Cheshire: A Lightweight, Linux-Capable RISC-V Host Platform for Domain-Specific Accelerator Plug-In

Cheshire: A Lightweight, Linux-Capable RISC-V Host Platform for Domain-Specific Accelerator Plug-In

Alessandro Ottaviano, *Student Member, IEEE*, Thomas Benz, *Student Member, IEEE*, Paul Scheffler, *Student Member, IEEE*, and Luca Benini, *Fellow, IEEE*

***Abstract*—Power and cost constraints in the internet-of-things (IoT) extreme-edge and TinyML domains, coupled with increasing performance requirements, motivate a trend toward heterogeneous architectures. These designs use energy-efficient application-class host processors to coordinate compute-specialized multicore accelerators, amortizing the architectural costs of operating system support and external communication. This brief presents *Cheshire*, a lightweight and modular 64-bit Linux-capable host platform designed for the seamless plug-in of domain-specific accelerators. It features a unique low-pin-count DRAM interface, a last-level cache configurable as scratchpad memory, and a DMA engine enabling efficient data movement to or from accelerators or DRAM. It also provides numerous optional IO peripherals including UART, SPI, I2C, VGA, and GPIOs. Cheshire's synthesizable RTL description, comprising all of its peripherals and its fully digital DRAM interface, is available free and open-source. We implemented and fabricated Cheshire as a silicon demonstrator called *Neo* in TSMC's 65nm CMOS technology. At 1.2 V, Neo achieves clock frequencies of up to 325 MHz while not exceeding 300 mW in total power on data-intensive computational workloads. Its RPC DRAM interface consumes only 250 pJ/B and incurs only 3.5 kGE in area for its PHY while attaining a peak transfer rate of 750 MB/s at 200 MHz.**

***Index Terms*—RISC-V, Linux, PHYs, memory controllers, domain-specific architectures, heterogeneous computing, VLSI**

I. Introduction

With Koomey's law [1] slowing down, computer architects turn to hardware specialization to meet the rising energy efficiency requirements of data-intensive applications like machine learning and near-sensor processing. Today's heterogeneous architectures couple conventional application-class (AC) host processors to domain-specific architectures (DSAs) or programmable many-core accelerators (PMCAs) [2]–[5], improving energy efficiency while maintaining programmability and general-purpose operating system (GPOS) support. Thus, to maximize compute efficiency, architects should maximize the area and energy resources spent on efficient accelerators and minimize the cost and energy footprint of AC hosts.

Heterogeneous architectures are established in mobile and high-performance applications: silicon-proven industrial [6]–[8] and academic [9], [10] systems-on-chip (SoCs) typically target large power envelopes or multiple AC cores. However, heterogeneity is now penetrating the internet of things (IoT) extreme-edge and TinyML domains where power and cost constraints are extremely tight. Heterogeneous SoCs in these domains [11], [12] typically use lightweight 32-bit microcontrollers without AC capabilities or GPOS support as management processors. The recent *HULK-V* [13] SoC demonstrates a more capable host by coupling a Linux-capable 64-bit RISC-V manager core to an eight-core PMCA for high-end tinyML applications. However, it has not been silicon-proven, and its PMCA's ad-hoc integration does not provide a reusable interface. Furthermore, HULK-V's external HyperRAM [14] memory interface is low-bandwidth, limiting scalability and potentially incurring host-accelerator memory bottlenecks.

To address these shortcomings, we present *Cheshire*, an energy-efficient, Linux-capable, 64-bit RISC-V host platform for the seamless heterogeneous plug-in of DSAs such as [15], [16]. Cheshire is fully modular and provides a configurable interconnect, numerous optional peripherals, and a direct memory access (DMA) engine to decouple host-DSA communication. It integrates the first synthesizable open-source interface for reduced pin count (RPC) DRAM, a recent low-pin-count DRAM solution incurring lower integration cost and effort than low-power double data rate (LPDDR) DRAM, but significantly higher bandwidth than HyperRAM at comparable energy efficiency. Cheshire's synthesizable register-transfer level (RTL) description and FPGA implementation are available open-source[1]. We present the following contributions:

- A minimal, customizable, and energy-efficient Linux-capable RV64GC host platform that can easily be co-integrated with on-chip or chiplet DSAs; we provide a configurable Advanced eXtensible Interface 4 (AXI4) interconnect and a digital die-to-die (D2D) interface.
- The first *fully digital*, technology-independent RPC-DRAM-compliant memory interface, which incurs only 22 switching IOs and $3.5\,\mathrm{kGE}$ in PHY area. It enables memory accesses at only $250\,\mathrm{pJ/B}$ while attaining a peak transfer rate of $750\,\mathrm{MB/s}$ at $200\,\mathrm{MHz}$, outperforming existing HyperRAM solutions. To the best of our knowledge, this is the first characterization of reduced-pin-count DRAM (RPC DRAM) in a taped-out open-source SoC.
- An agile memory system, enabling accesses to external RPC DRAM in only 8 cycles for a $32\,\mathrm{B}$ transfer while interacting with DSA memory space only when desired.
- A standalone demonstrator chip called *Neo*, fabricated in TSMC's $65\,\mathrm{nm}$ node. At $1.2\,\mathrm{V}$, Neo achieves clock frequencies of up to $325\,\mathrm{MHz}$ while consuming less than $300\,\mathrm{mW}$ during data-intensive computational workloads.

A. Ottaviano, T. Benz, and P. Scheffler contributed equally to this work.
A. Ottaviano, T. Benz, P. Scheffler and L. Benini are with the Integrated Systems Laboratory (IIS), ETH Zurich, Switzerland.
E-mail: {aottaviano,tbenz,paulsc,lbenini}@iis.ethz.ch
L. Benini is also with the Department of Electrical, Electronic and Information Engineering (DEI), University of Bologna, Bologna, Italy.

[1]https://github.com/pulp-platform/cheshire for Cheshire and https://github.com/pulp-platform/rpc_dram_controller for its RPC DRAM interface.

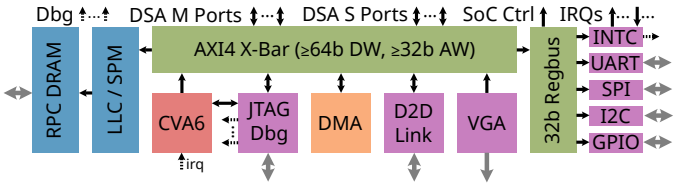

Fig. 1: Architecture of the Cheshire platform. The AXI4 crossbar provides a configurable number of **M**anager and **S**ubordinate ports toward a DSA.

II. Architecture

In the following section, we present the overall architecture of Cheshire, followed by an in-depth discussion of the fully-digital, technology-agnostic RPC DRAM interface.

A. Cheshire Platform

Cheshire is based on the energy-efficient 64-bit CVA6 [17] AC processor. It includes all hardware necessary to boot and run a GPOS like Linux autonomously, such as RISC-V-compliant core-local and platform interrupt controllers, various optional standard IO interfaces to access external storage and peripherals, and a DRAM interface. Off-chip DRAM is essential as the 8-16 MB memory footprint of simple embedded Linux systems [18] usually does not fit into on-chip memory.

Figure 1 shows Cheshire's architecture. A crossbar [19] using Arm's widespread AXI4 protocol [20] connects the CVA6 processor to RPC DRAM, the DSA, and other Cheshire-internal components. The crossbar's address width, data width, and the number of AXI4 DSA manager and subordinate ports are configurable to suit the target system's bandwidth and addressing needs. Simpler subordinates without burst or out-of-order transaction support are attached through a lightweight, extensible Regbus [21] demultiplexer, minimizing the crossbar's area and energy footprint.

Cheshire's RPC DRAM is connected through a configurable last-level cache (LLC). Each of the LLC's ways may individually be configured to serve as a scratchpad memory (SPM) at runtime, providing the host with fast internal static random access memory (SRAM) when needed. A RISC-V compliant debug module, backed by a JTAG transport module, enables live external debugging of CVA6 and any configured number of external RISC-V harts, e.g., those in PMCA DSAs. Likewise, the interrupt controllers support a configurable number of external sources and targets.

Cheshire provides various optional peripherals, including a flexible AXI4 DMA engine [22] for efficient data movement, a VGA controller for display output, and a digital D2D link for communication with off-chip systems. It also provides a UART for serial communication, a GPIO module, and I2C and SPI hosts to access external peripherals. All peripherals seamlessly integrate through AXI4 or Regbus interfaces and provide well-established feature sets for full compatibility with existing Linux drivers. An additional *SoC control* port connects to Cheshire-external on-chip devices essential for operation, such as clock generators, IO multiplexers, or clock and power domain controllers.

