# 8b/10b Encoder/Decoder

Introduction

Many serial data transmission standards utilize 8b/10b encoding to ensure sufficient data transitions for clock recovery. This reference design describes an encoder/decoder suitable for performing 8b/10b encoding/decoding within Lattice programmable logic devices (PLDs). This encoding/decoding logic can be combined with the high-speed serial interface blocks (sysHSI™ blocks) contained in the ispXPGA™ and ispGDX2™ devices to support high speed interfaces that utilize this encoding scheme. Two CPLD implementations for the ispGDX2 device family along with the ispXPGA implementation are provided for use with sysHSI blocks. A generic FPGA implementation in ORCA Series 4 FPGA is also shown with this reference design.

Features

- 8b to 10b encoder and 10b to 8b decoder
- Previous octet disparity input and current disparity output
- Output to indicate when invalid control character is requested to be encoded
- Output to indicate when invalid data/control character is received
- Running disparity checking
- Conform to 8b/10b specified in IEEE 802.3z and ANSI X3.230-1994
- Optimized for ispXPGA and Lattice CPLDs to work with sysHSI in ispXPGA and ispGDX2

Figure 1. The 8b/10b Encoder/Decoder in a System
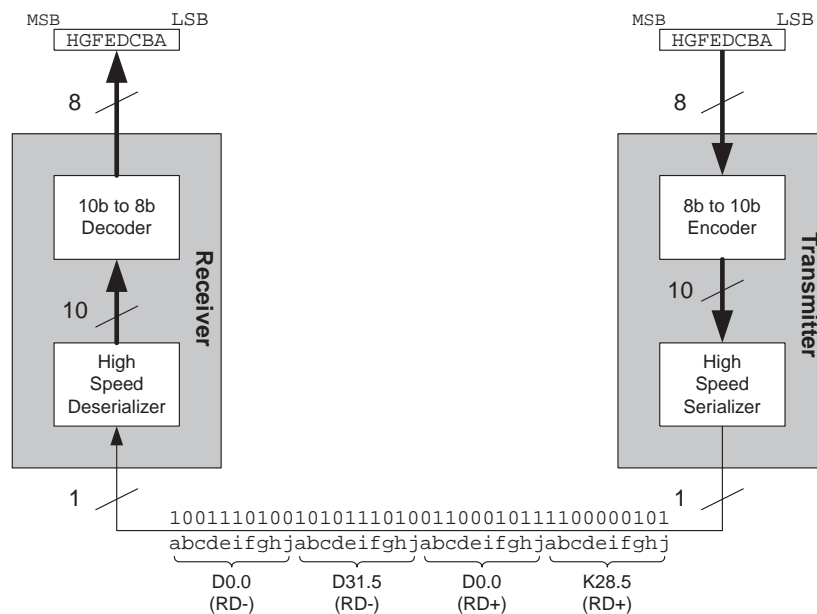
Functional Description

The 8b/10b coding scheme was initially proposed by Albert X. Widmer and Peter A. Franaszek of IBM Corporation in 1983. This coding scheme is used for high-speed serial data transmission. The encoder on the transmitter side maps the 8-bit parallel data input to 10-bit output. This 10-bit output is then loaded in and shifted out through a high-speed Serializer (Parallel-in Serial-out 10-bit Shift Register). The serial data stream will be transmitted through the transmission media to the receiver. The high-speed Deserializer (Serial-in Parallel-out 10-bit Shift Register) on the receiver side converts the received serial data stream from serial to parallel. The decoder will then re-map the 10-bit data back to the original 8-bit data. When the 8b/10b coding scheme is employed, the serial data stream is

DC-balanced and has a maximum run-length without transitions of 5. These characteristics aid in the recovery of the clock and data at the receiver. Figure 1 shows the 8b/10b encoder/decoder usage in a communication system.

DC Balance and Run Length

A DC-balanced serial data stream means that it has the same number of 0's and 1's for a given length of data stream. DC-balance is important for certain media as it avoids a charge being built up in the media.

The run-length is defined as the maximum numbers of contiguous 0's or 1's in the serial data stream. A small run-length data stream provides data transitions within a small length of data. Data transitions are essential for clock recovery. The PLL of the CDR generates a phase-adjustable output clock from the reference clock input. Transitions on the serial data stream provide the transmission clock phase information to the PLL and allow the PLL to recover the transmission clock with the correct phase. Note that the reference clock input is always necessary for the CDR. The serial data stream embeds the phase of the transmission clock, not the clock itself. This reference clock comes from the receiver system, not the transmitter system.

