Linux Base Driver for the Intel(R) Ethernet 10 Gigabit PCI Express Adapters

Contents

Identifying Your Adapter

The driver is compatible with devices based on the following:

- Intel(R) Ethernet Controller 82598
- Intel(R) Ethernet Controller 82599
- Intel(R) Ethernet Controller X520
- Intel(R) Ethernet Controller X540
- Intel(R) Ethernet Controller x550
- Intel(R) Ethernet Controller X552
- Intel(R) Ethernet Controller X553

For information on how to identify your adapter, and for the latest Intel network drivers, refer to the Intel Support website: https://www.intel.com/support

SFP+ Devices with Pluggable Optics

82599-BASED ADAPTERS

NOTES: - If your 82599-based Intel(R) Network Adapter came with Intel optics or is an Intel(R) Ethernet Server Adapter X520-2, then it only supports Intel optics and/or the direct attach cables listed below. - When 82599-based SFP+ devices are connected back to back, they should be set to the same Speed setting via ethtool. Results may vary if you mix speed settings.

Supplier	Type	Part Numbers
SR Modules		
Intel	DUAL RATE 1G/10G SFP+ SR (bailed)	FTLX8571D3BCV-IT
Intel	DUAL RATE 1G/10G SFP+ SR (bailed)	AFBR-703SDZ-IN2
Intel	DUAL RATE 1G/10G SFP+ SR (bailed)	AFBR-703SDDZ-IN1
LR Modules		
Intel	DUAL RATE 1G/10G SFP+ LR (bailed)	FTLX1471D3BCV-IT
Intel	DUAL RATE 1G/10G SFP+ LR (bailed)	AFCT-701SDZ-IN2
Intel	DUAL RATE 1G/10G SFP+ LR (bailed)	AFCT-701SDDZ-IN1

The following is a list of 3rd party SFP+ modules that have received some testing. Not all modules are applicable to all devices.

Supplier	Type	Part Numbers
Finisar	SFP+ SR bailed, 10g single rate	FTLX8571D3BCL
Avago	SFP+ SR bailed, 10g single rate	AFBR-700SDZ
Finisar	SFP+ LR bailed, 10g single rate	FTLX1471D3BCL
Finisar	DUAL RATE 1G/10G SFP+ SR (No Bail)	FTLX8571D3QCV-IT
Avago	DUAL RATE 1G/10G SFP+ SR (No Bail)	AFBR-703SDZ-IN1
Finisar	DUAL RATE 1G/10G SFP+ LR (No Bail)	FTLX1471D3QCV-IT
Avago	DUAL RATE 1G/10G SFP+ LR (No Bail)	AFCT-701SDZ-IN1
Finisar	1000BASE-T SFP	FCLF8522P2BTL
Avago	1000BASE-T	ABCU-5710RZ
HP	1000BASE-SX SFP	453153-001

82599-based adapters support all passive and active limiting direct attach cables that comply with SFF-8431 v4.1 and SFF-8472 v10.4 specifications.

Laser turns off for SFP+ when ifconfig ethX down

"ifconfig ethX down" turns off the laser for 82599-based SFP+ fiber adapters. "ifconfig ethX up" turns on the laser. Alternatively, you can use "ip link set [down/up] dev ethX" to turn the laser off and on.

82599-based QSFP+ Adapters

NOTES: - If your 82599-based Intel(R) Network Adapter came with Intel optics, it only supports Intel optics. - 82599-based QSFP+ adapters only support 4x10 Gbps connections. 1x40 Gbps connections are not supported. QSFP+ link partners must be configured for 4x10 Gbps. - 82599-based QSFP+ adapters do not support automatic link speed detection. The link speed must be configured to either 10 Gbps or 1 Gbps to match the link partners speed capabilities. Incorrect speed configurations will result in failure to link. - Intel(R) Ethernet Converged Network Adapter X520-Q1 only supports the optics and direct attach cables listed below.

Supplier	Type	Part Numbers
Intel	DUAL RATE 1G/10G QSFP+ SRL (bailed)	E10GQSFPSR

82599-based QSFP+ adapters support all passive and active limiting QSFP+ direct attach cables that comply with SFF-8436 v4.1 specifications.

