2022 IEEE International Symposium on Circuits and Systems (ISCAS) | 978-1-6654-8485-5/22/$31.00 ©2022 IEEE | DOI: 10.1109/ISCAS48785.2022.9937602

Enabling Software-Defined RF Convergence with a Novel Coarse-Scale Heterogeneous Processor

D. W. Bliss[1,2], T. Ajayi[1,3], A. Akoglu[5], I. Aliyev[5], T. Basaklar[4], L. Belayneh[3], D. Blaauw[3], J. Brunhaver[1,2], C. Chakrabarti[1,2], L. Chang[1,2], K.-Y. Chen[3], M.-H. Chen[3], X. Chen[1,2], A. R. Chiriyath[1,2], A. Daftardar[3], R. Dreslinski[3], A. Dutta[1,2], A. J. Farcas[6], Y. Fu[1,2], A. Goksoy[4], X. He[3], Md. S. Hassan[5], A. Herschfelt[1,2], J. Holtom[1,2], H.-S. Kim[3], A. N. Krishnakumar[4], Y. Li[1,2], O. Ma[1,2], J. Mack[5], S. Mallik[3], S. K. Mandal[4], R. Marculescu[6], B. McCall[1,2], T. Mudge[3], U. Y. Ogras[4], V. Pandey[4], S. Siddiqui[1,2], Y.-H. Sun[3], A. Venkataramani[1,2], X. Wei[3], B. R. Willis[1,2], H. Yu[1,2], Y. Yue[3]

***Abstract*—RF system development is traditionally constrained by a restrictive trade-off between power efficiency and programmatic flexibility. We outline a path towards achieving both, thereby enabling a range of new system concepts that better utilize limited resources. As an example, for many future applications, we consider RF convergence – reusing the same spectrum and waveforms to achieve multiple distributed system functions and goals, simultaneously. To enable this next step in processing, we develop a novel framework that includes both software and the system-on-chip (SoC) design.**

***Index Terms*—Communications; wireless; radar; positioning, navigation, and timing; processing; and computers.**

I. Introduction

For a variety of good engineering reasons, most of the use of the spectrum has historically been relatively rigid. Communications; radar; positioning, navigation, and timing; and other systems have been carefully spectrally isolated, often with inflexible standards. To efficiently implement these functions historically required rigid implementations. Because of advances in flexible frontend technologies [1]–[3] and new flexible, efficient computational capabilities – the focus of this paper – we enable novel, flexible, efficient use of the spectrum, such as radio frequency (RF) convergence. We use the term *RF* to indicate a broad range of frequencies from low to terahertz. The use of *convergence* indicates the implementation of multiple simultaneous functions through intelligent spectrum sharing [4], [5].

Many modern platforms for these capabilities – whether handheld, IoT device, small UAS, or even cubesat – have significant limitations on power consumption and heat dissipation. The maximum power consumption can be as low as 100 mW to a few Watts. We show that many interesting applications require somewhere between a fraction of a TOp/s to several. Thus, we need processor efficiency on the order 1 TOp/s/W or more, similar to full-custom, application-specific integrated circuits (ASICs). To enable complicated, flexible, multiple function RF convergence systems, we require these implementations to be easily programmable and continuously reconfigurable, similar the scalar processors in our laptop computer, which have orders of magnitude poorer efficiency than ASICs, as we depict in Figure 1.

To address the difficulties in programming such a processor, we developed an example coarse-scale heterogeneous system-

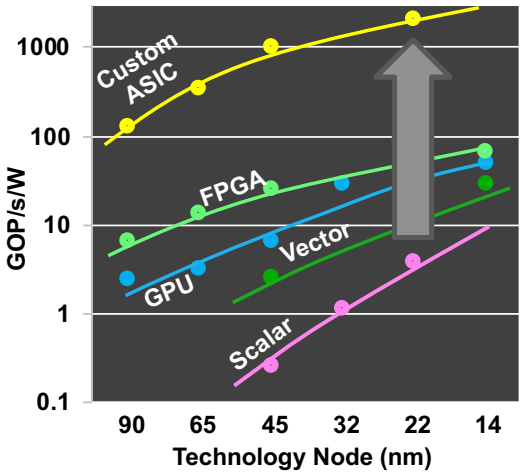

Fig. 1. Comparison of processing power efficiency for RF signal processing applications as a function of processor architecture and node size. To some extent, node size is a surrogate for time, as integrated circuit feature size continues to shrink. We compare examples of full-custom, application-specific integrated circuits (ASICs), field-programmable gate arrays (FPGAs), graphical processing units (GPUs), vector processors, and scalar general processors.

