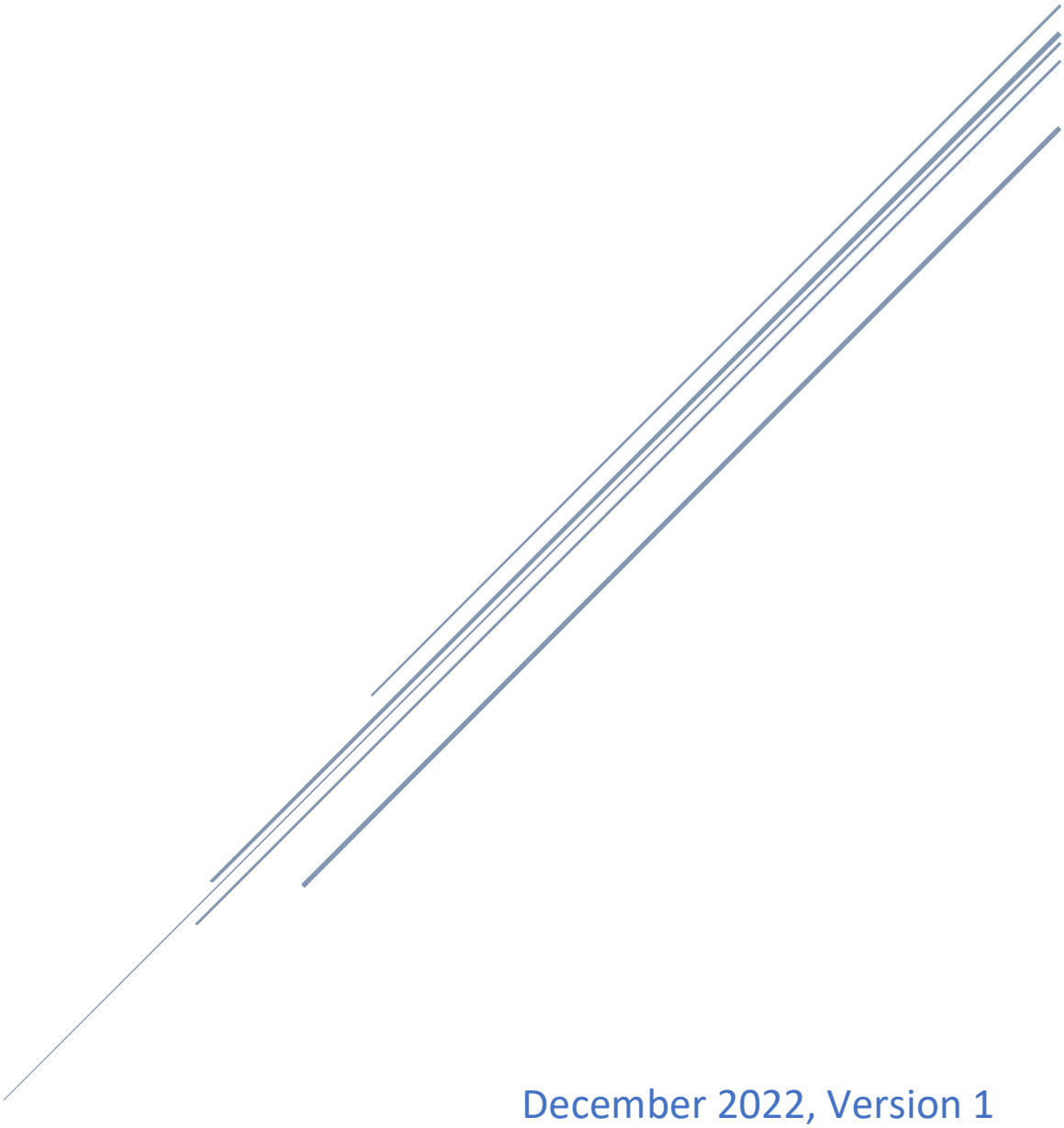
VSI OpenVMS LAN DEVICES, COUNTERS, AND FUNCTIONS (X86)



December 2022, Version 1

Contents

Introduction

This document describes the driver-specific counters and functions maintained by the LAN drivers internally.

The LAN drivers maintain this information that LANCP and the LAN SDA extension can display:

- Device counters, packets and bytes sent and received and error counters for transmit and receive.
- Driver-specific internal counters, device and driver settings and counters maintained internal to the driver. This includes driver messages showing settings and link state changes.
- Error log information.
- Driver tracing, by default mostly errors, state changes, and link events. Selecting which events to trace is done by LANCP command and LAN_FLAGS system parameter settings.
- Bandwidth monitoring. Adjusting bandwidth monitoring settings is done by LANCP command and LAN_FLAGS settings.

The LAN drivers provide additional capabilities, mostly for diagnostic and test purposes. These capabilities are called device-specific functions initiated by LANCP command. You can reset a device, cause various errors, set promiscuous mode and many other things. The selection of functions is driver-specific.

The devices covered in this document follow.

- Intel 1Gb/emulated Ethernet
- Broadcom 1Gb/emulated Ethernet
- Intel 10Gb Ethernet
- KVM Virtio Ethernet (Future)
- VMware VMXNET3 Ethernet (Future)
- Hyper-V VMBUS Ethernet (Future)

A section for each LAN device follows.

Intel 1Gb Ethernet

The Intel 1Gb devices are, via chip designation:

Chip type translation:

Chip name	Chip number
Kenai32	82540EM
Kenai32M	82540EP
Cordova	82544EI
Cordova-R	82544GC
Kenai64M	82545EM
Attla	82545GM
Anvik	82546EB
Anvik II	82546GB
Ophir	82571EB
Zoar	82575EB
Springville	I210
Pearsonville	I211
Powerville	I350

Device type translation:

Supported on X86

Vendor ID	SubIDs	Bus	DeviceName	Chip
15218086	various	PCIe	I350	I350 (UTP)
15338086	various	PCIe	I210	I210 (UTP)
157B8086	various	PCIe	I210	I210 (UTP)
15398086	various	PCIe	I211	I211 (UTP)

Virtual machines

Vendor ID	SubIDs	Bus	DeviceName	Chip
100E8086	001E8086	PCI	VBOX	82540VM
10048086	10048086	PCI	VBOX	82543VM
100F8086	075015AD	PCI	VBOX	82545VM
100E8086	11001AF4	PCIe	KVM	82540VM
100F8086	075015AD	PCI	VMware	82545VM

The driver is SYS$EI1000X.EXE.

Notes:

There are multiple names for each device:

- PCI ID repository (google search for it).

- Vendor names and part numbers (there may be multiple names and part numbers).

- Full name assigned by the driver, including a part number if known, with some effort for consistency. This name should identify the device sufficiently to allow someone to order it and receive what is expected.

  The full name is the "device =" parameter in the entry in SYS$CONFIG.DAT and is the same name assigned by the driver.

  The full name is displayed by LANCP SHOW DEV/INTERNAL, SDA CLUE CONFIG/ADAPTER, and by SEARCH SYS$SYSTEM:SYS$CONFIG.DAT DEVICE,"="/MATCH=AND.

- Short name assigned by the driver for printouts that need a name shorter than the full name (LANCP SHOW CONFIG, LANCP SHOW DEV/INTERNAL).

  The short name is displayed by LANCP SHOW CONFIG and LANCP SHOW DEV/INTERNAL.

- Additional names or aliases are given in the comments before the SYS$CONFIG.DAT entry, for example, this device:

  ```
  HPE 4-port Intel I350 366FLR Adapter (665240-B21) (Gigabit Ethernet)
  ```

  is in SYS$CONFIG.DAT as, where the comments give other names that identify the NIC:

  ```
  ! HP Ethernet 1Gb 4-port 366FLR Adapter (665240-B21)             I350
  ! HPE Ethernet 1Gb 4-port FLR-T I350-T4V2 (665240-B21)           I350
  ! HPE 4-port FLR-T Intel I350-T4V2 (665240-B21)                  I350
  device = "HPE 4-port Intel I350 366FLR Adapter (665240-B21) (Gigabit
  Ethernet)"
  ```

  The PCI Device IDs are stored in the busarray entry. To find for a particular device, do SDA CLUE CONFIG/ADAPTER and EXAMINE busarray address - the Device ID, Vendor ID, SubDevice ID, and SubVendor ID are the first two longwords.

