DPDK & DUNE-DAQ

Roland Sipos for UDAQ

Ethernet readoutm meeting
14th October 2021

Intro

- DPDK
- Integration with DUNE-DAQ
- Next steps

DPDK

DPDK is the Data Plane Development Kit that consists of libraries to accelerate packet processing workloads running on x86, POWER and ARM processors.

Licensed under the Open Source BSD License.

DPDK

Most of the codebase developed by Intel software and hardware engineers, but got substantial additions also from Mellanox, other bigger companies, and the general public.

Safe to say, that it became an industrial standard in the last few years.

Gold Members

arm AT&T ERICSSON f5
intel MARVELL Microsoft NVIDIA
NXP Red Hat ZTE

Silver Members

6WIND AMD BROADCOM HUAWEI

Associate Members

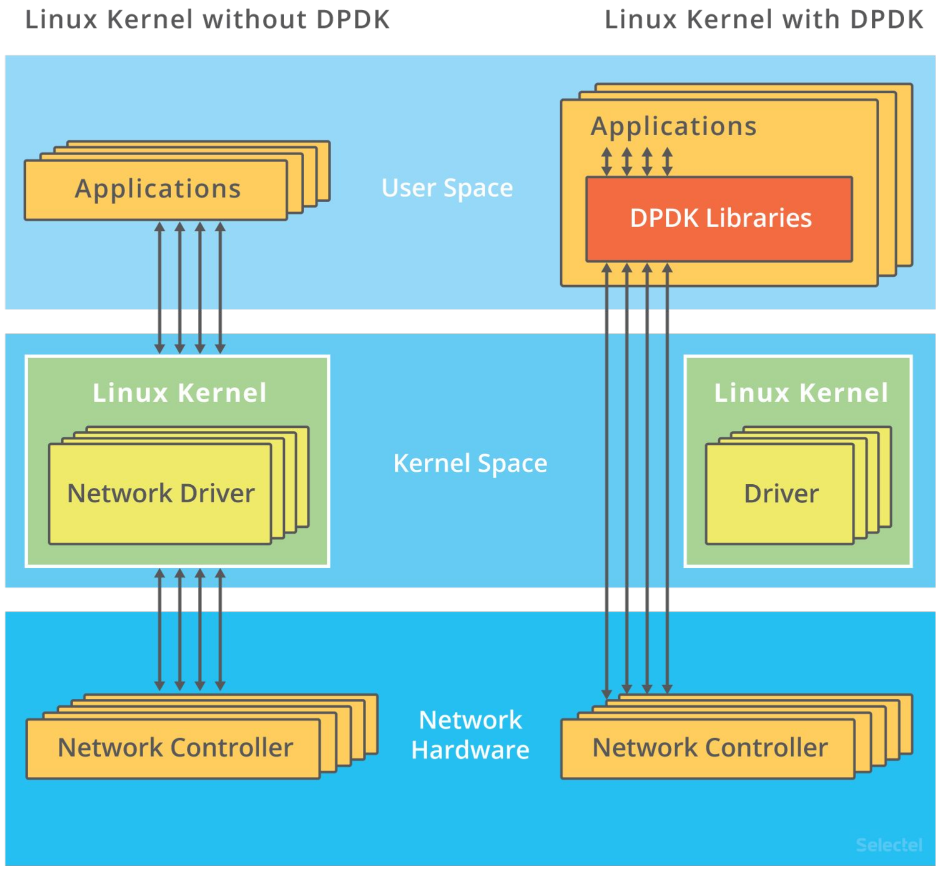
Eötvös Loránd University KAIST UMASS LOWELL University of Limerick
iol InterOperability Laboratory

