LogiCORE IP Viterbi Decoder v7.0

DS247 March 1, 2011 **Product Specification**

Introduction

The Viterbi Decoder is used in many Forward Error Correction (FEC) applications and in systems where data are transmitted and subject to errors before reception. The Viterbi Decoder is compatible with many common standards, such as DVB, 3GPP2, 3GPP LTE, IEEE 802.16, Hiperlan, and Intelsat IESS-308/309.

Features

- High-speed, compact Viterbi Decoder
- Available for Kintex™-7, Virtex®-7, Virtex-6, Virtex-5, Virtex-4, Spartan®-6, Spartan-3/XA, Spartan-3E/XA and Spartan-3A/AN/3A DSP/XA FPGAs
- Fully synchronous design using a single clock
- Parameterizable constraint length from 3 to 9
- Parameterizable convolution codes
- Parameterizable traceback length
- Decoder rates from 1/2 to 1/7
- Very low latency option
- Minimal block RAM requirements; two block RAMs for a constraint length 7 decoder
- Serial architecture for small area
- Soft decision with parameterizable soft width
- Multi-channel decoding
- Dual rate decoder
- Trellis mode
- Erasure for external puncturing
- BER monitor
- Normalization
- Synchronization
- Best state option
- Trellis Initialization options for packet handling
- Direct traceback options for packet handling
- For use with Xilinx CORE Generator™ software and Xilinx System Generator for DSP v13.1
- Compatible encoder core available in the Xilinx CORE Generator software

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/XA, Spartan-3E/XA, 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

This core implements a Viterbi Decoder for decoding convolutionally encoded data. For details of the encoding process, see the *Convolution Encoder (DS248)* data sheet. This is available in Xilinx CORE Generator software. The decoder core consists of two basic architectures: a fully parallel implementation which gives fast data throughput at the expense of silicon area and a serial implementation which occupies a small area but requires a fixed number of clock cycles per decoded result.

Viterbi decoding decodes the data originally input to the convolutional encoder by finding an optimal path through all the possible states of the encoder. For a constraint length 3 encoder, there are four possible states, and for a constraint length 7, there are 64 states. The basic decoder core consists of three main blocks as shown in Figure 1.

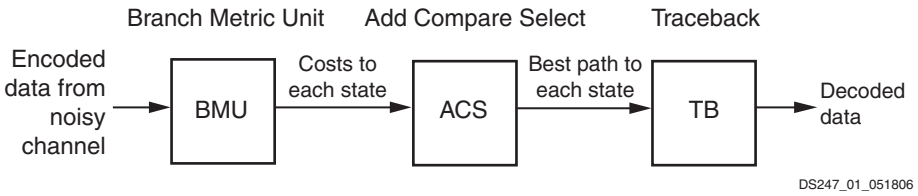

Figure 1: **Viterbi Decoder Block Diagram**

Costing

The first block is the branch-metric-unit (BMU). This module costs the incoming data. For the fully parallel decoder, the incoming data can be hard coded with bit width 1 or soft coded with a parameterizable bit width which can be set to any value from 3 to 8. Hard-coded data is decoded using the Hamming method of decoding, whereas soft data is decoded using an Euclidean metric. In hard coding, the demodulator makes a firm or hard decision on whether a one or zero is transmitted and provides no other information to the decoder on how reliable the decision is. For soft decoding, the demodulator provides the decoder with some side information together with the decision. The extra information provides the decoder with a measure of confidence for the decision. Soft-coded data gives a significantly better BER performance compared with hard-coded data. Soft decision offers approximately a 3 dB increase in coding gain over hard-decision decoding.

The increase of data width for the soft-coded data gives minimal benefit in the BER performance of the core and has a significant cost in terms of chip area; see Performance Characteristics, page 28. Hard coding is available only for the Standard parallel or Multi-Channel Viterbi. Erasure (external puncturing) is not available with hard coding.

There are two available data formats for soft coding: soft signed magnitude and offset binary. See Table 1 for the data formats for the case of soft width 3.

Table 1: **Data Format for Soft Width 3**

	Signed Magnitude	Offset-Binary
Strongest 1	111	111
	110	110
	101	101
Weakest 1	100	100
Weakest 0	000	011
	001	010
	010	001
Strongest 0	011	000

Decoding

The second block in the decoder is the add-compare-select (ACS) unit. This block selects the optimal path to each state in the Viterbi trellis. Figure 2 shows one stage in the Viterbi trellis for a constraint length 3 decoder. The ACS block uses the convolutional codes to extract the correct cost from the BMU for each branch in the trellis.

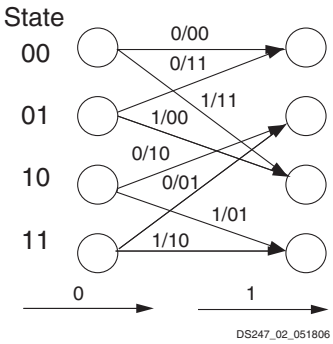

Figure 2: **State Transitions for Constraint Length 3**

The BER performance of the Viterbi algorithm varies greatly with different convolutional code sets; some of the standard convolutional codes are shown in Table 2.

Table 2: **Standard Convolution Codes**

Constraint Length	Output Rate = 2		Output Rate = 3	
	binary	octal	binary	octal
7	1111001 1011011	171 133	1001111 1010111 1101101	117 127 155
9	101110001 111101011	561 753	101101111 110110011 111001001	557 663 711

The ACS module decodes for each state in the trellis. Thus, for a constraint length 7 decoder which has 64 states, there are 64 sub-blocks in the ACS block.

If the core is implemented in serial mode, the amount of silicon required for each sub-block is only 2.5 slices. A decoder of constraint length 7 can be implemented on a small Spartan device, for example the XC3S100E. See Performance Characteristics, page 28 for further characterization of the decoder.

Traceback

The final block in the decoder is the traceback block. The actual decoding of symbols into the original data is accomplished by tracing the maximum likelihood path backwards through the trellis. Up to a limit, a longer sequence of tracing results in a more accurate path through the trellis. After a number of symbols equal to at least six times the constraint length, the decoded data is output. The traceback starts from zero or best state; the best state is estimated from the ACS costs. The traceback length is the number of trellis states processed before the decoder makes a decision on a bit. The decoded data is output only after a traceback length number of bits has been traced through. In other words, the traceback length determines the length of the training sequence for the Viterbi Decoder.

The length of the traceback is parameterizable and can be set to any value between 12 and 128. For the reduced latency option, the traceback length can only be a multiple of 6 between 12 and 126. The recommended value for

non-punctured decoding is at least 6 times the constraint length. For data that has been punctured, that is, symbols removed prior to transmission over the channel, a larger value traceback length is required; usually, it is at least 12 times the constraint length to obtain optimal BER performance.

A best state option is available to select the starting location for the traceback from the state with minimal cost. There is also a reduced latency option which reduces the latency on the core by approximately half. The latency is of the order of **two times the traceback length** for reduced latency; see Latency, page 27. The reduced latency option has a slight speed penalty and is not available with the multi-channel or serial core. The traceback block is implemented in block RAM, and the larger the value of the traceback length, the greater the block RAM requirements. See Core Resource Utilization, page 27 for additional details.

Multi-Channel Decoder

The multi-channel decoder decodes many interlaced channels using a single Viterbi Decoder. The input to the multi-channel decoder is interlaced encoded data on each DATA_IN bus. For a channel count of 3, channel 1 data is input followed by channel 2 and then channel 3 in a repeating sequence. The output is interlaced decoded data on the DATA_OUT pin. See Figure 3. The multi-channel decoder can decode from 2 to 32 channels. The larger the number of channels, the greater the block RAM requirements, as each channel requires its own traceback.

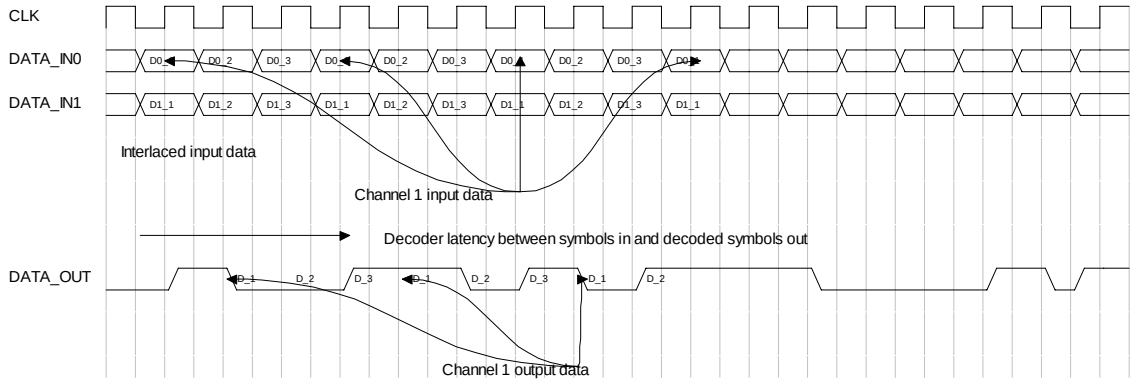

Figure 3: **Multi-Channel Decoder with Channel Count 3**

The BER from the multi-channel decoder, if selected, gives the average number of errors present over all the input channels.

The multi-channel decoder can decode at high speeds, but the true output rate of the decoder is equal to the speed divided by the number of channels. See Performance Characteristics, page 28 for characterization results on the multi-channel decoder.

Trellis Mode Decoder

Pragmatic Trellis Coded Modulation (PTCM) uses a standard rate 1/2 decoder to decode data. The data is costed externally to the decoder. In PTCM, the received symbol or phase angle is converted to four branch metrics and a sector number externally to the decoder using a lookup table. If the width of the generated costs is less than the soft width+1 then the data should be tied to the lower bits and the remaining TCM input bits tied to zero. The sector number identifies the part of the I-Q plane where the symbol was received. The branch metrics are then processed by the Trellis-Mode decoder. The sector number is delayed by the Viterbi latency in the Trellis-Mode decoder. The width of the TCM buses is always equal to the soft width of the decoder plus one. The decoded data is used with the delayed sector number to estimate the uncoded symbols.

Dual Rate Decoder

For a given constraint length and traceback-length, the core can function as a dual decoder, that is, two sets of convolutional codes and output rates can be used internally to the decoder. The dual-decoder offers significant chip area savings when two different decoders with the same constraint length are required. For example, as a constraint length 7 decoder the core can decode as a rate 1/2 decoder and a rate 1/3 decoder. The implementation requires only a little additional logic for the extra costing involved in the BMU and some mux'ing in the ACS unit (see Figure 1). The dual decoder can be implemented as either parallel or serial architecture, and erasure pins can be present on the input. The selection of the decoder rate and codes is through the SEL pin (see Figure 4). When the SEL pin is low, output rate0 and convolution0_codes are used in the decoding. When the SEL pin is high, then the rate is 1/output_rate1, and the convolution1_codes are used to decode the incoming data. The SEL_O pin shows the decoded data corresponding to the original SEL set of data points.The number of input data buses is equal to the max of the two output rates.

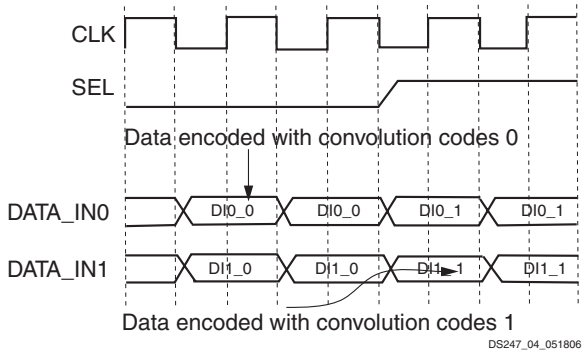

Figure 4: **SEL Input for Dual Decoder**

Erasure

If the data has been punctured prior to transmission, then depuncturing is carried out externally to the Viterbi Decoder (see Figures 5 and 6). The presence of null-symbols (that is, symbols which have been deleted prior to transmission across the channel) is indicated using the erasure input ERASE. The decoder functions exactly as a non-punctured Viterbi Decoder, except the corresponding DATA_IN inputs are ignored when the erased input bit is high. If ERASE(0) is high, then the input on DATA_IN0 is viewed as a null-symbol. If ERASE(1) is high, then the input on DATA_IN1 is viewed as a null symbol, etc. Although the normal usage of erasure is with a rate 1/2 decoder, the erasure pins can be present for output rates greater than 2. Erasure can be used with the Standard, Multi-Channel and Dual Decoder. Erasure cannot be present on the Trellis-Mode decoder.

