Mục lục

1 Tổng quan DPDK

DPDK là một framework đơn giản, toàn diện cho việc xử lý gói tin nhanh trong các ứng dụng data plane. DPDK chạy trên hệ điều hành Linux, có thể thay thế hoàn toàn network stack.

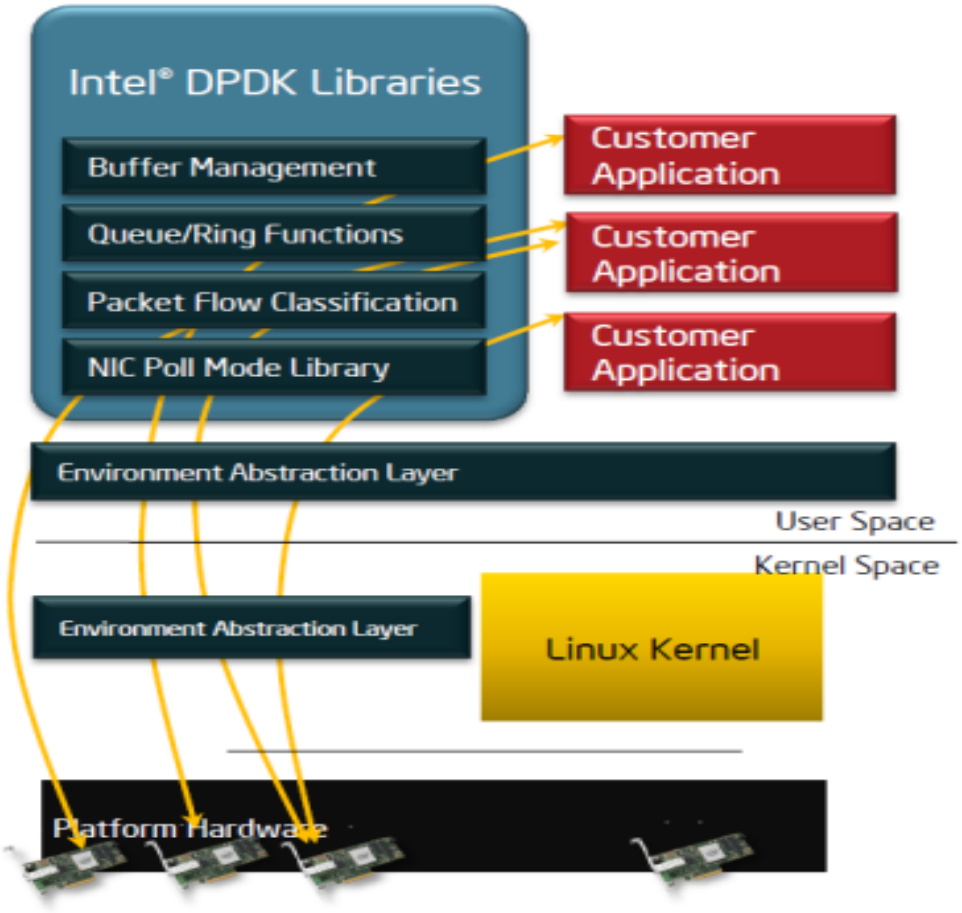
Để đạt được tốc độ đường truyền, 1,48 triệu của gói tin 64 byte/giây thì phải được gửi hoặc nhận trên giao diện 1G và tương ứng 14,8 triệu trên giao diện 10G.

Để tăng hiệu suất xử lý gói tin của máy chủ thì bypass kernel được thực hiện. Tất cả các hoạt động xử lý gói tin đều xảy ra trong user space.

DPDK cung cấp các thư viện được thực thi hoàn toàn trong user space bằng cách tạo Environment Abstraction Layer (EAL). EAL có thể hỗ trợ 32-bit hoặc 64-bit.

DPDK triển khai mô hình completion cho việc xử lý gói tin, trong đó các tài nguyên phải được phân bổ trước khi gọi các ứng dụng data plane, chạy như các unit trong core xử lý logic. Mô hình này không hỗ trợ lập lịch và tất cả thiết bị truy cập bằng cách bỏ phiếu. Ngoài ra,DPDK có thể triển khai theo mô hình pipeline, được dùng để trao đổi tin nhắn giữa các core khác nhau để thực hiện công viêc theo từng giai đoạn.

