Secure and Resilient SoCs for Autonomous Vehicles

Pradip Bose, Augusto Vega – *IBM T. J. Watson Research Center*
Sarita Adve, Vikram Adve, Sasa Misailovic – *University of Illinois at Urbana-Champaign (UIUC)*
Luca Carloni, Ken Shepard – *Columbia University*
David Brooks, Vijay Janapa Reddi, Gu-Yeon Wei – *Harvard University*
Contact emails: {pbose, ajvega}@us.ibm.com

***Abstract*—** Embedded systems-on-chip (SoCs) that are targeted to power autonomous vehicles of the future must meet stringent power, real-time performance, reliability, security and safety criteria in order to qualify for deployment in the field. General purpose processor cores, supported by carefully selected accelerators are needed to meet the power-performance requirements. Such heterogeneity at the hardware processing level immediately points to challenges in user-level programmability. In this paper, we present our solution strategy, and we present an updated results summary achieved after the first two phases of DARPA's DSSoC (Domain-Specific System-on-Chip) program.

Keywords—intelligent vehicles; system-on-chip; accelerators; agile design; ease of programmability; efficiency and resilience.

I. Introduction

A widely anticipated model of future vehicular transportation is that of artificially intelligent, connected, autonomous vehicles. In defense applications, the scope includes military vehicle convoys on the ground, drone swarms in the air, as well as swarms of miniature robot submarines. In the commercial sector, the dominant market potential at this point is that of wirelessly connected, cloud-backed autonomous and semi-autonomous land vehicles.

In this paper, we are concerned about the task of designing efficient and resilient (secure) embedded SoCs in support of the above-mentioned *vehicular swarm* problem space [1]. Heterogeneous, multi-core architecture is well-established in the art as an energy-efficient execution paradigm. The IBM-led project [8] under DARPA's DSSoC program is called: "EPOCHS: Efficient Programmability of Cognitive Heterogeneous Systems." In this project, our chosen application domain is that represented by smart, connected autonomous vehicles. In this paper, we touch upon key system resilience issues encountered in the targeted problem domain, and how they are being addressed in EPOCHS.

II. EPOCHS Overview

The EPOCHS agile design flow consists of a series of procedural steps that start with the EPOCHS Reference Application (ERA) all the way down to the final SoC implementation (Figure 1). The ERA code, developed by IBM, is an open-source software application representative of the connected, smart car application domain referred to above.

This research was developed with funding from the Defense Advanced Research Projects Agency (DARPA). The views, opinions and/or other findings expressed are those of the authors and should not be interpreted as representing the official views or policies of the Department of Defense or the U.S. Government. Distribution Statement "A": Approved for Public Release, Distribution Unlimited.

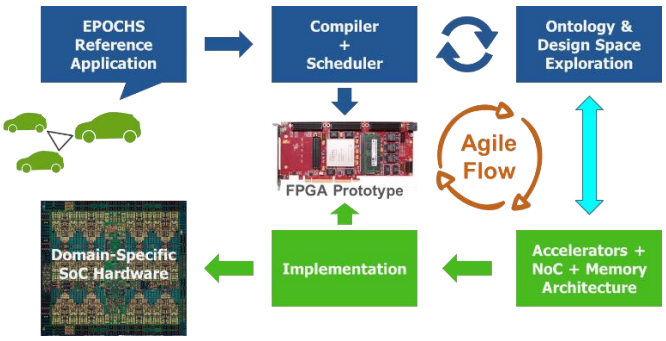

Fig. 1. EPOCHS agile flow methodology

In this *design and software enablement* flow, the compiler and the ontology exploration software interact iteratively to identify the most feasible candidates for offloading as synthesizable hardware accelerators. The recommended PE mix for the SoC is then passed on to the hardware design methodology. The latter invokes the embedded systems platform (ESP) [2] facility for rapid integration of the heterogeneous IP blocks into an FPGA-based emulation embodiment of the target SoC. The Spandex coherence interface [7] is a novel feature of the architecture that facilitates data movement reduction. The elements of this software-driven design flow were described at GomacTech-2020 [8]. In this particular paper, we focus on the "efficient and resilient autonomy" aspect of the overall solution space targeted by EPOCHS. In the next section, we first examine the desired *efficient resilience* features of our reference, domain-representative software application and then address the concomitant design requirements at the SoC level.

