“Bulldozer”

A new approach to multithreaded compute performance

Mike Butler, AMD Fellow
Chief Architect / Bulldozer Core

August 24, 2010

Two x86 Cores Tuned for Target Markets

"Bulldozer"

Performance & Scalability

Mainstream Client and Server Markets

"Bobcat"

Flexible, Low Power & Small

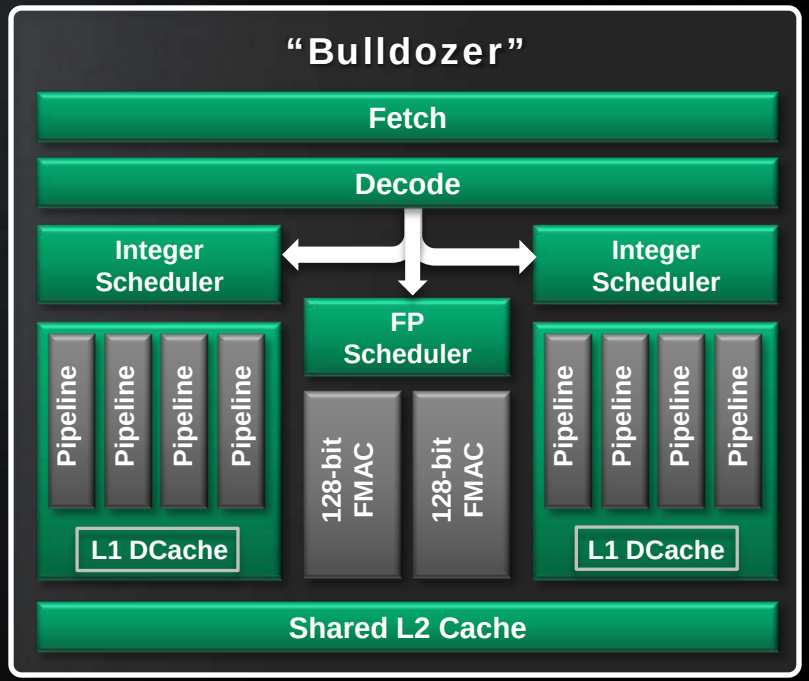
Low Power Markets

Small Die Area

Cloud Clients Optimized

"Bulldozer" x86 Architecture:
AMD's Latest Leap Forward

- Two tightly linked cores share resources to increase efficiency
- ISA extensions, including FP "FMAC"
- Extensive new power efficiency and management innovations
- Designed for knee-of-the-curve IPC features and low gates/clock
- 2011 desktop and server



Approaches for Supporting Multiple Threads

SMT

- Force two threads into one core
- Threads compete for resources
- Relies on under-utilization

CMP

- Dedicated cores for each thread
- Traditional brute force approach
- Each core is over-provisioned

However, there is another way . . .

Bulldozer Concept

Start with 2 cores:

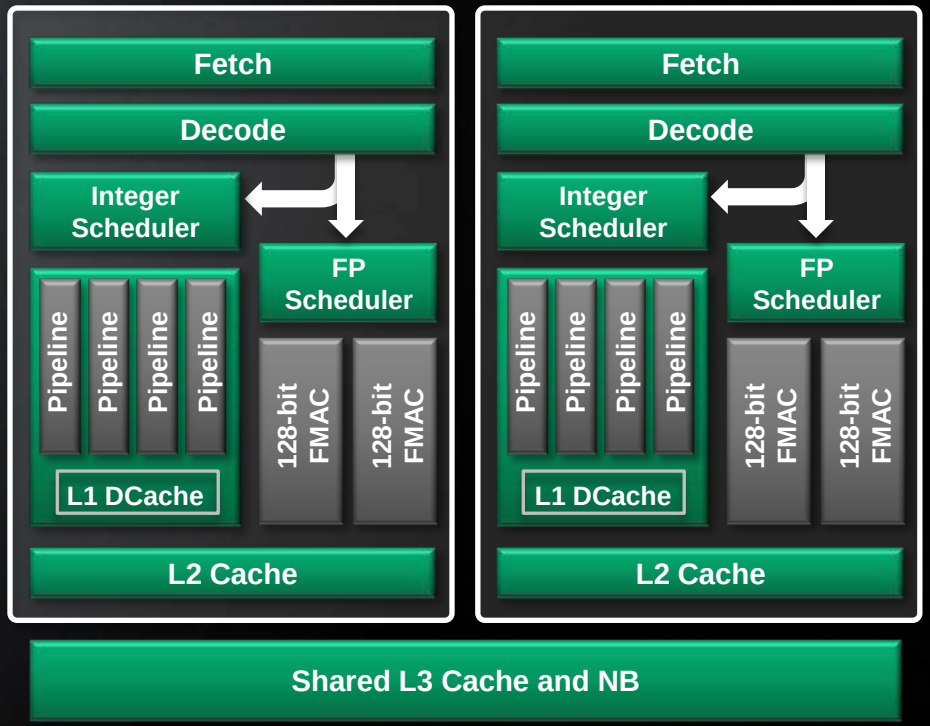
- Fully-capable core performance level

Share hardware when:

- Usage is naturally bursty for a single thread
- Little impact on timing and complexity of critical paths
- Benefit from increasing amortized bandwidth

Invest:

- Increase shared bandwidth/capacity
- Aggressive features to benefit both threads
  - E.g. data prefetch



Bulldozer

What it is:

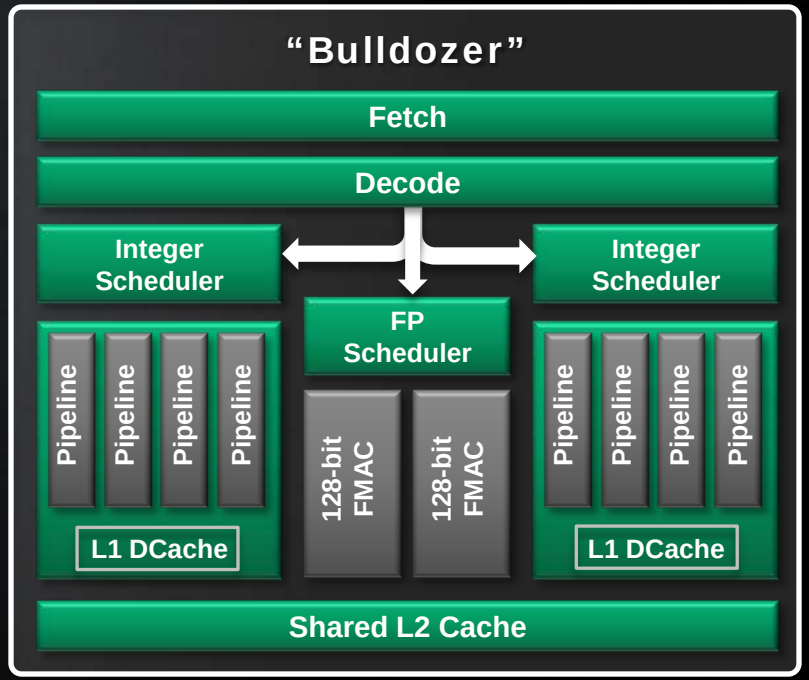
- A monolithic dual core building block that supports two threads of execution

How it works:

- Shares latency-tolerant functionality
- Smoothes bursty/inefficient usage
- Dynamic resource allocation between threads

Customer Benefits:

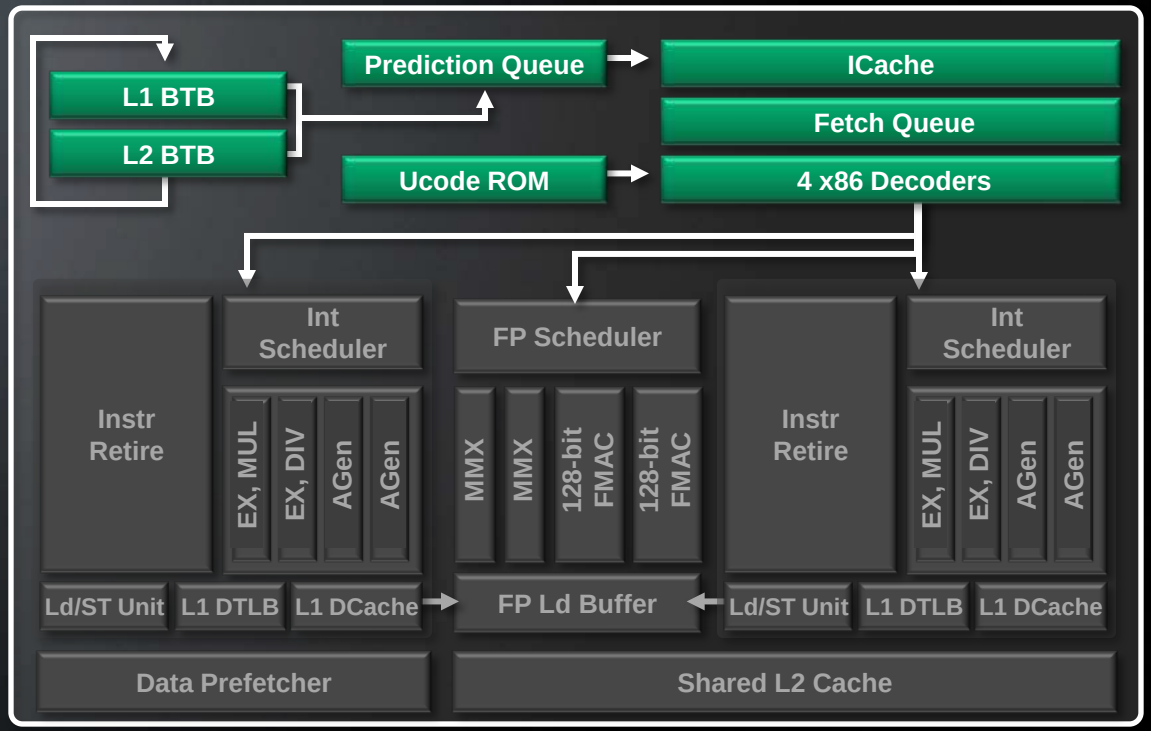
- Greater scalability and predictability than two threads sharing a single core
- Throughput advantages for multi-threaded workloads without significant loss on serial single-threaded workload components
- When only one thread is active, it has full access to all shared resources
- Estimated average of 80% of the CMP performance with much less area and power *



