



# Application of single electron threshold logic gates and memory elements to an up-down Counter

Anup Kumar Biswas, Assistant Professor,  
Department of Computer Science and Engineering  
Kalyani Govt. Engineering College, Kalyani,  
Nadia-741235, West Bengal, India

## ABSTRACT

Low cost, low power consumption, high operating speed, and high integration density-based electronic goods are economically requisite in business, engineering, science and technology in the present era. Single Electron tunneling based threshold phenomenon is one approach by which all the logic gates, combinational and sequential circuits can be implemented. Single Electron tunneling devices (SEDs) and Threshold Logic Gates (TLGs) have the capabilities of controlling the transport of only an electron through a tunnel junction at a time. A single electron bearing the charge is sufficient to store an information in a SED in the atmosphere of 0K. Power required in the single electron tunneling circuits is very low in comparison with the (CMOS) circuits. The speed of the processing of TLG based devices will be very near to electronic speed. The single-electron transistor (SET) and TLG both attract the scientists, technologists and researchers to design and implement their large scale circuits for small cost of the ultra-low power and its small size. All the tunneling events in the case of a TLG-based circuit happen when only a single electron tunnels from one conductor to other through the tunnel junction under the proper applied bias voltage and multiple input voltages. For implementing an up-down converter, TLG would be a best candidate to fulfill the necessities requiring for its implementation. So far as the Ultra-low noise is concerned, TLG based circuit would be a best selection for implementing the desired tunneling circuits. Different TLGs like 2- or 3- input AND and OR, RS Flip-flop and T-Flip-flop, an UP-Down counter are implemented by using linear or non-linear threshold logic gates or devices. And their corresponding truth table or simulated results are given in due places.

**Key words:** up-down Counter, Electron-tunneling, Coulomb-blockade, linear threshold gate

## 1. INTRODUCTION

It is transparent for the persons who are involved in doing research in semiconductor technology that the ever decreasing feature size and the corresponding increase in density of transistors facilitated many improvements in semiconductor based designs. We would be able to realize that such improvement will eventually come to an end. For ensuring further feature size reduction, possible coming technologies with greater scaling potential like electron tunneling technology are currently under the investigation of the researchers. Single electron tunneling based circuits or threshold logic gate based circuits are concentrated to tunnel junctions, through which individual electrons can be passed through in a controlled manner.

Single Electron tunneling based device is one such equipment by which all Boolean logic gates and more complicated circuits can be implemented. Tunneling events happen for TLG-based circuits if only a single electron passes the tunnel junction under the proper bias voltage and multiple input voltages connected to the islands via small capacitors. For implementing an Up-down Counter, TLG would be a suitable candidate for this case. Multiple input logic gates, memory cells are implemented, and based on them our desired counter is constructed and is verified by simulation.

## 2. Multiple input threshold logic gate

A multiple input threshold logic gate [1, 2, 3, 8, 9,10] is made up of a tunnel junction having its capacitance  $C_j$  and resistance  $R_j$ , two multiple input-signals  $V_k^P$ 's and  $V_l^n$ 's connected at points 'p' and 'q'. Each input voltage  $V_k^P$ , for the top left side is connected to the point "q" through the capacitors  $C_k^P$ 's and each input voltage  $V_l^n$ , for the bottom left side, is connected to the point "p" through the capacitors  $C_l^n$ 's. Bias voltage  $V_b$  is directly connected to point the "b" through a capacitor  $C_b$  also. Junction capacitor is connected to point "p" which is grounded through a capacitor  $C_0$ . This multiple input TLG can also be called a Junction-Capacitor (C-J) circuit. Using the Junction-Capacitor circuit we will be able to implement the LTG which can be presented by the signum function of  $h(x)$  expressed by equations (1) and (2).

$$g(x) = \text{sgn}\{h(x)\} = \begin{cases} 0, & \text{if } h(x) < 0 \\ 1, & \text{if } h(x) \geq 0 \end{cases} \quad (1)$$

$$h(x) = \sum_{k=1}^n (w_k \times x_k) - \theta \quad (2)$$

where  $x_k$  being the  $n$  Boolean inputs and  $w_k$  being their corresponding  $n$  integer weights.

The LTG compares the weighted sum of the inputs  $\sum_{k=1}^n (w_k \times x_k)$  with the threshold value  $\theta$ . If the weighted value of the sum becomes greater than or equal to the threshold or critical voltage value then the logic output of the LTG will be high (logical "1"), otherwise it will be logical "0".

The tunnel junction capacitance  $C_j$  and the output capacitance  $C_0$  are considered to be the two basic circuit elements in a LTG. The input signal vector elements ( $V_1^P, V_2^P, V_3^P, \dots, V_k^P$ ) are weighted by their corresponding vector capacitances ( $C_1^P, C_2^P, C_3^P, \dots, C_k^P$ ) and added to the junction voltage,  $V_j$ . Whereas, the input signal vector elements ( $V_1^n, V_2^n, V_3^n, \dots, V_l^n$ ) are weighted by their corresponding vector capacitances ( $C_1^n, C_2^n, C_3^n, \dots, C_l^n$ ) are being subtracted from the voltage,  $V_j$ .

The critical voltage  $V_c$  is essential to enable tunneling action, and which acts as the intrinsic threshold of the tunnel junction circuit. The bias voltage  $V_b$  connected to tunnel junction through the capacitance,  $C_b$ , is used in order to adjust the gate threshold to the desired value  $\theta$ . When a tunneling happens though the tunnel junction an electron goes through the junction from p to q as shown in Fig. 1.

The following notations are being used for the rest our discussion.

$$C_{\Sigma}^P = C_b + \sum_{k=1}^g C_k^P \quad (3)$$

$$C_{\Sigma}^n = C_0 + \sum_{l=1}^h C_l^n \quad (4)$$

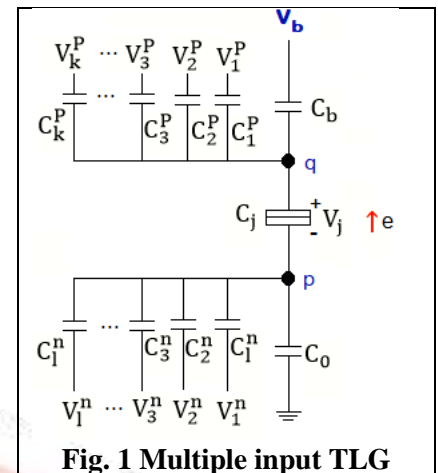
$$C_T = C_{\Sigma}^P C_j + C_{\Sigma}^P C_{\Sigma}^n + C_j C_{\Sigma}^n \quad (5)$$

When we are thinking of that all voltage sources in Fig. 1 are connected to ground, then the circuit can be considered to be made up of three capacitors namely,  $C_{\Sigma}^P$ ,  $C_{\Sigma}^n$  and  $C_j$ , connected in series. Here,  $C_T$  is thought of the sum of all 2-term products of these three capacitances  $C_{\Sigma}^P$ ,  $C_{\Sigma}^n$  and  $C_j$ .

It is the time to find the expression regarding the critical voltage  $V_c$  of the tunnel junction. We assume the capacitance of the tunnel junction to be  $C_j$  and the remainder of the circuit having the equivalent capacitance to be  $C_e$ , as observed from the viewpoint of tunnel junction, we can measure the threshold or critical voltage [7, 8] for the tunnel junction as

$$V_c = \frac{e}{2(C_j + C_e)} \quad (6)$$

$$V_c = \frac{e}{2[C_j + (C_{\Sigma}^P || C_{\Sigma}^n)]}$$



$$\begin{aligned}
 &= \frac{e}{2[C_j + \frac{(C_\Sigma^P) * (C_\Sigma^n)}{(C_\Sigma^P + C_\Sigma^n)}]} \\
 &= \frac{e(C_\Sigma^P + C_\Sigma^n)}{2[C_j * (C_\Sigma^P + C_\Sigma^n) + (C_\Sigma^P) * (C_\Sigma^n)]} \\
 &= \frac{e(C_\Sigma^P + C_\Sigma^n)}{2C_T} \dots\dots\dots(7)
 \end{aligned}$$

When the voltage across the junction is  $V_j$ , a tunnel event will happen through this tunnel junction if and only if the condition given below is satisfied.

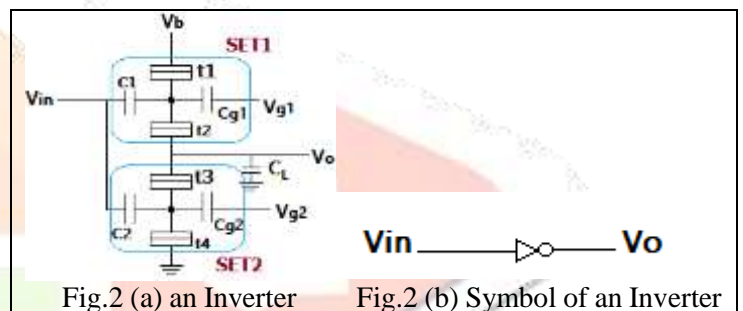
$$|V_j| \geq V_c \dots\dots\dots(8)$$

This indicates that if the junction voltage is less than the critical voltage i.e.  $|V_j| < V_c$ , there will be no tunneling events through the tunnel junction. As a consequence, the tunneling circuit stays in a *stable state*.

Theoretically, the thresholds are being integer numbers. And the threshold logic equations for 2-input or 3-input logic AND, OR, NAND and NOR gates are shown in the following sections.

### 3. Buffer

The buffer or inverter [2, 3, 4, 7, 8] depicted in Fig. 2(a) is made up of two single electron transistors (SETs) connected in series. The two inputs having the same values are directly coupled to the islands of the SET1 and SET2 through two capacitors  $C_1$  and  $C_2$  respectively. The islands of each SET have a size smaller than 10 nm diameter of gold and their capacitances should be less than  $10^{-17}$  F. The output terminal  $V_0$  is connected to the common channel between SET1 and SET2 and to the ground via a capacitor  $C_L$  to put down charging effects.



For the inverter, the parameters values chosen are:  $V_{g1}=0$ ,  $V_{g2}=0.1 \times \frac{q_e}{C}$ ,  $C_L = 9C$ ,  $t_4 = \frac{1}{10}C$ ,  $t_3 = \frac{1}{2}C$ ,  $t_2 = \frac{1}{2}C$ ,  $t_1 = \frac{1}{10}C$ ,  $C_1 = \frac{1}{2}C$ ,  $C_2 = \frac{1}{2}C$ ,  $C_{g1} = \frac{17}{4}C$  and  $C_{g2} = \frac{17}{4}C$ ,  $R_1 = R_2 = 50K\Omega$ . For simulation purpose, the value of  $C$  is taken 1aF.

