

ET3600 Intelligent Gateway

User Manual

Product Model	ET3600
Product Version	V1.0
Document Version	V1.1
Intended Audience	User/Maintainer/Developer

Contents

1 Preface	6
1.1 Reference	6
1.2 Abbreviations	6
1.3 Command Description	6
1.4 Naming Specifications for Helium Product Packages	6
2 Introduction.....	8
2.1 Product Overview	8
2.2 Physical Drawing of ET3600	8
3 Installation Process	11
3.1 Scope of Application.....	11
3.2 Installation Steps	11
3.3 Software Configuration Checklist	11
3.4 Compatibility List	12
4 Preparation before Installation	13
4.1 Confirmation of Hardware and Network Environment.....	13
4.2 Preparation of Static Information	13
5 Software Configuration of ET3600	14
5.1 ET3600 Login	14
5.1.1 Serial Port Login	14
5.1.2 Management Network Port Login.....	16
5.2 PCI Address of ET3600.....	16
5.3 Network-Related Configuration	17
5.3.1 Static IP Configuration	17
5.3.2 Port Rate Settings for Panel Ports	18
6 DPDK Runtime Test.....	19

6.1 Compile and Install DPDK.....	19
6.2 Configuring System HugePage	19
6.3 Switching Port Drivers.....	20
6.4 Run Testpmd.....	20
6.5 DPDK-testpmd Running Example	21
7 VPP Runtime Test.....	24
7.1 Compile and Install VPP	24
7.2 Configuring System HugePage	24
7.3 Switching Port Drivers.....	24
7.4 Edit Configuration File.....	24
7.5 Run VPP	25
7.6 VPP Running Example	25
8 Appendix A Operating System Installation	29
8.1 Boot Installation from USB.....	29
8.1.1 Preparation Work.....	29
8.1.2 Boot Parameter Settings.....	29
8.1.3 Boot from USB Device	30
8.1.4 Install Linux.....	30
8.2 Boot Installation from Network	31
8.2.1 Preparation Work.....	31
8.2.2 Download BusyBox and System Installation Package	31
8.2.3 Install Linux.....	32

About This Document:

This user manual introduces the Asterfusion ET3600 intelligent gateway, providing information on the hardware parameters and specifications related to the intelligent gateway, as well as details about the required software and firmware for the intelligent gateway. It also offers step-by-step instructions on how to install the operating system on the intelligent gateway to ensure the device functions properly.

The final section of the document includes detailed testing methods and procedures for conducting DPDK traffic tests and VPP performance tests based on the ET3600.

Intended Audience:

This user manual is intended for ET3600 intelligent gateway users, installers, and developers.

Revision History:

Revision Time	Revision Content	Note
2025/12/25	Create	V1.0
2026/01/26	Documentation enhancement	V1.1

1 Preface

1.1 Reference

Serial Number	Reference Documents
1	https://www.dpdk.org/
2	https://s3-docs.fd.io/vpp/24.02/
3	https://github.com/asterfusion/Helium_DPU/

Table 1-1 References

1.2 Abbreviations

Abbreviations	EN	CN
PCIe	Peripheral Component Interconnect Express	高速串行计算机扩展总线标准
DPDK	Data Plane Development Kit	数据平面开发套件
VPP	Vector Packet Processor	矢量报文处理器

Table 1-2 Abbreviations

1.3 Command Description

In this document, all commands displayed in gray indicate that they can be run directly on the device.

The sample is as follows:

```
# uname -a
```

1.4 Naming Specifications for Helium Product Packages

The naming specifications of the ET3600 software package are as follows:

Product - Function - Version

The explanations are as follows, where parentheses indicate non-essential items:

1. Product: ET3600
2. Function: Describes the functionality provided or available by the package
3. Version : Release version of the software package. The version format is Vx.yRz-(date)
 - a) x - major version number, used for incompatibility changes
 - b) y - minor version number, used to identify new or modified features
 - c) z - revised version, used to identify problem fixes

d) date - the release date corresponding to the software package

2 Introduction

2.1 Product Overview

The Asterfusion ET3600 is an open intelligent gateway designed to address the intricate networking requirements of modern enterprises. In recent years, businesses have embraced technologies like cloud computing, big data, video conferencing, live streaming, and notably, the application of AI and IoT technologies has facilitated business transformation through machine empowerment. Traditional enterprise networks typically employ various dedicated devices to handle different functions, making enterprise networks complex, expensive, and difficult to operate and maintain. To address this issue, the ET3600 adopts a Computing-Networking fusion chip architecture and a decoupled open design, allowing enterprises to use a single intelligent device to perform all of the aforementioned functions.

