Hardware Performance Counters Internals-Part1

By Mohit Kumar

Performance Counter on Intel Xeon

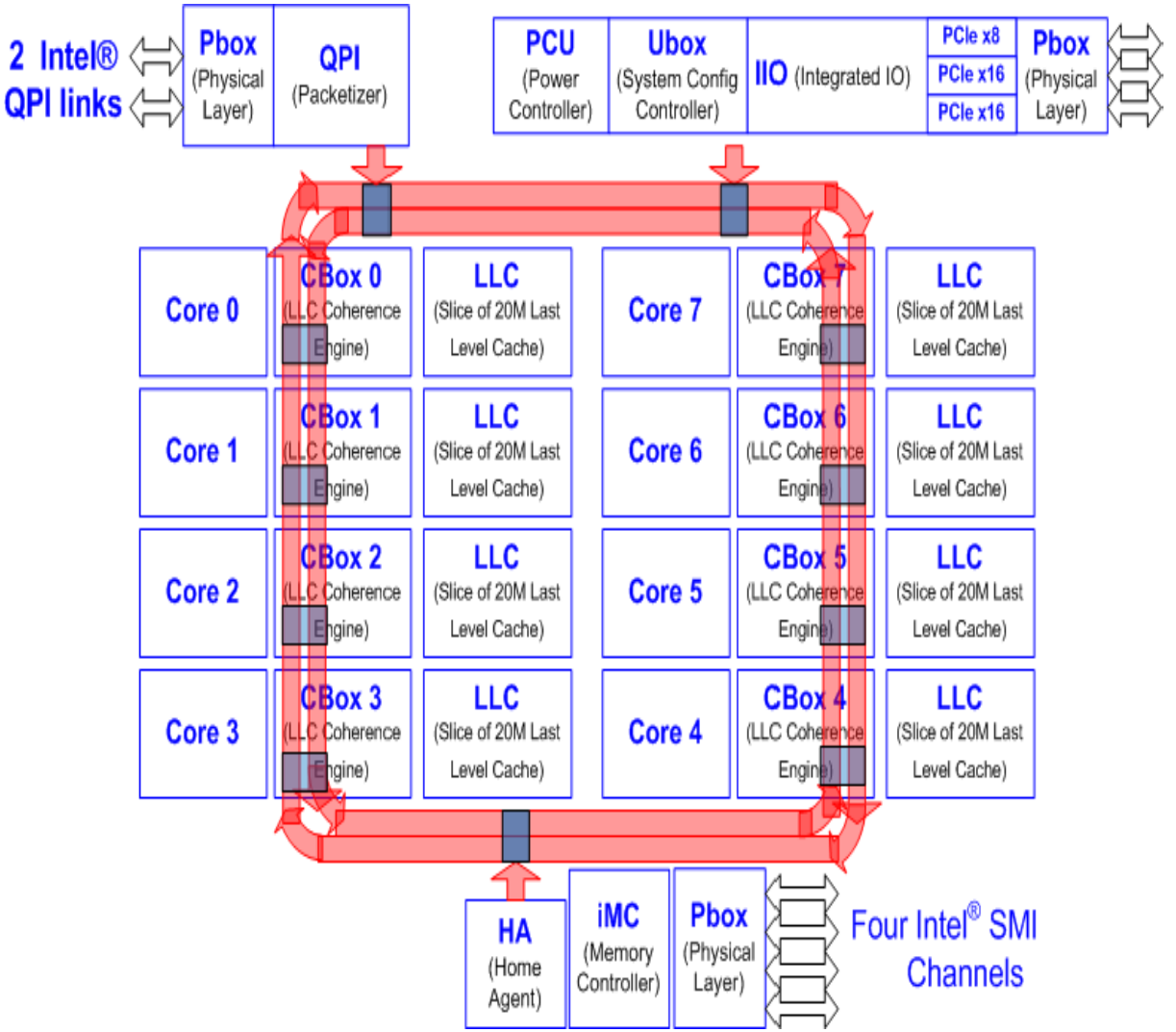
"**Uncore**"

per socket resources

"**Box**"

modular uncore unit

Intel Xeon E5-2600 Family (Sandy Bridge EP)