[1] Center for Wireless Information Systems and Computational Architectures (WISCA), Arizona State University, Tempe.
[2] School of Electrical, Computer, and Energy Engineering, Arizona State University, Tempe.
[3] Department of Electrical and Computer Engineering, University of Michigan, Ann Arbor.
[4] Department of Electrical and Computer Engineering, University of Wisconsin–Madison, Madison.
[5] Electrical and Computer Engineering Department, University of Arizona, Tucson.
[6] Electrical and Computer Engineering Department, University of Texas at Austin, Austin.

This material is based on research sponsored Air Force Research Laboratory (AFRL) and Advanced Research Projects Agency (DARPA) under agreement number FA8650-18-2-7860. The U.S. Government is authorized to reproduce and distribute reprints for Governmental purposes notwithstanding any copyright notation thereon. The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of Air Force Research Laboratory (AFRL) and Defense Advanced Research Projects Agency (DARPA) or the U.S. Government.

on-chip (SoC) and a supporting compile-time and run-time software suite. We denote our computational approach the *Domain-focused Advanced Software-reconfigurable Heterogeneous (DASH) SoC* framework. Our framework includes compile-time and run-time software and a general approach for SoC architectures.

Here it is worth noting that not all operations are the same cost. While one can find very high efficiency for certain types of computations that employ relatively low computational dynamic range – machine learning engines, for example – many important operations for RF systems need relatively high computational dynamic range with complex variables, so we focus on this class of operations.

In Section II, we introduce RF convergence. In Section III, we provide estimates of computational costs for example RF applications. In Section IV, we detail our framework to implement coarse-scale heterogeneous processors, and to address associated challenges.

II. RF Convergence

The concept of RF convergence [4]–[7] covers a range of techniques. The fundamental idea is that RF systems are sufficiently flexible to perform multiple functions simultaneously. We often consider the examples of communications, radar, and PNT, but other applications that employ the RF spectrum can be incorporated as well. We assume that radiated RF waveforms can be used for multiple functions, simultaneously. Similarly, we assume receivers can extract multiple types of information from received signals and potentially disentangle disparate temporally and spectrally overlapping waveforms from multiple sources. Ideally, system resources (spectral occupancy, waveforms, energy, receiver degrees of freedom, etc.) are continuously optimized to achieve the best overall system performance. In general, there is some high-dimensional space of performance metrics [6]–[9]. In this space, there is a manifold of best-case joint performance, as notionally depicted in Figure 2. By varying system parameters, we can move along this manifold as needed to achieve the time-varying goals of the system, which potentially consists of a large number of nodes, each with its own goals and capabilities.

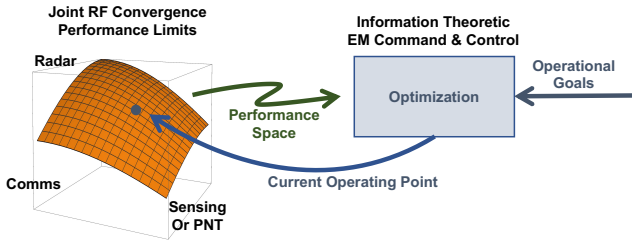

Fig. 2. For a space of joint performance metrics – for example, radar, communications, sensing – there exists a manifold of best performance for a given set of spectral, hardware, and processing resources. As the distributed system's needs vary, system parameters – such as waveform, power, and processing – are modified to best achieve overall multiple-function system goals.

III. Computations

As a motivation for advanced processors, we consider the computational complexity of a subset of tasks useful for RF convergence applications. Specifically, we evaluate computational complexity for communications and radar processing examples.