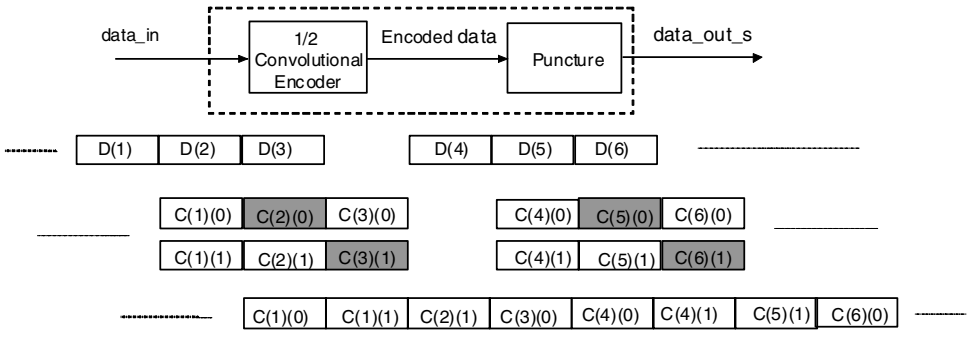

Figure 5: **Puncturing Encoded Data with 3/4 Puncture Rate with Single-Channel Output**

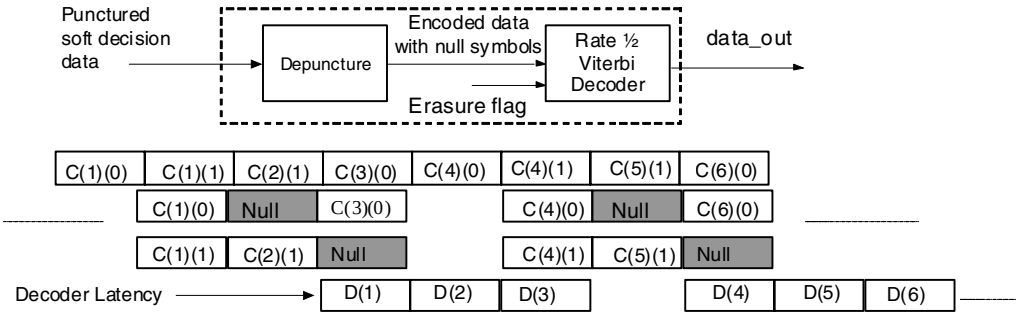

Figure 6: **Depuncturing Rate 3/4 Punctured Data with Single-Channel Soft Input and Erasure**

BER

The bit-error-rate (BER) option on the decoder monitors the error rate on the transmission channel. Decoded data from the Viterbi Decoder is re-encoded using the convolutional encoder and compared with a delayed version of the data input to the decoder. An error is indicated if the delayed and encoded data differ (see Figure 7). The count is incremented on a symbol-by-symbol basis. For example, if both the I & Q outputs differ from the expected I and Q for a rate 1/2 decoder, then this is only considered as one error on the BER count. The two sets of symbols can differ if there is an error on the channel or if the Viterbi Decoder has decoded incorrectly. The probability of the decoder incorrectly decoding is significantly smaller than the probability of a channel bit error; therefore the BER output gives a good estimate of the errors on the channel.

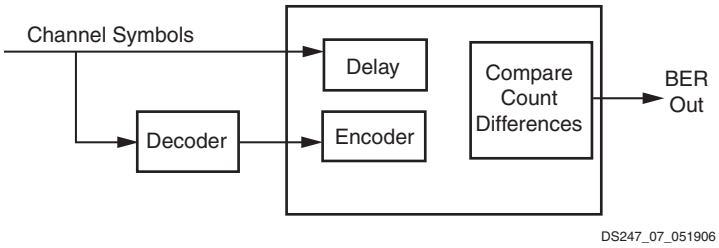

Figure 7: **Bit Error Rate Calculation**

The Number of BER symbols determines the number of input symbols over which the error count takes place. The output error count BER is the number of errors that has been counted during the Number of BER symbols count. Figure 8 shows the output BER_DONE which indicates that BER symbol count input symbols have been processed and that a new bit error rate value is present on BER. Note that the BER output always has a width of 16; therefore the maximum number of errors that can be counted is (2^{16}-1). If more errors occur than this upper limit, the maximum number of errors is output, that is, BER is set to all 1s.

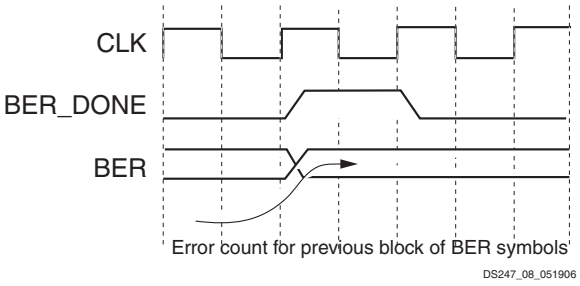

Figure 8: **BER_DONE with BER**

Normalization

The NORM signal is an optional output that gives immediate monitoring of the errors on the channel. As the metrics grow in the ACS unit, they must be normalized to avoid overflow. When normalization occurs the decoder subtracts a fixed value from all metrics and asserts the normalization signal. If the ACS unit requires normalization, then there are uncertainties or errors on the channel. The more frequent the normalization, the higher the rate of errors present. The actual frequency of the normalization depends on many factors, in particular the soft width and the output rate of the decoder.

The normalization signal can be used to detect Viterbi synchronization. A high normalization rate (exceeding a certain threshold) indicates loss of synchronization. A low normalization rate (less than a certain threshold) indicates Viterbi synchronization has been achieved. In general, the normalization rate is inversely proportional to the Signal-to-Noise Ratio (SNR) at the decoder input.

Synchronization

The decoder provides a method for monitoring the synchronization status of the core. The method involves the analysis of both the normalization output from the ACS modules within the core and the BER performance of the core. A complete description of the method used is provided in the Xilinx design brief, *Viterbi Synchronization (DS205 v1.0)*.

The Normalization rate by itself gives an indication of the Viterbi Decoder synchronization status. A high normalization rate, exceeding a predetermined threshold, implies a loss of synchronization. Similarly, the BER rate of the core, by itself, can indicate a loss of synchronization. Thus, if the BER rate, or the normalization rate, was used in isolation, the threshold required would vary with the noise on the channel. The BER rate, like the normalization rate, would therefore be dependent on the signal to noise ratio Eb/No. The core uses both the BER rate and the normalization rate to achieve a synchronization method that is independent of Eb/No.

Synchronization requires two thresholds – the normalization threshold and the BER threshold. These thresholds can be fixed within the core or varied dynamically through the NORM_THRESH and BER_THRESH inputs. The values of these thresholds depend on the Viterbi core and vary with the generics. Example thresholds found through simulation with an AWGN channel are given in Table 3.

Table 3: **Example Synchronization Thresholds for Various Constraint Length 7 Decoder**

Viterbi Type	Normalization Threshold	BER Threshold
Standard Parallel	250	600
Dual Parallel/Serial	15	600
Standard Serial	15	600

If the BER threshold is exceeded, then the signal OUT_OF_SYNC is asserted (high) and this indicates loss of synchronization. If the Norm threshold is exceeded, then the OUT_OF_SIGNAL is deasserted (low) indicating the decoder is synchronized. The internal counters are reset when any threshold is exceeded. In case of very low noise on the channel and synchronized input data, there is hardly any normalization within the ACS unit. Synchronization monitors for this situation and an internal counter deasserts the OUT_OF_SYNC flag after a fixed number of input symbols.

To determine threshold levels and monitor the behavior of the synchronization module, an out-of-synchronization flag is provided. The OOS_FLAG[2:0] indicates which threshold has been exceeded.

- OOS_FLAG[0] indicates the BER threshold has been exceeded. In this case, OUT_OF_SYNC is asserted high.

- OOS_FLAG[1] indicates the NORM threshold has been exceeded. In this case, OUT_OF_SYNC is deasserted low.
- OOS_FLAG[2] indicates that the internal data_count has been exceeded without either BER or NORM thresholds being exceeded, the core is in a very low noise state, and the OUF_OF_SYNC is deasserted low.

The synchronization option is not available for the multi-channel and trellis mode decoders.

Packet Handling

Viterbi decoding is a continuous operation, but the input to the decoder can be packet based rather than continuous streams. For data encoded in packets, it is necessary to terminate the encoder between the packets by the insertion of what is called zero tail bits. For a constraint length 7 decoder, there are 6 zero bits inserted into the encoder at the end of the packet. For a general Viterbi (constraint length -1), tail bits are required to return the encoder back to state zero. The effect of these zero tail bits is to return the Viterbi trellis to zero state and also the next packet starts from state zero. The Viterbi handles tail bits when working in any of the basic modes; no additional signals or control circuitry is required.

Although zero-tail bits or zero-tail termination is the standard method for handling packets within the Viterbi Decoder, there is a rate loss on the channel caused by constraint length -1 information bits being added to the original message. If the original packet contained m bits, the output code words are of length m + K -1, where K is the constraint length. Thus, the effective rate on the channel becomes:

$$\text{RateNew} = Rate \times \left(1 - \frac{K-1}{m+K-1}\right)$$

For large packets, the rate loss becomes insignificant. For smaller packets, the method of tail-biting avoids the problem of the fractional rate loss by letting the last K-1 information bits define the starting state of the encoder. Only the data is encoded, that is, exactly m encoded bits are produced. In this case, no rate loss occurs and the encoding always starts and ends in the same state, but not necessarily the zero state. For a full description of the Viterbi Decoder and trellis termination and tail-biting, see the Xilinx application note (*XAPP551*) *Viterbi Decoder Block Decoding - Trellis Termination and Tail-Biting*.

To handle different forms of trellis termination and trellis initialization, there are various options available in the Viterbi Decoder. See Figure 17.

Trellis Initialization

The trellis initialization options allow the user to specify the starting state of the received packet. When the PACKET_START signal is asserted, the costs in the ACS modules can be initialized to one of three options. Either state zero is the starting state (State Zero), or all the states are costed equally (Equal States), or the starting state is set to a dynamic input state (User Input). See Figure 9 for the user input state case.

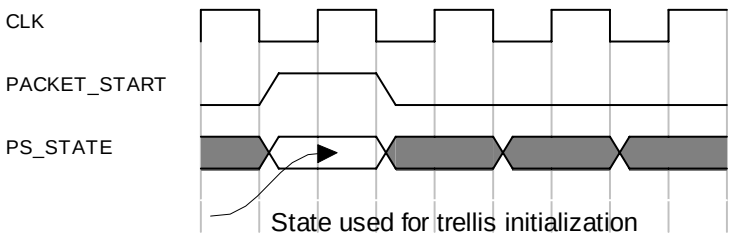

Figure 9: **Trellis Initialization with User Input PS_STATE**

Direct Traceback

The direct traceback options allow the user to specify the handling of the traceback and the end state of the packet. The data to be traced directly is marked by the TB_BLOCK signal. This signal must remain high for the number of encoded bits to be traced directly. The length of the TB_BLOCK signal cannot exceed the Maximum Traceback length. The TB_BLOCK signal normally marks the encoded bits at the end of a packet. See Figure 21 for an example of the TB_BLOCK signal and the output signals produced.

When any of the direct traceback options are selected, the decoder outputs four additional signals in addition to the DATA_OUT signal. The DATA_OUT_REVERSE signal is output immediately from the direct traceback process and is the decoded signal in reverse order without any training sequence. The decoder either starts decoding directly from state zero, the best state (determined by the ACS costs), or a user input state. See Figure 10 for the user input case.