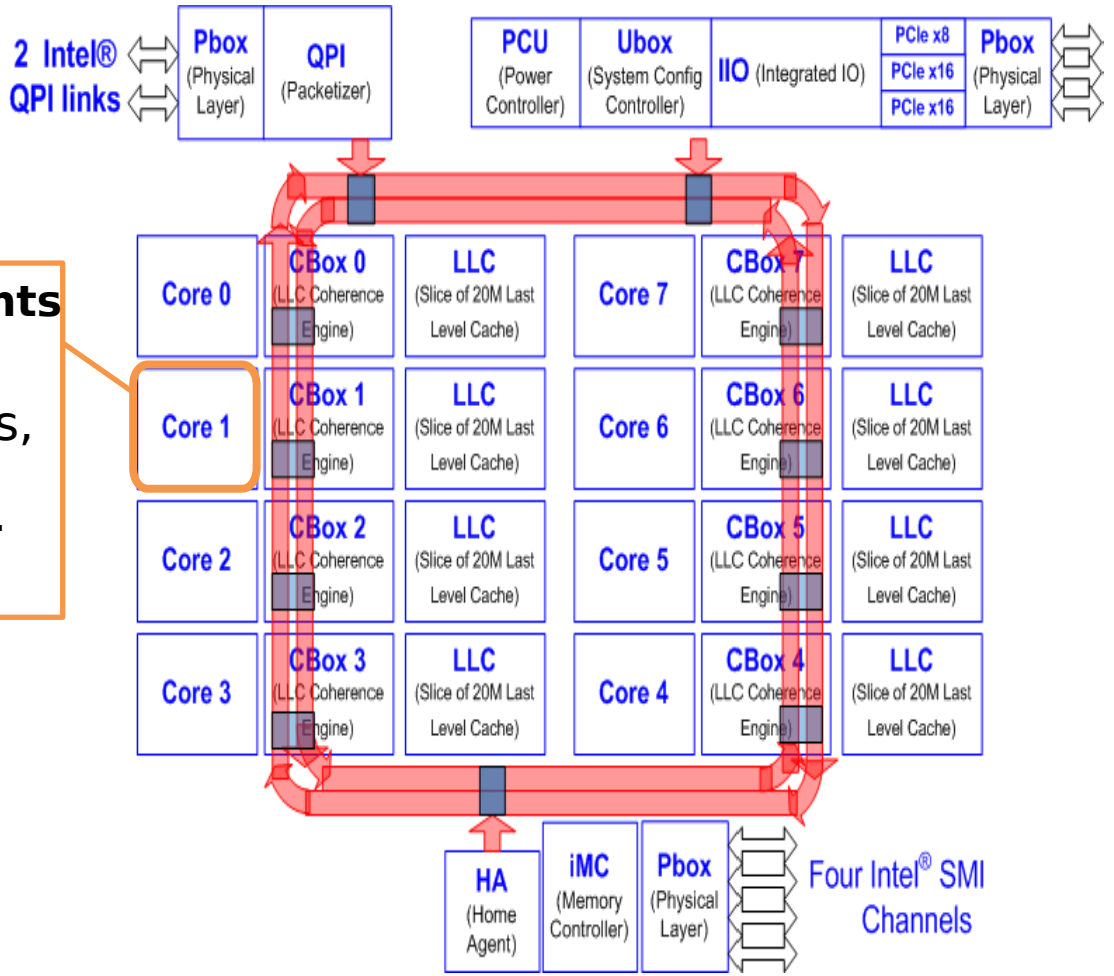


Core Performance Counter on Intel Xeon

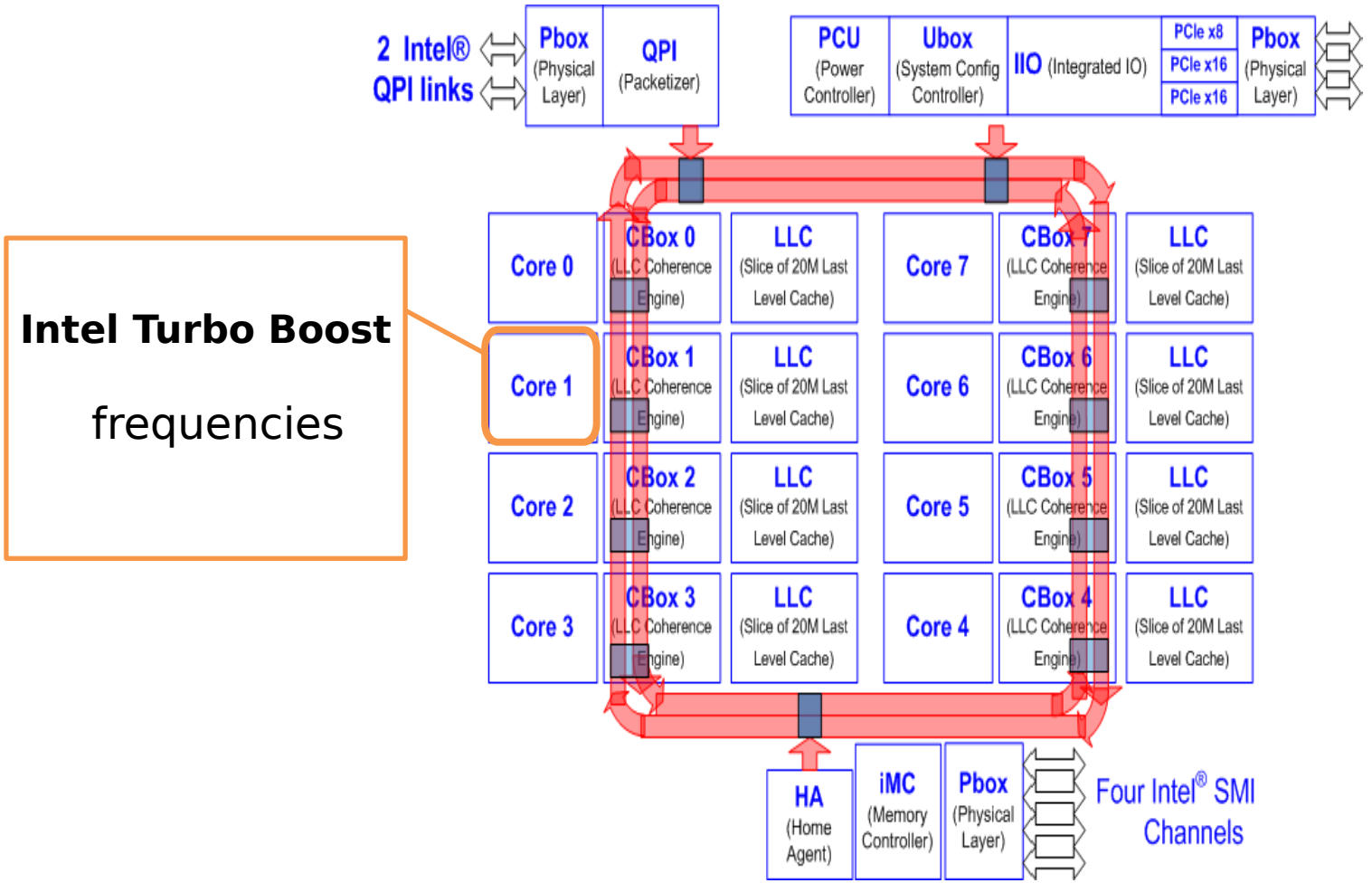
