LogiCORE IP Convolutional Encoder v7.0

DS248 March 1, 2011 **Product Specification**

Introduction

The Convolution Encoder core can be used in a wide variety of error correcting applications and is typically used in conjunction with the Viterbi Decoder (DS247).

Features

- High-speed compact convolution encoder with puncturing option
- Available for Kintex™-7, Virtex®-7, Virtex-6, Virtex-5, Virtex-4, Spartan®-6, Spartan-3 and Spartan-3A/3AN/3A DSP/XA
- Parameterizable constraint length from 3 to 9
- Parameterizable convolution codes
- Parameterizable puncture codes
- Puncturing rates from 2/3 to 12/23 available
- For use with Xilinx CORE Generator™ Version 13.1

LogiCORE IP Facts Table	
Core Specifics	
Supported Device Family[1]	Virtex-7 and Kintex-7, Virtex-6, Virtex-5, Virtex-4, Spartan-6, Spartan-3, Spartan-3A/3AN/3A DSP/XA
Supported User Interfaces	Not Applicable
Provided with Core	
Documentation	Product Specification
Design Files	Netlist
Example Design	Not Provided
Test Bench	Not Provided
Constraints File	Not Applicable
Simulation Model	VHDL behavioral model in the xilinxcorelib library VHDL UniSim structural model Verilog UniSim structural model
Tested Design Tools	
Design Entry Tools	CORE Generator tool 13.1 System Generator for DSP 13.1
Simulation	Mentor Graphics ModelSim 6.6d Cadence Incisive Enterprise Simulator (IES) 10.2 Synopsys VCS and VCS MX 2010.06 ISIM 13.1
Synthesis Tools	N/A
Support	
Provided by Xilinx, Inc.	

[1] For a complete listing of supported devices, see the release notes for this core.

Functional Description

The basic architecture of the encoder is shown in Figure 1. The incoming data is brought into the constraint register a bit at a time, and the output bits are generated by modulo-2 addition of the required bits from the constraint register. The bits to be XOR'd are selected by the convolution codes as shown in Figure 1.

Convolution encoding is used to encode data prior to transmission over a channel. The received data is decoded by the classic Viterbi decoder. In a basic convolution encoder, two or three bits (depending on the encoder output rate) are transmitted over the channel for every input bit.

