Technology Consulting Company
Research, Development &
Global Standard

Port Hot-plugging Framework for DPDK

November 2014

Tetsuya Mukawa
mukawa@igel.co.jp

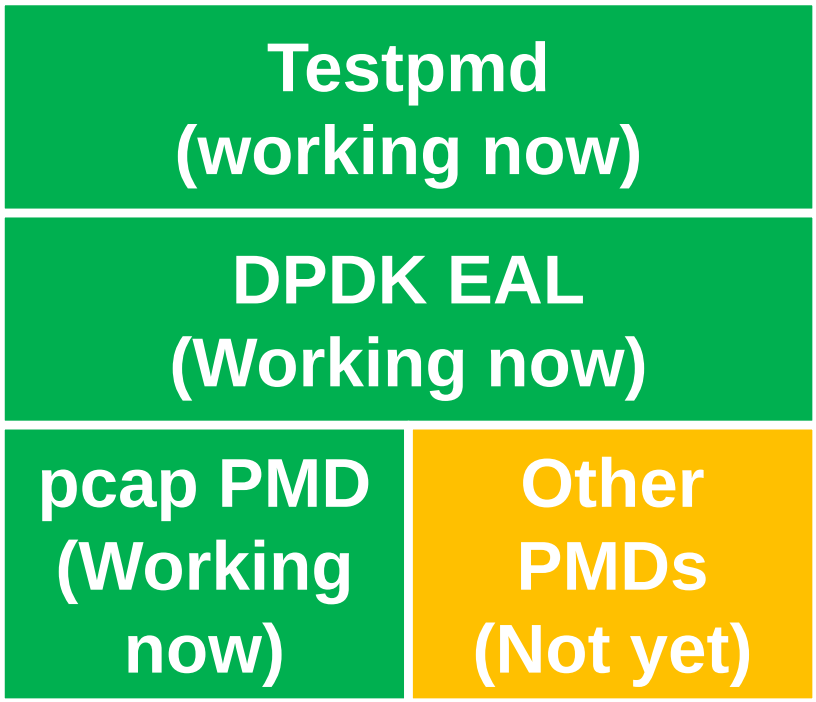
Overview

- I sent a patch-set to add a port hot-plugging feature into the DPDK EAL. It allows DPDK apps to attach or detach a physical or virtual device port at runtime.
- My implementation is straight-forward. When either the attach or detach function that my patch-set added is called, all the data structures for managing ports should be updated.
  - It assumes that added physical ports must be initialized appropriately through the PCI hot-plug framework in the kernel.

The Current Status of my Impl.

- We need to modify the following modules for the hot-plugging feature.
  - DPDK apps
    - testpmd … working now
  - EAL framework

    … working now
  - PMDs
    - pcap PMD … working now.
    - Other PMDs … not started yet.



- I’ve got some comments from Intel guy. Now I am preparing v2 patchset. But the design and behavior would be mostly unchanged.

Links to my (v1) patchset in the DPDK patchwork site (1/4)

- [01/25] eal/pci: Add a new flag indicating a driver can detach devices at runtime.
  - http://dpdk.org/dev/patchwork/patch/1364/
- [02/25] ethdev: Remove assumption that port will not be detached
  - http://dpdk.org/dev/patchwork/patch/1365/
- [03/25] eal/pci: Replace pci address comparison code by eal_compare_pci_addr
  - http://dpdk.org/dev/patchwork/patch/1366/
- [04/25] ethdev: Add rte_eth_dev_free to free specified device
  - http://dpdk.org/dev/patchwork/patch/1367/
- [05/25] eal, ethdev: Add function pointer for closing a device
  - http://dpdk.org/dev/patchwork/patch/1368/
- [06/25] ethdev: Add rte_eth_dev_shutdown for closing PCI devices.
  - http://dpdk.org/dev/patchwork/patch/1369/
- [07/25] ethdev: Add functions to know which port is attached or detached
  - http://dpdk.org/dev/patchwork/patch/1370/

