CoreSDLC

DirectCore

Product Summary

Intended Use

- ISDN D-Channel
- X.25 Networks
- Frame Relay Networks
- Custom Serial Interfaces

Key Features

- Based on Intel's 80C152 Global Serial Channel Working in SDLC Mode
- Single and Double-Byte Address Recognition
- Address Filtering Enables Multicast and Broadcast Addresses
- 16-bit (CRC-16) and 32-bit (CRC-32) Frame Check Sequence
- NRZ and NRZI Data Encoding
- Automatic Bit Stuffing/Stripping
- 3-Byte Deep Internal Receive and Transmit FIFOs
- Full or Half-Duplex Operation
- Variable Baud Rate
- External or Internal Transmit and Receive Clocks
- Optional Preamble Generation
- Programmable Interframe Space
- Raw Transmit and Receive Testing Modes
- All Major Actel Device Families Supported

Supported Families

- Fusion
- ProASIC3/E
- ProASIC$^{\underline{PLUS}}$
- Axcelerator
- SX-A
- RTSX-S

Core Deliverables

- Evaluation Version
  - Compiled RTL Simulation Model Fully Supported in Actel Libero® Integrated Design Environment (IDE)
- Netlist Version
  - Structural Verilog and VHDL Netlists (with and without I/O pads) Compatible with Actel's Designer Software Place-and-Route Tool
  - Compiled RTL Simulation Model Fully Supported in Actel Libero IDE
- RTL Version
  - Verilog and VHDL Core Source Code
  - Core Synthesis Scripts
- Testbench (Verilog and VHDL)

Synthesis and Simulation Support

- Synthesis: Synplicity®, Synopsys® (Design Compiler® / FPGA Compiler™ / FPGA Express™), Exemplar™
- Simulation: OVI-Compliant Verilog Simulators and Vital-Compliant VHDL Simulators

Core Verification

- Comprehensive VHDL and Verilog Testbenches
- User can Modify Testbench Using Existing Format to Add Custom Tests

Contents

General Description 1
CoreSDLC Device Requirements 3
CoreSDLC Verification 3
I/O Signal Descriptions 3
SDLC Protocol Overview 4
Data Encoding 7
Bit Stuffing 7
Special Function Registers 8
Modes of Operation 15
General Description of the Transmitter 16
General Description of the Receiver 17
Ordering Information 20
List of Changes 21
Datasheet Categories 21

General Description

The CoreSDLC macro provides a high-speed synchronous serial communication controller that utilizes the synchronous data link control (SDLC) protocol. Operation of the controller is similar to that used in the Intel 8XC152 Global Serial Channel (GSC) device working in SDLC mode under CPU control. Communication with a CPU is realized through the Special Function Register (SFR) interface and three interrupt sources. This enables interfacing CoreSDLC easily with any CPU.

CoreSDLC consists of three primary blocks, as shown in Figure 1:

1. Receive logic – decodes and bit strips incoming data stream, detects flags, checks CRC, and shifts data into an internal three-byte deep receive FIFO. The receive logic also performs address detection, clock recovery, and frame sequencing.
2. Transmit logic – shifts data out of an internal three-byte deep transmit FIFO, generates a CRC, performs bit stuffing, flag insertion, and encoding of the transmit data stream. The transmit logic also performs frame sequencing.
3. SFR logic – provides a simple interface to an external processor or controller.

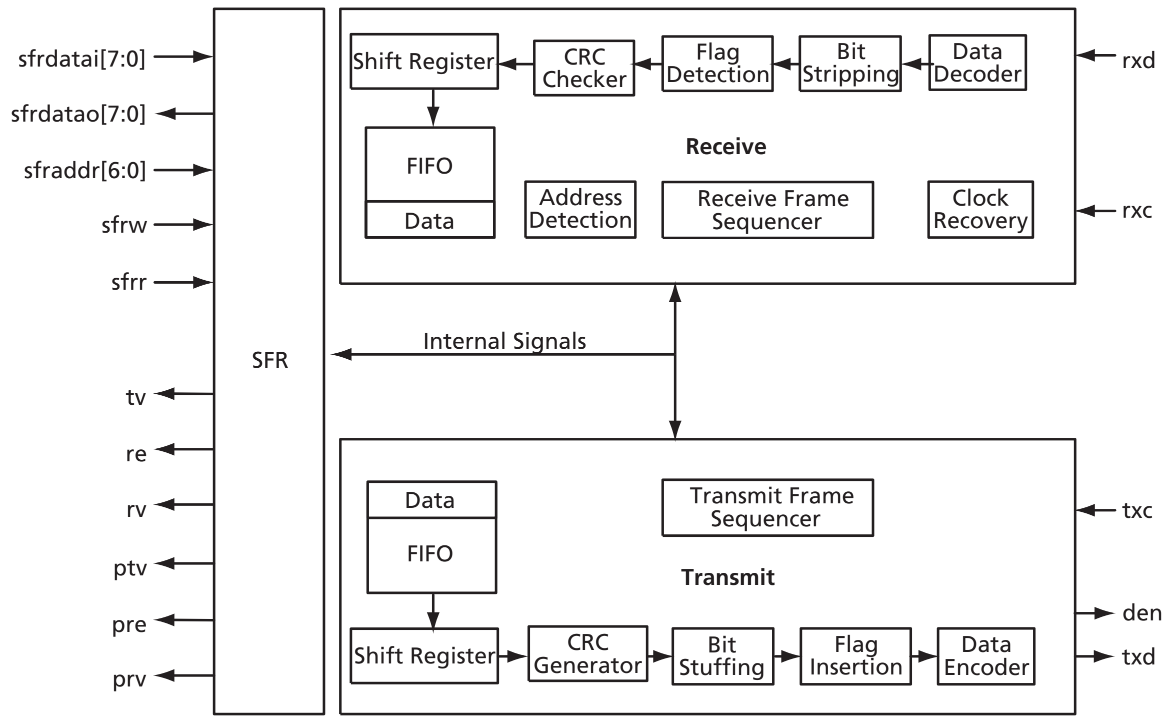


Figure 1 • **CoreSDLC Block Diagram**

CoreSDLC Device Requirements

CoreSDLC has been implemented in several of Actel's device families. A summary of the implementation data is listed in Table 1.

Table 1 • **CoreSDLC Device Utilization and Performance**

	Cells or Tiles			Utilization		
Family	**Sequential**	**Combinatorial**	**Total**	**Device**	**Total**	**Performance**
Fusion	408	878	1286	AFS600	10%	100 MHz
ProASIC3/E	408	878	1286	A3PE600-2	10%	100 MHz
ProASIC^{PLUS}	384	1337	1721	APA150-STD	28%	65 MHz
Axcelerator	400	537	577	AX125-3	47%	140 MHz

Note: Data in this table were achieved using typical synthesis and layout settings.

