



Intro to DPDK & HW

Network Platforms Group



Legal Disclaimer

INFORMATION IN THIS DOCUMENT IS PROVIDED IN CONNECTION WITH INTEL PRODUCTS. NO LICENSE, EXPRESS OR IMPLIED, BY ESTOPPEL OR OTHERWISE, TO ANY INTELLECTUAL PROPERTY RIGHTS IS GRANTED BY THIS DOCUMENT. EXCEPT AS PROVIDED IN INTEL'S TERMS AND CONDITIONS OF SALE FOR SUCH PRODUCTS, INTEL ASSUMES NO LIABILITY WHATSOEVER AND INTEL DISCLAIMS ANY EXPRESS OR IMPLIED WARRANTY, RELATING TO SALE AND/OR USE OF INTEL PRODUCTS INCLUDING LIABILITY OR WARRANTIES RELATING TO FITNESS FOR A PARTICULAR PURPOSE, MERCHANTABILITY, OR INFRINGEMENT OF ANY PATENT, COPYRIGHT OR OTHER INTELLECTUAL PROPERTY RIGHT.

A "Mission Critical Application" is any application in which failure of the Intel Product could result, directly or indirectly, in personal injury or death. SHOULD YOU PURCHASE OR USE INTEL'S PRODUCTS FOR ANY SUCH MISSION CRITICAL APPLICATION, YOU SHALL INDEMNIFY AND HOLD INTEL AND ITS SUBSIDIARIES, SUBCONTRACTORS AND AFFILIATES, AND THE DIRECTORS, OFFICERS, AND EMPLOYEES OF EACH, HARMLESS AGAINST ALL CLAIMS COSTS, DAMAGES, AND EXPENSES AND REASONABLE ATTORNEYS' FEES ARISING OUT OF, DIRECTLY OR INDIRECTLY, ANY CLAIM OF PRODUCT LIABILITY, PERSONAL INJURY, OR DEATH ARISING IN ANY WAY OUT OF SUCH MISSION CRITICAL APPLICATION, WHETHER OR NOT INTEL OR ITS SUBCONTRACTOR WAS NEGLIGENT IN THE DESIGN, MANUFACTURE, OR WARNING OF THE INTEL PRODUCT OR ANY OF ITS PARTS.

Intel may make changes to specifications and product descriptions at any time, without notice. Designers must not rely on the absence or characteristics of any features or instructions marked "reserved" or "undefined". Intel reserves these for future definition and shall have no responsibility whatsoever for conflicts or incompatibilities arising from future changes to them. The information here is subject to change without notice. Do not finalize a design with this information.

The products described in this document may contain design defects or errors known as errata which may cause the product to deviate from published specifications. Current characterized errata are available on request.

Contact your local Intel sales office or your distributor to obtain the latest specifications and before placing your product order.

Copies of documents which have an order number and are referenced in this document, or other Intel literature, may be obtained by calling 1-800-548-4725, or go to: <http://www.intel.com/design/literature.htm> Performance tests and ratings are measured using specific computer systems and/or components and reflect the approximate performance of Intel products as measured by those tests. Any difference in system hardware or software design or configuration may affect actual performance. Buyers should consult other sources of information to evaluate the performance of systems or components they are considering purchasing. For more information on performance tests and on the performance of Intel products, visit Intel Performance Benchmark Limitations

All products, computer systems, dates and figures specified are preliminary based on current expectations, and are subject to change without notice.

Celeron, Intel, Intel logo, Intel Core, Intel Inside, Intel Inside logo, Intel. Leap ahead., Intel. Leap ahead. logo, Intel NetBurst, Intel SpeedStep, Intel XScale, Itanium, Pentium, Pentium Inside, VTune, Xeon, and Xeon Inside are trademarks or registered trademarks of Intel Corporation or its subsidiaries in the United States and other countries.

Intel® Active Management Technology requires the platform to have an Intel® AMT-enabled chipset, network hardware and software, as well as connection with a power source and a corporate network connection. With regard to notebooks, Intel AMT may not be available or certain capabilities may be limited over a host OS-based VPN or when connecting wirelessly, on battery power, sleeping, hibernating or powered off. For more information, see <http://www.intel.com/technology/iamt>.

64-bit computing on Intel architecture requires a computer system with a processor, chipset, BIOS, operating system, device drivers and applications enabled for Intel® 64 architecture. Performance will vary depending on your hardware and software configurations. Consult with your system vendor for more information.

No computer system can provide absolute security under all conditions. Intel® Trusted Execution Technology is a security technology under development by Intel and requires for operation a computer system with Intel® Virtualization Technology, an Intel Trusted Execution Technology-enabled processor, chipset, BIOS, Authenticated Code Modules, and an Intel or other compatible measured virtual machine monitor. In addition, Intel Trusted Execution Technology requires the system to contain a TPMv1.2 as defined by the Trusted Computing Group and specific software for some uses. See <http://www.intel.com/technology/security/> for more information.

Hyper-Threading Technology (HT Technology) requires a computer system with an Intel® Pentium® 4 Processor supporting HT Technology and an HT Technology-enabled chipset, BIOS, and operating system. Performance will vary depending on the specific hardware and software you use. See www.intel.com/products/ht/hyperthreading_more.htm for more information including details on which processors support HT Technology.

Intel® Virtualization Technology requires a computer system with an enabled Intel® processor, BIOS, virtual machine monitor (VMM) and, for some uses, certain platform software enabled for it. Functionality, performance or other benefits will vary depending on hardware and software configurations and may require a BIOS update. Software applications may not be compatible with all operating systems. Please check with your application vendor.

* Other names and brands may be claimed as the property of others.

Other vendors are listed by Intel as a convenience to Intel's general customer base, but Intel does not make any representations or warranties whatsoever regarding quality, reliability, functionality, or compatibility of these devices. This list and/or these devices may be subject to change without notice.

Copyright © 2013, Intel Corporation. All rights reserved.

Topics

Why DPDK – PMD vs Linux interrupt driver, memory config, user space.

Licensing

Memory IA – NUMA, huge pages, TLBs on IA

Memory DPDK – mem pools, buffers, allocation etc.

Caching handling, DDIO

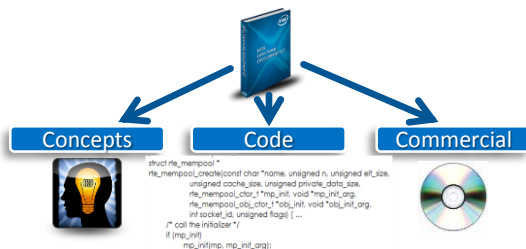
Intel® Data Plane Development Kit (Intel® DPDK)

• Big Idea

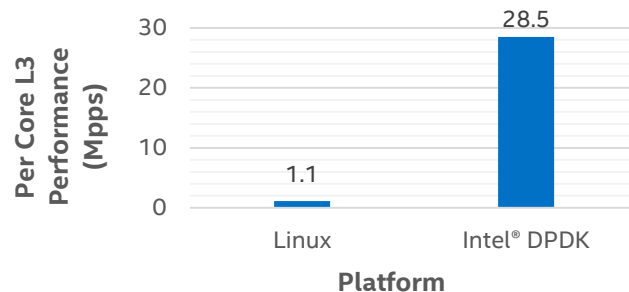
Software solution for accelerating Packet Processing workloads on IA.