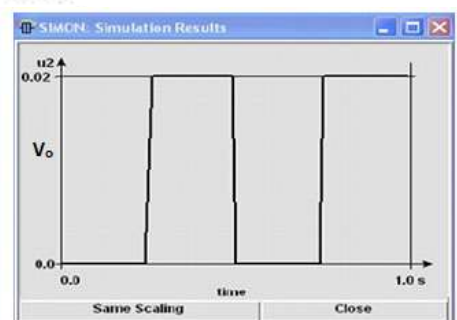
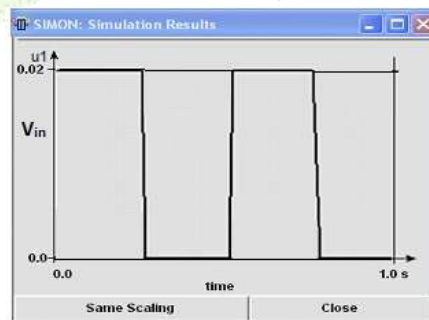
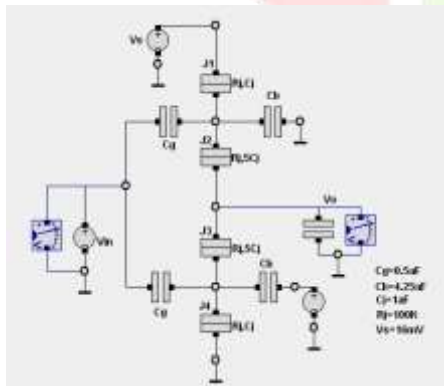


Fig. 2(c) Simulation set of Buffer

Fig.2 Simulation result of Buffer (d) input (e) output

The operation of the buffer will be described as: - the output  $V_0$  value will be high when the input voltage  $V_{in}$  is low and the  $V_0$  value will be low when the input voltage is high. To achieve this target, we set the voltages for  $V_{g1} = 0$  and  $V_{g2} = 16mV$  along with the tuning gate voltages, at present,  $V_{in}$  both for SET1 and SET2. SET1 is in conduction mode if  $V_{in}$  is set to low and the SET2 is in Coulomb blockade [2, 3, 4, 11, 15, 17]. This results the output voltage  $V_0$  is connected to  $V_b$  and therefore the output voltage becomes high. Coulomb blockade troubles the steady flow of current since as soon as the high voltage (logic 1) is applied to the input terminal, it makes shift the induced charge on each of



the islands of the two SETs by a fraction of an electron charge and enforces the SET1 in Coulomb blockade and the SET2 in conducting mode. So, the output shifts from high to low.

In this work we assume the Boolean logic inputs logic "0" = 0 Volts and logic "1" =  $0.1 \times \frac{q_e}{C}$ .

For simulation and other purposes, will consider that  $C = 1 \text{ aF}$  and Logic "1" =  $0.1 \times \frac{1.602 \times 10^{-19}}{1 \times 10^{-18}} = 0.1 \times 1.602 \times 10^{-2} = 16.02 \times 10^{-3} = 16.02 \cong 16 \text{ mV}$ .

#### 4. LOGIC GATES AND LINEAR SEPARATION

Let us see the input space for two inputs  $x_1$  and  $x_2$ , and how the basic gates are implemented by linear separation. Fig. 3 shows the functions of gates AND, OR and XOR with 4 vertices (combinations 00, 01, 11, 10) as points and red circle-points have the function value 1, and the rest of the points indicates 0 (zero) value for the function. These red-points or vertices are called 1-vertex and colorless points are called 0-vertices of the function of the gates. In Fig. 3 (a) and 3(b) and 3(c), we can draw many straight lines (in other words different weights and thresholds) to separate 1-vertices from 0-vertices and thus the functions of AND, OR and NOR can be implemented by a single neuron. This is not true for XOR in Fig. 3(d), since no line can separate the 1-vertices of XOR function from its 0-vertices. So, XOR is not a linearly separable function. In two dimensions, a line separates points in a plane, whereas a plane is a separator of points in three dimensional space. In general,  $n-1$  dimensional plane (also called hyper plane) is a separator points in  $n$ -dimensions. Small circles are considered vertices (combinations 00, 01, 10, 11) of  $x_1 x_2$ .

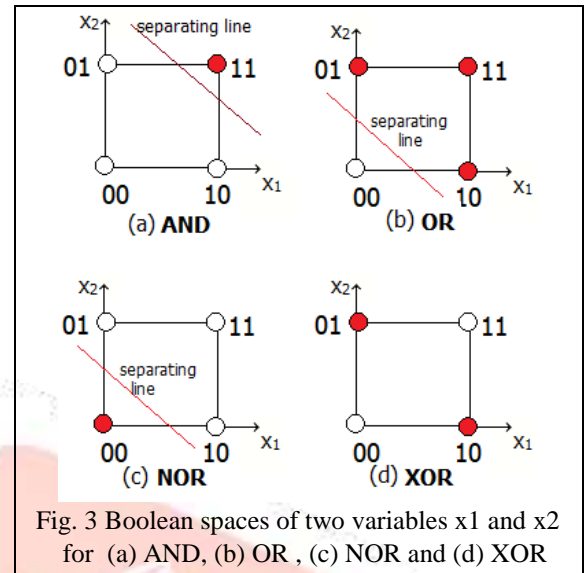


Fig. 3 Boolean spaces of two variables  $x_1$  and  $x_2$  for (a) AND, (b) OR, (c) NOR and (d) XOR

Neuron implementing a switching function when 1-vertex and 0-vertices can be separated by some straight lines (in red color straight lines). No line can separate the 1-vertices and 0-vertices in Fig. 3 (d), so XOR cannot be implemented by a neuron.

**Definition 1:** A switching function is considered to be linearly separable if and only if all of its 1-vertices (like red colored) can be separated from all of its 0-vertices with the help of a hyper plane. A neuron having  $n$ -inputs works upon an  $n$ -dimensional space and a single neuron which is capable of implementing any switching function is thought to be linearly separable.

##### Example 1:

A single neuron can implement a switching function  $f(x_1, x_2, x_3) = x_1 + x_2 x_3 + x_1 x_2$ , since its 1-vertices cannot be separated from 0-vertices by a plane as shown in Fig. 4 and its truth table is shown in Table-1.

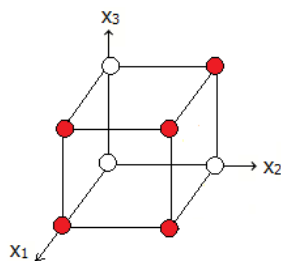


Fig. 4 Space of non-linear threshold logic function

Table-1			
$x_1$	$x_2$	$x_3$	$f(x_1, x_2, x_3)$
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	1
1	0	0	1
1	0	1	1
1	1	0	1
1	1	1	1

##### Example 2:

A single neuron can implement a switching function  $= x_1 + x_2 + \bar{x}_2 x_3$ , since its 1-vertices can be separated from all 0-vertices by a plane as shown in Fig. 5 and its truth table is shown in Table-2.

$$f(x_1, x_2, x_3)$$

Table-2

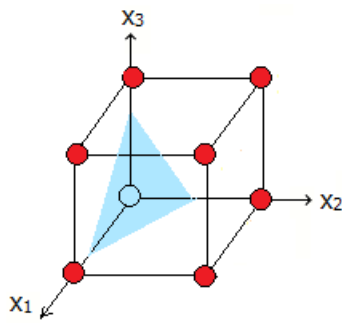


Fig. 5 Space of linear threshold logic function

Truth table of  $f(x_1, x_2, x_3) = x_1 + x_2 + \bar{x}_2 x_3$ ,

$x_1$	$x_2$	$x_3$	$f(x_1, x_2, x_3)$
0	0	0	0
0	0	1	1
0	1	0	1
0	1	1	1
1	0	0	1
1	0	1	1
1	1	0	1
1	1	1	1

**Example 3:**

A single neuron can implement a switching function  $f(x_1, x_2, x_3) = x_1 x_3 + x_2 \bar{x}_3$ , since its 1-vertices can be separated from all 0-vertices as shown in Fig. 6 and its truth table is shown in Table-3.

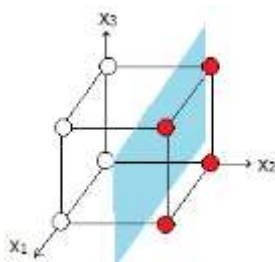
**Table-3**Truth table of  $f(x_1, x_2, x_3) = x_1 x_3 + x_2 \bar{x}_3$ 

Fig. 6 Space of a linear threshold logic function

$x_1$	$x_2$	$x_3$	$f(x_1, x_2, x_3)$
0	0	0	0
0	0	1	0
0	1	0	1
0	1	1	1
1	0	0	0
1	0	1	0
1	1	0	1
1	1	1	1

**5. Threshold logic equation for OR gate**

For making the threshold logic gate of OR gate, we draw the truth table Table-4 of OR gate and compare the weights of variables  $w_A$  and  $w_B$  of two variables A and B respectively with the threshold value  $\theta$ .

**Table-4**

A	B	F(A,B)	$\theta$
0	0	0	$0 < \theta$
0	1	1	$w_B \geq \theta$
1	0	1	$w_A \geq \theta$
1	1	1	$w_B + w_A \geq \theta$

For positive logic we assume weights of A and B are 1 each. Then from the three equations

$$w_B > \theta \dots\dots\dots (9)$$

$$w_A > \theta \dots\dots\dots (10) \text{ and}$$

$$w_B + w_A > \theta \dots\dots\dots (11)$$

As  $0 < \theta$ ,  $\theta$  must be positive, If we assume  $w_B=1$ ,  $w_A=1$  and  $\theta=0.5$ , then the three equations in (9), (10) and (11) Table-4 are satisfied. Hence the Threshold logic equation for OR gate is

$$OR(A, B) = \text{sgn}\{A + B - 0.5\} \dots\dots\dots (12)$$

For implementing the AND gate we will use the parameters  $C_1^n = C_2^n = 0.5aF$ ,  $C_3 = 11.7aF$ ,  $C_{b1} = C_{b2} = 4.25aF$ ,  $C_{g1} = C_{g2} = 0.5aF$ ,  $C_L = 9aF$ ,  $C_0 = 8aF$ ,  $R_j = 10^5 \Omega$ ,  $V_s = 16mV$  and accordingly after running the simulator the output we get is given in Fig. 5(b).

