CHAPTER 4
8T & 9T 1-BIT FULL ADDER CIRCUITS: A REVIEW

4.1 3T XOR GATE:

The 3T XOR gate [20], [22] is implemented using a modified version of a CMOS inverter and a pMOS pass transistor. The schematic of 3T XOR gate is shown in Fig. 4.1.

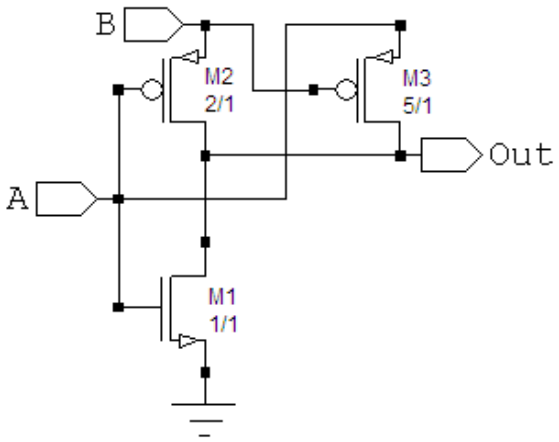

Fig. 4.1 Schematic of existing 3T XOR Gate

When the input B is at logic high, the inverter functions like a normal CMOS inverter. There is threshold loss when input B is at logic low. When the input B is at logic low, the CMOS inverter output is at high impedance. Here, the pass transistor M3 is enabled and the output terminal Out will get the same logic value as input A. The operation of the whole circuit is thus like a 2-input XOR gate. When A=0 and B=0, (logic '0' representing Ground and logic '1' representing Input supply voltage) there is voltage degradation due to threshold drop that occurs across transistor M3 and consequently the output 'Out' is degraded with respect to the input. A second problem of current feedback through transistor M1 also occurs when A=1 and B=0.

The output obtained is the parallel combination of complete '1' given by M3 and complete 0 given by M1. Also the output of the pass transistor is fed back through transistor M1 that is operating in the active region since its gate has a logic high input. Hence, when input B=0, output degrades with increasing value of input A.

One can manipulate the aspect ratios (W/L) of pMOS and nMOS transistors to solve these problems until threshold loss is minimized and an acceptable logic level is restored.

$$V_t = V_{t0} + \gamma\left(\sqrt{V_{SB} + \emptyset_0} - \sqrt{\emptyset_0}\right) - \alpha_l \frac{t_{ox}}{L}(V_{SB} + \emptyset_0) - \alpha_v \frac{t_{ox}}{L} V_{DS} + \alpha_w \frac{t_{ox}}{W}(V_{SB} + \emptyset_0) \quad (4.1)$$

Where, V_{t0} is the zero bias threshold voltage $\gamma = \left(\sqrt{2qN_A\varepsilon_{si}}\right)/C_{ox}$ is bulk threshold coefficient. $\emptyset_0$ is $2\emptyset_F$, where $\emptyset_F$ is the Fermi potential, t_{ox} is the thickness of the oxide layer. α_l, α_v and α_w are process dependent parameters.

The part of Eq. (4.1) [30] shows that the transistor channel width W has inverse effect on the threshold voltage. Channel length L is technology dependent. Therefore, it is evident that by increasing W, it is possible to decrease the threshold loss.

From Eq. (4.1) this degradation due to threshold drop can be considerably minimized by increasing the W/L ratio of transistor M3, increasing the channel width keeping the channel length constant. Also the difficulty in output when input is "10" can be minimized by decreasing the W/L ratio of transistor M1.

The output levels of 3T XOR cell justifying the above working are stated in Table II and also graphically represented in Fig. 4.2 as input-output waveform for all possible input combinations.

Table II. Performance Table of Existing 3T XOR cell

A (Volt)	**B (Volt)**	**Expected Output (Volt)**	**Obtained Output (Volt)**
0	0	0	-0.13
0	1	1	0.99
1	0	1	0.76
1	1	0	0

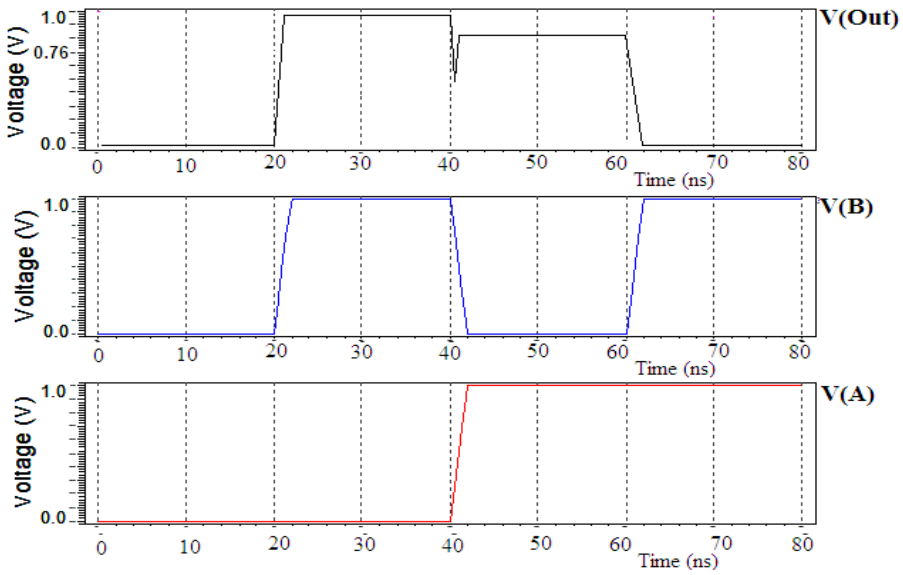

Fig.4.2 In-Out Waveform of existing3T XOR gate

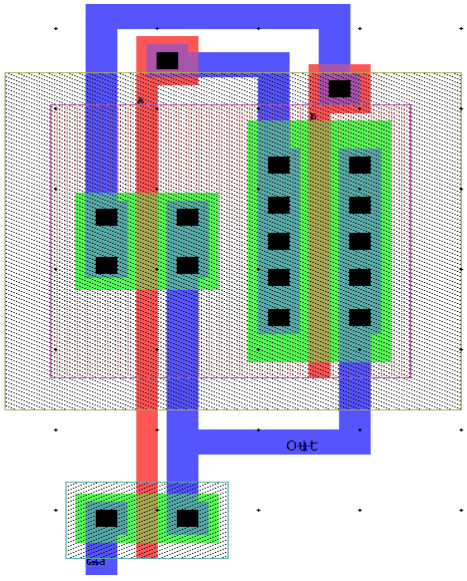

Fig. 4.3 Layout Design of existing 3T XOR Gate

Fig. 4.3 shows the layout design of existing 3T XOR gate designed at 45nm technology. In all the layout designs present in this report multiple active contacts are used for reduction of diffusion resistance. The source diffusion and the drain diffusion are filled with the maximum number of contacts to reduce the resistance of the connection from the metal to the diffusion, and to maximize the amount of current that can flow

through the contacts. Multiple contacts are not implemented for the MOSFETs having small aspect ratio as it violates the design rule for optimum distance between the contacts [7].

4.2 8T 1-BIT FULL ADDER:

The 8T full adder is designed by using conventional adder technique shown in Fig. 3.1 using combination 1 from Table I. i.e. two 3T XOR gate and a 2T multiplexer. The Sum output is obtained by a cascaded exclusive ORing of the three inputs and C_{out} module is implemented using 2T multiplexer. Given the three 1-bit inputs A, B, and C_{in}, it is desired to calculate the two 1-bit outputs Sum and Carry. The output Sum and C_{out} can be calculated using Eq. (4.2) and (4.3) respectively.

$$Sum = A \oplus B \oplus C_{in} \tag{4.2}$$

When $A \oplus B=0$;

$$C_{out} = A,$$

When $A \oplus B=1$;

$$C_{out} = C_{in}, \tag{4.3}$$

The implementation of this logic [20], [22] is shown in Fig.4.4. It is quite evident from the schematic of this adder that two stage delays are required to obtain Sum and Carry outputs.

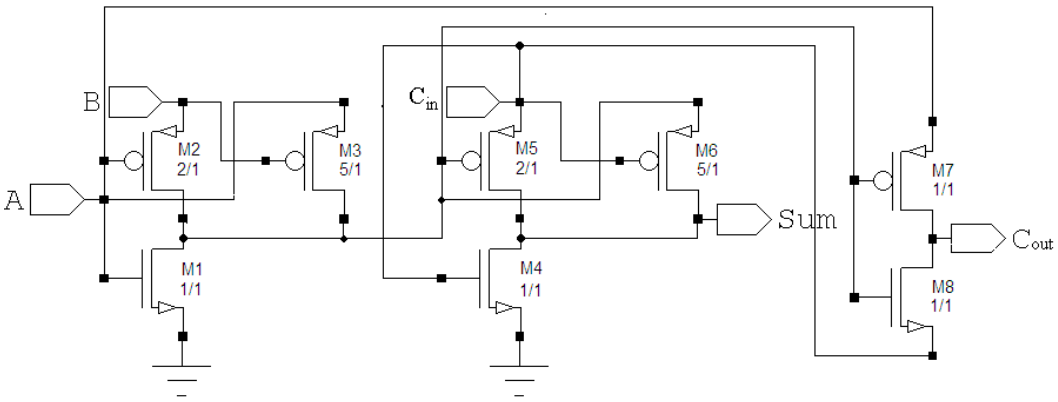

Fig. 4.4 Schematic of existing 8T Full Adder

The voltage drop due to the threshold loss in transistors M3 and M6 in Fig. 4.4 can be minimized by suitably increasing the aspect ratios of the two transistors. However, the threshold voltage drop of $|V_{tp}|$ provided by the pMOS pass transistor M3 when A=0 and B=0 is used to turn on the nMOS pass transistor M8 and therefore we get an output voltage equal to $|V_{tp}| - V_{tn}$, where $|V_{tp}|$ is the threshold voltage of the pMOS transistor and V_{tn} is the threshold voltage of the nMOS transistor. For input "010", output of the first stage XOR gate is complete "1". However in the second stage nMOS M4 and nMOS M6 get ON simultaneously. A close loop forms and current feedback through M4 occurs and the output Sum is degraded much less than V_{DD} (approx. $V_{DD}/2$). Similarly for input "100", expected Sum should be "1" and C_{out} to be "0". But the first stage XOR gate gives a degraded output less than V_{DD} (approx. $V_{DD}/2$). However at higher input voltages (greater than 0.6v) M1 & M3 and M4 & M6 get ON simultaneously. So, there should be minimum loss at first XOR gate, and driving capability of M6 should be high enough than M4. Also for inputs "000" and "110", output of first XOR gate is "0" and when C_{in}=0, this enables transistors M2 and M3 simultaneously giving a degraded Sum output. As it can be seen this adder design is confronted with serious problems especially when C_{in}=0. The outputs have good logic level for only a four input vectors. For the remaining input vectors, there is a major degradation in output voltage that may lead to functional failure as well as increased power consumption. As the voltage is scaled down, the signal integrity deteriorates and the speed decreases tremendously. Fig. 4.5 shows the layout design implemented using 45nm low power process technology. The obtained output levels are given in Table III and illustrated as In-Out waveform in Fig. 4.6.

Table III. Performance Table for Sum Module of Existing 8T Full Adder

A	**B**	**C_{in}**	**Sum**
0	0	0	<< $\lvert V_{Tp} \rvert$
0	0	1	1
0	1	0	***50% of logic "1"***
0	1	1	0
1	0	0	***50% of logic "1"***
1	0	1	0
1	1	0	<< $\lvert V_{Tp} \rvert$
1	1	1	1

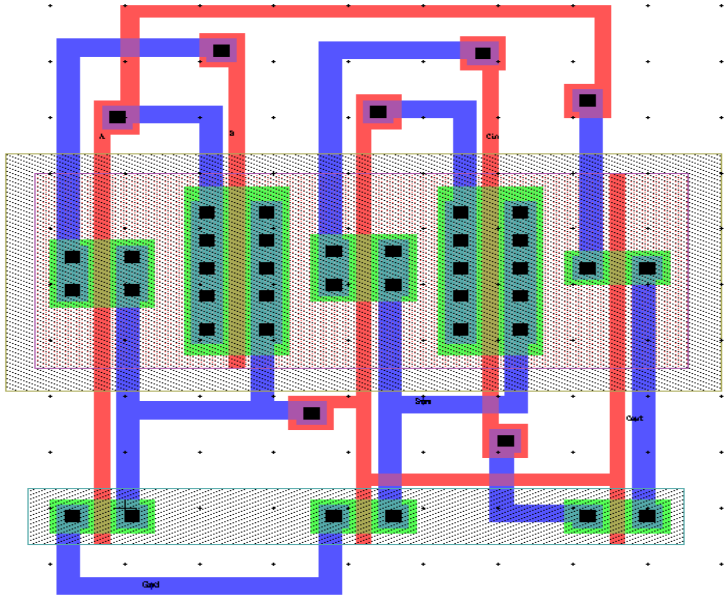

Fig. 4.5 Layout Design of existing 8T Full Adder

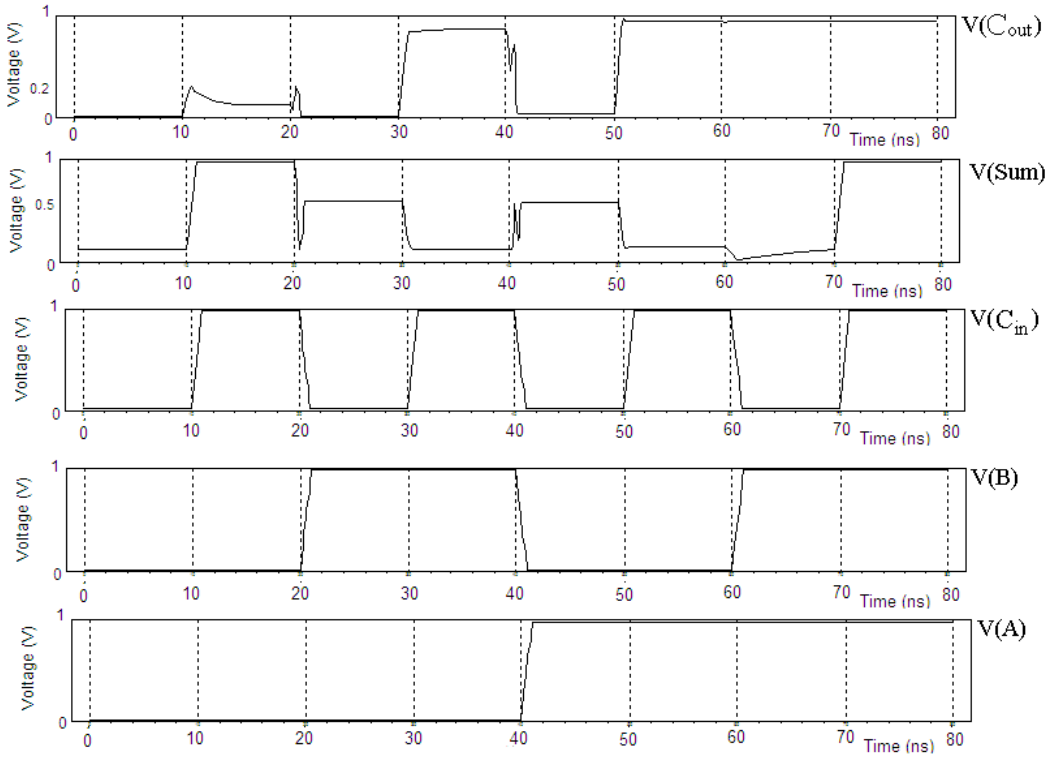

Fig.4.6 In-Out Waveform of existing 8T Full Adder

For inputs, '010' and '100' Sum bit is nearly equal to 50% of total amplitude which is very less to be correctly interpreted as logic high. Thus, using this design further in complex applications can give rise to this threshold loss and result into incorrect output.

4.3 9T 1-BIT FULL ADDER BASED ON CARRY LOGIC:

This technique for implementing full adder is different from conventional full adder design technique as it uses only input and output carry signals to generate Sum output [31], [32]. The Sum output is based on XORing between inputs A and B and thus, Sum and Carry output of the proposed adder cell are expressed as;

When $A \oplus B = 0$,

$$\text{Sum} = C_{in};\ C_{out} = A \tag{4.4}$$

When $A \oplus B = 1$,

$$\text{Sum} = \overline{C_{out}}\ ;\ C_{out} = C_{in} \tag{4.5}$$

Table IV given below states the dependency of Sum output on C_{in} and C_{out} signals.

Table IV. Truth Table of 9T Full Adder based on Carry Logic

A	B	C_{in}	Sum		C_{out}
			$A \oplus B$		
0	0	0	0	C_{in}	0
0	0	1	0	C_{in}	0
0	1	0	1	$\overline{C_{out}}$	0
0	1	1	1	$\overline{C_{out}}$	1
1	0	0	1	$\overline{C_{out}}$	0
1	0	1	1	$\overline{C_{out}}$	1
1	1	0	0	C_{in}	1
1	1	1	0	C_{in}	1

This logic for full adder cell is implemented and utilizes nine transistors. Fig. 4.7 shows the schematic of 9T full adder cell. In this design, C_{out} is implemented by using a 2T multiplexer which is controlled by output of 3T XOR gate and passes either A or C_{in} accordingly.

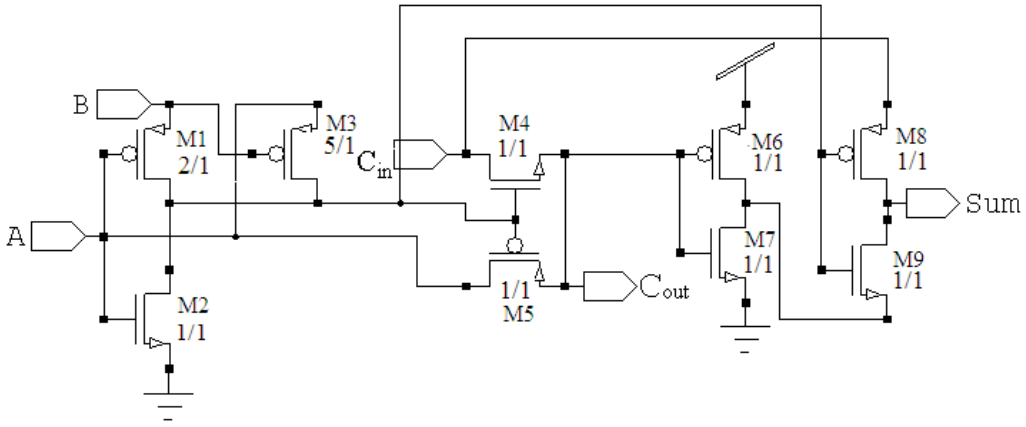

Fig. 4.7 Schematic of existing 9T full adder cell based on carry logic

Input to pMOS transistor M5 is A while to nMOS transistor M4, input is C_{in}. An inverter is connected at the output of C_{out} to generate its complement on which the Sum output is dependent. As stated above in Eq.(4.4) and (4.5) when A⊕ B is '0', Sum output is equal to input C_{in} and when A⊕ B is '1', Sum output is equal to $\overline{C_{out}}$. Thus, the Sum output is obtained by transferring C_{in} and $\overline{C_{out}}$ through a 2T multiplexer. Input to pMOS transistor M8 of this MUX is C_{in} while to nMOS transistor M9 of this MUX, input is $\overline{C_{out}}$. This 2T multiplexer is also controlled by output of 3T XOR gate i.e. A⊕ B.

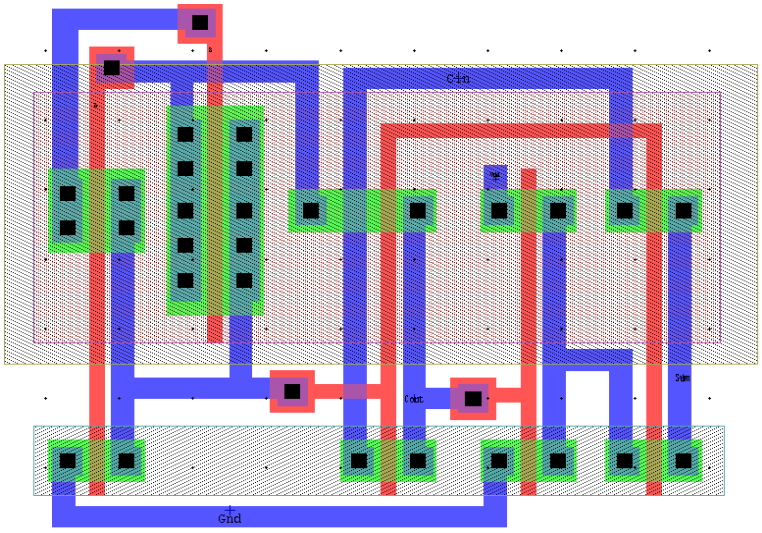

Fig. 4.8 Layout of existing 9T full adder cell based on carry logic

The substrate terminals of all the circuits included in the report are connected to their respective source terminals in order to nullify the substrate-bias effect. The layout shown in Fig. 4.8 has been designed 45nm technology. Fig. 4.9 illustrates its In-Out waveform.

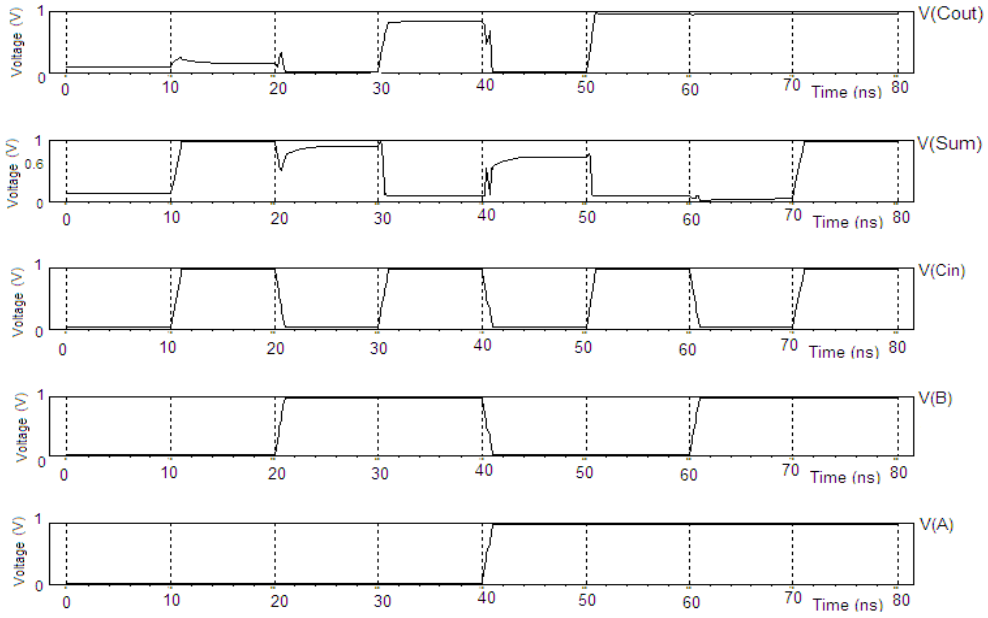

Fig. 4.9 In-Out waveform of existing 9T full adder based on carry logic

4.4 9T 1-BIT FULL ADDER BASED ON INVERSION LOGIC:

This technique is based on implementing XOR and XNOR operation between inputs B and C_{in}. An inverter is connected at the output of XOR gate to generate XNOR function [25], [33]-[35]. The truth table for this logic is stated in Table V.

Table V. Truth Table of 9T Full Adder based on Inversion Logic

A	B	C_{in}	$B \oplus C_{in}$	$B \odot C_{in}$	Sum	C_{out}
0	0	0	**0**	1	$B \oplus C_{in}$	0
0	0	1	**1**	0	$B \oplus C_{in}$	0
0	1	0	**1**	0	$B \oplus C_{in}$	0
0	1	1	**0**	1	$B \oplus C_{in}$	1
1	0	0	0	**1**	$B \odot C_{in}$	0
1	0	1	1	**0**	$B \odot C_{in}$	1
1	1	0	1	**0**	$B \odot C_{in}$	1
1	1	1	0	**1**	$B \odot C_{in}$	1

It is evident from Table V that for three 1 bit inputs A, B and C_{in}, the Sum and Carry outputs can be calculated from Eq. (4.6) and (4.7) as given below

When A = 0,

$$Sum = B \oplus C_{in}$$

When A =1,

$$Sum = B \odot C_{in} \quad (4.6)$$

When $B \oplus C_{in} = 0$,

$$C_{out} = A$$

When $B \oplus C_{in} = 0$,

$$C_{out} = C_{in} \quad (4.7)$$

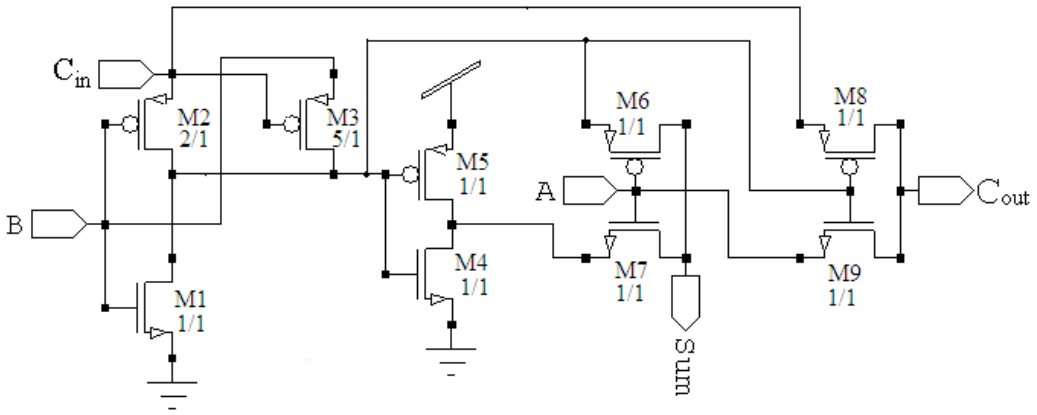

Fig. 4.10 Schematic of existing 9T full adder cell based on inversion logic

The schematic of 9T full adder cell based on inversion logic is shown in Fig. 4.10. For generating the Sum output in this design, the truth table has been divided into two parts, one for input A= '0' and another for A= '1' rather than implementing the conventional Sum module. From the truth table shown in Table V it is obvious that when A= '0', Sum can be produced by XORing inputs B and C_{in}. Similarly, when A= '1', Sum is showing the XNORing between inputs B and C_{in}. Therefore, the operation of Sum module is based on implementing XOR operation and XNOR operation between inputs B and C_{in}. An inverter is connected at the output of first stage XOR gate to generate XNOR function. Finally the Sum is implemented by transferring these output levels through 2T multiplexer. Input to pMOS transistor M6 is XOR of B and C_{in} while to nMOS M7, input

is XNOR of B and C_{in}. This 2T multiplexer is controlled by input A. C_{out} is implemented by using another 2T multiplexer which is controlled by output of first stage XOR gate and passes either A or C_{in} accordingly. Fig. 4.11 and 4.12 shows the layout and In-Out waveform of this 9T full adder cell.

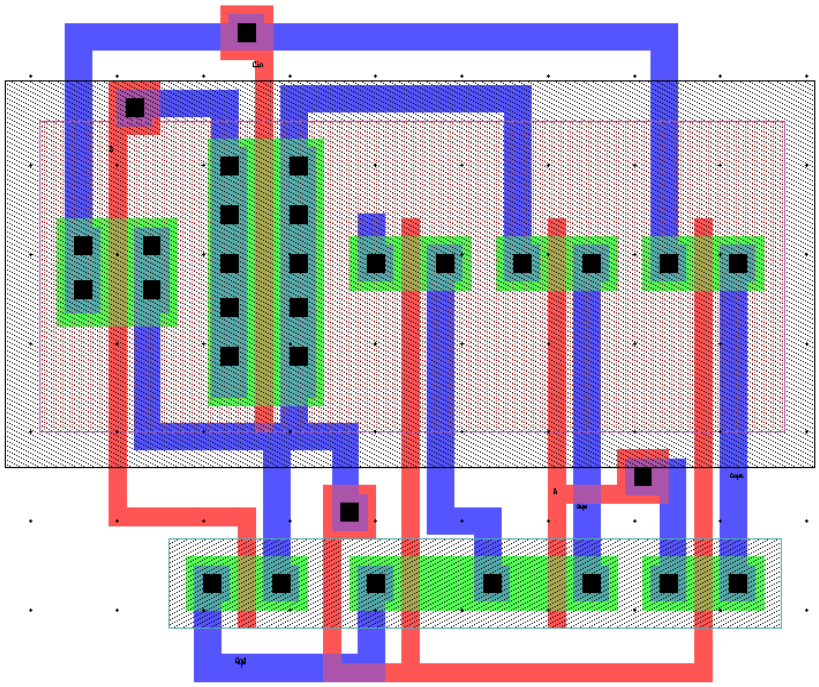

Fig. 4.11 Layout Design of existing 9T full adder cell based on inversion logic

The main idea behind designing the 9T full adders is to reduce the problem of threshold loss occurring in 8T full adder cell thereby improving the power consumption. The 9T full adder circuit reduces threshold loss problem by 30% as revealed in the Input-Output waveforms of 8T and 9T full adder circuits. 9T Full Adder has area overhead of one transistor. The 8T adder uses two 3T XOR cells having transistors up to 5/1 aspect ratio but on the other hand 9T adder employs only one XOR cell and all other transistors incorporated in the circuit of 9T adder have 1/1 aspect ratio only. Therefore instead of increased MOS count by one it does not affected the area by large extent. In a nutshell, the 9T full adder has better performance than 8T full adder.

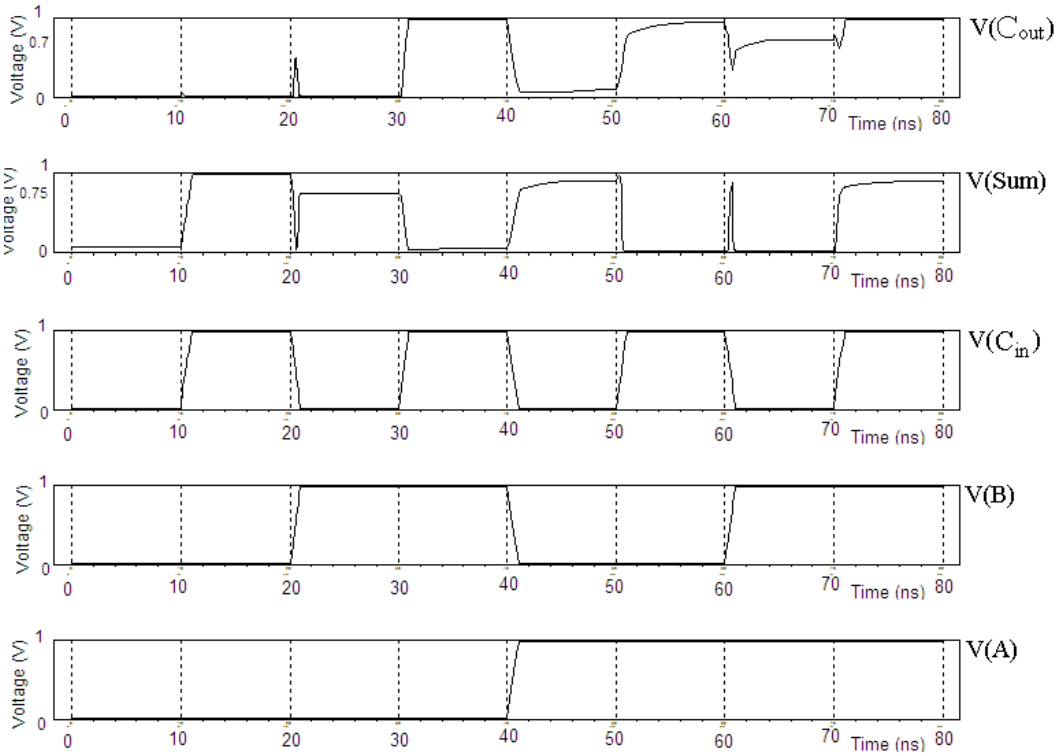

Fig. 4.12 In-Out Waveform of existing 9T full adder cell based on inversion logic

CHAPTER 5
PROPOSED PMOS BASED 3T XOR GATE

The XOR gate is the basic building block of a full adder circuit. The XOR gate can be implemented using AND, OR, and NOT gates with high redundancy. Optimized design of this gate enhances the performance of VLSI systems as these gates are utilized as sub blocks in larger circuits. Thus, design with less number of transistors; lesser power consumption and delay are highly desirable for efficient implementation of the large VLSI system.