- Delivers 25X performance jump over Linux
- Comprehensive Virtualization support
- Free, Open Source, BSD License
- Enjoys vibrant community support

• Deployment Models



• Performance



• Commercial Support

WIND RIVER

WIND

Intel

calsoft labs
An ALTEON Group Company

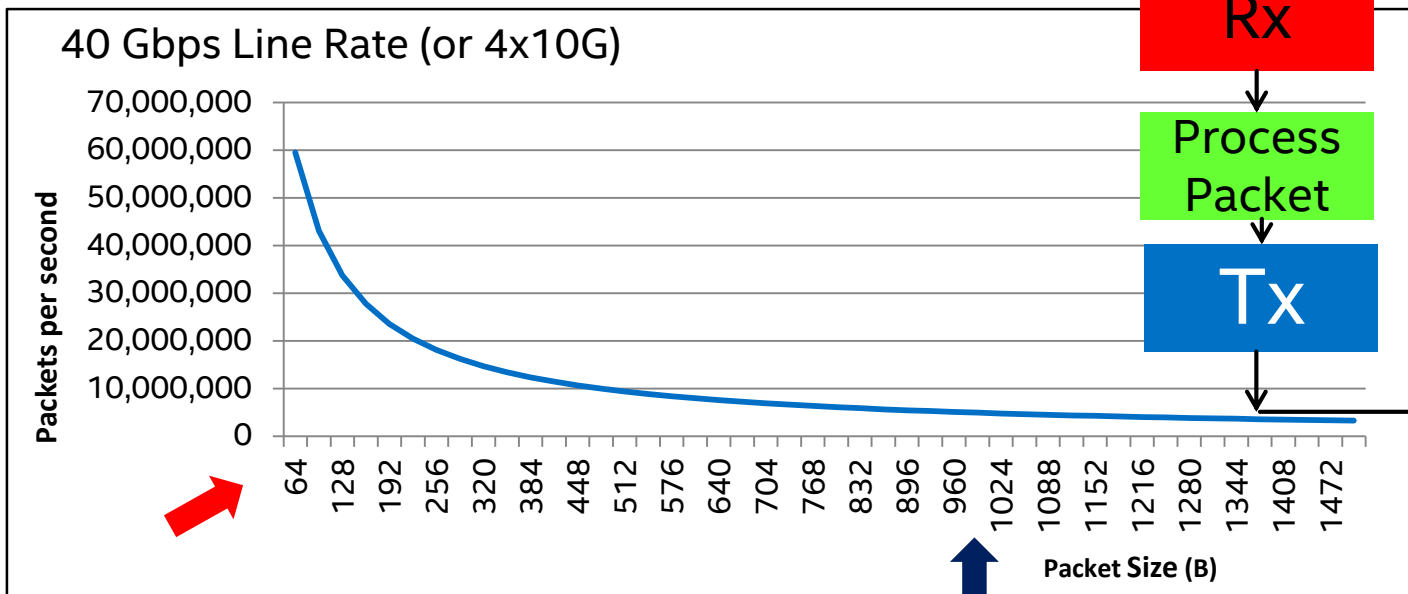
Disclaimer: Software and workloads used in performance tests may have been optimized for performance only on Intel microprocessors. Performance tests, such as SYSmark and MobileMark, are measured using specific computer systems, components, software, operations and functions. Any change to any of those factors may cause the results to vary. You should consult other information and performance tests to assist you in fully evaluating your contemplated purchases, including the performance of that product when combined with other products.

TRANSFORMING NETWORKING & STORAGE



What Problems Does DPDK Address?

What Problem Does DPDK address ?



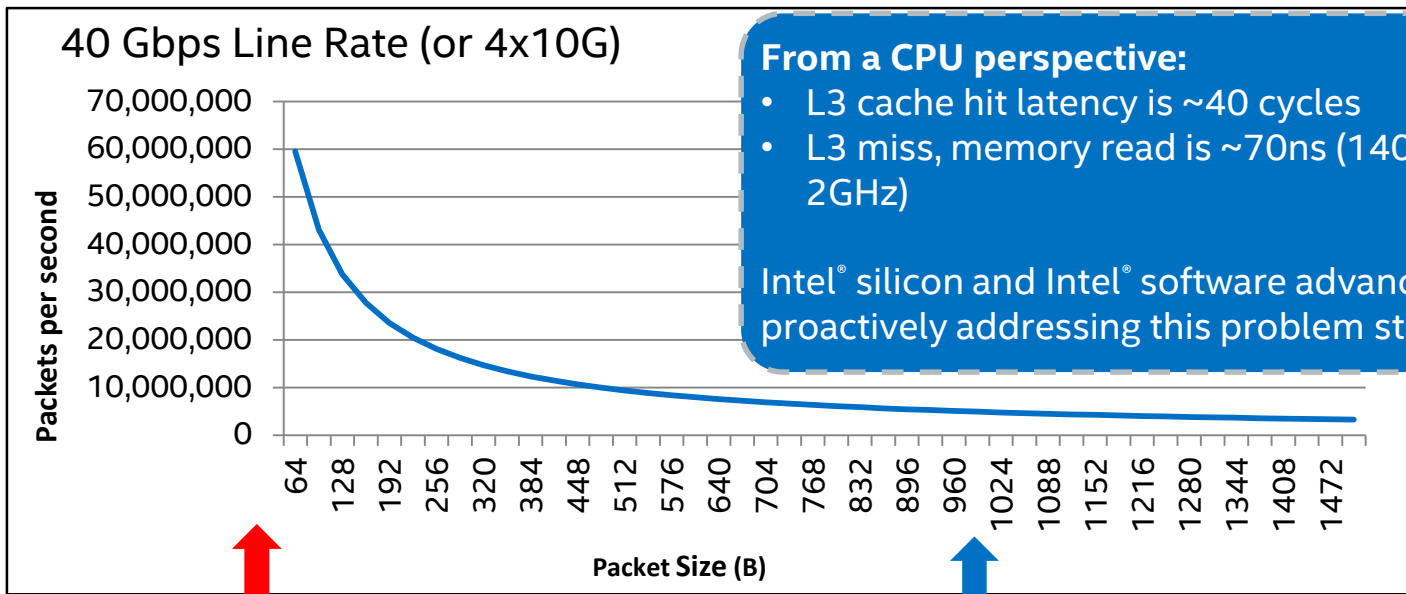
Network Infrastructure Packet Sizes

Packet Size	64 bytes
40G Packets/second	59.5 Million each way
Packet arrival rate	16.8 ns
2 GHz Clock cycles	33 cycles

Typical Server Packet Sizes

Packet Size	1024 bytes
40G Packets/second	4.8 Million each way
Packet arrival rate	208.8 ns
2 GHz Clock cycles	417 cycles

The Problem Intel® DPDK Addresses



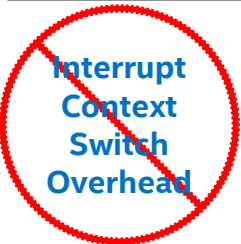


Network Infrastructure Packet Sizes

Packet Size	64 bytes
40G Packets/second	59.5 Million each way
Packet arrival rate	16.8 ns
2 GHz Clock cycles	33 cycles

Typical Server Packet Sizes

Packet Size	1024 bytes
40G Packets/second	4.8 Million each way
Packet arrival rate	208.8 ns
2 GHz Clock cycles	417 cycles

Benefits – Eliminating / Hiding Overheads

Eliminating	How?
 Interrupt Context Switch Overhead	Polling
 Kernel User Overhead	User Mode Driver
 Core To Thread Scheduling Overhead	Pthread Affinity

Licensing

- DPDK is BSD licensed:
- <http://opensource.org/licenses/BSD-3-Clause>
- User is free to modify, copy and re-use code
- No need to provide source code in derived software (unlike GPL license)

DPDK Packet Processing Concepts

- DPDK is designed for high-speed packet processing on IA. This is achieved by optimizing the software libraries to IA with some of the following concepts
 - Huge Pages Cache alignment Pthreads with Affinity
 - Prefetching New Instructions NUMA
 - Intel® DDIO Memory Interleave Memory Channel
- Intel® Data Direct I/O Technology (Intel® DDIO)
 - Enabled by default in all Intel® Xeon® processor E5-based platforms
 - Enables PCIe adapters to route I/O traffic directly to L3 cache, reducing unnecessary trips to system memory, providing more than double the throughput of previous-generation servers, while further reducing power consumption and I/O latency.
- Pthreads
 - On startup of the DPDK specifies the cores to be used via the Pthread call with affinity to tie an application to a core. Reducing the kernel's ability of moving the application to another local or remote core affecting performance.
 - The user may still use Pthreads or Fork calls after the DPDK has started to allow threads to float or multiple thread to be tied to a single core.

