*Research Paper* *IJMSRR*
*Impact Factor 0.348* *E- ISSN - 2349-6746*
*ISSN* -2349-6738

# POWER MINIMIZATION IN CARRY SELECT ADDER USING DUAL TRANSITION SKEWED LOGIC

**A.Pameela**
*Research Scholar, Bharath University, Thambaram, Chennai, India.*

***Abstract***
*In this paper, i present a low power and high performance Carry Select Adder using Dual Transition Skewed Logic which is used for high noise immunity. I compared DTSL Carry Select Adder with domino and static CMOS adders with respect to performance, power consumption and area, using UMC 0.18µm technology. The comparison shows the superior properties of the DTSL over domino and the static CMOS logic: 19% to 27% improvement in power dissipation over domino with similar performance is observed.*

***Keywords: CSA, DTSL.***

## 1. INTRODUCTION

The demand for high performance and low power consumption has become important in today's VLSI circuit design. The need for higher performance has led to the use of Domino circuits where conventional static CMOS circuits may not meet the demand for low critical path delay. However, Domino circuits are more susceptible to noise (for scaled technologies with low transistor threshold voltage) than static CMOS circuits because in the evaluation mode intermediate nodes of Domino circuits may be floating. Another drawback of Domino is its higher power consumption compared to standard complementary CMOS logic. Since a clock signal is necessary for every stage to pre-charge the output nodes of Domino circuits, power consumption due to clock is of concern. One of the solutions to these problems is to use skewed logic circuits, which have good noise-immunity and achieve high performance with low power consumption [1]. The circuit topology of skewed logic is the same as that of static CMOS logic, however, the PMOS or the NMOS transistors are preferentially sized to achieve fast high - to - low or low - to – high transitions. For example, to speed up high to low transition, the sizes of PMOS transistors are reduced while the NMOS transistors are sized up (Fig. 1).However, it should be noted that skewed CMOS logic gates may require the clock [2]. In this paper i propose Dual Transition Skewed Logic (DTSL) which consists of dual data paths (one for fast rising transition and the other for fast falling transition) using skewed circuits. Since DTSL uses skewed static CMOS logic style, it has good noise immunity and low power dissipation while achieving the performance of Domino. In this paper, i implement a Carry Select Adder (CSA) using DTSL, and compared the performance and power consumption of DTSL with Domino and Static CMOS logic. The rest of the paper is organized as follows. Section 2 presents a comparison between Domino and skewed style logic circuits. In section 3 i show the implementation of a CSA block using DTSL. Section 4 shows the simulation results.

## 2. CIRCUIT STYLES

The circuit styles that i will consider in this paper are static CMOS, Domino, skewed CMOS, and DTSL. Fig. 1 shows some examples of how to adjust the sizes of each circuit style, where Wp and Wn represent the optimal sizes of PMOS/NMOS of an inverter. Fig. 1(a) shows a 2 input NAND gate, whose optimal channel widths are Wp and 2Wn. Fig. 1(b) presents a 2 input Domino AND with a keeper transistor. Domino circuits are suitable for high performance because of the reduced junction capacitance at the output node due to the reduction in the number of PMOS transistors. The dynamic node of Domino circuits can be floating during evaluation, which may cause noise problems in wide input OR gates (especially with low transistor threshold voltage). In order to prevent this, a larger keeper transistor may be required which can have an impact on performance. Hence, the size of the keeper transistor should be selected carefully to optimize performance [3, 4].