Details for each device follow:

```
-------------------------------------------------------------------------------
----------------------- Intel 82540EM Device ID 100E ------------------------
-------------------------------------------------------------------------------

 Vendor/DevID  SubIDs   Bus     Device  PCIID Repository Name                 In SYS$CONFIG.DAT
    100E8086  001E8086  PCI     82540EM PRO/1000 MT Desktop Adapter                 Yes
       Full name:  Intel i82540 VBOX (Ethernet)
       Short name: i82540 VBOX

    100E8086  11001AF4  PCIe    82540EM QEMU Virtual Machine                        Yes
       Full name:  Intel i82540 KVM (Ethernet)
       Short name: i82540 VBOX

-------------------------------------------------------------------------------
----------------------- Intel 82543GC Device ID 1004 ------------------------
-------------------------------------------------------------------------------

 Vendor/DevID  SubIDs   Bus     Device  PCIID Repository Name                 In SYS$CONFIG.DAT
```

```
    10048086  Catch-all PCI     82543GC PRO/1000 T Server Adapter                     Yes
       Full name:  Intel i82543 VBOX (Ethernet)
       Short name: i82543 VBOX

------------------------------------------------------------------------------
----------------------- Intel 82545EM Device ID 100F -------------------------
------------------------------------------------------------------------------

 Vendor/DevID  SubIDs   Bus     Device  PCIID Repository Name                 In SYS$CONFIG.DAT
    100F8086  Catch-all PCI     82543GC PRO/1000 MT Server Adapter                    Yes
       Full name:  Intel i82545 VBOX/VMware (Ethernet)
       Short name: 82545 VB/VMware

------------------------------------------------------------------------------
-------------------------- Intel I350 Device ID 1521 --------------------------
------------------------------------------------------------------------------

 Vendor/DevID  SubIDs   Bus     Device  PCIID Repository Name                 In SYS$CONFIG.DAT
    15218086  06021028  PCIe    I350    Gigabit 2P I350-t LOM                         Yes
       Full name:  Intel I350-t 2P LOM (Gigabit Ethernet)
       Short name: I350 LOM

    15218086  06931028  PCIe    I350    Gigabit 2P I350-t LOM                         Yes
       Full name:  Intel I350-t 2P LOM (Gigabit Ethernet)
       Short name: I350 LOM

    15218086  06E21028  PCIe    I350    Gigabit 2P I350-t LOM                         Yes
       Full name:  Intel I350-t 2P LOM (Gigabit Ethernet)
       Short name: I350 LOM

    15218086  07571028  PCIe    I350    Gigabit I350-t LOM                            Yes
       Full name:  Intel I350-t LOM (Gigabit Ethernet)
       Short name: I350 LOM

    15218086  075A1028  PCIe    I350    Gigabit I350-t LOM                            Yes
       Full name:  Intel I350-t LOM (Gigabit Ethernet)
       Short name: I350 LOM

    15218086  1f601028  PCIe    I350    Gigabit 4P I350-t rNDC                        No
    15218086  1f621028  PCIe    I350    Gigabit 4P X540/I350 rNDC                     No
    15218086  1fa81028  PCIe    I350    Ethernet 10G 4P X550/I350 rNDC                No
    15218086  1fa91028  PCIe    I350    Ethernet 10G 4P X550 rNDC                     No
    15218086  1faa1028  PCIe    I350    Gigabit 4P X550/I350 rNDC                     No
    15218086  ff9a1028  PCIe    I350    Gigabit 4P X710/I350 rNDC                     No

    15218086  17d1103c  PCIe    I350    Ethernet 1Gb 4-port 366FLR Adapter            Yes
       Full name:  HPE 4-port Intel I350 366FLR Adapter (665240-B21) (Gigabit Ethernet)
       Short name: I350 HPE 366FLR
       Other name: HP Ethernet 1Gb 4-port 366FLR Adapter (665240-B21)
       Other name: HPE Ethernet 1Gb 4-port FLR-T I350-T4V2 (665240-B21)
       Other name: HPE 4-port FLR-T Intel I350-T4V2 (665240-B21)

    15218086  2003103c  PCIe    I350    Ethernet 1Gb 2-port 367i Adapter              No
    15218086  2226103c  PCIe    I350    Ethernet 1Gb 1-port 364i Adapter              No

    15218086  337f103c  PCIe    I350    Ethernet 1Gb 2-port 361i Adapter              Yes
       Full name:  HP 2-port Intel I350 361i Adapter (Gigabit Ethernet)
       Short name: I350 HP 361i

    15218086  3380103c  PCIe    I350    Ethernet 1Gb 4-port 366i Adapter              Yes
       Full name:  HP 4-port Intel I350 366i Adapter (Gigabit Ethernet)
       Short name: I350 HP 366i

    15218086  339e103c  PCIe    I350    Ethernet 1Gb 2-port 361T Adapter              Yes
       Full name:  HPE 2-port Intel I350 361T Adapter (652497-B21) (Gigabit Ethernet)
       Short name: I350 HPE 361T
       Other name: HP Ethernet 1Gb 2-port 361T Adapter (652497-B21)
       Other name: HPE Ethernet 1Gb 2-port BASE-T I350-T2V2 (HP 361T)(652497-B21)
       Other name: HPE 2-port BASE-T Intel I350-T2V2 (HP 361T) (652497-B21)
```

```
15218086  8157103c  PCIe    I350    Ethernet 1Gb 4-port 366T Adapter              Yes
   Full name:  HPE 4-port Intel I350 366T Adapter (811546-B21) (Gigabit Ethernet)
   Short name: I350 HPE 366T
   Other name: HP Ethernet 1Gb 4-port 366T Adapter (811546-B21)
   Other name: HPE Ethernet 1Gb 4-port BASE-T I350-T4V2 (811546-B21)
   Other name: HPE 4-port BASE-T Intel I350-T4V2 (811546-B21)

15218086  7b16108e  PCIe    I350    Quad Port GbE PCIe 2.0 ExpressModule, UTP     No
15218086  7b18108e  PCIe    I350    Quad Port GbE PCIe 2.0 Low Prof Adaptr, UTP   No
15218086  76481093  PCIe    I350    PCIe-8237R Ethernet Adapter                   No
15218086  76491093  PCIe    I350    PCIe-8236 Ethernet Adapter                    No
15218086  76b11093  PCIe    I350    PCIe-8237R-S Ethernet Adapter                 No
15218086  775b1093  PCIe    I350    PCIe-8237 Ethernet Adapter                    No
15218086  802a10a9  PCIe    I350    UV2-BaseIO dual-port GbE                      No

15218086  023e1137  PCIe    I350    1GigE I350 LOM                                Yes
   Full name:  Intel I350 LOM (Gigabit Ethernet)
   Short name: I350 LOM

15218086  000015d9  PCIe    I350    AOC-SGP-i4                                    No
15218086  065215d9  PCIe    I350    Dual Port i350 GbE MicroLP [AOC-CGP-i2]       No
15218086  107417aa  PCIe    I350    ThinkServer I350-T4 AnyFabric                 No

15218086  400517aa  PCIe    I350    I350 Gigabit Network Connection               Yes
   Full name:  Intel I350 Network Connection (Gigabit Ethernet)
   Short name: I350

15218086  0c0718d4  PCIe    I350    I350 1Gb 2-port RJ45 OCP Mezz MOP41-I-1GT2    No
15218086  1005193d  PCIe    I350    360T-B                                        No
15218086  1007193d  PCIe    I350    360T-L                                        No
15218086  1080193d  PCIe    I350    NIC-ETH360T-3S-4P                             No
15218086  001d1bd4  PCIe    I350    1G base-T QP EP014Ti1 Adapter                 No
15218086  00351bd4  PCIe    I350    1G base-T QP EP014Ti1 Adapter                 No
15218086  00661bd4  PCIe    I350    F014I350                                      No

15218086  00018086  PCIe    I350    Ethernet Server Adapter I350-T4               Yes
   Full name:  Server Adapter Intel I350-T4 (Gigabit Ethernet)
   Short name: I350-T4

15218086  00028086  PCIe    I350    Ethernet Server Adapter I350-T2               Yes
   Full name:  Server Adapter Intel I350-T2 (Gigabit Ethernet)
   Short name: I350-T2

15218086  00038086  PCIe    I350    Ethernet Network Adapter I350-T4 for OCP NIC  Yes
   Full name:  Network Adapter Intel I350-T4 for OCP NIC 3.0 (Gigabit Ethernet)
   Short name: I350-T4

15218086  00a18086  PCIe    I350    Ethernet Server Adapter I350-T4               Yes
   Full name:  Server Adapter Intel I350-T4 (Gigabit Ethernet)
   Short name: I350-T4

15218086  00a28086  PCIe    I350    Ethernet Server Adapter I350-T2               Yes
   Full name:  Server Adapter Intel I350-T2 (Gigabit Ethernet)
   Short name: I350-T2

15218086  00a38086  PCIe    I350    Ethernet Network Adapter I350-T4 for OCP NIC  Yes
   Full name:  Network Adapter Intel I350-T4 for OCP NIC 3.0 (Gigabit Ethernet)
   Short name: I350-T4

15218086  00aa8086  PCIe    I350    Ethernet Network Adapter I350-T4 for OCP NIC  Yes
   Full name:  Network Adapter Intel I350-T4 for OCP NIC 3.0 (Gigabit Ethernet)
   Short name: I350-T4

15218086  40178086  PCIe    I350    Ethernet Network Adapter I350-T4 for OCP NIC  No

15218086  50018086  PCIe    I350    Ethernet Server Adapter I350-T4               Yes
   Full name:  Server Adapter Intel I350-T4 (Gigabit Ethernet)
   Short name: I350-T4
```

```
    15218086  50028086  PCIe    I350    Ethernet Server Adapter I350-T2              Yes
       Full name:  Server Adapter Intel I350-T2 (Gigabit Ethernet)
       Short name: I350-T2

    15218086  50038086  PCIe    I350    Ethernet 1G 4P I350-t OCP                    Yes
       Full name:  4P Intel I350-t OCP (Gigabit Ethernet)
       Short name: I350

    15218086  Catch-all PCIe    I350    n/a                                        Yes
       Full name:  Intel I350 (Gigabit Ethernet)
       Short name: I350

-------------------------------------------------------------------------------
------------------------- Intel I210 Device ID 1533 ---------------------------
-------------------------------------------------------------------------------

 Vendor/DevID  SubIDs   Bus     Device  PCIID Repository Name                 In SYS$CONFIG.DAT
    15338086  0b351028  PCIe    I210    Gigabit Network Connection                  No
    15338086  0003103c  PCIe    I210    Ethernet I210-T1 GbE NIC                   No
    15338086  01801059  PCIe    I210    RD10019 1GbE interface                      No
    15338086  77061093  PCIe    I210    Compact Vision System Ethernet Adapter      No
    15338086  802c10a9  PCIe    I210    UV300 BaseIO single-port GbE                No
    15338086  802d10a9  PCIe    I210    UV3000 BaseIO GbE Network                   No
    15338086  110017aa  PCIe    I210    ThinkServer Ethernet Server Adapter         No
    15338086  00018086  PCIe    I210    Ethernet Server Adapter I210-T1             No
    15338086  00028086  PCIe    I210    Ethernet Server Adapter I210-T1             No
    15338086  Catch-all PCIe    I210    Intel I210                                  No
       Full name:  Intel I210 (Gigabit Ethernet)
       Short name: I210

-------------------------------------------------------------------------------
------------------------- Intel I210 Device ID 157B ---------------------------
-------------------------------------------------------------------------------

 Vendor/DevID  SubIDs   Bus     Device  PCIID Repository Name                 In SYS$CONFIG.DAT
    157B8086  Catch-all PCIe    I210    Intel I210                                  No
       Full name:  Intel I210 (Gigabit Ethernet)
       Short name: I210

-------------------------------------------------------------------------------
------------------------- Intel I211 Device ID 1539 ---------------------------
-------------------------------------------------------------------------------

 Vendor/DevID  SubIDs   Bus     Device  PCIID Repository Name                 In SYS$CONFIG.DAT
    15398086  Catch-all PCIe    I211    Intel I211                                  No
       Full name:  Intel I211 (Gigabit Ethernet)
       Short name: I211
```

Intel 1Gb Internal Counters

The LANCP command SHOW DEVICE/INTERNAL_COUNTERS EWA displays the entire set of internal counters maintained by the Intel 1Gb driver. Some counters are special debug counters. These are not displayed unless the additional qualifier /DEBUG is specified. Counters that are zero are not displayed unless the additional qualifier /ZERO is specified.

The LAN$SDA SDA extension also displays the complete set of internal counters with the command LAN INTERNAL/DEVICE=EIA.

The definition of these counters may change from one driver version to the next.

The counters are:

Device name (full)
Device name (short)

Name of the Intel 1gb device.

Driver timestamp

Compilation timestamp of the driver.

Driver version

Driver version numbered 1...n that usually is identical to the x-n ID displayed by an ANALYZE/IMAGE of the driver image. It includes variant information, if any. The full driver version includes the target OpenVMS release and is displayed by SDA LAN/DEV=EIA in the quadword driver version field.

Device revision

Hardware revision level of the chip.

Device interrupts

Number of times the driver interrupt service routine was called.

Link transitions

Number of link transitions seen by the driver.

Interrupt link state changes

Number of link state changes reported in the interrupt control register.

Link checks

Number of times the driver checked the current link status.

Link setup delay (usecs)

Total time (in microseconds) waiting for the device during setup of the link.

Maximum link setup delay (usecs)

Maximum time (in microseconds) waiting for the device during setup of the link.

VCI port usable/unusable events reported

Number of VCI user port events reported.

Device resets

Number of times the device was reset.

CSR Base Address

PCI BAR0 CSR base address.

Unit inits

Number of unit initializations executed; since unit initialization is only executed once, this counter will be one.

User start/change/stop requests

Number of user startup and shutdown requests processed by the driver, generally one or two when a user starts up, and one when a user stops.

Transmits queued

Number of transmit requests queued because the link was not available or because too many transmit requests were already outstanding.

Transmit timeouts

Number of times the driver has timed out a transmit and has reset the device and completed outstanding I/O with error status.

Transmit timeouts (averted)

Number of transmit timeouts averted by a final check for transmit completion.

Transmit chained (buffer address)

Number of chained transmits where the chaining is via SVA buffer address.

Transmit chained (svapte_sva)

Number of chained transmits where the chaining is via a buffer extent.

Transmit errors (too few segments)

Number of transmit requests completed with error status (SS$_INCSEGTRA) because the application did not specify the transmit buffer completely.

Transmit informationals (zero byte segments)

Number of transmit chain segments which specified a zero-byte length segment.

Transmit copies (too many segments)

Number of transmit requests that exceeded the maximum number of chain segments that the driver can handle; the driver then copied some of them to a temporary buffer so that it could transmit the packet.

Transmit copy failures

Number of transmit failures caused by too few chain segments when coalescing a request to reduce the number of segments needed.

Jumbo transmits issued

Number of transmit requests with a packet length exceeding 1514 bytes (excluding CRC).

Jumbo receives issued

Number of jumbo receive buffers allocated and given to the device.

Chained receives completed

Number of receives (jumbo packets) completed by the driver that span multiple receive buffers.

Receives discarded (bad frame)

Number of receive packets discarded by the driver due to receive error. By default, the chip does not deliver damaged receive packets to the driver, but can be asked to do so by a device-specific function.

Soft errors

Number of times errors were recovered in the driver by resetting and reinitializing the device, without notification of user applications.