DPDK Packet Processing Concepts

- NUMA
 - DPDK utilizes NUMA memory for allocation of resources to improve performance for processing and PCIe I/O local to a processor.
 - With out the NUMA set in a dual socket system memory is interleaved between the two sockets.
- Huge Pages
 - DPDK utilizes 2M and 1G hugepages to reduce the case of TLB misses which can significantly affect a cores overall performance.
- Cache Alignment
 - Better performance by aligning structures on 64 Byte cache lines.
- Software Prefetching
 - Needs to be issued “appropriately” ahead of time to be effective. Too early could cause eviction before use
 - Allows cache to be populated before data is accessed
- Memory channel use
 - Memory pools add padding to objects to ensure even use of memory channels
 - Number of channels specified at application start up

Memory configuration

Intel Architecture

Memory – performance topics

- NUMA architecture
- Caching
- TLBs
- Huge pages
- Memory allocation

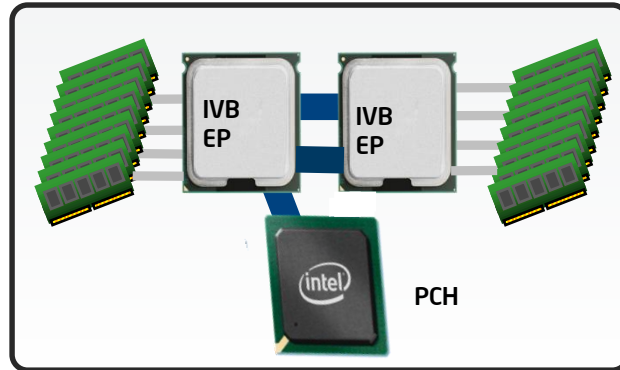
Intel® Core™ Microarchitecture Platform Architecture

Integrated Memory Controller

- 4 DDR3 channels per socket
- Massive memory **bandwidth**
- Memory Bandwidth scales with # of processors
- Very **low memory latency**

Intel® QuickPath Interconnect (Intel® QPI)

- New point-to-point interconnect
- Socket to socket connections



Significant performance leap for new platform

Non-Uniform Memory Access (NUMA)

FSB architecture (legacy)

- All memory in one location

Starting with Intel® Core™ microarchitecture (Nehalem)

- Memory located in multiple places

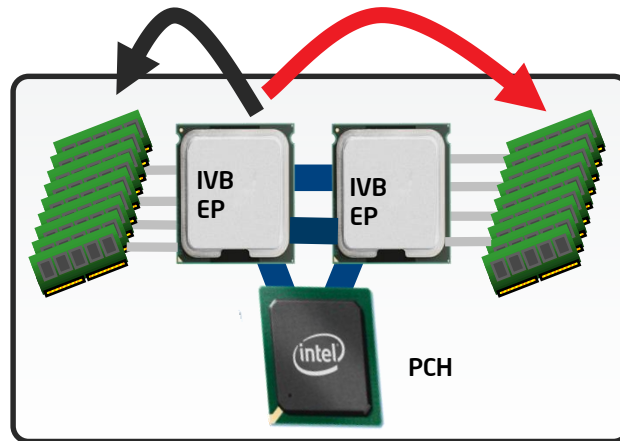
Latency to memory dependent on location

Local memory

- Highest BW
- Lowest latency

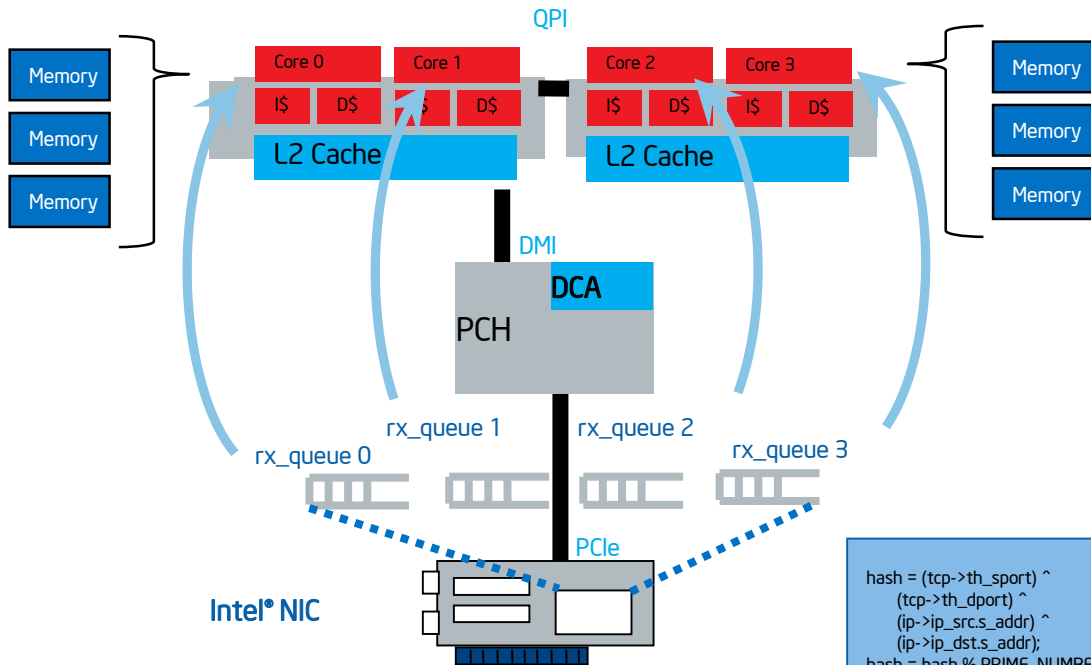
Remote Memory

- Higher latency



Ensure software is NUMA-optimized for best performance

NUMA Considerations for Data Structure Allocation



```
hash = (tcp->th_sport) ^
      (tcp->th_dport) ^
      (ip->ip_src.s_addr) ^
      (ip->ip_dst.s_addr);
hash = hash % PRIME_NUMBER;
return lookup_table[hash];
```

PTU Metrics

- MEM_UNCORE_RETIRED.REMOTE_DRAM
- MEM_INSTRUCTIONS_RETIRED.LATENCY_ABOVE_THRESHOLD

Caching on IA

Caching on IA

- IA Processors have cache integrated on processor die.
 - Fast access SRAM
 - Code & data from system memory (DRAM) stored in fast access cache memory
- Without a cache – CPU runs out of instructions from system memory
 - CPU Core “stalls” – waiting for data
- Cache miss (data not in cache)
 - CPU needs to get data from system memory
 - Cache populated with required data
 - Not just the data required, but a block of info is copied
 - “Cache line” – 64 Bytes on IA (IVB, HSW etc.)
- Cache hit – data present in cache