Main principle

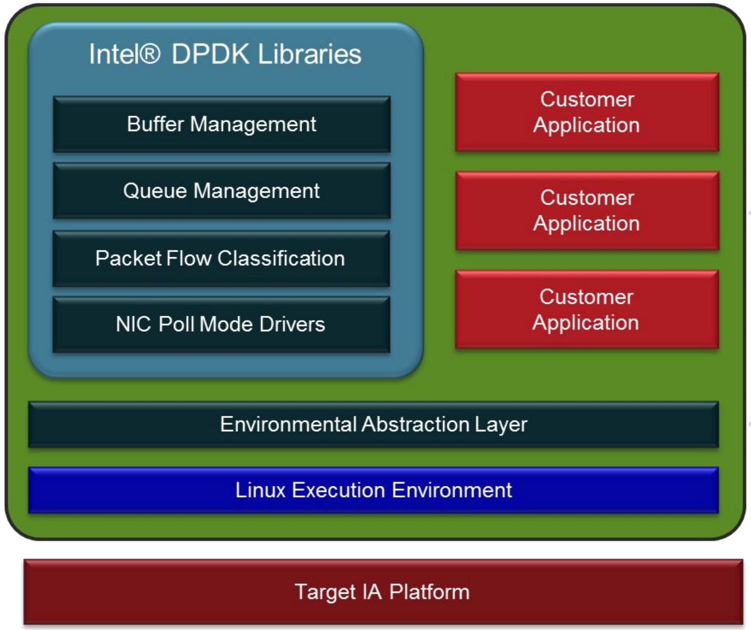
Bring packet processing via poll mode high performance drivers to user space.

Pre- 5.x kernel's interrupt based network driver has performance limitations.

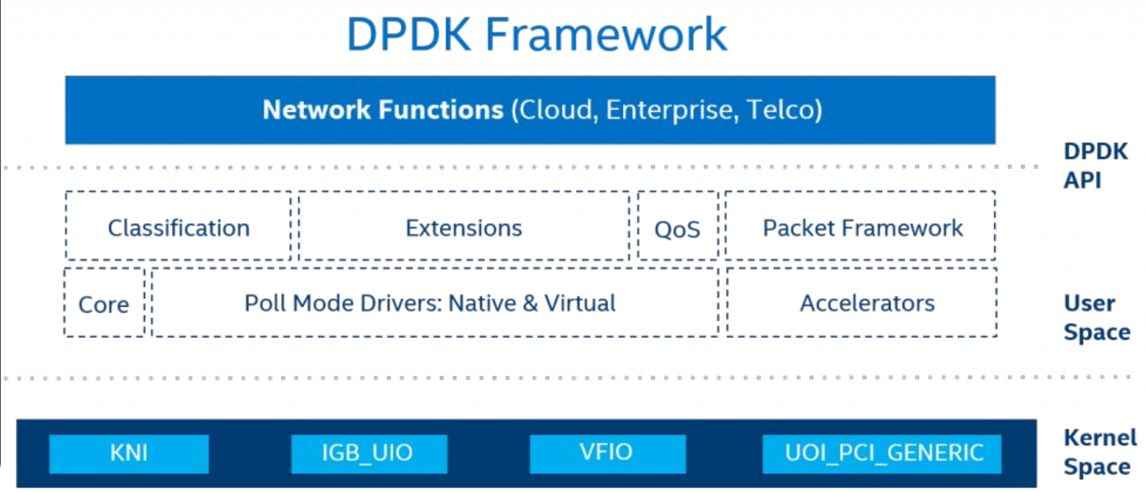
On the other hand, good to keep in mind, That post 5.x kernel has substantial improvements on networking, and some benchmarks and use-cases show close performance to DPDK variants.