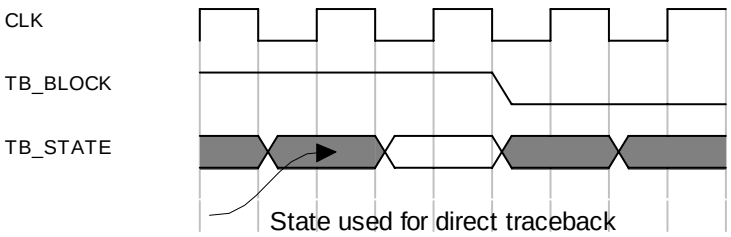

Figure 10: **Direct Traceback with User Input TB_STATE**

When decoding, the data is output in reverse order moving back through the trellis. This data is normally reversed by passing the data through a LIFO. The LIFO introduces an extra traceback length of latency to the decoding process. By outputting the data directly in reverse order without the LIFO, the decoded data is available with minimal latency. If the traceback blocks are of different lengths, then overlap can occur on the reverse output, unless the relationship K < M + N is preserved; where K is the length of the first TB_BLOCK, N is the length of the second block, and M is the separation between the blocks, as shown in Figure 11.

The DATA_OUT_DIRECT is the DATA_OUT_REVERSE signal, but now in the correct order. Both the reversed and direct data are marked with their corresponding RDY signals, REVERSE_RDY and DIRECT_RDY.

If any of the direct traceback options are selected, then the DATA_OUT_DIRECT signal is mux'ed into the output data DATA_OUT and the data is marked with the TB_BLOCK_O signal. See Figure 21.

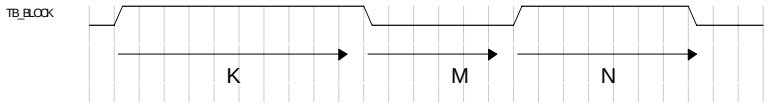

Figure 11: **Separation Between Traceback Blocks**

Core Pinout

A representative symbol of the Viterbi Decoder, with the signal names, is shown in Figure 12 and described in Table 4. 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. Timing diagrams for the signals are shown in Control Signal Operation, page 25.

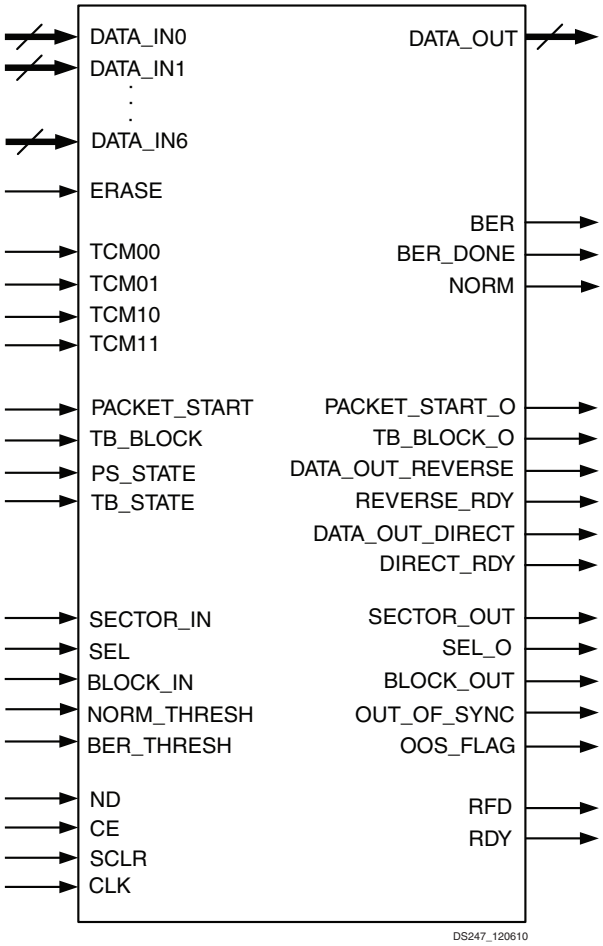

Figure 12: **Core Schematic Symbol**

Table 4: **Core Signal Pinout**

Signal	Direction	Description
DATA_IN0 DATA_IN1 DATA_IN2 DATA_IN3 DATA_IN4 DATA_IN5 DATA_IN6	Input	**Data In**: One for each output of the encoder. The width of each bus is 1 for hard-decision decoding or can be width 3 to 8 for soft-decision decoding.
ERASE (erasure only, optional)	Input	**Erasure**: Indicates if high that the corresponding DATA_IN is a null-symbol. ERASE(0) corresponds to DATA_IN0, ERASE(1) corresponds to DATA_IN1, etc.
TCM00 TCM01 TCM10 TCM11 (Trellis-Mode only, optional)	Input	**Trellis Coding Mode**: Input data for Trellis-Mode decoding, replacing DATA_IN. The width of each bus is equal to the soft width +1.
PACKET_START (Optional)	Input	**Packet Start**: Indicates the start of a packet for trellis initialization.
TB_BLOCK (Optional)	Input	**Traceback Block**: Indicates the block of input data at the end of a packet for direct traceback.

Table 4: **Core Signal Pinout *(Cont'd)***

Signal	Direction	Description
PS_STATE (Optional)	Input	**Packet start state:** Dynamically variable input state for trellis initialization.
TB_STATE (Optional)	Input	**Traceback state:** Dynamically variable state for direct traceback.
BLOCK_IN	Input	**Block In**: Marker for a packet of input data to the decoder.
SECTOR_IN (Trellis-Mode only, optional)	Input	**Sector In**: Sector data for Trellis-Mode decoding. The SECTOR_IN bus is of a fixed width of four bits.
NORM_THRESH (Optional)	Input	**Normalization Threshold:** Dynamic threshold for synchronization. The bus is of a fixed width of 16 bits.
BER_THRESH (Optional)	Input	**Bit Error Rate Threshold:** Dynamic threshold for synchronization. The bus is of a fixed width of 16 bits.
SEL (dual decoder only, optional)	Input	**Select**: Select pin for dual decoding. Indicates if high that the input data is to be decoded with the second set of convolutional codes.
ND (Serial only)	Input	**New Data**: Indicates new data on DATA_IN. Only present for the serial decoder.
CE (Optional)	Input	**Clock Enable**: Freezes state of core when low.
SCLR (Optional)	Input	**Synchronous Reset**: Reinitializes core control logic if core is enabled.
CLK	Input	**Clock**: Clock input. All core operation is synchronous with the CLK input.
DATA_OUT	Output	**Data Out**: Output decoded data.
BER (Optional)	Output	**Bit Error Rate**: Bit error rate monitor of fixed bus width16.
BER_DONE (Optional)	Output	**Bit Error Rate Done:** Indicates that the fixed number of BER symbol inputs has been received.
NORM (Optional)	Output	**Normalization**: Indicates when normalization has taken place internal to the ACS module.
PACKET_START_O (Optional)	Output	**Packet start output:** Marks the start of the output packet when trellis initialization is present.
TB_BLOCK_O (Optional)	Output	**Traceback block out:** Marks output data corresponding to the original traceback block.
DATA_OUT_REVERSE (Optional)	Output	**Data out reverse:** Reversed decoded data corresponding to the input traceback block.
REVERSE_RDY (Optional)	Output	**Reverse Ready**: Indicates valid data on output port DATA_OUT_REVERSE.
DATA_OUT_DIRECT (Optional)	Output	**Data out direct:** Direct traceback data corresponding to the input traceback block.
DIRECT_RDY (Optional)	Output	**Direct Ready**: Indicates valid data on output port DATA_OUT_DIRECT.
BLOCK_OUT (Optional)	Output	**Block Out:** Marks the output of decoded data corresponding to the input block marked by BLOCK_IN.
SECTOR_OUT (Only present for Trellis-Mode decoder)	Output	**Sector Out**: SECTOR_IN delayed by decoding delay for Trellis-Mode decoding. The SECTOR_OUT bus is a fixed width of four bits.
SEL_O (Only present for Dual-Decoder)	Output	**SEL_O**: SEL delayed by decoding delay for Dual-Rate decoding.

Table 4: **Core Signal Pinout *(Cont'd)***

Signal	Direction	Description
OUT_OF_SYNC (Optional)	Output	**Out of Synchronization:** Indicates when high that out the core input data is not synchronized.
OOS_FLAG (Optional)	Output	**Out of synchronization flag**: Indicates the synchronization status of the core relative to the synchronization thresholds.
RDY (Optional)	Output	**Ready**: Indicates valid data on output port DATA_OUT. Mandatory for the serial case.
RFD (Only present for serial decoder)	Output	**Ready for Data**: Indicates that the core is ready to receive new data. Mandatory for the serial decoder, not present otherwise.

DATA_IN0, DATA_IN1... DATA_IN6 Inputs

These are the bus inputs to be decoded. The encoded bits on each of the DATA_IN inputs can be hard coded (bus width 1) or soft coded (bus width 3 to 8). If the dual decoder is selected, the number of input symbols is equal to the maximum output rate.

ERASE Input

This erase input bus is optional and is only required where data on the channel has been punctured. The bus is present if the external puncturing option is selected. The inputs are used to indicate the presence of a null-symbol on the corresponding DATA_IN buses. ERASE(0) corresponds to DATA_IN0, ERASE(1) corresponds to DATA_IN1, etc. If an erase pin is high, the data on the corresponding DATA_IN bus is treated as a null-symbol internally to the decoder.

TCM00, TCM01... TCM11 Inputs

These bus inputs are only available if the Trellis-Mode decoder is selected The bus inputs replace the DATA_IN inputs to the decoder. The width of the Trellis-Mode inputs can range from 4 to 9 corresponding to a data width of 3 to 8. The Trellis-Mode inputs are the outputs from an external costing of the data. There are always four inputs to the decoder, and the decoder always functions as a rate 1/2 decoder when Trellis-Mode is selected.

SECTOR_IN Input

This bus input is only available if the Trellis-Mode decoder is selected. The SECTOR_IN bus has width 4. The SECTOR_IN input is delayed by the decoder delay and output to the SECTOR_OUT bus. See Trellis Mode Decoder, page 4.

SEL Input

The SEL pin is only available if the dual decoder has been selected. When SEL is low, the input data is decoded using the first set of convolutional codes. When it is high, the second set of convolutional codes is applied. See Figure 4.

BLOCK_IN Input

The BLOCK_IN pin is delayed by the latency of the decoder and output as BLOCK_OUT. The BLOCK_OUT pin shows the decoded data corresponding to the original BLOCK_IN set of data points.

PACKET_START Input

When PACKET_START is asserted, the trellis in the Viterbi Decoder is initialized to start from state zero, equal states, or from the dynamically variable input PS_STATE.

PS_STATE Input

The PS_STATE bus (width constraint length - 1) is the dynamically variable state for the initialization of the trellis. The bus is only available if the trellis initialization option with user input is selected.

TB_BLOCK Input

The TB_BLOCK signal marks the block of input data to be traced back directly in the decoder. This signal is only available if the direct traceback option is selected.

TB_STATE Input

The TB_STATE bus (width constraint length - 1) is the dynamically variable state for the initialization of the direct traceback. The bus is only available if the direct traceback option with user input is selected.

NORM_THRESH Input

The NORM_THRESH bus (fixed width 16) is the dynamically variable norm threshold for the synchronization of the core. If the threshold value is reached by the normalization counter prior to the bit error threshold, then the Viterbi is in-sync and the OUT_OF_SYNC pin is deasserted and OOS_FLAG(1) is asserted. This bus is only available if synchronization is selected with dynamic thresholds.

BER_THRESH Input

The BER_THRESH bus (fixed width 16) is the dynamically variable bit error rate threshold for the synchronization of the core. If the bit error threshold is reached by the bit error counter prior to the normalization threshold, then the Viterbi is out of sync and the OUT_OF_SYNC pin is asserted. See Synchronization, page 7. This bus is only available if synchronization is selected with dynamic thresholds.

