DPDK-accelerated Partial Offload for Fine-grained HQoS

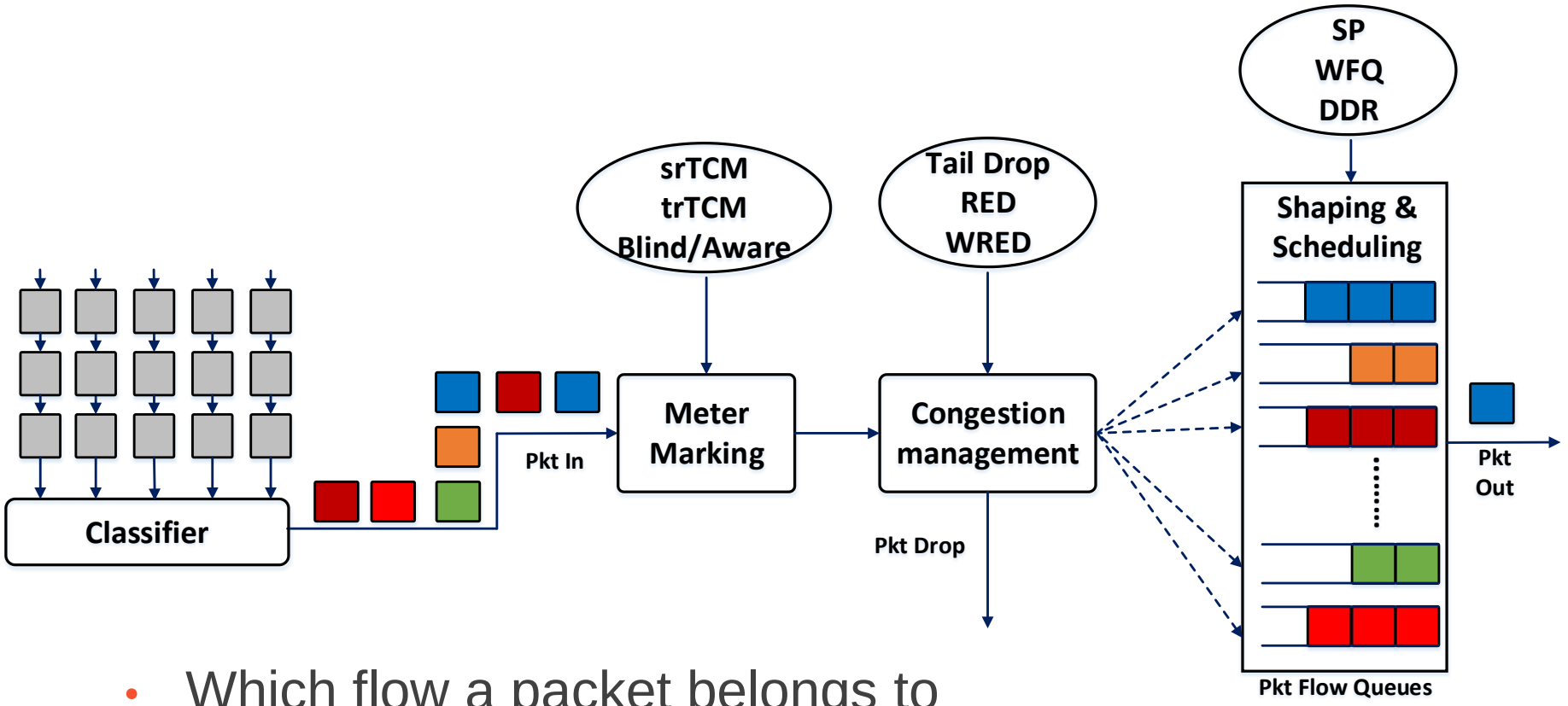
ROSEN XU, QINGSONG WANG

INTEL

Agenda

- A generic HQoS Use-case Overview
- Existing DPDK HQoS Capabilities
- Hardware Limitation for Fine-grained HQoS
- FPGA based Fine-grained HQoS reference design in Intel PAC N3000
- Fine-grained Partial Offload HQoS

A generic HQoS Use-case Overview



- Which flow a packet belongs to
- When a packet becomes eligible for scheduling
- What order to schedule packet flows

