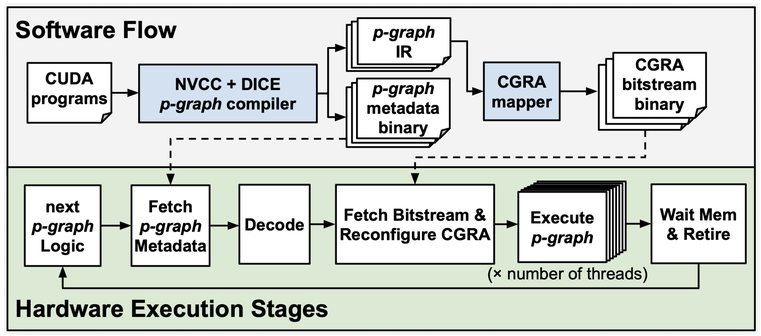
MiniDICE: General-Purpose Dataflow Intelligent Compute Engine

STUDENTS: AADITHYA MANOJ, ALBERT TON, ELLIOT NORMAN, JUWON JUN, SIBO ZHANG

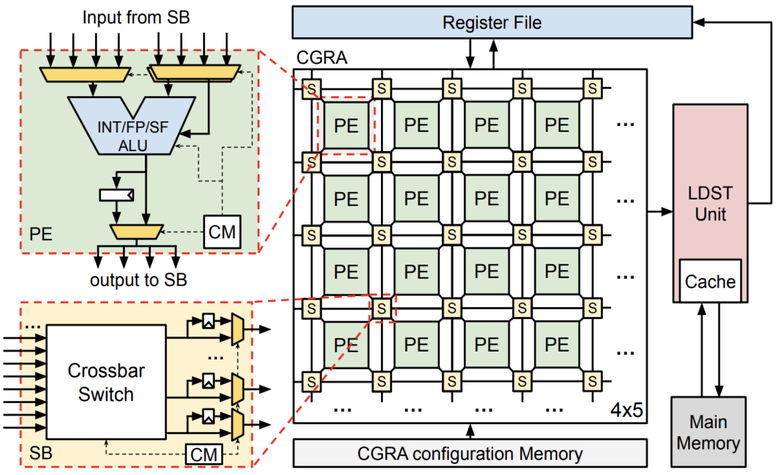
PNCEL

Introduction

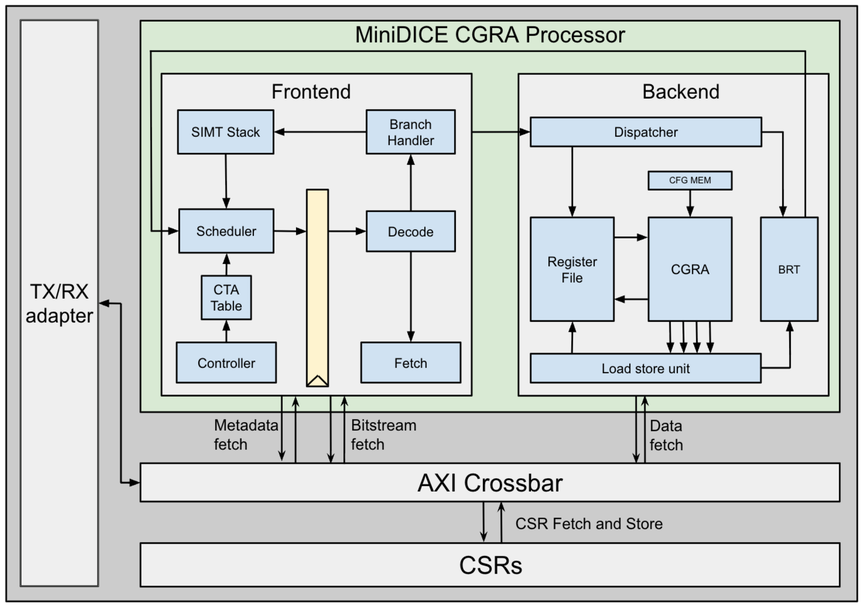
- General-purpose GPUs implement parallel computing through single-instruction, multiple thread (SIMT) execution, but the underlying single-instruction, multiple-data (SIMD) execution spends a large fraction of its energy on register-file traffic.
- MiniDICE replaces the SIMD backend with a 4x4 coarse-grained reconfigurable array (CGRA) fabric while keeping the SIMT execution model, supporting up to 16 threads.
- Instead of repeatedly accessing large register files, MiniDICE enables direct data movement between neighboring processing elements (PEs) within the CGRA fabric
- GEMM is used as a representative workload to evaluate how effectively MiniDICE maps dense computation onto the CGRA fabric.



