3GPP2 Turbo Encoder v2.0

Features

- Drop-in module for Virtex™-II, Virtex-II Pro, Virtex-4, Virtex-5, Spartan™-3, Spartan-3E, Spartan-3A/3AN/3A DSP FPGAs
- Implements the 3GPP2/CDMA-2000 Turbo Encoder specification [1]
- Double-buffered symbol memory for maximum throughput
- Flexible interfacing by means of optional control signals

Applications

The 3GPP2 Turbo Encoder core can be used in conjunction with the Xilinx 3GPP2 Turbo Decoder (available from the Xilinx CORE Generator™ system) to provide an extremely effective way of transmitting data reliably under low signal-to-noise conditions and to provide a performance close to the theoretical optimal performance as defined by the Shannon limit.

General Description

The 3GPP2 Turbo Encoder core is a parallel implementation of the convolutional turbo encoder specified by the 3GPP2/CDMA-2000 Turbo Encoder specification [1].

The theory of operation of the Turbo Codes is described in the paper by Berrou, Glavieux, and Thitimajshima [2].

LogiCORE Facts	
Core Specifics	
Supported Device Family	Virtex-II, Virtex-II Pro, Virtex-4, Virtex-5, Spartan-3, Spartan-3E, Spartan-3A/3AN/3A DSP
Provided with Core	
Documentation	Product Specification
Design File Formats	VHDL
Verification	VHDL Structural (UniSim) Model Verilog Structural (UniSim) Model
Instantiation Template	VHDL Wrapper Verilog Wrapper
Design Tool Requirements	
Xilinx Implementation Tools	ISE™ 9.1i or higher
Licensing	
Pay Core. Requires a full or evaluation license	
Support	
Provided by Xilinx, Inc @ www.xilinx.com	

The 3GPP2 Turbo Encoder input and output ports are shown in Figure 1.

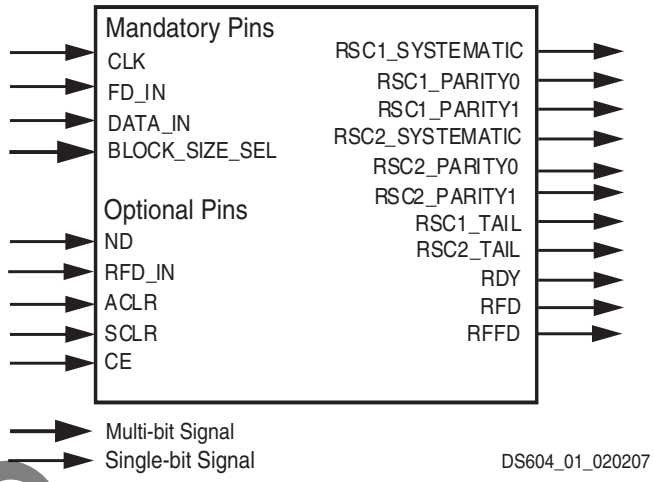


Figure 1: **TCC Encoder Pinout**

The encoder architecture is shown in Figure 2. It is a block-based processing unit. Each block of data is processed in two identical Recursive Systematic Convolutional (RSC) encoders, which generate high-weight codes. RSC1 processes the raw input data, while RSC2 processes an interleaved version of the input data. The coding operates on the principle that if an input symbol is corrupted in the sequence from RSC1, then it is unlikely also to be corrupted in the reordered sequence from RSC2, and vice versa.

The delay shown in Figure 2 is used to indicate that the input data is delayed before passing through RSC1. This ensures that the systematic data from RSC1 and RSC2 are block-aligned at the output. The input data is also double-buffered to maximize throughput.

Often some of the encoded output bits need not be transmitted, so they are omitted, or *punctured* from the output stream. Puncturing offers a dynamic trade off between code rate and error performance. When the channel is noisy or the data requires more protection, extra redundancy can be added, lowering the code rate. Puncturing is not implemented as part of the core.