82598-BASED ADAPTERS

NOTES: - Intel(r) Ethernet Network Adapters that support removable optical modules only support their original module type (for example, the Intel(R) 10 Gigabit SR Dual Port Express Module only supports SR optical modules). If you plug in a different type of module, the driver will not load. - Hot Swapping/hot plugging optical modules is not supported. - Only single speed, 10 gigabit modules are supported. - LAN on Motherboard (LOMs) may support DA, SR, or LR modules. Other module types are not supported. Please see your system documentation for details.

The following is a list of SFP+ modules and direct attach cables that have received some testing. Not all modules are applicable to all devices.

Supplier	Type	Part Numbers
Finisar	SFP+ SR bailed, 10g single rate	FTLX8571D3BCL
Avago	SFP+ SR bailed, 10g single rate	AFBR-700SDZ
Finisar	SFP+ LR bailed, 10g single rate	FTLX1471D3BCL

82598-based adapters support all passive direct attach cables that comply with SFF-8431 v4.1 and SFF-8472 v10.4 specifications. Active direct attach cables are not supported.

Third party optic modules and cables referred to above are listed only for the purpose of highlighting third party specifications and potential compatibility, and are not recommendations or endorsements or sponsorship of any third party's product by Intel. Intel is not endorsing or promoting products made by any third party and the third party reference is provided only to share information regarding certain optic modules and cables with the above specifications. There may be other manufacturers or suppliers, producing or supplying optic modules and cables with similar or matching descriptions. Customers must use their own discretion and diligence to purchase optic modules and cables from any third party of their choice. Customers are solely responsible for assessing the suitability of the product and/or devices and for the selection of the vendor for purchasing any product. THE OPTIC MODULES AND CABLES REFERRED TO ABOVE ARE NOT WARRANTED OR SUPPORTED BY INTEL. INTEL ASSUMES NO LIABILITY WHATSOEVER, AND INTEL DISCLAIMS ANY EXPRESS OR IMPLIED WARRANTY, RELATING TO SALE AND/OR USE OF SUCH THIRD PARTY PRODUCTS OR SELECTION OF VENDOR BY CUSTOMERS.

Command Line Parameters

max_vfs

Valid Range: 1-63

This parameter adds support for SR-IOV. It causes the driver to spawn up to max_vfs worth of virtual functions. If the value is greater than 0 it will also force the VMDq parameter to be 1 or more.

NOTE: This parameter is only used on kernel 3.7.x and below. On kernel 3.8.x and above, use sysfs to enable VFs. Also, for Red Hat distributions, this parameter is only used on version 6.6 and older. For version 6.7 and newer, use sysfs. For example:

```
#echo $num_vf_enabled > /sys/class/net/$dev/device/sriov_numvfs // enable VFs
#echo 0 > /sys/class/net/$dev/device/sriov_numvfs               //disable VFs
```

The parameters for the driver are referenced by position. Thus, if you have a dual port adapter, or more than one adapter in your system, and want N virtual functions per port, you must specify a number for each port with each parameter separated by a comma. For example:

```
modprobe ixgbe max_vfs=4
```

This will spawn 4 VFs on the first port.

```
modprobe ixgbe max_vfs=2,4
```

This will spawn 2 VFs on the first port and 4 VFs on the second port.

NOTE: Caution must be used in loading the driver with these parameters. Depending on your system configuration, number of slots, etc., it is impossible to predict in all cases where the positions would be on the command line.

NOTE: Neither the device nor the driver control how VFs are mapped into config space. Bus layout will vary by operating system. On operating systems that support it, you can check sysfs to find the mapping.

NOTE: When either SR-IOV mode or VMDq mode is enabled, hardware VLAN filtering and VLAN tag stripping/insertion will remain enabled. Please remove the old VLAN filter before the new VLAN filter is added. For example,

```
ip link set eth0 vf 0 vlan 100 // set VLAN 100 for VF 0
ip link set eth0 vf 0 vlan 0   // Delete VLAN 100
ip link set eth0 vf 0 vlan 200 // set a new VLAN 200 for VF 0
```

With kernel 3.6, the driver supports the simultaneous usage of max_vfs and DCB features, subject to the constraints described below. Prior to kernel 3.6, the driver did not support the simultaneous operation of max_vfs greater than 0 and the DCB features (multiple traffic classes utilizing Priority Flow Control and Extended Transmission Selection).

