Simulation of Convolutional Encoder and Viterbi Decoder using Verilog

Nehul Mathur
M.Tech. Scholar, Digital Communication, Pacific University
Udaipur, Rajasthan

Sunil Sharma
Assistant Professor, Department of Electronics and Communication
Pacific University
Udaipur, Rajasthan

ABSTRACT
In this paper, we are implementing the Convolutional encoder and viterbi decoder with code rate 2/3 using verilog.

The main issue of this paper is to implement the RTL level model of Convolutional encoder and viterbi decoder, with the testing results of behavior model. We tried to achieve a low silicon cost. The viterbi algorithm, used for Convolutional codes extensively employed decoding algorithm for Convolutional codes. This paper is realized using verilog HDL. It is simulated and synthesized using Modelsim Altera 10.1d.

General Terms
Convolutional codes were first introduced by Elias in 1955 as an alternative to block codes. Convolutional codes differ from block codes in that the encoder contains memory and the n encoder outputs at any time unit depend not only on the k inputs but also on m previous input block. In 1963, Massey proposed a less efficient but simpler to implement decoding method called threshold decoding. Then in 1967, Viterbi proposed a maximum likelihood decoding scheme that was easy to implement for codes. This scheme is called viterbi decoding.

Keywords
Convolutional encoder, viterbi decoder, Modelism10.1d, viterbi algorithm, trellis diagram, simulation.

1. INTRODUCTION
A Convolutional encoder operates on the incoming message sequence continuously in a serial manner.

The encoder of a binary Convolutional code with rate 1/n. measured in bits per symbol, may be viewed as a finite-state machine that consists of a M-stage shift register with prescribed connections to n modulo-2 adder, and a multiplexer that serializes the outputs of the adders. An 1-bit message sequence produces a coded output sequence of length n(L+M) bits[1]. The code rate is therefore given by

$$r = \frac{L}{n(L+M)} \text{ bits/symbol}$$

Typically, we have L>>M. Hence, the code rate is

$$r = \frac{1}{n} \quad \text{bits/symbol [2]}$$

The constraint length of a Convolutional code, expressed in terms of message bits, is defined as the number of shifts over which a single message bit can influence the encoder output. In an encoder with a M-stage shift register, the memory of the encoder equals M message bits, and K=M+1 shifts are required for a message bit to enter the shift register and finally come out. Hence, the constraint length of the encoder is K[4].

2. PROPOSED WORK AND METHODOLOGY

2.1 Convolutional encoder
A Convolutional code is generated by passing the information sequence to be transmitted through a linear finite-state shift Register. In General, the shift register consists of K (k-bit) stages and n Linear algebraic function generators[5].

Convolutional codes
k = number of bits shifted into the encoder at one time
k=1 is usually used!!
n = number of encoder output bits corresponding to the *k* Information bits
Rc = *k*/*n* = code rate
K = constraint length, encoder memory.
Each encoded bit is a function of the present input bits and their past ones[6].

Convolutional encoding is mostly used for satellite and other noise communication channels. There are two important components of a channel using Convolutional encoding: the viterbi encoder (at the transmitter) and the viterbi decoder (at the receiver)[7].

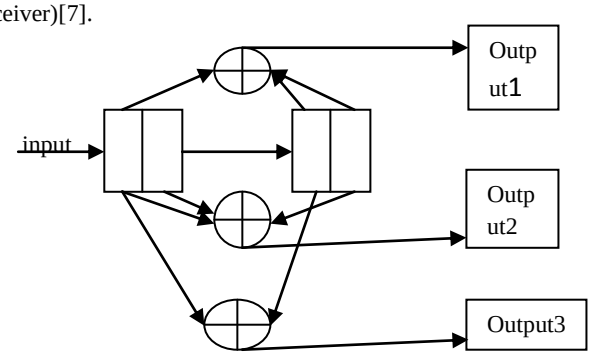

Figure 1. Block Diagram of code rate 2/3 Convolutional encoder

The generators are g1=[1011], g2=[1101], g3=[1010]. In octal form, these generators are (13, 15 and 12).