Existing DPDK HQoS Capabilities

```
struct rte_tm_ops {
    /** Traffic manager node type get */
    rte_tm_node_type_get_t node_type_get;

    /** Traffic manager capabilities_get */
    rte_tm_capabilities_get_t capabilities_get;
    /** Traffic manager level capabilities_get */
    rte_tm_level_capabilities_get_t level_capabilities_get;
    /** Traffic manager node capabilities get */
    rte_tm_node_capabilities_get_t node_capabilities_get;

    /** Traffic manager WRED profile add */
    rte_tm_wred_profile_add_t wred_profile_add;
    /** Traffic manager WRED profile delete */
    rte_tm_wred_profile_delete_t wred_profile_delete;
    /** Traffic manager shared WRED context add/update */
    rte_tm_shared_wred_context_add_update_t
        shared_wred_context_add_update;
    /** Traffic manager shared WRED context delete */
    rte_tm_shared_wred_context_delete_t
        shared_wred_context_delete;

    /** Traffic manager shaper profile add */
    rte_tm_shaper_profile_add_t shaper_profile_add;
    /** Traffic manager shaper profile delete */
    rte_tm_shaper_profile_delete_t shaper_profile_delete;
    /** Traffic manager shared shaper add/update */
    rte_tm_shared_shaper_add_update_t shared_shaper_add_update;
    /** Traffic manager shared shaper delete */
    rte_tm_shared_shaper_delete_t shared_shaper_delete;

    ......
}
```

