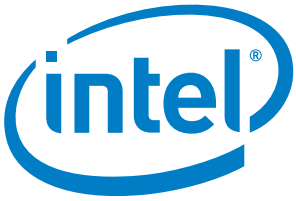


DPDK Vhost/Virtio Performance Report Release 21.08

Test Date: November 3, 2021

Author: Intel DPDK Validation Team

Revision History

Date	Revision	Comment
November 3 2021	1.0	Initial document for release

Contents

Audience and Purpose

The primary audience for this test report is architects and engineers implementing the Data Plane Development Kit (DPDK). This report provides information on packet processing performance testing for the specified DPDK release on Intel® architecture. The initial report may be viewed as the baseline for future releases and provides system configuration and test cases based on DPDK examples.

The purpose of reporting these tests is not to imply a single "correct" approach, but rather to provide a baseline of well-tested configurations and procedures with reproducible results. This will help guide architects and engineers who are evaluating and implementing DPDK solutions on Intel® architecture and can assist in achieving optimal system performance.

Test setup:

The device under test (DUT) consists of a system with an Intel® architecture motherboard populated with the following;

- A single or dual processor and PCH chip, except for System on Chip (SoC) cases
- DRAM memory size and frequency (normally single DIMM per channel)
- Specific Intel Network Interface Cards (NICs)
- BIOS settings noting those that updated from the basic settings
- DPDK build configuration settings, and commands used for tests

Connected to the DUT is an IXIA*, a hardware test and simulation platform to generate packet traffic to the DUT ports and determine the throughput at the tester side. The IXIA is used to implement RFC2544 on the DUT.

Benchmarking a DPDK system requires knowledge of networking technologies including knowledge of network protocols and hands-on experience with relevant open-source software, such as Linux*, and the DPDK. Engineers also need benchmarking and debugging skills, as well as a good understanding of the device-under-test (DUT) across compute and networking domains.

DPDK Testpmd Test Case: Documentation may be found at http://www.dpdk.org/doc/guides/testpmd_app_ug/index.html.

The testpmd application can be used to test the DPDK in a packet forwarding mode and also to access NIC hardware features. Note in the Testpmd example if the –i argument is used, the first core is used for the command language interface (CLI).

RFC2544 Zero packet loss test case: Used to determine the DUT throughput as defined in RFC1242(https://www.ietf.org/rfc/rfc1242.txt). Note RFC6201 https://www.ietf.org/rfc/rfc6201.txt has updated RFC2544 and RFC1242.Please check the link for more details. In this report, RFC2544 test uses DPDK testpmd as test application.

Procedure: Send a specific number of frames at a specific rate through the DUT and then count the frames that are transmitted by the DUT. If the count of offered frames is not equal to the count of

received frames, the rate of the offered stream is reduced and the test is rerun. The throughput is the fastest rate at which the count of test frames transmitted by the DUT is equal to the number of test frames sent to it by the test equipment.

DPDK Phy-VM-Phy(PVP) RFC2544 test case:
This test setup is shown in Figure1. The traffic is generated by Ixia running RFC2544(IxNetwork* 8.12 with 0.001% packet loss, and the duration for each round is 60 seconds). The flow is one fixed flow. In this test setup, one port(100G) of Intel ® Ethernet Converged Network Adapter E810-QDA2 is used to inject traffic to Vhost/virtio. The case is to measure vhost/virtio system forwarding throughput, and the theoretical system forwarding throughput is 100 Gbps. Both Vhost and Virtio is DPDK polling mode driver.
The flow is as below: IXIA→NIC port0→Vhost-user0→Virtio→Vhost-user0→NIC port0→IXIA.

