A Systolic Neural CPU Processor Combining Deep Learning and General-Purpose Computing With Enhanced Data Locality and End-to-End Performance

Yuhao Ju, *Student Member, IEEE*, and Jie Gu, *Senior Member, IEEE*

***Abstract*— While neural network (NN) accelerators are being significantly developed in recent years, CPU is still essential for data management and pre-/post-processing of accelerators in a commonly used heterogeneous architecture, which usually contains an NN accelerator and a processor core with data transfer performed by direct memory access (DMA) engine. This work presents a special neural processor, referred to as a systolic neural CPU processor (SNCPU), which is a unified architecture combining deep learning and general-purpose computing for fifth-generation of reduced instruction set computer (RISC-V) to improve end-to-end performance for machine learning (ML) tasks compared with a common heterogeneous architecture with CPU and accelerator. With 64%–80% processing elements (PEs) logic reuse and 10% area overhead, SNCPU can be configured into ten RISC-V CPU cores. Special bi-directional dataflow and four different working modes are developed to enhance the utilization of deep NN (DNN) accelerator and eliminate the expensive data transfer between CPU and DNN accelerator in existing heterogeneous architecture. A 65-nm test chip was fabricated demonstrating a 39%–64% performance improvement on end-to-end image classification tasks for ImageNet, Cifar10, and MNIST datasets with over 95% PE utilization and up to 1.8TOPs/W power efficiency.**

***Index Terms*— Bi-directional dataflow, CPU, deep neural network (DNN) accelerator, end-to-end performance, heterogeneous architecture, machine learning (ML), general-purpose computing for fifth-generation of reduced instruction set computer (RISC-V).**

I. Introduction

ACCELERATORS designed for machine learning (ML) tasks, especially for DNN, are being rapidly developed in recent years. Because of the broad application space of DNN and its tremendous computing workload, improving energy efficiency for DNN accelerator has become a dominant effort for accelerator design. Cross-layer approaches have been explored at the software level such as quantization [1], architecture level such as flexible dataflows [2], and bit-precision [3], [4], micro-architecture level such as adaptive clock [5], and circuit level such as compute-in-memory techniques [7], [8], [9], [10]. However, very few works have focused on improving the end-to-end performance of deep learning tasks for the system-on-chip (SoC) level considering the cooperation of the accelerator and other digital modules.

In an end-to-end ML task, besides DNN computing using the accelerator, pre-processing, and post-processing also consume significant latency and power [11], [12], especially for edge computing on Internet-of-Things (IoT) devices with real-time processing applications where data streaming is performed from sensory device to the digital backend. For example, in [11], the DNN inference accelerator is served as an "edge-gateway-cloud" which achieves on-the-fly visual recognition and classification of insect blobs. Preprocessing in [11], including image capture, segmentation, extraction, and loading DNN configuration settings, takes over 70% of run-time for the end-to-end procedure. In [12], the SoC is implemented for a vision artificial intelligence (AI) solution with the DNN accelerator and image sensor on one single chip. Neural network (NN) results need to pass through a digital signal processing (DSP) sub-system, a result-selector, and interfaces for post-processing. Before DNN processing, image readout and image signal processing (ISP) are necessary for the accelerator as preprocessing work. Overall, pre/post-processing and data management take around 60% of the total run time.

Even for commonly used neural processing units (NPUs), accelerators are only in charge of the multiplication-accumulation (MAC) operations for different layers of NNs. As for inter-layer data preparation such as padding, batch normalization, data alignment, and duplication, the accelerator needs cohesive cooperation with another CPU core and highly efficient core-to-core communication. Fig. 1 illustrates a typical heterogeneous configuration that has a DNN accelerator and a CPU pipeline core that handles the data preparation and pre-/post-processing [6], [11], [12], [13], [14], [15]. Direct memory access (DMA) engine is developed to perform data transfer between the CPU core and the DNN accelerator. Interface and global MEM provide the capability for data streaming with off-chip DRAMs. Within the DNN accelerator, the systolic array is a popular architecture for 2-D convolution with data being piped through PE units for easier dataflow management and less SRAM bandwidth requirements.

Manuscript received 29 May 2022; revised 25 August 2022; accepted 30 September 2022. Date of publication 27 October 2022; date of current version 28 December 2022. This article was approved by Associate Editor Sophia Shao. This work was supported by the National Science Foundation under Grant CCF-2008906. *(Corresponding author: Yuhao Ju.)*

The authors are with the Department of Electrical and Computer Engineering, Northwestern University, Evanston, IL 60208 USA (e-mail: yuhaoju2017@u.northwestern.edu; jgu@northwestern.edu).

Color versions of one or more figures in this article are available at https://doi.org/10.1109/JSSC.2022.3214170.

Digital Object Identifier 10.1109/JSSC.2022.3214170

一种结合深度学习和通用计算的收缩神经CPU处理器，具有增强的数据局部性和端到端性能

于耀菊 等，*IEEE学生会员*，葛杰，*IEEE高级会员*

***摘要*—尽管近年来神经网络加速器（NN）取得了显著进展，但在常见的异构架构中，CPU仍然是数据管理及加速器预后处理的核心组件。这种架构通常包含神经网络加速器和通过直接内存访问（DMA）引擎进行数据传输的处理器核心。本文提出了一种特殊的神经处理器——脉动神经CPU处理器（SNCPU），该处理器是第五代精简指令集计算机（RISC-V）中深度学习与通用计算融合的统一架构，相比传统的CPU-加速器异构架构，能显著提升机器学习任务的端到端性能。通过64%一80%的处理单元逻辑复用率和10%的面积开销，SNCPU可配置为十个RISC-V CPU核心。我们开发了特殊的双向数据流机制和四种工作模式，既能充分利用深度神经网络（DNN）加速器的优势，又能消除现有异构架构中CPU与DNN加速器之间昂贵的数据传输。一个65纳米的测试芯片被制造出来，在ImageNet、Cifar10和MNIST数据集上的端到端图像分类任务中，性能提高了39%一64%，PE利用率超过95%，功率效率高达1.8TOPs/W。**

***索引术语*——双向数据流、CPU、深度神经网络（DNN）加速器、端到端性能、异构体系结构、机器学习（ML）、用于第五代精简指令集计算机（RISC-V）的通用计算。**

I. 一引言

在电路层面，诸如内存计算技术[7][8][9][10]等方法已被广泛应用。然而，针对系统级芯片（SoC）层面的深度学习任务，结合加速器与其他数字模块协同工作的端到端性能优化研究却寥寥无几。

在端到端机器学习任务中，除了使用加速器进行DNN计算外，预处理和后处理也会消耗大量延迟和功耗[11][12]，这在物联网设备的边缘计算场景中尤为明显——当实时处理应用需要将数据流从传感器设备传输至数字后台时。例如[11]中的案例，DNN推理加速器作为“边缘网关云”实现昆虫斑点的实时视觉识别与分类。该方案中预处理环节（包括图像采集、分割提取及DNN配置加载）占整个流程运行时间的70%以上。而[12]方案则采用单芯片SoC架构，将DNN加速器与图像传感器集成于一体。神经网络（NN）结果需经过数字信号处理（DSP）子系统、结果选择器及后处理接口。在DNN处理前，加速器还需完成图像读取和图像信号处理（ISP）作为预处理工作。总体而言，预后处理与数据管理环节约占总运行时间的60%。

即便是常用的神经处理单元（NPU），其加速器也仅负责神经网络不同层级的乘积累加（MAC）运算。至于层间数据准备（如填充、批量归一化、数据对齐和复制等操作），加速器需要与另一个CPU核心协同工作，并实现高效的核心间通信。图1展示了一个典型的异构配置，其中包含一个DNN加速器和一个负责数据准备及预/后处理的CPU流水线核心[6][11][12][13][14][15]。直接内存访问（DMA）引擎被开发用于在CPU核心与DNN加速器之间进行数据传输。接口和全局内存模块提供了与片外DRAM进行数据流传输的能力。在DNN加速器内部，脉动阵列是二维卷积的常用架构，通过将数据通过PE单元进行流水线处理，既简化了数据流管理，又降低了SRAM带宽需求。

稿件于2022年5月29日接收，2022年8月25日修订，2022年9月30日录用。发表日期：2022年10月27日；当前版本日期：2022年12月28日。本文经副主编邵素芳审阅批准。本研究获美国国家科学基金会资助（项目编号：CCF-2008906）。*（通讯作者：居玉浩）*

作者来自美国伊利诺伊州埃文斯顿市西北大学电气与计算机工程系（邮编：60208），联系方式：yuhaoju2017@u.northwestern.edu；jgu@northwestern.edu。

本文中一个或多个图形的彩色版本可在https://doi.org/10.1109/JSSC.2022.3214170上找到。

数字对象标识符10.1109/JSSC.2022.3214170

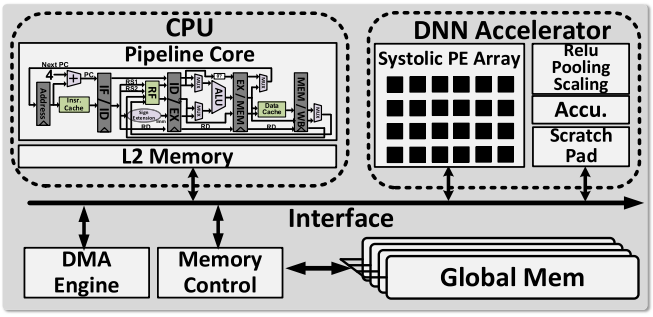


Fig. 1. Conventional heterogeneous architecture with a CPU pipeline core and a DNN accelerator.

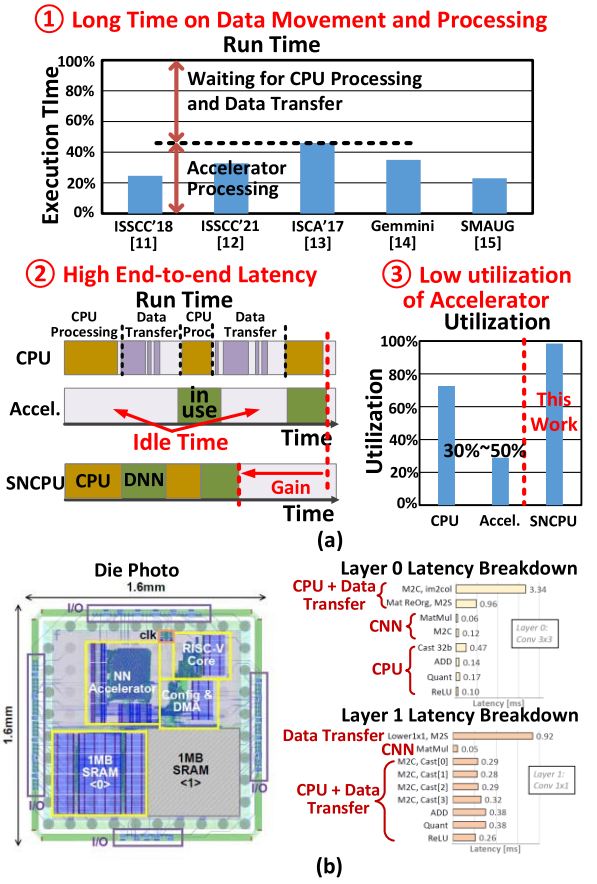


Fig. 2. (a) Challenges for heterogeneous architecture of the CPU and the accelerator. (b) Example of workload breakdown from the recent demonstration from meta [6].

There are several critical challenges for the existing heterogeneous architecture of CPU and accelerator to process end-to-end ML tasks. The challenges include: 1) large run time cost due to data movement across CPU and accelerators; 2) high latency for data preparation and pre-/post-processing using CPU; and 3) low utilization of accelerator due to imbalance of workload and waiting time on CPU and data transfer across processing cores [6], [11], [12], [13], [14], [15]. As shown in Fig. 2(a), for an end-to-end ML task, the accelerator is utilized for only 30%–50% of the total run time. The rest of the time is to wait for CPU processing and data movement between CPU and accelerator cores, which causes the stall of

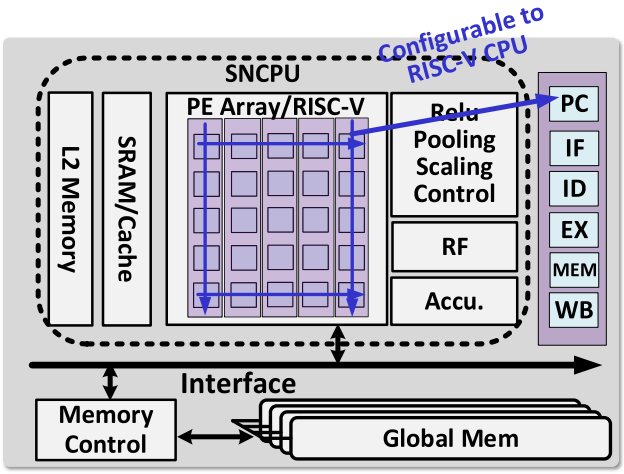


Fig. 3. Block diagram for the architecture of the proposed SNCPU.

the accelerator. Only 46.2% of utilization is reported from the tensor processing unit (TPU) [13].

Another detailed example is given in Fig. 2(b) for a test chip, which is recently published by Meta on AR/VR applications [6]. The SoC contains an NN accelerator with an general-purpose computing for fifth-generation of reduced instruction set computer (RISC-V) CPU pipeline core and a powerful DMA engine for data movement. For the first layer of the convolutional NN (CNN) model for eye gaze tracking in this SoC, the total latency is 5.36 ms with only 0.18 ms used by accelerator computing. Data movement to different cores and initial data preparation using im2col take more than 60% of the execution (EX) time. A similar story is for the second layer of CNN. Without initial image preprocessing, the data movement still takes more than 1 ms in terms of 3.16 ms total latency, which is over 30% of the total run time. NN accelerator only takes 0.05 ms for CNN processing. The rest of the EX time is for CPU processing such as casting, ReLU, batch normalization, and quantization.

To improve the efficiency of core-to-core communication, prior works have considered compressing data, accelerating data transfer, and increasing MEM bandwidth. As an example of [16], an accelerator coherency port (ACP) was designed to request data directly from the last level cache of the CPU instead of using the DMA engine to control the data transfer.