Links to my (v1) patchset in the DPDK patchwork site (2/4)

- [08/25] ethdev: Add rte_eth_dev_get_addr_by_port
  - http://dpdk.org/dev/patchwork/patch/1371/
- [09/25] ethdev: Add rte_eth_dev_get_port_by_addr
  - http://dpdk.org/dev/patchwork/patch/1372/
- [10/25] ethdev: Add rte_eth_dev_get_name_by_port
  - http://dpdk.org/dev/patchwork/patch/1373/
- [11/25] ethdev: Add rte_eth_dev_check_detachable
  - http://dpdk.org/dev/patchwork/patch/1374/
- [12/25] ethdev: Change scope of rte_eth_dev_allocated to global
  - http://dpdk.org/dev/patchwork/patch/1375/
- [13/25] eal/pci: Prevent double registration for devargs_list
  - http://dpdk.org/dev/patchwork/patch/1376/
- [14/25] eal/pci: Add rte_eal_devargs_remove
  - http://dpdk.org/dev/patchwork/patch/1377/

Links to my (v1) patchset in the DPDK patchwork site (3/4)

- [15/25] eal/pci: Add probe and close function for virtual drivers
  - http://dpdk.org/dev/patchwork/patch/1378/
- [16/25] eal/pci: Add port hotplug functions for virtual devices
  - http://dpdk.org/dev/patchwork/patch/1379/
- [17/25] eal/linux/pci: Add functions for unmapping igb_uio resources
  - http://dpdk.org/dev/patchwork/patch/1380/
- [18/25] eal/pci: Prevent double registrations for pci_device_list
  - http://dpdk.org/dev/patchwork/patch/1381/
- [19/25] eal/pci: Change scope of rte_eal_pci_scan to global
  - http://dpdk.org/dev/patchwork/patch/1382/
- [20/25] eal/pci: Add rte_eal_pci_close_one_driver
  - http://dpdk.org/dev/patchwork/patch/1383/
- [21/25] eal/pci: Fix pci_probe_all_drivers to share code with closing function
  - http://dpdk.org/dev/patchwork/patch/1384/

Links to my (v1) patchset in the DPDK patchwork site (4/4)

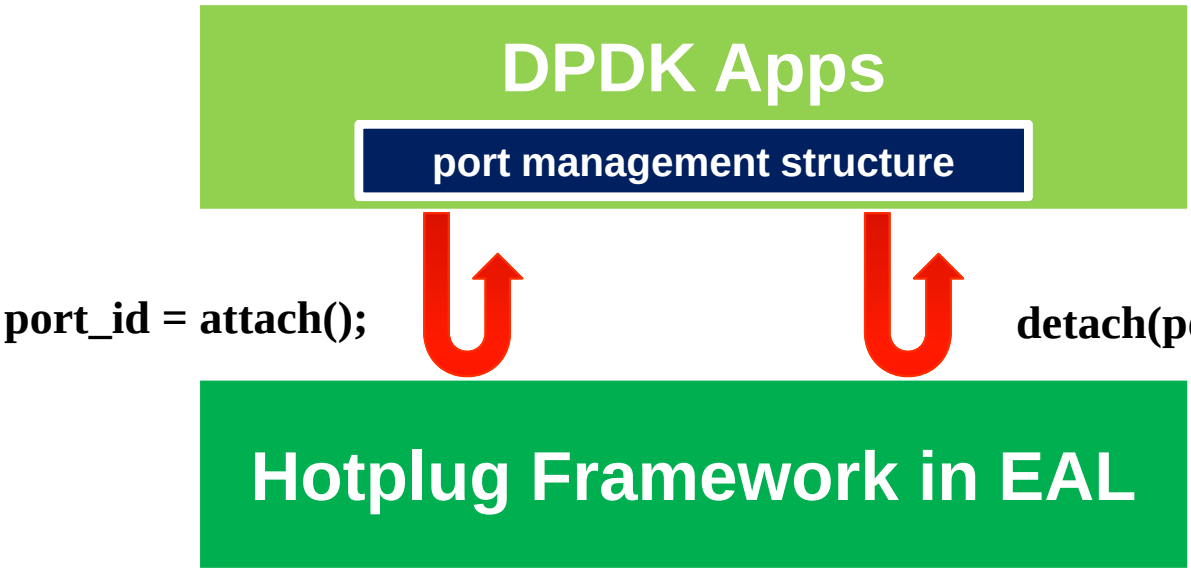
- [22/25] eal/pci: Add pci_close_all_drivers
  - http://dpdk.org/dev/patchwork/patch/1385/
- [23/25] eal/pci: Add rte_eal_pci_probe_one and rte_eal_pci_close_one
  - http://dpdk.org/dev/patchwork/patch/1386/
- [24/25] eal/pci: Add port hotplug functions for physical devices
  - http://dpdk.org/dev/patchwork/patch/1387/
- [25/25] eal: Enable port hotplug framework in Linux
  - http://dpdk.org/dev/patchwork/patch/1388/
- librte_pmd_pcap: Add port hotplug support
  - http://dpdk.org/dev/patchwork/patch/1389/
- testpmd: Add port hotplug support
  - http://dpdk.org/dev/patchwork/patch/1390/

