Scalable Open-Source System-on-Chip Design

(Invited Talk - Extended Abstract)

Luca P. Carloni
Department of Computer Science, Columbia University in the City of New York
New York, NY 10027
luca@cs.columbia.edu

***Abstract*—The system-on-chip is the dominant architecture in the age of heterogeneous computing, but its energy-efficient performance comes at the cost of higher design complexity. The open-source hardware movement responds to this challenge by promoting design reuse and collaboration. ESP is an open-source research platform for heterogeneous SoC design that combines a scalable tile-based architecture and a flexible system-level design methodology. Conceived as a heterogeneous integration platform, ESP is naturally suited to foster collaborative engineering of SoC designs across the open-source community.**

***Index Terms*—system-on-chip (SoC), open-source hardware (OSH), heterogeneous computing, system-level design, RISC-V.**

I. SoC Architectures Are Everywhere

Modern efficient computing is *heterogeneous computing*. The end of Dennard's ideal CMOS scaling [1], the slow-down of Moore's Law [2], [3], and the limits of effective parallelism that can be achieved with homogeneous multi-core processors [4] have pushed designers to achieve performance gains by specializing hardware. *Accelerators*—hardware computing engines that are specialized for a particular application or application domain—provide orders-of-magnitude gains in energy-efficient performance compared to general-purpose processors [5]. To balance specialization and programmability, designers combine accelerators and processors into heterogeneous architectures. Consequently, the *system-on-chip (SoC)* has become the principal computer architecture across the most important classes of computing. The SoC originally emerged in the design of embedded systems, where it continues to be the dominant computing engine for smartphones [6], automotive electronics [7], avionics [8] and the Internet-of-things [9]. Over the last decade, however, the continuous migration of devices from the board to the die [10] and the transfer of critical functionality from software to specialized hardware [4], [11] have made the SoC a popular choice also for personal computers and servers. More recently, as accelerators have gained ground in cloud computing [12], [13], the giants of the information technology industry have started designing their own SoCs [14].

II. SoC Design Is a Challenging Task

Since the quest to maximize energy-efficient performance passes through hardware specialization of computational kernels for critical application workloads, designers combine more and more heterogeneous components in the same SoC. A state-of-the-art SoC is a highly heterogeneous system that integrates many general-purpose processors, graphics processing units, digital signal processors, and accelerators [15]. Heterogeneity improves energy efficiency and performance, but it increases design complexity. A system consisting in a collection of diverse components is intrinsically more difficult to design, validate, and program than a system made only of homogeneous copies. At design time, the differences among heterogeneous components translate into diminished regularity in chip layout and the need to perform different verification tasks. At runtime, the presence of many heterogeneous components complicates the hardware-software interface and the management of shared resources, such as access to off-chip main memory and use of the tight on-chip power budget. With each SoC generation, the addition of new capabilities is increasingly limited by engineering effort and team sizes [16].

III. Open-Source Hardware to the Rescue

Open-source hardware (OSH) has been proposed as a vehicle to reenergize the innovation in the semiconductor industry in the mold of the proven success of the open-source software ecosystem [17]. The momentum of OSH has been building in recent years [18], [19], thanks particularly to the popularity of the RISC-V project [20]. The number of OSH projects is expected to grow steadily in the upcoming years, fueled by multi-institution organizations [21]–[23], government programs [24], and many diverse contributions from both academia [20], [25], [26] and industry [27], [28]. To date, however, most OSH projects are focused on the development of individual SoC components, such as a processor core or an accelerator. While certainly useful, this leaves open a critical challenge:

How can we realize a complete SoC for a given target application domain by efficiently reusing and combining a variety of independently developed, heterogeneous, OSH components, especially if these components are designed by separate organizations for separate purposes?

While the development of individual OSH components is a necessary precondition, the ultimate goal is the realization of complete SoC designs that leverage these components.