8b/10b Code Mapping

The 8b/10b encoder converts 8-bit code groups into 10-bit codes. The code groups include 256 data characters named Dx.y and 12 control characters named Kx.y.

Figure 2. The 8b/10b Coding Scheme
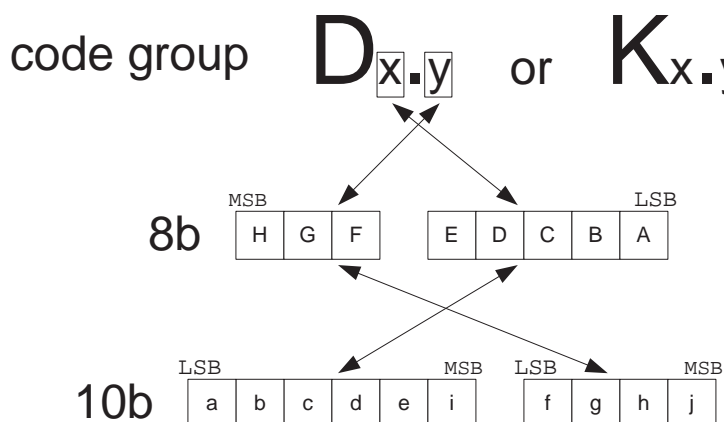
The coding scheme breaks the original 8-bit data into two blocks, 3 most significant bits (y) and 5 least significant bits (x). From the most significant bit to the least significant bit, they are named as H, G, F and E, D, C, B, A. The 3-bit block is encoded into 4 bits named j, h, g, f. The 5-bit block is encoded into 6 bits named i, e, d, c, b, a. As seen in Figure 2, the 4-bit and 6-bit blocks are then combined into a 10-bit encoded value.

Disparity

In order to create a DC-balanced data stream, the concept of disparity is employed to balance the number of 0's and 1's. The disparity of a block is calculated by the number of 1's minus the number of 0's. The value of a block that has a zero disparity is called disparity neutral.

If both the 4-bit and 6-bit blocks are disparity neutral, a combined 10-bit encoded data will be disparity neutral as well. This will create a perfect DC-balanced code. However, this is not possible. Because only 6 out of the 16 possible values of the 4-bit block are disparity neutral, they are not enough for encoding the 8 values of the 3-bit block. Likewise, only 20 values of the 6-bit block are disparity neutral and they are not enough for encoding the 32 values of the 5-bit block. Because both the 4-bit and 6-bit blocks have an even number of bits, the disparity is not possible to be +1 or -1. Therefore, the values with a disparity of +2 and -2 are also used in the 8b/10b coding scheme.

Table 1 and Table 2 are the values that are used for the 3-bit to 4-bit encoding and the 5-bit to 6-bit encoding respectively. Concatenating the 4-bit and 6-bit blocks together generates the 10-bit encoded value. Note that some

of the encoded values in Table 1 and Table 2 have two possible values, one with a disparity value of +2 and the other with a disparity value of -2. The 8b/10b coding scheme was designed to combine the values of the 4-bit and 6-bit blocks perfectly so that the worst case disparity value of the 10-bit code group will be at most +2 or -2. For example, the 4-bit encoded values with disparity value+2 will not be combined with the 6-bit encoded values with disparity value +2 because this will create a 10-bit value with disparity value +4.

Table 1. 3-Bit to 4-Bit Encoding Values

3b Decimal	3b Binary (HGF)	4b Binary (fghi)
0	000	0100 or 1011
1	001	1001
2	010	0101
3	011	0011 or 1100
4	100	0010 or 1101
5	101	1010
6	110	0110
7	111	0001 or 1110 or 1000 or 0111