When DCB is enabled, network traffic is transmitted and received through multiple traffic classes (packet buffers in the NIC). The traffic is associated with a specific class based on priority, which has a value of 0 through 7 used in the VLAN tag. When SR-IOV is not enabled, each traffic class is associated with a set of receive/transmit descriptor queue pairs. The number of queue pairs for a given traffic class depends on the hardware configuration. When SR-IOV is enabled, the descriptor queue pairs are grouped into pools. The Physical Function (PF) and each Virtual Function (VF) is allocated a pool of receive/transmit descriptor queue pairs. When multiple traffic classes are configured (for example, DCB is enabled), each pool contains a queue pair from each traffic class. When a single traffic class is configured in the hardware, the pools contain multiple queue pairs from the single traffic class.

The number of VFs that can be allocated depends on the number of traffic classes that can be enabled. The configurable number of traffic classes for each enabled VF is as follows: 0 - 15 VFs = Up to 8 traffic classes, depending on device support 16 - 31 VFs = Up to 4 traffic classes 32 - 63 VFs = 1 traffic class

When VFs are configured, the PF is allocated one pool as well. The PF supports the DCB features with the constraint that each traffic class will only use a single queue pair. When zero VFs are configured, the PF can support multiple queue pairs per traffic class.

allow_unsupported_sfp

Valid Range: 0,1
Default Value: 0 (disabled)

This parameter allows unsupported and untested SFP+ modules on 82599-based adapters, as long as the type of module is known to the driver.

debug

Valid Range: 0-16 (0=none,...,16=all)
Default Value: 0

This parameter adjusts the level of debug messages displayed in the system logs.

Additional Features and Configurations

Flow Control

Ethernet Flow Control (IEEE 802.3x) can be configured with ethtool to enable receiving and transmitting pause frames for ixgbe. When transmit is enabled, pause frames are generated when the receive packet buffer crosses a predefined threshold. When receive is enabled, the transmit unit will halt for the time delay specified when a pause frame is received.

NOTE: You must have a flow control capable link partner.

Flow Control is enabled by default.

Use ethtool to change the flow control settings. To enable or disable Rx or Tx Flow Control:

```
ethtool -A eth? rx <on|off> tx <on|off>
```

Note: This command only enables or disables Flow Control if auto-negotiation is disabled. If auto-negotiation is enabled, this command changes the parameters used for auto-negotiation with the link partner.

To enable or disable auto-negotiation:

```
ethtool -s eth? autoneg <on|off>
```

Note: Flow Control auto-negotiation is part of link auto-negotiation. Depending on your device, you may not be able to change the auto-negotiation setting.

NOTE: For 82598 backplane cards entering 1 gigabit mode, flow control default behavior is changed to off. Flow control in 1 gigabit mode on these devices can lead to transmit hangs.

Intel(R) Ethernet Flow Director

The Intel Ethernet Flow Director performs the following tasks:

- Directs receive packets according to their flows to different queues.
- Enables tight control on routing a flow in the platform.
- Matches flows and CPU cores for flow affinity.
- Supports multiple parameters for flexible flow classification and load balancing (in SFP mode only).

NOTE: Intel Ethernet Flow Director masking works in the opposite manner from subnet masking. In the following command:

```
#ethtool -N eth11 flow-type ip4 src-ip 172.4.1.2 m 255.0.0.0 dst-ip \
172.21.1.1 m 255.128.0.0 action 31
```

The src-ip value that is written to the filter will be 0.4.1.2, not 172.0.0.0 as might be expected. Similarly, the dst-ip value written to the filter will be 0.21.1.1, not 172.0.0.0.

To enable or disable the Intel Ethernet Flow Director:

```
# ethtool -K ethX ntuple <on|off>
```

When disabling ntuple filters, all the user programmed filters are flushed from the driver cache and hardware. All needed filters must be re-added when ntuple is re-enabled.