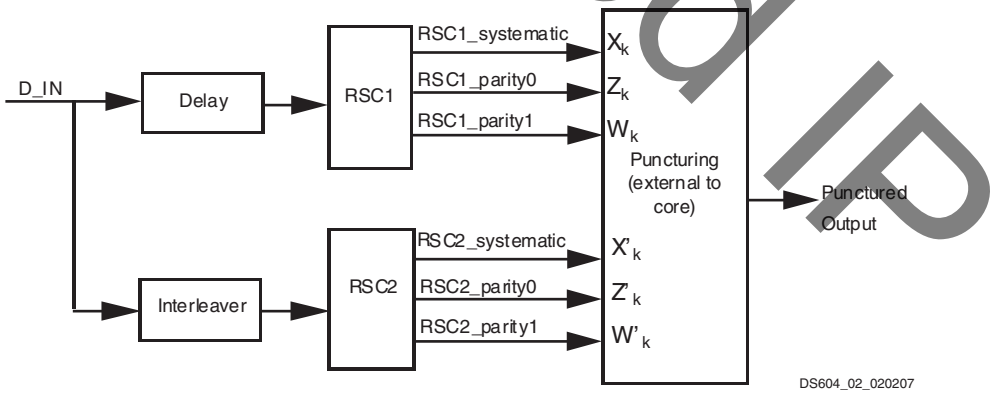


Figure 2: **TCC Encoder Structure**

Recursive Systematic Convolution (RSC) Encoder Structure

The schematic for each of the two RSCs and the transfer functions for the outputs are shown in Figure 3.

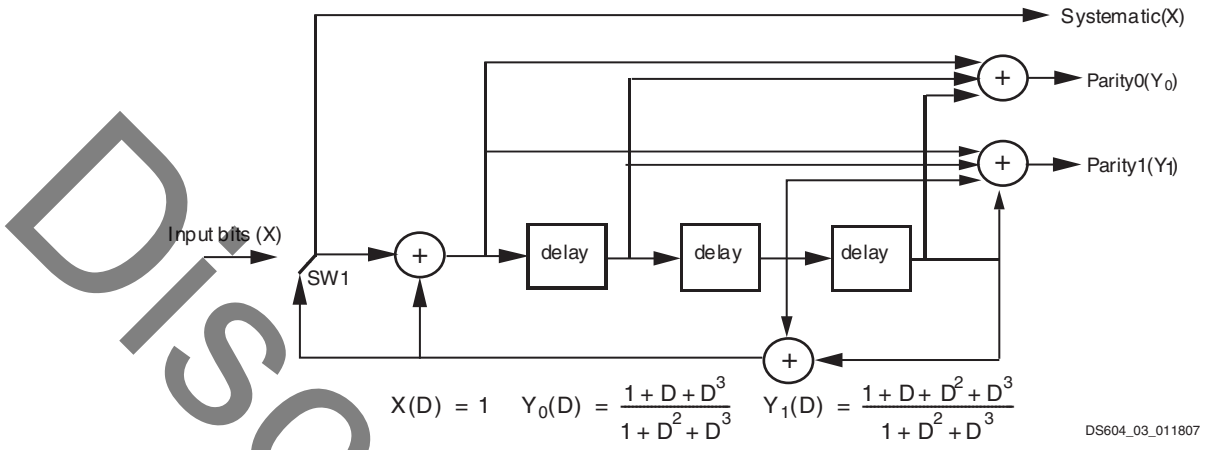


Figure 3: **TCC RSC Structure**

After a block of input data has been coded, the RSCs must return to the initial zero state. To force the RSC back to the all zero state, the input value is set equal to the feedback value by setting the control switch, SW1, to the lower position for three clock cycles.

During the first three tail bit periods, RSC2 is disabled, and the control switch of RSC1 is set to the lower position to output the RSC1 tail bits on the RSC1 systematic and parity outputs.

During the last three tail bit periods, RSC1 is disabled and the control switch of RSC2 is set to the lower position to output the RSC2 tail bits on the RSC2 systematic and parity outputs. The RSC2 systematic tail bits are duplicated on the `RSC1_SYSTEMATIC` output. This avoids the need for the user to connect the `RSC2_SYSTEMATIC` port.

