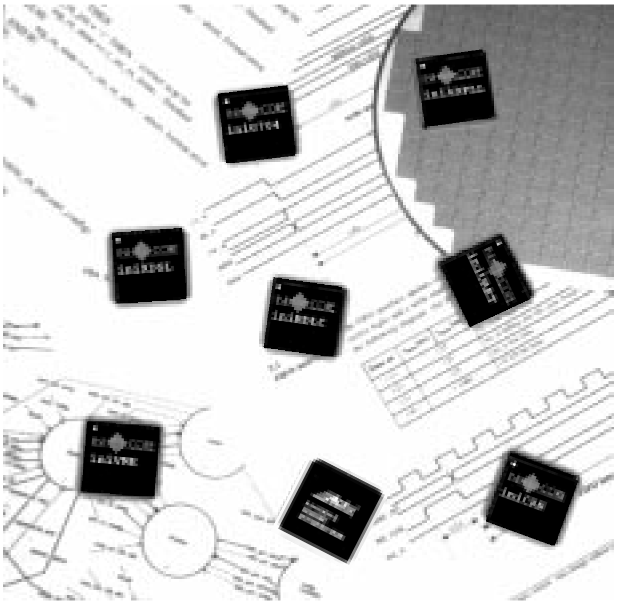
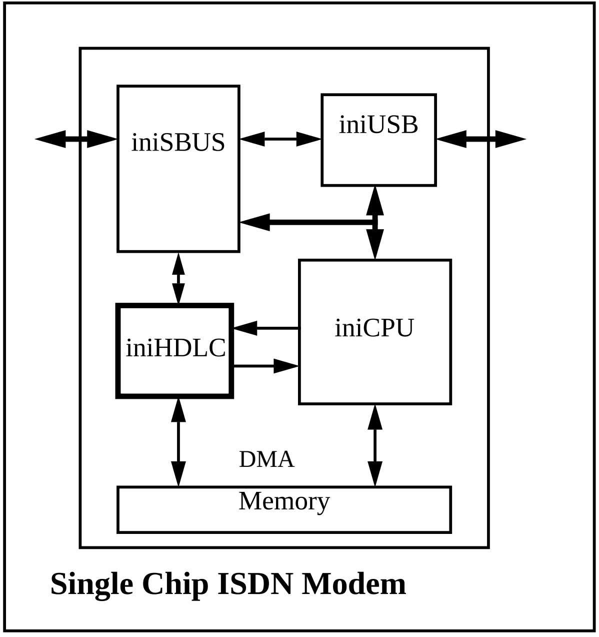
iniHDLC Controller data sheet



Features:

- HDLC Family Transmitter Receiver
- Single Port Interface
- Transparent Mode
- Start- and Stopflag Generation/Detection
- Frame Check Sequence Generation/ Verification (CRC-16)
- Built-in Bit Stuffing/Destuffing
- Frame Status Signalling
- Clk to Baud Ratio up to 1:1
- Supporting Protocols: Q.921(LAPD) ISDN, Q.922 Frame Relay, X.25 (LAPB) PBX , WAN...and more
- Flexible I/O Interface towards FIFO, uP, RAM and DMA

Example usage of iniHDLC in a System:



INICORE - the reliable Core and System Provider. We provide high quality IP, design expertise and leading edge silicon to the industry.

Inicore’s **iniHDLC** family of High-Level Data Link Controller (HDLC) cores consist a Receiver (FPR: from primary rate) and a Transmitter (TPR: to primary rate) unit. These single channel HDLC controllers handle all interframe flags, delimiting flags and Frame Check Sequence (FCS) pattern. The FCS is calculated using a CRC-16 polynomial.

The HDLC protocol has several subsets (e.g., Q.921, Q922, X.25 etc.). The structure of a frame (header, trailer, FCS, stuffing/destuffing) is the same but the procedures for Commands, Responses and Abort Conditions are different and are handled by software. iniHDLC cores can be used within the whole HDLC subset.

The iniHDLC cores are a synthesiszable, flexible, and structured VHDL implementation of the HDLC protocol. The cores are designed for easy interfacing to custom buffers (e.g., FIFO, DMA-Interface).
INICORE’s strategy is not to compete with the standard chip manufacturers, but to use the ASIC technologies for `system on chip’ design, which demands standards like HDLC with highly specific data processing capabilities.

