Open Source Used In C900 1.0

Cisco Systems, Inc.
www.cisco.com

Cisco has more than 200 offices worldwide.
Addresses, phone numbers, and fax numbers
are listed on the Cisco website at
www.cisco.com/go/offices.

Text Part Number: 78EE117C99-173217071

This document contains licenses and notices for open source software used in this product. With respect to the free/open source software listed in this document, if you have any questions or wish to receive a copy of any source code to which you may be entitled under the applicable free/open source license(s) (such as the GNU Lesser/General Public License), please contact us at external-opensource-requests@cisco.com.

In your requests please include the following reference number 78EE117C99-173217071

Contents

1.1 freeBSD Usb xHCI HCD 11.2

1.1.1 Available under license :

```
/* $FreeBSD: releng/11.1/sys/dev/usb/usb_util.h 227701 2011-11-19 10:11:50Z hselasky $ */
/*-
* Copyright (c) 2008 Hans Petter Selasky. All rights reserved.
*
* Redistribution and use in source and binary forms, with or without
* modification, are permitted provided that the following conditions
* are met:
* 1. Redistributions of source code must retain the above copyright
*    notice, this list of conditions and the following disclaimer.
* 2. Redistributions in binary form must reproduce the above copyright
*    notice, this list of conditions and the following disclaimer in the
*    documentation and/or other materials provided with the distribution.
*
* THIS SOFTWARE IS PROVIDED BY THE AUTHOR AND CONTRIBUTORS ``AS IS'' AND
* ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE
* IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE
* ARE DISCLAIMED.  IN NO EVENT SHALL THE AUTHOR OR CONTRIBUTORS BE LIABLE
* FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL
* DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS
* OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION)
* HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT
* LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY
* OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF
```

```
* SUCH DAMAGE.
*/

#ifndef _USB_UTIL_H_
#define _USB_UTIL_H_

uint8_t usb_make_str_desc(void *ptr, uint16_t max_len, const char *s);
void usb_printbcd(char *p, uint16_t p_len, uint16_t bcd);
void usb_trim_spaces(char *p);

#endif     /* _USB_UTIL_H_ */
```

1.2 FreeBSD-ixgbe ix-3.2.17

1.2.1 Available under license :

```
/******************************************************************************
```

Copyright (c) 2001-2017, Intel Corporation
All rights reserved.

Redistribution and use in source and binary forms, with or without
modification, are permitted provided that the following conditions are met:

1. Redistributions of source code must retain the above copyright notice,
   this list of conditions and the following disclaimer.

2. Redistributions in binary form must reproduce the above copyright
   notice, this list of conditions and the following disclaimer in the
   documentation and/or other materials provided with the distribution.

