Rockchip Linux DPDK Development Guide

ID: RK-YH-YF-987

Release Version: V1.0.0

Release Date: 2023-03-25

Security Level: □Top-Secret □Secret □Internal ■Public

Rockchip Electronics Co., Ltd.

No.18 Building, A District, No.89, software Boulevard Fuzhou, Fujian,PRC

Website: www.rock-chips.com

Customer service Tel: +86-4007-700-590

Customer service Fax: +86-591-83951833

Customer service e-Mail: fae@rock-chips.com

Preface

Overview

Rockchip Linux Platform DPDK Development Guide.

Product Version

Chip Name	Kernel Version
RK3588/RK3568/RK3566	Linux 4.19/5.10

Intended Audience

This document (this guide) is mainly intended for:

Technical Support Engineers

Software Development Engineers

Hardware Development Engineers

Revision History

Version Number	Author	Date	Change Description
V0.1.0	xy	2022-12	Initial version
V0.2.0	xy	2023-01	Added I320/350 PCIe network card forwarding data
V1.0.0	xy	2023-03	Added GMAC development guid

Contents

1. DPDK Overview

DPDK stands for Intel Data Plane Development Kit, a development toolkit provided by Intel. It currently supports efficient packet processing in user space for processor architectures such as ARM, X86, and PowerPC, offering library functions and driver support. Unlike the Linux system designed for generality, DPDK focuses on high-performance packet handling in network applications. It has been verified to run on most Linux operating systems; DPDK uses the BSD License, which greatly facilitates enterprises to implement their own protocol stacks or applications on this basis. This helps to deploy high-performance network applications more quickly and achieve more efficient computation than the standard interrupt-based kernel network stack.

2. DPDK Working Principle

DPDK utilizes a variety of techniques to optimize packet throughput, but its mode of operation (and the key to its performance) is based on kernel bypass and polling mode driver (PMD).

- **Kernel Bypass**: A path from the NIC to the application is created within user space, in other words, bypassing the kernel. This eliminates context switching when moving network frames between user space and kernel space. Furthermore, additional benefits can be gained by eliminating the NIC driver, kernel network stack, and the performance loss they introduce.
- **Polling Mode Driver (PMD)**: When the CPU receives a frame, instead of triggering an interrupt from the NIC, the CPU continuously runs the PMD to poll the NIC for new packets.

In addition, DPDK also uses other methods, such as **CPU affinity, CPU isolation, huge page memory, cache line alignment, and bulk operations**, to achieve the best possible performance.

3. Platform Support Status

Rockchip SOC platform supports network cards with GMAC/PCIe interfaces, with GMAC supporting up to two gigabit network cards; PCIe supports x1/x2/x4 types of network cards. The RK3588 supports up to six external network cards: 4 PCIe + 2 GMAC. The detailed resources for each chip are as follows:

3.1 RK3566

Resource	Mode	SOC Interconnect Support	Lane Splitting Support	DMA Support	MMU Support
PCIe Gen2 x 1	RC only	No	No	No	No
GMAC 1000M (**x2**)	RGMII	/	/	/	Yes

3.2 RK3568

Resources	Mode	SOC Interconnect Support	Lane Splitting Support	DMA Support	MMU Support
PCIe Gen2 x 1 lane	RC only	No	No	No	No
PCIe Gen3 x 2 lane	RC/EP	Yes	1 lane RC + 1 lane RC	Yes	No
GMAC 1000M (**x2**)	RGMII	/	/	/	Yes

3.3 RK3588

Resource	Mode	Support Lane Split	Available PHY	Internal DMA
PCIe Gen3 x 4 lane	RC/EP	Yes	pcie30phy	Yes
PCIe Gen2 x 1 lane (**x3**)	RC only	No	pcie30phy, combphy1_ps	No
GMAC 1000M (x2)	RGMII	/	/	Yes

3.4 RK3588S

Resource	Mode	Support Lane Split	Available PHY	Internal DMA
PCIe Gen3 x 1 lane	RC only	No	combphy2_psu	No
PCIe Gen2 x 1 lane	RC only	No	combphy0_ps	No
GMAC 1000M **(x2)**	RGMII	/	/	Yes

4. DPDK Kernel Configuration

Note:

- **GMAC Solution**: The latest SDK supports it by default. For older SDKs, patches are required. Please refer to the external/dpdk/gmac/README.txt in the SDK and apply the corresponding patches based on the kernel version.
- **PCIE Solution**: the external/dpdk/pcie/README.txt in the SDK .