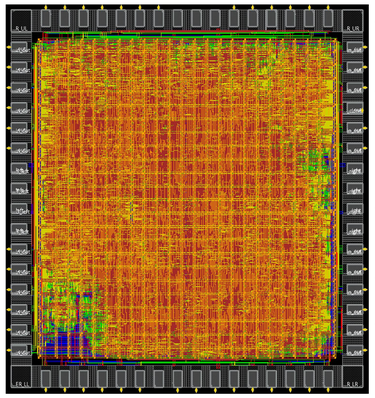
Coarse-Grained Reconfigurable Array (CGRA)



Architecture



Backend Flow



- **Cadence Genus** converts a high-level hardware description into an optimized gate-level circuit that can be physically implemented on silicon.
- **Cadence Innovus** physically implements the synthesized netlist by placing standard cells on the chip and routing wires between them, while meeting timing, power, area, and design-rule constraints.
- **Cadence Tempus** performs static timing analysis to verify timing closure on the post-layout design before tape-out
- **Cadence Conformal** uses formal verification to check RTL-to-netlist equivalence
- **Siemens Calibre** performs physical verification: DRC checks that the layout follows fabrication design rules by TSMC, while LVS confirms that the layout electrically matches the schematic or netlist.

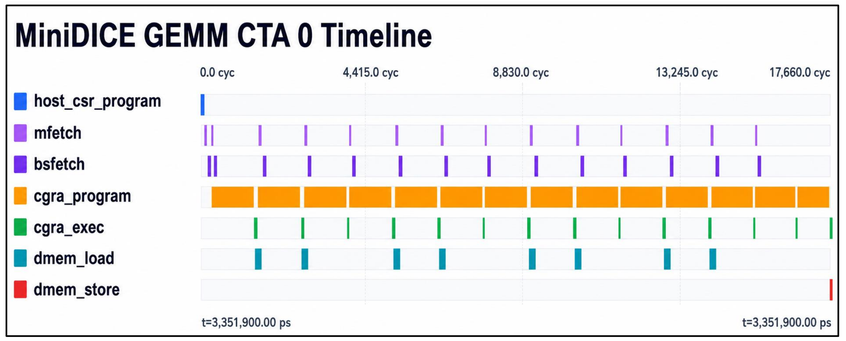
Future Work, References, and Acknowledgments

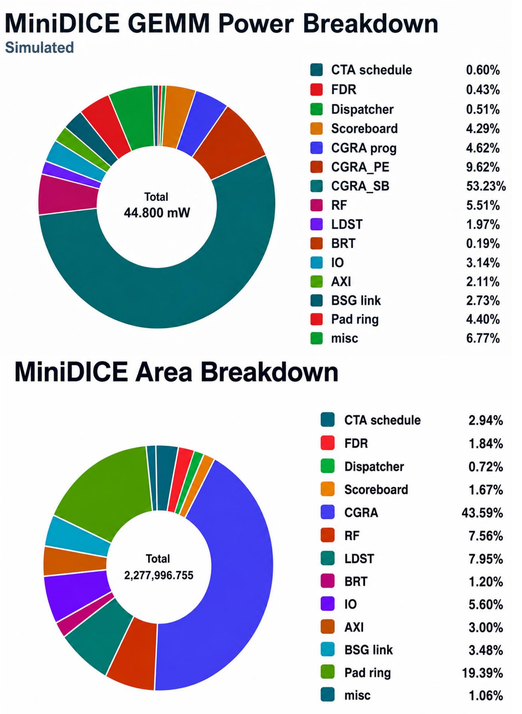
- Testing the chip with an AMD ZCU102 FPGA after fabrication
- Taping out full DICE in TSMC 28nm
- Explore further applications with a CGRA fabric

[1] Wang, J., Lu, A. D., Zeng, Z., & Li, A. DICE: Enabling Efficient General-Purpose SIMT Execution with Statically Scheduled Coarse-Grained Reconfigurable Arrays. ArXiv:2605.05496, 2026. To appear in ISCA 2026.
[2] M. Taylor, "BaseJump STL: SystemVerilog Needs a Standard Template for Hardware Design," in *Proc. 55th Annual Design Automation Conference (DAC '18)*, Article 73, 6 pp., 2018..

Results

Power	**137.85 mW**	Utilization	**77%**
Frequency	**51 MHz**	Std Cell Count	**76,669**
Die Area	**2.27 mm²**	Transistor Count	**278,850**
Core Area	**1.75 mm²**	Setup Slack	**0.362 ns**
Pad Ring Area	**0.44 mm²**	Hold Slack	**0.030 ns**





Design Insights:

- **Programmability + spatial**: Keeps SIMT execution model while using a CGRA fabric for parallel computing.
- **CGRA-based computation**: Most useful work happens inside the CGRA, where PEs communicate through SBs
- **Complete backend flow**: The design is synthesized, placed, routed, timing-checked, and verified using ASIC flow
- **GEMM stresses data movement**: Matrix multiplication repeatedly moves operands across the fabric, making interconnect activity a major power cost.
- **Datapath dominates overhead**: Results show that area are concentrated in compute, memory, and routing structures rather than control logic.

ELECTRICAL & COMPUTER ENGINEERING
UNIVERSITY of WASHINGTON

ADVISERS: ANG LI, JIAYI WANG, PATRICK HOWE

SPONSOR: TSMC, APPLE