Components



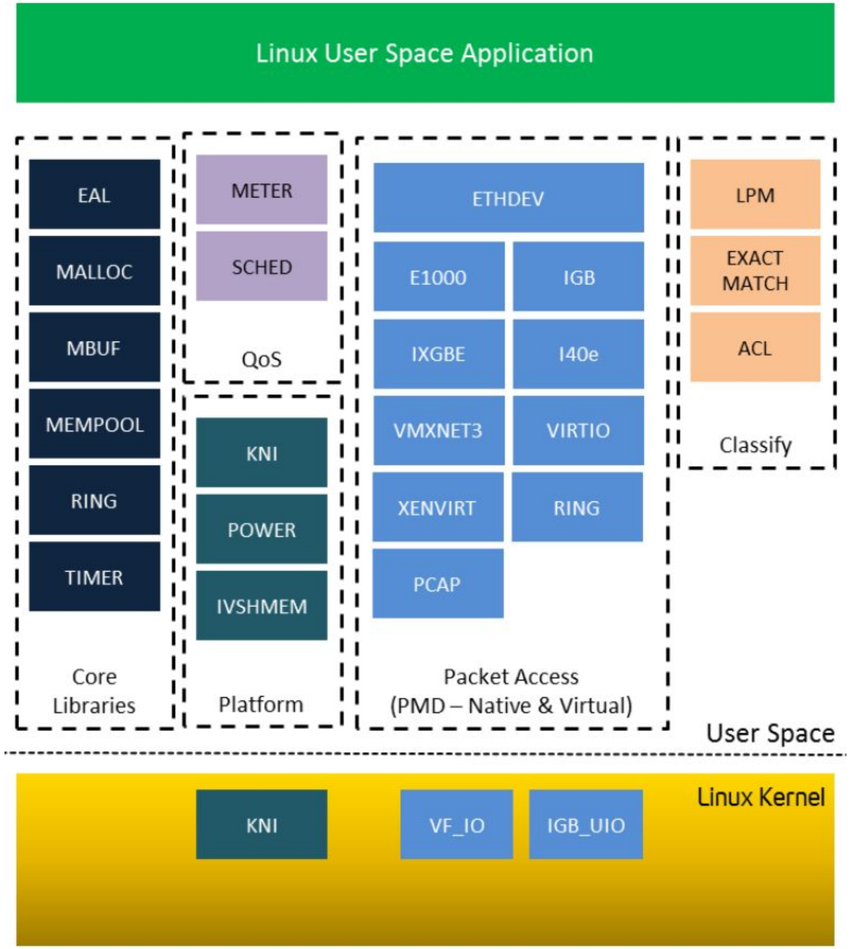
Intel® DPDK Libraries
Buffer Management
Queue Management
Packet Flow Classification
NIC Poll Mode Drivers
Customer Application
Customer Application
Customer Application
Environmental Abstraction Layer
Linux Execution Environment
Target IA Platform



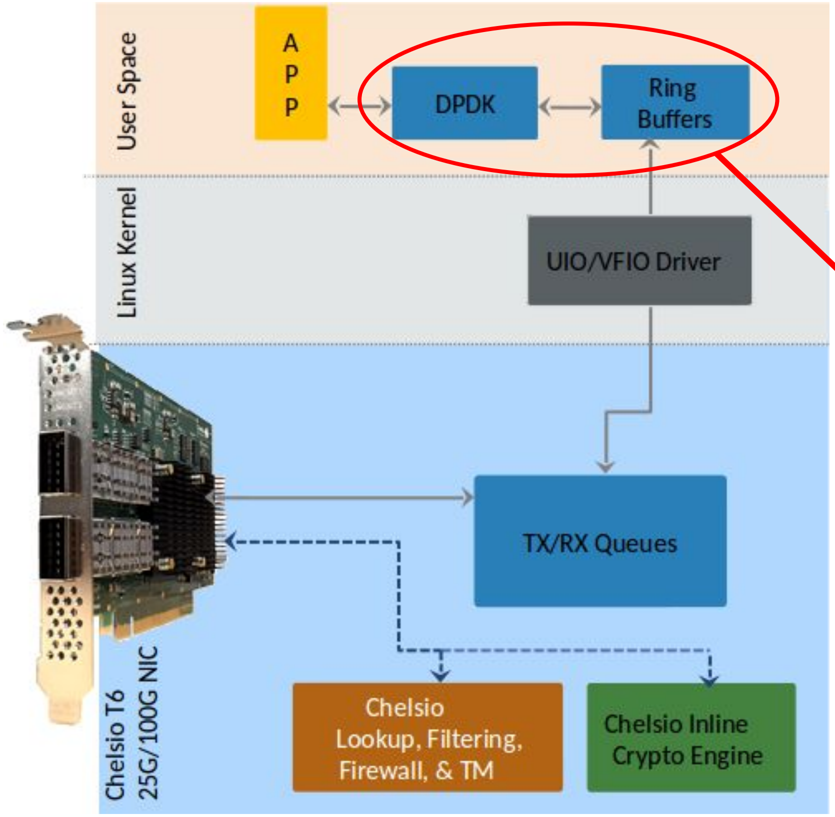
DPDK Framework
Network Functions (Cloud, Enterprise, Telco)
DPDK API
Classification
Extensions
QoS
Packet Framework
Core
Poll Mode Drivers: Native & Virtual
Accelerators
User Space
KNI
IGB_UIO
VFIO
UOI_PCI_GENERIC
Kernel Space

