A 1.4 GHz 695 Giga RISC-V Inst/s 496-core Manycore Processor with Mesh On-Chip Network and an All-Digital Synthesized PLL in 16nm CMOS

Austin Rovinski[1], Chun Zhao[2], Khalid Al-Hawaj[3], Paul Gao[2], Shaolin Xie[2], Christopher Torng[3], Scott Davidson[2], Aporva Amarnath[1], Luis Vega[2], Bandhav Veluri[2], Anuj Rao[4], Tutu Ajayi[1], Julian Puscar[4], Steve Dai[3], Ritchie Zhao[3], Dustin Richmond[2], Zhiru Zhang[3], Ian Galton[4], Christopher Batten[3], Michael B Taylor[2], Ronald G Dreslinski[1]

[1]U. Michigan, Ann Arbor, MI; [2]U. Washington, Seattle, WA; [3]Cornell U., Ithaca, NY; [4]UC - San Diego, San Diego, CA;

Abstract - This paper presents a 16nm 496-core RISC-V network-on-chip (NoC). The mesh achieves 1.4GHz at 0.98V, yielding a peak of 695 Giga RISC-V instructions/s (GRVIS) and a record 812,350 CoreMark benchmark score. The main feature is the NoC architecture, which uses only 1881µm^2 per router node, enables highly scalable and dense compute, and provides up to 361 Tb/s of aggregate bandwidth.

Introduction

Complex, data-parallel workloads continue to push towards edge devices, such as mobile and IoT platforms. In particular, streaming-based workloads like real-time computer vision are steadily increasing in demand. Mobile devices demand high energy efficiency to attempt these computationally-intensive workloads. At the same time, the hardware must remain flexible to perform state-of-the-art algorithms as well as workloads that emerge post-fabrication. Prior high-efficiency manycore architectures[1-3] that target streaming workloads have yielded high area and energy efficiencies (Table 3). However, much of the die area for these architectures were dedicated towards the NoC, including cache-coherence protocol controllers, which limits compute density and efficiency. We demonstrate a new NoC architecture that enables fast inter-node communication with a significantly reduced die area (3.7x-118x) compared to prior work. The processor is composed of a 496-core array of 5-stage, in-order RISC-V RV32IM cores in a mesh configuration (Fig. 1). It achieves a peak of 695 GRVIS and a record 812,350 CoreMark benchmark score.

Manycore Architecture

In order to achieve a high compute density, the network architecture (Fig. 1) differs significantly from a traditional coherent shared-memory model. Instead of caches, the manycore processor has a partitioned global physical address space across all network nodes. Each core's memory occupies an address range which is globally accessible by any core over the network. The network enforces remote stores as part of the Remote Store Programming (RSP) model, which both obviates logic and prevents pipeline stalls associated with long-latency remote loads. The network has a single virtual channel with dimension-ordered routing, which greatly simplifies the network logic. With RSP, this guarantees deadlock-free, in-order delivery. The routers are single-stage, which allows for minimal memory to hold in-transit messages and single-cycle latency per hop. Rate limiting and memory fences are implemented via source-controlled credit counters. Credits are returned over a separate 9-bit NoC with the same architecture as Fig. 1. To enter and exit the NoC, messages are sent to an address below the bottom of the mesh, which will be sent to a mesh-attached host processor running a full operating system, such as Linux.

Fig. 2 shows the layout of a single tile, which contains an RV32IM core and the routing logic for that node. The core contains 2x 4KB SRAMs for I- and D-MEMs, and a 32-entry, 32b register file implemented using two 1r1w latch-based memories. By cell area, the core occupies 17863µm^2 (90.5%) and the router occupies 1881µm^2 (9.5%). The router supports 80b transfers per cycle, which packages data, address, and commands into a single flit. This technique provides a faster and simplified model compared to traditional approaches (Fig. 5, Table 2). The router and core run on the same clock domain up to 1.4GHz, allowing each tile to both transfer 750 Gb/s and process 1.4 GRVIS. Several gaps were created between rows of tiles to allow for ESD cells and In-Cell Overlays (ICOVL) as required for fabrication (Fig. 6). The total die area of the manycore is 15.25mm^2 as fabricated with ESD+ICOVL (or 12.03mm^2 without). This yields an area efficiency of 45.57 GRVIS/mm^2 (57.77 GRVIS/mm^2).