*Based on internal AMD modeling using benchmark simulations

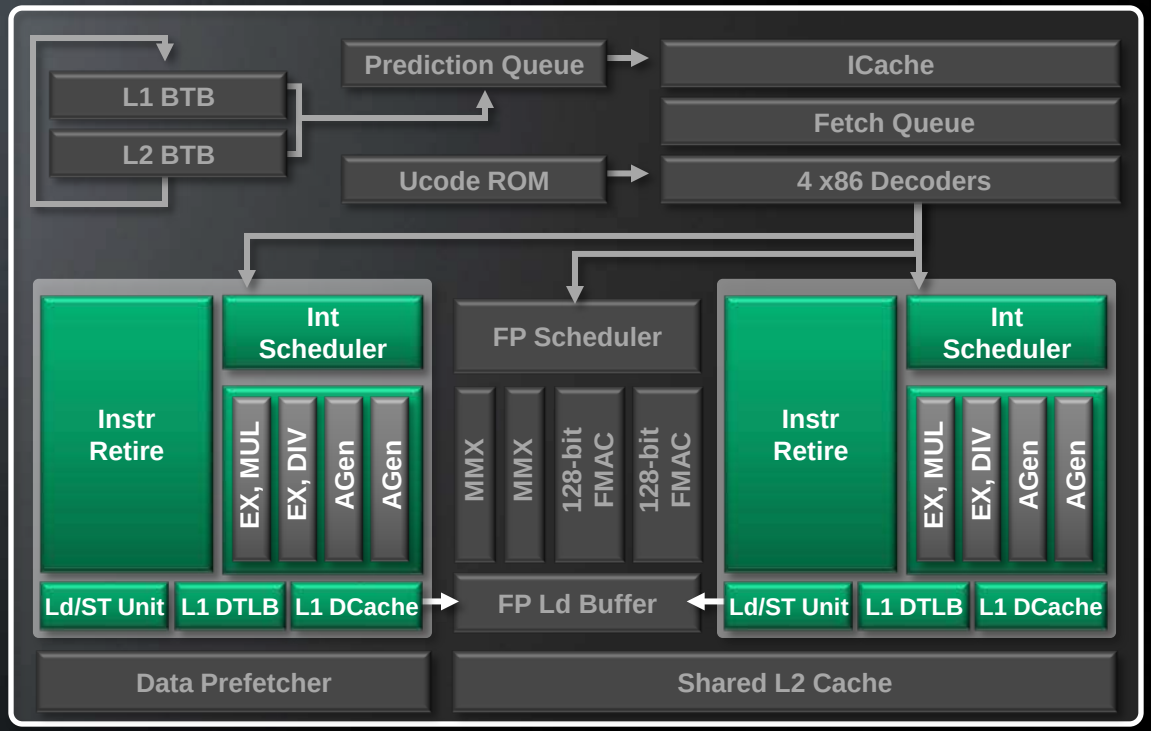
Core Microarchitecture – Shared Frontend

- Decoupled predict and fetch pipelines
- Prediction-directed instruction prefetch
- Icache: 64K Byte, 2-way
- 32-Byte fetch
- ITLBs:
  - L1: 72-entry, FA, mixed page sizes
  - L2: 512-entry, 4-way, 4K pages
- Branch fusion



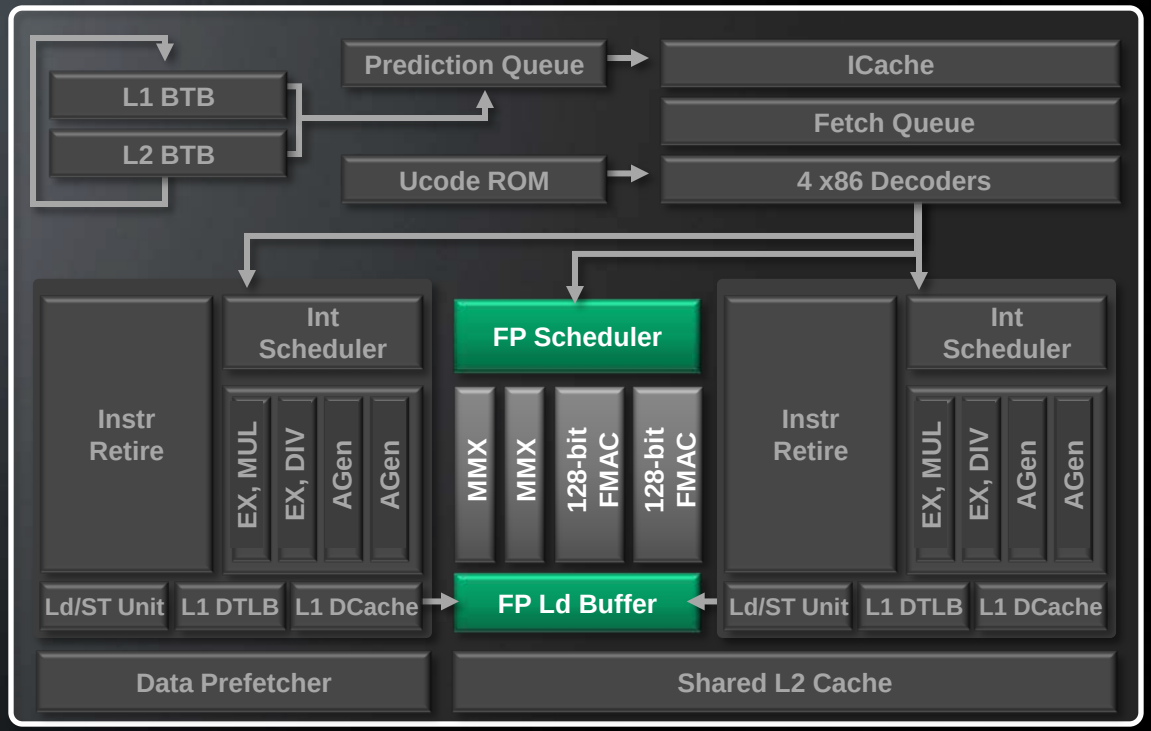
Core Microarchitecture – Dedicated Cores

