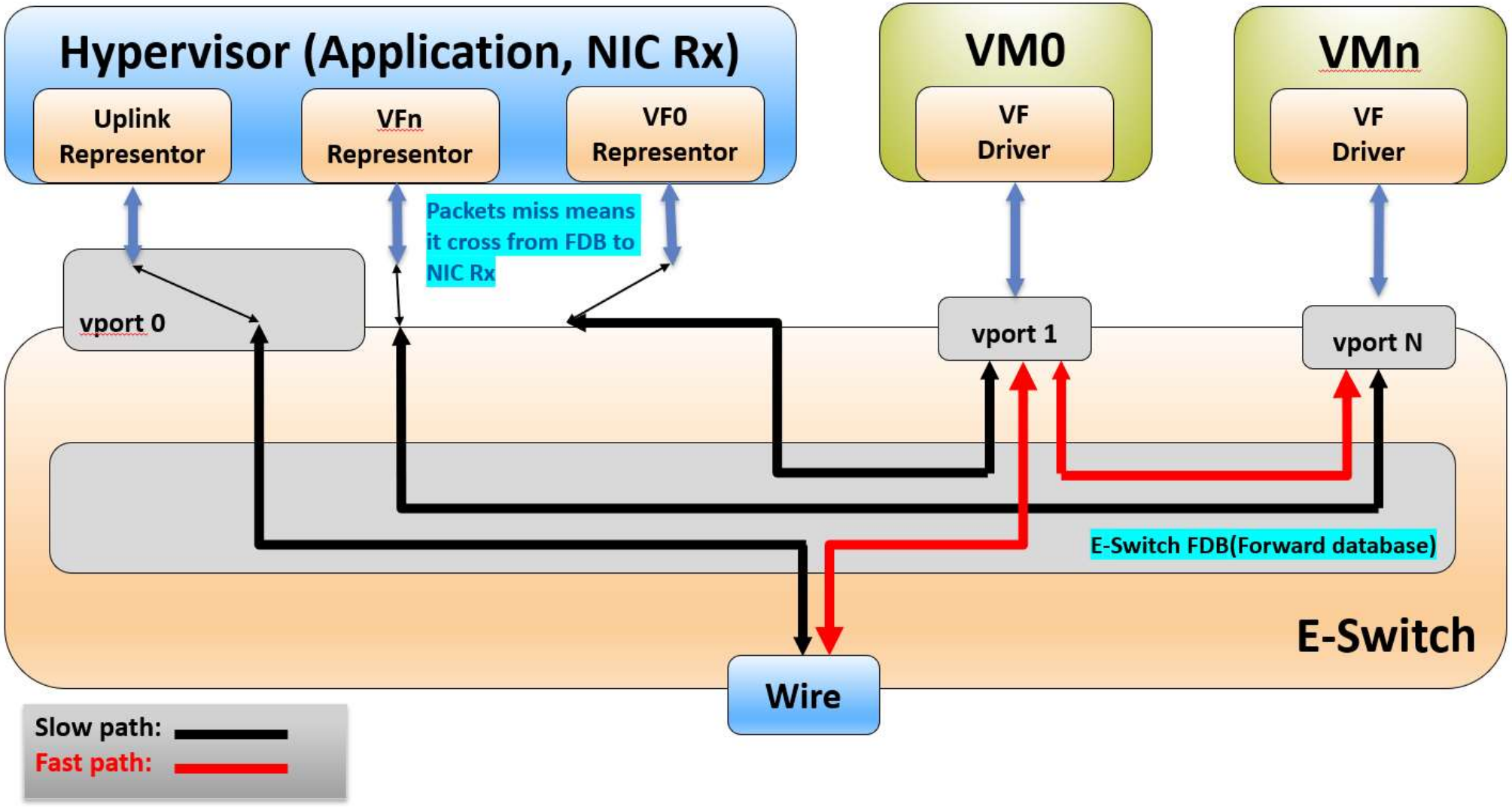
DPDK
SUMMIT APAC

Unified representor with large scale ports

Suanming Mou - Nvidia

Classic switch model

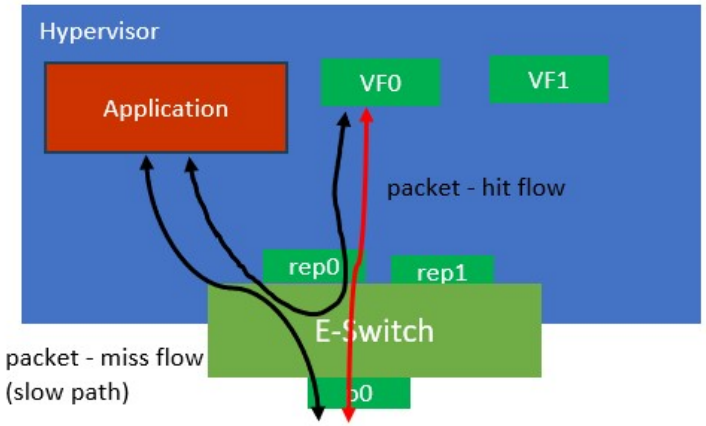


Switching model handling example