Cheshire has a built-in boot ROM, allowing for passive preloading through JTAG, UART, or the D2D link or autonomous boot from an external SPI Flash, I2C EEPROM, or SD card with Globally Unique Identifier Partition Table (GPT) support. Compiled with `-Os` flags and full-program link-time optimization, Cheshire's boot ROM is 7.2 KiB in size.

Although we use a single CVA6 as Cheshire's AC processor, multiple coherent CVA6 cores or another AC processor could be integrated instead; here, we focus on a minimal AC host.

We provide an open-source FPGA implementation of Cheshire, which enables the rapid prototyping and characterization of heterogeneous systems.

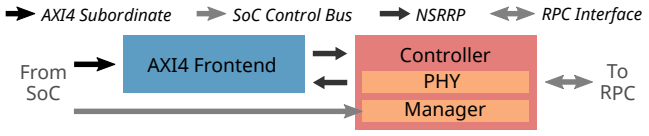

Fig. 2: Architecture of the RPC DRAM memory interface.

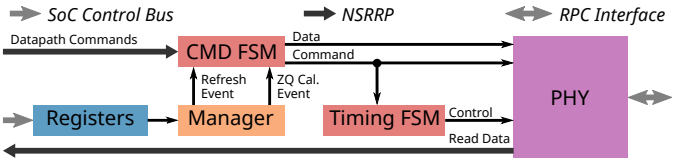

Fig. 3: Architecture of the RPC DRAM Controller.

B. RPC DRAM Interface

Background: RPC DRAM uses a reduced number of signals to deliver DDR3-level in-system bandwidth [23]. It multiplexes data, addresses, and commands onto a common data bus (*DB*) to minimize the needed pins. Command/address (CA) control can be sent concurrently with data by using a single multi-function pin, offering the high transaction efficiency of DDR3 at the cost of just 22 signals for a 16-bit wide *DB*.

Alternative DRAM solutions with low-pin-count interfaces have been proposed. Cypress' *HyperRAM* [14] requires only 12 switching IOs for an 8-bit shared bus. However, transfer rates are limited to 400 MB/s at 200 MHz or less, and its self-refresh precludes advanced controller-side scheduling. Antmicro proposes an FPGA-based Rowhammer testing platform based on the LiteDRAM memory controller [24]. The controller supports several DDR memory types and recently added an RPC physical interface circuit (PHY) implementation. Although open-source, this PHY targets FPGAs only.

Interface: Figure 2 depicts our RPC DRAM interface. It is comprised of two parts: a *controller* implementing the off-chip RPC protocol [23] and an AXI4 *frontend* implementing an AXI4-compliant subordinate. To enable easy adaptation to on-chip protocols other than AXI4, the controller and frontend are connected through a generic interface we call *non-stallable request-response protocol (NSRRP)*; its datawidth is 256 b or one *word* in the RPC DRAM standard. In the following, we will discuss the controller and AXI4 frontend in detail.

Controller: Figure 3 depicts the internal controller architecture. As shown, the controller receives *datapath commands* from the frontend. These generic commands are passed to the *command FSM*, which decomposes them into RPC DRAM-specific commands. For example, a generic datapath read is decomposed into 1) an *activate* of the corresponding bank and row, 2) a *read* of N consecutive RPC DRAM words, and 3) a *precharge* to close the bank and prepare it for the next access.

In addition to datapath commands issued by the frontend, the command FSM also handles *management commands* issued by a *manager* module inside the controller. The manager has three responsibilities: 1) it *initializes* the RPC DRAM device on startup, 2) it periodically *refreshes* active banks, and 3) it performs *ZQ calibration* when necessary. For these tasks, the manager uses configurable timing parameters, which

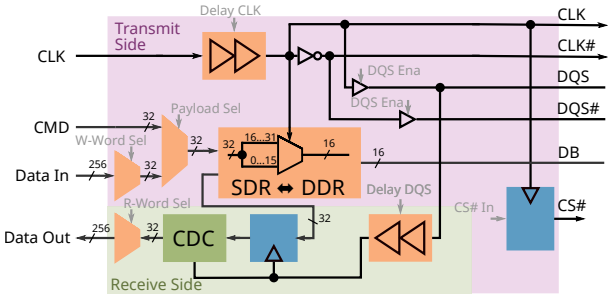

Fig. 4: Architecture of the RPC PHY.