IV. Scaling Up Open-Source Hardware

Achieving this goal requires enabling design reuse and collaboration to directly mitigate the design complexity challenge. A possible path goes through innovations of the SoC architectures as well as the methodologies used to design them. Indeed, architectures and methodologies must be developed together

978-1-7281-5409-1/20/$31.00 ©2020 IEEE

in order to be effective. This approach is captured by the concept of *platform*, which is precisely the combination of an architecture and methodology [29]. Although an architecture limits the space of possible SoC designs, its properties allow for the development of an effective design methodology and supporting CAD tools. In turn, the methodology and tools allow designers to focus on the most important and creative aspects of the design process, while leveraging automation for the repetitive and error-prone tasks.

An SoC architecture enables design reuse when it simplifies the integration of many components that are independently developed. An SoC methodology enables design collaboration when it allows designers to choose the preferred specification languages and design flows for the various components, particularly when these choices provide advantages in the context of important application-specific domains. An effective combination of architecture and methodology is a platform that maximizes the potential of open-source hardware by scaling-up the number of components that can be integrated in an SoC and by enhancing the productivity of the designers who develop and use them.

V. An Open-Source Platform for SoC Design

ESP is an open-source research platform for heterogeneous SoC design [30]. The System-Level Design Group at Columbia University has developed ESP by building on the foundations of communication-based system-level design [31] and on years of experience teaching SoC platforms [32]. ESP combines a scalable architecture and a flexible methodology [29]. Just like the architecture simplifies the integration of heterogeneous components developed by different teams, the methodology embraces the use of various design flows for component development [33].

The ESP architecture is structured as a tile grid. The tiles form a distributed system which is inherently scalable, modular and heterogeneous. The main types of tile are three: processor, accelerator and memory. For the processor tile, ESP currently allows a seamless choice between the 32-bit LEON3 SPARC core [34] and the 64-bit ARIANE RISC-V core [35]. An accelerator tile contains one or more loosely-coupled accelerators [36]; these can be accelerators developed with the ESP methodology [37], [38] as well as third-party OSH accelerators like the NVIDIA NVDLA [27]. Processors and accelerators [36] are given the same importance in the SoC. This system-centric view distinguishes ESP from other OSH platforms, most of which take a processor-centric view.

Each tile is encapsulated into a *modular socket* (aka *shell*) that interfaces it to a network-on-chip (NoC), which has a packet-switched 2D-mesh topology with multiple physical planes [39]. Following the principles of the *protocols and shells paradigm* of latency-insensitive design [31], [40], the shell decouples the design of the tile content from the design of the rest of the system, thereby simplifying the integration of an OSH component inside a tile, enabling late-stage optimizations as part of design-space exploration decisions, and promoting its reuse across different SoC designs. Furthermore, the shell implements a set of *platform services*, which provide pre-validated solutions for common design tasks like accelerator configuration [37], memory management [41], [42], and dynamic voltage-frequency scaling [43].

The ESP methodology guides the choice of the number, mix, and placement of tiles for a target SoC as well as the design of newly-developed components. Third-party IP blocks can be seamlessly integrated [44]. For the development of new components, ESP promotes system-level design [31] and, particularly, the application of high-level synthesis to design-space exploration [45]–[47]. Indeed, the ESP methodology is flexible because it embraces different design flows from specifications written in different languages, including: C with Xilinx Vivado HLS, SystemC with Cadence Stratus HLS, C++/SystemC with Mentor Catapult HLS, as well as SystemVerilog, VHDL, and Chisel. Recently, a flow to design embedded machine learning accelerators with Keras TensorFlow, PyTorch and ONNX through hls4ml [48] became the first example of a domain-specific design flow added to ESP [38].

A graphical user interface allows the selection of the tiles, the number and parallelism of the NoC planes, and the structure of the memory hierarchy, among many other configuration parameters. Once configured, the RTL implementation of the SoC is automatically generated together with all the hardware and software mechanisms for system integration of the chosen processor core. The automatic generation of device drivers from pre-designed templates simplifies the invocation of accelerators from user-level applications running on Linux [41]. The automatic generation of a multi-plane NoC from a parameterized model supports the scaling of the ESP architecture to accommodate multiple cores, many accelerators, and a distributed memory hierarchy [49].