2.2 Physical Drawing of ET3600



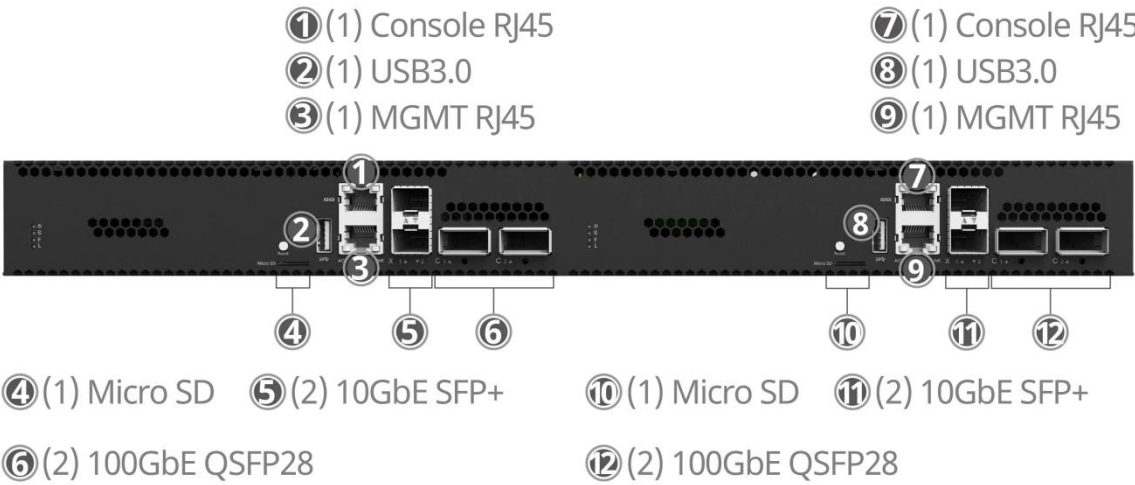


Figure 2-1 ET3600 Physical Products Front



Figure 2-2 ET3600 Physical Products Rear

Front panel status		
LED	Status	Description
Power LED	Green,always on	power ok
	Red,always on	power fail
SYS LED(S)	Green,always on	System is on and ready
	Green,blink	Booting
	Red,always on	When detect HW failed
	OFF	No power
FAN LED(F)	Green,always on	All fans are working on normal
	Red,always on	Any fan tray fault
	OFF	No power
Location LED (L)	Blue, Blink in 4Hz	Set by management to locate switch
	OFF	Function not active

Table 2-3 Front Panel Status Port LED

Front panel service port LED		
LED	Status	Description
SFP+ LED	Green,always on	link up,no data transmission
	Green,Blink	link up,data transmission
	OFF	link down
QSFP28 LED	Green,always on	link up,no data transmission
	Green,Blink	link up,data transmission
	OFF	link down

Table 2-4 Front Panel Service Port LED

3 Installation Process

3.1 Scope of Application

Scene	Description	Reference
Install operating system		8 Appendix A Operating System Installation
Set the port rate		5.3.2 Port Rate Settings for Panel Ports
DPDK runtime test		6 DPDK Runtime Test
VPP runtime test		7 VPP Runtime Test

Table 3-1 Scope of Application

3.2 Installation Steps

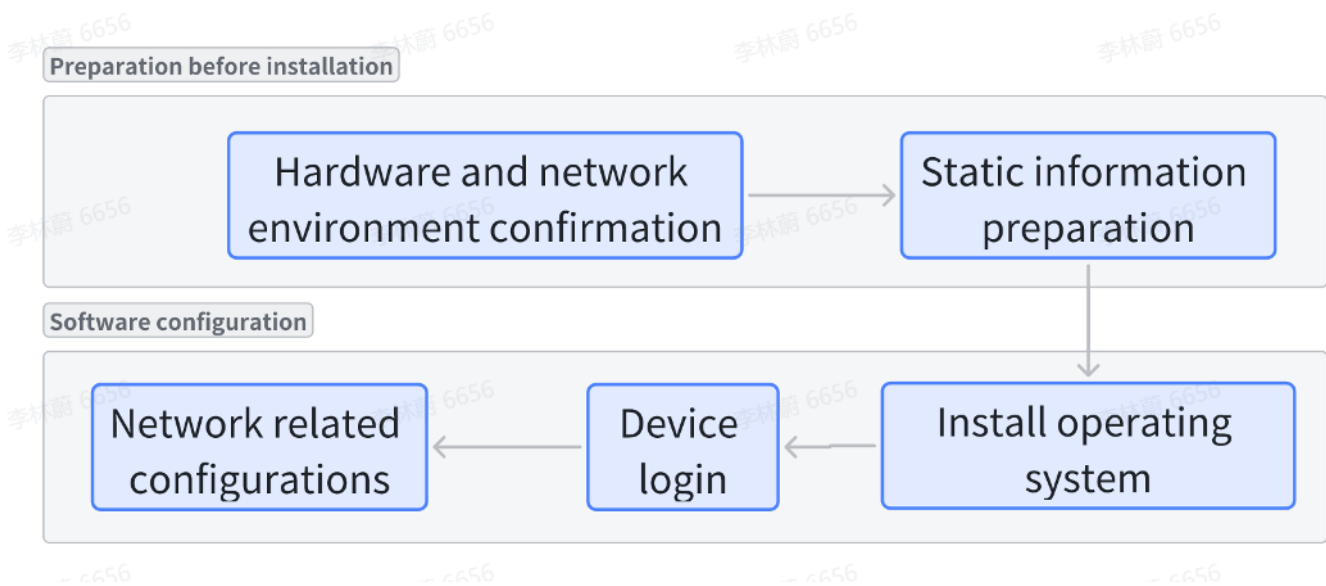


Figure 3-1 Installation Flow Chart

The overall installation process described in this manual is as follows: first, prepare for installation, then proceed with software configuration. Once the configuration is complete, the device can be used normally. After that, DPDK and VPP runtime tests can be conducted.

3.3 Software Configuration Checklist

Software Package Name	Content description	Download Location
ET3600-Debian12-Vx.yRz-date.tar	Debian12 OS for ET3600	Pre-installed with the operating system before shipment
ET3600-dpdk-24.11	DPDK for ET3600	ET3600-dpdk-24.11

Table 3-2 ET3600 Software Configuration Checklist

3.4 Compatibility List

OS	Kernel	gcc version
Debian12	6.6.46	12.2.0

Table 3-3 Compatibility List

4 Preparation before Installation

4.1 Confirmation of Hardware and Network Environment

Serial Number	Description
1	The ET3600 hardware has been completed, including the memory, power supply, fans, and other components.
2	The ET3600 is now ready for power-up.
3	The ET3600 is connected to the terminal host and can be accessed via the serial port.