As an example radar application, we consider correlation-based pulse-Doppler processing. For n_R range bins and n_D Doppler bins, the complex image $\mathbf{M} \in \mathbb{C}^{n_D \times n_R}$ is given by

$$\mathbf{M} = \mathbf{F}\,\mathbf{Z}\,\mathbf{S}^{\dagger}, \tag{1}$$

where $\cdot^{\dagger}$ indicates Hermitian conjugate, $\mathbf{F} \in \mathbb{C}^{n_D \times n_R}$ is the discrete Fourier transform which is typically implemented by fast Fourier transform (FFT), $\mathbf{Z}$ is the n_D number of pulses (same as Doppler bins) by n_s sample data matrix $\mathbf{Z} \in \mathbb{C}^{n_D \times n_s}$, and $\mathbf{S}$ is the n_R number of range bin by n_s number of sample reference matrix $\mathbf{S} \in \mathbb{C}^{n_R \times n_s}$. The computational complexity is proportional to $(n_D \log n_D)\, n_R$ and $n_D\, n_s\, n_R$ per $n_D\, n_S$ collected processing interval (CPI). This translates to order $(n_D \log n_D)\, n_R/(n_D\, n_s) = \log(n_D)\, n_R/n_s$ and $n_D\, n_s\, n_R/(n_D\, n_s) = n_R$ computations per sample. For reasonable parameters, the second term dominates, so n_R per sample. To approximate a modern automotive application, let's assume 400 MHz of bandwidth with around 250 m of range, so $n_R \approx 650$, which corresponds to a good fraction of a TOp/s.

To implement multiple-antenna interference mitigation in dispersive channels, we use space-time receive beamforming [10], [11]. The construction of the beamformer is given by

$$\hat{\underline{\mathbf{s}}} = \mathbf{w}^{\dagger}\, \tilde{\mathbf{Z}}_{data}, \tag{2}$$

$$\mathbf{w} = (\tilde{\mathbf{Z}}_t\, \tilde{\mathbf{Z}}_t^{\dagger})^{-1}\, \tilde{\mathbf{Z}}_t\, \mathbf{s}^{\dagger}, \tag{3}$$

$$\tilde{\mathbf{Z}}_t = \begin{pmatrix} \mathbf{Z}_{t,\delta_1} \\ \mathbf{Z}_{t,\delta_2} \\ \vdots \\ \mathbf{Z}_{t,\delta_{n_d}} \end{pmatrix}, \tag{4}$$

where the number of receive antenna n_r by number of samples n_s space-time data matrix that contains training data is given by $\tilde{\mathbf{Z}}_t \in \mathbb{C}^{(n_r \cdot n_d) \times n_s}$. The number of tap delays incorporated into the data matrix is given by n_d. We use $\mathbf{Z}_{t,\delta_m} \in \mathbb{C}^{n_r \times n_s}$ to indicate the data matrix shifted in time by δ_m.

The computational complexity is proportional to $(n_r\, n_d)^2\, n_s$ for space-time covariance estimation, $(n_r\, n_d)^3$ for matrix inversion, $(n_r\, n_d)\, n_s$ for cross-covariance estimation, and $(n_r\, n_d)\, n_{data}$ for filter application per communications data frame or super frame. Of these terms, the space-time covariance estimation is typically the most expensive. It would not be surprising to require beamforming updates every few milliseconds. For a system with n_r = 8 antennas, a delay range of n_d = 15 taps, and n_s = 10 $n_r\, n_d$, we have order of magnitude complexity of 10 $(n_r\, n_d)^3$ – on the order of 100 GOp/s.