Rescheduled forks (too long in fork)

Number of times that a rescheduled fork was done. In transmit and receive processing, the driver limits the amount of time spent in the fork process before rescheduling.

PHY register read/write errors

Number of errors reading or writing a PHY register.

Standard packet size (bytes)

This is the Ethernet packet size (1518 bytes).

Jumbo packet size (bytes)

This is the jumbo packet size (9018 bytes).

Requested link settings

Speed and duplex mode requested by a user.

Current link settings

Current link state.

Driver flags

Driver flags.

Driver state

Current driver state, one of the following:

0 - Driver state is undefined
1 - Driver is initializing the device
2 - Driver is running but the link is down, completing transmits with error status
4 - Driver is running but the link is down, queueing transmits for now
8 - Driver is running and the link is up, processing transmits, and receives
16 - Device is not usable

LAN_FLAGS system parameter

Current value of the LAN_FLAGS system parameter.

Packet buffer allocation (initial value)

Amount of the chip packet buffer allocated to transmit buffers vs receive, initial value.

Packet buffer allocation (current value)

Amount of the chip packet buffer allocated to transmit buffers vs receive, current value.

Interrupt delay value

Minimum delay between interrupts, in units of 256 nanoseconds.

Transmit interrupt delay (usec)

Amount of delay after transmit completion that an interrupt is generated.

Receive interrupt delay (usec)

Amount of delay after receive completion that an interrupt is generated.

Receive rings VA

System VA of the start of the receive rings.

Transmit rings VA

System VA of the start of the transmit rings.

LSB size

Total size of the LAN Station Block (LSB) structure.

Interrupt mode

Unknown, IOSAPIC, MSI, MSIX, or IOAPIC.

Transmit time limit

Transmit time limit in seconds after which a timeout is declared.

Timer routine interval

Resolution of the transmit timer (the real timeout is Limit + Interval).

Time Stamps

Current uptime

Current system uptime.

Last reset

Last time the device was reset.

Last link state change interrupt

Last time a link state change interrupt was fielded.

Previous link state change interrupt

Previous time a link state change interrupt was fielded.

Last link up

Last time a link up transition occurred.

Last link down

Last time a link down transition occurred.

Total link uptime

Total time the link has been up.

Total link downtime

Total time the link has been down.

Time of last uplink period
Time of last uplink period-1
Time of last uplink period-2
Time of last uplink period-3
Time of last uplink period-4
Time of last uplink period-5
Time of last uplink period-6
Time of last uplink period-7
Time of last uplink period-8
Time of last uplink period-9
Time of last uplink period-10
Time of last uplink period-11
Time of last uplink period-12
Time of last uplink period-13
Time of last uplink period-14

History of the last 15 link uptimes - the length of each uptime period.

Last transmit timeout

Time of the last transmit timeout.

Last transmit timeout (averted)

Time last averted a transmit timeout.

Last soft error

Time of the last soft error.

Last CRC error

Time of the last CRC error.

Last receive error

Time of the last receive error.

Last receive error packet header

First 24 bytes of the last packet received with error status.

Last unrecognized unicast packet

Time of the last unrecognized unicast packet received.

Last unrecognized unicast packet header

First 24 bytes of the last unicast packet received that was discarded because receive address filtering did not result in a matching user.

Last unrecognized multicast packet

Time of the last unrecognized multicast packet received.

Last unrecognized multicast packet header

First 24 bytes of the last multicast packet received that was discarded because receive address filtering did not result in a matching user.

Device Registers (read/wrote)

Consists of a list of the contents significant chip registers.

PCI Config Registers

- CFID register
- CFCR register
- CFRV register
- CFLT register
- PCI_ADDRESS0 register
- PCI_ADDRESS1 register
- SUB_VNDR / SUB_ID register
- CACHE_LINE_SIZE register

Receive Buffers

Minimum buffers requested

Minimum number of receive buffers as set by default or by management request (setmode or LANCP command). This is used to determine the "Current minimum limit".

Maximum buffers requested

Maximum number of receive buffers as set by default or by management request (setmode or LANCP command). This is used to determine the "Current maximum limit".

Current minimum limit

Minimum number of receive buffers as determined by the driver and by management request, "Minimum buffers requested". The driver will not allow the number of receive buffers in its receive queues to drop below this value. This does not include receive buffers handed up to applications (and not yet returned).

Current maximum limit

Maximum number of receive buffers as determined by the driver and by management request, "Maximum buffers requested". The driver will deallocate receive buffers until the number allocated does not exceed than this maximum. When applications return receive buffers, the number may exceed this maximum at which point they are deallocated. Note that the driver allows this maximum to be exceeded somewhat to limit allocation and deallocation activity. See "Target number of buffers maximum" below.

Current number of buffers

Current number of receive buffers owned by the driver.

Target number of buffers

Desired number of receive buffers. As receive activity does not result in lost packets, this number is slowly decreased toward the "Current minimum limit". If packets are lost this number is sharply increased toward the "Current maximum limit". When the driver allocates receive buffers, it allocates them until reaching this target number.

Target number of buffers maximum

Maximum desired number of receive buffers. When an application returns a buffer to the driver, if the current number does not exceed this target maximum, the buffer is kept by the driver. Otherwise, it is deallocated.

Fork Delay (after scheduled)

To help determine whether the buffering requirements of the driver and the chip are sufficient for the system configuration, the driver records the amount of time from fork scheduled to the time the fork is actually run. The data is recorded in 10-millisecond increments from 10 to 310 milliseconds.

This data can be used in conjunction with the number of packets discarded because there were insufficient buffers to determine whether the buffering settings of the driver (minimum and maximum receive buffers) and the amount of buffering on the chip are sufficient for normal operation. If packets are being discarded, the buffering should be increased until the number of packets lost is minimal.

Transmit Time

To help understand the operation of the system, driver, and chip, the driver records the time that each transmit buffer is given to the chip. When the transmit completes, the driver calculates the elapsed time, creating a histogram of transmit times from 10 to 310 milliseconds.

Receive completion time

To help understand the operation of the system and driver, the driver records the time taken to process each receive buffer. When the receive completes, the driver records the time that processing started. Then it delivers the packet to the user application. When the user returns it, the driver calculates this completion time, creating a histogram of receive times from 10 to 150 milliseconds.

One second timer time

The driver maintains a one-second timer (that runs 4 times a second). It expects the timer routine to be called close to the timer interval. By looking at the time difference between when a timer is expected to be called and when it actually is, you can understand the operation of the system and driver. The variance time is used to create a histogram of receive times from 10 to 150 milliseconds.

Statistics Block

These are the statistics kept by the device.

Driver Messages

Console messages issued by the driver.

Intel 1Gb Device-Specific Functions

The LANCP command SET DEVICE/DEVICE_SPECIFIC=(FUNCTION="FUNC",VALUE=(V1,V2,...)) provides a mechanism to issue device-specific functions to the Intel 1gb driver. Some functions are common to many LAN drivers. But most are useful in a diagnostic context and not necessarily appropriate for formal inclusion in the LANCP command repertoire.

Note that the function name is always 4 characters in length. The command requires the SYSPRV privilege.

The definition of these functions may change from one driver version to the next.

The functions are:

DEVICE_SPECIFIC=FUNCTION=("RCOR",VALUE=%Xptyvalue)
DEVICE_SPECIFIC=FUNCTION=("XCOR",VALUE=%Xptyvalue)

Corrupt a transmit or receive packet. When this function is done, the protocol type is specified, and if it matches on a transmit or receive packet, the next packet seen is corrupted by changing the last bit in the VCRP or CXB, XORing it with a 1. Then the protocol type is cleared so the corruption is not done again.

For example, to corrupt a PEDRIVER transmit packet:

```
$ mc lancp set dev/trace=(mask=xmtissue) eia
$ mc lancp set dev/dev=(func="XCOR", value=%x760) eia
```

For example, to corrupt a PEDRIVER receive packet:

```
$ mc lancp set dev/trace=(mask=rcvdone) eia
$ mc lancp set dev/dev=(func="RCOR", value=%x760) eia
```

DEVICE_SPECIFIC=FUNCTION=("LOSE",VALUE=%xaabbccdd)

Lose some packets, by corrupting the header of the packet. Corrupt packet 0xaabbccdd, where aa is seconds to corrupt, bb is number to corrupt, cc is byte number 0..255, dd is what to corrupt with.

For example, to send 32 transmit packets elsewhere (no more than 16 seconds), change the last byte of the destination address to 0x44:

```
$ mc lancp set dev/trace=(mask=xmtissue) eia
$ mc lancp set dev/dev=(func="LOSE", value=%x10200544) eia
```

To lose just 1 packet, change the 1st byte of the packet to 0x00:

```
$ mc lancp set dev/trace=(mask=xmtissue) eia
$ mc lancp set dev/dev=func="LOSE" eia
```

To lose 255 packets for the rest of the 10 millisecond tick, change the 1st byte of the packet to 0x00:

```
$ mc lancp set dev/trace=(mask=xmtissue) eia
$ mc lancp set dev/dev=(func="LOSE",value=%xFF0000) eia
```

DEVICE_SPECIFIC=FUNCTION=("FCOP",VALUE=setting)

Force DCBE transmit coalescing to force EI$COPY_TRANSMIT to get used (0 disable, 1 enable).

DEVICE_SPECIFIC=FUNCTION="DMIG"

Disable interrupt mitigation. Note that the TIDV register is set to the minimum value of 1 (spec says we can't set it to zero).

This is used for performance testing when it is desirable to get an interrupt immediately upon transmit and receive completion.

DEVICE_SPECIFIC=(FUNCTION="DXMT",VALUE=delayvalue)

Change the interrupt mitigation delay on transmit completion value, the "Transmit interrupt delay (usec)" value. This value is in microseconds, limited to 1000. If zero, the delay is disabled.

DEVICE_SPECIFIC=(FUNCTION="DRCV",VALUE=delayvalue)

Change the interrupt mitigation delay on receive completion value, the "Receive interrupt delay (usec)" value. This value is in microseconds, limited to 1000. If zero, the delay is disabled.

DEVICE_SPECIFIC=(FUNCTION="DINT",VALUE=delayvalue)

Change the interrupt delay value. The value is in microseconds, limited to 1000. If zero, the delay is disabled.

DEVICE_SPECIFIC=(FUNCTION="FLOW",VALUE=setting)

Disable flow control (value is zero), enable flow control (value is non-zero).

DEVICE_SPECIFIC=FUNCTION=("PROM",VALUE=setting)

1=Enable/0=disable force promiscuous mode, then reset device.

DEVICE_SPECIFIC=FUNCTION=("APRM",VALUE=setting)

1=Enable/0=disable auto promiscuous mode, then reset device.

Auto-promiscuous mode means the driver will enable promiscuous mode if it needs to according to the hypervisor in use. Enabling promiscuous mode may be needed if the MAC address is changed from the default power on address.

DEVICE_SPECIFIC=FUNCTION="SOFT"

Force a non-fatal, i.e., soft error. The device is restarted.

DEVICE_SPECIFIC=FUNCTION="HBUG"

Force a hardware bug (clear return rcv ring).

DEVICE_SPECIFIC=(FUNCTION="PRES",VALUE=periodicmask)

Schedule periodic random resets. The value supplied is AND'ed with (RSCC * 69069) and if the result is 3, a reset is done. So, for example, if the value supplied is 7, a reset is done, on average, every 8 seconds. (0 disable, n enable).

This is used to test error handling.

DEVICE_SPECIFIC=FUNCTION="FCRC"

Send the next packet without a valid CRC.

This is used to generate CRC errors for test purposes.

DEVICE_SPECIFIC=(FUNCTION="RBAD",VALUE=disableenable)

If value is zero, the chip will discard bad received packets. If value is non-zero, the bad packets will be received and discarded by the driver. This allows the packets to be traced and inspected.

DEVICE_SPECIFIC=FUNCTION=("WCSR",VALUE=(address,val)).

Write a value val to CSR address.

DEVICE_SPECIFIC=FUNCTION=("RCSR",VALUE=address).