III. Efficient Resilience Challenges

An SoC targeted for the autonomous vehicular swarm domain must meet stringent requirements of resilience, while meeting a tight power budget as well as real-time performance requirements. The term "resilience" includes aspects of in-field reliability as well as security. Meeting both criteria feeds into the system safety requirement for the overall solution.

A. EPOCHS Reference Application

The EPOCHS Reference Application (ERA) [3] is an evolving open-source offering from IBM that constitutes the main driver of the entire project. Each autonomous vehicle in ERA uses its on-board sensors to generate a local occupancy map, which it also communicates to the other nearby vehicles using dedicated short-range communications (DSRC). A given

vehicle in the swarm merges locally generated and remotely received maps into one that expands the scope of vision and enhances the object detection/recognition accuracy. ERA incorporates an open-source DSRC IEEE 802.11p transceiver implemented using GNU Radio [4]. It is useful to examine ERA-like application software in the overall envisaged context of cloud-backed "swarm-AI" systems as depicted in Figure 2. The component-level resilience at the individual SoC level must be assessed in a realistic manner, in the context of the entire system. Also, failure mitigation solutions at the SoC level must work in unison with the overall cloud-governed system resilience strategy.

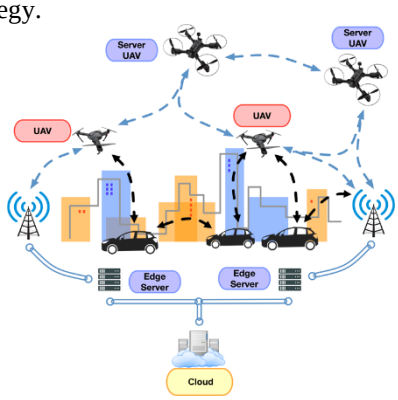

Fig. 2. Cloud-backed "swarm-AI": the larger context

B. *Mini-ERA and EPOCHS-0 SoC*

Let us consider an abstraction of the ERA workload via the Mini-ERA formulation [3] – as depicted in Figure 3. In this simplified view, we have only two sensors per vehicle: one dedicated to detection of obstructions with an estimate of distance; and the other involved in processing image data to classify the obstruction as one chosen from a modeled traffic dataset [18] that includes cars, buses, bicycles, pedestrians etc. There is a third input to the vehicular control state machine: namely, guidance messages received from nearby cars or infrastructure (see Figure 2). These encoded messages are decoded on-board using the Viterbi decoder module.

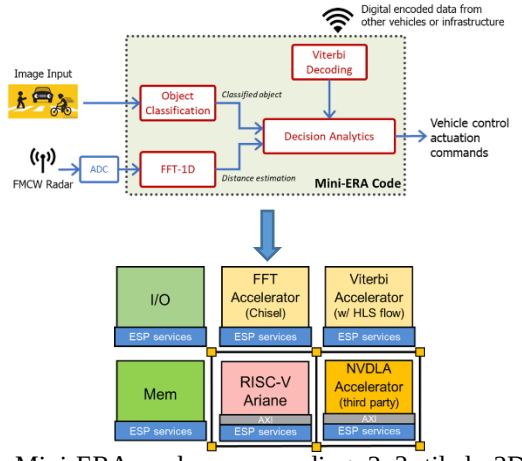

Fig. 3. Mini-ERA and corresponding 2x3 tiled, 2D-mesh connected SoC implemented using the ESP [2] FPGA platform.

The tiled architecture consisting of a RISC-V Ariane core [11], supported by accelerators, I/O and memory, as implemented in ESP-hosted FPGA apparatus is also sketched in Figure 3. The FFT-1D accelerator is used in the task of estimating the distance of an object sensed by the assumed FMCW (Frequency-Modulated Continuous Wave) radar sensor. The image classification task is executed via a CNN (convolutional neural network) implemented to run on an NVDLA engine [12]. The Viterbi accelerator is used to assist in decoding the Viterbi-encoded messages received from nearby vehicles or infrastructure. The compiler infrastructure [5, 6] along with our smart task scheduler [9] are currently operational on the above "minimal configuration" EPOCHS-0 FPGA embodiment. In fact, in preparation for the actual ASIC tape-out, the number of processing elements (PEs) was expanded to allow a larger number of concurrent Mini-ERA component tasks (or sub-tasks) to run on the SoC. In the updated FPGA, we now have four RISC-V Ariane cores, three NVDLA engines, three FFT accelerators and one Viterbi decoder PE. In addition, there are four memory tiles and one IO engine.

