Semiconductor Physics, Quantum Electronics & Optoelectronics. 2003. V. 6, N 3. P. 357-364.

PACS:

Error detection and debugging on information in communication system using single electron circuit based binary decision diagram

A.K. Biswas, S.K. Sarkar

Dept. of Electronics and Telecommunication Engineering, Jadavpur University, Kolkata-700032, India
e-mail: su_Sarkar@hotmail.com

Abstract. In digital information error happens in a communication system due to path delay or processing error/delay and can be detected by logical circuit which has been implemented here by binary decision diagrams with single electron movement from root node to leaf node. Binary decision diagrams are the representations of logic functions factored recursively with respect to input variables. Errors in the received information can be detected by the error detection circuit consisting of binary decision diagram circuits. In this technique a maximum errors of four bits can be detected in a received information of 32 bit length. However, by rearranging the received bit patterns it is possible to detect the error(s) in any bit of the received information. Every message signal is transmitted with a tag value. After detecting the error and removing the tag bit(s), the receiver finds out the corrected information.

Keywords: error detector, electron transport, tag value, output interface, tag-position eraser.

Paper received 19.05.03; accepted for publication 16.06.03.

1. Introduction

Binary Decision Diagram (BDD) method has widely been used for synthesis, analysis, optimization and minimization of both combinational as well as sequential circuits [1–5]. It has been widely used for design verification also. This BDD technique provides a useful model to allow logic designers to think about optimization and synthesis of large operations [6–8]. In the present work, BDD circuits are used for error detection and debugging of information in communication system. Tag values are also determined by separate BDD circuits. Data signal is transmitted with tag value. Tag is a combination of bits arranged in a predetermined fashion. The transmitter and receiver must agree with this tag value and positions of the bits of the tag. Transmitted signal at the receiver is checked if there is any error. If so, the receiver detects error(s) and corrects it (them) and finally corrected information is obtained. All the BDD circuits used in a module in Figs 10, 11 and 12 must be manipulated by controlling the transport of individual electrons [9–11].

In the following section, we shall stare at the definition of single electron BDD device, output interface, and other necessary BDD circuit modules [12]. We shall design X-OR, even or odd parity for 4, 5 and 6 variables and register (module-1). With the help of these X-OR, register, and even or odd parity logic circuits, desired error detector module is also constructed.

The electron can move through the BDD circuit with nearly electronic speed. Hence, the expected speed of operation of the designed "Error Detection and Debugging on Information in Communication System Using Single Electron Circuit Based Binary Decision Diagram" is as fast an electron [13].

Finally, we shall represent an algorithm that detects error, points the error positions, corrects the error of signal received and the algorithm that presents us an error-free message.

2. Definitions

Definition 1. A BDD is a rooted, directed graph, composed of many nodes and two terminals with each node labeled by a variable.

Definition 2. A binary decision diagram K having root vertex v denotes a function f_v defined recursively as:

I. If v is a terminal vertex
(a) If value $v = 1$, then $f_v = 1$
(b) If value $v = 0$, then $f_v = 0$

II. If v is a non-terminal vertex with index $(v) = i$ then $f_v(x_1,x_2,\ldots,x_n) = \bar{x}_i \cdot f_{\mathrm{low}(v)}(x_1,x_2,\ldots,x_n) + x_i \cdot f_{\mathrm{high}(v)}(x_1,x_2,\ldots,x_n)$, where x_i is called the decision variable for the vertex v. A non-terminal vertex has an attributes an arguments index index$(v) \in \{1,2,\ldots,n\}$ and two children low(v), high$(v) \in V$; where V indicates two types of vertex set.

Definition 3. A node, in a BDD, consists of four tunnel junctions $(J_1, J_2, J_3$, and $J_4)$ and three capacitors (C_1, C_2, C_3, C_4), an entry branch E and two exit branches F and G and the node is driven by a voltage clock (Φ_{i-1}). A binary voltage input x_i (and its compliment $\bar{x}_i$) used as variable is applied to point C (and B) through the capacitor C_3. The operation of this mode is:

I. It accepts an electron from the entry branch (E) and transmits the electron to any of the exit branches depending upon the input variable(s). If input variable is high, the electron follows branch 1 else branch 0.

II. The logical value of this mode is 1 if the electron passes through $\boxed{1}$ terminal and 0 if passes through $\boxed{0}$ terminal.