Table 4-1 Confirmation of Hardware and Network Environment

First, check the ET3600 hardware appearance for any damage, including the memory, power supply, fans, heat sinks, and other components. Then, insert the power cable into the device and turn on the power socket switch to power on the device. Next, prepare an RJ45 to USB serial cable; connect the RJ45 end to the serial port on the right side of the device's front panel and the USB end to the terminal host.

4.2 Preparation of Static Information

Content	Source	Explanation
MAC address to be assigned	Uniformly distributed from the factory or modified on-site	Uniformly distributed from the factory or modified on site
IP address of the device to be deployed	Factory default assignment 192.168.2.100 or modified on-site	For remote access and software updates, use the default IP address if the device is not online

Table 4-2 Preparation of Static Information

5 Software Configuration of ET3600

The ET3600 smart gateway is pre-installed with the Debian 12 operating system. For updating or reinstalling the operating system, please refer to Appendix A.

5.1 ET3600 Login

There are two login modes for ET3600: serial port, management network port.

Note: The serial port pin definition, most likely Cisco Standard, you can buy the USB to RJ45 console cable:

RJ45 Pin	Signal Name	Signal Direction	Description
1	N/A		
2	N/A		
3	RxD	DCE ← DTE	Receive Data (Device receives)
4	GND	—	Signal Ground
5	GND	—	Signal Ground
6	TxD	DCE → DTE	Transmit Data (Device transmits)
7	N/A		
8	N/A		

5.1.1 Serial Port Login

On a Windows system, the user can use a serial connection-supported application, such as MobaXterm or HyperTerminal. Set the connection speed to **115200**, and follow the rest of the settings as shown in Figure 5-2, MobaXterm Advanced Settings, to successfully establish the connection.