Caching on IA – some terms

- **Cache Consistency**
 - Cache is a copy of a piece of memory
 - Needs to always reflect what is contained in system memory
- **Snoop**
 - Cache watches address lines for transaction
 - Cache sees if any transactions access memory contained within cache
 - Cache keeps consistent with caches of other CPU cores
- **Dirty data**
 - Data modified in cache but not in main memory
- **Stale data**
 - Data modified in main memory, but not in cache

Caching on IA

- 3 Levels of cache (SNB, IVB, HSW processors)
 - L1 cache – 32KB data and 32KB instruction caches
 - L2 cache – 256KB – unified (holds code & data)
 - L3 cache (LLC) – 25MB (IVB) , 30MB (HSW) common cache for all cores in CPU socket.
- L1 cache is smallest, and fastest.
 - CPU tries to access data – not in L1 cache?
 - Try L2 cache - not in L2 cache?
 - Try L3 cache – not in L3 cache?
 - Cache miss - need to access system memory (DRAM).
- L1 & L2 cache is per physical core (shared per logical core)
- L3 cache is shared (per CPU socket)

Caching on IA

- What can be cached?
 - Only DRAM can be cached
 - IO, MMIO never cached
- L1 cache is smallest, and fastest.
- L1 Code cache is read-only
- Address residing in L1/L2 must be present in L3 cache – “inclusive cache”

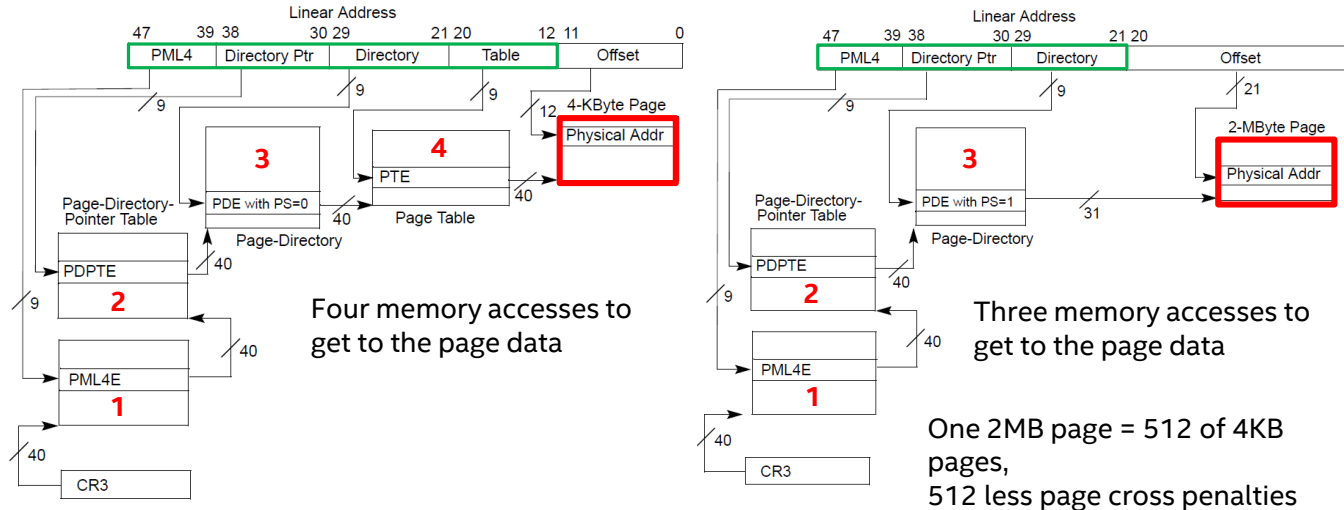
Huge Pages

Huge Pages

- All memory addresses virtual
 - Memory appears contiguous to applications, even if physically fragmented
- Map virtual address to physical address
 - Use page tables to translate virtual address to physical address
 - Default page size in Linux on IA is 4kB.
 - 4 layers of page tables

Why Hugepages?

TLB maps page numbers to page frames. Each TLB miss requires page walk.



DTLB:

- 4K pages 64 entries, maps 256 KB, so to access 16G of memory 32MB of PTE tables read by CPU
- 2M pages 32 entries, maps 64 MB, so to access 16G of memory 64Kb of PDE tables read by CPU, fits into CPU cache

Huge Pages

- Use Linux hugepage support through “hugetlbfs” filesystem
- Each page is 2MB in size equivalent to 512 4KB pages
- Each page requires only 1 DTLB entry
- Reduce DTLB misses, and therefore page walks
- Gives improved performance
- Need to enable & allocate huge pages with Linux boot command (in GRUB file)
 - Better to enable at boot time – prevents fragmentation in physical memory

Translation Lookaside Buffers

TLBs – virtual to physical memory address translation

*Intel® 64 and IA-32 Architectures Software Developer's Manual.
Volume 3. System Programming Guide. Chapter 4.10: Caching Translation Information
Intel® 64 and IA-32 Architectures Optimization Reference Manual.*

Translation Lookaside Buffers (TLBs)

- TLBs – Translation Lookaside Buffers – 2 types
 - Instruction TLB
 - Data TLB
- TLB is cache – maps virtual memory to physical memory
 - When memory requested by application, OS maps virtual address from process to physical address in memory
 - Mapping of virtual to physical memory – Page Table Entry (PTE)
 - TLB is a cache for the Page Table
 - If data is found in TLB during address lookup
 - TLB hit
 - Otherwise – TLB miss (page walk) - performance hit
 - Huge pages (Linux) – can alleviate

Translation Lookaside Buffers (TLBs)

- TLBs are a cache for page tables
- If memory address lookup is not in TLB -> TLB miss
 - We must then “walk the page tables”
 - This is slow, and costly
- We need to minimise TLB misses
- Solution is to use huge pages
 - Use 2M or 1G huge pages instead of default 4k pages

