$Id: INTERFACE,v 1.15.8.2 2001/03/13 16:17:07 kai Exp $

Description of the Interface between Linklevel and Hardwarelevel
of isdn4linux:

The Communication between Linklevel (LL) and Hardwarelevel (HL)
is based on the struct isdn_if (defined in isdnif.h).

An HL-driver can register itself at LL by calling the function
register_isdn() with a pointer to that struct. Prior to that, it has
to preset some of the fields of isdn_if. The LL sets the rest of
the fields. All further communication is done via callbacks using
the function-pointers defined in isdn_if.

Changes/Version numbering:

During development of the ISDN subsystem, several changes have been
made to the interface. Before it went into kernel, the package
had a unique version number. The last version, distributed separately
was 0.7.4. When the subsystem went into kernel, every functional unit
got a separate version number. These numbers are shown at initialization,
separated by slashes:

c.c/t.t/n.n/p.p/a.a/v.v

where

c.c is the revision of the common code.
t.t is the revision of the tty related code.
n.n is the revision of the network related code.
p.p is the revision of the ppp related code.
a.a is the revision of the audio related code.
v.v is the revision of the V.110 related code.

Changes in this document are marked with '***CHANGEx' where x representing
the version number. If that number starts with 0, it refers to the old,
separately distributed package. If it starts with one of the letters
above, it refers to the revision of the corresponding module.
***CHANGEIx refers to the revision number of the isdnif.h

1. Description of the fields of isdn_if:

int channels;

This field has to be set by the HL-driver to the number of channels
supported prior to calling register_isdn(). Upon return of the call,
the LL puts an id there, which has to be used by the HL-driver when
invoking the other callbacks.

int maxbufsize;

***CHANGE0.6: New since this version.

Also to be preset by the HL-driver. With this value the HL-driver
tells the LL the maximum size of a data-packet it will accept.

unsigned long features;

To be preset by the HL-driver. Using this field, the HL-driver announces the features supported. At the moment this is limited to report the supported layer2 and layer3-protocols. For setting this field the constants ISDN_FEATURE..., declared in isdnif.h have to be used.

***CHANGE0.7.1: The line type (1TR6, EDSS1) has to be set.

unsigned short hl_hdrlen;

***CHANGE0.7.4: New field.

To be preset by the HL-driver, if it supports sk_buff's. The driver should put here the amount of additional space needed in sk_buff's for its internal purposes. Drivers not supporting sk_buff's should initialize this field to 0.

void (*rcvcallb_skb)(int, int, struct sk_buff *)

***CHANGE0.7.4: New field.

This field will be set by LL. The HL-driver delivers received data-packets by calling this function. Upon calling, the HL-driver must already have its private data pulled off the head of the sk_buff.

Parameter:

int	driver-Id
int	Channel-number locally to the driver. (starting with 0)
struct sk_buff *	Pointer to sk_buff, containing received data.

int (*statcallb)(isdn_ctrl*);

This field will be set by LL. This function has to be called by the HL-driver for signaling status-changes or other events to the LL.

Parameter:
isdn_ctrl*

The struct isdn_ctrl also defined in isdn_if. The exact meanings of its fields are described together with the descriptions of the possible events. Here is only a short description of the fields:

driver = driver Id.
command = event-type. (one of the constants ISDN_STAT_...)
arg = depends on event-type.
num = depends on event-type.

Returnvalue:
0 on success, else -1

int (*command)(isdn_ctrl*);

This field has to be preset by the HL-driver. It points to a function,

to be called by LL to perform functions like dialing, B-channel setup, etc. The exact meaning of the parameters is described with the descriptions of the possible commands.

Parameter:
  isdn_ctrl*
    driver = driver-Id
    command = command to perform. (one of the constants ISDN_CMD_...)
    arg   = depends on command.
    num   = depends on command.

Returnvalue:
  >=0 on success, else error-code (-ENODEV etc.)

```
int (*writebuf_skb)(int, int, int, struct sk_buff *)
```

***CHANGE0.7.4: New field.
***CHANGEI.1.21: New field.

This field has to be preset by the HL-driver. The given function will be called by the LL for delivering data to be send via B-Channel.