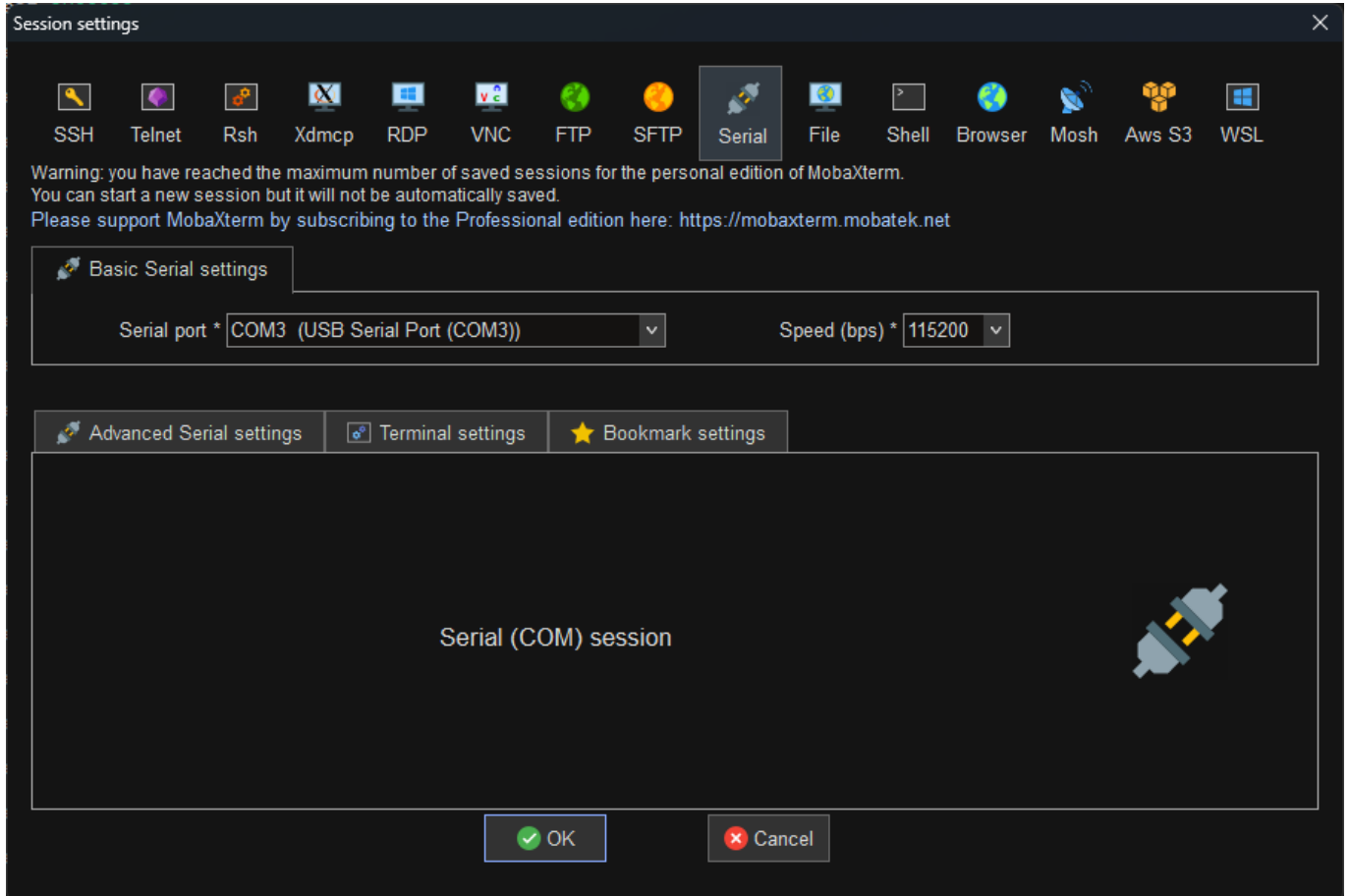
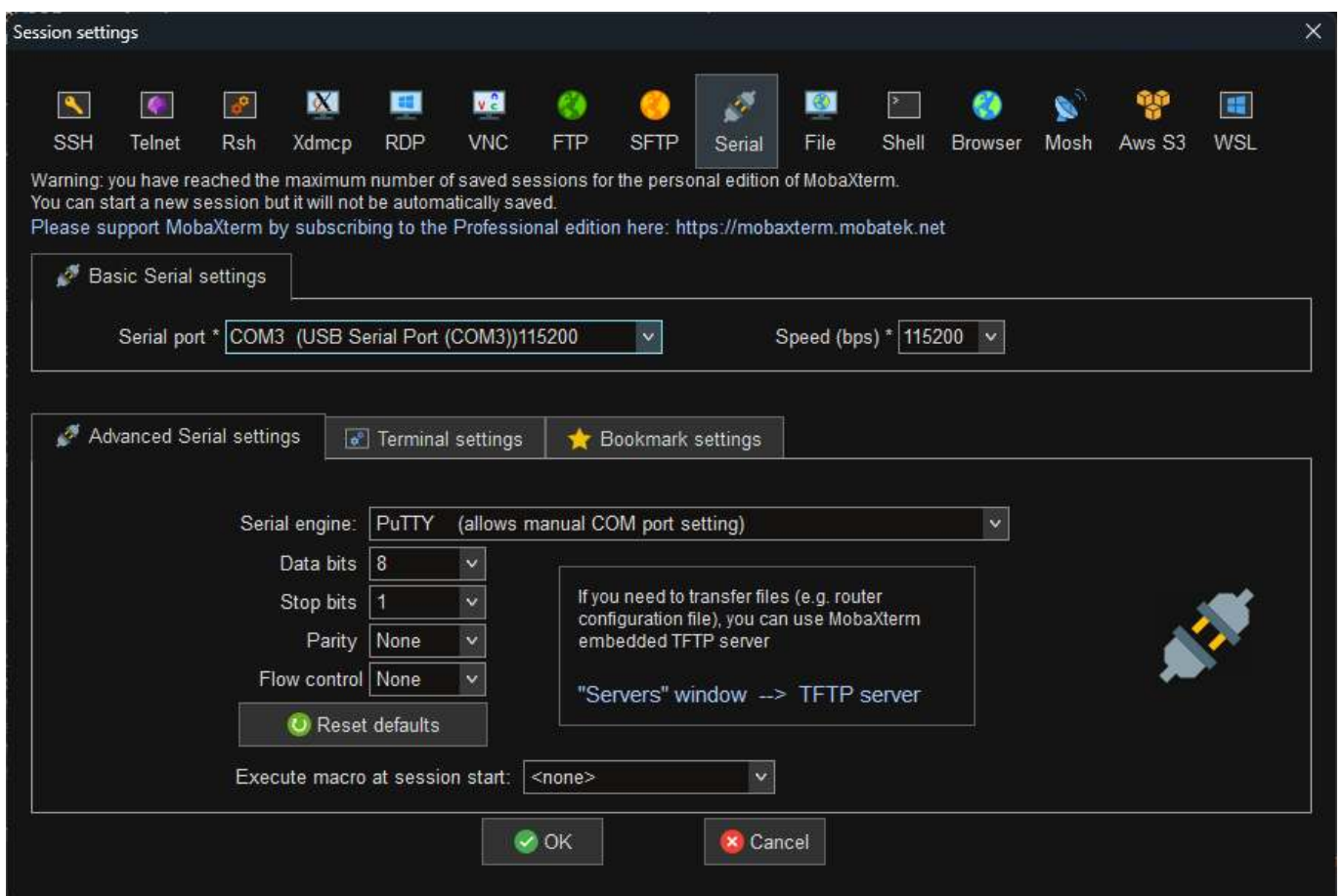


Figure 5-1 Mobaxterm Serial Port Connection



The configuration for Minicom in Linux is as follows:

```
+-----+
| A -   Serial Device       : /dev/ttyUSB0
| C -   Callin Program      :
| D -   Callout Program     :
| E -   Bps/Par/Bits        : 115200 8N1
| F -   Hardware Flow Control : No
| G -   Software Flow Control : No
|
| Change which setting? █
+-----+
```

Figure 5-3 Configuration for Minicom

The system's default username is `root`, and the password is `passok`.

5.1.2 Management Network Port Login

The default IP configuration for the management port eth0 is **192.168.2.100**, and you can log in to this address via SSH.

You can also configure the IP address manually by logging into the device through the serial port to set the management port IP address and configure SSH access permissions.

(For detailed network configuration, see section 5.3, "Network-Related Configuration.")

