802.16 LDPC Encoder v1.0

Introduction

The Xilinx 802.16 LDPC Encoder core provides designers with a Low Density Parity Check (LDPC) Code Encoding core, conforming to the IEEE P802.16e(draft) standard[1], for fixed and mobile wireless access systems.

Features

- Available for Virtex™-II, Virtex-II Pro, Virtex-4, Virtex-5, Spartan™-3, and Spartan-3E FPGAs
- Low Density Parity Check code for IEEE P802.16e supporting:
  - all block sizes
  - all code rates
- Very Low Latency, 7 clock cycles
- Fully optimized for speed and area
- Any required code rate subset selected at generation, enabling area minimization/speed maximization
- Full suite of handshaking signals for end-to-end
- Flow control
- Fully synchronous design using a single clock
- For use with Xilinx CORE Generator™ v8.2i and higher

Functional Overview

The LDPC Encoder core (see Figure 1) provides a complete encoding solution for the LDPC encoder as defined in section 8.4.9.2.5 of IEEE P802.16e(draft)[1]. A major feature of the core is that it has an extremely low latency. The encoded packet is available, at the output, seven cycles after the first bit of the unencoded packet is presented at the input. The output data is subsequently available as a continuous block.

The IEEE P802.16e LDPC encoder is based on a set of one or more fundamental LDPC codes, each of which is a systematic linear block code. The code Rate and Block Size can be adjusted dynamically (see "Run-Time Parameters" on page 8) to all code rates and packet sizes defined in the specification.

See Table 1 for the rates and sizes available in the IEEE P802.16e specification.

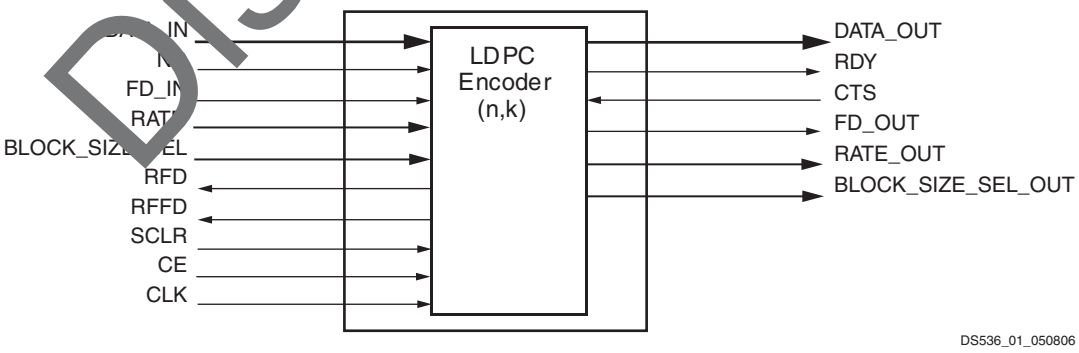

Figure 1: **IEEE P802.16e LDPC Core Block Diagram**

Table 1: **LDPC Block Sizes and Code Rates**

Output Packet Size, n(bits)	Output Packet Size, n (bytes)	Block Size (bits)	Input Packet Size, k (bytes)			
			Rate= 1/2	Rate= 2/3(A/B)	Rate= 3/4(A/B)	Rate = 5/6
576	72	24	36	48	54	60
672	84	28	42	56	63	70
768	96	32	48	64	72	80
864	108	36	54	72	81	90
960	120	40	60	80	90	100
1056	132	44	66	88	99	110
1152	144	48	72	96	108	120
1248	156	52	78	104	117	130
1344	168	56	84	112	126	140
1440	180	60	90	120	135	150
1536	192	64	96	128	144	160
1632	204	68	102	136	153	170
1728	216	72	108	144	162	180
1824	228	76	114	152	171	190
1920	240	80	120	160	180	200
2016	252	84	126	168	189	210
2112	264	88	132	176	198	220
2208	276	92	138	184	207	230
2304	288	96	144	192	216	240

There are six different LDPC code rates (1/2, 2/3A, 2/3B, 3/4A, 3/4B, and 5/6) defined in IEEE P802.16e(draft)[1] and shown in Table 1. The standard also defines the Block Size for each code. The Block Size controls the length of the output packet, and the code rate determines the ratio of the input packet length to the output packet length. Each encoding solution is defined by selecting one of the Code Rates and an associated Block Size.

Each of the six LDPC codes in the standard is defined by a matrix **H** of size *m*-by-*n*, where *n* is the length of the output packet and *m* is the number of parity check bits in the code. The number of input packet bits (systematic bits) required to generate the code is *k=n-m*.

The matrix **H** is expanded from a generator base matrix $\mathbf{H}_{bm}$ of size m_b-by-24, where $m=m_b.z$ and $n=24.z$, where *z* is the same as the Block Size. The generator base matrix is expanded by replacing each of its entries with a *z-by-z* matrix. Non-negative integers are replaced by a *z-by-z* permutation matrix, and each negative integer with a *z-by-z* zero matrix.