3. Neither the name of the Intel Corporation nor the names of its
   contributors may be used to endorse or promote products derived from
   this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS"
AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE
IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE
ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT OWNER OR CONTRIBUTORS BE
LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR
CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF
SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS
INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN
CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE)
ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE
POSSIBILITY OF SUCH DAMAGE.

```
******************************************************************************/
```

```
/*$FreeBSD$*/

#ifndef _IXGBE_X550_H_
#define _IXGBE_X550_H_

#include "ixgbe_type.h"

s32 ixgbe_dmac_config_X550(struct ixgbe_hw *hw);
s32 ixgbe_dmac_config_tcs_X550(struct ixgbe_hw *hw);
s32 ixgbe_dmac_update_tcs_X550(struct ixgbe_hw *hw);

s32 ixgbe_get_bus_info_X550em(struct ixgbe_hw *hw);
s32 ixgbe_init_eeprom_params_X550(struct ixgbe_hw *hw);
s32 ixgbe_update_eeprom_checksum_X550(struct ixgbe_hw *hw);
s32 ixgbe_calc_eeprom_checksum_X550(struct ixgbe_hw *hw);
s32 ixgbe_calc_checksum_X550(struct ixgbe_hw *hw, u16 *buffer, u32 buffer_size);
s32 ixgbe_validate_eeprom_checksum_X550(struct ixgbe_hw *hw, u16 *checksum_val);
s32 ixgbe_update_flash_X550(struct ixgbe_hw *hw);
s32 ixgbe_write_ee_hostif_buffer_X550(struct ixgbe_hw *hw,
         u16 offset, u16 words, u16 *data);
s32 ixgbe_write_ee_hostif_X550(struct ixgbe_hw *hw, u16 offset,
         u16 data);
s32 ixgbe_read_ee_hostif_buffer_X550(struct ixgbe_hw *hw,
        u16 offset, u16 words, u16 *data);
s32 ixgbe_read_ee_hostif_X550(struct ixgbe_hw *hw, u16 offset,
u16    *data);
s32 ixgbe_write_ee_hostif_data_X550(struct ixgbe_hw *hw, u16 offset,
       u16 data);
void ixgbe_set_source_address_pruning_X550(struct ixgbe_hw *hw, bool enable,
       unsigned int pool);
void ixgbe_set_ethertype_anti_spoofing_X550(struct ixgbe_hw *hw,
        bool enable, int vf);
s32 ixgbe_write_iosf_sb_reg_x550(struct ixgbe_hw *hw, u32 reg_addr,
    u32 device_type, u32 data);
s32 ixgbe_read_iosf_sb_reg_x550(struct ixgbe_hw *hw, u32 reg_addr,
 u32 device_type, u32 *data);
s32 ixgbe_set_fw_drv_ver_x550(struct ixgbe_hw *hw, u8 maj, u8 min,
        u8 build, u8 ver, u16 len, const char *str);
s32 ixgbe_get_phy_token(struct ixgbe_hw *);
s32 ixgbe_put_phy_token(struct ixgbe_hw *);
s32 ixgbe_write_iosf_sb_reg_x550a(struct ixgbe_hw *hw, u32 reg_addr,
 u32 device_type, u32 data);
s32 ixgbe_read_iosf_sb_reg_x550a(struct ixgbe_hw *hw, u32 reg_addr,
 u32 device_type, u32 *data);
void ixgbe_disable_mdd_X550(struct ixgbe_hw *hw);
void ixgbe_enable_mdd_X550(struct ixgbe_hw *hw);
void ixgbe_mdd_event_X550(struct ixgbe_hw *hw, u32 *vf_bitmap);
void ixgbe_restore_mdd_vf_X550(struct ixgbe_hw *hw, u32 vf);
```

```
enum ixgbe_media_type ixgbe_get_media_type_X550em(struct ixgbe_hw *hw);
s32 ixgbe_setup_sfp_modules_X550em(struct ixgbe_hw *hw);
s32 ixgbe_get_link_capabilities_X550em(struct ixgbe_hw *hw,
          ixgbe_link_speed *speed, bool *autoneg);
void ixgbe_init_mac_link_ops_X550em(struct ixgbe_hw *hw);
s32 ixgbe_reset_hw_X550em(struct ixgbe_hw *hw);
s32 ixgbe_init_phy_ops_X550em(struct ixgbe_hw *hw);
s32 ixgbe_setup_kr_x550em(struct ixgbe_hw *hw);
s32 ixgbe_init_ext_t_x550em(struct ixgbe_hw *hw);
s32 ixgbe_setup_internal_phy_t_x550em(struct ixgbe_hw *hw);
s32 ixgbe_setup_phy_loopback_x550em(struct ixgbe_hw *hw);
u64 ixgbe_get_supported_physical_layer_X550em(struct ixgbe_hw *hw);
void ixgbe_disable_rx_x550(struct ixgbe_hw *hw);
s32 ixgbe_get_lcd_t_x550em(struct ixgbe_hw *hw, ixgbe_link_speed *lcd_speed);
s32 ixgbe_enter_lplu_t_x550em(struct ixgbe_hw *hw);
s32 ixgbe_acquire_swfw_sync_X550em(struct ixgbe_hw *hw, u32 mask);
void ixgbe_release_swfw_sync_X550em(struct ixgbe_hw *hw, u32 mask);
s32 ixgbe_setup_fc_X550em(struct ixgbe_hw *hw);
s32 ixgbe_setup_mac_link_sfp_x550em(struct ixgbe_hw *hw,
       ixgbe_link_speed speed,
       bool autoneg_wait_to_complete);
s32 ixgbe_setup_mac_link_sfp_x550a(struct ixgbe_hw *hw,
       ixgbe_link_speed speed,
       bool autoneg_wait_to_complete);
s32 ixgbe_read_phy_reg_x550a(struct ixgbe_hw *hw, u32 reg_addr,
        u32 device_type, u16 *phy_data);
s32 ixgbe_write_phy_reg_x550a(struct ixgbe_hw *hw, u32 reg_addr,
   u32 device_type, u16 phy_data);
s32 ixgbe_setup_fc_fiber_x550em_a(struct ixgbe_hw *hw);
s32 ixgbe_setup_fc_backplane_x550em_a(struct ixgbe_hw *hw);
s32 ixgbe_setup_fc_sgmii_x550em_a(struct ixgbe_hw *hw);
void ixgbe_fc_autoneg_fiber_x550em_a(struct ixgbe_hw *hw);
void ixgbe_fc_autoneg_backplane_x550em_a(struct ixgbe_hw *hw);
void ixgbe_fc_autoneg_sgmii_x550em_a(struct ixgbe_hw *hw);
s32 ixgbe_handle_lasi_ext_t_x550em(struct ixgbe_hw *hw);
s32 ixgbe_setup_mac_link_t_X550em(struct ixgbe_hw *hw,
     ixgbe_link_speed speed,
     bool autoneg_wait_to_complete);
s32 ixgbe_check_link_t_X550em(struct ixgbe_hw *hw, ixgbe_link_speed *speed,
       bool *link_up, bool link_up_wait_to_complete);
s32 ixgbe_reset_phy_t_X550em(struct ixgbe_hw *hw);
s32 ixgbe_identify_sfp_module_X550em(struct ixgbe_hw *hw);
s32 ixgbe_led_on_t_X550em(struct ixgbe_hw *hw, u32 led_idx);
s32 ixgbe_led_off_t_X550em(struct ixgbe_hw *hw, u32 led_idx);
#endif /* _IXGBE_X550_H_ */
FreeBSD Driver for Intel(R) Ethernet 10 Gigabit PCI Express Server Adapters
==========================================================================
```