Instructions counts

total, branches,

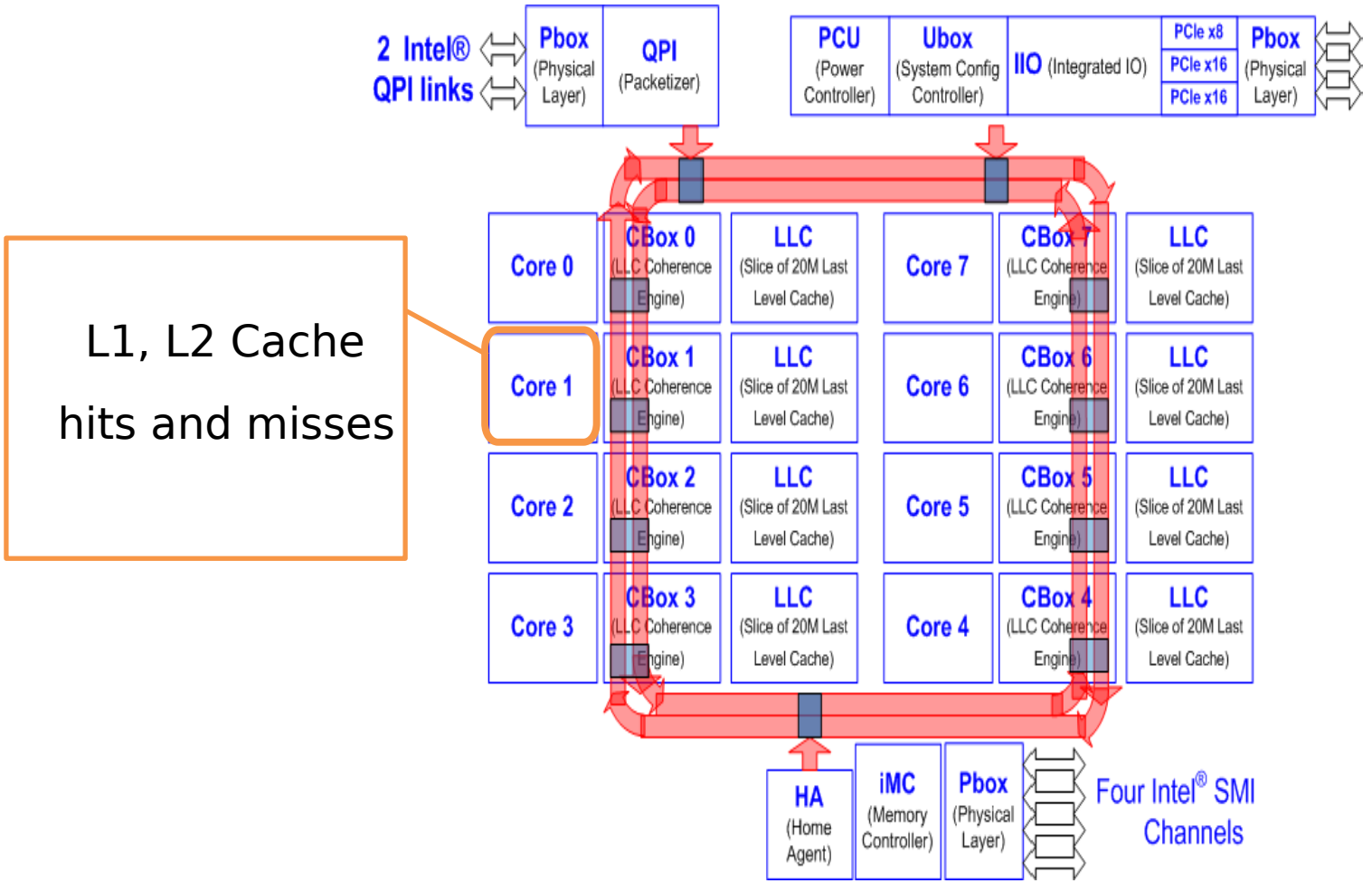
arithmetic, ...



