AMD Zen 2

- Tommy Tang, Shuning Liu, Jerry Huang

Core Microarchitecture

Architecture Overview

Front End (in-order):

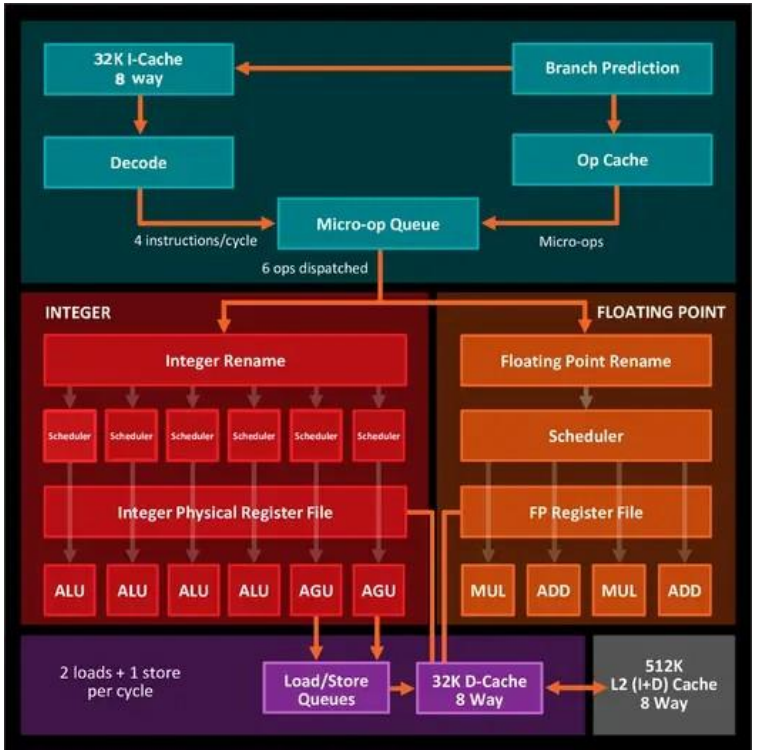
- **Branch Prediction:** Introduces a **TAGE** predictor, achieving **30%** lower misprediction rate.
- **Fetch & Decode:** The first fetch is from **L1 Instruction Cache (32 KB, 8-way)** and requires full decoding into µOPs. Subsequent fetches are from **µOP Cache** and skips decoding entirely.

Back End (out-of-order):

- Splitted execution pipelines between INT and FP.

Memory Subsystem:

- Data is fed into the execution units from the **L1 Data cache (32 KB, 8-way)** via the load and store queue at the rate of 2 loads and 1 store per cycle



https://en.wikichip.org/wiki/amd/microarchitectures/zen

Fetch

TLB: Allows instruction fetching to start earlier.

Branch Target Buffer (BTB) :

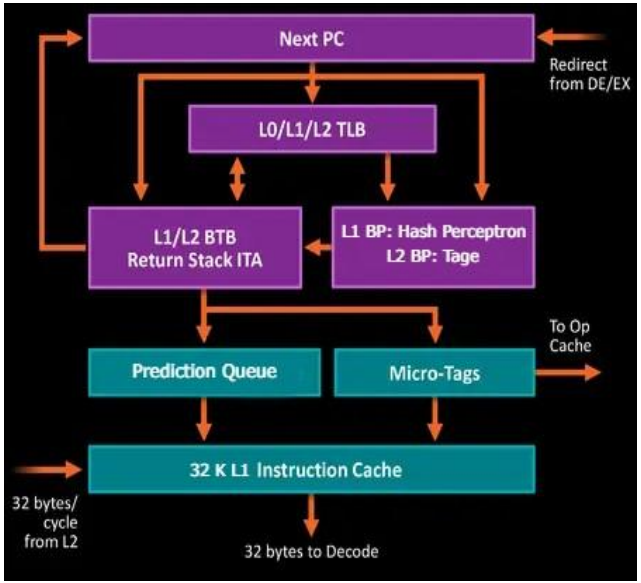
- Store branch targets for **conditional branches**.
- A cache line can hold up to **two** branches.
- Never taken branches are not entered into the BTB, and are implicitly predicted not-taken.

Return Address Stack (RAS):

- Predicts return addresses from a near call.
- Far control transfers (like jump, IRET) are not subject to branch prediction.