February 04, 2015

Contents
========

- Overview
- Supported Adapters
- Building and Installation
- Additional Configurations and Tuning
- Known Limitations

Overview

This file describes the FreeBSD* driver for Intel(R) Ethernet. This driver has
been developed for use with all community-supported versions of FreeBSD.

For questions related to hardware requirements, refer to the documentation
supplied with your Intel Ethernet Adapter. All hardware requirements listed
apply to use with FreeBSD.

NOTE: Devices based on the Intel(R) Ethernet Controller X552 and Intel(R)
Ethernet Controller X553 do not support the following features:
 * Low Latency Interrupts (LLI)

Identifying Your Adapter

The driver in this release is compatible with devices based on the following:
 * Intel(R) Ethernet Controller 82598
 * Intel(R) Ethernet Controller 82599
 * 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:
http://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.

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.

The VF Driver
==========
In the FreeBSD guest the ixgbe driver would be loaded and will function using
the VF device assigned to it.

The VF driver provides most of the same functionality as the CORE driver, but
is actually a slave to the Host, access to many controls is accomplished by a
request to the Host via what is called the "Admin queue". These are startup and
initialization events, however, once in operation the device is self-contained
and should achieve near native performance.

Some notable limitations of the VF environment: for security reasons the driver
is never permitted to be promiscuous, therefore a tcpdump will not behave the
same with the interface. Second, media info is not available from the PF, so it
will always appear as auto.

Building and Installation

NOTE: This driver package is to be used only as a standalone archive and the
user should not attempt to incorporate it into the kernel source tree.

In the instructions below, x.x.x is the driver version as indicated in the name of the driver tar file.

1. Move the base driver tar file to the directory of your choice. For example, use /home/username/ixgbe or /usr/local/src/ixgbe.

2. Untar/unzip the archive:

```
tar xzf ixgbe-x.x.x.tar.gz
```

This will create the ixgbe-x.x.x directory.

3. To install man page:

```
cd ixgbe-x.x.x
gzip -c ixgbe.4 > /usr/share/man/man4/ixgbe.4.gz
```

4. To load the driver onto a running system:

```
cd ixgbe-x.x.x/src
make load
```

5. To assign an IP address to the interface, enter the following:

```
ifconfig ixgbe<interface_num> <IP_address>
```

6. Verify that the interface works. Enter the following, where <IP_address> is the IP address for another machine on the same subnet as the interface that is being tested:

```
ping <IP_address>
```

7. If you want the driver to load automatically when the system is booted:

```
cd ixgbe-x.x.x/src
make
make install
```

Edit /boot/loader.conf, and add the following line:

```
if_ixgbe_load="YES"
```

Edit /etc/rc.conf, and create the appropriate ifconfig_ixgbe<interface_num> entry:

```
ifconfig_ixgbe<interface_num>="<ifconfig_settings>"
```

Example usage:

```
ifconfig_ixgbe0="inet 192.168.10.1 netmask 255.255.255.0"
```

NOTE: For assistance, see the ifconfig man page.