3. Explanation

All of the error detection/correction circuits are composed of the BDD circuits, buffers and output interfaces. To transfer the messenger electron in any circuit, four-phases (Φ_0 through Φ_3) clock is applied to node(s) and buffer(s). The phase-shifts of the clock are $\Phi_0 = 0$, $\Phi_1 = \pi/2$, $\Phi_2 = \pi$, and $\Phi_3 = 3\pi/2$. Hence, the desired result is available at the output of any expected circuit after some clock cycles. The input bit signals of a BDD circuit are applied in sequence to the nodes such that the bit signal for a BDD device is applied synchronously with the clock pulse for the consecutive BDD device(s) (or the consecutive buffers).

For successful operation in such a situation, buffers must be set up on the appropriate points on the path to ensure that a messenger electron will reach at BDD circuit (or will exit from BDD-circuit) at the correct time. Only one electron has been used in each BDD circuit for one or more clock cycle(s). The output signals of output interfaces have been used as the input signals of other BDDs (if necessary) with clock pulses. Where which pulses are necessary is shown in Figs 6, 8 and 9. As the electron moves from one node to another with consecutive clock pulses and the output of a BDD is acting as input of another BDD circuit, so pipelined operations are happening here.

4. Configuration of the output interface

The configuration of the output interface (symbol: 1-terminal 0-terminal ⟹▷— output) has been described and its operation has been explained below. The Fig. 7(a) shows the internal configuration of output interface. It contains six tunnel junction capacitances and six capacitances (their values are depicted in the Fig. 7(a)). It has two input terminals: input and *input* and two output terminals: output1 and *output*1. The output1 terminal is connected in series with 5 mV dc voltage.

The 1-terminal and 0-terminal of a BDD circuit are connected, respectively, to the input and *input* terminals of the output interface. If an electron reaches at the 1-terminal, the input potential becomes negative and the output1 voltage becomes approximately 10 mv and the output of the output interface is approximately 5 mv. If the electron reaches at 0-terminal then output1 will be approximately 0 (zero) and the output is negative (–5 mv). The input and output waveforms of output interface are shown in Fig. 7(d).

5. How fast is the error detection based on single-electron bdd devices?

A BDD device works with input signal(s) $[a_0, a_1, a_2,\ldots,]$ and a messenger electron. Each circuit (device) possesses some nodes and some buffers (if necessary). There must have some processing delays of such nodes and buffers but delays differ from node to node as well as buffer to buffer. The total delay of a BDD device is the sum of all such delays introduced by the nodes or buffer through which a messenger electron is passed. Such delays are insignificant and hence may be ignored. Following Heisenberg's uncertainty principle, it is predicted that the speed power product of an electron lie close to the quantum limit. The electron can move through the BDD circuit with nearly electronic speed. Hence, the expected speed of computation of the designed "Error Detection and Debugging on Information in Communication System Using Single Electron Circuit Based Binary Decision Diagram" based on single electron BDD circuits is as fast an electron.

6. Error detection by BDD circuit

By this BDD circuit (shown in Fig. 10) the message received by the receiver can be detected but cannot be corrected. Firstly, the special data of 5 bits (10011) must be known to transmitter and receiver. The length of the message to be sent (suppose it is of n-bit length) is increased by (5–1 = 4) four 0 bits (0000) after LSB. This (n + 4) length data bit is divided by special data 10011. In this case the long division is carried out the same way as it is done in binary system, but the subtraction is done by modulo-2 rule. In this modulo-2 rule, there are no carry for addition or borrow for subtraction, both addition and subtraction are identical to Exclusive-OR. In this division if the remainder becomes zero, there is no error in the message of n bits. If the remainder is non-zero, there is an error in the message. This process is implemented by BDD circuit shown in Fig. 10. First of all, all the registers (Fig. 9) are zeroized. Then the data bit stream (n bit) to be transmitted is shifted through the register, and its final state called the remainder, becomes the error check field. When the data and error check field (n + 4 bit long) are shifted through the receiver's BDD circuit (same as Fig. 10), a final state of zero of all register MSB X_2 X_3 LSB (0000) indicates there is no error in the message, else there are some errors. One input–output results of the error detector circuit depicted in the Table 1.

