

AmigaMail

COLLABORATORS

	<i>TITLE :</i> AmigaMail		
<i>ACTION</i>	<i>NAME</i>	<i>DATE</i>	<i>SIGNATURE</i>
WRITTEN BY		July 19, 2024	

REVISION HISTORY

NUMBER	DATE	DESCRIPTION	NAME

Contents

1	AmigaMail	1
1.1	Developer Support Package / sana2device.doc	1
1.2	sana2.device/AbortIO	1
1.3	sana2.device/CloseDevice	2
1.4	sana2.device/CMD_CLEAR	3
1.5	sana2.device/CMD_FLUSH	3
1.6	sana2.device/CMD_INVALID	4
1.7	sana2.device/CMD_READ	4
1.8	sana2.device/CMD_RESET	5
1.9	sana2.device/CMD_START	5
1.10	sana2.device/CMD_STOP	6
1.11	sana2.device/CMD_UPDATE	6
1.12	sana2.device/CMD_WRITE	7
1.13	sana2.device/OpenDevice	7
1.14	sana2.device/S2_ADDMULTICASTADDRESS	8
1.15	sana2.device/S2_BROADCAST	9
1.16	sana2.device/S2_CONFIGINTERFACE	10
1.17	sana2.device/S2_DELMULTICASTADDRESS	10
1.18	sana2.device/S2_DEVICEQUERY	11
1.19	sana2.device/S2_GETGLOBALSTATS	12
1.20	sana2.device/S2_GETSPECIALSTATS	13
1.21	sana2.device/S2_GETSTATIONADDRESS	14
1.22	sana2.device/S2_GETTYPESTATS	14
1.23	sana2.device/S2_MULTICAST	15
1.24	sana2.device/S2_OFFLINE	16
1.25	sana2.device/S2_ONEEVENT	16
1.26	sana2.device/S2_ONLINE	17
1.27	sana2.device/S2_READORPHAN	18
1.28	sana2.device/S2_TRACKTYPE	19
1.29	sana2.device/S2_UNTRACKTYPE	19

Chapter 1

AmigaMail

1.1 Developer Support Package / sana2device.doc

TABLE OF CONTENTS

```
sana2.device/AbortIO
sana2.device/CloseDevice
sana2.device/CMD_CLEAR
sana2.device/CMD_FLUSH
sana2.device/CMD_INVALID
sana2.device/CMD_READ
sana2.device/CMD_RESET
sana2.device/CMD_START
sana2.device/CMD_STOP
sana2.device/CMD_UPDATE
sana2.device/CMD_WRITE
sana2.device/OpenDevice
sana2.device/S2_ADDMULTICASTADDRESS
sana2.device/S2_BROADCAST
sana2.device/S2_CONFIGINTERFACE
sana2.device/S2_DELMULTICASTADDRESS
sana2.device/S2_DEVICEQUERY
sana2.device/S2_GETGLOBALSTATS
sana2.device/S2_GETSPECIALSTATS
sana2.device/S2_GETSTATIONADDRESS
sana2.device/S2_GETTYPESTATS
sana2.device/S2_MULTICAST
sana2.device/S2_OFFLINE
sana2.device/S2_ONEVENT
sana2.device/S2_ONLINE
sana2.device/S2_READORPHAN
sana2.device/S2_TRACKTYPE
sana2.device/S2_UNTRACKTYPE
```

1.2 sana2.device/AbortIO

NAME

AbortIO -- Remove an existing device request.

SYNOPSIS

```
error = AbortIO(Sana2Req)
D0          A1
```

```
LONG AbortIO(struct IOSana2Req *);
```

FUNCTION

This is an `exec.library` call.

This function aborts an `ioRequest`. If the request is active, it may or may not be aborted. If the request is queued it is removed. The request will be returned in the same way as if it had normally completed. You must `WaitIO()` after `AbortIO()` for the request to return.

INPUTS

Sana2Req - Sana2Req to be aborted.

RESULTS

error - Zero if the request was aborted, non-zero otherwise.
 `io_Error` in Sana2Req will be set to `IOERR_ABORTED`
 if it was aborted.