US Sales Office:
INICORE INC.
5600 Mowry School Road, Suite 180,
Newark, CA 94560
Tel: 510 445 1529 Fax: 510 656 0995
E-mail: ask_us@inicore.com
Web: www.inicore.com

INICORE AG
Mattenstrasse 6a,CH-2555 Brügg, Switzerland
Tel: ++41 32 374 32 00, Fax: ++41 32 374 32 01
E-mail: ask_us@inicore.ch
Web: www.inicore.ch

1 Overview

When building a system with a data link connection there are basically two possibilities to design the interface to the serial data stream:

- Software based data processing
- Hardware supported solution

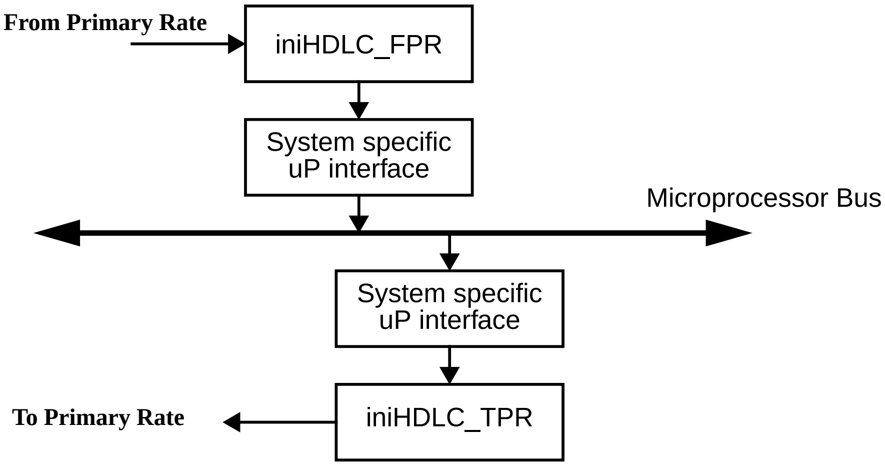
The software based implementation is very flexible but the whole data processing (framing, bit stuffing/destuffing, FCS calculation) may take a big part of the available processor performance. It may be necessary to use an other uP which has more performance.

An advanced solution is to implement the basic data processing in hardware.
iniHDLC_FPR/-TPR can be used to handle framing, bit stuffing/destuffing and FCS calculation/verification.

iniHDLC_FPR:
(From Primary Rate -> “Receiver”) Extracts a HDLC frame from the incoming data stream, performs frame alignment, bit destuffing and FCS verification.

iniHDLC_TPR:
(To Primary Rate -> “Transmitter”) This core generates a HDLC frame with header and trailer informations and send it to the primary rate interface.

The following block diagram shows how the two cores can be embedded into a system:



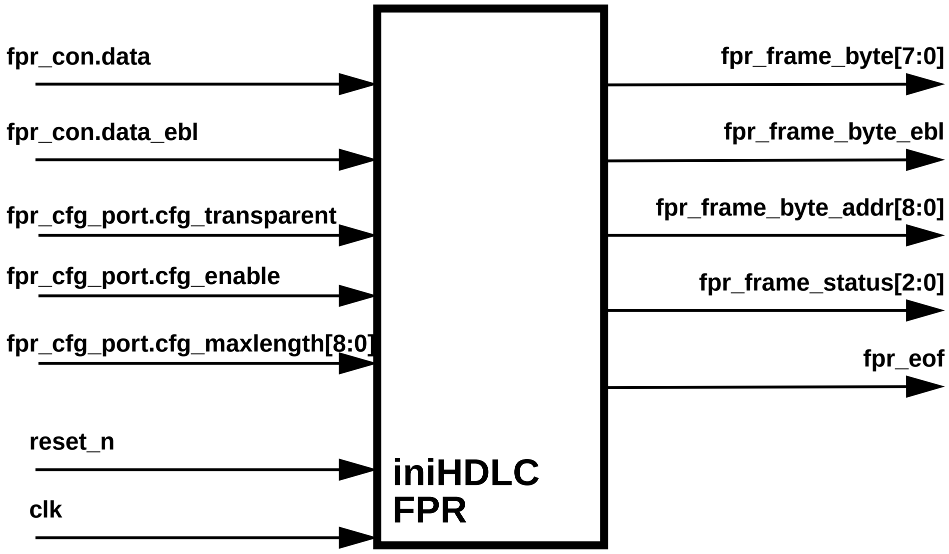
Depending on system requirements like data throughput and message response time the interface to the mircroprocessor can be designed to support features such as:

- Interrupt controlled data transfer
- Direct memory access using a built-in DMA controller
- On-chip dual-port memory for temporary buffering of a frame

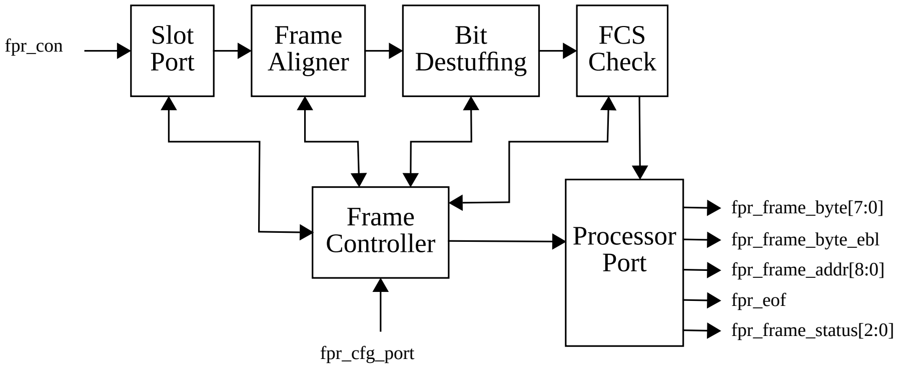
It is up to the system designer to choose the appropriate interface circuit. Both iniHDLC cores are designed to simplify the integration into a system.

1.1 Structure of iniHDLC_FPR

Pinout of iniHDLC_FPR:



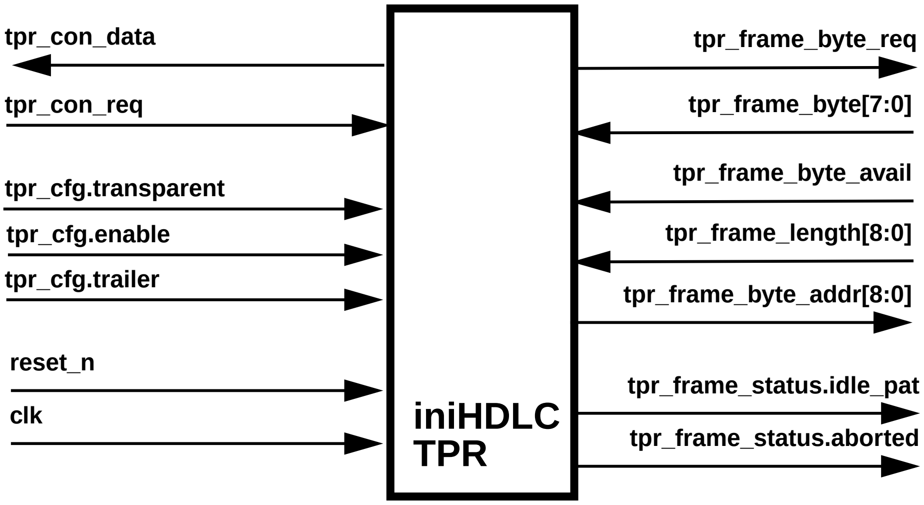
Structure of iniHLDC_FPR:



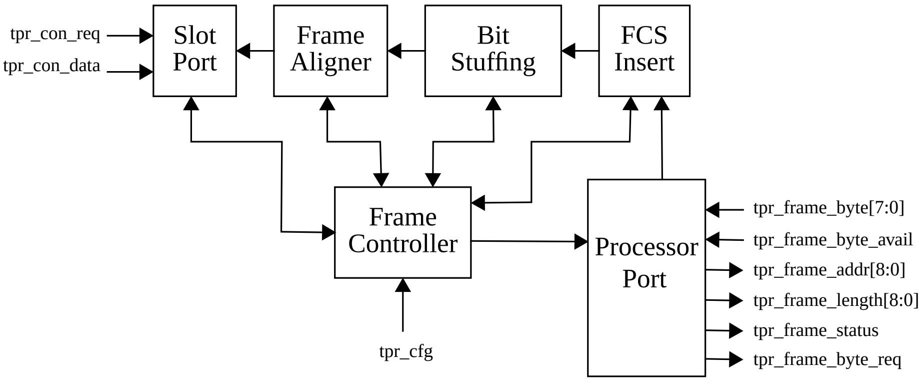
The processor port is the synchronous interface to an application dependent interface module, as e.g. a microprocessor interface module.