The matrix $\mathbf{H}_{bm}$ is defined as:

where **P**(*i*,*j*) is one of a set of *z*-by-*z* permutation matrices or a *z*-by-*z* zero matrix.

$$H_{bm} = \begin{bmatrix} P_{(0,0)} & P_{(0,1)} & P_{(0,2)} & \cdots & P_{(0,22)} & P_{(0,23)} \\ P_{(1,0)} & P_{(1,1)} & P_{(1,2)} & \cdots & P_{(1,22)} & P_{(1,23)} \\ P_{(2,0)} & P_{(2,1)} & P_{(2,2)} & \cdots & P_{(2,22)} & P_{(2,23)} \\ \cdots & \cdots & \cdots & \cdots & \cdots & \cdots \\ P_{(m_b-1,0)} & P_{(m_b-1,1)} & P_{(m_b-1,2)} & \cdots & P_{(m_b-1,22)} & P_{(m_b-1,23)} \end{bmatrix}$$

The permutations used are circular right shifts, and the set of permutation matrices contains the *z-by-z* identity matrix and circular right shifted versions of the identity matrix. Each permutation matrix is specified by a single circular right shift, the size of this shift is described by the non-negative integer entry at (i,j), $p(i,j) \geq 0$. The base matrix $\mathbf{H}_{bm}$ can then be directly expanded to $\mathbf{H}$ by replacing each entry with the correct *z*-by-*z* matrix. The base matrices defined in IEEE P802.16e(draft)[1] for each code rate of the LDPC encoder are included in "Appendix 1: Coding Matrices" on page 14.

The generator base matrix is defined for the largest Block Size of 96, which gives an output packet size of *n*=2304 for each code rate. The set of shifts, *p(i,j)*, in the generator base matrix are used to determine the shift sizes for all other packet sizes of the same code rate. Adjusting the value of Block Size changes the dimension of the *z-by-z* matrix used to expand $\mathbf{H}_{bm}$, which means that the size of the right shift applied to it must also be scaled with Block Size. The equations used to perform this scaling are provided by the IEEE P802.16e(draft) standard[1], and outlined below.

Let the scaled shift factor used at entry (i,j) be *p(f,i,j)*, where f is the index of the Block sizes, f = 0, 1, 2, ..., 18. In the equations below, the selected Block size is denoted z_f, and the maximum Block Size of 96 is designated z_0.

For code rates 1/2, 3/4A, 3/4B, 2/3B, and 5/6, the shift sizes, *p(f,i,j)*, for a packet size corresponding to Block Size z_f are derived by scaling *p(i,j)* proportionally:

$$P(f,i,j) = \begin{cases} p(i,j), p(i,j) \leq 0 \\ \left\lfloor \frac{p(i,j) z_f}{z_0} \right\rfloor, p(i,j) > 0 \end{cases}$$

where $\lfloor x \rfloor$ denotes the flooring function that gives the nearest integer towards $-\infty$.

For code rate 2/3A, the shift sizes, *p(f,i,j)*, for a code size corresponding to Block Size z_f are derived from *p(i,j)* by using a modulo function.

$$P(f,i,j) = \begin{cases} p(i,j), p(i,j) \leq 0 \\ \mathrm{mod}(p(i,j), z_f), p(i,j) > 0 \end{cases}$$

Thus, by selecting the code rate and the Block Size, the base generator matrices are scaled to meet the coding requirements. For a given code, the number of output bits in a packet, n, is $24 \times \text{Block Size}$. To generate an output of n bits, the input packet must be k bits long, where k is $n \times \text{Rate}$. For more information see [1].

Pinout

The ports employed by the core are shown on the block diagram in Figure 1 and are summarized in Table 2. All control signals are active high.

Table 2: **Core Signal Pinout**

Port Name	Port Width	Direction	Active State	Description
DATA_IN	1	INPUT		**Data Input**: Packet to be encoded, presented in slice of 1 bits.
RATE	3	INPUT		**Rate**: Rate to be used to encode the packet
BLOCK_SIZE_SEL	5	INPUT		**Block Size Select**: Block size to be used to encode the packet
ND	1	INPUT	High	**New Data**: Set High when data is valid on DATA_IN.
RFD	1	OUTPUT	High	**Ready For Data**: Set High by the core when it is ready for a slice of data on DATA_IN.
FD_IN	1	INPUT	High	**First Data**: Set High to indicate the start of a packet of input data.
RFFD	1	OUTPUT	High	**Ready For First Data**: Set High by the core when ready to accept FD_IN.
CE	1	INPUT	High	**Clock Enable** (optional).
SCLR	1	INPUT	High	**Synchronous Reset** (optional).
CLK	1	INPUT	Rising edge	Clock
DATA_OUT	1	OUTPUT		**Data Output:** Encoded packet output, provided in slice of 1 bit. Output width matches input width.
RATE_OUT	3	OUTPUT		**Rate Output**: Rate used to encode current output packet.
BLOCK_SIZE_SEL_OUT	5	OUTPUT		**Block Size Select Output**: Block size used to encode the current output packet
RDY	1	OUTPUT	High	**Ready**: Set High by core when output is valid on DATA_OUT.
CTS	1	INPUT	High	**Clear To Send**: Set High to enable output of data over DATA_OUT. Set Low to stall data output.
FD_OUT	1	OUTPUT	High	**First Data Output**: Set High by the core to indicate that the current data output is the first item of a packet.

