An Introduction to the FORZA Architecture

Shih-Hsiang Cheng, Pulkit Gupta, Ajay Mandadi, Jeff Young, Tom Conte, Vivek Sarkar

Georgia Institute of Technology

Georgia Tech College of Computing
Center for Research into Novel Computing Hierarchies

Background

Why FORZA?

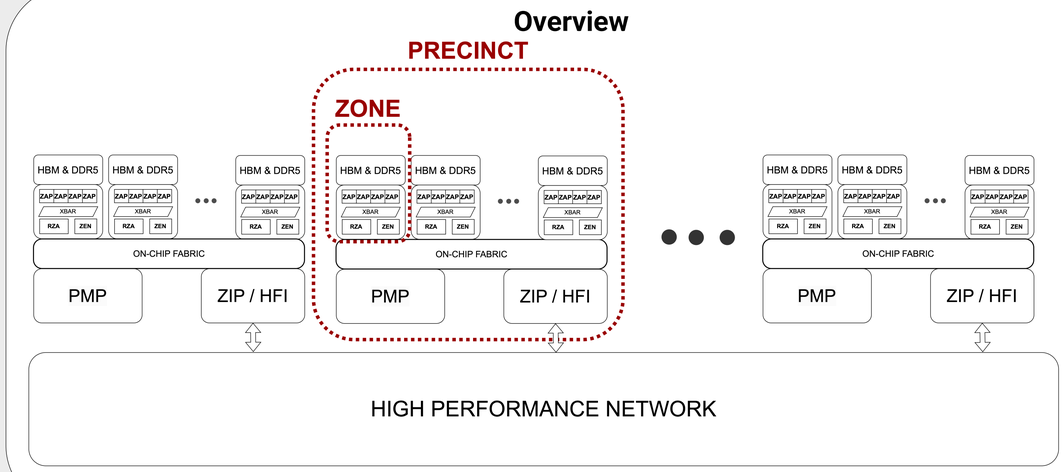
- Motivated by modern and future data analytics centered around distributed graph applications
- Future data analytics workloads are challenging:
  - Prevalence of sparse and irregular relational (graph) data
  - Streaming high-speed data sources
- Challenging because:
  - Exhibit fine-grained parallelism
  - Data movement combined with low operational intensities and poor locality
  - Large graphs (> 2^40 vertices)
- Conventional systems are compute-focused and inefficient on irregular workloads
- The FORZA architecture is data-focused, designed to deliver on the performance aspirations future demands

What's New?

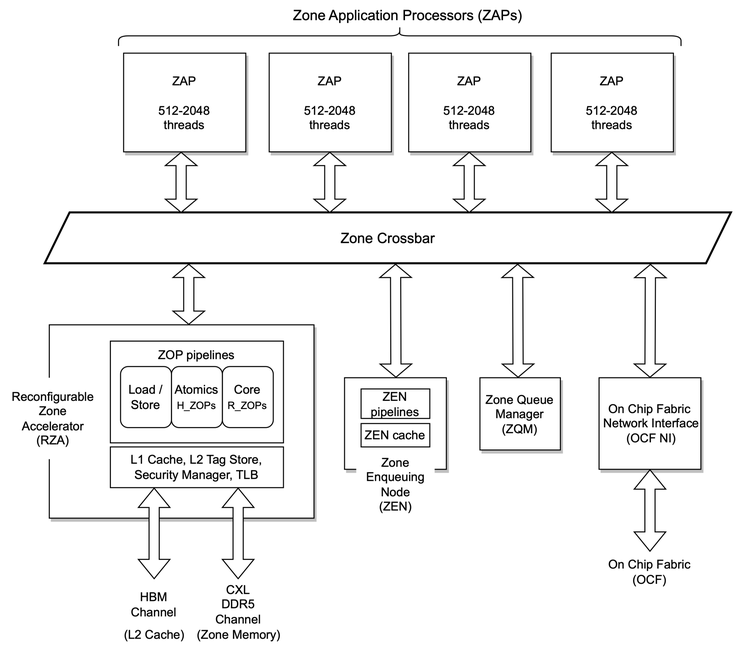
- *Move Compute to Data*: FORZA unifies three classes of compute mechanisms:
  - Actor model
  - Migrating threads
  - Remote atomic operations
- *Performance and Programmability*:
  - Supports new algorithms productively using advances in higher-level libraries and programming systems/models
  - Reduces overheads of traditional APIs by implementing them in hardware
- *Flexibility and Scalability*:
  - FORZA can be deployed across scales that range from deskside systems to multi-cabinet configurations

Architecture Model

Overview



Zone



Key Components

Precinct Management Processor (PMP)

Executes the OS and launch user applications

Network Processor (ZIP)

Serves as point of centralization before accessing the network

Zone Enqueuing Node (ZEN)

Buffers messages to accelerate message delivery message, especially for intra-zone message delivery

Reconfigurable Zone Accelerator (RZA)

Interface to the memory that recaptures the locality by concentrating access from across the system at the data element itself

Zone Application Processor (ZAP)

- Barrel processor with 512 threads interleaved to hide memory latency
- Instruction cache shared by all HARTs in the core
- Scratchpad memory instead of typical-sized L1 data cache to avoid coherence traffic

Georgia Tech