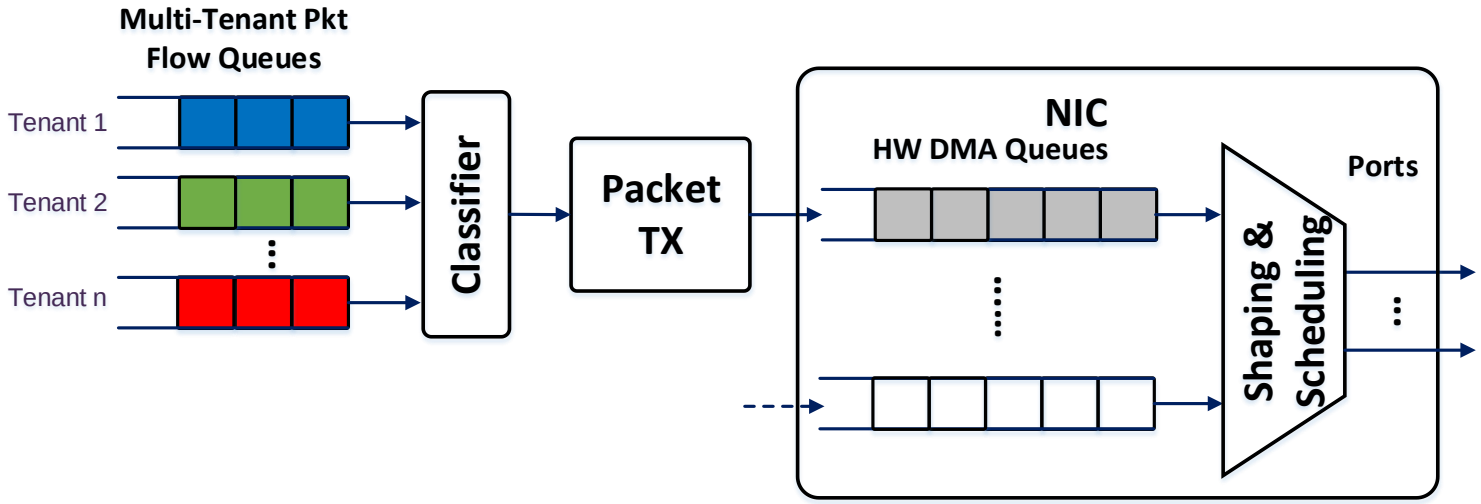
Meter

Congestion management

Shaping

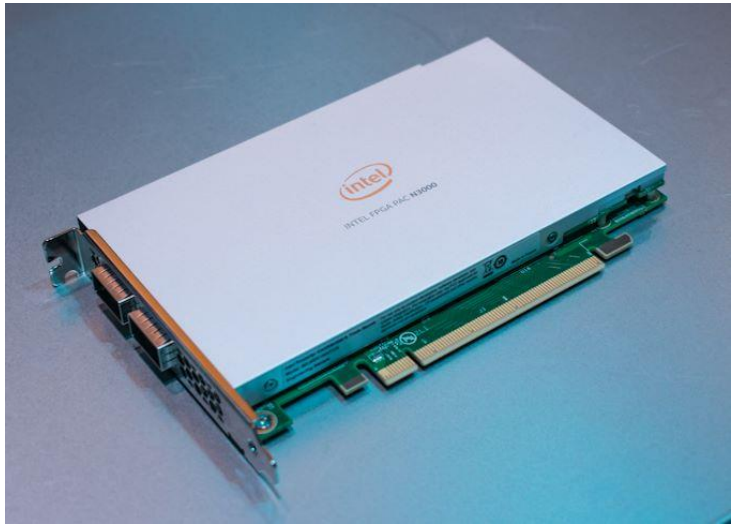
```
static inline enum rte_color
rte_meter_srtcm_color_blind_check(struct rte_meter_srtcm *m,
    struct rte_meter_srtcm_profile *p,
    uint64_t time,
    uint32_t pkt_len);

static inline enum rte_color
rte_meter_srtcm_color_aware_check(struct rte_meter_srtcm *m,
        struct rte_meter_srtcm_profile *p,
        uint64_t time,
        uint32_t pkt_len,
        enum rte_color pkt_color);

static inline enum rte_color
rte_meter_trtcm_color_blind_check(struct rte_meter_trtcm *m,
            struct rte_meter_trtcm_profile *p,
            uint64_t time,
            uint32_t pkt_len);
static inline enum rte_color
rte_meter_trtcm_color_aware_check(struct rte_meter_trtcm *m,
                struct rte_meter_trtcm_profile *p,
                uint64_t time,
                uint32_t pkt_len,
                enum rte_color pkt_color);

static inline enum rte_color __rte_experimental
rte_meter_trtcm_rfc4115_color_blind_check(
        struct rte_meter_trtcm_rfc4115 *m,
        struct rte_meter_trtcm_rfc4115_profile *p,
        uint64_t time,
        uint32_t pkt_len);
static inline enum rte_color __rte_experimental
rte_meter_trtcm_rfc4115_color_aware_check(
            struct rte_meter_trtcm_rfc4115 *m,
            struct rte_meter_trtcm_rfc4115_profile *p,
            uint64_t time,
            uint32_t pkt_len,
            enum rte_color pkt_color);
```

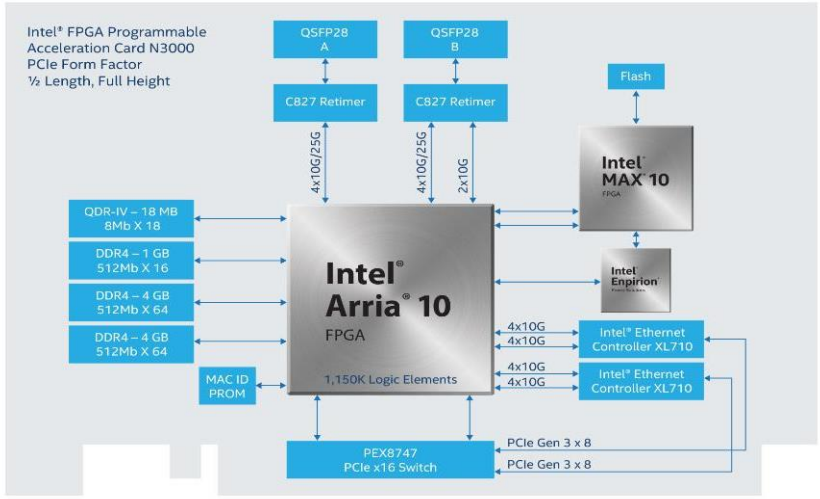
HW Limitation for Fine-grained HQoS



- Multi-tenant cloud network requires tens of thousands of flows
- Limited and none-scalable Hardware Queue Number
- Lack of flexibility for packet scheduler than software
- Limitation of enough Board Memory to keep packets