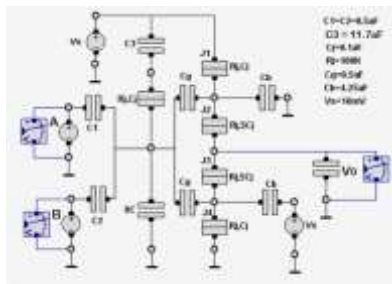


Fig. 5(a) OR gate

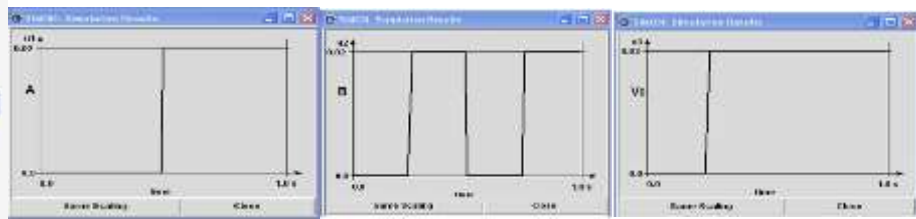


Fig. 5(b) simulation result of OR gate

## 6. 2-input AND gate

For making the threshold logic gate of AND gate, we draw the truth table Table-5 of AND gate and compare the weights of variables  $w_A$  and  $w_B$  of two variables A and B respectively with the threshold value  $\theta$  [7,8].

Table-5

A	B	F(A,B)	$\theta$
0	0	0	$0 < \theta$
0	1	0	$w_B < \theta$
1	0	0	$w_A < \theta$
1	1	1	$w_B + w_A \geq \theta$

For positive logic we assume weights of A and B are 1 each. Then from the above four inequalities if we assume  $w_B=1$ ,  $w_A=1$  and  $\theta=2$ , then the three inequalities or equations in 4<sup>th</sup> column in Table-5 are satisfied. Hence the Threshold logic equation for AND gate is

$$AND(A, B) = sgn\{A + B - 2\} \dots \dots \dots (13)$$

For implementing the AND gate we will use the parameters  $C_1^n = C_2^n = 0.5aF$ ,  $C_{b1} = C_{b2} = 4.25aF$ ,  $C_{g1} = C_{g2} = 0.5aF$ ,  $C_L = 9aF$ ,  $C_0 = 8aF$ ,  $R_j = 10^5 \Omega$  in Fig. 6(a) and accordingly after simulation the result we get is given in Fig. 6(b).

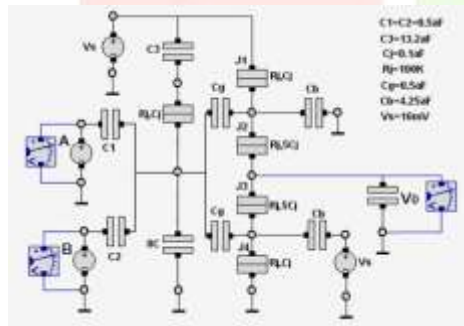


Fig.6 (a) AND Gate

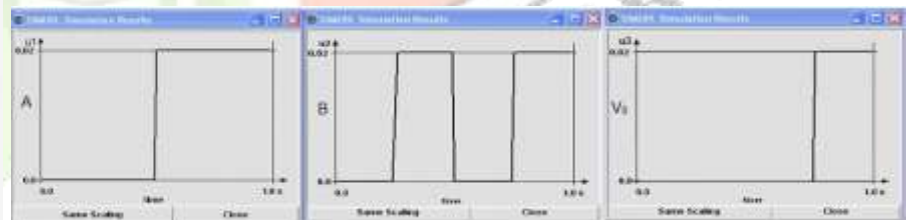


Fig.6 (b) Simulation result of AND gate

## 7. Threshold logic equation for 3-input AND gate

Table-6

A	B	C	F(A,B,C)=ABC	$\theta$
0	0	0	0	$0 < \theta$
0	0	1	0	$\theta > w_C$
0	1	0	0	$\theta > w_B$
0	1	1	0	$\theta > w_B + w_C$
1	0	0	0	$\theta > w_A$
1	0	1	0	$\theta > w_A + w_C$
1	1	0	0	$\theta > w_A + w_B$
1	1	1	1	$w_A + w_B + w_C \geq \theta$

As AND gate is a positive logic we shall assume that all the values of  $w_A$ ,  $w_B$ ,  $w_C$  and  $\theta$  are positive. If we take  $w_A = 1$ ,  $w_B = 1$ ,  $w_C = 1$  and  $\theta = 2.5$  ( or any value in the range  $2 < \theta \leq 3$ ), then all the inequality equations in the 5<sup>th</sup> column are satisfied. So the Threshold logic equation for 3-input AND gate is

$$AND(A, B, C) = sgn\{A + B + C - 2.5\} \dots \dots \dots (14)$$

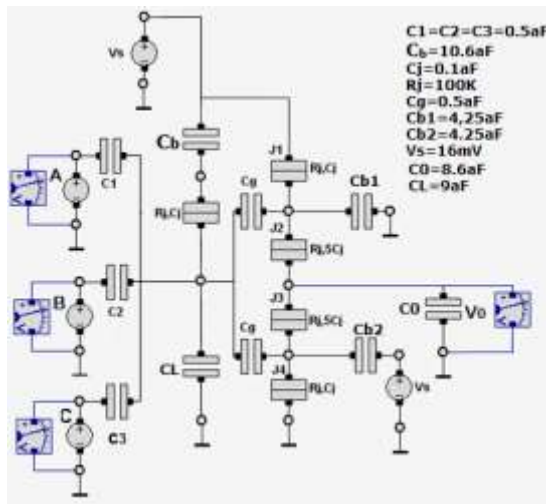
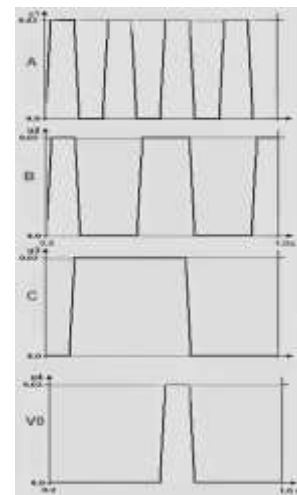


Fig. 6(c) Simulation set of 3-input AND gate



(b) Simulation result of 3-input AND gate

Simulation set of 3-input AND gate is depicted in Fig. 6(c). As said earlier, the threshold gates those are derived from the generic threshold gate given in Fig.1 require an output buffer for the correct operation in a network structure. It is known the applied buffer inverts its own input signal, we modify the threshold equation of  $AND(A, B, C)$  such that it determines  $AND(A, B, C)$ . So when we combine the result of  $AND(A, B, C)$  and its buffer, a buffered gate obtained calculates  $AND(A, B, C)$ . Truth table of 3-input NAND gate and the threshold relationships with the weights is given in Table-7

$$\begin{aligned} NAND=AND(A, B, C) &= sgn\{-A - B - C + 2.5\} \\ &= sgn\{-A - B - C - (-2.5)\} \dots \dots \dots (15) \end{aligned}$$

Table-7

Truth table of Threshold NAND gate

A	B	C	$\{w_A + w_B + w_C\}$	$\theta = -2.5$	$F(A, B, C) = \overline{ABC}$
0	0	0	0	$0 > -2.5$	1
0	0	1	-1	$-1 \geq -2.5$	1
0	1	0	-1	$-1 \geq -2.5$	1
0	1	1	-2	$-2 \geq -2.5$	1
1	0	0	-1	$-1 \geq -2.5$	1
1	0	1	-2	$-2 \geq -2.5$	1
1	1	0	-2	$-2 \geq -2.5$	1
1	1	1	-3	$-3 < -2.5$	0

From the Table-7, it is clear that equation (15) acts as the equation of a 3-input NAND gate. As a result, when we combine this NAND gate and the buffer serially, a 3-input AND gate providing correct output is obtained.

## 8. Threshold logic equation for OR and NOR Gate

We are interested in building a 2- input NOR gate to be derived from the generic threshold gate given in Fig.1. To implement the 2- input NOR gate, we need a threshold logic 2-input OR gate which will be connected with a buffer in series. Comparing the generic threshold gate based general equation given in equation (2) we consider a gate of two variables A and B of function  $F(A, B)$  as.

$$F(A, B) = sgn\{w_A \cdot A + w_B \cdot B - \theta\} \dots \dots \dots (11)$$

For a positive logic we shall assume that all the values of  $w_A, w_B$  and  $\theta$  are positive. If we take  $w_A = 1, w_B = 1$  and  $\theta = 0.5$  ( or any value in the range  $0 < \theta \leq 1$ ), then the above equation becomes



$$F(A, B) = \text{sgn}\{A + B - 0.5\} \dots\dots\dots (12)$$

And if we construct the truth table of it we get the Table-8.

**Table-8**

A	B	$\{W_A + W_B\}$	$\theta = 0.5$	$F(A, B)$
0	0	0	$0 < 0.5$	0
0	1	1	$1 \geq 0.5$	1
1	0	1	$1 \geq 0.5$	1
1	1	2	$2 \geq 0.5$	1

The Table -8 satisfies the condition of an OR gate, so equation (12) is written as

$$OR(A, B) = \text{sgn}\{A + B - 0.5\} \dots\dots\dots (13)$$

As already discussed that buffer inverts its own input signal, we modify the threshold equation of  $OR(A, B)$  such that it determines  $NOR(A, B)$ . So when we combine the result of  $OR(A, B)$  with a buffer, a new threshold logic buffered gate we get calculates the value of  $NOR(A, B)$  and the equation of the  $NOR$  logic gate will be

$$NOR(A, B) = \overline{OR(A, B)} \\ = \text{sgn}\{-A - B - (-0.5)\} \dots\dots\dots (14)$$

Here the value of the threshold voltage  $-0.5$  may be any value in the range of  $-1 > \theta \geq 0$ .

**Table-9**

A	B	$\{W_A + W_B\}$	$\theta = -0.5$	$F(A, B)$
0	0	0	$0 \geq -0.5$	1
0	1	-1	$-1 < -0.5$	0
1	0	-1	$-1 < -0.5$	0
1	1	-2	$-2 < -0.5$	0

From the Table -9 it is observed that  $F(A, B)$  satisfies all the conditions of an  $NOR$  gate, so the equation we derive is correct.