Table 1 • **CoreSDLC Device Utilization and Performance (Continued)**

Family	Cells or Tiles			Utilization		Performance
	Sequential	Combinatorial	Total	Device	Total	
SX-A	403	580	983	A54SX32A-3	35%	120 MHz
RTSX-S	391	545	936	RT54SX32S-2	33%	75 MHz

Note: Data in this table were achieved using typical synthesis and layout settings.

CoreSDLC Verification

The comprehensive verification simulation testbench (included with the Netlist and RTL versions of the core) verifies correct operation of the CoreSDLC macro with respect to the SDLC protocol.

The verification testbench applies over 90 tests to the CoreSDLC macro, including:

- Receive valid in normal mode tests
- Receive valid in raw mode tests
- Receive with errors tests
- Receive clock recovery tests
- Transmit tests

Using the supplied user testbench as a guide, the user can easily customize the verification of the core by adding or removing tests.

I/O Signal Descriptions

The port signals for the CoreSDLC macro are defined in Table 2 and illustrated in Figure 2 on page 4. All signals are either "Input" (input-only) or "Output" (output-only).

Table 2 • **CoreSDLC I/O Signal Descriptions**

Name	Type	Description
nreset	Input	Active-low asynchronous reset
clk	Input	System Clock: reference clock for all internal logic
sfrdatai[7:0]	Input	SFR data bus input
sfrdatao[7:0]	Output	SFR data bus output
sfraddr[6:0]	Input	SFR address bus
sfrwe	Input	SFR write enable
sfrra	Input	SFR read acknowledge
rv	Output	Receive valid interrupt
re	Output	Receive error interrupt
tv	Output	Transmit valid interrupt
prv	Output	Receive valid interrupt priority
pre	Output	Receive error interrupt priority
ptv	Output	Transmit valid interrupt priority
rxd	Input	Receive input
txd	Output	Transmit output
rxc	Input	Receive clock
txc	Input	Transmit clock
den	Output	Active-low external driver enable
rdn	Output	Receive done interrupt

Note: All signals are active-high unless otherwise indicated.

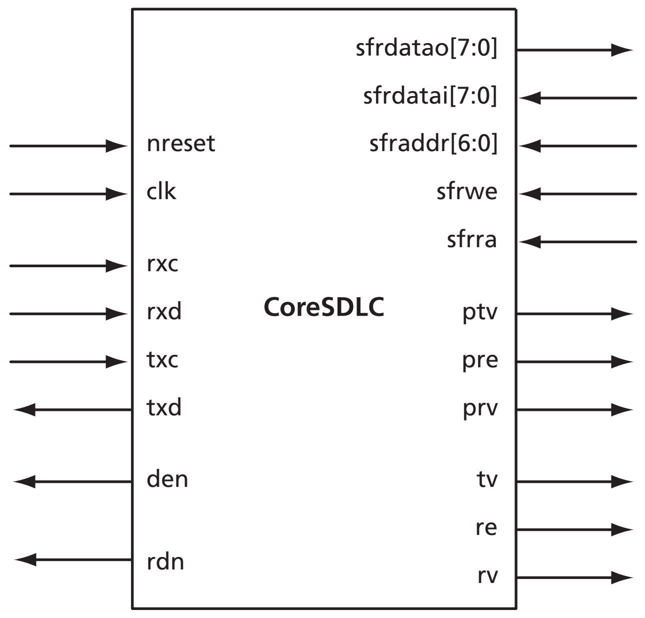


Figure 2 • **CoreSDLC I/O Signal Diagram**

SDLC Protocol Overview

The SDLC protocol has two types of network nodes: primary and secondary. There is always one primary node in the network, but there may be one or more secondary nodes. The primary node controls operation of the secondary nodes and manages the network. Secondary nodes can send information only if the primary node has given them permission. This is accomplished by the primary node polling the secondary nodes in a predetermined order to see if they need to send information.

As shown in Figure 3 on page 4, SDLC nodes are connected in one of the three following configurations:

- Point-to-point, when there is one primary and only one secondary node
- Multi-drop, when there is one primary and multiple secondary nodes
- Ring, when all nodes are connected in a loop and the output channel of one node is connected to the input channel of the next node.

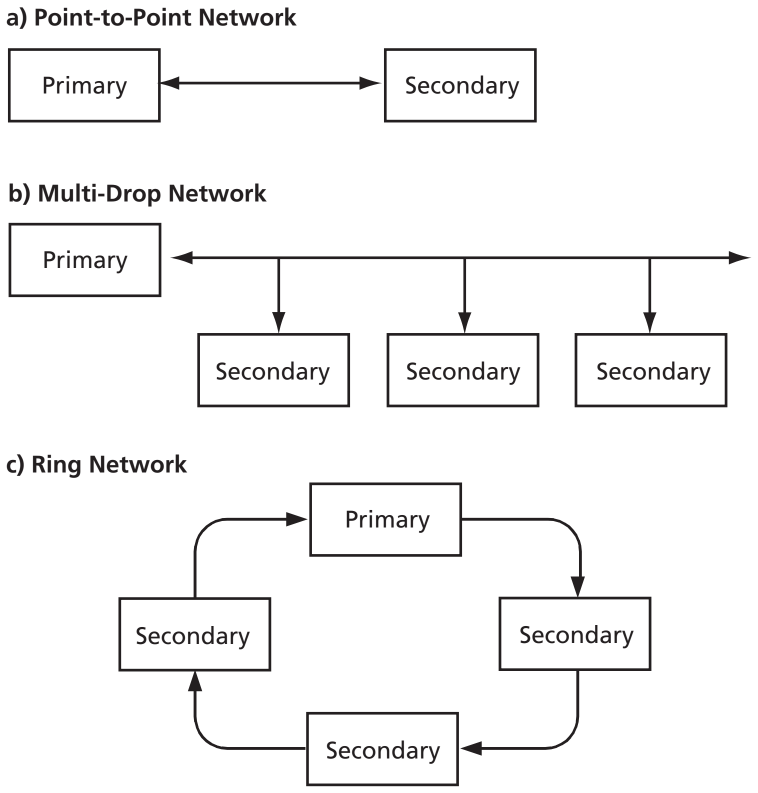


Figure 3 • **SDLC Network Configurations**

SDLC Frames

The SDLC frame consists of six fields. Table 3 shows the order of the fields in the SDLC frame.

Table 3 • **SDLC Frame**

BOF	ADDRESS	CONTROL	INFO	CRC	EOF

BOF (Begin of Frame)

The BOF flag, which indicates the beginning of a frame, is defined as the value 01111110. The controller's hardware properly distinguishes normal data from the BOF flag because of a process called *bit stuffing*, which is described in "Bit Stuffing" on page 7. Bit stuffing, performed by the transmit logic, is the process of inserting a '0' after each five consecutive '1' values. The receiver logic utilizes a process called *bit stripping*. Each time a sequence of five '1' values followed by a '0' is received, the controller automatically removes this '0' from the incoming bitstream. BOF is one of two possible bit combinations that consist of more than five consecutive '1' values. The BOF marks the beginning of a frame and also assures receive clock synchronization.

