Early-Stage System-on-Chip Design Space Exploration for Autonomous Vehicles

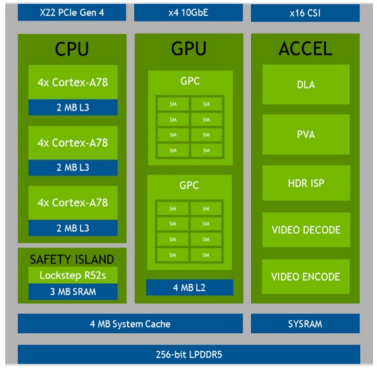
Subhankar Pal*
Aporva Amarnath*
Behzad Boroujerdian^
John-David Wellman*
Pradip Bose*

*IBM T.J. Watson Research Center, Yorktown Heights, NY
^Harvard University, Cambridge, MA

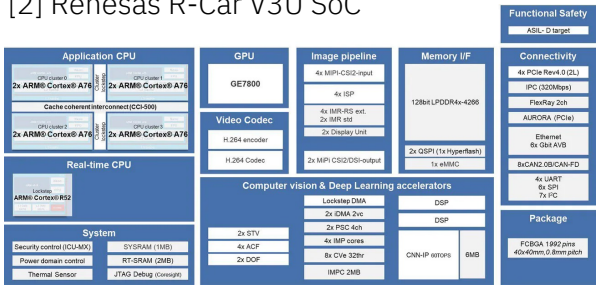
First Workshop on Open-Source Computer Architecture Research (OSCAR), 2022

Domain-Specific Systems-On-Chip for Autonomous Vehicles

[1] NVIDIA AGX Orin SoC



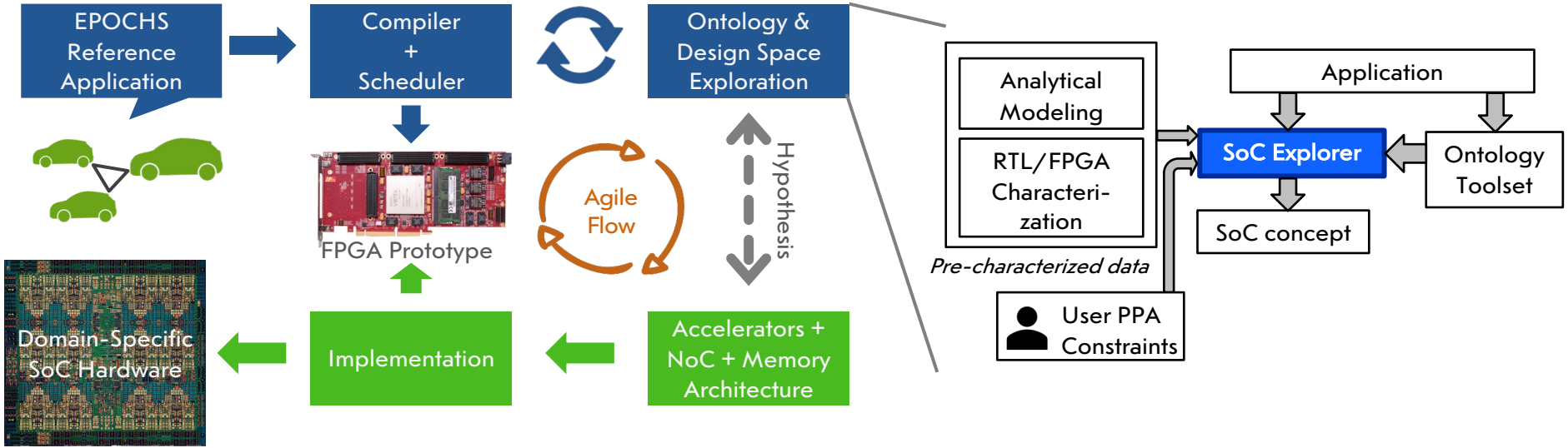
[2] Renesas R-Car V3U SoC



- Heterogeneous DSSoCs gaining rapid adoption in academia as well as industry
- Dominant trends particularly in AV DSSoCs
  - Rising specialization, i.e. hard accelerators for specific tasks
  - Increasingly stringent latency and energy constraints
- NVIDIA DRIVE platform, e.g., uses 2 Orin SoCs [1] and inputs from 28 sensors
- Extensive hardware design space: degree of **heterogeneity**, configuration and sizing of **interconnects** and **memory**, clock speed
- Application complexity added on top
  - E.g. for AVs, vehicle speed, traffic and weather conditions, etc.
- Calls for a fast, yet accurate tool for DSE of DSSoCs