The corresponding EPOCHS-0 ASIC chip was taped out (in mid-October 2020) for fabrication in a 14nm CMOS bulk FinFET technology made available by DARPA for the DSSoC program. Figure 4 shows the layout and post-synthesis area breakdown estimate. Note that one of the FFT tiles has been augmented to have support for local voltage regulation via on-board (off-chip) integrated voltage regulators and power management control loop. This is designed to be an initial test experiment leading to a comprehensive hardware-assisted, distributed power management architecture (targeted for EPOCHS-1) orchestrated via software using the smart task scheduler facility [13].

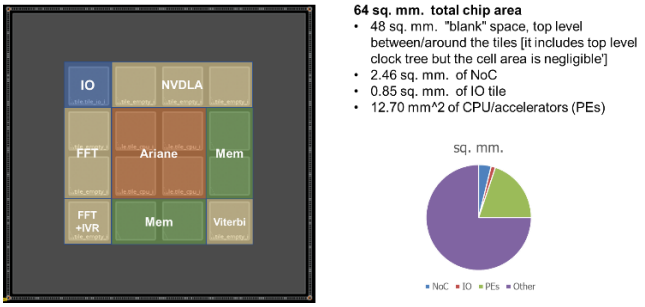

Fig. 4. EPOCHS-0 SoC taped out in 14nm technology; layout and area breakdown estimates.

The real-time constrained performance and energy efficiency metrics achievable at the system level will constitute the ultimate index of success in this project – as measured against the stipulated DSSoC program metrics. A compact summary of pre-ASIC measurements of Mini-ERA executions on the FPGA apparatus (see Figure 3) is provided in Table 1 below. We can see that the standalone speedups of the individual accelerator PEs, as measured over baseline RISC-V Ariane executions range from 12.8X through 38.9X, depending on the particular PE and the input data characteristics. The best-case average PE utilization is measured at 83.13%, for a

particular smart scheduler policy setting (see details in our companion paper [13]) in combination with optimal Viterbi message lengths and FFT sample sizes.

C. *Fault and Threat Models*

As we define the next generation ERA workload for the follow-on EPOCHS-1 chip, we are considering the following fault and threat models for exemplary demonstration of resilience features in the final cloud-to-edge solution:

- We assume that the ultra-efficient, low-voltage SRAM storage on-chip will be unreliable, with a pre-characterized error rate probability distribution.
- We assume that information from certain members of the vehicular swarm (a minority group) could be unreliable or tainted.
- We assume that the distributed on-chip power control mechanism could be subject to inadvertent or malicious "energy attacks" as anticipated in an earlier invited talk from the IBM authors [10].

To achieve low voltage efficiency (in EPOCHS-1) at targeted system resilience, we plan to experiment with robust DNNs that are pre-trained in error-prone environments, characterized by measured SRAM bit-error probabilities [28]. In incorporating secure resilience against adversarial attacks at the swarm connectivity level, we are looking into classical literature on adversarial robustness [30]. This is new work in progress that is expected to mature during Phase-3 of the DSSoC program.

Table 1. Measurement Summary: EPOCHS-0 SoC (pre-ASIC FPGA embodiment)

Accelerator	**Acceleration Speedup (standalone)**	**Best-case (avg.) accelerator (PE) utilization**
FFT (1K samples and 16K samples)	12.8X and 19.1X respectively	83.13 %
Viterbi (four different message lengths)	18.9X through 22.7X, depending on message length	
CNN (MIO dataset [18])	38.9X	

D. *Robust Map Fusion and Inference*

A key focus item in the final version of ERA [3] will be on the local and global map fusion algorithms carried out within the application workload.

Figure 5 depicts the initially targeted functional demonstration apparatus, involving proxies for multiple vehicles, each running ERA in its embedded SoC. The central software navigation controls will be implemented using CARLA [14] which is an open-source simulator for autonomous driving research. CARLA can provide customized sources of sensory inputs: e.g. the camera image stream and radar (or lidar) data stream, as envisaged also for the simplified Mini-ERA (see Figure 3).