ND Input

When the New Data (ND) input is sampled logic-high, it signals that a new symbol on DATA_IN should be sampled on the same rising clock edge. ND is only present for the serial decoder. ND differs from CE in that the core is not frozen when ND is low. In the serial case, the core processes the new sample as far as possible. Like all the synchronous inputs, ND is ignored if CE is low. ND is only valid if the Ready For Data (RFD) signal is high and is ignored if the RFD signal is low. See Control Signal Operation, page 25 for more details.

CE Input

The Clock Enable (CE) input is an optional input pin. When CE is deasserted (low), all the synchronous inputs (see Figure 13) are ignored and the core remains in its current state. The state of the core is frozen while CE is low.

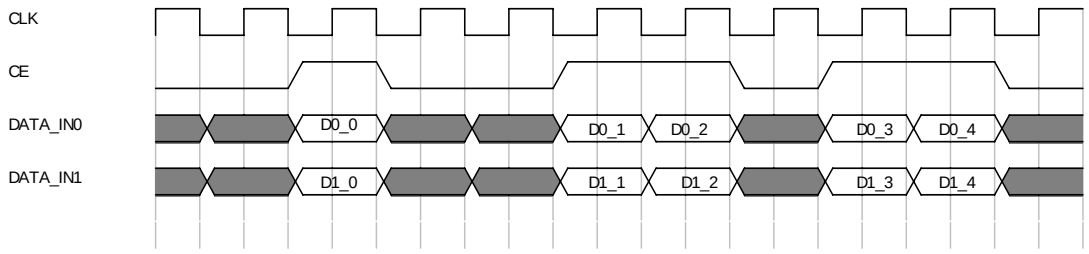

Figure 13: **Data input with CE**

SCLR Input

When SCLR is asserted (high), all the core flip-flops are synchronously initialized if the core is enabled. All the synchronous inputs are ignored and the core remains in its current state if there is a CE present and CE is low. The core remains in this state until SCLR is deasserted and CE, if present, is high. Note that the block RAM is not cleared with the SCLR signal and there is a block of previously decoded data output from the traceback prior to correct decoding resuming. The RDY signal is only asserted when valid data is available on the DATA_OUT pin. SCLR is an optional pin, as the core can function correctly without it.

DATA_OUT Output

This is the decoded data output. If Ready (RDY) is present, then the decoded output is only valid when RDY is high.

BER Output

The Bit Error Rate (BER) bus output (fixed width 16) gives a measurement of the channel bit error rate by counting the difference between the re-encoded DATA_OUT and the delayed DATA_IN to the decoder. For a full description of BER, see BER, page 6. Trellis-mode does not support a BER output.

BER_DONE Output

The BER measurement is carried out over a fixed number of input symbols, determined by the BER_RATE parameter. BER_DONE output indicates when BER_RATE inputs have been processed. The BER_DONE signal goes high for one symbol cycle. See Figure 8. See BER, page 6 for additional details.

NORM Output

The NORM output indicates when normalization has occurred within the core. It gives an immediate indication of the rate of errors in the channel. See Normalization, page 7 for additional details.

SECTOR_OUT Output

The SECTOR_OUT is a delayed version of the SECTOR_IN bus. Both buses have a fixed width of 4 bits. The delay equals the delay through the Trellis-Mode decoder.

SEL_O Output

SEL_O signal is a delayed version of the SEL signal. The delay equals the delay through the Dual decoder.

BLOCK_OUT Output

BLOCK_OUT pin is a delayed version of the BLOCK_IN signal. The BLOCK_OUT signal shows the decoded data corresponding to the original BLOCK_IN set of data points. The delay equals the delay through the decoder.

PACKET_START_O Output

The PACKET_START_O is a delayed version of the PACKET_START signal. The delay equals the delay through the standard parallel decoder.

TB_BLOCK_O Output

The TB_BLOCK_O is a delayed version of the TB_BLOCK signal. The delay equals the delay through the standard parallel decoder.

DATA_OUT_REVERSE Output

If one of the Direct Traceback options is selected, then the DATA_OUT_REVERSE signal is present on the decoder. The DATA_OUT_REVERSE signal is the decoded signal output directly from the traceback. Traceback in direct output does not involve any training, and the data is output immediately in reverse order. The reverse order avoids the latency of a LIFO and allows immediate access to the decoded data if required.

The traceback is started from one of three possible options selectable in the GUI. If State Zero is selected, then the traceback is started from state zero. For the decoder to correctly decode the data in the original packet, the data is terminated in the encoder with constraint length -1 zero tail bits. The second option for traceback is from a dynamically variable input state, TB_STATE. If the packet has been terminated in a special state in the encoder, then this state can be used in the decoder to start the traceback at the correct location, and the reversed data can be read immediately. The final option is to start the traceback from the best state. The best state is determined internally to the Viterbi Decoder from the ACS unit; this is the state with the minimal cost. Again, the data is output immediately in reverse order from the decoder. The length of data to be traced back in the direct traceback option can be any value between 10 and maximum direct.

REVERSE_RDY Output

The REVERSE_RDY signal indicates when the DATA_OUT_REVERSE data is valid, as shown in Figure 21.

DATA_OUT_DIRECT Output

If one of the Direct Traceback options is selected, then the DATA_OUT_DIRECT signal is present on the decoder. The DATA_OUT_DIRECT signal is the output from the direct traceback in the correct order. The output is generated by passing the DATA_OUT_REVERSE signal through a LIFO to give the correctly decoded data.

DIRECT_RDY Output

The DIRECT_RDY output indicates when the DATA_OUT_DIRECT signal is valid, as shown in Figure 21.

OUT_OF_SYNC Output

The OUT_OF_SYNC output indicates when the input data is out of synchronization. Using a comparison between internal normalization, bit error counts, and thresholds passed into the core either as parameters or dynamically, the synchronization of the core is evaluated. The OUT_OF_SYNC is asserted when the incoming data is not

synchronized. The pin is only present if synchronization is selected. Synchronization is not available for the multi-channel and trellis-mode cores.

OOS_FLAG Output

When the data is monitored for synchronization purposes, the OOS_FLAG output monitors which threshold has been exceeded. OOS_FLAG[0] indicates when the BER threshold has been exceeded, OOS_FLAG[1] indicates the normalization threshold has been exceeded, and OOS_FLAG[2] indicates that the data is synchronized with very low noise.

RDY Output

The Ready (RDY) output indicates valid data on DATA_OUT. This output is mandatory in the serial case.

RFD Output

Ready for Data (RFD) indicates the core is ready to sample new data on the DATA_IN buses. The pin is only available for the serial core. See Serial Decoder, page 26 for a full description of the serial core and the required control signals. If ND is asserted while RFD is low, the ND signal is ignored. Note that the core does not sample new data during synchronous resets, even though RFD is high.

CORE Generator Parameters

Figure 14 illustrates the main CORE Generator Viterbi Decoder screen. To generate a core, click Generate.

Screen 1 (Viterbi Type)

Figure 14 shows the Viterbi Decoder Screen 1. The parameter descriptions for this screen follow.

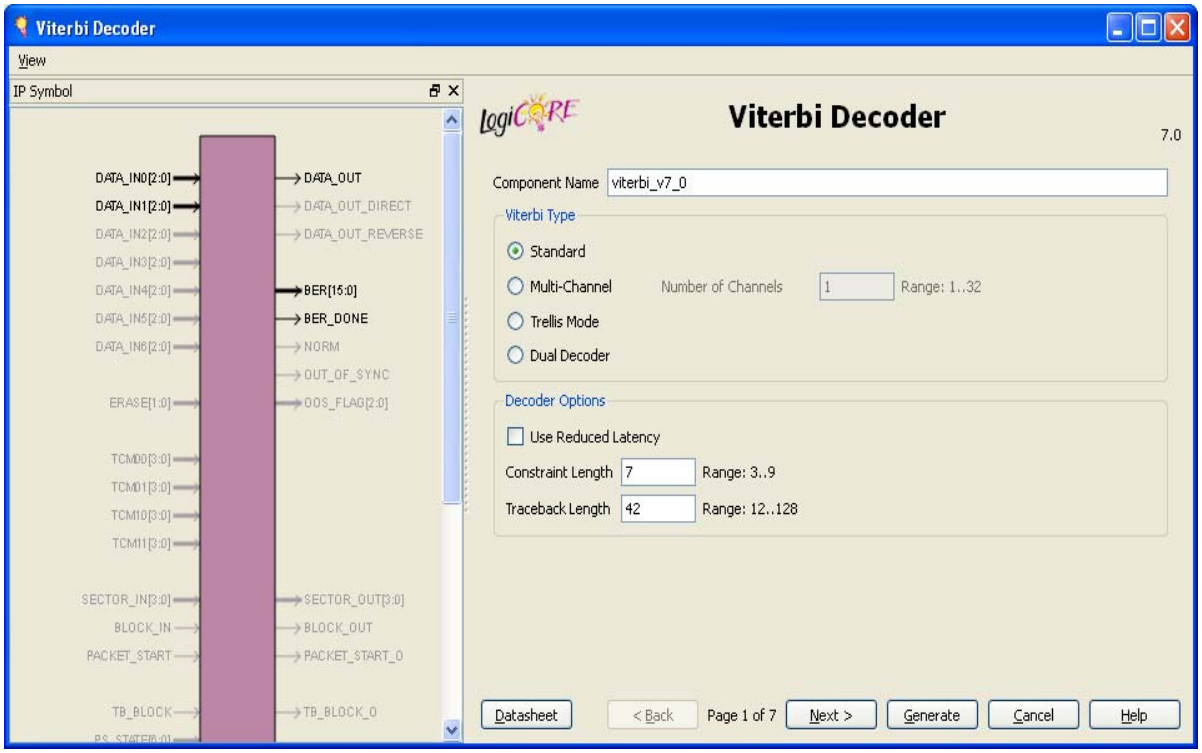

Figure 14: **Viterbi Decoder Screen 1 (Viterbi Type)**

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 of the following characters: a to z, 0 to 9, and "_".

Viterbi Type

There are four different types of Viterbi Decoders which can be selected:

- **Standard**: This type is the basic Viterbi Decoder.
- **Multi-Channel**: This type allows many interlaced channels of data to be decoded using a single Viterbi Decoder. The number of channels to be decoded can be any value between 2 and 32. See Multi-Channel Decoder, page 4.
- **Trellis Mode**: This type is a trellis mode decoder using the TCM and SECTOR_IN inputs. See Trellis Mode Decoder, page 4.
- **Dual Decoder**: The type is the dual decoder that can be operated in dual mode with two sets of convolutional codes. The SEL pin is present when the decoder operates in this mode. See Dual Rate Decoder, page 5.

Decoder Options

- **Constraint Length**: This is the length of the constraint register in the encoder plus 1. This value can be any integer in the range 3 to 9 inclusive.
- **Traceback Length**: This is the length of the survivor or training sequence in the traceback through the Viterbi trellis. Optimal length for the traceback is considered to be at least 6 times the constraint length for non-punctured data. For the multi-channel Viterbi, the traceback length is the length of the traceback for each channel in the decoder. For the reduced latency option, the traceback length must always be divisible by 6. For punctured data, the length should be at least 12 times the constraint length. Increasing traceback length may increase RAM requirements. See Block RAM Utilization, page 27 for more details. Increasing traceback length also increases the latency of the core. See Latency, page 27 for additional details.

- **Use Reduced Latency**: This option reduces the latency on the core by approximately half. The reduced latency option has a slight speed penalty and is not available with the parallel speed optimized or the multi-channel Viterbi. See Latency, page 27 section for additional details.

Screen 2 (Architecture and Data Format)

- See Figure 15. The parameter descriptions for this screen follow.