To address the challenges discussed above, in this work, we developed a new architecture, which is shown in Fig. 3, referred to as a systolic neural CPU processor (SNCPU), which combines an RISC-V CPU and a systolic CNN accelerator in one unified core. The contributions of this work include: 1) the flexible configuration for a multi-core RISC-V CPU or a systolic DNN accelerator with over 95% PE utilization for the end-to-end operation; 2) significant throughput improvement for CPU work because of ten-core CPU reconfiguration with 10% area overhead compared with a heterogeneous architecture for one CPU and one accelerator; 3) avoidance of expensive data movement across cores by using a special bi-directional dataflow for latency reduction; and 4) the demonstration of SNCPU through a 65-nm test chip with the

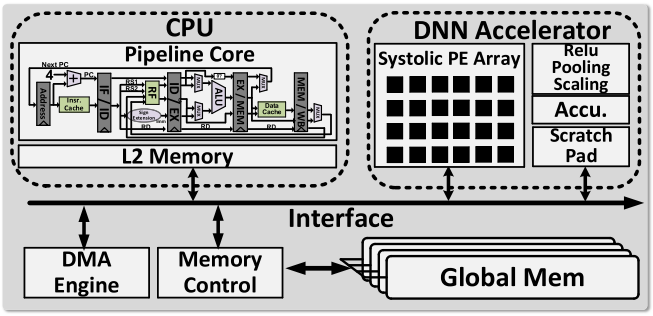


图1.传统异构架构，包含CPU流水线核心和DNN加速器。

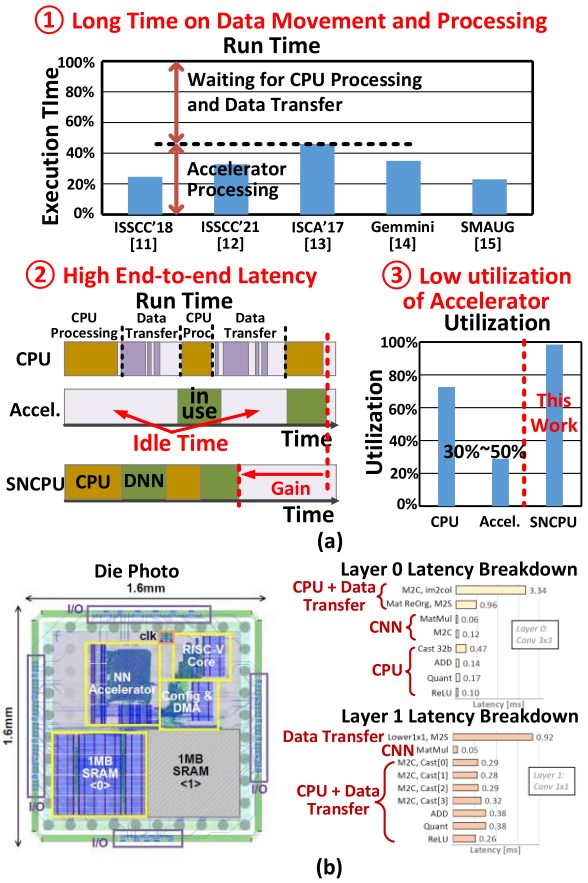


图2。(a)CPU与加速器异构架构面临的挑战。(b)来自meta [6]最新演示的工作负载分解示例。

现有的CPU与加速器异构架构在处理端到端机器学习任务时面临三大核心挑战：其一，数据在CPU与加速器之间的传输导致运行时间成本高昂；其二，使用CPU进行数据预处理和前后处理时存在显著延迟；其三，由于CPU工作负载与等待时间失衡，以及处理核心间的数据传输问题[6][11][12][13][14][15]，加速器利用率始终偏低。如图2(a)所示，在端到端机器学习任务中，加速器仅被利用了总运行时间的30%-50%。其余时间都耗费在等待CPU处理及数据在CPU与加速器核心间的传输上，这种滞留现象会导致

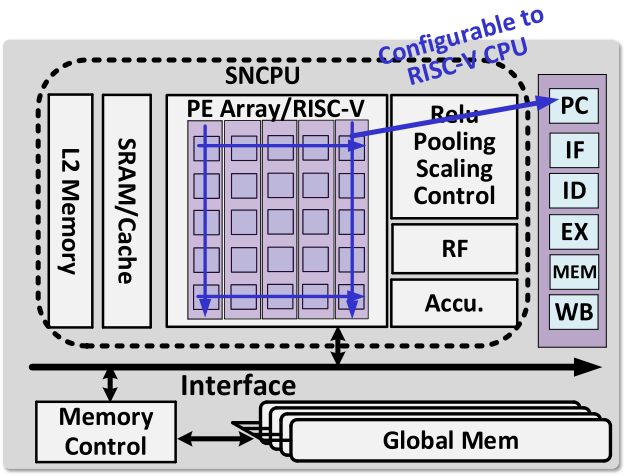


图3.所提出的SNCPU的体系结构框图。

加速器。据报告，张量处理单元（TPU）的利用率仅为46.2% [13]。

图2(b)展示了Meta公司最新发布的AR/VR应用测试芯片的详细案例[6]。该片上系统（SoC）搭载了第五代精简指令集计算机（RISC-V）CPU流水线核心的通用计算NN加速器，以及强大的数据移动引擎DMA。在用于眼动追踪的卷积神经网络（CNN）第一层中，总延迟为5.36毫秒，其中加速器计算仅占0.18毫秒。数据传输至不同核心及使用im2col进行初始数据准备耗时超过60%的执行时间。类似情况出现在CNN第二层：若未进行图像预处理，数据传输仍需耗时1毫秒以上，总延迟达3.16毫秒，超过运行总时长的30%。而NN加速器仅需0.05毫秒完成CNN处理，剩余执行时间则用于CPU处理如数据类型转换、ReLU激活函数、批量归一化和量化等操作。

为提升核心间通信效率，现有研究主要从压缩数据、加速数据传输和提升内存带宽三个方面展开。以文献[16]为例，该研究设计了加速器一致性端口（ACP），使数据直接从CPU的末级缓存获取，而无需通过DMA引擎进行控制。

为应对上述挑战，本研究开发了一种新型架构（如图3所示），称为脉动神经CPU处理器（SNCPU），该架构将RISC-V CPU与脉动CNN加速器整合于同一核心。本工作的主要贡献包括：

1) 该方案具备以下优势：1）多核RISC-V CPU或脉动式深度神经网络加速器的灵活配置，实现端到端运算中超过95%的处理器单元利用率；2）通过十核CPU重构技术实现CPU性能的显著提升，相比单核CPU与加速器的异构架构方案仅增加10%的面积开销；3）采用特殊双向数据流机制有效降低跨核心数据传输延迟，避免了昂贵的数据移动操作；4）通过65纳米制程测试芯片成功验证SNCPU架构的可行性。

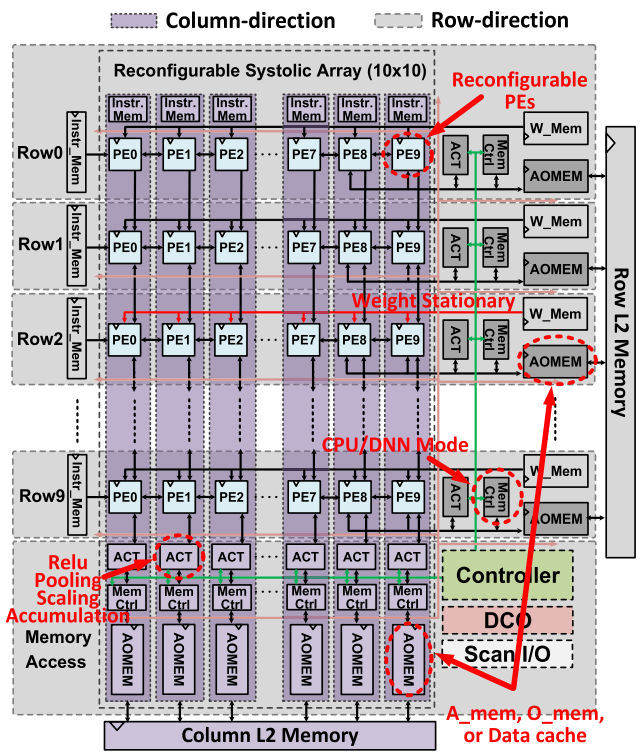


Fig. 4. Top-level architecture of SNCPU.

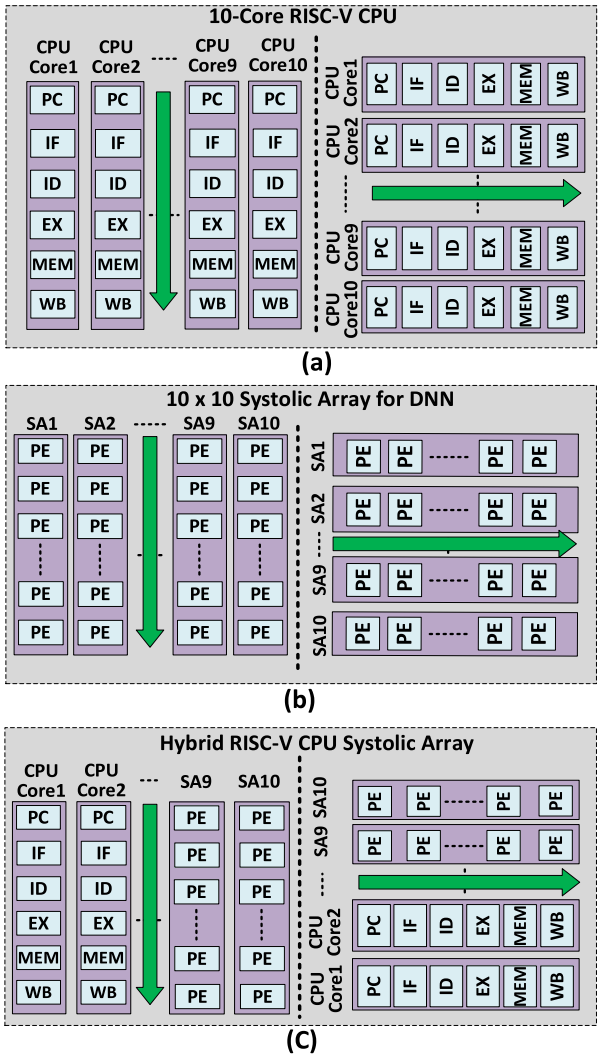


Fig. 5. Different configuration modes of SNCPU. (a) Multi-core CPU mode. (b) DNN mode. (c) Hybrid mode.

39%–64% latency improvement and the 0.65–1.8TOPS/W energy efficiency for end-to-end image-classification tasks.

The rest of this article is organized as follows. The overview of the top-level architecture of SNCPU is discussed in Section II. Section III shows the details for PE logic reuse and MEM reconfiguration supporting both CPU and accelerator functions. Special bi-directional dataflow is introduced in Section IV. The implementation and measurement results obtained by the test chip are shown in Section V, which also includes the test cases. Conclusions are in Section VI. This article is a detailed extension of the conference publication in ISSCC 2022 [22].

II. Top-Level Architecture of SNCPU

A. Design Overview

Fig. 4 shows the top-level architecture of SNCPU. A reconfigurable 10 × 10 PE array serves as the central computing tiles. This 2-D systolic array can be utilized as a baseline DNN accelerator or a ten-core five-stage RISC-V processor by reusing PE logic and MEM.

As shown in Fig. 4, the baseline DNN accelerator supports INT8 MAC operations with weight stationary dataflow. Memories are evenly separated and mapped to each row or column of the accelerator to enhance the flexibility and convenience of CPU mode reconfiguration. Each row or column of ten PEs has one reconfigurable "activation, output and data cache memory" (AOMEM), which can be reconfigured as an activation (ACT) SRAM bank, an output SRAM bank, or a data cache in different reconfigurations or dataflows. In addition, the ACT module is implemented for each row or column to deal with simple processing functions that DNN needs, such as partial sum accumulation, pooling, ReLU, and scaling. Furthermore, each row/column of 10 PEs owns a small controller to control the SRAM data arbitration and cohesive cooperation with different reconfiguration modes (DNN/CPU) and dataflows. A global digital controller oscillator (DCO) provides a tunable global clock. A top-level controller is designed to control the mode switching and dataflow switching. Each PE in the 2-D array is reconfigurable to satisfy the requirement of typical DNN operation and CPU operation. Additional SRAM banks are used to support multi-core CPU functionalities, such as the instruction SRAM bank of each row/column and two L2 SRAM banks for core-to-core communication in CPU mode.

B. Reconfiguration Modes and Dataflow Overview

Reconfiguration modes are shown in Fig. 5 where each lane of the PE array, i.e., each row or each column of PEs, can be configured as either systolic MAC operations for the DNN accelerator or CPU pipeline stages. Associated SRAM banks are also reconfigured for both purposes. Each mode contains

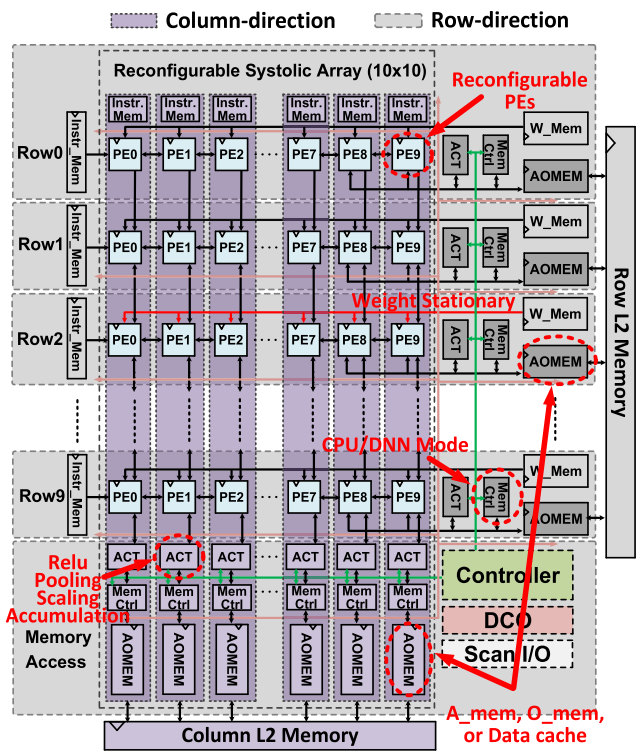


图4.SNCPU顶层架构。

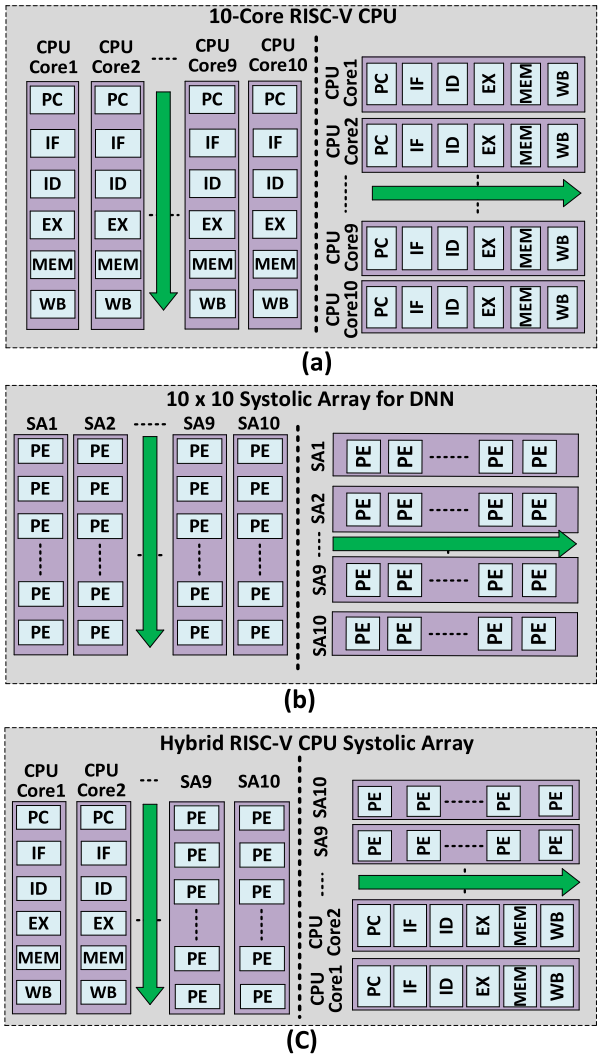


图5.SNCPU的不同配置模式。(a)多核CPU模式。(b) DNN模式。(c)混合模式。

在端到端图像分类任务中，延迟提高了39%-64%，0.65−1.8 TOPS/W能源效率也有所提高。

本文后续内容安排如下：第二部分将概述SNCPU的顶层架构；第三部分详细阐述支持CPU与加速器双重功能的PE逻辑复用及MEM重构技术；第四部分创新性地提出双向数据流机制；第五部分展示测试芯片的实现方案与实验数据，其中包含具体测试案例；第六部分为结论总结。本文是对ISSCC 2022会议论文[22]的深度拓展与技术延伸。

II.T OP-L ELE-V A-R-C-H-I-T-ACTION OF SNCPU

A. 设计概述

图4展示了SNCPU的顶层架构。一个可重构的10×10处理单元阵列作为中央计算核心。该二维脉动阵列通过复用处理单元逻辑和内存模块，既可作为基础DNN加速器使用，也能升级为十核五级RISC-V处理器。

如图4所示，基准版DNN加速器支持基于权重固定数据流的INT8 MAC运算。内存模块均匀分布并映射到加速器的每一行或列，从而提升CPU模式重构的灵活性与便捷性。每个行或列通常包含多个可重构的“激活、输出和数据缓存内存”（AOMEM）模块，这些模块可根据不同重构方式或数据流配置为激活（ACT）SRAM库、输出SRAM库或数据缓存。此外，每个行或列都配置了专用的ACT模块，用于处理深度神经网络（DNN）所需的简单处理功能包括部分求和累加、池化、ReLU激活函数和缩放操作。此外，每个10个处理单元（PE）的行/列都配备小型控制器，用于管理SRAM数据仲裁，并协调不同重构模式（DNN/CPU）与数据流的协同工作。全局数字控制器振荡器（DCO）提供可调谐的全局时钟信号。顶层控制器负责控制模式切换和数据流切换。二维阵列中的每个处理单元均可重构，以满足典型DNN运算和CPU操作的需求。额外的SRAM存储库支持多核CPU功能，例如每行/列的指令SRAM存储库，以及用于CPU模式下核心间通信的两个二级SRAM存储库。

B. 重构模式和数据流概述

图5展示了重新配置模式，其中PE阵列的每个通道（即PE的每一行或每一列）均可配置为DNN加速器的脉动式MAC操作或CPU流水线阶段。与之对应的SRAM存储库也会根据这两种用途进行重新配置。每种模式包含

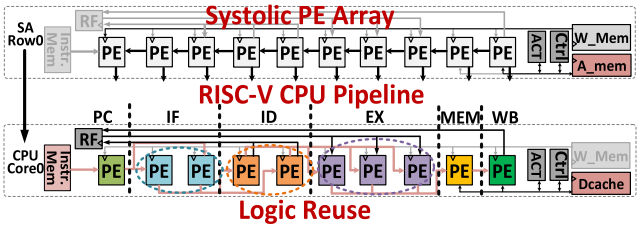


Fig. 6. Reconfiguration of rows of PE array for a single RISC-V pipeline core.

bi-directional dataflows and will be introduced in Section IV in detail. Fig. 5(a) illustrates two multi-core CPU modes as each row/column is utilized as one five-stage RISC-V pipeline core. Fig. 5(b) demonstrates DNN modes supporting DNN using MAC operation for convolutions. A special hybrid mode is also supported for imbalanced workload for end-to-end ML tasks between the CPU and the accelerator. In the hybrid mode, as shown in Fig. 5(c), half number of PEs are configured as DNN accelerators while the rest are configured into a five-core RISC-V CPU providing additional flexibility in workload assignment between the DNN accelerator and the CPU.