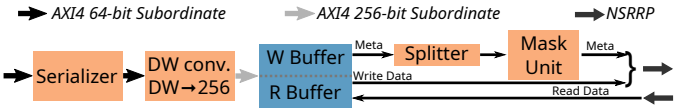

Fig. 5: Architecture of the RPC interface's AXI4 frontend.

can be set through a memory-mapped register file.

The command FSM passes its generated RPC DRAM commands to the *timing FSM*, which 1) *times commands*, ensuring that they adhere to protocol constraints like cycle alignment and minimum delays, and 2) *times the physical interface*, which includes controlling the chip select signals, gating the output strobe, and multiplexing the *DB*.

The *physical interface circuit* (PHY), shown in Figure 4, implements a low-power, digital-only, technology-agnostic RPC DRAM physical layer without internal clock generation. The PHY is composed of a *transmit side* sending data to the RPC DRAM, and a *receive side* accepting data from it.

The *transmit* side creates 90- and 270-degree phase-shifted clocks with a configurable delay line to drive the strobe signals DQS and DQS#, which are selectively enabled by the timing FSM. Any 256 b data words to be sent to the RPC DRAM are first serialized to 32 b *subwords*; the timing FSM then arbitrates between sending 32 b commands and 32 b data subwords using a multiplexer. The chosen payload is converted from single data rate (SDR) to DDR using clock-driven multiplexers as shown and placed on the *DB*.

The *receive* side accepts the strobe signal *DQS* in phase with the data read from the RPC DRAM chip through the *DB* pins. The received data is converted to SDR, then sampled by a delayed strobe generated with another configurable delay line. After passing through a clock domain crossing, the read data is packed to form complete 256 b words and then sent back to the AXI4 frontend over an NSRRP channel.

***AXI4 Frontend**:* The frontend, shown in Figure 5, implements an AXI4-compliant subordinate. Incoming requests are first *serialized* as the RPC DRAM controller operates strictly in order. Transfers from different AXI4 IDs are handled first come, first serve. After serialization, a *datawidth converter* converts the RPC DRAM interfaces' configured datawidth (64 b in the case of Neo) to RPC's 256 b word size.

After size conversion, protocol differences between AXI4 and RPC DRAM are reconciled. First, since AXI4 is fully stallable and RPC DRAM is not, the read and write channels are *buffered*. Next, a *splitter* splits NSRRP transactions at 2 KiB boundaries to comply with RPC protocol requirements. Finally, unaligned AXI4 transfers are handled by a *mask unit*.

III. Evaluation

We evaluate Cheshire through a silicon demonstrator named *Neo*, which was fabricated using TSMC's 65 nm node. Neo

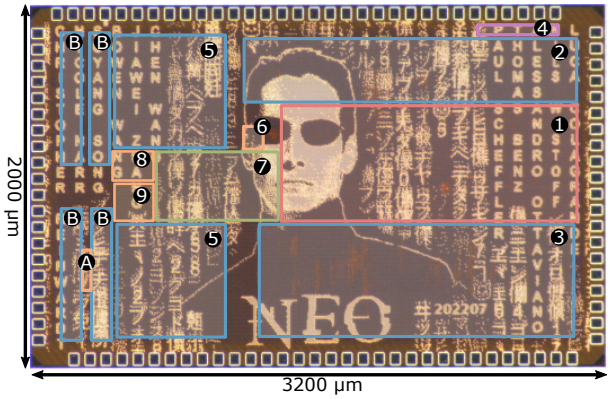

Fig. 6: Die shot of *Neo*: ❶ CVA6 and uncore region, ❷ D-cache, ❸ I-cache, ❹ D2D link, ❺ SPM, ❻ FLL, ❼ AXI4 interconnect and DMA, ❽ RPC manager, ❾ RPC AXI4 interface, ❿ RPC controller, ⓫ RPC buffer.