ADDRESS

In standard SDLC, an eight-bit field in the frame is used to identify the target controller for which the frame is intended. In CoreSDLC, this field may also be 16 bits in length, extending the addressing capability. The address length can be further extended by the user's software; however, the hardware address checking only works up to 16-bit addresses. There is also one special address defined in SDLC called "broadcast address," consisting of all '1' values. All stations connected to the network receive the frame containing the broadcast address. CoreSDLC transmits the address field's least significant bit (LSB) first.

CONTROL

This field is used for initializing the system and managing tasks, such as data acknowledge, identifying frame sequence numbers, and indicating the end of the message. CoreSDLC does not provide any functions for managing the CONTROL field, so the user's software is responsible for insertion and interpretation.

There are three types of control fields, depending on the type of SDLC frame used:

- Information frame (Table 4)
- Supervisory frame (Table 5)
- Nonsequenced (or unnumbered) frame (Table 6)

Table 4 • **Control Field – Information Format**

Bit Position	7	6	5	4	3	2	1	0
Function	Reception Sequence			Poll /Final	Sending Sequence			1

Table 5 • **Control Field – Supervisory Format**

Bit Position	7	6	5	4	3	2	1	0
Function	Reception Sequence			Poll / Final	Mode		0	1

Table 6 • **Control Field – Nonsequenced Format**

Bit Position	7	6	5	4	3	2	1	0
Function	Command / Response			Poll / Final	Command / Response		0	1

The CONTROL field of the information frame contains a three-bit sending sequence number (the number of the current frame) and a three-bit reception sequence number (the expected number of the next frame). The same reception sequence number is also part of the control field in the supervisory frame. In both cases, it is used for frame acknowledgement. If the receiving station has no data to send, it acknowledges the received frames by sending a supervisory frame in response. However, if the receiving station wants to send data, the response may be part of the information frame (piggybacking). This allows for full-duplex operation in which two continuous data streams are transmitted in both directions without supervisory frame insertion.

Up to seven information frames can be sent without acknowledgement. Due to this capability, continuous transmission (continuous ARQ) is possible, which means that the CoreSDLC transmitter does not need to wait for an acknowledgement.

The poll/final bit in each control field is used for polling secondary nodes by a primary node (poll) and for indicating the end of the message (final).

The supervisory frame contains an additional two mode bits, which affect the retransmission scheme. Although it is possible for four modes to exist, three of them are used in CoreSDLC:

- Receiver Ready (RR) – Indicates that the receive line of this station is ready to accept frames
- Receiver Not Ready (RNR) – Indicates that the receiver is not ready to accept frames (possible FIFO overflow)
- Reject (REJ) – Indicates that the previously received frame was rejected

The unnumbered (nonsequenced) frame contains five bits that indicate commands or responses used for initializing the network and eliminating errors. These commands include:

- Unnumbered information (UI)
- Set initialization mode (SIM)
- Disconnect (DISC)
- Response optional (UP)
- Function descriptor in information field (CFGR)
- Identification in information field (XID)
- Test pattern in information field (TEST)
- Request for initialization (RIM)
- Frame reject (FRMR)
- Unnumbered acknowledgment (UA)
- Signal loss of input (BCN)
- Station wants to disconnect (RD)
- Station in disconnected mode (DM)

INFO

This field contains data that is transmitted by one device to the other. It can be an arbitrary length, although it must be byte-aligned (its length must be a multiple of eight bits). It is possible that some frames may not contain an INFO field. The INFO field is transmitted LSB first.

CRC (Cyclic Redundancy Check)

This is the error checking sequence. It is a widely used method for detecting errors in messages transmitted over noisy channels. CoreSDLC offers two types of CRC: 16-bit CRC (often referred to as CRC-CCITT) and 32-bit CRC.

CRC is generated in the transmitter over the ADDRESS, CONTROL, and INFO fields before the bit-stuffing process. First, the CRC shift register is preset with an all '1' value. For each incoming data bit, the most significant bit (MSB) of the current CRC remainder is XORed with the data bit, and the remainder is shifted left with the LSB set to '0'. If the result of the XOR is '1,' the remainder is XORed with the generator polynomial.

For the 16-bit CRC, the polynomial is:

$$G(x) = x^{16} + x^{12} + x^{5} + 1$$

For the 32-bit CRC, the polynomial is:

$$G(x) = x^{32} + x^{26} + x^{23} + x^{22} + x^{16} + x^{12} + x^{11} + x^{10} + x^{8} + x^{7} + x^{5} + x^{4} + x^{2} + x + 1$$

After the last data bit has passed through the CRC generator, the current CRC generator value is inverted and sent to the receiver, MSB first.

The receiver operates exactly the same way as the transmitter by generating the CRC remainder over the incoming ADDRESS, CONTROL, INFO, and CRC fields. After all data has passed through, the current CRC remainder is checked. If no errors have occurred, this remainder will be equal to the polynomial residual. For a 16-bit CRC, this residual is 00011101 00001111 (0x1D0F). For a 32-bit CRC, it is 11000111 00000100 11011101 01111011 (0xC704DD7B). The CRC field is transmitted MSB first.

End of Frame (EOF)

The EOF flag consists of the same bit pattern as the BOF flag (01111110). EOF indicates when the transmission is completed and also can serve as the BOF for the next frame. Frames that share EOF and BOF flags are called *back-to-back frames*.

Data Encoding

CoreSDLC employs NRZI (Non-Return to Zero Inverted) data encoding when transmitting frames. In NRZI, a '1' is represented by no change in the output signal level, and a '0' is represented by a change in the level. A data stream consisting of all '0' values causes the NRZI output to toggle each bit time, while a stream of all '1' causes no transitions.

Although NRZI is typical for SDLC networks, CoreSDLC also performs NRZ (Non-Return to Zero) encoding. In this encoding method, a '1' is represented by a high level and a '0' by a low level.

Figure 4 shows an example of NRZ and NRZI encoding.

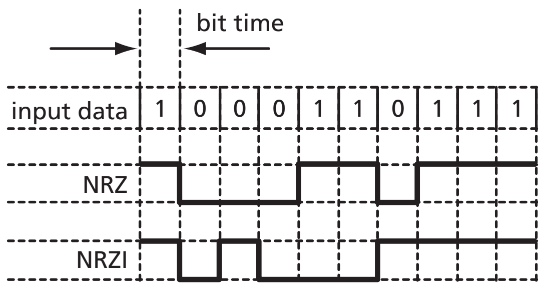


Figure 4 • **NRZI and NRZ Data Encoding**

Bit Stuffing

CoreSDLC employs bit stuffing to ensure a minimum number of signal transitions that are necessary for "recovering" the receiver clock. Bit stuffing is the process of inserting a '0' after every five consecutive '1' values in a data stream to force a transition in a NRZI output data stream. This guarantees that there will be at least one transition every six bit times while transmitting data.