A useful tool for RF convergence is temporal interference mitigation, which is effectively a projection operation. Here, we project onto a basis that is orthogonal to a signal sequence

and some delayed versions of that sequence. As an example, consider an n_r multiple antenna receiver providing the n_s sample data matrix $\mathbf{Z} \in \mathbb{C}^{n_r \times n_s}$. To remove known or estimated sequence $\underline{\mathbf{s}} \in \mathbb{C}^{1 \times n_s}$ and delayed versions of this sequence ($\underline{\mathbf{s}}_{\delta_m}$), we implement

$$\mathbf{Z}' = \mathbf{Z}\,\mathbf{P}_{\mathbf{S}}^{\perp} \tag{5}$$

$$= \mathbf{Z} - (\mathbf{Z}\,\mathbf{S}^{\dagger})\,(\mathbf{S}\,\mathbf{S}^{\dagger})^{-1}\,\mathbf{S} \tag{6}$$

$$\mathbf{S} = \begin{pmatrix} \underline{\mathbf{s}}_{\delta_1} \\ \underline{\mathbf{s}}_{\delta_2} \\ \vdots \\ \underline{\mathbf{s}}_{\delta_N} \end{pmatrix}, \tag{7}$$

where $\mathbf{Z}' \in \mathbb{C}^{n_r \times n_s}$ is the data matrix with $\underline{\mathbf{s}}$ removed. The number of complex operations for this task are proportional to n_d^3, $n_r\,n_s\,n_d$, and $n_d^2\,n_s$ per n_s samples. Because n_s is typically large compared to either n_r or n_d, the largest term is either $n_d^2\,n_s$ or $n_r\,n_s\,n_d$. The cost per sample is proportional to n_d^2 or $n_r\,n_d$. An 8-antenna system with 8-delay mitigation would require 100s of operations per sample. If the sample rate is order 100 MHz, the task would cost multiple 10s of GOp/s.

By considering the Tanner graph representation (as we depict in a toy example in Figure 3) of a low-density parity check (LDPC) code, we observe that number of operations per decoding iteration is proportional to at least the sum of paths connecting to each data vertex times the number of data vertices plus the sum of paths connecting to each parity check vertex times the number of parity check vertices. We consider an LDPC code with density α, with k information bits and code length n. We assume belief propagation is employed to decode with n_{iter} iterations. If the density nonzero entries in the parity check matrix is α, then the cost at each of the $\alpha\,n$ variable nodes is roughly proportional to $[\alpha\,(n-k)]$. The cost at each of the $\alpha\,(n-k)$ parity check (factor) nodes is roughly proportional to $\alpha\,n$. The computational cost of decoding includes terms approximately proportional to $[\alpha\,n]\,\alpha\,(n-k)\,n_{iter}$ and $[\alpha\,(n-k)]\,\alpha\,n\,n_{iter}$, with different coefficients, per n received likelihoods. By using typical parameters, it is easy to expect order hundreds of operations per information bit, so 100 Mb/s would correspond to order hundred GOp/s.

IV. Heterogeneous Processor Framework

We want a processor that can simultaneously perform the above tasks and many more. We need the implementation to be flexible, so resources can be adaptively used as application loads asynchronously wax and wane. We want to make these tasks relatively easy to program, and be able to add new application tasks without having to modify old tasks. All these tasks need to be performed with much greater efficiency than general processors would allow.

These requirements drive us to domain-specific coarse-scale heterogeneous SoC solutions. As we indicate in the notional task vs computational cost plot in Figure 4, there are typically a small number of tasks that consume most of the computational resources, while many tasks consume relatively little. Naturally, we focus our engineering efforts on accelerating those computational kernels that represent the largest computational costs.

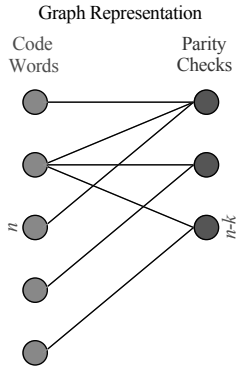

Fig. 3. Notional depiction of bipartite Tanner graph. Decode an LDPC code by updating likelihoods back and forth across sides of the graph.

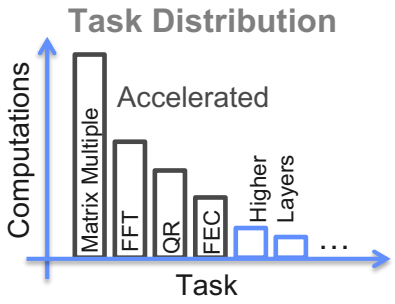

Fig. 4. Notional distribution of key computational kernel costs for an interference-mitigating multiple-antenna receiver for an orthogonal frequency division multiplexing (OFDM) waveform. The notation *FFT* indicates the fast Fourier transform. *QR* indicates a matrix decomposition that is often used as an intermediate step for matrix inversion. *FEC* indicates forward error correction.