Toolkit

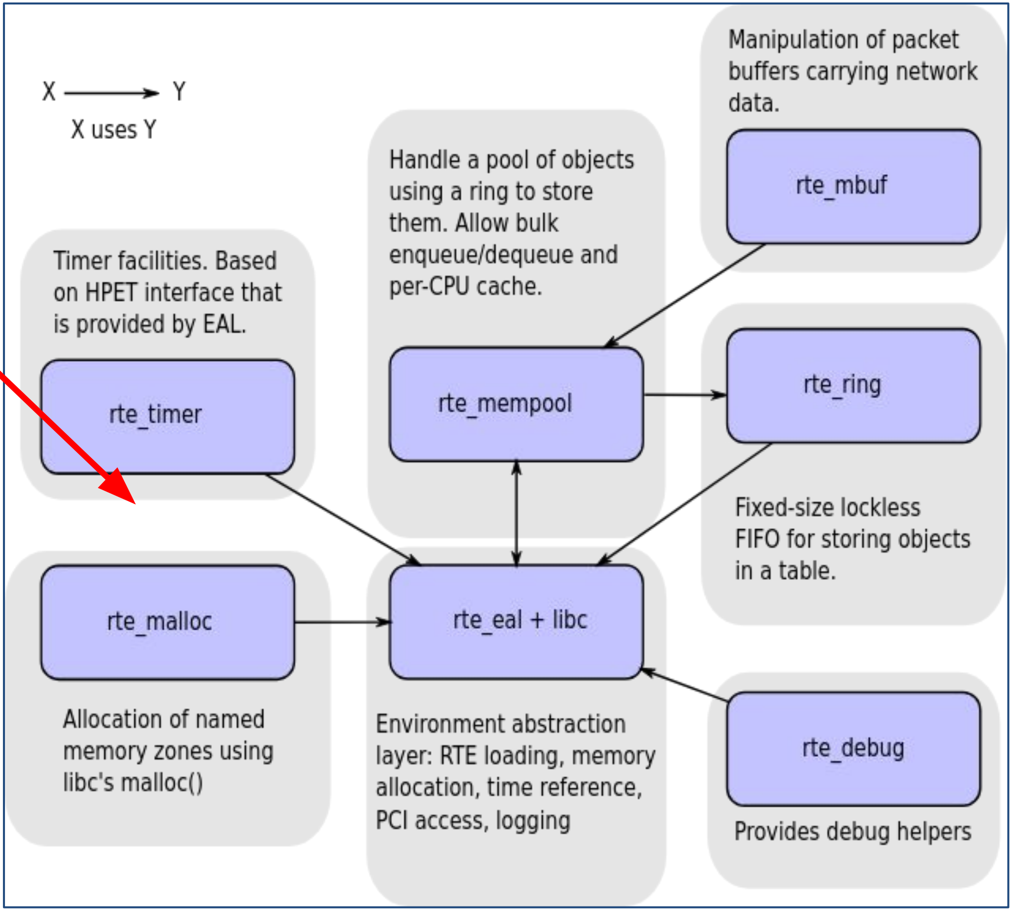
- Core libraries
  - EAL
  - Buffers
  - Timers
- Platform specifics
- Packet access
  - Drivers
- Classification
  - Algos, regex, etc.



Main idea



User Space
Linux Kernel
Chelsio T6 25G/100G NIC
APP
DPDK
Ring Buffers
UIO/VFIO Driver
TX/RX Queues
Chelsio Lookup, Filtering, Firewall, & TM
Chelsio Inline Crypto Engine



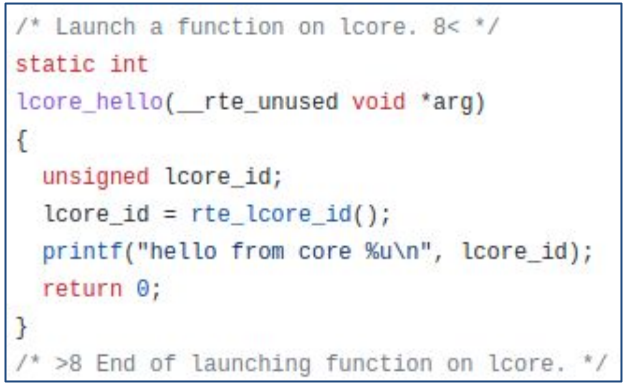
X → Y
X uses Y
Manipulation of packet buffers carrying network data.
rte_mbuf
Handle a pool of objects using a ring to store them. Allow bulk enqueue/dequeue and per-CPU cache.
rte_mempool
Timer facilities. Based on HPET interface that is provided by EAL.
rte_timer
rte_ring
Fixed-size lockless FIFO for storing objects in a table.
rte_malloc
Allocation of named memory zones using libc's malloc()
rte_eal + libc
Environment abstraction layer: RTE loading, memory allocation, time reference, PCI access, logging
rte_debug
Provides debug helpers