Indirect Target Array (ITA): predicts indirect branches (i.e. calls through a function pointer)

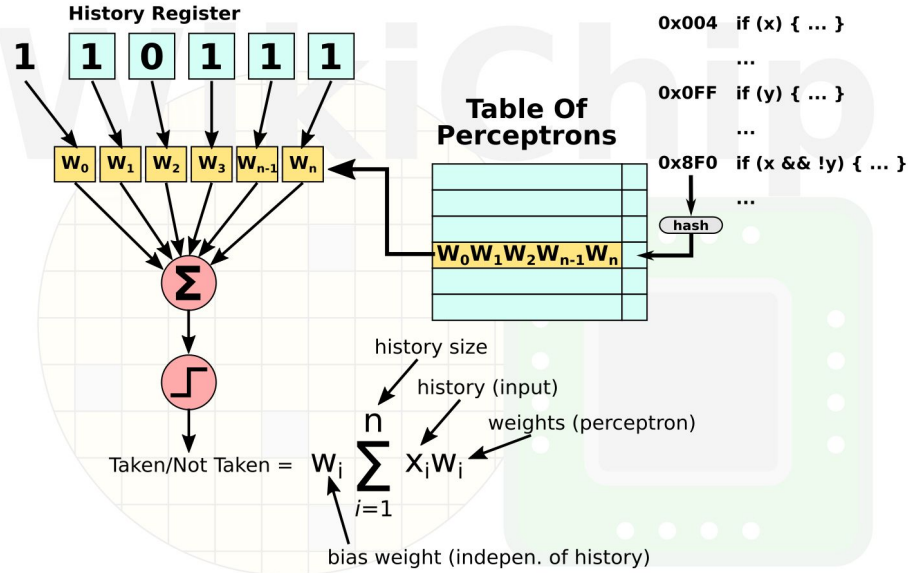
Micro-tags: Determines whether the instruction should be accessed from the µOP cache or from the L1 IC; predicts the way where the instruction block may be found.



https://en.wikichip.org/wiki/amd/microarchitectures/zen

Perceptron

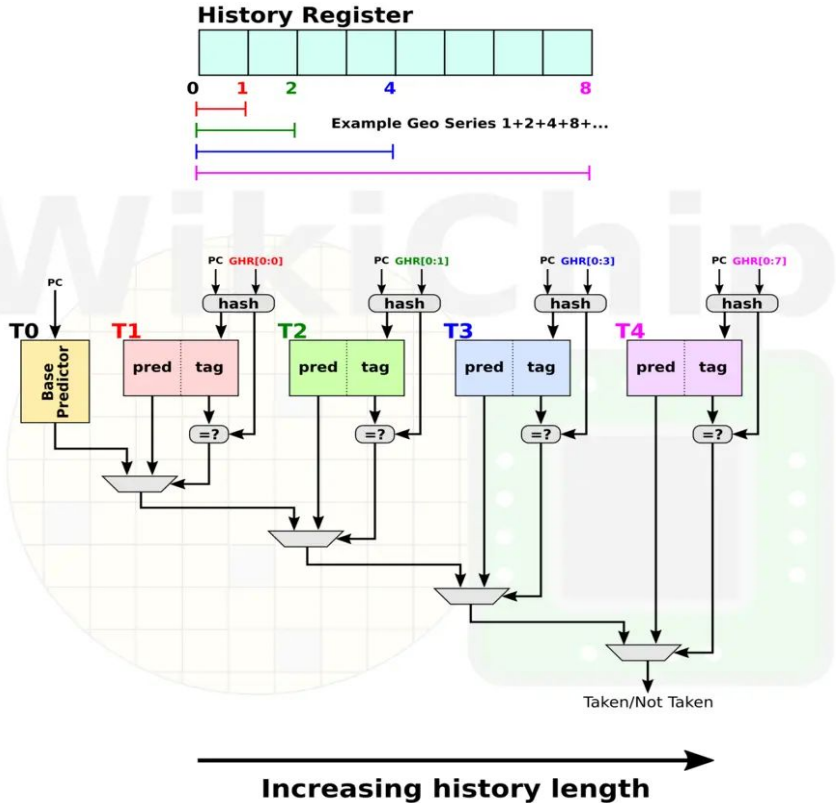
- Simple ML approach with hardware-friendly implementation.
- The address of conditional branch is used to fetch a perceptron (a vector of **weights**) from a **table of perceptrons**.
- Uses bits from the **Global History Register (GHR)** as inputs to predict outcomes of future branches based on prior branch results.
- Negative outputs are predicted as not taken, all other values are predicted as taken.



https://en.wikichip.org/wiki/amd/microarchitectures/zen_2

Tagged Geometric (TAGE) Predictor

- **Idea:** Different history length for different branches.
- TAGE uses multiple prediction tables indexed by varying-length registers to handle all cases.
- The basic predictor (L1 predictor) is the **perceptron.**
- L2 TAGE (slow/high-accuracy) validates L1 (fast/low-accuracy); discrepancies cause flushes and L2 overrides.