2. Kiến trúc DPDK



2.1 Thư viện DPDK

DPDK cung cấp một vài thư viện được tối ưu cho cho hiệu năng cao. Nó làm được những nhiệm vụ cơ bản giống như những gì network stack trong linux có thể làm: phân bổ bộ nhớ cho các gói tin, đệm các mô tả gói tin và chuyển các gói tin từ NIC đến ứng dụng (và ngược lại).

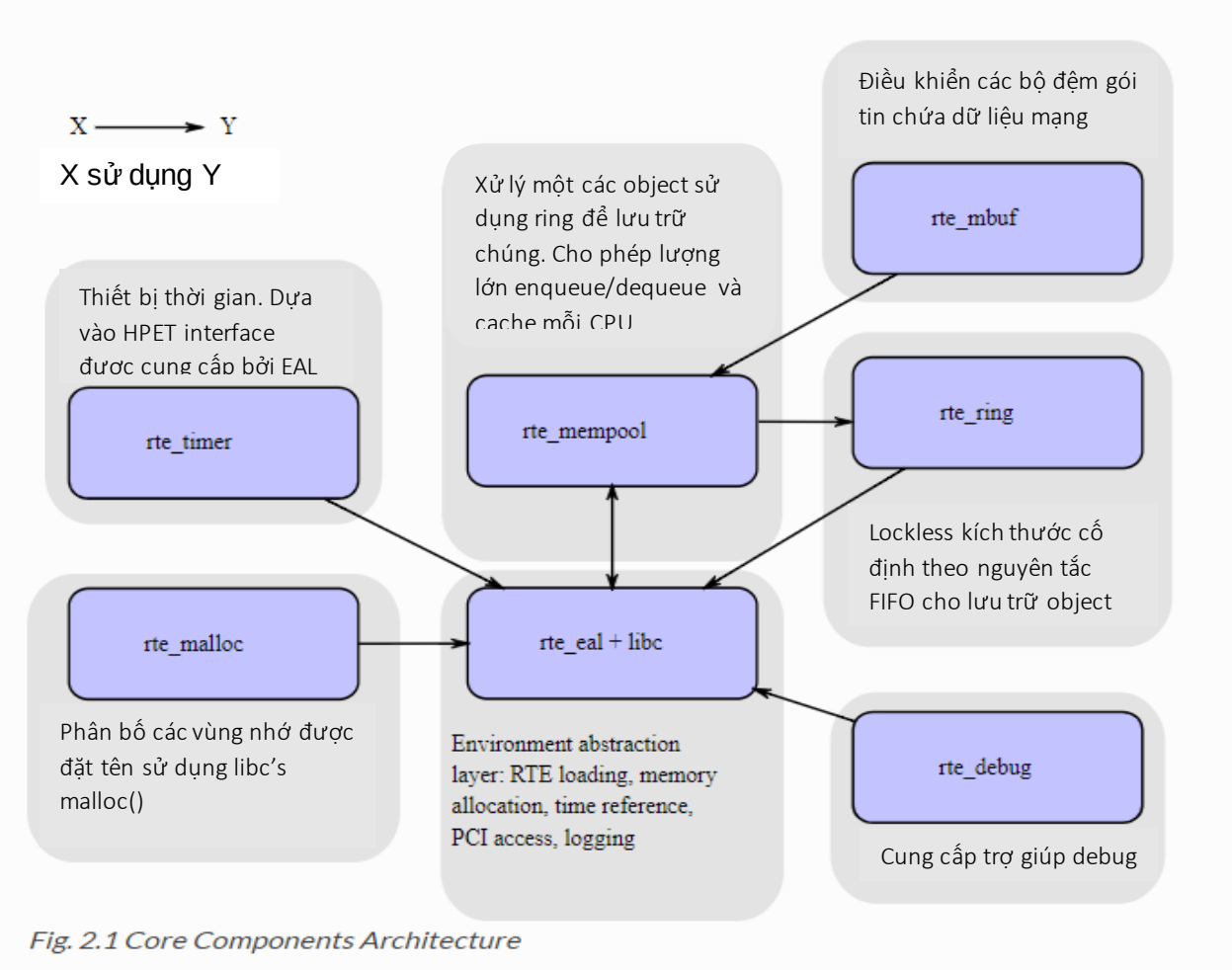


Fig. 2.1 Core Components Architecture

2.1.1 Quản lý hàng đợi

Gói tin được nhận trong DPDK được gửi đến hàng đợi đã được khởi tạo trong thư viện rte_ring.