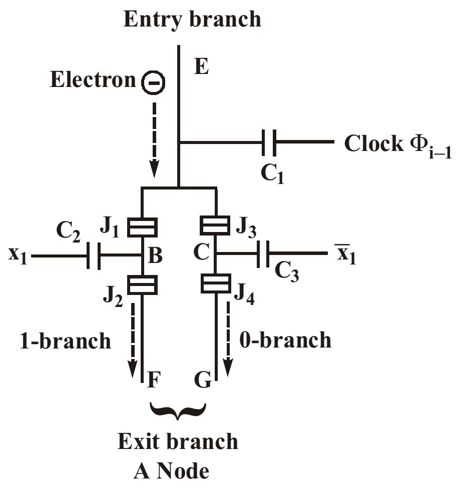
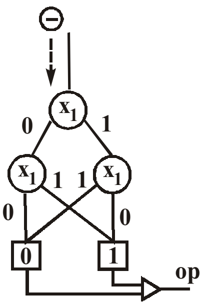


Fig. 1. Indicates a node of a BDD circuit.



x_1	x_2	op
0	0	0
0	1	1
1	0	1
1	1	0

X-OR

Fig. 2. An X-OR BDD circuit with truth table.

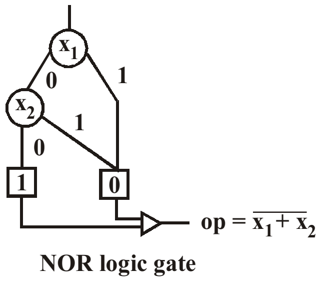


Fig. 3. A NOR logic circuit of two variables.

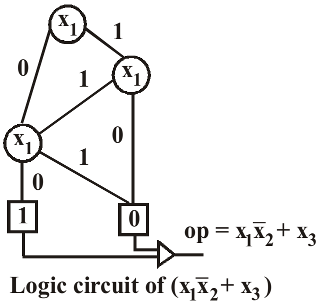

x_1
1
0
1
x_1
0
x_1
1
0
1
0
op = $x_1\overline{x}_2 + x_3$
Logic circuit of $(x_1\overline{x}_2 + x_3)$

Fig. 4. A logic circuit of $X_1\ \overline{X}_2 + X_{3.}$

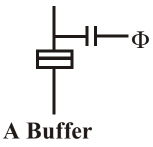
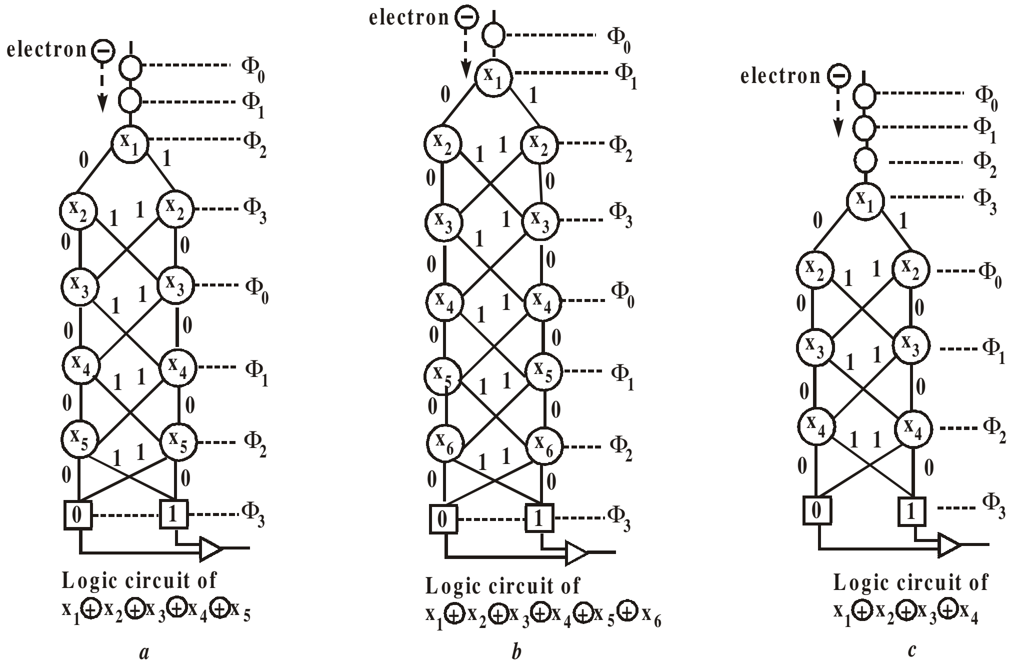


Fig. 5. A buffer.