NOTES

SEE ALSO

`exec.library/AbortIO()`, `exec.library/WaitIO()`

BUGS

1.3 sana2.device/CloseDevice

NAME

CloseDevice -- Close the device.

SYNOPSIS

```
CloseDevice(Sana2Req)
          A1
```

```
void CloseDevice(struct IOSana2Req *);
```

FUNCTION

This function is called by `exec.library/CloseDevice()`.

This function performs whatever cleanup is required at device closes.

Note that all `IORequests` MUST be complete before closing. If any are pending, your program must `AbortIO()` then `WaitIO()` each outstanding `IORequest` to complete them.

INPUTS

Sana2Req - Pointer to `IOSana2Req` initialized by `OpenDevice()`.

NOTES

SEE ALSO

`exec.library/CloseDevice()`, `exec.library/OpenDevice()`

BUGS

1.4 sana2.device/CMD_CLEAR

NAME

Clear -- Clear internal network interface read buffers.

FUNCTION

There are no device internal buffers, so CMD_CLEAR does not apply to this class of device.

IO REQUEST

`ios2_Command` - CMD_CLEAR.

RESULTS

`ios2_Error` - IOERR_NOCMD.

NOTES

SEE ALSO

BUGS

1.5 sana2.device/CMD_FLUSH

NAME

Flush -- Clear all queued I/O requests for the SANA-II device.

FUNCTION

This command aborts all I/O requests in both the read and write request queues of the device. All pending I/O requests are returned with an error message (IOERR_ABORTED). CMD_FLUSH does not affect active requests.

IO REQUEST

`ios2_Command` - CMD_FLUSH.

RESULTS

`ios2_Error` - Zero if successful; non-zero otherwise.

NOTES

SEE ALSO

BUGS

1.6 sana2.device/CMD_INVALID

NAME

Invalid -- Return with error IOERR_NOCMD.

FUNCTION

This command causes device driver to reply with an error IOERR_NOCMD as defined in <exec/errors.h> indicating the command is not supported.

IO REQUEST

ios2_Command - CMD_INVALID.

RESULTS

ios2_Error - IOERR_NOCMD.

NOTES

SEE ALSO

BUGS

Not known to be useful.

1.7 sana2.device/CMD_READ

NAME

Read -- Get a packet from the network.

FUNCTION

Get the next packet available of the requested packet type. The data copied (via a call to the requestor-provided CopyToBuffer function) into ios2_Data is normally the Data Link Layer packet data only. If bit SANA2IOB_RAW is set in ios2_Flags, then the entire physical frame will be returned.

Unlike most Exec devices, SANA-II device drivers do not have internal buffers. If you wish to read data from a SANA-II device you should have multiple CMD_READ requests pending at any given time. The functions provided by you the requestor will be used for any incoming packets of the type you've requested. If no read requests are outstanding for a type which comes in and no read_orphan requests are outstanding, the packet will be lost.

IO REQUEST

ios2_Command - CMD_READ
ios2_Flags - Supported flags are:
 SANA2IOB_RAW
 SANA2IOB_QUICK
ios2_PacketType - Packet type desired.
ios2_Data - Abstract data structure to hold packet data.

RESULTS

ios2_Error - Zero if successful; non-zero otherwise.
ios2_WireError - More specific error number.
ios2_Flags - The following flags may be returned:

```
        SANA2IOB_RAW
        SANA2IOB_BCAST
        SANA2IOB_MCAST
ios2_SrcAddr   - Source interface address of packet.
ios2_DstAddr   - Destination interface address of packet.
ios2_DataLength - Length of packet data.
ios2_Data      - Abstract data structure which packet data is
                  contained in.
```

NOTES

The driver may not directly examine or modify anything pointed to by `ios2_Data`. It *must* use the requester-provided functions to access this data.

SEE ALSO

`S2_READORPHAN`, `CMD_WRITE`, `any_protocol/CopyToBuffer`

BUGS

1.8 sana2.device/CMD_RESET

NAME

`Reset` -- Reset the network interface to initialized state.

FUNCTION

Currently, SANA-II devices can only be configured once (with `CMD_CONFIGINTERFACE`) and cannot be re-configured, hence, `CMD_RESET` does not apply to this class of device.