Block Size Selection

The encoding sequence is initiated from a single First Data (FD) pulse. When a valid FD is detected, the block or frame size is determined by the 4-bit code on the `BLOCK_SIZE_SEL` input.

The values of `BLOCK_SIZE_SEL` for all of the 3GPP2 valid block sizes are shown in Table 1.

Table 1: **Block Size Select Codes**

BLOCK_SIZE_SEL		Turbo Encoder Block Size
0	0000	122
1	0001	250
2	0010	506
3	0011	762
4	0100	1018
5	0101	1530
6	0110	2042
7	0111	3066

Table 1: **Block Size Select Codes *(Continued)***

BLOCK_SIZE_SEL		Turbo Encoder Block Size
8	1000	4090
9	1001	5114
10	1010	6138
11	1011	8186
12	1100	12282

Input/Output Ports

The I/O ports of the core are summarized in Table 2.

Table 2: **I/O Ports**

Pin	Sense	Port Width (bits)	Description
ACLR	Input (optional)	1	**Asynchronous Clear** - When this is asserted (High), the encoder is asynchronously reset.
CLK	Input	1	**Clock** - All synchronous operations occur on the rising edge of the clock signal.
CE	Input (optional)	1	**Clock Enable** - When this is deasserted (Low), rising clock edges are ignored and the core is held in its current state. A valid clock edge' is one on which CE is High.
SCLR	Input (optional)	1	**Synchronous Clear** - When this is asserted (High) on a valid clock edge, the encoder is reset.
DATA_IN	Input	1	**Data Input** - This is the data to be encoded.
ND	Input (optional)	1	**New Data** - When this is asserted (High) on a valid clock edge, a new input value is read from the DATA_IN port.
FD_IN	Input	1	**First Data -** When this is asserted (High) on a valid clock edge, the encoding process is started. Qualified by ND
BLOCK_SIZE_SEL	Input	4	**Block Size Select** - This 4-bit port, which is read when FD_IN is sampled High, selects the block size to be encoded. See Table 1.
RFFD	Output	1	**Ready For First Data** - When this is asserted (High), the core is ready to start another encoder operation.
RFD	Output	1	**Ready For Data** - When this is asserted (High), the core is ready to accept input on the DATA_IN port.
RSC1_SYSTEMATIC	Output	1	**RSC1_systematic** - The systematic output from RSC1.
RSC1_PARITY0	Output	1	**RSC1_parity0** - The parity0 output from RSC1.
RSC1_PARITY1	Output	1	**RSC1_parity1** - The parity1 output from RSC1.
RSC2_SYSTEMATIC	Output	1	**RSC2_systematic** - The systematic output from RSC2.
RSC2_PARITY0	Output	1	**RSC2_parity0** - The parity0 output from RSC2.

Table 2: **I/O Ports *(Continued)***

Pin	Sense	Port Width (bits)	Description
RSC2_PARITY1	Output	1	**RSC2_parity1** - The parity1 output from RSC2.
RSC1_TAIL	Output	1	**RSC1_tail** - This indicates that the tail bits are being output on RSC1 when asserted (High).
RSC2_TAIL	Output	1	**RSC2_tail** - This indicates that the tail bits are being output on RSC2 when asserted (High).
RDY	Output	1	**Ready** - This indicates that there is valid data on the systematic and parity outputs when asserted (High).
RFD_IN	Input (optional)	1	**Ready For Data Input** - An output strobe signal. When deasserted (Low), the output side of the encoder is suspended. The Systematic and Parity data outputs and the associated control signals RDY, RSC1_TAIL and RSC2_TAIL are frozen.

Asynchronous Clear (ACLR)

The ACLR input port is optional. When ACLR is driven High, the core is reset to its initial state, i.e., the core is ready to process a new block. Following the initial configuration of the FPGA, the core is automatically in the reset state, so no further ACLR is required before an encoding operation can take place. ACLR is the only asynchronous input to the core.