The ratio of input to output information in an encoder is the rate of the encoder; this is a rate 2/3 encoder. The following equations relate the three encoder output bits (Y_{2N} , Y_{1N} , and Y_{0N}) to the two encoder input bits (X_{2N} and X_{1N}) at a time nT:

$Y_{2N} = X_{2N}$, Y1N = X_{1N} xor X_{1N_2}, $Y_{0N} = X_{1N\_1}$

We can write the input bits as a single number. Thus, for example, if $X_{2N} = 1$ and $X_{2N} = 0$, we can write $X_N = 2$. Equation defines a state machine with two memory elements for the two last input values for X_{1N}: X_{1N_1} and X_{1N_2}. These two state variables define four state s :{ $X1N_{\_1}$, X_{1N_2}}, with $S_0 = \{0, 0\}$, $S_1 = \{1, 0\}$, $S_2 = \{0, 1\}$, and $S_3 = \{1, 1\}$. The 3-bit output Y_n is a function of the state and current 2-bit input X_n.

Table 2. A sequence of transmitted signals for the rate 2/3 Convolutional encoder

Present state	Inputs		State variables		outputs			Next state	
	X2n	X1n	X1n_1	X1n_2	Y2n	Y1n	Y0n	X1n_1	X1n_2
					Y2n =X2n	Y1n = X1n xor X1n_2	Yon = X1n_1	X1n_1	X1n_2
S0	0	0	0	0	0	0	0	00	S0
S0	0	1	0	0	0	1	0	10	S1
S0	1	0	0	0	1	0	0	00	S0
S0	1	1	0	0	1	1	0	10	S1
S1	0	0	1	0	0	0	1	01	S2
S1	0	1	1	0	0	1	1	11	S3
S1	1	0	1	0	1	0	1	01	S2
S1	1	1	1	0	1	1	1	11	S3
S2	0	0	0	1	0	1	0	00	S0
S2	0	1	0	1	0	0	0	10	S1
S2	1	0	0	1	1	1	0	00	S0
S2	1	1	0	1	1	0	0	10	S1
S3	0	0	1	1	0	1	1	01	S2
S3	0	1	1	1	0	0	1	11	S3
S3	1	0	1	1	1	1	1	01	S2
S3	1	1	1	1	1	0	1	11	S3

Table 1. State table for 2/3 rate Convolutional encoder

As an example, the repeated encoder input sequence X_N=0, 1, 2, 3 produces the encoder output sequence Y_N=1, 0, 5, 4 repeated.

Inputs		State variables		Outputs			Present state	Next state
X2n	X1n	X1n_1	X1n_2	Y2n	Y1n	Y0n		
1	1	X	X	1	X	X	S?	S?
1	1	0	0	1	1	0	S0	S1
0	0	1	0	0	0	1	S1	S2
0	1	0	1	0	0	0	S2	S1
1	0	1	0	1	0	1	S1	S2
1	1	0	1	1	0	0	S2	S1
0	0	1	0	0	0	1	S1	S2
0	1	0	1	0	0	0	S2	S1
1	0	1	0	1	0	1	S1	S2
1	1	0	1	1	0	0	S2	S1
0	0	1	0	0	0	1	S1	S2
0	1	0	1	0	0	0	S2	S1

Next I have transmit the eight possible encoder outputs (Yn = 0-7) as signals over our noisy communications channel (perhaps a microwave signal to a satellite) using the signal constellation. Typically this is done using phase-shift keying (PSK) with each signal position corresponding to a different phase shift in the transmitted carrier signal.

The noise signal enters the receiver, so my task to discover the 8 possible signals was transmitted at each time step. First I have calculated the distance of each received signal from each of known 8 positions according to signal constellation. The distance between signals in the 8PSK constellation. I am assuming that there is no error or noise in the channel to illustrate the operation of viterbi decoder, so that the distances in the table 3 represent the possible distance measures of received signal from the 8PSK signal.

