Coverage Analysis in the Verification of 8B/10B Encoder

Prateek Kumar Jana
Department of ECE
National Institute of Technology, Rourkela
Email:prateekjana@rediffmail.com

Debiprasad Priyabrata Acharya
Department of ECE
National Institute of Technology, Rourkela
Email:d_p_acharya@rediffmail.com

ABSTRACT
The Verification is a vital step of any ASIC development process. 8B/10B encoder is a very widely used block in communication systems. The RTL code of an 8B/10B encoder is simulated in Questasim environment. The 8B/10B encoder is put to formal verification process in this work. Coverage analysis gives a view of the efficiency of the code. The report of coverage analysis of 8B/10B encoder is presented in this work.

General Terms
ASIC development process, Coverage analysis

Keywords
8b/10b encoder, Verification, Code coverage

1. INTRODUCTION
Intellectual Property (IP) Cores are of first line of choice in the development of Systems-on-chip (SOC). IP Cores are register transfer level (RTL) codes which achieve certain desired functionality. These are well tested codes which must be ready for any use in SOC development. 8B/10B encoder is a block that is frequently used in communication systems; hence development of IP Cores for 8B/10B encoder is important. Predominant part of the development process of IP Cores using HDLs is verification. Verification is an exhaustive process of checking the function of this code. With the increase in complexity of designs, the functional verification have increased sharply in recent years mainly pushed by the major EDA companies. Today the verification engineers have outnumbered the design engineers for the most complex designs. Studies revealed that 74% of all respins of IC's are due to functional errors. Verification has become the bottleneck in a project's time-to-profit goal [1]. This paper uses an available RTL code of 8B/10B encoder, and carries out the verification process extensively. The 8B/10B encoder is taken up for verification using Questasim, such that the proposed verification methodology can be exercised using this simple RTL 8B/10B design. The coverage analysis is carried out to demonstrate the efficiency of performance of the RTL code of 8B/10B encoder. The remaining part of the paper is organized as follows; Section 2 describes the 8B/10B encoder structure. Section 3 describes the Coverage analysis, Section 4 shows results and discussion and Section 5 gives Conclusion.

2. 8B/10B ENCODER
A byte-oriented transmission code converts an 8 bit symbol to a 10 bit symbol [2].The converted 10 bit symbol should be such that it contains equal numbers of '1's and '0's. The application of scheme is such that at one time not more than five 0's or 1's are ever transmitted. The difference in the number of 1's and 0's is called disparity and can accept values of 0, -1 or +1 in the form of encoded 10-bit symbols. In order to maintain an overall DC balanced stream the disparity of one 10-bit output code is feed back to the encoder in order to compensate for non-zero disparity if any. The 8B/10B encoder have its application in PCI express, Serial ATA, USB 3.0, Fiber Channel, SSA and many more.

2.1 Structure of 8B/10B code
Special characters are included in the transmission codes known as D characters and K characters. In each byte of the transmission code the parity is monitored and accordingly D and K characters are related to positive or negative parity. Here the encoder selects parity for each code word for maintaining balance running parity. In this 8B/10B encoder the each incoming byte of data is fragmented into two different sub-blocks [3] which is shown in Fig 1.

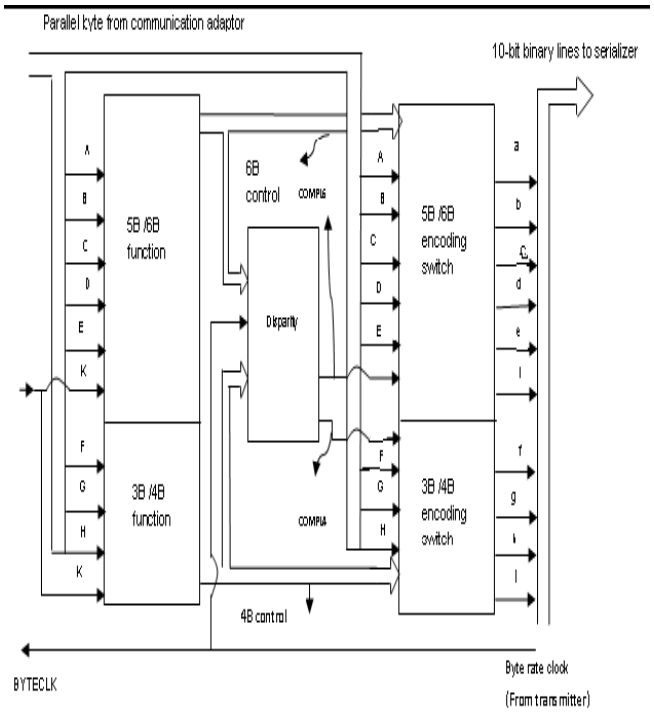

Fig 1: 8B/10B Encoding block diagram

The ABCDE serves as the first five inputs and gives abcdei as output and the remaining three inputs FGH yields the output fghj, following the encoding schemes 5B/6B and 3B/4B respectively. Every 10-bit encoded data group has one of the three possibilities to help in limiting the number of consecutive "1s" or "0s" in any 2- code words [3].