Uncore Performance Counter on Intel Xeon



Uncore Performance Counter on Intel Xeon

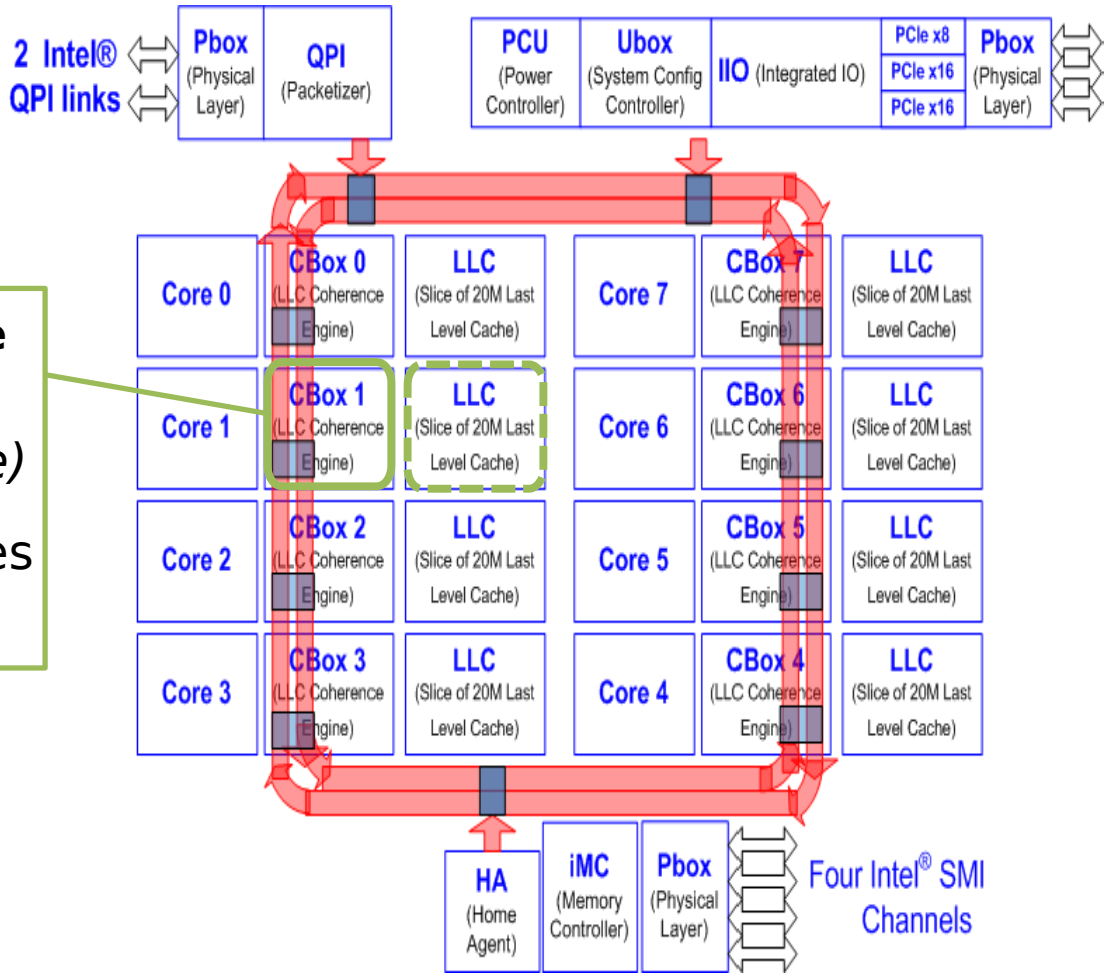


Uncore Performance Counter on Intel Xeon

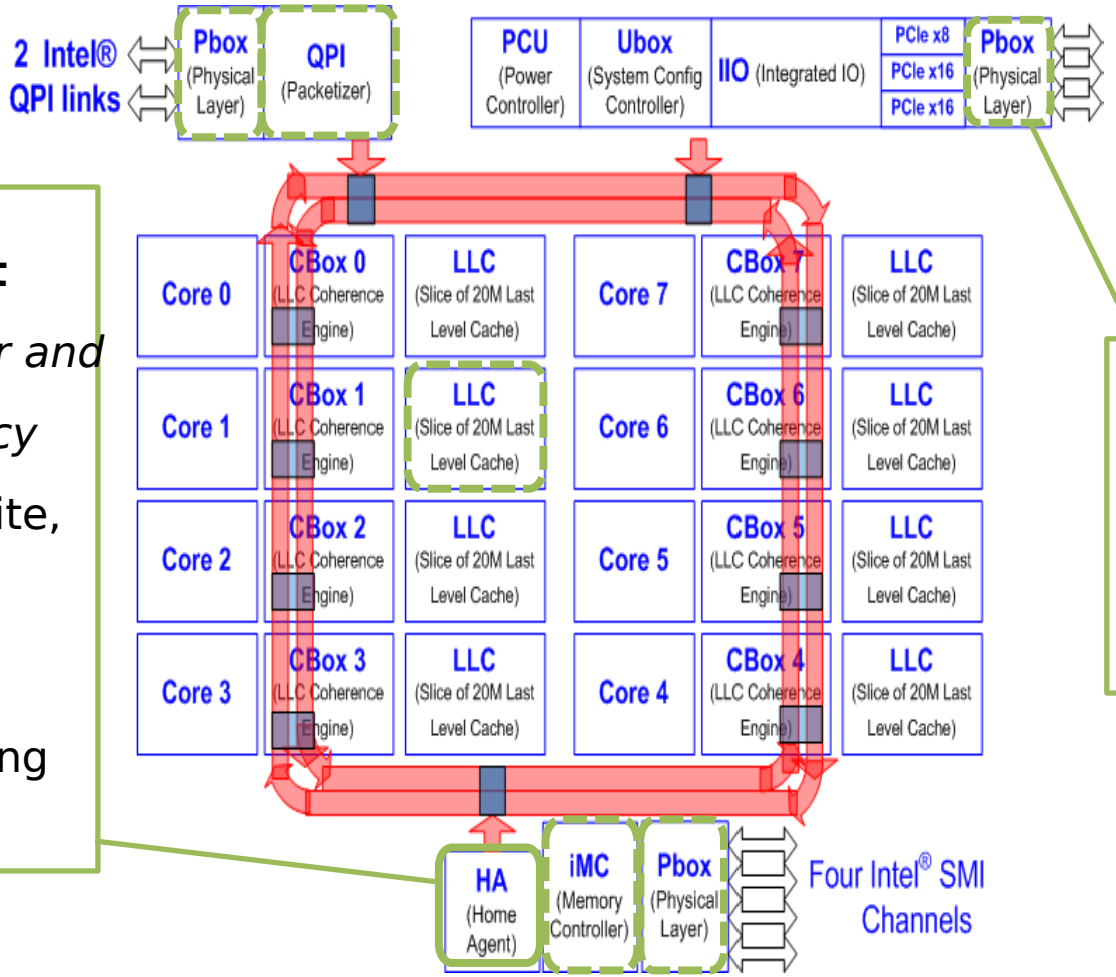
Last Level Cache

(shared L3 cache)