- Thread retire logic
- PRF-based register renaming
- Unified scheduler per core
- Way-predicted 16K Byte L1 Dcache
- DTLB: 32-entry fully associative
- Fully out-of-order ld/st
  - 2 128-bit loads/cycle
  - 1 128-bit store/cycle
  - 40-entry Load queue
    24-entry Store queue



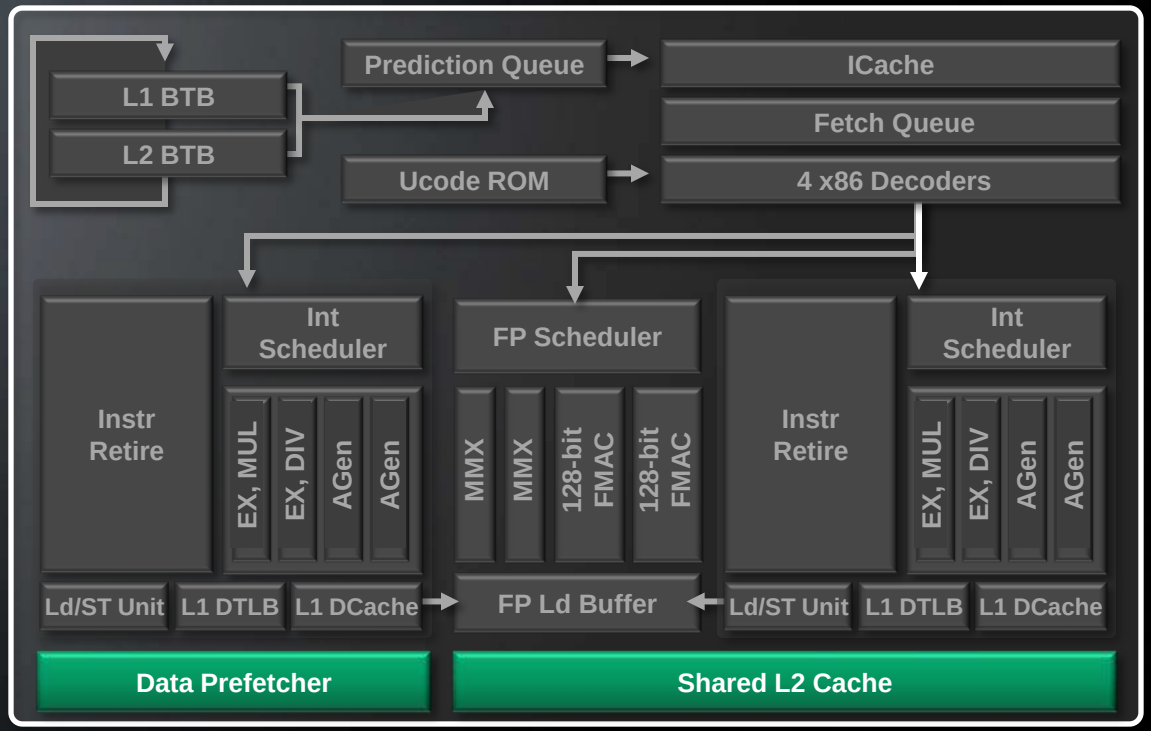
Core Microarchitecture – Shared FPU

- Co-processor organization
- Reports completion back to parent core
- Dual 128-bit FMAC pipes
- Dual 128-bit packed integer pipes
- PRF-based register renaming
- Unified scheduler (for both threads)