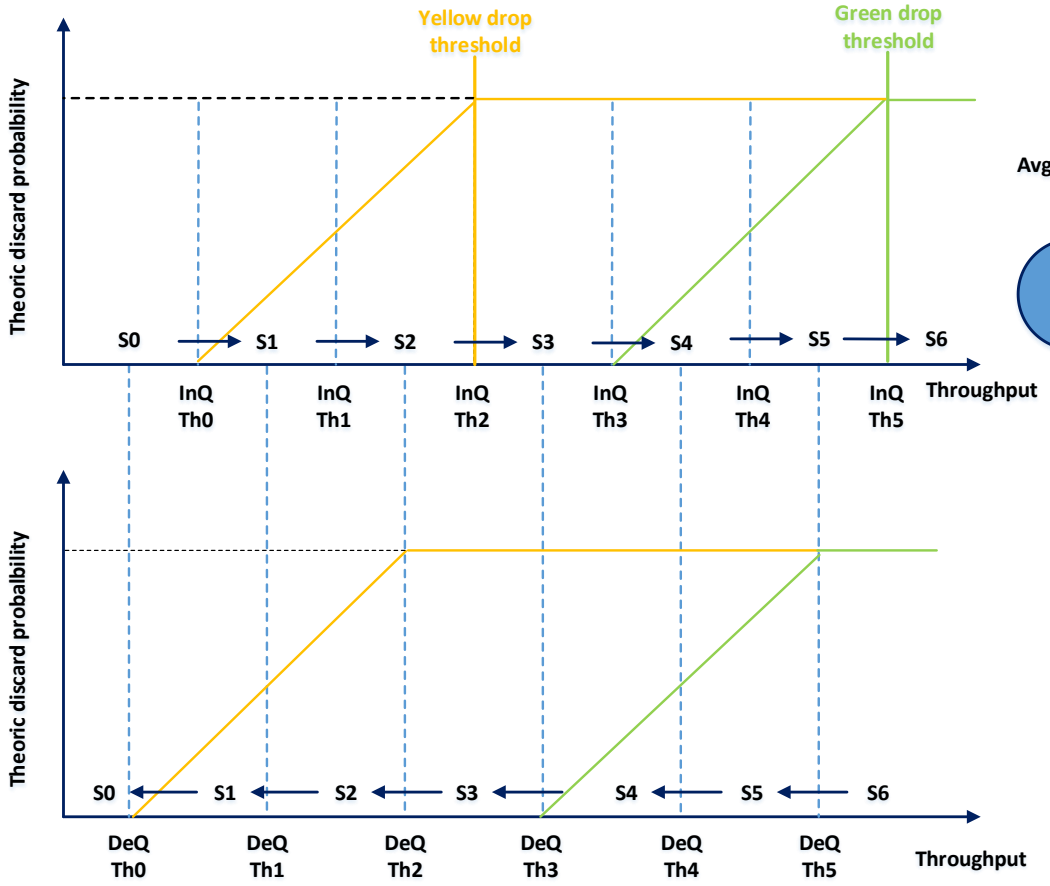
Fine-grained HQoS reference design – Intel PAC N3000

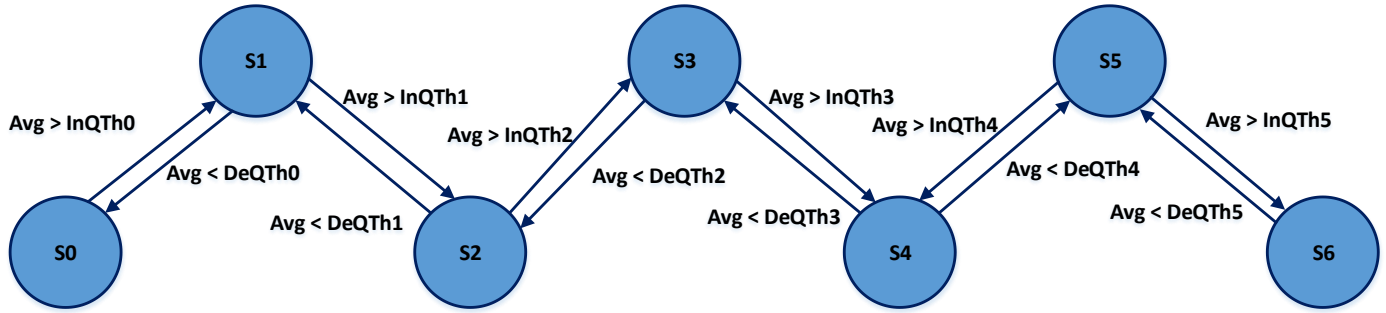




- Intel® Arria® 10 GT1150
  - 8x10GbE , 4x25GbE Network Interfaces
  - Local DDR4 and QDR Memory
- Dual Intel® Ethernet Controller XL710
  - Extensive OS support and Easier system integration