Để quản lý bất kì hàng đợi nào thì thư viện librte_ring cung cấp một cấu trúc vòng được gọi là rte_ring.

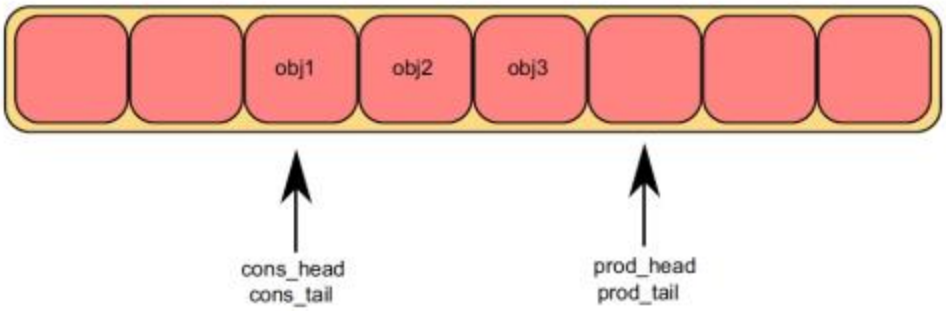


Figure 4: The *rte_ring* structure (image from [8])

Hàng đợi là một bộ đệm vòng không khóa được xây dựng theo cơ chế first in, first out. Bộ đệm vòng là một bảng chứa các con trỏ của các đối tượng có thể được lưu trong bộ nhớ để điều khiển truy nhập. Các con trỏ được chia thành 4 loại : prod_tail, prod_head, cons_tail, cons_head (prod: producer, cons: consumer)

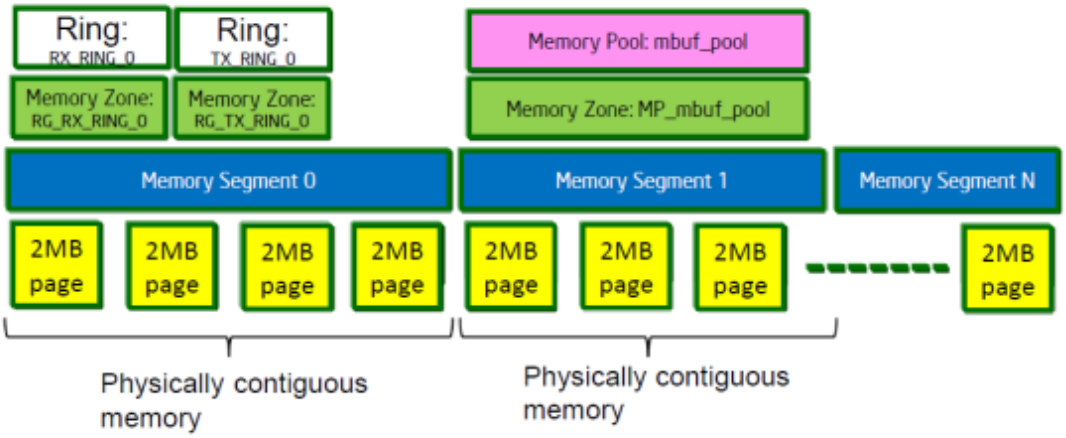
Producer là một tiến trình ghi dữ liệu vào bộ đệm tại một thời gian nhất đinh, và consumer là tiến trình đọc dữ liệu từ bộ đệm.

Tail là vị trí mà diễn ra tiến trình ghi trong bộ đệm vòng. Head là vị trí diễn ra tiến trình đọc tại một thời điểm.

Cách ring buffer hoạt động như sau : khi một đối tượng mới được thêm vào hàng đợi, con trỏ prod_tail-ring sẽ kết thúc bằng cách trỏ đến địa chỉ của ring-prod_head đã được chỉ ra trước đó.