Appendix

Basic concept

~ DPDK apps must have responsibility to manage ports ~

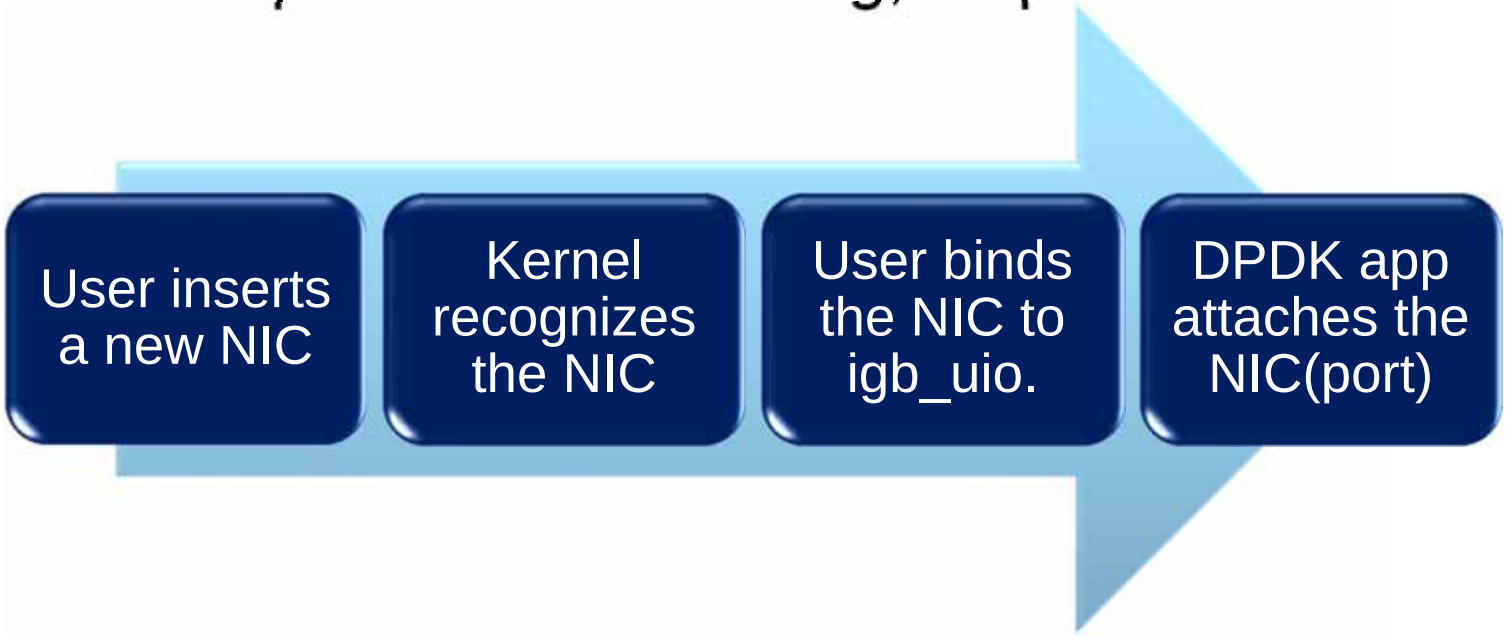
- DPDK apps only know which ports are attached or detached at the moment.
- The port hotplug framework is implemented to allow DPDK apps to manage ports. For example, when DPDK apps call port attach function, attached port number will be returned. Also DPDK apps can detach a port by specifying a port id.



