NAME

iflib - Network Interface Driver Framework

SYNOPSIS

device pci
device iflib

DESCRIPTION

iflib is a framework for network interface drivers for FreeBSD. It is designed to remove a large amount of the boilerplate that is often needed for modern network interface devices, allowing driver authors to focus on the specific code needed for their hardware. This allows for a shared set of sysctl(8) names, rather than each driver naming them individually.

SYSCTL VARIABLES

These variables must be set before loading the driver, either via loader.conf(5) or through the use of kenv(1). They are all prefixed by *dev.X.Y.iflib.* where X is the driver name, and Y is the instance number.

override_nrxds

Override the number of RX descriptors for each queue. The value is a comma separated list of positive integers. Some drivers only use a single value, but others may use more. These numbers must be powers of two, and zero means to use the default. Individual drivers may have additional restrictions on allowable values. Defaults to all zeros.

override_ntxds

Override the number of TX descriptors for each queue. The value is a comma separated list of positive integers. Some drivers only use a single value, but others may use more. These numbers must be powers of two, and zero means to use the default. Individual drivers may have additional restrictions on allowable values. Defaults to all zeros.

override_qs_enable

When set, allows the number of transmit and receive queues to be different. If not set, the lower of the number of TX or RX queues will be used for both.

override_nrxqs

Set the number of RX queues. If zero, the number of RX queues is derived from the number of cores on the socket connected to the controller. Defaults to 0.

override_ntxqs

Set the number of TX queues. If zero, the number of TX queues is derived from the number of

cores on the socket connected to the controller.

disable_msix
Disables MSI-X interrupts for the device.

core_offset
Specifies a starting core offset to assign queues to. If the value is unspecified or 65535, cores are assigned sequentially across controllers.

separate_txrx
Requests that RX and TX queues not be paired on the same core. If this is zero or not set, an RX and TX queue pair will be assigned to each core. When set to a non-zero value, TX queues are assigned to cores following the last RX queue.

These sysctl(8) variables can be changed at any time:

tx_abdicate
Controls how the transmit ring is serviced. If set to zero, when a frame is submitted to the transmission ring, the same task that is submitting it will service the ring unless there's already a task servicing the TX ring. This ensures that whenever there is a pending transmission, the transmit ring is being serviced. This results in higher transmit throughput. If set to a non-zero value, task returns immediately and the transmit ring is serviced by a different task. This returns control to the caller faster and under high receive load, may result in fewer dropped RX frames.

rx_budget
Sets the maximum number of frames to be received at a time. Zero (the default) indicates the default (currently 16) should be used.

There are also some global sysctls which can change behaviour for all drivers, and may be changed at any time.

net.iflib.min_tx_latency
If this is set to a non-zero value, iflib will avoid any attempt to combine multiple transmits, and notify the hardware as quickly as possible of new descriptors. This will lower the maximum throughput, but will also lower transmit latency.

net.iflib.no_tx_batch
Some NICs allow processing completed transmit descriptors in batches. Doing so usually increases the transmit throughput by reducing the number of transmit interrupts. Setting this to a non-zero value will disable the use of this feature.

These sysctl(8) variables are read-only:

driver_version
A string indicating the internal version of the driver.

There are a number of queue state sysctl(8) variables as well:

txqZ The following are repeated for each transmit queue, where Z is the transmit queue instance number:

r_abdications
Number of consumer abdications in the MP ring for this queue. An abdication occurs on every ring submission when tx_abdicate is true.

r_restarts
Number of consumer restarts in the MP ring for this queue. A restart occurs when an attempt to drain a non-empty ring fails, and the ring is already in the STALLED state.

r_stalls
Number of consumer stalls in the MP ring for this queue. A stall occurs when an attempt to drain a non-empty ring fails.

r_starts
Number of normal consumer starts in the MP ring for this queue. A start occurs when the MP ring transitions from IDLE to BUSY.

r_drops
Number of drops in the MP ring for this queue. A drop occurs when there is an attempt to add an entry to an MP ring with no available space.

r_enqueues
Number of entries which have been enqueued to the MP ring for this queue.

ring_state
MP (soft) ring state. This privides a snapshot of the current MP ring state, including the producer head and tail indexes, the consumer index, and the state. The state is one of "IDLE", "BUSY", "STALLED", or "ABDICATED".

txq_cleaned
The number of transmit descriptors which have been reclaimed. Total cleaned.

txq_processed
: The number of transmit descriptors which have been processed, but may not yet have been reclaimed.

txq_in_use
: Descriptors which have been added to the transmit queue, but have not yet been cleaned. This value will include both untransmitted descriptors as well as descriptors which have been processed.

txq_cidx_processed
: The transmit queue consumer index of the next descriptor to process.

txq_cidx
: The transmit queue consumer index of the oldest descriptor to reclaim.

txq_pidx
: The transmit queue producer index where the next descriptor to transmit will be inserted.

no_tx_dma_setup
: Number of times DMA mapping a transmit mbuf failed for reasons other than EFBIG.

txd_encap_efbig
: Number of times DMA mapping a transmit mbuf failed due to requiring too many segments.

tx_map_failed
: Number of times DMA mapping a transmit mbuf failed for any reason (sum of no_tx_dma_setup and txd_encap_efbig)

no_desc_avail
: Number of times a descriptor couldn’t be added to the transmit ring because the transmit ring was full.

mbuf_defrag_failed
: Number of times both m_collapse(9) and m_defrag(9) failed after an EFBIG error result from DMA mapping a transmit mbuf.

m_pullups
: Number of times m_pullup(9) was called attempting to parse a header.

mbuf_defrag
Number of times m_defrag(9) was called.

rxqZ The following are repeated for each receive queue, where Z is the receive queue instance number:

rxq_fl0.credits
Credits currently available in the receive ring.

rxq_fl0.cidx
Current receive ring consumer index.

rxq_fl0.pidx
Current receive ring producer index.

Additional OIDs useful for driver and iflib development are exposed when the INVARIANTS and/or WITNESS options are enabled in the kernel.

SEE ALSO

iflib(9)

HISTORY

This framework was introduced in FreeBSD 11.0.