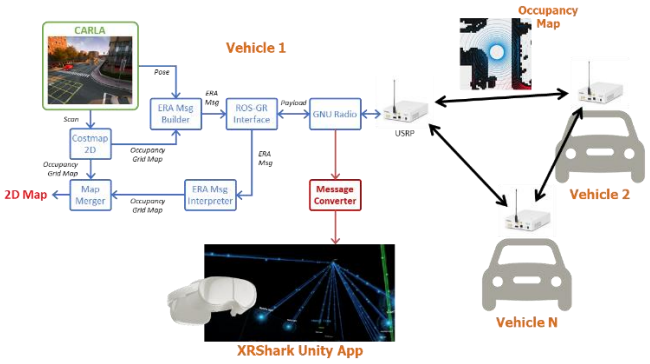

Fig. 5. ERA and targeted physical demonstration apparatus, with visualization of wireless communication traffic

Scanned images (from video camera and radar/lidar sensors) get converted into a 2-dimensional (2D) costmap within each vehicle. This is essentially a 2D grid matrix, with object occupancy indicators (e.g. grid values of '1' to indicate occupancy, '0' for absence), with embedded qualifying attributes that indicate whether the sensed object ahead is a known member within the modeled traffic data set. This initial map itself requires a coordinated intra-vehicular sensor fusion, combining the sensed information derived from multiple sensor types. Each vehicle then uses the generated costmap, embellished perhaps with other positional attributes (relative to adjacent vehicles) to build up a Viterbi-encoded message. The latter is to be broadcast wirelessly via a transceiver module crafted out of the GNURadio code [15]. Within each vehicle, messages received from other vehicles are decoded and interpreted using the ERA message interpreter. In our implementation, this is largely the Viterbi decoder module of ERA, whose output is used to reconstruct the costmap "seen" by an adjacent vehicle. The "Map Merger" module is the ultimate map fusion module that creates the most accurate costmap that incorporates the collaborative perception representing the vehicular swarm. We should mention here the use of the ROS-GR interface module that is needed to serve as the protocol layer between the GNURadio transceiver code and the ERA message builder (or the ERA message interpreter). ROS (Robot Operating System) [16] constitutes a popular set of (open source) software libraries and tools that help one build robotic applications. The initial version of ERA was built around ROS, which has its own built-in test stimuli environment (called Gazebo). We have recently moved over to CARLA, which provides seamless integration with ROS via its ROS-bridge [14]. This facility is utilized in crafting the ROS-GR (GR = GNURadio) interface module indicated in Figure 5.

Another targeted feature in the ERA-driven demonstration of EPOCHS technologies is the visualization of inter-vehicular wireless radio communication via XRShark [17] from University of California at Berkeley. This is a mixed reality network introspection tool for visualization of real-time wireless traffic in cyber-physical systems. As the moving vehicles communicate wirelessly, XRShark, suitably adapted, will allow us to visualize the wireless message traffic between

connected vehicles, to pinpoint and deduce the specific benefit of collaborative perception over non-swarm operation.

A key research challenge that needs to be tackled in realizing the goal of demonstrating robust functionality (Fig. 5) is: how to perform the map fusion steps using novel fault-screening methods; and, how the final inference step in interpreting the fused map is to be made error- and attack-tolerant using novel hardware-software architectural mechanisms. Methods of threat detection and adversarial robustness as practiced in current resilient AI/ML systems literature [30] are being incorporated into ERA in order to meet such requirements in a DoD military context. A few high-level elements of the planned security enhancements of ERA will be divulged in the actual presentation material at the conference. This is work in progress, targeted for completion before the end of 2021.

IV. Software and Programmability Challenges

A key feature of emphasis in the DSSoC program is that of ease of programming from a user perspective. The software elements of the system stack must be carefully organized and orchestrated in order to meet the programmability objective without compromising performance, energy efficiency or system resilience.

In EPOCHS, the foundation of programmability is built around the novel HPVM (heterogeneous parallel virtual machines) compiler [19-20], which was released as open-source software on January 27, 2020. The goal of the HPVM research project is to make heterogeneous systems easier to program. A key aspect of the technical strategy is to develop a uniform parallel program representation for heterogeneous systems that can capture a wide range of popular parallel hardware, including GPUs, vector instruction sets, multicore CPUs, FPGAs, and domain-specific accelerators.

Figure 6 shows the synergistic interplay between the HPVM compiler and the smart scheduler [13], as architected via the scheduler library (SL) software architecture. The compiler generates application binaries (specific to RISC-V as well as to specialized accelerator PEs where applicable). These binaries are annotated with calls to the SL application programming interface (API) – which is a relatively "thin" layer used to expose the main smart scheduler substrate to the user-level program suite (i.e. ERA in our case).