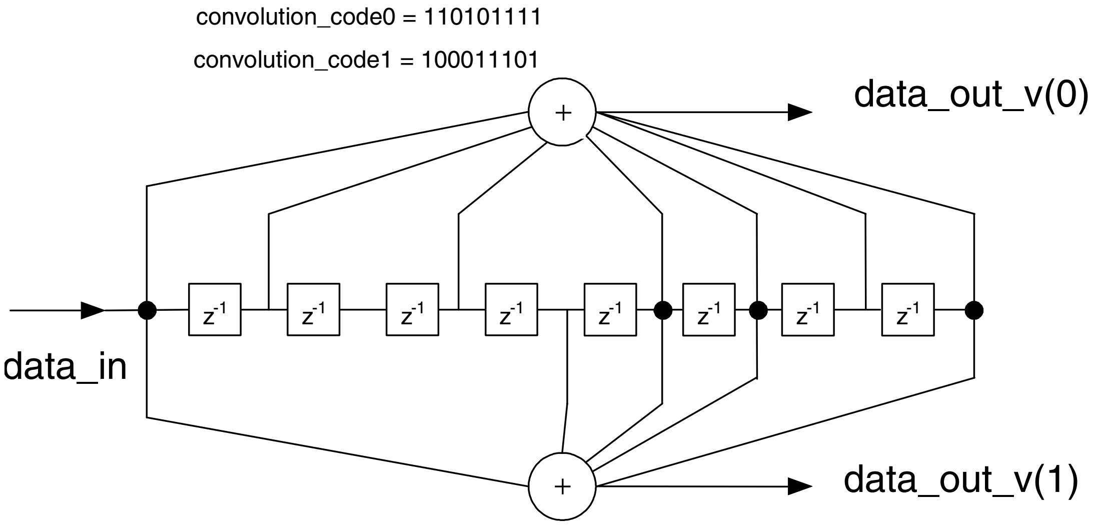


Figure 1: **Convolution Encoder Constraint Length 9**

Puncturing

The encoder can also puncture the data prior to transmission. In the punctured case, the basic convolution encoder is always a rate 1/2 encoder, two bits output for every one bit input. After the encoding, certain bits of the rate 1/2 encoded data are punctured (or deleted) and not transmitted. Thus, for a rate 3/4 punctured encoder, for every three input bits, only four of the six encoded bits generated by the encoder are actually transmitted, as shown in Figure 2. The two bits output from the encoder are punctured according to a pair of puncture codes. The puncture code is a bit pattern that identifies which bits from the encoder are to be transmitted. The use of puncturing significantly reduces the number of bits to be transmitted over the channel. For a puncture code of length n, exactly m bits are transmitted, that is, the rate of the encoder is n/m, where n<m<2n. For rate n/m puncturing, the length of each puncture code must be n bits and the total number of non-zero bits must be equal to m. The value of n can range from 2 to 12.

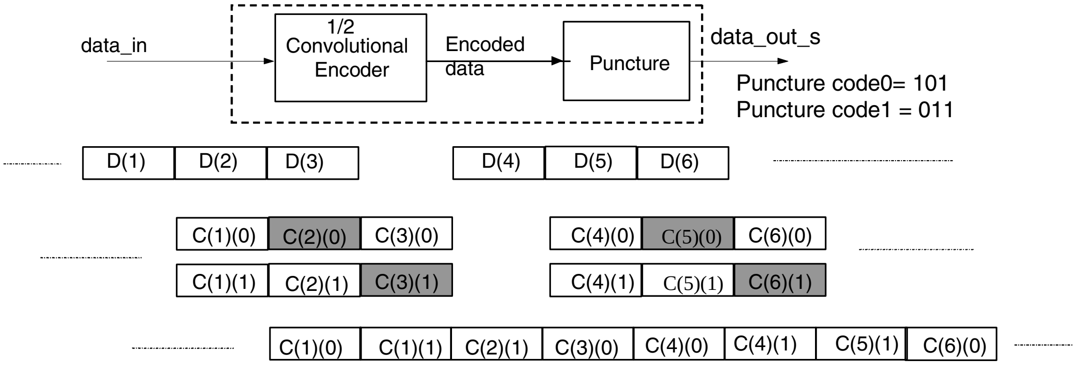

data_in
1/2 Convolutional Encoder
Encoded data
Puncture
data_out_s
Puncture code0= 101
Puncture code1 = 011
D(1) D(2) D(3) D(4) D(5) D(6)
C(1)(0) C(2)(0) C(3)(0) C(4)(0) C(5)(0) C(6)(0)
C(1)(1) C(2)(1) C(3)(1) C(4)(1) C(5)(1) C(6)(1)
C(1)(0) C(1)(1) C(2)(1) C(3)(0) C(4)(0) C(4)(1) C(5)(1) C(6)(0)

Figure 2: **Puncture Encoding with Single-Channel Output**

For the punctured encoder, the data can be output as a single bit stream, as shown in Figure 2. This single-channel output requires the use of the ND (new data) and RFD (ready for data) inputs as there are more bits output than bits input. Also, in the single-channel case, the RFD signal is low for m-n clock cycles after receiving the n input bits, as shown Figure 9.

The dual-channel output for puncturing outputs the data in pairs, as shown in Figure 3. In this situation, there is less data output than input and the RFD signal, if present, is always high. The RDY (ready) signal indicates when the output data is valid in both cases. For timing diagrams for the dual output case, see Figure 10.