hits and misses



Uncore Performance Counter on Intel Xeo



Home Agent

memory controller and cache coherency

memory read/write, local/remote, conflicts, directory/snooping

Pbox

Physical connection

between cores or sockets

Uncore Performance Counter on Intel Xeo

Integrated Memory Controller

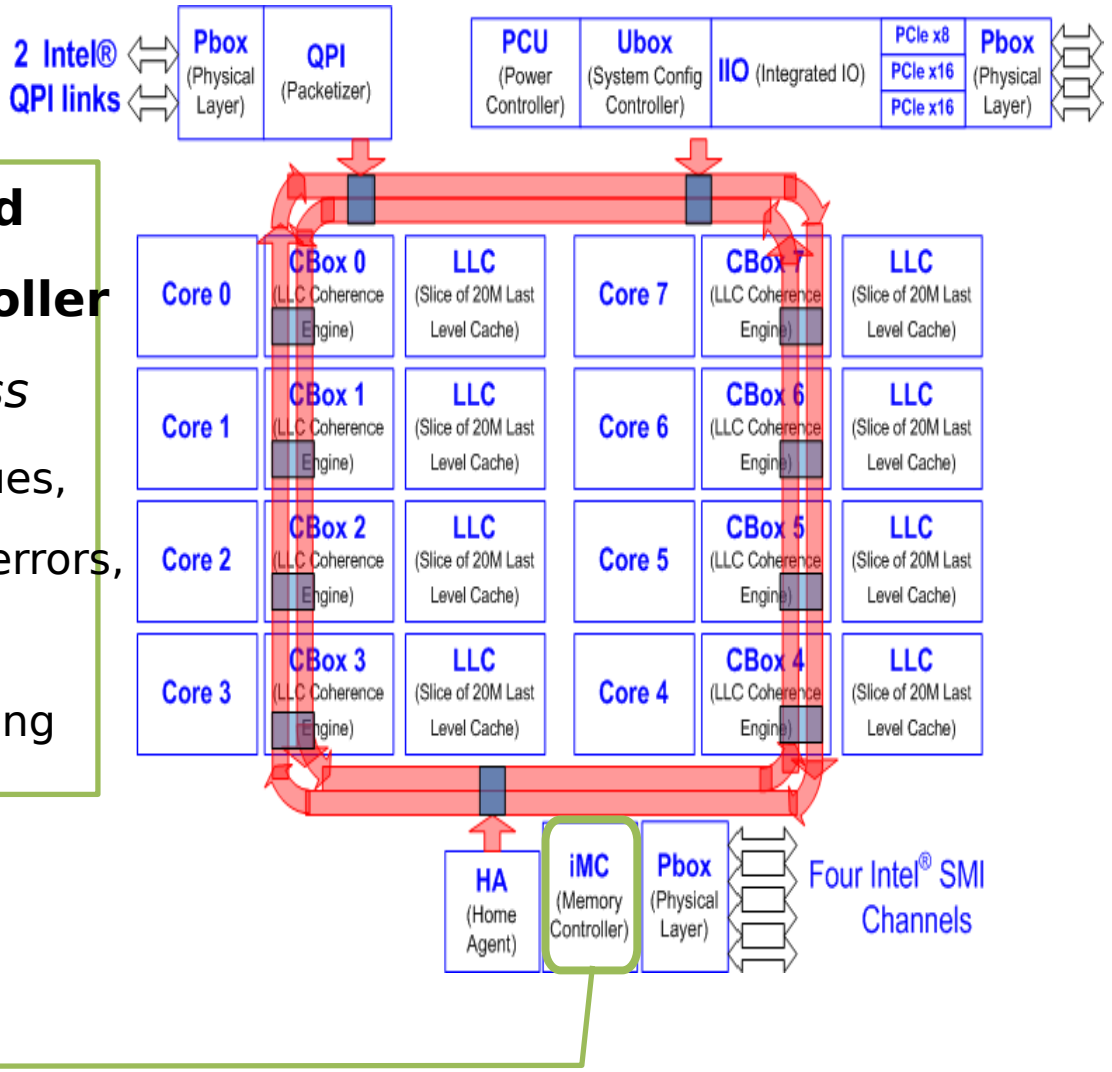
DRAM access

read/write queues,

ECC correctable errors,

refreshes,

thermal throttling



Uncore Performance Counter on Intel Xeo

QPI

Ring ↔ Link Layer

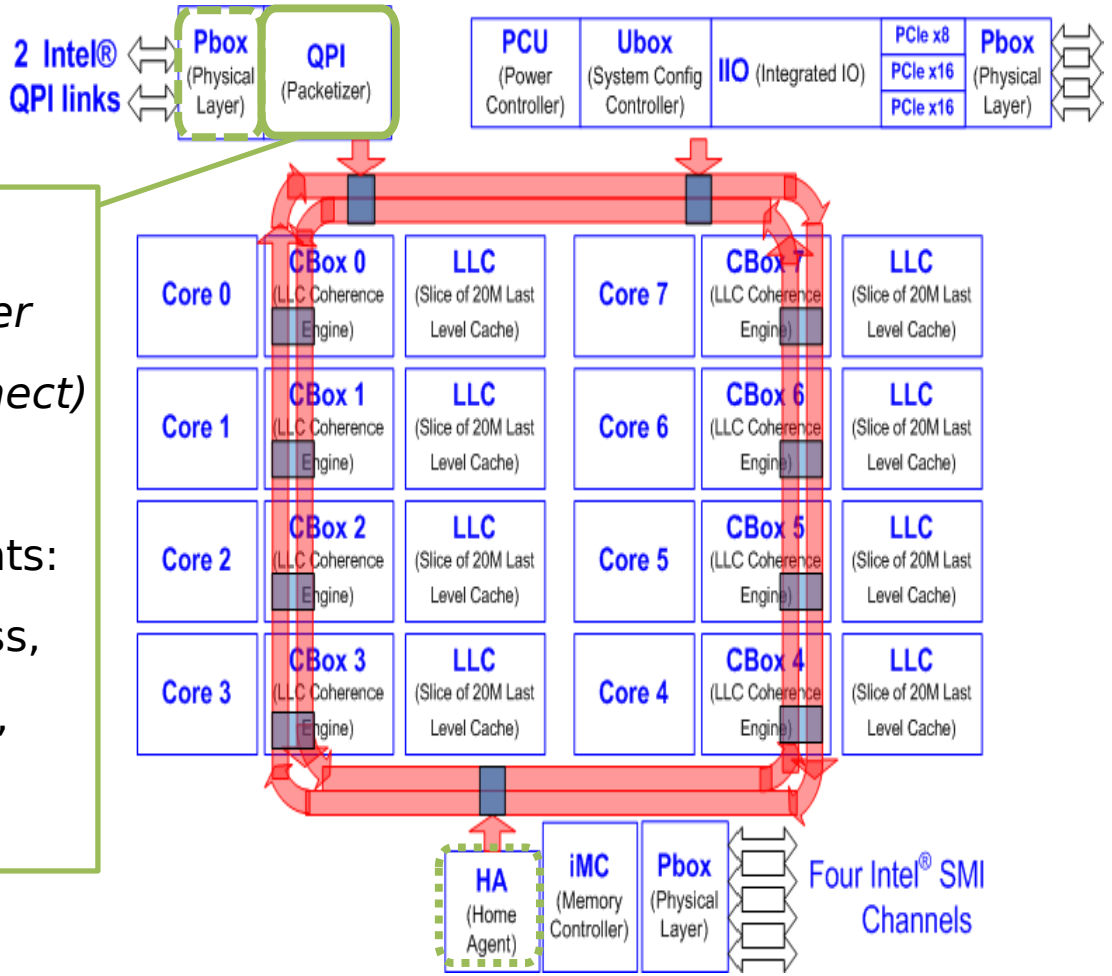
(socket interconnect)