Ưu Điểm	Nhược Điểm
- Dữ liệu được ghi vào bộ đệm nhanh - Khi thêm hoặc xóa một lượng lớn các đối tượng từ hàng đợi, lỗi cache xảy ra ít hơn vì các con trỏ được lưu trong một bảng	- Ring buffer có kích thước cố định nên không thể tăng kích thước trong lúc chạy - Tốn nhiều bộ nhớ là danh sách liên kết vì ring buffer chứa lượng tối đa con trỏ.

2.1.2 Quản lý bộ nhớ



Phân bố bộ nhớ vật lý thành các phân đoạn và vùng sử dụng mục đích khác

DPDK yêu cầu các hugepages.Theo các hướng dẫn cài đặt, nên tạo hugepages có kích thước 2MB. Những pages này sẽ kết hợp thành từng segment, sau đó sẽ được chia ra thành các vùng. Các đối tượng được tạo ra bởi ứng dụng hoặc các thư viện khác như là hàng đợi và bộ đệm gói tin sẽ được đặt trong những vùng này.

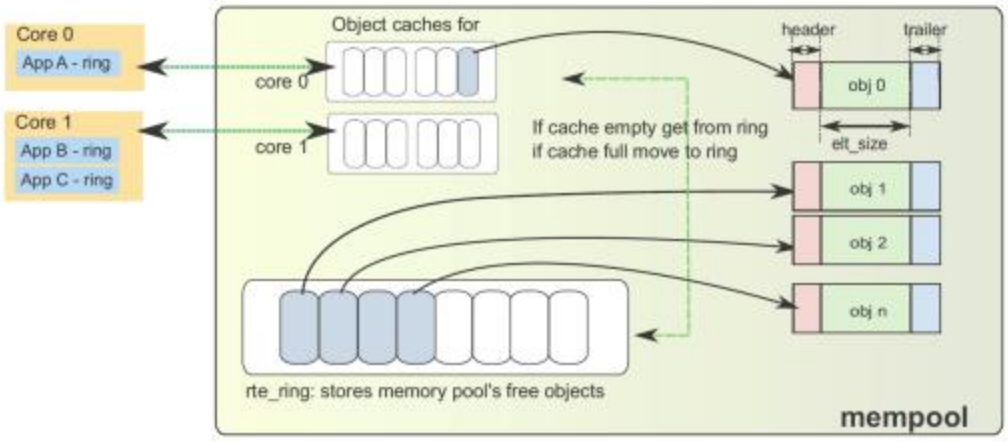


Figure 6: A *rte_mempool* with its associated *rte_ring* (image from [8])

Các đối tượng bao gồm các vùng bộ nhớ được tạo bởi thư viện rte_mempool. Đây là những nhóm đối tượng có kích thước cố định, sử dụng rte_ring để lưu trữ cấc đối tượng tự do và có thể xác định bằng tên riêng.

Kỹ thuật liên kết bộ nhớ có thể được thực hiện để cải thiện hiệu suất.

Mặc dù việc truy cập vào các đối tượng tự do được thiết kế trên một bộ đệm vòng không khóa, thì việc tiêu thụ tài nguyên hệ thống vẫn có thể rất cao. Khi nhiều lõi truy cập vào một vòng, một hoạt động so sánh và thiết lập (CAS) thường phải được thực hiện mỗi khi nó được truy cập.