The receiver must recognize the inserted bits and remove them from the data stream. This process is called bit stripping. Bit stuffing and stripping are automatically performed by CoreSDLC and are completely transparent to the user. Figure 5 shows an example of bit stuffing and bit stripping.

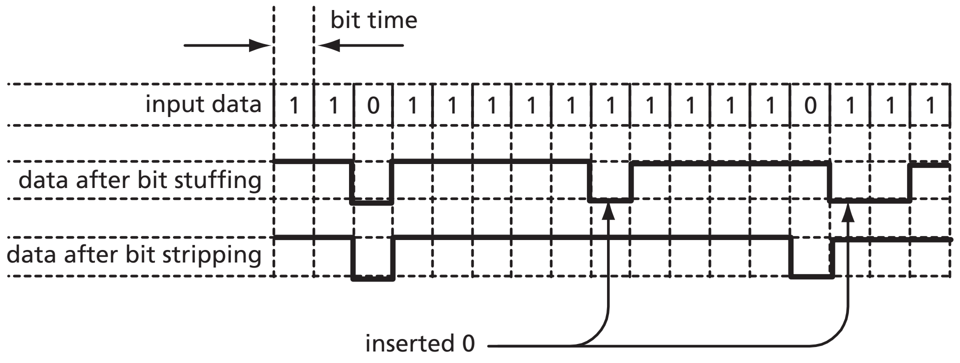
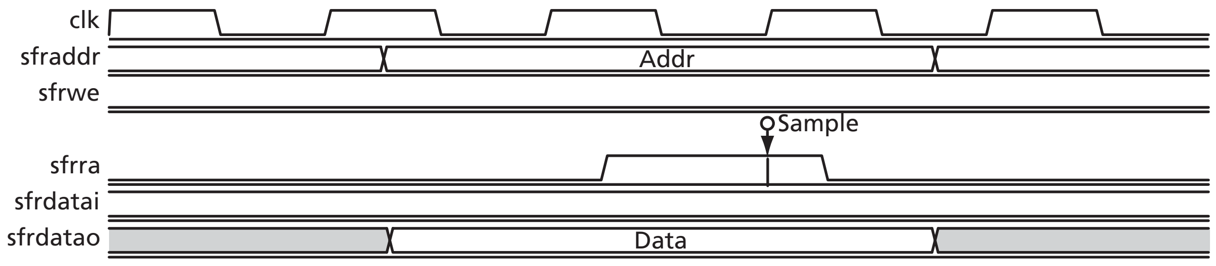


Figure 5 • **Bit Stuffing / Stripping**

Special Function Registers

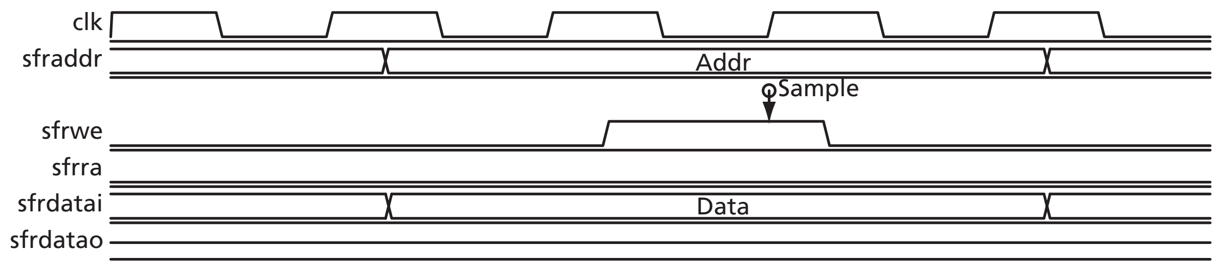
Special Function Registers Interface

The communication between CoreSDLC and the CPU is accomplished via the special function register (SFR) interface. An SFR read cycle is shown in Figure 6, and an SFR write cycle is shown in Figure 7 on page 8.



Note: Sample – point of registering the state of the signal into an internal flip-flop.

Figure 6 • **Special Function Register Read Cycle**



Note: Sample – point of registering the state of the signal into an internal flip-flop.

Figure 7 • **Special Function Register Write Cycle**

External Special Function Register Read Cycle

As shown in Figure 6 on page 7, the CPU reads an internal register within CoreSDLC by placing the address of the register on the sfraddr[6:0] input bus. The selected register presents its data onto the sfrdatao[7:0] output signals. The CPU acknowledges that data from the sfrdatao[7:0] bus has been read by setting the sfrra input signal.

External Special Function Register Write Cycle

As shown in Figure 7, the CPU writes to an internal register within CoreSDLC by placing the address of the register on the sfraddr[6:0] input bus, placing the data to be written on the sfrdatai[7:0] input bus, and by setting the sfrwe input signal to a logic '1' value. CoreSDLC samples the state of the sfrwe input signal. If sfrwe is active, the data is read from sfrdatai[7:0] and written into the selected register.

Special Function Register Map

The register map and reset values for each CPU-accessible register within CoreSDLC are shown in Table 7.

Table 7 • **Special Function Register Map**

Register	Address	Reset Value	Description
pcon	0x07	0x00	Power Control
gmod	0x04	0x00	GSC Mode
tfifo	0x05	0x00	Transmit FIFO
baud	0x14	0x00	Baud Rate
adr0	0x15	0x00	Address 0
ifs	0x24	0x00	Interframe Space
adr1	0x25	0x00	Address 1
adr2	0x35	0x00	Address 2
adr3	0x45	0x00	Address 3
ien1	0x48	0xC0	Interrupt Enable
amsk0	0x55	0x00	Address Mask 0
tstat	0x58	0x04	Transmit Status
amsk1	0x65	0x00	Address Mask 1
rstat	0x68	0x00	Receive Status
rfifo	0x74	0x00	Receive FIFO
ipn1	0x78	0xC0	Interrupt Priority

Special Function Register Descriptions

Table 8 on page 9 through Table 29 on page 14 describe the various CPU-accessible registers within CoreSDLC. Unless otherwise stated, each register can be read and written to by an external CPU.