Parameter:

int	driver-Id ***CHANGE0.7.4: New parameter.
int	channel-number locally to the HL-driver. (starts with 0)
int	ack ***ChangeI1.21: New parameter If this is !0, the driver has to signal the delivery by sending an ISDN_STAT_BSENT. If this is 0, the driver MUST NOT send an ISDN_STAT_BSENT.
struct sk_buff *	Pointer to sk_buff containing data to be send via B-channel.

Returnvalue:
  Length of data accepted on success, else error-code (-EINVAL on oversized packets etc.)

```
int (*writecmd)(u_char*, int, int, int, int);
```

This field has to be preset by the HL-driver. The given function will be called to perform write-requests on /dev/isdnctrl (i.e. sending commands to the card) The data-format is hardware-specific. This function is intended for debugging only. It is not necessary for normal operation and never will be called by the tty-emulation- or network-code. If this function is not supported, the driver has to set NULL here.

Parameter:

u_char*	pointer to data.
int	length of data.
int	flag: 0 = call from within kernel-space. (HL-driver must use memcpy, may NOT use schedule()) 1 = call from user-space. (HL-driver must use memcpy_fromfs, use of schedule() allowed)
int	driver-Id.
int	channel-number locally to the HL-driver. (starts with 0)

***CHANGEI1.14: The driver-Id and channel-number are new since this revision.

Returnvalue:
Length of data accepted on success, else error-code (-EINVAL etc.)

int (*readstat)(u_char*, int, int, int, int);

This field has to be preset by the HL-driver. The given function will be called to perform read-requests on /dev/isdnctrl (i.e. reading replies from the card) The data-format is hardware-specific. This function is intended for debugging only. It is not necessary for normal operation and never will be called by the tty-emulation- or network-code. If this function is not supported, the driver has to set NULL here.

Parameter:
u_char* pointer to data.
int length of data.
int flag: 0 = call from within kernel-space. (HL-driver must use memcpy, may NOT use schedule())
1 = call from user-space. (HL-driver must use memcpy_fromfs, use of schedule() allowed)
int driver-Id.
int channel-number locally to the HL-driver. (starts with 0)

***CHANGEI1.14: The driver-Id and channel-number are new since this revision.

Returnvalue:
Length of data on success, else error-code (-EINVAL etc.)

char id[20];
***CHANGE0.7: New since this version.

This string has to be preset by the HL-driver. Its purpose is for identification of the driver by the user. Eg.: it is shown in the status-info of /dev/isdninfo. Furthermore it is used as Id for binding net-interfaces to a specific channel. If a string of length zero is given, upon return, isdn4linux will replace it by a generic name. (line0, line1 etc.) It is recommended to make this string configurable during module-load-time. (copy a global variable to this string.) For doing that, modules 1.2.8 or newer are necessary.

2. Description of the commands, a HL-driver has to support:

All commands will be performed by calling the function command() described above from within the LL. The field command of the struct-parameter will contain the desired command, the field driver is always set to the appropriate driver-Id.

Until now, the following commands are defined:

***CHANGEI1.34: The parameter "num" has been replaced by a union "parm" containing
the old "num" and a new setup_type struct used for ISDN_CMD_DIAL and ISDN_STAT_ICALL callback.

ISDN_CMD_IOCTL:

This command is intended for performing ioctl-calls for configuring hardware or similar purposes (setting port-addresses, loading firmware etc.) For this purpose, in the LL all ioctl-calls with an argument >= IIOCDRVCTL (0x100) will be handed transparently to this function after subtracting 0x100 and placing the result in arg.
Example:
If a userlevel-program calls ioctl(0x101,...) the function gets called with the field command set to 1.

Parameter:
driver = driver-Id.
command = ISDN_CMD_IOCTL
arg = Original ioctl-cmd - IIOCDRVCTL
parm.num = first bytes filled with (unsigned long)arg

Returnvalue:
Depending on driver.

ISDN_CMD_DIAL:

This command is used to tell the HL-driver it should dial a given number.

