$Id: README.hysdn,v 1.3.6.1 2001/02/10 14:41:19 kai Exp $
The hysdn driver has been written by
Werner Cornelius (werner@isdn4linux.de or werner@titro.de)
for Hypercope GmbH Aachen Germany. Hypercope agreed to publish this driver
under the GNU General Public License.

The CAPI 2.0-support was added by Ulrich Albrecht (ualbrecht@hypercope.de)
for Hypercope GmbH Aachen, Germany.

Table of contents
=================

1. About the driver

The drivers/isdn/hysdn subdir contains a driver for HYPERCOPEs active
PCI isdn cards Champ, Ergo and Metro. To enable support for this cards
enable ISDN support in the kernel config and support for HYSDN cards in
the active cards submenu. The driver may only be compiled and used if
support for loadable modules and the process filesystem have been enabled.

These cards provide two different interfaces to the kernel. Without the
optional CAPI 2.0 support, they register as ethernet card. IP-routing
to a ISDN-destination is performed on the card itself. All necessary
handlers for various protocols like ppp and others as well as config info
and firmware may be fetched from Hypercopes WWW-Site www.hypercope.de.

With CAPI 2.0 support enabled, the card can also be used as a CAPI 2.0
compliant devices with either CAPI 2.0 applications

(check isdn4k-utils) or -using the capidrv module- as a regular isdn4linux device. This is done via the same mechanism as with the active AVM cards and in fact uses the same module.

2. Loading/Unloading the driver

The module has no command line parameters and auto detects up to 10 cards in the id-range 0-9.
If a loaded driver shall be unloaded all open files in the /proc/net/hysdn subdir need to be closed and all ethernet interfaces allocated by this driver must be shut down. Otherwise the module counter will avoid a module unload.

If you are using the CAPI 2.0-interface, make sure to load/modprobe the kernelcapi-module first.

If you plan to use the capidrv-link to isdn4linux, make sure to load capidrv.o after all modules using this driver (i.e. after hysdn and any avm-specific modules).

3. Entries in the /proc filesystem

When the module has been loaded it adds the directory hysdn in the /proc/net tree. This directory contains exactly 2 file entries for each card. One is called cardconfX and the other cardlogX, where X is the card id number from 0 to 9.
The cards are numbered in the order found in the PCI config data.

4. The /proc/net/hysdn/cardconfX file

This file may be read to get by everyone to get info about the cards type, actual state, available features and used resources.
The first 3 entries (id, bus and slot) are PCI info fields, the following type field gives the information about the cards type:

4 -> Ergo card (server card with 2 b-chans)
5 -> Metro card (server card with 4 or 8 b-chans)
6 -> Champ card (client card with 2 b-chans)

The following 3 fields show the hardware assignments for irq, iobase and the dual ported memory (dp-mem).
The fields b-chans and fax-chans announce the available card resources of this types for the user.
The state variable indicates the actual drivers state for this card with the following assignments.

0 -> card has not been booted since driver load
1 -> card booting is actually in progess
2 -> card is in an error state due to a previous boot failure
3 -> card is booted and active

And the last field (device) shows the name of the ethernet device assigned to this card. Up to the first successful boot this field only shows a - to tell that no net device has been allocated up to now. Once a net device has been allocated it remains assigned to this card, even if a card is

rebooted and an boot error occurs.

Writing to the cardconfX file boots the card or transfers config lines to the cards firmware. The type of data is automatically detected when the first data is written. Only root has write access to this file.
The firmware boot files are normally called hyclient.pof for client cards and hyserver.pof for server cards.
After successfully writing the boot file, complete config files or single config lines may be copied to this file.
If an error occurs the return value given to the writing process has the following additional codes (decimal):

1000 Another process is currently bootng the card
1001 Invalid firmware header
1002 Boards dual-port RAM test failed
1003 Internal firmware handler error
1004 Boot image size invalid
1005 First boot stage (bootstrap loader) failed
1006 Second boot stage failure
1007 Timeout waiting for card ready during boot
1008 Operation only allowed in booted state
1009 Config line too long
1010 Invalid channel number
1011 Timeout sending config data

Additional info about error reasons may be fetched from the log output.

5. The /proc/net/hysdn/cardlogX file

The cardlogX file entry may be opened multiple for reading by everyone to get the cards and drivers log data. Card messages always start with the keyword LOG. All other lines are output from the driver.
The driver log data may be redirected to the syslog by selecting the appropriate bitmask. The cards log messages will always be send to this interface but never to the syslog.

A root user may write a decimal or hex (with 0x) value t this file to select desired output options. As mentioned above the cards log dat is always written to the cardlog file independent of the following options only used to check and debug the driver itself:

For example:
echo "0x34560078" > /proc/net/hysdn/cardlog0
to output the hex log mask 34560078 for card 0.

The written value is regarded as an unsigned 32-Bit value, bit ored for desired output. The following bits are already assigned:

0x80000000 All driver log data is alternatively via syslog
0x00000001 Log memory allocation errors
0x00000010 Firmware load start and close are logged
0x00000020 Log firmware record parser
0x00000040 Log every firmware write actions
0x00000080 Log all card related boot messages
0x00000100 Output all config data sent for debugging purposes
0x00000200 Only non comment config lines are shown wth channel

0x00000400	Additional conf log output
0x00001000	Log the asynchronous scheduler actions (config and log)
0x00100000	Log all open and close actions to /proc/net/hysdn/card files
0x00200000	Log all actions from /proc file entries
0x00010000	Log network interface init and deinit

6. Where to get additional info and help

If you have any problems concerning the driver or configuration contact the Hypercope support team (support@hypercope.de) and or the authors Werner Cornelius (werner@isdn4linux or cornelius@titro.de) or Ulrich Albrecht (ualbrecht@hypercope.de).