Power Control Register (pcon)

Table 8 • **pcon Register**

MSB							LSB
–	–	–	garen	xrclk	gfien	–	–

Table 9 • **pcon Register Bit Functions**

Bit	**Symbol**	**Function**
7:5	–	Not used
4	garen	Auxiliary receive enable 1– The reception of back-to-back frames is enabled. The receiver is not disabled after receiving the EOF flag when this bit is set. 0 – Prevents reception of back-to-back frames. The receiver is disabled after receiving the EOF flag.
3	xrclk	External receive clock 1 – External clock and NRZ encoding scheme used by receiver 0 – Internal clock generator and NRZI encoding used by receiver
2	gfien	Flag idle enable 1 – Idle flags (01111110) are generated between transmitted frames representing the sequence 01111110 01111110. . . 0 – No idle flag generation
1:0	–	Not used

Note: This register has unimplemented bits (–). Unless otherwise noted, if these bits are read they will return '0'. Writing to these bits has no effect.

GSC Mode Register (gmod)

Table 10 • **gmod Register**

MSB							LSB
xtclk	m1	m0	a1	ct	pl1	pl0	–

Table 11 • **gmod Register Bit Functions**

Bit	**Symbol**	**Function**
7	xtclk	External transmit clock 1 – External clock and NRZ encoding used by transmitter 0 – Internal clock generator and NRZI encoding used by transmitter
6:5	m1 m0	Mode select 00 – normal 01 – raw transmit 10 – raw receive 11 – not allowed
4	a1	Address length 1 – 16-bit addressing is used 0 – 8-bit addressing is used
3	ct	The CRC type 1 – 32-bit CRC is used 0 – 16-bit CRC (CRC-CCITT) is used

Table 11 • **gmod Register Bit Functions (Continued)**

Bit	Symbol	Function
2:1	pl1 pl0	Preamble length 00 – 0 bit 01 – 8 bits 10 – 32 bits 11 – 64 bits
0	–	Not used

Note: This register has unimplemented bits (–). Unless otherwise noted, if these bits are read they will return '0'. Writing to these bits has no effect.

Transmit FIFO Register (tfifo)

Table 12 • **tfifo Register**

MSB							LSB
tfifo.7	tfifo.6	tfifo.5	tfifo.4	tfifo.3	tfifo.2	tfifo.1	tfifo.0

The tfifo register represents the three-byte deep transmit FIFO. Writing a byte to this register loads data into the transmit FIFO and automatically updates the FIFO pointers. Setting the ten bit in the tstat register clears the transmit FIFO. The tfifo is a write-only register from the perspective of the CPU.

Receive FIFO Register (rfifo)

Table 13 • **rfifo Register**

MSB							LSB
rfifo.7	rfifo.6	rfifo.5	rfifo.4	rfifo.3	rfifo.2	rfifo.1	rfifo.0

The rfifo register represents the three-byte deep receive FIFO. Reading a byte from rfifo loads a value from the receive FIFO and automatically updates the FIFO pointers. Setting the gren bit in the rstat register clears the receive FIFO. The rfifo is a read-only register from the perspective of the CPU.

Baud Rate Register (baud)

Table 14 • **baud Register**

MSB							LSB
baud.7	baud.6	baud.5	baud.4	baud.3	baud.2	baud.1	baud.0

The baud register is used to set the value of an internal programmable baud rate generator. The baud rate generator operates by down-counting the baud register. When baud decrements to an all '0' value, it is reloaded. Writing a value into baud stores the value in the reload register. Reading it gives the current count value.

The baud rate can only be programmed in multiples of 1/8th of the clk input frequency. This is accomplished by entering the appropriate value into the baud register as shown in the following formula:

baud rate = clk / ((baud + 1) x 8)

For example, if the clk input frequency is 20 MHz, and the baud register is set to 0x01, the baud rate will be set to 1.25 Mbps.

Address Match Registers

Table 15 • **adr0 Register**

MSB							LSB
adr0.7	adr0.6	adr0.5	adr0.4	adr0.3	adr0.2	adr0.1	adr0.0

Table 16 • **adr1 Register**

MSB							LSB
adr1.7	adr1.6	adr1.5	adr1.4	adr1.3	adr1.2	adr1.1	adr1.0

Table 17 • **adr2 Register**

MSB							LSB
adr2.7	adr2.6	adr2.5	adr2.4	adr2.3	adr2.2	adr2.1	adr2.0

Table 18 • **adr3 Register**

MSB							LSB
adr3.7	adr3.6	adr3.5	adr3.4	adr3.3	adr3.2	adr3.1	adr3.0

The address match registers contain values that are compared with the address in received frames. In an eight-bit addressing mode, the address match occurs when one of these four address values match. In a 16-bit addressing mode, address registers are combined into two 16-bit registers: adr1:adr0 and adr3:adr2. An address match occurs when one of these two 16-bit registers trigger a match.

Address registers are used only in the receive operation. When CoreSDLC transmits, the frame address is treated as normal data. Therefore, the user's software is responsible to load the address bytes into the transmit FIFO before other data.

Address Mask Registers (amsk0-1)

Table 19 • **amsk0 Register**

MSB							LSB
amsk0.7	amsk0.6	amsk0.5	amsk0.4	amsk0.3	amsk0.2	amsk0.1	amsk0.0

Table 20 • **amsk1 Register**

MSB							LSB
amsk1.7	amsk1.6	amsk1.5	amsk1.4	amsk1.3	amsk1.2	amsk1.1	amsk1.0

Bits in the address mask registers, amsk1 and amsk0, correspond to bits in the registers adr1 and adr0, respectively. Setting a mask register bit to '1' causes the corresponding bit in the address register to be omitted during the address matching process.

Transmit Status Register (tstat)

Table 21 • **tstat Register**

MSB							LSB
lni	–	–	–	tdn	tfnf	ten	–

All bits of the tstat register are read only, except for bit 1, which is read/write.

Table 22 • **tstat Register Bit Functions**

Bit	Symbol	Function
7	lni	Line idle 1 – receive line is idle (15 consecutive '1' values are received on rxd) 0 – receive line is not idle
6:4	–	Not used
3	tdn	Transmit done This bit is set after successful completion of a frame transmission and cleared after setting the ten bit.
2	tfnf	Transmit FIFO not full When set indicates that new data may be written into tfifo
1	ten	Transmit enable Setting this flag clears tdn and tfifo and enables transmission. This bit is automatically cleared after the end of transmission. If this bit is cleared to a '0' before the end of transmission, that transmission is aborted.
0	–	Not used

Note: This register has unimplemented bits (–). Unless otherwise noted, if these bits are read they will return '0'. Writing to these bits has no effect.

Receive Status Register (rstat)

Table 23 • **rstat Register**

MSB							LSB
ovr	rcabt	ae	crce	rdn	rfne	gren	–

Table 24 • **rstat Register Bit Functions**

Bit	Symbol	Function
7	ovr	Overrun error If set, ovr indicates that receive FIFO was full when attempting to store new data. The user can clear the ovr bit by setting the gren bit.
6	rcabt	Abort detect If set, rcabt indicates that seven consecutive '1' values were received before the EOF flag but after data had been loaded into rfifo. The user can clear the rcabt bit by setting the gren bit.
5	ae	Alignment error If set, ae indicates that a non byte-aligned flag was received after data had been loaded into rfifo. The user can clear the ae bit by setting the gren bit.
4	crce	The CRC error If set, crce indicates that a frame was received with a mismatched CRC. The user can clear the crce bit by setting the gren bit.