5.2 PCI Address of ET3600

The device's front panel ports correspond to the following PCI addresses:

Front Panel Ethernet Port Numbering	Linux System Network Interface Names	PCI address	Port Type
X1	X1	0002:03:00.0	10GE
X2	X2	0002:02:00.0	10GE
C1	C1	0002:05:00.0	100GE
C2	C2	0002:04:00.0	100GE
mgmt RJ45	eth0	usb device	1GE(RJ45)

Table 5-1 Device PCI Address Mapping

5.3 Network-Related Configuration

5.3.1 Static IP Configuration

The default management port is eth0, with a default IP of 192.168.2.100. If you need to modify the IP, follow these steps:

Example: Modify the default IP to 192.168.0.212

Step 1: Check the current system version

```
# cat /etc/os-release
```

If the current system is Debian 12:

Step 2: Modify the /etc/network/interfaces file

Open the file and input the following content, then save and close it:

```
# vim /etc/network/interfaces
```

```
auto eth0

iface eth0 inet static
    address 192.168.0.212
    netmask 255.255.255.0
    gateway 192.168.0.1
    broadcast 192.168.0.255
```

Step 3: Restart the network service

```
# sudo service networking restart
```

If the current system is Ubuntu 24:

Step 2: Modify the /etc/netplan/50-cloud-init.yaml file

```
# vim /etc/netplan/50-cloud-init.yaml
```

```

network:
  ethernet:
    eth0:
      dhcp4: no
      dhcp6: no
      addresses:
        -
192.168.0.212/24
      routes:
        - to: default
          via:
192.168.0.1
version: 2

```

Note: Pay attention to indentation when modifying the file.

Step 3: Restart the network service

```
# sudo netplan apply
```

5.3.2 Port Rate Settings for Panel Ports

The device can modify the X1/X2 panel port speed using the ethtool tool:

Panel Port Rate	ethtool Argument
10M	0x2
100M	0x8
1G	0x200000000000
10G	0x800000000000

Table 5-2 Ethtool Argument for Port Rate Settings

For example, change the panel port X1:

Modified to 10G:

```
# ethtool -s X1 advertise 0x800000000000
```

Modified to 1G:

```
# ethtool -s X1 advertise 0x200000000000
```

6 DPDK Runtime Test

6.1 Compile and Install DPDK

Download the DPDK source code from GitHub. Refer to the following link: [3.3 Software Configuration Checklist](#)

After the download is complete, navigate to the dpdk-24.10 folder:

```
# cd dpdk-24.10
```

Install the required dependencies, such as libnuma-dev:

```
# apt install -y libnuma-dev python3-pyelftools meson
```

Start the compilation process:

```
# meson build -Dmax_lcores=8
```

```
# ninja -C build
```

The compiled DPDK program is located in the build directory. If you want to install DPDK to the system, execute:

```
# ninja -C build install
```

6.2 Configuring System HugePage

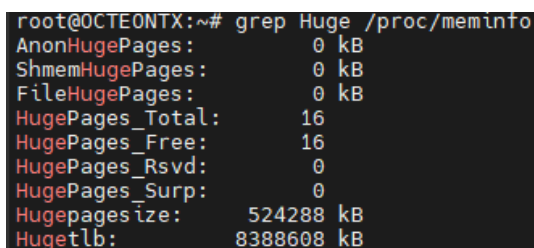
DPDK uses Huge Pages to improve memory management efficiency and reduce memory access latency, enhancing packet processing performance. Here are the configuration methods for Huge Pages:

The default Huge Page size for the ET3600 is 524288 kB(512MB). To allocate 8G of Huge Pages, you need to change the number of Huge Pages to 16:

```
# sysctl -w vm.nr_hugepages=16
```

The command to check the current system's Huge Pages is:

```
# grep Huge /proc/meminfo
```



```
root@OCTEONTX:~# grep Huge /proc/meminfo
AnonHugePages:      0 kB
ShmemHugePages:     0 kB
FileHugePages:      0 kB
HugePages_Total:    16
HugePages_Free:     16
HugePages_Rsvd:      0
HugePages_Surp:      0
Hugepagesize:       524288 kB
Hugetlb:             8388608 kB
```

Figure 6-1 check the current system's Huge Pages

Here are the meanings of each field:

HugePages_Total: The total number of Huge Pages configured in the system.

HugePages_Free: The number of Huge Pages that are currently free and available for allocation.

HugePages_Rsvd: The number of Huge Pages that are reserved but not currently in use.

HugePages_Surp: The number of Huge Pages that are allocated but not used by any process.

Hugepagesize: The size of each Huge Page (default 524288kB).

Hugetlb: The total Huge Page memory size.

6.3 Switching Port Drivers

Before running DPDK, it is necessary to switch the relevant port driver from `rvu_nicpf` to `vfio-pci`, allowing DPDK to access the port devices directly without going through the operating system. To switch the driver, use the `dpdk-24.10/usertools/dpdk-devbind.py` script. Here are the operation methods (the following commands should be executed after DPDK is installed; if not installed, include the path to `dpdk-devbind.py`):

To query the current system for all PCI devices and their status, use the following commands:

```
# dpdk-devbind.py -s
```

```
# dpdk-devbind.py -status
```

To bind a device to the DPDK driver, use the following commands:

```
# dpdk-devbind.py -bind vfio-pci 0002:02:00.0
```

```
# dpdk-devbind.py --bind=vfio-pci 0002:02:00.0
```

To unbind a device, you can check the help information with these commands:

```
# dpdk-devbind.py -u 0002:02:00.0
```

```
# dpdk-devbind.py --unbind 0002:02:00.0
```

To view the help documentation for `dpdk-devbind.py`, use the following command:

```
# dpdk-devbind.py -h
```

```
# dpdk-devbind.py -help
```

6.4 Run Testpmd

To run `testpmd` using 7 cores with a packet length of 1500 bytes, use the following command:

```
# dpdk-testpmd -l 0-7 -- --nb-cores=7 --txpkts=1500 -i
```

To start packet transmission, enter the following command in the `testpmd` prompt:

```
testpmd> start tx_first
```

To view information about all ports, use:

```
testpmd> show port info all
```

To see the statistics for all ports, use:

```
testpmd> show port stats all
```

To view the forwarding configuration, enter:

```
testpmd> show config fwd
```

To stop testpmd, use:

```
testpmd> stop
```

To quit testpmd prompt, use:

```
testpmd> quit
```

6.5 DPDK-testpmd Running Example

In this example, DPDK is installed on the system, using the network ports C1 and C2 as test ports. Connect C1 to C2 using optical fiber or a network cable, and then bind the ports using the DPDK driver.