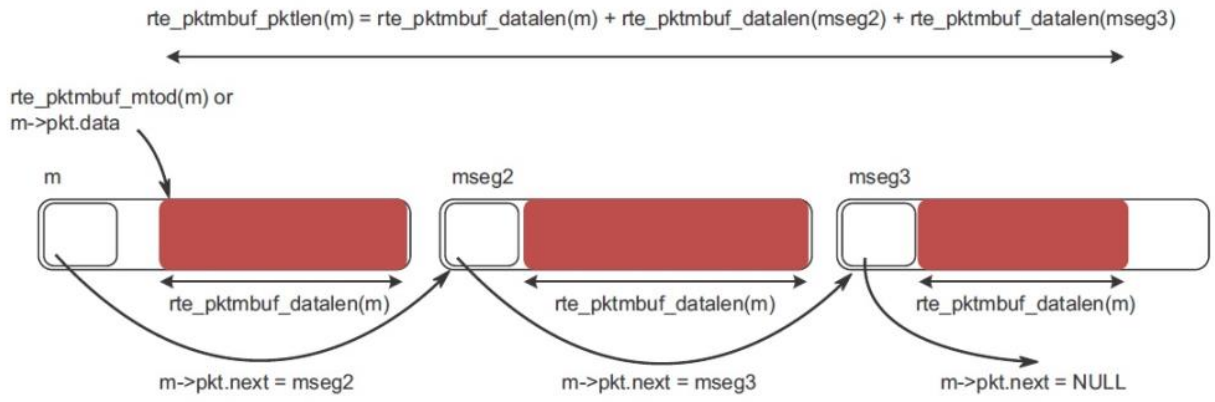
Để ngăn chặn sự "bottleneck", mỗi lõi sẽ được thêm bộ nhớ cache cục bộ bổ sung trong bộ nhớ. Sử dụng cơ chế khóa, lõi có thể truy cập hoàn toàn bộ nhớ cache đối tượng tự do. Khi bộ nhớ cache đầy hoặc trống rỗng, bộ nhớ sẽ trao đổi dữ liệu với bộ đệm vòng. Điều này cho phép core truy cập nhanh đến các đối tượng thường xuyên được sử dụng.

2.1.3 Quản lý bộ đệm

Đối với bất kỳ ứng dụng nào có thể truyền tải các gói tin mạng, thư viện librte mbuf cung cấp các bộ đệm được gọi là rte mbuf.

Các bộ đệm này được tạo ra trước thời gian chạy thực tế của ứng dụng và được lưu trữ trong một mempool. Trong thời gian chạy, người dùng có thể sử dụng một rte mbuf bằng cách chỉ định một mempool từ cái nó muốn lấy một bộ nhớ sẵn có. Giải phóng một rte mbuf bằng cách trả nó trở lại mempool mà nó bắt đầu.

Kích thước của rte_mbuf được giữ nhỏ nhất có thể để khớp với một dòng cache. Để lưu trữ thông tin của môt gói tin lớn, nhiều rte_mbuf có thể được xâu lại một chuỗi.



Mỗi rte_mbuf luôn luôn chứa metadata (message type, length, data segment starting address), được lấy từ network driver để có thể xử lý dễ dàng hơn. Rte_mbuf cũng chứa một con trỏ đến một rte_mbuf khác để cho phép bộ đệm xâu chuỗi lại khi cần xử lý gói tin với lượng lớn dữ liệu.

Cấu trúc này có thể dễ dàng mở rộng để chứa các dạng khác.

2.1.4 Thư viện rte_timer

Thư viện rte_timer cho phép thực hiện các chức năng đồng bộ. Bộ đếm thời gian có thể chạy một lần hoặc chạy theo định kỳ.

2.2. Environment Abstraction layer

EAL có trách nhiệm truy cập các tài nguyên cấp thấp như không gian bộ nhớ và phần cứng. Nó cung cấp một giao diện chung chung che giấu các đặc trưng môi trường từ các ứng dụng và

thư viện. Đó là trách nhiệm của quy trình khởi tạo để quyết định cách phân bổ các tài nguyên này (nghĩa là không gian bộ nhớ, thiết bị PCI, bộ định thời, bảng điều khiển, v.v.).Các dịch vụ được EAL cung cấp :

- Tải và khởi chạy DPDK
- Hỗ trợ đa tiến trình và đa luồng
- Các thủ tục quan hệ / chuyển nhương lõi
- Hoạt động khóa/atomic
- Time reference
- Truy cập PCI bus
- Chức năng tìm kiếm và debug
- Định danh các thuộc tính CPU
- Xử lý gián đoạn
- Chức năng alarm
- Quản lý bộ nhớ.

2.3. Poll mode driver

DPDK bao gồm 1 Gigabit, 10 Gigabit và 40 Gigabit và các trình điều khiển Chế độ Poll Mode ảo.

Trình điều khiển Chế độ Thăm dò (PMD) bao gồm các API, được cung cấp thông qua trình điều khiển BSD chạy trong user space, để định cấu hình thiết bị và hàng đợi tương ứng của chúng. Ngoài ra, một PMD truy cập các trình mô tả RX và TX trực tiếp mà không có bất kỳ ngắt đoạn nào (trừ trường hợp Ngắt kết nối Thay đổi Trạng thái) để nhanh chóng nhận, xử lý và phân phối các gói tin trong ứng dụng của người dùng.