Table 24 • **rstat Register Bit Functions (Continued)**

Bit	Symbol	Function
3	rdn	Receive done When set, rdn indicates successful completion of a frame receive operation. The user can clear the rdn bit by setting the gren bit.
2	rfne	Receive FIFO not empty When set, rfne indicates that new data can be read from rfifo
1	gren	Receive enable Setting this flag enables the receiver and clears the ovr, rcabt, ae, crce and rdn bits. This bit is automatically cleared after the end of the receive operation.
0	–	Not used

Note: This register has unimplemented bits (–). Unless otherwise noted, if these bits are read they will return '0'. Writing to these bits has no effect.

All bits of the rstat register are read only, except for bit 1, which is read/write.

Interframe Space Register (ifs)

Table 25 • **ifs Register**

MSB							LSB
ifs.7	ifs.6	ifs.5	ifs.4	ifs.3	ifs.2	ifs.1	ifs.0

The ifs register determines the minimum number of bit times that must elapse between two consecutive transmitted frames. CoreSDLC only takes the seven most significant bits of the written value (only even numbers can be used) and computes the interframe space by counting this seven-bit number down to an all '0' value twice. When read by the user's software, the seven most significant bits of the ifs register show the current count value, while the least significant bit is a '1' for first counting and a '0' for the second.

Interrupt Enable Register (ien1)

Table 26 • **ien1 Register**

MSB							LSB
–	–	–	–	egstv	–	egsre	egsrv

Table 27 • **ien1 Register Bit Functions**

Bit	Symbol	Function
7:4	–	Not used (fixed at 1100)
3	egstv	Transmit valid interrupt enable
2	–	Not used
1	egsre	Receive error interrupt enable
0	egsrv	Receive valid interrupt enable

Note: This register has unimplemented bits (–). Unless otherwise noted, if these bits are read they will return ‘0’. Writing to these bits has no effect.

Interrupt Priority Register (ipn1)

Table 28 • **ipn1 Register**

MSB							LSB
–	–	–	–	pgstv	–	pgsre	pgsrv

Table 29 • **ipn1 Register Bit Functions**

Bit	Symbol	Function
7:4	–	Not used (fixed at 1100)
3	pgstv	Transmit valid interrupt priority
2	–	Not used
1	pgsre	Receive error interrupt priority
0	pgsrv	Receive valid interrupt priority

Note: This register has unimplemented bits (–). Unless otherwise noted, if these bits are read they will return '0'. Writing to these bits has no effect.

Modes of Operation

CoreSDLC provides three modes of operation: normal mode, raw receive mode, and raw transmit mode. The first is normally used in standard communications within SDLC networks, while the others may be used for testing the controller's operation or transmit user data, (not necessarily SDLC-formatted). Information about all operational modes, as well as a summary of options available for each mode, are described in detail in the following text and in Table 30 on page 15. Each of the three modes of operation are selected by the CPU setting the m1 and m0 bits in the gmod register, as listed in Table 11 on page 9.

Normal Mode

In normal mode, data is transmitted in standard SDLC format. After transmission is enabled by the CPU setting the ten bit in the tstat register and loading the transmit FIFO with data, CoreSDLC tests if the interframe space (time from previous transmission) has expired. If this condition is met, the den (external driver enable) output is forced low. One bit time later, CoreSDLC begins transmission by sending the appropriate number of preamble bits and the BOF flag. Immediately after BOF is sent, a byte from tfifo is loaded into the shift register. As bits are shifted out of this register, they also pass through the CRC generator, updating the current CRC value. This process is performed as long as the transmit FIFO contains data. If the FIFO is empty when the transmitter is about to load the next byte, CoreSDLC assumes "end of data." Transmission ends with sending the current CRC generator value followed by the EOF flag. All transmitted bits are encoded with NRZI (if the internal clock generator is selected) or NRZ (if external clock input is selected).

The receiver working in normal mode searches the input for the BOF flag. Immediately after BOF is detected, the frame's address field is checked if it matches the address assigned with address registers addr3-0. When the address does not match, that frame is ignored. If the address matches, the receiver loads incoming bits including ADDRESS, CONTROL, and INFO fields into the shift register and then into the receive FIFO. The CRC is not loaded into the receive FIFO.

Raw Transmit Mode

In raw transmit mode, the transmit output is internally connected to the receive input. All data written to the transmit FIFO are transmitted without preamble, BOF and EOF flags, address, and CRC. Additionally, bit stuffing is disabled. The receiver operates as normal in this mode.

Raw transmit mode can be used for receiver testing or for transmitting user data (not necessarily SDLC-formatted).

Raw Receive Mode

In raw receive mode, the transmitter operates as in normal mode.

The receiver also operates as normal except that all bytes between the BOF and EOF flags are loaded into the receive FIFO, including the CRC field. The receiver does not check the CRC and no CRC error is set. In addition, address matching is not performed, and therefore, all frames are received.

To use raw receive as a test mode, the transmit output should be externally connected to the receive input. This allows most of the transmitter functions, as well as the external transceiver, to be checked.

Table 30 • **Functions Available in Individual Modes**

Function / Mode	**Normal (m1=0, m0=0)**		**Raw Transmit (m1=0, m0=1)**		**Raw Receive (m1=1, m0=0)**	
	Transmit	**Receive**	**Transmit**	**Receive**	**Transmit**	**Receive**
Preamble	O	NA	N	NA	O	NA
BOF and EOF	Y	Y	N	Y	Y	Y
Address Matching	NA	Y	NA	Y	NA	N
CRC check	Y	Y	N	Y	Y	N
Bit stuffing and stripping	Y	Y	N	Y	Y	Y
NRZ / NRZI	Y	Y	Y	Y	Y	Y

Notes:

1. m1, m0 – gmod register mode bits in Table 11 on page 9
2. Y – used
3. N – not used
4. O – optional
5. NA – not applicable

General Description of the Transmitter

Interframe Spacing

Interframe space is a period of time that must elapse between two consecutive transmissions. It is measured in bit times. Interframe space can be set by writing an appropriate value (number of bit times) into the ifs register. Note that only even numbers can be used (the LSB must always be set to '0'), because only the seven most significant bits are loaded into the ifs register. This means that interframe space can be set from two bit times to 256 bit times. A value of 0x02 written into ifs corresponds to two bit times, 0xFE corresponds to 254 bit times, while 0x00 corresponds to 256 bit times.