Read a CSR address and print the result.

DEVICE_SPECIFIC=FUNCTION=("WPHY",VALUE=(address,val)).

Write a value val to PHY address.

DEVICE_SPECIFIC=FUNCTION=("ALLM",VALUE=setting)

1=Enable/0=disable all messages flag.

DEVICE_SPECIFIC=FUNCTION=("FLIN",VALUE=setting)

Force link indication (0 disable, 1 set link up, 2 set link down).

DEVICE_SPECIFIC=FUNCTION=("XTOB",VALUE=setting)

1=Enable,0=disable Bugcheck on transmit timeout.

DEVICE_SPECIFIC=FUNCTION=("XLON",VALUE=setting)

1=Enable,0=disable Bugcheck on long transmit completion.

DEVICE_SPECIFIC=FUNCTION=("FLON",VALUE=setting)

1=Enable,0=disable Bugcheck on long fork.

DEVICE_SPECIFIC=FUNCTION=("SLON",VALUE=setting)

1=Enable,0=disable Bugcheck on long second.

DEVICE_SPECIFIC=FUNCTION="STOP"

Stop the chip, thereby inducing a transmit timeout on the next transmit request.

DEVICE_SPECIFIC=FUNCTION=("XTMO",VALUE=setting)

Set transmit timeout (2..10000 which is (1..9999)*25 * 10milliseconds.

DEVICE_SPECIFIC=FUNCTION=("AVER",VALUE=setting)

Disable transmit timeout averting if setting is 4747.

Set to 4747 to avoid trying to avert a transmit timeout. By setting XTMO to 1 to artificially cause transmit timeouts, we are causing many transmit timeouts, which exercises the transmit timeout handling. If not set to 4747, the transmit would have been found to have completed and the timeout would be counted as an 'averted' transmit timeout.

DEVICE_SPECIFIC=FUNCTION="OMSG"

Reset number of OPCOM messages.

DEVICE_SPECIFIC=FUNCTION="SYSI"

Send the periodic System ID messages now instead of waiting until expiry of the 8-12 minute System ID timer.

This is a convenient way to get the driver to send 2 packets.

DEVICE_SPECIFIC=FUNCTION="SYSZ"

Disable the sending of periodic System ID messages.

This is used to disable transmits that might distract from other traffic you are looking at.

DEVICE_SPECIFIC=(FUNCTION="WAIT",VALUE=timevalue)

Wait the specified time in nanoseconds, at IPL 8 with the driver port lock held.

This is a convenient way to introduce a delay at IPL 8.

The timevalue is 32 bits, limiting the wait to 4 seconds.

DEVICE_SPECIFIC=FUNCTION="CVME"
DEVICE_SPECIFIC=FUNCTION="SVME"

VME bit set/clear. If clear, the VLAN tag will not be stripped from the packet so it can be seen in the receive buffer.

DEVICE_SPECIFIC=FUNCTION="ARST"

Restart auto-negotiation.

This is used for link handling testing.

DEVICE_SPECIFIC=FUNCTION="FLSC"

Force a link state change interrupt.

DEVICE_SPECIFIC=FUNCTION=("DELA",VALUE=n)

Delay the next transmit by n 10-millisecond ticks.

DEVICE_SPECIFIC=FUNCTION="XDEL"

Call ini$brk.

DEVICE_SPECIFIC=FUNCTION="STIM"

Measure statistics time, view result in internal counters 'Soft errors'.

Broadcom 1Gb Ethernet

The Broadcom 5700 Ethernet devices supported for x86 are:

Vendor ID	SubIDs	Bus	DeviceName
15218086	various	PCIe	BCM5719
15338086	various	PCIe	BCM5720

The driver is SYS$EW5700X.EXE.

Note:

There are multiple names for each device:

- PCI ID repository (google search for it).
- Vendor names and part numbers (there may be multiple names and part numbers).
- Full name assigned by the driver, including a part number if known, with some effort for consistency. This name should identify the device sufficiently to allow someone to order it and receive what is expected.

  The full name is the "device =" parameter in the entry in SYS$CONFIG.DAT and is the same name assigned by the driver.

  The full name is displayed by LANCP SHOW DEV/INTERNAL, SDA CLUE CONFIG/ADAPTER, and by SEARCH SYS$SYSTEM:SYS$CONFIG.DAT DEVICE,"="/MATCH=AND.
- Short name assigned by the driver for printouts that need a name shorter than the full name (LANCP SHOW CONFIG, LANCP SHOW DEV/INTERNAL).

  The short name is displayed by LANCP SHOW CONFIG and LANCP SHOW DEV/INTERNAL.
- Additional names or aliases are given in the comments before the SYS$CONFIG.DAT entry if you search SYS$CONFIG.DAT for 5719 or 5720.

The PCI Device IDs are stored in the busarray entry. To find for a particular device, do SDA CLUE CONFIG/ADAPTER and EXAMINE busarray address - the Device ID, Vendor ID, SubDevice ID, and SubVendor ID are the first two longwords.

Details for each device follow:

```
-------------------------------------------------------------------------------
----------------------- Broadcom 5719 Device ID 1657 --------------------------
-------------------------------------------------------------------------------

 Vendor/DevID  SubIDs   Bus     Device  PCIID Repository Name                 In SYS$CONFIG.DAT
    165714E4  169D103C  PCIe    Ethernet 1Gb 4-port 331FLR Adapter                   Yes
       Full name:  HPE 4-port BCM5719 331FLR Adapter (629135-B22) (Gigabit Ethernet)
       Short name: 5719 HPE 331FLR
       Other name: HPE Ethernet 1Gb 4-port 331FLR Adapter (629135-B22)
```

```
    165714E4  3383103C  PCIe    Ethernet 1Gb 4-port 331T Adapter                      Yes
       Full name:  HPE 4-port BCM5719 331T Adapter (647594-B21) (Gigabit Ethernet)
       Short name: 5719 HPE 331T
       Other name: HPE Ethernet 1Gb 4-port 331T Adapter (647594-B21)

    165714E4  22BE103C  PCIe    Ethernet 1Gb 4-port 331i Adapter                       Yes
       Full name:  HPE 4-port BCM5719 331i Adapter (Gigabit Ethernet)
       Short name: 5719 HPE 331i
       Other name: HPE Embedded 1Gb Ethernet 4-port 331i Adapter

    165714E4  190414E4  PCIe    4-port 1Gb Ethernet Adapter                            Yes
       Full name:  BCM5719 4-port Adapter (Gigabit Ethernet)
       Short name: BCM5719
       Other name: BCM5719 Ethernet 1Gb 4-port Adapter

    165714E4  Catch-All PCIe    Broadcom BCM5719                                       Yes
       Full name:  Broadcom BCM5719 Adapter (Gigabit Ethernet)
       Short name: BCM5719

-------------------------------------------------------------------------------
----------------------- Broadcom 5720 Device ID 165F ------------------------
-------------------------------------------------------------------------------

 Vendor/DevID  SubIDs   Bus     Device  PCIID Repository Name                  In SYS$CONFIG.DAT
    165F14E4  04F71028  PCIe    PowerEdge R320 server                                  Yes
       Full name:  Dell PowerEdge R320 2-port BCM5720 LOM (Gigabit Ethernet)
       Short name: BCM5720 LOM
       Other name: Dell PowerEdge R320 2-port LOM

    165F14E4  08FD1028  PCIe    PowerEdge R6515/R7515 LOM                              Yes
       Full name:  Dell PowerEdge R6515/R7515 2-port BCM5720 LOM (Gigabit Ethernet)
       Short name: BCM5720 LOM
       Other name: Dell PowerEdge R6515/R7515 2-port LOM

    165F14E4  08FF1028  PCIe    PowerEdge Rx5xx LOM Board                              Yes
       Full name:  Dell PowerEdge Rx5xx 2-port BCM5720 LOM (Gigabit Ethernet)
       Short name: BCM5720 LOM
       Other name: Dell PowerEdge Rx5xx 2-port LOM

    165F14E4  09001028  PCIe    PowerEdge C6525 LOM                                    Yes
       Full name:  Dell PowerEdge C6525 2-port BCM5720 LOM (Gigabit Ethernet)
       Short name: BCM5720 LOM
       Other name: Dell PowerEdge C6525 2-port LOM

    165F14E4  09171028  PCIe    PowerEdge C6520 LOM                                    Yes
       Full name:  Dell PowerEdge C6520 2-port BCM5720 LOM (Gigabit Ethernet)
       Short name: BCM5720 LOM
       Other name: Dell PowerEdge C6520 2-port LOM

    165F14E4  1786103C  PCIe    NC332T Adapter                                         Yes
       Full name:  HPE 2-port BCM5720 332T Adapter (615732-B21) (Gigabit Ethernet)
       Short name: 5720 HPE 332T
       Other name: HPE Ethernet 1Gb 2-port 332T Adapter (615732-B21)

    165F14E4  193D103C  PCIe    NC332i Adapter                                         Yes
       Full name:  HPE 2-port BCM5720 332i Adapter (Gigabit Ethernet)
       Short name: 5720 HPE 332i
       Other name: HPE Embedded 1Gb Ethernet 2-port 332i Adapter

    165F14E4  2133103C  PCIe    NC332i Adapter                                         Yes
       Full name:  HPE 2-port BCM5720 332i Adapter (Gigabit Ethernet)
       Short name: 5720 HPE 332i
       Other name: HPE Embedded 1Gb Ethernet 2-port 332i Adapter

    165F14E4  22E8103C  PCIe    NC332i Adapter                                         Yes
       Full name:  HPE 2-port BCM5720 332i Adapter (Gigabit Ethernet)
       Short name: 5720 HPE 332i
       Other name: HPE Embedded 1Gb Ethernet 2-port 332i Adapter
```

```
165F14E4  22EB103C  PCIe    NC332i Adapter                                          Yes
   Full name:  HPE 2-port BCM5720 332i Adapter (Gigabit Ethernet)
   Short name: 5720 HPE 332i
   Other name: HPE Embedded 1Gb Ethernet 2-port 332i Adapter

165F14E4  Catch-All PCIe    Broadcom BCM5720                                        Yes
   Full name:  Broadcom BCM5720 Adapter (Gigabit Ethernet)
   Short name: BCM5720
```

Broadcom 1Gb Driver Internal Counters

The LANCP command SHOW DEVICE/INTERNAL_COUNTERS EWA displays the entire set of internal counters maintained by the 5700 driver. Some counters are special debug counters. These are not displayed unless the additional qualifier /DEBUG is specified. Counters that are zero are not displayed unless the additional qualifier /ZERO is specified.

The LAN$SDA SDA extension also displays the complete set of internal counters with the command LAN INTERNAL/DEVICE=EWA.

The definition of these counters may change from one driver version to the next.

The counters are:

Device name (full)
Device name (short)

Name of the Intel 1gb device.

Driver timestamp

Compilation timestamp of the driver.

Driver version

Driver version numbered 1...n that usually is identical to the x-n ID displayed by an ANALYZE/IMAGE of the driver image. It includes variant information, if any. The full driver version includes the target OpenVMS release and is displayed by SDA LAN/DEV=EWA in the quadword driver version field.

Device version (Broadcom 5719/5720 chip)

Hardware revision level identified by the Broadcom chip revision.

Device PHY ID

ID of the PHY chip found on the device. Bits <31:10> is the OUI, bits <9:4> is the Model number and bits <3:0> is the revision.

Device interrupts

Number of times the driver interrupt service routine was called.

Link transitions

Number of link transitions seen by the driver.

Status block errors

Number of times the error bit in the status block was set at the same time the fiber link partner was sending configuration data for auto-negotiation.

Status block link state changes

Number of times the link state change bit was set in the status block.

Link checks

Number of times the driver checked the current link status.

Device resets

Number of times the 5700 chip was reset.

Device resets retried (after 1.5 second timeout)

Number of times the 5700 chip reset was retried because the 5700 firmware did not complete initialization the first time.

Device initializations

Number of times the driver initialized the 5700 chip.