4.1 DTS

- The GMAC solution first enables the UIO node in DTS, with RK3568-evb1 as a reference:

```
diff --git a/arch/arm64/boot/dts/rockchip/rk3568-evb1-ddr4-v10.dtsi
b/arch/arm64/boot/dts/rockchip/rk3568-evb1-ddr4-v10.dtsi
index 0cb57e9d8529..c7729258e51d 100644
--- a/arch/arm64/boot/dts/rockchip/rk3568-evb1-ddr4-v10.dtsi
+++ b/arch/arm64/boot/dts/rockchip/rk3568-evb1-ddr4-v10.dtsi
@@ -262,6 +262,14 @@
        status = "okay";
 };

+&gmac_uio0 {
+       status = "okay";
+};
+
+&gmac_uio1 {
+       status = "okay";
+};
+
 /*
  * The power-supply should switch to vcc3v3_lcd1_n
  * when the MIPI panel is connected to DSI1.
```

4.2 Kernel Configuration

The following configurations are enabled in defconfig:

```
CONFIG_UIO=m
CONFIG_STMMAC_UIO=m
CONFIG_HUGETLBFS=y
```

4.3 Compile Kernel Object (KO)

```
# Modify according to the actual situation for different platforms
make CROSS_COMPILE=aarch64-linux-gnu- ARCH=arm64 rockchip_linux_defconfig
# Open the configuration for section 4.2
make CROSS_COMPILE=aarch64-linux-gnu- ARCH=arm64 menuconfig
make CROSS_COMPILE=aarch64-linux-gnu- ARCH=arm64 rk3568-evb1-ddr4-v10-linux.img -
j8

# Flash boot.img and push related ko
adb push igb_uio.ko  #Refer to the SDK's external/dpdk/pcie/README.txt
adb push drivers/uio/uio.ko
adb push drivers/net/ethernet/stmicro/stmmac/stmmac_uio.ko
```

4.4 DPDK Compilation

DPDK testing is conducted on a development board running Debian 11, with DPDK version 21.11. For compilation, refer to http://doc.dpdk.org/guides-21.11/linux_gsg/cross_build_dpdk_for_arm64.html

- Cross-compilation on PC

```
wget https://developer.arm.com/-/media/Files/downloads/gnu-a/9.2-
2019.12/binrel/gcc-arm-9.2-2019.12-x86_64-aarch64-none-linux-gnu.tar.xz
tar -xvf gcc-arm-9.2-2019.12-x86_64-aarch64-none-linux-gnu.tar.xz

export PATH=$PATH:<cross_install_dir>/gcc-arm-9.2-2019.12-x86_64-aarch64-none-
linux-gnu/bin

apt install build-essential
apt install pkg-config-aarch64-linux-gnu
apt-get install python3 python3-pip
apt install meson
apt install ninja-build

#Special attention: Modify the name of the compilation chain in the file below to
be consistent with the one downloaded and installed. If not, change it to be
consistent, otherwise it cannot be compiled!
config/arm/arm64_armv8_linux_gcc

meson aarch64-build-gcc --cross-file config/arm/arm64_armv8_linux_gcc
meson --reconfigure aarch64-build-gcc --cross-file
config/arm/arm64_armv8_linux_gcc -Dbuildtype=debug -Dplatform=arm64 -
Dexamples=l2fwd,l3fwd
ninja -C aarch64-build-gcc
```