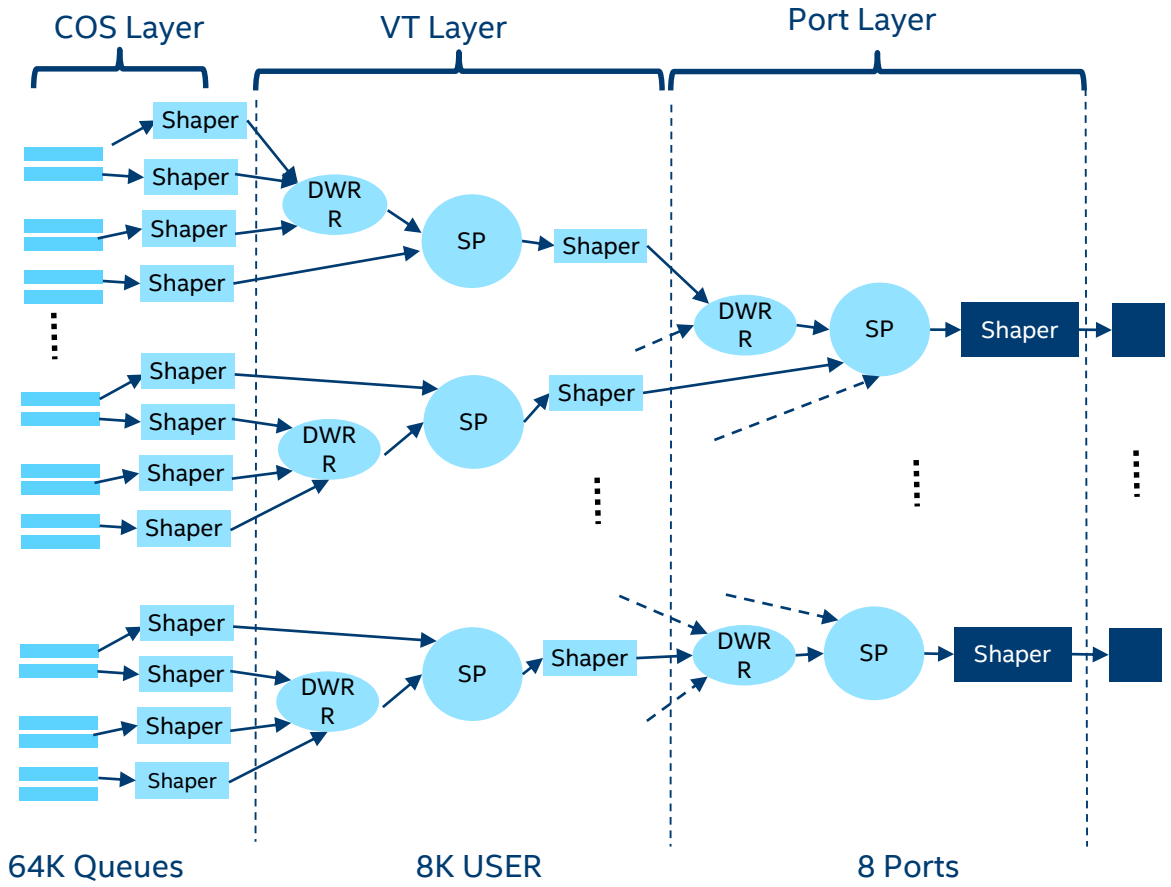
Fine-grained HQoS reference design - WRED





- Every traffic class has 7 states, different threshold for Inqueue and Dequeue
- WRED calculates average queue length when packets Inqueue and Dequeuer
- Packet drop or not depends on its color and current queue status

Fine-grained HQoS reference design- Shaper & Sche



- Queuing
  - Queue ID from Metadata
  - Traffic enqueued into 64K queues in Board Memory
- Scheduling
  - 3-layer scheduler
  - SP+DWRR
- Shaping
  - Input queue shaper per scheduler layer and port shaper
- Statistics
  - Port packet + byte counters

Intel PAC N3000 HQoS API

```
/* IPN3KE TM Node */
struct ipn3ke_tm_node {
    TAILQ_ENTRY(ipn3ke_tm_node) node;
    uint32_t node_index;
    uint32_t level;
    uint32_t tm_id;
    enum ipn3ke_tm_node_state node_state;
    uint32_t parent_node_id;
    uint32_t priority;
    uint32_t weight;
    struct ipn3ke_tm_node *parent_node;
    struct ipn3ke_tm_shaper_profile shaper_profile;
    struct ipn3ke_tm_tdrop_profile *tdrop_profile;
    struct rte_tm_node_params params;
    struct rte_tm_node_stats stats;
    uint32_t n_children;
    struct ipn3ke_tm_node_list children_node_list;
};

/* IPN3KE TM Hierarchy Specification */
struct ipn3ke_tm_hierarchy {
    struct ipn3ke_tm_node *port_node;
    /*struct ipn3ke_tm_node_list vt_node_list;*/
    /*struct ipn3ke_tm_node_list cos_node_list;*/

    uint32_t n_shaper_profiles;
    /*uint32_t n_shared_shapers;*/
    uint32_t n_tdrop_profiles;
    uint32_t n_vt_nodes;
    uint32_t n_cos_nodes;

    struct ipn3ke_tm_node *port_commit_node;
    struct ipn3ke_tm_node_list vt_commit_node_list;
    struct ipn3ke_tm_node_list cos_commit_node_list;

    /*uint32_t n_tm_nodes[IPN3KE_TM_NODE_LEVEL_MAX];*/
};
```

