rte_mempool_handler_register(&ops)

[New] Mempool Handler Registration for all mempool handlers

- Ops structure has the following members
  - .name[] : name of the mempool handler
  - .create : function callback on creation of the mempool
  - .get_bulk
  - .put_bulk
  - .get_count
  - .virt2phy

Current mempool handlers (default, XEN) will be converted to use this mechanism

Mempool ops.create

Called at memory pool initialization time only

Function is a callback to allocate mempool memory

- Current DPDK memory pool allocation code will be called for default handler (default, or XEN DOM0)
- Newer/different memory allocators can integrate in a fashion similar to XEN_DOM0

Function is expected to setup each rte_mbuf structure and call put_bulk() API to create mbuf pool

- External memory handlers could allocate a superstructure/wrapper around rte_mbuf to support their custom buffer structure

rte_mempool_def_handler()

Used to set the default memory pool handler

- Current alloc/get/put code is setup as default
- #ifdef XEN_DOM0 will install a different handler (with different create code)

This is usually what will be called back by the upper level helper functions (e.g. rte_pktmbuf_pool_create) that create mempools

- rte_pktmbuf_pool_create() currently has no way of selecting which specific mempool needs to be used
- [OPTION/TBD]: New API rte_pktmbuf_pool_create_ex() that takes a specific mempool handler name to associate with

How do I install my own memory handler?

INIT

1. Register the handler using rte_mempool_handler_register()
2. On the create callback, allocate your memory, initialize the mbufs and call the put function.

RUNTIME

3. You will get called back on every get/put call from the application/PMD
   - These functions are directly in the fast-path and any unoptimized handlers will become the performance limiter
   - Could also use the current rte_ring() mechanism (and the functions)
4. If indirect mbufs are used, there will be virt2phy() calls
5. Debugging/dump functions use the get_count() call to return number of elements allocated in the mempool

Current opens

Performance

- Likely negative impact (upto ~15%?) on mempools without cache
- Likely no negative impact on mempools with cache