DUNE-DAQ integration

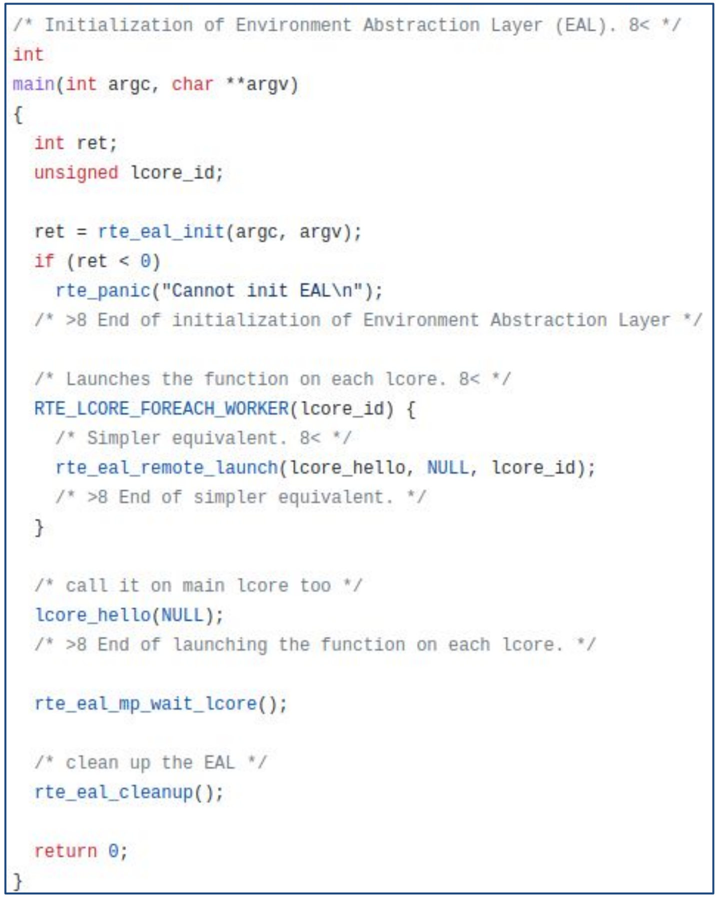
Dpdklibs - Libraries and plugins built against DPDK core and wrapping EAL.

Empty, with a dummy module.

Example application for EAL and lcore functions.



```
/* Launch a function on lcore. 8< */
static int
lcore_hello(__rte_unused void *arg)
{
  unsigned lcore_id;
  lcore_id = rte_lcore_id();
  printf("hello from core %u\n", lcore_id);
  return 0;
}
/* >8 End of launching function on lcore. */
```



