A 0.32–128 TOPS, Scalable Multi-Chip-Module-Based Deep Neural Network Inference Accelerator With Ground-Referenced Signaling in 16 nm

Brian Zimmer, *Member, IEEE*, Rangharajan Venkatesan, *Member, IEEE*, Yakun Sophia Shao, *Member, IEEE*, Jason Clemons, *Member, IEEE*, Matthew Fojtik, *Member, IEEE*, Nan Jiang, *Member, IEEE*, Ben Keller, *Member, IEEE*, Alicia Klinefelter, *Member, IEEE*, Nathaniel Pinckney, *Member, IEEE*, Priyanka Raina, *Member, IEEE*, Stephen G. Tell, *Member, IEEE*, Yanqing Zhang, *Member, IEEE*, William J. Dally, *Fellow, IEEE*, Joel S. Emer, *Fellow, IEEE*, C. Thomas Gray, *Senior Member, IEEE*, Stephen W. Keckler, *Fellow, IEEE*, and Brucek Khailany, *Senior Member, IEEE*

***Abstract*— Custom accelerators improve the energy efficiency, area efficiency, and performance of deep neural network (DNN) inference. This article presents a scalable DNN accelerator consisting of 36 chips connected in a mesh network on a multi-chip-module (MCM) using ground-referenced signaling (GRS). While previous accelerators fabricated on a single monolithic chip are optimal for specific network sizes, the proposed architecture enables flexible scaling for efficient inference on a wide range of DNNs, from mobile to data center domains. Communication energy is minimized with large on-chip distributed weight storage and a hierarchical network-on-chip and network-on-package, and inference energy is minimized through extensive data reuse. The 16-nm prototype achieves 1.29-TOPS/mm^2 area efficiency, 0.11 pJ/op (9.5 TOPS/W) energy efficiency, 4.01-TOPS peak performance for a one-chip system, and 127.8 peak TOPS and 1903 images/s ResNet-50 batch-1 inference for a 36-chip system.**

***Index Terms*— Deep neural networks (DNNs), ground-referenced signaling (GRS), inference accelerator, multi-chip modules, single-ended signaling.**

I. Introduction

DEEP neural networks (DNNs) are extremely popular and have been adopted to solve problems in a wide variety of fields, including image recognition [1]–[3], semantic segmentation [4], language translation [4], and autonomous driving [5]. DNN inference is currently performed on a range of traditional computing systems, including CPUs, field-programmable gate arrays (FPGAs), and GPUs, which provide different tradeoffs between efficiency, cost, performance, and programmability.

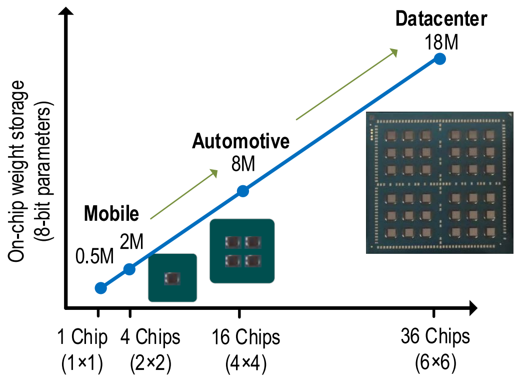


Fig. 1. Composing an MCM from different numbers of chiplets addresses a range of performance requirements with a single chip.

Due to the deterministic structure of DNNs, fixed-function accelerators have the potential to further improve area efficiency, energy efficiency, and performance relative to CPUs and GPUs [6]–[12]. However, the absolute performance requirements of DNNs vary from tiny networks on energy-constrained edge devices to large networks in data centers. It is extremely expensive to build a separate chip for each of these applications, as each domain has widely different compute and memory bandwidth requirements. Additionally, in such a rapidly changing field, it is difficult to predict DNN requirements years in advance, the lead time required to develop a custom accelerator for a target market.

The goal of this article is to design a system that can easily scale across many applications, where weight storage requirements range from 0.24 MB in DriveNet [5] to 24 MB in ResNet-50 [3], with a single silicon chip. The main idea is to build one small chip and connect a variable number of these chips together on a package to form a multi-chip module (MCM), as shown in Fig. 1. MCM-based systems offer a

Manuscript received August 15, 2019; revised November 2, 2019; accepted December 5, 2019. Date of publication January 14, 2020; date of current version March 26, 2020. This article was approved by Guest Editor Ken Takeuchi. This work was supported in part by the U.S. Government through the DARPA CRAFT Program. The views and conclusions contained in this document are those of the authors and should not be interpreted as representing the official policies, either expressed or implied, of the U.S. Government. Distribution Statement “A” (Approved for Public Release, Distribution Unlimited). *(Corresponding author: Brian Zimmer.)*

Brian Zimmer, Rangharajan Venkatesan, Ben Keller, Yanqing Zhang, and William J. Dally are with NVIDIA Corporation, Santa Clara, CA 94305 USA (e-mail: bzimmer@nvidia.com).

Yakun Sophia Shao is with the Department of Electrical Engineering and Computer Sciences, University of California at Berkeley, Berkeley, CA 94720 USA.

Jason Clemons, Nathaniel Pinckney, Stephen W. Keckler, and Brucek Khailany are with NVIDIA Corporation, Austin, TX 78717 USA.

Matthew Fojtik, Alicia Klinefelter, Stephen G. Tell, and C. Thomas Gray are with NVIDIA Corporation, Durham, NC 27712 USA.

Nan Jiang is with NVIDIA Corporation, St. Louis, MO 63144 USA.

Priyanka Raina is with the Department of Electrical Engineering, Stanford University, Stanford, CA 94305 USA.

Joel S. Emer is with NVIDIA Corporation, Westford, MA 01886 USA, and also with the Massachusetts Institute of Technology, Cambridge, MA 02139 USA.

Color versions of one or more of the figures in this article are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/JSSC.2019.2960488

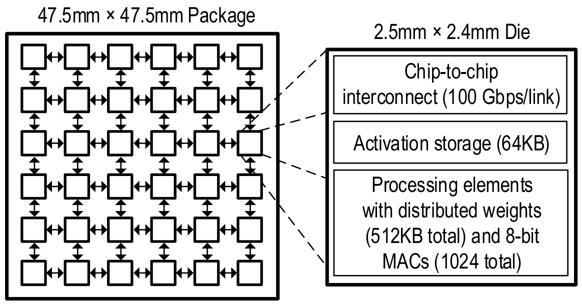

47.5mm × 47.5mm Package
2.5mm × 2.4mm Die
Chip-to-chip interconnect (100 Gbps/link)
Activation storage (64KB)
Processing elements with distributed weights (512KB total) and 8-bit MACs (1024 total)

Fig. 2. Prototype system with 36 chips connected on a package.

variety of benefits [13], [14]. Large designs are sometimes reticle limited, and MCMs provide a method to increase system capacity without requiring board-level or system-level integration. Smaller chips have a higher yield, as a fixed number of defects per wafer breaks fewer total chips as the number of chips on a wafer grows. Smaller chips are simpler and easier to design. A system can mix silicon dice from different process nodes to improve design reuse and reduce cost. The acceleration of new networks can be achieved by quickly and easily repackaging existing chips into an optimally sized system without waiting for the development and fabrication of new chips. Despite all these advantages, MCMs have area, performance, and power penalties relative to a large monolithic chip because chip-to-chip communication is more expensive than on-chip communication. The proposed system limits this penalty through energy-efficient chip-to-chip links, a hierarchical interconnect network, algorithmic optimizations to exploit the latency insensitive nature of architecture, and flexible dataflow mapping.

Implementing an MCM-based neural network inference accelerator posed some unique challenges. To achieve good performance for modern networks, such as ResNet-50 [3], the scale of the system must be very large—compared to recently published 8-bit neural network accelerators [6]–[9], the prototype system has 15–160× the on-chip static random-access memory (SRAM) storage, and over 67–1300× the performance. The system was designed to achieve strong scaling, in which increased compute capacity directly reduces latency, instead of weak scaling, where larger batches are used to perform more work in parallel but with the same latency. Because there is only one unique chip in the system, the architecture must be efficient for both small single-chip configurations and huge 36-chip configurations.

The prototype system architecture, shown in Fig. 2, combines 36 identical chips on an organic package to form a large-scale neural network accelerator. Each chip on the package is 2.5 mm × 2.4 mm, and 36 are connected on a 47.5 mm × 47.5 mm organic substrate. Each chip, described in Section II, has 752 kB of total SRAM storage and can perform 1024 multiply-and-accumulates (MACs) per cycle to execute smaller DNNs individually. Each package, described in Section III, combines 36 dice together in a mesh network with 100 Gb/s interconnect in each link direction to execute larger networks with 22.5 MB of on-chip SRAM and 36 864 MACs per cycle. The prototype system has no high-bandwidth external memory and is designed for networks with models that fit entirely on-package. Performance ranges from 4 TOPS for a one-chip system to 128 TOPS for a 36-chip system. The chip was fabricated in a TSMC 16-nm FinFET process and implemented with an agile design methodology further discussed in Section IV. Experimental results in Section V discuss energy-efficiency and performance measurements for a peak performance benchmark and ResNet-50.