TLB Invalidation

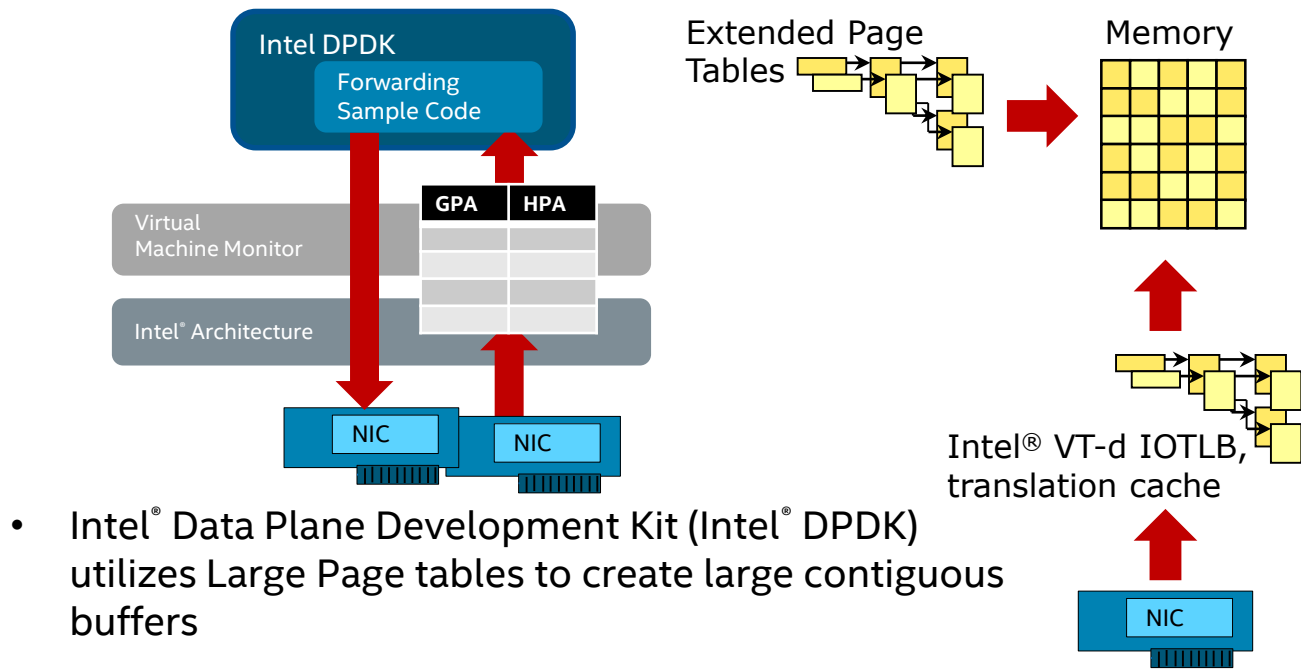
- On multi-core systems one core may change the page table which is used by other cores
- Page table change needs to be propagated to other cores TLBs
- This process is known as “TLB shutdown”
 - Need to invalidate the TLBs to avoid using “stale” data
- Need to be aware of other CPU cores invalidating TLBs
 - Costly for data plane applications.
 - Examples – page faults, VM transitions (VM exit & entry)
- More info in section 4.10.4 of *Volume 3A of Intel® 64 and IA-32 Architectures Software Developer’s Manual*
 - <https://www-ssl.intel.com/content/dam/www/public/us/en/documents/manuals/64-ia-32-architectures-software-developer-vol-3a-part-1-manual.pdf>

IOTLBs

- As well as TLBs for memory, there are TLBs for DMA – IOTLBs
 - Page table structure for DMA address translation
- Sandy Bridge – no huge page support in IOTLBs – page table fragmentation
 - 2M and 1G huge pages fragmented to 4k page size
 - Causes more IOTLB misses
 - SNB could not achieve near 64 byte line rates for 10G NIC
- Huge page support added in IVB
- SR-IOV performance in IVB greatly enhanced

Large Page Table Support

Reducing TLB and IOTLB misses with Large Page Table support



Memory Virtualization Challenges

Address Translation

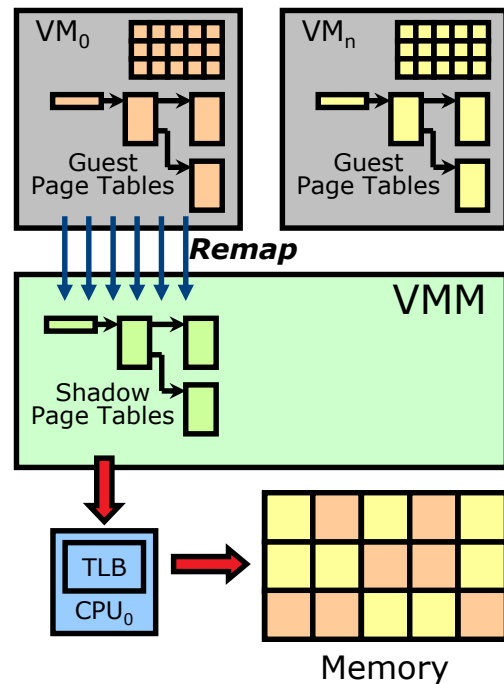
Guest OS expects contiguous,
zero-based physical memory
VMM must preserve this illusion

Page-table Shadowing

VMM intercepts paging operations
Constructs copy of page tables

Overheads

VM exits add to execution time
Shadow page tables consume
significant host memory



Memory Virtualization with EPT

Extended Page Tables (EPT)

Map guest physical to host address

New hardware page-table walker

Performance Benefit

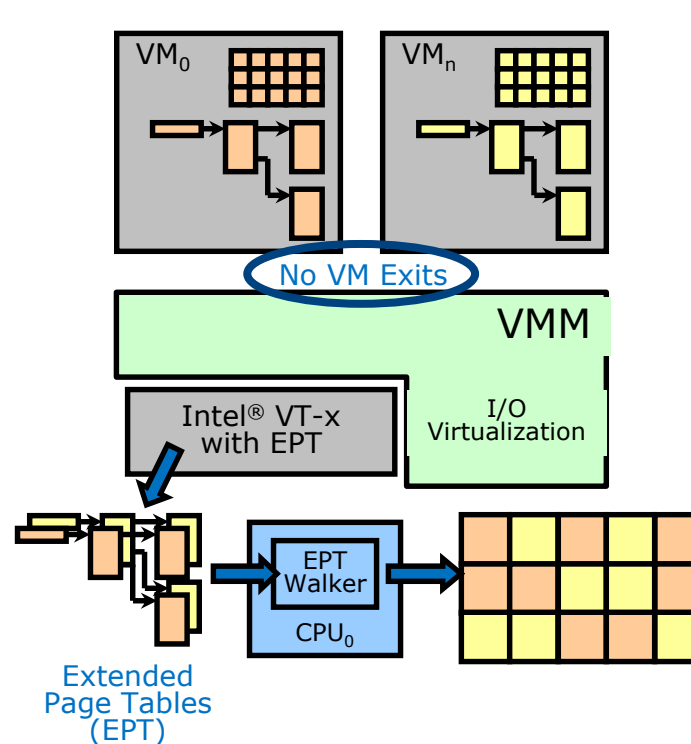
Guest OS can modify its own page tables freely

Eliminates VM exits

Memory Savings

Shadow page tables required for each guest user process (w/o EPT)