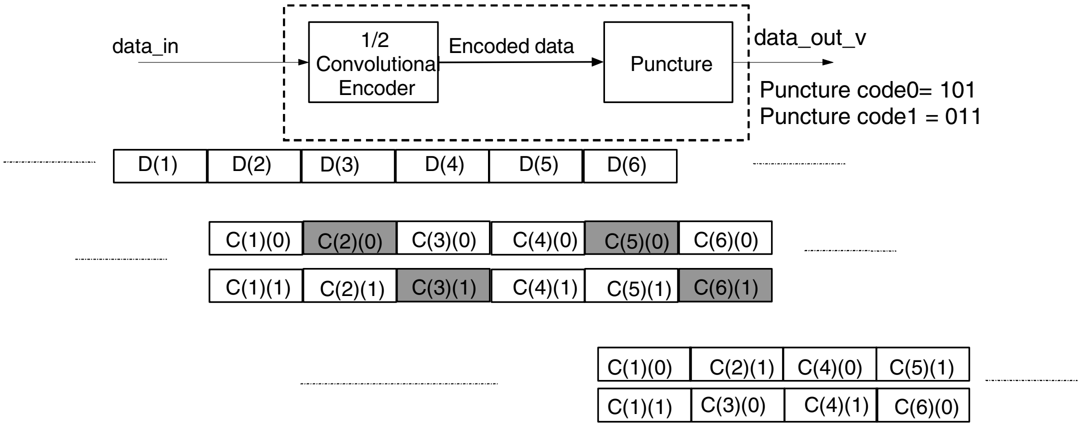

data_in
1/2 Convolutiona Encoder
Encoded data
Puncture
data_out_v
Puncture code0= 101
Puncture code1 = 011
D(1) D(2) D(3) D(4) D(5) D(6)
C(1)(0) C(2)(0) C(3)(0) C(4)(0) C(5)(0) C(6)(0)
C(1)(1) C(2)(1) C(3)(1) C(4)(1) C(5)(1) C(6)(1)
C(1)(0) C(2)(1) C(4)(0) C(5)(1)
C(1)(1) C(3)(0) C(4)(1) C(6)(0)

Figure 3: **Puncture Encoding with Dual-Channel Output**

Pinout

Some of the pins are optional. These should be selected only if they are genuinely required, as their inclusion might result in an increase in the core size.

A representative symbol with the signal names for the bus output case is shown in Figure 4. The single-bit output is shown in Figure 5.

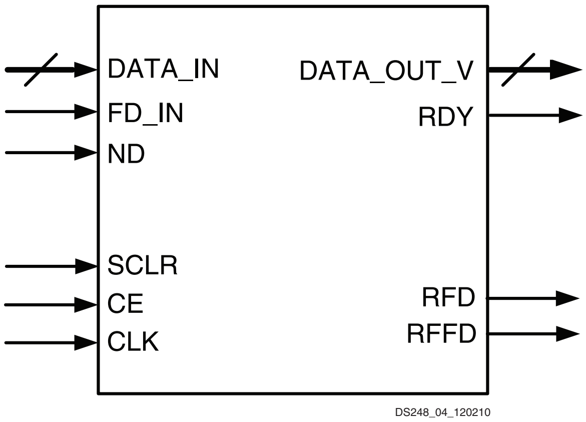


Figure 4: **Pinout for Bus Output**

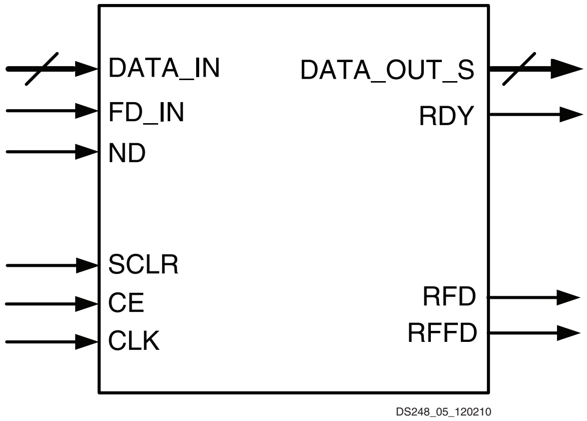


Figure 5: **Pinout for Single-Bit Output**

Table 1 summarizes the signal functions. They are described in more detail in the remainder of this section. Timing diagrams for the signals are shown in Control Signals, page 8.

Table 1: **Core Signal Pinout**

Signal	Direction	Description
DATA_IN	Input	**Data Input**: Input bit to be encoded.
FD_IN (puncture only)	Input	**First Data**: Indicates the start of a new puncture group, only available for a punctured core.
ND	Input	**New Data**: Indicates new data on DATA_IN. Optional for basic encoder must be present for punctured core in the single-channel output case
CE (Optional)	Input	**Clock Enable**: Freezes state of core when low.
SCLR (Optional)	Input	**Synchronous Reset**: Re-initializes core control logic.
CLK	Input	**Clock**: Clock input, all core operation is synchronous with the CLK input.
DATA_OUT_V (Optional)	Output	**Data Output**: Output encoded data as vector of width equal to output rate or of width 2 for punctured data with the dual-channel option.
DATA_OUT_S (Optional)	Output	**Data Output**: Output encoded data as a single bit. Available for only single-channel punctured cores.