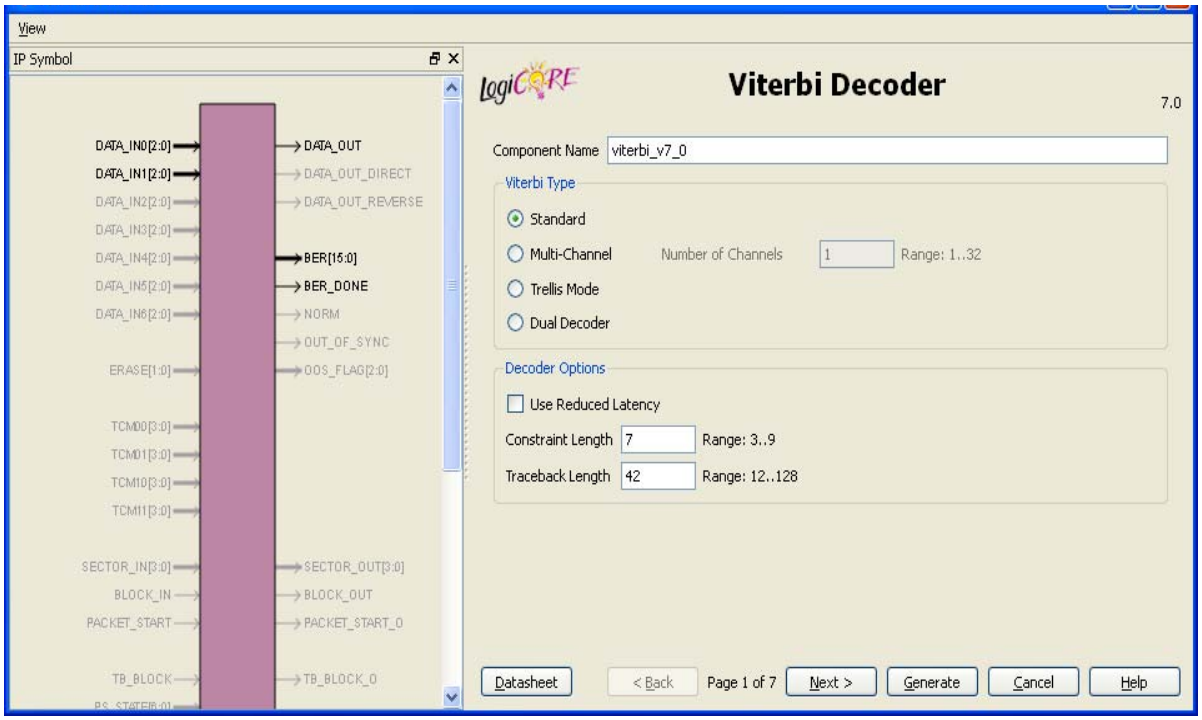

Figure 15: **Viterbi Decoder Screen 2 (Architecture, Best State, and Data Format)**

Architecture

- **Parallel**: Large but fast Viterbi Decoder.
- **Serial**: Small but serial processing of the input data. The number of clock cycles needed to process each set of input symbols depends on the output rate and the soft width of the data. See Serial Decoder, page 26 for more information. See Table 5 for the minimum number of clock cycles required for each set of input symbols.

Table 5: **Minimum Required Clock Cycles Per Input Symbol Set for a Rate 1/2 Serial Decoder**

Soft Width	Minimum Clock Cycles
3	11
4	12
5	13
6	14
7	15
8	16

Best State

The best state option starts the traceback of the core from the optimal state.

- **Use Best State**: The best state selection gives improved BER performance for highly punctured data.

- **Best State Width**: The best state selects the best state from the costs for each state. Most of the lower bits in the cost are redundant in the cost comparison, and the best state area requirements can be reduced by selecting a smaller width than the full acs width. If the width is set to 6, then the full cost is used in the best state selection for a soft width of 3. If the width is set to 3, then the lower three bits are ignored in the best state calculations.

Coding

There are two types of coding available: Soft Coding and Hard Coding.

- **Soft Coding**: Uses the Euclidean metric to cost the incoming data against the branches of the Viterbi trellis.
- **Hard Coding**: Uses the Hamming difference between the input data bits and the branches of the Viterbi trellis. Hard coding is only available for the standard parallel core.

Soft Width

The input width of soft-coded data can be anything in the range 3 to 8. Larger widths require more logic. If the core is implemented in serial mode, larger soft widths also increase the serial processing time. See Table 5 for the minimum number of clock cycles required for a rate 1/2 decoder in the serial case.

Data Format

There are two data formats available for Soft Coding: Signed Magnitude and Offset Binary (see Table 1).

Screen 3 (Output Rate and Convolution Code)

See Figure 16. The parameter descriptions for this screen follow.

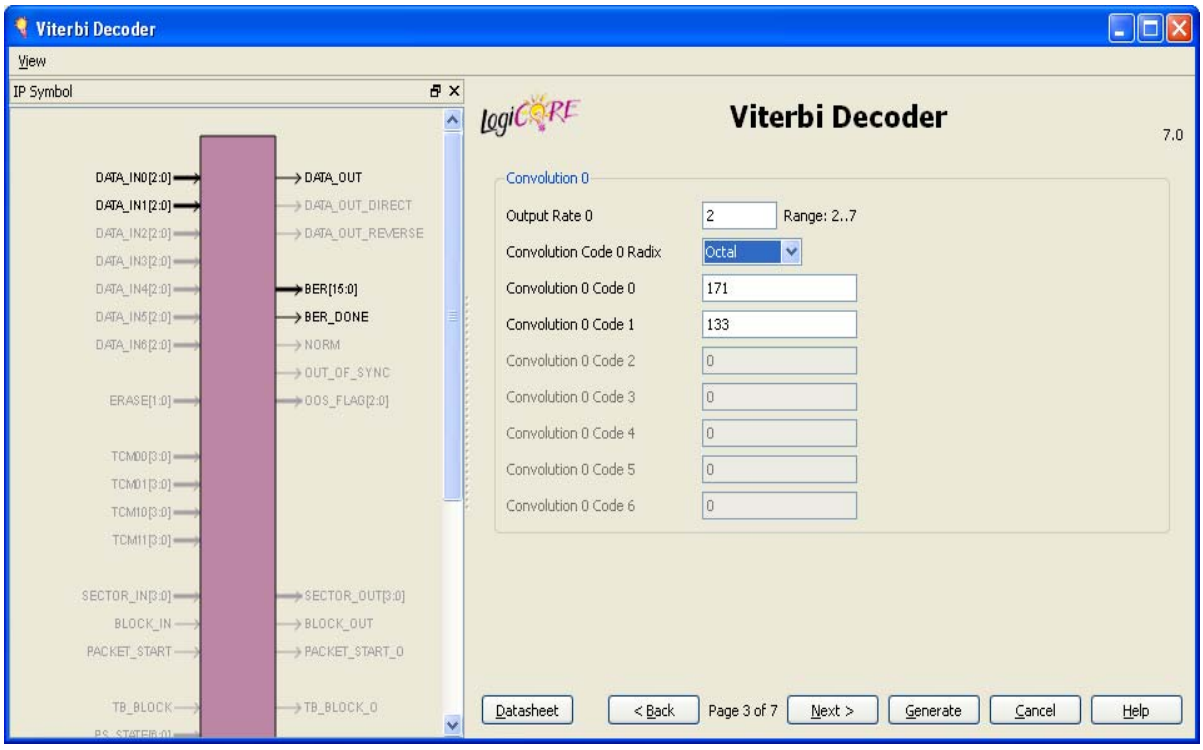

Figure 16: **Viterbi Decoder Screen 3 (Output Rate and Convolution Codes)**

Convolution Code 0 Radix

The convolutional codes can be input and viewed in binary, octal, or decimal.

Output Rate0

Output Rate is the symbol output rate at the Encoder. Output Rate0 can be any value from 2 to 7. Output Rate0 is the output rate used if the decoder is non-dual. If the decoder is dual, then Output Rate0 is the first output rate and the rate used by the decoder when the SEL input is low.

Convolution0 Codes

These codes are the convolutional codes used in the encoder. The codes can be entered (and viewed) in binary, octal, and decimal. If the decoder is dual, then Convolution0 Codes are the codes applied in the decoder when the SEL input is low.

If the dual decoder type is selected, then a second output rate and convolution code selection screen is shown:

Output Rate1

Output Rate1 can be any value from 2 to 7. This is the second output rate used if the decoder is dual. The incoming data is decoded at this rate when the SEL input is high. Output Rate 1 is not used for the non-dual decoder and the screen is only available if dual decoder is selected.

Convolution Code 1 Radix

The convolutional codes can be input and viewed in binary, octal, or decimal.

Convolution1 Codes

The convolutional codes are used in the decoder when the decoder is dual and the SEL input is high. The codes are entered in binary, octal, or decimal.

Screen 4 (Packet Options)

See Figure 17. The parameter descriptions for this screen follow.

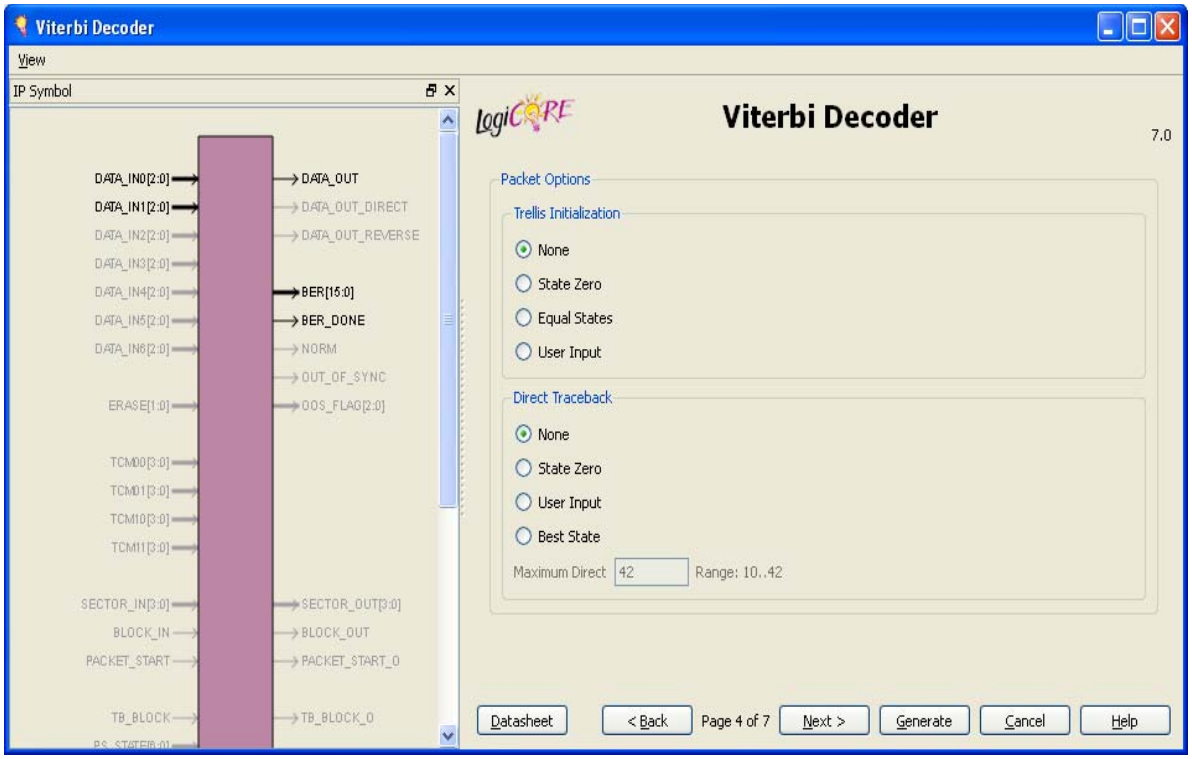

Figure 17: **Viterbi Decoder Screen 4 (Packet Options)**

Trellis Initialization Option

- None: There is no initialization on the trellis and the PACKET_START input signal is not present.
- **State Zero**: The trellis is initialized to state zero when the PACKET_START signal is asserted (high). The costs of the states are all initialized in the ACS module to a maximum value except for state zero.
- **Equal States**: All the states within the trellis are initialized to the same value when the PACKET_START signal is asserted (high).
- **User Input**: The trellis is initialized to the state on PS_STATE when the PACKET_START signal is asserted (high). The costs of the states are all initialized in the ACS module to a maximum value except for the dynamically input state, which is initialized to zero when the PACKET_START input is high. See Figure 9.