- Compile DPDK on the development board (suitable for Debian systems)

```
apt-get install python3 python3-pip
apt install meson
apt install ninja-build
pip3 install meson ninja
apt-get install libdpkg-perl
apt-get install build-essential
apt-get install python3-pyelftools
apt install libnuma-dev
apt install libpcap-dev

meson setup -Dplatform=generic build
cd build
ninja
ninja install
```

5. Running DPDK Program

5.1 Mounting Huge Pages

```
echo 1024 > /sys/kernel/mm/hugepages/hugepages-2048kB/nr_hugepages
```

5.2 Load KO

```
# Load UIO modules
insmod uio.ko
insmod stmmac_uio.ko  # GMAC
insmod igb_uio.ko     # PCIe

# Load GMAC PMD network card driver
dpdk/usertools/dpdk-devbind.py -b igb_uio 0000:01:00.0
dpdk/usertools/dpdk-devbind.py -b igb_uio 0000:01:00.1

# Load PCIe PMD network card driver (replace 0000:01:00.X with actual values)
dpdk/usertools/dpdk-devbind.py -b igb_uio 0000:01:00.0
dpdk/usertools/dpdk-devbind.py -b igb_uio 0000:01:00.1
dpdk/usertools/dpdk-devbind.py -b igb_uio 0000:01:00.2
dpdk/usertools/dpdk-devbind.py -b igb_uio 0000:01:00.3
... ...

# Enable performance mode (ignore any error messages)
echo performance | tee $(find /sys/ -name *governor) /dev/null || true

# Start huge page support
echo 1024 > /sys/kernel/mm/hugepages/hugepages-2048kB/nr_hugepages

# Start testing, --txpkts=1500 can set the packet size
dpdk-testpmd -l 5,6,7,8 -n 4 -- -i --nb-cores=1 --forward-mode=flowgen --
txpkts=1500
```

By default, the network traffic after booting up still goes through the native kernel network. Here, we switch between DPDK and the native kernel network by using `insmod stmmac_uio.ko` and `rmmod` for the `.ko` file, which means after unloading the `.ko` file, it will revert to the original kernel network state.

- Enter DPDK network control:

```
// Taking GMAC as an example:
root@linaro-alip:/home/linaro# insmod stmmac_uio.ko
[   41.232761] Generic PHY stmmac-1:00: attached PHY driver [Generic PHY]
(mii_bus:phy_addr=stmmac-1:00, irq=POLL)
[   41.236447] dwmac4: Master AXI performs any burst length
[   41.236497] rk_gmac-dwmac fe010000.ethernet eth0: No Safety Features support
found
[   41.236886] rockchip_eth_uio_drv fe010000.uio: Registered uio_eth0 uio
devices, 3 register maps attached
[   41.241453] Generic PHY stmmac-0:00: attached PHY driver [Generic PHY]
(mii_bus:phy_addr=stmmac-0:00, irq=POLL)
[   41.257084] dwmac4: Master AXI performs any burst length
[   41.257138] rk_gmac-dwmac fe2a0000.ethernet eth1: No Safety Features support
found
[   41.257523] rockchip_eth_uio_drv fe2a0000.uio: Registered uio_eth1 uio
devices, 3 register maps attached
```

- Return to kernel network control:

```
// Taking GMAC as an example:
root@linaro-alip:/home/linaro# rmmod stmmac_uio
[ 2200.304636] Generic PHY stmmac-0:00: attached PHY driver [Generic PHY]
(mii_bus:phy_addr=stmmac-0:00, irq=POLL)
[ 2200.318163] dwmac4: Master AXI performs any burst length
[ 2200.318205] rk_gmac-dwmac fe2a0000.ethernet eth1: No Safety Features support
found
[ 2200.318227] rk_gmac-dwmac fe2a0000.ethernet eth1: IEEE 1588-2008 Advanced
Timestamp supported
[ 2200.318443] rk_gmac-dwmac fe2a0000.ethernet eth1: registered PTP clock
[ 2200.319414] IPv6: ADDRCONF(NETDEV_UP): eth1: link is not ready
[ 2200.322971] Generic PHY stmmac-1:00: attached PHY driver [Generic PHY]
(mii_bus:phy_addr=stmmac-1:00, irq=POLL)
[ 2200.324883] dwmac4: Master AXI performs any burst length
[ 2200.324921] rk_gmac-dwmac fe010000.ethernet eth0: No Safety Features support
found
[ 2200.324945] rk_gmac-dwmac fe010000.ethernet eth0: IEEE 1588-2008 Advanced
Timestamp supported
[ 2200.325468] rk_gmac-dwmac fe010000.ethernet eth0: registered PTP clock
[ 2200.325814] IPv6: ADDRCONF(NETDEV_UP): eth0: link is not ready
[ 2205.385406] IPv6: ADDRCONF(NETDEV_CHANGE): eth0: link becomes ready
```