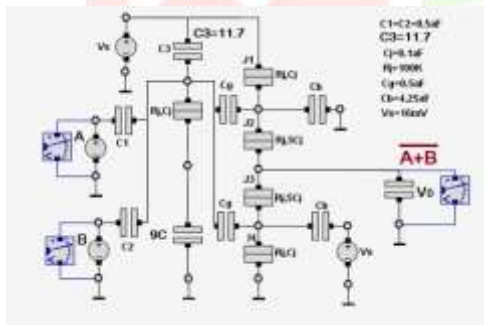
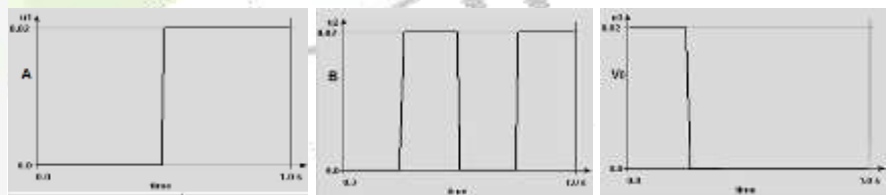


Fig 7(a) TLG based NOR gate



Fig, 7(b) Simulation result of NOR gate

For implementing the buffered Boolean logic  $NOR$  gate we will use the parameters logic input “0”=0V, logic “1”=16mV,  $C_1^n = C_2^n = 0.5aF$ ,  $C_3 = 11.7aF$ ,  $C_b = C_{b1} = C_{b2} = 4.25aF$ ,  $C_{g1} = C_{g2} = 0.5aF$ ,  $C_L = 9aF$ ,  $C_0 = 9aF$ ,  $R_j = 10^5 \Omega$ ,  $V_s = V_b = 16mV$  and accordingly after simulation the result we get is given in Fig. 7(b).

## 9. RS Flip-flop

RS Flip-flop based on Boolean logic if drawn in Fig.8. The output of new  $Q^t$  is  $Q^{t+1} = S + \bar{R}Q^t$ . The Truth table of this flip-flop is given in Table-10.

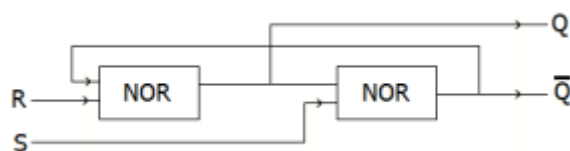


Fig. 8 RS Flip-flop based on Boolean gates



Table-10

Truth Table of R-S Flip-flop

R	S	Q	$\bar{Q}$	Description
0	0	last Q	last $\bar{Q}$	Keeps current o/p values of Q and $\bar{Q}$
0	1	1	0	o/p Q is set to 1 and $\bar{Q}$ to 0
1	0	0	1	Q is reset to 0 and $\bar{Q}$ to 1
1	1	?	?	Forbidden or Unspecified input combination

RS Flip-flop is a sequential circuit, its one output is feedback to an input of previous one. This flip-flop consisting of two NOR gates, two input terminals A and B, and two outputs Q and  $\bar{Q}$  is depicted in Fig. 8. The input-output relationship shown in the Table-10 is as follows. When  $R = 0$  and  $S = 0$ , the output of the Flip-flop remains the same.

If  $R = 1$  and  $S = 0$ , the output Q is reset to  $Q = 0$  and  $\bar{Q} = 1$ . If  $R = 0$  and  $S = 1$ , the output Q is reset to  $Q = 1$  and  $\bar{Q} = 0$ . But when the inputs are both 1, the output is unstable, this is why we are to avoid this input condition.

A Boolean gate-based implementation of the RS Flip-flop usually has two cross-coupled gates which constitute a feedback loop. When R and S are set both to zeros, the two NOR gates behave as chained inverters and they form bi-stable elements (where  $R=0$  and  $S=1$ , and  $R=1$  and  $S=0$  are indicating two stable states). Now if  $R=0$  and  $S=1$ , the output of the first NOR gate is forced to 1 and as a chained inverter so  $Q=1$  and  $\bar{Q}=0$ . In the same way when  $R=1$  and  $S=0$ , the output of the second NOR gate is forced to 1 and so  $Q=0$  and  $\bar{Q}=1$ . When both the input levels are high i.e.,  $R=1$  and  $S=1$ , both the NOR gates are forced to deliver the low outputs i.e.,  $Q=0$  and  $\bar{Q}=0$ . At this moment if we set both the inputs as 0s simultaneously, the output of the Flip-flop be either switches to  $Q=1$  and  $\bar{Q}=0$  or  $Q=0$  and  $\bar{Q}=1$ , it is observed they even oscillates between these two solutions and it is not desired. So these input are always avoided.

To implement the RS Flip-flop we require two buffered Boolean NOR gates which are elaborately discussed in section 8 and its correctness is verified with the help of SIMON as well. For testing the circuit we initiate with the inputs  $R=0$  and  $S=0$  while  $Q=0$  and  $\bar{Q}=1$ . Now, as we set  $S=1$  the output of first NOR becomes high i.e.,  $Q=1$  and  $\bar{Q}=0$ . These two values are memorized when S returns to 0. Next when the input value  $R=1$ , the outputs are  $Q=0$  and  $\bar{Q}=1$ . These two values are memorized when the value of R is returned to 0. At the last step when we set both the inputs high simultaneously, then both the outputs are forced to 0s. If we set, now,  $R=0$  and  $S=0$ , the simulator determines the possible tunnels events and after resolving them the simulator displayed the output results as  $Q=1$  and  $\bar{Q}=0$ . So, at last, we conclude that the RS Flip-flop gives the correct behavior shown in the Table-10 above.

## 10. Threshold gate based Memory cell

We have examined Boolean gate-based implementations of the RS flip-flop. The verification completed by means of simulation and the results got from simulation signifies that they operate as per their specified logic function for these implementations, every Boolean logic gates we utilized (except for the inverter) is derived from a generic threshold gate design in Fig.1. We can prove that any Boolean logic function can also be realized by a network of threshold logic gates [7, 11, 13, 18]. If anybody wants to compare the threshold gate-based realization with the Boolean gate-based realization of the same logic function, the threshold gate-based design will have, at most, the same number of logic gates and the same network depth. We shall describe threshold logic gate-based implementations of the memory cell like RS Flip-flop, T Flip-flop, and edge-triggered T Flip-flop. Every one of the utilized threshold gates is derived from the generic threshold gate design shown in Fig. 1.

### Characteristic equation of RS Flip:

For writing the characteristic equation of the RS Flip-flop we first write the truth table of it, then with the help of the K-Map the characteristic equation of  $Q^{t+1}$  and  $\bar{Q}^{t+1}$  are written in the equations (14) and (15) respectively. The truth table is given in Table-11.

$$Q^{t+1} = \bar{R}Q^t + \bar{R}S \quad \text{..... (14)}$$

$$\bar{Q}^{t+1} = \bar{S}\bar{Q}^t + RS \quad \text{..... (15)}$$

**Table-11****Truth table for RS Flip-flop**

R	S	Previous		New	
		$Q^t$	$\bar{Q}^t$	$Q^{t+1}$	$\bar{Q}^{t+1}$
0	0	1	0	1	0
0	1	1	0	1	0
1	0	1	0	0	1
1	1	1	0	invalid	
0	0	0	1	0	1
0	1	0	1	1	0
1	0	0	1	0	1
1	1	0	1	invalid	

RS $Q^t$				
	00	01	11	10
0	0	1	x	0
1	1	1	x	0

$Q^{t+1} = \bar{R}Q^t + \bar{R}S$

It is observed that Boolean gate based RS Flip-flop has the unstable behavior when both of the inputs R and S are high i.e.,  $R=S=1$ . This instability will happen for the cause of the cross-connection of the Two NOR gates by Boolean gate implementation. If we are able to implement an RS Flip-flop without cross-coupling gates, the instability problem created in present implementation can be resolved. In this situation, when we are intended in designing the RS Flip-flop, it costs more number of gates. On the other hand, for the sake of a threshold gate design, the gates required will be less.

A threshold gate based design of an RS Flip-flop can be implemented with the help of a single threshold gate only. By a small modification [7, 8] of the equations (14) and (15) we can get the Threshold gate equations (16) and (17) respectively, where there would not be any occurring events of uncertainty.

$$Q^{t+1} = \bar{R}Q^t + \bar{R}S + SQ^t \quad (16)$$

$$\bar{Q}^{t+1} = \bar{R}\bar{Q}^t + R\bar{S} + \bar{S}\bar{Q}^t \quad (17)$$

For the verification of equations (16) and (17), a table is provided without giving the input condition  $R=1$  and  $S=1$  in Table-12.

**Table-12**

R	S	Previous		New	
		$Q^t$	$\bar{Q}^t$	$Q^{t+1}$	$\bar{Q}^{t+1}$
0	0	1	0	1	0
0	1	1	0	1	0
1	0	1	0	0	1
1	1	1	0	1	0
0	0	0	1	0	1
0	1	0	1	1	0
1	0	0	1	0	1
1	1	0	1	0	1

Now for implementation of RS Flip-flop, we verify that the functions (16) and (17) are satisfying the specified function of RS Flip-flop. If we put  $R=0$  and  $S=0$ , we get  $Q^{t+1}=Q^t$  (Hold). If we put  $R=0$  and  $S=1$ , we get  $Q^{t+1}=1$  (Set). If we put  $R=1$  and  $S=0$ , we get  $Q^{t+1}=0$  (Reset). If we put  $R=1$  and  $S=1$ , we get  $Q^{t+1}=Q^t$  (Hold). And for all cases, the output  $\bar{Q}^{t+1}$  will be the complement of output  $Q^{t+1}$ . So we decide that equation (16) and (17) satisfy the characteristics of an RS Flip-flop without any forbidden input combination  $R=0, S=0$  or  $R=1, S=1$  and both of these combination corresponding to Hold function.

For implementing the Boolean gate based logic circuit of the equation (16) we require three 2-input AND gates and one 3-input OR gate and that will result in a logic circuit of a depth of two. On the other hand when we want to implement it with the help of threshold logic gate based circuit we need only a 3-input threshold gate which implements the same Boolean equation (16). The threshold logic equation of equation (16) is given in equation (18).

$$Q^{t+1} = \text{sgn}\{\bar{R}+S+Q^t-2\} \quad (18)$$

For any logic variable R we have always  $R + \bar{R} = 1$  or  $R = -\bar{R} + 1$  or  $\bar{R} = -R + 1$  so the above equation becomes

$$Q^{t+1} = \{-R+S+Q^t-1\} \quad (19)$$

For the verification of the correctness of equation (18) or (19), one can check the value of  $Q^{t+1}$  by putting the value of the variables R, S and  $Q^t$  in equation (19) and he gets the following data in the Table-13. From the table it is transparent that for  $R=S=0$ , and  $R=S=1$  the value of  $Q^{t+1}$  holds its previous values ( $Q^t$ ). When  $R=0$  and  $S=1$ , the value of  $Q^{t+1}$  is 1(Set) irrespective of its previous value. Similarly, when  $R=1$  and  $S=0$ , the value of  $Q^{t+1}$  is 0(Reset) irrespective of the value of  $Q^t$ . The corresponding threshold logic gate of the equation (19) has been depicted in Fig.9. In this, there are 3-input terminals one for R, one for S and the last one for a feedback signal from the output Q and the inputs having their weight values are -1, +1, and +1 respectively. In addition, the threshold value of the threshold gate is 1 shown in the circle.