Unit inits

Number of unit initializations executed; since unit initialization is only executed once, this counter will be one.

User start/change/stop requests

Number of user startup and shutdown requests processed by the driver, generally one or two when a user starts up, and one when a user stops.

Transmits queued

Number of transmit requests queued because the link was not available or because too many transmit requests were already outstanding.

Transmit timeouts

Number of times the driver has timed out a transmit and has reset the device and completed outstanding I/O with error status.

Transmit timeouts (averted)

Number of transmit timeouts averted by a final check for transmit completion.

Transmit chained (buffer address)

Number of chained transmits where the chaining is via SVA buffer address.

Transmit chained (svapte_sva)

Number of chained transmits where the chaining is via a buffer extent.

Transmit errors (too few segments)

Number of transmit requests completed with error status (SS$_INCSEGTRA) because the application did not specify the transmit buffer completely.

Transmit informationals (zero byte segments)

Number of transmit chain segments which specified a zero-byte length segment.

Transmit ring inconsistencies

Number of times the transmit ring entry did not match expected, so the device was reset and restarted.

Transmit copies (too many segments)

Number of transmit requests that exceeded the maximum number of chain segments that the driver can handle; the driver then copied some of them to a temporary buffer so that it could transmit the packet.

Transmit copy failures

Number of transmit failures caused by too few chain segments when coalescing a request to reduce the number of segments needed.

Jumbo transmits issued

Number of transmit requests with a packet length exceeding 1514 bytes (excluding CRC).

Jumbo receives issued

Number of jumbo receive buffers allocated and given to the device.

Chained receives completed

Number of receives (jumbo packets) completed by the driver that span multiple receive buffers.

Receives discarded (bad frame)

Number of receive packets discarded by the driver due to receive error. By default, the 5700 does not deliver damaged receive packets to the driver, but can be asked to do so by a device-specific function.

Receive entry mismatches (resulting in reset)

Number of times the receive return ring entry did not match expected, so the device was reset and restarted.

Soft errors

Number of times errors were recovered in the driver by resetting and reinitializing the device, without notification of user applications.

Rescheduled forks (too long in fork)

Number of times that a rescheduled fork was done. In transmit and receive processing, the driver limits the amount of time spent in the fork process before rescheduling.

Standard packet size (bytes)

This is the Ethernet packet size (1518 bytes).

Jumbo packet size (bytes)

This is the jumbo packet size (9018 bytes).

Buffers in standard ring

Number of 2304-byte receive buffers owned by the device and located in the standard size ring.

Buffers in jumbo ring

Number of 2304-byte receive buffers owned by the device and located in the jumbo size ring.

Requested link state

Speed and duplex mode requested by a user.

Current link state

Current link state.

Remote flow capabilities

Flow control capabilities reported for the link partner.

Current link up timer

Current value of the one-second timer that waits for the link to come back before resetting the device to free pending transmit requests.

Driver flags

Driver flags including the following bits:

- Promiscuous - Promiscuous mode enabled.
- All_Multicast - All multicast mode enabled.
- Jumbo_Frames - Jumbo packets enabled.
- 5719 - 5719 chip.
- 5720 - 5720 chip.
- Receive_Bad_Frames - Receive bad frames enabled.
- Bugcheck_XTMO - Bugcheck on transmit timeout.
- BugcheckLongXmt - Bugcheck on long transmit time.
- BugcheckLongFord - Bugcheck on long fork time.
- BugcheckLongSec - Bugcheck on long second time.
- Setup_Link_In_Progress - Currently setting up the link.
- All_Messages_Enabled - Display all messages.
- ForcePromiscuous - Force promiscuous mode enabled.
- DisableInterruptMitigation - Disable interrupt mitigation.
- PeriodicReset - Do period device reset.
- ForceLinkUp - Force the link state up.
- ForceLinkDOwn - Force the link state down.
- SendNoCRC - Don't include CRC in transmit requests.

Driver state

Current driver state, one of the following:

0 - Driver state is undefined
1 - Driver is initializing the device
2 - Driver is running but the link is down, completing transmits with error status
4 - Driver is running but the link is down, queueing transmits for now
8 - Driver is running and the link is up, processing transmits, and receives
16 - Device is not usable

LAN_FLAGS system parameter

Current value of the LAN_FLAGS system parameter.

MSI control (Alloc<6:4>,Req<3:1>,Enable<0>)

Contents of the MSI_CONTROL register that gives MSI status. If MSI is enabled, bit <0> is set. Bits <3:1> are the number of requested messages (always set to 3 by the device). Bits <6:4> are the number of messages allocated to the device by the system. If the /DEBUG qualifier is specified, more MSI context is displayed, such as MSI Address and Data registers.

DMA control value

Contents of the DMA control register which regulates read and write DMA operations.

Interrupt mode

Unknown, IOSAPIC, MSI, MSIX, or IOAPIC.

CPMU status (function <31:30>)

Central Power Management Unit (CPMU) Status register, containing function number in <31:30>.

Transmit interrupt delay (usec)

Transmit interrupts are generated every n "coalesce value" transmit completions, but no later than n "interrupt delay" microseconds after transmit completion.

Receive interrupt delay (usec)

Receive interrupts are generated every n "coalesce value" receive completions, but no more than n "interrupt delay" microseconds after receipt of a packet.

Initialization delay (usecs)

Total time (in microseconds) waiting for the device during initialization including shutdown.

Link setup delay (usecs)

Total time (in microseconds) waiting for the device during setup of the link.

Status block VA

System VA of the start of the status block.

Receive ring VA

System VA of the start of the receive ring.

Receive ring (jumbo) VA

System VA of the start of the jumbo receive ring.

Receive ring (rcv return) VA

System VA of the start of the return receive ring.

Transmit ring VA

System VA of the start of the transmit rings.

EEPROM VA

System VA of the start of the EEPROM data.

LSB size

Total size of the LAN Station Block (LSB) structure.

Transmit time limit

Transmit time limit in seconds after which a timeout is declared.

Timer routine interval

Resolution of the transmit timer (the real timeout is Limit + Interval).

Registers (wrote/read)

- Misc Host Control
- MAC Mode
- MAC Status
- MI Status
- RX Mode
- TX Status
- LED Control activity (LOM only)
- LED Control (wrote)

Time Stamps

Current uptime

Current system uptime.

Last reset

Last time the device was reset.

Last link state change interrupt

Last time a link state change interrupt was fielded.

Previous link state change interrupt

Previous time a link state change interrupt was fielded.

Last link up

Last time a link up transition occurred.

Last link down

Last time a link down transition occurred.

Total link uptime

Total time the link has been up.

Total link downtime

Total time the link has been down.

Time of last uplink period
Time of last uplink period-1
Time of last uplink period-2
Time of last uplink period-3
Time of last uplink period-4
Time of last uplink period-5
Time of last uplink period-6
Time of last uplink period-7
Time of last uplink period-8
Time of last uplink period-9
Time of last uplink period-10
Time of last uplink period-11
Time of last uplink period-12
Time of last uplink period-13
Time of last uplink period-14

History of the last 15 link uptimes - the length of each uptime period.

Last transmit timeout

Time of the last transmit timeout.

Last transmit inconsistency

Time of the last transmit ring inconsistency.

Last soft error

Time of the last soft error.

Last CRC error

Time of the last CRC error on receive.

Last late collision error

Time of the last late collision error reported.

Last receive error

Time of the last receive error.

Last receive error packet header

First 24 bytes of the last packet received with error status.

Last unrecognized unicast packet

Time of the last unrecognized unicast packet received.

Last unrecognized unicast packet header

First 24 bytes of the last unicast packet received that was discarded because receive address filtering did not result in a matching user.

Last unrecognized multicast packet

Time of the last unrecognized multicast packet received.

Last unrecognized multicast packet header

First 24 bytes of the last multicast packet received that was discarded because receive address filtering did not result in a matching user.

PCI Config Registers

- CFID register
- CFRV register
- CFLT register
- PCI_ADDRESS0 register
- PCI_ADDRESS1 register
- SUB_VNDR / SUB_ID register
- CACHE_LINE_SIZE register

Receive Buffers

Minimum buffers requested

Minimum number of receive buffers as set by default or by management request (setmode or LANCP command). This is used to determine the "Current minimum limit".

Maximum buffers requested

Maximum number of receive buffers as set by default or by management request (setmode or LANCP command). This is used to determine the "Current maximum limit".

Current minimum limit

Minimum number of receive buffers as determined by the driver and by management request, "Minimum buffers requested". The driver will not allow the number of receive buffers in its receive queues to drop below this value. This does not include receive buffers handed up to applications (and not yet returned).

Current maximum limit

Maximum number of receive buffers as determined by the driver and by management request, "Maximum buffers requested". The driver will deallocate receive buffers until the number allocated does not exceed than this maximum. When applications return receive buffers, the number may exceed this maximum at which point they are deallocated. Note that the driver allows this maximum to be exceeded somewhat to limit allocation and deallocation activity. See "Target number of buffers maximum" below.

Current number of buffers

Current number of receive buffers owned by the driver.

Target number of buffers

Desired number of receive buffers. As receive activity does not result in lost packets, this number is slowly decreased toward the "Current minimum limit". If packets are lost this number is sharply increased toward the "Current maximum limit". When the 5700 driver allocates receive buffers, it allocates them until reaching this target number.

Target number of buffers maximum

Maximum desired number of receive buffers. When an application returns a buffer to the driver, if the current number does not exceed this target maximum, the buffer is kept by the driver. Otherwise, it is deallocated.

Fork Delay (after scheduled)

To help determine whether the buffering requirements of the driver and the 5700 are sufficient for the system configuration, the driver records the amount of time from fork scheduled to the time the fork is actually run. The data is recorded in 10-millisecond increments from 10 to 310 milliseconds.

This data can be used in conjunction with the number of packets discarded because there were insufficient buffers to determine whether the buffering settings of the driver (minimum and maximum receive buffers) and the amount of buffering on the 5700 are sufficient for normal operation. If packets are being discarded, the buffering should be increased until the number of packets lost is minimal.

Transmit Time

To help understand the operation of the system, driver, and 5700, the driver records the time that each transmit buffer is given to the 5700. When the transmit completes, the driver calculates the elapsed time, creating a histogram of transmit times from 10 to 310 milliseconds.

Receive completion time

To help understand the operation of the system and driver, the driver records the time taken to process each receive buffer. When the receive completes, the driver records the time that processing started. Then it delivers the packet to the user application. When the user returns it, the driver calculates this completion time, creating a histogram of receive times from 10 to 150 milliseconds.

One second timer time

The driver maintains a one-second timer (that runs 4 times a second). It expects the timer routine to be called close to the timer interval. By looking at the time difference between when a timer is expected to be called and when it actually is, you can understand the operation of the system and driver. The variance time is used to create create a histogram of receive times from 10 to 150 milliseconds.

Status Block

Status

After an interrupt, this longword is nonzero. The driver fork process clears the status longword so the driver can recognize whether there is pending work to be done (checking the ring indexes for completions or checking the link status). The bits in the longword are defined as follows:

Bit 1 - Link changed status
Bit 2 - Status block updated flag
Bit 4 - Status block error

Receive Mini Consumer index

Unused.

Receive Jumbo Consumer index

Current position of the 5700 in the jumbo receive buffer ring.

Receive Standard Consumer index

Current position of the 5700 in the standard receive buffer ring.

Receive Return Producer index

Current position of the 5700 in the receive return ring.

Send ring 0 consumer index

Current position of the 5700 in the send ring.