Table 1: **Core Signal Pinout *(Cont'd)***

Signal	Direction	Description
RDY (Optional)	Output	**Output Ready**: Indicates valid data on output port DATA_OUT_V or DATA_OUT_S.
RFD (Optional)	Output	**Ready for Data**: Indicates that the core is ready to receive new data.
RFFD (Optional)	Output	**Ready for First Data**: Indicates that FD_IN may be asserted.

DATA_IN Input

This is the single-bit input for the bits to be encoded. Bits on DATA_IN are input to the constraint register and encoded; puncturing is then carried out if required and the encoded data is output on DATA_OUT_V or DATA_OUT_S.

FD_IN Input

The FD_IN (First Data) input is present only on punctured cores and is used to indicate the start of a new puncture group. The term "valid FD_IN pulse" is used to describe the case when FD_IN and ND (and CE, if appropriate) are asserted logic-high while a rising edge of CLK occurs. It is recommended that a valid FD_IN pulse last for only a single clock cycle. However, the core supports the case where it is asserted logic-high for multiple clock cycles. It is not required to assert the FD_IN signal; the data is punctured based on the first valid ND signal until FD_IN is asserted. A valid FD_IN signal can be asserted at any time to change the position of the puncturing on the encoded data.

ND Input

When the ND (New Data) input is sampled logic-high, it signals that a new symbol on DATA_IN should be sampled on the same rising clock edge. ND must be present for the punctured option. Like all the synchronous inputs, ND is ignored if CE is low. For the punctured case, ND is valid only if the RFD (ready for data) signal is high and is ignored if the RFD signal is low.

CE Input

The Clock Enable input is another optional pin. When CE is deasserted (low), all the synchronous inputs are ignored and the core remains in its current state. The state of the core is frozen while CE is low. See Figure 6.

SCLR Input

When SCLR is asserted (high), all the core flip-flops are synchronously initialized. The core remains in this state until SCLR is deasserted. SCLR is an optional pin; the core can function correctly without it.

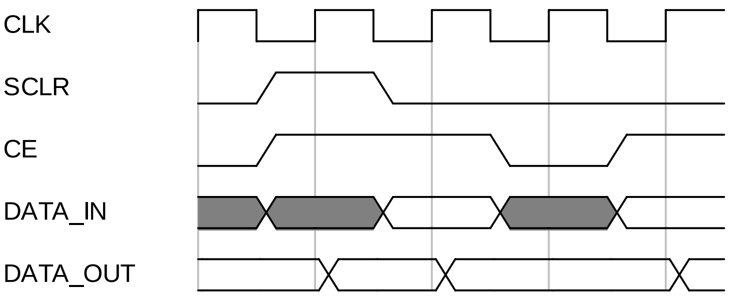


Figure 6: **CE and SCLR Timing**

DATA_OUT_V Output

This is the output bus for the non-punctured encoder and for the punctured encoder if the dual output option is selected. For the non-punctured encoder, the bus is of width equal to the output rate. For the punctured case, the bus always has a width equal to 2. For the non-punctured encoder, one new symbol is output on DATA_OUT_V for each input bit sampled on DATA_IN.

DATA_OUT_S Output

This is the output bus for the punctured encoder when the dual channel output is not selected. Punctured data in this case is output serially.

RDY Output

The RDY (Ready) output indicates valid data on DATA_OUT_V or DATA_OUT_S.

RFD Output

RFD (Ready for Data) indicates that the core is ready to sample new data on DIN. For the non-punctured core, RFD is permanently high because the core is always ready for new data. For the punctured core with dual output, again the RFD is permanently high. The output is provided in these cases only for consistency with other Xilinx cores that provide handshaking signals. The core does not sample new data during synchronous resets, even though RFD is high.