Clock (CLK)

With the exception of asynchronous clear, all operations of the core are synchronized to the rising edge of CLK. If the optional CE pin is enabled, an active rising clock edge occurs only when CE is High. If CE is Low, the core is held in its current state.

Clock Enable (CE)

Clock enable is an optional input pin that is used to enable the synchronous operation of the core. When CE is High, a rising edge of CLK is acted upon by the core, but if CE is Low, the core remains in its current state. An active rising clock edge is on one which CE (if enabled) is sampled High.

Synchronous Clear (SCLR)

The SCLR signal is optional. When it is asserted High on a valid clock edge, the core is reset to its initial state, and the core is ready to process a new block. Following the initial configuration of the FPGA, the core is automatically in the reset state, so no further SCLR is required before an encoding operation can take place. If the CE input port is selected, SCLR is ignored when CE is Low.

Data In (DATA_IN)

The DATA_IN port is a mandatory input port which carries the unencoded data. The input process is started with a valid-FD signal and data is read serially into the DATA_IN port on a clock-by-clock basis. Block size clock cycles are, therefore, required to input each block. DATA_IN may be qualified by the optional ND port. (see below)

New Data (ND)

The optional ND signal is used to indicate that there is new input data to be read from the DATA_IN port. For example, if the input block size is 122, then 122 active High *ND-samples* are required to load a block of data into the encoder. ND is also used to qualify the FD_IN input. (see First Data (FD_IN)).

First Data (FD_IN)

FD_IN is a mandatory input port which is used to start the encoder operation. FD_IN is qualified by the optional ND input. If ND is not selected, a *valid-FD* means simply that FD_IN is sampled High on an active rising clock edge. If ND is selected, a *valid-FD* means that FD_IN and ND are both sampled High on an active rising clock edge.

When a *valid-FD* occurs, the first data is read from the DATA_IN port, and the value of the BLOCK_SIZE_SEL port is sampled. The core then continues loading data until a complete block has been input.

The FD_IN input should only be asserted when the RFFD output is High. (see Ready For First Data (RFFD)). If FD_IN is asserted when RFFD is Low, the behavior of the core is not specified

Block Size Select (BLOCK_SIZE_SEL)

This 4-bit port determines the size of the block of data to be written into the encoder. The block size select value is sampled on an active rising clock edge when FD_IN is High and ND (if selected) is High. If an invalid block size select code (not in the range 0-12) is sampled, the behavior of the core is not specified.

Ready For First Data (RFFD)

When this output is asserted High, it indicates that the core is ready to accept an FD_IN signal to start a new encoding operation. When a valid-FD signal is sampled, the RFFD signal goes Low and remains Low until it is safe to start another block.

Ready For Data (RFD)

When this pin is asserted High, it indicates that the core is ready to accept new input data. If RFD is selected, then it is High during the period that a particular block is input. When *block size* samples of data have been input, the RFD signal goes Low to indicate that the core is no longer ready to accept data.

RSC1 Systematic Output (RSC1_SYSTEMATIC)

RSC1_SYSTEMATIC is a delayed version of the uninterleaved input data. During trellis termination, the RSC1_SYSTEMATIC port also carries systematic and parity tail bits.

RSC1 Parity0 Output (RSC1_PARITY0)

RSC1_PARITY0 is the Y0 output from RSC1. (See Figure 3.)

RSC1 Parity1 Output (RSC1_PARITY1)

RSC1_PARITY1 is the Y1 output from RSC1. (See Figure 3.)

RSC2 Systematic Output (RSC2_SYSTEMATIC)

RSC2_SYSTEMATIC is a delayed and interleaved version of the input data. This output is provided for diagnostic purposes. It is normally left unconnected.

RSC2 Parity0 Output (RSC2_PARITY0)

RSC2_PARITY0 is the Y0 output from RSC2. (See Figure 3.)

RSC2 Parity1 Output (RSC2_PARITY1)

RSC2_PARITY1 is the Y1 output from RSC2. (See Figure 3.)