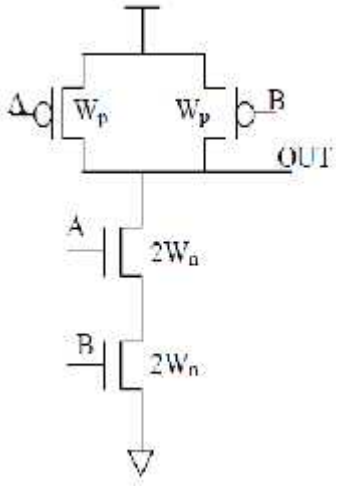


Fig. 1(a) Static CMOS

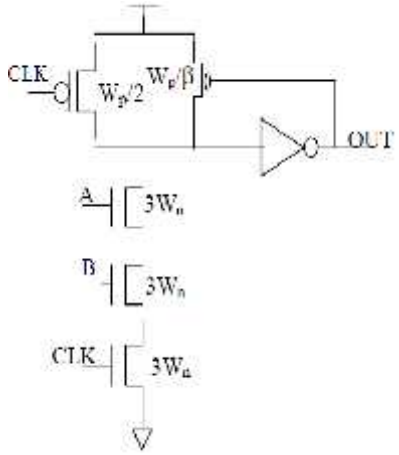


Fig. 1 (b) Domino circuit

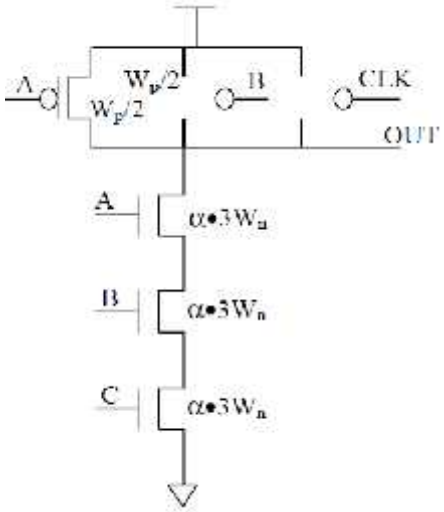


Fig. 1(c) Skewed CMOS with clock signal

*IJMSRR*
*E- ISSN - 2349-6746*
*ISSN -2349-6738*

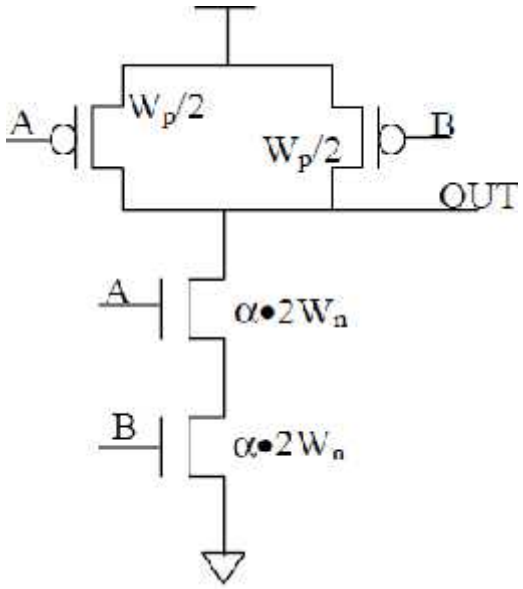


Fig.1 (d) Skewed CMOS without clock signal

The circuit topology of a skewed logic is the same as that of the conventional static CMOS logic, however, the sizes of PMOS and NMOS transistors are decided based on the preferred transition. Fig. 1 (c) and (d) show two different skewed 2 input NAND gates, in which transistors are sized to make fast high - to – low transitions. To speed up high to low transition we increase the driving capability of a pull down network by increasing the sizes of the NMOS transistors and reduce the capacitance of the output and input nodes by minimizing the sizes of the PMOS transistors. In Fig. 1 (c), to pre-charge the output node, an additional PMOS transistor having CLK signal as gate input is used. However, in Fig. 1 (d), we do not use CLK signal. We use this circuit style for implementing DTSL.

Let us define (α) as

$$\alpha = \frac{\text{Width of transistor used in skewed circuits}}{\text{Optimal width of static CMOS}}$$

We can control the skew ratio (α) to find the optimal point between performance and power consumption. By increasing the skew ratio (α) we can get higher performance.

Skewed circuits have performance comparable to Domino circuits [1]. However, skewed circuits require pre-charging (or selectively pre-charging) to each gate just like Domino circuit [2, 7]. However, this increases the clock load and hence, the power consumption (Fig. 2 (a)). The other solution is to use DTSL (Dual Transition Skewed Logic) as proposed in this paper, which does not require clock signal. Fig 2 (b) shows an example of DTSL that achieves high performance by duplicating signal paths: one signal path is for fast rising transition while other for fast falling transition. If the input of the first stage of the logic block toggles from high to low, faster data transition takes place through the top data path. On the other hand, if the input toggles from low to high, the data transits faster through the bottom path than through top path. The arrows represent the skew direction. The combiner detects earliest transition, latches it, and then transfers the data to the next stage. I implement a Carry Select Adder (CSA) using DTSL and compare its performance with that implemented using Domino and static CMOS logic style.

