



The SDLC controller is a synthesizable HDL core providing a high-speed synchronous serial communication interface.

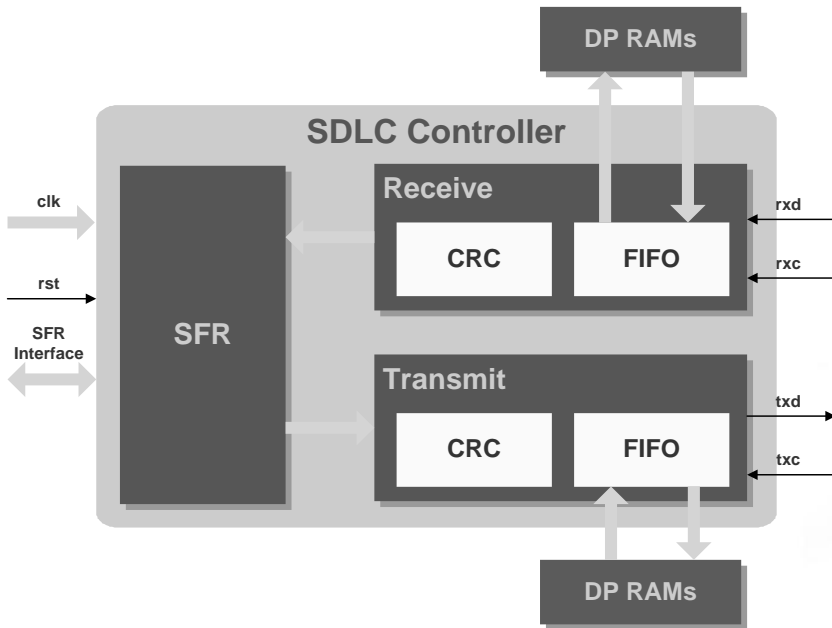
Operation of the controller is similar to that used in the Intel 8XC152 Global Serial Channel (GSC) working in SDLC mode under a CPU control. Communication with the CPU is realized through Special Function Registers (SFRs) and three interrupt sources. This allows the SDLC controller to be easily integrated with any CPU core.

The design is strictly synchronous, with positive-edge clocking, no internal tri-states and a synchronous reset.

Applications

- ISDN D-channel
- X.25 networks
- Frame Relay networks
- Custom serial interfaces

Block Diagram



Features

- Based on Intel's 80C152 Global Serial Channel
- Flexible addressing schemes
 - ▶ Single and double byte address recognition
 - ▶ Address filtering allowing multicast and broadcast addresses
- 16-bit CCITT or 32-bit frame check sequence
- NRZ or NRZI data encoding
- Full or half duplex operation
- Automatic bit stuffing/stripping
- 64-byte internal receive and transmit FIFOs
- Large FIFO sizes available
- External or internally generated transmit and receive clocks
- Optional preamble generation
- Programmable interframe space
- Raw transmit and receive testing modes

Benefits

- Simple use, low area design
- Optimized to support low cost and low power requirements of medium and low traffic systems
- Can be used as a general purpose synchronous interface with only single line (clock recovered from a data stream)
- Easy integration with SoC systems through a generic 8-bit interface

Functional Description

The SDLC core is partitioned into modules as shown in the block diagram and described below.

SFR Unit

This unit is responsible for communication with the external host controller. In a typical application, the external host may be a simple 8-bit embedded controller such as the 8051. Communication with that host is performed via a set of directly accessible 8-bit registers, and three interrupt lines. These interrupts notify the external CPU to write transmit data into the FIFO, to read received data, and about receive errors. The SFR unit also contains a baud rate generator used by the Transmit and Receive units.

Transmit Unit

This unit controls the transmit operation of the SDLC and is subdivided into several components responsible for generating properly formatted SDLC/HDLC frames. The whole unit is synchronous to the same global clock, which also feeds the SFR and Receive units.

The transmit frame sequencer is the main state machine responsible for generating all control signals used during frame formatting. Incoming data from the CPU goes directly into the transmit FIFO. After a transmit initialization sequence, the transmit frame sequencer automatically appends the HDLC frame flag and the optionally generated preamble sequence. It then periodically instructs the FIFO for writing data into the shift register.

After passing the shift register, the data goes to the crc generator, which computes the frame check sequence. Then the data goes to the bit stuffing unit which performs ones insertion procedure defined by the HDLC/SDLC protocol. The data encoder block forms data with NRZ or NRZI algorithms according to the current operating mode. If there is no transmit request, the Transmit unit can be idle or can generate a continuous idle flag sequence.

Receive Unit

This unit controls the receive operation of the SDLC and is subdivided into several components responsible for receiving properly formatted SDLC/HDLC frames. The whole unit is synchronous to the same global clock, which also feeds the SFR and Transmit units.

Incoming data together with clock information passes through the data decoder which separates data from the incoming bit stream. The same bit stream also feeds the clock recovery unit, which detects and recovers the receiver clock signal. Decoded data goes through the bit stripping circuit to remove any appended zeros and then to the flag detector.

The flag detector recognizes flags defined by the HDLC/SDLC and indicates events such as the start of a new frame, end of a frame, abort or idle state on the line. This indication is used by the frame sequencer unit, the main state machine of the Receive component. After passing the flag detector, data goes to the crc checker unit and to the shift register. The frame sequencer periodically instructs the shift register to write data into the receive FIFO.

The address check component performs receive address recognition and decides if the incoming frame should proceed. If the address checking fails, the frame is ignored by the other components.

The Receive unit is also able to detect frame errors such as missed crc checksum, misaligned flags, receive FIFO overflow or abort during a receive process.

Implementation Results

The following are typical performance and utilization results using various Lattice devices.

Lattice Device	LUT-4s	Registers	PFUs	SysMEM EBRs	External I/Os	Speed (f _{max} , MHz)
LFX125C-4	1568	408	411	2	39	81.0
OR4E02-3	1096	346	152	2	39	89.5

Lattice Device	LUT-4s	Registers	Slices	SysMEM EBRs	External I/Os	Speed (f _{max} , MHz)
LFE-2-50E-7	653	344	457	2	39	155
LFE-2M35E-7	596	344	416	2	39	156
LFXP2-17E-7	640	344	454	2	39	120
LFSC3GA25E-7	615	344	400	2	39	219

Options and Modifications

The SDLC core can be modified to include features such as:

- Custom interfaces for external CPU
- Custom baud rate generation variants
- Variable size of internal FIFOs

Support

The core as delivered is warranted against defects for ninety days from purchase. Thirty days of phone and email technical support are included, starting with the first interaction. Additional maintenance and support options are available.

Verification

The SDLC core's functionality was verified by means of a proprietary hardware modeler. The same stimulus was applied to a hardware model that contained the original Intel 80C152 chip, and the results compared with the core's simulation outputs.

Deliverables

The core includes everything required for successful implementation:

- Post-synthesis EDIF netlist (firm core) optimized for a specific Lattice device (HDL RTL source code (soft core) is also available)
- Testbenches (self-checking)
- Example testbench wrapper for post-route simulation Simulation script, vectors, and expected results
- Synthesis (soft) or place and route (firm) script
- Comprehensive user documentation