_ Five "1s' and five"0s"
_ Four "1s" and six"0s"
_ Six"1s" and four "1s"

2.2 8B/10B Coding Scheme

The encoding scheme of 8B/10B is done by following the coding plans and rules as describes in [2] for 5B/6B encoding Table 1 is referred and Table 2 is referred for 3B/4B encoding.

2.3 5B/6B Encoding Scheme

The column heading "Name" in the Table 1, gives the inputs ABCDE as the 32 decimal equivalents with A and E as low and high order bit. The K line must be held at 0 for regular data (D.x).Apart from this a few code points name K.x or D/K.x and have an 1 or x in column K can be a part of special characters that can be identifiable as other than data. The line K must be 1 for encoding special characters. In the classification columns, L04 means that there are no 1s but four 0s in ABCD; L13 convey that there are single 1 and three 0s in ABCD, etc. The L letter signifies that the logic function or classification used is a part of the 5B/6B encoder. Similar functions labeled P are defined for decoding. In order to symbolize negation of a symbol to the right an accent is used; E` means the complement of E; a dot (.) stands for the logical AND function. The code points generated from the inputs of ABCDE as a result of direct 5B/6B logic functions are all listed under the left heading abcdei in the column. In order to change positively the minimal number of bits while passing through the encoder the coding table was designed, so that the changes which are needed can be categories to relevant groups applicable to several code points. When the logical conditions listed on the left side under bit encoding meets the inputs it results in the change of bits values as shown in the left abcdei column, as shown for D.0 as well as D.16 the digits b and c are compel to 1s if L04 holds. In the bit encoding the second entry applies for the column D.16(L04.E) as well as D.31(L40.E).The added i-bit is a 0 and the bits ABCDE moved unchanged into abcde for those lines with no classification entry [2]. In the classification column if bit encoding is given and Disparity is not given then, the bits ABCDE move unchanged into abcde and the i-bit is 1. If bit encoding is not given and Disparity is given then, the ABCDE bits remains same and represented as abcde with the i-bit as 0.The another column with "alternate abcdei" to the right of the table 1 shows the complement for those ABCDE inputs, which have alternate code points. According to the disparity rules the complemented individual sub-blocks are 6B and 4B. The running disparity is never 0 it is either -1 or +1 at all sub-block boundaries.

2.4 3B/4B Encoding Scheme

The Table 2 follows the conventions and notations of Table 1 and converts the bits FGH into the digits fghj. In order to classify the disparity some columns in Table 2 have double entries.

2.5 Special Characters

The extra code points apart from the 256 needed for encoding a byte of data are called special characters. They are used to mark the start and end of packets, for establishment of byte synchronization, and sometimes for communicating control operations such as RESET, SHUT-OFF IDLE, ABORT, and link diagnostics. In Table 3 the set of twelve special characters depicted can be generated by the coding rules given in Table 1 and 2. They all follow with the general coding constraints of a maximum digital sum variation of 6 and a maximum run length of 5 [2].