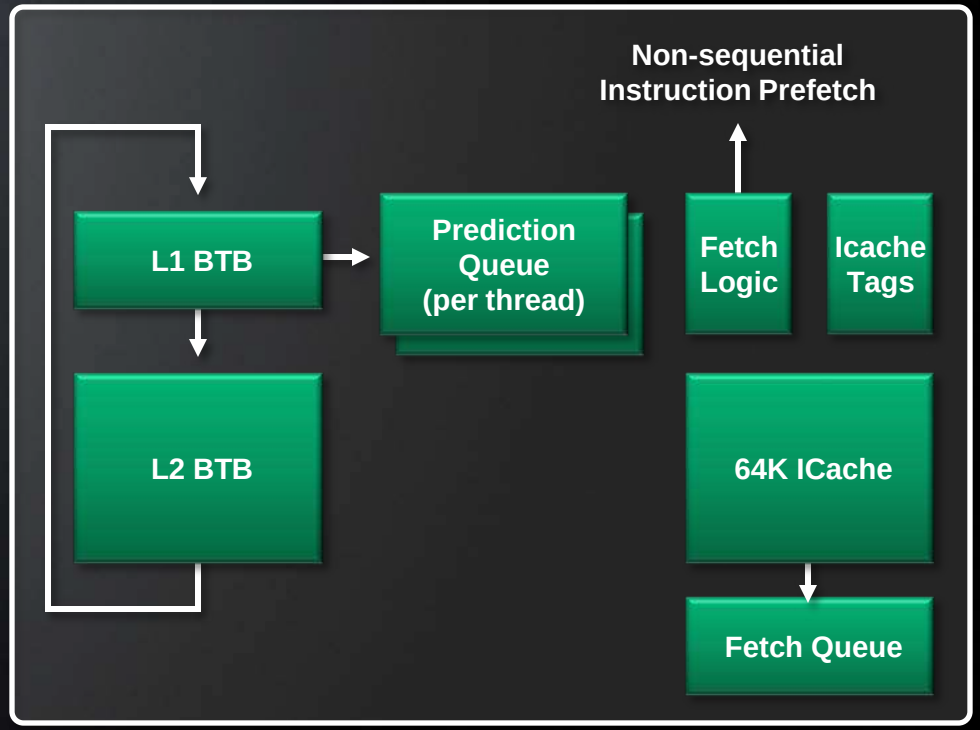
Core Microarchitecture – Shared L2

- 16-way unified L2 cache
- L2 TLB and page walker
  - 1024-entry, 8-way
  - Services both I-side and D-side requests
- Multiple data prefetchers (more on this later)
- 23 outstanding L2 cache misses for memory system concurrency



Prediction-Directed Instruction Prefetch

- Prediction Pipeline is free to run ahead and fill the prediction queue (per thread)
  - Produces sequence of future RIPs
  - Only back-pressure is via full prediction queue stall
- Instruction Fetch pipeline uses future RIPs to check for future misses in the shadow of a demand miss
  - Overlaps instruction miss requests to L2/memory
- Large L1 + L2 BTB capacity captures footprint