Basic concept

~ Kernel support is needed for plugging physical device ports ~

– To attach new device, the device will be recognized by kernel at first and controlled by kernel driver. Then user can bind the device to igb_uio by 'dpdk_nic_bind.py'. Finally, DPDK apps can call the port hotplug functions to attach ports. For detaching, steps are vice versa.

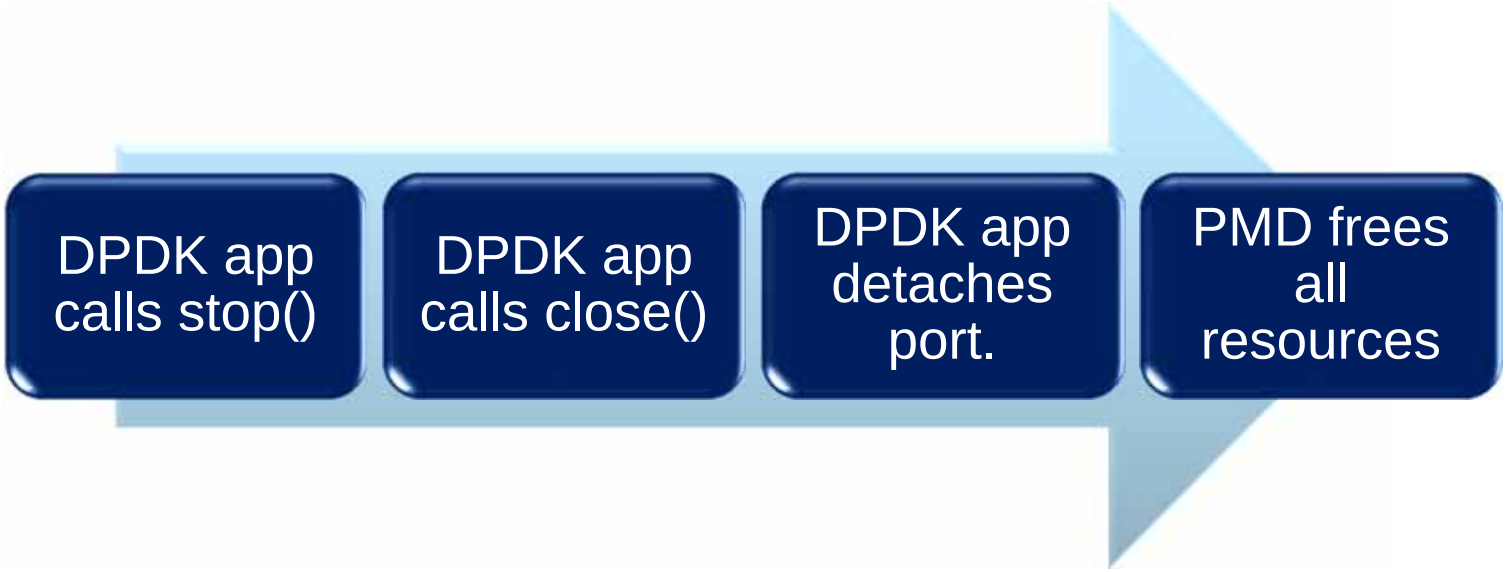


Basic concept

~ Before detach ports, ports must be stopped and closed ~

igel

- DPDK application must call rte_eth_dev_stop() and rte_eth_dev_close() before detaching ports.
- These function will call finalization codes of PMDs. But so far, no PMD frees all resources allocated by initialization. It means PMDs are needed to be fixed to support the port hotplug. 'RTE_PCI_DRV_DETACHABLE' is a new flag indicating a PMD supports detaching. Without this flag, detaching will be failed.