5.3 Set Performance Mode

```
echo performance > /sys/devices/system/cpu/cpufreq/policy0/scaling_governor
echo performance > /sys/devices/system/cpu/cpufreq/policy4/scaling_governor
echo performance > /sys/devices/system/cpu/cpufreq/policy6/scaling_governor
echo performance > /sys/class/devfreq/dmc/governor
#or
echo performance | tee $(find /sys/ -name *governor) /dev/null || true
```

5.4 Running testpmd

- Forwarding Mode Test Command

> --vdev=net_stmmac0 --vdev=net_stmmac1 indicates the specified virtual devices, currently the names are fixed (**PCIe does not need to be specified**)
>
> --main-lcore=0 indicates that core 0 is used for management, and cores 2 and 3 are used for forwarding
>
> --iova-mode=pa because the device does not support iommu, the iova-mode is set to pa mode
>
> -- is used to separate eal parameters and testpmd parameters
>
> -i indicates entering the interactive mode of the dpdk-testpmd command

```
# Taking GMAC as an example:
root@linaro-alip:/home/linaro# ./dpdk-testpmd  --iova-mode=pa --vdev=net_stmmac0
--vdev=net_stmmac1 -l 0,2,3 --main-lcore=0 -- -i
EAL: Detected CPU lcores: 4
EAL: Detected NUMA nodes: 1
EAL: Detected static linkage of DPDK
EAL: Multi-process socket /var/run/dpdk/rte/mp_socket
EAL: Selected IOVA mode 'PA'
TELEMETRY: No legacy callbacks, legacy socket not created
Interactive-mode selected
Warning: NUMA should be configured manually by using --port-numa-config and --
ring-numa-config parameters along with --numa.
testpmd: create a new mbuf pool <mb_pool_0>: n=163456, size=2176, socket=0
testpmd: preferred mempool ops selected: ring_mp_mc
Configuring Port 0 (socket 0)
stmmac_net: stmmac_eth_link_update()Port (0) link is Up
Port 0: BA:A0:3F:FD:B2:8C
Configuring Port 1 (socket 0)
stmmac_net: stmmac_eth_link_update()Port (1) link is Up
Port 1: B6:A0:3F:FD:B2:8C
Checking link statuses...
stmmac_net: stmmac_eth_link_update()Port (0) link is Up
stmmac_net: stmmac_eth_link_update()Port (1) link is Up
Done
```

- Starting Forwarding Mode Test

```
testpmd> start
io packet forwarding - ports=2 - cores=1 - streams=2 - NUMA support enabled, MP
allocation mode: native
Logical Core 2 (socket 0) forwards packets on 2 streams:
  RX P=0/Q=0 (socket 0) -> TX P=1/Q=0 (socket 0) peer=02:00:00:00:00:01
  RX P=1/Q=0 (socket 0) -> TX P=0/Q=0 (socket 0) peer=02:00:00:00:00:00

  io packet forwarding packets/burst=32
  nb forwarding cores=1 - nb forwarding ports=2
  port 0: RX queue number: 1 Tx queue number: 1
    Rx offloads=0x0 Tx offloads=0x0
    RX queue: 0
      RX desc=0 - RX free threshold=0
      RX threshold registers: pthresh=0 hthresh=0  wthresh=0
```

```
    RX Offloads=0x0
  TX queue: 0
    TX desc=0 - TX free threshold=0
    TX threshold registers: pthresh=0 hthresh=0  wthresh=0
    TX offloads=0x0 - TX RS bit threshold=0
port 1: RX queue number: 1 Tx queue number: 1
  Rx offloads=0x0 Tx offloads=0x0
  RX queue: 0
    RX desc=0 - RX free threshold=0
    RX threshold registers: pthresh=0 hthresh=0  wthresh=0
    RX Offloads=0x0
  TX queue: 0
    TX desc=0 - TX free threshold=0
    TX threshold registers: pthresh=0 hthresh=0  wthresh=0
    TX offloads=0x0 - TX RS bit threshold=0
```