CLK

All signals are synchronous to the CLK input.

CE

When CE is deasserted, all inputs are ignored and the core remains in its current state.

SCLR

When SCLR is asserted, the core is synchronously set to its initial state if CE is High. Any intermediate results are discarded. It takes one cycle to achieve this initial state, and the state is maintained until SCLR is deasserted. The initial state is where the core is awaiting FD_IN (i.e. RFFD and RFD are asserted).

FD_IN

FD_IN should be asserted to start the encoding of a packet of dat Note that th ND and FD_IN must be asserted while the core is asserting RFFD for the core to start. R is always asserted with RFFD.

RFFD

The RFFD output is asserted by the core when it is ready f a ew packet of data, as shown in Figure 1.

Note: RFD is asserted with RFFD.

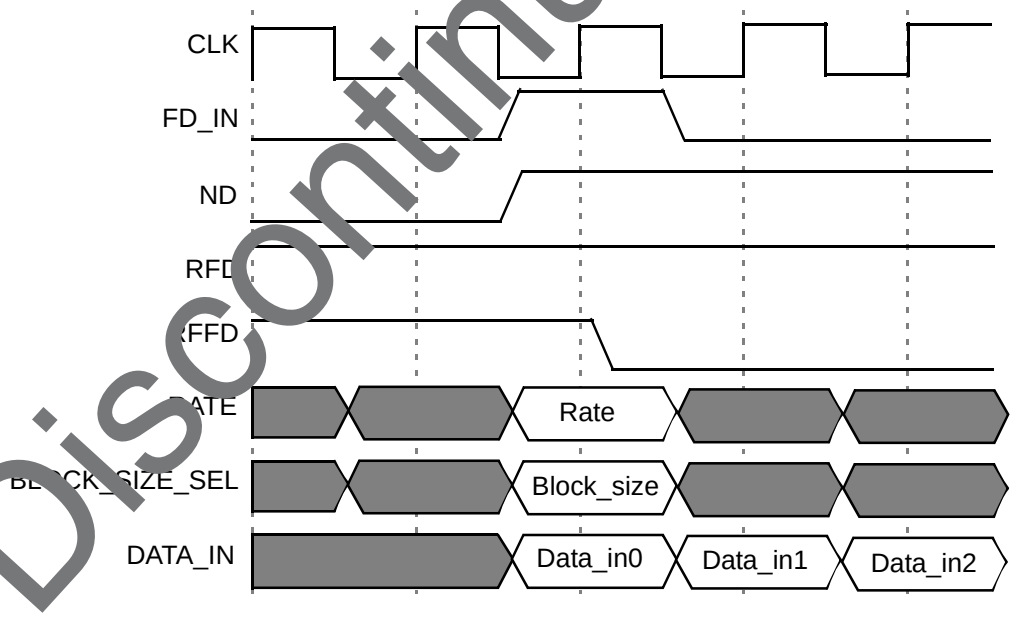

Figure 2: **FD_IN Input With RFFD (ND Must Be High for FD_IN to be Valid)**

ND

ND should be asserted when data is valid on DATA_IN. Deasserting ND stalls data input. The first bit of data is only input when ND, RFD, FD_IN and RFFD are all asserted. Subsequent data in the packet is only input when ND and RFD are asserted. See Figure 3 and Figure 4.