Parameter:
driver = driver-Id.
command = ISDN_CMD_DIAL
arg = channel-number locally to the driver. (starting with 0)

parm.setup.phone = An ASCII-String containing the number to dial.
parm.setup.eazmsn = An ASCII-Sting containing the own EAZ or MSN.
parm.setup.si1 = The Service-Indicator.
parm.setup.si2 = Additional Service-Indicator.

If the Line has been designed as SPV (a special german feature, meaning semi-leased-line) the phone has to start with an "S".
***CHANGE0.6: In previous versions the EAZ has been given in the highbyte of arg.
***CHANGE0.7.1: New since this version: ServiceIndicator and AddInfo.

ISDN_CMD_ACCEPTD:

With this command, the HL-driver is told to accept a D-Channel-setup. (Response to an incoming call)

Parameter:
driver = driver-Id.
command = ISDN_CMD_ACCEPTD
arg = channel-number locally to the driver. (starting with 0)
parm = unused.

ISDN_CMD_ACCEPTB:

With this command, the HL-driver is told to perform a B-Channel-setup.

(after establishing D-Channel-Connection)

Parameter:
driver = driver-Id.
command = ISDN_CMD_ACCEPTB
arg = channel-number locally to the driver. (starting with 0)
parm = unused.

ISDN_CMD_HANGUP:

With this command, the HL-driver is told to hangup (B-Channel if established first, then D-Channel). This command is also used for actively rejecting an incoming call.

Parameter:
driver = driver-Id.
command = ISDN_CMD_HANGUP
arg = channel-number locally to the driver. (starting with 0)
parm = unused.

ISDN_CMD_CLREAZ:

With this command, the HL-driver is told not to signal incoming calls to the LL.

Parameter:
driver = driver-Id.
command = ISDN_CMD_CLREAZ
arg = channel-number locally to the driver. (starting with 0)
parm = unused.

ISDN_CMD_SETEAZ:

With this command, the HL-driver is told to signal incoming calls for the given EAZs/MSNs to the LL.

Parameter:
driver = driver-Id.
command = ISDN_CMD_SETEAZ
arg = channel-number locally to the driver. (starting with 0)
parm.num = ASCII-String, containing the desired EAZ's/MSN's (comma-separated). If an empty String is given, the HL-driver should respond to ALL incoming calls, regardless of the destination-address.
***CHANGE0.6: New since this version the "empty-string"-feature.

ISDN_CMD_GETEAZ: (currently unused)

With this command, the HL-driver is told to report the current setting given with ISDN_CMD_SETEAZ.

Parameter:
driver = driver-Id.
command = ISDN_CMD_GETEAZ
arg = channel-number locally to the driver. (starting with 0)
parm.num = ASCII-String, containing the current EAZ's/MSN's

ISDN_CMD_SETSIL: (currently unused)

With this command, the HL-driver is told to signal only incoming calls with the given Service-Indicators.

Parameter:
driver = driver-Id.
command = ISDN_CMD_SETSIL
arg = channel-number locally to the driver. (starting with 0)
parm.num = ASCII-String, containing the desired Service-Indicators.

ISDN_CMD_GETSIL: (currently unused)

With this command, the HL-driver is told to return the current Service-Indicators it will respond to.

Parameter:
driver = driver-Id.
command = ISDN_CMD_SETSIL
arg = channel-number locally to the driver. (starting with 0)
parm.num = ASCII-String, containing the current Service-Indicators.

ISDN_CMD_SETL2:

With this command, the HL-driver is told to select the given Layer-2-protocol. This command is issued by the LL prior to ISDN_CMD_DIAL or ISDN_CMD_ACCEPTD.

Parameter:
driver = driver-Id.
command = ISDN_CMD_SETL2
arg = channel-number locally to the driver. (starting with 0)
logical or'ed with (protocol-Id << 8)
protocol-Id is one of the constants ISDN_PROTO_L2...
parm = unused.

ISDN_CMD_GETL2: (currently unused)

With this command, the HL-driver is told to return the current setting of the Layer-2-protocol.