IO REQUEST

`ios2_Command` - `CMD_RESET`.

RESULTS

`ios2_Error` - `IOERR_NOCMD`.

NOTES

SEE ALSO

BUGS

1.9 sana2.device/CMD_START

NAME

`Start` -- Restart device operation.

FUNCTION

There is no way for the driver to keep queuing requests without servicing them, so `CMD_STOP` does not apply to this class of device. `S2_OFFLINE` and `S2_ONLINE` do perform a similar function to `CMD_STOP` and `CMD_START`

IO REQUEST
 ios2_Command - CMD_START.

RESULTS
 ios2_Error - IOERR_NOCMD.

NOTES

SEE ALSO
 S2_ONLINE, S2_OFFLINE

BUGS

1.10 sana2.device/CMD_STOP

NAME
 Stop -- Pause device operation.

FUNCTION
 There is no way for the driver to keep queuing requests without servicing them, so CMD_STOP does not apply to this class of device. S2_OFFLINE and S2_ONLINE do perform a similar function to CMD_STOP and CMD_START

IO REQUEST
 ios2_Command - CMD_STOP.

RESULTS
 ios2_Error - IOERR_NOCMD.

NOTES

SEE ALSO
 S2_ONLINE, S2_OFFLINE

BUGS

1.11 sana2.device/CMD_UPDATE

NAME
 Update -- Force packets out to device.

FUNCTION
 Since there are no device internal buffers, CMD_UPDATE does not apply to this class of device.

IO REQUEST
 ios2_Command - CMD_UPDATE.

RESULTS
 ios2_Error - IOERR_NOCMD.

NOTES

SEE ALSO

BUGS

1.12 sana2.device/CMD_WRITE

NAME

Write -- Send packet to the network.

FUNCTION

This command causes the packet to be sent to the specified network interface. Normally, appropriate packet header and trailer information will be added to the packet data when it is sent. If bit SANA2IOB_RAW is set in io_Flags, then the ios2_Data is assumed to contain an entire physical frame and will be sent (copied to the wire via CopyFromBuffer()) unmodified.

Note that the device should not check to see if the destination address is on the local hardware. Network protocols should realize that the packet has a local destination long before it gets to a SANA-II driver.

IO REQUEST

ios2_Command - CMD_WRITE.
ios2_Flags - Supported flags are:
 SANA2IOB_RAW
 SANA2IOB_QUICK
ios2_PacketType - Packet type to send.
ios2_DstAddr - Destination interface address for this packet.
ios2_DataLength - Length of the Data to be sent.
ios2_Data - Abstract data structure which packet data is
 contained in.

RESULTS

ios2_Error - Zero if successful; non-zero otherwise.
ios2_WireError - More specific error number.

NOTES

The driver may not directly examine or modify anything pointed to by ios2_Data. It *must* use the requester-provided functions to access this data.

SEE ALSO

CMD_READ, S2_BROADCAST, S2_MULTICAST, any_protocol/CopyFromBuffer

BUGS

1.13 sana2.device/OpenDevice

NAME

Open -- Request an opening of the network device.

SYNOPSIS

```
error = OpenDevice(unit, IOSana2Req, flags)
```

```
D0          D0      A1          D1
```

```
BYTE OpenDevice(ULONG, struct IOSana2Req *, ULONG);
```

FUNCTION

This function is called by `exec.library OpenDevice()`.

This function performs whatever initialization is required per device open and initializes the `Sana2Req` for use by the device.

INPUTS

```
unit          - Device unit to open.
Sana2Req      - Pointer to IOSana2Req structure to be initialized by
                the sana2.device.
flags         - Supported flags are:
                SANA2OPB_MINE
                SANA2OPB_PROM
ios2_BufferManagement - A pointer to a tag list containing
                pointers to buffer management functions.
```

RESULTS

```
error         - same as io_Error
io_Error      - Zero if successful; non-zero otherwise.
io_Device     - A pointer to whatever device will handle the calls
                for this unit. This pointer may be different
                depending on what unit is requested.
ios2_BufferManagement - A pointer to device internal information
                used to call buffer management functions.
```

NOTES

SEE ALSO

`exec.library/OpenDevice()`, `exec.library/CloseDevice()`

BUGS

1.14 sana2.device/S2_ADDMULTICASTADDRESS

NAME

AddMulticastAddress -- Enable an interface multicast address.