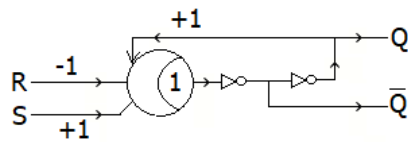


Fig.9 TLG based RS Flip-flop for eqn. (19)

Table-13

Verification of the correctness of equation (19)

R	S	$Q^t$	$Q^{t+1}$	function
0	0	0	0	Previous value
0	0	1	1	
0	1	0	1	Set
0	1	1	1	
1	0	0	0	Reset
1	0	1	0	
1	1	0	0	Previous value
1	1	1	1	

The RS Flip-flop we have proposed has been theoretically proven and it has also been verified by using the simulator and the simulation results are shown in the Fig 10. We can obtain the Simulation results by using the circuit parameters given below for the threshold logic gate logic input “0”=0V, logic “1” = 16mV,  $C_1^p(w_1 = 1) = C_2^p(w_2 = 1) = 0.5aF$ ,  $C_1^n(w_3 = -1) = 0.4aF$ ,  $C_3 = 12.7aF$ ,  $C_L = 9aF$ ,  $C_0 = 8.6aF$ ,  $R_j = 10^5 \Omega$ ,  $V_s = V_b = 16mV$ . For the buffer, the parameters taken are,  $C_{g1} = C_{g2} = 0.5aF$ ,  $C_1 = C_4 = 0.5aF$ ,  $C_2 = C_3 = 0.1aF$ ,  $C_{b1} = C_{b2} = 4.25aF$ ,  $C_L = 9aF$ ,  $R_j = 50K\Omega$ ,  $C_j = 0.1aF$ .

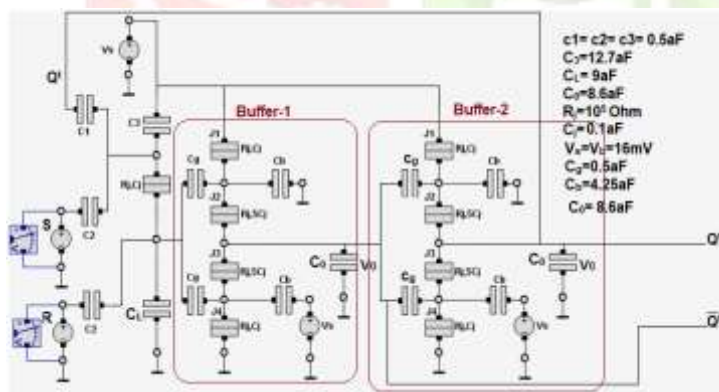


Fig. 10. Threshold Logic based RS Flip-flop

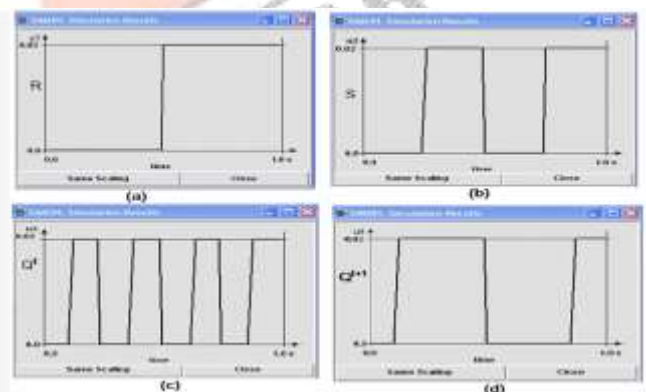


Fig. 10(a) R-input (b) S-input (c) Previous  $Q^t$  (c)  $Q^{t+1}$ =New  $Q^t$

## 11. T Flip-flop

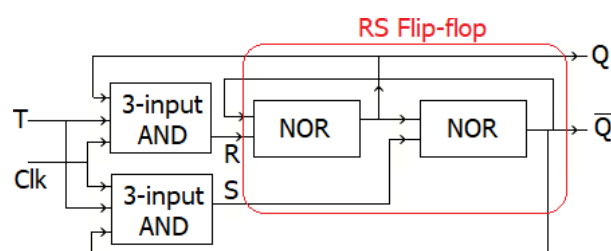






Fig. 11 Flip-flop based on Boolean gates

T Flip-flop is a sequential circuit, i.e., its one output is feedback to the input. This can be used as a memory cell or memory unit or memory element for retrieving or storing the information. T Flip-flop consists of two 3-input AND gates, Two NOR gates, two input terminals T and Clk, and two output terminals Q and  $\bar{Q}$ . The behavior of triggered T Flip-flop shown in Table-14 is as follows. For every positive clock pulse, a transition happens. If T and current Q are low, then the output Q i.e.,  $Q^{t+1}$  is low. If T and current Q are different values i.e., one is high and another is low, then  $Q^{t+1}$  is high. And when both T and current Q are high, then the output Q i.e.,  $Q^{t+1}$  is low. Unlike RS Flip-flop, a T Flip-flop is having no Forbidden or Unspecified input combination(s).

Table-14

Truth Table of T Flip-flop

Clk	T	Q	$Q^{t+1}$	$\bar{Q}^{t+1}$	Function
	0	0	0	1	If both T and Q values are "0", then $Q(t+1)=0$
	1	0	1	0	If T and Q values are different then $Q(t+1)=1$
	0	1	1	0	If T and Q values are different then $Q(t+1)=1$
	1	1	0	1	If both T and Q values are "1", then $Q(t+1)=0$

A possible T Flip-flop implementation on the basis of Boolean logic gates is drawn in Fig.11. The operation of the circuit is as follows. The two cross-coupled NOR gates forms an RS Flip-flop. If T=0, then irrespective of Q(current), both R and S will be 0 and value of  $Q^{t+1}$  will be last Q. Now if T=1 then the value of R and S will be depended upon the present value of Q and  $\bar{Q}$ . If Q=0 and  $\bar{Q}=1$ , the value of  $Q^{t+1}$  will be complement of last Q i.e.,  $Q^{t+1}=1$ , and if Q=1 and  $\bar{Q}=0$ , the value of  $Q^{t+1}$  will be complement of last Q i.e.,  $Q^{t+1}=0$ .

In no case, the values of R and S will be high (=1) simultaneously.

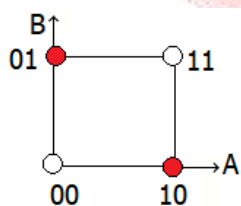
## 12. Implementation of T Flip-flop using Threshold Logic Gate

The logic function of T Flip-flop on the basis of Boolean gate can be expressed as

$$Q^{t+1} = T\bar{Q}^t + \bar{T}Q^t \quad (20)$$

and  $\bar{Q}^{t+1} = \bar{T}Q^t + T\bar{Q}^t \quad (21)$

It is observed that  $\bar{Q}^{t+1} = \text{NOT}(Q^{t+1})$ . We are familiar with the expression  $Y = A\bar{B} + \bar{A}B$  and it is similar to the equation (20). Space plot diagram of Y in 2D space is shown in Fig. 12. From this Fig.12, we observe that no linear separating line that is separating the red and white circles can be drawn. So the Boolean logic function is not linearly separable. Therefore we would not be able to draw a threshold logic gate that represents the equation  $Y = A\bar{B} + \bar{A}B$  or  $Q^{t+1} = T\bar{Q}^t + \bar{T}Q^t$ .

Fig.12 Space plot of  $Y = A\bar{B} + \bar{A}B$ 

Now for representing the Boolean function  $Q^{t+1} = T\bar{Q}^t + \bar{T}Q^t$  by a threshold logic gate, first we express  $P = (T\bar{Q}^t)$  with the help of threshold gate-based equation. As it is an AND function, so using equation (13) we can write for P as.

$$P = \text{sgn} \{ T + \bar{Q}^t - 2 \} \quad (22)$$

As we know  $\bar{Q}^t + Q^t = 1$  or  $\bar{Q}^t = 1 - Q^t$ , the equation (23) can be written as equation (23).

$$P = \text{sgn} \{ T - Q^t - (1) \} \quad (23)$$

The threshold Logic gate of equation (23) is drawn in Fig. 13(a). For stability of this gate we can connect two NOT gates in series to the output point of P which is shown in Fig. 13(b). The truth table of equation (23) is given in Table-15.



Table-15

Truth table of equation (23)

T	$Q^t$	P
0	0	0
0	1	0
1	0	1
1	1	0

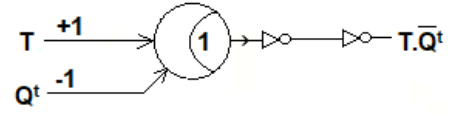
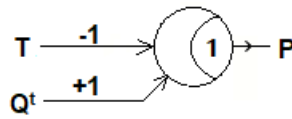


Fig. 13(a) threshold Logic gate of equation (23) Fig. 13(b) Modification of TLG of equation (23)

We again modify the threshold equation of P as we know that the buffer inverts its input signal, so that it calculates  $\bar{P} = \text{sgn} \{-T + Q^t - (0)\}$  and its corresponding truth table is given in Table-16. The threshold logic gate of equation  $\bar{P}$  and its complement are drawn in Fig.14 (a) and 14(b) respectively.

Table-16

Truth table of  $\bar{P}$ 

T	$Q^t$	$\bar{P}$
0	0	1
0	1	1
1	0	0
1	1	1

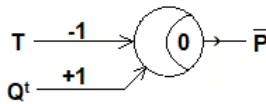
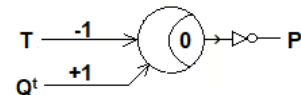
Fig. 14(a) Threshold logic gate of  $\bar{P}$ 

Fig. 14(b) Complement of Fig.14(a)

Now this is the time we are to express the Boolean expression  $Q^{t+1} = T.\bar{Q}^t + \bar{T}.Q^t$ . We assume  $P = T.\bar{Q}^t$

So  $Q^{t+1} = T.\bar{Q}^t + \bar{T}.Q^t = P + \bar{T}.Q^t$ .

$$Q^{t+1} = P + \bar{T}.Q^t \quad (24)$$

For finding out the threshold gate logic of equation (24), we draw the truth Table-17 with the assistance of Table-15 and from which we solve the minimum value solution to get the weight values.

$$0 < \theta \quad (25)$$

$$W_3 \geq \theta \quad (26)$$

$$W_1 + W_2 \geq \theta \quad (27)$$

$$W_2 + W_3 < \theta \quad (28)$$

From the equation (25), we have  $\theta$  is greater than 0, so it may be 1 or 2 or 3.