Logic circuit of $x_1 \oplus x_2 \oplus x_3 \oplus x_4 \oplus x_5$

Logic circuit of $x_1 \oplus x_2 \oplus x_3 \oplus x_4 \oplus x_5 \oplus x_6$

Logic circuit of $x_1 \oplus x_2 \oplus x_3 \oplus x_4$

Fig. 6. *a* – indicates an odd parity checker of 5 variables, *b* – an even parity of 6 variables and *c* an even parity of 4 variables respectively.

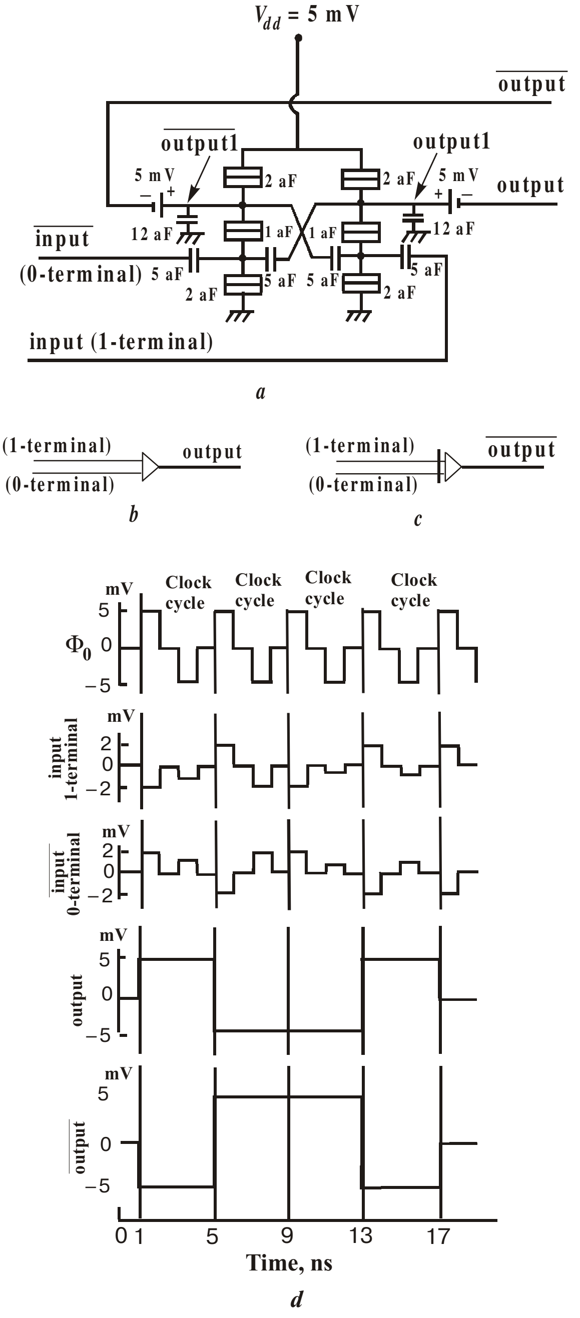


Fig. 7. *a* – shows internal connections of output interface, *b* – symbol and *c* complement symbol respectively, *d* – input and output waveforms of output interface.

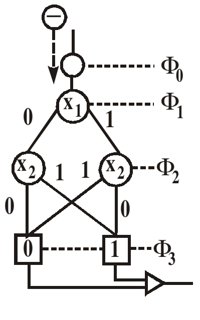


Fig. 8. X-OR BDD circuit of two variables.

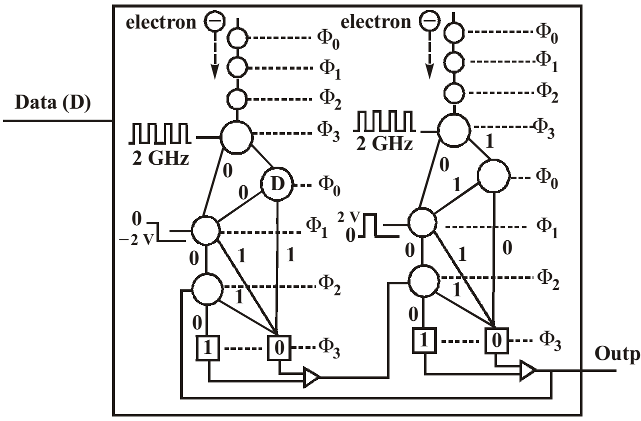


Fig. 9. A register that processes data after every 2 nanoseconds.