FUNCTION

This command causes the device driver to enable multicast packet reception for the requested address.

IO REQUEST

```
ios2_Command  - S2_ADDMULTICASTADDRESS.
ios2_SrcAddr  - Multicast address to enable.
```

RESULTS

ios2_Error - Zero if successful; non-zero otherwise.
 ios2_WireError - More specific error number.

NOTES

Multicast addresses are added globally -- anyone using the device may receive packets as a result of any multicast address which has been added for the device.

Since multicast addresses are not "bound" to a particular packet type, each enabled multicast address has an "enabled" count associated with it so that if two protocols add the same multicast address and later one removes it, it is still enabled until the second removes it.

SEE ALSO

S2_MULTICAST, S2_DELMULTICASTADDRESS

BUGS

1.15 sana2.device/S2_BROADCAST

NAME

Broadcast -- Broadcast a packet on network.

FUNCTION

This command works the same as CMD_WRITE except that it also performs whatever special processing of the packet is required to do a broadcast send. The actual broadcast mechanism is necessarily network/interface/device specific.

IO REQUEST

ios2_Command - S2_BROADCAST.
 ios2_Flags - Supported flags are:
 SANA2IOB_RAW
 SANA2IOB_QUICK
 ios2_PacketType - Packet type to send.
 ios2_DataLength - Length of the Data to be sent.
 ios2_Data - Abstract data structure which packet data is
 contained in.

RESULTS

ios2_DstAddr - The contents of this field are to be
 considered trash upon return of the IOReq.
 ios2_Error - Zero if successful; non-zero otherwise.
 This command can fail for many reasons and
 is not supported by all networks and/or
 network interfaces.
 ios2_WireError - More specific error number.

NOTES

The DstAddr field may be trashed by the driver because this function may be implemented by filling DstAddr with a broadcast address and internally calling CMD_WRITE.

SEE ALSO

CMD_WRITE, S2_MULTICAST

BUGS

1.16 sana2.device/S2_CONFIGINTERFACE

NAME

ConfigInterface -- Configure the network interface.

FUNCTION

This command causes the device driver to initialize the interface hardware and to set the network interface address to the address in ios2_SrcAddr. This command can only be executed once and, if successful, will leave the driver and network interface fully operational and the network interface in ios2_SrcAddr.

To set the interface address to the factory address, the network management software must use GetStationAddress first and then call ConfigInterface with the result. If there is no factory address then the network software must pick an address to use.

Until this command is executed the device will not listen for any packets on the hardware.

IO REQUEST

ios2_Command - S2_CONFIGINTERFACE.
ios2_Flags - Supported flags are:
 SANA2IOB_QUICK
ios2_SrcAddr - Address for this interface.

RESULTS

ios2_Error - Zero if successful; non-zero otherwise.
ios2_WireError - More specific error number.
ios2_SrcAddr - Address of this interface as configured.

NOTES

Some networks have the interfaces choose a currently unused interface address each time the interface is initialized. The caller must check ios2_SrcAddr for the actual interface address after configuring the interface.

SEE ALSO

S2_GETSTATIONADDRESS

BUGS

1.17 sana2.device/S2_DELMULTICASTADDRESS

NAME

DelMultiCastAddress -- Disable an interface multicast address.

FUNCTION

This command causes device driver to disable multicast packet reception for the requested address.

It is an error to disable a multicast address that is not enabled.

IO REQUEST

ios2_Command - S2_DELMULTICASTADDRESS
ios2_SrcAddr - Multicast address to disable.

RESULTS

ios2_Error - Zero if successful; non-zero otherwise.
ios2_WireError - More specific error number.

NOTES

Multicast addresses are added globally -- anyone using the device may receive packets as a result of any multicast address which has been added for the device.

Since multicast addresses are not "bound" to a particular packet type, each enabled multicast address has an "enabled" count associated with it so that if two protocols add the same multicast address and later one removes it, it is still enabled until the second removes it.