To simplify the test, the "send then forward" mode (tx_first) will be used.

To begin, compile and install DPDK as described in Section [6.1: Compiling and Installing DPDK](#). Below are partial screenshots from the compilation and installation process.

In this example, the command `# ninja -C build install` is executed to install the compiled DPDK into the system, making it easier to run. As shown in the figure, the top part displays the output of the compilation process, while the bottom part shows the output of the installation process.

After that, configure HugePages and drivers, allocate 8GB of HugePages, switch to the DPDK driver, and connect the two test ports using a network cable or optical fiber.

```
# sysctl -w vm.nr_hugepages=16
```

```
# grep Huge /proc/meminfo
```

```
root@OCTEONTX:~# sysctl -w vm.nr_hugepages=16
vm.nr_hugepages = 16
root@OCTEONTX:~# grep Huge /proc/meminfo
AnonHugePages:          0 kB
ShmemHugePages:         0 kB
FileHugePages:          0 kB
HugePages_Total:       16
HugePages_Free:        16
HugePages_Rsvd:         0
HugePages_Surp:         0
Hugepagesize:       524288 kB
Hugetlb:               8388608 kB
```

Figure 6-3 Allocating 16 HugePages

```
# dpdk-devbind.py -b vfio-pci 0002:05:00.0 0002:04:00.0
```

```
# dpdk-devbind.py -s
```

Then run testpmd:

```
# dpdk-testpmd -l 0-7 -- --nb-cores=7 --txpkts=1500 -i
```

```
root@OCTEONTX:~# dpdk-testpmd -l 0-7 -- --nb-cores=7 --txpkts=1500 -i
EAL: Detected CPU lcores: 8
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 'VA'
EAL: VFIO support initialized
EAL: Using IOMMU type 1 (Type 1)
CNXK: RoC Model: cn10kb_a0 (HW_PLATFORM)
CNXK: Skipping device, mcs device already probed
Interactive-mode selected
testpmd: create a new mbuf pool <mb_pool_0>: n=203456, size=2176, socket=0
testpmd: preferred mempool ops selected: cn10k_mempool_ops
Configuring Port 0 (socket 0)
CNXK: User passed link configuration: 0
CNXK: Port 0: Link Up - speed 100000 Mbps - full-duplex - Twisted Pair

Port 0: link state change event
Port 0: 72:A1:BE:5A:3F:DF
Configuring Port 1 (socket 0)
CNXK: User passed link configuration: 0
CNXK: Port 1: Link Up - speed 100000 Mbps - full-duplex - Twisted Pair

Port 1: link state change event
Port 1: 8A:A2:05:BC:EC:68
Checking link statuses...
Done
testpmd> █
```

Figure 6-5 run testpmd

As shown in the figure, the startup was successful, and commands can be entered normally in the testpmd prompt:

```
testpmd> start tx_first
```

```
testpmd> start tx_first
io packet forwarding - ports=2 - cores=2 - streams=2 - NUMA support enabled, MP allocation mode: native
Logical Core 1 (socket 0) forwards packets on 1 streams:
RX P=0/Q=0 (socket 0) -> TX P=1/Q=0 (socket 0) peer=02:00:00:00:00:01
Logical Core 2 (socket 0) forwards packets on 1 streams:
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=7 - nb forwarding ports=2
port 0: RX queue number: 1 Tx queue number: 1
Rx offloads=0x0 Tx offloads=0x10000
RX queue: 0
RX desc=4096 - RX free threshold=0
RX threshold registers: pthresh=0 hthresh=0 wthresh=0
RX offloads=0x0
TX queue: 0
TX desc=512 - TX free threshold=0
TX threshold registers: pthresh=0 hthresh=0 wthresh=0
TX offloads=0x10000 - TX RS bit threshold=0
```

Figure 6-6 Start Forwarding

When forwarding is started, display the port statistics:

```
testpmd> show port stats all
```

```
testpmd> show port stats all

##### NIC statistics for port 0 #####
RX-packets: 160658744 RX-missed: 0 RX-bytes: 240988116000
RX-errors: 0
RX-nombuf: 0
TX-packets: 160504568 TX-errors: 0 TX-bytes: 240756852000

Throughput (since last show)
Rx-pps: 4214786 Rx-bps: 50577434952
Tx-pps: 4210746 Tx-bps: 50528960008
#####

##### NIC statistics for port 1 #####
RX-packets: 160504657 RX-missed: 0 RX-bytes: 240756985500
RX-errors: 0
RX-nombuf: 0
TX-packets: 160658874 TX-errors: 0 TX-bytes: 240988311000

Throughput (since last show)
Rx-pps: 4210747 Rx-bps: 50528972312
Tx-pps: 4214789 Tx-bps: 50577468760
#####
testpmd>
```

Figure 6-7 display the port statistics

Use this command to view the packet counts, transmission rates, and packet loss for all ports. As shown in the figure, each port rate is 50 Gbps, total throughput is 100Gbps. Finally, use the stop command to stop forwarding and the quit command to exit the testpmd command line.

```
testpmd> stop
```

```
testpmd> quit
```

7 VPP Runtime Test

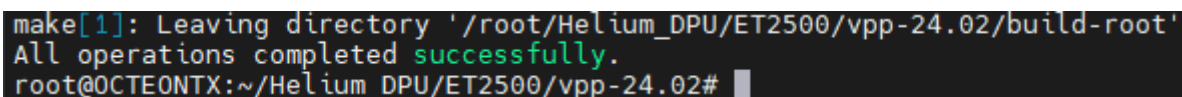
7.1 Compile and Install VPP

Download the VPP source code from GitHub, referring to the address in section [3.3 Software Configuration Checklist](#).