*IJMSRR*
*E- ISSN - 2349-6746*
*ISSN -2349-6738*

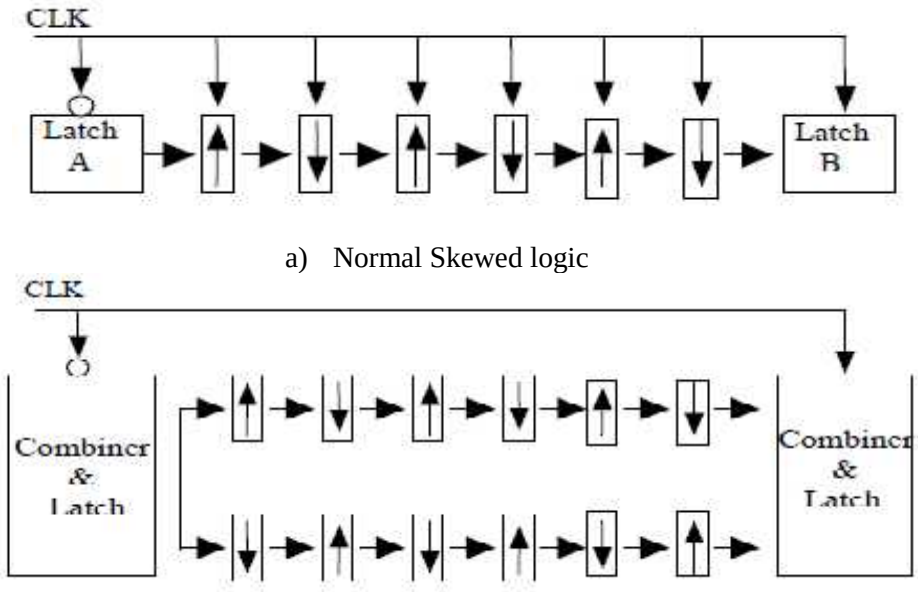


Fig. 2) DTSL block structures

**3. IMPLEMENTATION OF CSA USING DTSL**

DTSL has better noise immunity than the domino logic and comparatively high performance as described in the previous section. In general the number of transistors used in DTSL is more than twice the number of the transistors used in Domino logic, however, the total size of the PMOS and NMOS transistor compared to static logic style is not high. In some application this number can be reduced. For example, implementing CSA (Carry Select Adder) with DTSL, we can reduce the total number of transistors considerably.

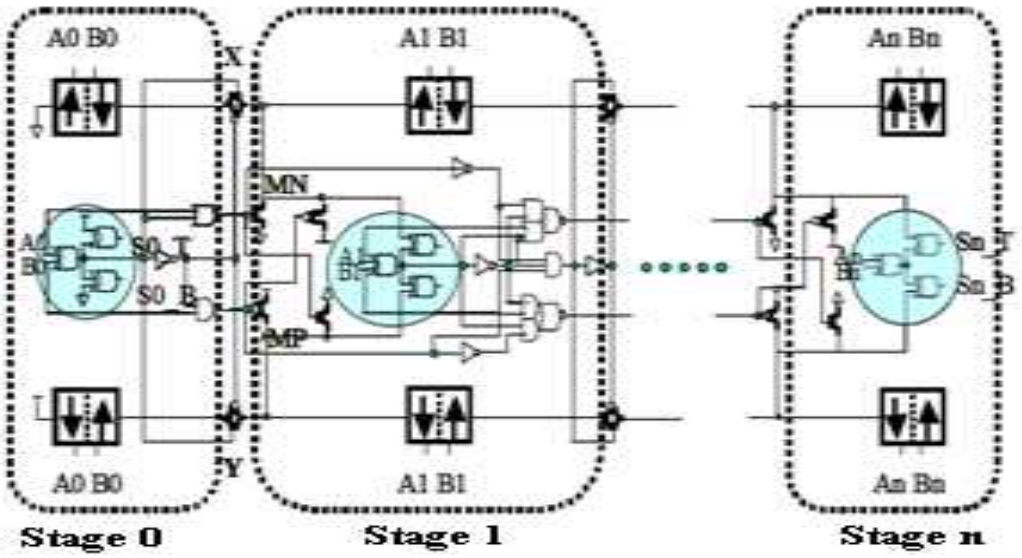


Fig. 3) Block Diagram of Carry Select Adder using DTSL

A general CSA consists of 2 pairs of n-bit wide blocks with each block consisting of a pair of ripple carry adders and two multiplexors controlled by the incoming carry signal: one of which is to select one of the two carry outputs and the other one is to select one of two nbit SUMs [5, 6].

Fig. 3 shows the implementation of the carry propagation logic of CSA using DTSL. It consists of two data paths for carry propagation, logic for generating SUM, and control logic. Control logic consists of transmission gates (X, Y) between each carry propagation circuit on the data path, switching transistors (MN, MP), and some static CMOS gates to control the transmission gates and switching transistors.