1.2 Structure of iniHDLC_TPR

Pinout of iniHDLC_TPR:



Structure of iniHLDC_TPR:



The processor port is the synchronous interface to an application dependent interface module, as e.g. a microprocessor interface module.

2 IO description

The following part lists the input and output ports of INICORE’s iniHDLC cores and explains their functionality.

2.1 General inputs

These pins are use to clock and initialize the whole iniHDLC core . There are no other clocks in these cores.

Pin Name	Type	Description
clk	in	System clock
reset_n	in	Asynchronous system reset, active low

2.2 HDLC_FPR IO's

2.2.1 Configuration FPR

The configuration pins are used to select the operating mode. They are static.

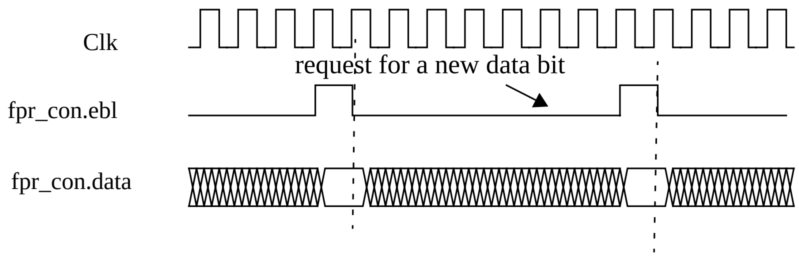
Pin Name	Type	Description
fpr_cfg_port. cfg_transparent	in	'1': The connection is transparent. The incoming data are unmodified transferred to the processor port. No address space overrun detection. '0': Normal function. Data is destuffed, unflagged (flag filtering), aligned and CRC-checked. (see cfg_maxlength for address overrun detection)
fpr_cfg_port. cfg_enable	in	'1': The connection is enabled. '0': HDLC_FPR is disabled. This signal may be used as a synchronous reset of the core.
fpr_cfg_port. cfg_maxlength [8:0]	in	Configuration of the maximum HDLC frame length including the 2 FCS bytes (no header- / trailerflag) Definition : value of cfg_max_length = #databytes + 2 bytes FCS) = n When HDLC_FPR receives longer frame than configured then status changes to "maximum length exceeded" When cfg_transparent = '1' then cfg_max_length has no function.

2.2.2 FPR Slot Port

The FPR slot port is connected to the synchronous input data stream.

Pin Name	Type	Description
fpr_con. data	in	Data bit (receiving HDLC serial bit stream)
fpr_con. data_ebl	in	Active ONE-pulse indicates valid data bit. The ratio clk to fpr_con.ebl can be from 1/n up to 1/1. The fpr_con.ebl can be set to '1' when baudrate = clk.

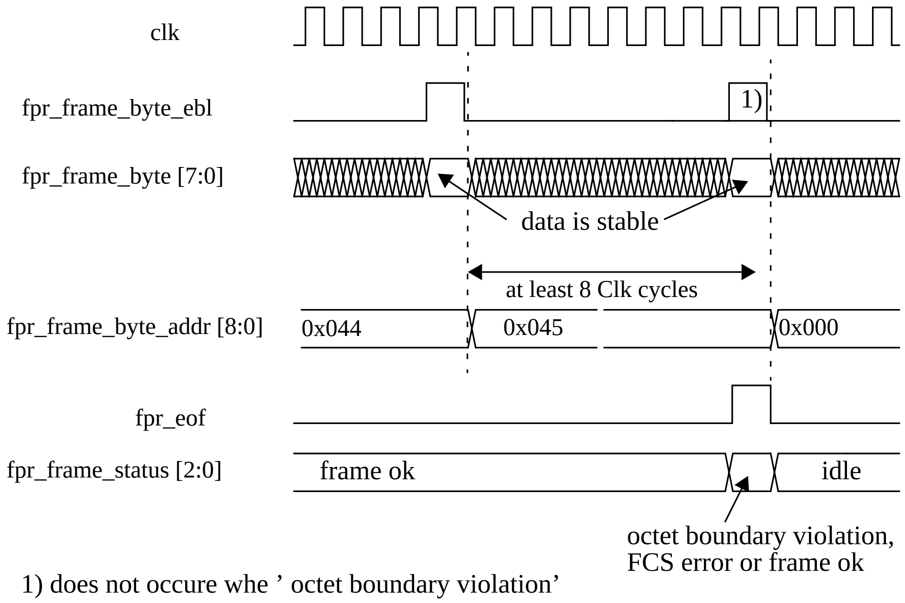
The functional behaviour is shown in the timing diagram below.