RSC1 Tail Output (RSC1_TAIL)

RSC1_TAIL is asserted High for three cycles at the end of each output block to indicate trellis termination of RSC1.

RSC2 Tail Output (RSC2_TAIL)

RSC2_TAIL is asserted High for three cycles at the end of each output block to indicate trellis termination of RSC2.

Ready (RDY)

This signal is asserted High when there is valid data on the SYSTEMATIC and PARITY output ports. RDY is asserted for block size+6 valid clock cycles each block, to include the tail bits.

Ready For Data In (RFD_IN)

RFD_IN is an optional pin which can be used to inhibit the output of the core while allowing current input operations to continue. RFD_IN is intended to be used as a means for a downstream system using the encoded data to indicate that it is not ready to handle any further data. If RFD_IN is Low, all systematic and parity outputs, RSC1_TAIL, RSC2_TAIL, and RDY outputs are inhibited. The operation of RFD_IN is described in further detail later in this document.

Functional Description

Double-Buffering

Figure 4 illustrates the LOAD and OUTPUT operations.

Data blocks are written alternately to the two symbol memories, MEM0 and MEM1, and also read out alternately, thereby, maximizing the data throughput.

If the block size increases or the core is inhibited by negating ND, the LOAD operation may take longer than the OUTPUT operation, in which case, the RDY output port may be driven Low to indicate the OUTPUT operation is complete. Similarly, if the block size decreases or the core is inhibited by negating RFD_IN, the OUTPUT operation may take longer than the LOAD operation. In which case, the RFD port may be driven Low to indicate that the LOAD operation is complete and the core is no longer ready for data. The effect of changing the block size is shown in Figure 4(a), and the effects of ND and RFD_IN are shown in Figure 4(b)