5.1 PROPOSED PMOS BASED 3T XOR GATE:

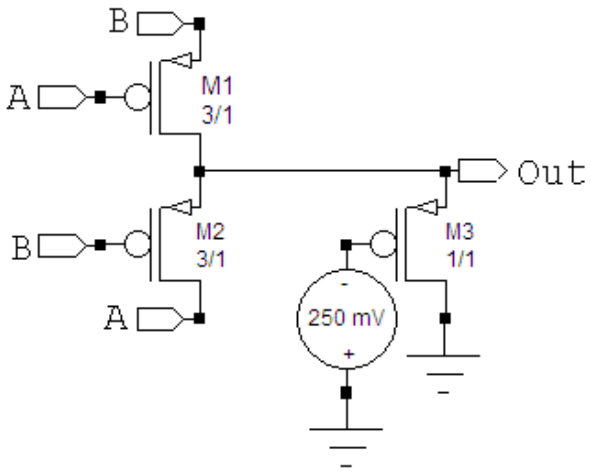

Fig. 5.1 Proposed pMOS based 3T XOR Gate

The design and performance of full adder is totally dependent on the design of XOR gate. Therefore, XOR cell being a heart of full adder needs to be improved for the betterment of full adder performance parameters. The schematic of proposed 3T XOR cell [36] is shown in Fig. 5.1. It consists of three pMOS transistors and a negative input supply voltage of 250 mV is given to M3 so that it eliminates the negative voltage spikes

for input combination '00'. The W/L ratios of transistors M1and M2 are taken as 3/1 to minimize the affect of M3 and to take output node at logic high.

Case1: When AB=00, all transistors are ON and as pMOS is weak '0' device, it will pass low logic signal with threshold loss. But at the same time due to parallel resistances of ON transistors there will be slight reduction in threshold loss and nearly 4% degradation in the output is obtained.

Case 2: When AB=01 and AB =10, M1 & M2 are ON respectively and pMOS being strong '1' device will allow to pass complete logic high at the output. But as M3 is always and connected to ground, hence it will try to pull the output node towards ground. Therefore to overcome the affect of M3 and to charge the output node at logic high, aspect ratios of M1 and M2 are increased to 3/1 with respect to M3 i.e.; 1/1.

Case 3: For AB=11, only M3 is ON and it will pass low logic signal with threshold loss.

The performance table shown in Table VI illustrates the small degradation in the output voltage with respect to the full scale input voltage value which can be easily interpreted as logic '1'. However, the proposed design for XOR gate gives better performance than existing one. Fig. 5.2 and Fig.5.3 shows the layout design at 45nm technology and In-Out waveform of proposed 3T XOR gate respectively.

Table VI. Performance Table of Proposed pMOS 3T XOR cell

A (Volt)	**B (Volt)**	**Expected Output (Volt)**	**Obtained Output (Volt)**
0	0	0	0.047
0	1	1	0.84
1	0	1	0.84
1	1	0	0.073

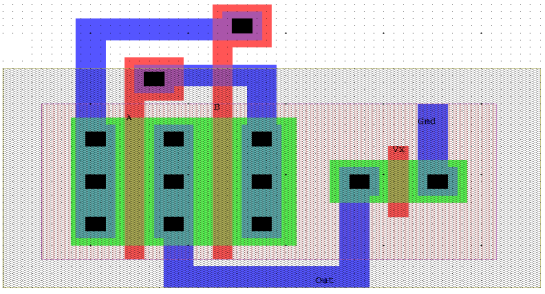

Fig. 5.2 Layout Design for proposed pMOS based 3T XOR Gate

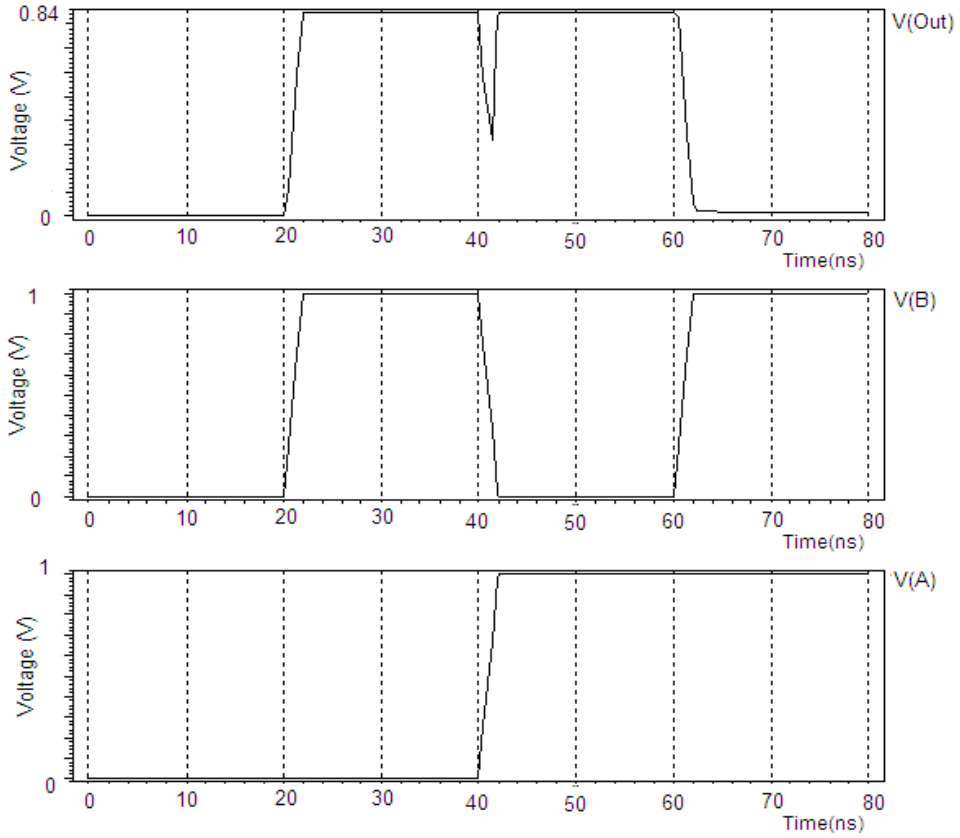

Fig. 5.3 In-Out waveform of proposed pMOS based 3T XOR gate

This proposed 3T XOR cell has also been designed with negative voltage supply of 440mV at the gate of transistor M3 [36]. The output voltage amplitudes for both the supply voltages are approximately same but when the inputs are having transitions from 0 to 1 and 1 to 0 the output of 440mV cell is showing large negative spike tending towards 0V. Also there is slight reduction in power consumption while using -250mV supply. Therefore, design using -250mV is taken for further analysis and applications.

For a pMOS transistor, Fermi potential $\emptyset_F$ [37]-[39] is expressed below as the temperature sensitive parameter;

$$\emptyset_F = -\frac{KT}{q}\ln\frac{N_{sub}}{n_i} \tag{5.1}$$

Where, K is Boltzmann's constant, T is the absolute temperature, q is the electron charge, N_{sub} is the substrate doping concentration and n_i is the intrinsic carrier concentration with a temperature dependence given by;

$$n_i = 3.88 \times 10^{16} T^{3/2} \exp\left(-\frac{E_{g0}}{2KT}\right) \quad (5.2)$$

Therefore, the gradient of $\emptyset_F$ with respect to the temperature is given by;

$$\frac{d\emptyset_F}{dT} = \frac{1}{T}\left[\emptyset_F + \frac{3}{2}\frac{KT}{q} + \frac{E_{g0}}{2q}\right] \quad (5.3)$$

The parameter $\emptyset_F$ always decreases in absolute value when the temperature increases. For a pMOS transistor, $\emptyset_F$ is negative and $d\emptyset_F/dT$ is positive while for nMOS, $\emptyset_F$ is positive and $d\emptyset_F/dT$ is negative. By assuming that the oxide charge is temperature independent, the temperature dependence of threshold voltage, V_{t0} is expressed as;

$$V_{t0} = \emptyset_{GB} - \frac{Q_{ox}}{C_{ox}} + 2\emptyset_F \pm \frac{Q_D}{C_{ox}} \quad (5.4)$$

Where, $\emptyset_{GB}$ is work function difference between channel and gate, Q_{ox} is charge density Q_D *is* depletion region charge density. Since, $\emptyset_F$ is negative for pMOS and is positive for nMOS, thus, it can be concluded from Eq. (5.4) given above that with increasing temperature, decrement in $\emptyset_F$ leads to decrement in threshold voltage of pMOS transistor and increment in threshold voltage of nMOS transistor. Thus, a gradual shift of threshold voltage over time is observed in pMOS transistor. This shift is caused by voltage stress on the gate oxide, temperature, and the duty cycle of the stressing voltage. This effect becomes more severe with reduced transistor dimensions and lower operating voltages. This phenomenon of negative shift of threshold voltage is called pMOS Negative Bias Temperature Instability (NBTI) [37], [38].

On the other hand, power consumption is proportional to threshold voltage thus; decrease in the value of threshold voltage will also reduce power consumption of the circuit.

As the proposed 3T XOR gate is implemented by pMOS transistor only thus, it will exhibit this effect. But as the existing design has been implemented using both pMOS and nMOS transistors, therefore it has the effect of pMOS and nMOS both depending upon the ON transistors at certain input combinations.

CHAPTER 6

PROPOSED 8T & 9T 1-BIT FULL ADDERS USING PMOS XOR CELL

6.1 PROPOSED 1-BIT 8T FULL ADDER USING PMOS XOR CELL:

Full adder circuit can be implemented with cascaded combinations of XOR gate and a multiplexer. Using the proposed pMOS 3T XOR cell [36] of Fig. 5.1 an 8T adder has also been realized [40]. The proposed design of full adder has been implemented using conventional technique of cascaded 3T XOR gate and 2T MUX in accordance with Eq. (3.5) and (3.6) as described in Chapter-3. Fig. 6.1 shows the schematic of proposed 8T full adder cell.

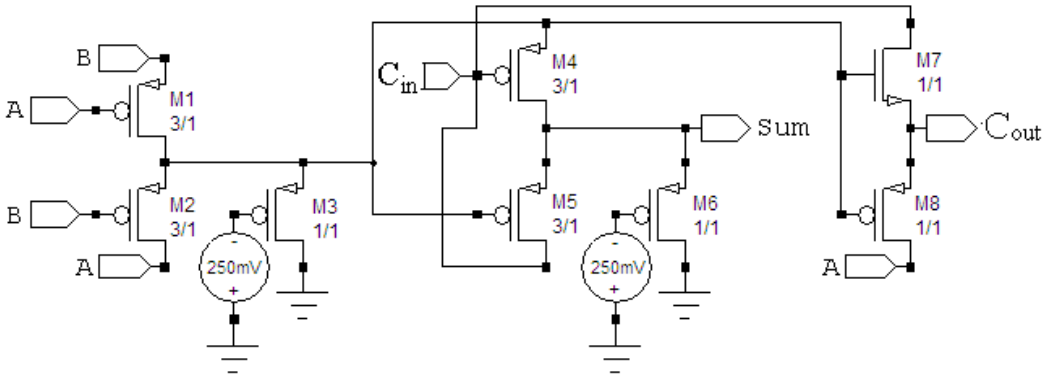

Fig. 6.1 Schematic of Proposed pMOS based 8T Full Adder

In this proposed full adder, due to the negative input supply voltage of 250mV given to the gate of M3 and M6, both transistors are always ON.

For inputs '000', all transistors are ON and pMOS being weak '0' device passes low logic signal with threshold loss of about 2%.

For inputs '001' logic low output at first stage XOR turns ON M5 which allows to pass complete logic high to the Sum output but at the same time as M6 is also ON, it

pulls Sum output to logic low. Due to higher aspect ratio of M5, Sum output will be charged to logic high and due to the parallel resistance of M5 and M6, output will be slightly degraded to 90% of logic '1'. Similar case happens with inputs '111'.

For inputs '010', first stage XOR gate passes degraded logic high output through M1 and M3 to the source terminal of M4. In the second stage XOR gate, M4 and M6 are ON therefore, it will pass more degraded output of about 70% than first stage XOR gate. Similar case happens with inputs '100'. The proposed full adder has overcome the threshold loss problem for these two input combinations as compared to existing design.

For inputs '011' and '101', first stage XOR gate passes degraded logic high which will not turn ON either M4 or M5, hence only M6 remains ON which pass low logic signal with threshold loss of about 6%.

For '110', first stage XOR gate passes low logic signal through M3 and turns ON M4 and M5. Thus, all the pMOS of second stage are ON and pMOS being weak '0' device passes low logic signal with threshold loss of about 2%.

Table VII illustrates the output levels of proposed full adder and its In-Out waveform is shown in Fig. 6.3.

Table VII. Performance Table for Sum Module of Proposed pMOS based 8T Full Adder cell

A	**B**	**C_{in}**	**Sum**
0	0	0	2% of logic "0"
0	0	1	90% of logic "1"
0	1	0	***70% of logic "1"***
0	1	1	6% of logic "0"
1	0	0	***70% of logic "1"***
1	0	1	6% of logic "0"
1	1	0	2% of logic "0"
1	1	1	90% of logic "1"

The proposed 8T full adder circuit operates efficiently in super threshold region to achieve low power. This circuit improves threshold loss up to 20% when compared with existing 8T full adder as shown in Table III. Although, it shows small voltage drop for certain input combinations but this drop is so less that it can be assumed as logic high. It also improves PDP, output noise voltage and temperature sustainability which is of great

interest in the complex circuit design. Fig. 6.2 shows the layout design at 45nm technology. The number of poly contacts and metal overlap are also reduced in the proposed design which will lead to reduced capacitance and thus power consumption.

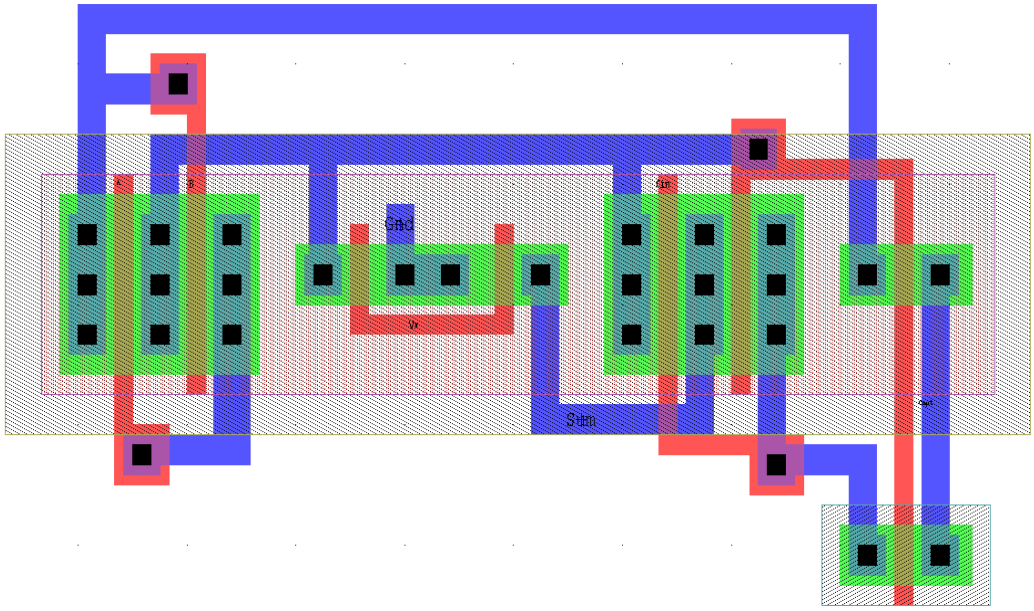

Fig. 6.2 Layout Design of proposed pMOS based 8T Full Adder

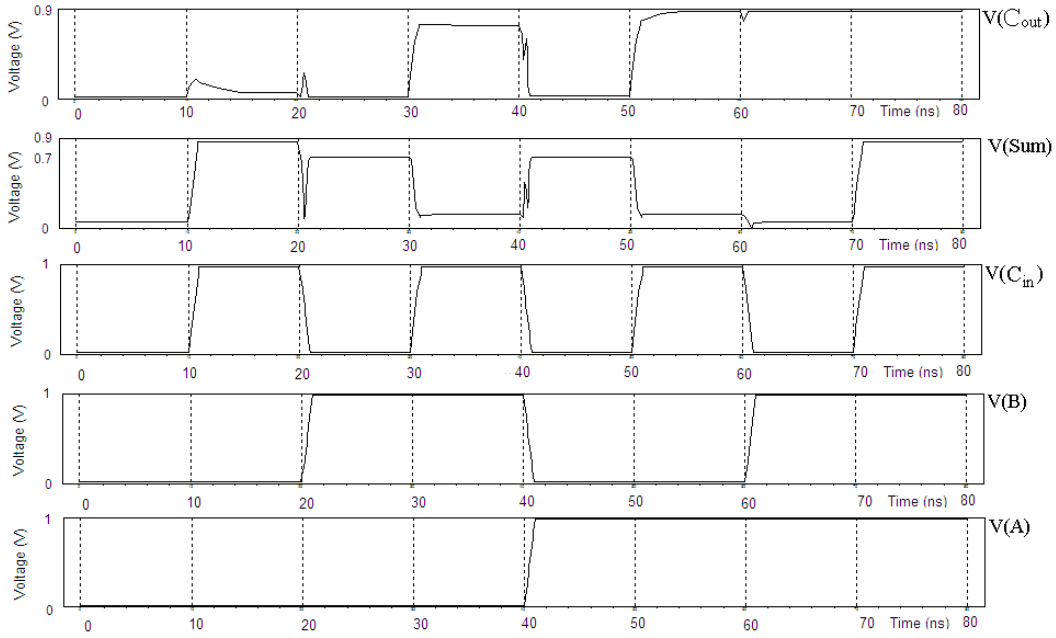

Fig. 6.3 In-Out Waveform of proposed pMOS based 8T full adder

6.2 PROPOSED 1-BIT 9T FULL ADDER BASED ON CARRY LOGIC:

The XOR gate is the basic building block of a full adder circuit. Using the proposed pMOS 3T XOR cell [36] of Fig. 5.1 a 9T adder based on carry logic [41] as discussed in Chapter-3 has also been realized. Fig. 6.4 shows the schematic of 9T full adder cell implemented in accordance with Eq. (3.5) and (3.6) and Table IV.

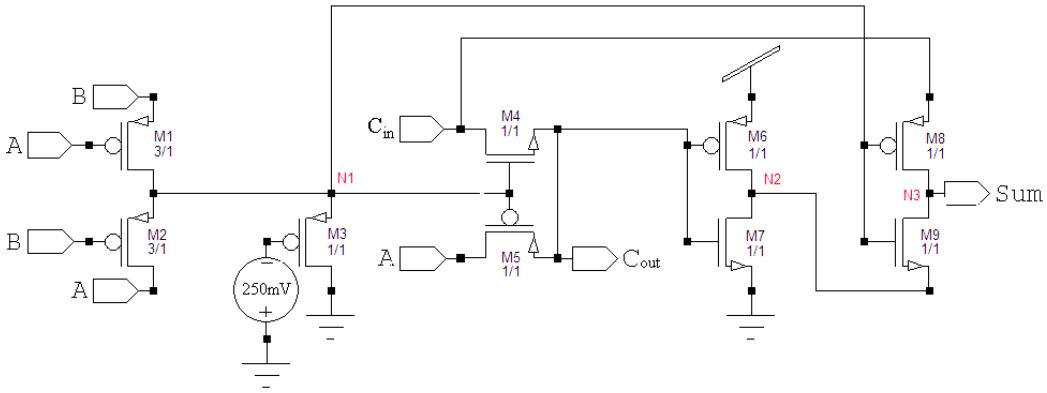

Fig. 6.4 Schematic of proposed pMOS 9T full adder cell based on carry logic

Table VIII. Power Consuming Transitions (Switching Activity) in Proposed pMOS 9T Full Adder Cell Based on Carry Logic

Power Consuming Transitions	**Node N1**	**Node N2**	**Node N3**	**Total**
	1	1	3	5

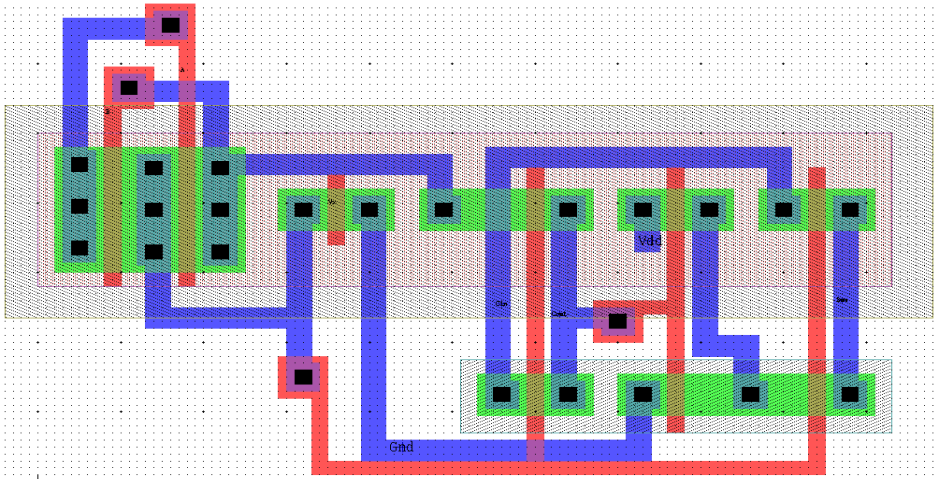

Fig. 6.5 Layout of proposed pMOS 9T full adder cell based on carry logic

The layout for this circuit has been designed 45nm technology. Fig 6.5 and Fig. 6.6 show its layout design designed at 45nm technology and In-Out waveform respectively.

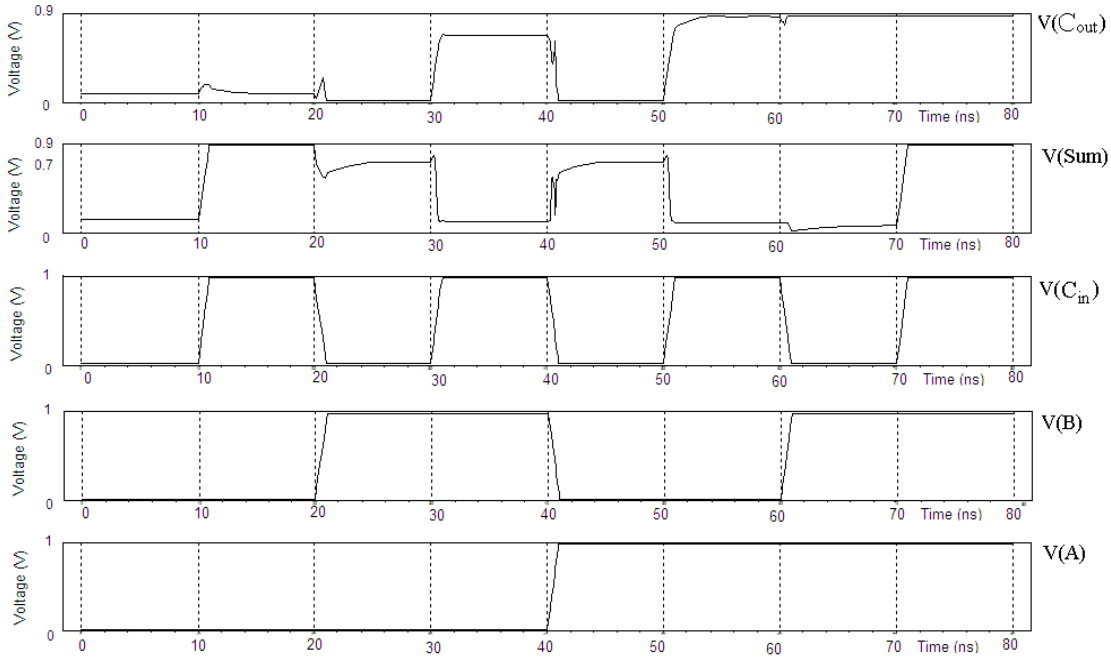

Fig. 6.6 In-Out Waveform of proposed pMOS 9T full adder cell based on carry logic

6.3 PROPOSED 1-BIT 9T FULL ADDER BASED ON INVERSION LOGIC:

The schematic of 9T full adder cell based on inversion logic [41] is shown in Fig. 6.7 implemented in accordance with Eq. (3.7) and (3.7) and Table V.

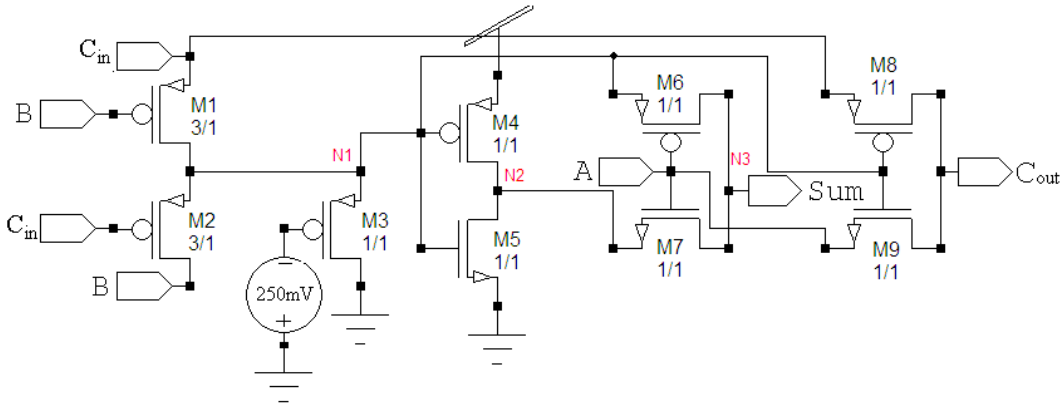

Fig. 6.7 Schematic of proposed pMOS 9T full adder cell based on inversion logic

Table IX. Power Consuming Transitions (Switching Activity) in Proposed pMOS 9T full Adder Cell Based on Inversion Logic

Power Consuming Transitions	**Node N1**	**Node N2**	**Node N3**	**Total**
	2	2	3	7

Fig. 6.8 and 6.9 shows the layout designed at 45nm technology and In-Out waveform of this 9T full adder cell.

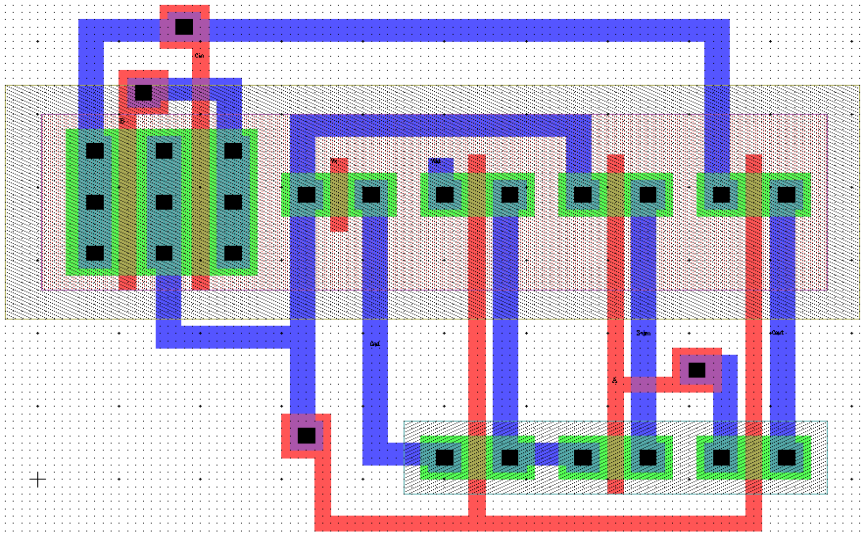

Fig. 6.8 Layout Design of proposed pMOS 9T full adder cell based on inversion logic

Table VIII and IX depicts the power consuming transitions at all the internal nodes of Sum module of both the proposed 9T full adders. Node N3 is same for both circuits as it the Sum node. The difference between both the proposed designs is at node N1 and N2 where the main logic has been implemented. The power consuming transition of 9T adder based on carry logic at node N1 and N2 are 50% of the transitions of 9T adder based on inversion logic. Therefore, the total power consuming transitions of 9T adder based on carry logic are less than that in 9T adder based on inversion logic. And, as discussed in Chapter-2 Eq. (2.1), the reduction of power consuming transition i.e.; switching activity will improve the total power consumption of the circuit which is further shown in the simulation results also.

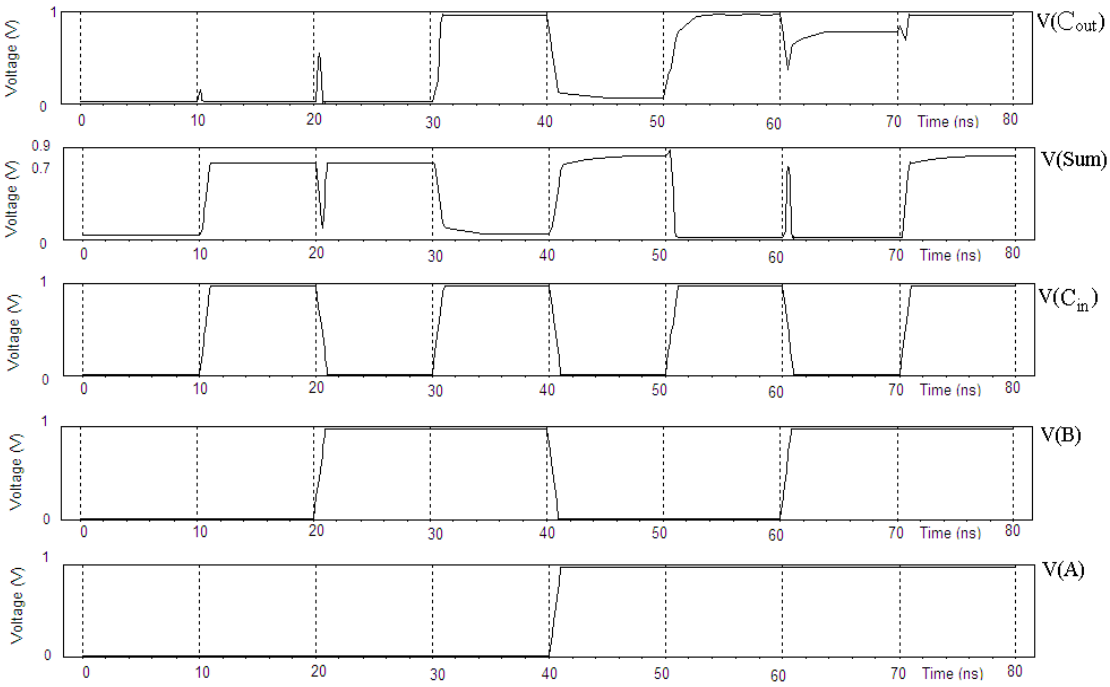

Fig. 6.9 In-Out Waveform of proposed pMOS 9T full adder cell based on inversion logic

CHAPTER 7
PROPOSED PTL 3T XOR GATE

As we know that the design of a full adder is based on the design of the XOR gate and a Carry out module, so we have proposed a 3T XOR gate to improve the basic building block and implemented it in proposed 1-bit 8T full adder cell.

7.1 PROPOSED PTL 3T XOR CELL:

The proposed design of XOR gate [42] is shown in Fig.7.1. It consists of 3 transistors, and input supply voltage (250mV) is given to the gate of nMOS. Due to the supply voltage the nMOS (M3) transistor is always ON. The Out terminal connected to all the Source/Drain of transistor.

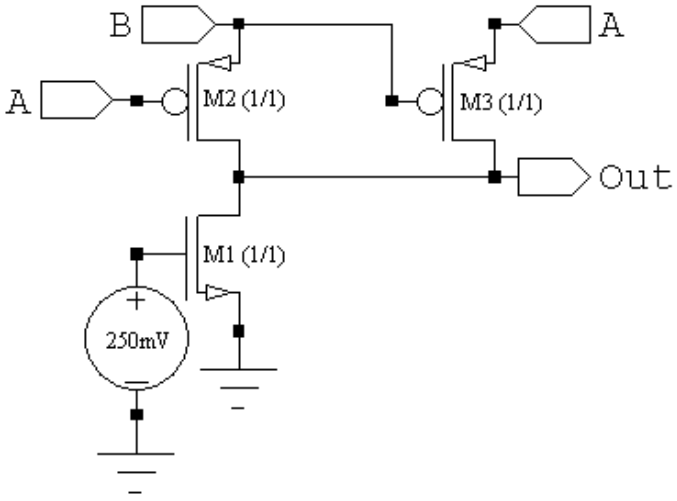

Fig.7.1 Proposed PTL 3T XOR Gate

Case 1: When AB=00, both the pMOS transistors are ON and nMOS is ON due to high gate voltage than threshold. As mobility of nMOS is nearly three times greater than pMOS, hence it will drive the` output ignoring the effect of ON pMOS transistors which results into complete zero output.

Case 2: When AB=01, pMOS M2 is ON and M3 is OFF. As pMOS is strong '1' device it will pass complete logic "high" signal at the output but as nMOS is always on in this design it will be responsible for degradation of voltage at the output. So the voltage at 'Out' terminal will be slightly less than logic "1". Similar is the case for input AB =10.