DASH Framework – As we depict in Figure 5, we developed a framework that enables the efficient implementation of RF processing applications. This spans a wide range of timelines, from SoC design, to code analysis, to compile-time, to run-time, to the intelligent scheduler (IS) adaptively making resource management decisions within a few nanoseconds to enable multiple applications to operate simultaneously and asynchronously.

Ontological Analysis – We analyze dynamic program traces to determine the flow of the program. We developed analysis tools that discover computational kernels, label these kernels, and produce parallel task graphs [12]. This information can be used to optimize SoC layout and to provide task graph information to the run-time software and intelligent scheduler.

Compile-Time and Run-Time Software – In our compile-time environment, we leverage our knowledge of accelerated kernels and produce "fat" binaries that incorporate multiple implementation approaches within the executable [13]–[15].

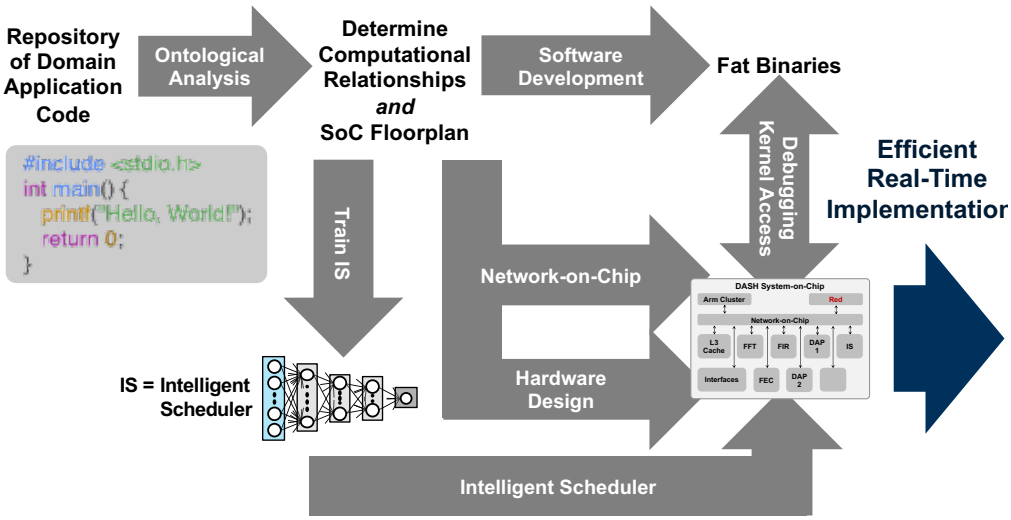

Fig. 5. The domain-focused advanced software-reconfigurable heterogeneous (DASH) framework incorporates a compile-time and run-time software suite along with an approach for designing advanced heterogeneous SoCs. The software informs and enables the intelligent scheduler (IS) to adaptively manage SoC resources.

For example, the FFT can be executed on the ARM processor, the DAP, and most efficiently on the FFT accelerator. In our run-time environment, the run-time software works with the intelligent scheduler to select the best version of the binary to execute.

IS – To enable real-time adaptive SoC resource management for scenarios that have multiple simultaneous applications running, we developed an intelligent scheduling (IS) capability. Because resource needs can vary dramatically and quickly, the IS is a necessary technology. It would be impossible to hand-schedule tasks in environments in which new applications are being added and removed in real time. Our IS employs a combination of approaches to reduce scheduling latency [16]. In static scenarios, the IS keeps the current resource allocation strategy. When there is a status change, we employ an imitation learning (IL) approach that was trained on high-performance strategies. Our implementation of IL allows us to execute resource allocation decisions more rapidly than the reference high-performance strategies [17]–[20].