Filter event counts:

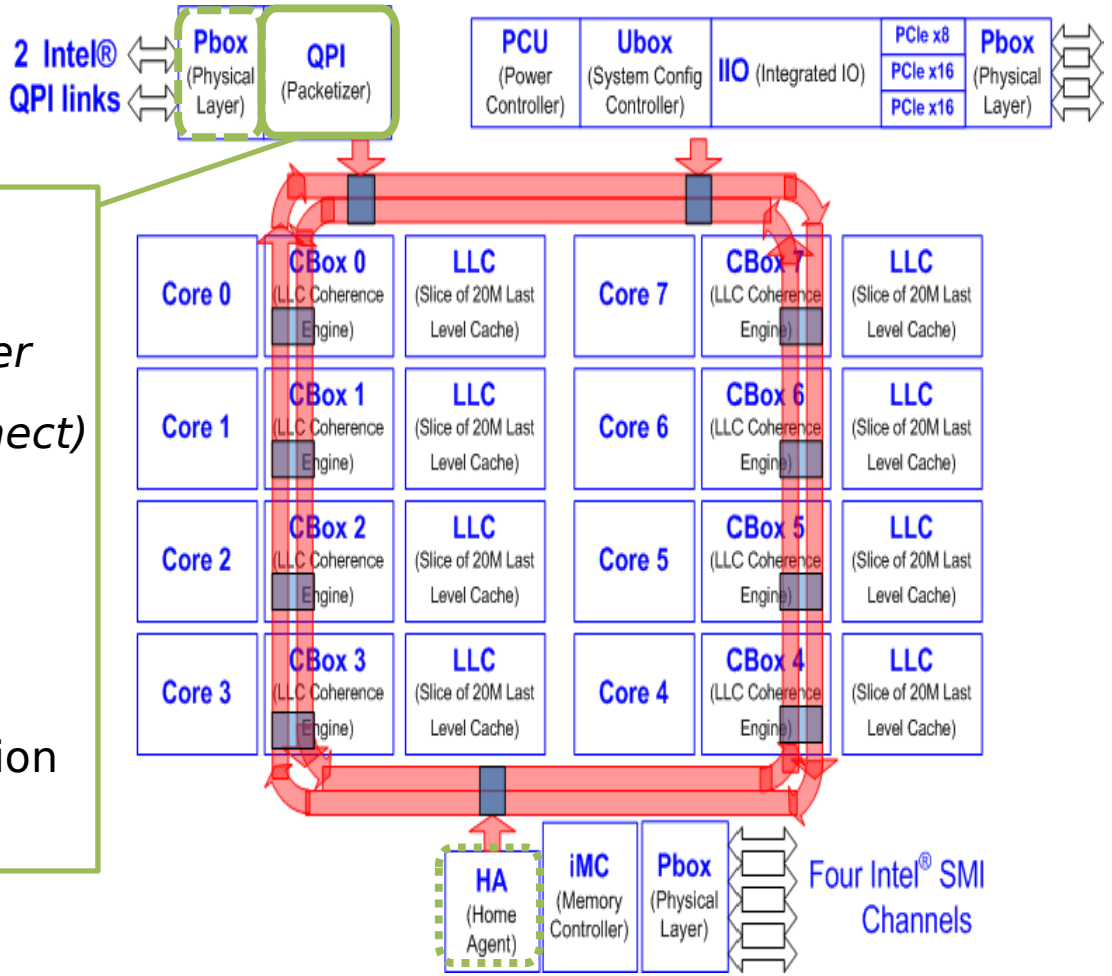
physical address,

Home Node ID,

instruction



Uncore Performance Counter on Intel Xeo



QPI

Ring ↔ Link Layer

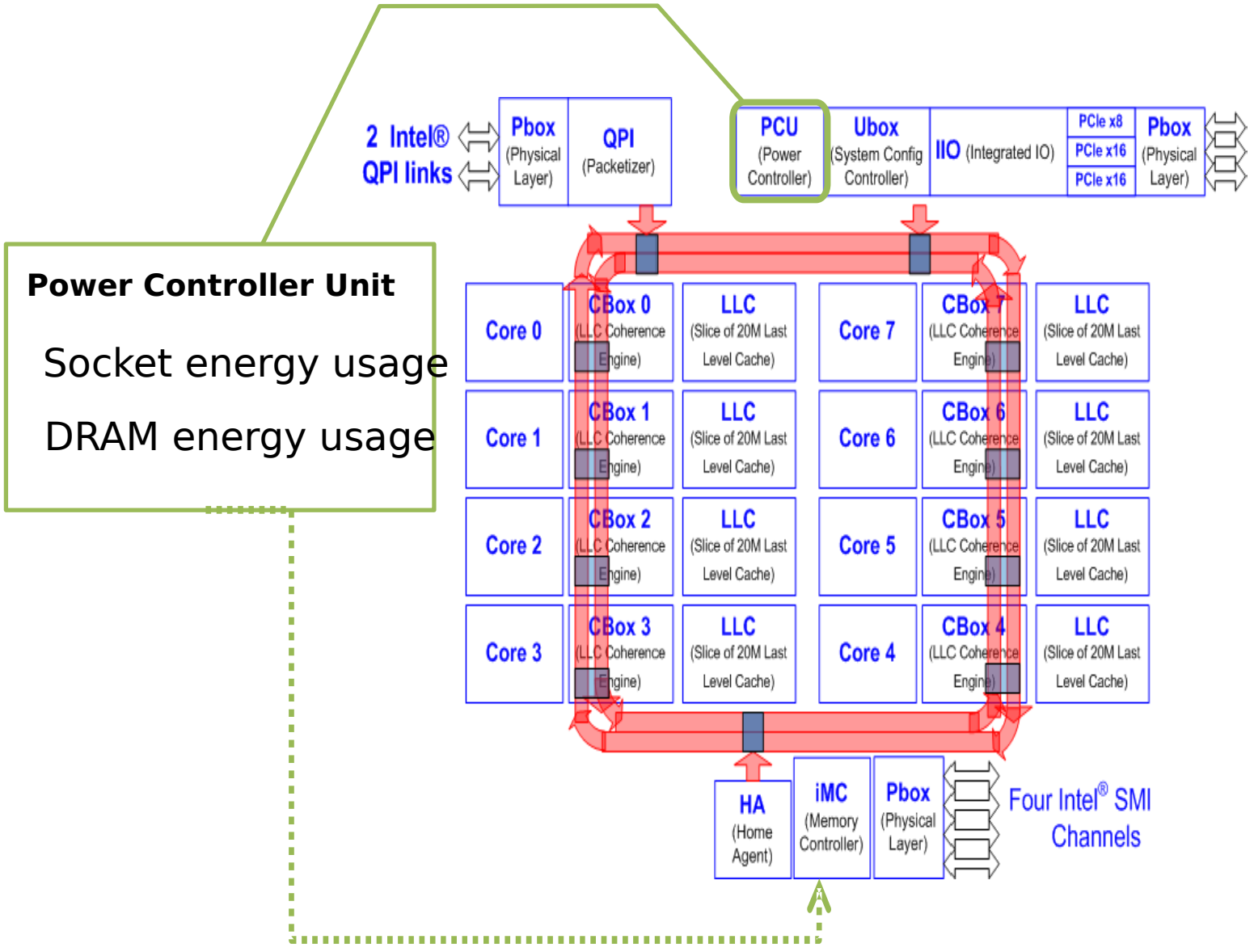
(socket interconnect)