Table 2. 5-Bit to 6-Bit Encoding Values

5b Decimal	5b Binary (EDCBA)	6b Binary (abcdei)
0	00000	100111 or 011000
1	00001	011101 or 100010
2	00010	101101 or 010010
3	00011	110001
4	00100	110101 or 001010
5	00101	101001
6	00110	011001
7	00111	111000 or 000111
8	01000	111001 or 000110
9	01001	100101
10	01010	010101
11	01011	110100
12	01100	001101
13	01101	101100
14	01110	011100
15	01111	010111 or 101000
16	10000	011011 or 100100
17	10001	100011
18	10010	010011
19	10011	110010
20	10100	001011
21	10101	101010
22	10110	011010
23	10111	111010 or 000101
24	11000	110011 or 001100
25	11001	100110
26	11010	010110
27	11011	110110 or 001001
28	11100	001110
29	11101	101110 or 010001
30	11110	011110 or 100001
31	11111	101011 or 010100

Running Disparity

Since the worst disparity of the 10-bit encoded data value is either +2 or -2, it is still possible that more 10-bit encoded data values with +2 (or -2) disparity are transmitted through the serial data stream. In this case, the data stream will no longer be DC-balanced. In order to maintain a DC-balance data stream, each code group will be converted to one of the two possible values as seen in the RD- and RD+ columns of the Table 3. The RD- disparity will be either +2 or 0 (disparity neutral) and the RD+ disparity will be either -2 or 0. The encoder will pick one of the two values based on the calculation of current Running Disparity.

Table 3. Portion of the 8b/10b Encoding/Decoding Mapping Table

Code Group	kin/ kout	8-bit data HGF EDCBA	10-bit data (RD-) abcdei fghj	10-bit data (RD+) abcdei fghj	Code Group	kin/ kout	8-bit data HGF EDCBA	10-bit data (RD-) abcdei fghj	10-bit data (RD+) abcdei fghj
D0.0	0	000 00000	100111 0100	011000 1011	D0.1	0	001 00000	100111 1001	011000 1001
D1.0	0	000 00001	011101 0100	100010 1011	D1.1	0	001 00001	011101 1001	100010 1001
D2.0	0	000 00010	101101 0100	010010 1011	D2.1	0	001 00010	101101 1001	010010 1001
D3.0	0	000 00011	110001 1011	110001 0100	D3.1	0	001 00011	110001 1001	110001 1001
:					:				
D31.0	0	000 11111	101011 0100	010100 1011	D31.1	0	001 11111	101011 1001	010100 1001
D0.2	0	010 00000	100111 0101	011000 0101	D0.3	0	011 00000	100111 0011	011000 1100
D1.2	0	010 00001	011101 0101	100010 0101	D1.3	0	011 00001	011101 0011	100010 1100
D2.2	0	010 00010	101101 0101	010010 0101	D2.3	0	011 00010	101101 0011	010010 1100
D3.2	0	010 00011	110001 0101	110001 0101	D3.3	0	011 00011	110001 1100	110001 0011
:					:				
D31.2	0	010 11111	101011 0101	010100 0101	D31.3	0	011 11111	101011 0011	010100 1100
D0.4	0	100 00000	100111 0010	011000 1101	D0.5	0	101 00000	100111 1010	011000 1010
D1.4	0	100 00001	011101 0010	100010 1101	D1.5	0	101 00001	011101 1010	100010 1010
D2.4	0	100 00010	101101 0010	010010 1101	D2.5	0	101 00010	101101 1010	010010 1010
D3.4	0	100 00011	110001 1101	110001 0010	D3.5	0	101 00011	110001 1010	110001 1010
:					:				
D31.4	0	100 11111	101011 0010	010100 1101	D31.5	0	101 11111	101011 1010	010100 1010
D0.6	0	110 00000	100111 0110	011000 0110	D0.7	0	111 00000	100111 0001	011000 1110
D1.6	0	110 00001	011101 0110	100010 0110	D1.7	0	111 00001	011101 0001	100010 1110
D2.6	0	110 00010	101101 0110	010010 0110	D2.7	0	111 00010	101101 0001	010010 1110
D3.6	0	110 00011	110001 0110	110001 0110	D3.7	0	111 00011	110001 1110	110001 0001
:					:				
D31.6	0	110 11111	101011 0110	010100 0110	D31.7	0	111 11111	101011 0001	010100 1110
K28.0	1	000 11100	001111 0100	110000 1011					
K28.1	1	001 11100	001111 1001	110000 0110					
K28.2	1	010 11100	001111 0101	110000 1010					
K28.3	1	011 11100	001111 0011	110000 1100					
K28.4	1	100 11100	001111 0010	110000 1101					
K28.5	1	101 11100	001111 1010	110000 0101					
K28.6	1	110 11100	001111 0110	110000 1001					
K28.7	1	111 11100	001111 1000	110000 0111					
K23.7	1	111 10111	111010 1000	000101 0111					
K27.7	1	111 11011	110110 1000	001001 0111					
K29.7	1	111 11101	101110 1000	010001 0111					
K30.7	1	111 11110	011110 1000	100001 0111					