https://en.wikichip.org/wiki/amd/microarchitectures/zen_2

Decode

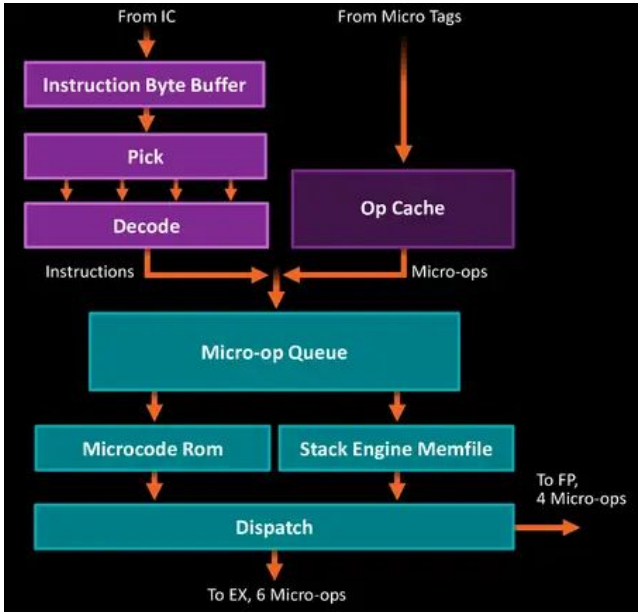
Instructions may be fetched from the **instruction cache (IC)** or **µOP cache** based on the branch prediction, eliminating the need for costly instruction decoding.

Instruction Cache (IC): On a miss, it requests the missing cache line and up to thirteen 64-byte blocks, using addresses from the **branch predictor** and preparing them in the prediction queue.

Decode: Complex **x86 instructions** are broken down to into simpler fixed-length **macro-operations (MOPs)**. **MOPs** are further cracked into smaller **micro-operations (µOPs)**.

µOP Cache: Stores decoded instructions. Avoids decoding, saving power. "**Micro-Tags**" is used to decide whether to fetch from the µOP cache or L1 IC.

Dispatch: sending up to 6 µOPs to Integer EX and an up to 4 µOP to the Floating Point EX



https://en.wikichip.org/wiki/amd/microarchitectures/zen

Integer & Memory Operations

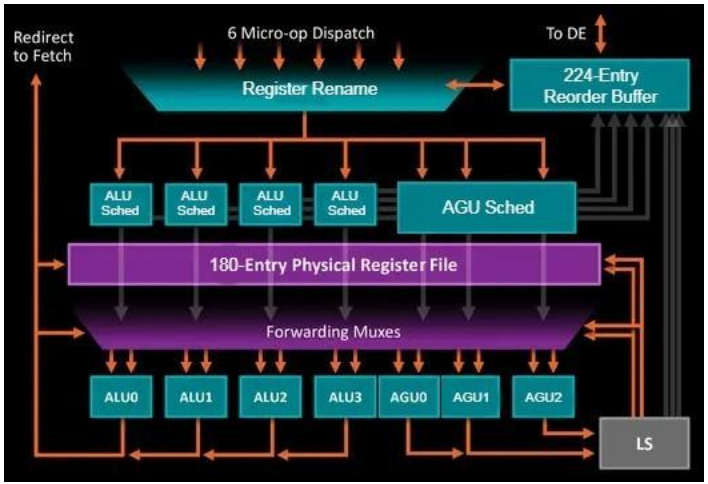
Rename Unit:
- Receives up to 6 µOPs per cycle.
- Zero latency for register to register moves (registers mapping to the same physical register).

Reorder Buffer (Retire Queue):
- Shared by both Integer and FP sections.
- Tracks up to 224 µOPs; retires up to 8 µOPs per cycle.

ALU pipelines:
- 4 ALUs, each with a ALU scheduler that tracks the operands availability and issues one micro-op per cycle to the ALU (FIFO order).
- Perform integer arithmetic, logic operations. 2 ALUs can perform branch.
- Each managed by a 16-entry **micro-op scheduler**.

AGU pipelines:
- 3 AGUs with a unified AGU scheduler that issues three micro-ops per cycle to the AGU.
- Perform memory computation for load or store micro-op.
- Two AGUs handle load addresses for the load queue, while all three AGUs handle store addresses for the store queue.