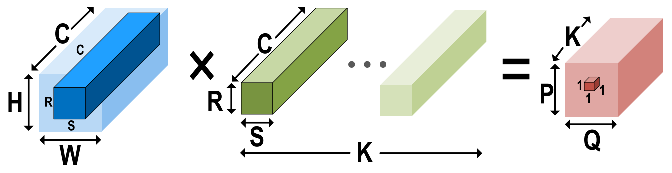


Fig. 3. Generation of one output element during convolution.

II. Single-Chip Architecture

Each chip in the system can operate as a standalone neural network accelerator for smaller networks.

A. System Operation

DNNs are composed of a series of many layers, including convolutional layers, pooling layers, activation layers, and fully connected layers. Each layer processes an input activation tensor from the previous layer and creates an output activation tensor for the next layer. The key workload kernel executed by this neural network accelerator is the convolution of a single layer shown in Fig. 3. An input activation tensor with size H × W and C input channels is convolved with a weight tensor with size R × S and C channels. After striding R × S across H × W, an output of size P × Q is formed. K different weight kernels contribute to each of the K output channels in the output activation tensor. Each element of the output activation tensor is formed from the MAC of R × S × C elements from the input and weight tensors, and this MAC is repeated P × Q × K times. Layer dimensions vary for each layer in the DNN, so the total number of MAC operations required can vary from 0.6–14 M in DriveNet [5] to 50–100 M in ResNet-50 [3]. These MAC operations are distributed spatially across multiple MAC datapaths and temporally within each datapath.

The system diagram shown in Fig. 4 describes the data movement required to execute a convolution. Weight tensors are loaded once at startup through general-purpose input–output (GPIO) from a host and distributed across the network-on-chip (NoC) to SRAMs inside each of the 16 processing elements (PEs) [Fig. 4(a)]. Each weight tensor can be split among the PEs if each PE computes separate input–output channels or replicated on different PEs so that each PE can work on a different portion of the output activation in parallel. Each PE and the global buffer (GB) operate autonomously after receiving a go command from the RISC-V processor (RVP) and are controlled by local configurable state machines [Fig. 4(b)]. Input activation tensors are loaded through GPIO and are stored in large SRAMs in the GB [Fig. 4(c)]. The GB sends the input activation tensor to the PEs, and the tensor can be split among the PEs along a

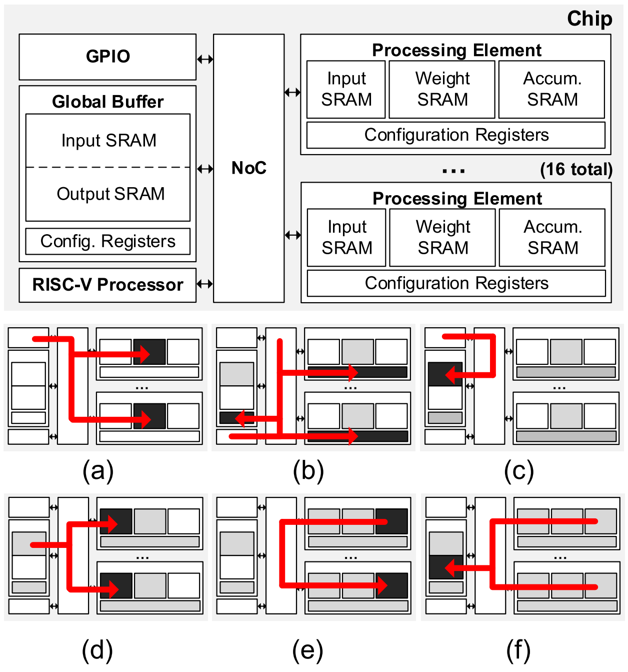


Fig. 4. Flow of data across the NoC router to execute a layer. (a) Weights. (b) Configuration. (c) Input activation. (d) Compute. (e) Accumulation. (f) Output activation.

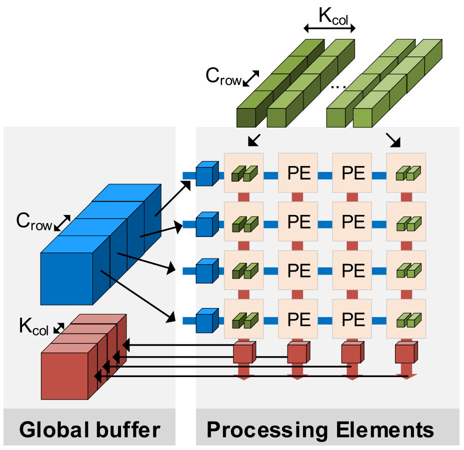


Fig. 5. Example mapping splits input channels along rows and output channels along columns.

layer dimension (such as input channel) or replicated so that each PE can compute a different output channel in parallel [Fig. 4(d)]. Each PE performs 64 MACs per cycle for a total of 1024 MACs per cycle in the chip. Accumulation of intermediate sums occurs either within the PE or across multiple PEs [Fig. 4(e)]. Each PE sends its respective portion of the output activation tensor to the GB to finish layer computation [Fig. 4(f)].

Each PE contains weights from multiple layers, and the chip cumulatively holds the weights for every layer, so the computation of subsequent layers and frames begins immediately without requiring redistribution of weights.

B. Mapping Convolutions

Fig. 5 describes one possible mapping of a convolution onto the chip. The input channels are split into four parts, and each part is multicast to the four PEs in each row of the PE array.

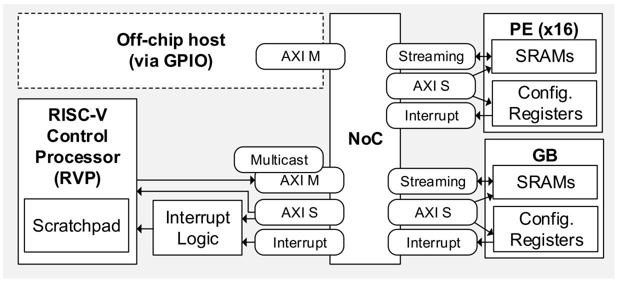


Fig. 6. Communication interface capabilities of the NoC.

The weight tensors are allocated between the four columns of the PE array and replicated into each PE along the column. All 16 PEs perform their MACs on the partitioned volumes in parallel. Then, the partial sums of the top row are accumulated row by row until the final row of PEs has accumulated the contributions from all C channels back together to form the final output activation.

Valid mappings and communication overheads are determined by the layer dimensions, as well as the PE and GB SRAM sizes. The architecture was designed to be very flexible and allows tiling across many different dimensions to achieve high efficiency across layers with very different dimensions. Layers with large K benefit from partitioning weight tensors between PEs and multicasting the input activation to each PE at the cost of more input activation traffic. Layers with large $H \times W$ benefit from replicating the weight kernels to minimize input activation traffic at the cost of greater weight storage requirements. Layers with large C benefit from splitting the input channels across PEs to minimize input traffic and weight storage at the cost of output accumulation traffic. The key to this architecture is that the weights remain stationary and are reused across multiple inputs. Only input activations, output partial sums, and output activations must be transported across the NoC, which reduces network bandwidth requirements. The main traffic demand on the network is input activation multicast, and the network bandwidth is provisioned to provide new 64-bit input to each PE in every cycle. Once the input is distributed, it is reused for many cycles of computing, which frees the network to send the smaller partial sums and output activations at the end of layer computation.

C. NoC

The on-chip network, shown in Fig. 6, serves as the transport layer for network transactions between the various blocks in the system. Each transaction can be one of three types: streaming data, interrupts, or Advanced eXtensible Interface (AXI) [15] transactions. Streaming data transactions are used to send input activations, partial sums, and output activations between the PEs and GB. Interrupts are single-flit packets generated by the PEs and GB and sent to the RVP to signal the completion of a layer. AXI transactions are used for all other reads and writes of registers and memory in the system. A protocol similar to 64-bit AXI4-Lite supports bursts up to eight words in length to allow the RVP to fetch entire cache lines with a single transaction. All architecturally visible state across the multi-chip system is globally addressable, including control registers, PE and GB buffers, and the RVP scratchpad. Only the RVP implements AXI masters that can

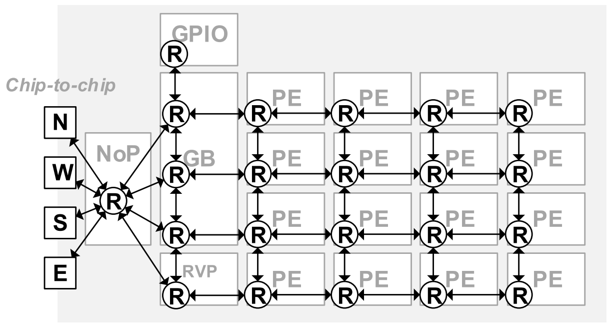


Fig. 7. Units are connected in a mesh NoC.

initiate requests, while the PEs and GBs implement AXI slaves to service requests to the local state. To simplify the AXI slave logic in the highly replicated PE, no write responses are generated anywhere in the system.

Custom hardware extensions to the AXI master block enable the RVP to exploit features of the system that reduce communication latency during runtime. A portion of the global address space is reserved for multicast requests: by writing to a particular global address, the RVP's AXI write is converted into a multicast packet that writes the same data to the same local address of a configurable subset of the PEs on the chip or the RVPs in the MCM system. The RVP's hardware interrupt lines are also memory-mapped so that a write to a particular address can trigger an interrupt.