3. Luồng gói tin trong DPDK

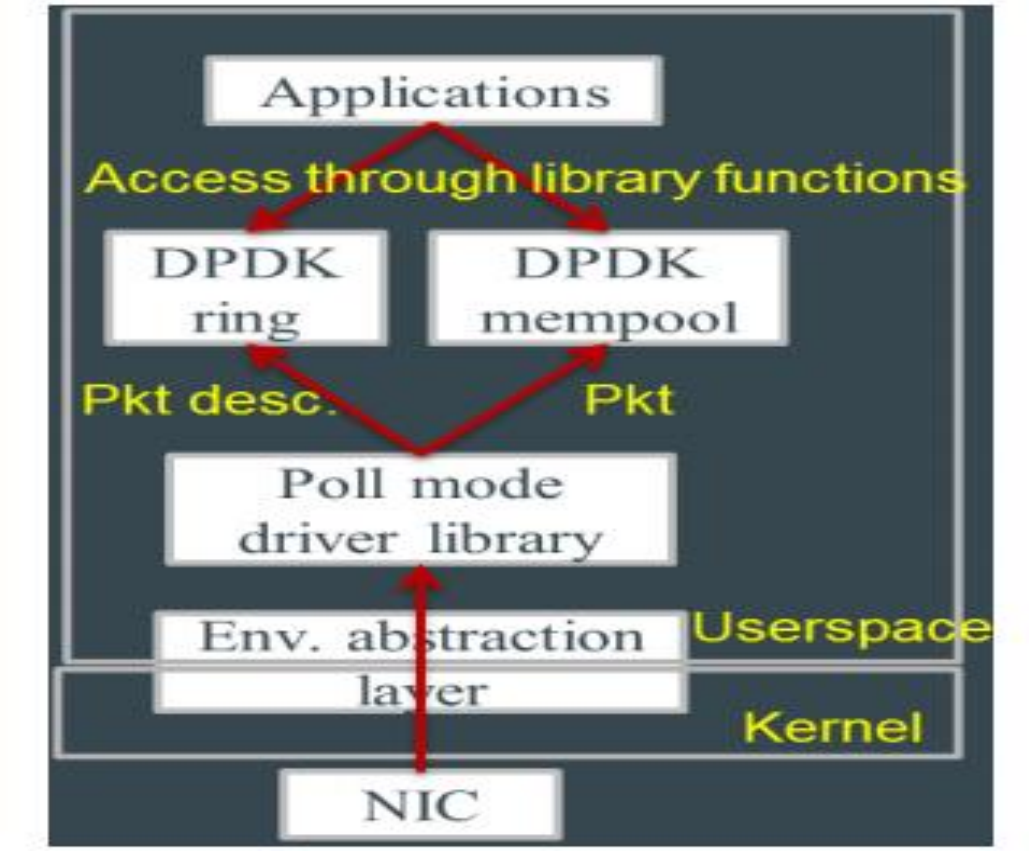


Figure 2.1: Packet I/O path in DPDK

- Ứng dụng kiểm tra cấu trúc vòng DPDK (nơi duy trì con trỏ tới các bộ đệm gói tin) với gói tin mới thông qua chức năng thư viện DPDK.

- Nếu các mô tả gói tin mới có sẵn trong ring, ứng dụng sẽ truy cập đến các bộ đệm gói tin trong DPDK mempool thông qua con trỏ trong mô tả gói tin.

- Nếu không có gói tin nào, ứng dụng sẽ thăm dò NIC thông qua poll mode driver library và sau đó cấu trúc ring được truy cập lần nữa.

4. Tham khảo

- http://dpdk.org/doc/guides/prog_guide/ — a detailed (but confusing in some places) description of all the DPDK libraries;
- http://www.it-sobytie.ru/system/attachments/files/000/001/102/original/LinuxPiter-DPDK-2015.pdf — a presentation explaining the DPDK structure.
- https://www.cse.iitb.ac.in/synerg/lib/exe/fetch.php?media=public:students:mitali:rnd_report.pdf