III. PE Logic Reuse and Memory Reconfiguration

A. PE Logic Reuse for CPU Reconfiguration

Fig. 6 shows the construction of a 32-b five-stage RISC-V CPU pipeline core from a lane of systolic PE array. Similar to a typical accelerator design, each PE in SNCPU contains a simple pipelined MAC unit with 8-b input and weight and maximum 32-b accumulation output. Fig. 6 also shows the logic reuse methodology for one row of ten PEs to realize pipeline functions. One column of ten PEs also supports CPU mode reconfiguration by using the same strategy of logic reuse. Based on the CPU mode reconfiguration for one row and one column, each PE needs to support two different pipeline stages of CPU mode, which means PE design is not unified in the 2-D PE array. While the design complexity has increased for the PE unit, the additional design efforts were managed by creating templates of pipeline stages and re-distribute them to the specific PE based on its location in the 2-D array. Besides one row/column PE reconfiguration, additional CPU-only SRAM banks were added to support complete CPU functions such as instruction caches and register files (RFs). CPU-only SRAM banks are gated for DNN modes.

Details of PE logic reuse for row reconfiguration are shown in Fig. 7, the very first PE in a row or column reuses the MAC's adder and 32-b registers as program counter (PC) to provide the instruction cache access for each pipeline core. Two PEs are used as the instruction fetch (IF) stage for instruction fetch with a reuse of the internal 32-b register and 8-b input registers. Two PEs are reconfigured into the instruction decoding (ID) stage, where the logic in the 8-b multiplier and 32-b adder are reconstructed to generate computing control signals by performing numerical/logical operations with the op-code or func-code of instructions. The next three PEs are combined into the EX stage, including one PE serving as an arithmetic logic unit (ALU) with additional

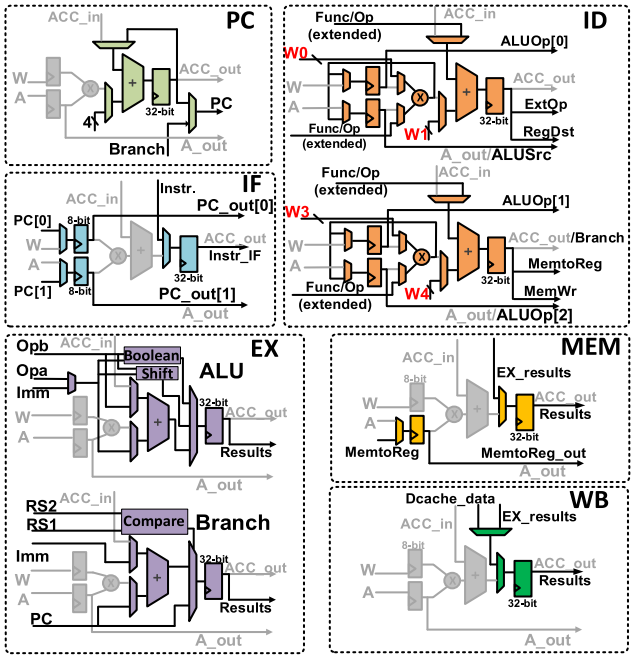


Fig. 7. PE logic reuse details of five pipeline stages for RISC-V ISA of CPU mode reconfiguration.

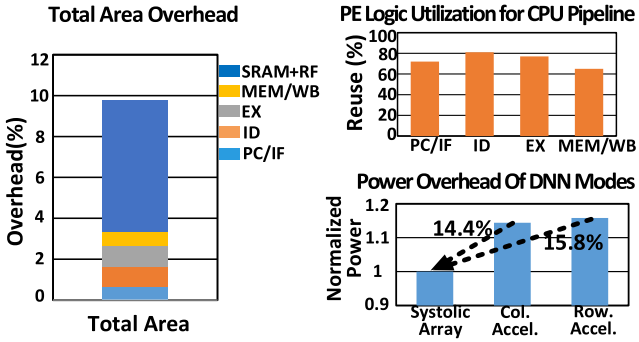


Fig. 8. Area and power overhead for CPU support and PE logic reuse rate.

logic for Boolean operations and a shifter, one PE supporting ALU branch calculation for new address from the instruction cache, and one PE used only for the registers to pass the EX results. The next PE is reconfigured into the CPU MEM stage for sending the ALU results to the data cache (Dcache) or bypassing the Dcache read/write. The last PE is utilized as the write-back (WB) stage to fetch the readout data from Dcache and send it to the RF by reusing registers with additional MUX logic. Several forwarding paths are also implemented to support CPU data dependency. Multipliers are also reused to realize multiplication instructions for RISC-V instruction set architecture (ISA). The multiplication instruction may take several cycles to finish in CPU modes.

B. Area, Power Overhead, and Utilization

As shown in Fig. 8, with an emphasis on logic sharing, the pipeline core reconfiguration utilizes 64%–80% of the original PE logic for CPU construction including the reconfiguration for multiplication instruction. The logic reuse rate is 69% at

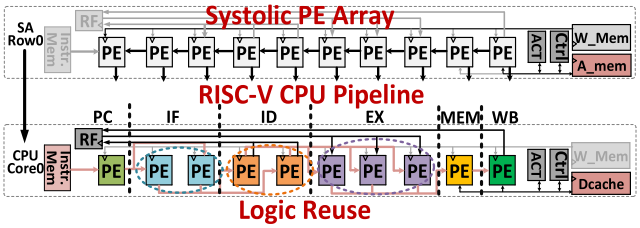


图6.针对单个RISC-V流水线内核的PE阵列行的重新配置。

双向数据流机制将在第四节详细阐述。图5(a)展示了两种多核CPU工作模式：每行/列可作为五级RISC-V流水线核心使用。图5(b)则演示了通过MAC运算实现卷积处理的深度神经网络（DNN）模式。针对CPU与加速器之间端到端机器学习任务的非均衡工作负载，还支持一种特殊混合模式。如图5(c)所示，在混合模式下，半数处理单元（PE）配置为DNN加速器，其余则组成五核RISC-V CPU，从而为DNN加速器与CPU之间的任务分配提供更大灵活性。

III. PE Logic Reuse和Memory Reconfiguration

A. 用于CPU重新配置的PE逻辑重用

图6展示了基于脉动处理单元阵列通道构建的32位五级RISC-V CPU流水线核心。与典型加速器设计类似，SNCPU中的每个脉动处理单元（PE）均配备带8位输入/权重和最大32位累加输出的简单流水线MAC单元。图6还展示了通过十行十列脉动处理单元实现流水线功能的逻辑复用方案。采用相同逻辑复用策略，一列十行脉动处理单元还能支持CPU模式重构。基于单行单列的CPU模式重构，每个脉动处理单元需同时支持两种不同流水线阶段，这意味着二维脉动处理单元阵列的设计并非统一标准。虽然脉动处理单元的设计复杂度有所增加，但通过创建流水线阶段模板并根据其在二维阵列中的位置进行重新分配，有效缓解了设计负担。除单行/单列脉动处理单元重构外，还新增了仅用于CPU的SRAM存储库以支持指令缓存和寄存器文件（RF）等完整CPU功能。这些仅用于CPU的SRAM存储库在深度神经网络模式下会触发门控机制。

图7展示了行/列重构中PE逻辑复用的详细架构。首列或首行的PE会复用MAC的加法器和32位寄存器作为程序计数器（PC），为各流水线核心提供指令缓存访问支持。两个PE被配置为指令获取阶段，通过复用内部32位寄存器和8位输入寄存器实现指令读取。另外两个PE则重构为指令解码阶段，其中8位乘法器和32位加法器的逻辑被重建，通过执行与指令操作码或功能码相关的数值/逻辑运算生成计算控制信号。接下来的三个PE组合成执行阶段，其中一个PE作为算术逻辑单元（ALU）使用，并配备额外的

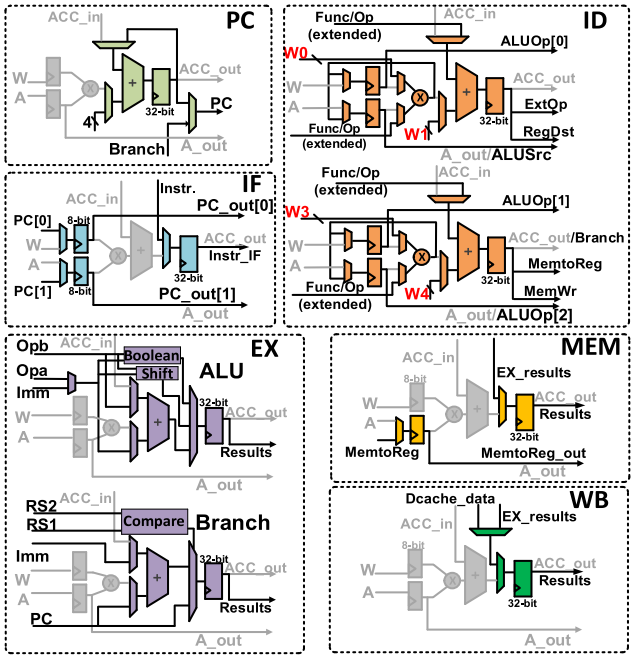


图7.CPU模式重新配置的RISC-V指令集架构中五个流水线阶段的PE逻辑复用细节。

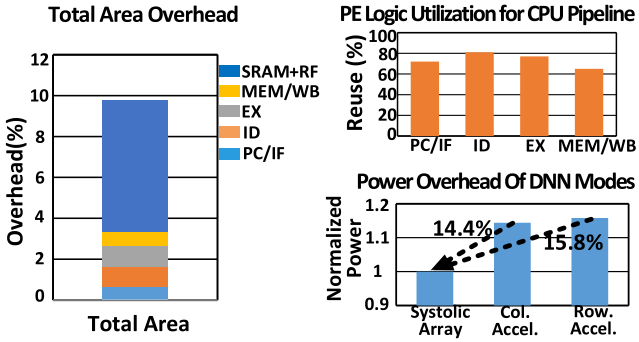


图8.支持CPU和PE逻辑复用率的面积和功率开销。

该架构包含布尔运算逻辑单元、移位器、一个支持ALU分支计算的新地址指令缓存处理单元（PE），以及仅用于传递EX结果寄存器的处理单元。后续PE模块可重新配置为CPU MEM阶段，用于将ALU运算结果传输至数据缓存（Dcache）或直接跳过Dcache的读写操作。最后一个PE模块作为写回（WB）阶段，通过复用寄存器并结合额外的多路复用逻辑，从Dcache中获取读取数据并传输至射频（RF）模块。系统还设计了多条数据转发路径以支持CPU的数据依赖性，同时复用乘法器来实现RISC-V指令集架构（ISA）的乘法指令运算。需要说明的是，乘法指令在CPU模式下可能需要多个时钟周期才能完成运算。

B. 面积、电源管理开销和利用率

如图8所示，通过重点采用逻辑共享技术，流水线核心重构在CPU构建中（包括乘法指令重构）可复用64%-80%的原始处理器单元逻辑，其逻辑复用率达69%。

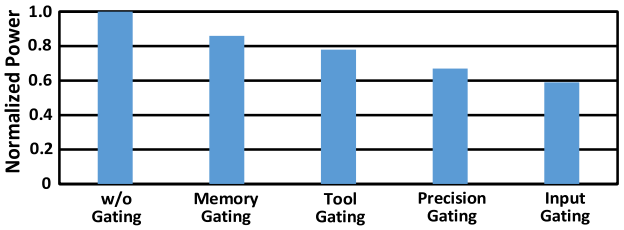


Fig. 9. Different implemented power gating techniques and their power benefits on DNN modes.

the PC and IF stages, 80% at the ID stage, 77% at the EX stage, and 64% at MEM/WB stage. The logic reuse rates are the average values for both row and column pipeline reconfigurations which differ slightly. For example, adders in PE are not reused in MEM/WB stage for row reconfiguration but are reused in column pipeline reconfiguration. Compared with the original systolic DNN accelerator, the area overhead to include CPU functions is 3.4% in the PE-array, 6.4% in the MEM, e.g., instruction and RF, and overall 9.8% for the whole SNCPU processor. Extensive clock gating is implemented to eliminate redundant power consumption from the additional logic and unused memories for DNN modes. As shown in Fig. 8, in two different accelerator modes of SNCPU, power overhead is about 15% compared with the baseline systolic array DNN accelerator.

Fig. 9 shows the improvement of different clock gating techniques for the DNN modes. Gating unused SRAM banks for the DNN modes such as instruction caches, L2 caches, and RFs can achieve 15% power saving. RTL-level optimization for better support of the gating function from the synthesis tool provides another 8% power saving. CPU reconfiguration with RV32I has 32-b precision while the DNN configuration only needs 26-b precision at most without overflow for the accumulator of each PE. Hence, unused flip-flops are gated for DNN modes achieving another 11% power saving with two different reconfigured precisions (20 b, 26 b) for different models. Special input gating is also used to eliminate the power consumption of unused CPU-only combinational logic leading to a 6% power saving. Overall, nearly 40% of energy saving for DNN modes is achieved by power gating.

Fig. 10 shows the power overhead for different CPU instructions from the reconfigurable PEs implemented in this work in comparison with baseline RISC-V pipelines excluding MEM power. The CPU operation from the reconfigurable PE array incurs around 14% power overhead compared with the scalar baseline RISC-V design (RV32I).

C. Customized Extension of RISC-V ISA for SNCPU

Customized instructions added into baseline RISC-V ISA as extension instructions are developed to support a smooth mode switching between CPU and DNN modes as shown in Fig. 11(a). Several control and status registers (CSRs) are used to store and configure the parameters required by the DNN accelerator mode processing. Accelerator setup instruction

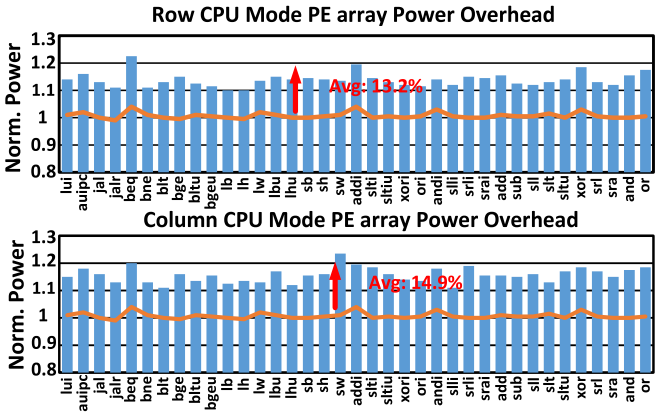


Fig. 10. Simulated power overhead from the reconfigurable PE array for RV32I instructions.

