Design of 8-bit Ripple Carry Adder Using Constant Delay Logic

K. Lavanya, V. Surendra Babu
E.C.E Department, Sir C.R. Reddy College of Engg., Andhra University
Email: lavanyasudeep@gmail.com, vsb.mtech14@gmail.com

Abstract: This paper new design a Proposed Logic design targeting at full-custom high speed applications. The constant delay characteristic of this logic style regardless of the logic expression makes it suitable in implementing complicated logic expression such as addition. This feature enables performance advantage over static and dynamic, CD logic styles in a single cycle, multi-stage circuit block. Several design considerations including appropriate timing window width adjustment to reduce power consumption and maintain sufficient noise margin to ensure robust operations are discussed and analyzed. Using 50nm general purpose CMOS technology, the proposed logic style high performance compare to Static, dynamic & CD logic respectively. Post layout simulation results of 8-bit ripple carry adders conclude that Proposed logic faster than the compare to CD-based design Dynamic static based adders respectively. For ultra-high speed applications. This circuits design DSCH Tool, Microwind Tool.

Index Terms—Domino logic, Dynamic CMOS logic, Feedthrough logic (FTL), Low power ripple carry adder (RCA)

I. INTRODUCTION:

High performance, energy- efficient logic style has always been a popular research topic in the field of very large scale integrated (VLSI) circuits because of the continuous demand of ever increasing circuit operating frequency.

The invention of the dynamic logic in 80s is one of the answers to this request as it allows designers to implement high performance circuit block, i.e., arithmetic logic unit (ALU), at an operating frequency that traditional static and pass transistor CMOS logic styles are difficult to achieve. However, the performance enhancement comes with several costs, including reduced noise margin, charge-sharing noise, and higher power dissipation due to higher data activity.

Logic Implementations: Circuit Families

A. Static Logic:

Static logic is the most widely used logic style in CMOS technology and its basic structure as shown in Figure 1. It consists of a NMOS pull-down network (PDN) and a PMOS pull-up network (PUN). The primary advantages of static logic are robustness, low power dissipation especially at low data activity factor, and adequate performance with no static power dissipation. Its most distinct characteristic is that at any given time, the gate output is connected to either VDD or GND via a low-resistance path.

While this unique feature ensures static logic's robustness, it is also a major drawback since static CMOS requires both NMOS and PMOS transistors on each input. During a falling output transition, PMOS transistors do not contribute to the pull-down transition current but only add significant capacitance. Hence, static CMOS has a relatively large logical effort and area penalty and is slow when implementing complicated logic expression such as 4-input XOR.

The PDN and PUN are implemented using NMOS and PMOS devices because they can pass strong logic “0” and “1” respectively. PMOS devices are typically sized up two times larger than NMOS devices to provide equal rise and fall delay due to lower hole mobility. Therefore, PMOS transistors have to be up sized four times larger than NMOS transistors to achieve equivalent rise and fall delay for the two-input NOR gate. The up-sized PMOS transistors contribute input capacitance for both transitions, while only helping the rise delay. In this regard, PMOS devices become the area bottleneck for static CMOS logic style when implementing NOR gate (PMOS devices in series). Furthermore, the up-sizing technique provides diminished rising delay improvement due to self-loading effect, since the additional drain capacitance introduced by up-sizing gradually offsets the performance enhancement contributed by higher pull-up current as a result of larger width device.
B. Dynamic & Compound Domino Logic: The invention of the dynamic logic in the 80s is one of the answers to the request of ever increasing IC operating speed as it allows designers to implement high performance circuit block, i.e., arithmetic logic unit (ALU), at an operating frequency that the traditional static and pass transistor CMOS logic styles are difficult to achieve.

A generalized schematic of a dynamic gate with footer CLK transistor is shown in Figure.

![Figure 1.1. Schematic of a dynamic gate with footer transistor](image)

The operation of dynamic logic is as follows: When CLK is low (precharge period), transistor M1 is on, and NMOS PDN is off because M2 is off. X is charged to VDD by transistor M1 and Out is maintained at GND.

Dynamic logic enters evaluation period when CLK rises to high.