The NoC routers are connected in a mesh network, as shown in Fig. 7. Each NoC transaction is encoded in a packet that is composed of one or more 66-bit flits. A flit is composed of 64 bits of data and 2 bits of flit identification, indicating a header, body, or tail flit. Singleton flits are indicated by setting both the header and tail bits. The header flit's 64 bits of data are for packet routing and other metadata; subsequent flits contain the packet payload. The NoC supports both unicast and multicast traffic, and routes are specified in the header flit with a bit indicating whether the packet is unicast or multicast. Multicast is one-hot encoded and can address all 36 chips in the system and 20 NoC destinations; this information consumes 56 header bits. Unicast packet destinations are binary encoded, consuming 6 bits for network-on-package (NoP) destinations and five for NoC destinations. Therefore unicast headers have available header bits to encode additional packet-specific information; only certain types of packets can be multicast due to the large bitwidth of the one-hot multicast address.

The NoC router implementation uses cut-through routing with credit-based flow control and is pipelined to operate at full throughput and two cycles of latency. At 0.72 V, each link of the NoC achieves 70-Gb/s bandwidth.

D. PE

Most computation is done by the PE in Fig. 8, which executes convolutional layers, fully-connected layers, and post-processing functions like bias addition, rectified linear unit (ReLU), and pooling. A PE has eight lanes, each using a different weight tensor to generate elements for a separate output channel (K). Within each lane, an 8-bit precision vector MAC multiplies eight input elements from separate input

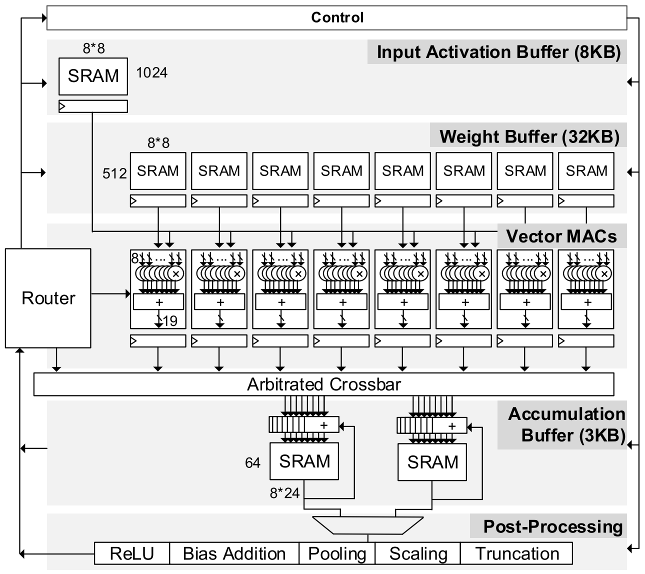


Fig. 8. Architecture of the PE.

channels (C) with eight weight elements and sums them to calculate a single output value. As the numbers of input and output channels (C and K) generally range from 64 to 1024, performing vector operations in sets of eight elements is very efficient. Local input activation, output activation, and accumulation SRAMs buffer data for the datapath. Minimizing the accesses to these SRAMs is critical to maximizing energy efficiency. The input activation SRAM is read every cycle, but the energy cost is amortized by distributing each element to eight lanes. The weight SRAM is much wider than the input activation SRAM as it needs to supply a separate vector of values to each lane in the datapath, but the weights remain constant for multiple inputs, so the values are reused $P \times Q$ times. The accumulation SRAM is written every cycle to hold partial sums, but energy is amortized by writing the accumulation of the eight-wide vector of C channels. The output size $P \times Q$ is generally larger than the number of entries in the accumulation buffer, so the computation is temporally tiled to generate a subset of output activation dimensions at a time. The accumulation buffer can also be written through the router from other PEs to perform cross-PE reduction when the weight kernel is split between multiple PEs. Once the full accumulation is complete, each PE performs the final post-processing functions, such as ReLU, bias addition, pooling, scaling, or truncation, to compute the final output activation. The accumulation buffer is split into two banks to allow simultaneous access for the local MACs, the router, and the post-processing unit; the arbitration crossbar resolves bank conflicts.

PE power was simulated on the post-synthesis gate-level netlist using activity traces from a representative workload, and a breakdown of PE energy is shown in Fig. 9 at 0.72 V. The measured PE energy correlates to the simulated energy within 10% across a range of benchmarks. The largest consumer of energy is the accumulation buffer SRAM, due to the writing of 192 bits every cycle. The second-largest consumer is the datapath, which performs the MAC operations. Since the input buffer output is shared between multiple lanes, and the

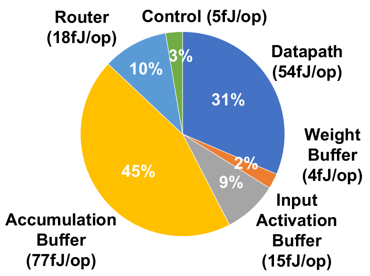


Fig. 9. Simulated energy breakdown of the PE at 0.72 V.

weight buffer output is used for many cycles, the contribution of these two SRAMs is small. PE energy efficiency could be optimized by using a generator to explore the design space of many different possible PE dataflows and precisions for different DNNs [16].

E. GB

As input activations are generally multicast to multiple PEs, and output activations collected from multiple PEs, the GB acts as a second level in the memory hierarchy to store these activations on a chip. The GB SRAM is partitioned into four 16-kB banks, which can be flexibly partitioned between input and output activations. The GB includes three routers that provide higher bandwidth into the NoC. In addition to managing activations, the GB can perform some forms of computation (such as element-wise computation) locally, without needing to send data to PEs.

F. RVP

The RISC-V core is an RV64IMAC implementation of the open-source rocket chip generator [17] based on the SmallCore instance. The RVP includes a 16-KiB instruction cache, an 8-KiB data cache, and a 16-KiB scratchpad. The 128 external interrupt lines are accessible via the NoC, which can trigger them either via interrupt transactions or AXI writes.

G. GPIO and JTAG

The chip uses a narrow, low-speed GPIO interface to communicate with a host for the purpose of loading weights, input activations, and RISC-V runtime software. The GPIO interface uses a divided on-chip clock and a ready-valid protocol to communicate with an FPGA, which recovers the clock and performs the skew alignment. JTAG is used to configure GPIO, clocking, and routing tables, toggle reset, and provide observability of key signals for debugging.

III. Multi-Chip Architecture

A full MCM-based neural network accelerator is formed by connecting 36 chips together on the package in a mesh network, as shown in Fig. 2. There are two general strategies to utilize the increased computation capacity to improve throughput: increase parallelism or pipeline multiple layers in the system. When increasing parallelism, the system executes a single layer at a time as it does in the one-chip architecture, except computation is split among chips in the same manner

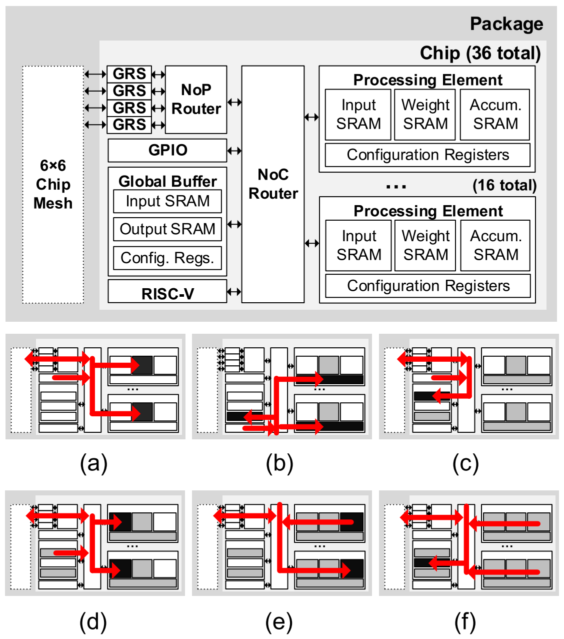


Fig. 10. Flow of data across the NoP router to execute a layer. (a) Weights. (b) Configuration. (c) Input activation. (d) Compute. (e) Accumulation. (f) Output activation.

as computation was split among PEs in the one-chip case. The latency of the layer computation is reduced (strong scaling), so more layers can be executed per unit time, improving throughput. An alternative strategy pipelines multiple layers, with groups of chips executing different layers simultaneously and forwarding their results to the next group. Pipelining improves throughput but does not decrease the latency of layer computation (weak scaling). While pipelining can improve the overall utilization when there is limited parallelism in a layer [18], this article focuses on the increased layer parallelism strategy to understand the limits of strong scaling.

The prototype system is optimal for networks that fit entirely in the on-chip SRAM storage. Future work can investigate supplementing the mesh with IO chips that interface between ground-referenced signaling (GRS) and a dynamic random-access memory (DRAM) controller to support larger DNNs.

A. System Operation

The system diagram, as shown in Fig. 10, describes the data movement required to execute a convolution on multiple chips. Weight tensors are loaded once at startup through GPIO from the host subsystem and distributed across the NoP and NoC to SRAMs inside each of the 576 PEs [Fig. 10(a)]. Each weight tensor can be split among the chips when each chip computes a separate output channel or replicated on different chips so that each chip can work on a different portion of the input activation in parallel. The RVPs on each chip configure state machines within each PE and the GB to match the layer dimensions [Fig. 10(b)]. Input activation tensors are loaded through GPIO from a host subsystem and are stored in large SRAMs in the GB [Fig. 10(c)].