Table 3. Representation of the possible distance measures of received signals from the 8PSK signals

Signal	Algebraic distance from signal 0	X = distance from signal 0	Euclidean distance E = X2	B = binary quantized value of E	D = decimal value of B	Q =D-1.75 E
0	$2\sin(0\pi/8)$	0.00	0.00	000	0	0
1	$2\sin(1\pi/8)$	0.77	0.59	001	1	-0.0325

2	$2\sin(2\pi/8)$	1.41	2.00	100	4	0.5
3	$2\sin(3\pi/8)$	1.85	3.41	110	6	0.0325
4	$2\sin(4\pi/8)$	2.00	4.00	111	7	0
5	$2\sin(5\pi/8)$	1.85	3.41	110	6	0.0325
6	$2\sin(6\pi/8)$	1.41	2.00	100	4	0.5
7	$2\sin(7\pi/8)$	0.77	0.59	001	1	-0.0325

Now table 4 shows the distance measures for the transmitted encoder output sequence Yn = 1, 0, 5, 4……. Corresponding to the encoder input of Xn = 0, 1, 2, 3……

Table 4. Receiver distance measure for transmission sequence

Input Xn	Output Yn	Present state	Next state	In0	In1	In2	In3	In4	In5	In6	In7
3	X	S?	S?	X	X	X	X	X	X	X	X
3	6	S0	S1	4	6	7	6	4	1	0	1
0	1	S1	S2	1	0	1	4	6	7	6	4
1	0	S2	S1	0	1	4	6	7	6	4	1
2	5	S1	S2	6	7	6	4	1	0	1	4
3	4	S2	S1	7	6	4	1	0	1	4	6
0	1	S1	S2	1	0	1	4	6	7	6	4
1	0	S2	S1	0	1	4	6	7	6	4	1
2	5	S1	S2	6	7	6	4	1	0	1	4
3	4	S2	S1	7	6	4	1	0	1	4	6
0	1	S1	S2	1	0	1	4	6	7	6	4
1	0	S2	S1	0	1	4	6	7	6	4	1

2.2 Viterbi decoder

The Viterbi decoder takes the distance measures and calculates the most likely transmitted signal. It does this by keeping a running history of the previously received signals in a path memory. The path-memory length of this decoder is 12. By keeping a history of possible sequences and using the knowledge that the signals were generated by a state machine, it is possible to select the most likely sequences.

The system input or message, X[1:0] , is driven by a counter that repeats the sequence 0, 1, 2, 3, ... incrementing by 1 at each positive clock edge (with a delay of one time unit), starting with X equal to 3 at t= 0. The active-high reset signal, Res, is asserted at t = 60 for 10 time units. The encoder output, Y [2:0], changes at t = 151, which is one time unit (the positive-edge-triggered D flip-flop model contains a one-time-unit delay) after the first positive clock edge (at t = 150) following the dissertation of the reset at t = 70. The encoder output sequence beginning at t= 151 is 2, 5, 4, 1, 0, and then the sequence 5, 4, 1, 0, repeats. This encoder output sequence is then imagined to be transmitted and received. The receiver module calculates the distance measures and passes them to the decoder. After 13 positive clock-edges (1300 time ticks) the transmitted sequence appears at the output, Out [2:0], beginning at t = 1451 with 2, 5, 4, 1, 0, exactly the same as the encoder output.

The Viterbi decoder model presented in this work is written for both simulation and synthesis. The Viterbi decoder makes extensive use of vector D flip-flops (registers). Early versions of Verilog-XL did not support vector instantiations of modules. In addition the inputs of UDPs may not be vectors and there are no primitive D flip-flops in Verilog. This makes instantiation of a register difficult other than by writing a separate module instance for each flip-flop.