The transmitter assumes a negative Running Disparity (RD-) at start up. When an 8-bit data is encoding, the encoder will use the RD- column for encoding. If the 10-bit data been encoded is disparity neutral, the Running Disparity will not be changed and the RD- column will still be used. Otherwise, the Running Disparity will be changed and the RD+ column will be used instead. Similarly, if the current Running Disparity is positive (RD+) and a disparity neutral 10-bit data is encoded, the Running Disparity will still be RD+. Otherwise, it will be changed from RD+

back to RD- and the RD- column will be used again. The state diagram in Figure 3 describes how the current Running Disparity is calculated.

Figure 3. Running Disparity State Machine
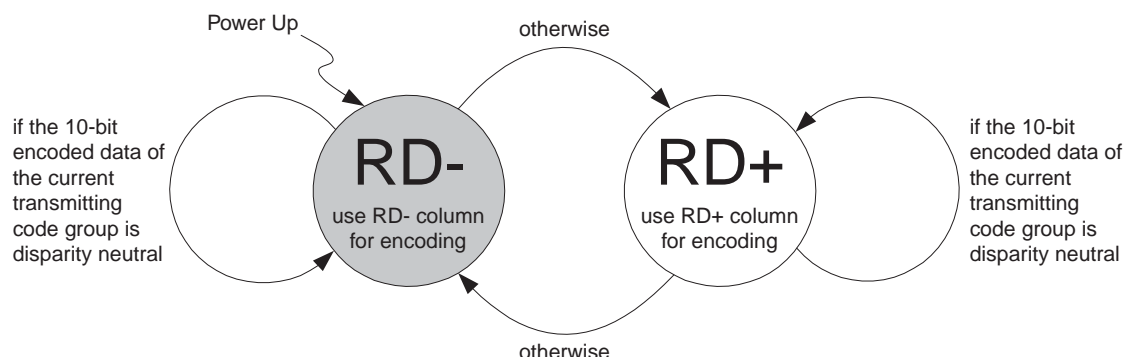
Encoder and Decoder Pin Descriptions

The 8b/10b encoder and 10b/8b decoder are implemented in HDL language. These designs can be combined to create a full duplex channel. The pin descriptions of the designs are shown in the following tables.

Table 4. Encoder Pin Descriptions

| Name	Type	Description
clk	I	**Encoder Clock.** This pin is the main clock of the encoder. All registered inputs and outputs of the encoder are based on the rising of this clock.
reset_n	I	**Master Reset.** This low active asynchronous reset will reset all internal registers of the encoder to their initial states.
datain_8b_[7:0]	I	**8-bit Data Input**. This is the 8-bit raw data to be encoded.
kin	I	**Character Type Control.** This high active signal indicates that the 8-bit data on datain_8b bus is going to be encoded to a control character instead of a data character.
rdispin	I	**Running Disparity Input.** This pin provides to the encoder the running disparity before the encoding of current 8-bit data on datain_8b bus.
dataout_10b_[9:0]	O	**10-bit Data Output.** This is the 10-bit encoder output and should be connected to the TXD port of the sysHSI macro TX_8B10B.
rdispout	O	**Running Disparity Output.** This is the running disparity of the present dataout_10b bus.
k_err	O	**Invalid Control Character Requested.** This high active signal indicates that a invalid control character is requested.