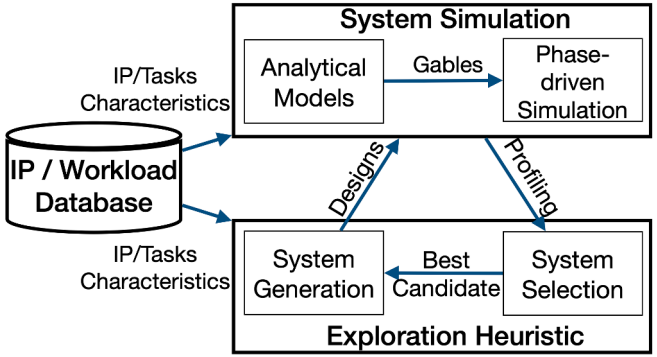
[1] https://www.nvidia.com/content/dam/en-zz/Solutions/gtcf21/jetson-orin/nvidia-jetson-agx-orin-technical-brief.pdf
[2] https://www.renesas.com/us/en/products/automotive-products/automotive-system-chips-socs/r-car-v3u-best-class-r-car-v3u-asil-d-system-chip-automated-driving#overview

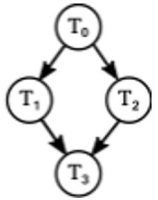
Framework for Agile Exploration of DSSoCs



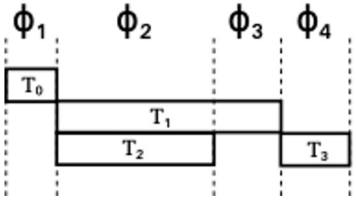
- Agile methodology as part of DARPA DSSoC to quickly design and implement an easily programmed domain-specific SoC for real-time cognitive decision engines in connected vehicles
- Given which "tasks" are the accelerable candidates, our SoC explorer in the agile design toolset answers:
  - How many accelerators of each kind should there be for given PPA constraints?
  - Where should these accelerators be placed in the SoC?