ESP allows SoC architects to rapidly implement FPGA-based prototypes of complex SoCs by combining third-party OSH components that use the AXI protocol (e.g. ARIANE and NVDLA) with newly-designed components. A growing set of tutorials and demos on how to realize these prototypes is available on the ESP website [30]. While the ESP release currently focuses on FPGA-based prototyping, the ESP methodology offers a natural and versatile front end, up to synthesizable RTL, for chip design.

VI. Conclusions

The concept of platform is the key to handling the complexity of SoC design in the age of heterogeneous computing. By combining a scalable architecture with a flexible methodology, ESP provides the open-source hardware community with a platform for collaborative engineering of SoC designs.

Acknowledgments. This work was sponsored in part by the Army Research Office and was accomplished under Grant Number W911NF-19-1-0476. 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 Army Research Office or the U.S. Government. The U.S. Government is authorized to reproduce and distribute reprints for Government purposes notwithstanding any copyright notation herein.

REFERENCES

[1] M. Bohr, "A 30 year retrospective on Dennard's MOSFET scaling paper," *IEEE Solid-State Circuits Society Newsletter*, vol. 12, no. 1, pp. 11–13, Winter 2007.

[2] R. K. Cavin, P. Lugli, and V. V. Zhirnov, "Science and engineering beyond Moore's Law," *Proc. of the IEEE*, vol. 100, pp. 1720–1749, May 2012.

[3] R. Colwell, "End of Moore's law," *IEEE Computer*, vol. 46, no. 12, p. 49, Dec. 2013.

[4] M. Horowitz, "Computing's energy problem (and what we can do about it)," in *ISSCC Digest of Technical Papers*, Feb. 2014, pp. 10–14.

[5] W. J. Dally, Y. Turakhia, and S. Han, "Domain-specific hardware accelerators," *Comm. of the ACM*, vol. 63, no. 7, pp. 48–57, Jun. 2020.

[6] Y. S. Shao, B. Reagen, G. Wei, and D. Brooks, "The Aladdin approach to accelerator design and modeling," *IEEE Micro*, vol. 35, no. 3, pp. 58–70, May-Jun 2015.

[7] G. P. Stein, E. Rushinek, G. Hayun, and A. Shashua, "A Computer Vision System on a Chip: a case study from the automotive domain," in *Conf. on Computer Vision and Pattern Recognition (CVPR'05)*, Sep. 2005, pp. 130–130.

[8] D. Keymeulen, S. Shin, J. Riddley, M. Klimesh, A. Kiely, E. Liggett, P. Sullivan, M. Bernas, H. Ghossemi, G. Flesch, M. Cheng, S. Dolinar, D. Dolman, K. Roth, C. Holyoake, K. Crocker, and A. Smith, "High performance space computing with system-on-chip instrument avionics for space-based next generation imaging spectrometers (NGIS)," in *NASA/ESA Conf. on Adaptive Hardware and Systems*, Aug. 2018, pp. 33–36.

[9] Y. Pu, C. Shi, G. Samson, D. Park, K. Easton, R. Beraha, A. Newham, M. Lin, V. Rangan, K. Chatha, D. Butterfield, and R. Attar, "A 9-mm2 ultra-low-power highly integrated 28-nm CMOS SoC for Internet of Things," *J. of Solid-State Circuits*, vol. 53, no. 3, pp. 936–948, Mar. 2018.

[10] S. Damaraju, V. George, S. Jahagirdar, T. Khondker, R. Milstrey, S. Sarkar, S. Siers, I. Stolero, and A. Subbiah, "A 22nm IA Multi-CPU and GPU System-on-Chip," in *ISSCC Digest of Technical Papers*, Feb. 2012, pp. 56–57.

[11] S. Borkar and A. Chen, "The future of microprocessors," *Communication of the ACM*, vol. 54, pp. 67–77, May 2011.

[12] A. M. Caulfield, E. S. Chung, A. Putnam, H. Angepat, J. Fowers, M. Haselman, S. Heil, M. Humphrey, P. Kaur, J. Kim, D. Lo, T. Massengill, K. Ovtcharov, M. Papamichael, L. Woods, S. Lanka, D. Chiou, and D. Burger, "A cloud-scale acceleration architecture," in *Proc. of the Intl. Symp. on Microarchitecture*, Oct. 2016, pp. 1–13.