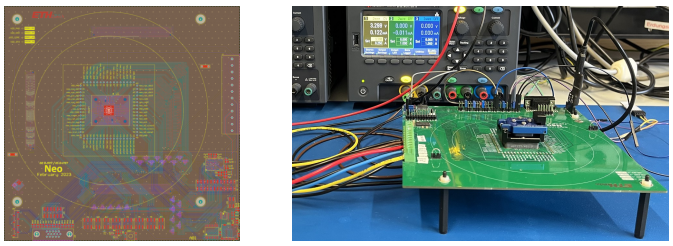

Fig. 7: Evaluation PCB to characterize *Neo* standalone and using industry-grade automated test equipment (ATE) equipment. (left) Layout of the board (right) fabricated PCB. DRAM traces are 25.23 cm long, 0.15 mm wide with JEDEC-compliant single-ended impedance of 50 Ω and differential of 90 Ω.

was synthesized using Synopsys DC 2019.03 and implemented using Cadence Innovus 2019.10, targeting a 200 MHz system clock in the SS corner at 125 °C. We will first describe how we configured Cheshire for Neo, then characterize the RPC DRAM and the AXI4 DSA interfaces in terms of functional performance and silicon implementation performance..

A. Silicon Demonstrator

Figure 6 shows an annotated die shot of Neo. Its 6.4 mm^2 die is housed in a QFN64 package. We use a core voltage of 1.2 V and a global IO voltage of 1.5 V. We configure Cheshire without DSA ports as we do not integrate any accelerator in this demonstrator. Neo features 128 KiB of SPM, 64 b data, and 48 b addresses; its CVA6 core is configured with 32 KiB 8-way level-one data and instruction caches. The RPC DRAM frontend is configured with 8 KiB buffers for read and write each. We include an on-chip frequency-locked loop (FLL)

We test Neo on a custom-made bring-up PCB shown in Figure 7. This board can be connected to industry-grade automated test equipment or operated standalone. It also features a set of peripherals and connectors to boot and run applications, including a 32 MiB RPC DRAM chip (EM6GA16LBXA-12H) connected to Neo and used for testing and measurements.

B. Functional Performance

Functional evaluation is performed through cycle-accurate RTL simulation. For all presented results, we leverage the efficient data movement capabilities of the DMA engine.

RPC DRAM specifies a maximum bus clock frequency of 933 MHz. Provided Neo's 16 b *DB* bus and an operating frequency of 200 MHz with DDR signaling, the peak attainable

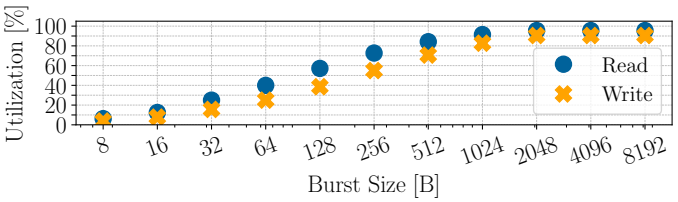

Fig. 8: Relative RPC DRAM bus utilization on reads and writes.

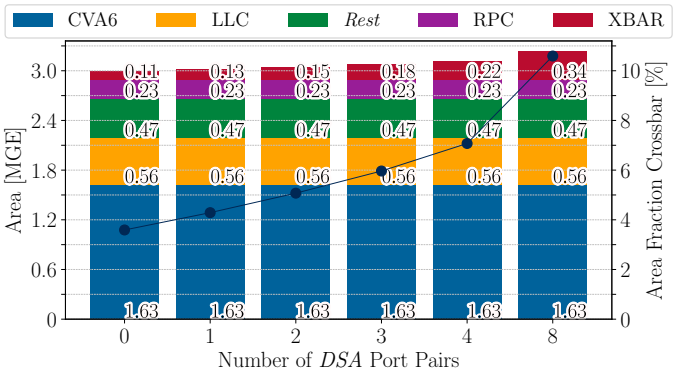

Fig. 9: Area breakdown of Cheshire implemented in TSMC65 and relative contribution of the crossbar for different numbers of DSA port pairs. *Rest* denotes the area of the DMA, peripherals, and interconnect adapters.

throughput is $\Theta = \alpha \cdot 800\,\mathrm{MB/s}$, where α is the relative bus utilization. Figure 8 shows the attained RPC bus utilization for both the read and write datapaths. The DMA is programmed to issue write and read transfers at increasing burst sizes starting from 8 B. The interface bus utilization plateaus close to peak utilization ($\alpha = 1$) for bursts 2 KiB in size or larger as they are decomposed into smaller transfers by the AXI4 frontend's transfer splitter. While the RPC pre- and postamble incur a constant, protocol-defined overhead independent of transfer size and controller implementation, bus utilization on reads is on average 1.3× higher than on corresponding writes; this is because read data is passed to the AXI4 bus as soon as possible whereas writes are deferred until enough data is buffered.