Table 1.

Sl.	clock	data	X_4	MSB	X_1	X_2	X_3	LSB
1		0	0	0	0	0	0	0
2		1	1	0	1	0	0	0
3	↑	1	1	1	0	1	0	0
4		0	0	1	1	1	0	0
5	↑	0	0	0	0	1	1	0
6		0	0	0	0	1	1	0
7	↑	0	1	0	1	0	1	1
8		1	0	0	0	0	1	1
9	↑	1	0	0	0	0	0	1
10		1	0	0	0	0	0	1
11	↑	1	1	0	1	0	0	0
12		1	1	0	1	0	0	0
13	↑	1	1	1	0	1	0	0
14		0	0	1	1	1	0	0
15	↑	0	0	0	0	1	1	0
16		1	1	0	1	1	1	0
17	↑	1	0	1	1	1	1	1
18		0	1	1	0	1	1	1
19	↑	0	1	1	0	0	1	1
20		0	1	1	0	0	1	1
21	↑	0	1	1	0	0	0	1
22		1	0	1	1	0	0	1
23	↑	1	1	0	1	1	0	0
24		1	1	0	1	1	0	0
25	↑	1	1	1	0	1	1	0
26		0	0	1	1	1	1	0
27	↑	0	1	0	1	1	1	1
28		1	0	0	0	1	1	1
29	↑	1	0	0	0	0	1	1
30		1	0	0	0	0	1	1
31	↑	1	0	0	0	0	0	1
32		1	0	0	0	0	0	1
33	↑	1	1	0	1	0	0	0