The HPVM compiler runtime scheduler has hooks into the *SL Manager*, with its embedded module dedicated to the set of available scheduling policies. In particular, the various trade-off curves sweeping the metric space of performance, computational (or inference) accuracy and energy consumption are made available to the *Scheduling Policy* module. This is possible because of the novel ApproxTuner [21] facility within the HPVM compiler toolset. Accelerator-specific binaries generated by the compiler are stored in the indicated kernel repository that is resident within the *SL Manager* layer. The *SL Manager* layer, in effect, serves as the interface between the compiler application binary suite and the actual heterogeneous SoC hardware, which is accessible through a set of hardware APIs. The latter constitute the gateway to read hardware execution and utilization data via implemented hardware counters; and, it also allows software to actuate resource (power-performance) control knobs as needed for efficiency and resilient operation.

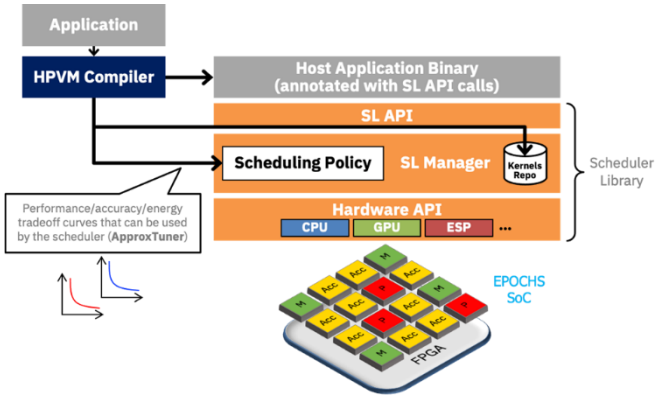

Fig. 6. HPVM and scheduler library (SL) synergistic interplay

The smart scheduler heuristics specific to EPOCHS and our main application domain of connected autonomous vehicles have been separately communicated through a pending publication under review [22]; and, exemplary features are also covered in a companion paper in this conference [13]. Hence, we do not cover the details of the scheduling policies and experimental evaluation thereof in this paper.

Within the software elements of the overall EPOCHS agile design flow (see Figure 1), there is a close synergy between the ontology toolset and the compiler toolset. In other words, the ontology software is driven by the intermediate code representation (IR) generated by the compiler from the application source code. In particular, the unique feature of the HPVM is that the parallelism opportunities for the targeted heterogeneous multi-processor are explicitly captured in the graph-oriented IR. This facilitates analysis that reasons about the minimal set of graph primitives that would "cover" a significant fraction of program execution time. The minimality analysis is pursued in synchrony with graph-node level merge-split transformations in order to maximize coverage while minimizing the number of accelerators. Having this ontology analysis in an iterative design loop with the compiler ensures *a priori* enforcement of end-user programmability criteria.

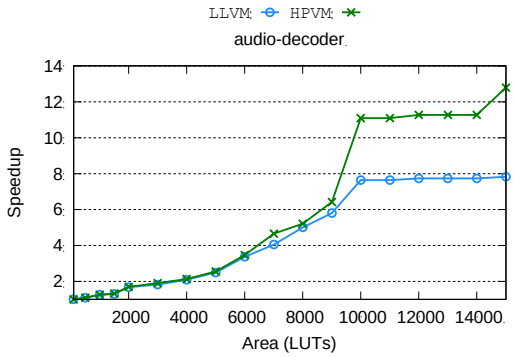

Fig. 7. Use of AcceSeeker/AccelMerger: Speedup estimation on ILLIXR audio-decoder module

The ontology software is collectively referred to as Jasmine. Key components of this toolset are AccelSeeker, AccelMerger and Novia, partially covered through recent and in-progress publications [23-26]. An important feature of the Jasmine methodology is the ability to automatically discover viable code segment candidates for implementation as accelerators in the target heterogeneous SoC.

The AccelSeeker-AccelMerger toolset is able to work with either standard LLVM IR code or HPVM IR code. In the former case, the toolset identifies sequential candidates for acceleration. In the latter case, it is able to identify a balanced combination of both sequential and parallel candidates for acceleration. As shown in Figure 7, working with HPVM IR, the toolset is able to discover significantly more efficient accelerators from an exemplary audio-decoder module chosen from an extended reality software application suite called ILLIXR [27]. In EPOCHS, we are treating the extended reality (XR) domain as one that is adjacent to the autonomous vehicle (AV) space; and, as such, we are interested in understanding how the AV-specific SoC design performs in an adjacent space exemplified by ILLIXR.