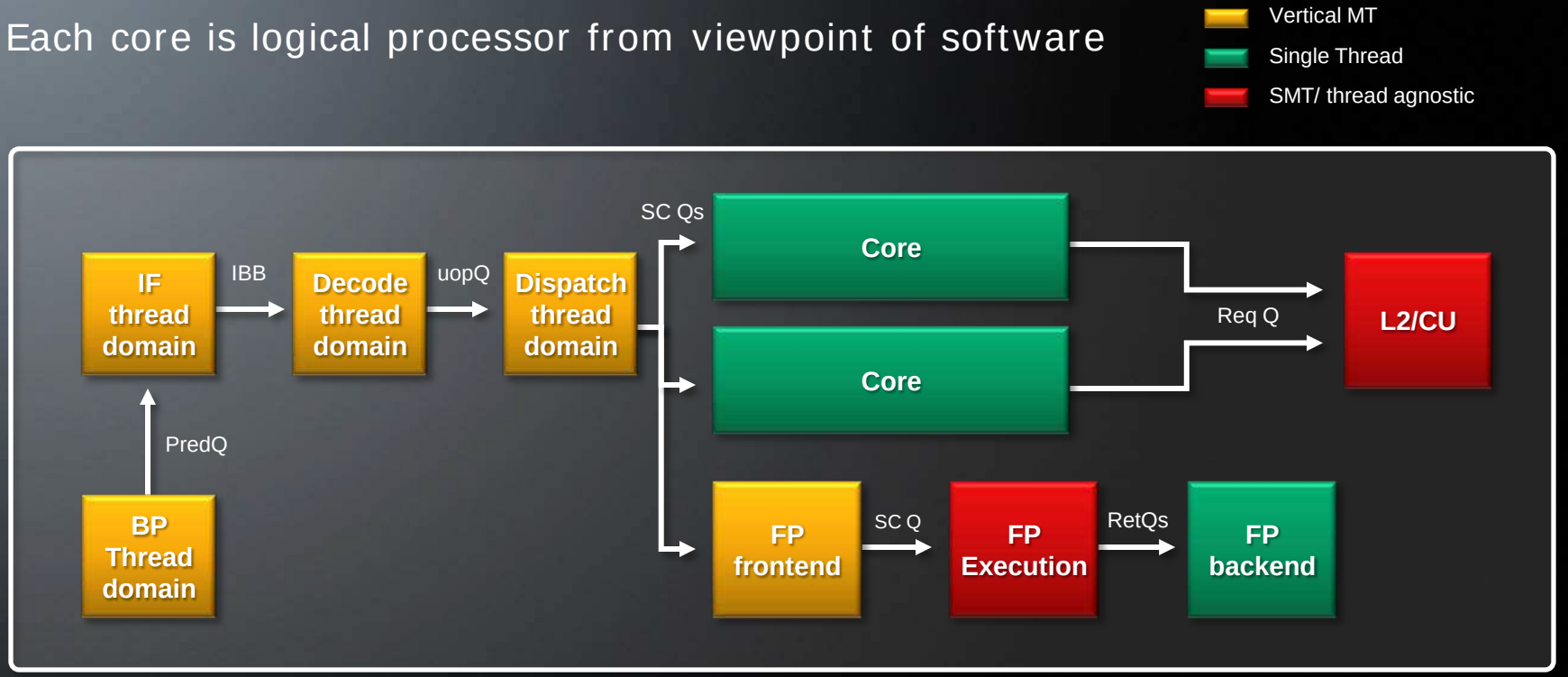


Multiple Data Prefetchers

- Aggressive Stride-based data prefetchers
  - Large number of strides
  - Large stride size
  - L1 and L2 predictors
- Non-strided data prefetcher
  - Captures correlated data accesses that don't have fixed stride relationship
- Robust performance characteristics
  - Applicability to wide range of client and server workloads
  - Backoff/throttling mechanism under heavy demand load