RPC DRAM proves its superiority in attainable bandwidth and area footprint against HyperBus, which was implemented in several academic works [12], [13]. HyperRAM occupies more PCB area, has a much lower maximum data rate of 400 MB/s, and is limited to a maximum frequency of only 200 MHz [14], restricting its applicability in high-bandwidth energy-efficient scenarios.

C. Silicon performance

Area breakdown: Figure 6 shows Neo's die, highlighting the approximate location and size of its main components. Figure 9 reports Cheshire's exact area breakdown in kGE, progressively increasing the number of DSA manager-subordinate port pairs attached to its main AXI4 crossbar. The leftmost bar corresponds to Neo's configuration without DSA ports. In all considered cases, CVA6 dominates Cheshire's area, while the RPC DRAM controller accounts for at most 7.6 %. As we increase the number of DSA ports, the all-to-all AXI4 crossbar grows from 3.6 % to 10.6 % of Cheshire, increasing its area by at most 7.8 % for eight port pairs compared to Neo's configuration without DSA ports. While crossbar scaling is a limiting factor for large numbers of port pairs, DSAs with many managers or subordinates can use a sub-interconnect, larger data widths, or sparse connectivity to facilitate scaling.

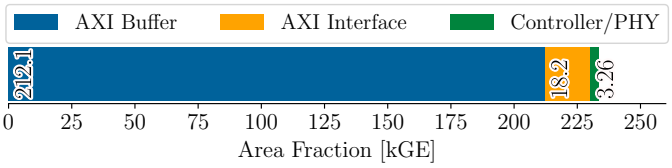

Fig. 10: Area breakdown of the RPC DRAM interface. When configured as in Neo, the *AXI4 buffer* and the *AXI4 Interface* occupy most of the area.

Figure 10 shows the area breakdown of Neo's RPC DRAM controller. The manager module, command/timing FSM, and digital PHY occupy only 1 % or 3.5 kGE of the controller's area, confirming the extremely low area overhead incurred by the RPC protocol for its memory interface. Most of the controller's area overhead is due to the buffers holding AXI4 beats. In Neo, these buffers are over-provisioned to simplify the initial design of the AXI4 frontend. Despite this, our controller occupies only 6.3 % of the area and 33 % of the beachfront of an existing 65 nm full-pin-count DDR3 controller [25].

Energy efficiency: Figure 11 shows Neo's power consumption for different scenarios and frequencies as measured on the bring-up board; each scenario explores an operational corner with different computational and memory intensities. We focus here on evaluating the Cheshire platform; for further benchmarks of the CVA6 processor, we refer the reader to its publication [17]. The bring-up board provides three power domains with their supplies: CORE, IO, and RAM. CORE feeds Neo's core area logic and SRAMs, IO provides power to its pads, and RAM supplies the RPC DRAM memory chips.

In the *WFI* scenario, CVA6 is waiting for an interrupt, idling without fetching or decoding instructions; this provides a power baseline with minimal switching. In *NOP*, CVA6 loops on a body of `nops`, establishing a floor for actively fetching, branching, and decoding workloads with few stalls. *2MM* runs an optimized double-precision floating-point matrix multiplication with arguments and results in RPC DRAM, keeping reusable matrix tiles in SPM. *MEM* writes high-throughput bursts to RPC DRAM using the DMA engine.

At a CORE supply of 1.2 V, Neo achieves clock frequencies of up to 325 MHz and remains within a 300 mW power envelope even in data-intensive computational scenarios like *2MM*. All power contributions scale linearly with frequency as expected, and CORE power dominates in almost all cases; at 200 MHz, 69 % of *MEM* power is consumed in CORE. Since the version of RPC DRAM interface proposed in this manuscript does not make use of the technology's *Deep Power Down* state, all benchmarks show an RAM idle power consumption. That being said, the RPC DRAM interface's IO power at 200 MHz for *MEM* is 45 % lower than that of an existing 65 nm DDR3 interface under high load [25].