Table 5. Decoder Pin Descriptions

Name	Type	Description
clk	I	**Decoder Clock.** This pin is the main clock of the decoder. All registered inputs and outputs of the decoder are based on the rising of this clock.
reset_n	I	**Master Reset.** This low active asynchronous reset will reset all internal registers of the decoder to their initial states.
datain_10b_[9:0]	I	**10-bit Data Input.** This is the 10-bit raw data to be decoded and should come from the RXD port of the sysHSI macro CDRX_8B10B.
rdispin	I	**Running Disparity Input.** This pin provides to the decoder the running disparity before the decoding of current 10-bit data on datain_10b bus.
dataout_8b_[7:0]	O	**8-bit Data Output.** This is the 8-bit decoder output.
kout	O	**Character Type Output.** This high active signal indicates that the 8-bit data on dataout_8b bus is a control character instead of a data character.
rdispout	O	**Running Disparity Output.** This is the running disparity of the present dataout_8b bus.
disp_err	O	**Disparity Error.** This high active signal indicates that a running disparity error has occurred.
code_err	O	**Invalid Code Group Error.** This high active signal indicates that an invalid 10-bit code group has been received.

Block Diagrams and Implementation Guidelines

The block diagrams of the encoder and decoder are shown in Figure 4 and Figure 5. Notice that the encoder contains a clock latency of 2 and the decoder contains a clock latency of 3. The clock latency allows the throughput to be increased. All signals in the designs are based on the rising edge of the main clock. Once the data inputs are provided and the setup/hold times are satisfied, the data inputs will be sampled at the rising edge of the clock. As seen in Figure 6, the data outputs will be available after another one or two rising edges and a clock-to-output delay.

For both the encoder and decoder designs, the rdispout output should be connected back to the rdispin input. Except clk, reset_n, and rdispin, the other inputs are pipelined to generate the outputs. However, since the running disparity of the data currently being transmitted is required for the encoding of the following data, the rdispin-to-rdispout loopback path can only contain one register level and cannot be pipelined.

Figure 4. Encoder Block Diagram
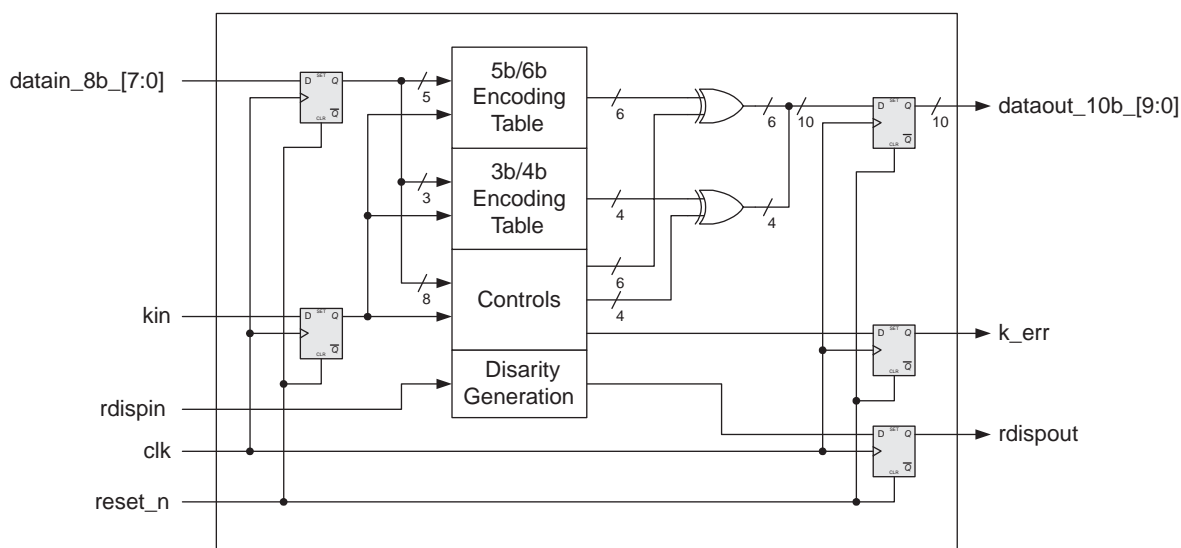
Figure 5. Decoder Block Diagram
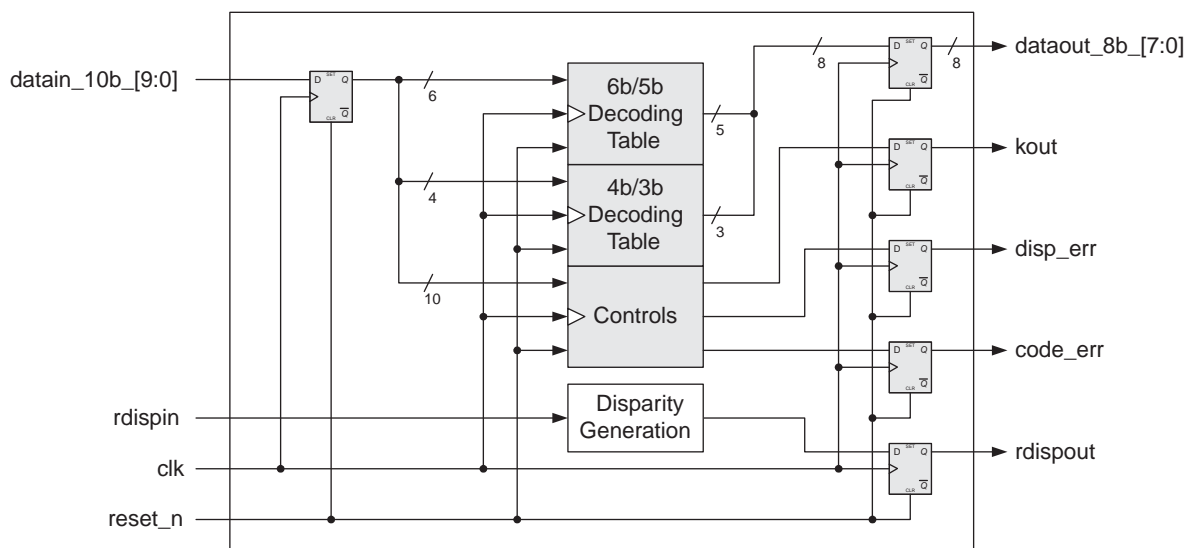
Figure 6. Timing Diagrams