Forwarding Muxes: forwards ALU and load results directly to earlier stages, enabling back-to-back execution of dependent instructions



https://en.wikichip.org/wiki/amd/microarchitectures/zen

Floating Point Operations

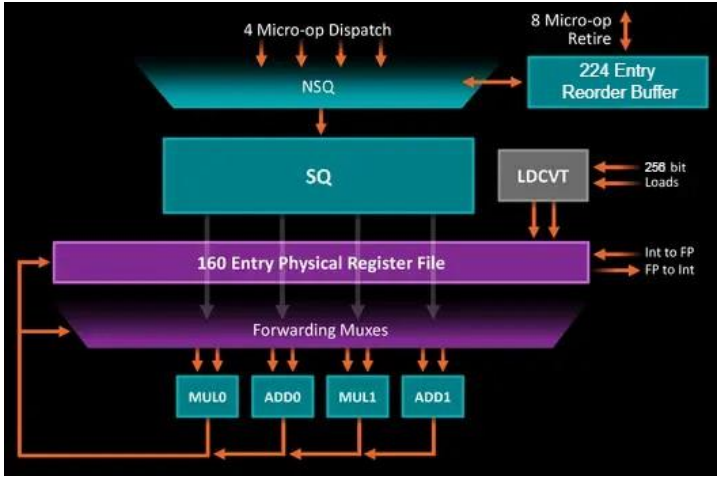
Non-Scheduling Queue (NSQ):
- Wait buffer for FP instructions, which have higher latency
- Allows dispatch to proceed with integer side while FP scheduler is busy.

Scheduling Queue (SQ) / Scheduler:
- Single 36-entry out-of-order FP scheduler.
- Issues up to 4 micro-ops per cycle to execution pipelines.

Load Convert Unit (LDCVT):
- Converts load data to the internal FP register file format.

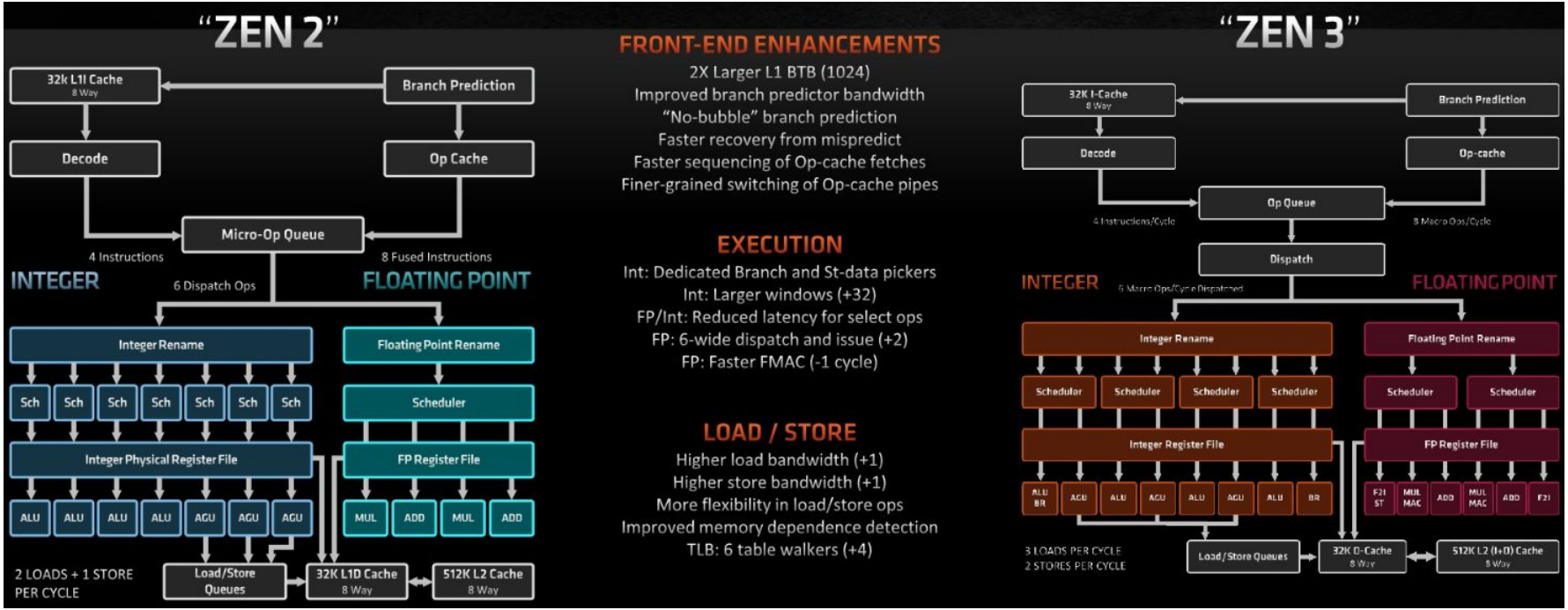
Physical Register File (PRF):
- Holds both speculative and committed values.
- Supports 8 read ports and 4 write ports.
- 2 additional write ports supporting up to two 256-bit load operations per cycle.