TABLE 1. 5B/6B Encoding

Name	ABCDEK	Classifications		D-1	abcdei	D0	abcdei
		Bit encoding	Disparity				Alternate
D.0	00000 0	L04	L22′•L31′•E′	+	011000	–	100111
D.1	10000 0	L13•E′	L22′•L31′•E′	+	100010	–	011101
D.2	01000 0	L13•E′	L22′•L31′•E′	+	010010	–	101101
D.3	11000 0	L22•E′		x	110001	0	
D.4	00100 0	L13•E′	L22′•L31′•E′	+	001010	–	110101
D.5	10100 0	L22•E′		x	101001	0	
D.6	01100 0	L22•E′		x	011001	0	
D.7	11100 0		L31•D′•E′	–	111000	0	000111
D.8	00010 0	L13•E′	L22′•L31′•E′	+	000110	–	111001
D.9	10010 0	L22•E′		x	100101	0	
D.10	01010 0	L22•E′		x	010101	0	
D.11	11010 0			x	110100	0	
D.12	00110 0	L22•E′		x	001101	0	
D.13	10110 0			x	101100	0	
D.14	01110 0			x	011100	0	
D.15	11110 0	L40	L22′•L31′•E′	+	101000	–	010111
D.16	00001 0	L04, L04•E	L22′•L13′•E	–	011011	+	100100
D.17	10001 0	L13•D′•E		x	100011	0	
D.18	01001 0	L13•D′•E		x	010011	0	
D.19	11001 0			x	110010	0	
D.20	00101 0	L13•D′•E		x	001011	0	
D.21	10101 0			x	101010	0	
D.22	01101 0			x	011010	0	
D/K.23	11101 x		L22′•L13′•E	–	111010	+	000101
D.24	00011 0	L13•D•E	L13•D•E	+	001100	–	110011
D.25	10011 0			x	100110	0	
D.26	01011 0			x	010110	0	
D/K.27	11011 x		L22′•L13′•E	–	110110	+	001001
D.28	00111 0			x	001110	0	
K.28	00111 1	L22•K	K	–	001111	+	110000
D/K.29	10111 x		L22′•L13′•E	–	101110	+	010001
D/K.30	01111 x		L22′•L13′•E	–	011110	+	100001
D.31	11111 0	L40, L40•E	L22′•L13′•E	–	101011	+	010100

TABLE 2. 3B/4B Encoding

Name	FGH K	Classifications		D-1	fghj	D0	fghj
		Bit encoding	Disparity				Alternate
D/K.x.0[a]	000 x	F′•G′•H′	F′•G′	+	0100	–	1011
D.x.1	100 0	(F≠G)•H′		x	1001	0	
D.x.2	010 0	(F≠G)•H′		x	0101	0	
D/K.x.3[a]	110 x		F•G	–	1100	0	0011
D/K.x.4[a]	001 x		F′•G′	+	0010	–	1101
D.x.5	101 0			x	1010	0	
D.x.6	011 0			x	0110	0	
D.x.P7	111 0		F•G, F•G•H	–	1110	+	0001
D/K.y.A7[b,c]	111 x	F•G•H•(S+K)	F•G, F•G•H	–	0111	+	1000
K.28.1	100 1	(F≠G)•H′	(F≠G)•K	+	1001	0	0110
K.28.2	010 1	(F≠G)•H′	(F≠G)•K	+	0101	0	1010
K.28.5	101 1		(F≠G)•K	+	1010	0	0101
K.28.6	011 1		(F≠G)•K	+	0110	0	1001

TABLE 3. Special characters(K=1)

Name	ABCDEFGH K	D-1 abcdei	fghj D0	D-1 abcdei Alternate	fghj D0
K.28.0	00111 000 1	- 001111 0100	0	+ 110000 1011	0
K.28.1*	00111 100 1	- 001111 1001	+	+ 110000 0110	-
K.28.2	00111 010 1	- 001111 0101	+	+ 110000 1010	-
K.28.3	00111 110 1	- 001111 0011	+	+ 110000 1100	-
K.28.4	00111 001 1	- 001111 0010	0	+ 110000 1101	0
K.28.5*	00111 101 1	- 001111 1010	+	+ 110000 0101	-
K.28.6	00111 011 1	- 001111 0110	+	+ 110000 1001	-
K.28.7**	00111 111 1	- 001111 1000	0	+ 110000 0111	0
K.23.7	11101 111 1	- 111010 1000	0	+ 000101 0111	0
K.27.7	11011 111 1	- 110110 1000	0	+ 001001 0111	0
K.29.7	10111 111 1	- 101110 1000	0	+ 010001 0111	0
K.30.7	01111 111 1	- 011110 1000	0	+ 100001 0111	0

3. COVERAGE ANALYSIS

Code coverage is used to evaluate the competency of verification implementation and also provide a quantitative measurement of the testing space. It also depict the extent to which the source code of a DUT has been tested. By applying code coverage analysis techniques to RTL codes of certain blocks, verification efficiency was improved by enabling a verification engineer to isolate areas of un-tested RTL code. The verification engineer analyze the coverage report, find out the low values and figure out the reasons for the particular code that hasn't been tested fully and writes more exhaustive tests or directs randomness to cover the untested areas where there could be a possibility of presence of a hidden bug.