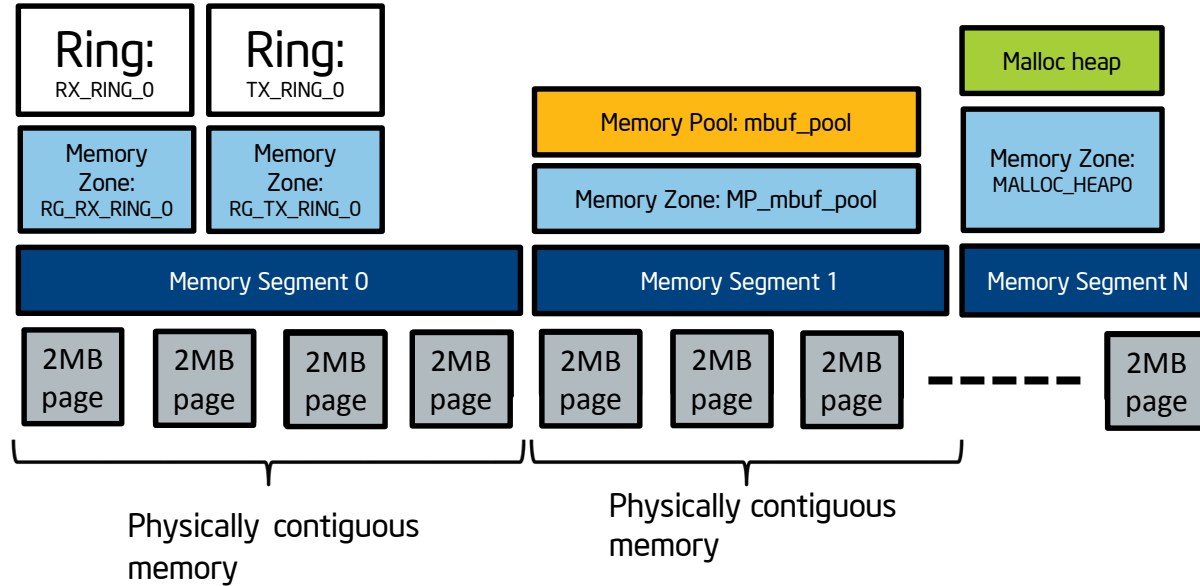
A single EPT supports entire VM



Memory Configuration

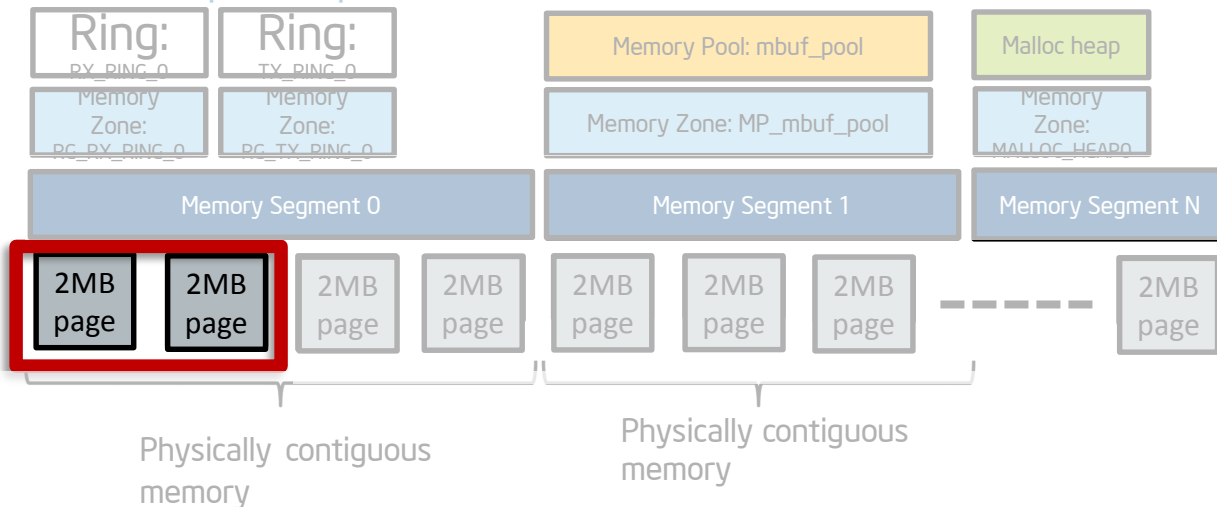
DPDK

Memory Object Hierarchy



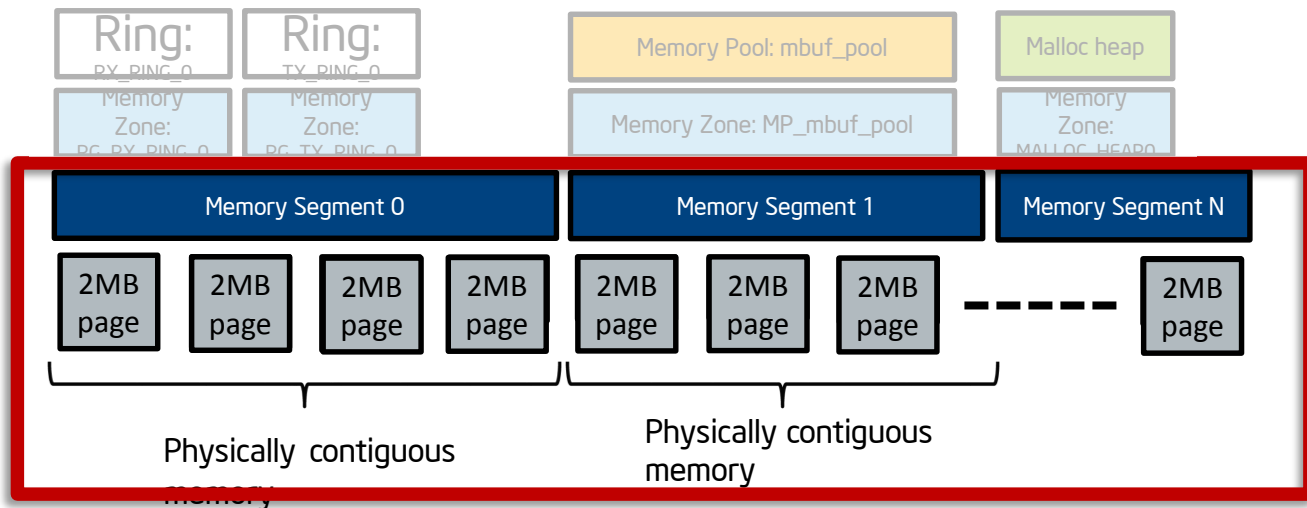
Hugepages

- Use Linux hugepage support through “hugetlbfs” filesystem
- Each page is 2MB in size equivalent to 512 4KB pages
- Each page requires only 1 DTLB entry
- Reduce DTLB misses, and therefore page walks
- Gives improved performance



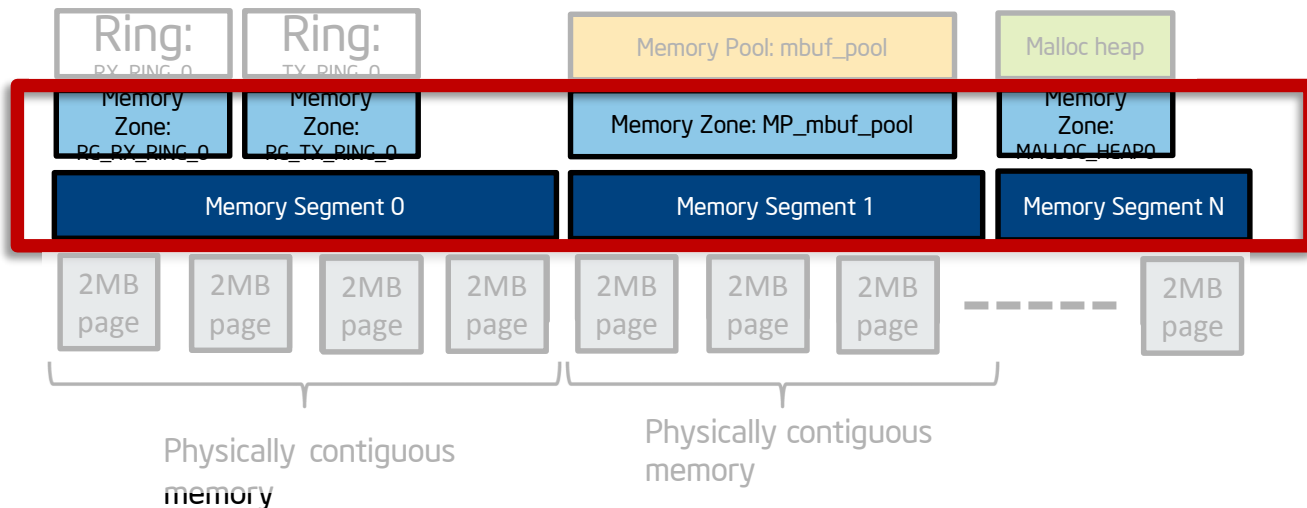
Memory Segments

- Internal unit for memory management is the memory segment
- Always backed by Huge Page (2 MB/1 GB page) memory
- Each segment is contiguous in physical and virtual memory
- Broken out into smaller memory zones for individual objects