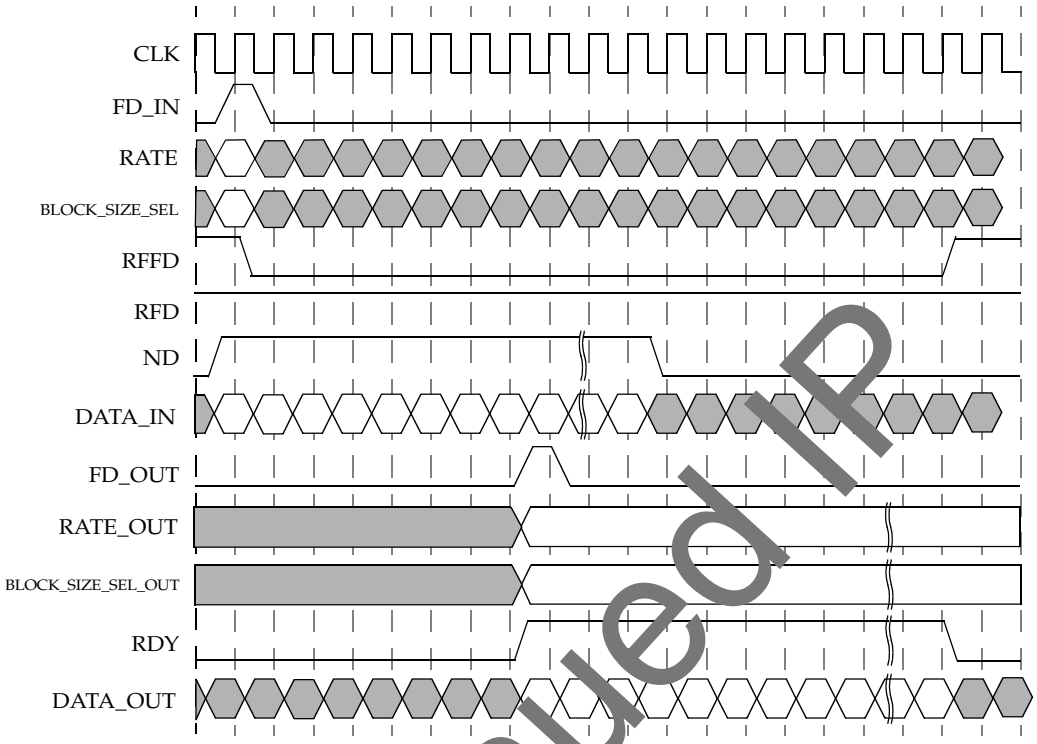

Figure 3: **Control Signals and Data Input and Output**

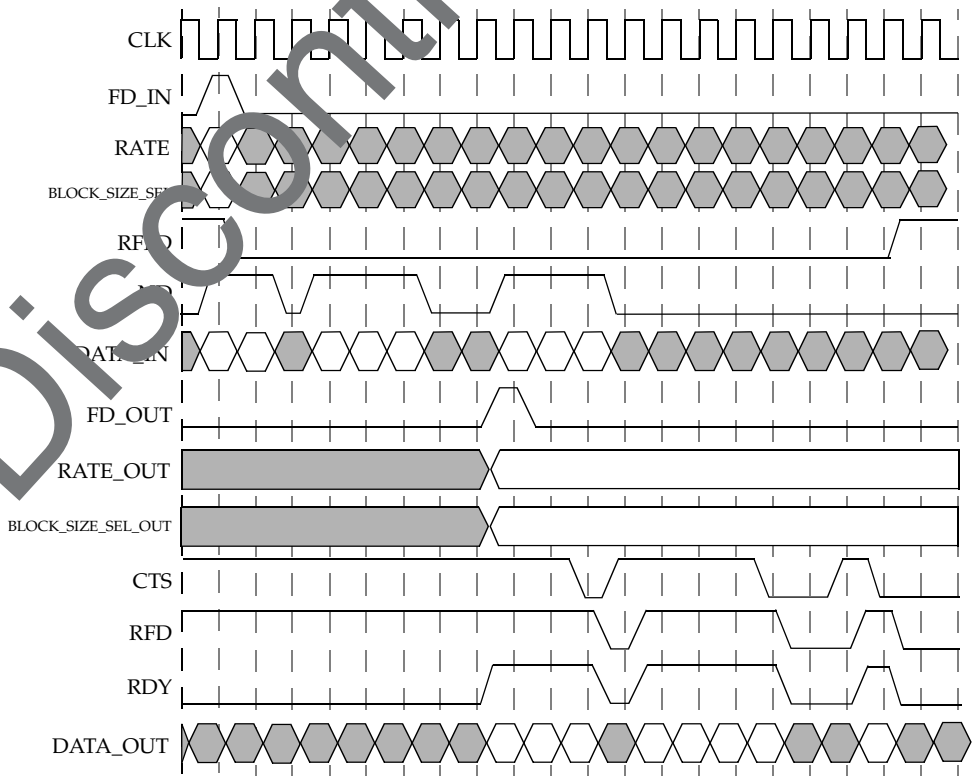

Figure 4: **End-To-End Flow Control with the ND, CTD, and RFD Signals**

RFD

RFD is asserted by the core to indicate that it is ready to accept a sample of data on DATA_IN. The first bit of data is only input when ND, RFD, FD_IN and RFFD are all asserted. Subsequent data in the packet is only input when ND and RFD are asserted. RFD is deasserted while the core is generating the code bits of the output packet. Also, deasserting CTS, during the systematic bits of the packet, causes RFD to be deasserted one cycle later. See Figure 3.

DATA_IN

The uncoded data input to the core is supplied via DATA_IN.

RATE

The run-time parameter used to select the coding matrix and code rate of the output signal. See "Functional Overview" on page 1. This port is read by the core on the start of a packet when the FD_IN, ND, and RFFD signals are asserted. The allowed values for this signal and their definitions can be found in Table 3.

BLOCK_SIZE_SEL

The run-time parameter used to select the Block Size used in the encoding algorithm; and the input and output packet length, see "Functional Overview" on page 1. This port is read by the core on the start of a packet when the FD_IN, ND, and RFFD signals are asserted. The allowed values for this signal and their definitions can be found in Table 4.

DATA_OUT

The encoded packet is output on DATA_OUT.