2.2.3 FPR Processor Port

The output port of HDLC_FPR gives access to the processed HDLC frame. This port can be connected to a DMA controller, a local uP interface or to custom on-chip logic. Core status information like FCS error, idle (flagpattern receiving) or frame aborted are supported.

Pin Name	Type	Description
fpr_frame_byte [7:0]	out	Received data byte. The last two received bytes are the CRC bytes.
fpr_frame_byte_ebl	out	Active ONE-pulse indicates valid fpr_frame_byte. Will NOT be active when FPR_Frame_Status = ‘octet boundary violation’ .
fpr_frame_byte_addr[8:0]	out	Can be used as write-address for storing fpr_frame_byte. The first word is written into address 0x0000. When FPR_EOF=’1’ byte address represents (frame length - 1). If cfg_transparent = ‘1’ then fpr_frame_byte_addr = not valid.
fpr_eof	out	Active ONE-pulse indicates end-of-frame.
fpr_frame_status[2:0]	out	The status is listed by priority (highest priority first) “110”: transparent transfer (config dependent) “101”: frame aborted (received more than 7 ones) “011”: idle (received “01111110” pattern) “100”: maximum length exceeded (occurs when Byte_Address = CfgMaxLenght) “010”: octet boundary violation (when 2nd flag received and not a complete octed ready) “001”: FCS error “000”: frame ok “111”: n.a.



1) does not occure whe ’ octet boundary violation’

fpr_frame_byte[7:0] is always valid when fpr_frame_byte_ebl is active. With this synchronous interface a lot of different applications are supported. fpr_frame_byte_ebl, fpr_frame_byte, fpr_frame_byte_addr and fpr_eof are registered outputs.

2.3 HDLC_TPR IO's

2.3.1 Configuration TPR

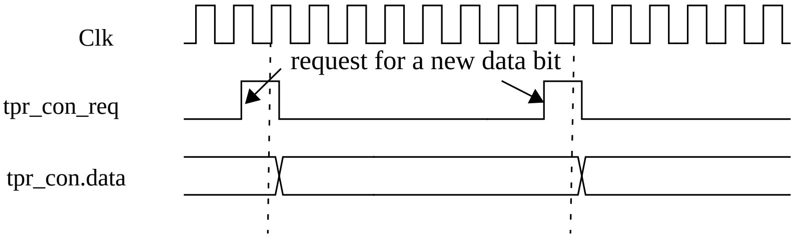
The configuration pins are used to select the operating mode. They are static.

Pin Name	Type	Description
tpr_cfg. cfg_transparent	in	'1': The connection is transparent. Frame data from the processor port are unmodified transferred to the slot port. '0': Normal mode.
tpr_cfg. cfg_enable	in	'1': The connection is enabled. '0': HDPC_TPR is reset. This signal may be used as a synchronous reset of the core.
tpr_cfg. cfg_trailer	in	'0': No trailer flag will be added to the frame '1': A trailer flag will be added to the frame

2.3.2 TPR slot port

Pin Name	Type	Description
tpr_con_data	out	Serial data bit
tpr_con_req	in	Active ONE-pulse indicates a request for a new data bit. The ratio clk to tpr_con_req can be from 1/n up to 1/1. The tpr_con_req can be set to '1' when baudrate = clk.

The functional behaviour is shown in the timing diagram below.