DASH SoC – We developed an enabling SoC architecture. As seen in Figure 6, we incorporated a cluster of ARM processors for general processing, a fast on-chip network, and a number of on-chip accelerators. We have dedicated accelerators for FFTs and finite-impulse-response (FIR) filters. We have developed a flexible forward error correction (FEC) decoder that efficiently implements a range of codes including turbo, low-density parity check, and polar codes. Prior to chip fabrication, we cannot know all required basic functions or kernels of future applications, so we developed a flexible high-performance systolic array processor, denoted domain-adaptive processor (DAP).

Flexible FEC – Our flexible FEC accelerator provides the tools to efficiently implement a range of error-correcting codes, including turbo, LDPC, and polar. In our preliminary

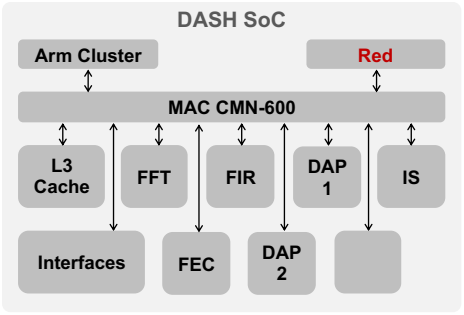

Fig. 6. The DASH SoC connects a cluster of ARM processors, accelerators, and interfaces via a high-speed network on-chip. Accelerators include FFT, FIR filters, DAPs, and flexible FECs.

efforts, we have shown that the power efficiency for FECs that can be implemented on the accelerator are as efficient – or at least within a reasonable factor – of full-custom equivalents.

DAP – Our DAP is a flexible high-performance systolic array processor that enables the implementation of a range of kernels. Multiple kernels can be executed simultaneously across the DAP. Furthermore, the functionality of the DAP can be nearly instantaneously changed as tasks change. For a range of kernels, we have empirically shown that the DAP is as efficient – or at least within a reasonable factor – of full-custom equivalents

V. Summary

We developed an approach that enables efficient processing by employing coarse-scale heterogeneous processors. Unlike some prior related efforts, we provide the framework to enable flexibility and relatively easy programming. From a system engineer's perspective, the DASH framework opens numerous new opportunities for flexibly and dynamically matching computations to the environmental and system's needs.

REFERENCES

[1] J. Ryynanen, K. Kivekas, J. Jussila, A. Parssinen, and K. A. Halonen, "A dual-band rf front-end for wcdma and gsm applications," *IEEE Journal of Solid-State Circuits*, vol. 36, no. 8, pp. 1198–1204, 2001.

[2] M. Feng, S.-C. Shen, D. C. Caruth, and J.-J. Huang, "Device technologies for rf front-end circuits in next-generation wireless communications," *Proceedings of the IEEE*, vol. 92, no. 2, pp. 354–375, 2004.

[3] A. Kalis, A. G. Kanatas, and C. B. Papadias, "A novel approach to mimo transmission using a single rf front end," *IEEE Journal on Selected Areas in Communications*, vol. 26, no. 6, pp. 972–980, 2008.

[4] B. Paul, A. R. Chiriyath, and D. W. Bliss, "Survey of RF communications and sensing convergence research," *IEEE Access*, vol. 5, pp. 252–270, 2016.

[5] A. R. Chiriyath, B. Paul, and D. W. Bliss, "Radar-communications convergence: Coexistence, cooperation, and co-design," *IEEE Transactions on Cognitive Communications and Networking*, vol. 3, no. 1, pp. 1–12, 2017.

[6] A. Herschfelt, A. R. Chiriyath, S. Srinivas, and D. W. Bliss, "An introduction to spectral convergence: Challenges and paths to solutions," in *2021 1st IEEE International Online Symposium on Joint Communications & Sensing (JC&S)*. IEEE, 2021, pp. 1–6.

[7] A. Herschfelt, "Vehicular rf convergence: Simultaneous radar, communications, and pnt for urban air mobility and automotive applications," in *2012 IEEE Radar Conference (RadarConf20)*. IEEE, 2020.