We use *MEM* to compute RPC DRAM's transfer efficiency. We consider only the write direction, representing the worst-case scenario regarding buffering as discussed in Section II. Given the maximum bandwidth measured in Section III-B, RPC DRAM's interface energy per transferred byte is $\Gamma = \frac{P_{tot}}{\Theta} \simeq 250\,\mathrm{pJ/B}$. This result is comparable to reported energy-per-byte consumptions in recent works integrating lower-bandwidth memories like HyperRAM [12].

IV. Conclusion

This work presents Cheshire, a lightweight, modular, and open-source 64-bit Linux-capable host platform designed

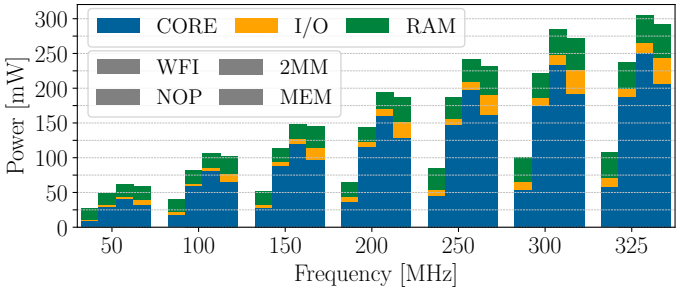

Fig. 11: Power consumption of Neo for the four workloads: *WFI*, *NOP*, *2MM*, and *MEM*. The power is split into the three power domains of Neo.

to plug in DSAs seamlessly. Cheshire features a unique RPC DRAM interface, a last-level cache, configurable as scratchpad memory, and a DMA engine enabling efficient data movement to or from accelerators or DRAM. It also provides numerous optional peripherals, including UART, SPI, I2C, a GPIO module, and VGA for display output. We implement and fabricate Cheshire as a silicon demonstrator called *Neo* in TSMC's 65nm CMOS technology. At 1.2 V, Neo achieves clock frequencies of up to 325 MHz while not exceeding 300 mW in total power even in data-intensive computational scenarios. Its DRAM interface consumes only 250 pJ/B while incurring only 3.5 kGE in area for its PHY and attaining a peak transfer rate of 750 MB/s at 200 MHz.

ACKNOWLEDGMENTS

We thank the reviewers for their feedback. We thank C. Jinfan, V. Krishna, R. Zhou, N. Narr, C. Reinwardt, S. Song, B. Wang, C. Wang, J. Zhang, N. Wistoff, and L. Colagrande for their contributions to this research. We also thank Etron Technology Inc., in particular R. Crisp, for their support. This work is supported in part through the FRACTAL (877056) project, which received funding from the ECSEL Joint Undertaking (JU) and TRISTAN (101095947) project, which received funding from the HORIZON KDT-JU programme.

REFERENCES

[1] J. Koomey, S. Berard, M. Sanchez, and H. Wong, "Implications of historical trends in the electrical efficiency of computing," *IEEE Ann. Hist. Comput.*, vol. 33, no. 3, pp. 46–54, 2011.

[2] Y. Chi, W. Qiao, A. Sohrabizadeh, J. Wang, and J. Cong, "Democratizing domain-specific computing," *Commun. ACM*, vol. 66, no. 1, p. 74–85, dec 2022.

[3] J. L. Hennessy and D. A. Patterson, "A new golden age for computer architecture," *Commun. ACM*, vol. 62, no. 2, p. 48–60, jan 2019.

[4] Y. Zhao, R. Xie, G. Xin, and J. Han, "A high-performance domain-specific processor with matrix extension of risc-v for module-lwe applications," *IEEE Trans. Circuits Syst. I: Reg. Papers*, vol. 69, no. 7, pp. 2871–2884, 2022.

[5] R. Paludo and L. Sousa, "Ntt architecture for a linux-ready risc-v fully-homomorphic encryption accelerator," *IEEE Trans. Circuits Syst. I: Reg. Papers*, vol. 69, no. 7, pp. 2669–2682, 2022.

[6] Raspberry Pi Foundation, "Rasperry pi zero," https://www.raspberrypi.com/products/raspberry-pi-zero, 2020.

[7] M. Ditty, "Nvidia orin system-on-chip," in *2022 IEEE Hot Chips 34 Symp.*, 2022, pp. 1–17.