To initiate layer execution, a lead RVP multicasts a go command to the other worker RVPs, which then multicast local start commands to every unit on the chip. The GB sends

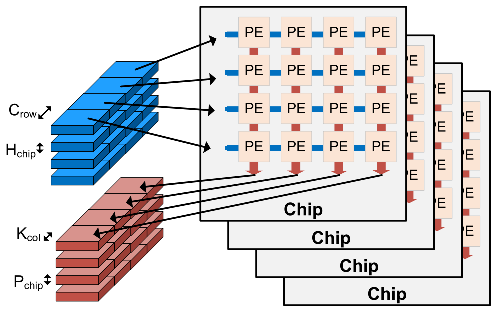


Fig. 11. Mapping a convolution to an example four-chip system.

the input activation tensor to the local PEs across the NoC, as well as remote PEs across the NoP [Fig. 10(d)]. Each PE performs 64 MACs per cycle for a total of 36 864 MACs per cycle in the package. Accumulation of intermediate sums to compute the final output element occurs either within the PE, across multiple PEs in one chip, or across multiple PEs in one package [Fig. 10(e)]. Then each PE sends its respective portion of the output activation tensor to the local or remote GB, and the layer's computation is finished [Fig. 10(f)]. The RVPs across the system synchronize after the completion of each layer using an interrupt-based barrier system controlled by the lead RVP. The worker RVPs wait for interrupts from all local units that indicate the completion of work. Once received, the worker RVPs send an interrupt to the lead RVP. Meanwhile, the lead RVP first waits for all local interrupts, then waits for interrupts from the other RVP participating in the computation. The lead RVP uses built-in counters to time the execution of each layer.

B. Mapping Convolutions

The mapping strategy across multiple chips in a package is almost identical to mapping across multiple PEs in a chip, except that traffic flows across the NoP in addition to the NoC, and there are now multiple GBs in the system. Mapping remains flexible, and computation can be split along any dimension between chips (K, P, Q, R, S, H, W, and C). Fig. 11 shows an example of mapping a layer onto a four-chip system. Each chip computes a subset of the output rows (P), so each chip stores the corresponding input activation rows (H) in their local GB, and the weights are replicated across all four chips. Within each chip, input channels (C) are split between rows and output channels (K) between columns, as described in Fig. 5. Each PE executes 8 K and 8 C per cycle, each chip operates on four segments of K and four segments of C in parallel, and each package operates on four segments of P in parallel for 4096 MACs per cycle. The other layer dimensions are looped over temporally to complete the convolution. Detailed descriptions and studies of various mapping strategies and measurements of resulting latencies can be found in [18].

C. NoP

Simply connecting the edges of the single-chip NoC mesh to adjacent chips in one large multi-chip mesh would inhibit scalability. Instead, a hierarchical network connects 36 chips in a 6-by-6 chip mesh NoP, and each chip's NoP router connects to four NoC local ports, as shown in Fig. 7. The NoP routers use the same design and packet format as the NoC routers. NoP routing tables are configurable via JTAG-configurable lookup tables. Multicast-multicast deadlocks are avoided by enforcing cut-through flow control, in which a packet can only advance to the next router when there is enough buffering for the entire packet. Multicast-unicast routing deadlocks are avoided by using the base-routing-conformed-path model [19], in which multicast and unicast routing tables are programmed such that they share the same network paths. This approach ensures that if the unicast routing algorithm is deadlock-free, all possible interaction between unicast and multicast is also deadlock-free.

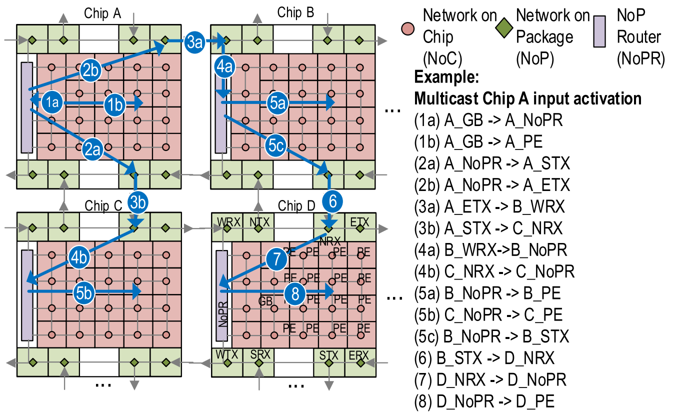


Fig. 12. Example multicast operation between four chips.

Fig. 12 shows an example of input activation multicast operation in a four-chip system. The GB on chip A sends data into local PEs through the NoC and other chips through the NoP. The data moves to chips B and C through the chip-to-chip interconnect, and arrives at the local GB, where it is forwarded to chip D and deposited into the local NoCs to send to local PEs. Like the NoC, the bandwidth requirement of the NoP is reduced because the weights remain stationary within the PEs, and only input activations, partial sums, and output activations need to be sent between chips.

D. Chip-to-Chip GRS

The scalability of the MCM-based accelerator relies on efficient chip-to-chip communication within the NoP. For achieving high bandwidth and energy efficiency, each chip in the package is connected with single-ended GRS [20]. For implementing a package mesh, every chip has eight chip-to-chip GRS transceivers, where four are configured as transmitters (TX) and four as receivers (RX) and communicate to adjacent chips in a mesh. Each TX and RX pair has four data wires, and one forwarded clock wire, as shown in Fig. 13. Each signal is ground-referenced instead of differential, only requiring one bump, and the transmitters drive a low-swing signal of about 200 mV around the ground to improve energy efficiency. Each link has configurable equalization and termination at both the receiver and transmitter. An alternative MCM technology to GRS across organic substrates is the silicon interposer [13], which allows much finer pitch bumps, but the wires support

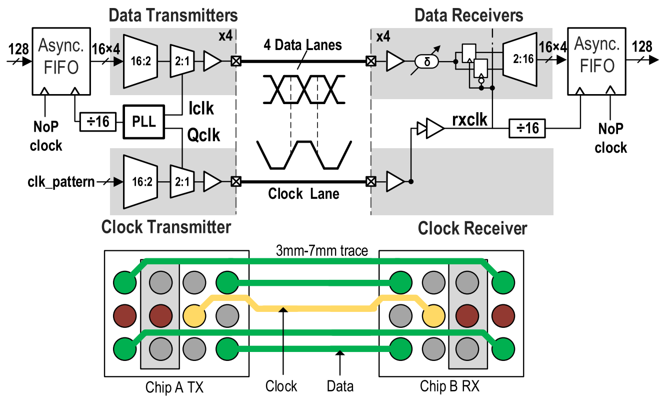


Fig. 13. Chips communicate on the package using GRS.

much lower data rates, and the expense is impractical for many markets. Unlike silicon interposer transceivers, GRS can communicate with other packages through the printed circuit board (PCB) at the same speed with the same circuits, so the prototype system size is not limited to a single package and could scale further through either denser packing of chips on the package or multiple packages on a PCB.

Since each GRS link is unidirectional, credit-based flow control is used where credits are returned using the GRS link running in the opposite direction. Data sent from the NoP router to the GRS TX is written to a 15-word, 128-bit FIFO memory using the NoP router clock. This same memory is read as a 32-word, 60-bit FIFO by a 1.56-GHz GRS word clock, such that the same FIFO memory is used for both clock domain crossing and splitting the data into 60-bit words. Four bits of header are added per word to signify when data is valid or partially valid and to pass credits. The 64-bit words are then sent to high speed 16:1 serializers to drive the four data wires. This process is reversed in the GRS RX to reconstitute data for the receiving NoP router.

Buffering of full packets is used along with the GRS NoP and GPIO interfaces; in the former case, to ensure that no stalling occurs mid-packet when sending across the GRS interface, and in the latter case to ensure that the much slower GPIO interface does not tie up routing resources while flits are being transmitted or received off-chip. The total packet length is limited by these interfaces to 17 flits (1 header and 16 payloads).

IV. MCM SoC Implementation

The 6 mm^2 inference accelerator [21] was fabricated in a TSMC 16-nm FinFET process, and 36 chips were assembled on a 12-layer organic substrate.

A. HLS-Based Agile Design Methodology

The test chip was designed with a high-productivity very large scale integration (VLSI) design methodology [22], which enabled 24-h turnaround from design changes to a tape-out-ready GDS. Most of the design was described in C++ using an open-source library of commonly used micro-architectural components called MatchLib [23] and synthesized into Verilog using an industry-standard high-level synthesis (HLS) tool.