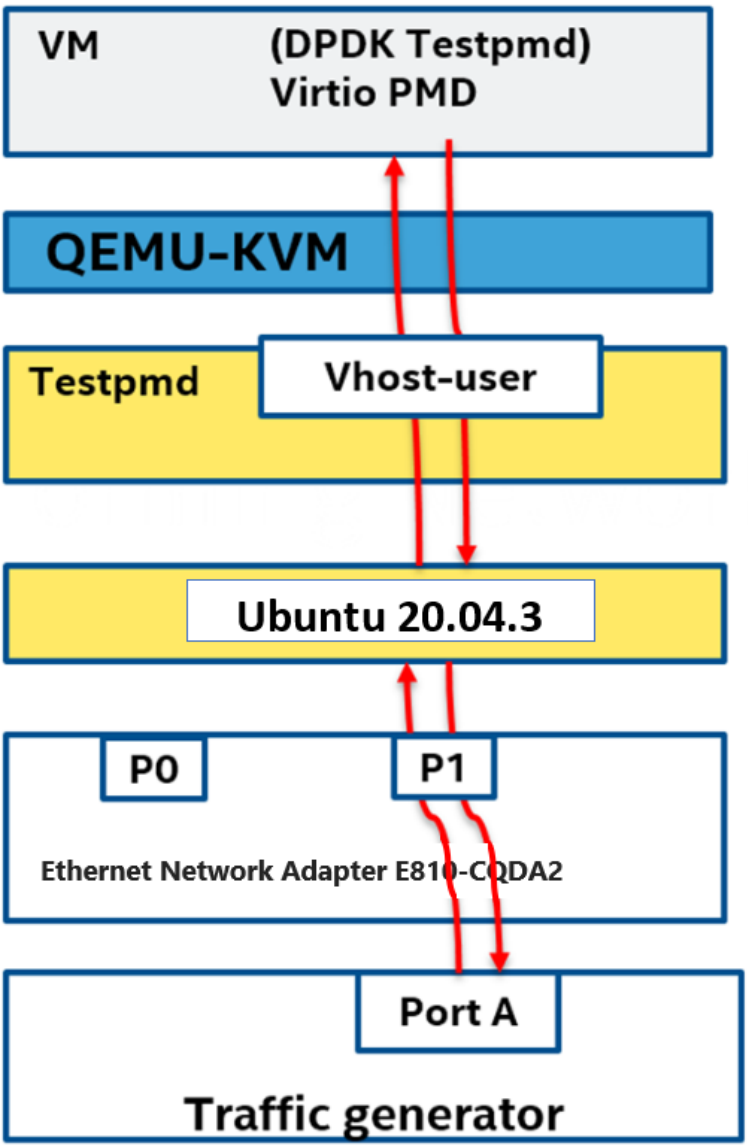


Figure1. DPDK PVP test setup

Intel® Xeon® Processor Platinum 8380 (60M Cache, 2.30 GHz)

Hardware & Software Ingredients

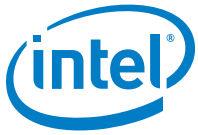
Item	Description
Server Platform	Inspur® Server Board NF5280M6
CPU	Intel® Xeon® Platinum 8380 CPU @ 2.30GHz 40 CPU cores * 2 NUMA nodes
Memory	256GB: 16GB x 8 DIMMs x 2 NUMA nodes @ 3200MHz
PCIe	PCIe 3.0/4.0 x16
NICs	1x Intel® Ethernet Network Adapter E810-CQDA2
BIOS	04.12.02
Microcode version	0xd0002a0
Host Operating System	Ubuntu 20.04.3 LTS QEMU emulator version 4.2.1 (Debian 1:4.2-3ubuntu6.18)
Host Linux kernel version	5.11.0-38-generic
Host GCC version	gcc (Ubuntu 9.3.0-17ubuntu1~20.04) 9.3.0
Host DPDK version	21.08
Guest Operating System	Ubuntu 20.04 LTS
Guest GCC version	gcc (Ubuntu 9.3.0-10ubuntu2) 9.3.0
Guest DPDK version	21.08
Guest Linux kernel version	5.11.0-38-generic

Boot and BIOS settings

Item	Description
Host Boot Settings	`hugepagesz=1G hugepages=120 default_hugepagesz=1G isolcpus=1-15 intel_iommu=on iommu=pt intel_pstate=disable numa_balancing=disable nmi_watchdog=0 mitigations=off` **Note: nohz_full and rcu_nocbs is to disable Linux* kernel interrupts, and it's important for zero-packet loss test. Generally, 1G huge pages are used for performance test.**
VM Boot Settings	`hugepagesz=2M hugepages=512 isolcpus=1-2 nohz_full=1-2 rcu_nocbs=1-2`
BIOS	CPU Power and Performance Policy <Performance> CPU C-state Disabled CPU P-state Disabled Enhanced Intel® Speedstep® Tech Disabled Turbo Boost Disabled