Once the download is complete, navigate to the vpp-24.02 folder and execute the build script:

```
# cd vpp-24.02
# ./vpp-build.sh -release
```

During the compilation process, you may encounter some options that require manual confirmation. Follow the prompts and input y or take action based on the specific situation.



```
make[1]: Leaving directory '/root/Helium_DPU/ET2500/vpp-24.02/build-root'
All operations completed successfully.
root@OCTEONTX:~/Helium_DPU/ET2500/vpp-24.02#
```

Figure 7-1 Compile VPP

Printing this message indicates successful compilation.

Install VPP:

```
# make pkg-deb
# dpkg -i build-root/*.deb
```

7.2 Configuring System HugePage

Refer to section [6.2 Configuring System HugePage](#).

7.3 Switching Port Drivers

The script path for switching drivers is:

vpp-24.02/extras/vpp_config/scripts/[dpdk-devbind.py](#)

Refer to section [6.3 Switching Port Drivers](#).

7.4 Edit Configuration File

The configuration file can refer to the example configuration file provided by VPP:

vpp-24.02/build-root/install-vpp-native/vpp/etc/vpp/onp-startup.conf

Note: This configuration file is generated only after the compilation is complete.

The following are explanations for some fields in the configuration file:

unix Configuration Block: Configures some basic behaviors of VPP.

api-trace Configuration Block: Controls the tracing functionality of API calls. This feature helps with debugging and tracking API calls.

cpu Configuration Block: Configures the CPU cores used by VPP.

main-core Block: Sets the CPU core to which the main thread is bound, defaulting to core 1.

corelist-workers Block: Sets the range of CPU cores to which worker threads are bound.

buffers Configuration Block: Adjusts buffer allocation settings.

onp Configuration Block: Configures whitelist devices and queue parameters.

dev: Specifies the PCIe address of the device.

plugins Configuration Block: Enables or disables plugins within VPP.

statseg Configuration Block: Configures the statistics segment for monitoring and collecting statistical data.

7.5 Run VPP

After writing the configuration file, use the -C parameter to specify the configuration file for VPP and run VPP:

```
# vpp -c ET3600_startup.conf
```

For additional commands, please refer to the VPP documentation: <https://s3-docs.fd.io/vpp/24.02/>

7.6 VPP Running Example

This run example uses another device, TRex, to send packets, while VPP enables Layer 3 routing forwarding. C1(renamed with Ethernet12 in vpp) and C2(renamed with Ethernet13 in vpp) are bound to VPP and connected to another device.

The test network topology is as follows:

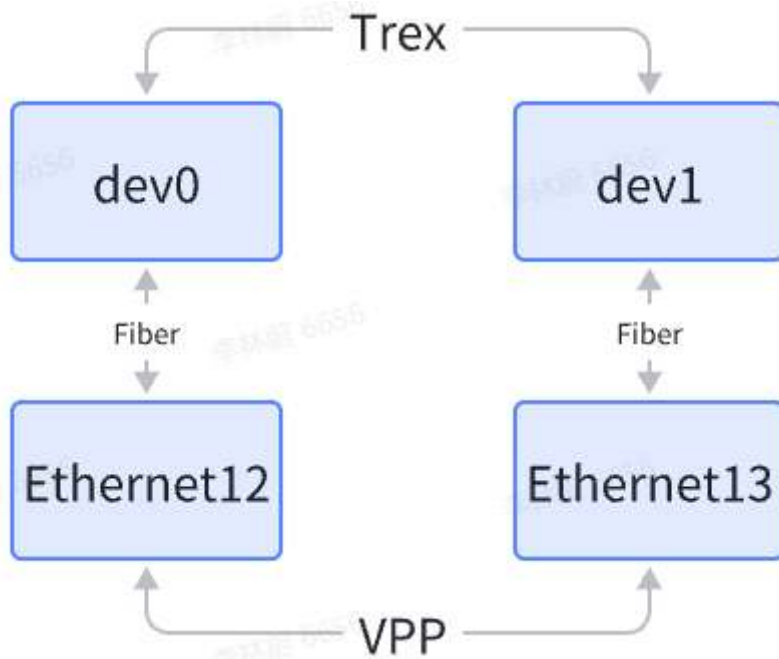


Figure 7-2 Test Network Topology

The VPP configuration file is as follows

```

unix {
    interactive
}

api-trace {
    on
}

memory {
    main-heap-size 1G
    main-heap-page-size default-hugepage
}

cpu {
    main-core 1
    corelist-workers 2-3
}
  
```

```
}

buffers {
    buffers-per-numa 128000
    page-size default-hugepage
}

plugins {
    plugin default { disable }
    plugin dpdk_plugin.so { disable }
    plugin onp_plugin.so { enable }
    plugin ikev2_plugin.so { enable }
    plugin idpf_plugin.so { enable }
    plugin oddbuf_plugin.so { enable }
}

onp {
    dev 0002:05:00.0 {
        name Ethernet12
    }
    dev 0002:04:00.0 {
        name Ethernet13
    }
}

statseg {
    size 1G
}
```