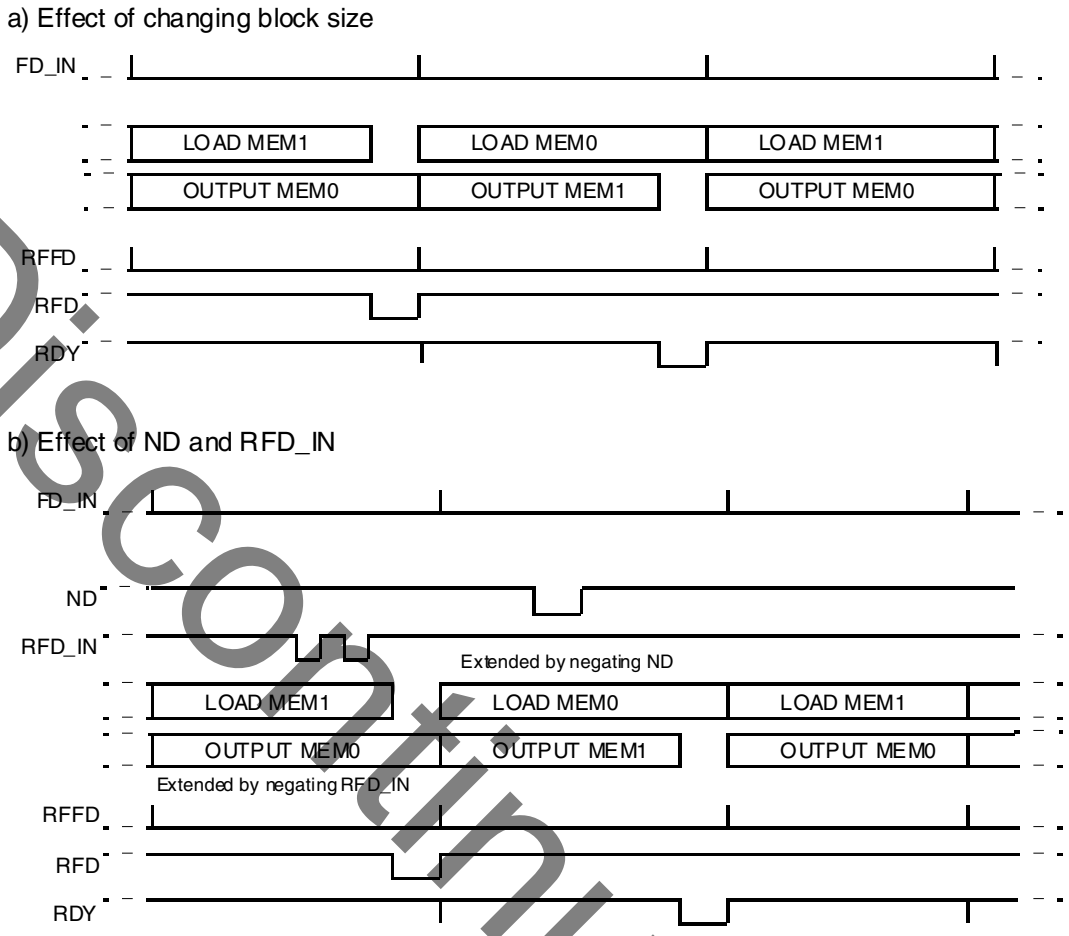


Figure 4: **Double-Buffered Mode Data Throughput**

Input Control Signals

Figure 5 shows the signals associated with the data input side of the core. If, on an active rising edge of CLK, FD_IN and ND (if selected) are both sampled High, this is known as a valid-FD, or valid First Data signal.

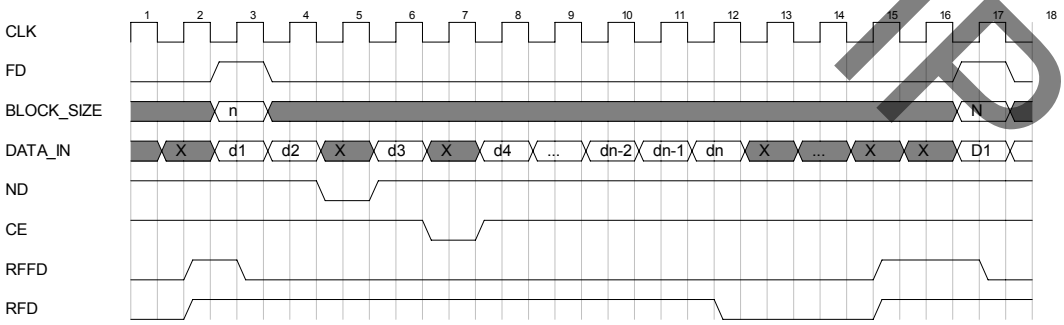


Figure 5: **Input Timing**

When a valid-FD is sampled, the RFFD signal is driven Low to indicate that the core is no longer waiting for FD_IN. The block size, in this case *n*, of the current input block is sampled on the BLOCK_SIZE port, and the first data symbol, *d1*, is sampled on the DATA_IN port.

A High on the ND port indicates that the value on the DATA_IN port is new data. If ND is sampled Low, then the DATA_IN port is not sampled, and the internal write address does not advance.

The core continues to input data, until *n* new data samples have been accepted, whereupon RFD is normally driven Low to indicate that the core is no longer ready for data and the core stops sampling the DATA_IN port.

When the core is ready to accept a new block of data, RFFD and RFD are both driven High. The time at which this occurs is determined by how long it takes the core to output the current output block, which depends on its block size, and on whether or not the output side of the core is inhibited by negating RFD_IN. If the input operation completes and the output cycle is already complete, it is possible for RFFD to be asserted, and a new input cycle to be started without RFD going Low.

After asserting RFFD, the core waits until the next valid-FD is sampled, whereupon a write cycle is started, in this case, with block size *N*.

The behavior of the core is not specified if, on a valid-FD, an invalid BLOCK_SIZE_SEL is sampled, or RFFD is sampled Low (core not ready for a new block). However, the core will recover if a valid-FD is sampled, with a valid BLOCK_SIZE_SEL value when RFFD is High.