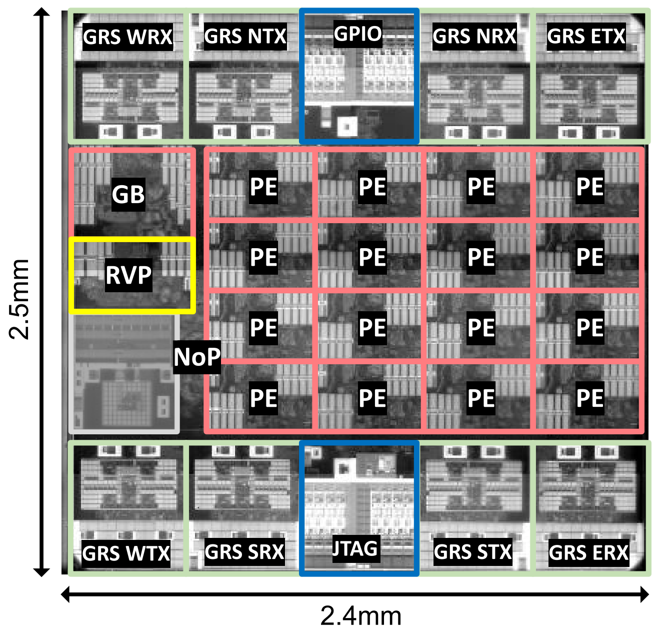


Fig. 14. Chip micrograph annotated with the floorplan of the chip.

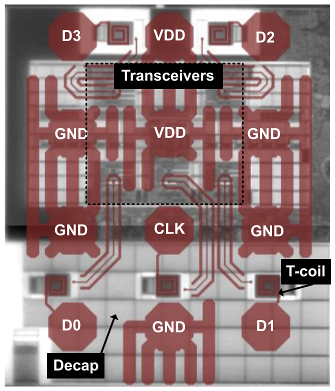


Fig. 15. Floorplan of the chip-to-chip GRS interconnect partition.

The design was intentionally modularized into partitions of around 200 000 gates that avoid tight communication or timing constraints to other units by using latency-insensitive (LI) channels [24], [25]. The main partitions in the design are the PE, GB, RVP, NoP, and GRS, shown as rectangles in the floorplan in Fig. 14; each was implemented independently and in parallel with the others to improve turnaround time. Partitioning the design into smaller units increases the number of cross-unit boundaries, while larger units increase place-and-route runtime. An agile hardware implementation flow using fully automated synthesis and place-and-route tools provided daily feedback about timing, area, and power consumption as the design was optimized. Engineering change orders (ECOs) were avoided entirely by making changes directly in the source code and reimplementing the entire unit. In parallel with VLSI trials, the entire design was prototyped on an FPGA so that software development occurred in parallel with hardware development, which revealed many critical bugs well before tape-out. Overlapping architectural design, VLSI implementation, and software design ensured that effort was focused on improving the final quality-of-result.

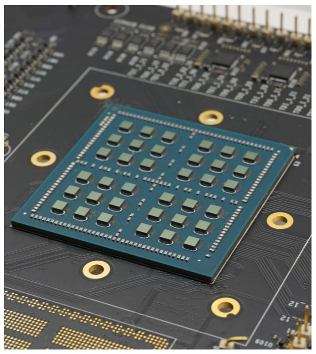


Fig. 16. Prototype system connects 36 chips on the package.

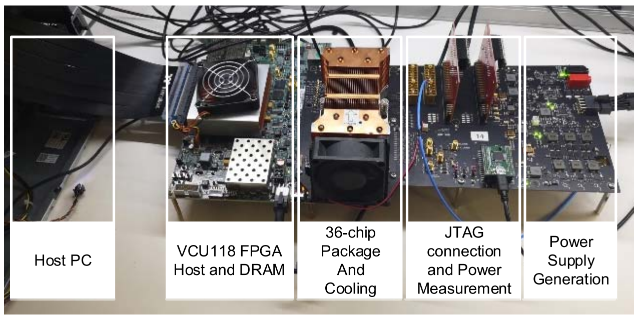
B. Floorplan

The physical floorplan in Fig. 14 largely reflects the logical NoC mesh network shown in Fig. 7. Physical partition reuse was critical to reducing the design effort. Each of the 16 PE partitions is identical and is designed so that the IO pins connect by abutment. The off-chip communication partitions (GRS, JTAG, and GPIO) are placed on the edge of the chip to avoid disturbing power delivery to the PEs in the center of the chip. The eight GRS partitions are identical and designed so that they can be mirrored across the X and Y dimensions while still abutting correctly to the power grid. The GRS partition floorplan, shown in Fig. 15, contains the custom layout transceivers in the center. The connections to the 140-μm pitch bumps are made with length-matched and shielded wires on the redistribution layer. The t-coils, electrostatic discharge (ESD) devices, decoupling capacitance, and link calibration circuits are implemented with digital place-and-route. The JTAG and GPIO partitions use standard 1.8-V IO devices to communicate off-chip. The NoP contains the most difficult timing paths because it synchronously communicates with every GRS macro, so it requires careful pipelining and clock distribution.

C. Clocking

Each partition in the design is clocked by an adaptive clock generator in the center of the partition and is asynchronous to other partitions. The latency cost of synchronization between clock domains is mitigated with the use of pausible bisynchronous FIFOs [26], [27]. Each partition can run at independent frequencies, so physically large partitions, such as the NoP, can run at a slower frequency than the PEs. The FIFOs between each partition are five entries deep to support the latency of conventional synchronizers in a backup mode.

A JTAG interface is used to configure the chip during the boot process. Every partition has a separate JTAG tap controller to avoid synchronous paths between partitions, and the entire chip consists of a chain of 31 controllers with the JTAG signals serially snaked through the chip. The reset is toggled through the JTAG interface and is synchronized into each local clock domain. A high-frequency reference clock for GRS (1.56 GHz), a low-frequency testing clock for measurement circuits in each partition (100 MHz), and a global on-chip clock are distributed as a tree from the JTAG partition.


Host PC
VCU118 FPGA Host and DRAM
36-chip Package And Cooling
JTAG connection and Power Measurement
Power Supply Generation
Fig. 17. Bench measurement setup.

V. Experimental Results

A. Test Setup

Fig. 16 shows the prototype package with 36 chips and Fig. 17 shows the bench measurement setup. The test package is mounted on a custom PCB with voltage regulators, clock generators, and power measurement circuitry. The test board connects via FPGA Mezzanine Card (FMC) to a Xilinx VCU118 FPGA board, which is connected to a host PC via peripheral component interconnect express (PCIE). The FPGA communicates with the prototype via the GPIO interface of one of the chips. The FPGA fabric implements an AXI interconnect that shares the global memory map of the prototype system, allowing the RVPs to access the FPGA state that includes 4 GB of DRAM. To execute an inference operation, a RISC-V program, which includes all weights, input activations, and configuration settings, is loaded into FPGA DRAM. The RVPs then execute the program, fetching from DRAM, and loading state into the PEs and GBs before initiating layer execution.

B. NoP Performance

Each data lane operates at a configurable speed between 11 and 25 Gb/s and consumes 0.82–1.75 pJ/bit. The power breakdown is similar to a prior implementation [20], except that there are four forwarded data lanes per clock lane instead of eight. Power is constant regardless of traffic because the links in this prototype have no optimizations, such as sleep mode, to reduce power during periods of inactivity. Compared to previous interconnect on organic substrates for MCM systems [14], GRS has about 3.5× higher bandwidth per chip area and lower energy per bit. Measurements show an eye opening of 0.7 UI at 25 Gb/s.

C. Peak Performance

Table I compares our system to prior inference accelerators with 8-bit precision running a peak performance benchmark that saturates MACs on each chip. The voltage range differs between single-chip and multi-chip systems because GRS has a minimum frequency requirement that limits the minimum operating voltage. The digital core efficiency numbers exclude chip-to-chip interconnect power for comparison purposes. At peak performance mode, this fixed power overhead is less than 5% of overall power, but it becomes more significant at minimum voltage. Overall the prototype achieves between

TABLE I
COMPARISON TO OTHER INFERENCE ACCELERATORS FOR PEAK PERFORMANCE BENCHMARK

	B. Moons, ENVISION [7]	Z. Yuan, STICKER [8]	J.Lee, UNPU [9]	J. Song, Exynos [6]	This work*		
					1 Chip	4 Chip (2 × 2)	36 Chip (6 × 6)
Technology	28nm	65nm	65nm	8nm	16nm		
Cumulative Core Area	1.87 mm^2	7.8 mm^2	13 mm^2	5.5 mm^2	3.1 mm^2	12.4 mm^2	111.6 mm^2
Cumulative Chip Area	unknown	12 mm^2	16 mm^2	unknown	6 mm^2	24 mm^2	216 mm^2
Precision	4b,8b,16b	8b	1-16b	8b,16b	8b		
On-Chip SRAM (MB)	0.14	0.17	0.25	1.53	0.625	2.5	22.5
Supply Voltage (V)	1	0.67-1.1	0.63-1.1	0.5-0.8	0.41-1.2	0.52-1.2	0.52-1.1
Frequency (MHz)	200	200	5-200	67-933	161-2001	515-1998	484-1797
Core Power (mW)	165	21-248	3.2-297	39-1,553	30-4160	630-16,420	5,310-106,090
GRS Power† (mW)	n/a	n/a	n/a	n/a	n/a	215-220	3,840-4,090
MACs per cycle	512 @8b	256	1,728@8b	1,024	1,024	4,096	36,864
Performance (TOPS)	~0.15@8b	0.1	0.69@8b	1.91	0.32-4.01	3.93-15.7	32.5-127.8
Core Energy Efficiency (pJ/op)	~1.1@8b	0.96	~0.18@8b	0.087@8b	0.105-1.04	0.160-1.05	0.164-0.83
Core Area Efficiency (TOPS/mm^2)	0.08	0.013	0.053	0.35	0.10-1.29	0.32-1.27	0.29-1.15