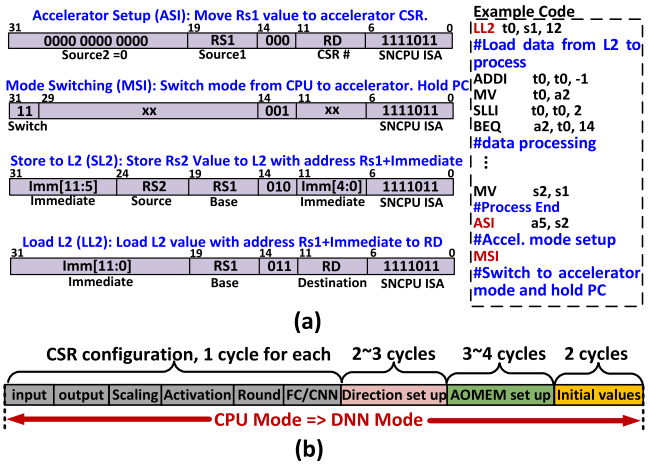


Fig. 11. (a) Customized RISC-V ISA extension for CSR, mode switching, and multi-core communication for CPU modes. (b) Detailed mode switching process from CPU to accelerator.

(ASI) is utilized to save values to those CSRs before the DNN accelerator starts to run. Mode switching instruction (MSI) is used to switch from CPU mode to DNN accelerator mode with holding the PC for future processing. Store L2 instruction (SL2) and load L2 instruction (LL2) provide a direct communication method to store/reload data from L2 for multi-core CPU data sharing.

The switching process from CPU to DNN accelerator is shown in Fig. 11(b). CSR configuration takes 6 cycles to save required configuration parameters for DNN running, such as input vector numbers, output vector numbers, scaling factor, and convolution/fully-connected (FC) calculation selection, 2–3 cycles are needed to set up the flow direction, i.e., row-based or column-based flow. The 3–4 cycles are used for "AOMEM" SRAM bank selection and configuration. It also takes two additional cycles to set up initial values for the DNN controller and initialize input gating. The total switching time is 13–15 clock cycles for switching from CPU to DNN configuration. On the other hand, when the DNN accelerator finishes the work, a trigger signal is generated to switch back into CPU modes from accelerator modes consuming only 3 cycles.

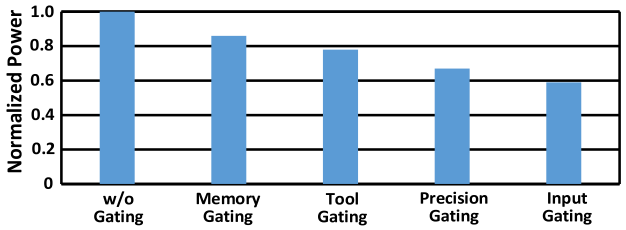


图9.不同的实现功率门控技术及其对DNN模式的功率效益。

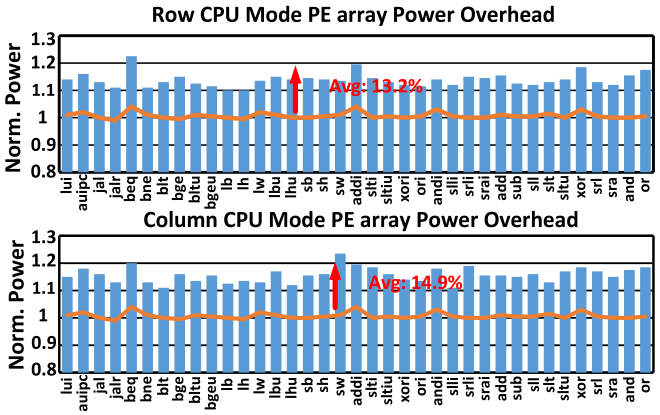


图10.RV32I指令的可重构PE阵列模拟功耗开销。

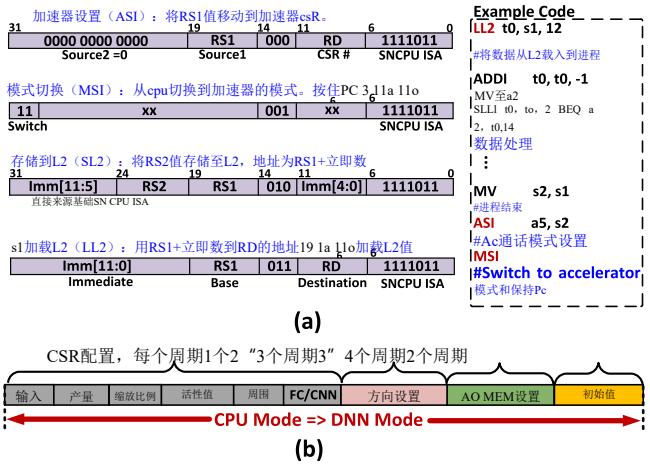


图11。(a)针对CPU模式下的CSR、模式切换和多核通信进行定制的RISC-V指令集架构扩展。(b)从CPU到加速器的详细模式切换流程。

在PC和IF阶段，ID阶段占比80%，EX阶段77%，MEM/WB阶段64%。逻辑复用率是行流水线和列流水线重构的平均值，两者存在细微差异。例如，在PE单元中，加法器在MEM/WB阶段的行重构中未被复用，但在列流水线重构中得以复用。与原始脉动式DNN加速器相比，PE阵列在CPU功能集成方面增加了3.4%的面积开销，在MEM（如指令存储器和射频模块）方面增加6.4%，整个SN-CPU处理器整体增加9.8%。通过实施广泛的时钟门控技术，有效消除了DNN模式下额外逻辑和未使用存储器造成的冗余功耗。如图8所示，在SNCPU的两种不同加速模式下，与基准脉动式阵列DNN加速器相比，功耗开销约为15%。

图9展示了不同时钟门控技术在DNN模式下的节能效果。通过门控未使用的SRAM存储库（如指令缓存、L2缓存和射频单元），可实现15%的能耗节省。通过RTL级优化增强合成工具对门控功能的支持，还能额外节省8%的功耗。采用RV32I架构的CPU重新配置需要32位精度，而DNN配置最多仅需26位精度即可避免每个处理单元累加器发生溢出。因此，针对不同模型采用两种精度(20位/26位）重新配置门控电路后，未使用的触发器门控技术又实现了11%的节能增益。此外，特殊输入门控技术有效消除了仅用于CPU的组合逻辑未使用部分的功耗，带来6%的节能提升。总体而言，通过电源门控技术，DNN模式下实现了近40%的能耗降低。

图10展示了本研究中采用可重构处理单元（PE）实现的各类CPU指令功耗开销，并与排除内存模块功耗的基准RISC-V流水线进行对比。相较于标量型基准RISC-V设计（RV32I），可重构PE阵列的CPU运算功耗开销约为14%。

C. 针对SNCPU的RISC-V指令集架构定制扩展

如图11(a)所示，开发了作为扩展指令添加到RISC-V指令集架构基线中的定制化指令，以支持CPU与深度神经网络（DNN）模式之间的平滑切换。通过多个控制寄存器和状态寄存器（CSRs），存储并配置DNN加速器模式处理所需的参数。加速器配置指令

在DNN加速器启动运行前，系统会通过（ASI）将数据保存至各个计算服务器单元（CSR）。模式切换指令（MSI）用于在CPU模式与DNN加速器模式之间进行切换，并保持程序计数器（PC）以便后续处理。存储L2指令（SL2）和加载L2指令（LL2）则提供了直接通信机制，实现L2缓存数据的存储与重载操作，从而支持多核CPU的数据共享。

CPU与DNN加速器之间的切换流程如图11(b)所示。配置CSR需要6个时钟周期来保存DNN运行所需参数，包括输入向量数量、输出向量数量、缩放系数以及卷积/全连接（FC）计算模式选择等。设置数据流方向（行或列）需2-3个时钟周期，而“AOEM”SRAM库的选择与配置则需要3-4个时钟周期。此外还需两个时钟周期用于设置DNN控制器的初始值并初始化输入门控。从CPU切换到DNN配置的总耗时为13-15个时钟周期。当DNN加速器完成工作后，系统会生成触发信号，仅需3个时钟周期即可从加速模式切换回CPU模式。

D. Reconfiguration Modes and Memory Reuse

The SNCPU architecture allows the majority of data to be retained inside the SRAM banks of the processor core even when operation modes change. It can eliminate the expensive data movement and the DMA engine in the heterogeneous architecture. To achieve high data locality, a special dual-mode bi-directional dataflow is developed resulting in four different reconfiguration modes. The four reconfiguration modes include two different operations, i.e., CPU and DNN, two directional dataflows, i.e., row-based dataflow and column-based dataflow.

Fig. 12 shows the detailed four different operation modes with activated modules highlighted in this figure. As shown in Fig. 12(a), column accelerator mode is the typical accelerator mode with weight stationary dataflow. The "AOMEM" SRAM bank in each row is used as input MEM to provide input data from right to left. Bottom "AOMEM" SRAM banks are used for output MEM for the accumulation results from top to bottom. Instruction caches and L2 SRAM banks are gated in this mode. As for the row CPU mode, which is shown in Fig. 12(b), each row of ten PEs can be configured as a CPU pipelined core, with each "AOMEM" SRAM bank on the right serving as a data cache for each CPU core. Instructions caches on the left are activated to pass instructions through every pipeline stage from left to right.

In the other directional scenario, as shown in Fig. 12(c), the PEs in row accelerator mode receives the input data from the bottom AOMEM banks and store the results in the right AOMEM banks. Bottom "AOMEM" banks are reused as input MEM for each column. Right "AOMEM" banks are utilized as output MEM. The accumulation dataflow for row accelerator mode is from left to right, which is an orthogonal direction compared to the column accelerator mode. All CPU-only SRAM banks are gated in this mode. The last mode, column CPU mode, is introduced in Fig. 12(d). It receives the instructions from the top instruction caches and the bottom AOMEM banks are reconfigured to data caches, which allows one column of ten PEs to be reconfigured to one RISC-V pipeline core.

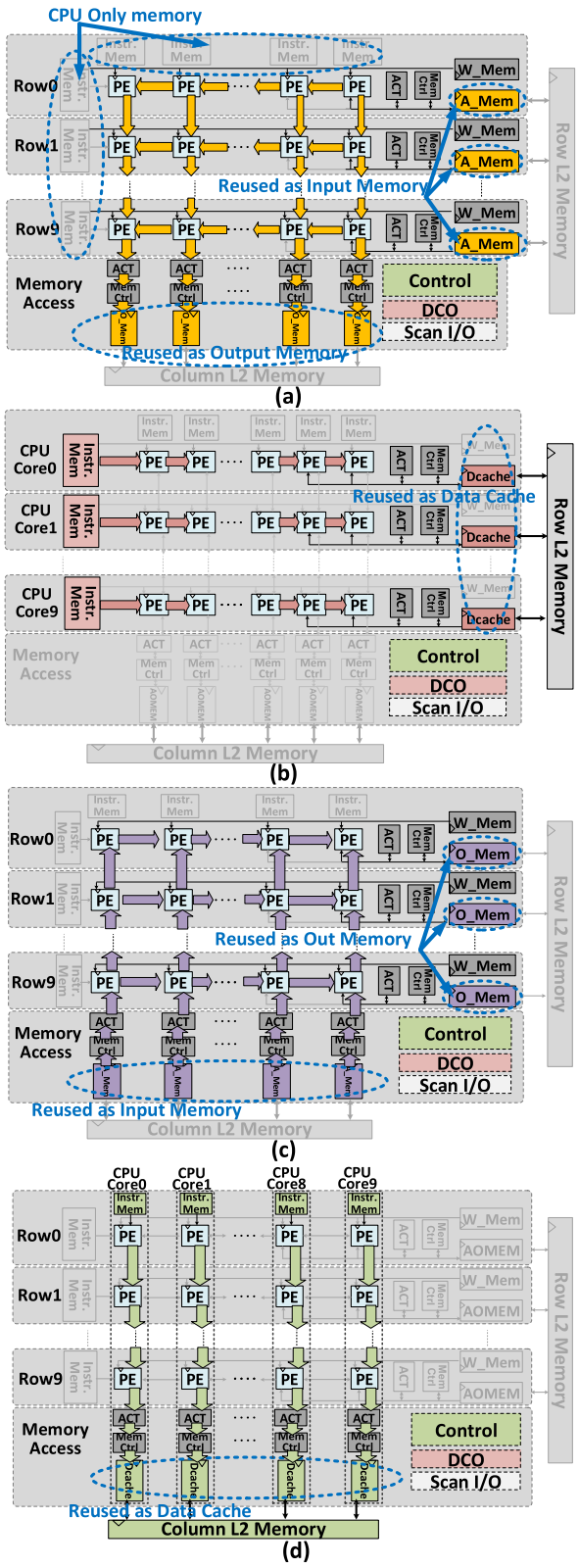


Fig. 12. Four different reconfiguration modes for two directions of SNCPU. (a) Column accelerator mode. (b) Row CPU mode. (c) Row accelerator mode. (d) Column CPU mode.

IV. Dual-Mode Bi-Directional Dataflow for Enhanced Data Locality

Fig. 13(a) shows the diagram of a special four-step dataflow with the four different configurations from Fig. 12 using the image classification task as an example. The four-step dataflow is designed to avoid data movement across MEM banks at different operation phases. In step 1 and step 3, different CPU modes are responsible for the pre/post-processing and inter-layer data preparation. Step 2 and Step 4 are used for DNN operations. In image classification tasks, preprocessing includes reshape, grayscale, rotation, and normalization. The inter-layer data processing for some commonly used DNN model contains data alignment, padding, duplication, and batch normalization.

Details are shown in Fig. 13(b). First, the SNCPU operates in row CPU mode to perform image preprocessing to generate the input data for the DNN operation of the first layer of the DNN model. Finished input data is stored in the "AOMEM" SRAM banks on the right, which are utilized as data caches

D. 重新配置模式和内存复用

SNCPU架构的核心优势在于：即使在不同运行模式切换时，也能将大部分数据保留在处理器核心的SRAM存储单元中。这种设计不仅省去了异构架构中昂贵的数据传输机制，还取消了DMA引擎。为实现高效的数据局部性，该架构特别开发了双模双向数据流技术，从而形成四种不同的重构模式。这四种重构模式包含两种操作模式（CPU与深度神经网络）和两种数据流向（行向数据流与列向数据流），形成灵活的数据处理架构。

图12展示了四种不同操作模式的详细示意图，其中激活模块在图中特别标出。如图12(a)所示，列加速器模式是典型的权重固定数据流加速器模式。每行中的“AOMEM”SRAM存储库作为输入MEM，负责从右向左提供输入数据。底部的“AOMEM”SRAM存储库则作为输出MEM，用于从上至下传递累加结果。在此模式下，指令缓存和L2 SRAM存储库处于门控状态。至于图12(b)所示的行CPU模式，每行十个处理单元可配置为一个CPU流水线核心，右侧每个“AOMEM”SRAM存储库充当各CPU核心的数据缓存。左侧的指令缓存被激活，使指令能够从左到右依次通过每个流水线阶段。

在另一种方向性场景中（如图12(c)所示），行加速模式下的处理单元从底部的AOMEM存储库接收输入数据，并将结果存入右侧的AOMEM存储库。底部的“AOMEM”存储库被重新用作每列的输入存储单元，而右侧的“AOMEM”存储库则作为输出存储单元。行加速模式下的数据流从左至右进行累积运算，这与列加速模式形成正交方向。在此模式下，所有仅用于CPU的SRAM存储库均处于门控状态。最后一种模式——列CPU模式（见图12(d)）——接收来自顶部指令缓存的指令，此时底部的AOMEM存储库被重新配置为数据缓存，使得每列十个处理单元可重构为一个RISC-V流水线核心。