In this case, depending on the input patterns two possible scenarios can take place. If NMOS PDN is off, X will be floated because both M1 and PDN are off. Therefore, a small PMOS keeper (M3) is required to fight against the leakage and to help maintaining the voltage of node X at VDD. On the other hand, if NMOS PDN is on, then X is quickly discharged to GND and Out is charged up to VDD via the inverter. Dynamic logic does not have the problem of static power dissipation because when X is at GND (Out is at VDD), PMOS keeper M3 is guaranteed to be off. When Out is discharged, it cannot be charged again until the next precharge period begins. Thus the inputs to the gate of NMOS PDN can make at most one transition during evaluation.

Summary, the unique characteristics of dynamic logic are:

1. The logic function is implemented with NMOS transistors only.
2. The number of transistors for complicated logic expression implemented with dynamic logic is substantially lower than the static case.
3. Dynamic logic has faster switching speed because less number of transistors (especially without any PMOS logic transistors) contributes less load capacitance.
4. It only consumes dynamic power since no static current path ever exists between VDD and GND. However, the overall power consumption can be significantly higher than the static design because of the higher switching activity.

The performance enhancement comes with several costs however, including reduced noise margin, charge-sharing noise, and higher power dissipation due to higher data activity. In a traditional dynamic logic, an output inverter is required between dynamic logics to satisfy the data monotonicity requirement and to ensure proper logic evaluation. This not only increases the overall delay but also the power consumption as well. Two variations of the dynamic logic have been proposed to mitigate this problem. NP domino, or also known as NORA domino, replaces this inverter with pre-discharged dynamic gates using PMOS logic. However, NORA is extremely susceptible to noise and has not been used extensively. Zipper domino attempts to achieve the same objective by a slightly different implementation, but is never widespread in the VLSI industry. Furthermore, dynamic logic has gradually lost its performance advantage over static logic due to the increased self-loading ratio in deep-submicron technology (65nm and below) because of the additional NMOS CLK footer transistor (Figure 1.12). This phenomena has been demonstrated in , which concludes that at processes such as 180nm and 130nm, the optimal adder architecture is radix-4 (5 transistors in series, including the footer transistor); however, radix-2 (3 transistors in series, including the footer transistor) configuration becomes optimal at 65nm technology and beyond because the increased self-loading ratio has made radix-4 architecture slower than radix-2, even though radix-2 configuration requires more number of stages to complete the addition.

Compound domino logic (CDL) where dynamic and static CMOS gates alternating between each other mitigates the two aforementioned problems and has become the most popular logic style in high performance circuit block, i.e., 64-bit adder in modern central processing unit (CPU). In this design, the output inverter is replaced with a more complex inverting static CMOS gates (Figure 1.13), i.e., NAND or NOR, such that the monotonicity requirement is satisfied while conducting complex logic operations without wasting the one inverter delay. Moreover, all the dynamic stages except the first stage can be footless (the footer transistor is eliminated) in CDL, thus reduce the total stack height by one. However, this implementation comes at the expense of increased .

schematic of Dynamic Logic vs. Compound Domino Logic

![Fig 1.3 schematic of Dynamic Logic vs. Compound Domino Logic](image)
respectively, its noise margin is significantly degraded as in a CDL design, the output of the dynamic logic without any buffer is required to drive the next stage via a long interconnect and with other signal wires running in parallel. The crosstalk of the adjacent wire can potentially be the state of the dynamic logic, and results in false logic evaluation. As a result, extra distance among wires running in parallel has to be enforced in laying out such a design at the expense of increased total wire length.

In the extreme case, power rails are placed in between adjacent wires to eliminate the crosstalk problem. This technique nevertheless, causes significant performance degradation and increased power consumption as a result of increased parasitic capacitance. Because of this reliability concern, CDL is regarded as a less robust logic style and is not considered.

II. EVOLUTION OF CD LOGIC

A. FTL Logic

FTL logic [Fig. 1(b)] in CMOS technology was first introduced in a basic operation as follows: when CLK is high, the predischARGE period begins and Out is pulled down to GND through M2. When CLK becomes low, M1 is on, M2 is off, and the gate enters the evaluation period. If inputs (IN) are logic “1,” Out enters the contention mode where M1 and transistors in the nMOS pull-down network (PDN) are conducting current simultaneously. If PDN is off, then the output quickly rises to logic “1.” In this case, FTL’s critical path is always a single pMOS transistor. Despite its performance advantage, FTL suffers from reduced noise margin, excess direct path current, and nonzero nominal low output voltage, which are all caused by the contention between M1 and nMOS PDN during the evaluation period. Furthermore, cascading multiple FTL stages together to perform complicated logic evaluations is not practical. Consider a chain of inverters implemented in FTL cascaded together and driven by the same clock, as shown in Fig. 2.