3. SIMULATION RESULT ANALYSIS ON MODELSIM 10.1d

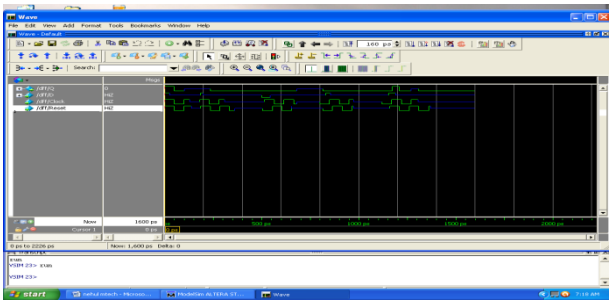

Figure 2 Simulation Wave form of D flip-flop using Modelsim 10.1d.

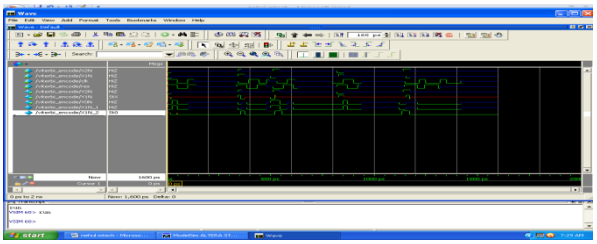

Figure 3 Simulation waveform of Convolutional encoder using Modelsim 10.1d

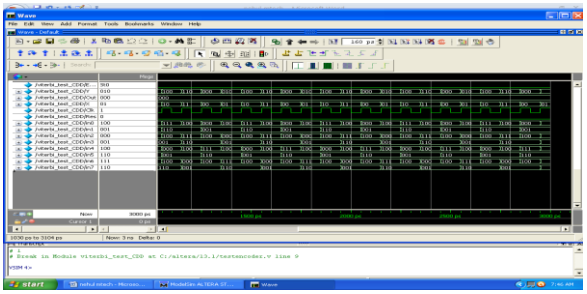

Figure 4 Simulation waveform of test bench using modelsim 10.1d

4. CONCLUSION

The main air was to implement convolution encoder and viterbi decoder with code rate 2/3 in compact verilog style. As verilog implementation works in module form and it is comparatively simple than other HDL language. Second, the design of viterbi decoder is in the receiver end so that, implementation would take less memory space. While tracing path back towards front end, look up method saves lot of time and complexity.

The modulation scheme 8 PSK was used here for faster coding with three bit input frame.

By building the Convolutional encoder and viterbi decoder in the behavior model, the Modelsim simulation result give us a light on its performance.

In future it will be used to measure & improve the performance of designed encoder-decoder pair. The next level implementation will also consist power saving coding style. Here the code is written in specific way to make the design low power design.

It can also be prefered for comparatively low power consumption with other rtl code.

5. ACKNOWLEDGMENTS

The author would like to express their sincere thanks to T.A. Kaji Director, Pacific Institute of Technology, Udaipur, for his support during research. Mr. Ashok Kherodia, HOD, Dept. of ECE, Mr. Sunil Sharma, Assistant Professor, Dept. of ECE from Pacific university Udaipur, Rajasthan for the encouragement given throughout this development.

6. REFERENCES

[1] Simon Haykin and Michael Moher: Modern wireless communications; Pearson Prentice Hall, 2005

[2] Simon Haykin; Digital communications; Wiley, cop. 1988

[3] Samir Palnitkar: Verilog HDL a Guide to Digital Design and Synthesis; sun soft press (1996);

[4] Chip Fleming: A Tutorial on Convolutional Coding with Viterbi Decoding; Spectrum Application; Jun, 2003;

[5] Pravallika.kolakaluri, R.Suryaprakash: HDL Implementation of Convolutional encoder and viterbi decoder; July, 2012;

[6] Rohan M. Pednekar, Dayanand B M: Design and Implementation of Convolutional encoder and viterbi decoder; 2013;

[7] Swati Gupta, Rajesh Mehra: FPGA Implementation of viterbi decoder using track back architecture; June, 2011;

[8] Mahe Jabeen, Salma Khan: Design of Convolutional encoder and Reconfigurable viterbi decoder; (sept 2012);