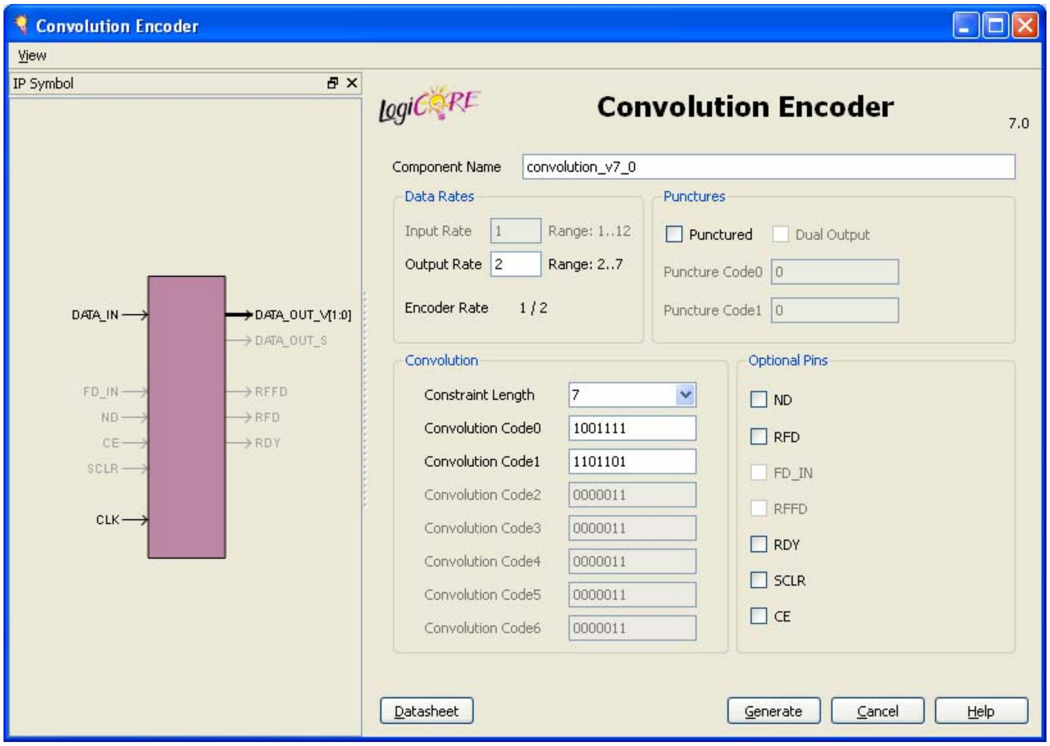
For the punctured core with single bit output, the RFD signal is mandatory, as the core generates m outputs for every n inputs if the puncture code is n/m. The RFD signal is low for m-n clock cycles after n valid ND signals have been received. If ND is asserted while RFD is low, the ND signal is ignored and input data is not sampled.

RFFD Output

When RFFD (Ready for First Data) is high indicates that FD_IN can be asserted. When a valid FD pulse is received, RFFD is deasserted immediately. It then remains low until the core is ready to accept the first data bit of a new block; see Figure 9 and Figure 10. An FD_IN pulse can be safely applied at any time and the puncturing pattern is reset to start at the new location.

CORE Generator Parameters

Figure 7 illustrates the main CORE Generator screen. Each parameter is defined below. To generate a core, click Finish.



Convolution Encoder 7.0

Component Name: convolution_v7_0

Data Rates

Input Rate: 1 Range: 1..12

Output Rate: 2 Range: 2..7

Encoder Rate: 1 / 2

Punctures

□ Punctured □ Dual Output

Puncture Code0: 0

Puncture Code1: 0

Convolution

Constraint Length: 7

Convolution Code0: 1001111

Convolution Code1: 1101101

Convolution Code2: 0000011

Convolution Code3: 0000011

Convolution Code4: 0000011

Convolution Code5: 0000011

Convolution Code6: 0000011

Optional Pins

□ ND
□ RFD
□ FD_IN
□ RFFD
□ RDY
□ SCLR
□ CE

Figure 7: **Convolution Encoder Graphical User Interface**

Component Name

The component name is used as the base name of the output files generated for the core. Names must begin with a letter and must be composed from the following characters: a to z, 0 to 9 and "_".

Data Rate

Not Punctured: Only the output rate can be modified. Its value can be integer values from 2 to 7, resulting in a rate 1/2 or rate 1/7 encoder, respectively.

Punctured: Only the input rate can be modified. Its value can range from 2 to 12, resulting in a rate n/m encoder where n is the input rate and n<m<2n.

Constraint Length

The length of the constraint register plus 1 in the encoder. This value can be in the range 3 to 9, inclusive.

Convolution Codes

Convolution codes are used to generate the encoder outputs. For a punctured core, only two are required, because the encoder internal to the punctured core is always rate 1/2. The codes are entered in binary and have length equal to the Constraint Length.