From equation (26) it is transparent that  $W_3$  must be equal to or greater than  $\theta$ . For minimum value condition we can take as.

$$\theta = W_3 = 1 \quad (29)$$

In equation (28) if we put  $W_2 = -1$  and  $W_3 = 1$  then the equation is satisfied, so we take  $W_2 = -1$

Next in equation (27) if we put  $W_2 = -1$ ,  $W_1 = 2$  and  $\theta = 1$  then it is satisfied. So a solution set is  $\{W_1, W_2, W_3; \theta\} = \{2, -1, 1; 1\}$

Hence the Threshold equation for the T Flip-flop is

$$Q^{t+1} = \text{sgn} \{ 2P - T + Q^t - (1) \} \quad (30)$$

And its corresponding Threshold logic gate based T Flip-flop is depicted in Fig. 15. The simulation set and its simulated results are shown in Fig. 16 and Fig. 16(a) respectively.

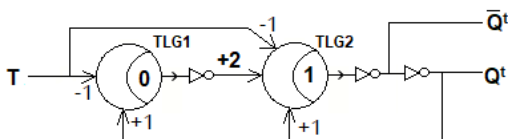


Fig. 15 Threshold logic gate based T Flip-flop

For the simulation purpose of the T Flip-flop we have used the following parameters. For Threshold logic gate 1 (TLG1):  $C_0 = 8.1 \text{ aF}$ ,  $R_j = 10^5 \Omega$ ,  $V_s = V_b = 16 \text{ mV}$ ,  $C_1^p(w_1 = 1) = 0.5 \text{ aF}$ ,  $C_1^n(w_3 = 1) = 0.4 \text{ aF}$ , For the Threshold Logic gate 2 (TLG2):  $C_1^p(w_1 = 2) = 1 \text{ aF}$ ,  $C_2^p(w_2 = 1) = 0.5 \text{ aF}$ ,  $C_1^n(w_3 =$

–1) = 0.4aF,  $C_3 = 13.4aF$ ,  $C_L = 9aF$ ,  $C_0 = 8.6aF$ ,  $R_j = 10^5 \Omega$ ,  $V_s = V_b = 16mV$ . For the buffer, the parameters taken are,  $C_{g1} = C_{g2} = 0.5aF$ ,  $C_1 = C_4 = 0.5aF$ ,  $C_2 = C_3 = 0.1aF$ ,  $C_{b1} = C_{b2} = 4.25aF$ ,  $C_L = 9aF$ ,  $R_j = 50K\Omega$ ,  $C_j = 0.1aF$ .

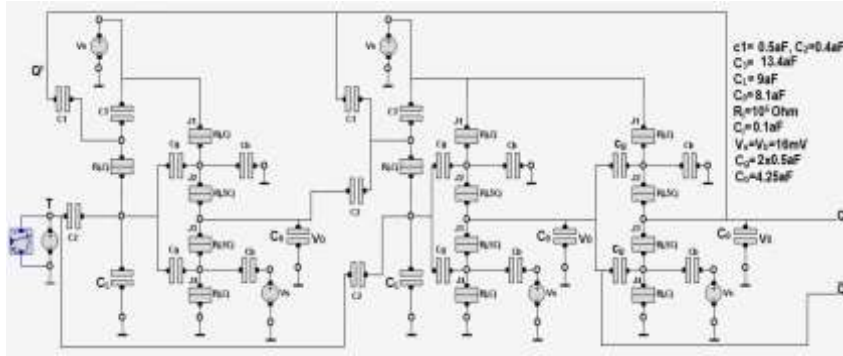
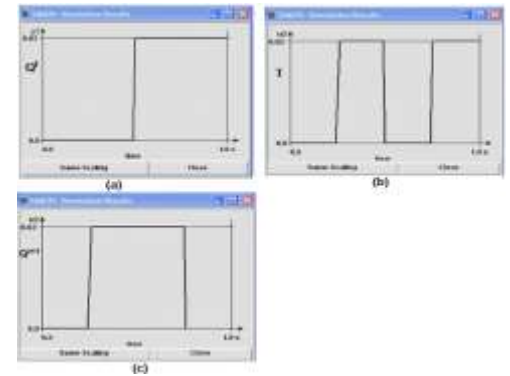


Fig. 16 T-Flip-flop Threshold logic gate

Fig.16 (a)  $Q^t$ -input (b) T-input (c)  $Q^{t+1}$ =New  $Q^t$ 

Till now we have been able to implement a T-Flip-flop without trigger signal. When anyone is interested in making a register or counter he is to add a clock input signal for synchronization purpose, he must include clock signal as a new input signal. To do this, we may connect a AND gate at the input side of Fig. 15 and the combination circuit is shown in Fig.16. The truth table of edge-triggered T Flip-flop is shown in Table-10.

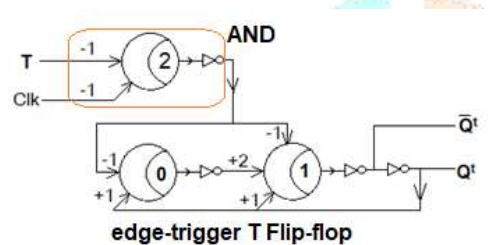


Fig. 17 edge triggered T Flip-flop

**Table-10**  
Truth table of edge triggered T Flip-flop

Clk	T	$Q^t$	$Q^{t+1}$
	0	0	0
	0	1	1
	1	0	1
	1	1	0

For simulation set of this edge triggered T Flip-flop, parameters like capacitances, input voltages, bias voltage etc. are to be specified. The parameters we used are given as: For AND gate we will use the parameters  $C_1^n = C_2^n = 0.5aF$ ,  $C_{b1} = C_{b2} = 4.25aF$ ,  $C_{g1} = C_{g2} = 0.5aF$ ,  $C_L = 9aF$ ,  $C_0 = 8aF$ ,  $R_j = 10^5 \Omega$ . For Threshold logic gate1 (TLG1):  $C_0 = 8.1aF$ ,  $R_j = 10^5 \Omega$ ,  $V_s = V_b = 16mV$ ,  $C_1^P(w_1 = 1) = 0.5aF$ ,  $C_1^n(w_3 = -1) = 0.4aF$ , For the Threshold Logic gate 2 (TLG2) :  $C_1^P(w_1 = 2) = 1aF$ ,  $C_2^P(w_2 = 1) = 0.5aF$ ,  $C_1^n(w_3 = -1) = 0.4aF$ ,  $C_3 = 13.4aF$ ,  $C_L = 9aF$ ,  $C_0 = 8.6aF$ ,  $R_j = 10^5 \Omega$ ,  $V_s = V_b = 16mV$ . For the buffer, the parameters taken are,  $C_{g1} = C_{g2} = 0.5aF$ ,  $C_1 = C_4 = 0.5aF$ ,  $C_2 = C_3 = 0.1aF$ ,  $C_{b1} = C_{b2} = 4.25aF$ ,  $C_L = 9aF$ ,  $R_j = 50K\Omega$ ,  $C_j = 0.1aF$ . Taking these parameters we have set the simulation circuit in Fig. 17(a) and input-output simulation result is shown in Fig. 17(b), (c), (d) and (e).

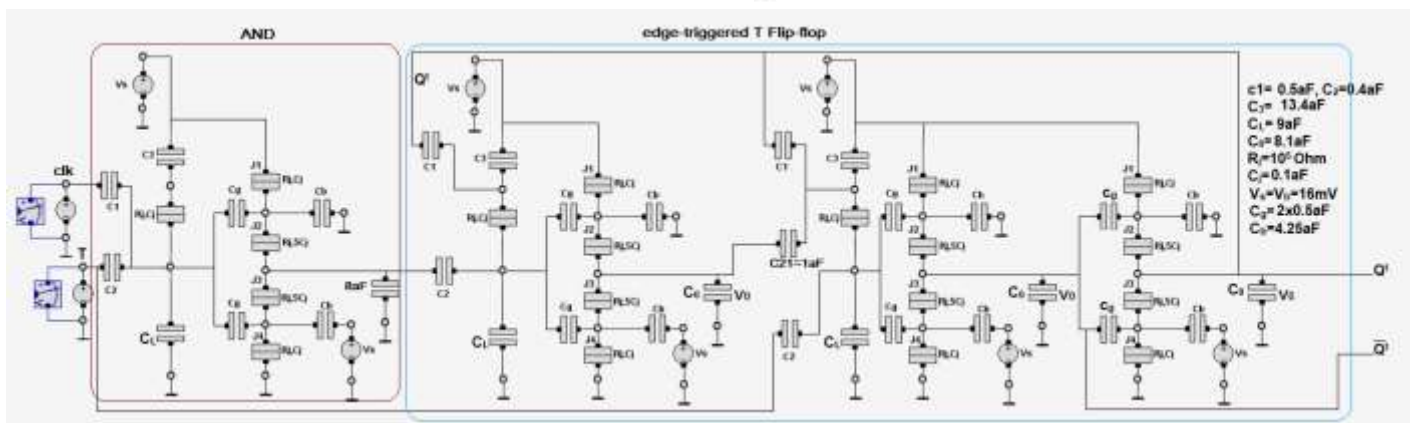


Fig. 17(a) edge-triggered T Flip-flop

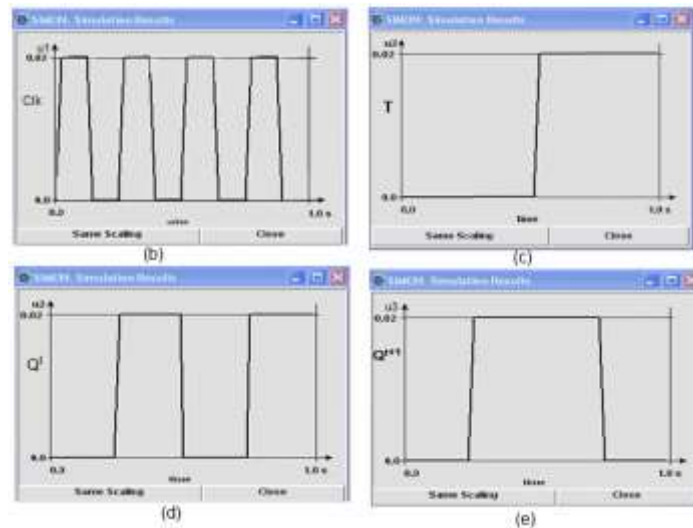


Fig. 17(b) Clock, (c) T-input (d),  $Q^t$  previous (e)  $Q^{t+1}$  = New  $Q^t$

### 13. Description of an Up-Down counter made up of Threshold logic gate.

For the count-down counter when we set down-input High (i.e.,=1) in Fig. 18(a), the “T flip-flop 0” in the lowest-order placed in the top is complemented with every clock input pulse. A T Flip-flop in any other places will be complemented with a positive clock pulse provided that all the lower-order bits are equal to 0. For instance, if the present state of a 4-bit count-down binary counter is  $A_3A_2A_1A_0 = 1000$ , then the next count will be 0111.  $A_0$  is always complemented with clock.  $A_2$  will be complemented because the present state of  $A_1 = 0$ .  $A_3$  will be complemented as the present state of  $A_2 A_1 = 00$ .  $A_4$  will be complemented since the present state of  $A_3A_2 A_1 = 000$ .