[13] N. P. Jouppi, C. Young, N. Patil, and D. Patterson, "A domain-specific architecture for deep neural networks," *Comm. of the ACM*, vol. 61, no. 9, pp. 50–59, Aug. 2018.

[14] E. Jhonsa, "Why tech giants like Amazon are designing their own chips – and who benefits," https://www.thestreet.com/opinion/why-tech-giants-are-designing-their-own-chips-14807638, Dec. 2018.

[15] M. Ditty, A. Karandikar, and D. Reed, "NVIDIA's Xavier SoC," Intl. Symp. on High Performance Chips (HotChips'30), 2018.

[16] B. Khailany, E. Khmer, R. Venkatesan, J. Clemons, J. S. Emer, M. Fojtik, A. Klinefelter, M. Pellauer, N. Pinckney, Y. S. Shao, S. Srinath, C. Torng, S. L. Xi, Y. Zhang, and B. Zimmer, "A modular digital VLSI flow for high-productivity SoC design," in *Proc. of the Design Automation Conf. (DAC)*, Jun. 2018, pp. 72:1–72:6.

[17] G. Gupta, T. Nowatzki, V. Gangadhar, and K. Sankaralingam, "Kick-starting semiconductor innovation with open source hardware," *IEEE Computer*, vol. 50, no. 6, pp. 50–59, Jun. 2017.

[18] B. Bailey, "Open-source hardware momentum builds," https://semiengineering.com/riding-the-risc-v-wave/, Jun. 2020.

[19] The Economist, "The rise of open-source computing," Oct. 2019.

[20] K. Asanovic and D. Patterson, "The case for open instruction sets," *Microprocessor Report*, Aug. 2014.

[21] RISC-V Foundation, https://riscv.org/.

[22] CHIPS Alliance, https://chipsalliance.org/.

[23] OpenHWGroup, https://www.openhwgroup.org/.

[24] S. Moore, "DARPA picks its first set of winners in electronics resurgence initiative," https://spectrum.ieee.org/tech-talk/semiconductors/design/darpa-picks-its-first-set-of-winners-in-electronics-resurgence-initiative, Jul. 2018.

[25] 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. on Very Large Scale Integration Systems*, vol. 27, no. 11, pp. 2629–2640, Nov. 2019.

[26] J. Balkind, K. Lim, F. Gao, J. Tu, D. Wentzlaff, M. Schaffner, F. Zaruba, and L. Benini, "OpenPiton+Ariane: the first SMP Linux-booting RISC-V system scaling from one to many cores," in *Workshop on Computer Architecture Research with RISC-V (CARRV)*, 2019.

[27] NVIDIA, "NVIDIA Deep Learning Accelerator," www.nvdla.org, 2018.

[28] L. Armasu, "Western Digital bets big on RISC-V with own processor, other innovations," https://www.tomshardware.com/news/western-digital-risc-v-processor-open-source,38200.html, Feb. 2019.

[29] L. P. Carloni, "The case for embedded scalable platforms," in *Proc. of the Design Automation Conf. (DAC)*, Jun. 2016, pp. 17:1–17:6.

[30] Columbia SLD Group, "ESP Release," www.esp.cs.columbia.edu, 2019.

[31] L. P. Carloni, "From latency-insensitive design to communication-based system-level design," *Proc. of the IEEE*, vol. 103, no. 11, pp. 2133–2151, Nov. 2015.

[32] L. P. Carloni, E. G. Cota, G. D. Guglielmo, D. Giri, J. Kwon, P. Mantovani, L. Piccolboni, and M. Petracca, "Teaching heterogeneous computing with system-level design methods," in *Workshop on Computer Architecture Education*, Jun. 2019.

[33] P. Mantovani, D. Giri, G. D. G. L. Piccolboni, J. Zuckerman, E. G. Cota, M. Petracca, C. Pilato, and L. P. Carloni, "Agile SoC development with Open ESP," in *Proc. of the Intl. Conf. on Computer-Aided Design (ICCAD)*, Nov. 2020.