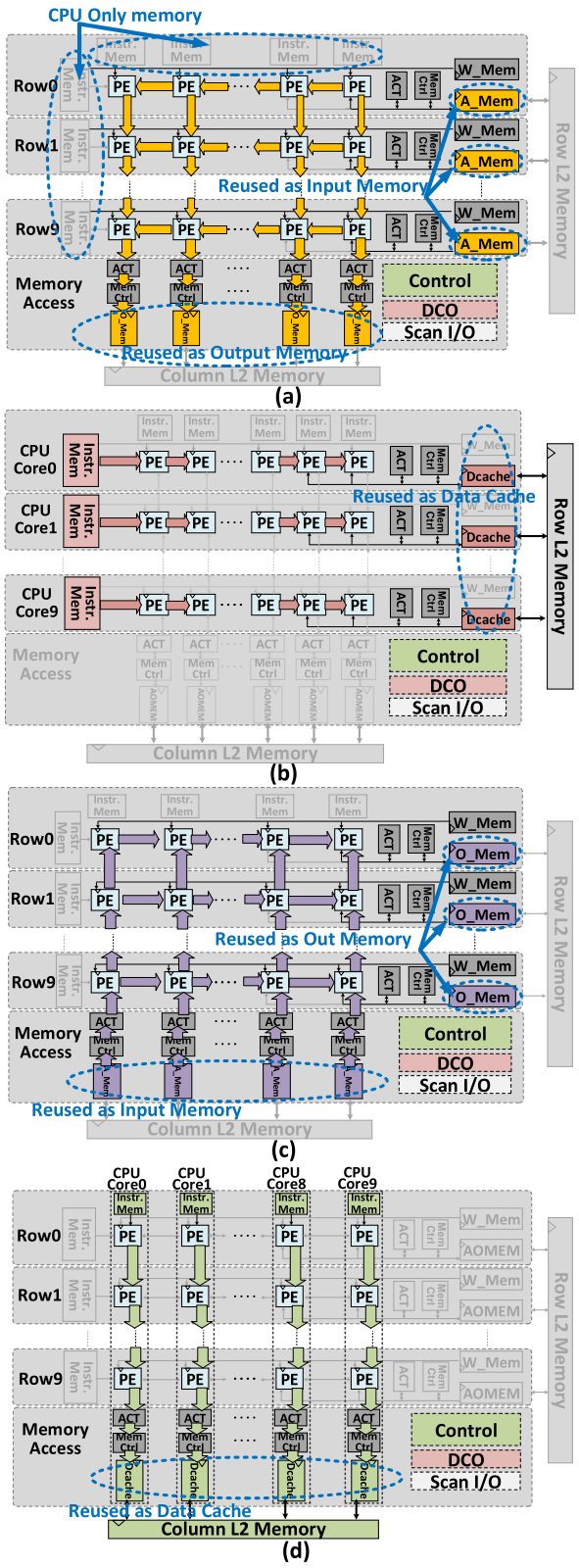


图12.SNCPU两个方向的四种不同重构模式。
(a) (b)行CPU模式。(c)行加速器模式。
(d) 列CPU模式。

IV.D UAL-M ODE B-D IRECTIONAL D ATAFLOWFOR E NHANCED D ATA L OCALITY

图13(a)展示了以图像分类任务为例的特殊四步数据流架构示意图，该架构基于图12中的四种不同配置方案。该四步数据流设计旨在避免在不同操作阶段跨MEM存储库的数据传输。其中步骤1和步骤3分别由不同CPU模式负责预处理/后处理及层间数据准备，步骤2和步骤4则用于深度神经网络运算。在图像分类任务中，预处理包含数据重塑、灰度化、旋转和归一化等操作。对于某些常用深度神经网络模型，其层间数据处理通常涉及数据对齐、填充、复制和批量归一化等操作。

具体细节如图13(b)所示。首先，SNCPU以行CPU模式运行，执行图像预处理以生成第一层DNN运算的输入数据。DNN模型。完成的输入数据存储在右侧的“AOMEM”SRAM存储库中，这些存储库用作数据缓存。

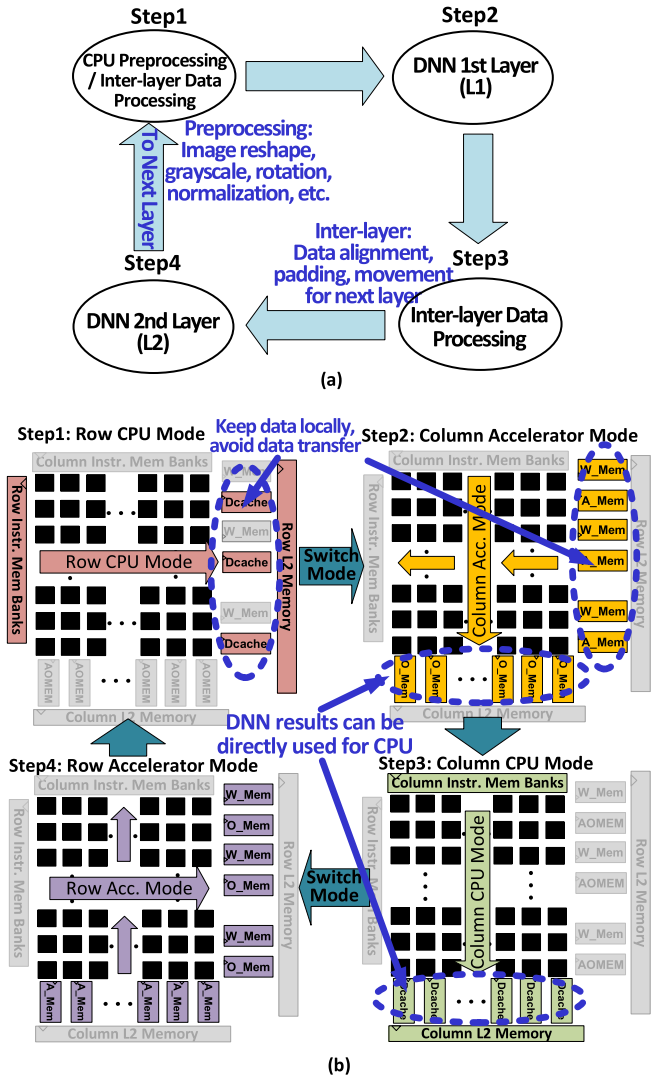
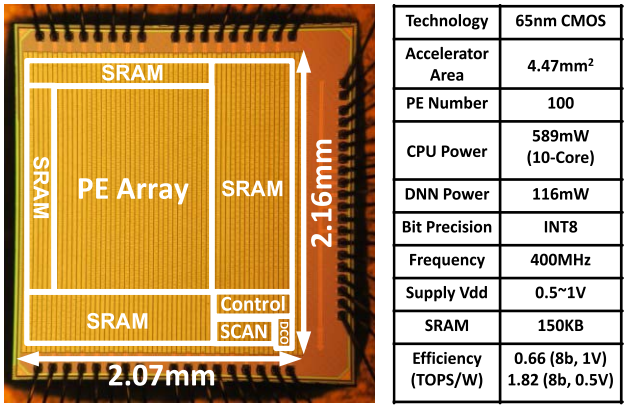


Fig. 13. (a) Block diagram and (b) architecture level illustration for four-step dual-mode bi-directional dataflow in SNCPU using image classification as an example.

for row CPU mode. Second, the SNCPU operates in column accelerator mode which reuses the right "AOMEM" banks as input MEM for the DNN accelerator. The DNN accelerator can directly use the data from the right "AOMEM" SRAM banks as input data, which is also the result of row CPU mode. Considering the same scenario in heterogeneous architecture, the DMA engine has to be engaged to transfer preprocessed data from the CPU data cache to the input scratchpad of the accelerator. Moreover, after the column accelerator finishes the entire first layer of the DNN model, the SNCPU is reconfigured to column CPU mode to perform data preparation work such as data alignment, padding, duplication, and batch normalization by directly using the data in the output MEM from the previous accelerator mode. Finally, the SNCPU switches to row accelerator mode to process the second layer of the CNN by directly using the data cache from the previous CPU mode as input MEM. The four-step operation repeats until all CNN layers are finished. All the data can stay either in the bottom "AOMEM" SRAM banks or right



Technology	65nm CMOS
Accelerator Area	4.47mm²
PE Number	100
CPU Power	589mW (10-Core)
DNN Power	116mW
Bit Precision	INT8
Frequency	400MHz
Supply Vdd	0.5~1V
SRAM	150KB
Efficiency (TOPS/W)	0.66 (8b, 1V) 1.82 (8b, 0.5V)

Fig. 14. Chip micrograph and specifications.

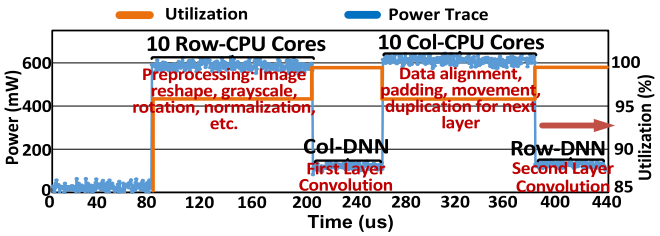


Fig. 15. Measured power trace and PE utilization during processing of the VGG16 model for CIFAR10 dataset.

"AOMEM" SRAM banks without any data transfer between CPU modes and DNN modes. Multi-core CPU improves the CPU throughput because of the parallel processing.

V. Chip Implementation, Measurement Results, and Case Study

A. Chip Implementation

A 2-D 10 × 10 SNCPU processor was designed and fabricated using a 65-nm CMOS process. The chip micrograph and implementation details are shown in Fig. 14. The active die area is 4.47 mm^2 (2.07 × 2.16 mm) with 1.0-V nominal supply voltage and 400-MHz operating frequency. The chip can support 8-b integer bit precision for DNN accelerator modes and RISC-V 32-b integer ISA for CPU modes with several customized ISA extensions. The chip was tested with a supply voltage scaled down to 0.5 V. The total on-chip SRAM is around 150 kB. This chip provides a scan IO interface to load and read out all on-chip SRAM content. A field-programmable gate array (FPGA) board is engaged in chip testing for data streaming in and out of the test chip through scan IO ports for verification and measurement.

B. Performance Measurement Results

Fig. 15 shows the measured first a few hundred microseconds of power trace on different modes using an image classification task. The row CPU mode starts to work at around 80 μs for ten cores running together to perform preprocessing work, such as image reshape, rotation, and normalization. After that, the SNCPU switches to column accelerator mode

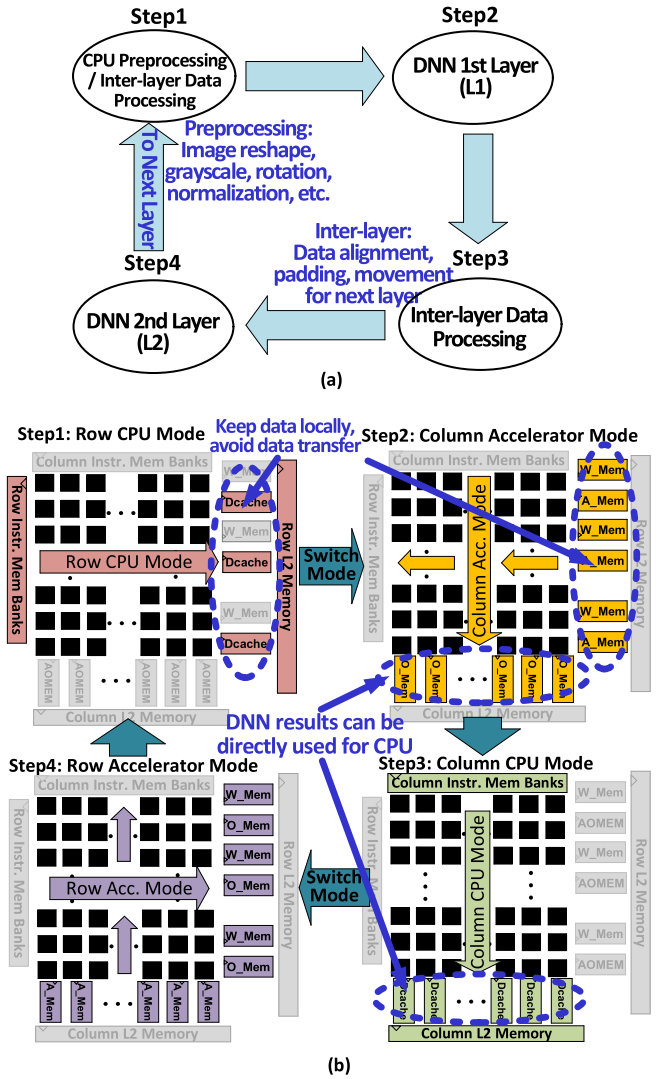
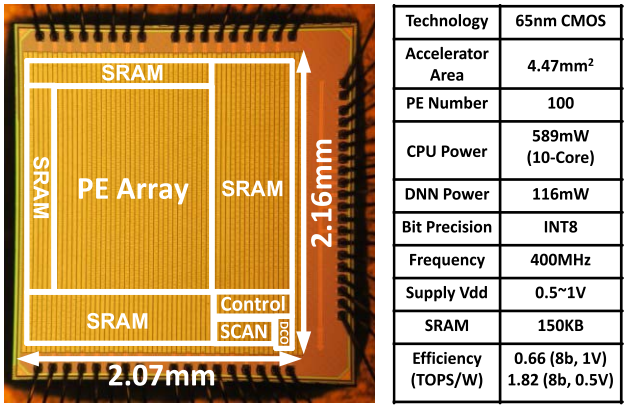


图13. (a)以图像分类为例的SNCPU中四步双模式双向数据流的框图和(b)体系结构级示意图。

在行CPU模式下，SNCPU首先以列加速器模式运行，该模式会将右侧的“AOMEM”存储库作为DNN加速器的输入内存进行复用。DNN加速器可直接使用右侧“AOMEM”SRAM存储库的数据作为输入数据，这些数据同样源自行CPU模式的处理结果。若在异构架构中遇到相同场景，则需启动DMA引擎将预处理数据从CPU数据缓存传输至加速器的输入暂存区。当列加速器完成DNN模型第一层运算后，SNCPU会切换至列CPU模式，直接利用前一加速器模式输出的内存数据执行数据对齐、填充、复制及批量归一化等准备工作。最后，SNCPU切换至行加速器模式，通过直接使用前一CPU模式的数据缓存作为输入内存来处理CNN的第二层运算。整个四步操作循环直至所有CNN层完成运算。所有数据可暂存于底部“AOMEM”SRAM存储库或右侧



Technology	65nm CMOS
Accelerator Area	4.47mm²
PE Number	100
CPU Power	589mW (10-Core)
DNN Power	116mW
Bit Precision	INT8
Frequency	400MHz
Supply Vdd	0.5~1V
SRAM	150KB
Efficiency (TOPS/W)	0.66 (8b, 1V) 1.82 (8b, 0.5V)

图14.芯片显微照片和规格。

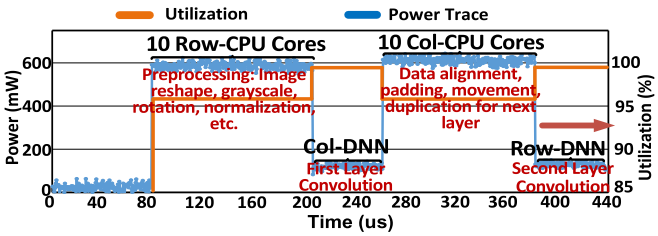


图15.CIFAR10数据集上VGG16模型处理过程中的实测功率轨迹和PE利用率。

在“AOMEM”SRAM存储库中，CPU模式与DNN模式之间不进行任何数据传输。多核CPU通过并行处理提高了CPU吞吐量。

V. C HIP I MPLEMENTATION，M EASUREMENT R ESULTS，以及C ASE S TUDY

A. 芯片实现

我们采用65纳米CMOS工艺设计并制造了一款2D 10×10 SNCPU处理器。如图14所示，该芯片的显微结构与具体实现细节清晰可见。其有源晶圆面积为4.47平方毫米（实际尺寸2.07×2.16毫米），标称供电电压为1.0V，工作频率达400MHz。该芯片在DNN加速器模式下支持8位整数精度，在CPU模式下则采用带有多种定制指令集扩展的RISC-V 32位整数指令集架构。测试时将供电电压降至0.5 V，芯片总片上静态随机存取存储器容量约为150千比特。芯片配备专用扫描输入输出接口，可实现片上SRAM的全部内容读写操作。此外，通过扫描输入输出端口建立现场可编程门阵列（FPGA）测试板，实现芯片数据流式传输，用于验证与性能测量。

B. 性能测量结果

图15展示了通过图像分类任务在不同模式下测量的前几百微秒功耗曲线。行CPU模式在约80 μ秒时开始运行，此时十个核心协同工作以执行图像重塑、旋转和归一化等预处理操作。此后，SNCPU切换至列加速器模式。

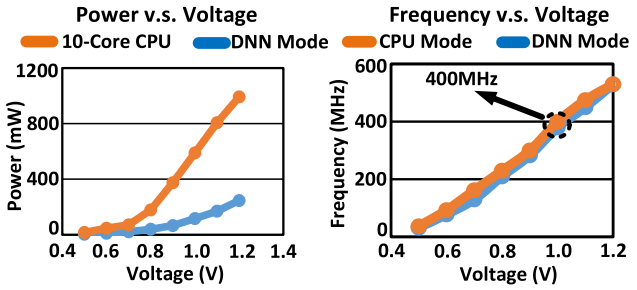


Fig. 16. Measured power and frequency with voltage scaling.