FARSI: A Tool for Early-Stage Exploration of DSSoCs

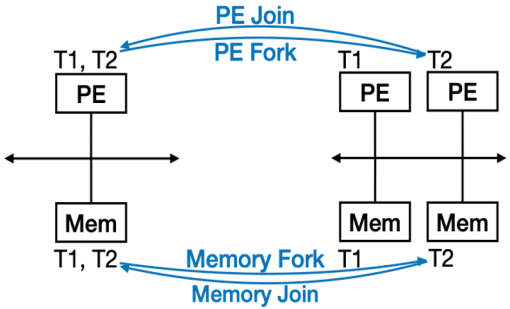




Input as DAG



Phase-driven Simulation

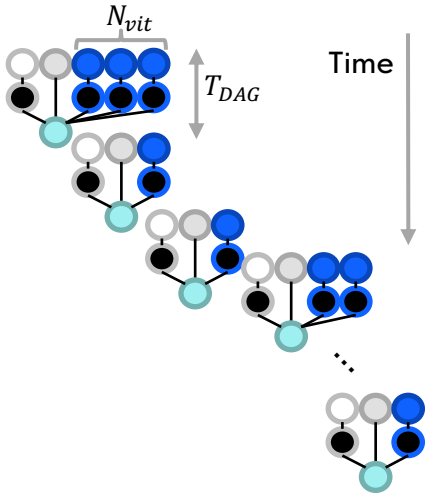


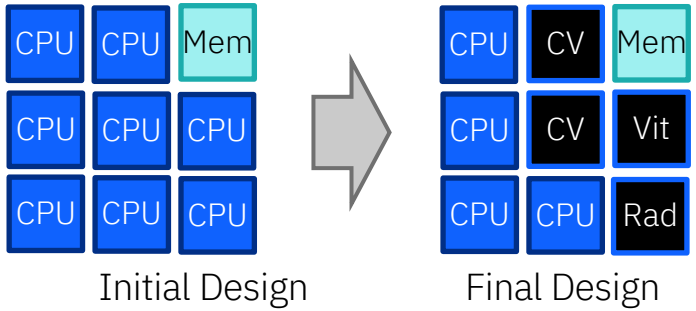
Exploration Moves

- FARSI [3] is a tool that uses analytical modeling with phase-driven simulation for fast, early-stage DSE
  - IP/Workload database populated using offline analysis
    - Workloads represented as directed acyclic graphs where tasks represent functions that can be offloaded to a PE
  - The simulator uses the Gables SoC roofline models and augments it with features such as task-to-task dependency to perform phase driven simulation
  - The explorer uses simulated annealing with architecture aware optimization moves to iterative explore the “best” system, i.e. the system that can fit the PPA budgets the best
  - Can produce an SoC that optimizes for multiple workloads with different PPA constraints
- Previously demonstrated for AR applications with errors of 1.5% compared to Synopsys Platform Architect (PA), while executing 8,400× faster than PA

[3] Boroujerdian, Behzad, et al. "FARSI: An Early-stage Design Space Exploration Framework to Tame the Domain-specific System-on-chip Complexity.”, ACM Transactions on Embedded Computing Systems, 2022.

DSSoC Exploration for AVs: Our Contributions

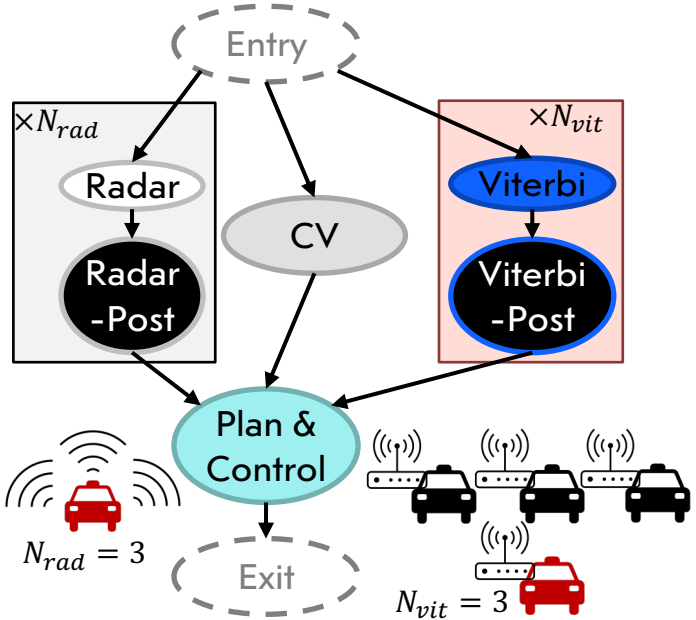




- We generate **DAG-based inputs using probabilistic sampling** for an AV's SoC, based on tasks from the Mini-ERA application [1]
  - We use this to perform DSE of the SoC under latency constraints from pre-characterized data
- We augment FARSI to consume a representative AV application, by incorporating features to **simulate independent DAGs that are staggered in time**
- We incorporate the ability to **constrain the SoC to a certain network topology**
  - Specifically, we augment the SA process with a new heuristic to accelerate the DSE process under this constraint

Input Representation of Mini-ERA

- Mini-ERA [4] is a simple but representative workload for collaborative AVs
- Workload represented as a directed acyclic graph (DAG) with four tasks (nodes):
  - **Radar**: used to model the radar range-finding to the nearest obstacle in a given lane
  - **Viterbi decoder**: used to decode messages received, e.g. via Wi-Fi
  - **CV/CNN** (Computer Vision Object Recognition and Labeling): which identifies the type of obstacle in a given lane
  - The outputs of these three kernels (after post-processing, if required) are used to update the navigation plan and generate actuation signals to control the AV