Fig. 4 shows the implementation of one stage of the carry propagation logic of CSA using DTSL. The arrows indicate the skew direction. The logic in the circle is for generating SUM.

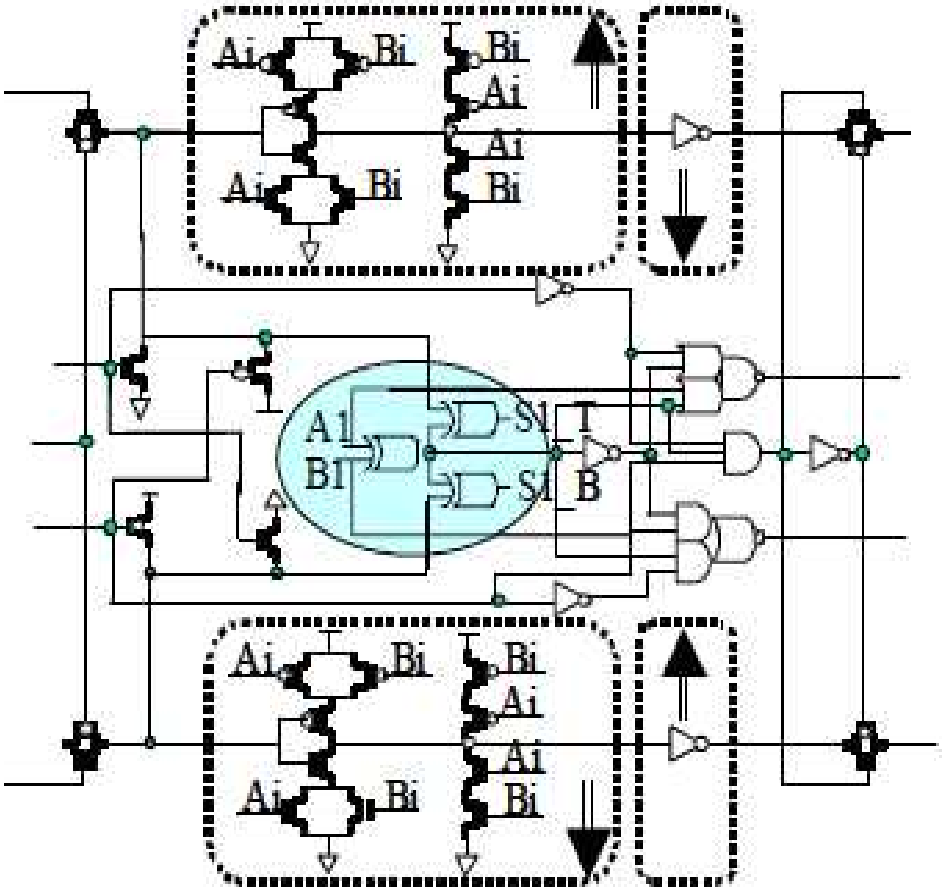


Fig. 4) Block diagram of one stage of CSA using DTSL

As shown in Fig. 3, the carry propagation logic of each block of CSA has two data paths: one has '0' as its CARRY input to the first stage and the other has '1' as its CARRY input. We can, therefore, improve performance by properly skewing CMOS logic in the upper and lower logic blocks.

The skew direction (high → low) on the top data path should be opposite to that (low → high) on the bottom. It can be observed that there is no problem in the operation of the skewed circuits when inputs Ai's are different from Bi's (Ai≠Bi, for i = 0 to n) because the direction of CARRY transition is always the same as the skew direction.

However, if any Ai is equal to Bi at Stage i (i = 0 to n), the CARRY outputs on both paths at that stage will be the same.

This means that the skewed circuits on one data path at that stage are evaluated in the opposite direction of skew. This may introduce larger propagation delay. To mitigate such a problem, we have used control logic as follows: First, if the data inputs are different (Ai < Bi, for i = 0 to n) for every stage, then transmission gates X, Y will turn on and the switching transistors (MN, MP) will be disabled. Carry-out will be the same as Carry-in for every stage because each carry propagation logic propagates Carry-in to Carry-out without change.

Hence, every skewed circuit will be evaluated in the preferential skew direction. This has the largest propagation delay because Carry-in goes through all stages in the data path. However, if Ai is equal to Bi at any stage, we have to switch Carry-in of the next stage ($[i+1]^{th}$) to low or high depending on the value of Ai and Bi.