SEE ALSO

S2_ADDMULTICASTADDRESS

BUGS

1.18 sana2.device/S2_DEVICEQUERY

NAME

DeviceQuery -- Return parameters for this network interface.

FUNCTION

This command causes the device driver to report information about the device. Up to SizeAvailable bytes of the information is copied into a buffer pointed to by ios2_StatData. The format of the data is as follows:

```
struct Sana2DeviceQuery
{
/*
** Standard information
*/
    ULONG SizeAvailble; /* bytes available */
    ULONG SizeSupplied; /* bytes supplied */
    LONG  DevQueryFormat; /* this is type 0 */
    LONG  DeviceLevel;    /* this document is level 0 */

/*
** Common information
*/
    UWORD AddrFieldSize; /* address size in bits */
    ULONG MTU;           /* maximum packet data size */
}
```

```

        LONG   bps;                /* line rate (bits/sec) */
        LONG   HardwareType;       /* what the wire is */

/*
** Format specific information
*/
};

```

The SizeAvailable specifies the number of bytes that the caller is prepared to accomodate, including the standard information fields.

SizeSupplied is the number of bytes actually supplied, including the standard information fields, which will not exceed SizeAvailable.

<devices/sana2.h> includes constants for these values. If your hardware does not have a number assigned to it, you must contact CATS to get a hardware number.

IO REQUEST

```

ios2_Command      - S2_DEVICEQUERY.
ios2_StatData     - Pointer to Sana2DeviceQuery structure to fill in.

```

RESULTS

```

ios2_Error        - Zero if successful; non-zero otherwise.
ios2_WireError    - More specific error number.

```

NOTES

SEE ALSO

BUGS

1.19 sana2.device/S2_GETGLOBALSTATS

NAME

GetGlobalStats -- Get interface accumulated statistics.

FUNCTION

This command causes the device driver to retrieve various global runtime statistics for this network interface. The format of the data returned is as follows:

```

struct Sana2DeviceStats
{
    ULONG PacketsReceived;
    ULONG PacketsSent;
    ULONG BadData;
    ULONG Overruns;
    ULONG UnknownTypesReceived;
    ULONG Reconfigurations;
    timeval LastStart;
};

```

IO REQUEST

```

ios2_Command      - S2_GETGLOBALSTATS.
ios2_StatData     - Pointer to Sana2DeviceStats structure to fill.

```

RESULTS

```

ios2_Error        - Zero if successful; non-zero otherwise.
ios2_WireError    - More specific error number.

```

NOTES

SEE ALSO

S2_GETSPECIALSTATS

BUGS

1.20 sana2.device/S2_GETSPECIALSTATS

NAME

GetSpecialStats -- Get network type specific statistics.

FUNCTION

This function returns statistics which are specific to the type of network medium this driver controls. For example, this command could return statistics common to all Ethernets which are not common to all network mediums in general.

The supplied Sana2SpecialStatData structure is given below:

```

struct Sana2SpecialStatData
{
    ULONG RecordCountMax;
    ULONG RecordCountSupplied;
    struct Sana2StatRecord[RecordCountMax];
};

```

The format of the data returned is:

```

struct Sana2StatRecord
{
    ULONG Type;        /* commodore registered */
    LONG Count;        /* the stat itself */
    char *String;      /* null terminated */
};

```

The RecordCountMax field specifies the number of records that the caller is prepared to accomodate.

RecordCountSupplied is the number of record actually supplied which will not exceed RecordCountMax.

IO REQUEST

```

ios2_Command      - S2_GETSPECIALSTATS.
ios2_StatData     - Pointer to a Sana2SpecialStatData structure to fill.
                   RecordCountMax must be initialized.

```

RESULTS

ios2_Error - Zero if successful; non-zero otherwise.
ios2_WireError - More specific error number.

NOTES

Commodore will maintain registered statistic Types.

SEE ALSO

S2_GETGLOBALSTATS, <devices/sana2specialstats.h>

BUGS

1.21 sana2.device/S2_GETSTATIONADDRESS

NAME

GetStationAddress -- Get default and interface address.