The processor clock is supplied by a fully synthesized and automatically placed-and-routed clock generator. It operates from an isolated 0.8V supply and occupies 5898µm^2. The output frequency is tunable from 10MHz to 3.3GHz in steps of ≤ 2%, with a (simulated) period jitter of < 2.5ps. The core is a 1st-order FDC-based PLL (Fig. 5). The 16 ring DCOs together cover 1.3-3.3 GHz. Each DCO inverter delay element is loaded with a bank of NAND gate frequency control elements (FCEs)[4], 37 of which are controlled by the DCO drift compensator to adjust for temperature and supply variations. The DCO control logic partitions its input into integer and fractional parts. The former drives 8 FCEs with an update rate of f_{ref} = 26 MHz. The latter is oversampled by a 2nd-order ΔΣ modulator which drives 8 FCEs through a dynamic element matching encoder.

Experimental Results

We run the industry-standard CoreMark benchmark. The benchmark was slightly modified by combining two loops in order to reduce the binary size by 80B to fit within a tile's I-MEM. Fig. 3 identifies the operating configurations where CoreMark reports a correct result for all tiles. The processor achieves a max throughput of 695 GRVIS at 1.4GHz and 0.98V – *the highest single-chip RISC-V throughput to date* – and a max energy efficiency of 314.89 GRVIS/W at 500MHz and 0.60V. It achieves a record CoreMark score of 812,350, *outperforming the next best score by more than 2x*.

Our NoC router outperforms all compared works for normalized area (3.7x-118x), minimum latency, and overhead (Table 2). Kilocore[3] modestly outperforms this work in network bandwidth, however it uses a circuit switched network which is statically routed prior to runtime. The performance exceeds TILE64[1] and Piton[2] by at least 27.6x for normalized area efficiency, 7.0x for energy efficiency, and 4.8x for throughput. Kilocore[3] performs similarly for area efficiency, although it uses a 16-bit datapath and a small memory size (7x less I+D-Mem than this work). We still achieve a 2.1x-46.8x higher energy efficiency than the compared works. ESSCIRC '14[5] reports the state-of-the-art in GRVIS throughput, which we outperform by 267x.

This research employed a BaseJump ASIC Motherboard;
the bringup effort was partly funded by the DARPA/SRC JUMP ADA center.

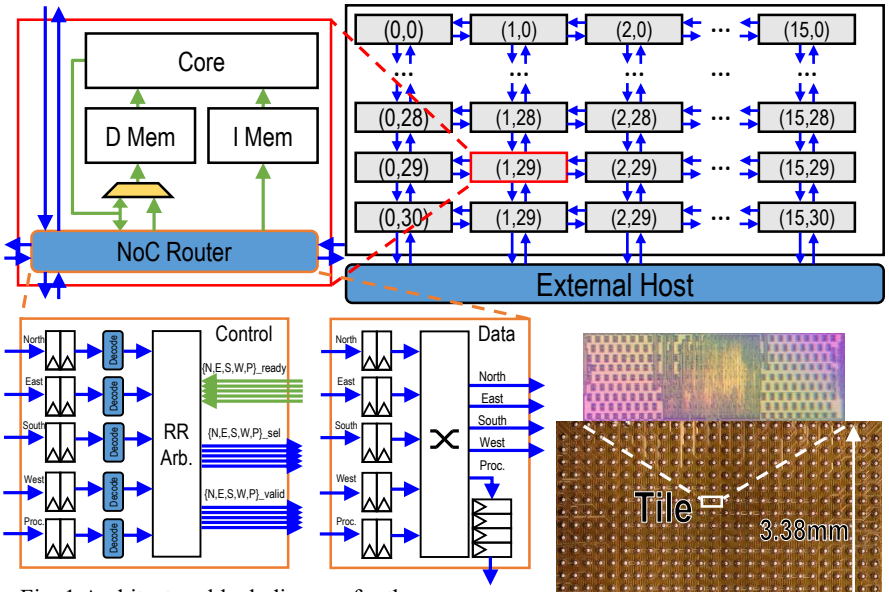

Fig. 1 Architecture block diagram for the manycore, manycore tile, and NoC.