[8] SiFive, "Hifive unmatched," https://www.sifive.com/boards/hifive-unmatched, 2022.

[9] J. Zuckerman, P. Mantovani, D. Giri, and L. P. Carloni, "Enabling heterogeneous, multicore soc research with risc-v and esp," 2022.

[10] J. Balkind, K. Lim, M. Schaffner, F. Gao, G. Chirkov, A. Li, A. Lavrov, T. M. Nguyen, Y. Fu, F. Zaruba, K. Gulati, L. Benini, and D. Wentzlaff, *BYOC: A "Bring Your Own Core" Framework for Heterogeneous-ISA Research*. New York, NY, USA: ACM, 2020, p. 699–714.

[11] A. Di Mauro, M. Scherer, D. Rossi, and L. Benini, "Kraken: A direct event/frame-based multi-sensor fusion soc for ultra-efficient visual processing in nano-uavs," in *IEEE Hot Chips 34 Symp.*, 2022, pp. 1–19.

[12] D. Rossi, F. Conti, M. Eggiman, A. D. Mauro, G. Tagliavini, S. Mach, M. Guermandi, A. Pullini, I. Loi, J. Chen, E. Flamand, and L. Benini, "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.

[13] L. Valente, Y. Tortorella, M. Sinigaglia, G. Tagliavini, A. Capotondi, L. Benini, and D. Rossi, "HULK-V: a Heterogeneous Ultra-low-power Linux capable RISC-V SoC," in *2023 Design, Automation & Test in Europe Conference & Exhibition (DATE)*, 2023.

[14] Cypress, "Hyperram & octal xspi ram memory," https://www.cypress.com/products/hyperram-octal-xspi-ram-memory.

[15] A. Garofalo, M. Rusci, F. Conti, D. Rossi, and L. Benini, "Pulp-nn: A computing library for quantized neural network inference at the edge on risc-v based parallel ultra low power clusters," in *26th IEEE Int. Conf. Electronics, Circuits and Syst.*, 2019, pp. 33–36.

[16] E. Manor and S. Greenberg, "Custom hardware inference accelerator for tensorflow lite for microcontrollers," *IEEE Access*, vol. 10, pp. 73 484–73 493, 2022.

[17] F. Zaruba and L. Benini, "The cost of application-class processing: Energy and performance analysis of a linux-ready 1.7-ghz 64-bit risc-v core in 22-nm fdsoi technology," *IEEE Trans. Very Large Scale Integration (VLSI) Syst.*, vol. 27, no. 11, pp. 2629–2640, Nov 2019.

[18] M. Opdenacker, "Linux in less than 4 mb of ram," https://bootlin.com/pub/conferences/2017/jdll/opdenacker-embedded-linux-in-less-than-4mb-of-ram/, 2017.

[19] A. Kurth, W. Ronninger, T. Benz, M. Cavalcante, F. Schuiki, F. Zaruba, and L. Benini, "An open-source platform for high-performance non-coherent on-chip communication," *IEEE Trans. Comput.*, pp. 1–1, 2021.

[20] A. Limited, "Amba axi and ace protocol specification version h.c," https://developer.arm.com/documentation/ihi0022/hc, 2021.

[21] *Generic Register Interface*, PULP Platform Contributors. [Online]. Available: https://github.com/pulp-platform/register_interface

[22] T. Benz, M. Rogenmoser, P. Scheffler, S. Riedel, A. Ottaviano, A. Kurth, T. Hoefler, and L. Benini, "A high-performance, energy-efficient modular dma engine architecture," 2023.

[23] Etron Technology, "256Mb high bandwidth rpc dram," https://etronamerica.com/wp-content/uploads/2019/05/EM6GA16LGDABMACAEA-RPC-DRAM_Rev.-1.0.pdf, 2019.

[24] Antmicro, "Open source ddr controller framework for mitigating rowhammer," https://antmicro.com/blog/2021/08/open-source-ddr-test-framework-for-rowhammer/, 2021.

[25] C. Sudarshan, J. Lappas, C. Weis, D. M. Mathew, M. Jung, and N. Wehn, "A lean, low power, low latency dram memory controller for transprecision computing," in *Embedded Comput. Syst.: Architectures, Model., Simul.* Springer Int. Publishing, 2019, pp. 429–441.