FUNCTION

This command causes the device driver to copy the current interface address into ios2_SrcAddr, and to copy the factory default station address (if any) into ios2_DstAddr.

IO REQUEST

ios2_Command - S2_GETSTATIONADDRESS.

RESULTS

ios2_Error - Zero if successful; non-zero otherwise.
ios2_WireError - More specific error number.
ios2_SrcAddr - Current interface address.
ios2_DstAddr - Default interface address (if any).

NOTES

SEE ALSO

S2_CONFIGINTERFACE

BUGS

1.22 sana2.device/S2_GETTYPESTATS

NAME

GetTypeStats -- Get accumulated type specific statistics.

FUNCTION

This command causes the device driver to retrieve various packet type specific runtime statistics for this network interface. The format of the data returned is as follows:

```
struct Sana2TypeStatData
{
    LONG PacketsSent;
    LONG PacketsReceived;
    LONG BytesSent;
```

```

        LONG BytesReceived;
        LONG PacketsDropped;
    };

```

IO REQUEST

```

    ios2_Command      - S2_GETTYPESTATS.
    ios2_PacketType   - Packet type of interest.
    ios2_StatData     - Pointer to TypeStatData structure to fill in.

```

RESULTS

```

    ios2_Error        - Zero if successful; non-zero otherwise.
    ios2_WireError    - More specific error number.

```

NOTES

Statistics for a particular packet type are only available while that packet type is being ``tracked``.

SEE ALSO

S2_TRACKTYPE, S2_UNTRACKTYPE

BUGS

1.23 sana2.device/S2_MULTICAST

NAME

Multicast -- Multicast a packet on network.

FUNCTION

This command works the same as CMD_WRITE except that it also performs whatever special processing of the packet is required to do a multicast send. The actual multicast mechanism is necessarily network/interface/device specific.

IO REQUEST

```

    ios2_Command      - S2_MULTICAST.
    ios2_Flags        - Supported flags are:
                        SANA2IOB_RAW
                        SANA2IOB_QUICK
    ios2_PacketType   - Packet type to send.
    ios2_DstAddr      - Destination interface address for this packet.
    ios2_DataLength   - Length of the Data to be sent.
    ios2_Data         - Abstract data structure which packet data is
                        contained in.

```

RESULTS

```

    ios2_Error        - Zero if successful; non-zero otherwise.
                        This command can fail for many reasons and
                        is not supported by all networks and/or
                        network interfaces.
    ios2_WireError    - More specific error number.

```

NOTES

The address supplied in ios2_DstAddr will be sanity checked (if possible) by the driver. If the supplied address fails this sanity check, the multicast request will fail immediately with ios2_Error

set to S2WERR_BAD_MULTICAST.

Another Amiga will not receive a multicast packet unless it has had the particular multicast address being used S2_ADDMULTICASTADDRESS'd.

SEE ALSO

CMD_WRITE, S2_BROADCAST, S2_ADDMULTICASTADDRESS

BUGS

1.24 sana2.device/S2_OFFLINE

NAME

Offline -- Remove interface from service.

FUNCTION

This command removes a network interface from service.

IO REQUEST

ios2_Command - S2_OFFLINE.

RESULTS

ios2_Error - Zero if successful; non-zero otherwise.

ios2_WireError - More specific error number.

NOTES

Aborts all pending reads and writes with ios2_Error set to S2ERR_OUTOFSERVICE.

While the interface is offline, all read, writes and any other command that touches interface hardware will be rejected with ios2_Error set to S2ERR_OUTOFSERVICE.

This command is intended to permit a network interface to be tested on an otherwise live system.

SEE ALSO

S2_ONLINE

BUGS

1.25 sana2.device/S2_ONEVENT

NAME

OnEvent -- Return when specified event occurs.

FUNCTION

This command returns when a particular event condition has occurred on the network or this network interface.

IO REQUEST

ios2_Command - S2_ONEVENT.

ios2_Flags - Supported flags are:
 SANA2IOB_QUICK
ios2_WireError - Mask of event(s) to wait for
 (from <devices/sana2.h>).