Case 3: When AB=11, both the pMOS transistors are OFF and only nMOS will be responsible for driving the output to complete zero. Table X describes the performance of proposed 3T XOR gate shown in Fig.7.1.

Table X. Performance Table of Proposed PTL 3T XOR cell

A(volt)	B(volt)	Expected Output (volt)	Obtained Output (volt)
0	0	0	0
0	1	1	0.985
1	0	1	0.985
1	0	0	0

Fig.7.2 shows the layout of proposed XOR gate and Fig.7.3 shows its input-output waveform.

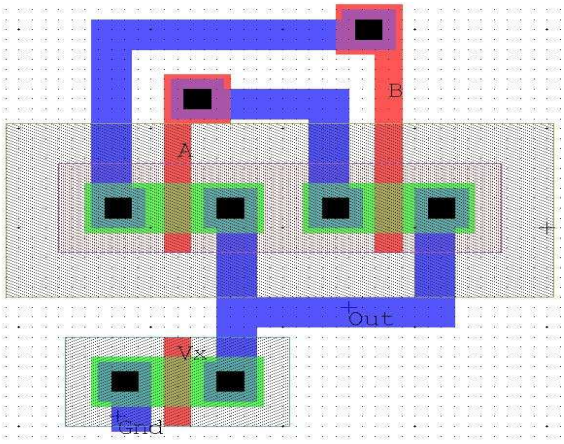

Fig.7.2 Layout Cell Design for proposed PTL 3T XOR gate

Table XI. Performance Table of Existing and Proposed 3T XOR cells

A (Volt)	B (Volt)	Expected Output (Volt)	Obtained Output (Volt)		
			Existing	Proposed pMOS	Proposed PTL
0	0	0	-0.13	0.047	0
0	1	1	0.99	0.84	0.985
1	0	1	0.76	0.84	0.985
1	1	0	0	0.073	0

The data stated in Table XI makes it obvious that the threshold loss has been reduced in the two proposed designs of XOR cells as compared to existing one. Also, the PTL based XOR cell is showing complete output at low levels and merely 1.5% degradation is observed at high level.

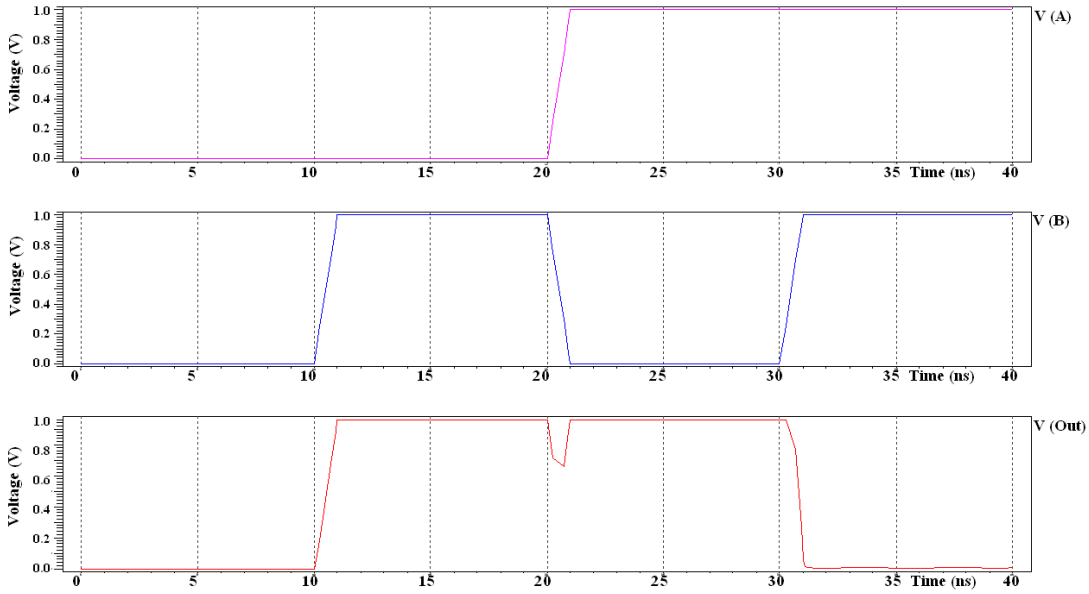

Fig.7.3 In-Out Waveform of proposed PTL 3T XOR Gate

CHAPTER 8
PROPOSED 8T & 9T 1-BIT FULL ADDERS USING PTL XOR CELL

8.1 PROPOSED 1-BIT 8T FULL ADDER USING PTL XOR CELL:

In the proposed 8T full adder [43] shown in Fig.8.1 the Sum module has been implemented using cascaded 3T XOR gate and C_{out} module is designed by 2T. When input AB=00 and 11, the output of first stage of XOR gate is exact '1', it is when XORed with C_{in}='0' results to complete output 'Sum'. However there is a 1.5% of threshold drop when C_{in}=1 for AB=00 and 11. Similarly when AB=01 and 10, the output of first XOR gate degrades to Input voltage$-V_{tn}$. Since pMOS is a strong '1' device it will pass complete logic "high" signal at the output node 'X' and as nMOS is strong '0' is always ON in this design it will pass compete '0', this results to the degraded output at node 'X'. As both nMOS and pMOS get enabled try to transfer opposite signals on the output result into voltage degradation. This Similar is the case for when AB =10, pMOS M3 is ON. However this drop is so less that it can easily be assumed as '1' and gives correct outputs for C_{in}=0 and 1.

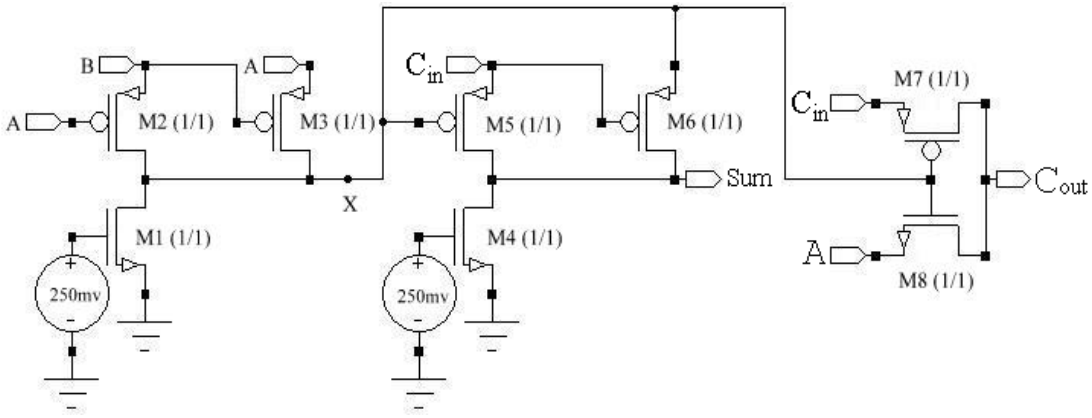

Fig.8.1 Proposed PTL based 8T Full Adder

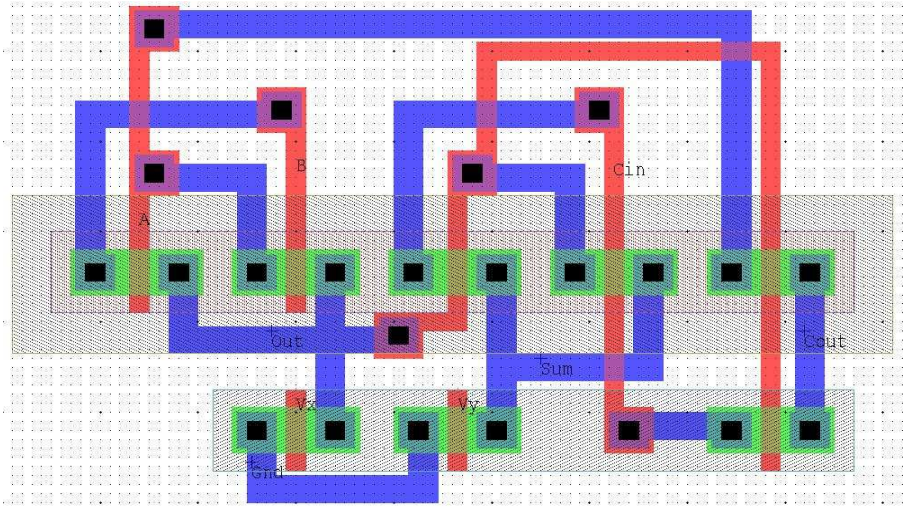

Fig. 8.2 Layout Cell Design for Proposed PTL based 8T Full Adder

Fig. 8.2 shows the Layout of proposed 8T and Fig. 8.3 shows its Input-Output waveform which reveals that the threshold loss is remarkably reduced in the proposed design than the existing 8T full adder design which is of great interest in the complex circuit design.

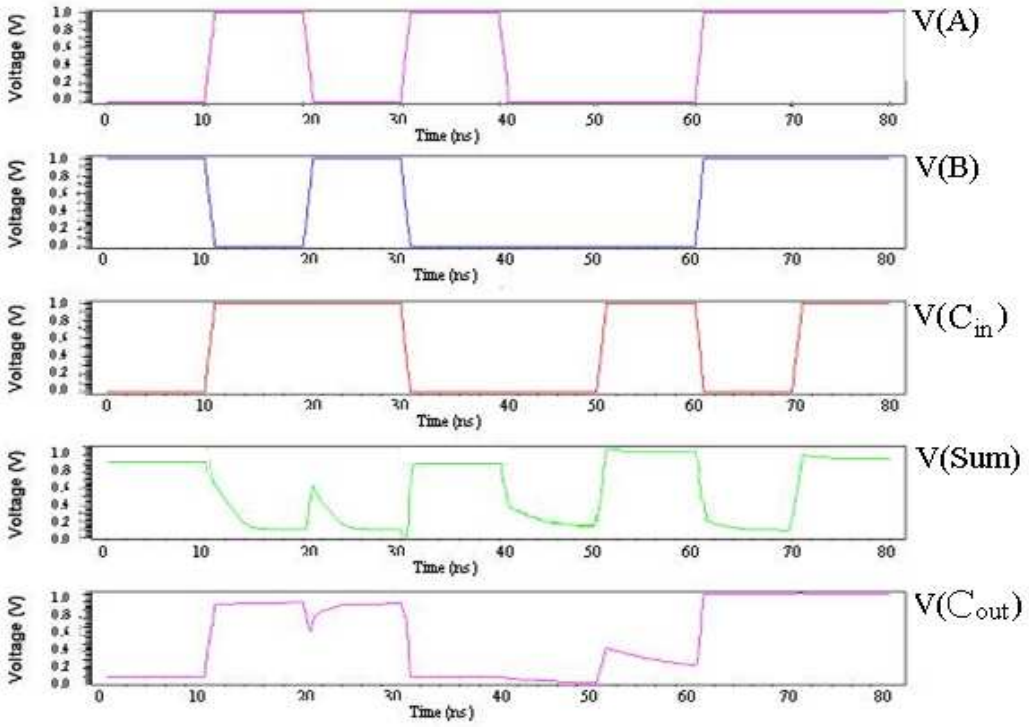

Fig.8.3 In-Out Waveform of Proposed PTL based 8T Full Adder Cell

8.2 PROPOSED 1-BIT 9T FULL ADDER BASED ON CARRY LOGIC:

The XOR gate is the basic building block of a full adder circuit. Using the proposed PTL 3T XOR cell of Fig. 8.1 the 9T adder based on carry logic as discussed in Chapter-3 has also been realized. Fig. 8.4 shows the schematic of 9T full adder cell implemented in accordance with Eq. (3.5) and (3.6) and Table IV.

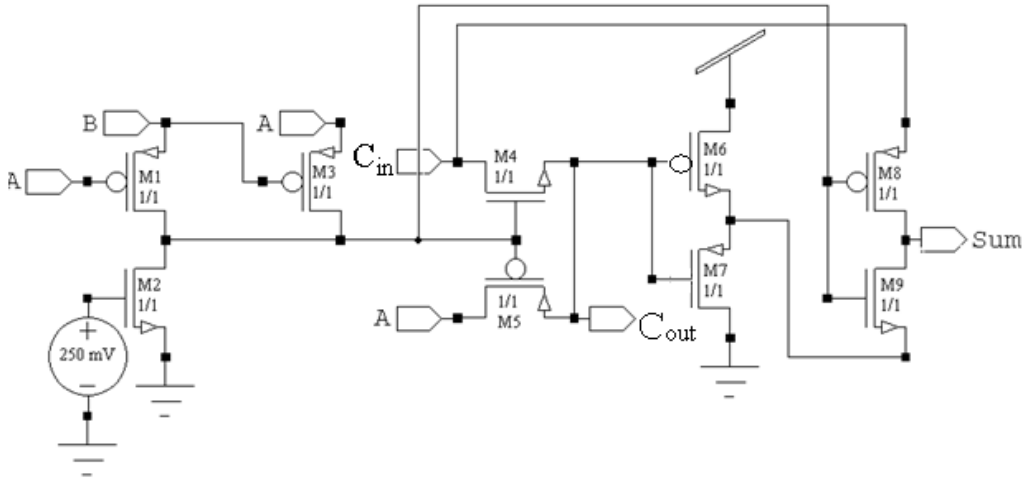

Fig. 8.4 Schematic of proposed PTL 9T full adder cell based on carry logic

Table XII. Power Consuming Transitions (Switching Activity) in proposed 9T full adder cell based on carry logic

Power Consuming Transitions	**Node N1**	**Node N2**	**Node N3**	**Total**
	1	1	3	5

The layout for this circuit has been designed 45nm technology. Fig 8.5 and Fig. 8.6 show its layout design designed at 45nm technology and In-Out waveform respectively.

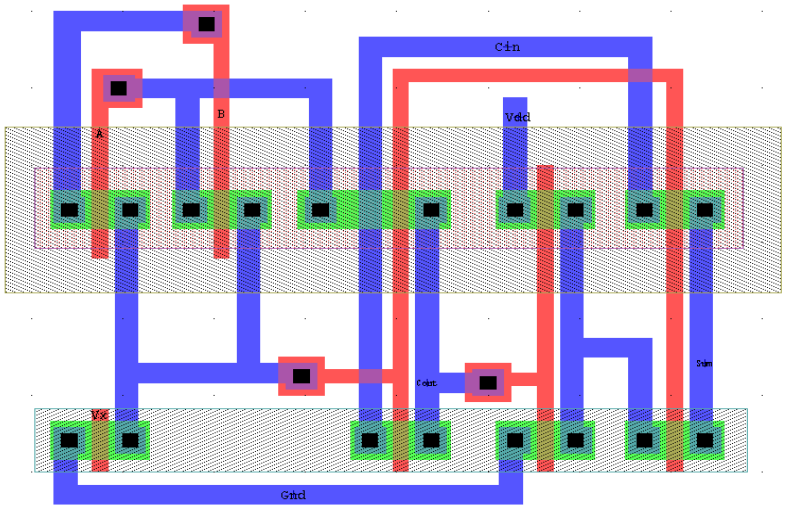

Fig. 8.5 Layout of proposed PTL 9T full adder cell based on carry logic

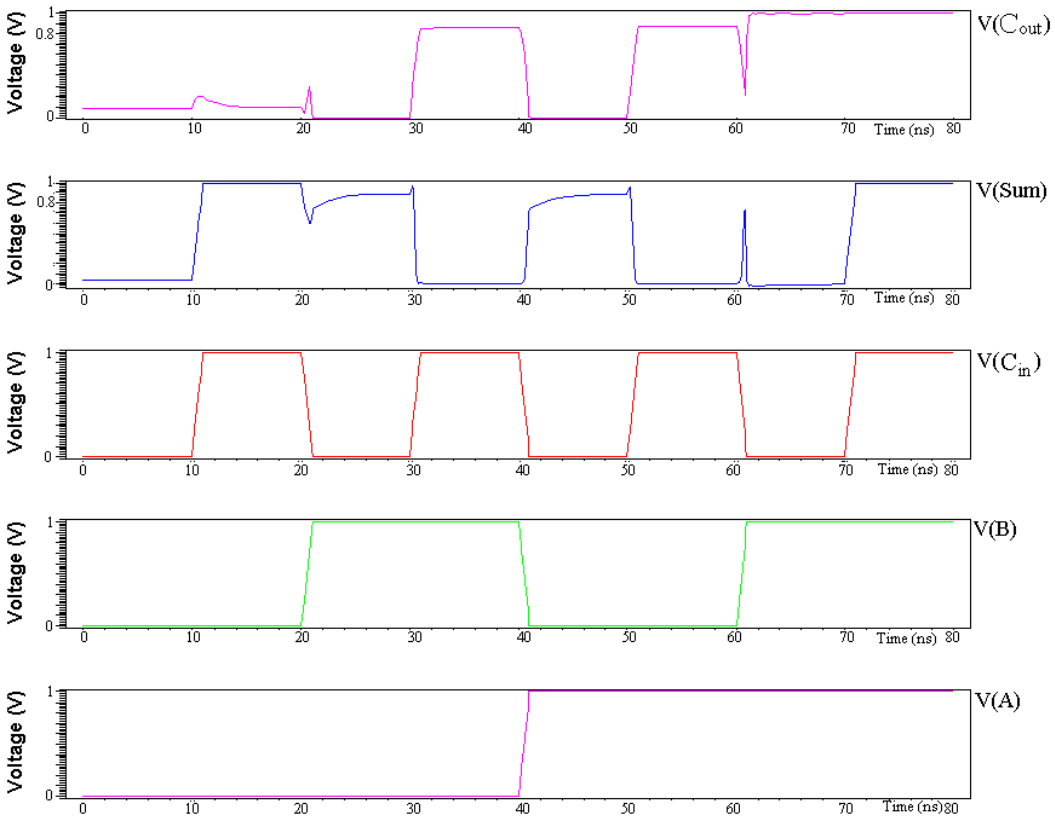

Fig.8.6 In-Out Waveform of proposed PTL 9T full adder cell based on carry logic

8.3 PROPOSED 1-BIT 9T FULL ADDER BASED ON INVERSION LOGIC:

The schematic of 9T full adder cell based on inversion logic is shown in Fig. 8.7 implemented in accordance with Eq. (3.7) and (3.7) and Table V.

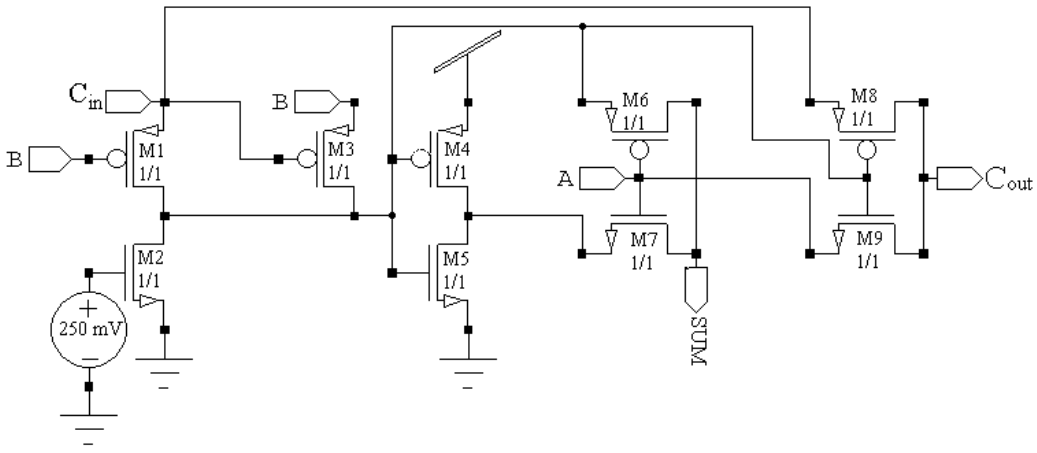

Fig. 8.7 Schematic of proposed PTL 9T full adder cell based on inversion logic

Table XIII. Power Consuming Transitions (Switching Activity) in proposed PTL 9T full adder cell based on inversion logic

Power Consuming Transitions	**Node N1**	**Node N2**	**Node N3**	**Total**
	2	2	3	7

Fig. 8.8 and 8.9 shows the layout designed at 45nm technology and In-Out waveform of this 9T full adder cell.

Table XII and XIII depicts the power consuming transitions at all the internal nodes of Sum module of both the proposed 9T full adders. Node N3 is same for both circuits as it the Sum node. The difference between both the proposed designs is at node N1 and N2 where the main logic has been implemented. The power consuming transition of 9T adder based on carry logic at node N1 and N2 are 50% of the transitions of 9T adder based on inversion logic. Therefore, the total power consuming transitions of 9T adder based on carry logic are less than that in 9T adder based on inversion logic. And, as discussed in Chapter-2 Eq. (2.1), the reduction of power consuming transition i.e.; switching activity will improve the total power consumption of the circuit which is further shown in the simulation results also.

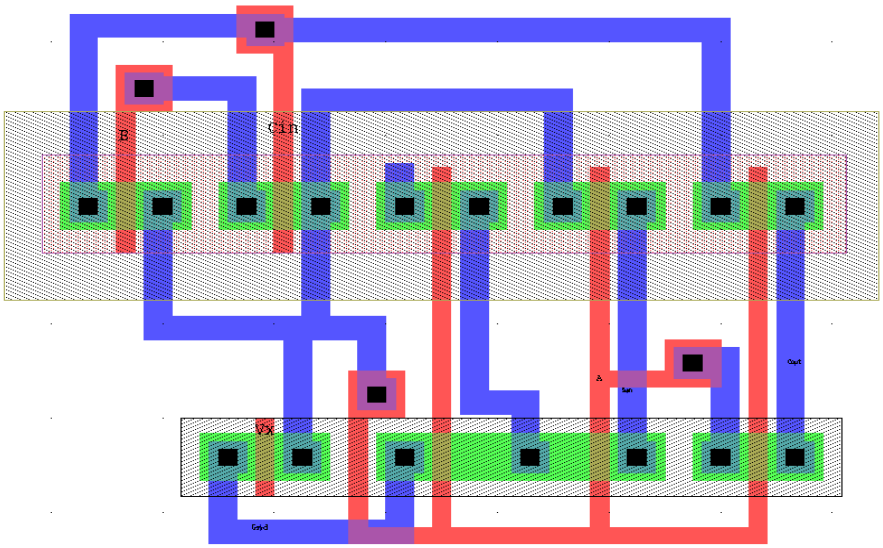

Fig. 8.8 Layout Design of proposed PTL 9T full adder cell based on inversion logic

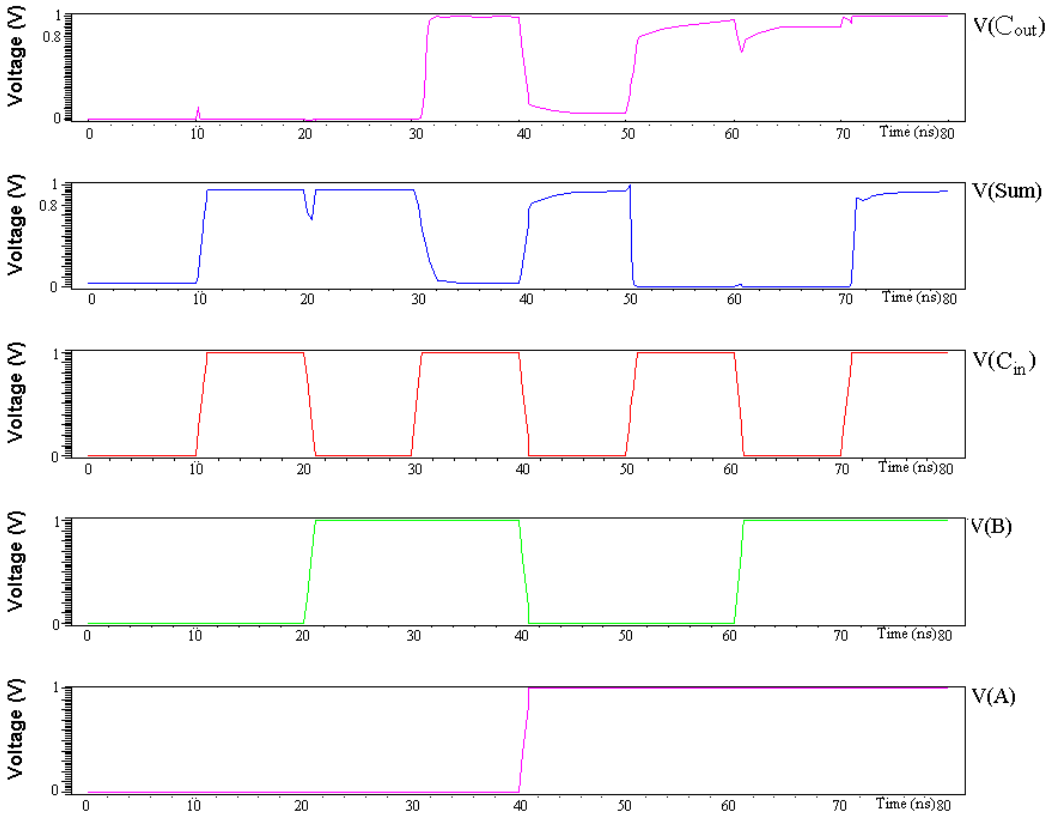

Fig. 8.9 In-Out Waveform of proposed PTL 9T full adder cell based on inversion logic

CHAPTER 9
SIMULATION TOOL AND ENVIRONMENT

The tool used for simulation purpose for the entire research work is Tanner EDA tool version 13.0. The features and functionality of this tool has been described below:

9.1 SIMULATION TOOL AND COMMANDS:

The design cycle for the development of electronic circuits includes an important pre-fabrication verification phase. Because of the expense and time pressures associated with the fabrication step, accurate verification is crucial to efficient design. The role of EDA tool is to help design and verify a circuit's operation by numerically solving the differential equations describing the circuit. These simulation results allow circuit designers to verify and fine-tune designs before submitting them for fabrication.

Tanner EDA tool is a complete circuit design and analysis system that includes:

- **Schematic Editor (S-Edit):** Schematic editor is a powerful design capture and analysis package that can generate netlist directly usable in T-Spice simulations.
- **T-Spice Circuit Simulator:** T-Spice performs fast and accurate simulation of analog and mixed analog/digital circuits. The simulator includes the latest and best device models available, as well as coupled line models and support for user-defined device models via tables or C functions. T-Spice uses an extended version of the SPICE input language that is compatible with all industry standard SPICE simulation programs. All of SPICE's device models are incorporated, as well as resistors, capacitors, inductors, mutual inductors, single and coupled transmission lines, current sources, voltage sources, controlled sources, and a full complement of the latest advanced semiconductor device models from Berkeley and Philips Labs.

- **Waveform Editor (W-Edit):** W-Edit displays T-Spice simulation output waveforms as they are being generated during simulation. Visualizing the complex numerical data resulting from VLSI circuit simulation is critical to testing, understanding, and improving those circuits. W-Edit is a waveform viewer that provides ease of use, power, and speed in a flexible environment designed for graphical data presentation.
- **Layout Editor (L-Edit):** Tanner EDA tool includes L-Edit for layout editing, Interactive DRC for real-time design rule checking during editing, Standard DRC for hierarchical DRC, Standard Extract for netlist extraction, Standard LVS for layout versus schematic, Node Highlighting for highlighting all geometry associated with a node and SPR for standard cell place & route.

T-SPICE [44]

To transform your ideas into designs, you must be able to simulate large circuits quickly and with a high degree of accuracy. That means you need a simulation tool that offers fast run times, integrates with your other design tools, and is compatible with industry standards. Tanner T-Spice Circuit Simulator puts you in control of simulation jobs with an easy-to-use graphical interface and a faster, more intuitive design environment. With key features such as multi-threading support, device state plotting, real-time waveform viewing and analysis, and a command wizard for simpler SPICE syntax creation, T-Spice saves you time and money during the simulation phase of your design flow.

T-Spice enables more accurate simulations by supporting the latest transistor models—including BSIM4 and the Penn State Philips (PSP) model. Given that T-Spice is compatible with a wide range of design solutions and runs on Windows and Linux platforms, it fits easily and cost effectively into your current tool flow.

T-Spice incorporates numerous innovations and improvements not found in other SPICE and SPICE-compatible simulators:

- **Speed:** T-Spice provides highly optimized code for evaluating device models, formulating the systems of linear equations, and solving those systems. In addition to the standard direct model evaluation, T-Spice also provides the option

of table-base transistor model evaluation, in which the results of device model evaluations are stored in tables and reused. Because evaluation of device models can be computationally expensive, this technique can yield dramatic simulation speed increases.

- **Convergence**: T-Spice uses advanced mathematical methods to achieve superior numerical stability. Large circuits and feedback circuits, impossible to analyze with other SPICE products, can be simulated in T-Spice.
- **Accuracy**: T-Spice uses very accurate numerical methods and charge conservation to achieve superior simulation accuracy.
- **Macro modeling:** T-Spice simulates circuits containing "black box" macro devices. A macro device can directly use experimental data as its device model. Macro devices can also represent complex devices, such as logic gates, for which only the overall transfer characteristics, are of interest.
- **Input language extensions:** The T-Spice input language is an enriched version of the standard SPICE language. It contains many enhancements, including parameters, algebraic expressions, and a powerful bit and bus input wave specification syntax.
- **External model interface:** You can develop custom device models using C or C++.
- **Runtime waveform viewing:** The W-Edit waveform viewer displays graphical results during simulation. T-Spice analysis results for voltages, currents, charges, and power can be written to single or multiple files.

T-Spice also supports foundry extensions, including HSPICE foundry extensions to models.

- Supports PSP, BSIM3.3, BSIM4.6, BSIM SOI 4.0, EKV 2.6, MOS 9, 11, 20, 30, 31, 40, PSP, RPI a-Si & Poly-Si TFT, VBIC, Modella, and MEXTRAM models.
- Includes two stress effect models, from the Berkeley BSIM4 model and from TSMC processes, in the BSIM3 model to provide more accuracy in smaller geometry processes.

- Supports gate and body resistance networks in RF modeling.
- Performs non-quasi-static (NQS) modeling.
- Supports comprehensive geometry-based parasitic models for multi-finger devices.
- Models partially depleted, fully depleted, and unified FD-PD SOI devices.
- Models self-heating and RF resistor networks.
- Performs table-based modeling for using measured device data to model a device.
- Includes enhanced diode and temperature equations to improve compatibility with many foundry model libraries.

Work in a faster, easier design environment

T-Spice helps integrate your design flow from schematic capture through simulation and waveform viewing. An easy-to-use point-and-click environment gives you complete control over the simulation process for greater efficiency and productivity.

- Enables easy creation of syntax-correct SPICE through a command wizard.
- Highlights SPICE Syntax through a text editor.
- Provides Fast, Accurate, and Precise options to enable optimal balance of accuracy and performance.
- Enables you to link from syntax errors to the SPICE deck by double clicking.
- Supports Verilog-A for analog behavioral modeling, allowing designers to prove system level designs before doing full device level design.
- Provides ".alter" command for easy what-if simulations with netlist changes.

Perform sophisticated analysis

T-Spice uses superior numerical techniques to achieve convergence for circuits that are often impossible to simulate with other SPICE programs. The types of circuit analysis it performs include:

- DC and AC analysis.
- Transient analysis with Gear or trapezoidal integration.
- Enhanced noise analysis.

- Monte Carlo analysis over unlimited variables and trials with device and lot variations.
- Virtual measurements with functions for timing, error, and statistical analysis including common measurements such as delay, rise time, frequency, period, pulse width, settling time, and slew rate.
- Parameter sweeping using linear, log, discrete value, or external file data sweeps.
- 64-bit engine for increased capacity and higher performance.

With T-Spice, you can

- Optimize designs with variables and multiple constraints by applying a Levenberg-Marquardt non-linear optimizer.
- Perform Safe Operating Area (SOA) checks to create robust designs.
- Use bit and bus logic waveform inputs.

Benefit from flexible licensing

When you purchase a new design tool, licensing options can greatly affect your total cost of ownership. T-Spice is available in node-locked and networked configurations offering you the most flexible licensing possible. With a single solution, T-Spice will work whenever and wherever meeting the design needs of your main workgroup and remote workers. If you offshore design projects, T-Spice does not have geographic restriction on its licenses, thus, lowering your total cost of ownership.

Schematic Editor (S-Edit) [45]

S-Edit is an easy-to-use PC-based design environment for schematic capture. It gives you the power you need to handle your most complex full custom IC design capture. S-Edit is tightly integrated with Tanner EDA's T-Spice simulation, L-Edit layout, and HiPer verification tools. S-Edit helps you meet the demands of today's fast-paced market by optimizing your productivity and speeding your concepts to silicon. Its efficient design capture process integrates easily with third-party tools. S-Edit enables you to explore design choices and provides an easy-to-use view into the consequences of those choices. A faster design cycle gives you more flexibility in moving to an optimal

solution—freeing up more time and resources for process corner validation. The results are less risk downstream, higher yield, and quicker time to market.