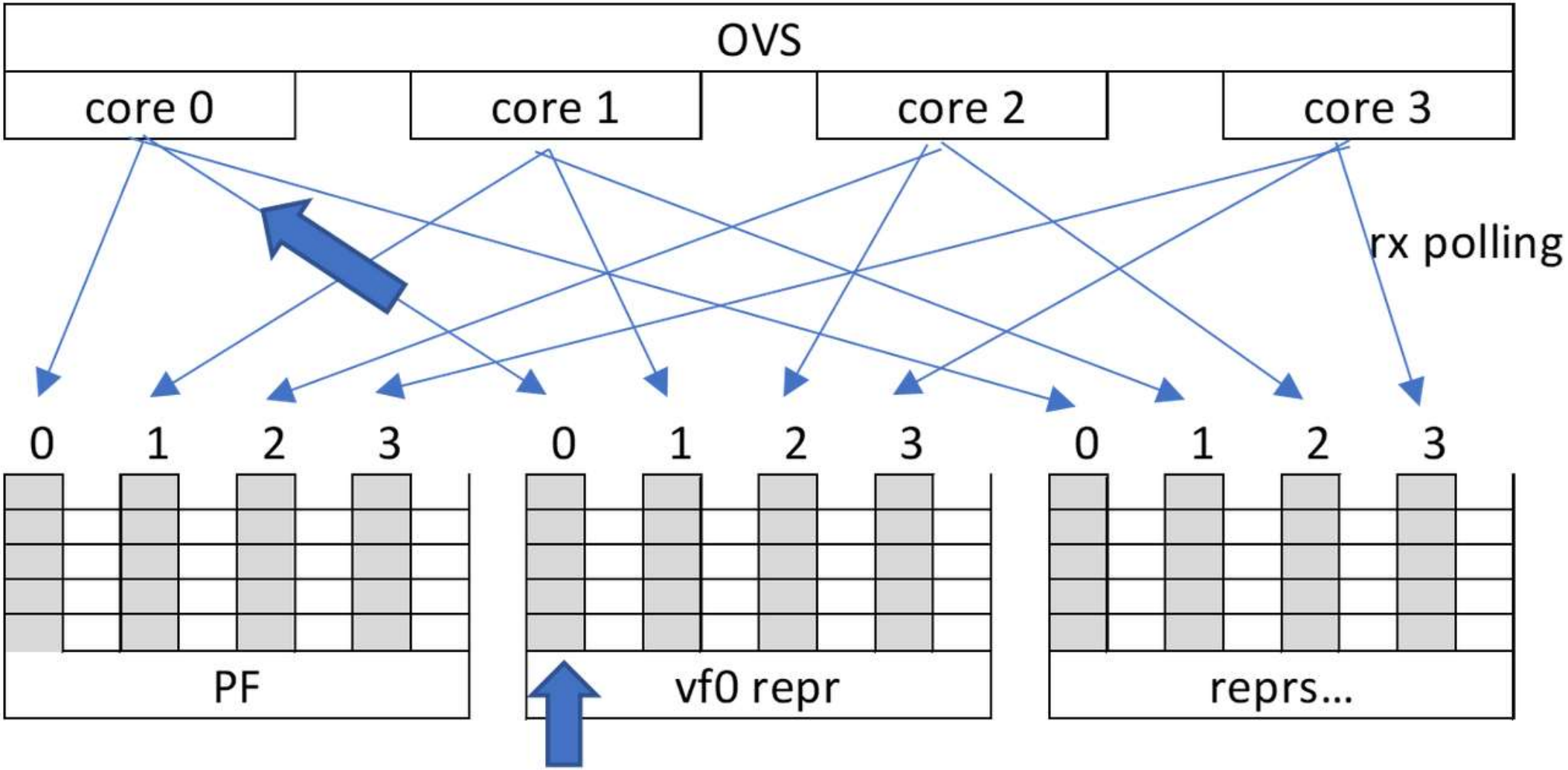
- Slow path:
  Packet missed from wire will be received by application from PF.
  Packet missed from VFn/VMn will be received by application from VFn representor port.
  (Packet missed from FDB domain to NIC Rx domain)