RESULTS

ios2_Error - Zero if successful; non-zero otherwise.
ios2_WireError - Mask of events that occurred.

NOTES

If this device driver does not understand the specified event condition(s) then the command returns immediately with ios2_Req.io_Error set to S2_ERR_NOT_SUPPORTED and ios2_WireError S2WERR_BAD_EVENT. A successful return will have ios2_Error set to zero ios2_WireError set to the event number.

All pending requests for a particular event will be returned when that event occurs.

All event types that cover a particular condition are returned when that condition occurs. For instance, if an error is returned by a buffer management function during receive processing, events of types S2EVENT_ERROR, S2EVENT_RX and S2EVENT_BUFF would be returned if pending.

Types ONLINE and OFFLINE return immediately if the device is already in the state to be waited for.

SEE ALSO

BUGS

1.26 sana2.device/S2_ONLINE

NAME

Online -- Put a network interface back in service.

FUNCTION

This command places an offline network interface back into service.

IO REQUEST

ios2_Command - S2_ONLINE.

RESULTS

ios2_Error - Zero if successful; non-zero otherwise.
ios2_WireError - More specific error number.

NOTES

This command is responsible for putting the network interface hardware back into a known state (as close as possible to the state before S2_OFFLINE) and resets the unit global and special statistics.

SEE ALSO

S2_OFFLINE

BUGS

1.27 sana2.device/S2_READORPHAN

NAME

ReadOrphan -- Get a packet for which there is no reader.

FUNCTION

Get the next packet available that does not satisfy any then-pending CMD_READ requests. The data returned in the ios2_Data structure is normally the Data Link Layer packet type field and the packet data. If bit SANA2IOB_RAW is set in ios2_Flags, then the entire Data Link Layer packet, including both header and trailer information, will be returned.

IO REQUEST

ios2_Command - CMD_READORPHAN.
 ios2_Flags - Supported flags are:
 SANA2IOB_RAW
 SANA2IOB_QUICK
 ios2_DataLength - Length of the Data to be sent.
 ios2_Data - Abstract data structure which packet data is contained in.

RESULTS

ios2_Error - Zero if successful; non-zero otherwise.
 ios2_WireError - More specific error number.
 ios2_Flags - The following flags may be returned:
 SANA2IOB_RAW
 SANA2IOB_BCAST
 SANA2IOB_MCAST
 ios2_SrcAddr - Source interface address of packet.
 ios2_DstAddr - Destination interface address of packet.
 ios2_DataLength - Length of the Data to be sent.
 ios2_Data - Abstract data structure which packet data is contained in.

NOTES

This is intended for debugging and management tools. Protocols should not use this.

As with 802.3 packets on an ethernet, to determine which protocol family the returned packet belongs to you may have to specify SANA2IOB_RAW to get the entire data link layer wrapper (which is where the protocol type may be kept). Notice this necessarily means that this cannot be done in a network interface independent fashion. The driver will, however, fill in the PacketType field to the best of its ability.

SEE ALSO

CMD_READ, CMD_WRITE

BUGS

1.28 sana2.device/S2_TRACKTYPE

NAME

TrackType -- Accumulate statistics about a packet type.

FUNCTION

This command causes the device driver to accumulate statistics about a particular packet type. Packet type statistics, for the particular packet type, are zeroed by this command.

IO REQUEST

ios2_Command - S2_TRACKTYPE.
ios2_PacketType - Packet type of interest.

RESULTS

ios2_Error - Zero if successful; non-zero otherwise.
ios2_WireError - More specific error number.

NOTES

SEE ALSO

S2_UNTRACKTYPE, S2_GETTYPESTATS

BUGS

1.29 sana2.device/S2_UNTRACKTYPE

NAME

UntrackType -- End statistics about a packet type.

FUNCTION

This command causes the device driver to stop accumulating statistics about a particular packet type.

IO REQUEST

ios2_Command - S2_UNTRACKTYPE.
ios2_PacketType - Packet type of interest.

RESULTS

ios2_Error - Zero if successful; non-zero otherwise.
ios2_WireError - More specific error number.

NOTES

SEE ALSO

S2_TRACKTYPE, S2_GETTYPESTATS

BUGS