Basic concept

~ This framework mustn't affect legacy DPDK apps ~

- If the port hotplug functions aren't called, no EAL behavior is changed.
- So all legacy DPDK apps can still work without modifications.

Limitations

- The port hotplug functions are not thread safe. DPDK apps should handle it.
- Only support Linux and igb_uio so far.
  - BSD, pci_uio_generic and VFIO is not supported.

Let’s see how it works

~ Attaching a virtual device port ~

Termnal1

```
$ sudo testpmd -c f -n 1  -d librte_pmd_null.so -- -i --no-flush-rx
.....
testpmd> show port info all
testpmd>
testpmd> port attach v eth_null0
Attaching a new port...
PMD: Initializing pmd_null for eth_null0
PMD: Configure pmd_null: packet size is 64, packet copy is disabled
PMD: Creating null ethdev on numa socket 0
Port 0 is attached. Now total ports is 1
Done
testpmd> show port info all

********************* Infos for port 0  *********************
MAC address: 00:00:00:00:00:00
Connect to socket: 0
memory allocation on the socket: 0
Link status: down
Link speed: 10000 Mbps
Link duplex: full-duplex
Promiscuous mode: disabled
Allmulticast mode: disabled
Maximum number of MAC addresses: 1
Maximum number of MAC addresses of hash filtering: 0
VLAN offload:
  strip off
  filter off
  qinq(extend) off
```

No port here

Attach Port

We got Port0

- Execute testpmd.
- Attach a virtual port.

- Null PMD
  - http://dpdk.org/dev/patchwork/patch/686/
  - I will show how it’s useful for testing and developping.To merge the PMD, it must be reviewed. Someone, could you please do it?
  - When you use Null PMD with testpmd, --no-flush-rx option is needed.
  - To work with hotplug framework, PMD should be fixed. The patch for Null PMD has not been submitted yet.

Let’s see how it works

~ Attaching a physical device port ~

Termnal2

```
$ sudo ./tools/dpdk_nic_bind.py --status

Network devices using DPDK-compatible driver
============================================
<none>

Network devices using kernel driver
===================================
0000:02:00.0 '82572EI Gigabit Ethernet Controller (Copper)' if=p4p1
drv=e1000e unused=igb_uio

Other network devices
=====================
<none>
$ sudo ./tools/dpdk_nic_bind.py -b igb_uio 0000:02:00.0
$ sudo ./tools/dpdk_nic_bind.py --status

Network devices using DPDK-compatible driver
============================================
0000:02:00.0 '82572EI Gigabit Ethernet Controller (Copper)' drv=igb_uio
unused=

Network devices using kernel driver
===================================
<none>

Other network devices
=====================
<none>
```

No port here

Attach Port

We got port

- Open “Terminal2”.
- Switch to igb_uio.

Let’s see how it works

~ Attaching a physical device port ~ (contd.)

Termnal1

```
testpmd> port attach p 0000:02:00.0
Attaching a new port...
EAL: PCI device 0000:02:00.0 on NUMA socket -1
EAL:   probe driver: 8086:10b9 rte_em_pmd
EAL:   PCI memory mapped at 0x7ffff7f81000
EAL:   PCI memory mapped at 0x7ffff7f61000
PMD: eth_em_dev_init(): port_id 1 vendorID=0x8086 deviceID=0x10b9
Port 1 is attached. Now total ports is 2
Done
testpmd> show port info all

********************* Infos for port 0  *********************
.....snip.....
********************* Infos for port 1  *********************
MAC address: 00:1B:21:2D:E5:53
Connect to socket: 0
memory allocation on the socket: 0
Link status: up
Link speed: 1000 Mbps
Link duplex: full-duplex
Promiscuous mode: enabled
Allmulticast mode: disabled
Maximum number of MAC addresses: 15
Maximum number of MAC addresses of hash filtering: 0
VLAN offload:
  strip off
  filter off
  qinq(extend) off
```