Statistics Block

Statistics - Transmit

- Bytes sent
- Unicast packets sent
- Multicast packets sent
- Broadcast packets sent
- Collisions experienced
- Internal MAC transmit errors

- Single collisions
- Multiple collisions
- Excessive collisions
- Late collisions
- Deferred transmits
- Carrier sense errors
- Outbound discards
- Send failure (link down)

Statistics - Receive

- Bytes received
- Bad byte received
- Runt packets received (bad CRC)
- Unicast packets received
- Multicast packets received
- Broadcast packets received
- CRC errors
- Alignment errors
- MAC Control other Frames received
- Frame too long errors
- Frame exceeded jabber time errors
- Runt packets received (good CRC)

Statistics - Flow Control

- Transmitting disabled (xoff)
- XON sent
- XOFF sent
- XON received
- XOFF received
- No more RXBDs
- In discards
- In errors

Debug Data

Consists of a list of the contents significant 5700 chip registers, collected when internal driver counters were requested.

EEPROM Contents

Consists of the contents of the non-volatile memory on the 5700 NIC, which contains manufacturing information.

Driver Messages

Console messages issued by the driver.

Broadcom 1Gb Device-Specific Functions

The LANCP command SET DEVICE/DEVICE_SPECIFIC=(FUNCTION="FUNC",VALUE=(V1,V2,...)) provides a mechanism to issue device-specific functions to the 5700 driver. Some functions are common to many LAN drivers. But most are useful in a diagnostic context and not necessarily appropriate for formal inclusion in the LANCP command repertoire.

Note that the function name is case sensitive and is always 4 characters in length. The command requires the SYSPRV privilege.

The functions are:

DEVICE_SPECIFIC=FUNCTION=("RCOR",VALUE=%Xptyvalue)
DEVICE_SPECIFIC=FUNCTION=("XCOR",VALUE=%Xptyvalue)

Corrupt a transmit or receive packet. When this function is done, the protocol type is specified, and if it matches on a transmit or receive packet, the next packet seen is corrupted by changing the last bit in the VCRP or CXB, XORing it with a 1. Then the protocol type is cleared so the corruption is not done again.

For example, to corrupt a PEDRIVER transmit packet:

```
$ mc lancp set dev/trace=(mask=xmtissue) ewa
$ mc lancp set dev/dev=(func="XCOR", value=%x760) ewa
```

For example, to corrupt a PEDRIVER receive packet:

```
$ mc lancp set dev/trace=(mask=rcvdone) ewa
$ mc lancp set dev/dev=(func="RCOR", value=%x760) ewa
```

DEVICE_SPECIFIC=FUNCTION=("LOSE",VALUE=%xaabbccdd)

Lose some packets, by corrupting the header of the packet. Corrupt packet 0xaabbccdd, where aa is seconds to corrupt, bb is number to corrupt, cc is byte number 0..255, dd is what to corrupt with.

For example, to send 32 transmit packets elsewhere (no more than 16 seconds), change the last byte of the destination address to 0x44:

```
$ mc lancp set dev/trace=(mask=xmtissue) ewa
$ mc lancp set dev/dev=(func="LOSE", value=%x10200544) ewa
```

To lose just 1 packet, change the 1st byte of the packet to 0x00:

```
$ mc lancp set dev/trace=(mask=xmtissue) ewa
$ mc lancp set dev/dev=func="LOSE" ewa
```

To lose 255 packets for the rest of the 10 millisecond tick, change the 1st byte of the packet to 0x00:

```
$ mc lancp set dev/trace=(mask=xmtissue) ewa
$ mc lancp set dev/dev=(func="LOSE",value=%xFF0000) ewa
```

DEVICE_SPECIFIC=FUNCTION=("FCOP",VALUE=setting)

Force DCBE transmit coalescing to force EW$COPY_TRANSMIT to get used (0 disable, 1 enable).

DEVICE_SPECIFIC=FUNCTION="INTR"

Generate an interrupt.

DEVICE_SPECIFIC=(FUNCTION="DMAC",VALUE=value)

Write the DMA Control Register with the value given.

DEVICE_SPECIFIC=(FUNCTION="DXMT",VALUE=delayvalue)

Change the interrupt mitigation delay on transmit completion value, the "Transmit interrupt delay (usec)" value. This value is in microseconds, limited to 1000. If zero, the delay is disabled.

DEVICE_SPECIFIC=(FUNCTION="DRCV",VALUE=delayvalue)

Change the interrupt mitigation delay on receive completion value, the "Receive interrupt delay (usec)" value. This value is in microseconds, limited to 1000. If zero, the delay is disabled.

DEVICE_SPECIFIC=(FUNCTION="CXMT",VALUE=packetvalue)

Change the interrupt mitigation coalesce on transmit completion value, the "Transmit coalesce value". This value is in number of packets. If less than 1, it is set to 1. If greater than 64, it is set to 64.

DEVICE_SPECIFIC=(FUNCTION="CRCV",VALUE=packetvalue)

Change the interrupt mitigation coalesce on receive completion value, the "Receive coalesce value". This value is in number of packets. If less than 1, it is set to 1. If greater than 128, it is set to 128.

DEVICE_SPECIFIC=(FUNCTION="FLOW",VALUE=setting)

Disable flow control (value is zero), enable flow control (value is non-zero).

DEVICE_SPECIFIC=FUNCTION=("PROM",VALUE=setting)

1=Enable/0=disable force promiscuous mode, then reset device.

DEVICE_SPECIFIC=FUNCTION="SOFT"

Force a non-fatal, i.e., soft error. The device is restarted.

DEVICE_SPECIFIC=FUNCTION="MACH"

Check state machines for operational status.

DEVICE_SPECIFIC=FUNCTION="DIAG"

Read and print the diagnostic registers.

DEVICE_SPECIFIC=FUNCTION="HBUG"

Force a hardware bug (clear return rcv ring).

DEVICE_SPECIFIC=FUNCTION=("SPAR",VALUE=setting)

Set spare value for debug purposes.

DEVICE_SPECIFIC=(FUNCTION="PRES",VALUE=periodicmask)

Schedule periodic random resets. The value supplied is AND'ed with (RSCC * 69069) and if the result is 3, a reset is done. So, for example, if the value supplied is 7, a reset is done, on average, every 8 seconds. (0 disable, n enable).

This is used to test error handling.

DEVICE_SPECIFIC=FUNCTION="FCRC"

Send the next packet without a valid CRC.

This is used to generate CRC errors for test purposes.

DEVICE_SPECIFIC=(FUNCTION="RBAD",VALUE=disableenable)

If value is zero, the chip will discard bad received packets. If value is non-zero, the bad packets will be received and discarded by the driver. This allows the packets to be traced and inspected.

DEVICE_SPECIFIC=FUNCTION=("WCSR",VALUE=(address,val)).

Write a value val to CSR address.

DEVICE_SPECIFIC=FUNCTION=("ALLM",VALUE=setting)

1=Enable/0=disable all messages flag.

DEVICE_SPECIFIC=FUNCTION=("FLIN",VALUE=setting)

Force link indication (0 disable, 1 set link up, 2 set link down).

DEVICE_SPECIFIC=FUNCTION=("XTOB",VALUE=setting)

1=Enable,0=disable Bugcheck on transmit timeout.

DEVICE_SPECIFIC=FUNCTION=("XLON",VALUE=setting)

1=Enable,0=disable Bugcheck on long transmit completion.

DEVICE_SPECIFIC=FUNCTION=("FLON",VALUE=setting)

1=Enable,0=disable Bugcheck on long fork.

DEVICE_SPECIFIC=FUNCTION=("SLON",VALUE=setting)

1=Enable,0=disable Bugcheck on long second.

DEVICE_SPECIFIC=FUNCTION="STOP"

Stop the chip, thereby inducing a transmit timeout on the next transmit request.

DEVICE_SPECIFIC=FUNCTION=("XTMO",VALUE=setting)

Set transmit timeout (2..10000 which is (1..9999)*25 * 10milliseconds.

DEVICE_SPECIFIC=FUNCTION=("AVER",VALUE=setting)

Disable transmit timeout averting if setting is 4747.

Set to 4747 to avoid trying to avert a transmit timeout. By setting XTMO
to 1 to artificially cause transmit timeouts, we are causing many transmit timeouts, which exercises the transmit timeout handling. If not set to 4747, the transmit would have been found to have completed and the timeout would be counted as an 'averted' transmit timeout.

DEVICE_SPECIFIC=FUNCTION="OMSG"

Reset number of OPCOM messages.

DEVICE_SPECIFIC=FUNCTION="SYSI"

Send the periodic System ID messages now instead of waiting until expiry of the 8-12 minute System ID timer.

This is a convenient way to get the driver to send 2 packets.

DEVICE_SPECIFIC=FUNCTION="SYSZ"

Disable the sending of periodic System ID messages.

This is used to disable transmits that might distract from other traffic you are looking at.

DEVICE_SPECIFIC=(FUNCTION="WAIT",VALUE=timevalue)

Wait the specified time in nanoseconds, at IPL 8 with the driver port lock held.

This is a convenient way to introduce a delay at IPL 8.

The timevalue is 32 bits, limiting the wait to 4 seconds.

DEVICE_SPECIFIC=FUNCTION="FLSC"

Force a link state change interrupt.

DEVICE_SPECIFIC=FUNCTION=("DELA",VALUE=n)

Delay the next transmit by n 10-millisecond ticks.

DEVICE_SPECIFIC=FUNCTION="XDEL"

Call ini$brk.

Intel 10Gb Ethernet

The Intel 10Gb devices are, via chip designation:

Chip type translation:

Chip name	Chip number
Twinfille	X540
Sageville	X550

Device type translation:

Vendor ID	SubIDs	Bus	Name	Description
15288086	192D103C	PCIe	X540	HPE 2-port Intel 561FLR-T Adapter (700699-B21)
15288086	211A103C	PCIe	X540	HPE 2-port Intel 561T Adapter (716591-B21)
15288086	00BF1137	PCIe	X540	Converged Network Adapter Intel X540-T2
15288086	00018086	PCIe	X540	Converged Network Adapter Intel X540-T2
15288086	00028086	PCIe	X540	Converged Network Adapter Intel X540-T1
15288086	001A8086	PCIe	X540	Converged Network Adapter Intel X540-T2
15288086	00A28086	PCIe	X540	Converged Network Adapter Intel X540-T1
15288086	catchall	PCIe	X540	2P Intel X540-t Adapter
15608086	catchall	PCIe	X540	2P Intel X540T1 Adapter
15638086	00011170	PCIe	X550	Intel Controller X550-T2 OCP card
15638086	120114C0	PCIe	X550	Intel X550 2P RJ45 OCP Mezz
15638086	00D11590	PCIe	X550	HPE 2-port Intel 561T Adapter (817738-B21)
15638086	00D21590	PCIe	X550	HPE 2-port Intel 561FLR-T Adapter (817745-B21)
15638086	0C0818D4	PCIe	X550	Intel X550 2-port RJ45 OCP Mezz Card MOP81-I-10GT2
15638086	00018086	PCIe	X550	Converged Network Adapter Intel X550-T2
15638086	001A8086	PCIe	X550	Converged Network Adapter Intel X550-T2
15638086	001B8086	PCIe	X550	Server Adapter Intel X550-T2 for OCP
15638086	00228086	PCIe	X550	Converged Network Adapter Intel X550-T2
15638086	catchall	PCIe	X550	2P Intel X550-t Adapter
15D18086	catchall	PCIe	X550	2P Intel X550T1 Adapter

Notes:

There are multiple names for each device:

- PCI ID repository (google search for it).
- Vendor names and part numbers (there may be multiple names and part numbers).
- Full name assigned by the driver, including a part number if known, with some effort for consistency. This name should identify the device sufficiently to allow someone to order it and receive what is expected.

  The full name is the "device =" parameter in the entry in SYS$CONFIG.DAT and is the same name assigned by the driver.

The full name is displayed by LANCP SHOW DEV/INTERNAL, SDA CLUE CONFIG/ADAPTER, and by SEARCH SYS$SYSTEM:SYS$CONFIG.DAT DEVICE,"="/MATCH=AND.

- Short name assigned by the driver for printouts that need a name shorter than the full name (LANCP SHOW CONFIG, LANCP SHOW DEV/INTERNAL).

  The short name is displayed by LANCP SHOW CONFIG and LANCP SHOW DEV/INTERNAL.