Punctured

This parameter determines whether the core is punctured.

Punctured Codes

The two puncture pattern codes used to remove bits from the encoded data prior to output. The length of each puncture code must be equal to the puncture input rate, and the total number of bits set to 1 in the two codes must equal the puncture output rate (m) for the codes to be valid. A 0 in any position indicates that the output bit from the encoder is not transmitted. See Figure 2 for an example of a rate 3/4 punctured encoder.

Optional Pins

Check the boxes of the optional pins that are required. Select only pins that are genuinely required, as each selected pin uses more FPGA resources and can result in a less-than-maximum operating frequency.

Parameter Ranges

Valid ranges for the parameters are given in Table 2.

Table 2: **Parameter Ranges**

Parameter	Min.	Max	Notes
Output Rate	2	7	Range for non-punctured encoder.
Input Rate	2	12	Input rate range for punctured cores.
Output Rate	input_rate+1	(2xinput_rate)-1	Output rate range for punctured cores.
Constraint Length	3	9	-
Convolution Code	Bit width = Constraint length		-
Punctured Code	Bit width = Input rate		-

Control Signals

Non-punctured Encoder

For the non-punctured case, data input can be controlled by the ND (new data) signal, or data can be continuously input if no ND signal is present or ND is held permanently high. If data is continuously input, the convolved data is available on the output one clock cycle after the data is input. If the ND signal is present, the enabling of the data through the cores is controlled by the ND signal and the RDY (ready) signal matches the ND input, as displayed in Figure 8.

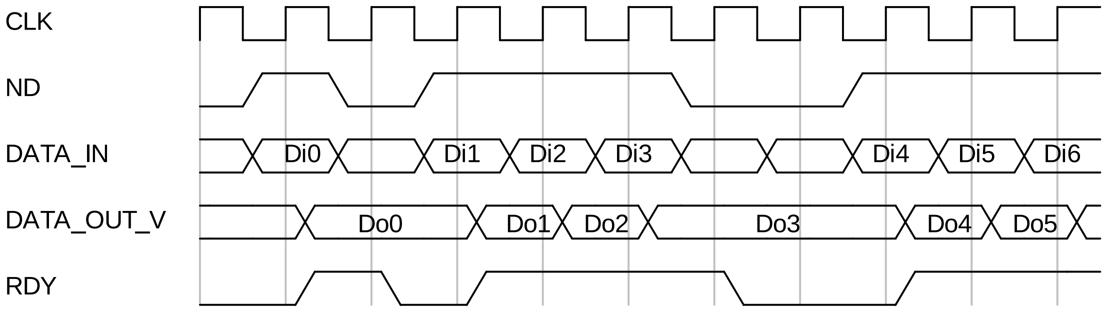


Figure 8: **Non-punctured Encoder with ND Signal**

Punctured Encoder

For the puncturing case, the control signals vary depending on whether the data is output in a single bit stream or in the dual-channel mode.

Single-Channel Output

For the single-channel output, when using puncturing, the core must have ND (new data) and RFD (ready for data) signals. After n input bits (puncture input rate) have been received, the RFD (ready for data) signal goes low for m-n clock periods. The RDY signal is used to indicate that valid output data is available which is output in a block of m bits (puncture output rate), as shown in Figure 9.

If there is no FD_IN (first data) input on the core, the puncturing pattern starts with the first ND input. If there is an FD_IN input is present, puncturing again starts with the first ND and continues in a repeating pattern until FD_IN is asserted. FD can be asserted only when ND is high. FD_IN resets the puncturing pattern to start on the current ND.

Dual-Channel Output

For the dual-channel punctured case, the ND and RFD are optional pins on the core. If RFD is present, it is always high. After 2*n input bits have been received, where n is the punctured input rate, m symbols are output on DATA_OUT_V (width 2) after a certain latency. The RDY signal indicates the valid outputs. If the ND signal is always high or not present, the data is output in blocks of m symbols (width 2) and the RDY signal is high for m clock cycles and low for the next n-1. See Figure 10 for the 3/4 rate dual-channel punctured encoder. The FD_IN, as in the single-channel case, resets the start location of the puncturing pattern.