- Viewing Forwarding Data:

```
testpmd> show port stats all

######################## NIC statistics for port 0  ########################
RX-packets: 5240160    RX-missed: 0          RX-bytes:  314409600
RX-errors: 0
RX-nombuf:  0
TX-packets: 0          TX-errors: 0          TX-bytes:  0

Throughput (since last show)
Rx-pps:            0          Rx-bps:            0
Tx-pps:            0          Tx-bps:            0
############################################################################

######################## NIC statistics for port 1  ########################
RX-packets: 0          RX-missed: 0          RX-bytes:  0
RX-errors: 0
RX-nombuf:  0
TX-packets: 5180224    TX-errors: 0          TX-bytes:  310813440

Throughput (since last show)
Rx-pps:            0          Rx-bps:            0
Tx-pps:            0          Tx-bps:            0
############################################################################
testpmd> show port stats all

######################## NIC statistics for port 0  ########################
RX-packets: 6382336    RX-missed: 0          RX-bytes:  382940160
RX-errors: 0
RX-nombuf:  0
TX-packets: 0          TX-errors: 0          TX-bytes:  0

Throughput (since last show)
Rx-pps:       681809          Rx-bps:    327268480
Tx-pps:            0          Tx-bps:            0
############################################################################

######################## NIC statistics for port 1  ########################
RX-packets: 0          RX-missed: 0          RX-bytes:  0
RX-errors: 0
```

```
  RX-nombuf:  0
  TX-packets: 6322397    TX-errors: 0          TX-bytes:  379343820

  Throughput (since last show)
  Rx-pps:            0          Rx-bps:            0
  Tx-pps:       681810          Tx-bps:    327269160
  ############################################################################
```

- Other Common Settings:

  --nb-cores indicates the number of cores specified for dpdk-testpmd to be used for forwarding work

  --rxd receive queue descriptor

  --txd transmit queue descriptor

  --rxq indicates the number of receive queues specified for dpdk-testpmd, RK3568 has 1 queue

  --txq indicates the number of transmit queues specified for dpdk-testpmd, RK3568 has 1 queue

5.5 Running l2fwd

l2fwd requires at least two cores to test forwarding by default.

```
./dpdk-l2fwd -l 0,2,3  --main-lcore=0 --iova-mode=pa --vdev=net_stmmac0 --
vdev=net_stmmac1 -- -q 1 -p 0x3
```

The console information of l2fwd is refreshed every 10 seconds, which may cause packet loss. It is recommended to redirect the console information to a file.

Note: **--vdev=net_stmmac0 --vdev=net_stmmac1** indicates the specified virtual devices, which currently have fixed names (**PCIe does not need to be specified**).

5.6 Running l3fwd

```
./dpdk-l3fwd -l 3 -n 1 --iova-mode=pa --vdev=net_stmmac0 --vdev=net_stmmac1 -- -p
0x3 -P --config="(0,0,3),(1,0,3)" --parse-ptype
```

> -p PortMask parameter specifies the port mask used;
>
> --vdev=net_stmmac0 --vdev=net_stmmac1 indicates the specified virtual devices, which are currently fixed in name (**PCIe does not need to be specified**);
>
> -P parameter means setting all ports to promiscuous mode to receive all packets;
>
> --config (port,queue,lcore),[(port,queue,lcore)] parameter is used to configure the correspondence between ports, queues, and cores. For example, --config (0,0,3) means that port 0's queue 0 is processed by core 3;