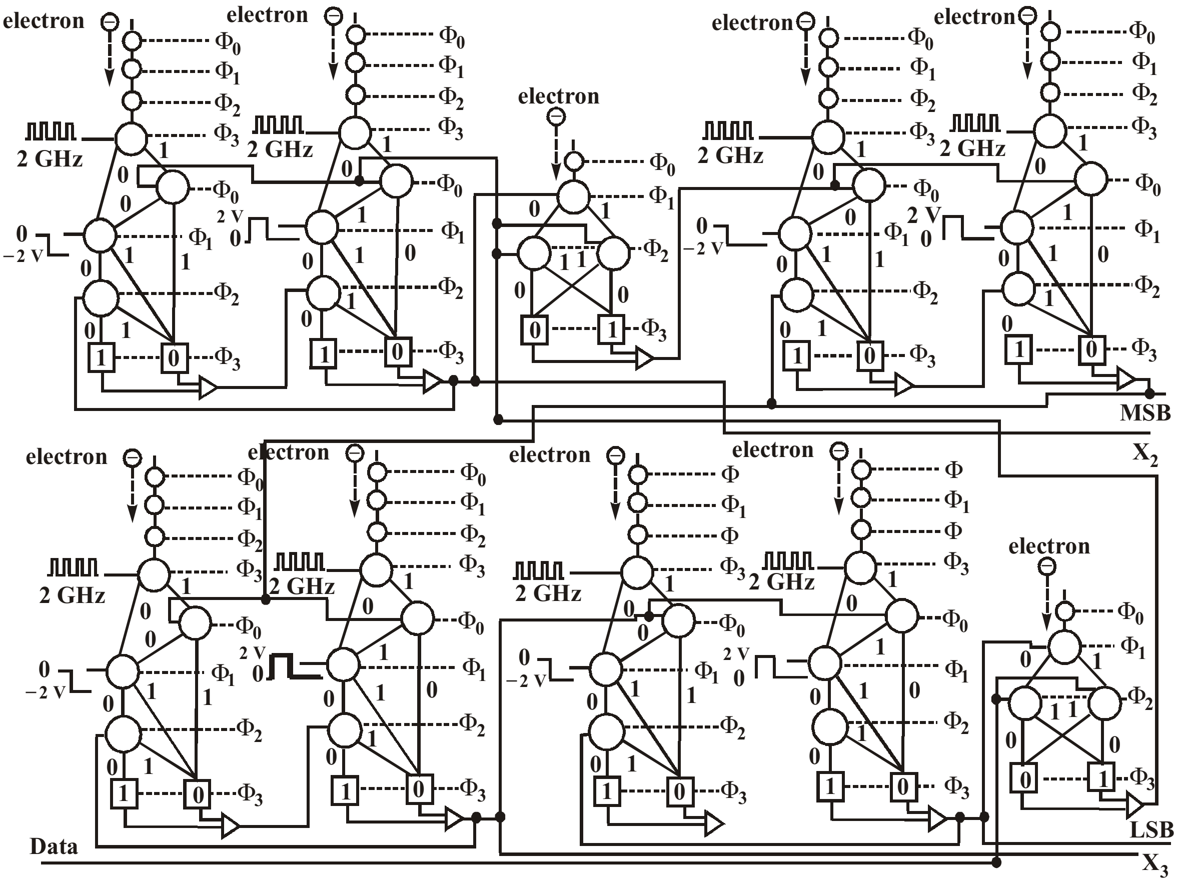


Fig. 10. A BDD circuit that detects the error(s), if any, of an information.

Example: Modulo-2 division.

Suppose the message data is 1 0 1 1 1 0, the special data is 1 0 0 1 1, then the division procedure is:

```
                       1 0 1 0 0 1
           1 0 0 1 1 | 1 0 1 1 1 0 0 0 0 0
                       1 0 0 1 1
        Exclusive-OR   0 0 1 0 0 0 0
                           1 0 0 1 1
        Exclusive-OR   0 0 0 1 1 0 0 0
                                 1 0 0 1 1
        Exclusive-OR             0 1 0 1 1
```

7. Error detection and error correction

To detect and correct the transmitted data we take the following steps:

I. Suppose the data to be transmitted be $I_1 I_2 I_3 I_2 I_3 I_4 I_5 I_6 I_7 I_8$. These data bits are processed through the circuit shown in Fig. 11. From this circuit we get tag value $P_1 P_2 P_3 P_4$. Now we arrange the signal to be transmitted in the manner: P_1 P_2 I_1 P_3 I_2 I_3 P_4 I_4 I_5 I_6 I_7 I_8. This arranged data is transmitted through the transmitted media.

II. When the transmitted signal is arrived at receiver, the receiver processed the signal and gets the four bit data C_1 C_2 C_3 C_4 from the Fig. 12. This four bit data determines the error position. After finding the error bit positions, the error bit(s) is (are) corrected. As the tag value position is known, except the tag position the bits are collected and pure data bits are found out. In Table 2 the simulated results are given.

8. Algorithm

I. Write input data to be transmitted.

II. Perform X-OR operation on input data for finding tag value. Arrange the data with tag value in a manner T_1 T_2 I_1 T_3 I_2 I_3 T_4 I_4 I_5 I_6 I_7 I_8 I_9 I_{10} I_{11} T_5... (where T_i indicates tag bit and I_i input data).

III. X-OR operation is to be performed on the arrived data for calculating C_1 C_2 C_3 C_4... for determining the error position(s), if any. If so, do complement of all data of the error position(s).If no error occurs all bit(s) of C_i will be zero.