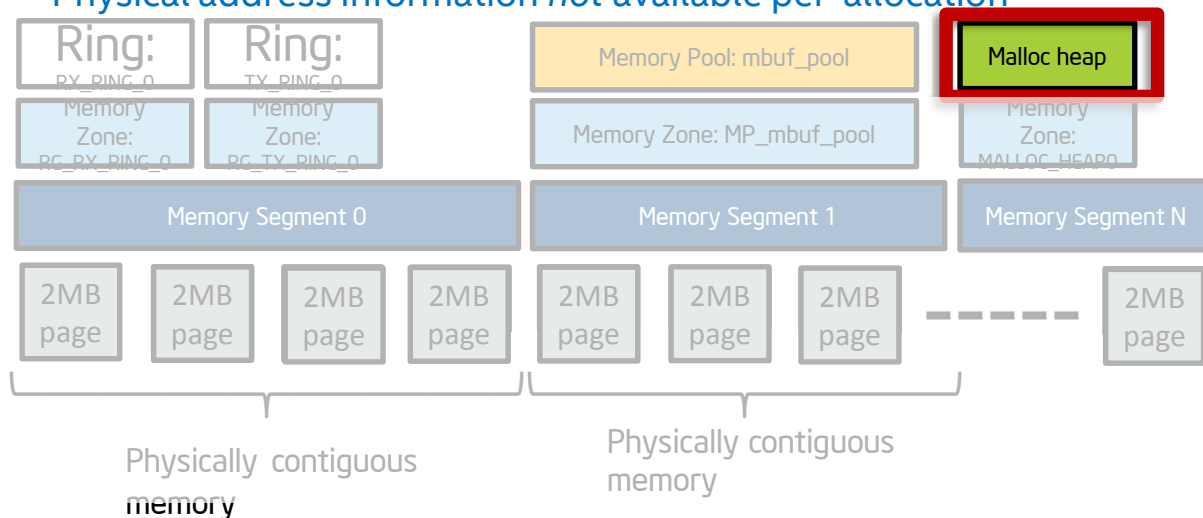
Memory Zones

- Most basic unit of memory allocation – named block of memory
- Allocate-only, cannot free
- Cannot span a segment boundary – contiguous memory
- Physical address of allocated block available to caller



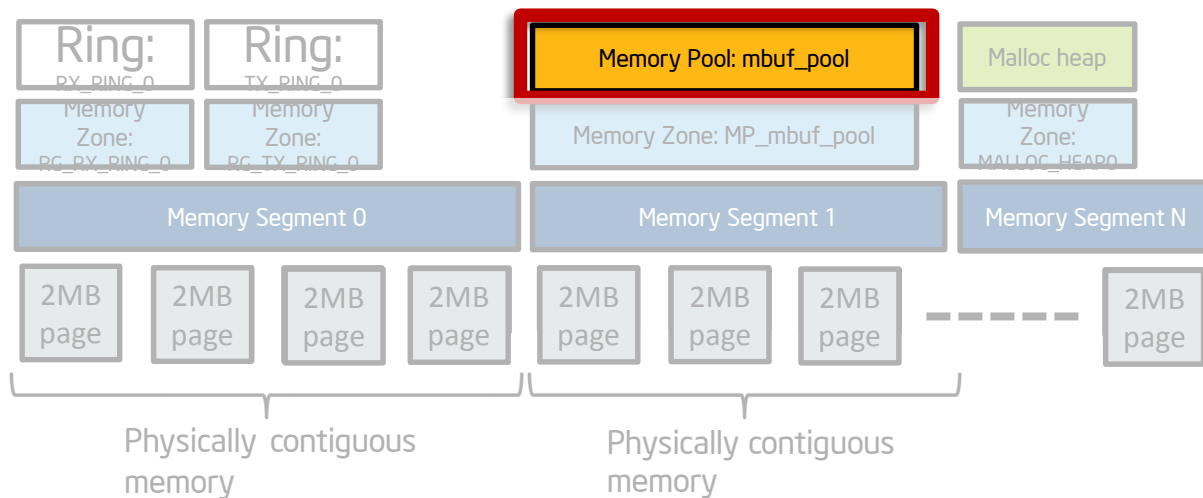
Malloc support – rte_malloc/rte_free

- Malloc library provided to allow easier application porting
- Backed by one or more memzones
- Uses hugepage memory, but supports memory freeing
- Not lock-free – avoid in data path
- Physical address information *not* available per-allocation



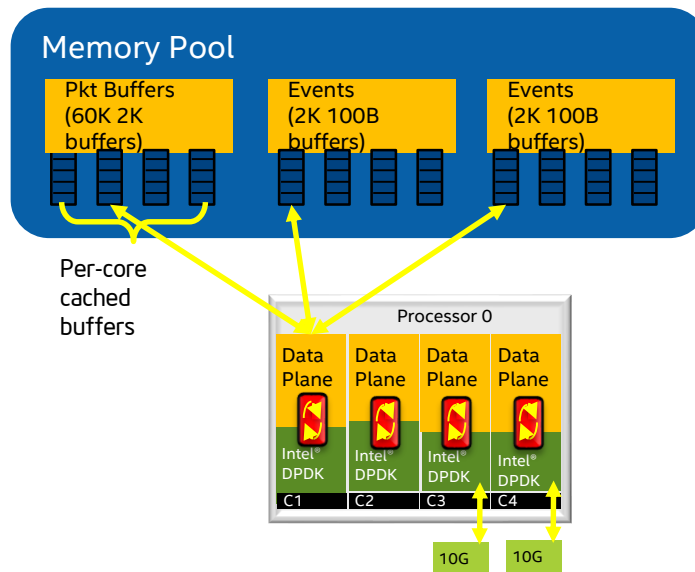
Memory Pools

- Pool of fixed-size buffers
- One pool can be safely shared among many threads
- Lock-free allocation and freeing of buffers to/from pool
- Designed for fast-path use



Memory Pools (continued)

- **Size fixed at creation time:**
 - Fixed size elements
 - Fixed number of elements
- **Multi-producer / multi-consumer safe**
- **Safe for fast-path use**
- **Typical usage is packet buffers**
- **Optimized for performance:**
 - No locking, use CAS instructions
 - All objects cache aligned
 - Per core caches to minimise contention / use of CAS instructions
 - Support for bulk allocation / freeing of buffers



Memory allocation - summary

- For DPDK application – allocated all memory from huge pages
- Allocate all memory at initialisation time (not during run time).
- Pools of buffers created.
 - Buffers taken from pools as needed for packet processing
 - Returned to pool after use
 - Never need to use “malloc” at runtime.
 - DPDK takes care of aligning memory to cache lines

Memory allocation

- `rte_eal_init()`
 - Initialises Environment Abstraction Layer
 - Takes care of allocating memory from huge pages
- `rte_mempool_create()`
 - Create pool of message buffers (mbufs)
 - This pool is used to hold packet data
 - mbufs taken from and returned to this pool