- Fast path:
  Application will offload flow rules for the missed packets and next time the packets will be handled by HW offloaded flow rules.(Handled by FDB domain flow rules)

- Challenge:
  Large scale of ports? Think about allocating and preparing mbuf memory, setup queues and polling all the queues for each port.



The polling mode in slow path



Challenges of large number representor ports

- OVS polls PF and all registered representors:
  Usually, N cores and N queues each port, core <x> polls queue <x> of all ports

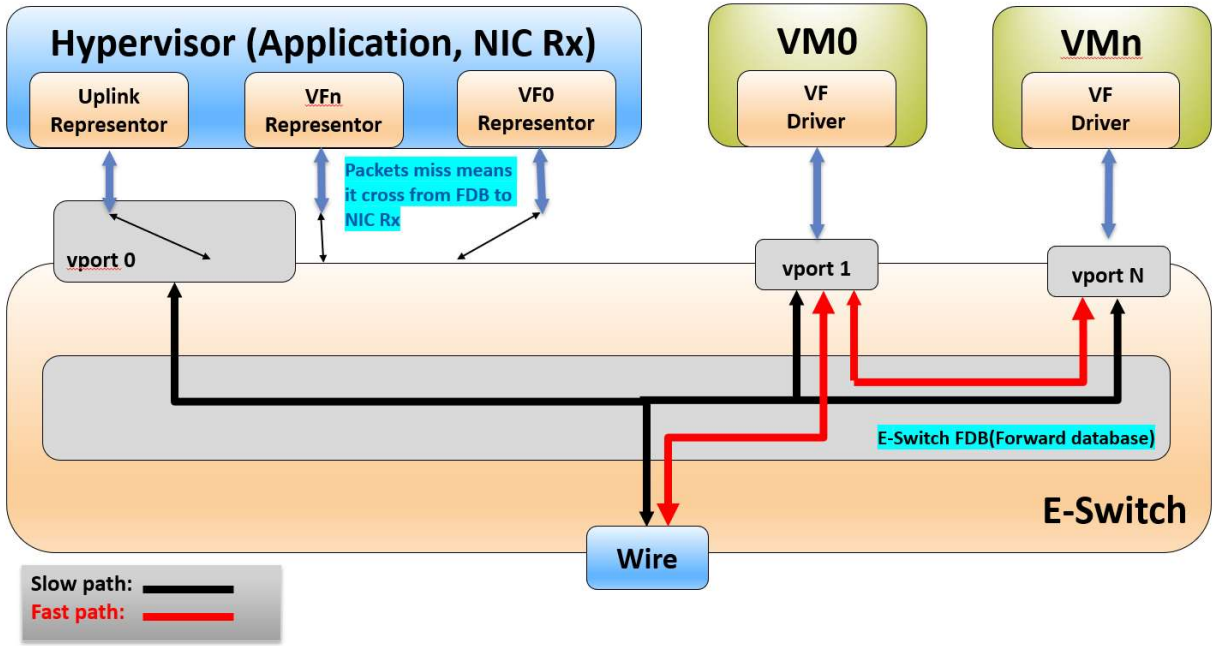
- High memory consumption:
  For 1k SF+VF representor ports: 4 * 1024(rxd) * 3kB(mbuf) * 1024(ports) = 12GB

- High CPU usage, high cache miss:
  Cache get flushed when polling so many ports, low performance