Schematic capture for the most complex full custom IC design

- Bus support speeds the creation of mixed signal designs.
- Advanced array support enables easy creation and editing of memory, imaging, or circuits with repetitive blocks.
- Rubberband connectivity editing enables faster design modifications.
- S-Edit displays evaluated parameters in real time over the course of the design process. Parameters with formulas based on other circuit parameters can be displayed or evaluated.
- Auto symbol generation enables you to easily create symbols from schematics, and synchronize any changes.
- All actions are fully scriptable through the TCL/ Tk command language.
- Recordable scripts enable you to automate tasks or expand the tool for application-specific needs.
- Replayable logs permit recovery if there is an unexpected network or hardware failure.
- S-Edit performs net highlighting and keeps the net highlighted as you move through the hierarchy.
- Cross probe from SPICE netlists and LVS to highlighting nets or devices.
- Schematic ERC enables you to check your design for common errors such as undriven nets, unconnected pins and multiple output pins connected together. The design checks are fully configurable, including custom validation scripts.

Tight integration with simulation

- S-Edit is tightly integrated with simulation. You can drive the simulator from within the schematic capture environment, viewing operating point results directly on the schematic and performing waveform cross-probing to view node voltages and device terminal currents or charges.

- S-Edit creates an efficient flow for the iterative loop of design, simulation, analysis, and tweaking of circuit parameters. The IC designer can focus on the design and not on data processing—thereby speeding up the design process.

Easy interoperability with third party tools and legacy data

- S-Edit imports schematics via EDIF from third party tools, including Cadence®, Mentor, Laker and ViewDraw with automatic conversion of schematics and properties for seamless integration of legacy data.
- Netlists can be exported in flexible, user-configurable formats, including SPICE and CDL variants, EDIF, structural Verilog, and structural VHDL.
- Library support in S-Edit maximizes the reuse of IP developed in previous projects, or imported from third- party vendors.

Powerful and easy-to-use interface

- S-Edit brings to front-end design capture the ease-of-use and design productivity for which Tanner Tools are known.
- A fully user-programmable design environment allows you to remap hotkeys, create new toolbars, and customize the view to your preference—all in a streamlined GUI.
- The complete user interface is available in multiple languages. S-Edit currently supports English, Japanese, Simplified and Traditional Chinese.
- S-Edit provides Unicode support. All user data can be entered in international character sets.

Cost-effective

- S-Edit provides an ideal performance to-cost ratio, allowing you to maximize the number of designers on a project.
- Since S-Edit is Windows-based, designers can work on cost-effective workstations or laptops. This means you can take your work with you anywhere—even home—and continue working to meet time-to-market pressures.
- Available in two configurations—full schematic editor, and schematic viewer.

Easy to manage

- Human-readable technology files and design databases are revision-control system-compatible.
- CAD managers can control distribution and access rights to the technology or design. The format allows revision control systems to manage revisions over the course of the design process.

Benefit from flexible licensing

When you purchase a new design tool, licensing options can greatly affect your total cost of ownership. S-Edit is available in node-locked and networked configurations offering you the most flexible licensing possible. With a single solution, S-Edit will work whenever and wherever meeting the design needs of your main workgroup and remote workers. If you offshore design projects, S-Edit does not have geographic restriction on its licenses, thus, lowering your total cost of ownership.

Layout Editor (L-Edit) [46]

In today's analog design world, speed is more important than ever. To compete in a high-efficiency, high-productivity marketplace, you need a toolset that has proven its ability to accelerate the design cycles of commercially successful projects.

Tanner EDA's L-Edit meets your needs by combining the fastest rendering available with powerful features that exceed the needs of the most demanding user. This leading analog/ mixed signal IC design tool for the PC platform enables you to get started with minimal training. You can draw and edit quickly, with fewer keystrokes and mouse clicks than other layout tools. Using powerful features such as interactive DRC, object snapping, and alignment, you can work more efficiently to save time and money.

L-Edit increases your productivity by enabling you to import Cadence Virtuoso technology and display files for quick setup. Save time by using foundry-provided files directly, allowing you to avoid having to set up technology information manually. Once you've begun using L-Edit, the CAD support burden for your physical design tools will be reduced, enabling you to focus on other mission critical tasks.

Create layout with precision

L-Edit gives you greater precision by enabling you to perform complex Boolean and derived layer operations with polygons of arbitrary shape and curvature. Perform AND, OR, XOR, Subtract, Grow, and Shrink on groups of objects. You can display coordinate and distance values in any technology unit, and automatically add guard rings around any shape. Further increase your productivity by mapping multiple layout functions to a single keystroke.

- Perform complete hierarchical physical layout with all-angle and curved polygons on an unlimited number of layers.
- Use orthogonal, 45°, all-angle, and curved drawing modes.
- View your design with the fastest rendering on the market.
- Use a command line interface for run-time automation.

L-Edit schematic driven layout (SDL) provides capabilities that enable you to:

- Read in a netlist and automatically generate parameterized cells and instance them into your design.
- Display flylines allowing you to place your blocks to minimize routing congestion.
- Mark existing geometry as part of a specific net allowing selection and rip-up of geometry by net.
- Perform engineering change orders (ECO) and highlight differences in the netlists.
- Use netlist files in T-Spice, HSPICE, Pspice, structural Verilog, or CDL formats.

L-Edit also supports parameterized cells called T-Cells With T-Cells, you can create versatile source cells that consist of user-defined input parameters and layout-generating code. T-Cells extend traditional geometry cells to include the flexibility and automation of L-Edit's user-programmable interface (UPI).

L-Comp, a set of high-level composition functions, provides a simple toolkit for creating T-Cell code.

Use L-Comp to efficiently create, place, and align cell instances in a design.

Navigate efficiently

L-Edit provides a built-in Design Navigator that enables you to:

- Efficiently traverse design hierarchy with top-down and bottom-up hierarchical view, non-instanced cells view, or view cells sorted by their modified date.
- Drag and drop cells into layout from library files, other design files, or the current design database.
- View layout details down to any level of the hierarchy.
- Lock and unlock cells to protect the design from any changes.
- Easily replace instances of one cell with another cell, at the current level or throughout the design.
- Maximize IP reuse or partition your design for multiple designers with L-Edit's multiple library support (XrefCells).

Gain complete control over editing

L-Edit gives you the flexibility and control you need to master the editing process. You can dramatically streamline the process by editing the properties of multiple objects simultaneously. With L-Edit you can instantly push down the hierarchy to any object, making it easy to edit edges, corners, and arcs. You can quickly stretch or shrink multiple edges to make room for more layouts.

- Change the current drawing layer directly from the layout using the virtual layer palette.
- Perform unlimited undo and redo operations.
- Perform all-angle rotate, flip, merge, nibble, and slice operations.
- Speed drawing and editing by snapping the cursor to object vertices, edges, midpoints, center points, and instances.
- Perform one-click horizontal or vertical object alignment, equally space objects, or tile objects horizontally, vertically, or in a 2D array.
- Specify a reference point for editing operations such as object rotation, flip, move, or instance placement using the base point feature.

Work in a versatile environment

Save time and money with L-Edit's ease-of-use benefits:

- Delivers powerful features from an affordable, customizable, easy-to manage tool.
- Offers a short learning curve.
- Enables you to import and export GDS, DXF, and CIF file formats.
- Provides multi-language menus (English, Japanese, Simplified and Traditional Chinese, German, Italian, and Russian).
- Customize and filter the Layer Palette to show only layers used in the file, current cell, or cell and its hierarchy allowing you to finish your layout faster.
- Enables you to easily cut and paste layout into your documentation flow.

Generate layout for parameterized devices

L-Edit's DevGen feature provides layout generators for most common devices. It is easy to configure for any process to help ensure DRC correct layout. Devices include Capacitor Generator, Inductor Generator, Resistor Generator, and MOSFET Generator. For specialized devices, use L-Edit's automatic layout to T-Cell generator to quickly complete your T-Cell library.

Create UPI macros

L-Edit's powerful user programmable interface (UPI) allows you to create macros that automate layout manipulations, geometric synthesis, batch verification, and advanced analysis. You can further increase your productivity by mapping multiple layout functions to a single keystroke. UPI macros are written in C/C++ language and can be executed with L-Edit's built in interpreter or compiled as a DLL.

Speed layout with standard cell place and route

L-Edit standard cell place and route (SPR) performs place and route, pad frame generation, and pad routing. To minimize total area, a built in placement and routing optimizer automatically reduces net length and the number of vias. L-Edit SPR includes:

- Three-layer channel routing.
- Cell clustering options.

- Specification of critical nets.
- Global signal routing for clock nets.
- Back annotation with SDF.
- EDIF input.

Further streamline verification with Interactive DRC

L-Edit Interactive DRC displays violations in real time while you edit your layout, helping you create compact, error-free layouts the first time. Interactive DRC simultaneously checks for violations between objects in the same cell and down through the cell hierarchy.

Correct layout as you goes

L-Edit Node Highlighting offers node highlighting for connectivity visualization so you can quickly find and fix LVS problems.

- Point to an object in the layout, regardless of hierarchy, and display all the geometry connected to it based on a set of connectivity rules.
- Highlight discrepancies between the schematic netlist and the extracted netlist in the layout.
- View multiple nodes in different colors.
- Track down shorts and opens.
- Improve design productivity significantly during LVS.

Verify complex designs with DRC

L-Edit DRC is a high performance, all angle, hierarchical design rule checker. Setting up rules is configurable for any technology, with a graphical interface for easy setup. Hierarchical error output displays your errors at the level of the hierarchy where they occur, and the error navigator opens the cell and zooms automatically to the location of the error. Your productivity improves via speedy runtimes and quick location of errors.

L-Edit DRC features include:

- Support for an unlimited number of width, spacing, surround, enclose, extension, overlap, not exist, and density rules.

- High performance all-angle Boolean and Select layer generation.
- Flag off grid vertices, self-intersecting polygons, all-angle edges, and polygons with more than a specific number of vertices.
- Full chip and local region DRC.
- Import of Calibre® DRC results for browsing in L-Edit.
- Seamless integration into the layout environment using the DRC Error Navigator.
- See the DRC status (pass/fail/needed) of each cell.

Integrated Verification Error Navigator allows you to:

- Navigate instantly down the hierarchy to locate errors.
- View errors grouped by rule or by cell.
- See rule distance and actual violation distance.
- View errors in the top cell or in the cell where the error occurred.
- Mark or remove errors that have been fixed.

Extract SPICE netlists

L-Edit Extract generates a SPICE netlist from L-Edit layout for LVS comparison or for post-layout simulation with T-Spice. It extracts active and passive devices and user-definable sub circuits, with support for orthogonal, 45°, all-angle, and curved layout. To aid in verification against the schematic, L-Edit enables you to zoom or click in the layout to display a specific node or element. For increased accuracy, you can extract parasitic node capacitances, including fringing effects. You can extract the most common device parameters, including:

- MOSFET width, length, source/drain area, and perimeter.
- Areas of diodes, BJTs, MESFETs, and JFETs.
- Capacitance and resistance.
- Sub circuit extraction for functional blocks with user parameters.

Compare layout to schematic

L-Edit layout versus schematic (LVS) accurately and efficiently compares two SPICE netlists to determine whether they contain equivalent circuit descriptions. LVS can use

topological information, parametric values, and geometric values to compare netlists according to your specifications. The program quickly traces element and node mismatches back to their origins, pinpointing irresolvable nodes and devices using fragmented class reporting.

LVS offers a full range of pre-processing options to optimize netlists for comparison, including:

- Merging of parallel or series devices, where options can be set independently for different device types and for specific device models.
- Elimination of shorted and disconnected (open) devices.
- Elimination of parasitic resistors and capacitors that exceed user-supplied min/max thresholds.
- Removal of user-specified device models so that the nodes spanned by their terminals can be shorted or opened.
- Omission of user-selected device parameters from the compared netlists.
- Checking for soft-connections.
- Supporting asymmetrical MOSFETs with user-defined pin swapping.
- Matching device parameter values, which are crucial for analog design.
- Specifying pre- and post-iteration matches to speed the comparison process.

LVS reads netlist files in T-Spice, HSPICE, Pspice, or CDL formats, with support for all device types and key parameters. Input netlists do not need matching formats or hierarchy to be compared. LVS supports length and width parameters, with accompanying model definitions, in capacitor I and resistor I device statements. Zip through LVS with cross-probing from SPICE and LVS results to layout or schematic and with enhanced navigation of SPICE files.

Benefit from flexible licensing

When you purchase a new design tool, licensing options can greatly affect your total cost of ownership. L-Edit is available in node-locked and networked configurations offering you the most flexible licensing possible. With a single solution, L-Edit will work whenever and wherever meeting the design needs of your main workgroup and remote

workers. If you offshore design projects, L-Edit does not have geographic restriction on its licenses, thus, lowering your total cost of ownership.

Waveform Editor (W-Edit): Waveform Viewing & Analysis

The W-Edit waveform analysis tool is a comprehensive viewer for displaying, comparing, and analyzing simulation results. W-Edit provides an intuitive multiple-window, multiple-chart interface for easy viewing of waveforms and data in highly configurable formats.

- W-Edit is dynamically linked to T-Spice and S-Edit with a run-time update feature that displays simulation results as they are being generated and allows waveform cross-probing directly in the schematic editor for faster design cycles.
- Focus on and optimize your design with W-Edit's advanced features such as automatically calculating and displaying FFT results in a variety of formats, including dB or linear magnitude, wrapped or unwrapped phase, and real or imaginary parts.
- W-Edit allows creation of new traces based on mathematical expressions of other traces for advanced analysis and easy comparison with measured data.

The advantages of W-Edit include:

- Tight integration with T-Spice, Tanner EDA's circuit-level simulator. W-Edit can chart data generated by T-Spice directly, without modification of the output data files. W-Edit charts data dynamically as it is produced during the simulation.
- Charts are automatically configured for the type of data being presented.
- A data set is treated by W-Edit as a unit called a *trace*. Multiple traces from different output files can be viewed simultaneously, in single or multiple windows. You can copy and move traces between charts and windows. You can perform trace arithmetic or spectral analysis on existing traces to create new ones.
- You can pan back and forth and zoom in and out of chart views, including specifying the exact *x-y* coordinate range W-Edit displays. You can measure positions and distances between points easily and precisely with the mouse.
- You can customize properties of axes, traces, grids, charts, text, and colors.

- Numerical data is input to W-Edit in the form of plain or binary text files. Header and comment information supplied by T-Spice is used for automatic chart configuration. Runtime update of results is made possible by linking W-Edit to a running simulation in T-Spice.
- W-Edit saves information on chart, trace, axis, and environment settings in files with the **.wdb** (W-Edit Database) extension. You can load a **.wdb** file in conjunction with a data file to automatically apply saved formats and environment settings. The **.wdb** file does not contain any trace data.

Simulation Commands Used for Coding

1) .dc

Performs DC transfer analysis to study the voltage or current at one set of points in a circuit as a function of the voltage or current at another set of points. Can also be used for linear or logarithmic sweeps of DC voltage or current; for sweeps of parameters other than voltage and current source values; and for Monte Carlo analysis or optimization.

- Transfer analysis is done by *sweeping* the source variables over specified ranges, and recording the output.
- Up to three parameters can be specified per **.dc** command
- When two or more sources are specified, the last-named source "controls" the sweeping process.
- The specified current or voltage sources must exist—that is, be defined by **i** or **v** device statements elsewhere in the input file
- DC transfer analysis results can be reported with the **".print" dc, ".probe" dc**, and **".macro /.eom" dc** commands.

Syntax

.dc *swinfo* [[**sweep**] *swinfo* [[**sweep**] *swinfo*]] Refer to **".step"** (page 138) for a syntax description of ***swinfo***.

Examples

.dc isrc 0 1e-6 0.1e-6

Current source **isrc** is swept from 0 to 1 microampere in 0.1-microampere steps.

.dc vin 0 5 0.05 VCC 4 6 0.5

Names two voltage sources:

vin, to be swept from 0 to 5 volts in 0.05-volt steps, and **VCC**, to be swept from 4 to 6 volts in 0.5-volt steps. The second source "controls" the sweep: **VCC** is initially set to 4 volts, while **vin** is swept over its specified range. Then **VCC** is incremented to 4.5 volts, and **vin** is again swept over its range. This process is repeated until **VCC** reaches the upper limit of its specified range.

2) .end

Signifies the end of the circuit description.

- Any text in the input file after the **.end** command is ignored.
- The **.end** command is optional in T-Spice but is included for compatibility with generic SPICE.

Syntax

.end [*comment*]

3) .measure

Used to compute and print electrical specifications of a circuit, such as delay between signals, rise and fall times, and minimum and maximum values of a signal. Also used for optimization in conjunction with the following commands:

- **".ac"**
- **".dc"**
- **".connect"**
- **".step"**
- **".tran"**

For parameter sweeps, T-Spice generates a separate output section plotting measurement results versus swept parameter values. A data set can be used with the error measurement syntax of **.measure** to compute differences between simulation output curves and corresponding external data.

Trigger/Target Measurements

The trigger/target format of the **.measure** command is used to make independent variable (time, frequency, or swept parameter) difference measurements. The ***trigger*** and ***target*** specifications determine the beginning and end, respectively, of the measurement. The value of the measurement is the difference in the independent variable value between the trigger and the target. Common examples of trigger/target measurements include delay time, rise time, fall time, and bandwidth measurements.

Syntax

trig *outvar* **val=***val* [**td=***td*] {**cross=***cross* | **rise=***rise* | **fall=***fall*}

+ **targ** *outvar* **val=***val* [**td=***td*] {**cross=***cross* | **rise=***rise* | **fall=***fall*}

or

trig at=*at_value* **targ** *outvar* **val=***val* [**td=***td*] {**cross=***cross* | **rise=***rise* | **fall=***fall*}

at_value	Specifies an explicit independent variable value for the event.
Outvar	Specifies an output signal on which the trigger or target measurement is performed. This can be any output plot item that is legal on a **.print** command for the appropriate analysis type. For **.measure ac** commands, **.print noise** plot items are allowed. ***Outvar*** may be an output expression enclosed in single quotes.
Val	Specifies the value of ***outvar*** at which the trigger or target counter for ***cross***, ***rise***, or ***fall*** is incremented.
Td	Specifies a time delay before the measurement is enabled and crossings, rises, and falls are counted. Default: 0.
Cross	Indicates which occurrence of the trigger or target crossing is to be used for the measurement. A crossing occurs when the trigger or target ***outvar*** takes on the value *val*. The special syntax **cross=last** indicates that the last crossing is to be used.
Rise	Indicates which occurrence of the trigger or target rise crossing is to be used for the measurement. A rise crossing occurs when the trigger or target ***outvar*** takes on the value ***val*** while increasing. The

special syntax **rise=last** indicates that the last rise crossing is to be used.

Fall Indicates which occurrence of the trigger or target fall crossing is to be used for the measurement. A fall crossing occurs when the trigger or target ***outvar*** takes on the value ***val*** while decreasing. The special syntax **fall=last** indicates that the last fall crossing is to be used.

Note: For a particular trigger or target, only one of ***cross***, ***rise***, or ***fall*** may be specified.

Trigger/target measurement output reports the independent variable value difference between the trigger and the target, as well as the independent variable values at the trigger and target. The measurement result may be negative if the target independent variable value is less than the trigger independent variable value.

Examples

.measure tran delaytime trig v(1) val=2.5 fall=3 targ v(2) val=2.5 rise=3

measures the time delay from the third falling edge of signal **v(1)** to the third rising edge of signal **v(2)**. The measurement begins when **v(1)** falls through 2.5V for the third time, and ends when **v(2)** rises through 2.5V for the third time.

.measure tran risetime trig v(1) val=0.5 rise=1 targ v(1) val=4.5 rise=1

measures the first rise time of the voltage at node 1.

4) .model

Specifies device model parameters to be used by one or more devices.

Syntax

.model *modelname type* [**level=***L*][*parameter*=*value* [*parameter*=*value* [...]]]

modelname Model name.

type Device type (see below).

L Model level, required for device models with multiple levels.

Parameter The ***parameter*** list is predefined for each standard device model.

Type is one of the following:

c	Capacitor
cpl	Coupled transmission line.
Csw	Current-controlled switch element.
D	P-N diode.
Npn	NPN-type BJT.
Pnp	PNP-type BJT.
Njf	N-type JFET.
Pjf	P-type JFET.
Nmf	N-type MESFET.
Opt	Controls the optimization algorithm.
Pmf	P-type MESFET.
Nmos	N-type MOSFET.
Pmos	P-type MOSFET.
R	Two-terminal resistor (or three-terminal resistor if a capacitance is specified).
Sw	Voltage-controlled switch element.

Example

.model nmos nmos + Level=2 Ld=0.0u Tox=225.00E-10 + Nsub=1.066EV+16 Vto=0.622490 Kp=6.326640E-05 + Gamma=.639243 Phi=0.31 Uo=1215.74 + Uexp=4.612355E-2 Ucrit=1746677 Delta=0.0 + Vmax=177269 Xj=.9u Lambda=0.0 + Nfs=4.55168E+12 Neff=4.68830 Nss=3.00E+10 + Tpg=1.000 Rsh=60 Cgso=2.89E-10 + Cgdo=2.89E-10 Cj=3.27E-04 Mj=1.067 + Cjsw=1.74E-10 Mjsw=0.195

Specifies the parameters for an *n*-type MOSFET model called **nmos**.

5) .noise

Computes the effect of circuit noise on output voltage in conjunction with AC analysis.

- If the **".ac**" command is missing from the input file, the **.noise** command is ignored.

- Noise analysis is performed at the same frequencies as specified by the **".ac**" command.
- Noise models take the form of frequency-dependent mean-square currents (since the underlying phenomena are "random") generated by adding a current source to the circuit for each modeled noise source.
- Noise sources at different points in the circuit are uncorrelated.
- Noise models are available for resistors and semiconductor devices (diodes, BJTs, JFETs, MESFETs, and MOSFETs). Semiconductor device models may contain noise model parameters which affect the size of noise sources.
- External model devices may also contain noise sources.
- Noise analysis results can be reported with the **".print**" **noise** command.

Syntax

.noise v(*node1* [[**,**] *node2*]**)** *source interval*

node1	Output node.
Node2	Reference node. (Default: ground.)
source	Input voltage or current source, at which noise can be considered to be concentrated for the purposes of estimating the equivalent noise spectral density.
Interval	Report interval. A noise report will be printed to the simulation log which lists every device and the noise contribution and noise components. This report will be printed for the first frequency and each *interval* frequency. (Default: 0)

6) .power

Computes power dissipation in conjunction with transient analysis.

- If the **".tran**" command is missing from the input file, the **.power** command is ignored.

- The average power consumption, the instantaneous maximum power, and the instantaneous minimum power (in watts) and the times of maximum and minimum power consumption (in seconds) are reported at the end of the transient simulation.
- The instantaneous power $P(t)$ dissipated by a voltage source at time t is the current through the source multiplied by the voltage drop across the source. The average power P for a time interval ($t1$,$t2$) is computed by using the trapezoidal rule approximation to evaluate the integral
- Multiple **.power** commands can be used in a single simulation.
- Power results can also be reported with the **".print" tran** command.

Syntax

.power *source* [*A* [*Z*]]

source	Voltage source whose power consumption is to be computed.
A	Time at which power recording begins. (Unit: seconds. Default: simulation start time.)
Z	Time at which power recording ends. (Unit: seconds. Default: simulation end time.)

Example

.power vtest 3e-7

Computes the power dissipated by voltage source **vtest** between the given time (0.3 microseconds) and the end of the simulation. It might produce the following sample output:

Power Results
vtest from time 0 to 3e-007
Average power consumed -> 1.249948e-002 watts
Max power 2.490143e-002 at time 9.76e-006
Min power 2.282279e-030 at time 1e-005

7) .print

Reports simulation results.

- If an output filename is not specified, the output file specified in the Start Simulation dialog is used. (If that file cannot be opened, the results are written to the Simulation Window.)
- Multiple **.print** commands can be used to direct different types of output to separate files.
- Transient analysis results are printed in columns, with time values in the first column.
- AC and noise analysis results are printed in columns, with frequency values in the first column.
- DC transfer analysis results are printed in columns, with sweep values from the first source listed on the **".dc**" command in the first column.
- DC operating point analysis results are printed line by line, each argument to a line.
- Expressions can be printed by themselves, with or without reference to physical quantities at specific nodes.
- Nodes and devices within subcircuits can be accessed with hierarchical notation in the form **x*instance*.x*instance*.*node***.
- **.print** commands inside subcircuit definition blocks are replicated for each instance.
- If no arguments are given, all node voltages and source currents are printed. If neither mode nor arguments are given, **.print** applies to all analysis types. If arguments are given, a mode must also be specified.
- Wildcards may be entered as part of the **.print** command node names, devices names, terminal names, or terminal numbers. Wildcards are expanded to match any available elements which match the name specification.
- *Device State* print statements are a means of obtaining very detailed information about devices and device internal states. Data such as the current, charge,

capacitance, and voltage values can be listed, as well as certain model evaluation variable (threshold voltage, beta, etc.).

Syntax

.print [*mode*] [**"***filename***"**] [*arguments*]

mode	Analysis mode (see below).
Filename	Output filename. Must be enclosed by double quotes.
Arguments	Information to be printed (see below). ***Arguments*** may include valid expressions involving other arguments or global parameters. ***Arguments*** take one or more of a number of values, depending on ***mode***. When no arguments are given, all node voltages and source currents are printed.

Mode is one of the following:

tran	Print results from transient analysis.
Dc	Print results from DC transfer analysis and DC operating point analysis.
Ac	Print results from AC analysis.
Noise	Print results from noise analysis.

Examples

.print tran in out i1(r2) id(M2)

Prints transient analysis results: the voltages at nodes **in** and **out** and the currents into terminal **1** of device **r2** and the **drain** terminal of device **M2**.

.print v(n*)

Prints the voltages for all nodes whose name begins with the letter 'n'.

8) .subckt

Defines a hierarchical set of devices and nodes to be used repeatedly in a higher-level circuit.

- Subcircuits are replicated by means of the instance (**x**) statement.
- When invocations of the following commands appear within subcircuit definitions and refer to nodes inside the subcircuit, the commands are executed for each instance of the node: **".acmodel**" , **".hdl**", **".macro /.eom**", **".nodeset**", **".noise**", **".print**", and **".probe**", and **".tf**".
- Node and device names have local scope in subcircuits unless global node names (defined elsewhere with the **".global**" command are used.
- Subcircuit blocks cannot be nested: after one **.subckt** command, the **".ends**" command must appear before another **.subckt** command can be used.

Syntax

.subckt *name node1* [*node2* ...] [*parameter*=*X* ...]

subcircuit

.ends

name	Name of subcircuit.
Node1 node2	Nodes used as "external" connections to the subcircuit.
Parameter	Parameter(s), with default value(s) assigned. ***X*** can be a number or an expression. Subcircuit parameters have local scope. Parameters can be written in any order in both definition and instances. Parameter values specified in the definition are used as defaults when not specified in instances. Within the definition, parameter values are referenced (in place of numbers) by enclosing their names in single quotes. Alternatively, the **".param**" command may be used within the definition, with the same results. Parameters created outside the definition with the **".param**" command may be used inside the definition, but an assignment made with the **.subckt**

command to an externally defined parameter always overrides its external value.

Subcircuit Subcircuit definition (may be multiple lines).

Example

.subckt inv in out Vdd length=1.25u nwidth=2u pwidth=3u
mt1 out in GND GND nmos l='length' w='nwidth'
mt2 out in Vdd Vdd pmos l='length' w='pwidth'
c2 out GND 800f .ends inv

This subcircuit could be instanced as follows:
xinv1 a1 a2 Vdd inv nwidth=4u pwidth=6u

9) .temp

Specifies the temperatures at which the circuit is to be simulated.

- Changing the temperature affects the behavior of diode, resistor, BJT, JFET, MESFET, and MOSFET models. It may also affect the behavior of user-defined external models.
- The **.temp** command has no effect on external tables, which should be regenerated to reflect the new temperature.

Syntax

.temp *temperature* [*temperature* [*temperature* [...]]]

temperature Temperature. (Unit: °C. Default: 25.)

Using **.TEMP** and **.STEP** displays voltage vs. voltage plots with different temperatures displaying as different traces.

Example

.DC vin 0 5 0.1

.TEMP 25 30 35 40 50

To plot voltage vs. temperature, use the following so that the first sweep variable in the DC analysis will become the X axis:

.DC temp 25 40 5 VIN 0 5 0.1

10) .tf

Computes and reports the value of the small-signal DC transfer function between the specified output and input, and the corresponding input and output resistances, at the DC operating point.

- The **.tf** command automatically performs (but does not report the results from) a DC operating point calculation.
- Results are reported under the heading **SMALL-SIGNAL TRANSFER FUNCTION** (to the specified output file or in the Simulation Window).
- The transfer function value corresponds to a voltage (V/V) or current (I/I) *gain*, a *transconductance* (I/V), or a *transresistance* (V/I).

Syntax

.tf *arguments source*

arguments	Any arguments appropriate for the **".print" dc** command.
Source	Voltage or current input source.

Example

.tf i(mb1,out1) ii1

Computes transfer function results between node **out1** of device **mb1** and current source **ii1**.

11) .tran

Performs large-signal time-domain (transient) analysis of the circuit to determine its response to initial conditions and time-dependent stimuli.

- The time step is adaptively varied throughout the simulation to ensure accuracy.
- Results for nodes selected by the **.print tran**, **.probe tran**, and **.measure tran** commands will be output for every time step, unless otherwise specified by the **.options prtdel** command. For additional information on these commands, see **".print"**, **".probe"**, **".macro /.eom"** and **".options"**.

Syntax

.tran[*/mode*] *S L* [**start=***A*] [**UIC**] [**sweep** *sweep*]

mode	Analysis mode (see below). This parameter must immediately follow the keyword **.tran** and be preceded by a slash (/).
S	Maximum time step allowed. By default, the time step is dynamically adapted to resolve the output values. (Unit: seconds.)
L	Total simulation time. (Unit: seconds.)
A	Output start time. Execution of the **.print tran** command will not start until this time. (Unit: seconds. Default: 0.)
UIC	Instructs T-Spice to skip the DC operating point analysis for determining the **time=0** circuit state. Instead, only the initial conditions specified using **.ic** commands are used to set the **time=0** voltages. Voltages which cannot be determined using **.ic** commands are set to zero.
Sweep	For a description of the syntax for this field, see **".step"**.

mode takes one of the following values:

op	Performs a DC operating point calculation before simulation to determine initial steady-state node voltages. The commands **".nodeset"** or **".hdl"** can be used to impose initial conditions.

Powerup	Performs a "powerup" simulation. All nodes are at the same potential at time zero, and the voltage sources are ramped gradually to their final values.
Preview	Steps through the input signals without simulating the circuit. Input waveforms will be reported as specified by the **".print" tran** command.

If ***mode*** is not specified, T-Spice first performs a DC operating point analysis, without printing the DC operating point analysis results.

Sweep indicates the beginning of the next nested sweep variable specification. The **sweep** keyword can be omitted if the previous sweep is not of the **list** or **poi** type or if one of the keywords **lin**, **dec**, **oct**, **list**, **poi**, **temp**, **param**, **source**, or **modparam** follows immediately.

Using the ***sweep*** option with **.tran** or **".ac**" causes that analysis to be performed for all parameter values of the sweep. It is equivalent to **".step**", except that it applies only to one analysis command, while **.step** applies to all analysis commands in the input file. If ***sweep*** is specified on an analysis command and **.step** is present, the ***sweep*** sweep is nested inside the **.step** sweep. The ***sweep*** parameter may be used to specify a parametric sweep, Monte Carlo analysis, or optimization.

Examples

.tran 0.5n 100n

Defines a transient simulation lasting 100 nanoseconds, using time steps of at most 0.5 nanosecond. By default, a DC operating point calculation will first be performed to define a starting condition.

.tran/preview 4n 4000n

The input waveforms are reported for 4000 nanoseconds; the rest of the circuit is ignored.

.tran 1ns 100ns Specifies a maximum time step of 1 nanosecond and a total simulation time of 100 nanoseconds.

.tran/powerup 1ns 100 ns

Specifies a powerup simulation with no operating point computation.

.tran 1ns 100 ns start=50ns

Produces output starting at time 50 nanoseconds.

.tran 1n 100n sweep temp list 0 27 100 150 -50

Performs five transient analysis runs, one for each temperature listed. The keyword **temp** specifies the sweep *variable*, as defined in **".step**".

.tran 1n 400n sweep temp -50 150 50

This performs five transient analyses at temperatures -50, 0, 50, 100, and 150 degrees Celsius.

.tran 1n 200n sweep temp list 0 25 75 150

This example performs four transient analysis runs at temperatures 0, 25, 75, and 150 degrees Celsius.

12) MOSFET (m)

A transistor with four terminals: drain, gate, source, and bulk. (MOSFET stands for *metal oxide semiconductor field effect transistor*.)