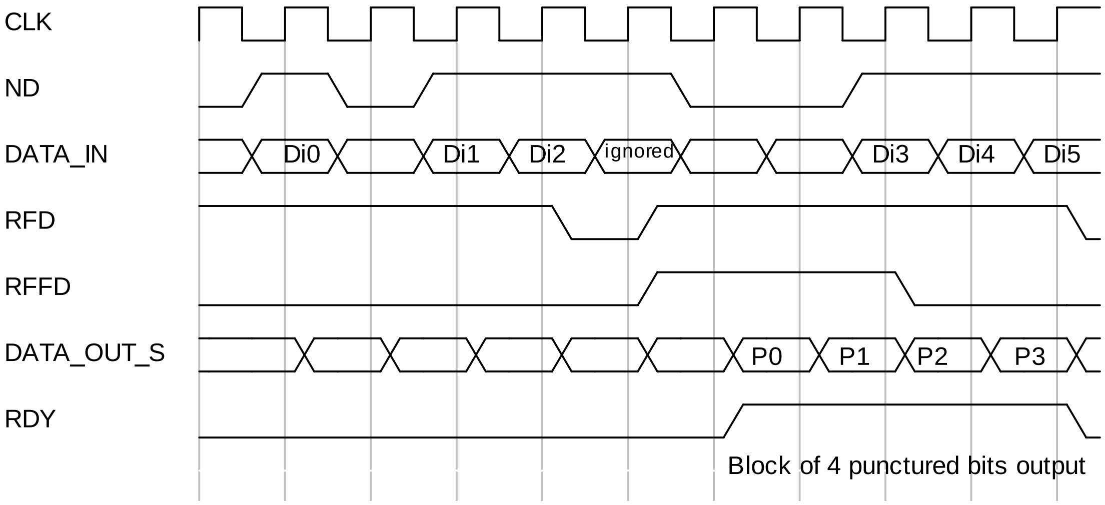


Figure 9: **Example 3/4 Single-Output Punctured Cod**

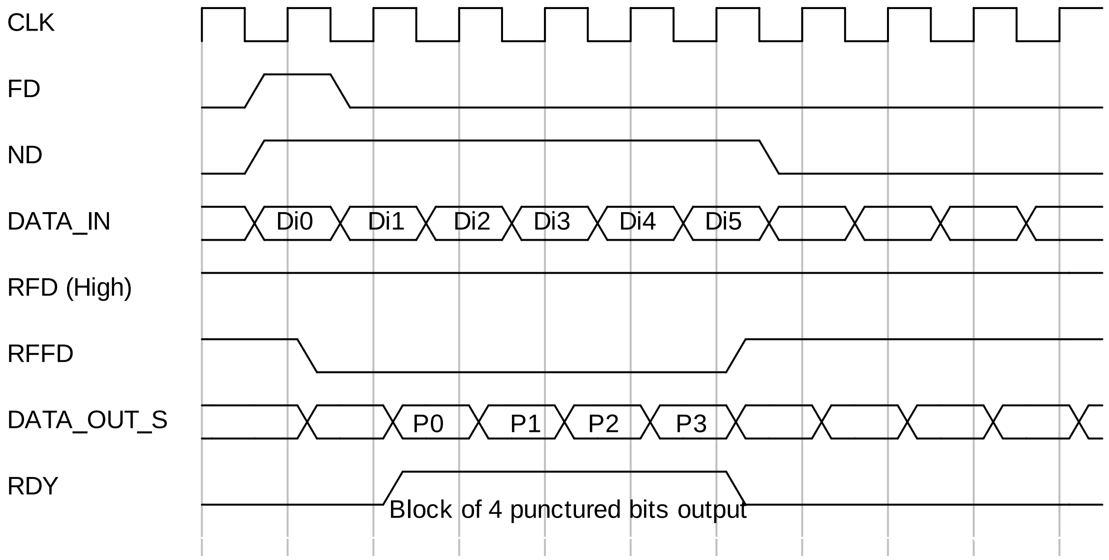


Figure 10: **Punctured Dual-Channel Rate 3/4**

Punctured Pipelining

The convolution core uses a highly pipelined approach to ensure maximum encoding throughput. This pipelined delay must be taken into account when associating a particular input data block with a corresponding output block.

Following a core reset or on the first data block entered into the core, the first data output block, as indicated by the first active RDY block, consists always of zeros. The output corresponding to the first data input block is associated with the second RDY block. This has been done so that when an FD_IN pulse is asserted (when RFFD is active), the previous results continue to be output and correctly identified by the RDY signal. If the user asserts FD_IN when RFFD is not active, a partial output block can result.