* Measured results reported for 40% density weights and input activations †11Gbps mode

TABLE II
MEASUREMENT OF A 36-CHIP SYSTEM RUNNING RESNET-50 AT 0.80 V

Layer	Latency (µS)	Core Energy (µJ)	GRS Energy (µJ)
conv1-pool1	41.00	902.90	147.70
res2a_branch1	8.87	209.00	32.02
res2a_branch2a	6.44	141.21	23.26
res2[a-c]_branch2b	9.26	250.84	33.40
res2[a-c]_branch2c	8.87	209.00	32.02
res2[b-c]_branch2a	14.04	417.68	50.56
res3a_branch1	8.92	281.39	32.15
res3a_branch2a	7.59	199.90	27.41
res3[a-d]_branch2b	9.11	237.57	32.91
res3[a-d]_branch2c	8.18	220.74	29.52
res3[b-d]_branch2a	8.40	232.08	30.29
res4a_branch1	8.11	264.19	29.21
res4a_branch2a	6.06	154.99	21.87
res4[a-f]_branch2b	11.98	302.36	43.35
res4[a-f]_branch2c	6.64	187.68	23.94
res4[b-f]_branch2a	6.86	194.77	24.77
res5a_branch1	12.49	326.72	45.18
res5a_branch2a	21.09	464.69	76.28
res5[a-c]_branch2b	13.33	349.58	48.20
res5[a-c]_branch2c	7.38	181.21	26.74
res5[b-c]_branch2a	8.23	203.74	29.78
fc1000	3.32	27.37	3.29
Total (batch=1)	0.525 ms	16.3 mJ/image	2.33 mJ/image

67–1280× higher TOPS, 0.8–10× higher energy efficiency, and 4–100× higher area efficiency.

D. Application Measurements

Table II demonstrates the architecture's scalability with the measured performance of a 32-chip datacenter-scale system running each layer of ResNet-50 [3]. GRS chip-to-chip interconnect enables multi-chip scalability while consuming 12% of the total energy, and a batch size of one minimizes inference latency and energy. Further analysis of the tradeoff between chip count and latency, strategies to efficiently map workloads onto the hardware, and the impact of differing NoP latencies on performance is available in [18].

E. Strong Scaling

Fig. 18 shows that measured latency in terms of PE cycles is reduced by 16× when executing the res4a_branch1 layer of ResNet-50 [3] on 32 chips instead of 1 chip. The measured latency is higher than the ideal latency because it includes the NoC and NoP communication overhead of operations, such as distributing the input activation across the package. One chip maintains 63% utilization of the MAC units. With 32 chips, the computation is spread across so many PEs that the number of cycles spent doing computation is only 4096 cycles, and 6000 cycles of synchronization between chips across the NoP starts to dominate runtime. Overall, a 32-chip system can execute 128-million MACs in 11 μs, and design improvements to the synchronization scheme could further improve strong scaling.

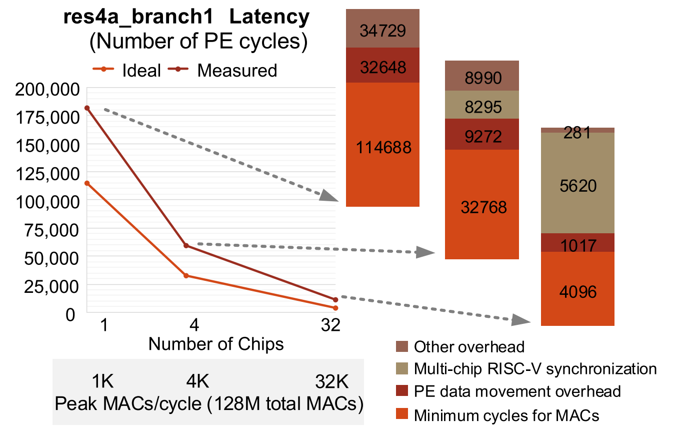


Fig. 18. Increasing the number of chips decreases the latency of layer execution.

VI. CONCLUSION

This article presents a scalable DNN inference accelerator that uses MCM assembly of multiple chips on an organic substrate to improve yield, reduce design cost, and address different market segments with a single chip. Scalability is enabled by a flexible multi-chip architecture and hierarchical NoC and NoP. The 36-chip system achieves high energy efficiency (9.5 TOPS/W), high area efficiency (1.29 TOPS/mm^2), and high performance (128 TOPS).

ACKNOWLEDGMENT

The authors would like to thank D. Templeton, G. Peled, J. Dobbins, B. Boudaoud, R. Laperriere, B. Moghadam, S. Sudhakaran, Z. Bokharey, S. Rajapandian, J. Chen, J. Hu, and V. Iyer for package, PCB, signal integrity, fabrication, and prototyping support.

REFERENCES

[1] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna, "Rethinking the inception architecture for computer vision," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 2818–2826, doi: 10.1109/CVPR.2016.308.

[2] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," in *Proc. 25th Int. Conf. Neural Inf. Process. Syst.*, vol. 1, 2012, pp. 1097–1105.

[3] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 770–778.

[4] L.-C. Chen, Y. Zhu, G. Papandreou, F. Schroff, and H. Adam, "Encoder-decoder with atrous separable convolution for semantic image segmentation," in *Proc. 15th Eur. Conf.*, Munich, Germany, Sep. 2018, pp. 833–851.

[5] M. Bojarski *et al.*, "End to end learning for self-driving cars," 2016, *arXiv:1604.07316*. [Online]. Available: https://arxiv.org/abs/1604.07316

[6] J. Song *et al.*, "An 11.5TOPS/W 1024-MAC butterfly structure dual-core sparsity-aware neural processing unit in 8 nm flagship mobile SoC," in *IEEE Int. Solid-State Circuits Conf. (ISSCC) Dig. Tech. Papers*, Feb. 2019, pp. 130–132.

[7] B. Moons, R. Uytterhoeven, W. Dehaene, and M. Verhelst, "Envision: A 0.26-to-10TOPS/W subword-parallel dynamic-voltage-accuracy-frequency-scalable convolutional neural network processor in 28 nm FDSOI," in *IEEE Int. Solid-State Circuits Conf. (ISSCC) Dig. Tech. Papers*, Feb. 2017, pp. 246–247.

[8] Z. Yuan *et al.*, "Sticker: A 0.41-62.1 TOPS/W 8Bit neural network processor with multi-sparsity compatible convolution arrays and online tuning acceleration for fully connected layers," in *Proc. IEEE Symp. VLSI Circuits*, Jun. 2018, pp. 33–34.

[9] J. Lee, C. Kim, S. Kang, D. Shin, S. Kim, and H.-J. Yoo, "UNPU: A 50.6TOPS/W unified deep neural network accelerator with 1b-to-16b fully-variable weight bit-precision," in *IEEE Int. Solid-State Circuits Conf. (ISSCC) Dig. Tech. Papers*, Feb. 2018, pp. 218–220.

[10] S. Han *et al.*, "EIE: Efficient inference engine on compressed deep neural network," in *Proc. 43rd Int. Symp. Comput. Archit.*, 2016, pp. 243–254.

[11] Y. 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. 2017, doi: 10.1109/JSSC.2016.2616357.

[12] N. P. Jouppi *et al.*, "In-Datacenter Performance Analysis of a Tensor Processing Unit," in *Proc. 44th Annu. Int. Symp. Comput. Archit.*, 2017, pp. 1–12.

[13] K. Saban. (2012). *Xilinx Stacked Silicon Interconnect Technology Delivers Breakthrough FPGA Capacity, Bandwidth, and Power Efficiency*. [Online]. Available: https://www.xilinx.com/support/documentation/white_papers/wp380_Stacked_Silicon_Interconnect_Technology.pdf

[14] N. Beck, S. White, M. Paraschou, and S. Naffziger, "'Zeppelin': An SoC for multichip architectures," in *IEEE Int. Solid-State Circuits Conf. (ISSCC) Dig. Tech. Papers*, Feb. 2018, pp. 40–42.

[15] (2017). *AMBA AXI and ACE Protocol Specification AXI3, AXI4, AXI5, ACE and ACE5*. [Online]. Available: https://arm.com

[16] R. Venkatesan *et al.*, "MAGNet: A modular accelerator generator for neural networks," in *Proc. Int. Conf. Comput. Aided Design (ICCAD)*, 2019.

[17] K. Asanovic *et al.*, "The rocket chip generator," Dept. EECS, Univ. California, Berkeley, CA, USA, Tech. Rep. UCB/EECS-2016-17, 2016.

[18] Y. S. Shao *et al.*, "Simba: Scaling deep-learning inference with multi-chip-module-based architecture," in *Proc. 52nd Annu. IEEE/ACM Int. Symp. Microarchitecture (MICRO)*, Oct. 2019, pp. 14–27.

[19] D. K. Panda, S. Singal, and R. Kesavan, "Multidestination message passing in wormhole k-ary n-cube networks with base routing conformed paths," *IEEE Trans. Parallel Distrib. Syst.*, vol. 10, no. 1, pp. 76–96, Jan. 1999.