Encoder Timing   Decoder Timing
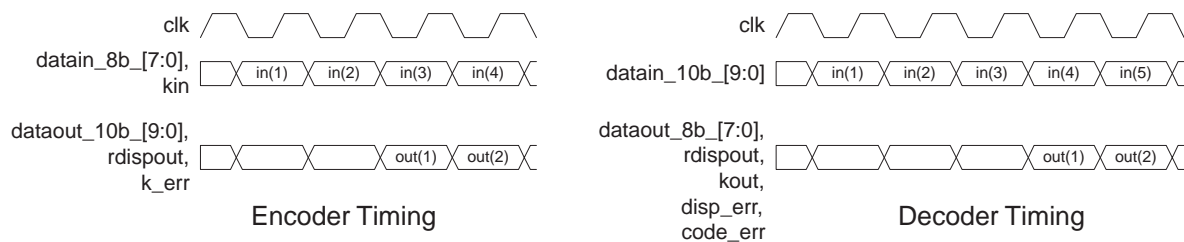
Encoder and Decoder Behavior

This 8b/10b encoder/decoder is fully tested to cover all possible cases. The encoder will generate the correctly encoded 10-bit codes based on the kin, datain_8b_[7:0], and rdispin inputs. The total number of combination cases is 2 x 256 x 2 = 1024. In these 1024 cases, only (256 + 12) x 2 = 536 cases are valid and the k_err will be generated for the rest of the 488 cases. The decoder will convert 10-bit codes into 8-bit outputs. The code_err will be generated for all invalid 10-bit codes the decoder received. However, in some cases even though the received 10-bit input data is one of the 536 valid codes, the disp_err will be generated if the running disparity rule is violated. The behavior of all possible input combinations of the 8b/10b encoder/decoder is shown in Table 6 and Table 7.

Table 6. 8b/10b Encoder Behavior

Inputs			Outputs		
kin	datain_8b_[7:0]	rdispin	dataout_10b_[9:0]	rdispout	k_err
0	Any of the 256 Dx.y 8-bit Data Characters	0	10-bit dataout from the (RD-) column of the corresponding Dx.y row in Table 3	0, If dataout has 5 one's 1, If dataout has 6 one's	0
		1	10-bit dataout from the (RD+) column of the corresponding Dx.y row in Table 3	0, If dataout has 4 one's 1, If dataout has 5 one's	
1	Any of the 12 Kx.y 8-bit Control Characters	0	10-bit dataout from the (RD-) column of the corresponding Kx.y row in Table 3	0, If dataout has 5 one's 1, If dataout has 6 one's	
		1	10-bit dataout from the (RD+) column of the corresponding Kx.y row in Table 3	0, If dataout has 4 one's 1, If dataout has 5 one's	
	Other undefined 244 Kx.y 8-bit Control Characters	0	10-bit dataout from the (RD-) column of the corresponding Dx.y row in Table 3	0, If dataout has 5 one's 1, If dataout has 6 one's	1
		1	10-bit dataout from the (RD+) column of the corresponding Dx.y row in Table 3	0, If dataout has 4 one's 1, If dataout has 5 one's	