Parameter:
driver = driver-Id.
command = ISDN_CMD_GETL2
arg = channel-number locally to the driver. (starting with 0)
parm = unused.
Returnvalue:
current protocol-Id (one of the constants ISDN_L2_PROTO)

ISDN_CMD_SETL3:

With this command, the HL-driver is told to select the given Layer-3-protocol. This command is issued by the LL prior to ISDN_CMD_DIAL or ISDN_CMD_ACCEPTD.

```
  Parameter:
    driver      = driver-Id.
    command     = ISDN_CMD_SETL3
    arg         = channel-number locally to the driver. (starting with 0)
                  logical or'ed with (protocol-Id << 8)
                  protocol-Id is one of the constants ISDN_PROTO_L3...
    parm.fax    = Pointer to T30_s fax struct. (fax usage only)

ISDN_CMD_GETL2: (currently unused)

  With this command, the HL-driver is told to return the current
  setting of the Layer-3-protocol.

  Parameter:
    driver      = driver-Id.
    command     = ISDN_CMD_GETL3
    arg         = channel-number locally to the driver. (starting with 0)
    parm        = unused.
  Returnvalue:
    current protocol-Id (one of the constants ISDN_L3_PROTO)

ISDN_CMD_PROCEED:

  With this command, the HL-driver is told to proceed with a incoming call.

  Parameter:
    driver      = driver-Id.
    command     = ISDN_CMD_PROCEED
    arg         = channel-number locally to the driver. (starting with 0)
    setup.eazmsn= empty string or string send as uus1 in DSS1 with
                  PROCEED message

ISDN_CMD_ALERT:

  With this command, the HL-driver is told to alert a proceeding call.

  Parameter:
    driver      = driver-Id.
    command     = ISDN_CMD_ALERT
    arg         = channel-number locally to the driver. (starting with 0)
    setup.eazmsn= empty string or string send as uus1 in DSS1 with
                  ALERT message

ISDN_CMD_REDIR:

  With this command, the HL-driver is told to redirect a call in proceeding
  or alerting state.

  Parameter:
    driver      = driver-Id.
    command     = ISDN_CMD_REDIR
    arg         = channel-number locally to the driver. (starting with 0)
    setup.eazmsn= empty string or string send as uus1 in DSS1 protocol
    setup.screen= screening indicator
```

    setup.phone = redirected to party number

  ISDN_CMD_PROT_IO:

    With this call, the LL-driver invokes protocol specific features through
    the LL.
    The call is not implicitely bound to a connection.

    Parameter:
      driver   = driver-Id
      command   = ISDN_CMD_PROT_IO
      arg   = The lower 8 Bits define the addressed protocol as defined
                    in ISDN_PTYPE..., the upper bits are used to differentiate
                    the protocol specific CMD.

      para   = protocol and function specific. See isdnif.h for detail.

  ISDN_CMD_FAXCMD:

    With this command the HL-driver receives a fax sub-command.
    For details refer to INTERFACE.fax

    Parameter:
      driver   = driver-Id.
      command   = ISDN_CMD_FAXCMD
      arg   = channel-number locally to the driver. (starting with 0)
      parm   = unused.

3. Description of the events to be signaled by the HL-driver to the LL.

  All status-changes are signaled via calling the previously described
  function statcallb(). The field command of the struct isdn_cmd has
  to be set by the HL-driver with the appropriate Status-Id (event-number).
  The field arg has to be set to the channel-number (locally to the driver,
  starting with 0) to which this event applies. (Exception: STAVAIL-event)

  Until now, the following Status-Ids are defined:

  ISDN_STAT_AVAIL:

    With this call, the HL-driver signals the availability of new data
    for readstat(). Used only for debugging-purposes, see description
    of readstat().

    Parameter:
      driver   = driver-Id
      command   = ISDN_STAT_STAVAIL
      arg   = length of available data.
      parm   = unused.

  ISDN_STAT_ICALL:
  ISDN_STAT_ICALLW:

    With this call, the HL-driver signals an incoming call to the LL.

If ICALLW is signalled the incoming call is a waiting call without
a available B-chan.