Table 1 Tile area breakdown.

	Cell type	Area	%
	IMEM	6691um²	27.59%
	DMEM	6691um²	27.59%
	RF	2008um²	8.28%
	Core logic	2473um²	10.20%
	ALU	485um²	2.00%
	Div	412um²	1.70%
	Mult	301um²	1.24%
	Pipeline/other	1275um²	5.26%
	NoC	1881um²	7.76%
	Endpoint FIFO	303um²	1.25%
	Credit counter	23um²	0.09%
	Router	1555um²	6.41%
	Endcap/welltap	281um²	1.16%
	Filler	1635um²	6.74%
	Unutilized	2591um²	10.68%
	Total	24251um²	100.00%

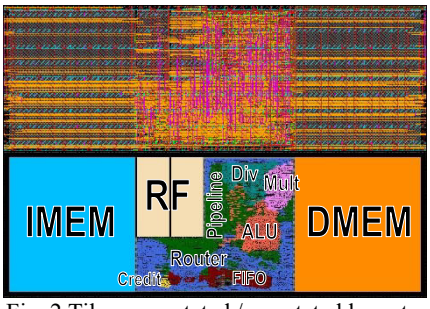

Fig. 2 Tile unannotated / annotated layouts.

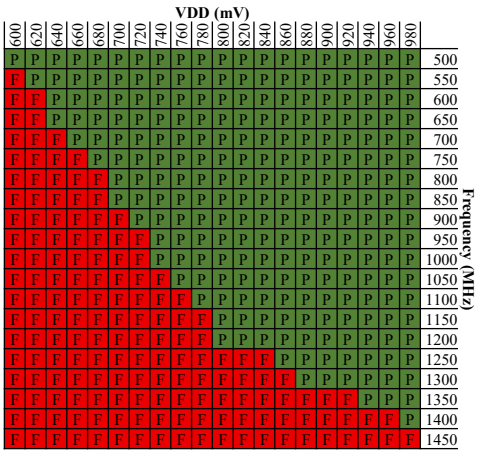

Fig. 3 Shmoo plot of operational configurations

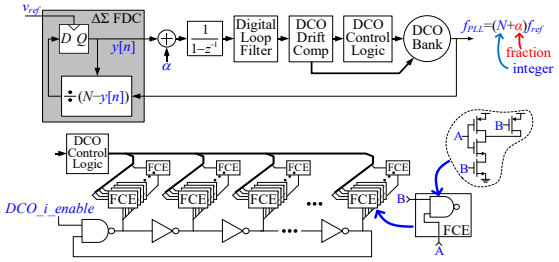

Fig. 4 High-level block diagram of the clock generator's core PLL, and one of its DCOs.

Fig. 6 Annotated die micrograph with callout to a delayered tile

References

[1] S. Bell, et al., *ISSCC*, pp. 567-568, Feb 2008.
[2] M. McKeown, et al., *HPCA*, Feb 2018.
[3] B. Bohnenstiehl, et al., *JSSC*, Apr 2017.
[4] P.-L. Chen, et al., *Trans. on Circ. and Sys.*, May 2005.
[5] Y. Lee, et al., *ESSCIRC*, 2014.

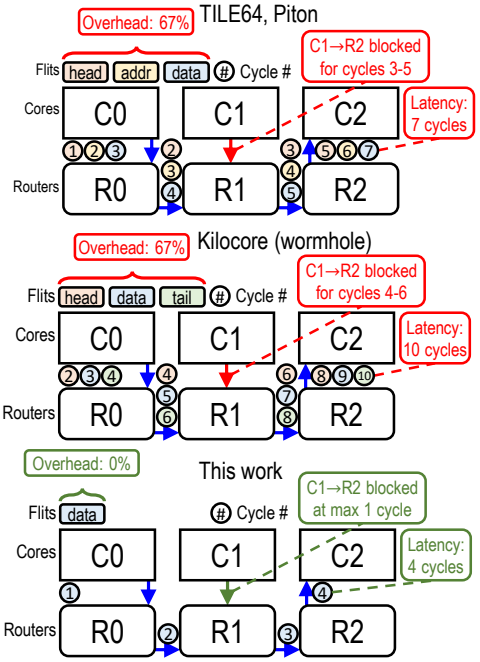

Fig. 5 Examples of sending a message from Core 0 to Core 2 for each architecture (4 hops, 1 data flit, 0 turns). Core → router hops count as turns for TILE64/Piton.