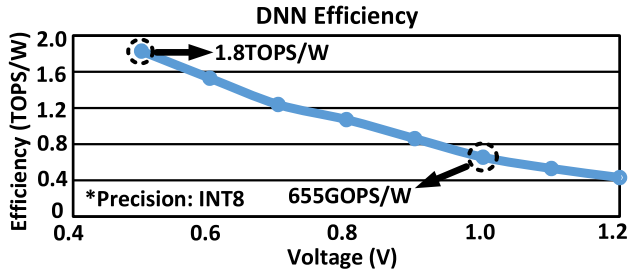


Fig. 17. Energy efficiency of SNCPU for DNN operation with voltage scaling.

to process the first layer of the DNN model for about 60 μs. Then the SNCPU switches to column CPU mode to prepare the input data for the next layer with padding and data alignment. Starting from 380 μs, the SNCPU does the DNN operations of the second layer with row accelerator mode for ten cores. Only the starting of the second layer is shown for the row accelerator mode in Fig. 15.

The PE utilization of each mode is also shown in Fig. 15. The average PE utilization for DNN modes is around 99% and the average PE utilization for CPU modes is around 96%. Total average PE utilization is 97% which is significantly higher than the conventional heterogeneous architectures [11], [12].

Fig. 16 shows the measured power and frequency with the voltage scaled down to 0.5 V. The nominal supply voltage for both DNN modes and CPU modes is 1.0 V with 589-mW ten-core CPU power and 116-mW DNN accelerator power at 400-MHz frequency. Fig. 17 shows the energy efficiency with a scaled supply voltage down to 0.5 V. The results are based on the INT8 bit precision. The DNN modes achieve 655GOPS/W at 1.0 V and increase to 1.8TOPS/W at 0.5 V.

As shown in Fig. 18, there are three reasons for elevated power in CPU modes compared with DNN modes. First, there are several CPU-only memories such as instruction caches, L2 caches, and RFs which added significantly more power consumption. Second, although extensive power gating for DNN modes was performed, the same level of optimization for CPU modes was not carried out due to the stringent tapeout time. Many unused SRAM banks and logic are ungated in CPU mode causing a power waste of about 45% in CPU mode. Third, additional power saving was achieved in DNN mode through programmable clock gating in the design by adjusting the required bit precision. For example, the accumulator in each PE only utilizes 20-b precision (based on the VGG16

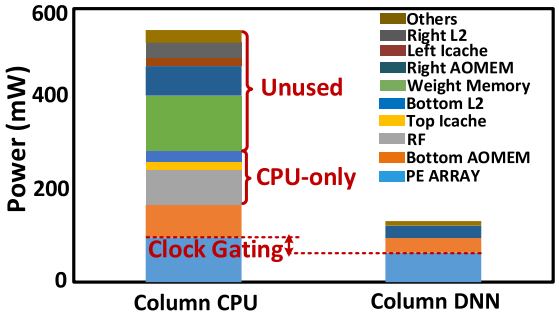


Fig. 18. Power breakdown for column CPU and DNN.

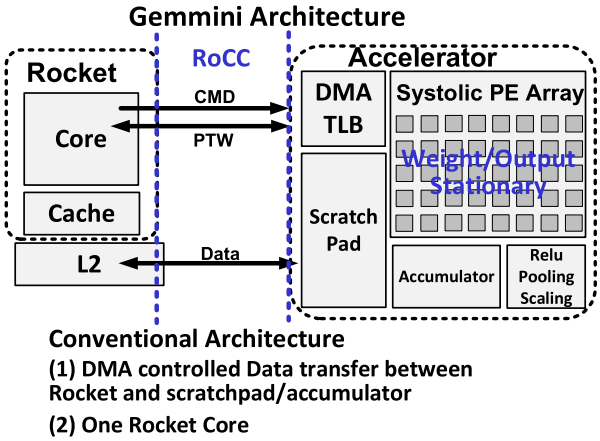


Fig. 19. Description of Gemmini architecture [14].

model evaluated) out of 32-b available logics for DNN modes. Hence, unused bits are gated off through the chip global setting.

C. Benchmark Results

The open-source RISC-V-based heterogeneous architecture, Gemmini [14], is utilized as the reference for comparison with SNCPU. Image classification tasks are evaluated with three different datasets, MNIST, CIFAR10, and ImageNet, using three different NN models, i.e., VGG16, ResNet18, and a simple ELU network. The ELU network with the MNIST database is a much smaller workload which helps to evaluate the performance of our design under a different model setting.

Fig. 19 shows a brief description of Gemmini SoC [14] for comparison. The Rocket core is one open-source in-order scalar processor with RISC-V ISA and five pipeline stages. Here, we use a Rocket core with a two-level MEM system with 32-b RISC-V ISA. RoCC is a RISC-V-based interface connecting the Rocket core with extensional modules. As shown in Fig. 19, by utilizing the RoCC interface, a 2-D systolic PE array with scratchpads and an accumulator is physically connected to the Rocket CPU core as a DNN accelerator. DMA engine is used to control the data transfer between the caches of the Rocket core and the scratchpads or accumulators of the accelerator.

We implemented end-to-end image classification operation using an 8-b quantized model for all three datasets and models.

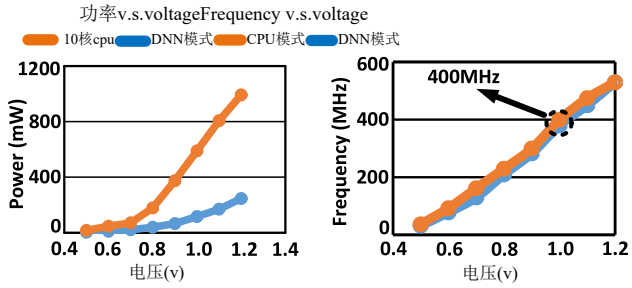


图16.测量功率和频率，带电压缩放。

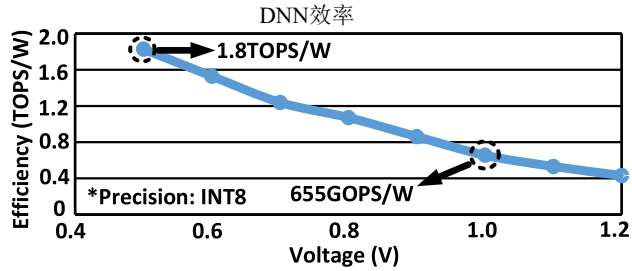


图17.采用电压缩放的DNN操作中SNCPU的能效。

首先处理DNN模型的第一层，耗时约60 μ秒。随后SN-CPU切换至列式CPU模式，通过填充和数据对齐为下一层准备输入数据。从380 μ秒开始，SNCPU以行加速器模式对DNN第二层进行十核运算。图15中仅展示了该模式下第二层的起始阶段运算过程。

各模式的PE利用率如图15所示。DNN模式的平均PE利用率约为99%，
CPU模式下的平均PE利用率约为96%，而整体平均PE利用率高达97%，这比传统异构架构[11][12]的性能表现显著提升。

图16展示了将电压缩放至0.5 V后的测量功率与频率数据。DNN模式和CPU模式的标称供电电压均为1.0 V，其中589mW的十核CPU功耗与116mW的DNN加速器功耗均在400 MHz频率下运行。图17则展示了将供电电压缩放至0.5 V后的能效表现，该结果基于INT8位精度计算得出。数据显示：DNN模式在1.0 V时实现655GOPS/W的性能，当电压降至0.5 V时，其运算能力提升至1.8TOPS/W。

如图18所示，与DNN模式相比，CPU模式下功耗升高的原因主要有三点。首先，指令缓存、二级缓存和射频单元等专用存储器显著增加了功耗。其次，虽然DNN模式采用了广泛的功率门控技术，但由于严格的芯片测试时间限制，CPU模式未进行同等程度的优化。许多未使用的SRAM存储库和逻辑电路在CPU模式下处于无门控状态，导致约45%的功耗浪费。第三，DNN模式通过可编程时钟门控技术调整所需位精度实现了额外节能。例如，每个处理单元（PE）中的累加器仅使用20位精度（基于VGG16模型）。

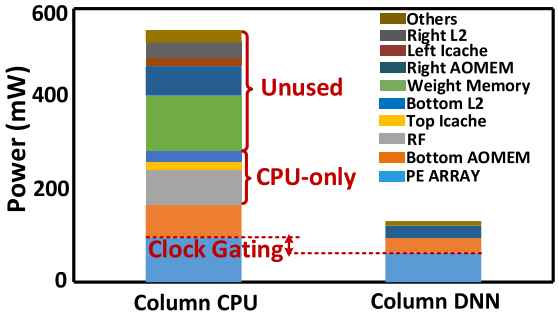


图18.柱CPU和DNN的功率损耗。

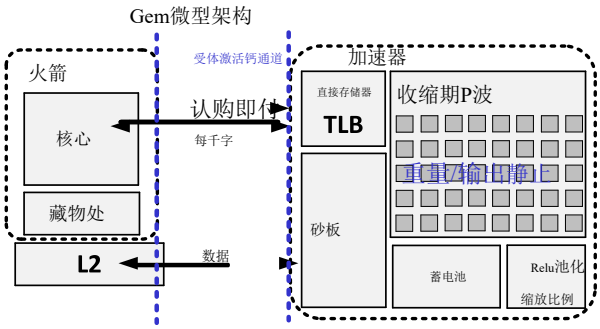


图19.Gemmini体系结构的描述[14]。

在DNN模式的32位可用逻辑中，模型评估结果显示未被使用。因此，未使用的位通过芯片全局设置被关闭。

C. 基准结果

我们采用基于开源RISC-V架构的异构系统Gemmini [14]作为与SNCPU对比的基准模型。针对图像分类任务，我们使用MNIST、CIFAR10和ImageNet三个不同数据集进行评估，并采用VGG16、ResNet18以及一个简单的ELU网络三种神经网络模型。其中，使用MNIST数据集的ELU网络工作量显著降低，这有助于在不同模型配置下评估我们设计方案的性能表现。

图19展示了Gemmini SoC [14]的简要说明以作对比。该芯片采用基于RISC-V指令集架构的开源顺序标量处理器Rocket核心，包含五个流水线阶段。我们在此使用配备32位RISC-V指令集的双级内存系统与Rocket核心配合工作。RoCC作为基于RISC-V的接口，将Rocket核心与扩展模块连接起来。如图所示，通过RoCC接口，一个包含暂存垫和累加器的二维脉动处理单元阵列被物理连接到Rocket CPU核心，作为深度神经网络加速器。DMA引擎则用于控制Rocket核心缓存与加速器暂存垫或累加器之间的数据传输。

我们使用8位量化模型对全部三个数据集和模型进行端到端图像分类操作。

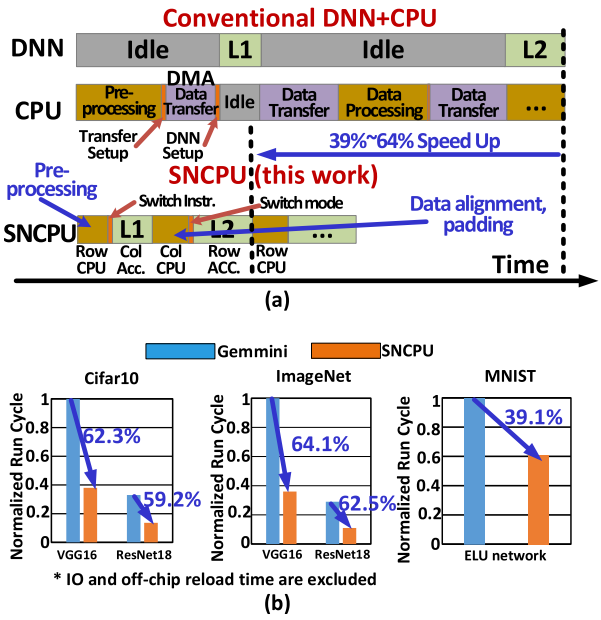


Fig. 20. Performance improvement compared with Gemmini [14]. (a) Explanation for the reasons of benefit for SNCPU. (b) Improvement details for CIFAR10, ImageNet, and MNIST datasets using VGG16, ResNet18, and simple ELU models.

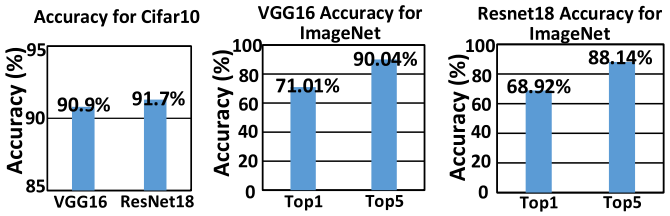


Fig. 21. Benchmark accuracy in this work.

As shown in Fig. 20(a), for each model, certain preprocessing works such as image processing, normalization, and inter-layer data preparation work between layer 1 processing (L1) and layer 2 processing (L2), are needed to be performed by CPU. For conventional heterogeneous architecture, there are two different parts of data transfer. First of all, input data prepared by the CPU needs to be transferred to the input scratchpad of the accelerator. In addition, finished data from the accelerator needs to be sent back to the CPU for general-purpose computation such as batch normalization, data alignment, and padding. The data transfer is controlled by the DMA engine with extra power consumption and takes around 30% of the run cycles in total. It also needs to pay extra run cycles for setting up the data transfer and the initialization of the accelerator before the work starts. Data transfer between different cores and the imbalance of CPU workload and accelerator workload cause the idle time for both the accelerator and CPU pipeline core. Especially for the DNN accelerator, it can reduce the core utilization rate and sacrifice the end-to-end latency of the image classification tasks.

In SNCPU, dual-mode bi-directional dataflow can keep data locally for both CPU and DNN processing avoiding data transfer effort compared with conventional architecture. Also, the unified architecture of the CPU and DNN accelerator eliminates the idle time for the SNCPU core leading to improvement of the core utilization. Because of the ten-core configuration, SNCPU can provide parallel processing capability for CPU work, which is much faster than the single-core CPU in Gemmini. As shown in Fig. 20, a 39%–64% total latency improvement was observed throughout image classification tasks. By processing identical programs for the same DNN model and the same dataset for both SNCPU and Gemmini design, the detailed improvement for image classification is shown in Fig. 20(b). As for the Cifar10 dataset, SNCPU shows 62.3% latency improvement for VGG16 and 59.2% for ResNet18. The SNCPU achieves 64.1% latency improvement for VGG16 and 62.5% for ResNet18 for ImageNet. The average latency benefit is around 61% due to 30% from the removal of data transfer and 31% from parallel processing by multi-core CPU. The workload of CPU processing impacts the latency improvement from SNCPU. Benefits drop from 61% to 39.1% with the three-layer FC ELU network on the simple MNIST dataset. The dropped benefit is due to the less CPU workload for preprocessing and less inter-layer data movement on the FC layers and in the simpler MNIST dataset. Because of the limitation of the test chip, MB level on-chip SRAM size which is enough for one layer processing of well-known DNN model such as VGG cannot be afforded. The benefit results assume that SNCPU and the heterogeneous architecture have the same SRAM size and the data reload latency from off-chip MEM such as DRAM is ignored. The reason is that off-chip data reload is necessary and similar for both architectures if the on-chip SRAM size is the same. SNCPU can only save on-chip data transfer as well as CPU processing acceleration. If considering off-chip data reload time, it may dominate the end-to-end latency for both architectures which is beyond the scope of chip design and make the discussion really complicated.