IV. Delete the tag position bit(s) to find corrected data.

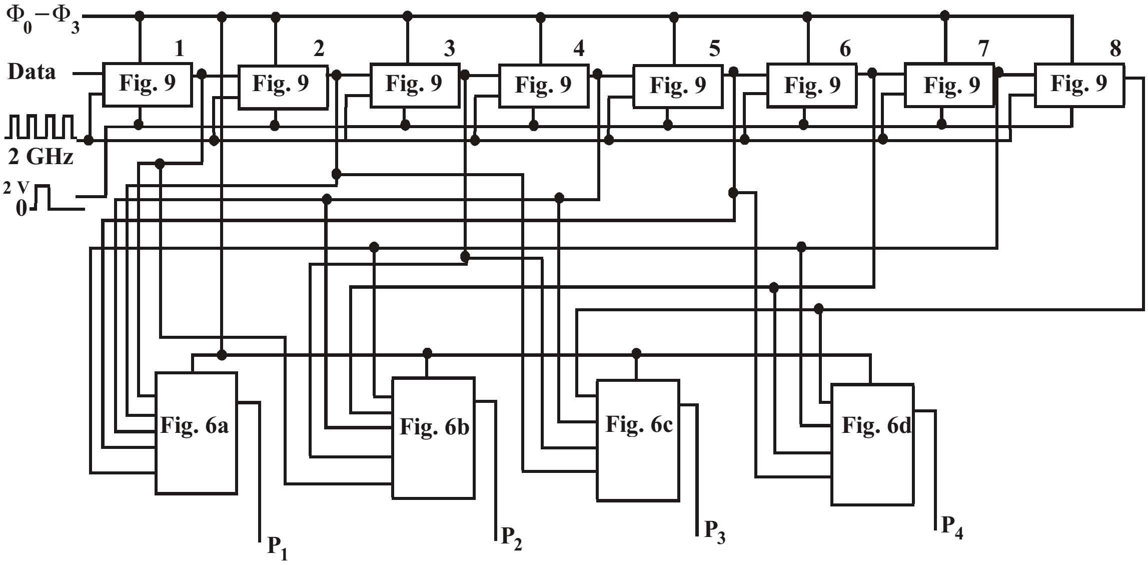


Fig. 11. This is another BDD circuit which determines the tag value(s).

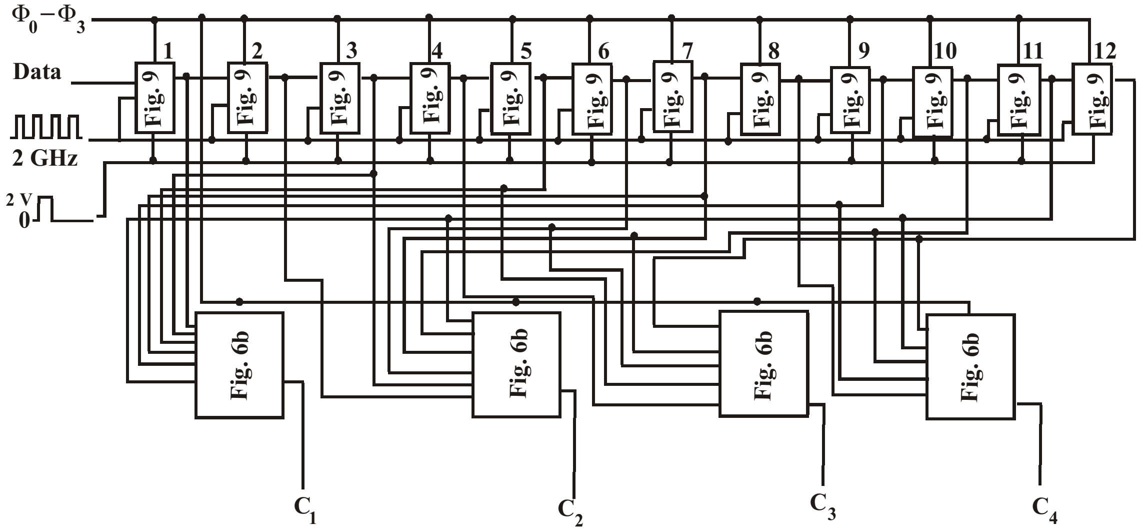


Fig. 12. This BDD circuit finds the error position.

9. Conclusion

This paper is basically concentrated at 'error detection' and 'error correction' with the help of single electron logic systems based on binary decision diagram. The error detection module, tag value module are designed by combining the single electron BDD devices. The outputs of the modules are shown. The designed modules produce output data flow in response to the input data flow through pipelined processing. In the computer simulation process, 32-bits data are considered as data. But any number of bits for a data can be simulated in the same manner, if necessary. The error detection and correction has been simulated. The thermal agitation is excluded here, because thermal agitation causes an operation error caused by an "unexpected tunneling" which means a tunneling that does not follow the BDD path [6].

Acknowledgement

S. K. Sarkar thankfully acknowledges the financial support obtained from University Grants Commission vide order no. F. 14-32 2000 dated 12th October 2000.

Table 2.