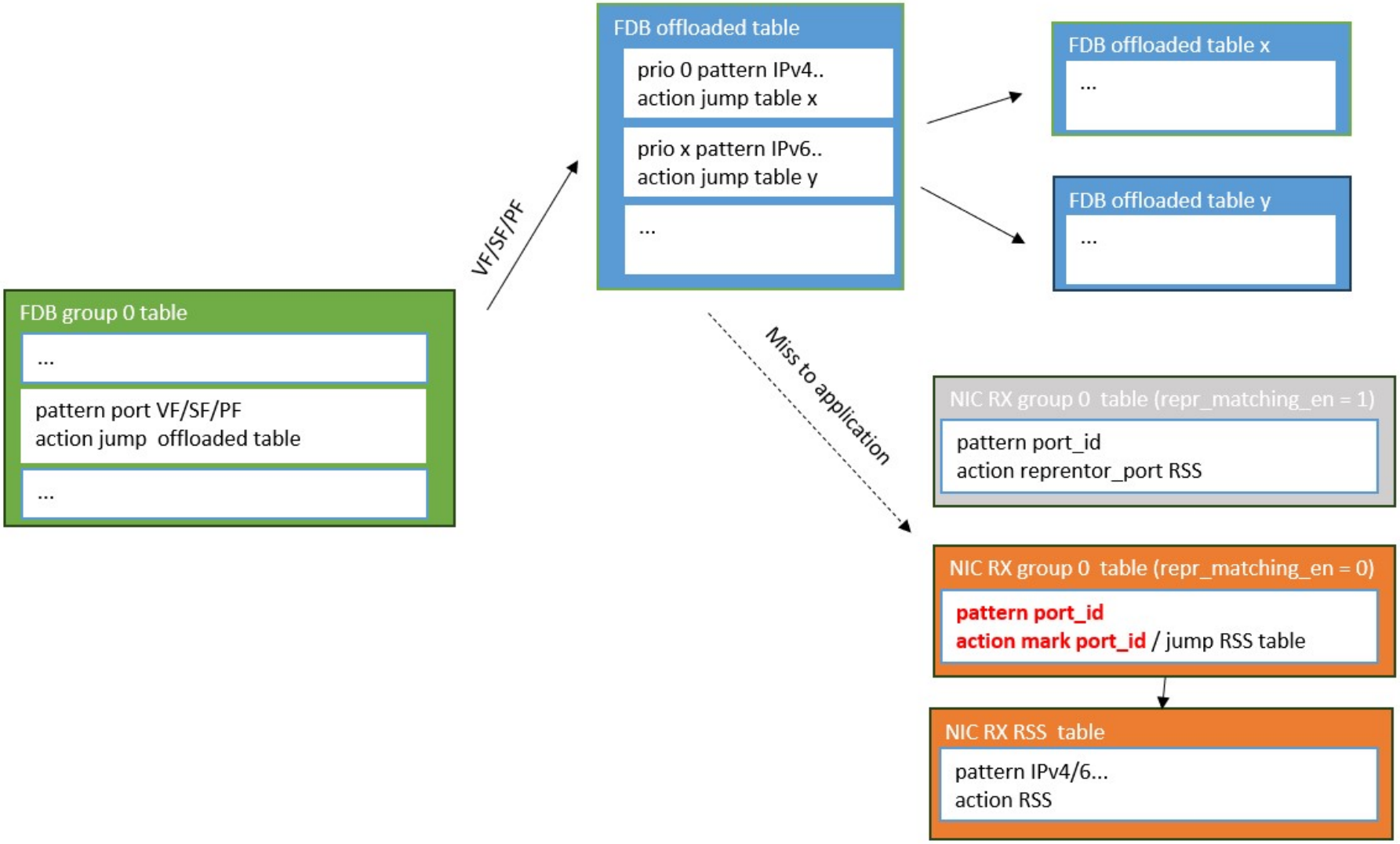
- High latency:
  CPU handles traffic not offloaded(customized crypto, legacy tunnel)
  Even if all packet come from one VF, OVS must poll all ports to get next burst

Optimization in mlx5 PMD - flows

- The port to representor behavior is controlled by **repr_matching_en** devarg in mlx5 PMD(1 by default). **repr_matching_en = 1**, the internal flow rules will steer the packets to relevant representor port.
- Setting **repr_matching_en = 0**, add flow rules to E-Switch manager, let the flow rules to steer all the packets to the PF(uplink as proxy) port.
- The offloaded flow rule will also copy the source port_id information to packet metadata(CQE). And later the port_id will be written to mbuf in rx_burst function.