3.1 Code coverage Analysis of encoder

The 8b/10b encoder is processed through various sections such as simulation and circuit interconnects there after it is processed for Code coverage which includes Statement coverage, Block coverage, Conditional coverage, Toggle coverage etc. and the report were generated to get the satisfactory result, in the course of coverage.

4. RESULTS & DISCUSSIONS

The simulation results for the 8B/10B encoder has been presented in Fig 2.,Fig. 3 along with the comments in the transcript window, which states that the encoder output and disparity matches also it matches to the decoder. The Dataflow in circuit interconnects as shown in Fig. 4 was also verified. Furthermore when complete Code Analysis was done for the encoder and coverage report as shown in Fig.5 was generated it is found that the coverage is less than 100%.

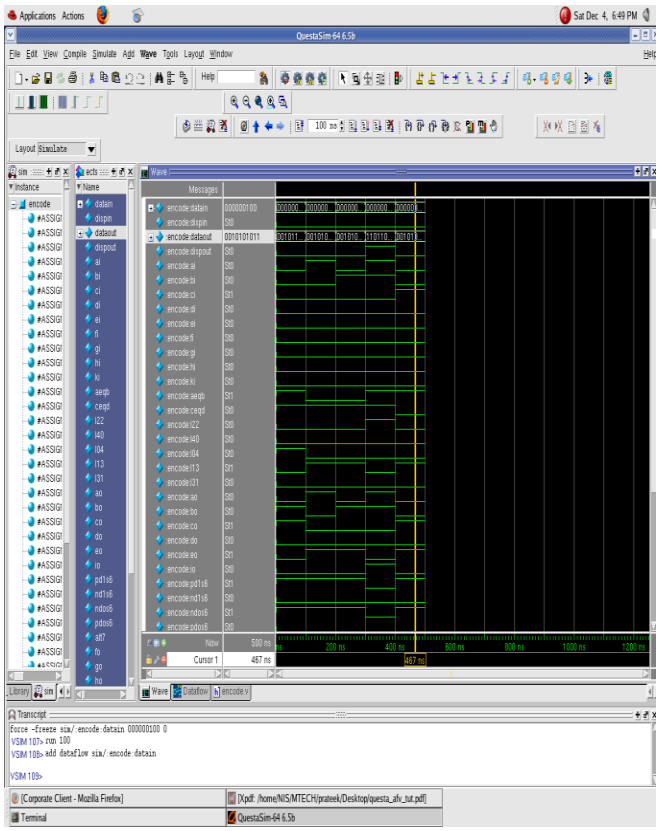

Fig 2: Simulation and timing analysis

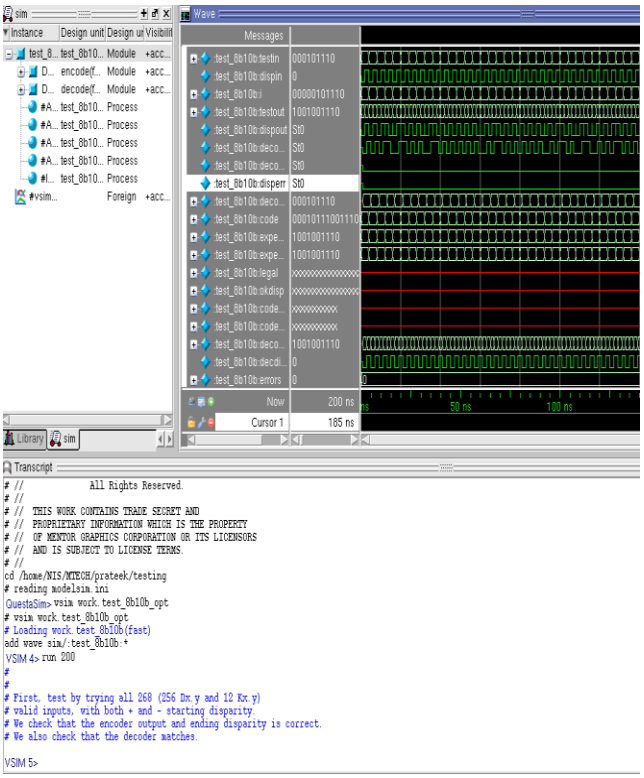

Fig 3: Test matching of encoder circuit.