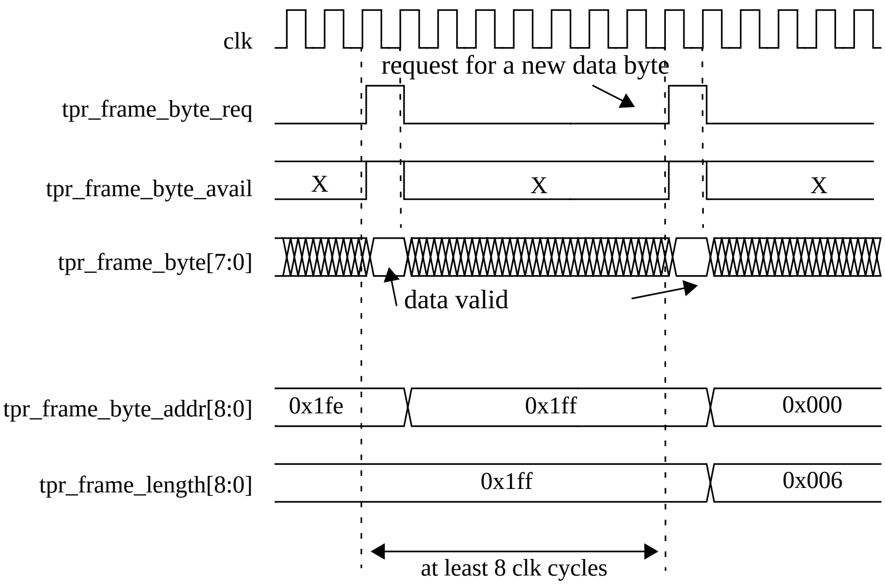


2.3.3 TPR processor port

The TPR processor port is the system interface to a DMA controller, a local uP interface or to custom on-chip logic. Core status information like frame aborted or idle pattern are available too.

Pin Name	Type	Description
tpr_frame_byte [7:0]	in	Data byte to be transmitted.
tpr_frame_byte_ avail	in	If ‘1’ indicates valid tpr_frame_byte.
tpr_frame_byte_ request	out	Active ONE-pulse signalling that the data byte will be fetched at next rising clk edge. If data is not valid (tpr_frame_data_avail = ‘0’) then a idle pattern will be inserted. But while the FPR is sending a frame (inside frame state) and no valid data byte is available when requested then a abort byte is transmitted. The definition of the abortbyte (constant name = c_abort) is in the FPR typedefintion file. The default value of c_abort is “01111111”.
tpr_frame_length [8:0]	in	Number of data bytes to be packet into the current frame. Signal must be stable during frame transmission. tpr_frame_length may not be ‘0’!
tpr_frame_byte_ addr [8:0]	out	Can be used as read-address for reading tpr_frame_byte. The first word is read from address 0x0000. End-of-frame is indicated when tpr_frame_byte_addr = tpr_frame_length; tpr_frame_byte_addr will restart at 0x0000. tpr_frame_byte_addr is not valid when TPR is in transparent mode.
tpr_frame_status. idle_pat	out	Active ONE-pulse indicates data idle pattern has been sent
tpr_frame_status. frame_aborted	out	Active ONE-pulse indicates that the current frame has been aborted.

Functional timing:



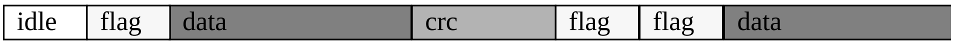
If the input port doesn’t get a valid framebyte after the request the core will insert a idle pattern. If this happens during a frame transfer tpr_frame_status will indicate ‘frame aborted’.

The behaviour of the tpr_frame_length and tpr_frame_byte_addr signals are illustrated above using examples numbers.

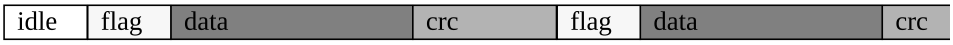
2.3.4 General Frame Cases

Following some framing cases are illustrated.

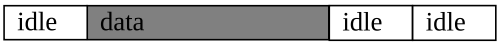
Frame with trailer flag:



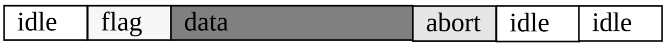
Framing without trailer flag:



Framingmode transparent:



Data not available during a frame transfer:



Default values of flag-, idle- and abortbyte:

idle = “01111110” flag = “01111110” abort = “01111111”

The default values can be easily modified by changing constant values in the typedefinition file.