[8] O. Ma, A. Herschfelt, H. Yu, S. Wu, S. Srinivas, Y. Li, H. Lee, C. Chakrabarti, and D. W. Bliss, "Communications and high-precision positioning (chp2): Secure traffic and resource management using reinforcement learning," in *2020 AIAA/IEEE 39th Digital Avionics Systems Conference (DASC)*. IEEE, 2020, pp. 1–6.

[9] O. Ma, A. R. Chiriyath, A. Herschfelt, and D. W. Bliss, "Cooperative radar and communications coexistence using reinforcement learning," in *2018 52nd Asilomar Conference on Signals, Systems, and Computers*. IEEE, 2018, pp. 947–951.

[10] D. W. Bliss and S. Govindasamy, *Adaptive Wireless Communications: MIMO Channels and Networks*. New York, New York: Cambridge University Press, 2013.

[11] J. R. Guerci, *Space-Time Adaptive Processing for Radar*. Norwood, Massachusetts: Artech House, 2003.

[12] R. Uhrie, D. W. Bliss, C. Chakrabarti, U. Y. Ogras, and J. Brunhaver, "Machine understanding of domain computation for Domain-Specific System-on-Chips (DSSoC)," in *Open Architecture/Open Business Model Net-Centric Systems and Defense Transformation 2019*, R. Suresh, Ed., vol. 11015, International Society for Optics and Photonics. SPIE, 2019, pp. 180 – 187. [Online]. Available: https://doi.org/10.1117/12.2519264

[13] J. Mack, N. Kumbhare, A. NK, U. Y. Ogras, and A. Akoglu, "User-space emulation framework for domain-specific soc design," in *2020 IEEE International Parallel and Distributed Processing Symposium Workshops (IPDPSW)*, 2020, pp. 44–53.

[14] J. Mack, N. Kumbhare, R. Uhrie, A. Krishnakumar, S. Arda, L. Chang, A. Amarnath, R. Dreslinski, C. Chakrabarti, U. Ogras, and A. Akoglu, "Automating Programming and Development of Heterogeneous SoCs with LLVM Tools," Talk at LLVM Devroom, 2020 Free and Open-source Software Developers' European Meeting (FOSDEM), Brussels, Feb 2020.

[15] J. Mack, N. Kumbhare, S. Hassan, M. Castro, S. Arda, L. Chang, J. Brunhaver, C. Chakrabarti, U. Ogras, and A. Akoglu, "Runtime Strategies and Task Scheduling of Software-Defined Radio on Heterogeneous Hardware," Talk at LLVM Devroom, 2021 Free and Open-source Software Developers' European Meeting (FOSDEM), Brussels, Feb 2021.

[16] A. A. Goksoy, A. Krishnakumar, M. S. Hassan, A. J. Farcas, A. Akoglu, R. Marculescu, and U. Y. Ogras, "DAS: Dynamic adaptive scheduling for energy-efficient heterogeneous SoCs," *IEEE Embedded Systems Letters*, 2021.

[17] A. Krishnakumar, S. E. Arda, A. A. Goksoy, S. K. Mandal, U. Y. Ogras, A. L. Sartor, and R. Marculescu, "Runtime task scheduling using imitation learning for heterogeneous many-core systems," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 39, no. 11, pp. 4064–4077, 2020.

[18] A. L. Sartor, A. Krishnakumar, S. E. Arda, U. Y. Ogras, and R. Marculescu, "Hilite: Hierarchical and lightweight imitation learning for power management of embedded socs," *IEEE Computer Architecture Letters*, no. 01, pp. 63–67, jan 2020.

[19] X. Chen, U. Ogras, and C. Chakrabarti, "Probabilistic risk-aware scheduling with deadline constraint for heterogeneous socs," *ACM Trans. Embed. Comput. Syst.*, vol. 21, no. 2, feb 2022. [Online]. Available: https://doi.org/10.1145/3489409

[20] S. E. Arda, A. Krishnakumar, A. A. Goksoy, N. Kumbhare, J. Mack, A. L. Sartor, A. Akoglu, R. Marculescu, and U. Y. Ogras, "DS3: A system-level domain-specific system-on-chip simulation framework," *IEEE Transactions on Computers*, vol. 69, no. 8, pp. 1248–1262, 2020.