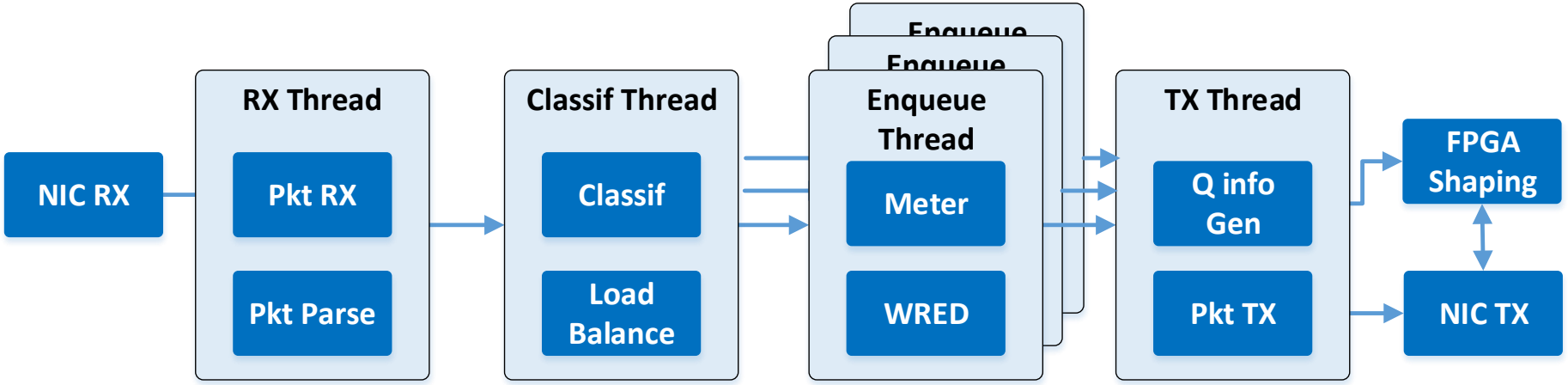
```
const struct rte_tm_ops ipn3ke_tm_ops = {
    .node_type_get = ipn3ke_pmd_tm_node_type_get,
    .capabilities_get = ipn3ke_tm_capabilities_get,
    .level_capabilities_get = ipn3ke_tm_level_capabilities_get,
    .node_capabilities_get = ipn3ke_tm_node_capabilities_get,

    .wred_profile_add = ipn3ke_tm_tdrop_profile_add,
    .wred_profile_delete = ipn3ke_tm_tdrop_profile_delete,

    .shaper_profile_add = ipn3ke_tm_shaper_profile_add,
    .shaper_profile_delete = ipn3ke_tm_shaper_profile_delete,

    .node_add = ipn3ke_tm_node_add,
    .node_delete = ipn3ke_pmd_tm_node_delete,
    .hierarchy_commit = ipn3ke_tm_hierarchy_commit,
};
```

- 2018’Q3 DPDK with FPGA TM Offloading POC, enable 64k flows
- 2019’Q1 DPDK with PAC N3000 FPGA TM Offloading upstream 64K flows

Summary of FPGA based HQoS

- More configurable and flexible than NIC
- Provide Fine-grained HQoS to Multi-tenant cloud network
- Need lots of Board Memory to keep all packets
- Can software take up some parts of HQoS?

Fine-grained Partial Offload HQoS



- [RX Thread] Packet receive and parse
- [Classifier Thread] Packet classifier, divides input packets into different flow, load balance and dispatch flow to different queue
- [Enqueue Thread] Meter, mark color and calculate WRED for different queue and enqueue
- [TX Thread] Generate queue info for FPGA Shaping, packets are sent from NIC DMA


DPDK
DATA PLANE DEVELOPMENT KIT
Q & A