It is worth noting that the above output printed the default routing rules of l3fwd, that is

```
LPM: Adding route 198.18.0.0 / 24 (0) [net_stmmac0]
LPM: Adding route 198.18.1.0 / 24 (1) [net_stmmac1]
LPM: Adding route 2001:200:: / 64 (0) [net_stmmac0]
LPM: Adding route 2001:200:0:1:: / 64 (1) [net_stmmac1]
```

That is to say, packets with a destination IP of the 198.18.0.0/24 segment will be forwarded through port 0, and packets with a destination IP of the 198.18.1.0 / 24 segment will be forwarded through port 1. The above default routing rules are configured in the source code, so when testing l3fwd, it is necessary to set the destination IP and source IP of the test data properly.

6. Pktgen

Running Pktgen on the board, which is based on DPDK, requires first compiling and installing DPDK. For compilation, refer to section 1.2 on compiling DPDK for the development board.

6.1 Download pktgen-dpdk Source Code

```
git clone http://dpdk.org/git/apps/pktgen-dpdk
apt-get install libpcap-dev
apt-get install libnuma-dev
apt install meson
apt install ninja-build pkg-config
```

6.2 DPDK Compilation

```
cd build
ninja
ninja install
ldconfig
export PKG_CONFIG_PATH=/usr/local/lib/aarch64-linux-gnu/pkgconfig/
```

6.3 Pktgen Compilation

```
cd pktgen-dpdk
meson build
cd build
ninja
```

6.4 Running the Pktgen Program

```
./build/app/pktgen --iova-mode=pa --vdev=net_stmmac0 -l 6,7 --proc-type auto --
log-level debug -- -P -m 7.0
```

In the EAL options section, you can refer to the DPDK EAL parameters. The most important parameter is the `-l` option, which is used to specify the list of cores to be used, for example: `-l 1,2` or `-l 1-2`, indicating the use of cores 1 and 2.

It is worth noting that at least two cores must be specified for pktgen, as pktgen requires one core to interact with users, such as responding to user input during the test process.
The most important parameter in the pktgen's own parameters is the `-m` option, which is used to specify the correspondence between the network ports and the cores, for example:
`-m 2.0`: It means that core 2 is responsible for handling network port 0. It is worth noting that if you want to specify multiple correspondences (using multiple network cards and multiple cores), you need to use the `-m` option multiple times.
If you want to receive packets, it is best to specify the `-P` option as well, indicating that all network ports enter promiscuous mode to receive all data packets.

Setting the packet format and starting Pktgen:

```
set 0 size 64
set 0 src ip 198.18.0.100/24
set 0 dst ip 198.18.1.101
set 0 dst mac ba:a0:3f:fd:b2:8c
start 0
```

7. Performance Metrics

Environment (Single Core)	L2 Forwarding	L3 Forwarding
RK3588 GMAC + 64 Byte	650Mbit/s	530Mbit/s
RK3588 GMAC + 1500 Byte	940Mbit/s	940Mbit/s
RK3588 Intel I210/350 PCIe + 64 Byte	N/A	737Mbit/s
RK3588 Intel I210/350 PCIe + 1500 Byte	940Mbit/s	940Mbit/s
RK3568 GMAC + 64 Byte	600Mbit/s	580Mbit/s
RK3568 GMAC + 1500 Byte	940Mbit/s	940Mbit/s

Note:

1. For multi-core, efficiency can be converted at around 85%, for example, if a single core is 1000Mbps, then 8 cores would be 1000 x 8 x 0.85 = 6.8Gbit/s;
2. N/A indicates that actual test data has not yet been provided and will be supplemented later, but L2 is generally greater than L3 as it lacks the operation of looking up the routing table.

8. Network Card Support List

Module	Interface	Rate
GMAC+PHY	GMII	1000M
Intel I210	PCIe x1	1000M
Intel I350	PCIe x4	1000M x4
RTL8111H	PCIe x1	1000M