Flow rule tables

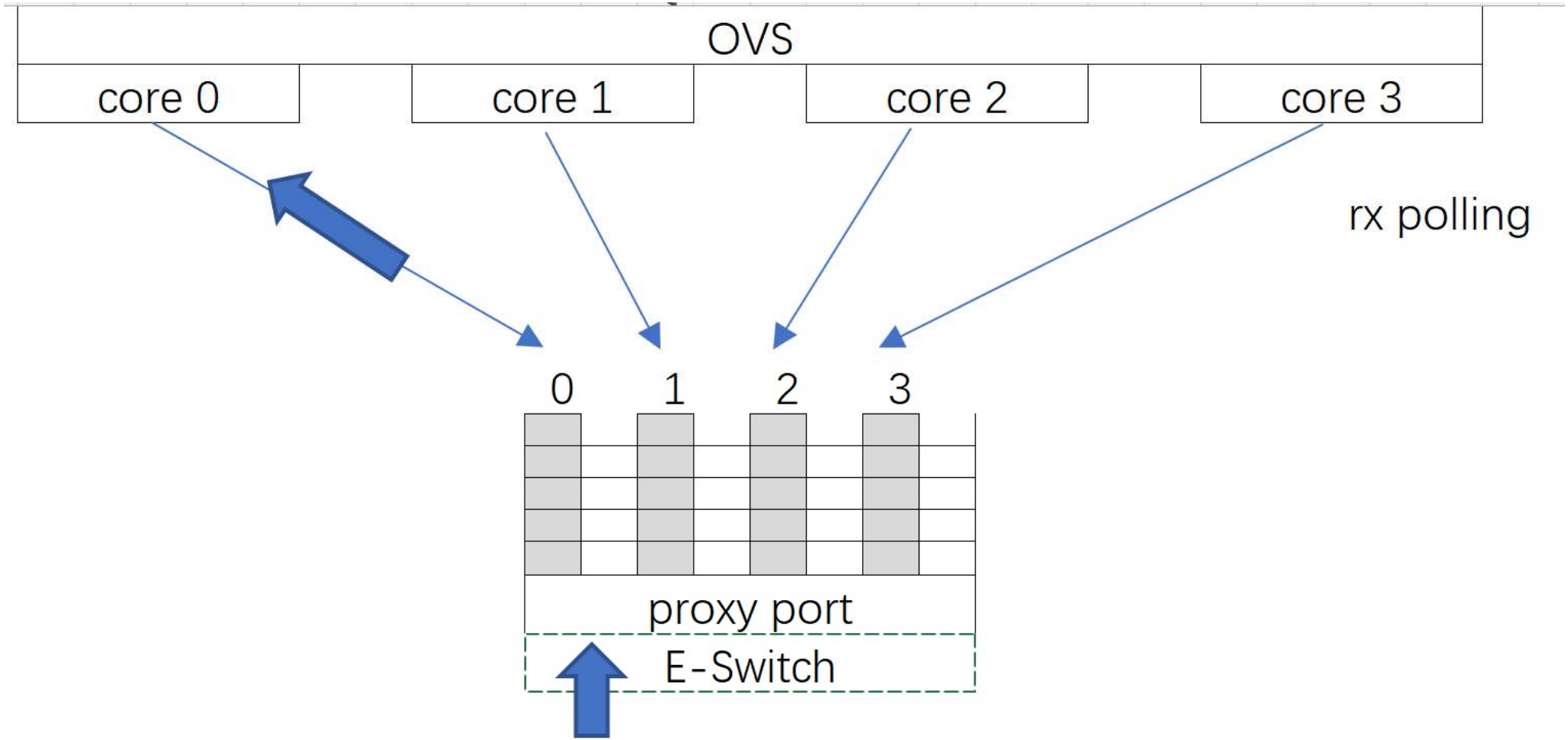


Optimization – queue preparation

- While **repr_matching_en** = 0, no need to configure and setup the datapath queues for VF/SF ports anymore. That will save huge numbers of memory.

```
/* Configure the Ethernet device for proxy port only */
ret = rte_eth_dev_configure(port, rx_rings, tx_rings, &port_conf);

/* Allocate and set up RX queues according to number of cores */
for (q = 0; q < rx_rings; q++) {
    ret = rte_eth_rx_queue_setup(port, q, RX_RING_SIZE, rte_eth_dev_socket_id(port), NULL, mbuf_pool);
    if (ret < 0) {
        …
    }
}

/* Allocate and set up TX queues according to number of cores */
for (q = 0; q < tx_rings; q++) {
    ret = rte_eth_tx_queue_setup(port, q, TX_RING_SIZE, rte_eth_dev_socket_id(port), NULL);
    if (ret < 0) {
        ..
    }
}
```