The Novia tool [26], on the other hand is focused on discovering *inline* accelerators, that could be directly coupled to the general purpose RISC-V Ariane core. The motivational insight here is that strengthening the general purpose core by adding a small suite of inline hardware accelerators would yield much better speed-up. This expectation is in the context of application codes that are limited by a significant "scalar" component that is not easily amenable to coarse-grain off-load acceleration. Novia engages in a bottoms-up analysis regimen starting with fine-grain code segments at the level of basic block primitives. For the ERA workload, Novia is able to propose an inline accelerator that would result in a 13% speedup over plain RISC-V core execution, while incurring only a 2% area overhead to the core design. For the popular SpecFP application workload, Novia discovers a set of three inline accelerators that provides 14% speedup with 5% core area overhead.

At present, the EPOCHS-1 SoC architecture definition team is examining the feasibility of integrating non-conventional accelerators (NCA) – inline, off-load or both – as discovered (or proposed) by the Jasmine ontology toolset. The decision must balance programmability, verification and design integration complexity against benefits in performance and/or energy efficiency.

V. Technology Transition

The EPOCHS team has made a significant investment of resources in concurrent technology transition activity. There are two broad impact targets: (a) commercial sector; and (b) DoD military sector.

Under (a), we are engaged in trying to transition our agile application-driven SoC design methodology to design teams within IBM as well as to other companies (e.g., in particular, companies involved in the automotive electronics sector). In addition, in partnership with a group of private software and hardware vendors, we are attempting an entrepreneurial venture in terms of developing cloud-to-edge solutions in support of future connected autonomous vehicles.

Under (b) we are engaged in a collaborative deployment of specific EPOCHS capabilities in the military combat vehicle space under the guidance of our project's COR (contracting officer's representative) Mr. James Hilger, who is affiliated with NVESD (Night Vision and Electronic Sensors Directorate).

We will provide specific details of the above two categories of engagement in the conference presentation slides. In this paper (and specifically, in this section), we provide a more elaborate view of the physical demonstration plan. We had introduced the elements of the vehicular swarm-AI collaborative perception demo apparatus in section III-D in the context of Figure 5. Now, we provide a description of the targeted demonstrative test plan under the guidance of DoD/NVESD experts, as depicted in Figure 8. In this figure, we illustrate the human-in-the-loop automated system for threat detection in the context of next generation combat vehicles (NGCVs).

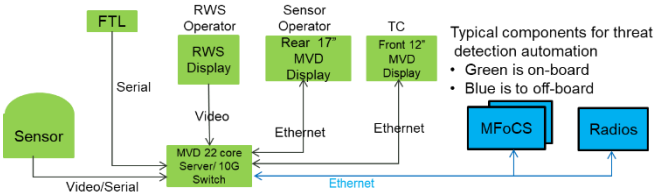

Fig. 8. Demo system composed of MVD (multi-function video display) in the context of NGCV (next generation combat vehicles). Legend: FTL = far target locator; TC = turret commander; RWS = remote weapon station; MFoCS = mounted family of computer systems.

The on-vehicle compute server processes a complex of sense-and-control tasks, driven by various types of video sensor inputs. On detecting an inferred threat, the system raises a communication alert via local/global, wired/wireless ethernet connectivity. The targeted adaptation of the originally-conceived collaborative perception demo (Figure 5) to an autonomous threat detection system atop wirelessly-connected NGCVs is an obvious DoD-relevant extension. In this new context, our goal is to test out the basic real-time functionality of the EPOCHS SoC-family for elementary *Sense-Classify-ThreatDetect-Notify* control loops.

VI. Conclusion and Acknowledgements

In this paper, we provide a technical overview of the IBM-led EPOCHS project, where the goal is to demonstrate agile software-hardware development of edge embedded applications, in the specific context of next generation connected autonomous vehicles. We describe key software and hardware components of the overall methodology and the end demonstration objectives. We have designed and taped out EPOCHS-0, the first of two SoCs targeted for use in final demonstration systems. EPOCHS-1 is targeted to be a larger (6x6 tiled) SoC with added new PE tiles: e.g. an independently tested NLP (natural language processing) engine [29].