Host Real Time Settings	`echo -1 > /proc/sys/kernel/sched_rt_period_us`
	`echo -1 > /proc/sys/kernel/sched_rt_runtime_us`
	`echo 10 > /proc/sys/vm/stat_interval`
	`echo 0 > /proc/sys/kernel/watchdog_thresh`

VM Real Time Settings	`echo 0 > /proc/sys/kernel/watchdog`
	`echo 0 > /proc/sys/kernel/nmi_watchdog`
	`echo -1 > /proc/sys/kernel/sched_rt_period_us`
	`echo -1 > /proc/sys/kernel/sched_rt_runtime_us`



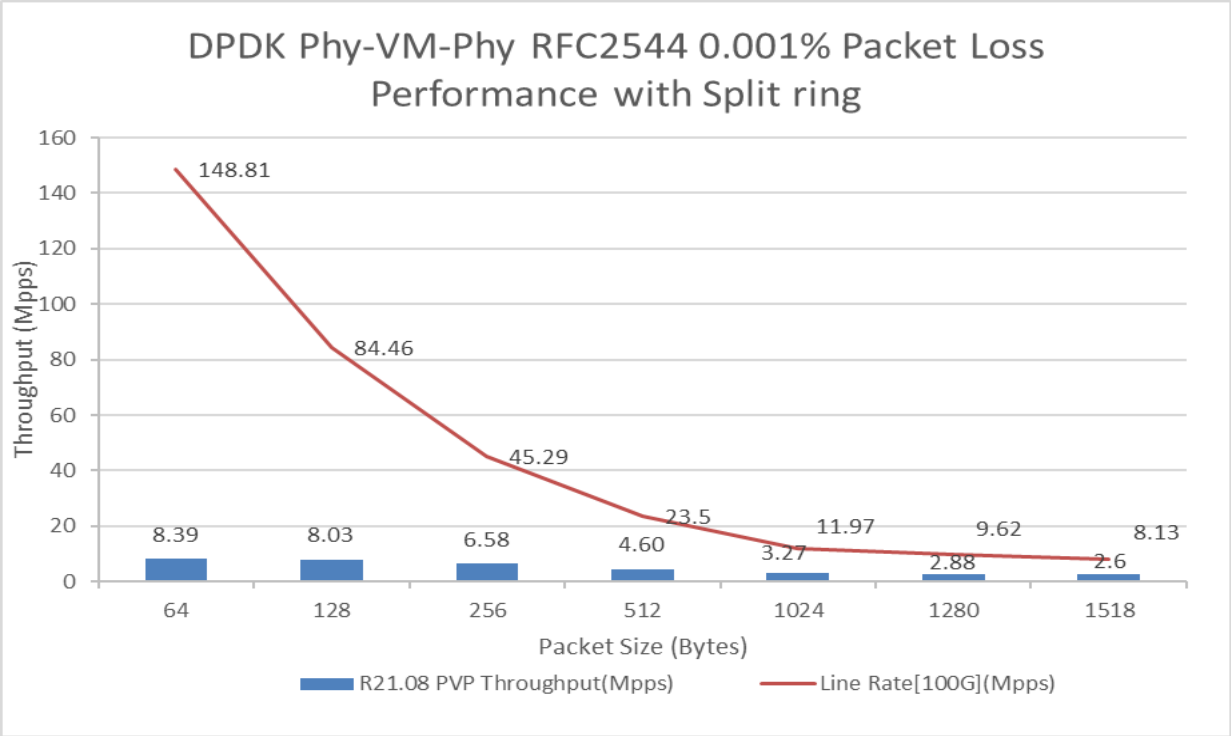
Test Case 1 – RFC2544 0.001% packet loss test for Split ring Vhost/Virtio PVP Mergeable