The simplified polling mode

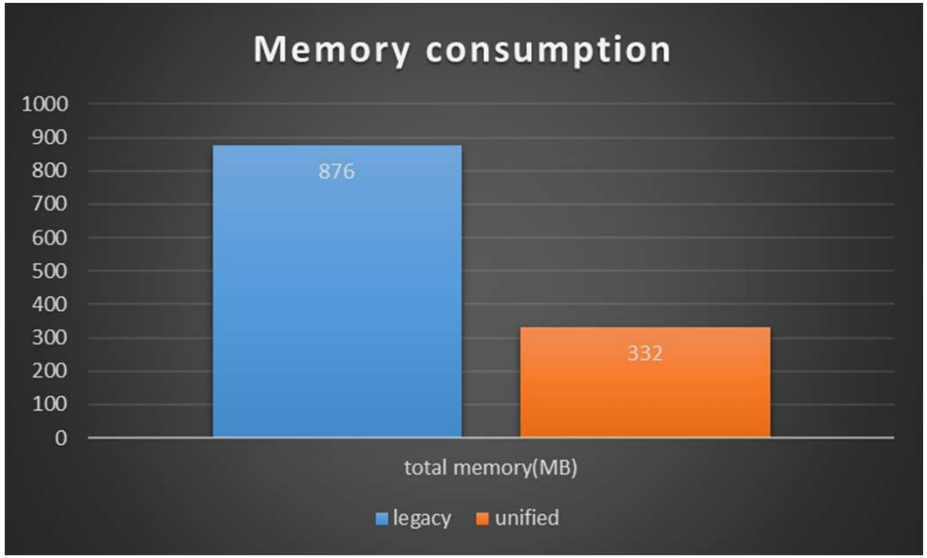


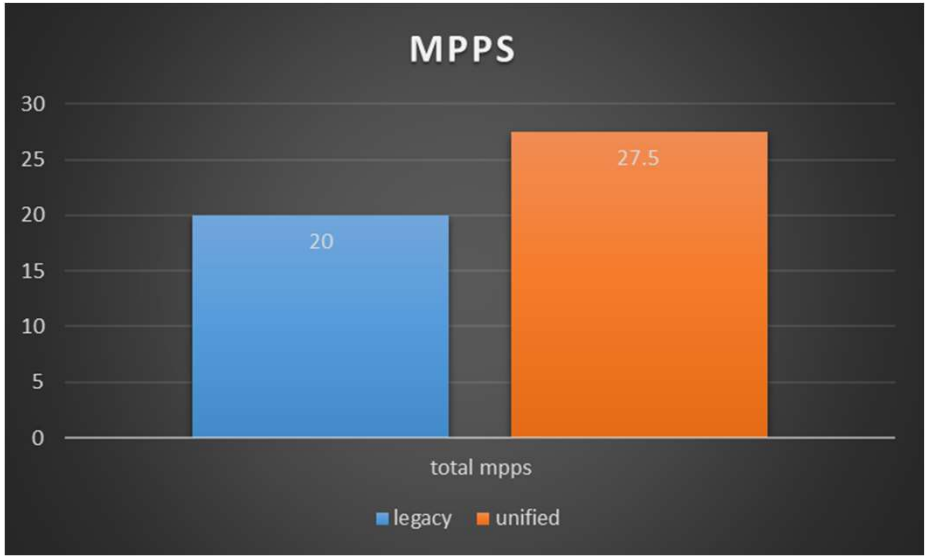
The polling differences

```
/* Before */
while (!force_quit) {
        for (port_id = 0; port_id < total_port; port_id++) {
                nb_rx = rte_eth_rx_burst(port_id, queue_id, mbufs, MAX_PKTS);
                if (!nb_rx)
                        continue;
                for (i = 0; i < nb_rx; i++) {
                    /* application logic */
                }
                /* Tx packet */
        }
}

/* After */
while (!force_quit)
{

    nb_rx = rte_eth_rx_burst(port_id, queue_id, mbufs, MAX_PKTS);
    if (!nb_rx)
        continue;
    for (i = 0; i < nb_rx; i++) {
        src_port_id = mbufs[i]->hash.fdir.hi;
        /* application logic */
    }
    /* Tx packet */
}
```

Results

- In the test model with 4 (1PF + 3VFs) ports and 8 queues with 8 cores (1 queue per core), use traffic generator to send 64bytes packets to wire port. Comparing the memory consumption and pps:






DPDK
SUMMIT APAC