Syntax

m*name drain gate source bulk model* [**l=***L*] [**w=***W*] [**ad=***Ad*] [**pd=***Pd*] [**as=***As*] [**ps=***Ps*]

name	MOSFET name.
drain	Drain terminal.
Gate	Gate terminal.
Source	Source terminal.
Bulk	Bulk terminal.
Model	MOSFET model name. The model is declared elsewhere in the input file in the form: **.model** *name* **nmos | pmos level=1 | 2 | 3 | 4 | 5 | 9|13|20|28|30|31|40|47 | 49 |52|100…** [*parameters*]
L	Channel length. (*Unit:* meters. *Default:* set by the **.options defl** command.)
W	Channel width. (*Unit:* meters. *Default:* set by the **.options defw** command.)

Ad	Drain area. (*Unit:* square meters)
Pd	Drain perimeter. (*Unit:* meters)
As	Source area. (*Unit:* square meters.)
Ps	Source perimeter. (*Unit:* meters)

Example

m12 n1 n2 GND GND ndep l=10u w=5u ad=100p as=100p pd=40u ps=40u

13) Voltage Source (v)

A two-terminal ideal voltage supply.

Exponential, pulse, piecewise linear, frequency-modulated, sinusoidal, and customizable (vectorized) waveforms are available. Voltage sources whose waveform is described using an expression can be created using the **e**-element with an expression and the **time()** function.

Syntax

v*name node1 node2* [[**DC**] *V*] [**AC** *M* [*P*]] [*waveform*]

name	Voltage source name.
node1	Positive terminal
node2	Negative terminal.
V	DC level between ***node1*** and ***node2***. (*Unit:* volts. *Default:* 0.)
waveform	Waveform identifier and parameters
M	AC magnitude. (*Unit:* volts.)
P	AC phase. (*Unit:* degrees. *Default:* 0.)

DC, AC, and transient values can be specified independently and in any order.

Waveform is one of the following:

Exponential Waveform

exp (*Vi Vp* [*Dr* [*Tr* [*Df* [*Tf*]]]])

Vi	Initial voltage. (*Unit:* volts.)
Vp	Peak voltage. (*Unit:* volts.)
Dr	Rise time delay. (*Unit:* seconds. *Default:* 0.)
Tr	Rise time constant. (*Unit:* seconds. *Default:* 0.)
Df	Fall time delay. (*Unit:* seconds. *Default:* 0.)
Tf	Fall time constant. (*Unit:* seconds. *Default:* 0.)

Pulse Waveform

pulse (*Vi Vp* [*D* [*Tr* [*Tf* [*Pw* [*Pp*]]]]]) [**ROUND=***RND*]

Vi	Initial voltage. (*Unit:* volts.)
Vp	Peak voltage. (*Unit:* volts.)
D	Initial delay. (*Unit:* seconds. *Default:* 0.)
Tr	Rise time. (*Unit:* seconds. *Default:* time step from **.tran**.)
Tf	Fall time. (*Unit:* seconds. *Default:* time step from **.tran**.)
Pw	Pulse width. (*Unit:* seconds. *Default:* stop time from **.tran**.)
Pp	Pulse period. (*Unit:* seconds. *Default:* stop time from **.tran**.)
RND	Rounding half-interval. A corner at time *T* is replaced by a smoothly differentiable polynomial in the interval (*T*–***RND***,*T*+***RND***). The maximum ***RND*** is half the distance to the nearest neighboring corner. (*Default:* 0—no rounding.)

Note that rise time is not necessarily a ’rise’ time, but is the time to go from the initial voltage to the pulse voltage, regardless of whether it’s smaller or larger.

Vectorized Waveform

bit| bus ({*pattern*} [**on=***A*] [**off=***Z*] [**delay=***D*] [**pw=***P*] [**rt=***R*] [**ft=***F*] [**lt=***L*] [**ht=***H*]) **[ROUND=***RND***]**

pattern	An expression consisting of one or more string or string-multiplier combinations.
A	On voltage (*Unit:* volts. *Default:* 0.001.)
Z	Off voltage (*Unit:* volts. *Default:* 0.)
D	Delay time. (*Unit:* seconds. *Default:* 0.)
P	Pulse width: $P = R + TA = F + TZ$, where TA is the time during a pulse where the voltage is "on" ($V = A$) and TZ is the time during a pulse where the voltage is "off" ($V = Z$). (*Unit:* seconds. *Default:* $10 \times 10–9$.)
R	Rise time. (*Unit:* seconds. *Default:* $1 \times 10–9$.)
F	Fall time. (*Unit:* seconds. *Default:* $1 \times 10–9$.)
L	Low time: $L = F + TZ$, where TZ is the time during a pulse where the voltage is "off" ($V = Z$). (*Unit:* seconds. *Default:* $10 \times 10–9$.)
H	High time: $H = R + TA$, where TA is the time during a pulse where the voltage is "on" ($V = A$). (*Unit:* seconds. *Default:* $10 \times 10–9$.)
RND	Rounding half-interval. A corner at time T is replaced by a smoothly differentiable polynomial in the interval ($T–RND$,$T+RND$). The maximum ***RND*** is half the distance to the nearest neighboring corner. (*Default:* 0—no rounding.)

Examples

v2 n2 GND bit ({01010 11011} on=5.0 off=0.0 pw=50n rt=10n ft=30n)

v2 generates a **bit** input. Enclosed in braces **{ }** are two binary-valued five-bit patterns specifying the waveform. The two patterns alternate in time. The **on** voltage value is 5.0 volts; the **off** voltage value is zero. The pulse width (**pw**), 50 nanoseconds, is the time the wave is either (ramping up and) on, or (dropping down and) off. The rise time (**rt**), 10 nanoseconds, is the time given for the wave to ramp from off to on; and the fall time (**ft**), 30 nanoseconds, the time given for the wave to drop from on to off.

V3 n3 GND bit ({5(01010 5(1))} pw=10n on=5.0 off=0.0)

v3 generates a *repeating* **bit** input. Two distinct patterns are given again, but now *multiplier factors* are included. The wave consists of two alternating patterns: the first pattern contains five bits; the second is a single bit. The five-bit pattern is followed by five successive repetitions of the single-bit pattern, and this combination is repeated five times. (The same pattern could be described by **{5(3(01) 4(1))}**.) The pulse width and on and off voltages are again specified, but the rise and fall times take default values.

9.2 SIMULATION ENVIRONMENT:

The pre-layout and post-layout simulations of the proposed and the existing designs have been performed using above discussed Tanner EDA Tool version 13.0. All the pre layout and post layout simulations are performed on 45nm process technology [47] for low-power applications with increasing input voltages from 0.5V to 1V in steps of 0.1V. The fabrication data of 45nm low power process technology is given in Appendix-A.

In order to prove that proposed design is consuming low power and have high performance, simulations are carried out for power consumption, delay, power-delay product at increasing input voltage, operating frequency and temperature. Simulations are also done for extracting noise immunity of existing and proposed designs. The complete input stream covering all possible combinations has been used to prove the proper functioning of the circuits.

All the simulations with increasing temperature and frequency have been done at 0.6V input voltage and supply voltage.

CHAPTER 10
SIMULATION & COMPARISON OF XOR GATES & VARIOUS ADDERS

10.1 PRE-LAYOUT PERFORMANCE COMPARISONS:

10.1.1 Performance Comparison of Existing and Proposed 3T XOR Gates:

The graphs shown in Fig. 10.1 – Fig. 10.4 depicts that the proposed designs of 3T XOR cell are a viable option for power efficient design. Between the two proposed cells the PTL based XOR cell proves its superiority over the pMOS XOR gate; but the noise immunity of both the proposed cell is same and much less than the existing one. The simulated data of Fig. 10.1 – Fig.10.4 is given in Appendix-B. The curves in Fig. 10.1-Fig.10.3 are plotted in logarithmic scale for the better view of comparison.

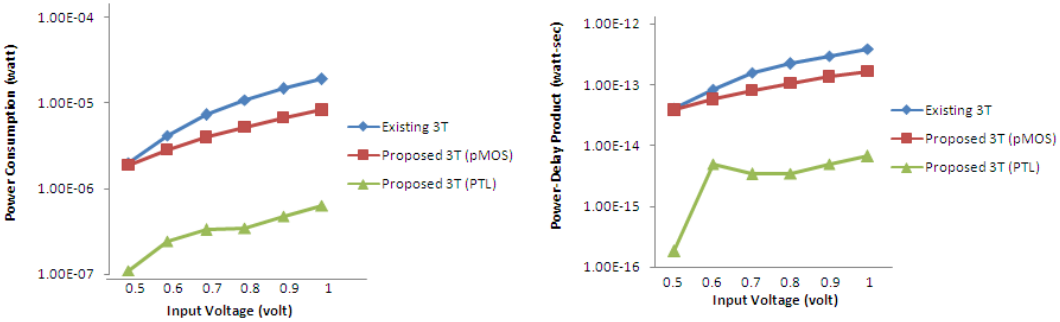

Fig. 10.1 Power Consumption & PDP at varying Input Voltages

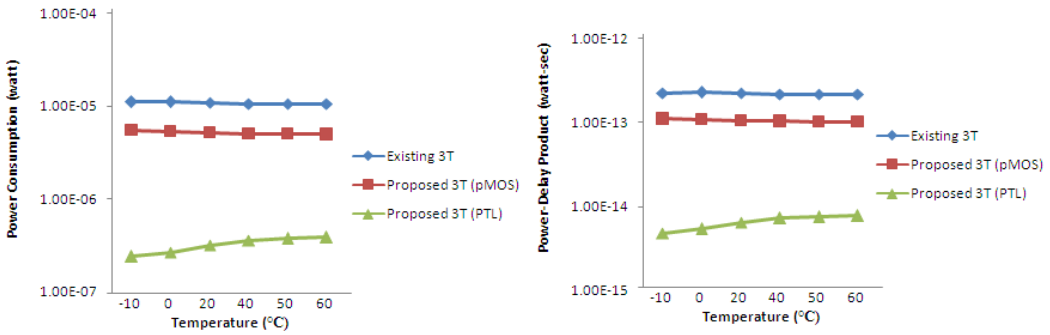

Fig. 10.2 Power Consumption & PDP at increasing Temperature

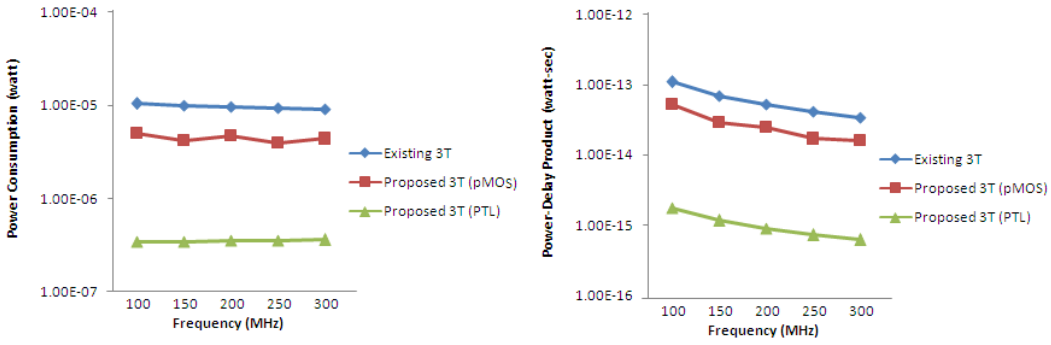

Fig. 10.3 Power Consumption & PDP at increasing Frequency

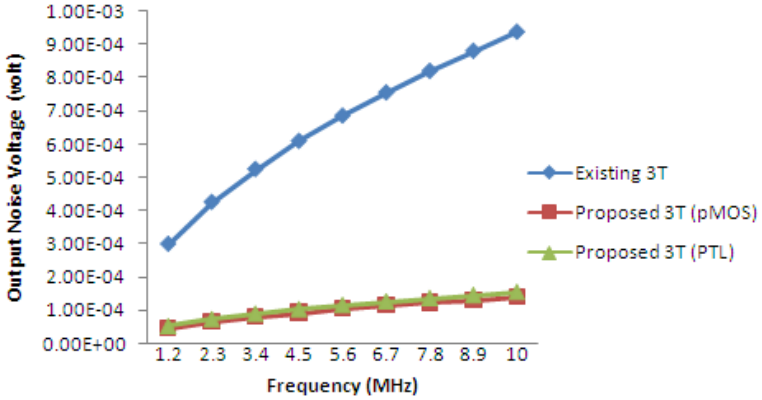

Fig. 10.4 Output Noise Voltage at varying Frequency

The graph shown in Fig. 10.1 reveals that the power consumption of proposed designs is 20% to 98% reduced than the existing one but the delay of proposed pMOS cell is slightly more than that of existing design for the input voltage 0.5V and 0.6V whereas the overall delay is decreasing with increasing input voltage. This decrease in delay is because of the inverse relationship between power and delay with respect to voltage. As depicted in Fig. 10.1, the decrease in power consumption results into reduced PDP for the proposed designs. The results shown in Fig. 10.2 with increasing temperature gives the graphical proof for the Eq. (5.1)-Eq. (5.3) included in Chapter- 5 and hence the proposed pMOS based design shows better temperature sustainability than existing one. On the other side the proposed PTL based design is showing the best results but its power

consumption and power-delay product is slightly increasing with increasing in temperature due to the characteristics of the nMOS transistor.

The result of Fig. 10.3 reveals that the power consumption and PDP with varying operating frequency of proposed 3T XOR gates is much better than the existing 3T XOR cell. As stated below, the capacitance of MOSFET is voltage dependent:

$$C = dQ/dV \tag{10.1}$$

In accumulation, the capacitance is basically the insulator capacitance $C_{i.}$ As the voltage increases, the semiconductor surface is depleted. Thus a depletion layer capacitance C_d is added in series with C_i. The total capacitance is expressed as:

$$C = \frac{C_i C_d}{C_i + C_d} \tag{10.2}$$

The capacitance decreases as width of depletion region, W grows until final inversion is reached at threshold voltage. After inversion is reached, the capacitance depends on whether measurements are made at high or low frequency. For increasing input frequency, since the gate voltage is changed rapidly, the charge in the inversion layer cannot change in response to the changing voltage, and thus does not contribute to the capacitance. Hence, corresponding to the higher frequency, the semiconductor capacitance is at minimum value. And as discussed in Chapter- 2 that power consumption is proportional to capacitance, thus reduction of capacitance at higher frequency leads to lesser power consumption. The delay of the circuit also varies linearly with capacitance [2] thus; reduction of capacitance with increasing operating frequency also decreases delay of the circuit.

Fig. 10.3 also shows the improvement in PDP of proposed cells with varying operating frequency and Fig. 10.4 reveals that the output noise voltage of the proposed designs is less than the existing one which results into better noise immunity of the proposed cells. Also the difference in output noise voltage between existing and proposed ones is increasing with the increase in frequency.

10.1.2 Performance Comparison of Existing and Proposed 8T Full Adders:

The graphs based on schematic simulations shown in Fig. 10.5 - Fig. 10.8 results into the better performance of proposed designs in comparison to existing design. The data for graphs shown in Fig. 10.5 – Fig. 10.8 is included in Appendix-C.

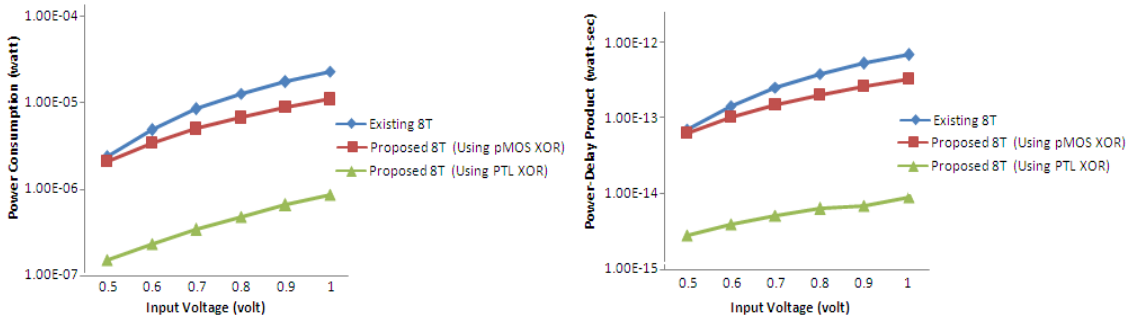

Fig. 10.5 Power Consumption & PDP at varying Input Voltages

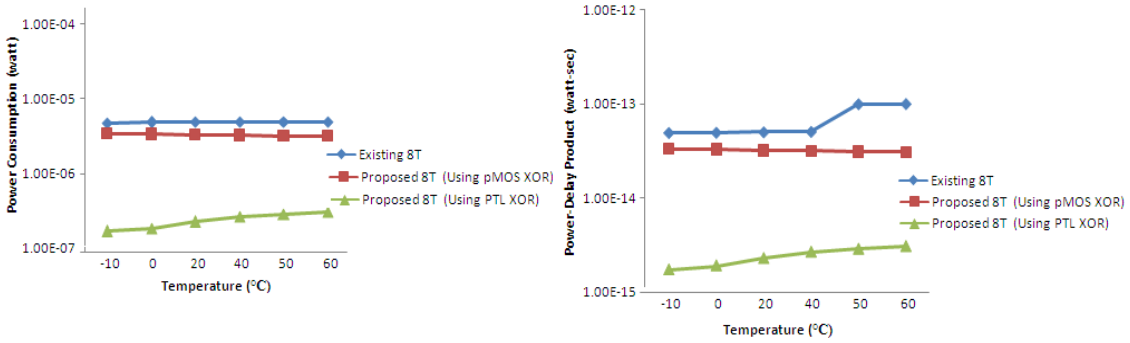

Fig. 10.6 Power Consumption & PDP at varying Temperature

Fig. 10.5 reveals the superiority of proposed 8T full adder cells in terms of power consumption and PDP at different input voltages. The PDP curve shown in Fig. 10.5 reveals that for low input voltages the existing design and the proposed pMOS based adder cell have comparable performance but as the input voltage is increased the proposed pMOS adder cell overcomes the existing one. Even at varying temperature, power consumption of the proposed designs is reduced by large scale. The results shown in Fig. 8.9 and Fig. 8.10 with increasing temperature follows Eq.(5.1) - Eq.(5.3) and shows the decrement in power consumption with increasing temperature. Fig. 10.6 also reveals the unpredictable behavior of existing circuit from 40°C to 50°C temperature which demonstrates that existing design may not be a good option for higher temperatures. The proposed pMOS based design has almost constant as well as improved

power consumption and hence acquire better temperature sustainability over existing design. Also Fig. 10.6 shows that the temperature sustainability of the proposed PTL based cell is best among the three designs but it due to the nMOS used the power consumption and hence PDP is gradually increasing with rising temperature.

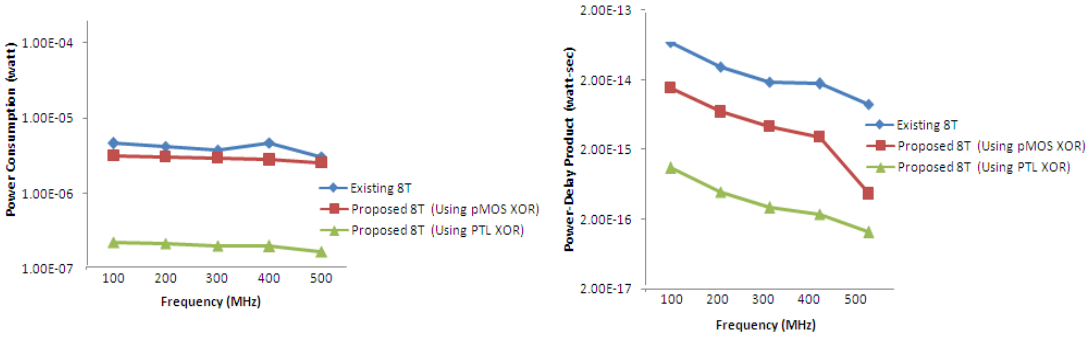

Fig. 10.7 Power Consumption & PDP at varying Frequency

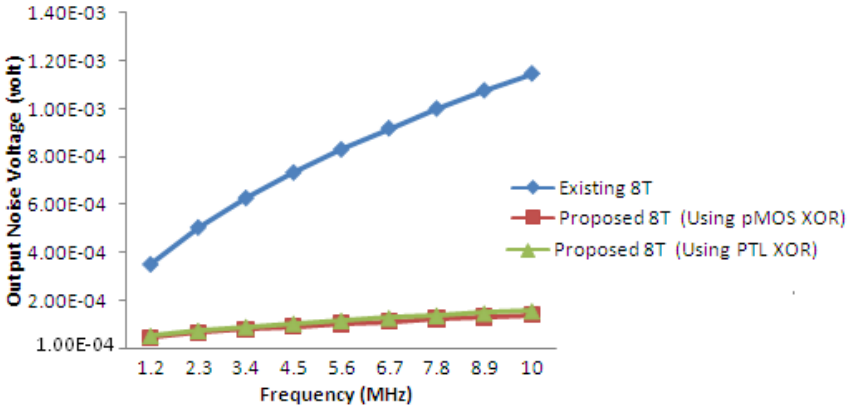

Fig. 10.8 Output Noise Voltage at increasing Frequency

The comparison of same parameters of existing and proposed adders with increasing operating frequency is shown in Fig. 10.7 and reveals that the proposed full adders can be the better option for high frequency applications. Also, as shown in Fig. 10.8 the noise immunity of the proposed cells is remarkably better than the existing one.

10.1.3 Performance Comparison of Existing and Proposed 9T Full Adders Based on Carry Logic:

The ***abbreviations $9E_C$, $9P_{C(pMOS)}$ and $9P_{C(PTL)}$ are used for representing existing 9T full adder based on carry logic as well as proposed 9T full adder based on carry logic using pMOS XOR gate and PTL XOR gate respectively***. Fig. 10.9 –Fig. 10.12 shows the performance of the designs with increasing input voltage, temperature and frequency. The data for these graphs is included in Appendix-D.

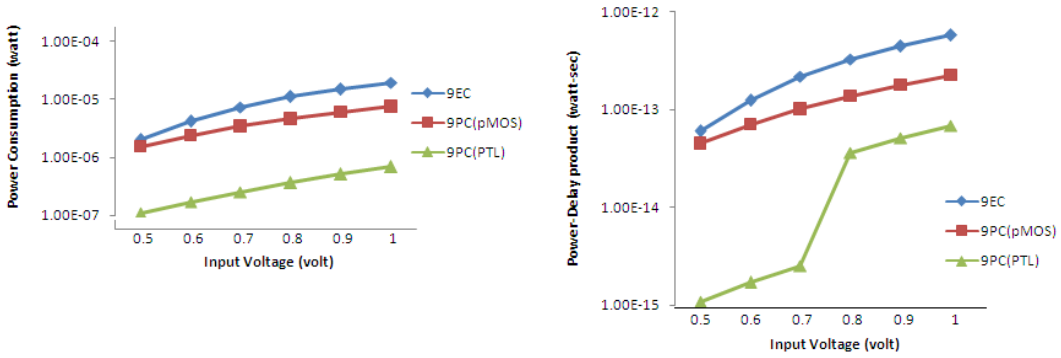

Fig. 10.9 Power Consumption & PDP at varying Input Voltages

Fig. 10.9 depicts remarkable reduction in power consumption of the proposed designs but with a slight increase in delay. Since, PDP is a better metric to evaluate circuit's performance; curve for PDP has also been plotted and reveals improved PDP of proposed designs thus ensuring the better performance of proposed design at higher voltages. The power consumption of the proposed full adder based on pMOS XOR cell is in the same powers as that of existing one but the proposed adder designed using PTL XOR cell is showing ten times reduced power consumption than the existing adder. Similarly, the PDP of PTL based full adder is much less at low voltages and as we are moving towards high voltage range the difference between the curves is reducing but still the PDP of proposed cells is better than the existing one.

The better performance of proposed designs is again shown in Fig. 10.10 and Fig. 10.11 with varying temperature and frequency. In Fig. 10.10 the plot for power consumption as well as PDP of pMOS based adder is slowly decreasing with increase in temperature which ensures that the proposed pMOS based adder is suitable for high

temperature applications. On the other side the curves of existing and proposed PTL based adders are slightly increasing with increase in temperature. Fig. 10.12 also reveals the less output noise and thus better noise immunity of the proposed designs.

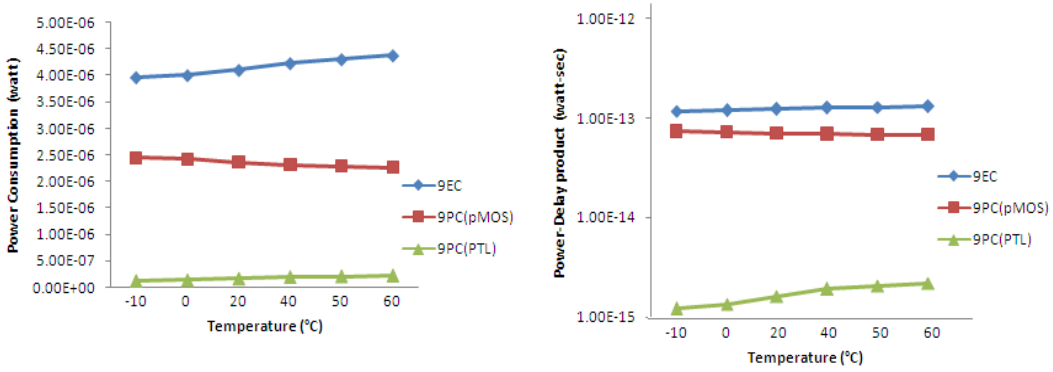

Fig. 10.10 Power Consumption & PDP at increasing Temperature

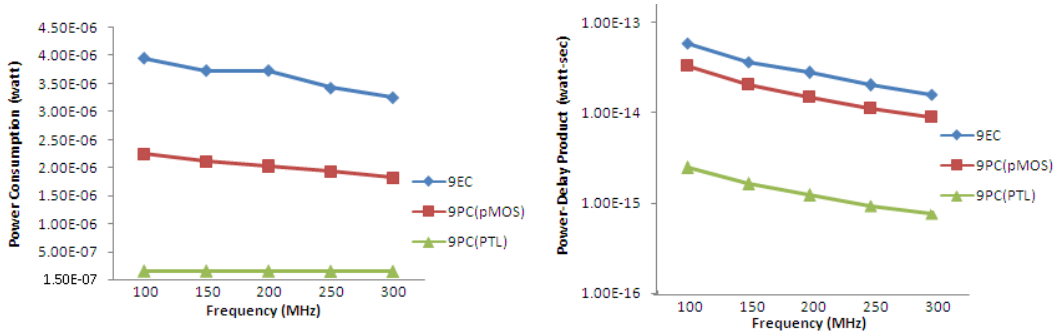

Fig. 10.11 Power Consumption & PDP at increasing Frequency

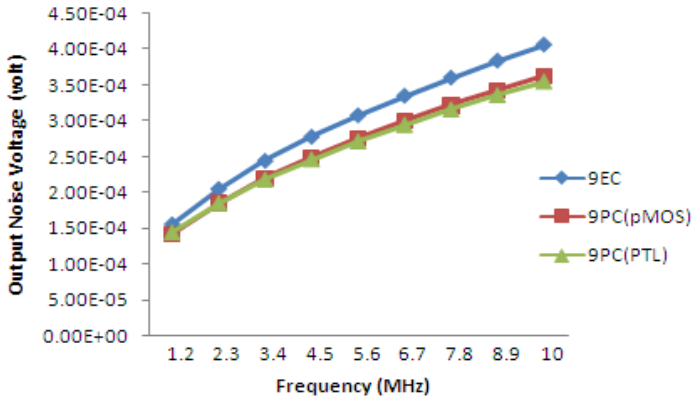

Fig. 10.12 Output Noise Voltage with increasing Frequency

The above curves for 9T full adder based on carry logic reveals that the proposed designs of 9T full adder has much better performance in terms of power, PDP, noise immunity and temperature sustainability than existing design.

10.1.4 Performance Comparison of Existing and Proposed 9T Full Adders Based on Inversion Logic:

The ***abbreviations $9E_I$, $9P_{I(pMOS)}$ and $9P_{I\ (PTL)}$ are used for representing existing 9T full adder based on inversion logic as well as proposed 9T full adder based on inversion logic using pMOS XOR gate and PTL XOR gate respectively***. Fig. 10.13 –Fig. 10.16 shows the performance of the 9T adder designs with increasing input voltage, temperature and frequency. The data for these graphs is included in Appendix-E.

Fig. 10.13 depicts 4%-10% improved power consumption of the proposed pMOS based design as compared to existing one and the proposed PTL based design is showing the power consumption in nanowatts rather than microwatts and proves itself to be the best option for low power applications. Similarly, the PDP of the proposed pMOS based design is improved to 3%-10% than the existing one and still the proposed PTL based adder design is maintaining its superiority among all.

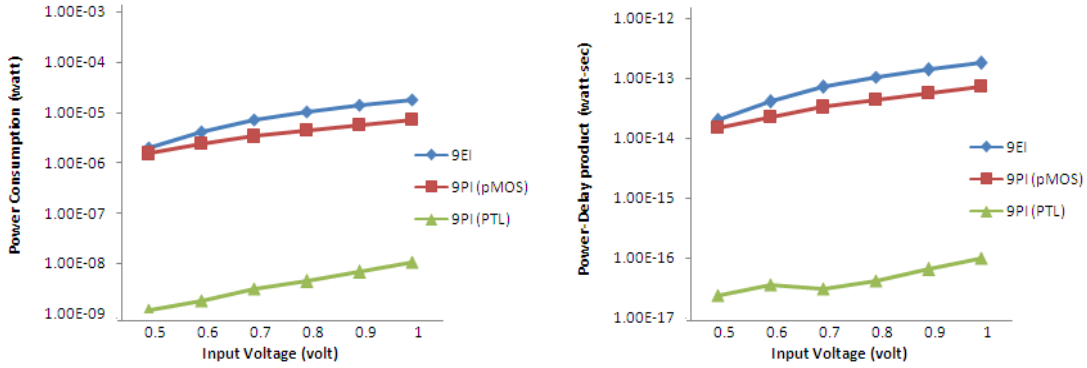

Fig. 10.13 Power Consumption & PDP at varying Input Voltages

The results shown in Fig.10.14 depict 2% improvement in power consumption of proposed pMOS based adder with varying temperature and results are in accordance with Eq. (5.1) - Eq. (5.3). Again the proposed PTL based 9T adder achieves the highest rank

among the three designs ensuring the power results in nanowatts. The delay of existing design as stated in Appendix-E is drastically increased below room temperature which may lead to failure of device at higher temperatures. The proposed designs overcome all the faults of the existing design and maintain lower power consumption as well as delay at higher temperatures. Fig. 10.14 also demonstrates better PDP of proposed designs maintaining its consistency with varying temperatures.

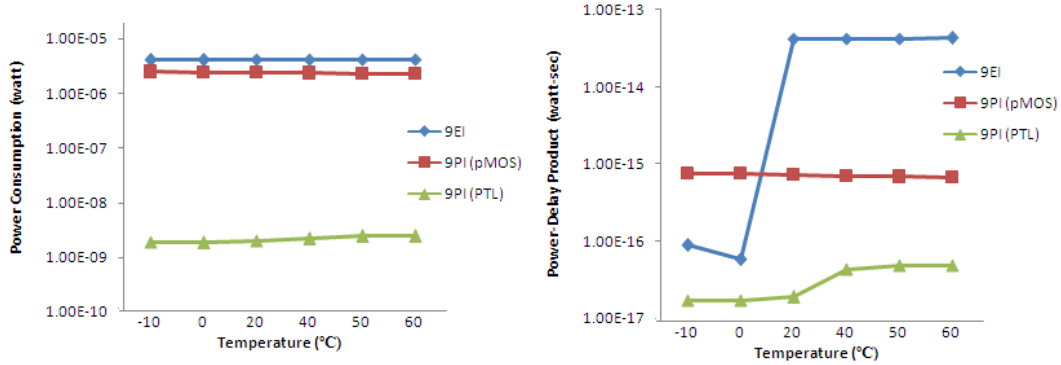

Fig. 10.14 Power Consumption & PDP at varying Temperature

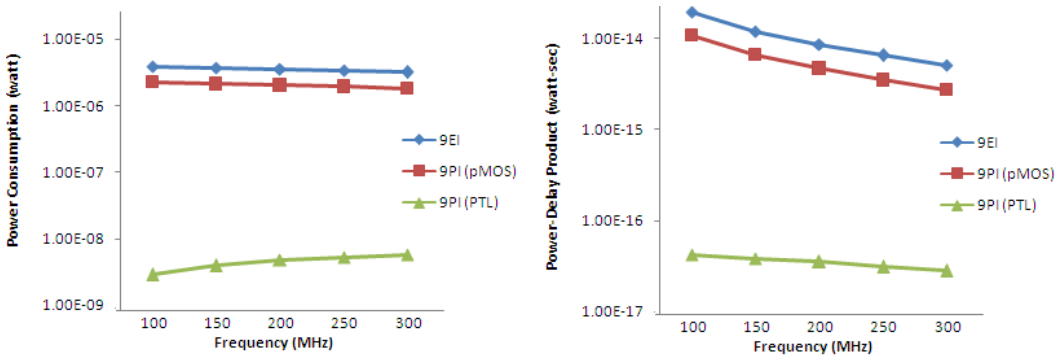

Fig. 10.15 Power Consumption & PDP at varying Frequency

Fig. 10.15 shows 18% less power consumption of proposed design using pMOS XOR cell with varying operating frequency having almost similar delay to existing design. Even at same delay, due to lesser power consumption, PDP of proposed pMOS based design is 9%-25% better than that of existing one which is clearly evident from Fig. 10.15. The proposed 9T adder designed using PTL XOR cell maintains similar and best results as discussed earlier. On the other hand, the output noise voltage of proposed

pMOS based adder and the existing adder is same which indicates that both are equally noise immune but the noise voltage of proposed PTL based adder is reducing gradually with increasing frequency.