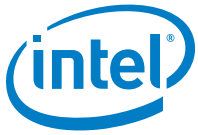
Item	Description
Test Case	RFC2544 0.001% packet loss test for Split ring Vhost/Virtio PVP Mergeable
NIC	1x Intel® Ethernet Network Adapter E810-CQDA2(2x100G)
Driver	ice DPDK PMD
Test Configuration	Test tool: IxNetwork 9.00.1915.16 **Qemu Version: 4.2.1** **Vring size: 1024,** the max Vring size Qemu support Hugepage size: 1G Virtio Mergeable: On Forward Mode: testpmd mac forward **Vhost: 1 queue 1 logic core** **Virtio: 1 queue 1 logic core** **Totally 2 logic cores from 2 physical cores are used.**
Flow Configuration	1 Flow with fixed source and destination IP.
Test Step	1. Bind one 100G NIC port to vfio-pci 2. Launch vhost: chrt -f 95 ./x86_64-native-linuxapp-gcc/app/testpmd -l 9-10 -n 4 -v --file-prefix=vhost\ --vdev 'eth_vhost0,iface=vhost-net,queues=1' -- -i --txd=1024 --rxd=1024 --nb-cores=1 testpmd>set fwd mac testpmd>start 3. Launch VM : taskset -c 11,12,13 qemu-system-x86_64 \ -name us-vhost-vm1 -cpu host -enable-kvm -m 2048 \ -object memory-backend-file,id=mem,size=2048M,mem-path=/mnt/huge,share=on \ -numa node,memdev=mem -mem-prealloc \ -smp cores=2,sockets=1 -drive file=/home/osimg/ubuntu20-04.img \ -chardev socket,id=char0,path=./vhost-net \ -monitor unix:/tmp/vm2_monitor.sock,server,nowait -device e1000,netdev=ntts1 -netdev user,id=ntts1,hostfwd=tcp:127.0.0.1:6013-:22 \ -netdev type=vhost-user,id=mynet1,chardev=char0,vhostforce \ -device virtio-net-pci,mac=52:54:00:00:00:01,netdev=mynet1,mrg_rxbuf=on,rx_queue_size=1024,tx_queue_size=1024 -vnc :10 --monitor stdio 4. Use monitor to bind vcpu with physical cpu on host machine: qemu monitor: info cpus #check pid taskset -cp 12 xxx #xxx is the pid number taskset -cp 13 xxx

	5. Enter vm and bind port to vfio-pci: modprobe vfio-pci echo 1 > /sys/module/vfio/parameters/enable_unsafe_noiommu_mode 6. Bind vdev to vfio-pci and launch virtio in VM: ./x86_64-native-linuxapp-gcc/app/testpmd -c 0x3 -n 4 -- -i --txd=1024 --rxd=1024 testpmd>set fwd mac testpmd>start

Test Result:

Packet Size (Bytes)	Throughput (Mpps)	Line Rate[100G] (Mpps)
64	8.39	148.81
128	8.03	84.46
256	6.58	45.29
512	4.60	23.50
1024	3.27	11.97
1280	2.88	9.62
1518	2.60	8.13





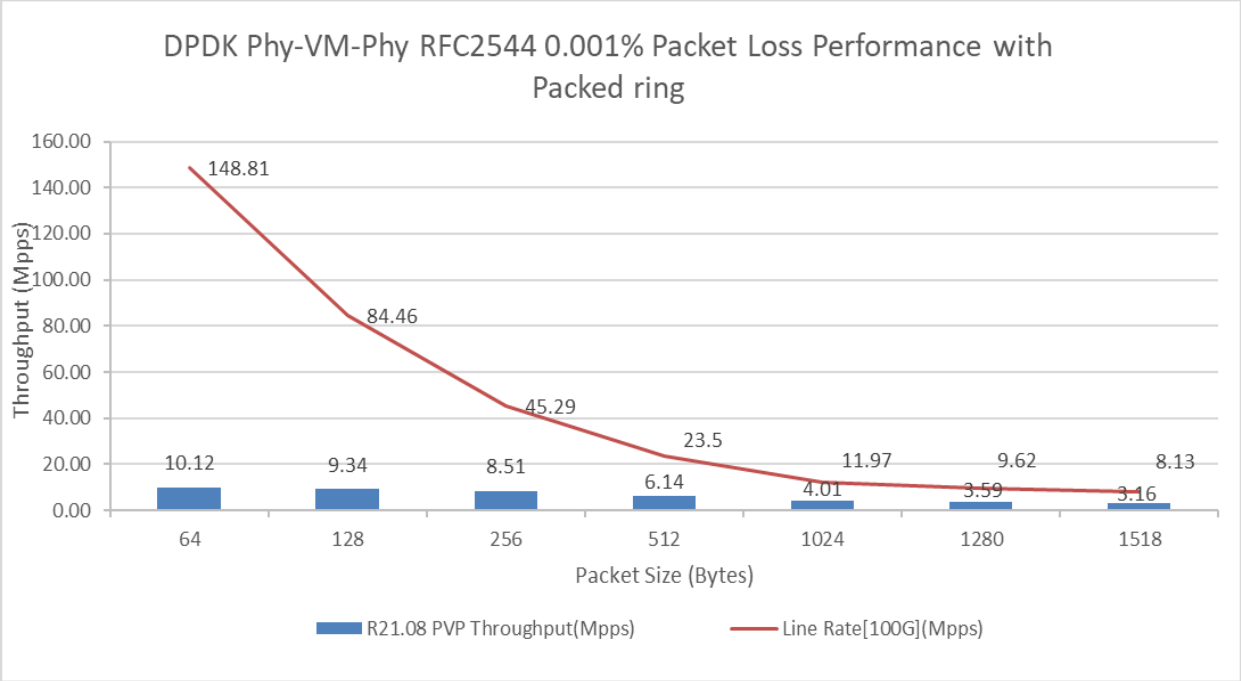
Test Case 2 – RFC2544 0.001% packet loss test for Packed ring Vhost/Virtio PVP Mergeable