- Open “Terminal1”.
- Attach a physical port

Let’s see how it works

~ Packet forwarding ~

Termnal1

```
testpmd> port start all
testpmd> start
testpmd> stop
Telling cores to stop...
Waiting for lcores to finish...

  ---------------------- Forward statistics for port 0  ----------------------
  RX-packets: 92453792       RX-dropped: 0             RX-total: 92453792
  TX-packets: 0              TX-dropped: 0             TX-total: 0
  ----------------------------------------------------------------------------

  ---------------------- Forward statistics for port 1  ----------------------
  RX-packets: 0              RX-dropped: 0             RX-total: 0
  TX-packets: 2714270        TX-dropped: 89738977      TX-total: 92453247
  ----------------------------------------------------------------------------

  ++++++++++++++++ Accumulated forward statistics for all
ports++++++++++++++++
  RX-packets: 92453792       RX-dropped: 0             RX-total: 92453792
  TX-packets: 2714270        TX-dropped: 89738977      TX-total: 92453247

++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
++++++++++++++++

Done.
```

- Open “Terminal1”.
- Start all ports.
- Start packet forwarding.
- Stop packet forwarding.

Let’s see how it works

~ Detach ports ~

Termnal1

```
testpmd> port stop all
Stopping ports...
Checking link statuses...
Port 0 Link Down
Port 1 Link Down
Done
testpmd> port close all
Closing ports...
Done
testpmd> port detach p 1
Detaching a port...
EAL: PCI device 0000:02:00.0 on NUMA socket -1
EAL:   remove driver: 8086:10b9 rte_em_pmd
EAL:   PCI memory mapped at 0x7ffff7f81000
EAL:   PCI memory mapped at 0x7ffff7f61000
Port '0000.02.00.0' is detached. Now total ports is 1
Done
testpmd> port detach v 0
Detaching a port...
PMD: Closing null ethdev on numa socket 0
Port 'eth_null0' is detached. Now total ports is 0
Done
testpmd> show port info all
testpmd>
```

Detach Port1

Detach Port0

No port here

- Open “Terminal1”.
- Stop all ports
- Close all ports
- Detach ports
  - You can detach ports in any order.
- After detaching a physical port, you can switch to kernel device driver again using “dpdk_nic_bind.py”

Let’s see how it works

~ Example ~

Termnal1

```
testpmd> port attach v eth_null0
testpmd> port attach v eth_null1
testpmd> port attach v eth_null2
testpmd> port attach v eth_null3
testpmd> port stop 0
testpmd> port stop 2
testpmd> port close 0
testpmd> port close 2
testpmd> port detach v 0
testpmd> port detach v 2
testpmd> port start all
testpmd> start
testpmd> stop
 ---------------------- Forward statistics for port 1  ----------------------
  RX-packets: 31774496       RX-dropped: 0          RX-total: 31774496
  TX-packets: 31774496       TX-dropped: 0          TX-total: 31774496
  ----------------------------------------------------------------------------
  ---------------------- Forward statistics for port 3  ----------------------
  RX-packets: 31774496       RX-dropped: 0          RX-total: 31774496
  TX-packets: 31774496       TX-dropped: 0          TX-total: 31774496
  ----------------------------------------------------------------------------
  ++++++++++++++++ Accumulated forward statistics for all ports++++++++++++++++
  RX-packets: 63548992       RX-dropped: 0          RX-total: 63548992
  TX-packets: 63548992       TX-dropped: 0          TX-total: 63548992
  ++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
Done.
testpmd>
```

Attach 4 ports

Detach Ports0 and Port3

Packet forwarding between port1 and port3

- It’s very flexible. We can do like that.