Multiplication and Addition Pipelines:
- vector integer operations, floating point operations, and store operations
- Each pipe receives two operands from the PRF or bypass network.
- Pipes 0 and 1 support **fused multiply-add** with 3 operands, and take 5 cycles.
- Pipe 2 executes floating point store micro-ops, sending data to the store queue, one 256-bit store per cycle
- Double-precision **vector multiplication** latency reduced from 4 to 3 cycles.



https://en.wikichip.org/wiki/amd/microarchitectures/zen

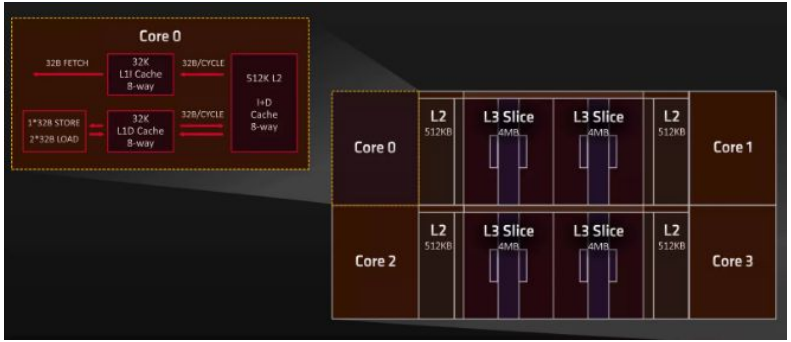
Changes in Higher Versions



https://www.anandtech.com/show/16214/amd-zen-3-ryzen-deep-dive-review-5950x-5900x-5800x-and-5700x-tested/2

Memory Hierarchy

Memory Hierarchy Overview



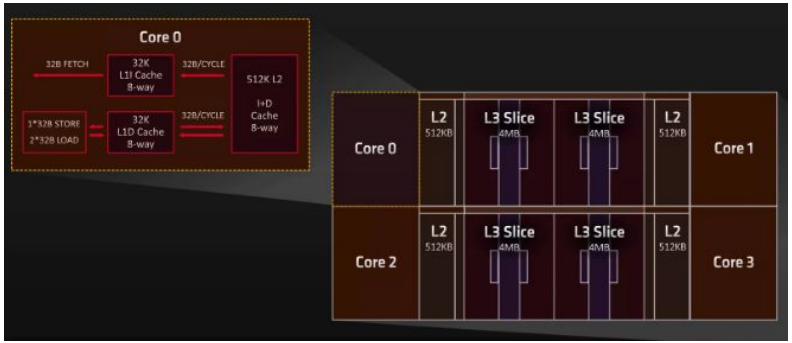
- Private L1 and L2 caches *per core*
- Shared L3 cache per CCX (Core Complex; group of *4 cores*)
- Memory controller on centralized I/O die which connects to main memory
  - Responsible for when there are cache misses
- Infinity Fabric connects to the CCDs and I/O die
  - CCDs (core complex die) contain 2 CCXs

Goals: reduce access latency, maximize bandwidth across cores

L1 and Op (µOP) Cache

L1 Cache:

- L1I: 32 KB, 8-way associative *instruction* cache
- L1D: 32 KB, 8-way associative *data* cache



https://www.slideshare.net/slideshow/the-path-to-zen-2/164957690#2

L2 & L3 Caches

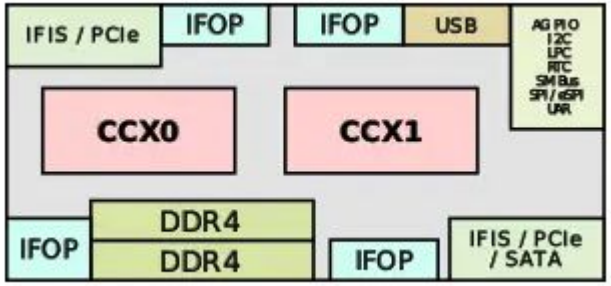
L2 Cache (Per Core) 512 KB, private, 8-way associative

- Inclusive of L1 caches Low-latency buffer for per-core workloads (*consistency*)
- Goal: low-latency buffer per core

L3 Cache (Per CCX) 16 MB, 16-way associative, shared across 4 cores

- Reduces DRAM traffic, improves inter-core communication
- Goal: fast sharing between cores & reduce DRAM traffic; e.g., if multiple threads are reading from the the same L3 cache

Chiplet Architecture and Infinity Fabric



New Chiplet Design:

- CCDs: up to 8 cores (2 CCX per CCD / 8 cores + 32MB if L3 cache)
- Each CCD communicates with a centralized I/O die
- Built separately for better *manufacturing yield* and *scalability*

Infinity Fabric: communication bus that connects CCDs to I/O Die; enables scalable, high-speed communication

- Scalable so that we can scale from 4- to 64-core CPUs
- Maintains cache coherency and synchronizes memory traffic across chiplets

https://en.wikichip.org/wiki/amd/infinity_fabric

Memory Subsystem and NUMA

Unified Memory Controller (UMC): Sits on the I/O die

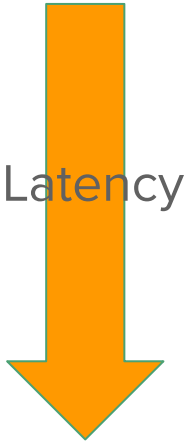
- Supports DDR4-3200 memory and maximizes memory bandwidth across many cores (3.2k mega transfers/sec)

NUMA (Non-Uniform Memory Access):

- Access latency depends on core and memory proximity
- NUMA-aware software optimizes access patterns

Memory Access Workflow

1. Check L1 ➜ miss?
2. Check L2 ➜ miss?
3. Checks L3 in the CCX ➜ miss?
4. Fetch from main memory
    a. Send a request from the CCD via Infinity Fabric to the I/O die
    b. UMC fetches data from DRAM
    c. Data is returned to the specific core via Infinity Fabric



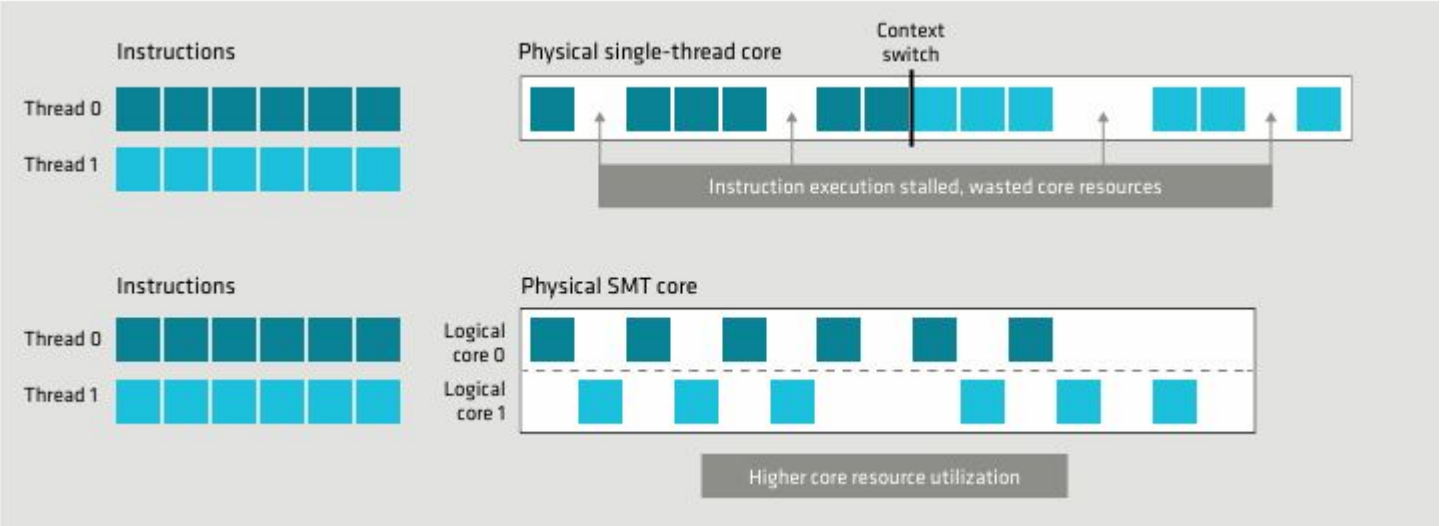
Parallelism Support

Vector / SIMD (Instruction-Level Parallelism)

- **256-bit Wide Execution Units**: Zen 2 architecture doubles the width of the floating-point execution units and data paths from 128 bits (in Zen and Zen+) to 256 bits, enhancing throughput for vector operations.
- Later, Zen 4 introduced support for AVX-512 instructions, executing 512-bit operations internally as two 256-bit micro-operations to maximize energy efficiency; Zen 5 retains AVX-512 support with further refinements for scheduling efficiency.