link speed,

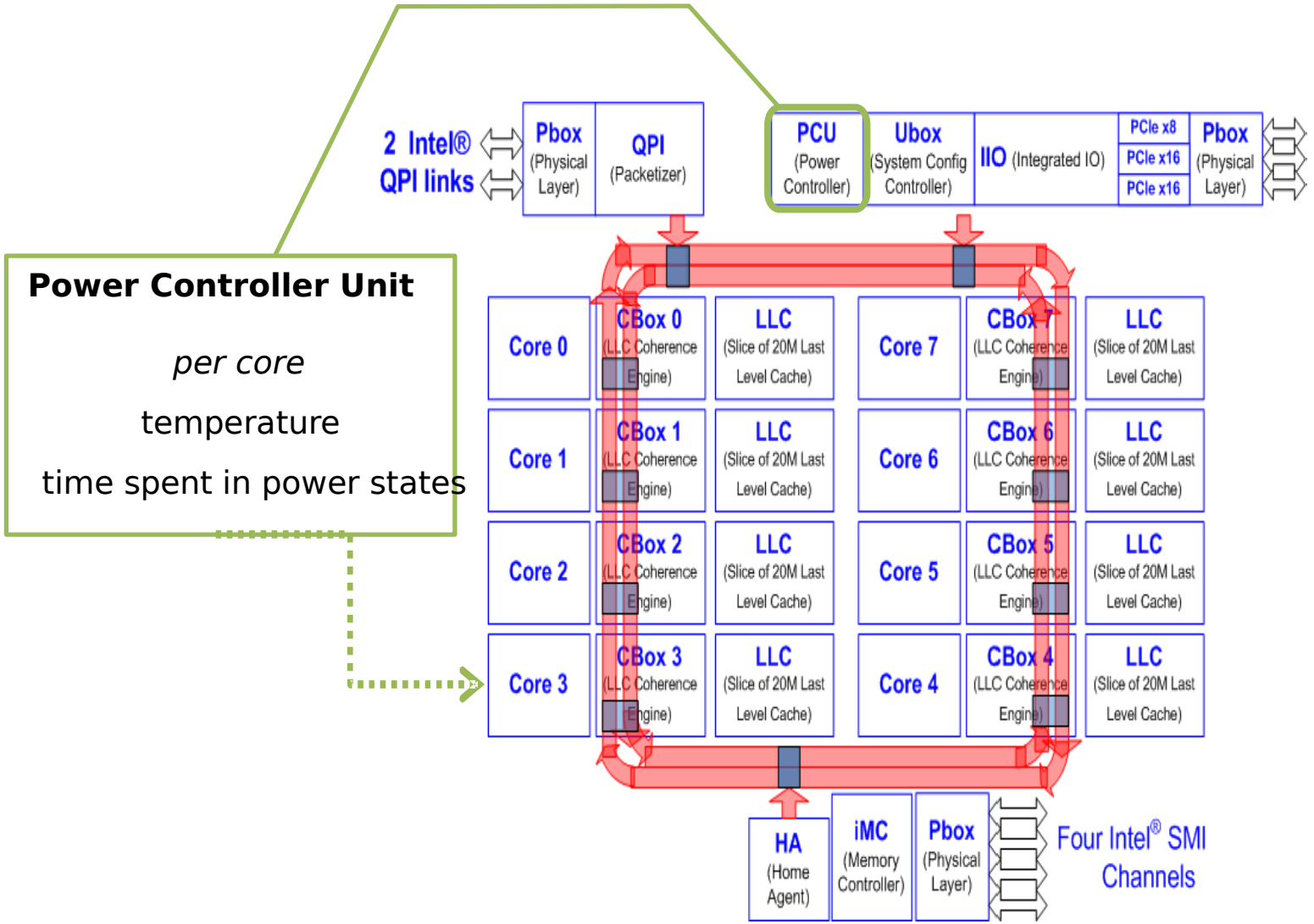
transfers,

total link utilization

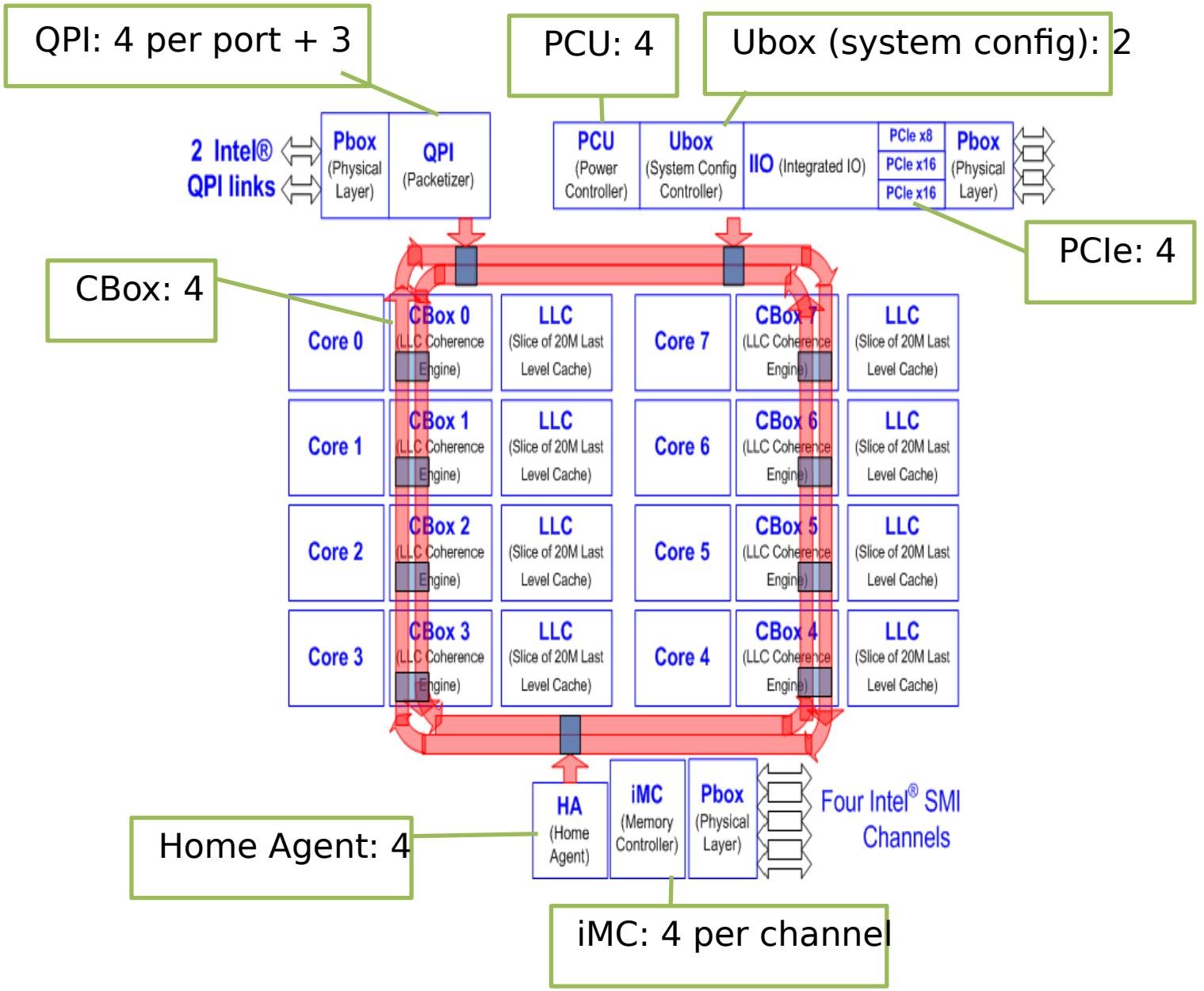
Uncore Performance Counter on Intel Xeo



Uncore Performance Counter on Intel Xeo



Performance Counters per Box



Performance Counters per Box

ubuntu-numa0101.fsoc: Linux 3.13, 2x Intel Xeon E5-2620

- 57 (+x) Performance Monitoring Units per socket
- 634 countable events
- Allowing comprehensive runtime analysis
- Mostly focused on a few context specific events

HPC: How to?

Use Profilers first

- Using automated tests
- Partly implemented with Performance Counter
- Optimize as much as possible

Low level analysis with Performance Counters

- Optimize problematic code sections
- Platform specific optimization
- Benefit from minimal overhead

HPC: Using

Tools and C++ programming interface

Full support for Intel core/uncore events

Supports newer Intel Xeon, Core i, Atom

- Uncore mainly available on server platforms

HPC:Tools and Libraries

- Inspect Raw Counters
- Linux Perf
- PAPI
- Perfmon/libpfm
- Intel Performance Counter Monitor

HPC:Raw Counters

- General
  - /proc/stat
  - /proc/meminfo
  - /proc/interrupts
- Process specific
  - /proc/[pid]/statm – process memory
  - /proc/[pid]/stat – process execution times
  - /proc/[pid]/status – human readable
- Device specific
  - /sys/block/[dev]/stat
  - /proc/dev/net
- Hardware
  - smartctl

	CPU0	CPU1	CPU2	CPU3	CPU4
0:	60	0	0	0	
3:	584	0	0	0	
4:	12	0	0	0	
8:	1	0	0	0	
9:	2	0	0	0	
10:	248	0	0	0	
22:	129	0	0	0	
23:	289	0	0	0	
104:	0	0	0	0	
105:	0	0	0	0	
106:	0	0	0	0	
107:	0	0	0	0	
108:	0	0	0	0	
109:	0	0	0	0	
110:	0	0	0	0	
111:	0	0	0	0	
112:	0	0	0	0	
113:	12497	0	301	0	1871
114:	2	0	0	0	
115:	2	0	0	0	
116:	2	0	0	0	

/proc/interrupts

HPC:Raw Counters

/sys/block/[dev]/stat

Name	units	description
read I/Os	requests	number of read I/Os processed
read merges	requests	number of read I/Os merged with in-queue I/O
read sectors	sectors	number of sectors read
read ticks	milliseconds	total wait time for read requests
write I/Os	requests	number of write I/Os processed
write merges	requests	number of write I/Os merged with in-queue I/O
write sectors	sectors	number of sectors written
write ticks	milliseconds	total wait time for write requests
in_flight	requests	number of I/Os currently in flight
io_ticks	milliseconds	total time this block device has been active
time_in_queue	milliseconds	total wait time for all requests

HPC:What is Perf?

- Part of Linux since v2.6.31 (2009)
  - before: patch and compile your kernel
- Command-line tool “perf” (userspace)
  - Debian/Ubuntu-Package “linux-tools”
- perf list
  - Detects supported events
  - But no support for finding relevant events
- perf stat -e [eventName] command
  - Run command and count eventName