To add a filter that directs packet to queue 2, use -U or -N switch:

```
# ethtool -N ethX flow-type tcp4 src-ip 192.168.10.1 dst-ip \
192.168.10.2 src-port 2000 dst-port 2001 action 2 [loc 1]
```

To see the list of filters currently present:

```
# ethtool <-u|-n> ethX
```

Sideband Perfect Filters

Sideband Perfect Filters are used to direct traffic that matches specified characteristics. They are enabled through ethtool's ntuple interface. To add a new filter use the following command:

```
ethtool -U <device> flow-type <type> src-ip <ip> dst-ip <ip> src-port <port> \
dst-port <port> action <queue>
```

Where:

<device> - the ethernet device to program <type> - can be ip4, tcp4, udp4, or sctp4 <ip> - the IP address to match on <port> - the port number to match on <queue> - the queue to direct traffic towards (-1 discards the matched traffic)

Use the following command to delete a filter:

```
ethtool -U <device> delete <N>
```

Where <N> is the filter id displayed when printing all the active filters, and may also have been specified using "loc <N>" when adding the filter.

The following example matches TCP traffic sent from 192.168.0.1, port 5300, directed to 192.168.0.5, port 80, and sends it to queue 7:

```
ethtool -U enp130s0 flow-type tcp4 src-ip 192.168.0.1 dst-ip 192.168.0.5 \
src-port 5300 dst-port 80 action 7
```

For each flow-type, the programmed filters must all have the same matching input set. For example, issuing the following two commands is acceptable:

```
ethtool -U enp130s0 flow-type ip4 src-ip 192.168.0.1 src-port 5300 action 7
ethtool -U enp130s0 flow-type ip4 src-ip 192.168.0.5 src-port 55 action 10
```

Issuing the next two commands, however, is not acceptable, since the first specifies src-ip and the second specifies dst-ip:

```
ethtool -U enp130s0 flow-type ip4 src-ip 192.168.0.1 src-port 5300 action 7
ethtool -U enp130s0 flow-type ip4 dst-ip 192.168.0.5 src-port 55 action 10
```

The second command will fail with an error. You may program multiple filters with the same fields, using different values, but, on one device, you may not program two TCP4 filters with different matching fields.

Matching on a sub-portion of a field is not supported by the ixgbe driver, thus partial mask fields are not supported.

To create filters that direct traffic to a specific Virtual Function, use the "user-def" parameter. Specify the user-def as a 64 bit value, where the lower 32 bits represents the queue number, while the next 8 bits represent which VF. Note that 0 is the PF, so the VF identifier is offset by 1. For example:

```
... user-def 0x800000002 ...
```

specifies to direct traffic to Virtual Function 7 (8 minus 1) into queue 2 of that VF.

Note that these filters will not break internal routing rules, and will not route traffic that otherwise would not have been sent to the specified Virtual Function.

Jumbo Frames

Jumbo Frames support is enabled by changing the Maximum Transmission Unit (MTU) to a value larger than the default value of 1500.

Use the ifconfig command to increase the MTU size. For example, enter the following where <x> is the interface number:

```
ifconfig eth<x> mtu 9000 up
```

Alternatively, you can use the ip command as follows:

```
ip link set mtu 9000 dev eth<x>
ip link set up dev eth<x>
```

This setting is not saved across reboots. The setting change can be made permanent by adding 'MTU=9000' to the file:

```
/etc/sysconfig/network-scripts/ifcfg-eth<x> // for RHEL
/etc/sysconfig/network/<config_file> // for SLES
```

NOTE: The maximum MTU setting for Jumbo Frames is 9710. This value coincides with the maximum Jumbo Frames size of 9728 bytes.

NOTE: This driver will attempt to use multiple page sized buffers to receive each jumbo packet. This should help to avoid buffer starvation issues when allocating receive packets.

NOTE: For 82599-based network connections, if you are enabling jumbo frames in a virtual function (VF), jumbo frames must first be enabled in the physical function (PF). The VF MTU setting cannot be larger than the PF MTU.