Table 7. 10b/8b Decoder Behavior

Inputs		Outputs				
datain_10b_[9:0]	rdispin	code_err	disp_err	kout	dataout_8b_[7:0]	rdispout
Any 10-bit data from either the (RD-) or (RD+) column of the Dx.y rows in Table 3	0	0	0, if data is from (RD-) column 1, if data is from (RD+) column	0	Corresponding 8-bit Data	0, If dataout has 5 one's 1, If dataout has 6 one's
	1		0, if data is from (RD+) column 1, if data is from (RD-) column			0, If dataout has 4 one's 1, If dataout has 5 one's
Any 10-bit data from either the (RD-) or (RD+) column of the Kx.y rows in Table 3	0		0, if data is from (RD-) column 1, if data is from (RD+) column	1		0, If dataout has 5 one's 1, If dataout has 6 one's
	1		0, if data is from (RD+) column 1, if data is from (RD-) column			0, If dataout has 4 one's 1, If dataout has 5 one's
Others	0	1	X	X	X	0, If rdispin is 0 1, If rdispin is 1
	1					

Note: X means that the outputs may be 0 or 1 and should be ignored because of the code_err.

Implementation

The design software used for this implementation is Lattice ispLEVER™ 2.0 Service Pack 5. Table 8 lists design implementation information using Verilog and the LeonardoSpectrum synthesizer.

Table 8.

Family	Device	Macrocells/PFU Used total (Encoder Only/Decoder Only)	Max. Clock Freq.
ispMACH 4000	LC4128V-5T100C	109 (58 / 77)	85.5 MHz
ispXPLD 5000MX	LC5512MV-5Q208C	132 (49 / 85)	95.7 MHz
ispXPGA	LFX1200B-03FE680C	92 (40 / 49)	91.4 MHz
ORCA Series 4	OR4E02-3BA352	30 (11 / 19)	88.6 MHz

Figure 7 shows the timing simulation of the 8b/10b encoder/decoder implemented in a LC4128V-5T100C device. After reset, D3.7 (RD-), D0.6 (RD+), D0.0 (RD-), D0.0 (RD-), ..., are transmitted.

Figure 7. Timing Simulation Diagrams
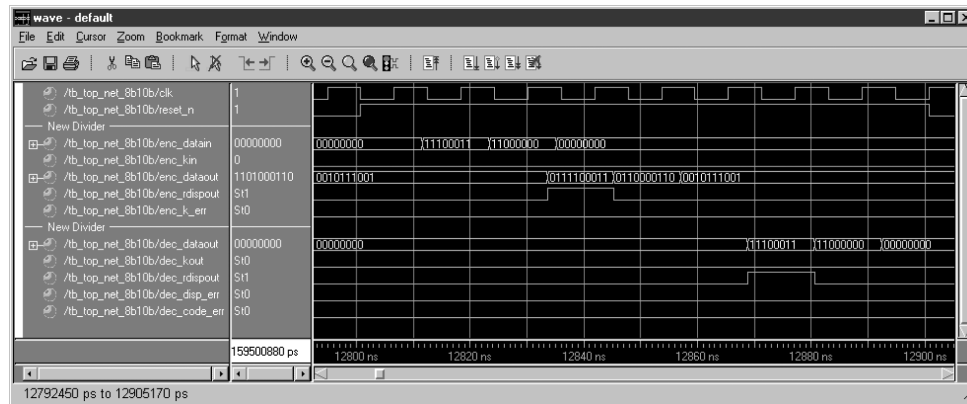
Technical Support Assistance

Hotline: 1-800-LATTICE (Domestic)

1-408-826-6002 (International)

e-mail: techsupport@latticesemi.com