As shown in Fig. 21, the VGG16 and ResNet18 for the Cifar10 dataset can achieve around 91% accuracy with INT8 bit precision by using SNCPU. VGG16 for ImageNet dataset has 71.01% top 1 accuracy and 90.04% top 5 accuracy. ResNet18 can reach 68.92% top 1 accuracy and 88.14% top 5 accuracy on the ImageNet dataset on SNCPU.

D. Comparison Results

The comparison results are shown in Table I. Compared with prior reconfigurable binary NN (BNN) accelerator-based design [17], SNCPU achieves much higher throughput and can support the most commonly used DNN models. Compared with regular accelerator design [2], [3], SNCPU can support general-purpose computing which is important for real-time embedded applications. Table I also shows the comparison with three prior RISC-V-based heterogeneous SoC works, SamurAI [23] (1 × RISC-V + 1 × accelerator), Vega [24] (9 × RISC-V + 1 × accelerator) and Dustin [25] (16 × RISC-V). Compared with those SoC works, the DNN efficiency for 8 b is compatible with existing SoC demonstration at 0.66GOPS/W at 1 V and 1.8TOPS/W at 0.5 V. Compared with the 16-core RISC-V design in Dustin, the

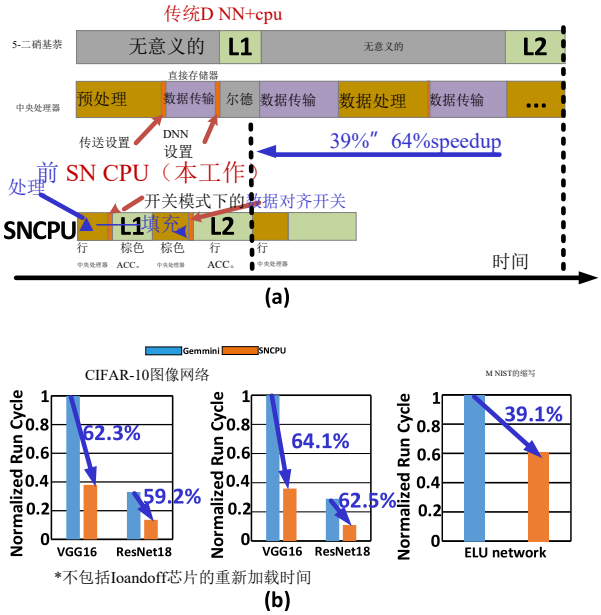


图20.与Gemmini [14]相比的性能提升对比。(a)SNCPU优势的原理说明。(b) 使用VGG16、ResNet18和简单ELU模型在CIFAR10、ImageNet和MNIST数据集上的改进细节。

如图20(a)所示，每个模型都需要CPU执行图像处理、归一化以及层间数据准备等预处理工作，这些工作发生在第一层处理（L1）与第二层处理（L2）之间。在传统异构架构中，数据传输存在两个不同环节：首先，CPU准备的输入数据需要传输到加速器的输入暂存区；其次，加速器完成的数据需要回传至CPU进行批量归一化、数据对齐和填充等通用计算操作。这种数据传输由DMA引擎控制，虽然功耗较高，但仅占总运行周期的30%左右。此外，在任务启动前还需额外消耗运行周期来配置数据传输参数并初始化加速器。不同核心之间的数据传输以及CPU与加速器工作负载的不平衡，导致加速器和CPU流水线核心都存在空闲时间。特别是对于深度神经网络加速器而言，这会降低核心利用率，甚至牺牲图像分类任务的端到端延迟。

在SNCPU架构中，双模式双向数据流可同时为CPU和DNN处理保留本地数据，相比传统架构显著减少数据传输需求。此外，CPU与DNN加速器的统一设计消除了SNCPU核心的空闲时间，从而提升核心利用率。得益于十核配置，SNCPU能为CPU任务提供并行处理能力，其速度远超Gemmini平台的单核CPU。如图20所示，在图像分类任务中观察到总延迟提升了39%-64%。通过针对相同DNN模型和数据集对SNCPU与Gemmini设计进行完全相同的程序处理，我们获得了图像处理的详细性能提升数据。

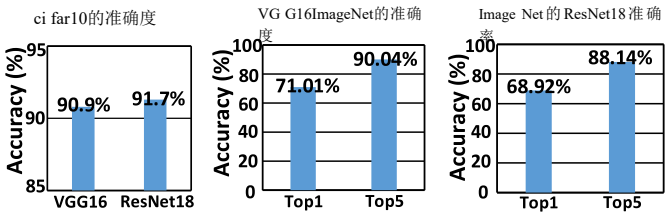


图21.本工作中的基准精度。

分类结果如图20(b)所示。针对Cifar10数据集，SNCPU在VGG16模型上实现了62.3%的延迟优化，在ResNet18模型上达到59.2%。在ImageNet数据集上，SNCPU对VGG16模型的延迟优化提升达64.1%，对ResNet18模型则提升62.5%。平均延迟优化率约为61%，其中30%得益于数据传输的省去，31%来自多核CPU的并行处理能力。CPU处理工作负载对SNCPU的延迟优化效果存在显著影响：在简单的MNIST数据集上，采用三层全连接ELU网络时，优化率从61%降至39.1%。这种性能下降主要源于预处理阶段CPU工作负载减少，以及全连接层间数据传输量降低和MNIST数据集本身的简化特性。受限于测试芯片规格，MB级片上静态随机存取存储器（SRAM）容量无法满足像VGG这类知名深度神经网络模型单层处理需求。上述优化效果假设SNCPU与异构架构共享相同SRAM容量，并忽略来自DRAM等片外存储器的数据重载延迟。这是因为若片上SRAM容量相同，两种架构都需要进行片外数据重载操作。SNCPU只能节省芯片上的数据传输和CPU处理加速，如果考虑片外数据的重新加载时间，它可能会主导两种体系结构的端到端延迟，这超出了芯片设计的范围，使讨论变得非常复杂。

如图21所示，在Cifar10数据集上使用SNCPU时，VGG16和ResNet18模型在INT8位精度下可达到约91%的准确率。针对ImageNet数据集的VGG16模型，其前1名准确率为71.01%，前5名准确率达90.04%。而ResNet18模型则能实现68.92%的前1名准确率和88.14%的前5名准确率。

在SNCPU上的ImageNet数据集上获得5的准确率。

D. 比较结果

对比结果详见表I。相较于先前基于可重构二进制神经网络（BNN）加速器的设计方案[17]，SNCPU展现出显著提升的吞吐量，并能支持主流深度神经网络模型。与常规加速器设计[2][3]相比，SNCPU可支持通用计算需求，这对实时嵌入式应用至关重要。表I还展示了与三款基于RISC-V架构的异构SoC方案的对比：SamurAI [23] (1×RISC-V + 1×加速器）、Vega [24] (9×RISC-V + 1×加速器）以及Dustin [25]（16×RISC-V）。在8位架构下，其深度神经网络效率与现有SoC方案表现相当——1V电压时达到0.66GOPS/W，0.5 V电压时可达1.8TOPS/W。相较于Dustin方案中的16核RISC-V设计，

TABLE I
COMPARISON WITH PRIOR WORK

	[17] NCPU	[2] Eyeriss	[3] DNPU	[24] SamurAI	[25] Vega	[26] Dustin	SNCPU
Process	CMOS 65nm	CMOS 65nm	CMOS 65nm	CMOS 28nm	CMOS 22nm	CMOS 65nm	CMOS 65nm
Area (mm2)	2.86	12.25	16	4.5	12	10	4.47
Architecture	CPU/BNN	CNN	CNN,FC,RNN	Hete. CPU+DNN	Hete. CPU+DNN	Hete. CPU	CPU/DNN
CPU	1x RISC-V	-	-	1x RISC-V	9x RISC-V	16x RISC-V	10x RISC-V
Application	IoT GP BNN	DNN	DNN	IoT GP DNN	IoT GP DNN	IoT GP DNN	IoT GP DNN
DNN Power	241mW	278mW	279mW	96mW (total power)	49.4mW (total power)	-	116mW
CPU Power	106mW	-	-	24.5mW (1x)	-	156mW (16x)	589mW (10x)
Int Precision	2,32	16	16	8,16,32	8,16,32	2,4,8,16,32	8,32
Max Freq.	960MHz	250MHz	200MHz	350MHz	450MHz	205MHz	400MHz
Supply Voltage	0.5-1.0V	0.82-1.17V	0.77-1.1V	0.45-0.9V	0.5-0.8V	0.8-1.2V	0.5-1.0V
SRAM	55.5KB	181KB	290KB	464KB	128KB (L1)	208KB	150KB
CPU Perf. (GOPS)	0.96 @1.0V (8b)	-	-	1.5 @ 0.9V (8b)	15.6 @ 0.8V (8b)	15 @ 1.2V (8b)	16 @ 1.0V (8b)
DNN Efficiency (TOPS/W)	1.6 @ 1V(1b) 6.0 @ 0.4V(1b)	0.24 @ 1V(16b)	1.0 @ 1.1V(16b) 2.1 @ 0.77V(16b)	0.38 @ 0.9V (8b) 1.3 @ 0.5V (8b)	0.9 @ 0.8V(8b) 1.35 @ 0.6V (8b)	0.303 @ 0.8V (8b) 1.15 @ 0.8V(2b)	0.66 @ 1V(8b) 1.8 @ 0.5V(8b)

TABLE II
COMPARISON WITH EXISTING CPUS

	[18] Microchip	[19] TI	[20] Microchip	[21] SiFive	SNCPU (CPU)
Datapath	**32b**	**32b**	**8b**	**32b**	**32b**
CPU	**ARM**	**ARM**	**RISC-V**	**RISC-V**	**RISC-V**
Pipe Stage	**8**	**3**	**2**	**5**	**5**
Voltage (V)	**1.26**	**3**	**3**	**1**	**1**
Freq (MHz)	**600**	**48**	**64**	**250**	**400**
Power (mW)	**229**	**22.8**	**37.2**	**150**	**58.9**
Performance (DMIPS/MHz)	**1.57**	**1.22**	**0.25**	**1.61**	**0.93**
Efficiency (DMIPS/mW)	**4.11**	**2.57**	**0.43**	**2.68**	**6.31**

performance of CPU mode of SNCPU achieves 16GOPS at 1 V which is similar to the performance of Dustin. The power of SNCPU is higher than Dustin due to the lack of optimization in CPU mode in SNCPU.

Since one CPU core in CPU modes of SNCPU is an in-order scalar processor, Table II lists the comparison of the CPU performance of SNCPU with four similar types of low-cost embedded commercial processors [18], [19], [20], [21]. In order to support the related general-purpose computing for DNN, complicated MEM system and pipeline optimizations are not very critical compared with commercial processors, which also provide competitive performance and power efficiency for CPU modes of SNCPU.

VI. CONCLUSION

This article presented a unified architecture, i.e., SNCPU, combining the systolic DNN accelerator and the RISC-V CPU core. SNCPU is designed based on a 10 × 10 2-D systolic array which can be reconfigured to a ten-core 32 b in-order RISC-V CPU. The design achieves over 95% PE utilization for ML tasks. It is implemented by using logic sharing inside the PE array and MEM reuse with 9.8% area overhead and 15% power overhead for DNN modes. A test chip was fabricated using 65-nm CMOS technology under 1.0-V supply voltage and 400-MHz frequency. The SNCPU achieves an energy efficiency of 655GOPS/W to 1.8TOPS/W from 1.0 to 0.5 V. Based on the evaluation using popular DNN datasets and models, the SNCPU achieves a 39%–64% speedup for end-to-end image classification tasks due to the enhanced data locality and parallel data processing.

REFERENCES

[1] K. Ueyoshi et al., "QUEST: A 7.49TOPS multi-purpose log-quantized DNN inference engine stacked on 96MB 3D SRAM using inductive-coupling technology in 40nm CMOS," in *IEEE Int. Solid-State Circuits Conf. (ISSCC) Dig. Tech. Papers*, Feb. 2018, pp. 160–161.

[2] Y. H. Chen, T. Krishna, J. S. Emer, and V. Sze, "Eyeriss: An energy-efficient reconfigurable accelerator for deep convolutional neural networks," *IEEE J. Solid-State Circuits*, vol. 52, no. 1, pp. 127–138, Jan. 2016.

[3] D. Shin, J. Lee, J. Lee, and H.-J. Yoo, "14.2 DNPU: An 8.1TOPS/W reconfigurable CNN-RNN processor for general-purpose deep neural networks," in *IEEE Int. Solid-State Circuits Conf. (ISSCC) Dig. Tech. Papers*, Feb. 2017, pp. 240–241.

[4] J. Lee, C. Kim, S. Kang, D. Shin, S. Kim, and H.-J. Yoo, "UNPU: An energy-efficient deep neural network accelerator with fully variable weight bit precision," *IEEE J. Solid-State Circuits*, vol. 54, no. 1, pp. 173–185, Jan. 2019.

[5] T. Jia, Y. Ju, and J. Gu, "31.3 A compute-adaptive elastic clock-chain technique with dynamic timing enhancement for 2D PE-array-based accelerators," in *IEEE Int. Solid-State Circuits Conf. (ISSCC) Dig. Tech. Papers*, Feb. 2020, pp. 482–484.

[6] H. E. Sumbul et al., "System-level design and integration of a prototype AR/VR hardware featuring a custom low-power DNN accelerator chip in 7nm technology for codec avatars," in *Proc. IEEE Custom Integr. Circuits Conf. (CICC)*, Apr. 2022, pp. 1–8.

[7] Y.-D. Chih et al., "16.4 an 89TOPS/W and 16.3TOPS/mm^2 all-digital SRAM-based full-precision compute-in memory macro in 22nm for machine-learning edge applications," in *IEEE Int. Solid-State Circuits Conf. (ISSCC) Dig. Tech. Papers*, Feb. 2021, pp. 252–254.

[8] J. Yue et al., "14.3 A 65nm computing-in-memory-based CNN processor with 2.9-to-35.8TOPS/W system energy efficiency using dynamic-sparsity performance-scaling architecture and energy-efficient inter/intra-macro data reuse," in *IEEE Int. Solid-State Circuits Conf. (ISSCC) Dig. Tech. Papers*, Feb. 2020, pp. 234–236.