For example, let us assume A1=B1=0 at Stage 1, then the Carry-out at that stage will be low regardless of Carry-in of Stage 1. Hence, we do not need to wait for the Carry-in to propagate to the output node of Stage 1, i.e. when inputs A1, B1 of Stage 1 are set, we can switch Carry-in of the next stage (Stage 2) immediately to low after turning off the transmission gates on data path. Similarly, if A1=B1=1 at Stage 1, then we change Carry-in of the next stage (Stage 2) to high.

For such cases, the total propagation delay will be shorter than the total delay of the previous case (Ai < Bi, for i = 0 to n) because the time taken to switch Carry-in of the next stage (Stage 2) is shorter than the time in which Carry-in of the first stage (Stage 0) propagates to the Carry-in node of the Stage 2 having A2, B2 as inputs.

In the case of A0=B0, the time taken to switch Carryin node of the next stage (Stage 1) may be longer than the carry propagation delay of the node, so we have to increase the drive of the XOR gate at the Stage 1 to minimize the switching time. Finally, let us consider the case when Ai = Bi and Aj < Bj (j=i+1). In this case, even though Aj is not the same as Bj, we have to switch Carry-in of $(j+1)^{th}$ stage on the data path because the skew direction of one data path at Stage (j+i) is opposite Carry-in of $(j+1)^{th}$ stage.

## 4. RESULTS

To compare the propagation delay and the power consumption of DTSL with domino logic and static CMOS logic, we implemented blocks of CSA having n stage ripple carry adders. I use UMC 0.18µm CMOS technology with Vdd=1.5 volts. Table 1 shows simulation results of CSA for each logic style and the comparison of DTSL with Domino and Static CMOS logic.

The simulation results shown in Table 1 are the values with inputs Ai > Bi (i = 0 to 6) and A0 = B0 & Ai < Bi (i = 1 to 6). Under these conditions the propagation delay is largest for CSA implemented using DTSL.

Table 1 show that the performance of DTSL is comparable to Domino and is better than standard CMOS logic. DTSL consumes less power than the Domino. Comparison shows that DTSL has 19% to 27% power improvements with performance comparable to Domino. However, the number of transistor used in implementing the CSA using DTSL increases by 37% over static CMOS logic.

## 5. CONCLUSIONS

In this paper, I proposed Dual Transition Skewed Logic for robust high performance low power circuit design. I have implemented a Carry Select Adder using DTSL and compared its performance with Domino and static CMOS logic. I have shown that the performance of CSA using DTSL is comparable with CSA implemented with Domino while being more power efficient.

**Table 1. CSA Simulation and Comparison Results**

| Parameter | | Domino Logic | Static CMOS Logic | DTSL | Domino-DTSL | Static-DTSL |
|---|---|---|---|---|---|---|
| | | | | | Domino | Static |
| Ai>Bi (I=0≈6) | Delay(ns) | 0.520 | 0.750 | 0.518 | -0.4% | -31% |
| | Power(mW) | 1.58 | 0.81 | 1.15 | 27% | -42% |
| A0=B0, Ai<Bi (I=0≈6) | Delay(ns) | 0.520 | 0.750 | 0.537 | 3% | -28% |
| | Power(mW) | 2.01 | 0.935 | 1.628 | 19% | -74% |
| Number of Transistors | | 280 | 392 | 536 | 91% | 37% |

## 6. REFERENCES

1. D. Somasekhar, "Power and dynamic noise considerations in high performance CMOS VLSI design", PhD Thesis, Purdue University, August 1999.
2. A. Solomatmikov, D. Somasekhar, K. Roy, "Skewed CMOS: Noise-immune high-performance low-power static circuits family", ESSCIRC, 2000.
3. D.Somasekhar, S.H.Choi and K.Roy, "Dynamic Noise Immunity in Precharge-Evaluate Circuits", DAC, 2000.
4. R. Krambeck et al., "High–Speed Compact Circuits with CMOS," IEEE Journal of Solid State Circuit,
5. vol. SC-17, no 3, pp. 614-619, June 1982.
6. O. J. Bedrij, "Carry-Select Adder," IRE Transaction on Electronic Computers, Vol. EC-11, pp.340-346, 1962.
7. N. Weste, K. Eshraghian, "Principles of CMOS VLSI Design", Addison Wesley Publishers, 1994.
8. N. Sirisantana, A. Caoa, S. Davidson, C. Koh, and K. Roy, "Selectively Clocked Skewed Logic (SCSL): A Robust Low Power Logic Style for High Performance Applications," ACM/IEEE International Symposium on Low Power Design, Aug. 2001, to appear.