Start VPP and Configure:

```
# ./vpp -c ../ET3600_startup.conf
```

```
vpctl> set interface ip address Ethernet12 10.10.13.2/24
```

```
vpctl> set interface ip address Ethernet13 10.10.14.2/24
```

```
vppctl> set interface state Ethernet12 up
```

```
vppctl> set interface state Ethernet13 up
```

```
vppctl> set interface mac address Ethernet12 aa:bb:cc:dd:ee:13
```

```
vppctl> set interface mac address Ethernet13 aa:bb:cc:dd:ee:14
```

```
vppctl> set ip neighbor Ethernet12 10.10.13.1 d2:d2:1c:43:a6:13
```

```
vppctl> set ip neighbor Ethernet13 10.10.14.1 d2:d2:1c:43:a6:14
```

At this point, start the packet generation software Trex and check the port status:

```
vppctl> show interface
```

8 Appendix A Operating System Installation

8.1 Boot Installation from USB

8.1.1 Preparation Work

Connect the device via the serial port.

Backup important files and data from the eMMC device.

Prepare a USB storage device containing a complete Linux file system and Linux kernel image, such as the BusyBox system.

Download the latest Linux image compressed package to the prepared storage device, for example ET3600-Debian12-V1.0R2-20240903.tar.

8.1.2 Boot Parameter Settings

In this step, the boot disk needs to be changed from eMMC to a USB device.

First, restart the device:

```
# reboot
```

When the message "Autoboot in 5 seconds" appears, press the **S** and **T** keys on the keyboard to enter the Uboot interface.

In the Uboot interface, print the current boot parameters:

```
ET3600> printenv
```

Insert the prepared USB and start Uboot's USB:

```
ET3600> usb start
```

If you see the message "0 Storage Device(s) found," you need to run:

```
ET3600> usb reset
```

Repeat until the storage device appears.

If you are unsure about the device, you can check the USB storage contents using the ls command. For example, if the current storage device is formatted as ext4, you can use ext4ls:

```
ET3600> usb start
```

```
ET3600> ext4ls usb 0:1
```

This represents the first partition of the 0th USB device.

When there are multiple storage devices, they are sorted in order as **sda**, **sdb**, **sdc**, etc.

When a storage device has multiple partitions, they are sorted as **sda1**, **sda2**, **sda3**, etc.

Modify the bootargs to specify the prepared storage device, for example root=/dev/sda1:

```
ET3600> editenv bootargs
```

The Uboot command line will display the current bootargs, modify it to:

```
console=ttyAMA0,115200n8 earlycon=pl011,0x87e028000000 maxcpus=8 rootwait rw  
root=/dev/sda1 coherent_pool=16M net.ifnames=0
```

Note: After modifying with editenv bootargs, use printenv to verify the changes.

(Optional) Store the environment variables: You can save the environment variables to non-volatile storage:

```
ET3600> saveenv
```

8.1.3 Boot from USB Device

Load the kernel onto the device:

```
ET3600> ext4load usb 0:1 $loadaddr boot/Image
```

This represents the Image kernel located in the /boot folder of the first partition of the 0th USB device.

Adjust according to your situation.

Start BusyBox:

```
ET3600> booti $loadaddr - $fdtcontroladdr
```

8.1.4 Install Linux

Extract the installation package into a new folder:

```
# mkdir temp
```

```
# tar -xvf ET3600-Debian12-V1.0R2-20240903.tar -C temp
```

```
# cd temp
```

Note: If you are using a single USB device to install multiple systems, you need to extract each system into an empty folder each time.

After successful extraction, you should see the following folders:

driver.tar.gz

nvmeAutoInstall.sh

ET3600-uboot.img

extra_files

Image

partition.sh

```
rootfs_debian12.tar.gz
```

```
rootfs_patch.sh
```

Run the nvmeAutoInstall.sh script:

```
# ./nvmeAutoInstall.sh
```

If you do not have execution permissions, you need to run:

```
# sudo chmod +x nvmeAutoInstall.sh
```

Wait for the installation to complete, then restart the device to boot into the newly installed system.

8.2 Boot Installation from Network

8.2.1 Preparation Work

Connect to the device via the serial port.

Backup important files and data from the eMMC device.

Prepare a TFTP server and connect it to the device's management network port.

Prepare the BusyBox system and the Linux installation package, such as ET3600-Debian12-V1.0R2-20240903.tar

8.2.2 Download BusyBox and System Installation Package

The default management network port for the ET3600 device is ax88179_eth. Execute `ET3600> ethlist` to confirm whether the ax88179_eth port exists. If it does not, enter `ET3600> usb reset` to activate the management network port.

Note: After rebooting, you need to re-confirm whether the management network port is activated.

Place the BusyBox image on the TFTP server, configure the management network port's IP and TFTP server IP to ensure both are on the same subnet. Then, use TFTP to download the BusyBox image and enter BusyBox.

```
ET3600> usb reset
```

```
ET3600> setenv ipaddr {ip}
```

```
ET3600> setenv serverip {tftp_server_ip}
```

```
ET3600> ethact
```

```
ET3600> tftpboot $loadaddr ET3600-busybox-Vx.yRz.img;booti $loadaddr -  
$fdtcontroladdr
```

Enter the BusyBox terminal to download the Linux system package:

```
# ifconfig eth4 <ip>  
# ip route add default via <gateway_ip>  
# scp root@<server_ip>:~/ET3600-Debian12-Vx.yRz.tar ./
```

8.2.3 Install Linux

Refer to Section [8.1.4 Install Linux](#).