[34] Cobham Gaisler, "Leon3," www.gaisler.com/index.php/products/processors/leon3.

[35] Ariane, www.github.com/pulp-platform/ariane.

[36] E. G. Cota, P. Mantovani, G. Di Guglielmo, and L. P. Carloni, "An analysis of accelerator coupling in heterogeneous architectures," in *Proc. of the Design Automation Conf. (DAC)*, Jun. 2015, pp. 202:1–202:6.

[37] P. Mantovani, G. D. Guglielmo, and L. P. Carloni, "High-level synthesis of accelerators in embedded scalable platforms," in *Proc. of the Asia and South Pacific Design Automation Conf.*, Jan. 2016, pp. 204–211.

[38] D. Giri, K.-L. Chiu, G. D. Guglielmo, P. Mantovani, and L. P. Carloni, "ESP4ML: platform-based design of systems-on-chip for embedded machine learning," in *Conf. on Design, Automation and Test in Europe*, Mar. 2020, pp. 1049–1054.

[39] Y. Yoon, N. Concer, and L. P. Carloni, "Virtual channels and multiple physical networks: Two alternatives to improve NoC performance," *IEEE Trans. on Computer-Aided Design of Integrated Circuits and Systems*, vol. 32, no. 12, pp. 1906–1919, Dec. 2013.

[40] L. P. Carloni, K. L. McMillan, and A. L. Sangiovanni-Vincentelli, "Theory of latency-insensitive design," *IEEE Trans. on Computer-Aided Design of Integrated Circuits and Systems*, vol. 20, no. 9, pp. 1059–1076, Sep. 2001.

[41] P. Mantovani, E. G. Cota, C. Pilato, G. Di Guglielmo, and L. P. Carloni, "Handling large data sets for high-performance embedded applications in heterogeneous systems-on-chip," in *Intl. Conf. on Compilers, Architecture and Synthesis for Embedded Systems*, Oct. 2016, pp. 1–10.

[42] D. Giri, P. Mantovani, and L. P. Carloni, "Accelerators and coherence: An SoC perspective," *IEEE Micro*, vol. 38, no. 6, pp. 36–45, Nov-Dec 2018.

[43] P. Mantovani, E. G. Cota, K. Tien, C. Pilato, G. Di Guglielmo, K. Shepard, and L. P. Carloni, "An FPGA-based infrastructure for fine-grained DVFS analysis in high-performance embedded systems," in *Proc. of the Design Automation Conf. (DAC)*, Jun. 2016, pp. 157:1–157:6.

[44] D. Giri, K.-L. Chiu, G. Eichler, P. Mantovani, N. Chandramoorth, and L. P. Carloni, "Ariane + NVDLA: seamless third-party IP integration with ESP," in *Workshop on Computer Architecture Research with RISC-V (CARRV)*, May 2020.

[45] H.-Y. Liu, M. Petracca, and L. P. Carloni, "Compositional system-level design exploration with planning of high-level synthesis," in *Conf. on Design, Automation and Test in Europe*, Mar. 2012, pp. 641–646.

[46] C. Pilato, P. Mantovani, G. Di Guglielmo, and L. P. Carloni, "System-level optimization of accelerator local memory for heterogeneous systems-on-chip," *IEEE Trans. on Computer-Aided Design of Integrated Circuits and Systems*, vol. 36, no. 3, pp. 435–448, Mar. 2017.

[47] L. Piccolboni, P. Mantovani, G. D. Guglielmo, and L. P. Carloni, "COSMOS: Coordination of high-level synthesis and memory optimization for hardware accelerators," *ACM Trans. on Embedded Computing Systems*, vol. 16, no. 5s, pp. 150:1–150:22, Sep. 2017.

[48] hls4ml, https://fastmachinelearning.org/hls4ml.

[49] D. Giri, P. Mantovani, and L. P. Carloni, "NoC-based support of heterogeneous cache-coherence models for accelerators," in *Proc. of the Intl. Symp. on Networks-on-Chip (NOCS)*, Oct. 2018, pp. 1:1–1:8.