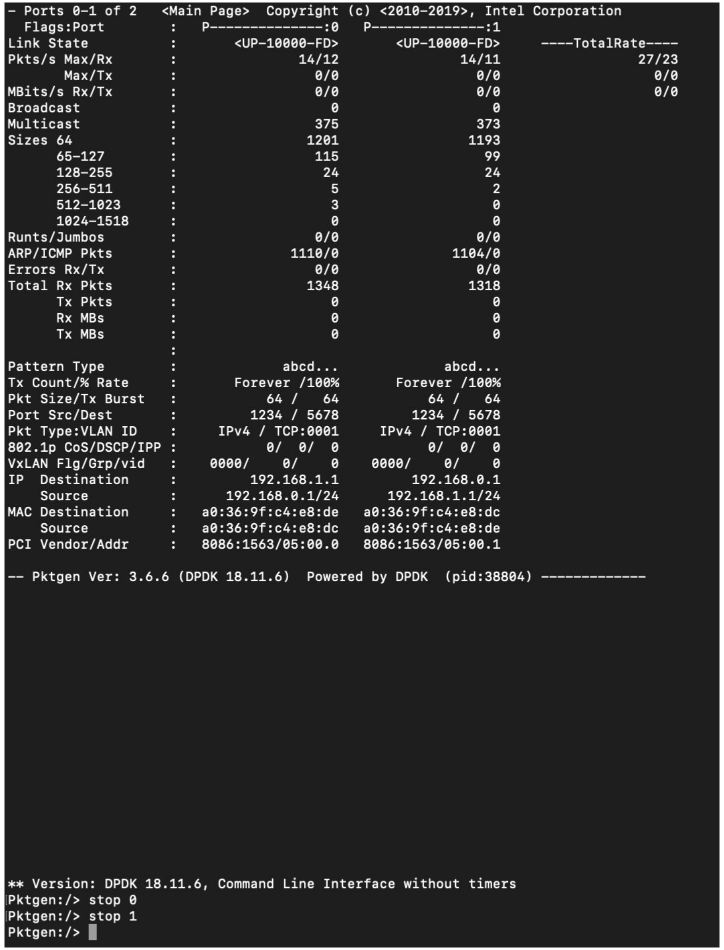
```
/* Initialization of Environment Abstraction Layer (EAL). 8< */
int
main(int argc, char **argv)
{
  int ret;
  unsigned lcore_id;

  ret = rte_eal_init(argc, argv);
  if (ret < 0)
    rte_panic("Cannot init EAL\n");
  /* >8 End of initialization of Environment Abstraction Layer */

  /* Launches the function on each lcore. 8< */
  RTE_LCORE_FOREACH_WORKER(lcore_id) {
    /* Simpler equivalent. 8< */
    rte_eal_remote_launch(lcore_hello, NULL, lcore_id);
    /* >8 End of simpler equivalent. */
  }

  /* call it on main lcore too */
  lcore_hello(NULL);
  /* >8 End of launching the function on each lcore. */

  rte_eal_mp_wait_lcore();

  /* clean up the EAL */
  rte_eal_cleanup();

  return 0;
}
```

Next steps

- Wrapper lib for EAL in C++/C hybrid
    - NIC port handler
    - Buffer and allocator manager

- Lcore processors for package processing
    - Wrap to ReadoutType in order to interface with readout
    - Connect to generic readout

- Software emulator of UDP packets
- (Port majority of dpdk-pktgen to make UDP packet generators from COTS NICs)



```
- Ports 0-1 of 2   <Main Page>  Copyright (c) <2010-2019>, Intel Corporation
  Flags:Port        :   P--------------:0   P--------------:1
Link State          :        <UP-10000-FD>       <UP-10000-FD>     ----TotalRate----
Pkts/s Max/Rx       :                14/12               14/11                 27/23
       Max/Tx       :                  0/0                 0/0                   0/0
MBits/s Rx/Tx       :                  0/0                 0/0                   0/0
Broadcast           :                    0                   0
Multicast           :                  375                 373
Sizes 64            :                 1201                1193
      65-127        :                  115                  99
      128-255       :                   24                  24
      256-511       :                    5                   2
      512-1023      :                    3                   0
      1024-1518     :                    0                   0
Runts/Jumbos        :                  0/0                 0/0
ARP/ICMP Pkts       :               1110/0              1104/0
Errors Rx/Tx        :                  0/0                 0/0
Total Rx Pkts       :                 1348                1318
      Tx Pkts       :                    0                   0
      Rx MBs        :                    0                   0
      Tx MBs        :                    0                   0
                    :
Pattern Type        :              abcd...             abcd...
Tx Count/% Rate     :        Forever /100%       Forever /100%
Pkt Size/Tx Burst   :            64 /   64           64 /   64
Port Src/Dest       :          1234 / 5678         1234 / 5678
Pkt Type:VLAN ID    :      IPv4 / TCP:0001     IPv4 / TCP:0001
802.1p CoS/DSCP/IPP :            0/   0/  0           0/   0/  0
VxLAN Flg/Grp/vid   :     0000/    0/    0    0000/    0/    0
IP  Destination     :          192.168.1.1         192.168.0.1
    Source          :       192.168.0.1/24      192.168.1.1/24
MAC Destination     :    a0:36:9f:c4:e8:de   a0:36:9f:c4:e8:dc
    Source          :    a0:36:9f:c4:e8:dc   a0:36:9f:c4:e8:de
PCI Vendor/Addr     :    8086:1563/05:00.0   8086:1563/05:00.1

-- Pktgen Ver: 3.6.6 (DPDK 18.11.6)  Powered by DPDK  (pid:38804) -------------






























** Version: DPDK 18.11.6, Command Line Interface without timers
Pktgen:/> stop 0
Pktgen:/> stop 1
Pktgen:/>
```

Outlook

Xilinx has