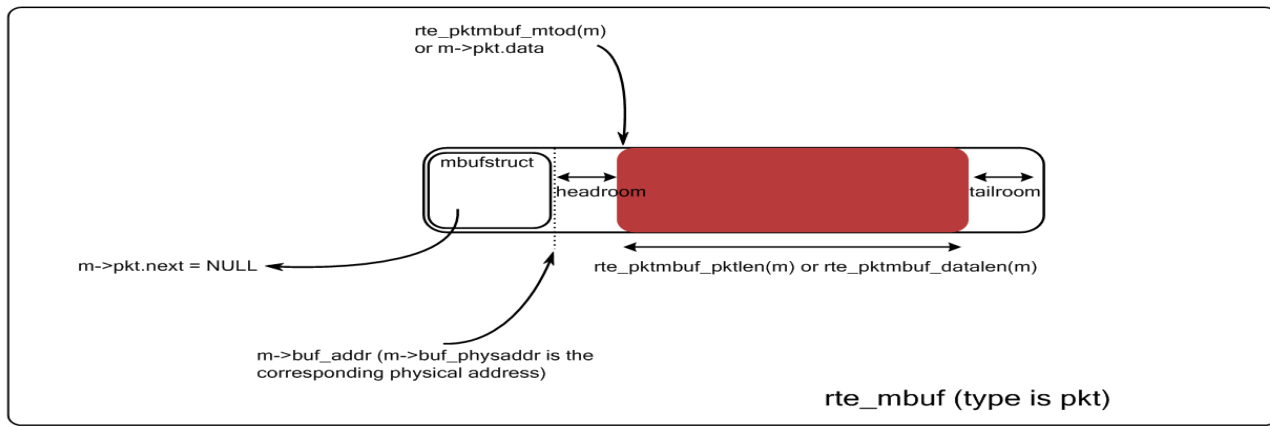
Memory Buffer - mbuf

Memory buffer structure used throughout the Intel® DPDK

Header holds meta-data about packet and buffer

- Buffer & packet length
- Buffer physical address
- RSS hash or flow director filter information
- Offload flags

Body holds packet data plus room for additional headers and footers.

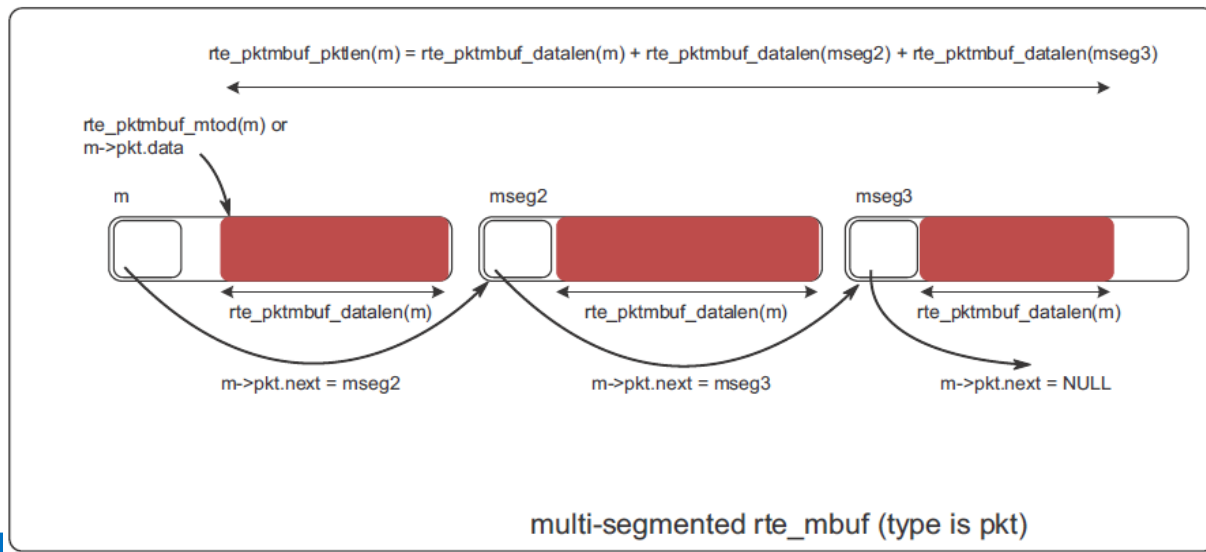


Memory Buffer – chained mbuf

Mbufs generally used with memory pools

Size of mbuf fixed when the mempool is created

For packets too big for a single mbuf, the mbufs can be linked together in an “mbuf chain”



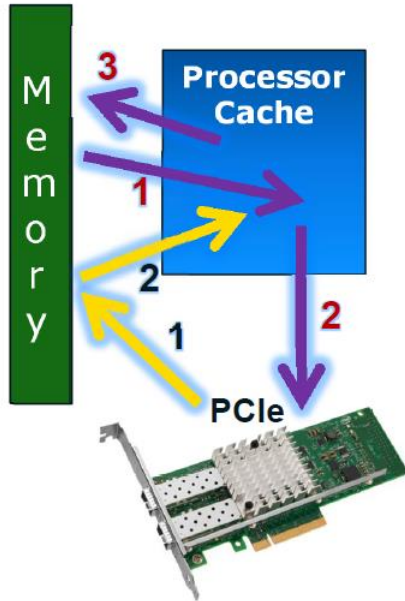
DDIO

Data Direct I/O (DDIO)

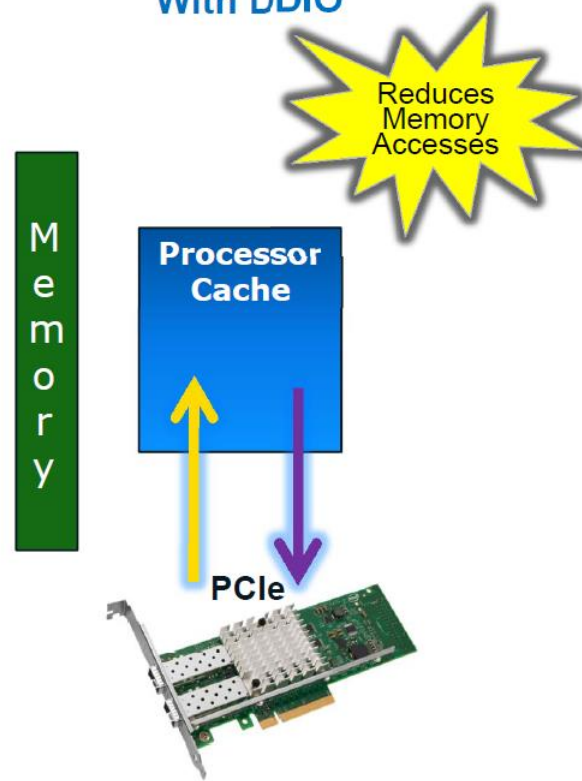
- Ethernet controllers & NICs talk directly with CPU cache
- DDIO makes processor cache the primary source and destination of I/O data, rather than main memory
- DDIO reduces latency, power consumption, and memory bandwidth
 - Lower latency – I/O data does not need to go via main memory
 - Lower power consumption – reduced memory access
 - More scalable I/O bandwidth – reduced memory bottlenecks

How Does DDIO Work?

“Classical” Approach
Before DDIO



Romley Approach
With DDIO



DDIO requires no complex setup

- DDIO is enabled by default on all Romley platforms, including pre-released platforms for OEMs, IHVs, and ISVs
 - DDIO has been active on all Intel and industry Romley development and validation
- DDIO has no hardware dependencies
- DDIO is invisible to software
 - No driver changes are required
 - No OS or VMM changes are required
 - No application changes are required