This affects the punctured code case only; an example is shown in Figure 11.

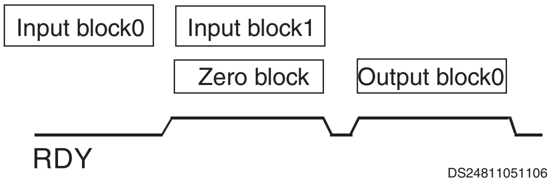


Figure 11: **Note: First RDY Block is Zeros**

Core Resource Requirements and Performance

The area of the core increases with the constraint length and the punctured input and output rates if the core is punctured. Some example configurations are shown in Table 3.

Table 3: **Examples of Non-Punctured Convolution Encoder Implementations on Spartan-6 FPGAs**

	Rate 1/2	Rate 3/4	Rate 5/6	Rate 12/23
Constraint Length	7	7	7	7
Xilinx Part	XC6SLX45T	XC6SLX45T	XC6SLX45T	XC6SLX45T
LUT/FF Pairs	9	41	47	86
Maximum Clock Frequency(1)(2)	370	243	250	230

Notes

1. Area and maximum clock frequencies are provided as a guide. They may vary with new releases of the Xilinx implementation tools.
2. Maximum clock frequencies are shown in MHz for -2 parts. Clock frequency does not take jitter into account and should be de-rated by an amount appropriate to the clock source jitter specification.

Support

Xilinx provides technical support at www.xilinx.com/support for this LogiCORE product when used as described in the product documentation. Xilinx cannot guarantee timing, functionality, or support of product if implemented in devices that are not defined in the documentation, if customized beyond that allowed in the product documentation, or if changes are made to any section of the design labeled *DO NOT MODIFY*.

Refer to the IP Release Notes Guide (XTP025) for further information on this core. On the first page there is a link to "All DSP IP." The relevant core can then be selected from the displayed list.

For each core, there is a master Answer Record that contains the Release Notes and Known Issues list for the core being used. The following information is listed for each version of the core:

- New Features
- Bug Fixes
- Known Issues

Ordering Information

This LogiCORE IP module is included at no additional cost with the Xilinx ISE Design Suite software and is provided under the terms of the Xilinx End User License Agreement. Use the CORE Generator software included with the ISE Design Suite to generate the core. For more information, please visit the core page.

Information about additional Xilinx LogiCORE modules is available at the Xilinx IP Center. For pricing and availability of other Xilinx LogiCORE modules and software, please contact your local Xilinx sales representative.

Revision History

The following table shows the revision history for this document.

Date	Version	Revision
03/28/02	1.0	Initial Revision History added to document.
03/16/04	4.0	Updated to version 4.0 standards.
04/28/05	4.1	Added support for Spartan-3E FPGAs and Xilinx software v7.1i.
01/18/05	5.0	Updated to version 5.0; ISE tools 8.1i.
07/13//06	6.0	Added support for Virtex-5 FPGAs.
05/17/07	6.1	Added support for Spartan-3A DSP FPGAs.
06/24/09	7.0	Removed aclr pin and added support for Spartan-6 and Virtex-6
03/01/11	7.1	Support added for Virtex-7 and Kintex-7. ISE Design Suite 13.1

Notice of Disclaimer

Xilinx is providing this product documentation, hereinafter “Information,” to you “AS IS” with no warranty of any kind, express or implied. Xilinx makes no representation that the Information, or any particular implementation thereof, is free from any claims of infringement. You are responsible for obtaining any rights you may require for any implementation based on the Information. All specifications are subject to change without notice. XILINX EXPRESSLY DISCLAIMS ANY WARRANTY WHATSOEVER WITH RESPECT TO THE ADEQUACY OF THE INFORMATION OR ANY IMPLEMENTATION BASED THEREON, INCLUDING BUT NOT LIMITED TO ANY WARRANTIES OR REPRESENTATIONS THAT THIS IMPLEMENTATION IS FREE FROM CLAIMS OF INFRINGEMENT AND ANY IMPLIED WARRANTIES OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. Except as stated herein, none of the Information may be copied, reproduced, distributed, republished, downloaded, displayed, posted, or transmitted in any form or by any means including, but not limited to, electronic, mechanical, photocopying, recording, or otherwise, without the prior written consent of Xilinx.