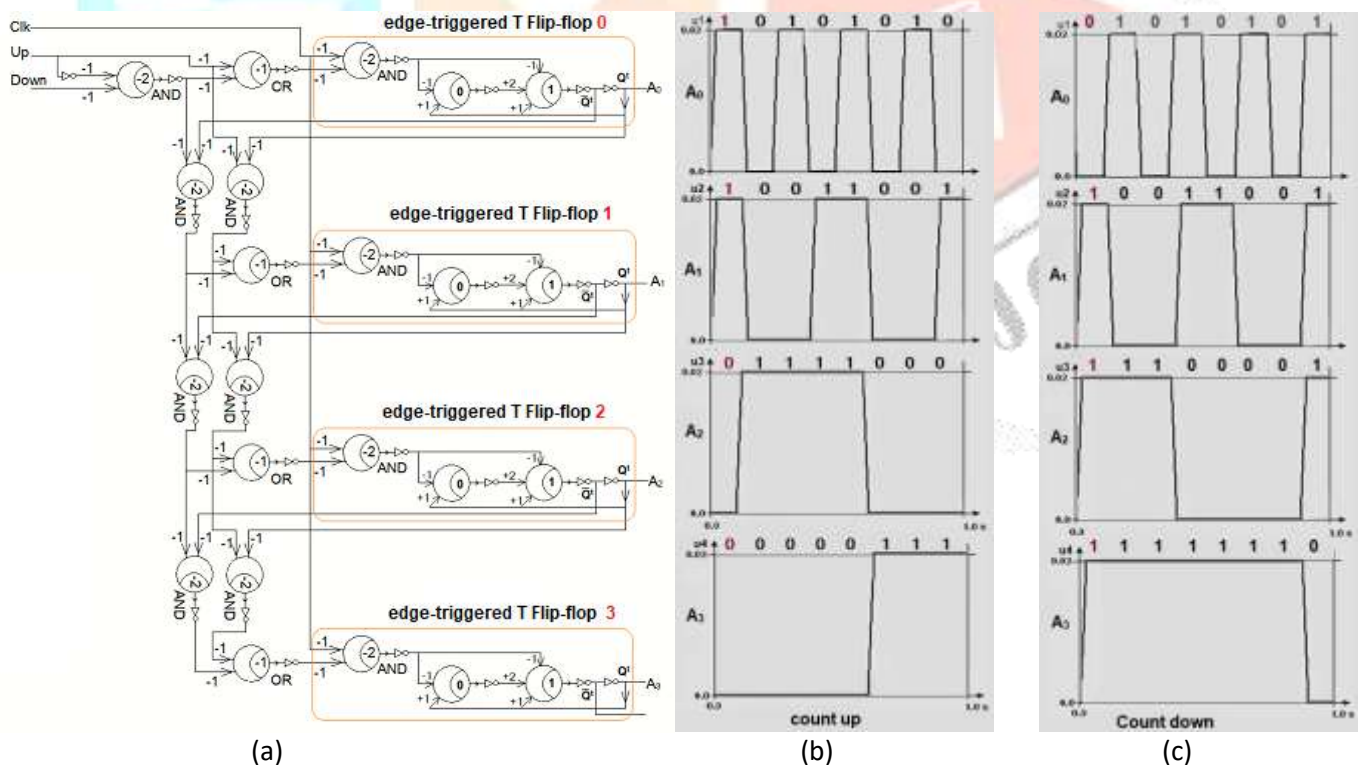


Fig.18 (a) Threshold Logic gate based 4-bit Up-Down Counter (b) simulation of count up and (c) count down

A binary counter that is eligible for counting either up or down is drawn in Fig.18 based on threshold logic gate. The T flip-flops employed in this circuit can be considered to be a JK Flip-flops where both J and K terminals are tied together. When the up-input is 1 in Fig. 18(a), the circuit counts up, because the T inputs get their signals from the previous normal outputs of the T flip-flops. When the down-input control is 1 and the up input is 0, the circuit starts counting down, since the complemented outputs of the previous T flip-flops are applied to the T inputs. When both the up-input and down-input are being 0, the circuit does not change any state yet keeps in the same count. If the up and down inputs are both high (=1), the circuit counts



up. This ensures us that only one operation is performed at any given time. The related up- and down- counter data are provided in Table-10.

## 14. Discussion

What we have discussed different gates, combinational and sequential circuits based on the LTG gates. Whether they are slow or fast, we are to observe it. For determining the processing delay of any logic gates, we must involve critical voltage  $V_c$  given in equations (6) and (7), as well as the tunnel junction capacitance  $C_j$ . However, assuming the atmosphere temperature at  $T = 0K$ , the switching/processing delay of a logic gate can be calculated using the approaches [7, 8].

$$\text{Delay} = -(e|\ln(P_{\text{error}})|R_t) / (|V_j| - V_c) \dots\dots\dots (31)$$

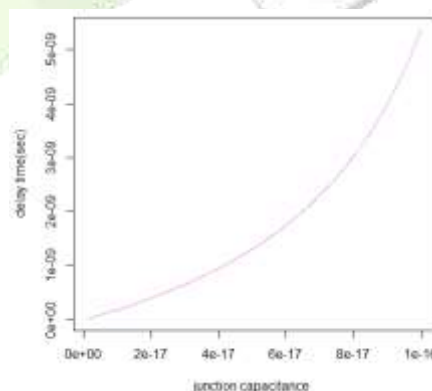
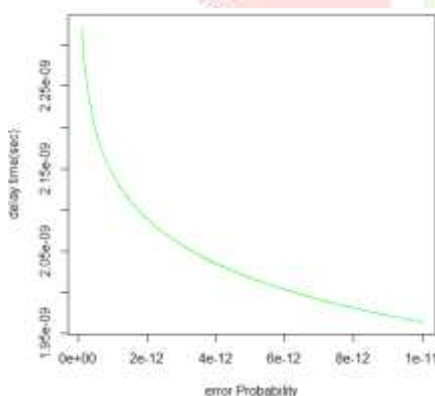
where  $V_j$  is the junction voltage and  $V_c$  is the critical r threshold voltage and  $R_t$  is the junction resistance.

The slowest switching will happen whenever the critical voltage  $V_c$  has the value which is lesser than the tunnel junction voltage  $V_j$ , i.e.,  $V_c < |V_j|$ , but very near to it. This must happen, for example when  $V_{in1}$  is logic 1, resulting  $V_j = 11.8mV$  for the case of a 2-input NOR gate in Fig-7(a), the critical voltage of the tunnel junction voltage  $V_c$  is 11.58mV. Given that the probability of error change  $P_{\text{error}} = 10^{-12}$ , resistance  $R_t = 10^5\Omega$ . After calculation we get a gate delay equal to  $0.07281|\ln(P_{\text{error}})|ns = 1.675 ns$ . In the same manner, we can calculate the circuit delays written in Table-11. Whenever a charge passes through the tunnel junction, the amount of total energy in the circuit changes before and after the tunneling events. The difference between the energy levels before and after the tunneling event is found out by the relation

$$\begin{aligned} \Delta E &= E_{\text{before tunnel}} - E_{\text{after tunnel}} \\ &= -e(V_c - |V_j|) \dots\dots\dots (32) \end{aligned}$$

and it is the amount of switching energy consumed when a tunnel event occurs in the tunneling circuit.

We have drawn curves as to the switching delay as a function of the switching error probability in Fig. 19(a) and the switching delay as a function of the unit capacitance  $C$  shown in Fig. 19(b).



**Table-10**  
Truth table of Up-down Counter

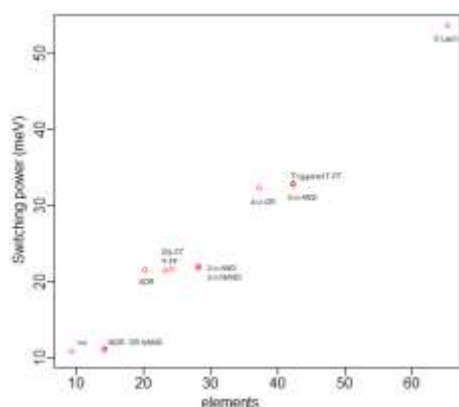
Initial state	Down count	Up count
Present state	Next state	Next state
$A_3A_2A_1A_0$	$A_3A_2A_1A_0$	$A_3A_2A_1A_0$
0000	1111	0001
0001	0000	0010
0010	0001	0011
0011	0010	0100
0100	0011	0101
0101	0100	0110
0110	0101	0111
0111	0110	1000
1000	0111	1001
1001	1000	1010
1010	1001	1011
1011	1010	1100
1100	1011	1101
1101	1100	1110
1110	1101	1111
1111	1110	0000

Fig. 19(a) Delay vs. Error Probability    Fig. 19(b) Delay Vs. capacitance

We have found out the element numbers for each and every case of gates or circuits, their switching delays, and switching energy consumptions for the corresponding individual LTGs (using the same methodology as adopted for the Boolean gates). All the calculated or found out parameters are shown in tabular form in Table-11.

The switching energy vs. elements diagram regarding our present LTG based circuits is shown in Fig. 20.





**Fig. 20.** Switching energy vs. elements

Next, we have calculated those parameters for T Flip flop and Up-down counter circuits and that are presented in Table-11.

The time delays or processing delays for different gates and circuits are different. For RS Flip-flop switching delay is  $0.082|\ln(P_{error})|$  ns, for T Flip-flop it is  $0.082|\ln(P_{error})|$  ns, for edge triggered T Flip-flop delay is  $0.144|\ln(P_{error})|$  ns and for 4-bit Up-down counter the delay is  $0.224|\ln(P_{error})|$  ns. Given that the value of  $P_{error}$  equals to  $10^{-12}$ , so the time after which the 1<sup>st</sup> output of each Flip-flop of the 4-bit up-down counter will be fanned out is  $0.224|\ln(P_{error})|$  ns = 6.19 ns. i.e., after every 6.19 ns, the next output bit will be taken from the counter. Therefore clock time/duration of the clock signal must be greater than or equal to 6.19 ns.