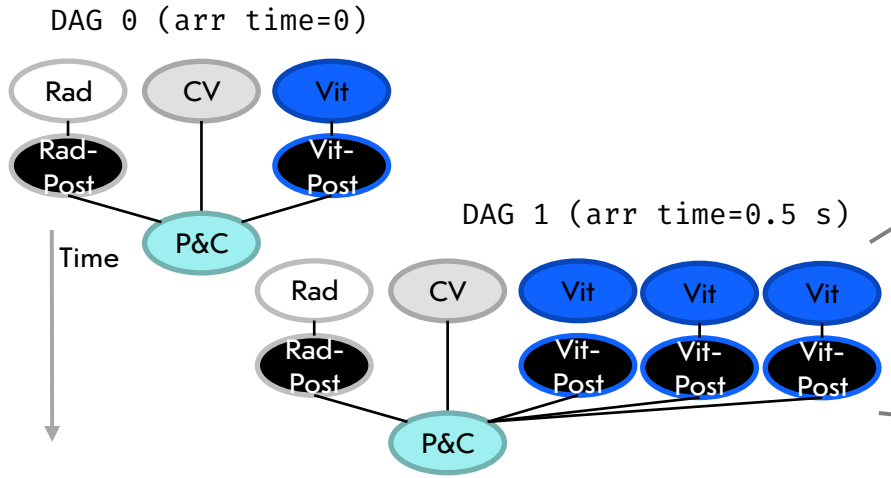


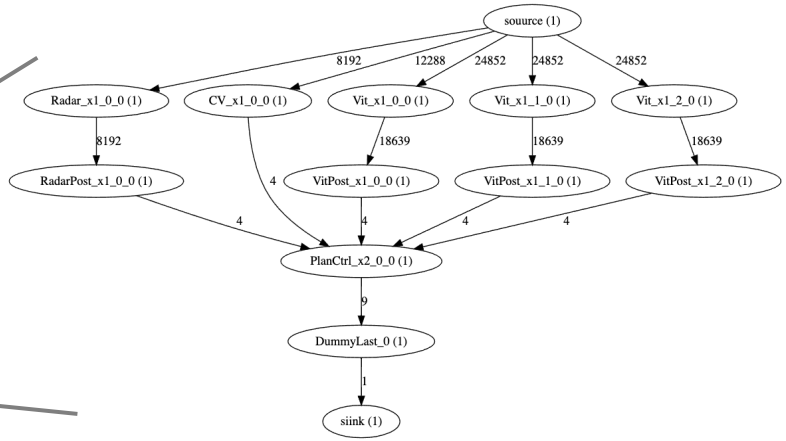
- We consider N_{rad} static radar kernels in each DAG, corresponding to varying number of sensors in the AV
- We also consider N_{vit} dynamic Viterbi decoder kernels, corresponding to different number of connected vehicles in the AV's proximity

[4] https://github.com/IBM/mini-era

DSE for Temporally Staggered DAGs

```
DAG ID, Time (s)
0,      0.0
1,      0.5
```