Trellis Termination

During the first three tail bit periods, RSC2 is disabled and the control switch of RSC1 is set to the lower position to output the RSC1 tail bits on the RSC1 systematic and parity outputs. During the last three tail bit periods, RSC1 is disabled and the control switch of RSC2 is set to the lower position to output the RSC2 tail bits on the RSC2 systematic and parity outputs.

The RSC2 systematic tail bits are also output on the RSC1 Systematic output. Typically, the RSC2 systematic data is only transmitted during the tail bit period, so by multiplexing the RSC2 systematic data onto the RSC1_SYSTEMATIC port, the user can usually ignore the RSC2_SYSTEMATIC port. However, it is provided to maximize flexibility.

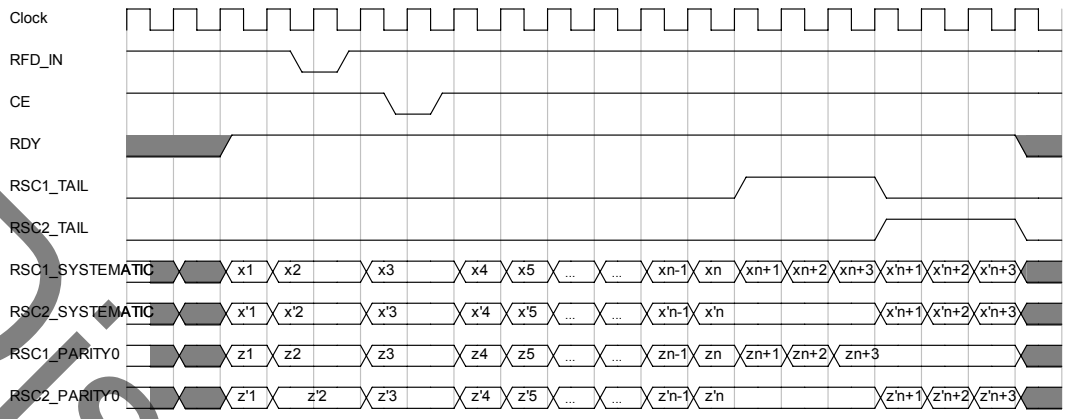
Output Control Signals

The RDY signal is driven High to indicate that there is valid data on the systematic and parity ports. In addition, the RSC1_TAIL and RSC2_TAIL outputs are provided to indicate trellis termination of RSC1 and RSC2, respectively. A High on RSC1_TAIL indicates that RSC1 tail bits are being output, and a High on RSC2_TAIL indicates that RSC2 tail bits are being output. The output timing is shown in Figure 6.

Flow control on the output side can be implemented with the optional RFD_IN input port. If the RFD_IN port is sampled Low on an active rising clock edge, the RSC output ports, RDY, and the internal circuitry associated with these outputs are frozen.

The input side of the core is not directly affected by the RFD_IN port. However, if RFD_IN is deasserted often enough, the time taken to output a block can be extended such that the assertion of RFFD is delayed, which acts to prevent overrun on the input side.

RSC1_PARITY1 and RSC2_PARITY1 have the same timing as RSC1_PARITY0 and RSC2_PARITY0, respectively.



x1 ... xn are the uninterleaved input bits, delayed, for block size n
x'1 ... x'n are the interleaved input bits
z1 ... zn+3 are the RSC1 Parity1 output bits
z'1 ... z'n+3 are the RSC2 Parity0 output bits

Figure 6: **Output Timing**

Throughput

Throughput is the average number of bits per second, as measured at the input. As mentioned previously, the maximum throughput is achieved when the block size is constant. For each block, there is an overhead of 6 cycles due to the time required for the tail bits, plus another 3 cycles due to internal delays. Therefore, throughput is also maximized by using the largest available block size. As a percentage of clock speed, for any fixed block size, the throughput is given by:

$$\text{Input data throughput} = (\text{blocksize} / (\text{blocksize}+9)) * 100\%$$

Latency

The latency of the encoder is the number of clock cycles between a valid-FD being sampled and the assertion of the RDY at the start of the corresponding output block. As the encoder is double-buffered, this latency depends upon the relative sizes of the current block and the preceding block, and partly on internal delays.

