

NAME

iflibdi - Device Independent Configuration Functions

SYNOPSIS

```
#include <ifdi_if.h>
```

Device Independent Functions

int

```
iflib_device_attach(device_t dev);
```

int

```
iflib_device_detach(device_t dev);
```

int

```
iflib_device_suspend(device_t dev);
```

int

```
iflib_device_resume(device_t dev);
```

int

```
iflib_device_register(device_t dev, void *softc, if_shared_ctx_t sctx, if_ctx_t *ctxp);
```

int

```
iflib_device_deregister(if_ctx_t ctx);
```

int

```
iflib_irq_alloc(if_ctx_t ctx, if_irq_t irq_info, int rid, driver_filter_t filter, void *filter_arg,  
          driver_intr_t handler, void *arg, char *name);
```

int

```
iflib_irq_alloc_generic(if_ctx_t ctx, if_irq_t irq, int rid, intr_type_t type, driver_filter_t *filter,  
          void *filter_arg, int qid, char *name);
```

void

```
iflib_led_create(if_ctx_t ctx);
```

void

```
iflib_tx_intr_deferred(if_ctx_t ctx, int txqid);
```

void

```
iflib_rx_intr_deferred(if_ctx_t ctx, int rxqid);
```

void

```
iflib_link_intr_deferred(if_ctx_t ctx);
```

void

```
iflib_link_state_change(if_ctx_t ctx, int linkstate);
```

void

```
iflib_add_int_delay_sysctl(if_ctx_t ctx, const char *, const char *, if_int_delay_info_t, int, int);
```

Global Variables

extern struct if_txx

DATA STRUCTURES

The *if_ctx_t* Structure is the device independent data structure that contains statistics and identifying information used to transmit and receive data packets. The interface is associated with an array of queues assigned sequentially. Each queue has its own transmit (*iflib_txq_t*) and receive (*iflib_rxq_t*) queue. The transmit queue is used to hold packets while the interface is in the process of sending another. The receive queue is used to receive packets that are awaiting processing.

The *if_ctx_t* Structure

The fields of *struct if_ctx_t* are as follows:

<i>if_softc</i>	(<i>void</i>) A pointer to the driver's private state block.
<i>ifc_dev</i>	(<i>device_t</i>) The underlying device structure.
<i>ifc_ip</i>	(<i>if_t</i>) A link back to the interface structure
<i>ifc_cpus</i>	(<i>cpuset_t</i>)
<i>ifc_mutex</i>	(<i>struct mtx</i>) Mutex lock used to maintain data integrity
<i>ifc_mtx_name</i>	(<i>char *</i>) The name of the mutex
<i>ifc_txqs</i>	(<i>iflib_txq_t</i>) Device independent transmit queue maintained internally by iflib
<i>ifc_rxqs</i>	(<i>iflib_rxq_t</i>) Device independent receive queue maintained internally by iflib

- ifc_qsets* (*iflib_qset_t*) Output queue that contains a single transmit (*ifc_txq_t*) and receive (*ifc_rxq_t*) queue
- ifc_if_flags* (*uint32_t*) Flags describing the operational parameter of the interface
- ifc_in_detach* (*int*)
- ifc_link_state* (*int*) Describes the current link state of the Ethernet interface. Its possible values are either active or inactive.
- ifc_link_irq* (*int*)
- ifc_vlan_attach_event*
(*eventhandler_tag*)
- ifc_vlan_detach_event*
(*eventhandler_tag*)
- ifc_pause_frames*
(*int*)
- ifc_watchdog_events*
(*int*)
- ifc_mac* (*uint8_t*)
- ifc_msix_mem*
(*struct resource **)
- ifc_legacy_irq*
(*struct if_irq*)
- ifc_admin_task*
(*struct grouptask*) Taskqueue task scheduled for link state change events of the interface
- ifc_filter_info* (*struct iflib_filter_info*) Statistics and information relating to the interface device filter
- ifc_media* (*struct ifmedia*)

ifc_txx (struct *if_txx*)

FUNCTIONS

The above named functions are found exclusively in `iflib`. They are independent of the underlying hardware type or configuration.

Device Independent Functions

`iflib_device_attach()`

Function initiates a device registration with the `iflib` framework. It calls the `iflib_register` function, which is responsible for allocating and initializing the `if_ctx_t` structure.

`iflib_device_detach()`

Shutdown and detach the device. Unregister vlan events, drain any dependent tasks, and release irq, pci, and msix memory.

`iflib_device_suspend()`

Suspend a device by calling the device dependent suspend function and `bus_generic_suspend`.

`iflib_device_resume()`

Resume a device by calling the device dependent resume function, the `iflib_init_locked` function, and `bus_generic_resume`.

`iflib_device_register()`

Register a device with the `iflib` framework. Allocate and initialize the `if_ctx_t` structure. Setup and initialize the MSI or MSI/X interrupt queues if necessary. Allocate memory for queues and qset structure setup.

`iflib_irq_alloc()`

Allocate an interrupt resource for a given rid value with an associated filter and handler function.

`iflib_irq_alloc_generic()`

Performs the same function as `iflib_device_irq_alloc` along with the additional functionality of adding a taskgroup. The data fields and callback function are determined by the type of interrupt, such as `IFLIB_INTR_TX`, `IFLIB_INTR_RX`, and `IFLIB_INTR_ADMIN`.

`iflib_led_create()`

Calls `led_create` to initialize the `ctx->ifc_led_dev` field

`iflib_tx_intr_deferred()`

Calls `GROUPTASK_ENQUEUE` to enqueue the transfer queues `ift_task`.

iflib_rx_intr_deferred()

Calls GROUPTASK_ENQUEUE to enqueue the receive queues ifr_task.

iflib_link_intr_deferred()

Calls GROUPTASK_ENQUEUE to enqueue the link task

iflib_link_state_change()

Change the interface link status to either LINK_STATE_UP or LINK_STATE_DOWN as specified by the second argument to the function.

Interface Link States The following link states are currently defined:

LINK_STATE_UP

The link is up.

LINK_STATE_DOWN

The link is down.

iflib_add_int_delay_sysctl()

Modifies settings to user defined values for a given set of variables.

SEE ALSO

iflibdd(9), iflibtxrx(9)

AUTHORS

This manual page was written by Nicole Graziano