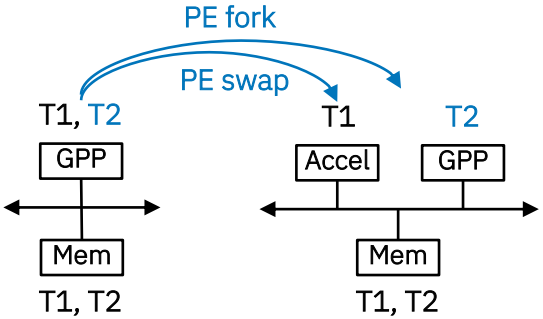




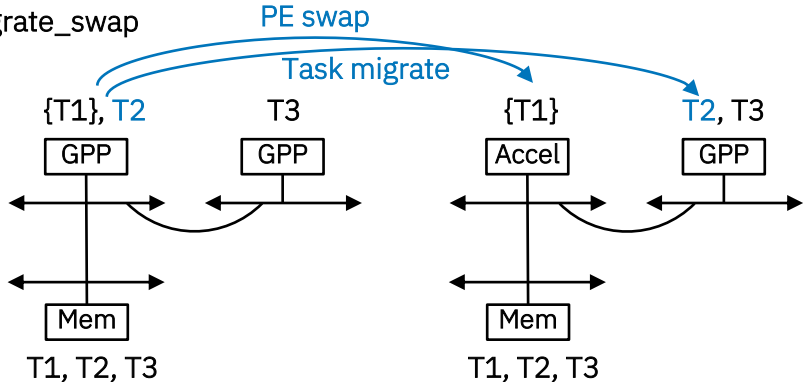
- We enhance FARSI to encode within each task the time at which its host DAG "arrives" into the system
  - FARSI does not schedule a task until the current time is > the task's parent DAG arrival time
- We model a dynamic workload where multiple DAGs arrive at a rate dependent upon the congestion in the system and AV speed
  - E.g. an AV experiences shorter DAG inter-arrival times while driving within a city, compared to a rural highway
- Each DAG corresponds to a different "workload" in FARSI, and thus allows for us to explore SoCs with different deadlines per DAG

DSE under Topological Constraints

Original FARSI move:
fork_swap

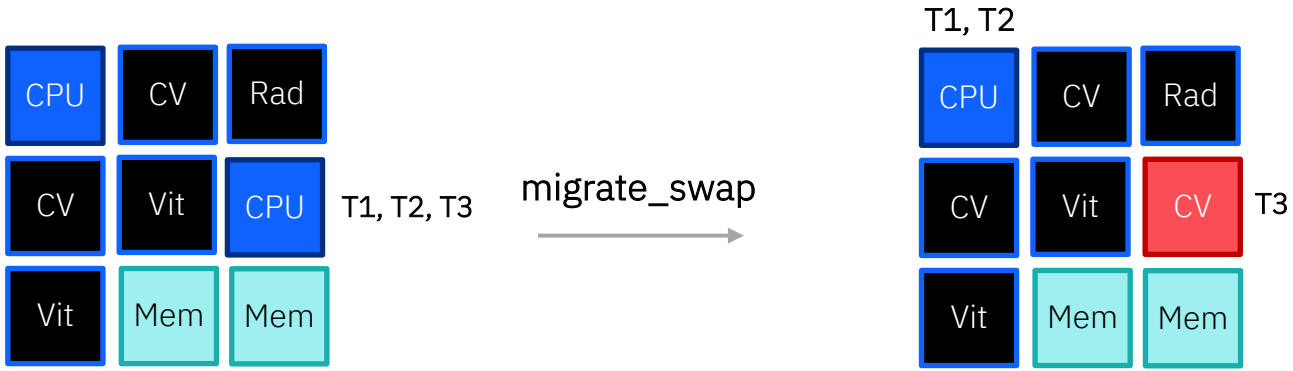


New FARSI move:
migrate_swap



- We introduce several changes in the simulated annealing based DSE part of FARSI
- First, we forbid existing optimization moves in FARSI that involve “forks”
  - Replace them with “migrates”
- Second, we replace fork_swap with migrate_swap that allows for specialization without introducing a new PE
  - New clustering function for migrant task selection: Determine tasks to keep ({T1}) by clustering those that map to the same IP + removing a random number of them for stochasticity

DSE under Topological Constraints



- FARSI originally selects migrant destinations based on locality, but we break this constraint in order to allow more specialization
- Allow the same accelerator to (serially) run multiple tasks and for tasks to migrate to it
- Additional changes e.g. forbidding hardware graph optimizations to maintain the total number of PEs+Mems

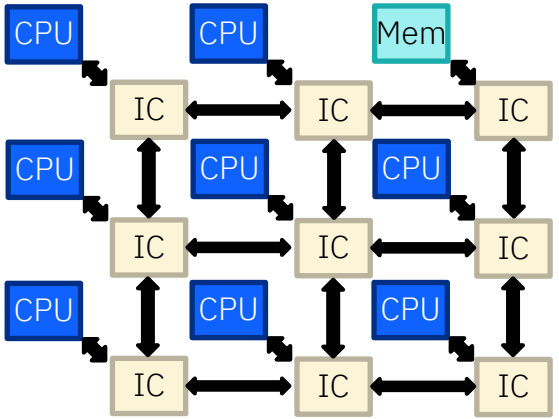
Experimental Setup

PE Performance Characterization Table (values are in cycles)