The research, development and technology transition tasks covered in this summary overview are being pursued by a diverse team of talented scientists, engineers, students and post-doctoral fellows spread over four locations: IBM Research headquarters in Yorktown Heights, NY; Harvard University at Cambridge, MA; Columbia University in New York, NY; and, University of Illinois at Urbana-Champaign, IL. The following project performers are specifically acknowledged in light of the particular material covered in this paper:

Hardware design and verification: Tianyu Jia, Jeff Zhang, Thierry Tambe, Paolo Mantovani, Maico Cassel, Erik Loscalzo, Joseph Zuckerman, Martin Cochet, Nandhini Chandramoorthy, Karthik Swaminathan, Kevin Tien.

Ontology software, plus ESP/FPGA accelerator integration, emulation and measurements: Georgios Zacharopoulos, Iulian Brumar, David Trilla Rodriguez, Alper Buyuktosunoglu, John-David Wellman, Davide Giri, Kuan-lin Chiu, Giuseppe Di Guglielmo.

System architecture, adjacent-domain workloads and compiler toolset: Akash Kothari, Yifan Zhao, Hashim Sharif, Adel Ejjeh, Sam Grayson, Huzaifa Muhammad, Zeran Zhu, Keyur Joshi, Srijan Chakraborty, Andrew Fortunat.

Smart task scheduler library and policies; full system integration and demo apparatus: Aporva Amarnath, Hubertus Franke, Akin Sisbot (plus contributors from University of Michigan at Ann Arbor – fellow DSSoC performer team).

There are several other technical team members that have contributed (and/or are currently contributing) to the success of the EPOCHS project – as evident from the author names in quoted team papers and sub-projects. The project principals writing this paper would like to express their sincere gratitude to all team members (past and present) that have made this research endeavor a success, despite the severe impediments created by the pandemic over the past year.

REFERENCES

[1] A. Vega, A. Buyuktosunoglu, P. Bose:, "Towards "Smarter" Vehicles Through Cloud-Backed Swarm Cognition," Intelligent Vehicles Symposium 2018: 1079-1086.

[2] L. Carloni, "The case for embedded systems platforms," in Proc. of the 53rd Ann. Design Automation Conf. (DAC), 2016, pp. 17:1 – 17:6. https://github.com/sld-columbia/esp .

[3] IBM Corp., "EPOCHS Reference Application (ERA) and the Mini-ERA abstraction," https://github.com/IBM/era, https://github.com/IBM/Mini-ERA, 2019.

[4] B. Bloessl, "IEEE 802.11 a/g/p transceiver for GNU Radio," https://github.com/bastibl/gr-ieee802-11, 2019.

[5] M. Kotsifakou, P. Srivatsava, M. Sinclair, R. Komuravelli, V. Adve, S. Adve, "HPVM: Heterogeneous parallel virtual machine," in Proc. of the 23rd ACM SIGPLAN Symp. on Principles and Practice of Parallel Programming (PPoPP), 2018, pp. 68-80.

[6] H. Sharif, P. Srivastava, M. Huzaifa, M. Kotsifakou, K. Joshi, V. Adve, S. Adve, "ApproxHPVM: a portable compiler IR for acciuracy-aware optimization," in Proc. of the 34th ACM SIGPLAN conf. on Object-Oriented Programming, Systems, Languages and Applications (OOPSLA), 2019, pp. 186-215.

[7] J. Alsop, M. Sinclair, S. Adve, "Spandex: a flexible interface for efficient coherent interface," Proc. Int'l. Symp. on Comp. Arch. (ISCA), 2018.

[8] P. Bose et al., "SoCs for autonomous vehicles: agile design and programmability challenges," full paper at GOMACTech-2019.

[9] E. Akin Sisbot, A. Vega, A. Paidimarri, J-D. Wellman, A. Buyuktosunoglu, P. Bose, "Multi-vehicle map fusion using GNU Radio," presented at GNU Radio Conference, 2019.

[10] A. Vega, A. Buyuktosunoglu, P. Bose, "Energy-secure swarm power management," Proc. DATE-2018, pp. 1652-1657.

[11] 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," in IEEE Transactions on Very Large Scale Integration (VLSI) Systems, vol. 27, no. 11, pp. 2629-2640, Nov. 2019, doi: 10.1109/TVLSI.2019.2926114.