Direct Traceback Option

If direct traceback is selected, then the direct traceback data is output on the DATA_OUT_REVERSE and DATA_OUT_DIRECT signals. The DATA_OUT_DIRECT data is also mux'ed into the DATA_OUT data.

- **None**: There is no direct traceback.
- **State Zero**: When the TB_BLOCK signal is asserted (high), the input data is traced back directly without a training sequence from state zero.
- **User Input**: When the TB_BLOCK signal is asserted (high), the input data is traced back directly without a training sequence from the user input TB_STATE. The value of the TB_STATE is selected on the last clock edge of the TB_BLOCK signal high. See Figure 10.
- **Best State**: When the TB_BLOCK signal is asserted (high), the input data is traced back directly without a training sequence from the best state. The best state is generated internally to the decoder from the costs on the ACS modules.

Screen 5 (Puncturing and BER Options)

See Figure 18. The parameter descriptions for this screen follow.

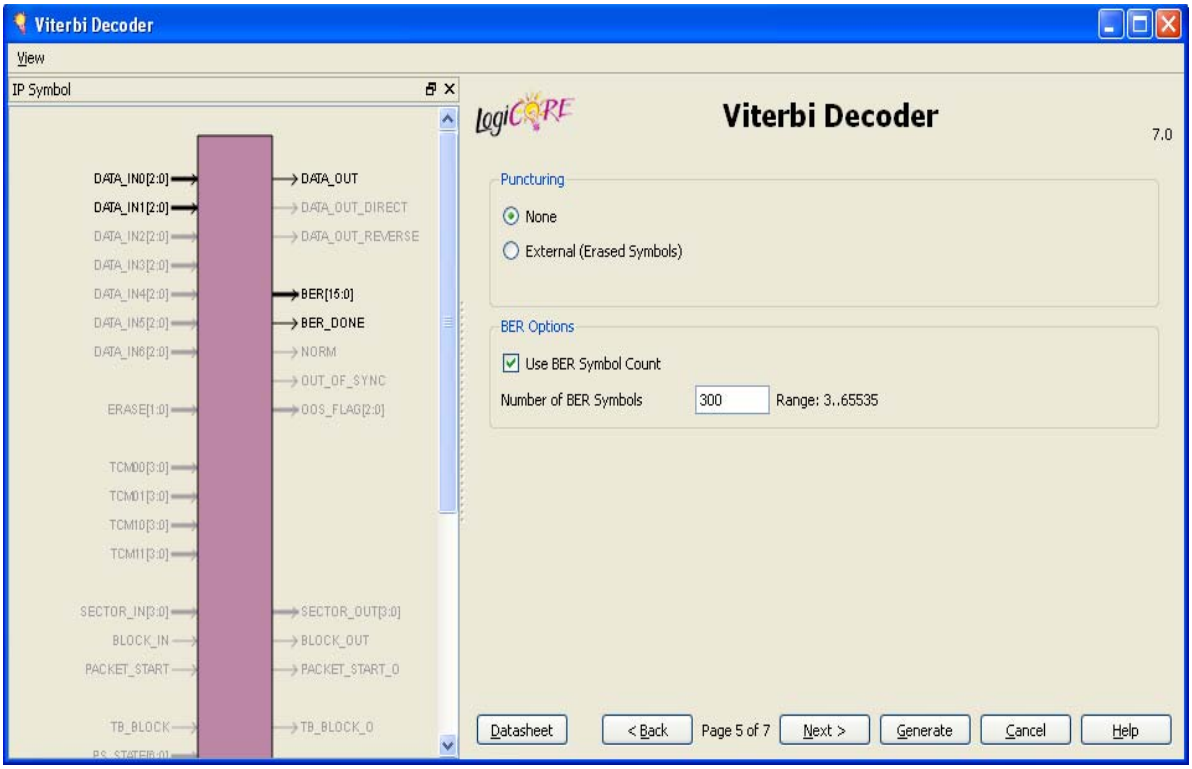

Figure 18: **Viterbi Decoder Screen 5 (Erasure and BER)**

Puncturing Options

- **None**: This indicates there is no external puncturing on the core.
- **External (Erased Symbols)**: This indicates the presence of the erased bus ERASE and allows the core to be depunctured externally prior to decoding. ERASE(0) high indicates that the sample on DATA_IN0 is a null symbol, ERASE(1) high indicates that the data on DATA_IN1 is a null symbol, etc. The size of the erase bus is equal to the output rate of the decoder, or the maximum output rate if the dual decoder is selected.

BER Options

- **Use BER Symbol Count**: Check this box if a Bit Error Rate (BER) monitor is required. The core compares the delayed incoming data with an encoded version of the outgoing decoded data to obtain an estimate of the BER on the channel. See BER, page 6 for additional details.
- **Number of BER Symbols**: The Number of BER symbols is the number of samples over which the BER measurement is carried out. This value can range from 3 to (2^{16}-1).

Screen 6 (Synchronization Options)

See Figure 19. The parameter descriptions for this screen follow.

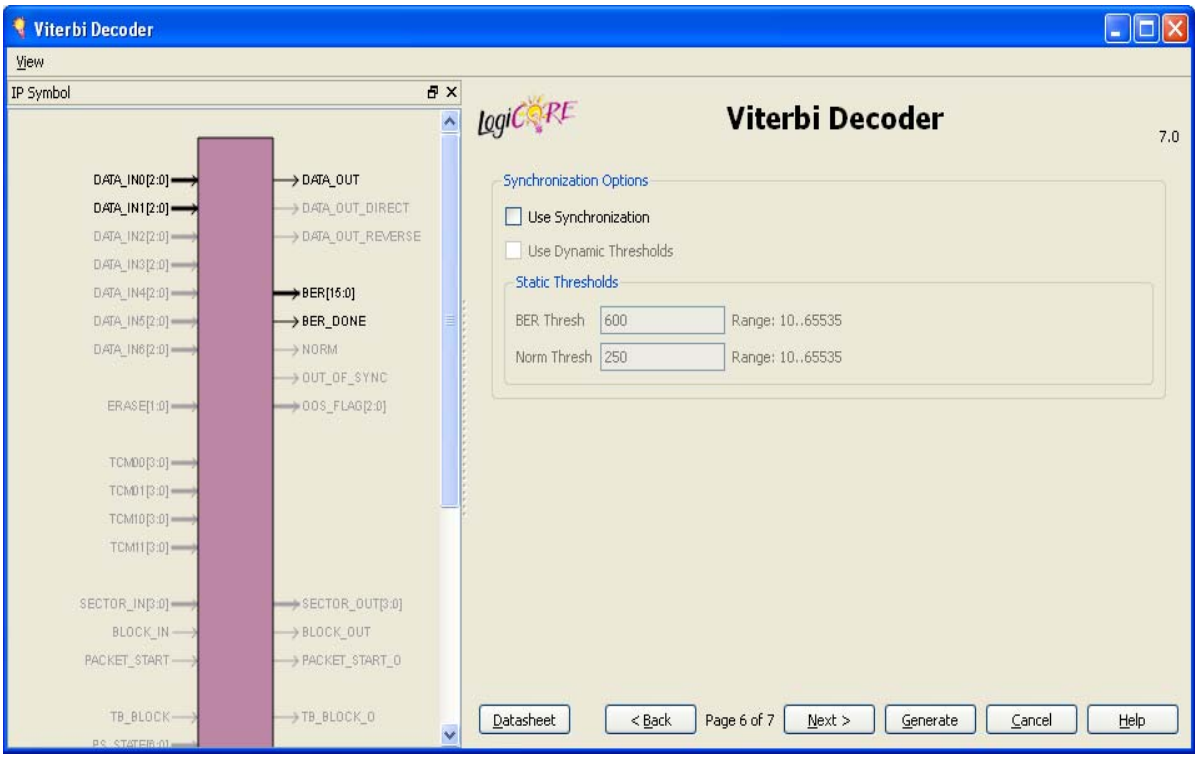

Figure 19: **Viterbi Decoder Screen 6 (Synchronization Options)**

Synchronization Options

- **Use Synchronization**: Check this box if an out of synchronization (OUT_OF_SYNC) output is required. The core evaluates whether the input data is synchronized by examining the normalization and bit error counts and comparing these values to the preset or dynamic thresholds. For additional details, see Synchronization, page 7.
- **Use Dynamic Thresholds**: If this check box is selected, then the synchronization inputs buses NORM_THRESH and BER_THRESH are added to the core. These 16-bit input buses correspond to the BER thresh and Norm thresh above, but allow the thresholds for synchronization evaluation to be dynamically modified.

Static Thresholds

- **BER Thresh:** This is the preset threshold for synchronization evaluation. If the bit error count reaches this threshold before the normalization threshold is obtained, then the core is considered to be out of synchronization and the OUT_OF_SYNC output is asserted.
- **Norm Thresh**: This is the preset threshold for synchronization evaluation. If the normalization count reaches this threshold before the bit error threshold is obtained, then the core is considered to be synchronized and the OUT_OF_SYNC output is deasserted.

Screen 7 (Control Signals)

See Figure 20. The parameter descriptions for this screen follow.

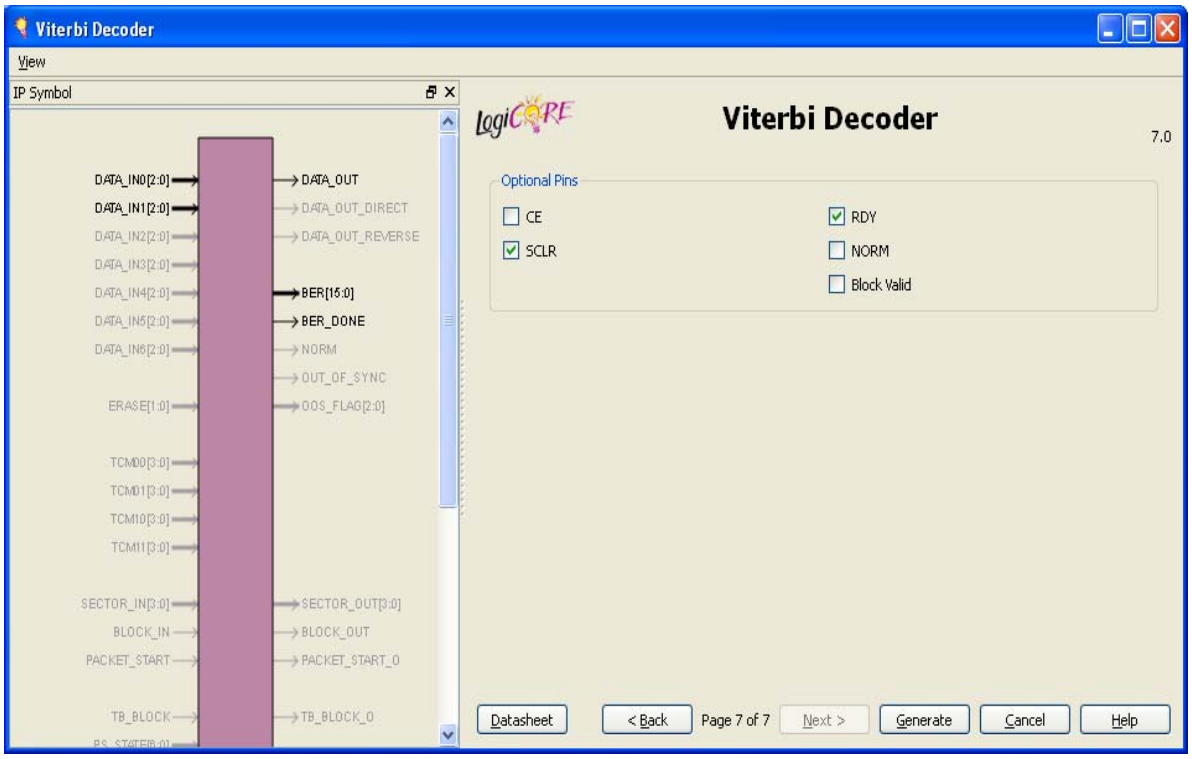

Figure 20: **Viterbi Decoder Screen 7 (Control Signals)**

Optional Pins

Check the boxes of the optional pins that are required: CE, SCLR, RDY, NORM, and Block Valid. Select only pins that are genuinely required, because each selected pin results in more FPGA resources being used and can result in a reduced maximum operating frequency.