Task Name	Ariane CPU	NVDLA	Radar Accel	Viterbi Accel
Plan & Ctrl	39,000			
Radar	1,794,000		117,000	
Radar-Post	117,000			
Viterbi	9,360,000			464,100
Viterbi-Post	390,000			
CV	97,890,000*	11,700,000		

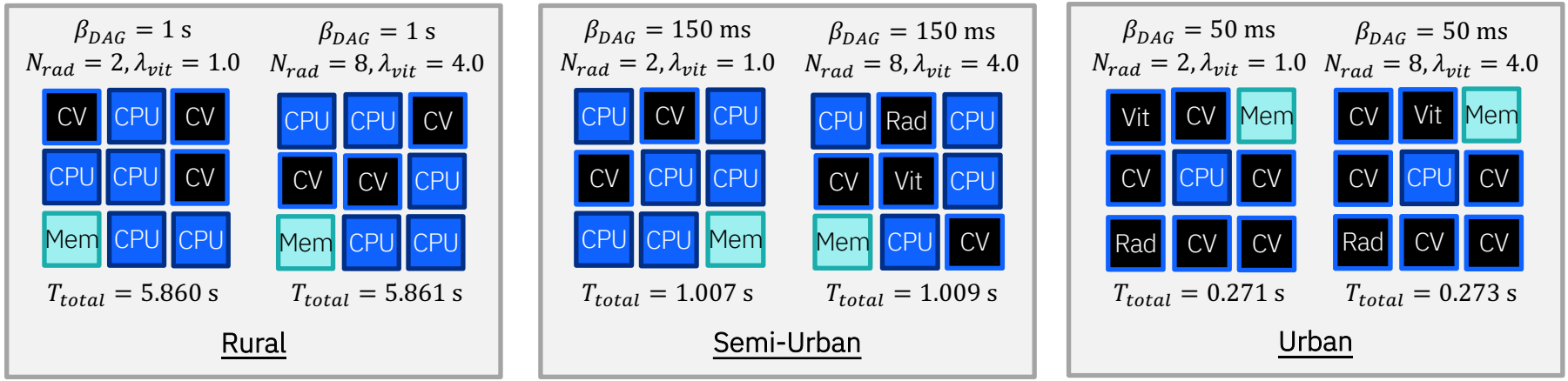
*due to software issues, the CV task could not be run on the Ariane core and its performance is estimated by scaling up its execution time on a Xeon CPU (single-core run)

Initial SoC Design (Hardware Graph) fed to FARSI



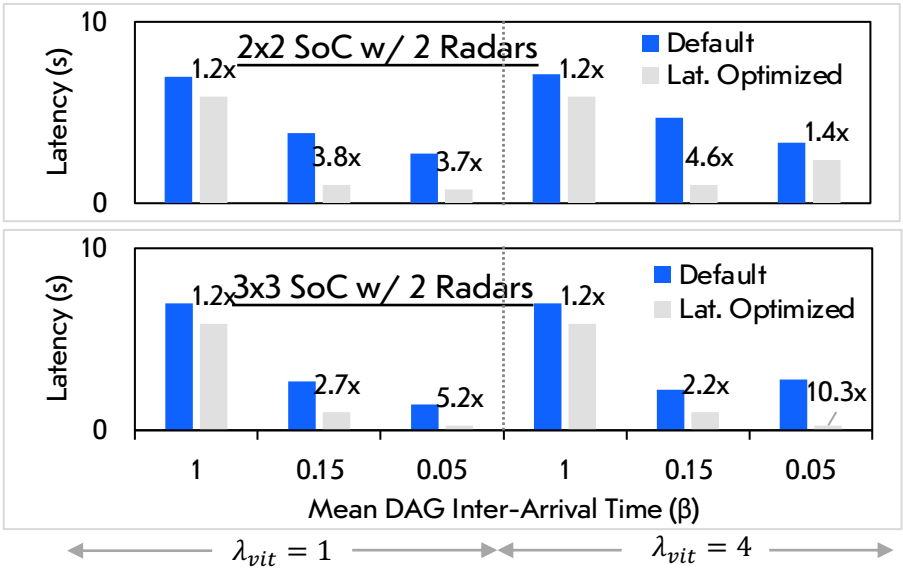
- Evaluate tiled SoC designs (2×2 and 3×3) for Mini-ERA composed of Ariane (in-order RISC-V) CPUs, and accelerators for each of the CV, Viterbi and radar tasks
- Cycle counts obtained from implementations of each PE on a VCU118 FPGA running @ 78 MHz
- We consider generic bus-based interconnects also running @ 78 MHz with widths from 4 to 256 B
- T_{DAG} values sampled from an exponential distribution; scale (β) representing the congestion/vehicle speed
- N_{vit} is sampled from a Poisson distribution; mean arrival rate (λ) representing the traffic on the roadway