Parameter:

driver	= driver-Id
command	= ISDN_STAT_ICALL
arg	= channel-number, locally to the driver. (starting with 0)
para.setup.phone	= Callernumber.
para.setup.eazmsn	= CalledNumber.
para.setup.si1	= Service Indicator.
para.setup.si2	= Additional Service Indicator.
para.setup.plan	= octet 3 from Calling party number Information Element.
para.setup.screen	= octet 3a from Calling party number Information Element.

Return:

0	= No device matching this call.
1	= At least one device matching this call (RING on ttyI). HL-driver may send ALERTING on the D-channel in this case.
2	= Call will be rejected.
3	= Incoming called party number is currently incomplete. Additional digits are required. Used for signalling with PtP connections.
4	= Call will be held in a proceeding state (HL driver sends PROCEEDING) Used when a user space prog needs time to interpret a call para.setup.eazmsn may be filled with an uus1 message of 30 octets maximum. Empty string if no uus.
5	= Call will be actively deflected to another party Only available in DSS1/EURO protocol para.setup.phone must be set to destination party number para.setup.eazmsn may be filled with an uus1 message of 30 octets maximum. Empty string if no uus.
-1	= An error happened. (Invalid parameters for example.)

The keypad support now is included in the dial command.

ISDN_STAT_RUN:

With this call, the HL-driver signals availability of the ISDN-card.
(after initializing, loading firmware)

Parameter:

driver	= driver-Id
command	= ISDN_STAT_RUN
arg	= unused.
parm	= unused.

ISDN_STAT_STOP:

With this call, the HL-driver signals unavailability of the ISDN-card.
(before unloading, while resetting/reconfiguring the card)

Parameter:

driver	= driver-Id

```
    command     = ISDN_STAT_STOP
    arg         = unused.
    parm        = unused.
```

ISDN_STAT_DCONN:

With this call, the HL-driver signals the successful establishment of a D-Channel-connection. (Response to ISDN_CMD_ACCEPTD or ISDN_CMD_DIAL)

```
  Parameter:
    driver      = driver-Id
    command     = ISDN_STAT_DCONN
    arg         = channel-number, locally to the driver. (starting with 0)
    parm        = unused.
```

ISDN_STAT_BCONN:

With this call, the HL-driver signals the successful establishment of a B-Channel-connection. (Response to ISDN_CMD_ACCEPTB or because the remote-station has initiated establishment)

The HL driver should call this when the logical l2/l3 protocol connection on top of the physical B-channel is established.

```
  Parameter:
    driver      = driver-Id
    command     = ISDN_STAT_BCONN
    arg         = channel-number, locally to the driver. (starting with 0)
    parm.num    = ASCII-String, containing type of connection (for analog
                  modem only). This will be appended to the CONNECT message
                  e.g. 14400/V.32bis
```

ISDN_STAT_DHUP:

With this call, the HL-driver signals the shutdown of a D-Channel-connection. This could be a response to a prior ISDN_CMD_HANGUP, or caused by a remote-hangup or if the remote-station has actively rejected a call.

```
  Parameter:
    driver      = driver-Id
    command     = ISDN_STAT_DHUP
    arg         = channel-number, locally to the driver. (starting with 0)
    parm        = unused.
```

ISDN_STAT_BHUP:

With this call, the HL-driver signals the shutdown of a B-Channel-connection. This could be a response to a prior ISDN_CMD_HANGUP, or caused by a remote-hangup.