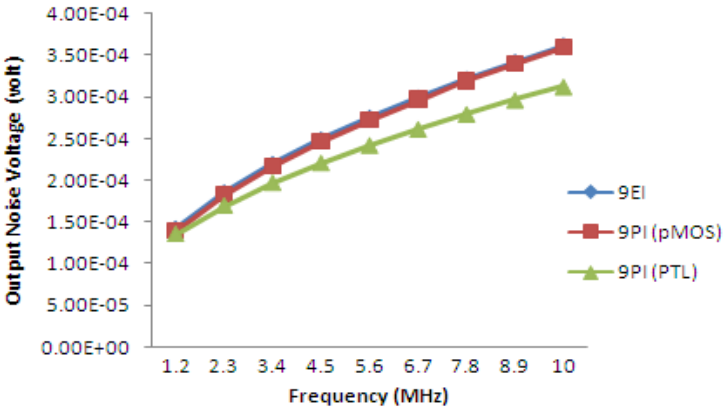

Fig. 10.16 Output Noise Voltage at increasing Frequency

10.2 POST-LAYOUT PERFORMANCE COMPARISONS:

After the physical layout designing post-layout simulations are carried out with parasitic extracted in RC extraction embedded into the schematic. Power consumption is a function of load capacitance, frequency of operation, and supply voltage hence, reduction of any one of these is beneficial. A reduction in power consumption provides several benefits. Less heat is generated, which reduces problems associated with high temperature, such as the need for heat sinks. An additional benefit of the reduced power consumption is the extended life of the battery in battery powered systems.

Table XIV – Table XVII shows significant improvement in total as well as output parasitic capacitances. The proposed designs of XOR cell and full adder cell have less parasitic capacitance than existing design and, as power consumption is directly proportional to capacitance, therefore proposed design will also have remarkable reduction in power consumption. The results for the same are shown in Fig. 10.17- Fig. 10.24.

10.2.1 Performance Comparison of Existing and Proposed 3T XOR Gates:

Table XIV. Extracted Parasitic Capacitances for Existing and Proposed 3T XOR Cells

Designs	**Parasitic Capacitance (fF)**					
	Existing		**Proposed pMOS**		**Proposed PTL**	
	Total	**Output**	**Total**	**Output**	**Total**	**Output**
3T XOR Cell	0.33122	0.16367	0.25124	0.09792	0.12550	0.05790

The graphs based on post layout simulations shown in Fig. 10.17 and Fig. 10.18 results into the better performance of the proposed designs in comparison to existing design. Fig.10.17 shows that the PDP of proposed pMOS XOR cell is same as that of existing one at the start of super threshold region but as the curve moves deeper into super threshold region the difference between the two is increasing with increase in input voltage and at the same time the difference between the power-delay product of proposed pMOS XOR cell and proposed PTL XOR cell is reducing with increasing voltage. The PDP versus temperature plot shown in Fig.10.17 again satisfies the relation between power consumption and temperature for the pMOS. The results for power consumption and delay extracted from the simulation from which PDP has been calculated for various parameters are included in Appendix-B.

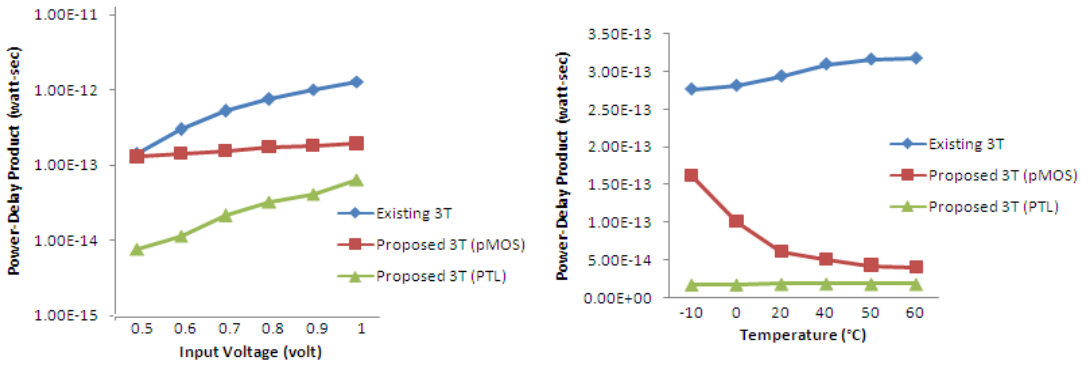

Fig. 10.17 PDP with increasing Input Voltages & Temperature

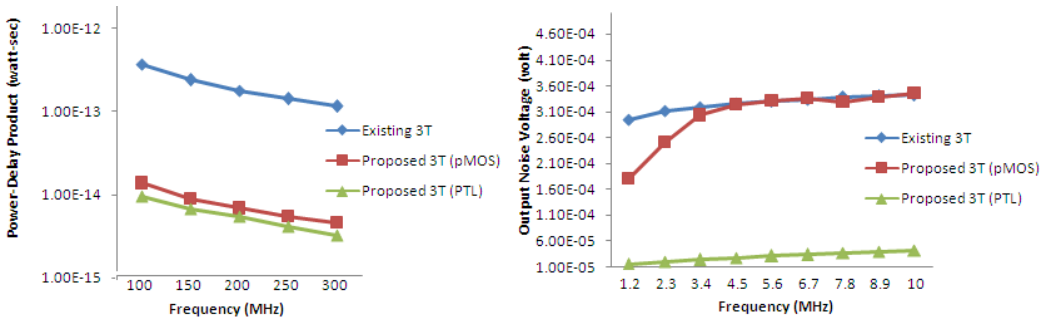

Fig. 10.18 PDP & Output Noise Voltage with increasing Frequency

The curve for output noise voltage of proposed pMOS based adder in Fig. 10.18 depicts that at lower frequency the circuit is more noise immune than the existing one but as we are moving towards the high frequencies both are comparable. Therefore, if the noise immunity is concerned the proposed PTL based design outperforms remaining ones.

10.2.2 Performance Comparison of Existing and Proposed 8T Full Adders:

Table XV. Extracted Parasitic Capacitances for Existing and Proposed 8T Adder Cells

Designs	**Parasitic Capacitance (fF)**					
	Existing		**Proposed pMOS**		**Proposed PTL**	
	Total	**Output**	**Total**	**Output**	**Total**	**Output**
8T Adder Cell	0.82214	0.22163	0.56415	0.12523	0.28552	0.07138

The graphs based on post layout simulations shown in Fig. 10.19 and Fig. 10.20 results into the better performance of proposed designs in comparison to existing design. The consistency of the data extracted for the 3T XOR cells has still been maintained while analyzing the performance of 8T adder cells and as a result the proposed designs outperforms the existing one. The simulated data for power consumption and delay used to calculate PDP for these graphs is included in Appendix-C.

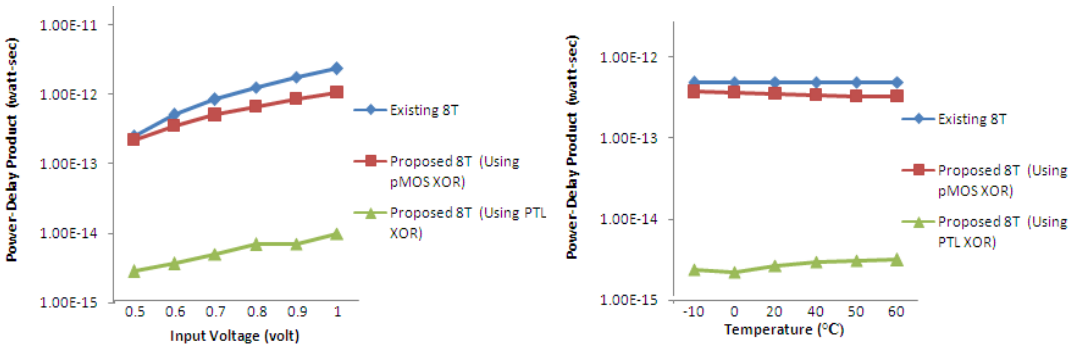

Fig. 10.19 PDP with increasing Input Voltages & Temperature

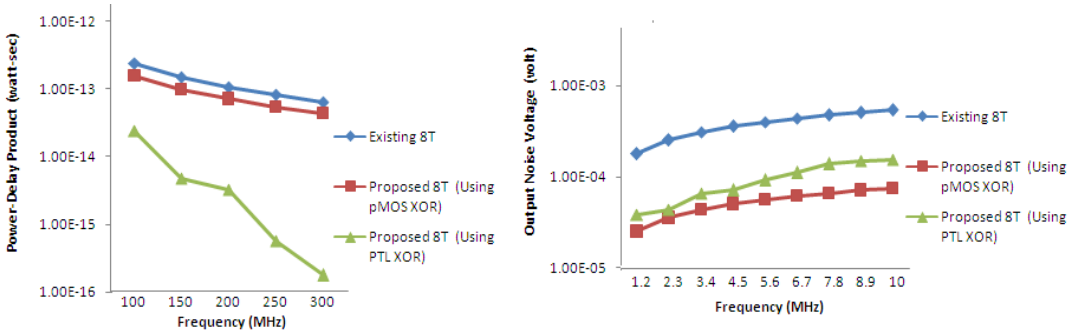

Fig. 10.20 PDP & Output Noise Voltage with increasing Frequency

10.2.3 Performance Comparison of Existing and Proposed 9T Full Adders Based on Carry Logic:

Table XVI. Extracted Parasitic Capacitances for Existing and Proposed 9T Full Adders Based on Carry Logic

Designs	**Parasitic Capacitance (fF)**					
	Existing		**Proposed pMOS**		**Proposed PTL**	
	Total	**Output**	**Total**	**Output**	**Total**	**Output**
9T Adder cell based on carry logic	0.54031	0.66773	0.50493	0.60875	0.34937	0.06677

The graphs based on post layout simulations shown in Fig. 10.21 and Fig. 10.22 proves the superiority of proposed designs over existing one in terms of different performance parameters with varying voltage, temperature and operating frequency. The

simulation results for power consumption and delay from which PDP has been calculated is included in Appendix-D.

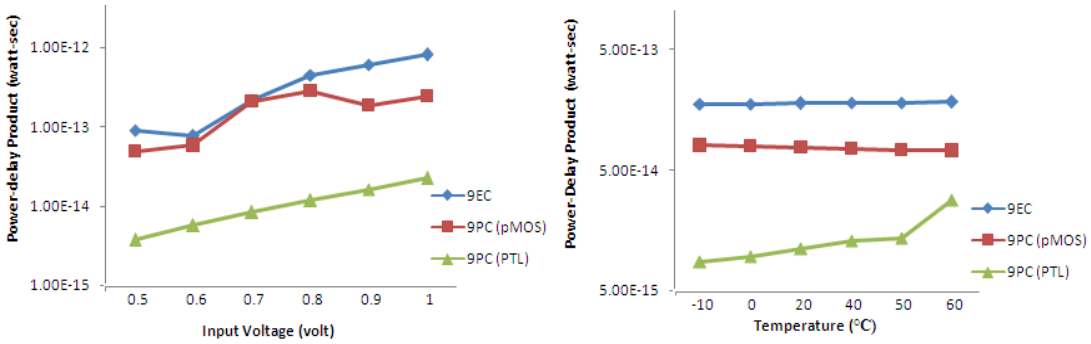

Fig. 10.21 PDP with increasing Input Voltages & Temperature

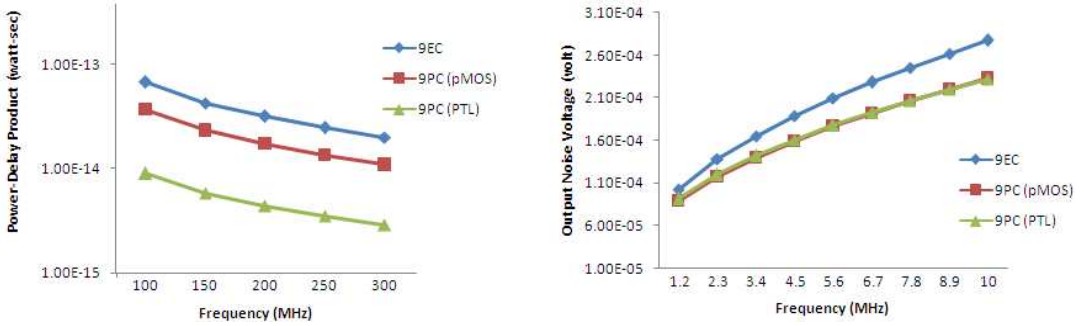

Fig. 10.22 PDP & Output Noise Voltage with increasing Frequency

10.2.4 Performance Comparison of Existing and Proposed 9T Full Adders Based on Inversion Logic:

Table XVII. Extracted Parasitic Capacitances for Existing and Proposed 9T Full Adders Based on Inversion Logic

Designs	**Parasitic Capacitance (fF)**					
	Existing		**Proposed pMOS**		**Proposed PTL**	
	Total	**Output**	**Total**	**Output**	**Total**	**Output**
9T Adder cell based on inversion logic	0.54869	0.71647	0.50198	0.71647	0.34446	0.07106

The graphs based on post layout simulations shown in Fig. 10.23 and Fig. 10.24 results into the better performance of proposed designs in comparison to existing design. The extracted data of power consumption and delay used for calculating PDP is included in Appendix-E.

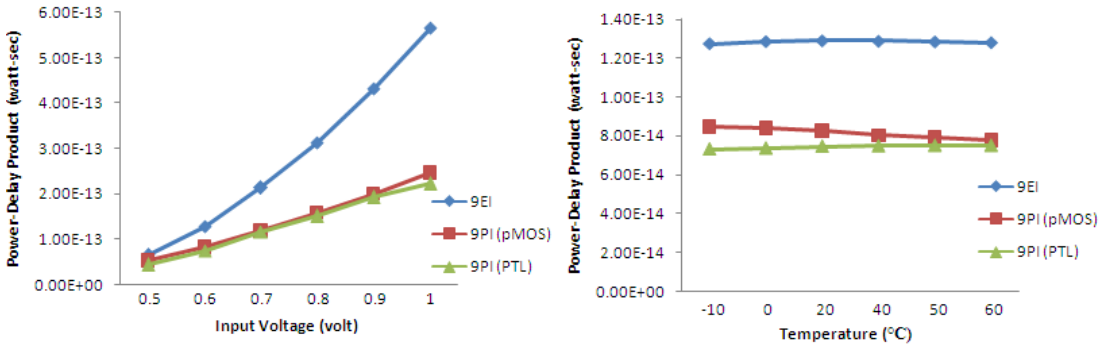

Fig. 10.23 PDP with increasing Input Voltages & Temperature

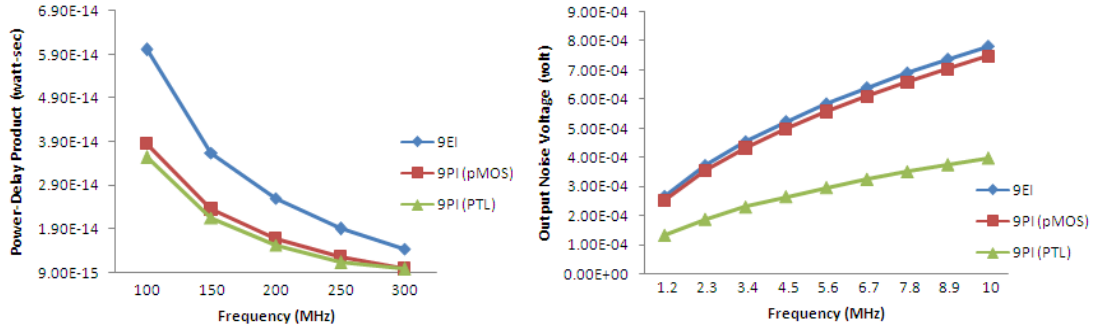

Fig. 10.24 PDP & Output Noise Voltage with increasing Frequency

From the above results it is evident that the proposed 9T full adders based on carry logic as well as proposed 9T full adders based on inversion logic outperforms their peer designs. Also the variations in the results with respect to various design parameters are same as in the basic building cell i.e.; 3T XOR cells.

CHAPTER 11
4-BIT RIPPLE CARRY ADDER

The full adder circuit can now be used as the basic building block of a 4-Bit binary adder [4], which accepts two 4-Bit binary numbers as input and produces the binary sum of 4-Bit or 8-Bit at the output. The simplest such adder can be constructed by a cascade connection of four full adders, where each adder performs a binary addition, produces the corresponding sum bit, and passes the carry output on to the next stage. Hence, this cascade configuration is called the 4-bit ripple carry adder since each carry bit ripples to the next full adder. The overall speed of the carry ripple adder is obviously limited by the delay of the carry bits rippling through the carry chain. Fig. 11.1 shows the interconnection of four *full adder* circuits to provide a 4-Bit ripple carry adder.

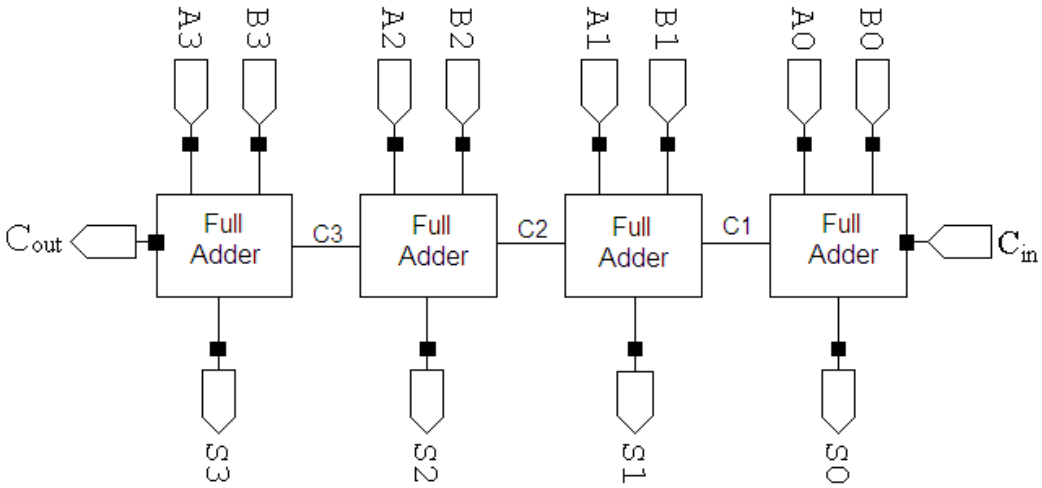

Fig. 11.1 Block Representation of 4-Bit RCA

It has many advantages which include *low power, low area* and a *simple layout* however; the ripple carry adder is relatively *slow*, since each full adder must wait for the carry bit to be calculated from the *previous* full adder. The delay of the adder is linearly dependent on the *bit-width* (N) of the adder i.e.;

the delay increases with the increase in the number of *bits* to be added. The *critical path* of the ripple carry adder consists of the *carry chain* from the *first* full adder to the *last*. Therefore, during circuit-level design, the carry signal is frequently assigned to the transistor closest to the gate output for the carry computation.

The existing and proposed 8T and proposed 9T full adder cells have been used to implement 4-Bit RCA [48] and their performance have been analyzed to verify the driving capability and performance of full adder cell in complex circuitry. Fig. 11.2, Fig. 11.3 and Fig. 11.4 shows the schematic of 4-Bit RCA with existing, proposed pMOS based 8T full adder and proposed PTL based full adder respectively.

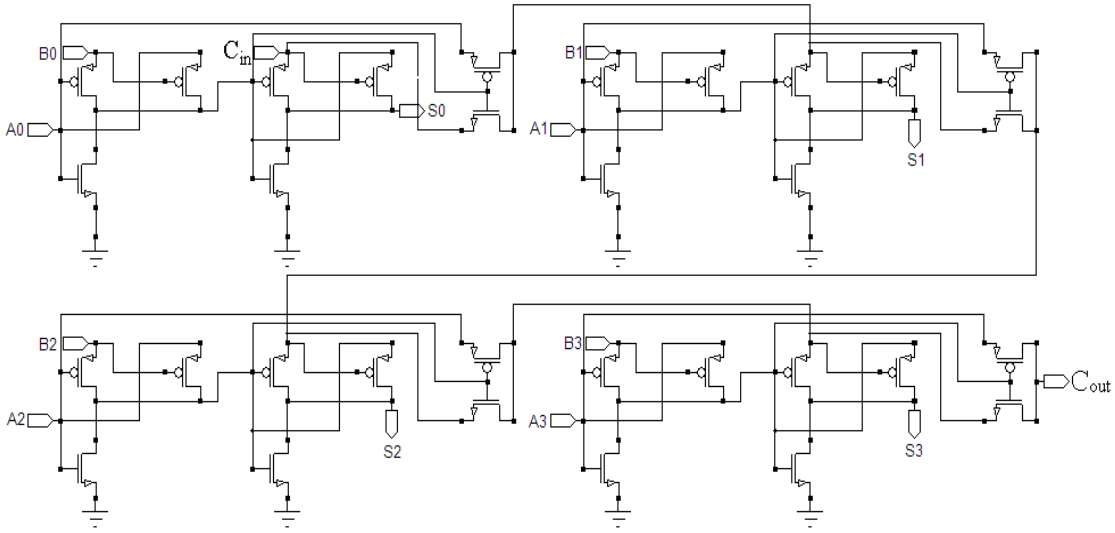

Fig.11.2 Schematic of 4-bit RCA using existing 8T full adder

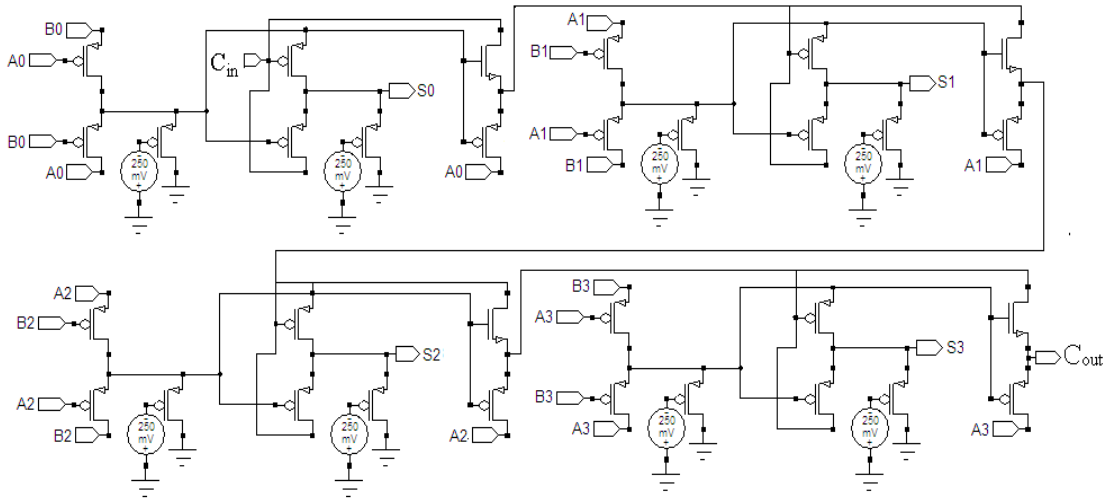

Fig. 11.3 Schematic of 4-bit RCA using proposed pMOS 8T full adder

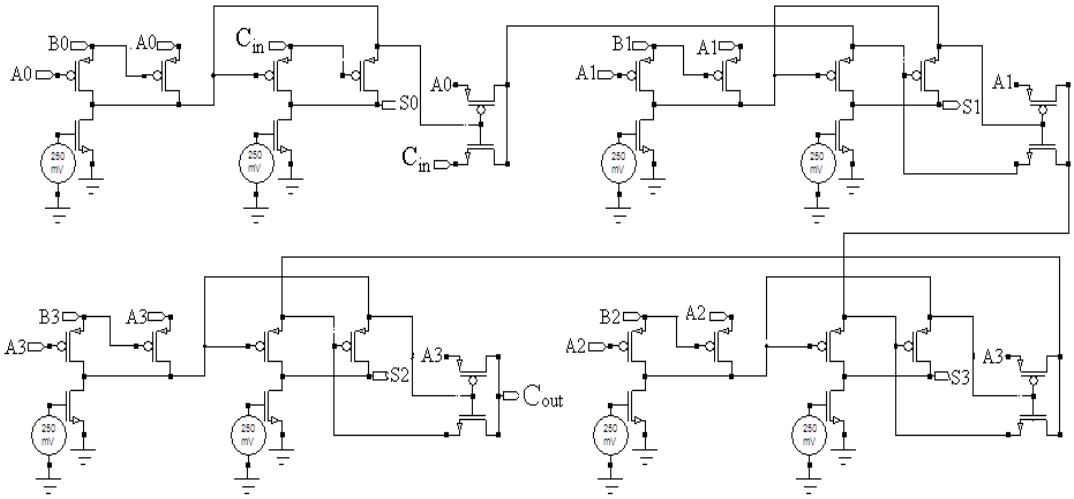

Fig. 11.4 Schematic of 4-bit RCA using proposed PTL 8T full adder

Fig. 11.5 - Fig. 11.7 shows the In-Out waveforms of the above drawn 4-bit RCA circuits. The threshold loss problem of existing 8T full adder cell is also obtained in RCA. For certain input combinations as shown in Fig. 11.5, the output level is at 50% of logic high or below which can be interpreted as logic low signal and thus result into improper functioning of the systems which is not desirable. While, proposed 8T full adders have consistent performance in RCA implementation as seen in Fig. 11.6 and Fig. 11.7.