NORM

Check this box if a normalization output is required. For additional details, see Normalization, page 7.

Block Valid

Check this box if BLOCK_IN and BLOCK_OUT signals are required. These signals track the movement of a block of data through the decoder. BLOCK_OUT corresponds to BLOCK_IN delayed by the decoder latency.

Parameter Ranges

Valid ranges for the parameters are shown in Table 6.

Table 6: **Parameter Ranges**

Parameter	Min	Max	Notes
Channels	1	32	
Traceback Length	12	128	[1]

Table 6: **Parameter Ranges *(Cont'd)***

Parameter	Min	Max	Notes
Constraint Length	3	9	-
Maximum Direct	10	Traceback Length	-
Best State Width	3	8	-
Soft Width	3	8	-
Output Rates	2	7	[2]
Convolution Code	Bit width = constraint length		-
Number of BER symbols	3	65536	-
BER Thresh	10	65536	-
Norm Thresh	10	65546	-

Notes:
1. Traceback length must be divisible by 6 for the reduced latency case and ranges from 12 to 126.
2. For the Trellis-Mode, the output rate is always 2.

Control Signal Operation

Parallel Decoder

For the parallel case, data input can be controlled by the CE input. If data is continuously input, the decoded data is available on the DATA_OUT after a fixed latency, determined by the constraint length and the traceback length. See Latency, page 27.

For the processing of blocks of data, the data to the encoder must have zero tail-bits inserted, that is, the input block of data must end with at least a (constraint length -1) of zero bits. The tail-bits return the decoder to state zero. As the decoder requires to trace through a traceback length of data prior to outputting the decoded data, encoded zero DATA_IN data should be applied to the decoder between the blocks of decoded data. The combination of zero tail-bits and zero input to the decoder between the encoded blocks allows the decoder to correctly decode encoded blocks of data. The BLOCK_OUT signal is the BLOCK_IN signal delayed by the decoder latency. The BLOCK_IN and BLOCK_OUT signals can be used to identify the decoded blocks as they are output from the decoder. See Figure 21.

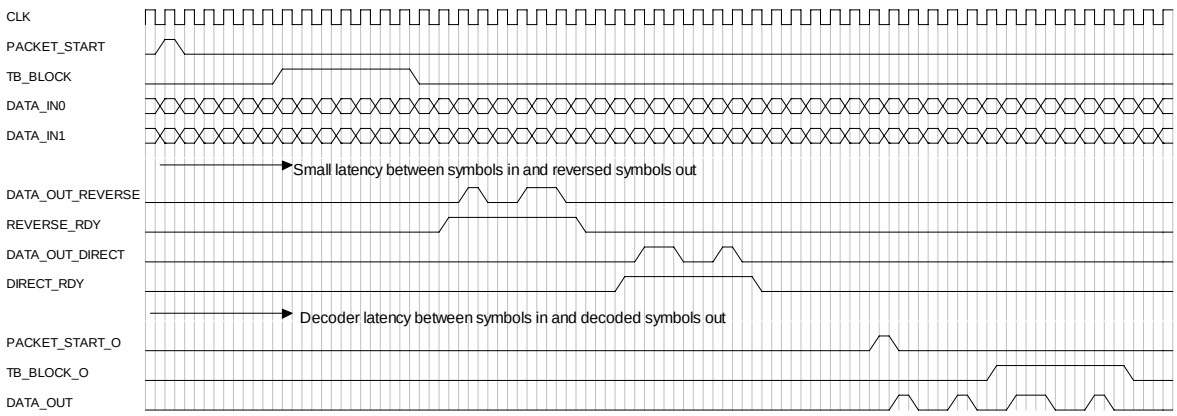

Figure 21: **Packet Handling**

Decoder with External Puncturing

For a decoder with erasure, the presence of a null symbol on the DATA_IN pins is indicated by the erasure pin going high. ERASE(0) corresponds to DATA_IN0, ERASE(1) corresponds to DATA_IN1, etc. The data on the DATA_IN bus is ignored when the corresponding erasure pin is high. See the timing diagram in Figure 22 for a decoder working with external rate 3/4 erasure.

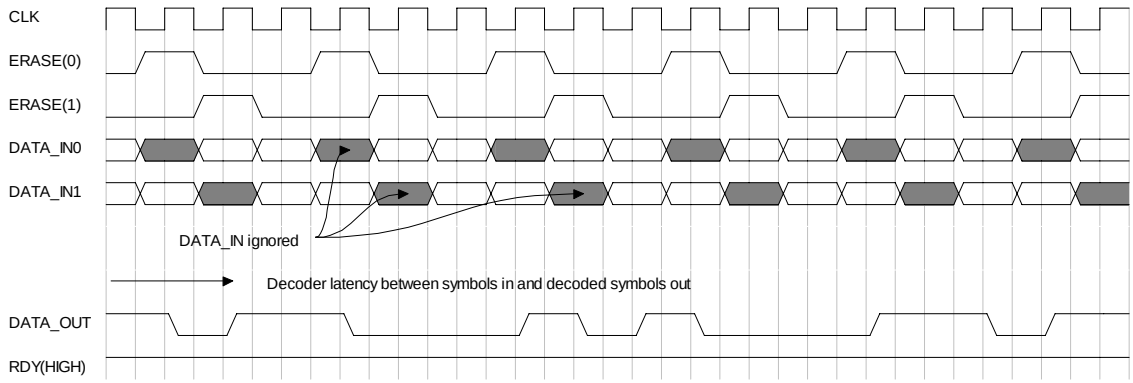

Figure 22: **Viterbi Decoder with Erasure Input Following a Rate 3/4 Pattern**

Serial Decoder

For the serial case, data input is controlled by the New Data (ND) signal. The core must have an ND input and Ready-For-Data (RFD) output. Every new input symbol must be qualified by an ND. In the serial case, ND differs from CE in that the core is not frozen when ND is low. The core processes the new sample as far as possible. Like all the synchronous inputs, ND is ignored if CE is low. After ND goes high, RFD immediately goes low while the input data is processed. RFD remains low for a fixed number of clock cycles. The number of clock cycles required depends on the soft width of the input data and the output rate:

number of clock cycles = soft_width + output_rate + 6

The enabling of the data through the core is controlled by the ND signal. The RDY signal goes high a fixed number of clock cycles after the ND signal has gone high. The delay is equal to:

number of clock cycles = soft_width + output_rate + 12

The DATA_OUT output only changes when the RDY output is high. The DATA_OUT remains valid until the next RDY. See Figure 23 for more information.

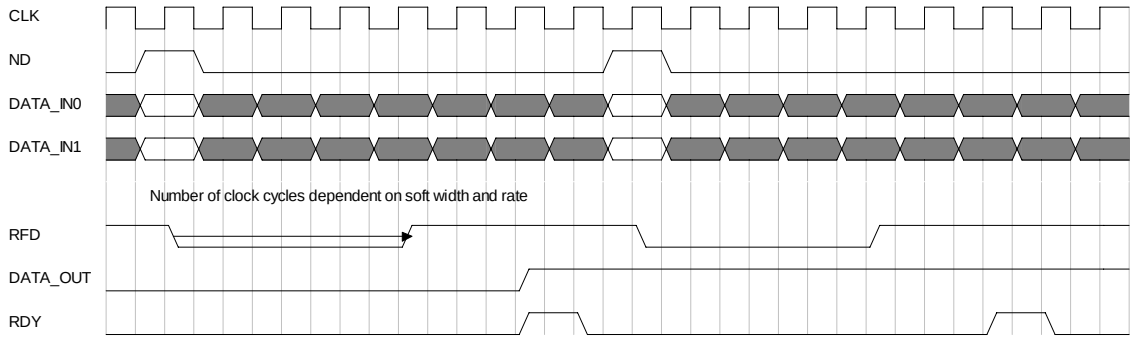

Figure 23: **Serial Decoder Rate 1/2**

Core Resource Utilization

The area of the core increases with the constraint length and the soft width of the input data. Some example configurations are shown in Performance Characteristics, page 28. The slice counts can be reduced slightly by selecting the option to map primary I/O registers into IOBs during placement. This option should certainly be selected if the core I/Os are to be connected directly onto a PCB via the FPGA package pins. This gives lower output clock-to-out times and predictable setup and hold times.

Block RAM Utilization

The block RAM requirements of the core depend on the constraint length and the traceback length. If the reduced latency option is selected, an extra block of RAM is required for the traceback addressing. Multi-channel cores also require extra traceback as each channel requires its own traceback; thus the internal traceback length of the multi-channel core is (channel count * traceback length). See Table 7 for block RAM usage.

Table 7: Block RAM Requirements for the Viterbi Decoder with Standard Latency

Constraint Length	Block RAM
3	1
4	1
5	1
6	1
7	2
8	4
9	8

Latency

The latency of the core depends on the traceback length and the constraint length. If the reduced latency option is selected, then the latency of the core is approximately halved and the latency is only 2 times the traceback length. The latencies given below are a count of the number of symbol inputs between DATA_IN and the decoded data result output on DATA_OUT. Note that the actual latency depends on the parameters selected for the core and the true value of the latency for a given set of parameters can be found through simulation. The rdy signal indicates when there is valid data on the output of the core.

Without Reduced Latency

For a parallel core, the latency is of the order

$$\text{Latency} \cong 4*\text{traceback\_length} + \text{constraint\_length} + \text{output\_rate}$$

For a serial core, the latency is of the order shown. Note that the latency is in terms of valid ND inputs in the serial case.

$$\text{Latency} \cong 4*\text{traceback\_length} + \text{constraint\_length}$$

With Reduced Latency

Reduced latency is only available for the parallel core. The latency is given by

$$\text{Latency} \cong 2*\text{traceback\_length} + \text{constraint\_length} + \text{output\_rate}$$

Multi-Channel Latency

The multi-channel core always uses the standard latency option. The latency in the multi-channel case is given by

$$Latency \cong 4*\text{traceback\_length}*\text{channel\_count} + (\text{constraint\_length}*\text{channel\_count}) + \text{output\_rate}$$

This corresponds to the reduced latency single-channel case above with channel count set to 1.

Trellis Mode Latency

The latency of the Trellis-Mode decoder is as the above equations, but reduced by the output_rate as the branch metric costing unit is not present in the core.

Performance Characteristics

It is important to set a maximum period constraint on the core clock input. The data in Tables 8, 9, and 10 show clock speeds that can be achieved when this is done. It might be possible to improve slightly on these values by trying different options for the place and route software. If necessary, performance can be increased by selecting a part with a faster speed grade.

Table 8: **Characterizations Parallel Viterbi Decoder Constraint Length 7, Output Rate 1/2, Traceback 96, Soft Width 3, and Best State on Virtex-6 FPGAs**

	Parallel	Serial	Multi-Channel 3 Channels
Xilinx Part	6VLX75T	6VLX75T	6VLX75T
LUT/FF Pairs	2794	1642	3763
LUTs[3]	2573	1326	2410
FFs	1863	1497	3434
Block RAMs (36k)[4]	2	2	2
DSP Blocks	0	0	0
Max Clock Freq[1][2]	284/347	316/422	361/478
Mbps	284/347	26/35	120/159 per channel

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 –1/-3 parts for Virtex-6 FPGAs. Clock frequency does not take jitter into account and should be de-rated by an amount appropriate to the clock source jitter specification.
3. LUT count includes route-thrus and may vary when the core is packed with other logic.
4. This is the total number of 36k block RAMs used when map was run. In reality, two 18k block RAM primitives can usually be packed together, giving an absolute minimum total block RAM usage of block RAMs (36k) + (block RAMs (18k) /2) (rounded up).