[20] J. W. Poulton *et al.*, "A 1.17-pJ/b, 25-Gb/s/pin ground-referenced single-ended serial link for off- and on-package communication using a process- and temperature-adaptive voltage regulator," *IEEE J. Solid-State Circuits*, vol. 54, no. 1, pp. 43–54, Jan. 2019.

[21] B. Zimmer *et al.*, "A 0.11 pJ/Op, 0.32-128 TOPS, scalable multi-chip-module-based deep neural network accelerator with ground-reference signaling in 16 nm," in *Proc. Symp. VLSI Circuits*, 2019, pp. C300–C301.

[22] B. Khailany *et al.*, "A modular digital VLSI flow for high-productivity SoC design," in *Proc. 55th Annu. Design Autom. Conf. (DAC)*, 2018, pp. 1–6.

[23] *MATCHLIB*. Accessed: Aug. 14, 2019. [Online]. Available: https://github.com/NVlabs/matchlib

[24] K. E. Fleming, M. Adler, M. Pellauer, A. Parashar, A. Mithal, and J. Emer, "Leveraging latency-insensitivity to ease multiple FPGA design," in *Proc. ACM/SIGDA Int. Symp. Field Program. Gate Arrays (FPGA)*, 2012, pp. 175–184.

[25] L. P. Carloni, K. L. McMillan, A. Saldanha, and A. L. Sangiovanni-Vincentelli, "A methodology for correct-by-construction latency insensitive design," in *IEEE/ACM Int. Conf. Comput.-Aided Design, Dig. Tech. Papers*, Nov. 1999, pp. 309–315.

[26] B. Keller, M. Fojtik, and B. Khailany, "A pausible bisynchronous FIFO for GALS systems," in *Proc. 21st IEEE Int. Symp. Asynchronous Circuits Syst.*, May 2015, pp. 1–8.

[27] M. Fojtik *et al.*, "A fine-grained GALS SoC with Pausible adaptive clocking in 16 nm FinFET," in *Proc. 25th IEEE Int. Symp. Asynchronous Circuits Syst. (ASYNC)*, May 2019, pp. 27–35.

Brian Zimmer (Member, IEEE) received the B.S. degree in electrical engineering from the University of California at Davis, Davis, CA, USA, in 2010, and the M.S. and Ph.D. degrees in electrical engineering and computer sciences from the University of California at Berkeley, Berkeley, CA, in 2012 and 2015, respectively.

He is currently a Senior Research Scientist with the Circuits Research Group, NVIDIA, Inc., Santa Clara, CA. His research interests include soft error resilience, energy-efficient digital design, low-voltage static random-access memory (SRAM) design, machine learning accelerators, productive design methodologies, and variation tolerance.

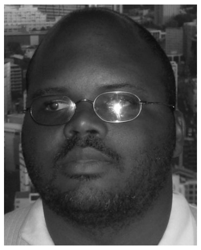
Rangharajan Venkatesan (Member, IEEE) received the B.Tech. degree in electronics and communication engineering from the IIT Roorkee, Roorkee, India, in 2009, and the Ph.D. degree in electrical and computer engineering from Purdue University, West Lafayette, IN, USA, in 2014.

He is currently a Senior Research Scientist with NVIDIA, Santa Clara, CA, USA. His research interests include machine learning accelerators, high-level synthesis, variation-tolerant design methodologies, spintronics, and approximate computing.

Dr. Venkatesan has been a member of the technical program committees of several leading IEEE conferences, including the International Solid-State Circuits Conference (ISSCC), the Design Automation Conference (DAC), and the International Symposium on Low Power Electronics and Design (ISLPED). He received the Best Paper Award for the paper on scalable deep learning accelerator design at the International Symposium on Microarchitecture (MICRO) in 2019. His work on spintronic memory design was recognized with the Best Paper Award at the ISLPED in 2012 and the Best Paper Nomination at the Design, Automation and Test in Europe (DATE) in 2017. His work on FinFET-based static random-access memory (SRAM) also received the Best Paper Nomination at the DATE in 2015.

Yakun Sophia Shao (Member, IEEE) received the B.S. degree in electrical engineering from Zhejiang University, Hangzhou, China, and the M.S. and Ph.D. degrees in computer science from Harvard University, Cambridge, MA, USA, in 2014 and 2016, respectively.

She was a Senior Research Scientist with NVIDIA, Inc., Santa Clara, CA, USA. She is currently an Assistant Professor with the Electrical Engineering and Computer Sciences Department, University of California at Berkeley, Berkeley, CA. Her research interest is in the area of computer architecture, with a special focus on specialized accelerators, heterogeneous architecture, and agile very large scale integration (VLSI) design methodology.



Jason Clemons (Member, IEEE) was born in Michigan, USA, in 1978. He received the B.S. degree in electrical engineering from Michigan Technological University, Houghton, MI, USA, in 2000, and the M.S. and Ph.D. degrees in computer science and engineering from the University of Michigan, Ann Arbor, MI, USA, in 2002 and 2013, respectively.

In 2000, he joined Whirlpool Appliances, St. Joseph, MI, USA, as an Embedded Systems Engineer, where he was promoted to Senior Engineer while transitioning to the role of Motor Control System Engineer. Following the completion of his Ph.D. degree, he joined NVIDIA's Architecture Research Group, Austin, TX, USA, where he is currently a Senior Research Scientist. His current research focuses on domain-specific computing, in particular the intersection of machine learning, computer vision, and computer architecture.

Dr. Clemons is a member of ACM, the National Society of Black Engineers, and the Society of Women Engineers. He was a Co-Guest Editor of the IEEE MICRO Automotive Computing edition in January 2018. He has served on the International Symposium on Performance Analysis of Systems and Software (ISPASS), International Symposium on High-Performance Computer Architecture (HPCA), and the Design Automation Conference Program Committees (DAC PCs). He has also served on the external review committees for the International Symposium on Microarchitecture (MICRO), ISCA, HPCA, DAC, and International Conference on Artificial Intelligence Circuits and Systems (AICAS).

Matthew Fojtik (Member, IEEE) received the B.S., M.S., and Ph.D. degrees in electrical engineering from the University of Michigan, Ann Arbor, MI, USA, in 2008, 2010, and 2013, respectively.

In 2013, he joined NVIDIA, Durham, NC, USA, as a member of the Circuits Research Group and is currently a member of NVIDIA's ASIC/VLSI Research Group. His research interests include timing margin reduction techniques, clocking and synchronization, power supply noise tolerance, low-power on-chip communication, and efficient very large scale integration (VLSI) methodologies.

Nan Jiang (Member, IEEE) received the B.S. degree in engineering from the Harvey Mudd College, Claremont, CA, USA, in 2007, and the M.S. and Ph.D. degrees in electrical engineering from Stanford University, Stanford, CA, in 2010 and 2013, respectively.

In 2012, he joined NVIDIA, Inc., St. Louis, MO, USA, as a Founding Member of the Network Research Group, where he works on designing supercomputing network systems and NVIDIA's NVSwitch technology. His areas of research include switch architecture, routing algorithms, congestion control protocols, and accelerator-centric network fabrics.

Ben Keller (Member, IEEE) received the B.S. degree in engineering from the Harvey Mudd College, Claremont, CA, USA, in 2010, and the M.S. and Ph.D. degrees in electrical engineering and computer sciences from the University of California at Berkeley, Berkeley, CA, in 2015 and 2017, respectively.

He held research internships at the National Institute of Standards and Technology in 2010 and NVIDIA Corporation, Santa Clara, CA, in 2014. In 2017, he joined the ASIC & VLSI Research Group, NVIDIA Corporation, where he works as a Senior Research Scientist. His research interests include digital clocking and synchronization techniques, fine-grained adaptive voltage scaling, and hardware design productivity.

Alicia Klinefelter (Member, IEEE) received the B.S. degree in electrical and computer engineering from Miami University, Oxford, OH, USA, in 2010, and the Ph.D. degree in electrical engineering from the University of Virginia, Charlottesville, VA, USA, in 2015.

From 2015 to 2017, she was with Intel's Digital Communication Lab, Hillsboro, OR, USA. Since 2017, she has been with NVIDIA, Inc., Durham, NC, USA, where she is currently a Senior Research Scientist. She has authored or coauthored publications in the areas of ultra-low-power circuits, SoC design, and high-level synthesis methodologies.

Dr. Klinefelter is currently a member of SSCS. She has been a member of the Technical Program Committee of the International Solid-State Circuits Conference (ISSCC) since 2018. She is also a Reviewer of the IEEE JOURNAL OF SOLID-STATE CIRCUITS.

Nathaniel Pinckney (Member, IEEE) received the B.S. degree in engineering from the Harvey Mudd College, Claremont, CA, USA, in 2008, and the M.S. and Ph.D. degrees in electrical engineering from the University of Michigan, Ann Arbor, MI, USA, in 2012 and 2015, respectively.

From 2008 to 2010, he was with Sun Microsystems' VLSI Research Group, Menlo Park, CA. Since 2015, he has been with NVIDIA, Inc., Austin, TX, USA, where he is currently a Senior Research Scientist. He has authored or coauthored over 40 publications in the areas of high-level synthesis methodologies, low-power very large scale integration (VLSI) design, and cryptographic accelerators.