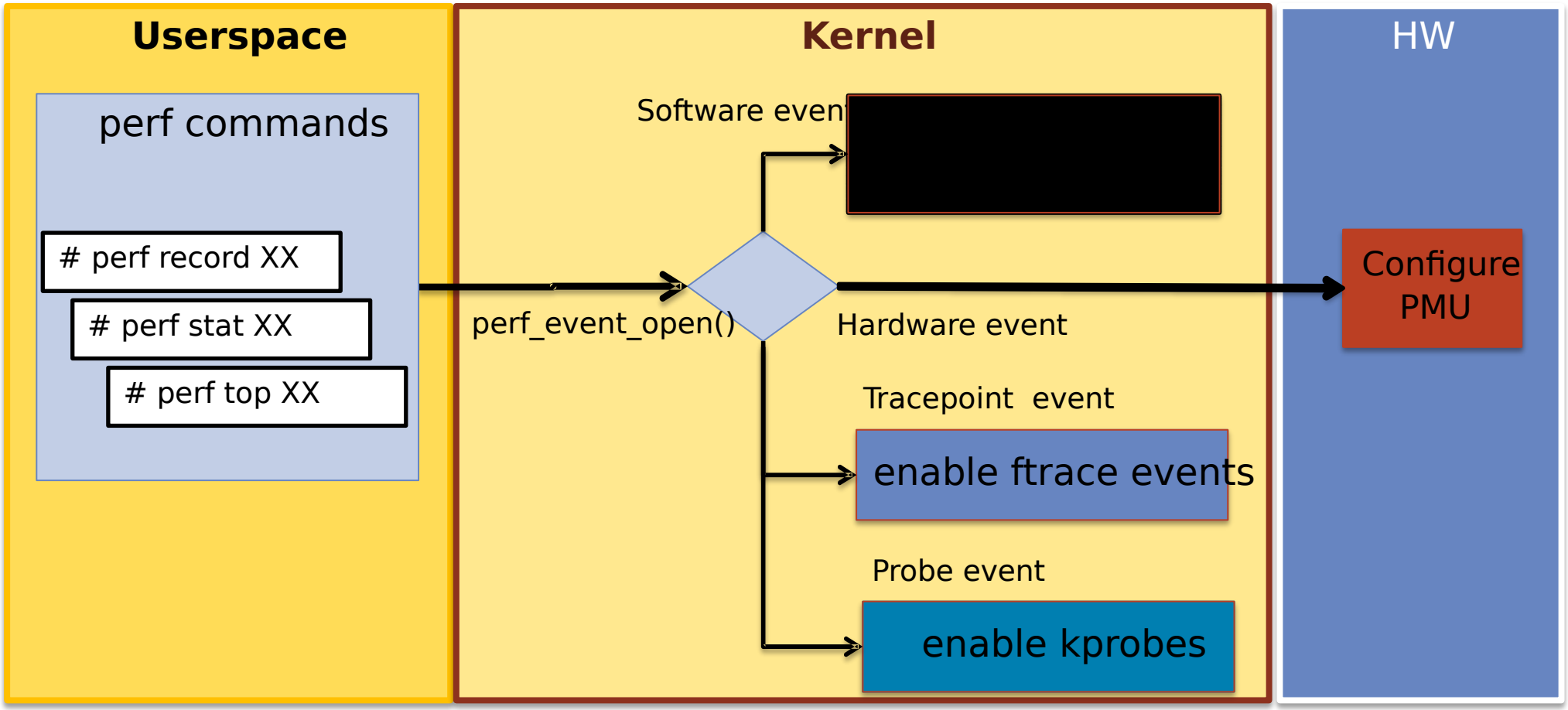
HPC:What is Perf?

- An integrated performance analysis tool on Linux kernel
- Basically a profiler, but its tracer capabilities is enhanced,and becomes an all-round performance analysis tool.
  - Events
    - hardwre event
    - swevent
- tracing
  - trace point
  - probe point
- Samples
  - Events related information
  - IP
  - CALLCHAIN
  - STACK
  - TIME

HPC:What is Perf?:Events

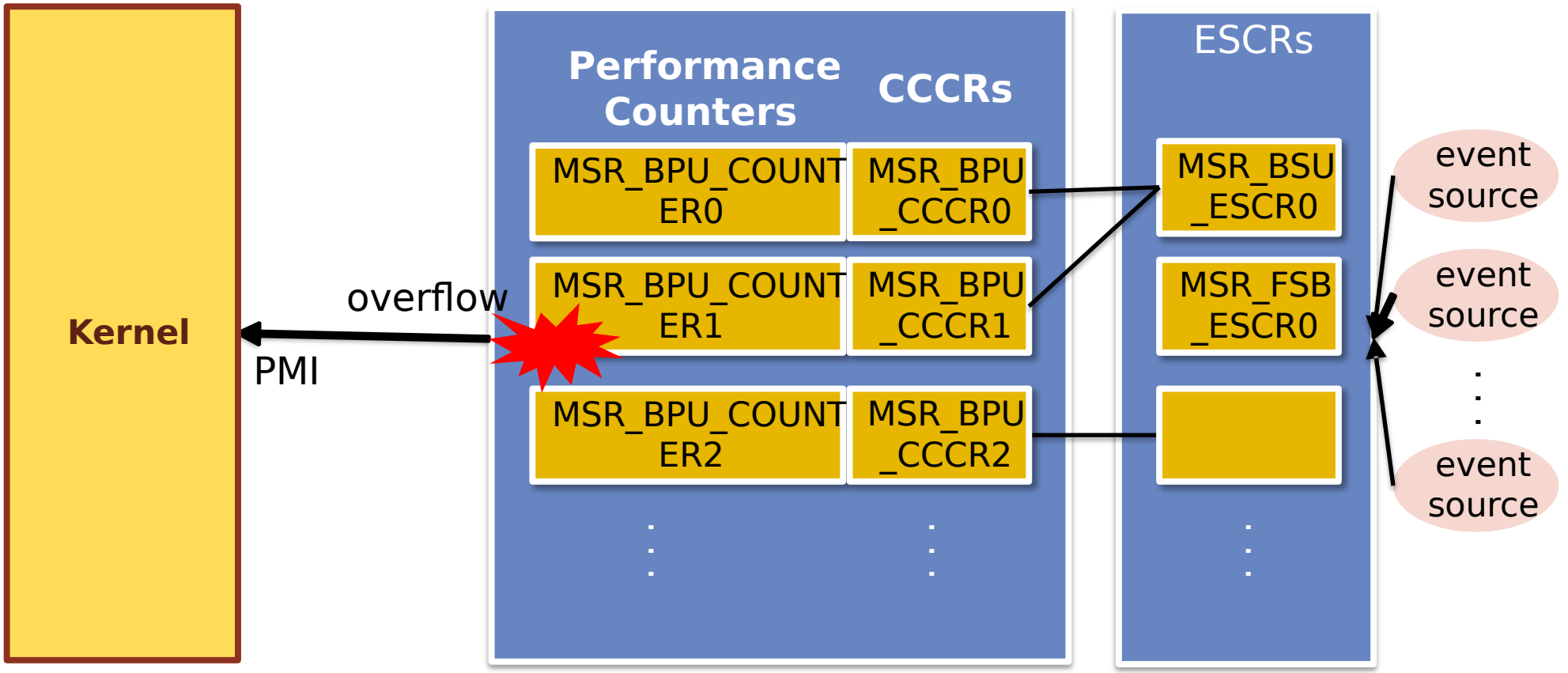
Categories	Descriptions	Examples
Hardware events	Event measurable by PMU of processor. Data can be collected without the overhead though the contents is dependent on type of processors.	Cpu-cycles and cache-misses , etc.
Hardware cache events		L1-dcache-load-misses and branch-loads, etc.
Software events	Event measurable by kernel counter.s	Cpu-clock and page-faults, etc.
Tracepoint events	Code locations built into the kernel where trace information can be collected.	Sched:sched_stat_runtime and syscalls:sys_enter_socket, etc.
Probe events	User-defined events dynamically inserted into the kernel.	-

HPC:What is Perf?:perf event registration



HPC:What is Perf?:perf event registration

- Collection of sample data on hardware events are mostly done by hardware. A pair of Performance counter and CCCR(configuration control registers (CCCR)) records data at events selected by an ESCR(event select control register). Only when a Performance Counter overflows when the kernel receives a PMI interrupt and copies information from the registers.



HPC:What is Perf?:perf event registration

- The perf module collects samples when an event like HWevent occurs. Data to be collected is specified as sample types when a user invokes the perf command.
- They include IP (Instruction Pointer), user or kernel stack, timer and mostly taken from hardware. Samples collected are written to memory area mapped by the perf command so that it can retrieve them without kernel-to-user copying.