Table 9: **Characterizations Parallel Viterbi Decoder Constraint Length 7, Output Rate 1/2, Traceback 96, Soft Width 3, and Best State on Spartan-6FPGAs**

	Parallel	Serial	Multi-Channel 3 Channels
Xilinx Part	XC6SLX45T	XC6SLX45T	XC6SLX45T
LUT/FF Pairs	2601	1246	3133
LUTs[3]	2442	1196	2914
FFs	1980	1590	3507
Block RAMs [4]	2	2	4
DSP Blocks	0	0	0
Max Clock Freq[1][2]	126	158	179
Mbps	126	13	59 per channel

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 for Virtex-6 FPGAs. Clock frequency does not take jitter into account and should be de-rated by an amount appropriate to the clock source jitter specification.
3. LUT count includes route-thrus and may vary when the core is packed with other logic.
4. This is the total number of 18k block RAMs used when map was run. In reality, two 18k block RAM primitives can usually be packed together, giving an absolute minimum total block RAM usage of block RAMs (36k) + (block RAMs (18k) /2) (rounded up).

Table 10: **Characterizations Parallel Viterbi Decoder Constraint Length 7, Output Rate 1/2, Traceback 96, Soft Width 3, and Best State on Virtex-5FPGAs**

	Parallel	Serial	Multi-Channel 3 Channels
Xilinx Part	5VLX30	5VLX30	5VLX30
LUT/FF Pairs	2906	1640	3763
LUTs[3]	2457	1326	2410
FFs	1538	1497	3434
Block RAMs [4]	2	2	2
DSP Blocks	0	0	0
Max Clock Freq[1][2]	213/272	267/364	295/387
Mbps	213/272	22/30	98/129 per channel

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 –1/3 parts for Virtex-6 FPGAs. Clock frequency does not take jitter into account and should be de-rated by an amount appropriate to the clock source jitter specification.
3. LUT count includes route-thrus and may vary when the core is packed with other logic.
4. This is the total number of 36k block RAMs used when map was run. In reality, two 18k block RAM primitives can usually be packed together, giving an absolute minimum total block RAM usage of block RAMs (36k) + (block RAMs (18k) /2) (rounded up).

BER Performance

BER performance curves were generated using the Xilinx XtremeDSP™ kit. A BER tester design is downloaded to the board through a MATLAB® interface. The BER tester design consists of the Xilinx Convolutional Encoder, the Xilinx AWGN core, the Viterbi Decoder, and the Nallatech BER tester circuit as shown in Figure 24.

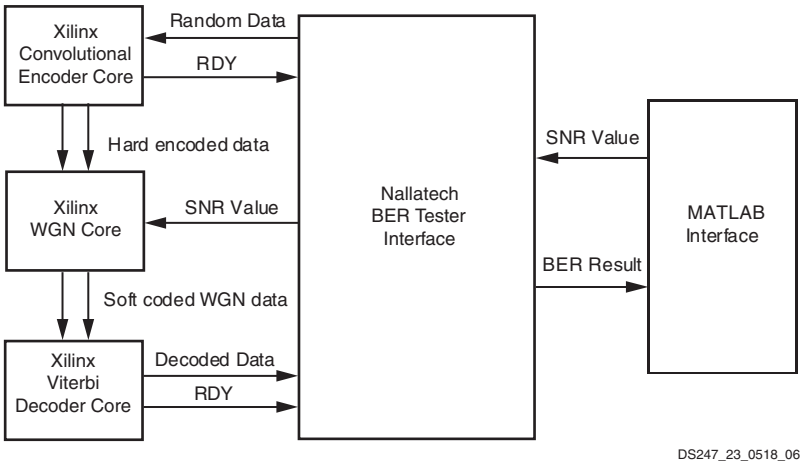

Figure 24: **BER Testing Circuit for Use with Xilinx XtremeDSP Kit**

Figure 25 shows the BER performance of the core for constraint lengths 5, 7, and 9 with soft width 4. Figure 26 shows the BER performance of the core for a constraint length 7 decoder with various channel rates. Figure 27 shows the effect of varying the soft width and the traceback length for a constraint length 7 decoder. The plot clearly demonstrates the need for the traceback length to be at least equal to 6 times the constraint length for a rate 1/2 decoder.

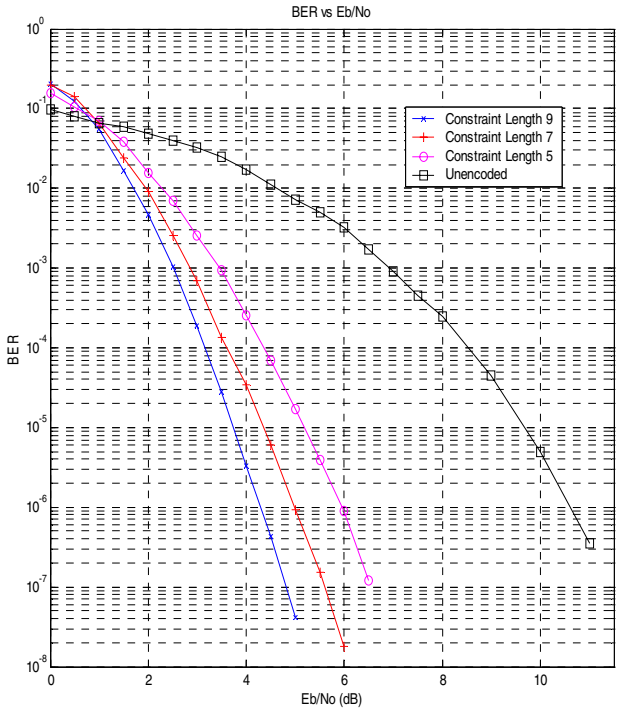

Figure 25: **BER Testing Circuit for Use with Xilinx XtremeDSP Kit**

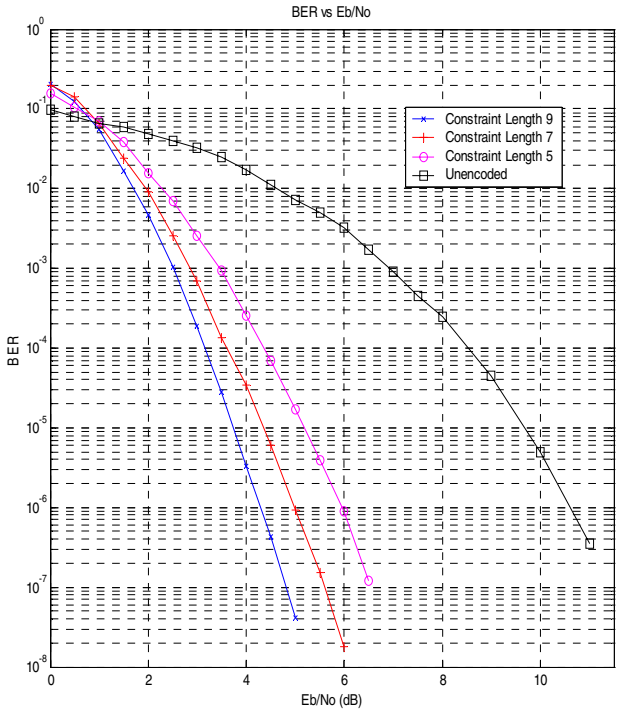

Figure 26: **BER Testing Circuit for Use with Xilinx XtremeDSP Kit**

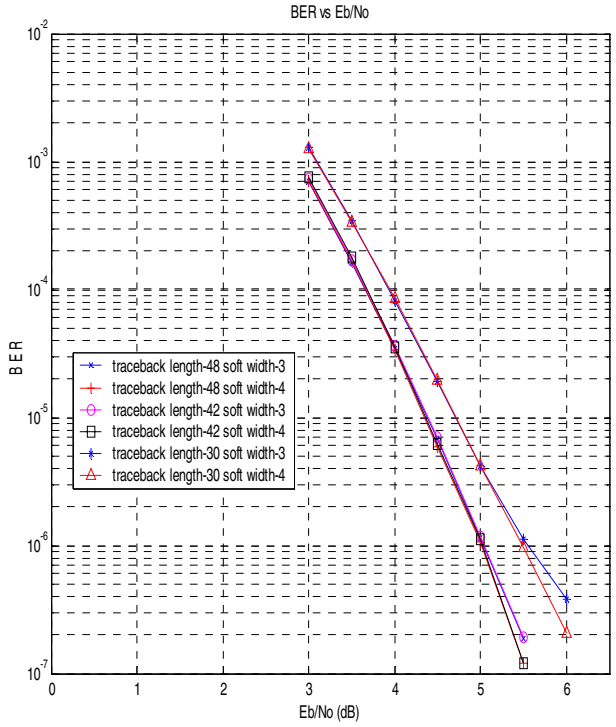

Figure 27: **Constraint Length 7 Decoder with Varying Soft Width and Traceback Length**

An example BER test design is also available in the Xilinx System Generator, as shown in Figure 28. The design can be used for hardware co-simulation with any of the currently supported boards.

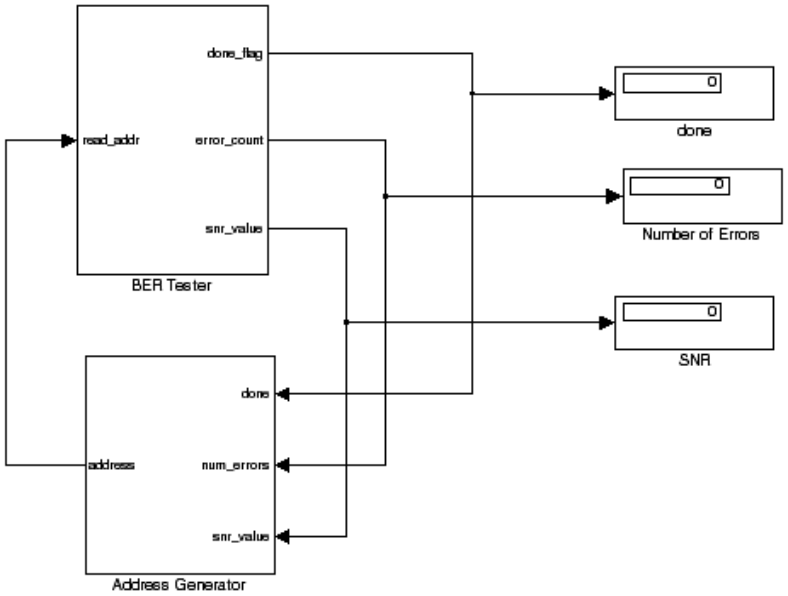

Figure 28: **BER Example Circuit Available in System Generator**

References

1. LogiCORE IP Viterbi Decoder v7.0 User Guide (UG745)

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

The Viterbi Decoder core is provided under the SignOnce IP Site License and can be generated using the Xilinx® CORE Generator v13.1. The CORE Generator software is shipped with ISE® Design Suite software v13.1.

To access the full functionality of the core, including simulation and FPGA bitstream generation, a full license must be obtained from Xilinx. For more information, visit the Viterbi Decoder product page.

Contact your local Xilinx sales representative for pricing and availability of additional Xilinx LogiCORE modules and software. Information about additional Xilinx® LogiCORE modules is available on the Xilinx IP Center.

Revision History

The following table shows the revision history for this document.

Date	Version	Revision
03/28/03	1.0	Revision history initiated.
03/16/04	2.0	Updated to version 4.0 standards.
11/11/04	3.0	Updated to support Viterbi Decoder core v6.0, including software support of v6.3i.
4/28/05	4.0	Updated to support Spartan-3E FPGAs; Xilinx tools 7.i1.
9/28/06	6.0	Updated to support Virtex-5 LX/LXT, Spartan-3/XA, and Spartan-3E/XA FPGAs.
05/17/07	6.1	Updated to support Spartan-3A DSP FPGAs.
10/10/07	6.2	Updated to support Viterbi Decoder core v6.2.
06/24/09	7.0	Removed aclr pin, the speed option, 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