![Fig. 2. Simulated unwanted glitch at different logic depths in a chain of inverters implemented with FTL.](image)

When CLK is low, M1 of every stage turns on, and the output of every stage begins to rise. This will result in false logic Fig. 2. Simulated unwanted glitch at different logic depths in a chain of inverters implemented with FTL evaluations at even numbered (i.e., 2, 4, 6, etc.) stages since initially there is no contention between M1 and nMOS PDN because all inputs to nMOS transistors are reset to logic “0” during the reset period.

B. CD Logic

To mitigate the above-mentioned problems, CD logic is proposed with a schematic shown in Fig. 2.1(a). Timing block (TB) creates an adjustable window period to reduce the static power dissipation. Logic Block (LB) helps to reduce the unwanted glitch and also makes cascading CD logic feasible. A buffer implemented in CD logic with schematics of TB and LB is shown in Fig. 2.1(b). 1) CD Logic Operation: Fig. 2.2 depicts the corresponding CD logic timing diagram and flowchart. For simplicity, we assume that IN come from dynamic domino logic gates. When CLK is high, CD logic predischarges both X and Y to GND. When CLK is low, CD logic enters the evaluation period and three scenarios can take place: namely, the contention, C–Q delay, and D–Q delay modes. The contention mode happens when CLK is low while IN remain at logic “1.” In this case, X is at a nonzero voltage level which causes Out to experience a temporary glitch.

![Fig. 2.1. CD logic (a) block diagram and (b) buffer.](image)

The duration of this glitch is determined by the local window width, which is determined by the delay between CLK and CLK_d. When CLK_d becomes high, and if X remains low, then Y rises to logic “1,” and turns off M1. Thus the contention period is over, and the temporary glitch at Out is eliminated. C–Q delay mode takes places when IN make a transition from high to low before CLK becomes low. When CLK becomes low, X rises to logic “1” and Y remains at logic “0” for the entire evaluation cycle. The delay is measured by the falling edge of both CLK and Out: hence the name C–Q delay. D–Q delay mode utilizes the pre-evaluated characteristic of CD logic to enable high-performance operations. In this mode, CLK falls from high to low before IN transit, hence X initially rises to a nonzero voltage level. As soon as IN become logic “0,” while Y is still low, then X quickly rises to logic “1.” A race condition exists in this case between X and Y. If CLK_d rises much earlier than X and Y will go to logic “1,” turn off M1, and result in a false logic evaluation. If CLK_d rises slightly slower than X, then Y will initially rise (thus slightly turns off M1) but eventually settle back to logic “0.” CD logic can still perform the correct logic operation in this case, however, its performance is degraded because of M1’s reduced current drivability.

Therefore, it is important to maintain a sufficient window width under process–voltage–temperature (PVT) variations. Table I presents a summary of CD logic’s operations.
Compared to FTL, where the contention lasts for the entire evaluation period, TB effectively reduces CD logic’s power consumption during the contention mode. The local window technique in the proposed CD gate allows designers to customize the window width for different logic expressions to achieve minimal power dissipation while not sacrificing the performance. For instance, a multiple input NAND gate will require a longer window width than a NOR gate because of the larger internal capacitive due to the stacked NMOS transistors. Another advantage of CD logic is that the internal node (X) is always connected to either VDD or GND, thus making the robustness of CD logic comparable to static logic, except during the contention mode. CD logic eliminates the problem of false logic evaluation associated with cascaded FTL. Consider a cascaded CD logic system, in which the inputs to nMOS PDN are always at logic “1” when first entering the evaluation period, because X and Out are always precharged to logic “0” and “1,” respectively. Therefore, when CLK is low, CD gates will always first enter the contention mode and conditionally make a low-to-high transition depending on the inputs. This is not the case for the first stage CD gate, however, as there is no guarantee that the inputs will always be at logic “1.” In other words, designers need to ensure that the input signals to the first CD gate arrive earlier than the clock signal, i.e., operate in C–Q delay mode only.

III. 8-BIT RIPPLE CARRY ADDER

Analysis:

As shown in the previous section, CD logic demonstrates speed advantage over static and dynamic logics only during D-Q mode, which only takes places in a single cycle, multi-stage system. In this chapter, three 8-Bit ripple carry adders (RCAs) using static, dynamic, and CD logic style are laid out and simulated to compare their performance.