- Additional names or aliases are given in the comments before the SYS$CONFIG.DAT entry.

  The PCI Device IDs are stored in the busarray entry. To find for a particular device, do SDA CLUE CONFIG/ADAPTER and EXAMINE busarray address - the Device ID, Vendor ID, SubDevice ID, and SubVendor ID are the first two longwords.

Details for each device follow:

```
-------------------------------------------------------------------------------
----------------------- Intel X540 Device ID 1528 ----------------------------
-------------------------------------------------------------------------------

Vendor/DevID  SubIDs   Bus   Device  PCIID Repository Name               In SYS$CONFIG.DAT
15288086  192D103C  PCIe  HPE 2-port Intel 561FLR-T Adapter (700699-B21)     Yes
Full name:  HPE Ethernet 10Gb 2-port 561FLR-T Adapter (700699-B21)
Short name: X540 HPE 561FLR

15288086  211A103C  PCIe  HPE 2-port Intel 561T Adapter (716591-B21)        Yes
Full name:  HPE Ethernet 10Gb 2-port 561T Adapter (716591-B21)
Short name: X540 HPE 561T

15288086  00BF1137  PCIe  Converged Network Adapter Intel X540-T2           Yes
Full name:  Ethernet Converged Network Adapter X540-T2
Short name: X540 CNA

15288086  00018086  PCIe  Converged Network Adapter Intel X540-T2           Yes
Full name:  Ethernet Converged Network Adapter X540-T2
Short name: X540 CNA

15288086  00028086  PCIe  Converged Network Adapter Intel X540-T1           Yes
Full name:  Ethernet Converged Network Adapter X540-T1
Short name: X540 CNA

15288086  001A8086  PCIe  Converged Network Adapter Intel X540-T2           Yes
Full name:  Ethernet Converged Network Adapter X540-T2
Short name: X540 CNA

15288086  00A28086  PCIe  Converged Network Adapter Intel X540-T1           Yes
Full name:  Ethernet Converged Network Adapter X540-T1
Short name: X540 CNA

15288086  catchall  PCIe  2P Intel X540-t Adapter                           Yes
Full name:  10G 2P X540-t Adapter
Short name: X540 NIC

-------------------------------------------------------------------------------
----------------------- Intel X540 Device ID 1560 ----------------------------
-------------------------------------------------------------------------------

Vendor/DevID  SubIDs   Bus   Device  PCIID Repository Name               In SYS$CONFIG.DAT
15608086  catchall  PCIe  2P Intel X540T1 Adapter                          Yes
Full name:  10G 2P X540T1 Adapter
Short name: X540 NIC
```

```
------------------------------------------------------------------------------
----------------------- Intel X550 Device ID 1563 ----------------------------
------------------------------------------------------------------------------

Vendor/DevID  SubIDs   Bus   Device  PCIID Repository Name              In SYS$CONFIG.DAT
15638086  00011170  PCIe  Intel Controller X550-T2 OCP card                 Yes
Full name:  Intel Ethernet Controller X550-T2 OCP card
Short name: X550 NIC

15638086  120114C0  PCIe  Intel X550 2P RJ45 OCP Mezz                       Yes
Full name:  X550 10Gb 2P RJ45 OCP Mezz
Short name: X550 Mezz

15638086  00D11590  PCIe  HPE 2-port Intel 561T Adapter (817738-B21)       Yes
Full name:  HPE Ethernet 10Gb 2-port 561T Adapter (817738-B21)
Short name: X550 HPE 561T

15638086  00D21590  PCIe  HPE 2-port Intel 561FLR-T Adapter (817745-B21)    Yes
Full name:  HPE Ethernet 10Gb 2-port 561FLR-T Adapter (817745-B21)
Short name: X550 HPE 561FLR

15638086  0C0818D4  PCIe  Intel X550 2-port RJ45 OCP Mezz Card MOP81-I-10GT2 Yes
Full name:  X550 10Gb 2-port RJ45 OCP Mezz Card MOP81-I-10GT2
Short name: X550 Mezz

15638086  00018086  PCIe  Converged Network Adapter Intel X550-T2          Yes
Full name:  Ethernet Converged Network Adapter X550-T2
Short name: X550 NIC

15638086  001A8086  PCIe  Converged Network Adapter Intel X550-T2          Yes
Full name:  Ethernet Converged Network Adapter X550-T2
Short name: X550 NIC

15638086  001B8086  PCIe  Server Adapter Intel X550-T2 for OCP             Yes
Full name:  Ethernet Server Adapter X550-T2 for OCP
Short name: X550 NIC

15638086  00228086  PCIe  Converged Network Adapter Intel X550-T2          Yes
Full name:  Ethernet Converged Network Adapter X550-T2
Short name: X550 NIC

15638086  catchall  PCIe  2P Intel X550-t Adapter                          Yes
Full name:  10G 2P X550-t Adapter
Short name: X550 NIC

------------------------------------------------------------------------------
----------------------- Intel X550 Device ID 15D1 ----------------------------
------------------------------------------------------------------------------

Vendor/DevID  SubIDs   Bus     Device  PCIID Repository Name            In SYS$CONFIG.DAT
15D18086  catchall  PCIe  2P Intel X550T1 Adapter                           Yes
Full name:  10G 2P X550T1 Adapter
Short name: X550 NIC
```

Intel 10Gb Internal Counters

The LANCP command SHOW DEVICE/INTERNAL_COUNTERS EWA displays the entire set of internal counters maintained by the Intel 10Gb driver. Some counters are special debug counters. These are not displayed unless the additional qualifier /DEBUG is specified. Counters that are zero are not displayed unless the additional qualifier /ZERO is specified.

The LAN$SDA SDA extension also displays the complete set of internal counters with the command LAN INTERNAL/DEVICE=EIA.

The definition of these counters may change from one driver version to the next.

The counters are:

```
Device name (full)
Device name (short)

    Name of the Intel 1gb device.

Driver timestamp

    Compilation timestamp of the driver.

Driver version

    Driver version numbered 1...n that usually is identical to the x-n ID displayed by an ANALYZE/IMAGE
    of the driver image. It includes variant information, if any. The full driver version includes the
    target OpenVMS release and is displayed by SDA LAN/DEV=EIA in the quadword driver version field.

Device revision

    Hardware revision level of the chip.

Device interrupts

    Number of times the driver interrupt service routine was called.

Link transitions

    Number of link transitions seen by the driver.

Interrupt link state changes

    Number of link state changes reported in the interrupt control register.

Link checks

    Number of times the driver checked the current link status.

Link setup delay (usecs)

    Total time (in microseconds) waiting for the device during setup of the link.

Maximum link setup delay (usecs)

    Maximum time (in microseconds) waiting for the device during setup of the link.

VCI port usable/unusable events reported

    Number of VCI user port events reported.

Device resets

    Number of times the device was reset.

CSR Base Address

    PCI BAR0 CSR base address.

Unit inits

    Number of unit initializations executed; since unit initialization is only executed once, this
    counter will be one.
```

User start/change/stop requests

Number of user startup and shutdown requests processed by the driver, generally one or two when a user starts up, and one when a user stops.

Transmits queued

Number of transmit requests queued because the link was not available or because too many transmit requests were already outstanding.

Transmit timeouts

Number of times the driver has timed out a transmit and has reset the device and completed outstanding I/O with error status.

Transmit timeouts (averted)

Number of transmit timeouts averted by a final check for transmit completion.

Transmit chained (buffer address)

Number of chained transmits where the chaining is via SVA buffer address.

Transmit chained (svapte_sva)

Number of chained transmits where the chaining is via a buffer extent.

Transmit errors (too few segments)

Number of transmit requests completed with error status (SS$_INCSEGTRA) because the application did not specify the transmit buffer completely.

Transmit informationals (zero byte segments)

Number of transmit chain segments which specified a zero-byte length segment.

Transmit copies (too many segments)

Number of transmit requests that exceeded the maximum number of chain segments that the driver can handle; the driver then copied some of them to a temporary buffer so that it could transmit the packet.

Transmit copy failures

Number of transmit failures caused by too few chain segments when coalescing a request to reduce the number of segments needed.

Jumbo transmits issued

Number of transmit requests with a packet length exceeding 1514 bytes (excluding CRC).

Jumbo receives issued

Number of jumbo receive buffers allocated and given to the device.

Chained receives completed

Number of receives (jumbo packets) completed by the driver that span multiple receive buffers.

Receives discarded (bad frame)

Number of receive packets discarded by the driver due to receive error. By default, the chip does not deliver damaged receive packets to the driver, but can be asked to do so by a device-specific function.

Soft errors

Number of times errors were recovered in the driver by resetting and reinitializing the device, without notification of user applications.

Rescheduled forks (too long in fork)

Number of times that a rescheduled fork was done. In transmit and receive processing, the driver limits the amount of time spent in the fork process before rescheduling.

PHY register read/write errors

Number of errors reading or writing a PHY register.

Standard packet size (bytes)

This is the Ethernet packet size (1518 bytes).

Jumbo packet size (bytes)

This is the jumbo packet size (9018 bytes).

Requested link settings

Speed and duplex mode requested by a user.

Current link settings

Current link state.

Driver flags

Driver flags.

Driver state

Current driver state, one of the following:

0 - Driver state is undefined
1 - Driver is initializating the device
2 - Driver is running but the link is down, completing transmits with error status
4 - Driver is running but the link is down, queueing transmits for now
8 - Driver is running and the link is up, processing transmits, and receives
16 - Device is not usable

LAN_FLAGS system parameter

Current value of the LAN_FLAGS system parameter.

Interrupt delay value

Minimum delay between interrupts, in units of 256 nanoseconds.

Receive rings VA

System VA of the start of the receive rings.

Transmit rings VA

System VA of the start of the transmit rings.

LSB size

Total size of the LAN Station Block (LSB) structure.

Interupt mode

Unknown, IOSAPIC, MSI, MSIX, or IOAPIC.

Transmit time limit

Transmit time limit in seconds after which a timeout is declared.

Timer routine interval

Resolution of the transmit timer (the real timeout is Limit + Interval).

Time Stamps

Current uptime

Current system uptime.

Last reset

Last time the device was reset.

Last link state change interrupt

Last time a link state change interrupt was fielded.

Previous link state change interrupt

Previous time a link state change interrupt was fielded.

Last link up

Last time a link up transition occurred.

Last link down

Last time a link down transition occurred.

Total link uptime

Total time the link has been up.

Total link downtime

Total time the link has been down.

Time of last uplink period
Time of last uplink period-1
Time of last uplink period-2
Time of last uplink period-3
Time of last uplink period-4
Time of last uplink period-5
Time of last uplink period-6
Time of last uplink period-7
Time of last uplink period-8
Time of last uplink period-9
Time of last uplink period-10
Time of last uplink period-11
Time of last uplink period-12
Time of last uplink period-13
Time of last uplink period-14

History of the last 15 link uptimes - the length of each uptime period.

Last transmit timeout

Time of the last transmit timeout.

Last transmit timeout (averted)

Time last averted a transmit timeout.

Last soft error

Time of the last soft error.

Last CRC error

Time of the last CRC error.

Last receive error

Time of the last receive error.

Last receive error packet header

First 24 bytes of the last packet received with error status.

Last unrecognized unicast packet

Time of the last unrecognized unicast packet received.

Last unrecognized unicast packet header

First 24 bytes of the last unicast packet received that was discarded because receive address filtering did not result in a matching user.

Last unrecognized multicast packet

Time of the last unrecognized multicast packet received.

Last unrecognized multicast packet header

First 24 bytes of the last multicast packet received that was discarded because receive address filtering did not result in a matching user.

Device Registers (read/wrote)

Consists of a list of the contents significant chip registers.

PCI Config Registers

- CFID register
- CFCR register
- CFRV register
- CFLT register
- PCI_ADDRESS0 register
- PCI_ADDRESS1 register
- SUB_VNDR / SUB_ID register
- CACHE_LINE_SIZE register

Receive Buffers

Minimum buffers requested

Minimum number of receive buffers as set by default or by management request (setmode or LANCP command). This is used to determine the "Current minimum limit".