Preamble

The preamble is a series of toggling '1' and '0' values. The length of preamble can be set to 0, 8, 32 or 64 bits by writing appropriate values into pl1 and pl0 bits in the gmod register. The preamble is not defined in the standard SDLC protocol and thus it is not considered part of the SDLC frame. The purpose of the preamble is only for synchronization between stations in the network.

Note that if idle flags are used in conjunction with a preamble, the addresses 0x00 and 0x55 should not be assigned to the controller. Otherwise, a preamble following the idle flags will be interpreted as a matching address.

Sending an Abort Flag

An abort flag is the sequence of seven or more '1' values. If the receiver detects an abort flag between EOF and BOF, it immediately ends reception. There are three ways to generate an abort flag using CoreSDLC:

- One method is to clear the ten bit in the tstat register and wait at least seven bit times. In this case, the delay necessary to transmit seven bits must be measured by the user's software.
- The second method is based on programmable interframe space. The first step is to write into the ifs register, a value greater than or equal to eight. Then the user's software must clear the ten bit, which disables transmission and forces the output to a high level. With this method, the ten bit can be immediately re-enabled. The output will remain at a high level until the interframe space expires, which is accomplished automatically by CoreSDLC.
- The third method is to use raw transmit mode. Writing a value of 0xFF into tfifo generates a high output for eight bit times. This is possible because the transmitter does not use bit stuffing in raw transmit mode.

External Driver Interface

CoreSDLC uses the den output for an external driver interface. This output is activated one bit time before transmission begins and remains active until the last bit of the EOF is transmitted, or until the ten bit is cleared by the user's software.

Transmission with an External Clock

An external clock is selected by setting the xtclk bit in the gmod register. When this bit is set, NRZI decoding is also disabled. Data is transmitted with the NRZ decoding scheme on the falling edge of the txc clock. Due to the txc input synchronization with the global clock, there can be a delay of up to two clock periods until the data begins to transmit, as shown in Figure 8 and Table 31.

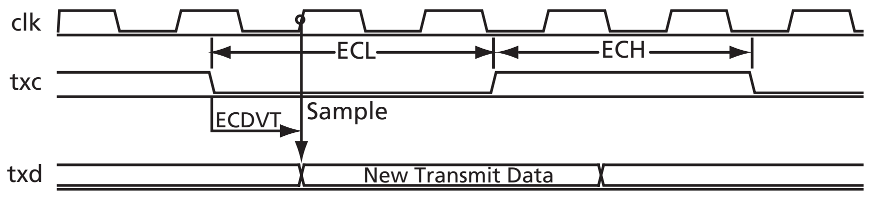


Figure 8 • **External Transmit Clock Timing**

Table 31 • **External Transmit Clock Timing**

Symbol	**Parameter**	**Minimum Value**
ECL	External clock low	2*(period of clk)
ECH	External clock high	2*(period of clk)
ECDVT	External clock to data valid transmit	1 to 2*(period of clk)

General Description of the Receiver

Receive Clock Recovery

In synchronous serial protocols like SDLC, data and clock are both transmitted over the same medium. Receiver clock recovery is the process of separating the clock signal from the incoming data bitstream (Figure 9). The receiver performs this action. In CoreSDLC, the receiver input is always monitored at eight times the baud rate frequency and searched for the level transitions. Every transition causes the receiver to correct its own clock for proper synchronization with the transmitter. Additionally, CoreSDLC performs digital filtering by ignoring input pulses shorter than four baud-rate periods.

The only exception to this rule is when the xrclk bit in the pcon register is set. In that case, an external receive clock and NRZ decoding are used. Receive data input is then sampled on the external clock rising edge. No clock recovery and no digital filtering are performed in that case.

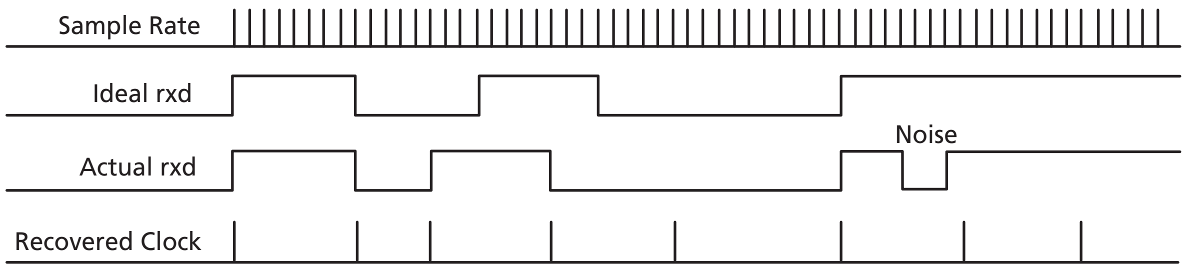


Figure 9 • **Receive Clock Recovery**

Receive Error Conditions

CoreSDLC detects four kinds of receive errors represented by bits in the rstat register (Table 32):

- crce – CRC error
- ae – alignment error
- rcabt – receive abort
- ovr – overrun in receive FIFO

The user's software can read these bits, but only CoreSDLC can write them in response to the various error conditions that they represent. When an error occurs, CoreSDLC sets one or more of these bits and also clears the gren bit in the rstat register. When the user's software sets the gren bit re-enabling the receiver, all error bits are cleared. This is the only method for clearing error bits.

It is possible that multiple error bits get set in response to certain errors:

- rcabt and ae can be set when receiving misaligned abort flag
- ovr and crce can be set when an overrun error is forced
- ae and crce can be set when an alignment error occurs

In order to determine the correct cause of the receiver error, the user's software should poll error bits in the following sequence:

1. rcabt
2. ovr
3. crce
4. ae

Table 32 • **Receive Error Conditions**

Error Bit	Condition
crce	This bit will be set if the CRC remainder after passing all ADDRESS, CONTROL, INFO and CRC fields through the CRC generator is not equal to the polynomial residual. For a 16-bit CRC, this residual is 00011101 00001111, and for a 32-bit CRC it is 11000111 00000100 11011010 01111011. This bit will also be set when alignment or overrun errors occur.
ae	This bit will be set if the number of bits received between BOF and EOF flags are not a multiple of eight (INFO field is not byte-aligned). This bit will also be set when the abort flag is detected.
rcabt	This bit will be set if the receiver detects an abort sequence (7 or more consecutive '1' values) in an incoming frame between the BOF and EOF flags but after the first received data has already passed to the receive FIFO. If the abort flag is detected before loading the first byte into the FIFO, the incoming frame is ignored and no error bits are set.
ovr	This flag will be set if the receiver gets new data and the receive FIFO is already full.