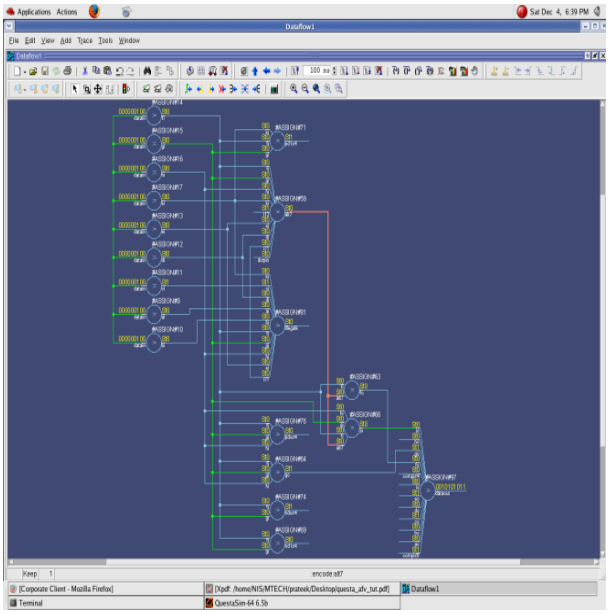

Fig 4: Circuit interconnects

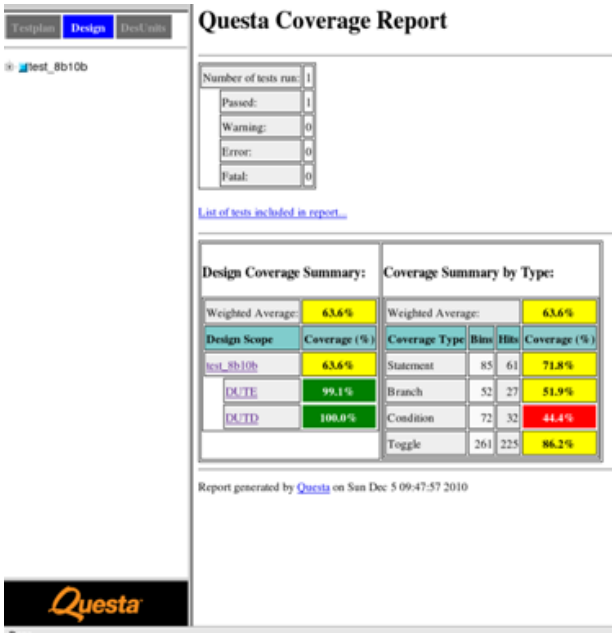

Fig 5: Coverage Report

5. CONCLUSIONS

The paper presents the code coverage details of an 8B/10B encoder RTL code. The encoder IP is verified for Code coverage using Questasim. During the process of code coverage it is found that though the encoder gave the expected output, still its overall code coverage is not 100%, indicating that all paths of design are not used to their full strength. So it is clear that higher order complete verification has to be taken up to reduce the errors in the design and increase the efficiency of the 8B/10B encoder IP which will make it robust in the design of communication system on chip.

6. REFERENCES

[1] M. E. Radu, S. M. Sexton, "Integrating extensive functional verification into digital design education," IEEE Trans. On Education, Vol. 51, No. 3, August 2008.

[2] A. X. Widmer, P. A. Franaszek, "A DC-balanced, partitioned-block 8b/10b transmission code,"IBM Journal of Reasearch and Development, vol. 27, no. 5, pp. 440, 1983.

[3] M.S.Suma and S.S.Rekha, "16B/20B CODEC Development and its ASIC Implementation." in Proc. International Conference on Future Computer and Communication, pp.622-626, April,2009.

[4] O.Cadenas, E.Todorovich, "Experiences applying OVM 2.0 to an 8B/10B RTL design," in Proc. SPL. 5th Southern Conference on Programmable Logic, pp. 1-8, April,2009.

[5] http://www.systemverilog.org/products/products_solu.html

[6] M. Mintz, R. Ekendahl, "Hardware verification with SystemVerilog: An object-oriented framework", Springer ISBN-10: 0-387-71738-2, 2007.

[7] http://www.opencores.org/projects.cgi/web/8b10b_encdec/ overview;

[8] http://asics.chuckbenz.com/

[9] B. Wile, J. C. Goss and W. Roesner,Comprehensive Functional Verification", Morgan Kaufmann, ISBN-10: 0-12-751803-7, Elsevier 2005.

[10] Cadence Designs Systems and Mentor Graphics Inc., "Open Verification Methodology User Guide" Version 2.0, Sept.2008 available from http://www.ovmworld.org

[11] J.Bergeron,"Writing Testbenches Using System Verilog"Springer, ISBN-10: 0-387-29221-7, Business Media, 2006.

[12] http://www.doulos.com/knowhow/sysverilog/ovm