Table 2 Impact of routing models on network performance.

Work	Routing model	Arbitrary destination	Head-of-line blocking	Packet throughput	Min. latency (cycles)	Overhead flit fraction
TILE64[1], Piton[2]	Wormhole	Yes	Yes	0.33 / cycle	$h+n+t+1$	$2/(n+1)$
Kilocore[3]	Wormhole	Yes	Yes	0.33 / cycle	$2h+n+1$*	$2/(n+2)$
	Circuit switched	No	N/A⁺	Not Reported	Not Reported	Not Reported
This work	Single-flit packet	Yes	No	1 / cycle	$h+n-1$	0

h hops, n data flits, t turns in the network path. Core↔router is 1 hop.
* Kilocore's network is GALS and requires synchronization for each hop
⁺ Kilocore's circuit switch NoC can only be reprogrammed during the processor configuration phase

Table 3 Comparison to prior works.

	ISSCC '08[1]	HPCA '18[2]	JSSC '17[3]	ESSCIRC '14[6]	**This work**
ISA	VLIW	SPARC V9	RISC	RISC-V	RISC-V
Datapath Width	32-bit	64-bit	16-bit	64-bit	32-bit
Technology	90nm Planar	32nm SOI	32nm SOI	45nm SOI	16nm FinFET
Voltage	0.90 - 1.30 V	0.80 - 1.20 V	0.67 - 1.10 V	0.65 - 1.20 V	0.60 – 0.98 V
Area[a]	~232.16 mm²	29.37 mm²	57.41 mm²	3.08 mm²	15.25 (12.03[c]) mm²
Normalized Area[ab]	~87.06 mm²	20.33 mm²	39.75 mm²	1.46 mm²	15.25 (12.03[c]) mm²
Normalized NoC Router Area[ab]	~221051 um² (5x32 bit)	23420 um² (3x64 bit)	6910 um² (16 + 2x16 bit)	—	**1881 um² (80 + 9 bit)**
Cores (Threads)	64 (64)	25 (50)	1000 (1000)[d]	2 (2)	496 (496)
Frequency	750 MHz	500 MHz	1770 MHz	200 - 1300 MHz	10 - 1400 MHz
Power	10.8 W	2 W	39.6 W[d]	0.96 W	7.47 W
Throughput	144 GOPS	2.5 GOPS	1770 GOPS	2.6 GRVIS	695 GRVIS
Network Aggregate Bandwidth[e]	33.79 Tb/s	11.33 Tb/s	53.4 Tb/s (w-hole) 335 Tb/s (circuit)	—	361 Tb/s
Network Bisection Bandwidth[f]	1.92 Tb/s	0.96 Tb/s	0.58 Tb/s (w-hole) 3.65 Tb/s (circuit)	—	4.00 Tb/s
Routing Model	Wormhole	Wormhole	Wormhole+circuit	—	**Single-flit packet**
Energy Efficiency	13.33 GOPS/W (32-bit)	1.25 GOPS/W (64-bit)	44.70 GOPS/W[d] (16-bit)	2.71 GRVIS/W (64-bit)	**93.04 GRVIS/W** (32-bit)
Normalized Area Efficiency[ab]	1.65 GOPS/mm² (32-bit)	0.12 GOPS/mm² (64-bit)	44.52 GOPS/mm² (16-bit)	1.78 GRVIS/mm² (64-bit)	**45.57 (57.77[c]) GRVIS/mm²** (32-bit)

[a] Area only includes die area allocated to tiles.
[b] Area normalized to 16nm based on Contacted Poly Pitch (CPP) scaling
[c] Excluding ESD and ICOVL area
[d] Kilocore can only power 160 cores from its package. Power extrapolated to 1000 cores.
[e] Network Aggregate Bandwidth = (# usable links) * (link bandwidth)
[f] Network Bisection Bandwidth = (min. # links cut to bisect network) * (link bandwidth)