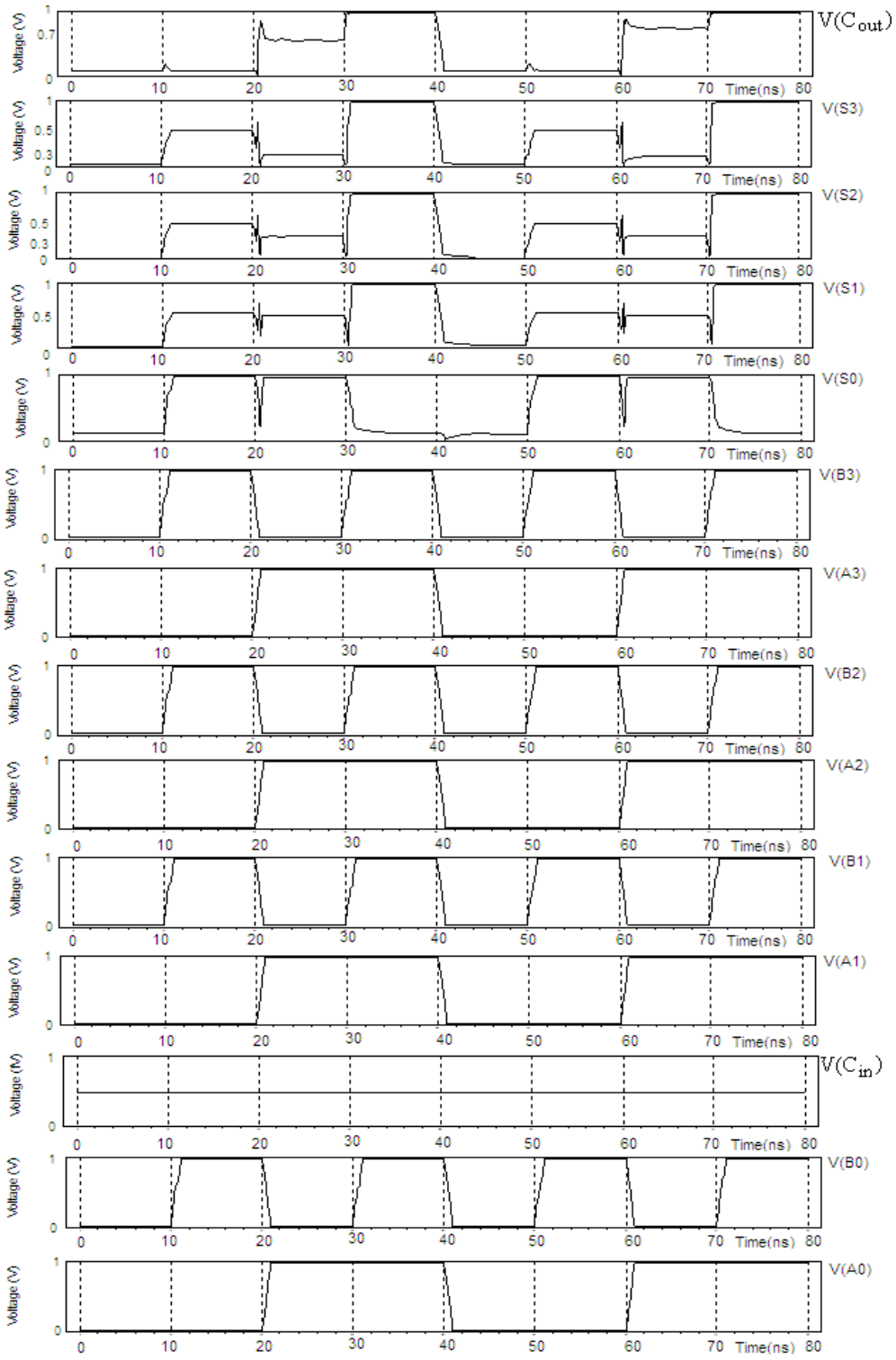

Fig. 11.5 In-Out waveform of 4-bit RCA using existing 8T full adder

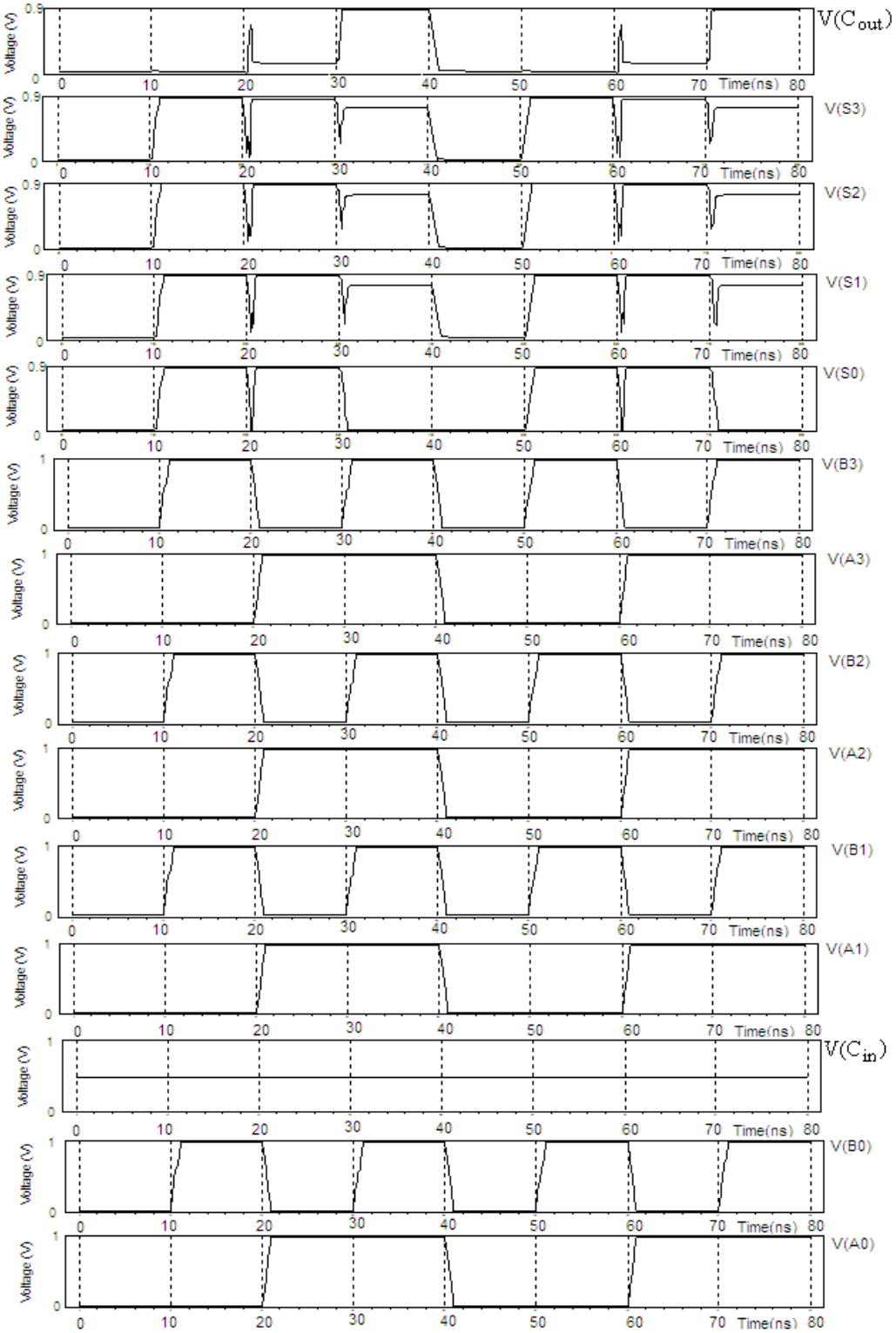

Fig. 11.6 In-Out waveform of 4-bit RCA using proposed pMOS 8T full adder

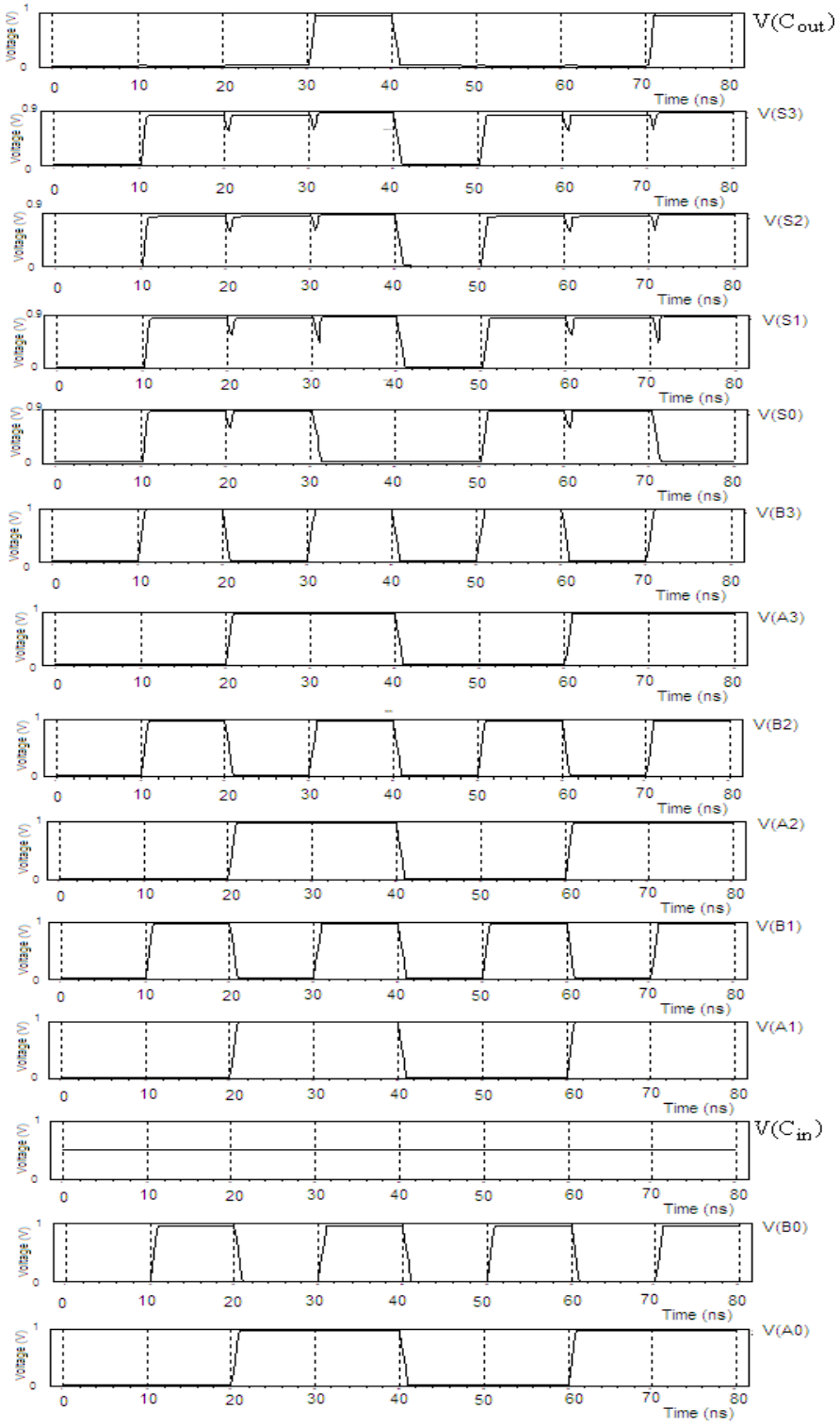

Fig. 11.7 In-Out waveform of 4-bit RCA using proposed PTL 8T full adder

The proposed 9T adders have been chosen to implement 4-bit RCA because of their better performance over the existing 9T adder designs as demonstrated in section 10.1 and 10.2. The schematic of RCA using proposed pMOS 9T full adder based on carry logic and RCA using proposed pMOS 9T full adder based on inversion logic are shown in Fig. 11.8 and Fig. 11.9 respectively. Fig. 11.10 and Fig. 11.11 are the input output waveform of the circuits shown in Fig. 11.8 and Fig. 11.9.

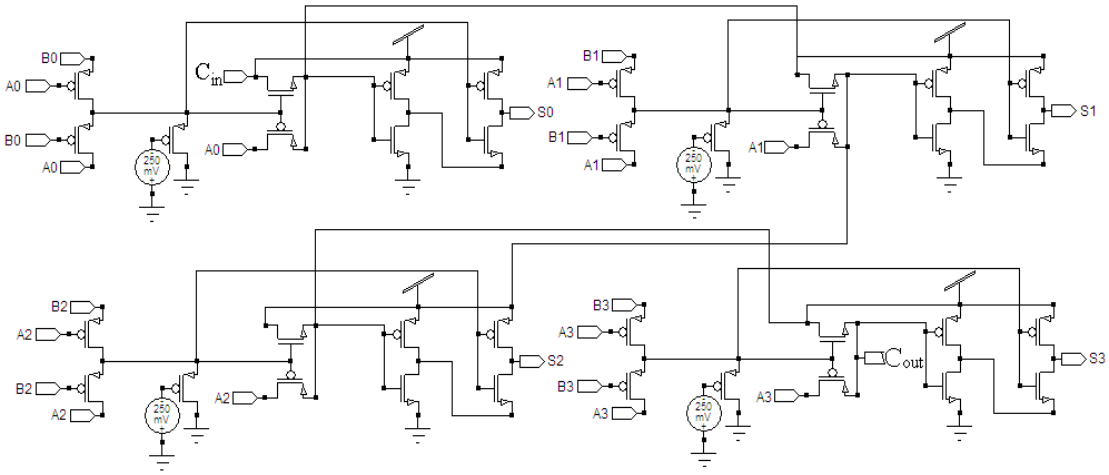

Fig. 11.8 Schematic of 4-bit RCA using proposed pMOS 9T full adder based on carry logic

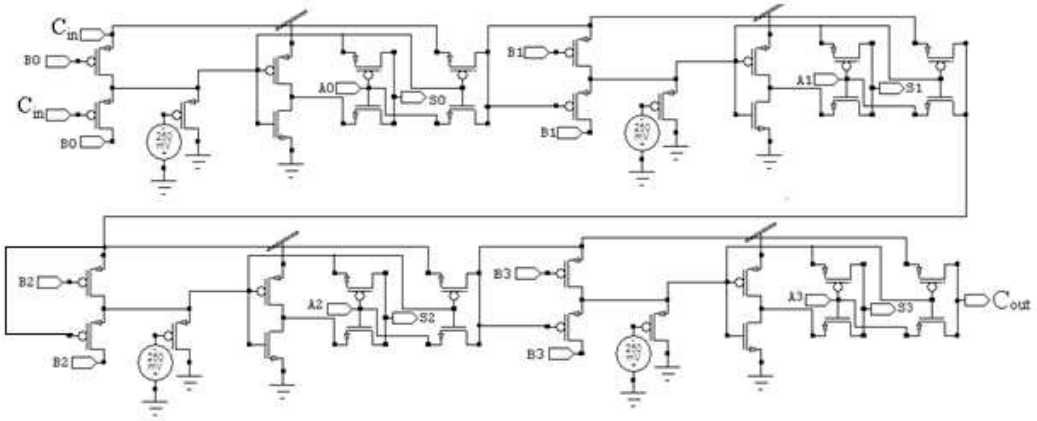

Fig. 11.9 Schematic of 4-bit RCA using proposed pMOS 9T full adder based on inversion logic

Both the proposed design gives acceptable performance and good driving capability along with negligible threshold loss which makes them suitable to be used in

cascade configuration. Thus, the basic cell behind RCA i.e. 9T full adder cell can be chosen as a suitable option for large power efficient systems.

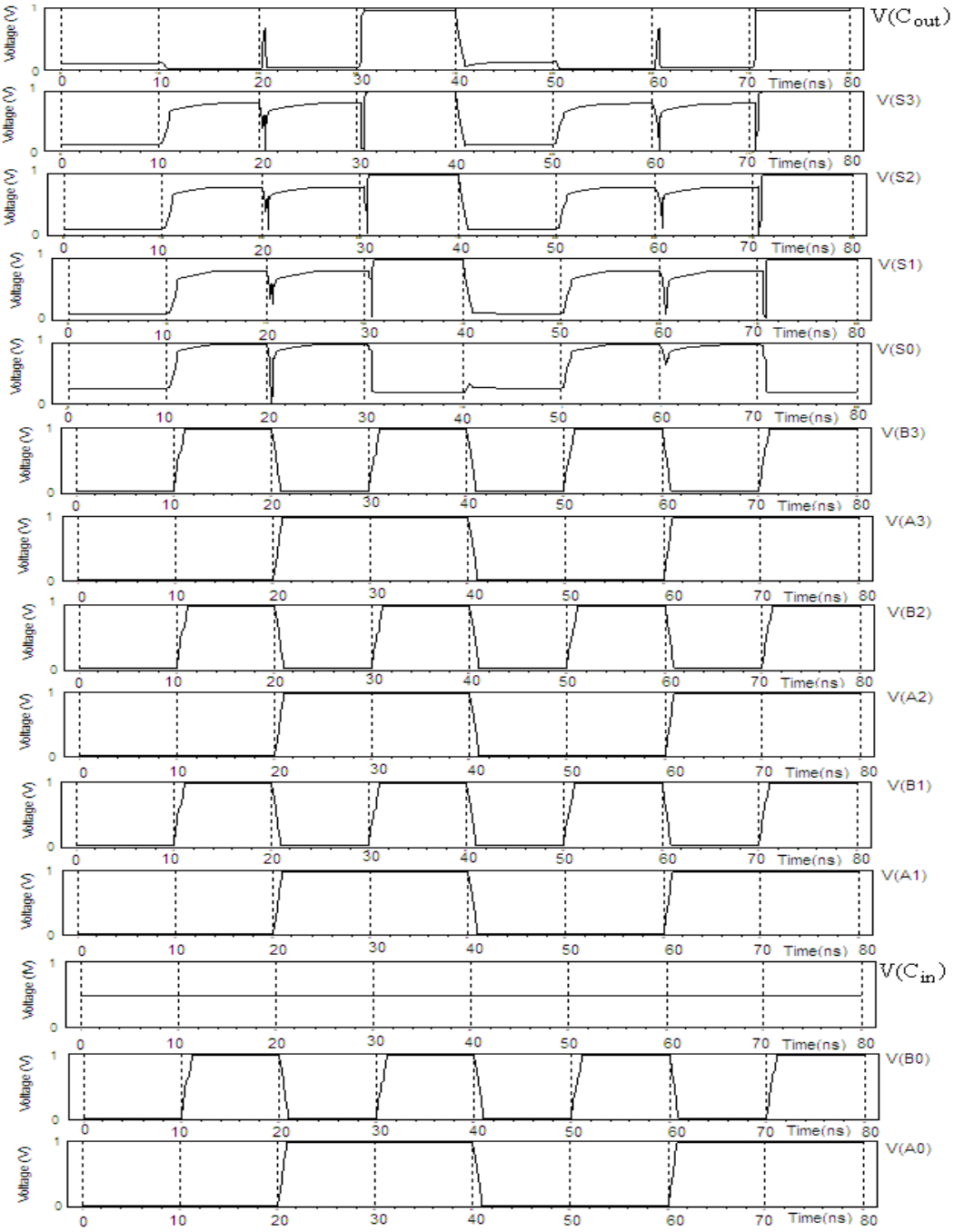

Fig.11.10 In-Out Waveform of RCA using proposed pMOS 9T full adder based on carry logic

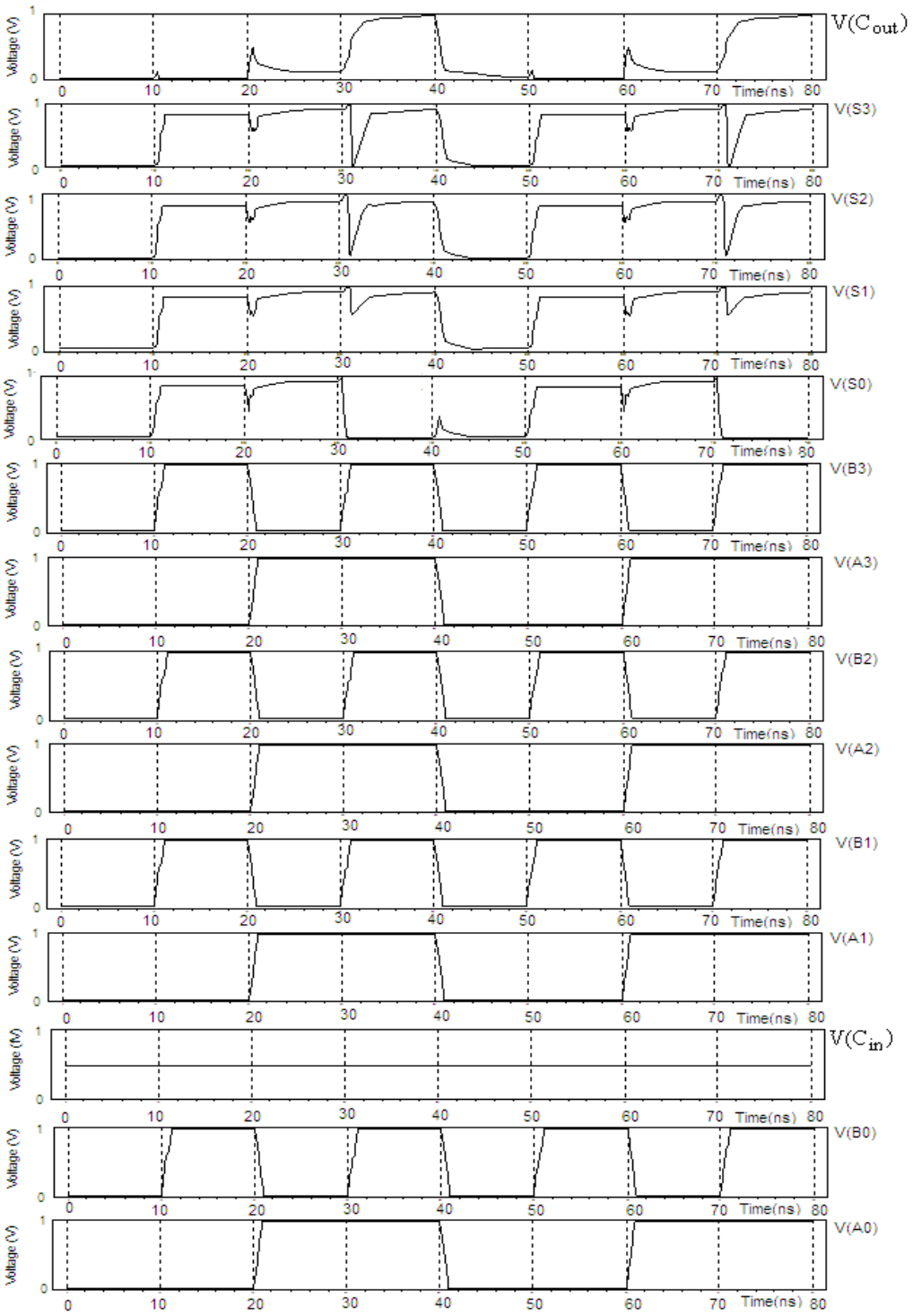

Fig. 11.11 In-Out Waveform of RCA using proposed pMOS 9T full adder based on inversion logic

The schematic of RCA using proposed PTL 9T full adder based on carry logic and RCA using proposed PTL 9T full adder based on inversion logic are shown in Fig. 11.12 and Fig. 11.13 respectively. Fig. 11.14 and Fig. 11.15 are the input output waveform of the circuits shown in Fig. 11.12 and Fig. 11.13. The waveforms shown in Fig. 11.14 and 11.15 reveals the reduced threshold loss than the previous designs implemented using pMOS cell. Therefore, PTL based RCA designs are more appropriate to be used as they have better driving capability.

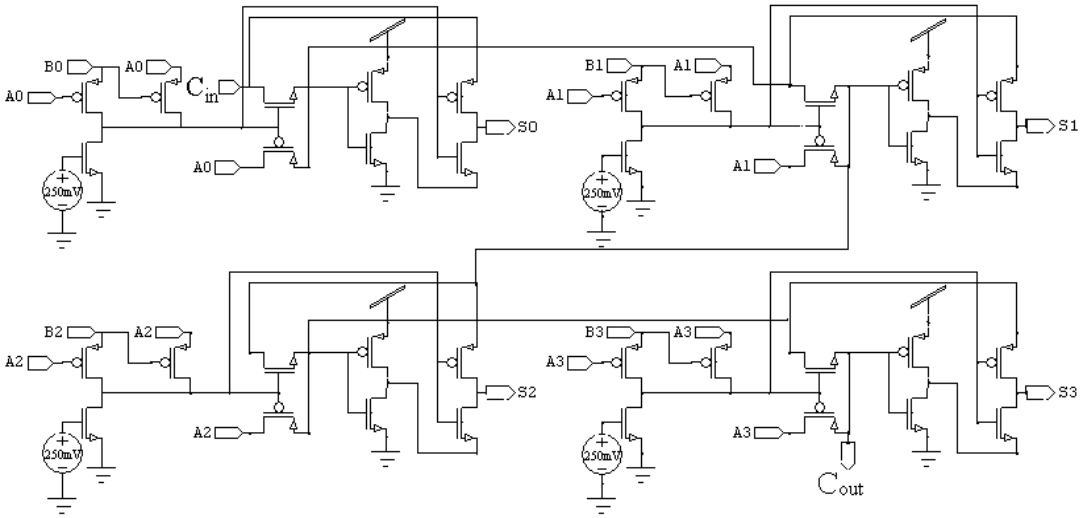

Fig. 11.12 Schematic of 4-bit RCA using proposed PTL 9T full adder based on carry logic

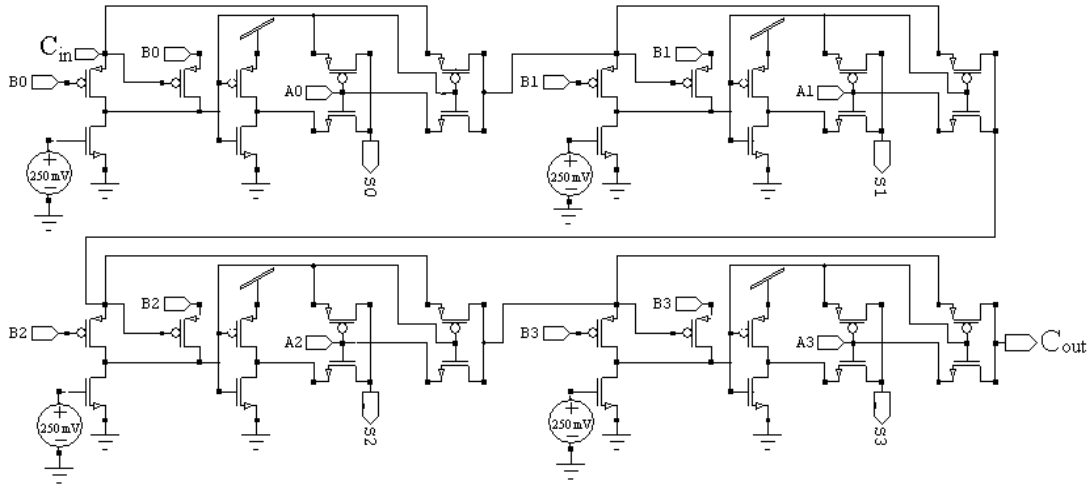

Fig. 11.13 Schematic of 4-bit RCA using proposed PTL 9T full adder based on inversion logic

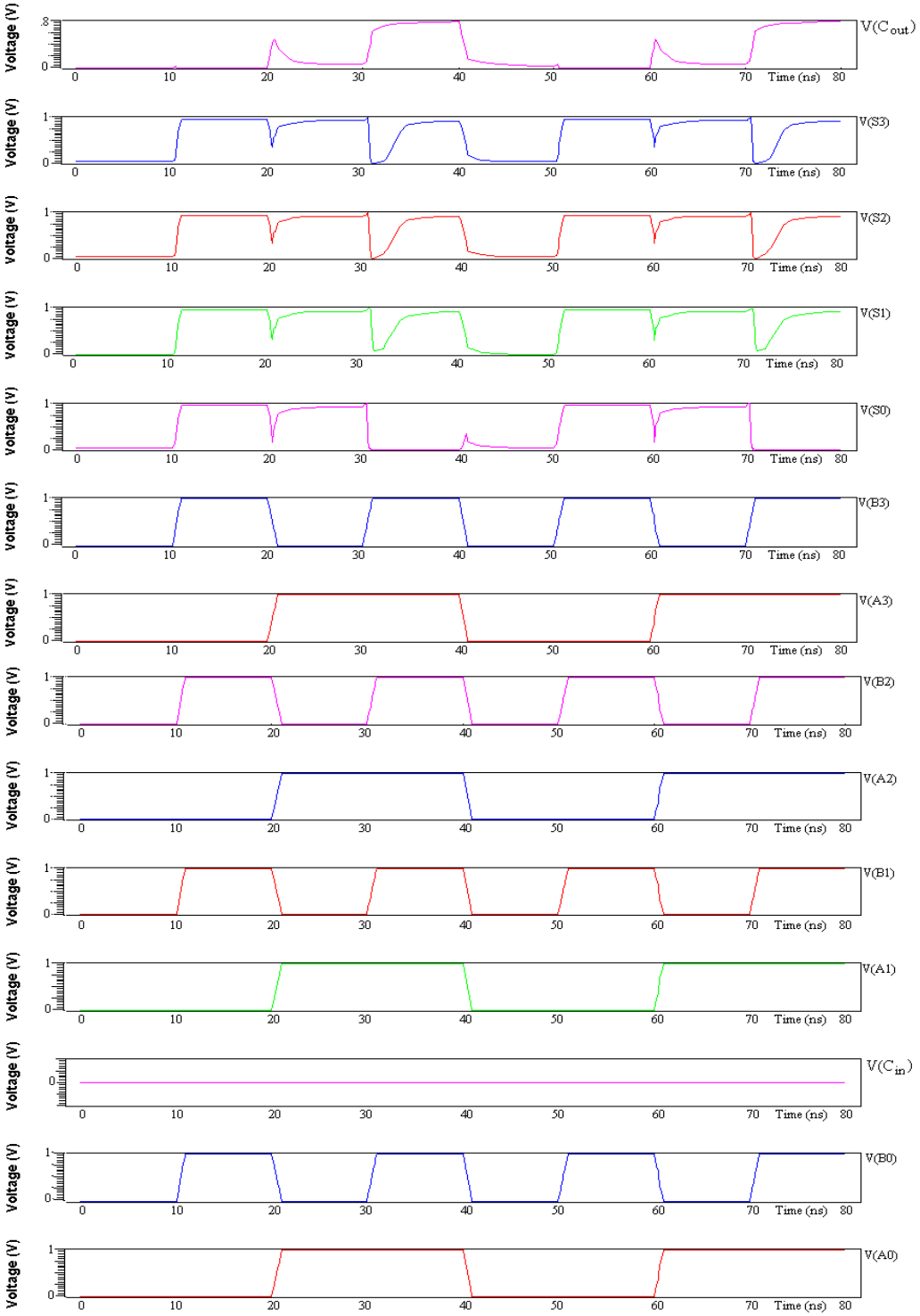

Fig. 11.14 In-Out Waveform of RCA using proposed PTL 9T full adder based on carry logic

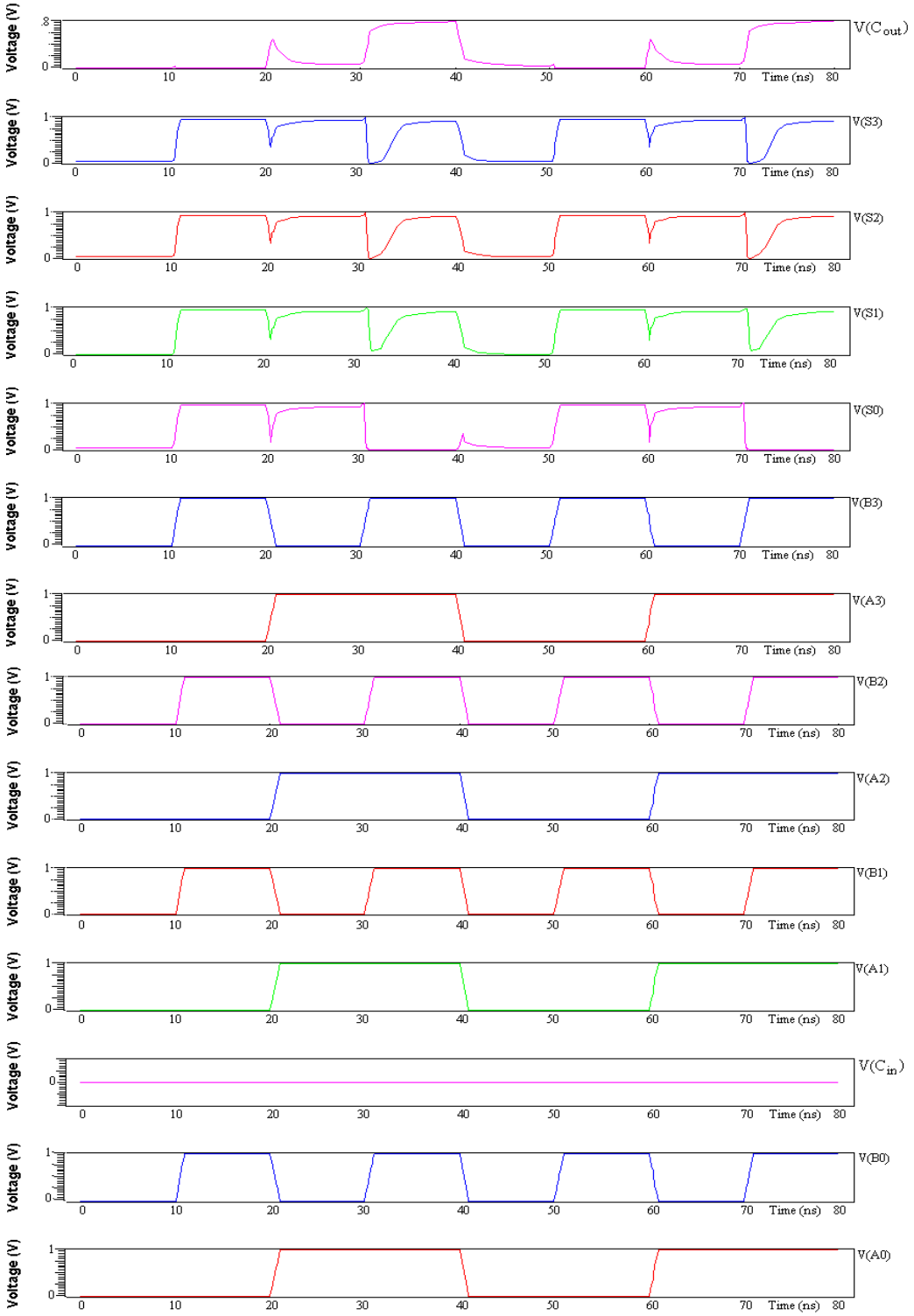

Fig. 11.15 In-Out Waveform of RCA using proposed PTL 9T full adder based on inversion logic

CHAPTER 12

SIMULATION & COMPARISON OF 4-BIT RCA

To evaluate the driving capability of proposed adder cells in complex designs, simulations on RCA have also been done. The curves in Fig.12.1 – Fig.12.4 show the performance comparison for 4-bit Ripple Carry Adders. The simulated data for power consumption and delay based on which these curve for PDP are plotted is given in Appendix-F and Appendix-G.

12.1 SCHEMATIC SIMULATIONS OF 4-BIT RCA USING EXISTING AND PROPOSED 8T FULL ADDER CELLS:

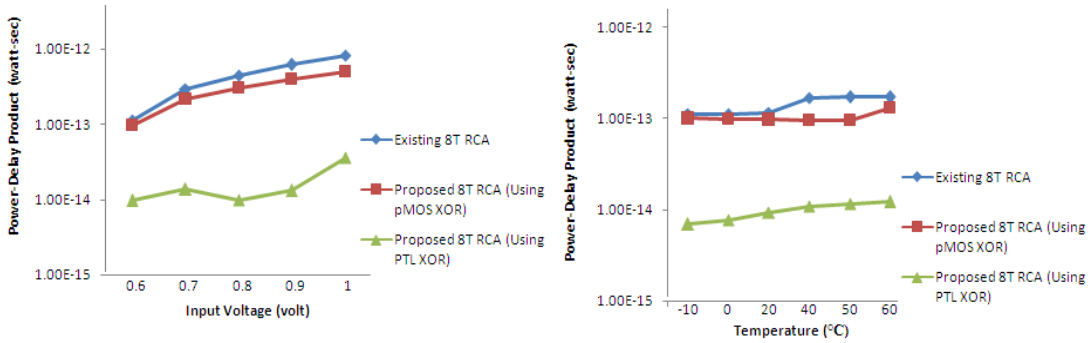

Fig. 12.1 PDP with increasing Input Voltages & Temperature

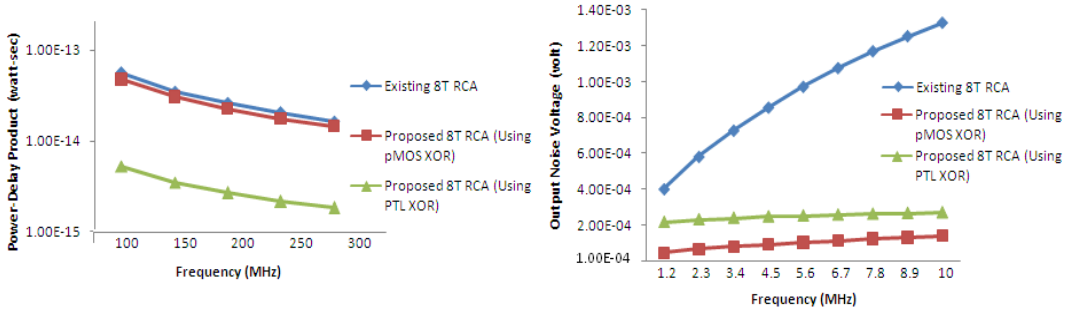

Fig. 12.2 PDP & Output Noise Voltage with increasing Frequency

Fig. 12.1 reveals that the PDP of RCA implemented using proposed pMOS 8T adder is slightly less than the existing one but the 4-bit RCA using PTL based 8T adder has still ensuring its consistent performance as in the 1-bit adder cell.. Fig. 12.2 shows the superiority of RCA using proposed 8T full adder cells for varying frequency and output noise voltage.

12.2 SCHEMATIC SIMULATIONS OF 4-BIT RCA USING PROPOSED 9T FULL ADDER CELLS:

The curves in Fig.12.3 – Fig.12.4 shows the performance comparison for 4-bit Ripple Carry Adders using proposed 9T adder cells because of its better performance than the existing 9T adder cells. The simulated data based on which these curves are drawn is given in Appendix-G.