RATE_OUT

The RATE input is propagated to the output via RATE_OUT, using the same encoding as RATE, see Table 3. RATE_OUT is set when FD_OUT and RDY are asserted, and then stays valid for the duration of the packet. See Figure 3 and Figure 4.

BLOCK_SIZE_SEL_OUT

The BLOCK_SIZE_SEL input is propagated to the output via BLOCK_SIZE_SEL_OUT, using the same encoding as BLOCK_SIZE_SEL. See Table 4. BLOCK_SIZE_SEL_OUT is set when FD_OUT and RDY are asserted, and then stays valid for the duration of the packet. See Figure 3 and Figure 4.

RDY

RDY is asserted by the core to indicate that data on DATA_OUT is valid.

CTS

The output from the core is stalled when CTS is deasserted. Deasserting CTS causes RDY to deassert in the next clock cycle. Also, deasserting CTS causes RFD to deassert in the following cycle. See Figure 4.

FD_OUT

FD_OUT is asserted when the first item of data in a packet is presented on DATA_OUT. Note that RDY is always asserted with FD_OUT. FD_OUT is therefore only asserted, when CTS is asserted in the previous cycle. See Figure 3 and Figure 4.

Run-Time Parameters

All the values required by the core are supplied through the input ports, RATE and BLOCK_SIZE_SEL. These inputs consist of 3-bit and 5-bit fields, respectively.

Code Rate

The core supports the code rates defined in the IEEE P802.16e(draft) standard[1]. The code rate applied to a packet is set using the RATE input. The RATE input is encoded according to Table 3. The code rates supported by the core are specified when the core is generated, which means the available code rates can be a subset of the six valid values listed. See "CORE Generator Parameters" on page 9. If a value is presented to the RATE input that has not been specified at generation, or is a reserved value, the rate used by the core defaults to 000, Code Rate 5/6.

Table 3: **Valid Values for the RATE input**

Rate Values (RATE[2:0])	Code Rate
000	5/6
001	3/4B
010	3/4A
011	2/3B
100	2/3A
101	1/2
110,111	RESERVED

Block Size

The block size used in the encoding algorithm is defined using the BLOCK_SIZE_SEL input. The BLOCK_SIZE_SEL input is encoded according to Table 4. The maximum allowable block size is specified when the core is generated. See "CORE Generator Parameters" on page 9.

If the BLOCK_SIZE_SEL input has a value that specifies a block size greater than the maximum specified at generation, the block size defaults to the maximum. If a value is selected that is less than the minimum (i.e., in the reserved section between 0-5), the block size defaults to 24.

Table 4: **Valid Values for the BLOCK_SIZE_SEL Input**

Block Size Select Values (BLOCK_SIZE_SEL[4...0])	Block Size (bits)
00000 - 00101	RESERVED
00110	24
00111	28
01000	32
01001	36
01010	40
01011	44

Table 4: **Valid Values for the BLOCK_SIZE_SEL Input *(Continued)***

Block Size Select Values (BLOCK_SIZE_SEL[4...0])	Block Size (bits)
01100	48
01101	52
01110	56
01111	60
10000	64
10001	68
10010	72
10011	76
10100	80
10101	84
10110	88
10111	92
11000	96
11001 - 11111	RESERVED

CORE Generator Parameters

The CORE Generator provides a Graphical User Interface (GUI), shown in Figure 5, that allows only required rates to be selected when building the core. By reducing the number of Code Rates used within the core, the slice count is reduced. The options available in the GUI are:

- **Rate 1/2:** Allows the generated core to perform 1/2 rate encoding.
- **Rate 2/3A:** Allows the generated core to perform 2/3A rate encoding.
- **Rate 2/3B:** Allows the generated core to perform 2/3B rate encoding.
- **Rate 3/4A:** Allows the generated core to perform 3/4A rate encoding.
- **Rate 3/4B:** Allows the generated core to perform 3/4B rate encoding.
- **Rate 5/6:** Allows the generated core to perform 5/6 rate encoding.

Any combination of these options is permitted, although at least one Code Rate must be selected.

The GUI also provides the option of specifying the biggest block size used within the core. By reducing the maximum block size used by the core, the slice count is decreased and the maximum clock rate increased.

- **Maximum Block Size:** Specifies the largest Block Size that can be used in the encoder and, therefore, the maximum input/output packet size. All block sizes from 24 (the minimum specified in IEEE P802.16e) to Maximum Block Size is available (selected by the BLOCK_SIZE_SEL input). Maximum Block Size cannot exceed 96 (the maximum specified in IEEE P802.16e).

Additional options available in the GUI are:

- **Optional Pins:** Synchronous clear and CE.