NBASE-T Support

The ixgbe driver supports NBASE-T on some devices. However, the advertisement of NBASE-T speeds is suppressed by default, to accommodate broken network switches which cannot cope with advertised NBASE-T speeds. Use the ethtool command to enable advertising NBASE-T speeds on devices which support it:

```
ethtool -s eth? advertise 0x1800000001028
```

On Linux systems with INTERFACES(5), this can be specified as a pre-up command in /etc/network/interfaces so that the interface is always brought up with NBASE-T support, e.g.:

```
iface eth? inet dhcp
     pre-up ethtool -s eth? advertise 0x1800000001028 || true
```

Generic Receive Offload, aka GRO

The driver supports the in-kernel software implementation of GRO. GRO has shown that by coalescing Rx traffic into larger chunks of data, CPU utilization can be significantly reduced when under large Rx load. GRO is an evolution of the previously-used LRO interface. GRO is able to coalesce other protocols besides TCP. It's also safe to use with configurations that are problematic for LRO, namely bridging and iSCSI.

Data Center Bridging (DCB)

NOTE: The kernel assumes that TC0 is available, and will disable Priority Flow Control (PFC) on the device if TC0 is not available. To fix this, ensure TC0 is enabled when setting up DCB on your switch.

DCB is a configuration Quality of Service implementation in hardware. It uses the VLAN priority tag (802.1p) to filter traffic. That means that there are 8 different priorities that traffic can be filtered into. It also enables priority flow control (802.1Qbb) which can limit or eliminate the number of dropped packets during network stress. Bandwidth can be allocated to each of these priorities, which is enforced at the hardware level (802.1Qaz).

Adapter firmware implements LLDP and DCBX protocol agents as per 802.1AB and 802.1Qaz respectively. The firmware based DCBX agent runs in willing mode only and can accept settings from a DCBX capable peer. Software configuration of DCBX parameters via dcbtool/lldptool are not supported.

The ixgbe driver implements the DCB netlink interface layer to allow user-space to communicate with the driver and query DCB configuration for the port.

ethtool

The driver utilizes the ethtool interface for driver configuration and diagnostics, as well as displaying statistical information. The latest ethtool version is required for this functionality. Download it at: https://www.kernel.org/pub/software/network/ethtool/

FCoE

FCoE

The ixgbe driver supports Fiber Channel over Ethernet (FCoE) and Data Center Bridging (DCB). This code has no default effect on the regular driver operation. Configuring DCB and FCoE is outside the scope of this README. Refer to http://www.open-fcoe.org/ for FCoE project information and contact ixgbe-eedc@lists.sourceforge.net for DCB information.

MAC and VLAN anti-spoofing feature

When a malicious driver attempts to send a spoofed packet, it is dropped by the hardware and not transmitted.

An interrupt is sent to the PF driver notifying it of the spoof attempt. When a spoofed packet is detected, the PF driver will send the following message to the system log (displayed by the "dmesg" command):

```
ixgbe ethX: ixgbe_spoof_check: n spoofed packets detected
```

where "x" is the PF interface number; and "n" is number of spoofed packets. NOTE: This feature can be disabled for a specific Virtual Function (VF):

```
ip link set <pf dev> vf <vf id> spoofchk {off|on}
```

IPsec Offload

The ixgbe driver supports IPsec Hardware Offload. When creating Security Associations with "ip xfrm ..." the 'offload' tag option can be used to register the IPsec SA with the driver in order to get higher throughput in the secure communications.

The offload is also supported for ixgbe's VFs, but the VF must be set as 'trusted' and the support must be enabled with:

```
ethtool --set-priv-flags eth<x> vf-ipsec on
ip link set eth<x> vf <y> trust on
```

Known Issues/Troubleshooting

Enabling SR-IOV in a 64-bit Microsoft Windows Server 2012/R2 guest OS

Linux KVM Hypervisor/VMM supports direct assignment of a PCIe device to a VM. This includes traditional PCIe devices, as well as SR-IOV-capable devices based on the Intel Ethernet Controller XL710.

Support

For general information, go to the Intel support website at:

https://www.intel.com/support/

or the Intel Wired Networking project hosted by Sourceforge at:

https://sourceforge.net/projects/e1000

If an issue is identified with the released source code on a supported kernel with a supported adapter, email the specific information related to the issue to e1000-devel@lists.sf.net.