$9P_{I\ (pMOS)}$ RCA: 4-bit RCA implemented using 9T full adder based on inversion logic designed by pMOS XOR cell

$9P_{I\ (PTL)}$ RCA: 4-bit RCA implemented using 9T full adder based on inversion logic designed by PTL XOR cell

$9P_{C\ (pMOS)}$ RCA: 4-bit RCA implemented using 9T full adder based on carry logic designed by pMOS XOR cell

$9P_{C\ (PTL)}$ RCA: 4-bit RCA implemented using 9T full adder based on carry logic designed by PTL XOR cell

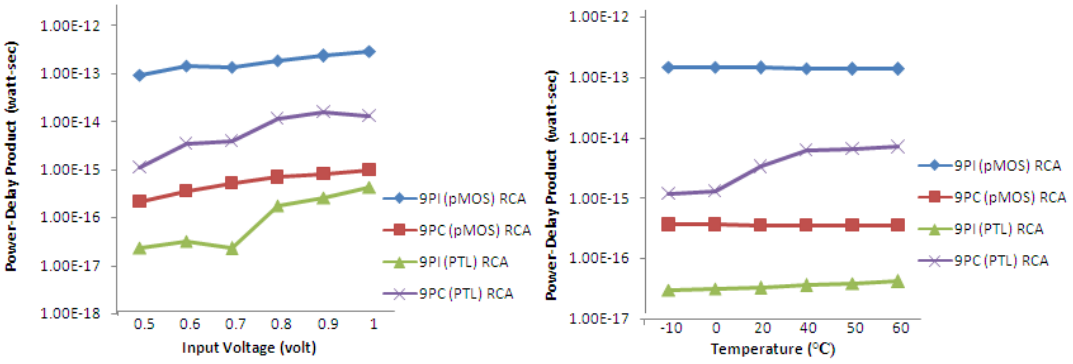

Fig. 12.3 PDP with increasing Input Voltages & Temperature

Fig. 12.3 and Fig. 12.4 depicts that the 4-bit RCA implemented using 9T full adder based on inversion logic is performing best among all others. It is operating better and showing improved results in terms of PDP at different input voltages and frequencies and thus has better temperature sustainability in comparison to others. But the noise immunity of 4-bit RCA implemented using 9T full adder based on inversion logic designed by pMOS XOR cell and 4-bit RCA implemented using 9T full adder based on carry logic designed by pMOS XOR cell is better than the other two designs.

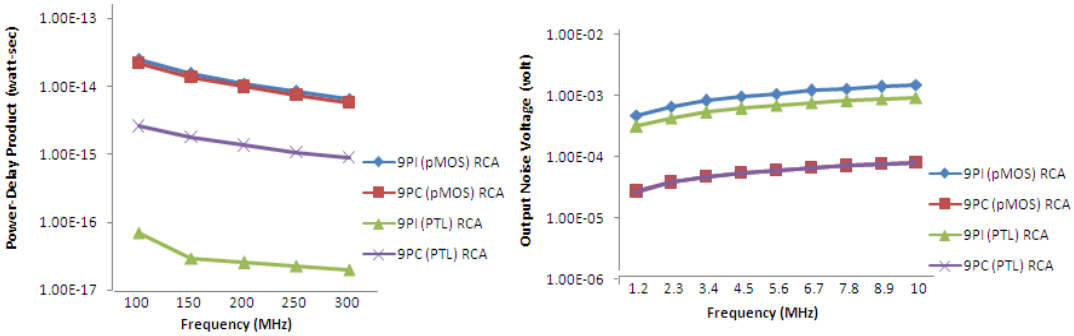

Fig. 12.4 PDP & Output Noise Voltage with increasing Frequency

CHAPTER 13
ARRAY MULTIPLIER

Multiplication is a less common operation than addition, but is still essential for microprocessors, digital signal processors, and graphics engines. The most basic form of multiplication consists of forming the product of two unsigned number. M×N-bit multiplication can be viewed as forming N partial products of M bits each and then summing the appropriately shifted partial products to produce an M+N-bit result P. Binary multiplication is equivalent to a logical AND operation. Therefore, generating partial products consists of a logical ANDing of appropriates bits of the multiplier and multiplicand. Each column of partial products must then be added and, if necessary, any carry values passed to a next column.

13.1 AN ARRAY MULTIPLIER:

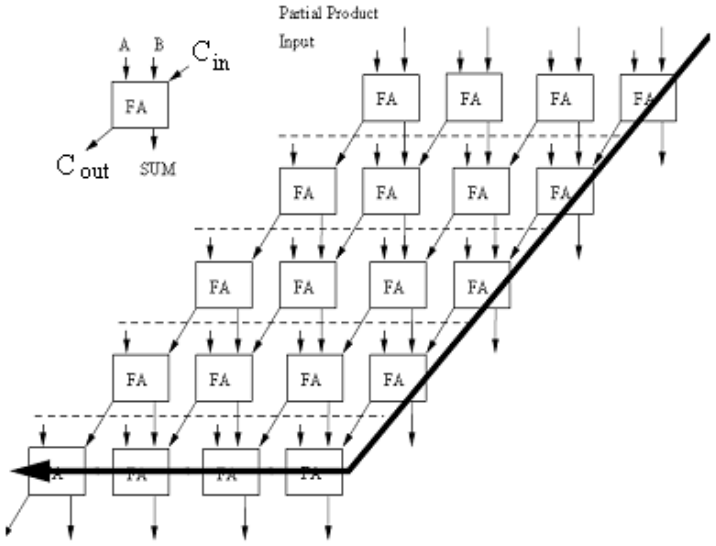

Fig. 13.1 Block Representation of Array Multiplier

Array multiplier is well known due to its regular structure [49]. The array multiplier originates from the multiplication parallelogram. As shown in Fig.13.1, each stage of the parallel adders should receive some partial product inputs. The bold line is the critical path of the multiplier. In array multiplier, all of the partial products are generated at the same time. It is observed that the critical path consists of two parts: vertical and horizontal. Both have the same delay in terms of full adder delays and gate delays. For an M-bit by N-bit array multiplier, the vertical and the horizontal delays are both the same as the delay of an N-bit full adder.

13.1.1 Advantages of Array Multiplier:

- Layout formation is easy.
- Design time of an array multiplier is much less.
- It is easy to design because of its regular structure.
- Array multipliers may be pipelined to decrease clock period at the expense of latency.

13.1.2 Limitation of Array Multiplier:

The limitation of an array multiplier is the increase in size with increasing operands. As operand size increases, linear array grows in size at a rate equal to the square of the operand size. This is because the number of rows in the array is equal to the length of the multiplier, with the width of each row equal to the width of multiplicand. The large size of full arrays typically prohibits their use, except for small operand sizes, or on special purpose math chips where a major portion of the silicon area can be assigned to the multiplier array.

13.2 2×2 ARRAY MULTIPLIER:

Array multiplier has a regular structure that simplifies the wiring and the layout. Therefore, among other multiplier structures, array multiplier takes up the least amount of area. The basic process of binary array multiplication involves the AND operation of multiplicand and multiplier bits and subsequent addition.

N×N array multiplier requires:

No. of full adders: N (N-2)

No. of half adders: N

No. of AND gates: N^2

Therefore from the above formulas the design of 2×2 Array Multiplier requires 04 AND gates and 02 half adders [49]. Full adders are not needed. These results are justified below explaining the logic and block diagram for the same.

A (A1, A0) and B (B1, B0) are 2-bit inputs given to multiplier results in output M (M3, M2, M1, M0). Fig. 13.2 shows the logic of 2×2 Array Multiplier.

```
                    A1 A0
                  × B1 B0
          A1B0   A0B0
     A1B1  A0B1
M3   M2    M1     M0
```

Fig.13.2 Multiplication Logic of 2×2 Array Multiplier

The output of 2×2 array multiplier can be obtained using following equations:

M0=A0.B0

M1=A0.B1+A1B0

M2=A1B1+Carry from M1

M3=Carry from M2

Fig.13.3 shows the basic architecture representing 2×2 array multiplier. The main building blocks for an array multiplier are AND gate and XOR gate.

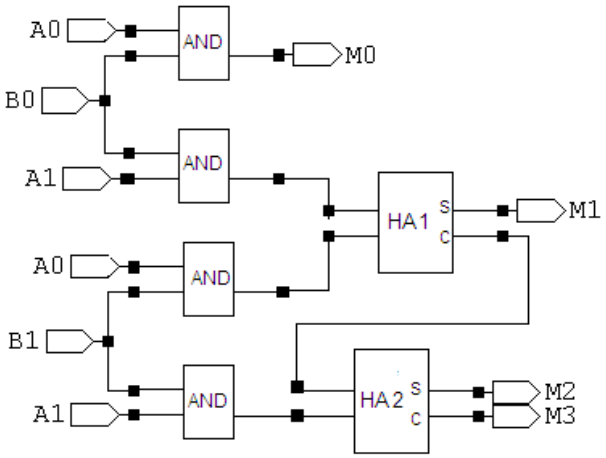

Fig.13.3 Block Diagram of 2x2 Array Multiplier

The 2×2 array multiplier consists of:

- Four 2-input AND gates: The AND gate used in design of array multiplier is implemented using conventional CMOS technology in order to provide full voltage swing at the first stage of multiplier design. The schematic of AND gate [1] - [3] is shown below in Fig. 13.4.

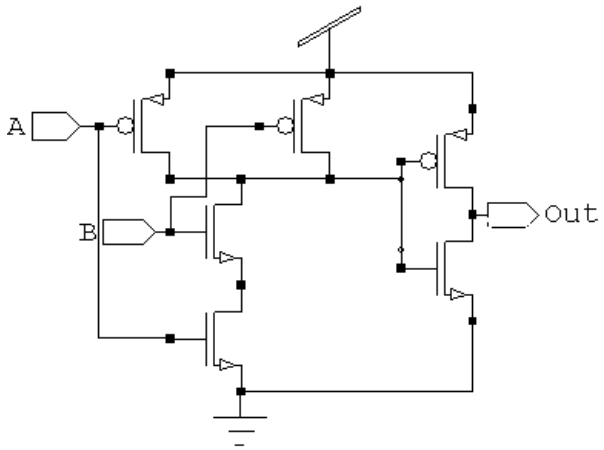

Fig. 13.4 Schematic of CMOS AND gate

- Two half adders: The half adders used in the circuit are implemented by existing and proposed 3T XOR gates as shown in Fig. 5.1, Fig. 6.1 and Fig. 7.1 respectively and the AND gate shown in Fig. 13.4.

The existing and proposed 3T XOR cells have been used to implement 2×2 array multiplier to significantly reduce the power consumption and the chip area of array multipliers, without sacrificing performance. The approach behind this is to use low power, minimal transistor count XOR gates that are the most power consuming blocks in the design of multiplier.

The schematic of 2x2 array multiplier using existing [50] and proposed 8T full adder cells are shown in Fig. 13.5 – Fig. 13.7. The In-Out waveform for the same is shown in Fig. 13.8 –Fig. 13.10.

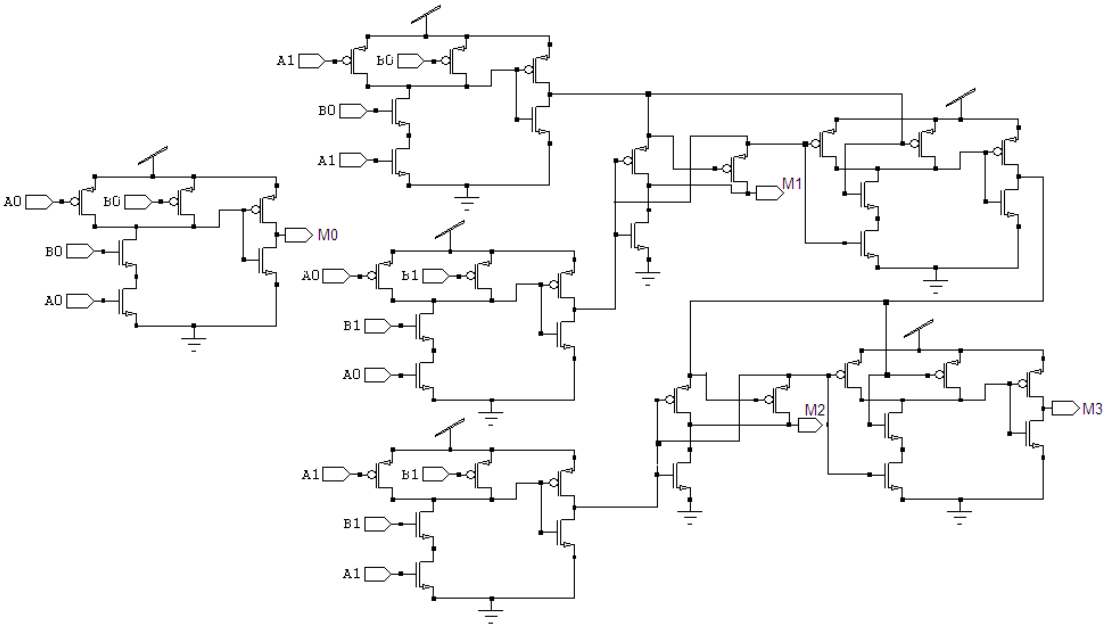

Fig. 13.5 Schematic of 2×2 array multiplier using existing 3T XOR gate [50]

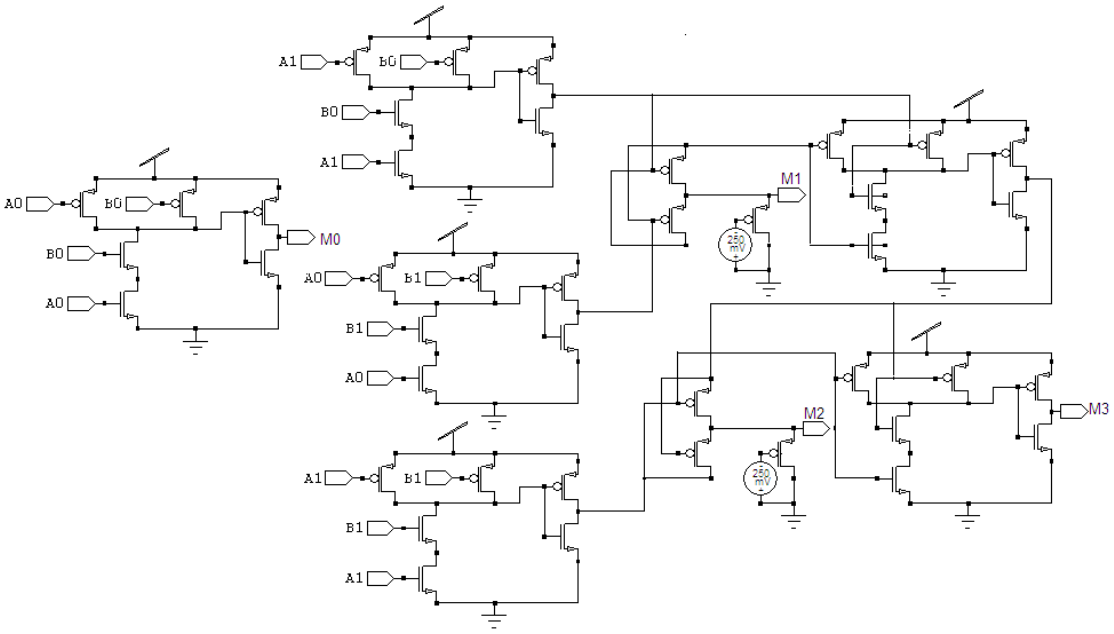

Fig. 13.6 Schematic of 2×2 array multiplier using proposed pMOS 3T XOR gate [51]

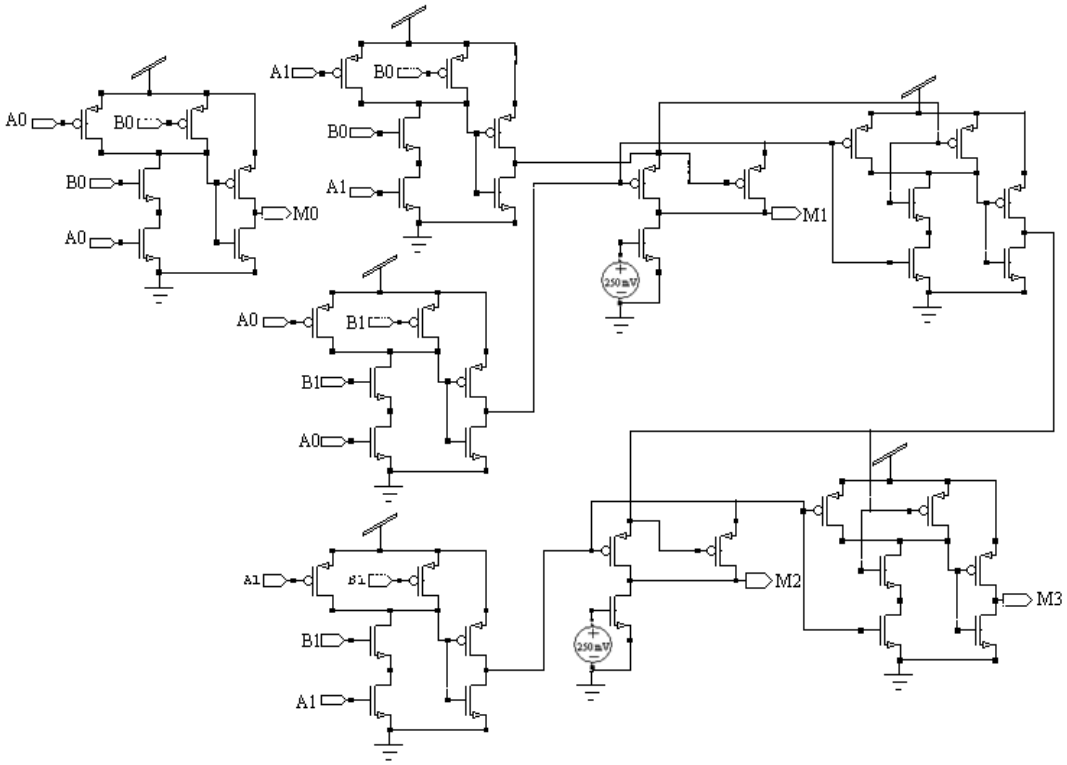

Fig. 13.7 Schematic of 2×2 array multiplier using proposed PTL 3T XOR gate

The 2×2 array multiplier designed using existing XOR gate gives threshold loss for certain input combinations as shown in Fig. 13.8 and thus the output may mistakenly interpreted as logic low which is not desirable for high performance systems.

On the other hand, the 2×2 array multiplier designed using proposed XOR gates overcome this threshold loss and thus maintain their performance in the circuit. Hence these proposed cells act as better alternative to be used in power efficient complex systems with optimum performance.

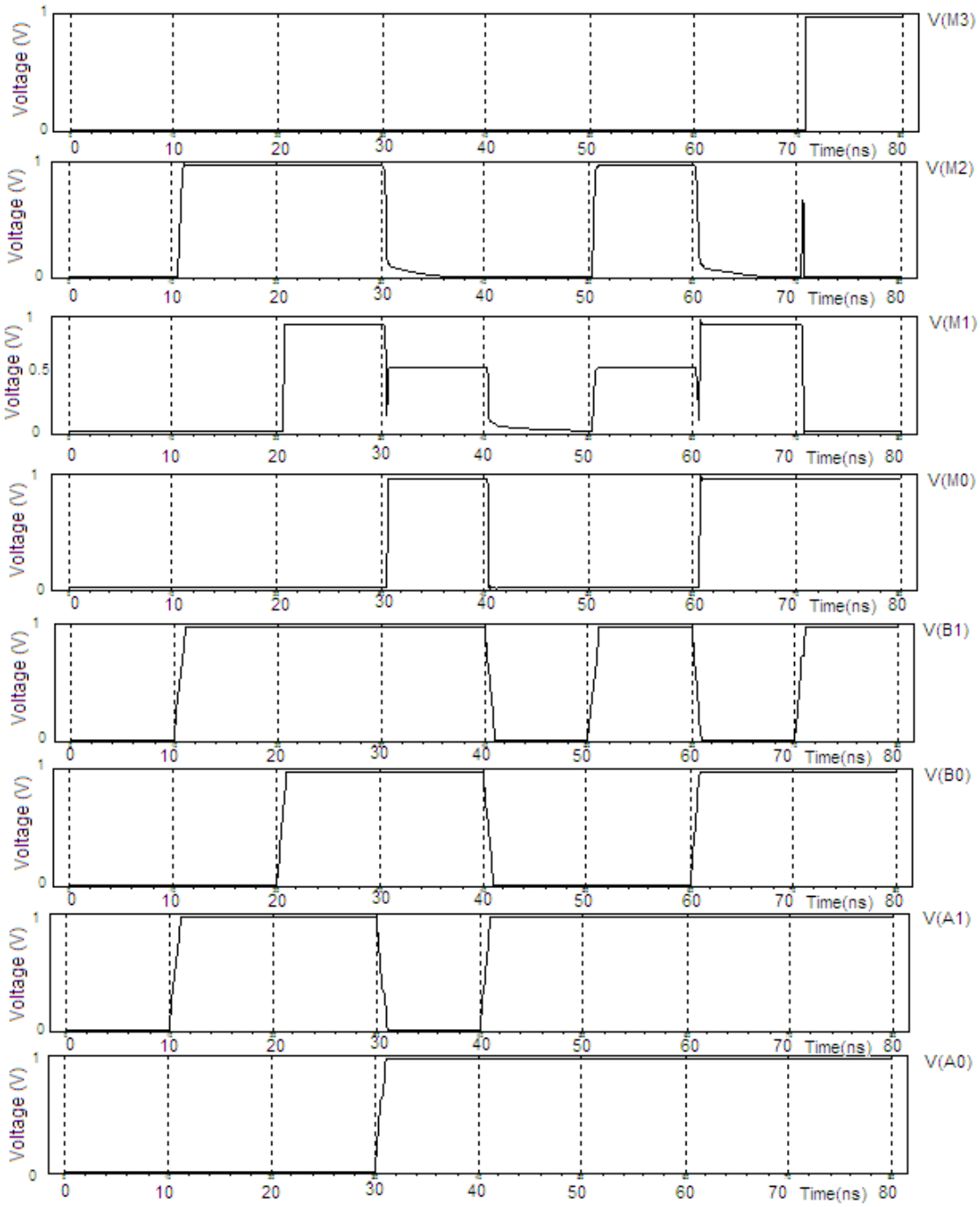

Fig. 13.8 In-Out waveform of 2×2 array multiplier using existing 3T XOR gate

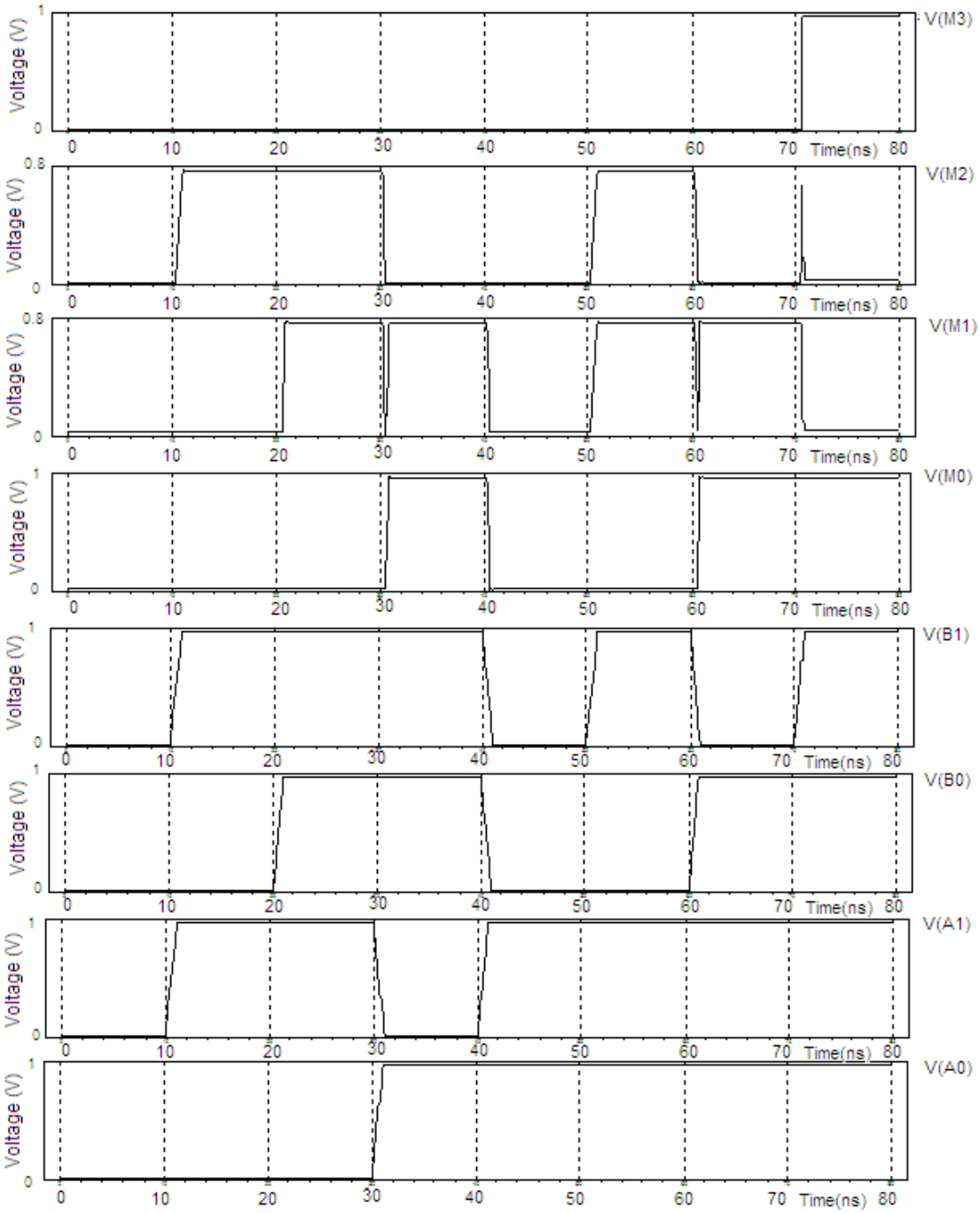

Fig. 13.9 In-Out waveform of 2×2 array multiplier using proposed pMOS 3T XOR gate

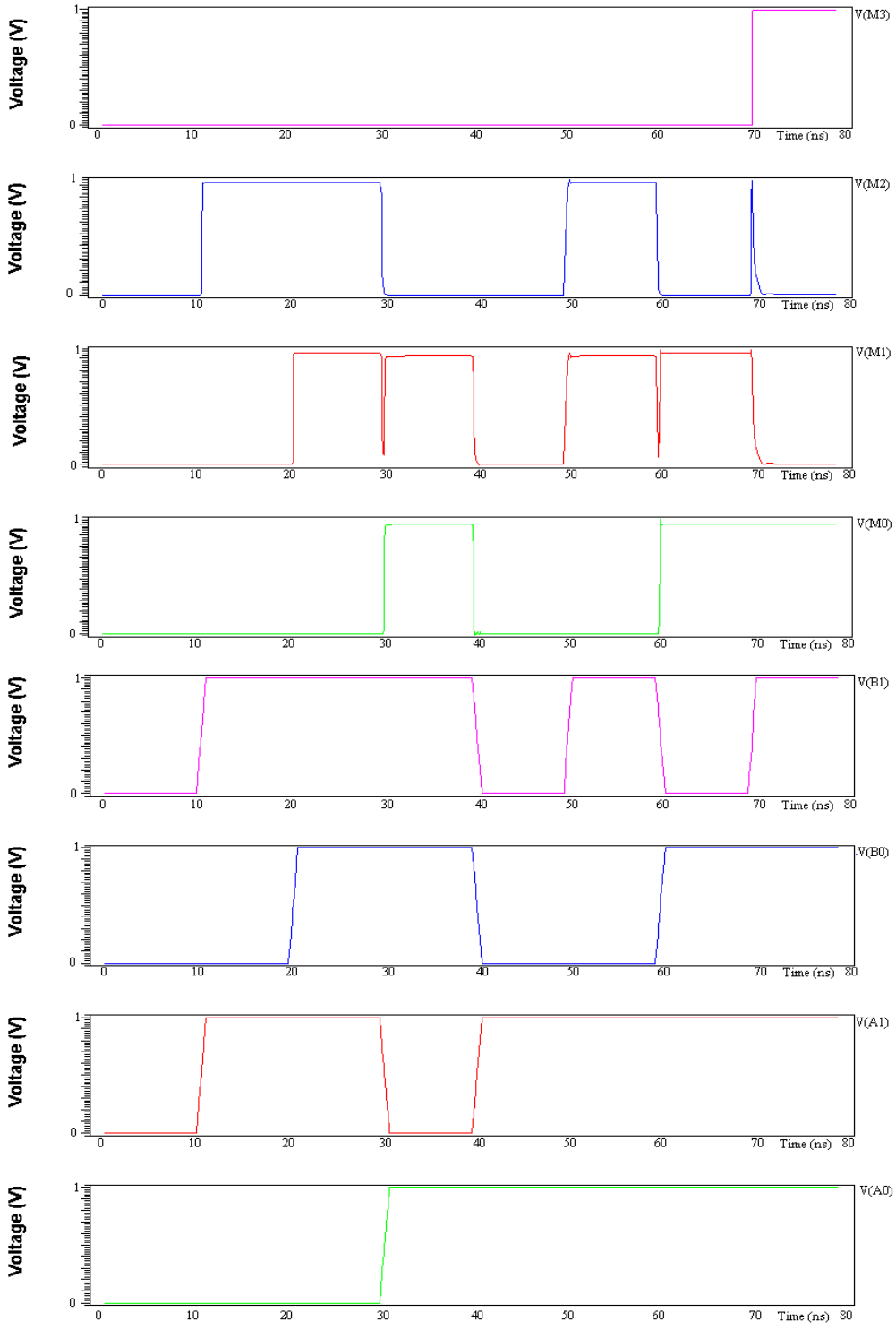

Fig. 13.10 In-Out waveform of 2×2 array multiplier using proposed PTL 3T XOR gate

CHAPTER 14

SIMULATION & COMPARISON OF ARRAY MULTIPLIER

14.1 PERFORMANCE COMPARISON OF 2×2 ARRAY MULTIPLIER USING EXISTING AND PROPOSED 3T XOR GATE:

After designing 4-bit RCA, performance of XOR cell has been analyzed with 2×2 array multiplier. Fig. 14.1 and Fig. 14.2 show the schematic comparison of existing and proposed multiplier designs. The simulated data based on which these curves are drawn is given in Appendix-H.

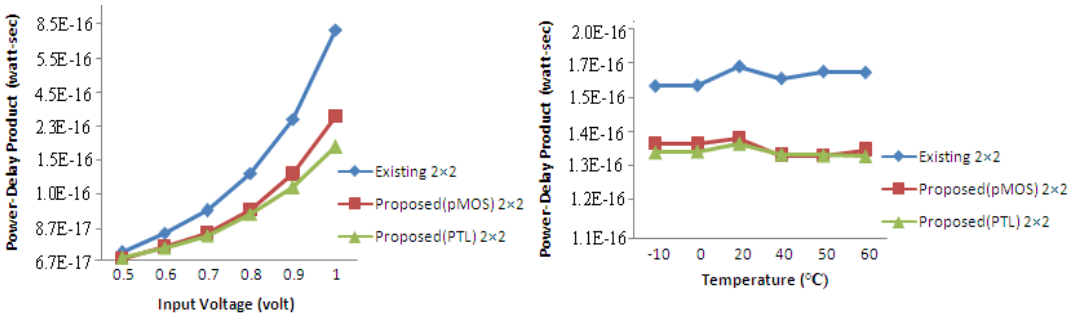

Fig. 14.1 PDP with increasing Input Voltage & Temperature

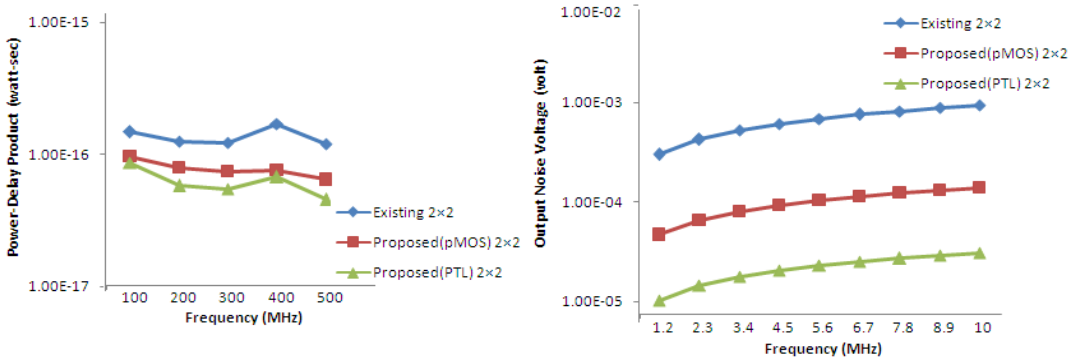

Fig. 14.2 PDP & Output Noise Voltage with increasing Frequency

Fig. 14.1 depicts 20% to 30% improvement in PDP of proposed designs with increasing input voltage as well as with varying temperature. Improvement of nearly 30% to 35% is obtained with varying frequency and also the proposed multipliers posses' better noise immunity than existing one as can be concluded from Fig. 14.2.