[12] NVDLA: The NVIDIA Deep Learning Accelerator: http://nvdla.org/ .

[13] A. Vega, J-D. Wellman, H. Franke, A. Buyuktosunoglu, P. Bose, A.Amarnath, H. Kassa, S. Pal, R. Dreslinski, " STOMP: agile evaluation of scheduling policies in heterogeneous multi-processors," DOSSA-3 Workshop@HPCA-2021.; see also: https://arxiv.org/abs/2007.14371 .

[14] CARLA open source simulator: https://carla.org/ .

[15] GNURadio: a free and open source software toolkit for software radio: https://www.gnuradio.org/ .

[16] Robot Operating System (ROS): https://www.ros.org/ .

[17] XRShark visualizer demonstration video: https://youtu.be/Fw6Z4OZd2KI

[18] Z. Luo *et al.*, "MIO-TCD: A New Benchmark Dataset for Vehicle Classification and Localization," in IEEE Transactions on Image Processing, vol. 27, no. 10, pp. 5129-5141, Oct. 2018, doi: 10.1109/TIP.2018.2848705.

[19] M. Kotsifakou, P. Srivastava, M. Sinclair, R. Komuravelli, V. Adve and S. Adve, "HPVM: Heterogeneous Parallel Virtual Machine.**"** Proceedings of Principles and Practice of Parallel Programming **(**PPoPP), Feb 2018; https://dl.acm.org/doi/10.1145/3178487.3178493 .

[20] HPVM portal: https://publish.illinois.edu/hpvm-project/; open-source release, Jan, 27, 2020: https://gitlab.engr.illinois.edu/llvm/hpvm-release

[21] H. Sharif, Y. Zhao, M. Kotsifakou, A. Kothari, B. Schreiber, E. Wang, Y. Sarita, N. Zhao, K. Joshi, V. Adve, S. Misailovic, S. Adve, "ApproxTuner: a compiler and runtime system for adaptive approximations," accepted for presentation at PPoPP, March 2021.

[22] A. Amarnath, H. Kassa, S. Pal, A. Vega, A. Buyuktosunoglu, H. Franke, J-D. Wellman, R. Dreslinski, P. Bose, "AVSched: mission-aware scheduling in heterogeneous SoCs for autonomous vehicles," cleared by DARPA, in submission for publication, 2021.

[23] G. Zacharopoulos, L. Ferretti, G. Ansaloni, G. Di Guglielmo, L. Carloni, L. Pozzi, "Compiler-Assisted Selection of Hardware Acceleration Candidates from Application Source Code," Proc. IEEE Int'l. Conf. on Computer Design (ICCD) 2019, pp. 129-137.

[24] AccelSeeker: open-source accelerator discovery toolset: https://github.com/GiorgioZacharo/AccelSeeker .

[25] G. Zacharopoulos, L. Ferretti, E. Giaquinta, G. Ansaloni, L. Pozzi, "RegionSeeker: Automatically identifying and selecting accelerators from application source code," in IEEE Trans. Comput. Aided Des. Integr. Circuits Syst. 38(4): 741-754 (2019).

[26] D. Trilla, J-D. Wellman, A. Buyuktosunoglu, P. Bose, "Novia: A framework for discovering non-conventional inline accelerators," cleared by DARPA, in submission for publication, 2021.

[27] M. Huzaifa, R. Desai, X. Jiang, J. Ravichandran, F. Sinclair, S. Adve, "Exploring extended reality with ILLIXR: a new playground for architecture research," https://arxiv.org/abs/2004.04643. See also: https://illixr.github.io/ .

[28] D. Stutz, N. Chandramoorthy, M. Hein, B. Schiele, "Bit error robustness for energy-efficient DNN accelerators," Proc. 3rd Conf. on Machine Learning and Systems (MLSys), April 2021 (to appear).

[29] T. Tambe, E-Y. Yang, G. Ko, Y. Chai, C. Hooper, M. Donato, P. Whatmough, A. Rush, D. Brooks, G-Y. Wei, "A 25 mm2 SoC for IoT devices with 18ms noise-robust speech-to-text latency via Bayesian speech denoising and attention-based sequence-to-sequence DNN speech recognition in 16nm FinFET," Proc. ISSCC, Feb. 2021.

[30] Z. Kolter and A. Madry, "Adversarial robustness – theory and practice," https://adversarial-ml-tutorial.org.