Speed and Duplex Configuration

In addressing speed and duplex configuration issues, you need to distinguish between copper-based adapters and fiber-based adapters.

In the default mode, an Intel(R) Ethernet Network Adapter using copper connections will attempt to auto-negotiate with its link partner to determine the best setting. If the adapter cannot establish link with the link partner using auto-negotiation, you may need to manually configure the adapter and link partner to identical settings to establish link and pass packets. This should only be needed when attempting to link with an older switch that does not support auto-negotiation or one that has been forced to a specific speed or duplex mode. Your link partner must match the setting you choose. 1 Gbps speeds and higher cannot be forced. Use the autonegotiation advertising setting to manually set devices for 1 Gbps and higher.

Caution: Only experienced network administrators should force speed and duplex or change autonegotiation advertising manually. The settings at the switch must always match the adapter settings. Adapter performance may suffer or your adapter may not operate if you configure the adapter differently from your switch.

An Intel(R) Ethernet Network Adapter using fiber-based connections, however, will not attempt to auto-negotiate with its link partner since those adapters operate only in full duplex and only at their native speed.

By default, the adapter auto-negotiates the speed and duplex of the connection. If there is a specific need, the ifconfig utility can be used to configure the speed and duplex settings on the adapter.

NOTE: For the Intel(R) Ethernet Connection X552 10 GbE SFP+ you must specify the desired speed.

Example usage:
ifconfig emX <IP_address> media 100baseTX mediaopt full-duplex

NOTE: Only use mediaopt to set the driver to full-duplex. If mediaopt is not specified and you are not running at gigabit speed, the driver defaults to half-duplex.

If the interface is currently forced to 100 full duplex, you must use this command to change to half duplex:

ifconfig emX <IP_address> media 100baseTX -mediaopt full-duplex

This driver supports the following media type options:

Media Type Description
---------- -----------
autoselect Enables auto-negotiation for speed and duplex.
10baseT/UTP Sets speed to 10 Mbps. Use the ifconfig mediaopt
  option to select full-duplex mode.
100baseTX Sets speed to 100 Mbps. Use the ifconfig mediaopt
  option to select full-duplex mode.
1000baseTX Sets speed to 1000 Mbps. In this case, the driver
  supports only full-duplex mode.
1000baseSX Sets speed to 1000 Mbps. In this case, the driver
  supports only full-duplex mode.

For more information on the ifconfig utility, see the ifconfig man page.

Configuration and Tuning
========================

Important system configuration changes:

- Change the file /etc/sysctl.conf, and add the line:

 hw.intr_storm_threshold: 0 (the default is 1000)

- Best throughput results are seen with a large MTU; use 16114if possible.
 The default number of descriptors per ring is 1024. Increasing this may
 improve performance, depending on your use case.
- If you have a choice, run on a 64-bit OS rather than a 32-bit OS.

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

To confirm an interface's MTU value, use the ifconfig command.

To confirm the MTU used between two specific devices, use:

  route get <destination_IP_address>

NOTE: The maximum MTU setting for Jumbo Frames is 16114. This value coincides
with the maximum Jumbo Frames size of 16136 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.

VLANS

To create a new VLAN interface:

```
ifconfig <vlan_name> create
```

To associate the VLAN interface with a physical interface and assign a VLAN ID,
IP address, and netmask:

```
ifconfig <vlan_name> <ip_address> netmask <subnet_mask> vlan
  <vlan_id> vlandev <physical_interface>
```

Example:

```
ifconfig vlan10 10.0.0.1 netmask 255.255.255.0 vlan 10 vlandev ixgbe0
```

In this example, all packets will be marked on egress with 802.1Q VLAN tags,
specifying a VLAN ID of 10.

To remove a VLAN interface:

```
ifconfig <vlan_name> destroy
```

Checksum Offload

Checksum offloading supports both TCP and UDP packets and is supported for both
transmit and receive.

Checksum offloading can be enabled or disabled using ifconfig. Both transmit
and receive offloading will be either enabled or disabled together. You cannot
enable/disable one without the other.

