DPDK

USERSPACE October 8-9 Dublin

The 7 Deadly Sins of Packet Processing

Venky Venkatesan & Bruce Richardson

The CPU Core

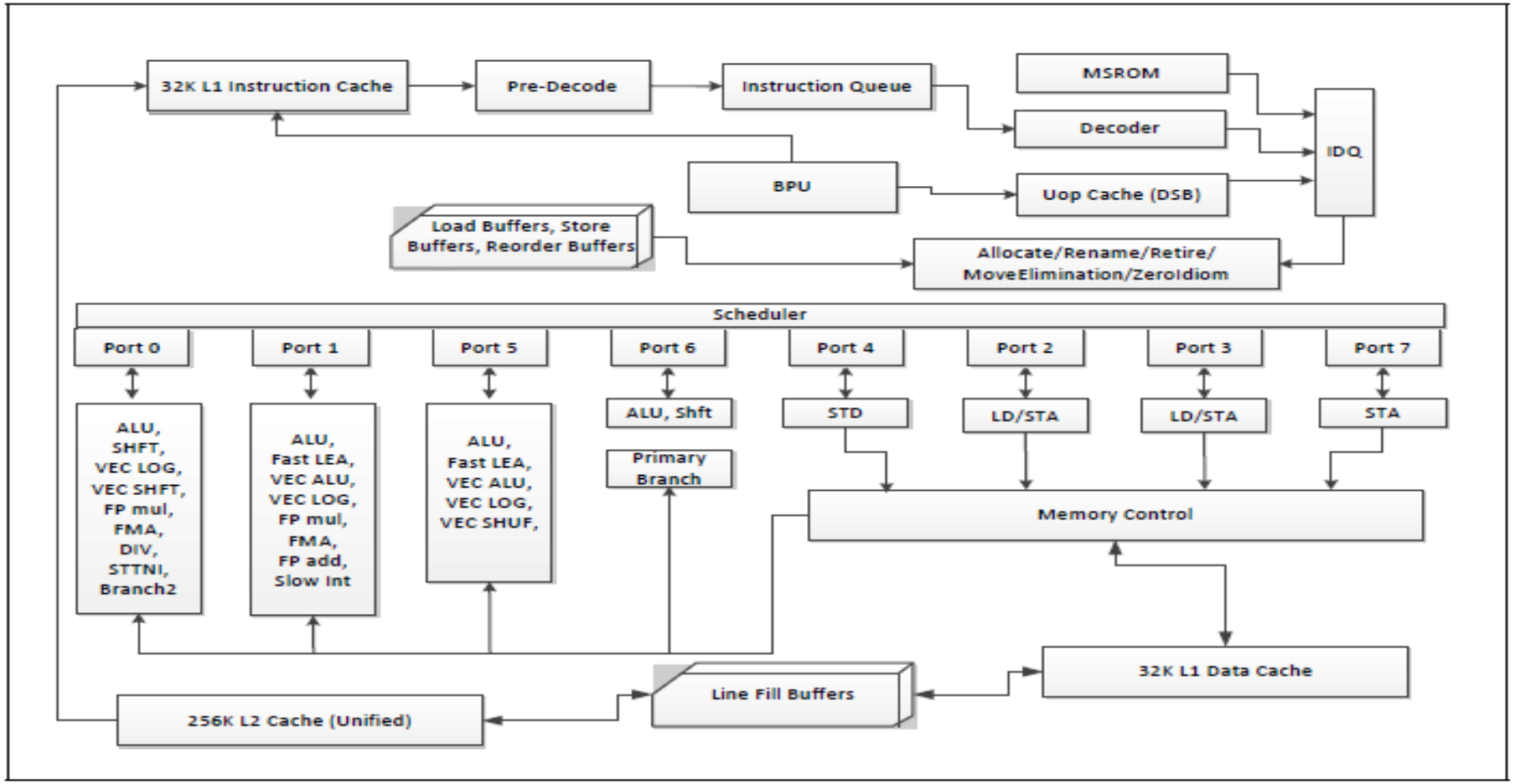

Figure 2-2. CPU Core Pipeline Functionality of the Haswell Microarchitecture

Source: Intel® 64 and IA-32 Architectures: Optimization Reference Manual

DPDK
DATA PLANE DEVELOPMENT KIT

Unpredictable Branches

Joe Thorn / CC-BY-NC-ND [https://www.flickr.com/photos/joethorn/272181350]

Not all branches are bad!

If branch is unpredictable, performance will suffer!

The first time a branch is encountered …

Helping the branch predictor …

- Predicts conditional branches, direct & indirect calls & jumps, returns, loop iterations

➢Guide the compiler with likely()/unlikely() on error cases, and where humans can be certain

  ➢Wrongly structured code can waste fetch/decode bandwidth

➢Let the branch predictor work on runtime data dependent branches

➢Inline … gives the BTB more context, but bloats code

Source: Intel® 64 and IA-32 Architectures: Optimization Reference Manual

Sharing Cache Lines

Pierre-Yves Beaudouin / Wikimedia Commons / CC-BY-SA-3.0

Need to avoid the latency of having cache-lines ping-pong between different cores on a system.

Some basics …

	Sandy Bridge Ivy Bridge	Haswell	Skylake
L1 data access (cycles)	4	4	4
L1 Peak Bandwidth (bytes/cycle)	2x16	2x32 load 1x32 store	2x32 load 1x32 store
L2 data Access (cycles)	12	11	12
L2 peak bandwidth (bytes/cycle)	1x32	64	64
Shared L3 Access (cycles)	26-31	34	44
L3 peak bandwidth (bytes/cycle)	32	-	32
Data hit in L2 or L1D Dcache of another core	43 – clean hit 60 – modified hit		

- BUT memory is ~70+ ns away (i.e. 2.0 GHz = 140+ cycles)

Source: Intel® 64 and IA-32 Architectures: Optimization Reference Manual

Incorrect Prefetching

A cache miss can use up your full packet budget, so make sure you pull in your data before you need it!

Prefetching …

- Hardware pre-fetcher will try to predict … and will fetch data that isn't needed (adds overhead)
- With packed data structures, it sometimes fetches data that an other core uses (inadvertently sharing cache lines)
- But in most cases, hardware prefetchers hugely improve application performance

Per-Packet Operations

Any overhead gets magnified when done per-packet.

DPDK
DATA PLANE DEVELOPMENT KIT

Some of these aren’t quite obvious …

- Memory mapped I/O & UC (Uncacheable) operations
- Atomic increment/decrement/Compare-swap
- Ring enq/deq (especially those that use atomics)
- Locks

Incorrect Inlining

Trade-off: function calls have an overhead, but add flexibility

In-lining …

- Eliminates parameter passing overhead
- Increases optimization opportunities for the compiler
- More specific branch prediction context
- Mis-predicted branch penalties in a small function are higher e.g. if a branch misprediction results in a return being prematurely taken

Bad Data Structures

Terence Ong / CC-BY-2.5 [from: https://en.wikipedia.org/wiki/PET_bottle_recycling#/media/File:NEA_recycling_bins,_Orchard_Road.JPG]

Remember to separate per-lcore data onto different cachelines

DPDK
DATA PLANE DEVELOPMENT KIT

Making System Calls

Like flushing away cycles….