Thread Control and Selection Mechanisms

Each core is logical processor from viewpoint of software

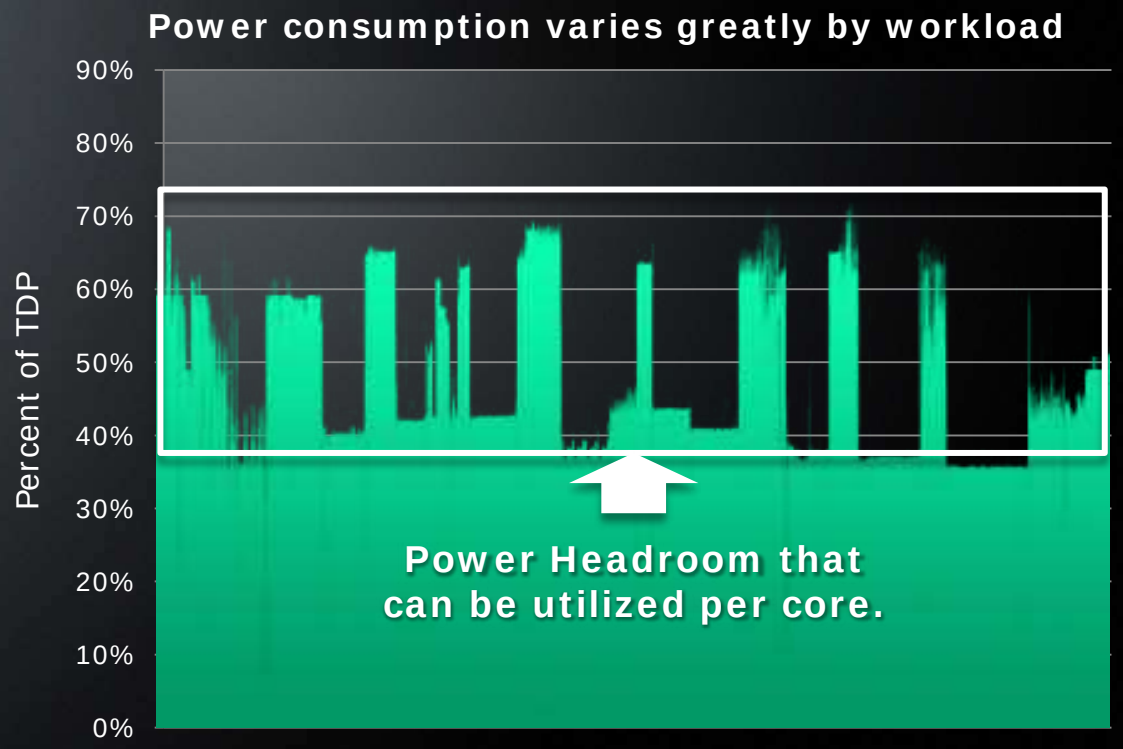


Bulldozer ISA and Feature Extensions

- **Instruction Set Extensions**
  - SSE 4.1 and 4.2
  - AVX
    - 256-bit YMM registers
    - Non-destructive source operand capability
    - AES subset
    - FMAC subset (AMD 4-operand form)
  - XSAVE state space management
  - XOP Instructions
- **Light Weight Profiling (LWP)**
  - Low-overhead user-level profiling
  - Uses XSAVE state space
  - Stores records for configured events
    - Instructions retired
    - Branches retired

Power Efficiency and APM

- Start with inherently power-efficient micro-architecture and implementation:
  - Dynamic sharing of shared resources
  - Minimize data movement
  - Extensive clock and power gating
- Add active management support:
  - Digitally measure activity to estimate power
  - Hardware uses higher frequency when power limit allows
- Support for chip-level core power gating



* Based on internal AMD modeling using benchmark simulations

Concluding Remarks

- Bulldozer at the heart of AMD's 2011 family of mainstream and high-performance processors
- Major investments in
  - Power / Area efficiency
  - New ISA support
  - Scalability of Cores
  - Modular Design Approach
- Significant improvement in Performance/Watt/mm2
  - General purpose throughput
  - Estimated average of 80% of the CMP performance with much less area and power*
  - Single-thread performance

*Based on internal AMD modeling using benchmark simulations

Trademark Attribution

AMD, the AMD Arrow logo and combinations thereof are trademarks of Advanced Micro Devices, Inc. in the United States and/or other jurisdictions. Other names used in this presentation are for identification purposes only and may be trademarks of their respective owners.