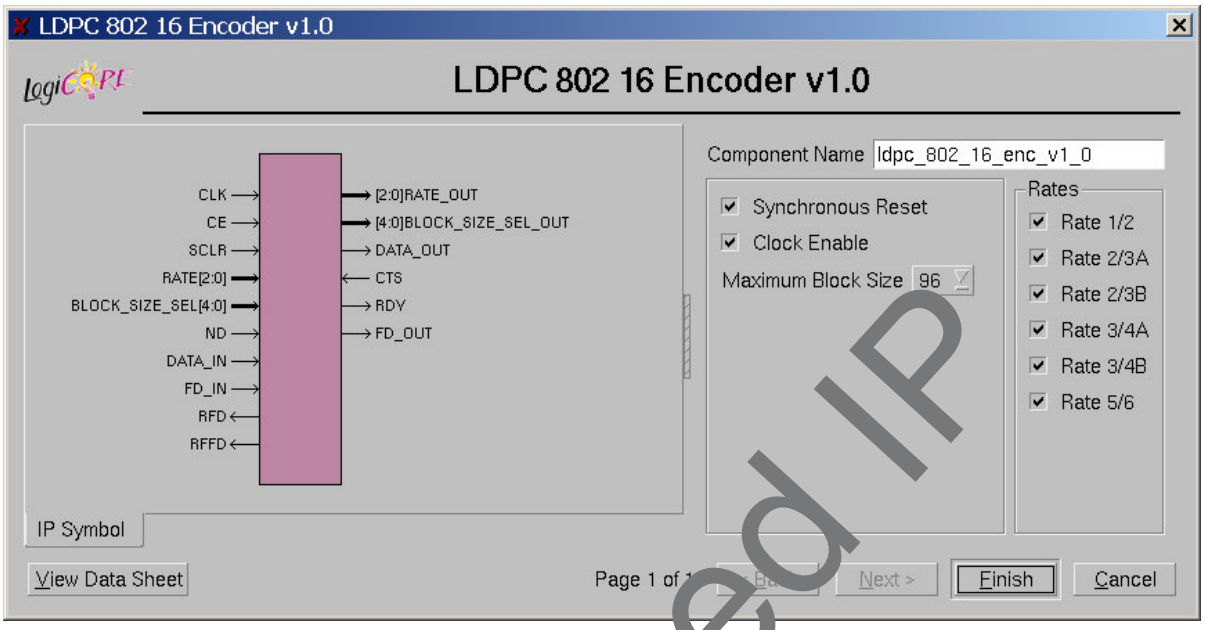

Figure 5: **LDPC Encoder GUI Screen Shot**

Latency

The latency of the core is defined as the number of cycles between input of the first bit of an uncoded input packet to the first bit of an encoded output packet. If ND and CTS are always asserted, then it takes 7 clock cycles for the data to propagate through the core. This gives a total latency for the core of 7 cycles. If ND is deasserted during the input of the uncoded packet bits, the core stalls while it waits for more data. This adds to the time it takes to encode a packet. If CTS is deasserted, the data coming out of the core is stalled, and additional latency (equal to the number of cycles CTS is deasserted) is added.

System Integration

Data input

Data flow into the core is controlled at two levels. The core input FD_IN and output RFFD provide a handshake to control the start of each packet, and during a packet, the input ND and output RFD provide a handshake to control the flow of data into and out of the core on a cycle-by-cycle basis.

Note: Data must be available when the core is started, so ND must be asserted with FD_IN (and RFFD) for the input of a packet to start.

Also note, if the core is ready for a new packet (i.e., RFFD is asserted), then asserting ND without FD_IN does not result in data being consumed by the core. The core must be started by asserting FD_IN with ND for this to happen. In summary, data is only consumed by the core after the core has started (i.e., RFFD is deasserted) and both ND and RFD are asserted, or when the core is started (i.e., RFFD is asserted by the core and FD_IN is asserted with ND).

If packet-level flow control is not required, then FD_IN can be tied to RFFD. The core automatically starts processing data once it becomes available (i.e., when ND is asserted, RFD will have been asserted with RFFD).

Data Output

The first bit of the encoded packet becomes available at the output of the core 7 cycles after the first bit of data in a packet is input (see "Latency" on page 10). Packets of data can be streamed out continuously. If the data rate is too high, then it can be modified by periodically deasserting CTS to stall output. Because there is no internal buffering, deasserting CTS also deasserts RFD to stop the flow of data into the core to prevent data loss.

End-to-End Flow Control

The inclusion of data flow handshake signals on both the core's input and output provides the means to implement end-to-end flow control, whereby the data flow can be either regulated by the source of uncoded input data or the destination of the encoded output data.

Throughput

The throughput of the core, which can be measured in terms of bits per second or packets per second, depends upon three limits:

- Time to input a packet
- Time to process a packet
- Minimum time to output a packet, as dictated by the core for a particular clock-rate.

The nature of the coding algorithm means that the output packet is always longer than the input packet. If the input packet is received as a continuous block of data, it does not restrict the throughput, since this is set by the time to output a packet. However, if ND is deasserted while the packet is being input, the encoder stalls and the throughput is reduced. When the core is generating the code bits, the core cannot accept any input and it deasserts RFD. Therefore, the rate at which packets can be input is the same as the output rate.

The encoding process consists of two stages: first, systematic bits are passed, unaltered to the output, with a small delay through the encoding block and, second, the code bits are appended to the end of the systematic bits to create the coded packet.