Sl.No.	Description of data	Data value
1	Message data	00000001 00000010 00000011 00000100
	Tag value	0011 1101 1110 0101
	Transmitted data	000100010001 110000010010 110100000011 010000010100
	Data received	100100010001 010000010010 010100000011 110000010100
	Error position(s)	1,13,25,37
	Data corrected	000100010001 110000010010 110100000011 010000010100
	Actual corrected message data	00000001 00000010 00000011 00000100
2	Message data	00000101 00000110 00000111 00001000
	Tag value	0110 1000 1011 1001
	Transmitted data	010100000101 100000000110 100100010111 100000011000
	Data received	010100000100 000000000110 100100010011 100000001000
	Error position(s)	12,13,34,44
	Data corrected	010100000101 100000000110 100100010111 100000011000
	Actual corrected message data	00000101 00000110 00000111 00001000
3	Message data	00001001 00001010 00001011 00001100
	Tag value	1010 0100 0111 1100
	Transmitted data	100100001001 010000001010 010100011011 110000001100
	Data received	100000001001 000000001010 000100011011 110000001000
	Error position(s)	4,14,26,46
	Data corrected	100100001001 010000001010 010100011011 110000001100
	Actual corrected message data	00001001 00001010 00001011 00001100
4	Message data	00001101 00001110 00001111 00010000
	Tag value	1111 0001 0010 1110
	Transmitted data	110100011101 000000011110 000100001111 110100100000
	Data received	110000011101 000000001110 000000001111 110100000000
	Error position(s)	4,20,28,43
	Data corrected	110100011101 000000011110 000100001111 110100100000
	Actual corrected message data	00001101 00001110 00001111 00010000
5	Message data	00010001 00010010 00010011 00010100
	Tag value	1101 0011 0000 1011
	Transmitted data	110000110001 000100110010 000000100011 100100110100
	Data received	110000100001 000000110010 000000000011 100000110100
	Error position(s)	8,16,31,40
	Data corrected	110000110001 000100110010 000000100011 100100110100
	Actual corrected message data	00010001 00010010 00010011 00010100

References

1. N. Asahi, M. Akazawa, Y. Amemiya, Single-electron Logic System Based on the Binary Decision // *IEICE.Trans, Electron*, **E81-c**(1), January 1998.
2. K. Nakazato, Single-ElectronSwitch for phase- locked single-electron logic devices // *in IEDM Tech. Dig.*, 1992, pp. 487-490.
3. N. Kuwamura, K. Taniguchi, and C. Hamaguchi, Simulation of single-electron logic circuits // *IEICE Trans*, **J77-C-II**(5), pp. 221-228, May 1994.
4. J.E. Mooji ext. abstract SSDM 93 p. 339.
5. Neil M. Zimmerman, Mark W. Keller // *Journal of Applied Physics*, **87**(12), pp. 8570-8574, 15 June 2000.
6. N. Asahi, M. Akazawa, Y. Ameniya, Single-electron logic device based on the binary decision diagram // *EEE Trans. Electron Devices*, **44**(7), pp. 1109-1116 (1997).

7. N. Asahi, M. Akazawa, Y. Amemiya, Binary-decision-diagram device // *IEEE Trans. Electron Devices*, **42**(11), pp. 1990-2003, 1995.
8. A.K. Biswas, A.K. Dey, B. Maji, S.K. Sarkar, Single-electron Transport and Realization of a Sequence Generation circuit // *Proceeding of the International workshop on Physics of semiconductor devices*, **2**, p. 1275 (2001).
9. K. Nakazato, Single-electron switch for Phase-locked single-electron logic devices // *in IEDM Tech. Disest*, p. 487-490 (1992).
10. L.J. Geeligs, V.F. Anderegg, P.A.M. Holweg, J.E. Mooij, H. Pothier, D. Esteve, C. Urbina, M.H. Devoret, Frequency-locked turnsile device for single electrons // *Phys. Rev. Lett.*, **64**, pp. 2691-2694 (1990).
11. H. Pothier, P. Lafage, C. Urbina, D. Esteve, M.H. Devoret, Single-electron pump based on charging effects // *Europhys. Lett.*, **17**, pp. 249-254 (1992).
12. S. Aborhey, Binary decision graph reduction // *IEE Proc. part-E*, **136**(4), pp. 277-283 (1989).
13. A.K. Biswas, Subir Kumar Sarkar, An arithmetic logic unit of a computer based on single-electron transport // *Semiconductor Physics, Quamtum Electrtonics & Optoelectronics*, **6**, pp. 91-96 (2003) (in press).