Maximum buffers requested

Maximum number of receive buffers as set by default or by management request (setmode or LANCP command). This is used to determine the "Current maximum limit".

Current minimum limit

Minimum number of receive buffers as determined by the driver and by management request, "Minimum buffers requested". The driver will not allow the number of receive buffers in its receive queues to drop below this value. This does not include receive buffers handed up to applications (and not yet returned).

Current maximum limit

Maximum number of receive buffers as determined by the driver and by management request, "Maximum buffers requested". The driver will deallocate receive buffers until the number allocated does not exceed than this maximum. When applications return receive buffers, the number may exceed this maximum at which point they are deallocated. Note that the driver allows this maximum to be exceeded somewhat to limit allocation and deallocation activity. See "Target number of buffers maximum" below.

Current number of buffers

Current number of receive buffers owned by the driver.

Target number of buffers

Desired number of receive buffers. As receive activity does not result in lost packets, this number is slowly decreased toward the "Current minimum limit". If packets are lost this number is sharply increased toward the "Current maximum limit". When the driver allocates receive buffers, it allocates them until reaching this target number.

Target number of buffers maximum

Maximum desired number of receive buffers. When an application returns a buffer to the driver, if the current number does not exceed this target maximum, the buffer is kept by the driver. Otherwise, it is deallocated.

Fork Delay (after scheduled)

To help determine whether the buffering requirements of the driver and the chip are sufficient for the system configuration, the driver records the amount of time from fork scheduled to the time the fork is actually run. The data is recorded in 10-millisecond increments from 10 to 310 milliseconds.

This data can be used in conjunction with the number of packets discarded because there were insufficient buffers to determine whether the buffering settings of the driver (minimum and maximum receive buffers) and the amount of buffering on the chip are sufficient for normal operation. If packets are being discarded, the buffering should be increased until the number of packets lost is minimal.

Transmit Time

To help understand the operation of the system, driver, and chip, the driver records the time that each transmit buffer is given to the chip. When the transmit completes, the driver calculates the elapsed time, creating a histogram of transmit times from 10 to 310 milliseconds.

Receive completion time

To help understand the operation of the system and driver, the driver records the time taken to process each receive buffer. When the receive completes, the driver records the time that processing started. Then it delivers the packet to the user application. When the user returns it, the driver calculates this completion time, creating a histogram of recieve times from 10 to 150 milliseconds.

One second timer time

```
    The driver maintains a one-second timer (that runs 4 times a second). It expects the timer routine
    to be called close to the timer interval. By looking at the time difference between when a timer is
    expected to be called and when it actually is, you can understand the operation of the system and
    driver. The variance time is used to create create a histogram of recieve times from 10 to 150
    milliseconds.

Statistics Block
```

- Good Packets Received
- Good Octets Received
- Broadcast Packets Received
- Multicast Packets Received
- Packets Received [64 Bytes]
- Packets Received [65-127 Bytes]
- Packets Received [128-255 Bytes]
- Packets Received [256-511 Bytes]
- Packets Received [512-1023 Bytes]
- Packets Received [1024 to Max Bytes]
- Good Packets Transmitted
- Good Octets Transmitted
- Packets Transmitted [64 Bytes]
- Packets Transmitted [65-127 Bytes]
- Packets Transmitted [128-255 Bytes]
- Packets Transmitted [256-511 Bytes]
- Packets Transmitted [512-1023 Bytes]
- Packets Received [1024 to Max Bytes]
- Total Octets RX
- Total Packets RX
- Total Packets TX
- CRC Errors
- Illegal Byte Errors
- Error Byte Packets
- MAC Short Packet Discards
- MAC Bad SFDs
- MAC Local Faults
- MAC Remote Faults
- Receive Length Errors
- Receive Undersizes
- Receive Fragments
- Receive Oversizes
- Receive Jabbers
- XSUM Errorx
- Link XON Received
- Link XFF Received
- Priority XON Received
- Priority XOFF Received

```
Driver Messages

    Console messages issued by the driver.
```

Intel 10Gb Device-Specific Functions

The LANCP command SET DEVICE/DEVICE_SPECIFIC=(FUNCTION="FUNC",VALUE=(V1,V2,...)) provides a mechanism to issue device-specific functions to the Intel 1gb driver. Some functions are common to many LAN drivers. But most are useful in a diagnostic context and not necessarily appropriate for formal inclusion in the LANCP command repertoire.

Note that the function name is always 4 characters in length. The command requires the SYSPRV privilege.

The definition of these functions may change from one driver version to the next.

The functions are:

```
DEVICE_SPECIFIC=FUNCTION=("RCOR",VALUE=%Xptyvalue)
DEVICE_SPECIFIC=FUNCTION=("XCOR",VALUE=%Xptyvalue)

    Corrupt a transmit or receive packet. When this function is done, the protocol type is specified,
    and if it matches on a transmit or receive packet, the next packet seen is corrupted by changing the
    last bit in the VCRP or CXB, XORing it with a 1. Then the protocol type is cleared so the corruption
    is not done again.

    For example, to corrupt a PEDRIVER transmit packet:

    $ mc lancp set dev/trace=(mask=xmtissue) eia
    $ mc lancp set dev/dev=(func="XCOR", value=%x760) eia

    For example, to corrupt a PEDRIVER receive packet:

    $ mc lancp set dev/trace=(mask=rcvdone) eia
    $ mc lancp set dev/dev=(func="RCOR", value=%x760) eia

DEVICE_SPECIFIC=FUNCTION=("LOSE",VALUE=%xaabbccdd)

    Lose some packets, by corrupting the header of the packet. Corrupt packet 0xaabbccdd, where aa is
    seconds to corrupt, bb is number to corrupt, cc is byte number 0..255, dd is what to corrupt with.

    For example, to send 32 transmit packets elsewhere (no more than 16 seconds), change the last byte
    of the destination address to 0x44:

    $ mc lancp set dev/trace=(mask=xmtissue) eia
    $ mc lancp set dev/dev=(func="LOSE", value=%x10200544) eia

    To lose just 1 packet, change the 1st byte of the packet to 0x00:

    $ mc lancp set dev/trace=(mask=xmtissue) eia
    $ mc lancp set dev/dev=func="LOSE" eia

    To lose 255 packets for the rest of the 10 millisecond tick, change the 1st byte of the packet to
    0x00:

    $ mc lancp set dev/trace=(mask=xmtissue) eia
    $ mc lancp set dev/dev=(func="LOSE",value=%xFF0000) eia

DEVICE_SPECIFIC=FUNCTION=("FCOP",VALUE=setting)

    Force DCBE transmit coalescing to force EI$COPY_TRANSMIT to get used (0 disable, 1 enable).

DEVICE_SPECIFIC=FUNCTION="DMIG"

    Disable interrupt mitigation.  Note that the TIDV register is set to the minimum value of 1 (spec
    says we can't set it to zero).

    This is used for performance testing when it is desirable to get an interrupt immediately upon
    transmit and receive completion.

DEVICE_SPECIFIC=(FUNCTION="DINT",VALUE=delayvalue)

    Change the interrupt delay value. The value is in microseconds, limited to 1000. If zero, the delay
    is disabled.

DEVICE_SPECIFIC=(FUNCTION="FLOW",VALUE=setting)

    Disable flow control (value is zero), enable flow control (value is non-zero).
```

DEVICE_SPECIFIC=FUNCTION=("PROM",VALUE=setting)

    1=Enable/0=disable force promiscuous mode, then reset device.

DEVICE_SPECIFIC=FUNCTION="SOFT"

    Force a non-fatal, i.e., soft error. The device is restarted.

DEVICE_SPECIFIC=FUNCTION="HBUG"

    Force a hardware bug (clear return rcv ring).

DEVICE_SPECIFIC=(FUNCTION="PRES",VALUE=periodicmask)

    Schedule periodic random resets. The value supplied is AND'ed with (RSCC * 69069) and if the result
    is 3, a reset is done. So, for example, if the value supplied is 7, a reset is done, on average,
    every 8 seconds. (0 disable, n enable).

    This is used to test error handling.

DEVICE_SPECIFIC=FUNCTION="FCRC"

    Send the next packet without a valid CRC.

    This is used to generate CRC errors for test purposes.

DEVICE_SPECIFIC=(FUNCTION="RBAD",VALUE=disableenable)

    If value is zero, the chip will discard bad received packets. If value is non-zero, the bad packets
    will be received and discarded by the driver. This allows the packets to be traced and inspected.

DEVICE_SPECIFIC=FUNCTION=("WCSR",VALUE=(address,val)).

    Write a value val to CSR address.

DEVICE_SPECIFIC=FUNCTION=("RCSR",VALUE=address).

    Read a CSR address and print the result.

DEVICE_SPECIFIC=FUNCTION=("WPHY",VALUE=(address,val)).

    Write a value val to PHY address.

DEVICE_SPECIFIC=FUNCTION=("ALLM",VALUE=setting)

    1=Enable/0=disable all messages flag.

DEVICE_SPECIFIC=FUNCTION=("FLIN",VALUE=setting)

    Force link indication (0 disable, 1 set link up, 2 set link down).

DEVICE_SPECIFIC=FUNCTION=("XTOB",VALUE=setting)

    1=Enable,0=disable Bugcheck on transmit timeout.

DEVICE_SPECIFIC=FUNCTION=("XLON",VALUE=setting)

    1=Enable,0=disable Bugcheck on long transmit completion.

DEVICE_SPECIFIC=FUNCTION=("FLON",VALUE=setting)

    1=Enable,0=disable Bugcheck on long fork.

DEVICE_SPECIFIC=FUNCTION=("SLON",VALUE=setting)

1=Enable,0=disable Bugcheck on long second.

DEVICE_SPECIFIC=FUNCTION="STOP"

Stop the chip, thereby inducing a transmit timeout on the next transmit request.

DEVICE_SPECIFIC=FUNCTION=("XTMO",VALUE=setting)

Set transmit timeout (2..10000 which is (1..9999)*25 * 10milliseconds.

DEVICE_SPECIFIC=FUNCTION=("AVER",VALUE=setting)

Disable transmit timeout averting if setting is 4747.

Set to 4747 to avoid trying to avert a transmit timeout. By setting XTMO to 1 to artificially cause transmit timeouts, we are causing many transmit timeouts, which exercises the transmit timeout handling. If not set to 4747, the transmit would have been found to have completed and the timeout would be counted as an 'averted' transmit timeout.

DEVICE_SPECIFIC=FUNCTION="OMSG"

Reset number of OPCOM messages.

DEVICE_SPECIFIC=FUNCTION="SYSI"

Send the periodic System ID messages now instead of waiting until expiry of the 8-12 minute System ID timer.

This is a convenient way to get the driver to send 2 packets.

DEVICE_SPECIFIC=FUNCTION="SYSZ"

Disable the sending of periodic System ID messages.

This is used to disable transmits that might distract from other traffic you are looking at.

DEVICE_SPECIFIC=(FUNCTION="WAIT",VALUE=timevalue)

Wait the specified time in nanoseconds, at IPL 8 with the driver port lock held.

This is a convenient way to introduce a delay at IPL 8.

The timevalue is 32 bits, limiting the wait to 4 seconds.

DEVICE_SPECIFIC=FUNCTION="CVME"
DEVICE_SPECIFIC=FUNCTION="SVME"

VME bit set/clear. If clear, the VLAN tag will not be stripped from the packet so it can be seen in the receive buffer.

DEVICE_SPECIFIC=FUNCTION="ARST"

Restart auto-negotiation.

This is used for link handling testing.

DEVICE_SPECIFIC=FUNCTION="FLSC"

Force a link state change interrupt.

DEVICE_SPECIFIC=FUNCTION=("DELA",VALUE=n)

Delay the next transmit by n 10-millisecond ticks.

DEVICE_SPECIFIC=FUNCTION="XDEL"

```
Call ini$brk.
```