Priyanka Raina (Member, IEEE) received the B.Tech. degree in electrical engineering from IIT Delhi, New Delhi, India, in 2011, and the S.M. and Ph.D. degrees in electrical engineering and computer science from the Massachusetts Institute of Technology, Cambridge, MA, USA, in 2013 and 2018, respectively.

She was a Visiting Research Scientist with NVIDIA Corporation, Santa Clara, CA, USA. She is currently an Assistant Professor of electrical engineering with Stanford University, Stanford, CA, where she works on domain-specific hardware architectures and design methodology.

Stephen G. Tell (Member, IEEE) received the B.S.E. degree in electrical engineering from Duke University, Durham, NC, USA, in 1989, and the M.S. degree in computer science from the University of North Carolina at Chapel Hill (UNC/Chapel Hill), Chapel Hill, NC, in 1991.

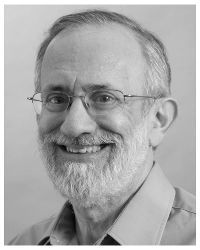
From 1991 to 1999, he worked on parallel graphics systems and high-speed signaling as a Senior Research Associate with UNC/Chapel Hill. In 1999, he joined Velio, Inc., to develop circuits and control systems for high-speed SerDes products. This work continued at Rambus, where he designed the logic for a SerDes with the lowest energy per bit demonstrated up to that time. In 2009, he joined NVIDIA, Inc., Durham, NC, as a Founding Member of the Circuits Research Group, where he works as a Senior Research Scientist. He holds more than ten U.S. patents. His current research interests include custom circuit design and the surrounding logic for intra- and inter-chip communication.

Yanqing Zhang (Member, IEEE) received the Ph.D. degree in electrical engineering from the University of Virginia, Charlottesville, VA, USA, in 2013.

Since January 2014, he has been a Research Scientist with NVIDIA, Inc., Santa Clara, CA, USA. His research interests include machine learning for electronic design automation (EDA) applications, digital very large scale integration (VLSI) methodology, variation-resilient digital design, and latch-based timing.

William J. Dally (Fellow, IEEE) is currently a Chief Scientist and a Senior Vice President of Research at NVIDIA, Inc., Santa Clara, CA, USA, and a Professor (Research) and a former Chair of Computer Science at Stanford University, Stanford, CA. He is currently working on developing hardware and software to accelerate demanding applications, including machine learning, bioinformatics, and logical inference. He has a history of designing innovative and efficient experimental computing systems. At Bell Labs, he contributed to the BELLMAC32 microprocessor and designed the MARS hardware accelerator. At the California Institute of Technology, Pasadena, CA, he designed the MOSSIM Simulation Engine and the Torus Routing Chip that pioneered wormhole routing and virtual-channel flow control. At the Massachusetts Institute of Technology, Cambridge, MA, USA, his group built the J-Machine and the M-Machine, experimental parallel computer systems that pioneered the separation of mechanisms from programming models and demonstrated very low overhead synchronization and communication mechanisms. At Stanford University, his group has developed the Imagine processor, which introduced the concepts of stream processing and partitioned register organizations, the Merrimac supercomputer, which led to GPU computing, and the ELM low-power processor. He also leads projects on computer architecture, network architecture, circuit design, and programming systems. He has published over 250 articles in these areas, holds over 160 issued patents, and is an author of the textbooks: *Digital Design: A Systems Approach*, *Digital Systems Engineering*, and *Principles and Practices of Interconnection Networks*.

Dr. Dally is also a member of the National Academy of Engineering and a fellow of the ACM and the American Academy of Arts and Sciences. He received the ACM Eckert-Mauchly Award, the IEEE Seymour Cray Award, the ACM Maurice Wilkes Award, the IEEE-CS Charles Babbage Award, and the IPSJ FUNAI Achievement Award.



Joel S. Emer (Fellow, IEEE) received the B.S. (Hons.) and M.S. degrees in electrical engineering from Purdue University, West Lafayette, IN, USA, in 1974 and 1975, respectively, and the Ph.D. degree in electrical engineering from the University of Illinois at Urbana–Champaign, Champaign, IL, USA, in 1979.

He was with Intel, Hudson, MA, USA, where he was an Intel Fellow and the Director of Microarchitecture Research. At Intel, he led the VSSAD Group. He was an employee of Compaq, Hudson, and Digital Equipment Corporation, Hudson. He is currently a Senior Distinguished Research Scientist with the NVIDIA's Architecture Research Group, Westford, MA, USA, where he is responsible for exploration of future architectures and modeling and analysis methodologies. He is also a Professor of the Practice at the Massachusetts Institute of Technology, Cambridge, MA, where he teaches computer architecture and supervises graduate students. He has held various research and advanced development positions investigating processor microarchitecture and developing performance modeling and evaluation techniques. He made architectural contributions to a number of VAX, Alpha, and X86 processors and is recognized as one of the developers of the widely employed quantitative approach to processor performance evaluation. He has been recognized for his contributions in the advancement of simultaneous multithreading technology, processor reliability analysis, cache organization, and spatial architectures for deep learning.

Dr. Emer is a fellow of the ACM. He was a recipient of numerous public recognitions. In 2009, he received the Eckert-Mauchly Award for lifetime contributions in computer architecture, the Purdue University Outstanding Electrical and Computer Engineer Alumni Award, and the University of Illinois Electrical and Computer Engineering Distinguished Alumni Award in 2010 and 2011, respectively. His 1996 article on simultaneous multithreading received the ACM/SIGARCH-IEEE-CS/TCCA: Most Influential Paper Award in 2011. He was named to the ISCA and Micro Halls of Fame in 2005 and 2015, respectively. He has had six articles selected for the IEEE Micro's Top Picks in Computer Architecture in 2003, 2004, 2007, 2013, 2015, and 2016. He was the Program Chair of ISCA in 2000 and the 2017 Program Chair for the International Symposium on Microarchitecture (MICRO).

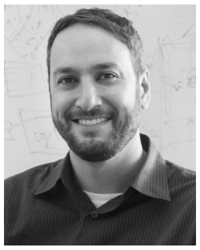
C. Thomas Gray (Senior Member, IEEE) received the B.S. degree in computer science and mathematics from the Mississippi College, Clinton, MS, USA, and the M.S. and Ph.D. degrees in computer engineering from North Carolina State University, Raleigh, NC, USA.

From 1993 to 1998, he was an Advisory Engineer at IBM, Research Triangle Park, NC, working in transceiver design for communication systems. From 1998 to 2004, he was a Senior Staff Design Engineer with the Analog/Mixed Signal Design Group, Cadence Design Systems, Cary, NC, USA, working on SerDes system architecture. From 2004 to 2010, he was Consultant Design Engineer with Artisan/ARM, Cary, and the Technical Lead of SerDes architecture and design. In 2010, he joined Nethra Imaging, Cary, as a System Architect. His work experience includes digital signal processing design and CMOS implementation of DSP blocks, as well as high-speed serial-link communication systems, architectures, and implementation. In 2011, he joined NVIDIA, Durham, NC where he is currently the Sr. Director of Circuits Research leading activities related to high-speed signaling, low-energy, and resilient memories, circuits for machine learning, and variation tolerant clocking and power delivery.

Stephen W. Keckler (Fellow, IEEE) received the B.S. degree in electrical engineering from Stanford University, Stanford, CA, USA, in 1990, and the M.S. and Ph.D. degrees in computer science from the Massachusetts Institute of Technology, Cambridge, MA, USA, in 1992 and 1998, respectively.

From 1998 to 2012, he was tenured-track Professor with The University of Texas at Austin (UT-Austin), Austin, TX, USA, where he developed scalable parallel processor and memory system architectures, including non-uniform cache architectures, explicit data graph execution processors, which merge dataflow execution with sequential memory semantics, and micro-interconnection networks to implement distributed processor protocols. All of these technologies were demonstrated in the TRIPS experimental computer system. He joined NVIDIA, Austin, in 2010, where he is currently the Vice President of Architecture Research and focuses on architectures for massively parallel and energy-efficient systems, domain-specific accelerators, and memory systems.

Dr. Keckler is a fellow of the ACM and an Alfred P. Sloan Research Fellow. He was a recipient of the NSF CAREER Award, the ACM Grace Murray Hopper Award, the President's Associates Teaching Excellence Award at UT-Austin, and the Edith and Peter O'Donnell Award for Engineering.



Brucek Khailany (Senior Member, IEEE) received the Ph.D. degree in electrical engineering from Stanford University, Stanford, CA, USA, in 2003, and the B.S.E. degree in electrical engineering from the University of Michigan, Ann Arbor, MI, USA, in 1997.

From 2004 to 2009, he was a Co-Founder and a Principal Architect at Stream Processors, Inc. (SPI), Sunnyvale, CA, USA, where he led research and development activities related to parallel processor architectures. He joined NVIDIA, Austin, TX, USA, in 2009, where he is currently the Director of the ASIC and VLSI Research Group. He leads research into innovative design methodologies for integrated circuit (IC) development, machine learning (ML) and GPU-assisted electronic design automation (EDA) algorithms, and energy-efficient ML accelerators. Over ten years at NVIDIA, he has contributed for many projects in research and product groups spanning computer architecture and very large scale integration (VLSI) design.