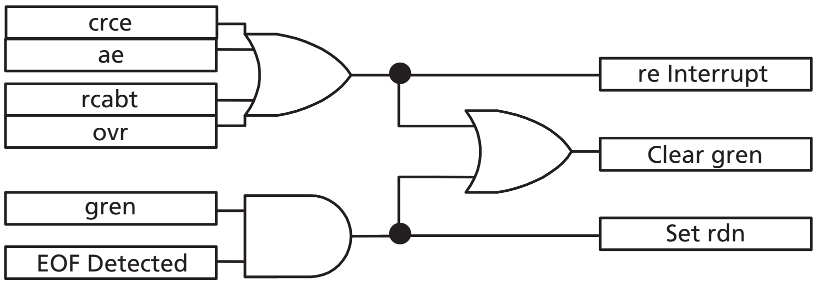
Receive Enable Bits

There are two receive enable bits: gren (receive enable, in the rstat register) and garen (auxiliary receive enable, in the pcon register). In order to enable the receiver, at least one of these bits should be set. Although setting only the garen bit enables the receiver, the rdn (receive done, in the rstat register) bit, which indicates the end of a valid reception, will only be set if the gren bit is set (Figure 10 on page 18).

When the frame reception is in process, clearing both gren and garen causes the receiver to end reception. There will be a 0 to 1 bit time delay between clearing the receive enable bits and end of reception.

Receive with External Clock

The external clock is selected by setting the xrclk bit in the pcon register. When this bit is set, NRZI encoding is also disabled. Data is received with NRZ decoding on the rxc clock rising edge. Due to rxc input synchronization with the global clock, as shown in Figure 11 on page 18 and listed in Table 33 on page 18, there is a maximum of one clock period delay from rxc rising edge to receive data (ECDR).



Note: crce, ae, rcabt, ovr and gren are rstat special function register bits. The re interrupt is an external interrupt output pin. The "clear gren" and "set rdn" actions are both taken automatically by CoreSDLC.

Figure 10 • **Receive Error Logic**

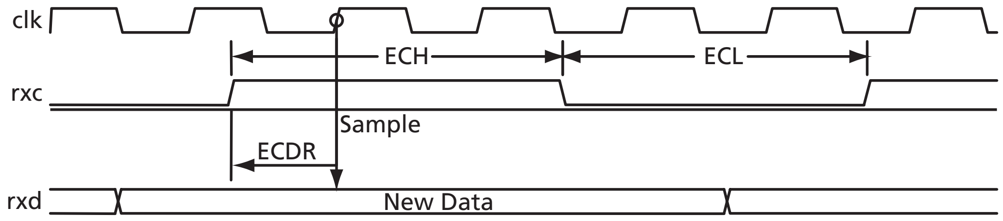


Figure 11 • **External Receive Clock Timing**

Table 33 • **External Receive Clock Timing**

Symbol	**Parameter**	**Minimum Value**
ECL	External clock low	2*(period of clk)
ECH	External clock high	2*(period of clk)
ECDR	External clock to receive data sample	0 to 1*(period of clk)

Interrupt Structure

There are three interrupt sources in CoreSDLC: transmit valid, receive valid, and receive error interrupt (Table 34).

The transmit valid interrupt flag is set when tfnf (transmit FIFO not full) is set. End of frame transmission is not indicated by an interrupt, so the user must poll the tdn (transmit done) bit in order to know if transmission has ended.

The receive valid interrupt flag is set when rfne (receive FIFO not empty) is set. End of frame reception is not indicated by an interrupt, so the user must poll the rdn (receive done) bit in order to know if reception has ended.

The receive error interrupt is set in response to a receive error indicated by any of the error bits in the rstat register. These error bits are: crce, ae, rcabt, and ovr.

Table 34 • **Interrupt Summary**

Name	**Output**	**Enable Bit**	**Priority Bit**	**Condition**
Transmit Valid	tv	egstv	pgstv	This flag is set when tfnf is set (transmit FIFO is not full). Setting the ten bit by the user's software will clear this flag.
Receive Valid	rv	egsrv	egstv	This flag is set when rfne is set (receive FIFO is not empty). Setting the gren bit by the user's software will clear this flag.
Receive Error	re	egsre	pgstv	This flag is set when at least one of the error bits (crce, ae, rcabt, ovr) is set. Setting the gren bit by the user's software will clear this flag.

Clock and Reset Control

CoreSDLC is fully synchronous with respect to the global clock clk. In other words, there is only one clock domain in the core. All internal registers operate synchronous to the rising edge of clk. All input signals (except the reset signal nreset) including txc and rxc, are sampled with the rising edge of clk. The nreset input signal is asynchronous with respect to the global clock clk. For proper operation, nreset should be active for at least one global clock period. When nreset is active, all registers return to their default states.

Ordering Information

Order CoreSDLC through your local Actel sales representative. Use the following numbering convention when ordering: CoreSDLC-XX, where XX is listed in Table 35.

Table 35 • **Ordering Codes**

XX	Description
EV	Evaluation Version
SN	Netlist for single-use on Actel devices
AN	Netlist for unlimited use on Actel devices
SR	RTL for single-use on Actel devices
AR	RTL for unlimited use on Actel devices
UR	RTL for unlimited use and not restricted to Actel devices

List of Changes

The following table lists critical changes that were made in the current version of the document.

Previous Version	Changes in Current Version (v4.0)	Page
v3.0	The "Supported Families" section was updated to include Fusion.	1
	Table 1 was updated to include Fusion data.	2
v2.0	"Supported Families" section was updated to include ProASIC3/E.	1
	Table 1 • CoreSDLC Device Utilization and Performance was updated to include ProASIC3/E data.	2

Datasheet Categories

In order to provide the latest information to designers, some datasheets are published before data has been fully characterized. Datasheets are designated as "Product Brief," "Advanced," and "Production." The definition of these categories are as follows:

Product Brief

The product brief is a summarized version of an advanced or production datasheet containing general product information. This brief summarizes specific device and family information for unreleased products.

Advanced

This datasheet version contains initial estimated information based on simulation, other products, devices, or speed grades. This information can be used as estimates, but not for production.

Unmarked (production)

This datasheet version contains information that is considered to be final.

www.actel.com

Actel Corporation

2061 Stierlin Court
Mountain View, CA
94043-4655 USA

Phone 650.318.4200
Fax 650.318.4600

Actel Europe Ltd.

Dunlop House, Riverside Way
Camberley, Surrey GU15 3YL
United Kingdom

Phone +44 (0) 1276 401 450
Fax +44 (0) 1276 401 490

Actel Japan
www.jp.actel.com

EXOS Ebisu Bldg. 4F
1-24-14 Ebisu Shibuya-ku
Tokyo 150 Japan

Phone +81.03.3445.7671
Fax +81.03.3445.7668

Actel Hong Kong
www.actel.com.cn

Suite 2114, Two Pacific Place
88 Queensway, Admiralty
Hong Kong

Phone +852 2185 6460
Fax +852 2185 6488

51700016-2/12.05