To enable checksum offloading:

```
ifconfig ixgbeX rxcsum
```

To disable checksum offloading:

```
ifconfig ixgbeX -rxcsum
```

To confirm the current setting:
 ifconfig ixgbeX

Look for the presence or absence of the following line:
 options=3 <RXCSUM,TXCSUM>

See the ifconfig man page for further information.

TSO

TSO (TCP Segmentation Offload) supports both IPv4 and IPv6. TSO can be disabled and enabled using the ifconfig utility or sysctl.

NOTE: TSO requires Tx checksum, if Tx checksum is disabled, TSO will also be disabled.

To enable/disable TSO in the stack:
 sysctl net.inet.tcp.tso=0 (or 1 to enable it)

Doing this disables/enables TSO in the stack and affects all installed adapters.

To disable BOTH TSO IPv4 and IPv6:
 ifconfig ixgbe<interface_num> -tso

To enable BOTH TSO IPv4 and IPv6:
 ifconfig ixgbe<interface_num> tso

You can also enable/disable IPv4 TSO or IPv6 TSO individually. Simply replace tso|-tso in the above command with tso4 or tso6. For example, to disable TSO IPv4:
 ifconfig ixgbe<interface_num> -tso4

LRO

LRO (Large Receive Offload) may provide rx performance improvement. However, it is incompatible with packet-forwarding workloads. You should carefully evaluate the environment and enable LRO when possible.

To enable:

  ifconfig ixgbe<interface_num> lro

It can be disabled by using:

  ifconfig ixgbe<interface_num> -lro

DMAC

Valid Range: 0, 41-10000
This parameter enables or disables DMA Coalescing feature. Values are in microseconds and set the internal DMA Coalescing internal timer.
DMAC is available on Intel(R) X550 (and later) based adapters.
DMA (Direct Memory Access) allows the network device to move packet data directly to the system's memory, reducing CPU utilization. However, the frequency and random intervals at which packets arrive do not allow the system to enter a lower power state. DMA Coalescing allows the adapter to collect packets before it initiates a DMA event. This may increase network latency but also increases the chances that the system will enter a lower power state.
Turning on DMA Coalescing may save energy.
DMA Coalescing must be enabled across all active ports in order to save platform power.
InterruptThrottleRate (ITR) should be set to dynamic. When ITR=0, DMA Coalescing is automatically disabled.
A whitepaper containing information on how to best configure your platform is available on the Intel website.

Known Issues/Troubleshooting

UDP Stress Test Dropped Packet Issue

Under small packet UDP stress with the ixgbe driver, the system may drop UDP packets due to socket buffers being full. Setting the driver Intel Ethernet Flow Control variables to the minimum may resolve the issue.

Attempting to configure larger MTUs with a large numbers of processors may generate the error message "ix0:could not setup receive structures"

When using the ixgbe driver with RSS autoconfigured based on the number of cores (the default setting) and that number is larger than 4, increase the memory resources allocated for the mbuf pool as follows:

Add to the sysctl.conf file for the system:

kern.ipc.nmbclusters=262144
kern.ipc.nmbjumbop=262144

Lower than expected performance

Some PCIe x8 slots are actually configured as x4 slots. These slots have
insufficient bandwidth for full line rate with dual port and quad port devices.
In addition, if you put a PCIe v3.0-capable adapter into a PCIe v2.x slot, you
cannot get full bandwidth. The driver detects this situation and writes the
following message in the system log:

"PCI-Express bandwidth available for this card is not sufficient for optimal
performance. For optimal performance a x8 PCI-Express slot is required."

If this error occurs, moving your adapter to a true PCIe v3.0 x8 slot will
resolve the issue.

Support

For general information, go to the Intel support website at:
www.intel.com/support/

or the Intel Wired Networking project hosted by Sourceforge at:
http://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 freebsdnic@mailbox.intel.com

License

This program is free software; you can redistribute it and/or modify it under
the terms and conditions of the GNU General Public License, version 2, as
published by the Free Software Foundation.

This program is distributed in the hope it will be useful, but WITHOUT ANY
WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR
A PARTICULAR PURPOSE. See the GNU General Public License for more details.

You should have received a copy of the GNU General Public License along with
this program; if not, write to the Free Software Foundation, Inc., 51 Franklin
St - Fifth Floor, Boston, MA 02110-1301 USA.

The full GNU General Public License is included in this distribution in the
file called "COPYING".