表一

COMPARISON WITH PRIOR WORK

	[17] NCPU	[2] Eyeriss	[3] DNPU	[24] SamurAI	[25] Vega	[26] Dustin	SNCPU
Process	CMOS 65nm	CMOS 65nm	CMOS 65nm	CMOS 28nm	CMOS 22nm	CMOS 65nm	CMOS 65nm
Area (mm2)	2.86	12.25	16	4.5	12	10	4.47
Architecture	CPU/BNN	CNN	CNN,FC,RNN	Hete. CPU+DNN	Hete. CPU+DNN	Hete. CPU	CPU/DNN
CPU	1x RISC-V	-	-	1x RISC-V	9x RISC-V	16x RISC-V	10x RISC-V
Application	IoT GP BNN	DNN	DNN	IoT GP DNN	IoT GP DNN	IoT GP DNN	IoT GP DNN
DNN Power	241mW	278mW	279mW	96mW (total power)	49.4mW (total power)	-	116mW
CPU Power	106mW	-	-	24.5mW (1x)	-	156mW (16x)	589mW (10x)
Int Precision	2,32	16	16	8,16,32	8,16,32	2,4,8,16,32	8,32
Max Freq.	960MHz	250MHz	200MHz	350MHz	450MHz	205MHz	400MHz
Supply Voltage	0.5-1.0V	0.82-1.17V	0.77-1.1V	0.45-0.9V	0.5-0.8V	0.8-1.2V	0.5-1.0V
SRAM	55.5KB	181KB	290KB	464KB	128KB (L1)	208KB	150KB
CPU Perf. (GOPS)	0.96 @1.0V (8b)	-	-	1.5 @ 0.9V (8b)	15.6 @ 0.8V (8b)	15 @ 1.2V (8b)	16 @ 1.0V (8b)
DNN Efficiency (TOPS/W)	1.6 @ 1V(1b) 6.0 @ 0.4V(1b)	0.24 @ 1V(16b)	1.0 @ 1.1V(16b) 2.1 @ 0.77V(16b)	0.38 @ 0.9V (8b) 1.3 @ 0.5V (8b)	0.9 @ 0.8V(8b) 1.35 @ 0.6V (8b)	0.303 @ 0.8V (8b) 1.15 @ 0.8V(2b)	0.66 @ 1V(8b) 1.8 @ 0.5V(8b)

表II

C OMPARISON W Ith E Xistindo CUPOS S

	[18] Microchip	[19] TI	[20] Microchip	[21] SiFive	SNCPU (CPU)
Datapath	**32b**	**32b**	**8b**	**32b**	**32b**
CPU	**ARM**	**ARM**	**RISC-V**	**RISC-V**	**RISC-V**
Pipe Stage	**8**	**3**	**2**	**5**	**5**
Voltage (V)	**1.26**	**3**	**3**	**1**	**1**
Freq (MHz)	**600**	**48**	**64**	**250**	**400**
Power (mW)	**229**	**22.8**	**37.2**	**150**	**58.9**
Performance (DMIPS/MHz)	**1.57**	**1.22**	**0.25**	**1.61**	**0.93**
Efficiency (DMIPS/mW)	**4.11**	**2.57**	**0.43**	**2.68**	**6.31**

SNCPU的CPU模式性能在1V电压下达到16GOPS，与Dustin的性能相当。由于SNCPU的CPU模式缺乏优化，其性能优于Dustin。

由于SNCPU的CPU模式中单个核心属于顺序标量处理器，表II对比了SNCPU与四种类似低成本商用嵌入式处理器[18][19][20][21]的CPU性能表现。相较于商用处理器，SNCPU在CPU模式下同样能提供具有竞争力的性能和能效，因此在支持深度神经网络相关通用计算任务时，复杂的内存系统和流水线优化措施并非关键所在。

VI. CONCLUSION

本文提出了一种统一架构——SNCPU，将脉动式深度神经网络加速器与RISC-V CPU核心相结合。该架构基于10×10二维脉动阵列设计，可重构为十核32位顺序RISC-V CPU。通过优化设计，实现了超过95%的处理单元利用率。多任务处理模块通过在处理器阵列内部实现逻辑共享，并采用内存复用技术来实现。在深度神经网络（DNN）模式下，该方案的面积开销为9.8%，功耗开销达15%。测试芯片采用65纳米CMOS工艺制造，工作电压1.0V，频率400MHz。SNCPU在1.0-0.5V电压范围内实现了655GOPS/W至1.8TOPS/W的能效比。基于对主流DNN数据集和模型的评估，得益于增强的数据局部性和并行数据处理能力，该处理器在端到端图像分类任务中实现了39%-64%的加速效果。

REFERENCES

[1] K.上吉等人，《QUEST：一种基于40纳米CMOS工艺、采用电感耦合技术堆叠在96MB三维静态随机存取存储器上的7.49TOPS多用途对数量化深度神经网络推理引擎》，载于*IEEE国际固态电路会议（ISSCC）数字技术论文集*，2018年2月，第160–161页。

[2] 陈永华、T·克里希纳、J·S·埃默与V·斯在《Eyeriss：一种适用于深度卷积神经网络的节能可重构加速器》中提出，该技术发表于2016年1月的*《IEEE固态电路杂志》*第52卷第1期，页码127-138。

[3] D. Shin，J. Lee，J. Lee和H.-J. Yoo，《14.2 DNPU：一种8.1TOPS/W可重构CNN-RNN处理器用于通用深度神经网络》，载于*IEEE国际固态电路会议（ISSCC）数字技术论文集*，2017年2月，第240–241页。

[4] J. Lee、C. Kim、S. Kang、D. Shin、S. Kim和H.-J. Yoo，“UNPU：一种具有完全可变权重位精度的节能深度神经网络加速器”，*IEEE固态电路杂志*，第54卷，第1期，第173-185页，2019年1月。

[5] 贾涛、朱宇和顾军，《31.3 A计算自适应弹性时钟链技术及动态时序增强方案在二维PE阵列加速器中的应用》，载于*IEEE国际固态电路会议（ISSCC）数字技术论文集*，2020年2月，第482-484页。

[6] H·E·桑布尔等人，《基于7纳米制程定制低功耗深度神经网络加速器芯片的AR/VR原型硬件系统级设计与集成——以编解码器虚拟形象为例》，载于*《IEEE定制集成电路会议（CICC）论文集》*，2022年4月，第1–8页。

[7] Y.-D. Chih等，《基于全数字SRAM的22纳米全精度计算存储器宏单元（支持89TOPS/W功耗与16.3TOPS/mm 2），专用于机器学习边缘计算应用》，载于*IEEE国际固态电路会议（ISSCC）数字技术论文集*，2021年2月，第252–254页。

[8] J. Yue等人在《基于动态稀疏性性能缩放架构与节能宏/中宏数据复用的14.3A 65nm计算内存型CNN处理器：2.9-to-35.8TOPS/W系统能效》一文中提出，该方案通过动态稀疏性性能缩放架构和节能宏/中宏数据复用技术，在2020年2月*IEEE国际固态电路会议（ISSCC）数字技术论文集*第234–236页发表。

[9] J. Wang et al., "A 28-nm compute SRAM with bit-serial logic/arithmetic operations for programmable in-memory vector computing," *IEEE J. Solid-State Circuits*, vol. 55, no. 1, pp. 76–86, Jan. 2020.
[10] X. Si et al., "15.5 A 28nm 64Kb 6T SRAM computing-in-memory macro with 8b MAC operation for AI edge chips," in *IEEE Int. Solid-State Circuits Conf. (ISSCC) Dig. Tech. Papers*, Feb. 2020, pp. 246–248.
[11] T. Karnik et al., "A cm-scale self-powered intelligent and secure IoT edge mote featuring an ultra-low-power SoC in 14nm tri-gate CMOS," in *IEEE Int. Solid-State Circuits Conf. (ISSCC) Dig. Tech. Papers*, Feb. 2018, pp. 46–48.
[12] R. Eki et al., "9.6 A 1/2.3inch 12.3Mpixel with on-chip 4.97TOPS/W CNN processor back-illuminated stacked CMOS image sensor," in *IEEE Int. Solid-State Circuits Conf. (ISSCC) Dig. Tech. Papers*, Feb. 2021, pp. 154–156.
[13] N. P. Jouppi et al., "In-datacenter performance analysis of a tensor processing unit," in *Proc. ACM/IEEE 44th Annu. Int. Symp. Comput. Archit. (ISCA)*, Jun. 2017, pp. 1–12.
[14] H. Genc et al., "Gemmini: Enabling systematic deep-learning architecture evaluation via full-stack integration," in *Proc. 58th ACM/IEEE Design Autom. Conf. (DAC)*, Dec. 2021, pp. 769–774.
[15] S. L. Xi, Y. Yao, K. Bhardwaj, P. Whatmough, G.-Y. Wei, and D. Brooks, "SMAUG: End-to-end full-stack simulation infrastructure for deep learning workloads," *ACM Trans. Archit. Code Optim.*, vol. 17, no. 4, pp. 1–26, Nov. 2020.
[16] P. N. Whatmough et al., "A 16nm 25 mm2 SoC with a 54.5x flexibility-efficiency range from dual-core arm cortex-A53 to eFPGA and cache-coherent accelerators," in *Proc. Symp. VLSI Circuits*, Jun. 2019, pp. C34–C35.
[17] T. Jia, Y. Ju, R. Joseph, and J. Gu, "NCPU: An embedded neural CPU architecture on resource-constrained low power devices for real-time end-to-end performance," in *Proc. 53rd Annu. IEEE/ACM Int. Symp. Microarchitecture (MICRO)*, Oct. 2020, pp. 1097–1109.
[18] Online Resource, Microchip. *ATSAMA5D44*. Accessed: Mar. 23, 2022. [Online]. Available: https://www.microchip.com/wwwproducts/en/ATsama5d44
[19] Online Resource, Texas Instruments. *MSP432P401R*. Accessed: Mar. 23, 2022. [Online]. Available: https://www.ti.com/product/MSP432P401R
[20] Online Resource, Microchip. *PIC18F13K22*. Accessed: Mar. 23, 2022. [Online]. Available: https://www.microchip.com/wwwproducts/en/PIC18F13K22
[21] Online Resource, SiFive. *E31*. Accessed: Mar. 23, 2022. [Online]. Available: https://www.sifive.com/cores/e31
[22] Y. Ju and J. Gu, "A 65nm systolic neural CPU processor for combined deep learning and general-purpose computing with 95% PE utilization, high data locality and enhanced end-to-end performance," in *IEEE Int. Solid-State Circuits Conf. (ISSCC) Dig. Tech. Papers*, Feb. 2022, pp. 1–3.
[23] I. Miro-Panades et al., "SamurAI: A 1.7MOPS-36GOPS adaptive versatile IoT node with 15,000× peak-to-idle power reduction, 207ns wake-up time and 1.3TOPS/W ML efficiency," in *Proc. IEEE Symp. VLSI Circuits*, Jun. 2020, pp. 1–2.
[24] D. Rossi et al., "Vega: A ten-core SoC for IoT endnodes with DNN acceleration and cognitive wake-up from MRAM-based state-retentive sleep mode," *IEEE J. Solid-State Circuits*, vol. 57, no. 1, pp. 127–139, Jan. 2022.
[25] A. Garofalo et al., "A 1.15 TOPS/W, 16-cores parallel ultra-low power cluster with 2b-to-32b fully flexible bit-precision and vector lockstep execution mode," in *Proc. IEEE 47th Eur. Solid State Circuits Conf. (ESSCIRC)*, Sep. 2021, pp. 267–270.

Yuhao Ju (Student Member, IEEE) received the B.S. degree from the University of Electronic Science and Technology of China, Chengdu, China, in 2017, and the M.S. degree from Northwestern University, Evanston, IL, USA, in 2019, where he is currently pursuing the Ph.D. degree in computer engineering.

His current research interests include computer architecture and machine learning accelerator design.

Jie Gu (Senior Member, IEEE) received the B.S. degree from Tsinghua University, Beijing, China, in 2001, the M.S. degree from Texas A&M University, College Station, TX, USA, in 2003, and the Ph.D. degree from the University of Minnesota, Minneapolis, MN, USA, in 2008.

He was an IC Design Engineer with Texas Instruments, Dallas, TX, USA, from 2008 to 2010, with a focus on ultralow-voltage mobile processor design and integrated power management techniques. He was a Senior Staff Engineer with Maxlinear, Inc., Carlsbad, CA, USA from 2011 to 2014, focusing on low-power mixed-signal broadband SoC design. He is currently an Associate Professor with Northwestern University, Evanston, IL, USA. His current research interests include novel circuits and architectures for emerging applications.

Dr. Gu was a recipient of the NSF CAREER Award. He has served as program committee and conference co-chair for numerous low-power design conferences and journals, such as ISPLED, DAC, ICCAD, and ICCD.

[9] 王杰等，《一种采用位串行逻辑/算术运算的28纳米计算SRAM用于可编程内存矢量计算》，*IEEE固态电路杂志*，第55卷，第1期，第76-86页，2020年1月。
[10] X. Si等，《15.5 A 28nm 64Kb 6T SRAM计算内存宏单元及8位MAC运算架构用于AI边缘芯片》，载于*IEEE国际固态电路会议（ISSCC）数字技术论文集*，2020年2月，第246–248页。
[11] T. Karnik等人，“一种厘米级自供电智能安全物联网边缘mote，采用14nm三栅极CMOS超低功耗SoC”，*IEEE国际固态电路会议（ISSCC）数字技术论文集*，2018年2月，第46-48页。
[12] R. Eki等人，“9.6 A 1/2.3英寸12.3Mpixel芯片，配备片上4.97TOPS/W CNN处理器背照式堆叠CMOS图像传感器”，载于*IEEE国际固态电路会议（ISSCC）数字技术论文集*，2021年2月，第154–156页。
[13] N. P. Jouppi等人，《张量处理单元的数据中心性能分析》，载于*《ACM/IEEE第44届国际计算机体系结构年会（ISCA）论文集》*，2017年6月，第1–12页。
[14] H. Genc等人，《Gemmini：通过全栈集成实现系统深度学习架构评估》，收录于*第58届ACM/IEEE设计自动化会议（DAC）论文集*，2021年12月，第769–774页。
[15] S·L·希、Y·姚、K·巴德瓦杰、P·沃特莫、G·Y·魏和D·布鲁克斯，《SM-AUG：面向深度学习工作负载的端到端全栈仿真基础设施》，*《ACM体系结构与代码优化汇刊》*，第17卷第4期，第1–26页，2020年11月。
[16] P. N. Whatmough等人的论文《16纳米25平方毫米SoC：从双核arm cortex-A53到eFPGA和缓存一致性加速器的54.5倍灵活性-效率范围》，发表于2019年6月*《超大规模集成电路研讨会论文集》*，第C34–C35页。
[17] 贾涛、朱宇、约瑟夫·R.和顾杰，《NCPU：一种基于资源受限低功耗设备的嵌入式神经CPU架构，用于实时端到端性能》，载于*第53届IEEE/ACM国际微架构年会（MICRO）论文集*，2020年10月，第1097–1109页。
[18] 在线资源，Microchip。*ATSAMA5D44*。访问日期：2022年3月23日。[在线]。来源：https://www.microchip.com/wwwproducts/en/ ATsama5d44
[19] 在线资源，德州仪器。*MSP432P401R*。访问日期：2022年3月23日。[在线]。可获取于：https://www.ti.com/product/ MSP432P401R
[20] 在线资源，Microchip。PIC18F13K22。访问日期：2022年3月23日。[在线]。来源：https://www.microchip.com/wwwproducts/en/ PIC18F13K22
[21] 在线资源，SiFive。*E31*。访问日期：2022年3月23日。[在线资源]。可获取地址：https://www.si five.com/cores/e31
[22] Y. Ju和J. Gu，“一种65纳米脉动神经CPU处理器，用于深度学习与通用计算的结合，具有95%的PE利用率、高数据局部性和增强的端到端性能”，见于*IEEE国际固态电路会议（ISSCC）数字技术论文集*，2022年2月，第1–3页。
[23] I.米罗-帕纳德斯等人，《SamurAI：一款1.7MOPS-36GOPS自适应多功能物联网节点，具备15,000×峰值至空闲功耗降低、207ns唤醒时间及1.3 TOPS/W机器学习效率》，载于《*IEEE超大规模集成电路研讨会论文集*》2020年6月卷第1–2页。
[24] D. Rossi等人，“Vega：一种用于物联网终端节点的十核SoC，具有DNN加速和基于MRAM的状态保持睡眠模式的认知唤醒”，*IEEE固态电路杂志*，第57卷，第1期，第127-139页，2022年1月。
[25] A. Garofalo等人，《一种1.15 TOPS/W、16核并行超低功耗集群，支持2b至32b全灵活位精度及矢量同步执行模式》，载于*《IEEE第47届欧洲固态电路会议（ESSCIRC）论文集》*，2021年9月，第267–270页。

Ju Yuhao（IEEE学生会员）于2017年获得中国电子科技大学学士学位，2019年获得美国西北大学硕士学位，目前在美国西北大学攻读计算机工程博士学位。

目前的研究方向包括计算机体系结构和机器学习加速器设计。

Jie Gu（IEEE高级会员）于2001年获得中国北京清华大学学士学位，2003年获得美国得克萨斯州College Station得克萨斯农工大学硕士学位，2008年获得美国明尼苏达州Minneapolis明尼苏达大学博士学位。

2008年至2010年，他在美国得克萨斯州达拉斯市的德州仪器公司担任集成电路设计工程师，专注于超低电压移动处理器设计与集成电源管理技术。2011年至2014年，他出任美国加利福尼亚州卡尔斯巴德市麦克斯线性公司高级工程师，主要研究低功耗混合信号宽带SoC设计。目前，他担任美国伊利诺伊州埃文斯顿市西北大学副教授，研究方向涵盖新兴应用领域的新颖电路与架构设计。

顾博士曾获美国国家科学基金会职业奖（NSF CAREER Award）。他担任过多个低功耗设计会议和期刊的程序委员会和会议联合主席，例如ISPLED、DAC、ICCAD和ICCD。