The HL driver should call this as soon as the logical l2/l3 protocol connection on top of the physical B-channel is released.

```
  Parameter:
    driver      = driver-Id
```

```
    command     = ISDN_STAT_BHUP
    arg         = channel-number, locally to the driver. (starting with 0)
    parm        = unused.

ISDN_STAT_CINF:

 With this call, the HL-driver delivers charge-unit information to the
 LL.

  Parameter:
    driver      = driver-Id
    command     = ISDN_STAT_CINF
    arg         = channel-number, locally to the driver. (starting with 0)
    parm.num    = ASCII string containing charge-units (digits only).

ISDN_STAT_LOAD: (currently unused)

ISDN_STAT_UNLOAD:

 With this call, the HL-driver signals that it will be unloaded now. This
 tells the LL to release all corresponding data-structures.

  Parameter:
    driver      = driver-Id
    command     = ISDN_STAT_UNLOAD
    arg         = unused.
    parm        = unused.

ISDN_STAT_BSENT:

  With this call the HL-driver signals the delivery of a data-packet.
  This callback is used by the network-interfaces only, tty-Emulation
  does not need this call.

  Parameter:
    driver      = driver-Id
    command     = ISDN_STAT_BSENT
    arg         = channel-number, locally to the driver. (starting with 0)
    parm.length = ***CHANGEI.1.21: New field.
                  the driver has to set this to the original length
                  of the skb at the time of receiving it from the linklevel.

ISDN_STAT_NODCH:

  With this call, the driver has to respond to a prior ISDN_CMD_DIAL, if
  no D-Channel is available.

  Parameter:
    driver      = driver-Id
    command     = ISDN_STAT_NODCH
    arg         = channel-number, locally to the driver. (starting with 0)
    parm        = unused.

ISDN_STAT_ADDCH:

  This call is for HL-drivers, which are unable to check card-type
```

or numbers of supported channels before they have loaded any firmware
using ioctl. Those HL-driver simply set the channel-parameter to a
minimum channel-number when registering, and later if they know
the real amount, perform this call, allocating additional channels.

Parameter:
  driver   = driver-Id
  command   = ISDN_STAT_ADDCH
  arg   = number of channels to be added.
  parm   = unused.

ISDN_STAT_CAUSE:

With this call, the HL-driver delivers CAUSE-messages to the LL.
Currently the LL does not use this messages. Their contents is simply
logged via kernel-messages. Therefore, currently the format of the
messages is completely free. However they should be printable.

Parameter:
  driver   = driver-Id
  command   = ISDN_STAT_NODCH
  arg   = channel-number, locally to the driver. (starting with 0)
  parm.num   = ASCII string containing CAUSE-message.

ISDN_STAT_DISPLAY:

With this call, the HL-driver delivers DISPLAY-messages to the LL.
Currently the LL does not use this messages.

Parameter:
  driver   = driver-Id
  command   = ISDN_STAT_DISPLAY
  arg   = channel-number, locally to the driver. (starting with 0)
  para.display= string containing DISPLAY-message.

ISDN_STAT_PROT:

With this call, the HL-driver delivers protocol specific infos to the LL.
The call is not implicitely bound to a connection.

Parameter:
  driver   = driver-Id
  command   = ISDN_STAT_PROT
  arg   = The lower 8 Bits define the addressed protocol as defined
                in ISDN_PTYPE..., the upper bits are used to differentiate
                the protocol specific STAT.

  para   = protocol and function specific. See isdnif.h for detail.

ISDN_STAT_DISCH:

With this call, the HL-driver signals the LL to disable or enable the
use of supplied channel and driver.
The call may be used to reduce the available number of B-channels after
loading the driver. The LL has to ignore a disabled channel when searching
for free channels. The HL driver itself never delivers STAT callbacks for

disabled channels.
The LL returns a nonzero code if the operation was not successful or the selected channel is actually regarded as busy.

Parameter:
driver = driver-Id
command = ISDN_STAT_DISCH
arg = channel-number, locally to the driver. (starting with 0)
parm.num[0] = 0 if channel shall be disabled, else enabled.

ISDN_STAT_L1ERR:

***CHANGEI1.21 new status message.
A signal can be sent to the linklevel if an Layer1-error results in packet-loss on receive or send. The field errcode of the cmd.parm union describes the error more precisely.

Parameter:
driver = driver-Id
command = ISDN_STAT_L1ERR
arg = channel-number, locally to the driver. (starting with 0)
parm.errcode= ISDN_STAT_L1ERR_SEND: Packet lost while sending.
ISDN_STAT_L1ERR_RECV: Packet lost while receiving.

ISDN_STAT_FAXIND:

With this call the HL-driver signals a fax sub-command to the LL.
For details refer to INTERFACE.fax

Parameter:
driver = driver-Id.
command = ISDN_STAT_FAXIND
arg = channel-number, locally to the driver. (starting with 0)
parm = unused.