**Table-11**

Gate/Device	elements	Delay	Switching Energy
inverter	09 elements	$0.022 \ln(P_{error}) $ ns	10.4 meV
2-input NOR	14 elements	$0.072 \ln(P_{error}) $ ns	10.7 meV
2-input OR	14 elements	$0.062 \ln(P_{error}) $ ns	10.8 meV
2-input NAND	14 elements	$0.080 \ln(P_{error}) $ ns	10.7 meV
2-input AND	14 elements	$0.062 \ln(P_{error}) $ ns	10.8 meV
3-input AND	28 elements	$0.104 \ln(P_{error}) $ ns	21.6 meV
3-input NAND	28 elements	$0.072 \ln(P_{error}) $ ns	21.4 meV
2-input XOR	20 elements	$0.102 \ln(P_{error}) $ ns	21.2 meV
D latch	65 elements	$0.342 \ln(P_{error}) $ ns	53.2 meV
3-input OR	28 elements	$0.104 \ln(P_{error}) $ ns	21.6 meV
4-input OR	42 elements	$0.104 \ln(P_{error}) $ ns	32.4 meV
4-input AND	42 elements	$0.104 \ln(P_{error}) $ ns	32.4 meV
RS Flip-flop	24 elements	$0.082 \ln(P_{error}) $ ns	21.2 meV
T Flip-flop	23 elements	$0.082 \ln(P_{error}) $ ns	21.1 meV
Edge triggered T Flip-flop	37 elements	$0.144 \ln(P_{error}) $ ns	31.9 meV
4-bit Up-down Counter	302 elements	$0.224 \ln(P_{error}) $ ns	246.4 meV

We are interested in comparing the circuit delays of CMOS, SET-based and LTG-based. The processing delay or switching delay for a CMOS logic gate like AND, NAND, NOR, XOR is 12 ns [18, 19], on the other hand the time required for tunneling through a single electron transistor (SET) is approximately 4 ns [4, 5, 16, 17].

Table-12

## Switching delays of SET and LTG

Gate/Device	SET-based delay	LTG-based delay
inverter	8	0.60ns
2-input NOR	4	1.67ns
2-input OR	4	1.71ns
2-input NAND	4	2.21ns
2-input AND	4	1.71ns
3-input AND	8	2.87ns
3-input NAND	8	1.98ns
2-input XOR	4	2.81ns
D latch	32	9.44ns
3-input OR	8	2.87ns
4-input OR	12	2.87ns
4-input AND	12	2.87ns
RS Flip-flop	8	2.26ns
T Flip-flop	8	2.26ns
Edge triggered T Flip-flop	12	3.98ns
4-bit Up-down Counter	20	6.19ns

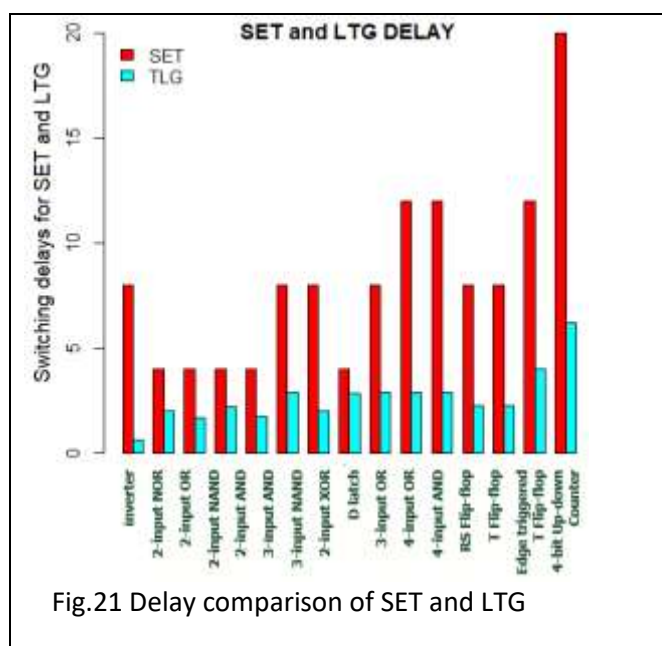


Fig.21 Delay comparison of SET and LTG

The XOR gate using conventional logic circuits needs 16 transistors, whereas the same function can be implemented with the help of just one SET [3, 5, 6, 11] i.e. number of nodes can be reduced to 1 instead of 16 and for TLG-based time delay is 2.81 ns.

Assuming that the error probability is  $10^{-12}$  then the delay for the 2-input OR gate will be 1.71ns and similarly the delays for the other gates can be calculated and are all shown in Table-12. It is transparent that the LTG based circuit is faster than the SET based circuit when  $P_{error}=10^{-12}$ . The comparison of delays for SET and LTG gate based circuits is drawn by a bar diagram in Fig.21.

## 15. Conclusion

How an electron tunnels through a single electron transistor and an inverter is discussed first. A generic Linear Threshold logic Gate implementation is discussed about its construction and from which we have been able to derive a family of logic gates like AND, NAND, OR. All the gates along with a 4-bit up-down counter, RS Flip-flop, T Flip-flop and edge-trigger T Flip-flop have been implemented and are verified by means of simulation using SIMON. The number of elements requiring for logic gates, their delays, power consumed by them are given in tabular form and their related curves or bar diagram are also given in the connected figures. By dint of threshold logic equation all the LTG gates have been depicted. In single electron tunneling technology, it is observed that the threshold logic gates are faster than SET based logic gates. The atmosphere temperature should be kept at 0K in real operation.

## 16. REFERENCES

- [1] Anup Kumar Biswas, "Implementation of A 4n-Bit Comparator based on IC Type 74L85 using Linear Threshold Gate Tunneling Technology" International Journal of Engineering Research & Technology (IJERT), Vol. 10 Issue 05, May-2021 pp.299-310, ISSN: 2278-0181
- [2] Anup Kumar Biswas, "State Transition Diagram for A Pipeline Unit based on Single Electron Tunneling" International Journal of Engineering Research & Technology (IJERT) Vol. 10 Issue 04, April-2021pp.325-336, ISSN: 2278-0181
- [3] Anup Kumar Biswas, "Design of A Pipeline for A Fixed-Point Multiplication using Single Electron Tunneling Technology", International Journal of Engineering Research & Technology (IJERT), Vol. 10 Issue 04, April-2021 pp. 86-98, ISSN: 2278-0181
- [4] Souvik Sarkar<sup>1</sup>, Anup Kumar Biswas<sup>2</sup>, Ankush Ghosh<sup>1</sup>, Subir Kumar Sarkar<sup>1</sup> "Single electron based binary multipliers with overflow detection", International Journal of Engineering, Science and Technology Vol. 1, No. 1, 2009, pp. 61-73
- [5] A. K. Biswas and S. K. Sarkar: "An arithmetic logic unit of a computer based on single electron transport system": Semiconductor Physics, Quantum Electronics & Opt-Electronics. 2003. Vol 6. No.1, pp 91-96

- [6] A.K. Biswas and S. K. Sarkar: "Error Detection and Debugging on Information in Communication System Using Single Electron Circuit Based Binary Decision Diagram." Semiconductor Physics Quantum electronics and opt electronics, Vol. 6, pp.1-8, 2003
- [7] Alexander N. Korotkov, "Single-electron logic and memory devices" INT. ELECTRONICS, 1999, Vol. 86, No. 5, 511- 547
- [8] Casper Lageweg, Student Member, IEEE, Sorin Cotofană, Senior Member, IEEE, and Stamatis Vassiliadis, Fellow, IEEE "Single Electron Encoded Latches and Flip-Flops" IEEE TRANSACTIONS ON NANOTECHNOLOGY, VOL. 3, NO. 2, JUNE 2004
- [9] C. Lageweg, S. Cotofană, and S. Vassiliadis, "A linear threshold gate implementation in single electron technology," in IEEE Computer Society VLSI Workshop, Apr. 2001, pp. 93– A. Korotkov, "Single-electron logic and memory devices," Int. J. Electron., vol. 86, no. 5, pp. 511–547, 1999.
- [10] K. Likharev, "Single-electron devices and their applications," Proc. IEEE, vol. 87, pp. 606–632, Apr. 1999.
- [11] A. Korotkov, R. Chen, and K. Likharev, "Possible performance of capacitively coupled single-electron transistors in digital circuits," J. Appl. Phys., vol. 78, pp. 2520–2530, Aug. 1995.
- [12] J. R. Tucker, "Complementary digital logic based on the "Coulomb blockade"," J. Appl. Phys., vol. 72, no. 9, pp. 4399–4413, Nov. 1992.
- [13] A. Korotkov and K. Likharev, "Single-electron-parametron-based logic devices," J. Appl. Phys., vol. 84, no. 11, pp. 6114–6126, Dec. 1998.
- [14] C. Wasshuber, H. Kosina, and S. Selberherr, "SIMON—A simulator for single-electron tunnel devices and circuits," IEEE Trans. Computer- Aided Design, vol. 16, pp. 937–944, Sept. 1997.
- [15] A. B. Zorin, S. V. Lotkhov, H. Zangerle, and J. Niemeyer, "Coulomb blockade and cotunneling in single electron circuits with on-chip resistors: Toward the implementation of the R pump," J. Appl. Phys., vol. 88, no. 5, pp. 2665–2670, Sept. 2000.
- [16] T. Oya, T. Asai, T. Asai, T. Fukui, and Y. Amemiya, "A majority-logic device using and irreversible single-electron box," IEEE Trans. Nanotechnol., vol. 2, pp. 15–22, Mar. 2003.
- [17] Z. Durrani, A. Irvine, and H. Ahmed, "Coulomb blockade memory using integrated single-electron transistor/metal–oxide–semiconductor transistor gain cells," IEEE Trans. Electron Devices, vol. 47, pp. 2334–2339, Dec. 2000.
- [18] J. Millman and C. C. Halkias; Integrated Electronics- Analog and Digital Circuits and Systems\_ McGraw Hill Education; 2 edition
- [19] Millman's Electronic Devices & Ciruits 4th Edition (English, Paperback, Millman Jacob)

## BIOGRAPHY

Anup Kumar Biswas is an Assistant Professor in the department of Commuter Science and Engineering in Kalyani Govt. Engineering College. He is awarded his PhD [Engg.] degree in the stream of Electronics and Telecommunication Engineering from Jadavpur University in the year 2006. He has engaged in teaching and research activities since the last 16 years. His Specialization field is nanotechnology especially single electron tunneling technology. Dr. Biswas has published several papers in various national, international conferences and journals.