A. Addition

Addition is the most frequently used arithmetic operation and is often the speed-limiting element of arithmetic logic unit in modern CPUs. Addition forms the basis for many processing operations, including counting, multiplication, and digital signal filtering. As a result, adder circuits that add two binary numbers are of great interest in the field of digital IC and are often used as a test bench to compare different logic styles.

Table 1 shows the truth table of a binary full adder. A and B are the adder inputs, Ci is the carry input, S is the sum output, and Cout is the carry output. Based on this truth table.

A full adder (FA) unit can be implemented with the above boolean equations using CMOS transistors. To construct a N-bit adder, N FA units can be cascaded in series, connecting Cout of a FA cell to the next stage FA. Ci input. This configuration is known as ripple-carry adder, since the carry bit “ripples” from one stage to the other. A block diagram of a N-bit ripple carry adder (RCA) is shown in Figure 3.

For a RCA, the worst case delay is the propagation of carry signal from the least significant carry bit (Ci) to the most significant carry bit. Furthermore, an additional stage is required to produce the sum based on this carry signal. In the case of RCA, the delay is then proportional to number.

3.2 8-Bit Ripple carry adders:

The basic full adder (FA) is implemented with an energy efficient, mirror structured 2T cell with sizing strongly favoured to compute Cout, PMOS and NMOS transistors in the carry generation are sized to be six times and three times larger than the sum generation transistors respectively. Since the critical path of RCAs goes from Cin to Cout, in dynamic and CD logic implementations only the carry generation circuitry is replaced while the sum computation remains static. During this experiment we observe that the parasitic capacitance contributed by both internal (within a cell) and global (stage to stage) interconnects are critical and can increase the total delay by up to 40% from schematic to post-layout simulation. Therefore, to ensure fair comparison we first laid out each RCA, then extracted the corresponding parasitic capacitance at every node, and finally back-annotated the extracted lump capacitance value to the schematic.

Compared to post layout simulations, extensive studies reveal that schematic simulations with back-annotated capacitance achieve a result difference of less than 2% but are at least 3X faster in terms of simulation time. Therefore, all the data are generated from simulations with back-annotated lumped capacitance for the rest of the comparisons.

IV. SIMULATION RESULTS

Design a RCA using Static, Dynamic, CD logics as shown in figers.
Figure 4 (a) - dynamic domino logic with a footer transistor Schematic using in DSCH Tool.

Figure 4 (b) - dynamic domino logic without a footer transistor Schematic using in DSCH Tool.

Figure 4 (c) - constant delay logic Schematic using in DSCH Tool.

Figure 4 (d) – Proposed Full adder Schematic using in DSCH Tool.

Figure 4 (e) 8 Bit RCA using dynamic domino logic layout in 50nm.

Figure 4 (f) 8 Bit RCA using CD logic with a footer transistor layout in 50nm.

Figure 4 (g) Proposed 8 Bit RCA layout in 50nm.

Figure 4 (h) Simulation output 8 Bit RCA using dynamic domino logic.

Figure 4 (i) Simulation output 8 Bit RCA using CD logic.

Figure 4 (j) Simulation output Proposed 8 Bit RCA.

Table 2. shows RCA performance comparison of different logic styles.

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Dynamic</th>
<th>CD</th>
<th>Proposed Logic</th>
</tr>
</thead>
<tbody>
<tr>
<td>Power</td>
<td>35.76</td>
<td>48</td>
<td>44.6</td>
</tr>
<tr>
<td>Delay(ps)</td>
<td>100</td>
<td>60</td>
<td>40</td>
</tr>
<tr>
<td>Area µm2</td>
<td>1185.5</td>
<td>1563.8</td>
<td>1080.2</td>
</tr>
</tbody>
</table>

V. CONCLUSION

A new high-performance proposed logic style main goal was to make the domino circuits more robust and with low leakage without significant performance degradation or increased power consumption. Relative decrease in power consumption and delay is achieved by...
using current comparison domino. Performance analysis of 8-bit RCAs reveals that proposed logic is faster than static and dynamic domino logic & CD logic respectively. A RCA using Static Full adder, Dynamic full adder CD logic Full adder & Proposed Full adder is designed in MicroWind Tool.

REFERENCES