For two consecutive input blocks, of sizes blocksize^{i-1} and blocksize^{i}, respectively:

If the preceding block size, blocksize^{i-1}, is greater than the current block size, blocksize^{i}, the latency is determined by the time required to finish reading the preceding block. Therefore, the latency for the current block is given by:

$$\text{Latency} = \text{blocksize}^{i-1} + 20$$

If the current block size, blocksize^{i}, is greater than or equal to the preceding block size, blocksize^{i-1}, the latency is determined by the time required to write the current block to memory. In this case, the latency for the current block is:

$$\text{Latency} = \text{blocksize}^{i} + 17$$

Performance and Resource Usage

Table 3: **Performance and Resource Usage**

Option Pins	Resource	Notes	Virtex-4 XC4VSX55-10
None	Area (slices)	1,2	253
	Block Memories (18K)		2
	MULT18x18s or DSP48s		4
	Speed (MHz)	1,3	368
All (CE, SCLR, ACLR, RFD_IN, ND)	Area (slices)	1,2	288
	Block Memories (18K)		2
	MULT18x18s or DSP48s		4
	Speed (MHz)	1,3	322
RFD_IN	Area (slices)	1,2	269
	Block Memories (18K)		2
	MULT18x18s or DSP48s		4
	Speed (MHz)	1,3	350
ND	Area (slices)	1,2	254
	Block Memories (18K)		2
	MULT18x18s or DSP48s		4
	Speed (MHz)	1,3	366

Notes:
1. Area and maximum clock frequencies are provided as a guide. They may vary with new releases of Xilinx implementation tools, etc.
2. Area figures obtained with the core instantiated within a single registered wrapper, with the wrapper registers in the IOBs.
3. Speed figures obtained with the core instantiated within a double registered wrapper, with the outer wrapper registers in the IOBs. Clock frequency does not take clock jitter into account and should be derated by an amount appropriate to the clock source jitter specification.

Evaluation License

Xilinx provides a full system hardware evaluation of the IP core. After filling out the evaluation request form, you will be able to download the evaluation core. You will be able to parameterize, generate and instantiate this IP in your design. You will be able to perform functional and timing simulation and download and configure your design in hardware. The evaluation license allows for a bitstream to be generated. The resulting IP will be fully functional for 2-3 hours. After that, the IP times out and you will need to download and reconfigure the FPGA.

References

1. *CDMA2000 High Rate Packet Data Air Interface Specification* 3GPP2 C.S0024-B Version 1.0, May 2006
2. *Near Shannon Limit Error-correcting Coding and Decoding Turbo Codes*, C. Berrou, A. Glavieux, and P. Thitimajshima, IEEE Proc 1993 International Conference Committee, pp1064-1070.

Ordering Information

This Xilinx LogiCORE™ product is provided under the terms of the SignOnce IP Site License.

To evaluate this core in hardware, the user can generate an evaluation license, which is accessed from the Xilinx IP Evaluation page.

After purchasing the core, the user will receive instructions for registering and generating a full license. The full license can be requested and installed from the Xilinx IP Center for use with the Xilinx CORE Generator™. The CORE Generator is bundled with all Alliance™ series software package at no additional charge.

Contact the local Xilinx sales representative for pricing and availability on Xilinx LogiCORE products and software.

France Telecom, for itself and certain other parties, claims certain intellectual property rights covering Turbo Codes technology, and has decided to license these rights under a licensing program called the Turbo Codes Licensing Program. Supply of this IP core does not convey a license nor imply any right to use any Turbo Codes patents owned by France Telecom, TDF or GET. Please contact France Telecom for information about its Turbo Codes Licensing Program at the following address: France Telecom R&D, VAT/TURBOCODES 38, rue du Général Leclerc 92794 Issy Moulineaux Cedex 9.

Revision History

The following table shows the revision history for this document.

Date	Version	Revision
02/15/07	2.0	Initial Xilinx release
04/02/07	2.5	Added support for Spartan-3A DSP devices.