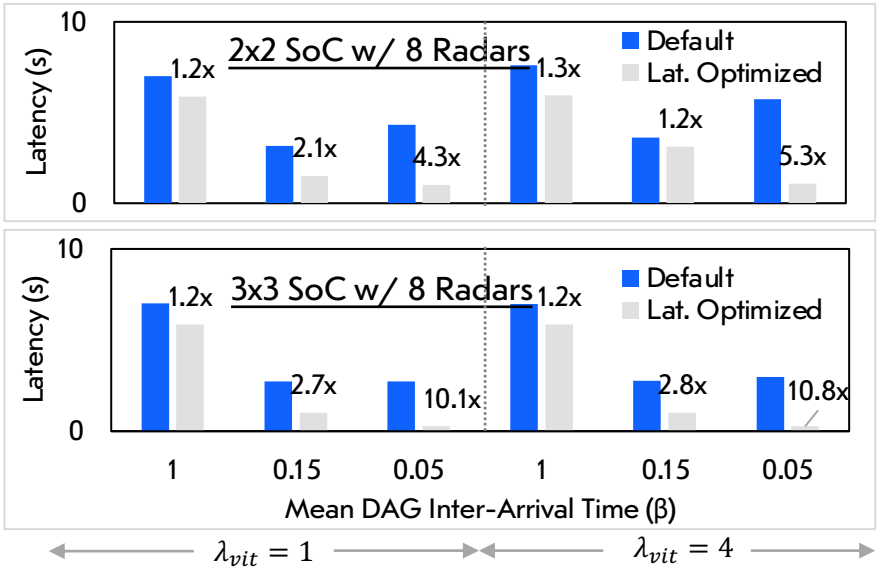
Evaluation – 3x3 SoC Configuration



- More congestion in the environment leads to more need for specialization of the DSSoC
- For the rural scenario, we converge on an SoC design with only CV accelerators and CPUs
  - We are bottlenecked only on the CV task, since its execution time on the CPU is > β_{DAG}
- The semi-urban case sees a crossover from CV-dominated execution to a case each of the three tasks demand acceleration
- The urban case sees a design with maxed-out degree of specialization, with just one CPU
- Using the enhanced FARSI framework for DSE, the end-to-end SoC latency increase is within 0.02-0.73% even with 4× more sensors and 4× more number of connected cars on the road

Evaluation





- Two SoCs for Mini-ERA runs of 5 DAGs with the inter-arrival time swept on the x-axis, representing rural, semi-urban and urban scenarios, also sweeping N_{rad} and N_{vit} to illustrate scaling
- Gains are usually higher for the urban case ($\beta = 0.05$s), compared to semi-urban ($\beta = 0.15$s)
- With the rural scenario, there is a small gain of 1.2× for the FARSI-generated SoC
  - The system mostly idle and waiting for tasks
- Sweeping N_{rad} and N_{vit} makes a smaller difference for semi-urban, as these are bottlenecked by CV

Conclusion and Future Work

We augmented a recent SoC DSE framework called FARSI to consume workloads representative of AV applications

We further enhanced DSE heuristics to explore a fixed topology, which may be required by the designer for IP reuse

Our code will be available on GitHub soon for download!

Future Work

- Explore an average-best case SoC for a suite of probabilistic inputs instead of just one sample
- Explore SoC designs with constraints of energy in addition to latency
- Evaluate a more complex and representative AV application called ERA
- Validate against fabricated SoCs designs

Acknowledgments

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.

Thank You!