Versa: A Dataflow-Centric Multiprocessor with 36 Systolic ARM Cortex-M4F Cores  
and a Reconfigurable Crossbar-Memory Hierarchy in 28nm

Sung Kim1, Morteza Fayazi1, Alhad Daftardar1, Kuan-Yu Chen1, Jielun Tan1, Subhankar Pal1, Tutu Ajayi1, Yan Xiong2, Trevor Mudge1, Chaitali Chakrabarti2, David Blaauw1, Ronald Dreslinski1, Hun-Seok Kim1

1University of Michigan, Ann Arbor, MI
2Arizona State University, Tempe, AZ

Abstract: We present Versa, an energy-efficient processor with 36 systolic ARM Cortex-M4F cores and a runtime-reconfigurable memory hierarchy. Versa exploits algorithm-specific characteristics in order to optimize bandwidth, access latency, and data reuse. Measured on a set of kernels with diverse data access, control, and synchronization characteristics, reconfiguration between different Versa modes yields median energy-efficiency improvements of 11.6× and 37.2× over mobile CPU and GPU baselines, respectively.

Introduction: Conventional programmable architectures have structurally limited support for diverse, dynamically-varying algorithm characteristics. CPUs provide broad programmability, but are burdened by large out-of-order cores and costly data transfers through fixed-function, coherent caches. GPUs have large SIMD thread arrays and employ scratchpad memories but suffer on irregular workloads. FPGAs are highly configurable but incur high hardware overheads and long reconfiguration times. In contrast, Versa is a highly-parallel architecture that supports regular, irregular, and systolic workloads via fast (2 cycle) reconfiguration. The flexible architecture enables algorithm-specific runtime optimizations. Finally, a tree-based method accelerates multi-threaded barrier synchronization operations. Versa is a standalone flexible architecture, in contrast to coarse-grained systems like [2].

Architecture Overview: The Versa architecture (Fig. 1) consists of compute tiles and a 3-level memory hierarchy. Each tile has a cluster of 8 ARM Cortex-M4F worker cores equipped with IEEE 754-compliant floating point units (FPUs). We introduce register-to-register (R2R) links between workers in a 4x2 spatial array, extending transparently across tiles, to support chip-wide systolic algorithms with efficiency gains up to 66.8%. A reconfigurable crossbar (RXB) and 8-slice reconfigurable on-chip memory (ROCM) provide a number of data transfer patterns, each optimized for access latency and throughput. Fast access to cross-mode persistent data (e.g., mutexes) is supported with fixed-function scratchpads in each tile and at the global level. Supervisory tasks and runtime reconfiguration are offloaded to a manager core in each tile.

Reconfigurable Crossbar and Memory: The RXB and ROCM (Fig. 2) are co-designed to provide reconfigurable memory slices that are private per worker, shared across all workers in a tile, or FIFO-buffered between core-pairs. The RXB has 1 bidirectional port per worker/ROCM slice and is reconfigurable to either 1) RXB-shared, 2) RXB-private, or 3) RXB-queue modes. RXB-shared provides all-to-all connectivity between workers and ROCM slices with least-recency-granted arbitration [1]. From the perspective of worker cores, shared slices function as single memory that has 8x the capacity of private slices. RXB-shared is also beneficial for workloads where common data is accessed by multiple cores. In RXB-private, RXB crosspoints lock worker-to-slice connections vertically and horizontal arbitration cycles while dissipating 7.9 mW at 31 MHz. RXB-queue supports common streaming DSP and filtering crossbar and cache overheads (Fig. 3). If enabled at runtime (i.e., in software), the FPU registers are aliased to scalar data links in the <W, E, N, S> directions, respectively. When an instruction writes to an R2R register, data from register writeback - normally FIFO buffered between the CPU and GPU, respectively. Energy-efficiency improvements between Versa modes extend up to 3.17×, illustrating how reconfiguration captures workload-dependent variation (Fig. 6). For instance, Stencil2D with Versa private cache yields 1.37× higher GFLOPS/W relative to private SPM+R2R at small data sizes, but the advantage between modes is inverted at larger sizes. This result is largely due to the use of R2R to share and reuse overlapped input patches across cores, and cache pressure as dataset footprint increases. On Mergesort, Versa attains 2.33× and 71.6× speedups over the CPU and GPU, respectively, translating to 14.4× and 105× energy-efficiency improvements. GPU profiling indicates bottlenecks in parallel synchronization and branch-heavy comparison operations. Results from Mergesort suggest that Versa's independent scalar cores and tree-based scratchpad barriers are effective in-practice.

Acknowledgements: This work was sponsored in part by the U.S. Government under the DARPA SDH program, agreement number FA8650-18-2-7864. We thank ARM for IP and design support.

References
Fig. 1: Main components of a Versa tile (left), and the chip overview (right). Worker cores connected by systolic R2R links and dynamically-reconfigurable memories exploit algorithm-dependent characteristics.

Fig. 3: R2R tunneling and integration in the ARM Cortex M4.

Fig. 4: Illustration of reduced serialization with tree-based barrier synchronization (left), and latency improvements over phreaths (right).

Denominator Platform CPU (ARM A57) GPU (Tegra X1)

Fig. 6: Trends (top) in energy efficiency across dataset sizes with different Versa configurations, and summary boxplots (bottom).

Fig. 7: Power (left), energy-per-cycle (middle), and compute efficiency (right) under VDD scaling between 0.55 - 1.0V. 0.6V coincides with the minimum-energy point.

Fig. 8: Die photo and chip summary.