Item	Description
Test Case	RFC2544 0.001% packet loss test for Packed ring Vhost/Virtio PVP Mergeable
NIC	1x Intel® Ethernet Network Adapter E810-CQDA2(2x100G)
Driver	ice DPDK PMD
Test Configuration	Test tool: IxNetwork 9.00.1915.16 **Qemu Version: qemu-4.2.1** **Vring size: 1024,** the max Vring size Qemu support Hugepage size: 1G Virtio Mergeable: On Forward Mode: testpmd mac forward **Vhost: 1 queue 1 logic core** **Virtio: 1 queue 1 logic core** **Totally 2 logic cores from 2 physical cores are used.**
Flow Configuration	1 Flow with fixed source and destination IP.
Test Step	1. Bind one 100G NIC port to vfio-pci 2. Launch vhost: chrt -f 95 ./x86_64-native-linuxapp-gcc/app/testpmd -l 9-10 -n 4 -v --file-prefix=vhost\ --vdev 'eth_vhost0,iface=vhost-net,queues=1' -- -i --txd=1024 --rxd=1024 --nb-cores=1 testpmd>set fwd mac testpmd>start 3. Launch VM : taskset -c 11,12,13 qemu-system-x86_64 \ -name us-vhost-vm1 -cpu host -enable-kvm -m 2048 \ -object memory-backend-file,id=mem,size=2048M,mem-path=/mnt/huge,share=on \ -numa node,memdev=mem -mem-prealloc \ -smp cores=2,sockets=1 -drive file=/home/osimg/ubuntu20-04.img \ -chardev socket,id=char0,path=./vhost-net \ -monitor unix:/tmp/vm2_monitor.sock,server,nowait -device e1000,netdev=nttsip1 -netdev user,id=nttsip1,hostfwd=tcp:127.0.0.1:6013-:22 \ -netdev type=vhost-user,id=mynet1,chardev=char0,vhostforce \ -device virtio-net-pci,mac=52:54:00:00:00:01,netdev=mynet1,mrg_rxbuf=on,rx_queue_size=1024,tx_queue_size=1024,packed=on -vnc :10 --monitor stdio 4. Use monitor to bind vcpu with physical cpu on host machine: qemu monitor: info cpus #check pid taskset -cp 12 xxx #xxx is the pid number

	taskset -cp 13 xxx 5. Enter vm and bind port to vfio-pci: modprobe vfio-pci echo 1 > /sys/module/vfio/parameters/enable_unsafe_noiommu_mode 6. Bind vdev to igb_uio and launch virtio in VM: ./x86_64-native-linuxapp-gcc/app/testpmd -c 0x3 -n 4 -- -i --txd=1024 --rxd=1024 testpmd>set fwd mac testpmd>start

Test Result:

Packet Size (Bytes)	Throughput (Mpps)	Line Rate[100G] (Mpps)
64	10.12	148.81
128	9.34	84.46
256	8.51	45.29
512	6.14	23.50
1024	4.01	11.97
1280	3.59	9.62
1518	3.16	8.13



DISCLAIMERS