The code bits are available in the cycle after the last systematic bit is output and are appended as a contiguous block with no processing delay. Encoding a packet does not take any longer than the time it takes to output a packet.

The time to output a packet is minimized when data exits on every clock cycle. For example, if the clock rate is 100 MHz, then 1-bit of the encoded packet can be output on every clock cycle. Therefore, the minimum time to output a maximum size packet (Block Size=96) is $(24*96)/(100x10^6)=23$ µs. The associated throughput rate is 100 Mbits/s or 4348 packets/s.

Since the time to input a packet (ND always asserted) and the time to encode a packet add no delay, the throughput of the core is the same as the output rate of the core. If CTS is deasserted, it stalls the core and increases the time taken to output a packet. This reduces the throughput of the core.

Core Resource Requirements and Performance

The core resource requirements and the clock rates achievable in Virtex-II Pro and Virtex-4 FPGAs are summarized in Table 5 and Table 6.

Table 5: **Core Resource Requirements and Maximum Clock Frequency for Virtex-II Pro FPGAs**

Enabled Rates	Maximum Block Size	Slices	Block RAM	MULT 18x18	Max. Clock Frequency (MHz)		Maximum Encoded Output Rate (Mbits/s)	
					-7	-5	-7	-5
All Rates	24							
All Rates	96							
5/6	24							
5/6	96							

Notes:
1. Packfactor option on mapper set to c-1.
2. Area and maximum clock frequencies are provided as a guide. They may vary with new releases of Xilinx implementation tools, etc.
3. Clock frequency does not take clock jitter into account and should be derated by an amount appropriate tot he clock source jitter specification.

Table 6: **Core Resource Requirements and Maximum Clock Frequency for Virtex-4 FPGAs**

Enabled Rates	Maximum Block Size	Slices	Block RAM	MULT 18x18	Max. Clock Frequency (MHz)		Maximum Encoded Output Rate (Mbits/s)	
					-10	-12	-10	-12
All Rates	24							
All Rates	96							
5/6	24							
5/6	96							

Notes:
1. Packfactor option on mapper set to c-1.
2. Area and maximum clock frequencies are provided as a guide. They may vary with new releases of Xilinx implementation tools, etc.
3. Clock frequency does not take clock jitter into account and should be derated by an amount appropriate tot he clock source jitter specification.

Ordering Information

This core can be downloaded from the Xilinx IP Center for use with the Xilinx CORE Generator v8.2i and later. The Xilinx CORE Generator is bundled with the ISE™ Foundation software at no additional charge.

To order Xilinx software, contact your local Xilinx sales representative. Information about additional Xilinx LogiCORE™ modules is available on the Xilinx IP Center.

References

1. *IEEE P802.16e/D12, "Draft IEEE Standard for Local and Metropolitan Area Networks. Part 16: Air Interface for Fixed and Mobile Broadband Wireless Access Systems,"* 2005-10-14.

Revision History

Date	Version	Revision
07/13/06	1.0	Initial Xilinx release.

Appendix 1: Coding Matrices

The Base Matrices defined in the IEEE 802.16e Standard are:

Rate 1/2:

-1	94	73	-1	-1	-1	-1	-1	55	83	-1	-1	7	0	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1
-1	27	-1	-1	-1	22	79	9	-1	-1	-1	12	-1	0	0	-1	-1	-1	-1	-1	-1	-1	-1	-1
-1	-1	-1	24	22	81	-1	33	-1	-1	-1	0	-1	-1	0	0	-1	-1	-1	-1	-1	-1	-1	-1
61	-1	47	-1	-1	-1	-1	-1	65	25	-1	-1	-1	-1	-1	0	0	-1	-1	-1	-1	-1	-1	-1
-1	-1	39	-1	-1	-1	84	-1	-1	41	72	-1	-1	-1	-1	-1	0	0	-1	-1	-1	-1	-1	-1
-1	-1	-1	-1	46	40	-1	82	-1	-1	-1	79	0	-1	-1	-1	-1	0	-1	-1	-1	-1	-1	-1
-1	-1	95	53	-1	-1	-1	-1	-1	14	18	-1	-1	-1	-1	-1	-1	-1	0	-1	-1	-1	-1	-1
-1	11	73	-1	-1	-1	2	-1	-1	47	-1	-1	-1	-1	-1	-1	-1	-1	0	0	-1	-1	-1	-1
12	-1	-1	-1	83	24	-1	43	-1	-1	-1	51	-1	-1	-1	-1	-1	-1	0	0	-1	-1	-1	-1
-1	-1	-1	-1	-1	94	-1	59	-1	-1	70	72	-1	-1	-1	-1	-1	-1	-1	0	0	-1	-1	-1
-1	-1	7	65	-1	-1	-1	-1	39	49	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	0	0	-1	-1
43	-1	-1	-1	-1	66	-1	41	-1	-1	-1	26	-1	-1	-1	-1	-1	-1	-1	-1	-1	0	-1	-1

Rate 2/3 A code:

3	0	-1	-1	2	0	-1	3	7	-1	1	-1	-1	-1	-1	1	0	-1	-1	-1	-1	-1	-1	-1
-1	-1	1	-1	36	-1	-1	34	10	-1	-1	18	2	-1	3	0	-1	0	0	-1	-1	-1	-1	-1
-1	-1	12	2	-1	15	-1	40	[illegible]	3	-1	15	-1	2	13	-1	-1	-1	0	0	-1	-1	-1	-1
-1	-1	19	24	-1	3	0	-1	6	-1	17	-1	-1	-1	8	39	-1	-1	-1	0	0	-1	-1	-1
20	-1	6	-1	-1	10	[illegible]	[illegible]	[illegible]	28	-1	14	-1	38	-1	-1	0	-1	-1	-1	0	0	-1	-1
-1	-1	10	-1	28	20	[illegible]	-1	8	-1	36	-1	9	-1	21	45	-1	-1	-1	-1	-1	0	0	-1
35	25	-1	37	-1	21	[illegible]	-1	5	-1	-1	0	-1	4	20	-1	-1	-1	-1	-1	-1	-1	0	0
-1	6	6	-1	-1	-1	4	-1	14	30	-1	3	36	-1	14	-1	1	-1	-1	-1	-1	-1	-1	0

Rate 2/3 B code:

[illegible]	-1	19	-1	47	-1	48	-1	36	-1	82	-1	47	-1	15	-1	95	0	-1	-1	-1	-1	-1	-1
-1	[illegible]	-1	[illegible]	-1	33	-1	3	-1	16	-1	37	-1	40	-1	48	-1	0	0	-1	-1	-1	-1	-1
10	-1	86	-1	62	-1	28	-1	85	-1	16	-1	34	-1	73	-1	-1	-1	0	0	-1	-1	-1	-1
-1	28	-1	32	-1	81	01	27	-1	88	-1	5	-1	56	-1	37	-1	-1	-1	0	0	-1	-1	-1
23	-1	29	-1	15	-1	30	-1	66	-1	24	-1	50	-1	62	-1	-1	-1	-1	-1	0	0	-1	-1
-1	30	-1	65	-1	54	-1	14	-1	0	-1	30	-1	74	-1	0	-1	-1	-1	-1	-1	0	0	-1
32	-1	0	-1	15	-1	56	-1	85	-1	5	-1	6	-1	52	-1	0	-1	-1	-1	-1	-1	0	0
-1	0	-1	47	-1	13	-1	61	-1	84	-1	55	-1	78	-1	41	95	-1	-1	-1	-1	-1	-1	0

Rate 3/4 A code:

6	38	3	93	-1	-1	-1	30	70	01	86	01	37	38	4	11	-1	46	48	0	-1	-1	-1	-1
62	94	19	84	-1	92	78	-1	15	-1	-1	92	-1	45	24	32	30	-1	-1	0	0	-1	-1	-1
71	-1	55	-1	12	66	45	79	-1	78	-1	-1	10	-1	22	55	70	82	-1	-1	0	0	-1	-1
38	61	-1	66	9	73	47	64	-1	39	61	43	-1	-1	-1	-1	95	32	0	-1	-1	0	0	-1
-1	-1	-1	-1	32	52	55	80	95	22	6	51	24	90	44	20	-1	-1	-1	-1	-1	-1	0	0
-1	63	31	88	20	-1	-1	-1	6	40	56	16	71	53	-1	-1	27	26	48	-1	-1	-1	-1	0

Rate 3/4 B code:

-1	81	-1	28	-1	-1	14	25	17	-1	-1	85	29	52	78	95	[illegible]	[illegible]	0	0	-1	-1	-1	-1
42	-1	14	68	32	-1	-1	-1	-1	70	43	11	36	40	33	57	38	[illegible]	-1	0	0	-1	-1	-1
-1	-1	20	-1	-1	63	39	01	70	67	-1	38	4	72	47	[illegible]	60	5	80	-1	0	0	-1	-1
64	2	-1	-1	63	-1	-1	3	51	-1	81	15	94	9	8[illegible]	36	[illegible]	19	-1	-1	-1	0	0	-1
-1	53	60	80	-1	26	75	-1	-1	-1	-1	86	77	[illegible]	[illegible]	[illegible]	60	25	-1	-1	-1	-1	0	0
77	-1	-1	-1	15	28	-1	35	-1	72	30	68	85	[illegible]	26	64	11	89	0	-1	-1	-1	-1	0

Rate 5/6 code:

1	25	55	-1	47	4	-1	91	84	8	8[illegible]	52	82	33	5	0	36	20	4	77	80	0	-1	-1
-1	6	-1	36	40	47	12	79	47	-1	41	21	12	71	14	72	0	44	49	0	0	0	0	-1
51	81	83	4	67	-1	21	-1	31	[illegible]	91	61	81	9	86	78	60	88	67	15	-1	-1	0	0
68	-1	50	15	-1	36	13	10	11	20	[illegible]	90	29	92	57	30	84	92	11	66	80	-1	-1	0