SMT Support (Thread-Level Parallelism)

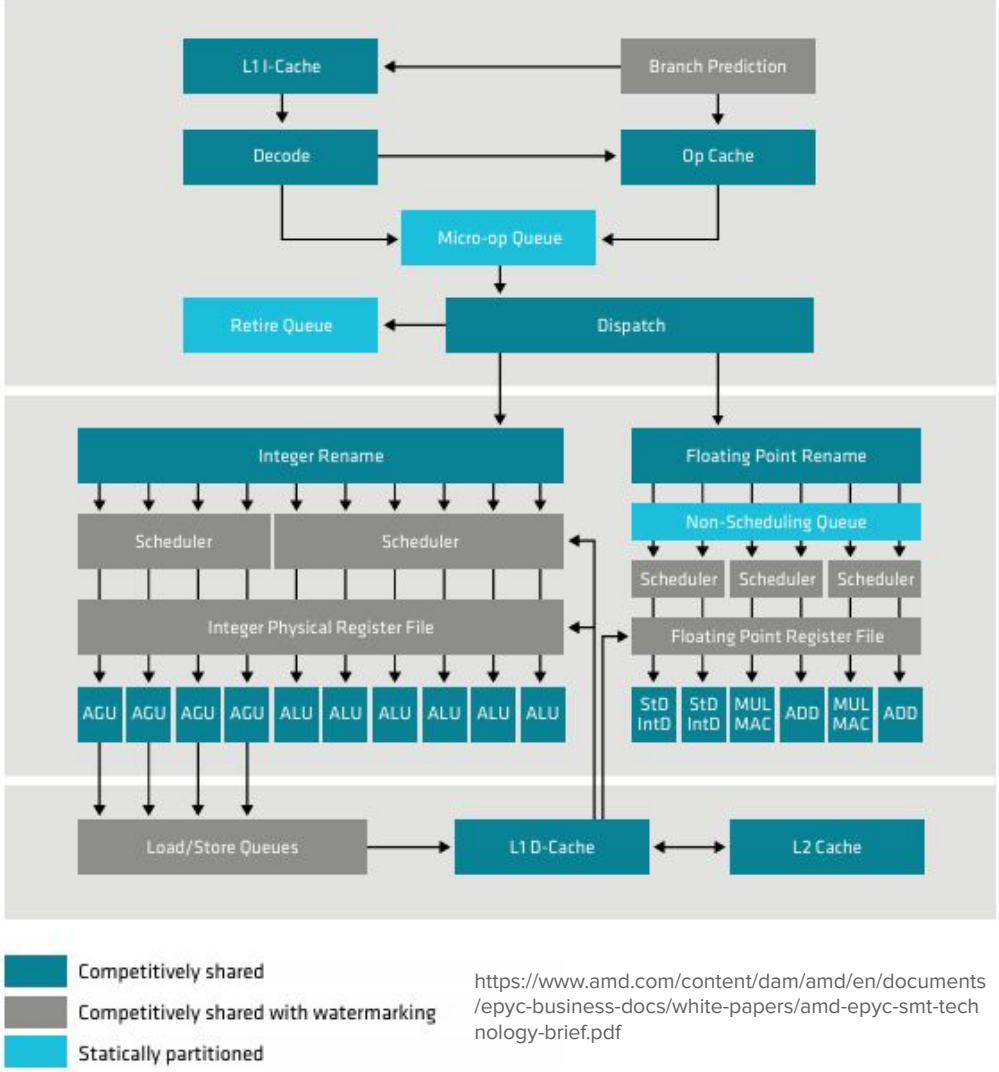
Simultaneous multithreading (SMT) is a technology that allows a CPU core to execute multiple threads simultaneously.



https://www.amd.com/content/dam/amd/en/documents/epyc-business-docs/white-papers/amd-epyc-smt-technology-brief.pdf

Resource sharing on SMT

- Improved core resource utilization
- Increased throughput
- No Physical cost
- Do require security measures



https://www.amd.com/content/dam/amd/en/documents/epyc-business-docs/white-papers/amd-epyc-smt-technology-brief.pdf

Multicore (Core-Level Parallelism)

Zen 2 packages consist of up to 8 Core Complex Dies (CCDs) plus 1 I/O Die (IOD).

Each CCD:

- Contains 8 CPU cores, divided into 2 Core Complexes (CCX).
- Each CCX = 4 cores sharing 16 MiB of L3 cache.
- Therefore, 32 MiB of L3 cache per CCD.

L2 cache:

- Each core has 512 KiB private L2 cache.

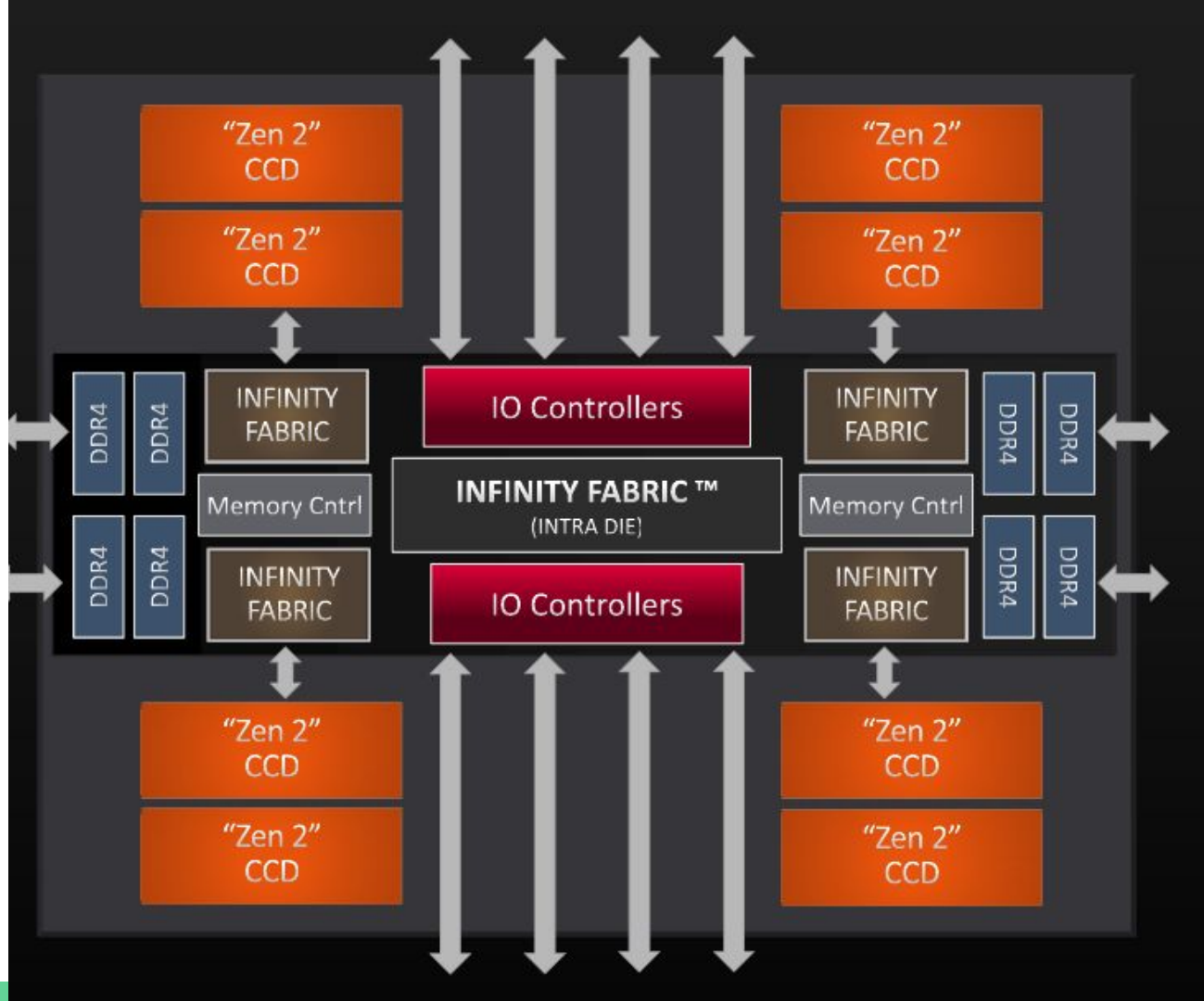


https://www.amd.com/content/dam/amd/en/documents/epyc-technical-docs/tuning-guides/amd-epyc-7002-tg-hpc-56827.pdf

Multi-Socket

Zen 2 supports multi-socket (2P) configurations in server platforms.

- Two sockets are connected via external Infinity Fabric.



Zen2 Key feature Summary

- Improved Frontend: TAGE and perceptron branch predictors enhance accuracy; µOP cache bypasses decode to save power and latency.
- Out-of-Order Execution: Separate INT and FP pipelines with deep scheduling and high µOP throughput (6 INT, 4 FP per cycle).
- Efficient Memory Hierarchy: Private L1/L2 caches, shared 16 MiB L3 per CCX, and µOP cache for faster instruction delivery.
- Chiplet Design: Up to 8 CCDs + 1 I/O Die connected by Infinity Fabric for better scalability and yield.
- Strong Parallelism: Supports SIMD (256-bit), SMT (2 threads/core), multicore (up to 64 cores), and dual-socket NUMA configurations.

Thank you for watching