Comparison of a 17b Multiplier in Dual-Rail Domino and in Dual-Rail D^{3}L (D^{4}L) Logic Styles

*R. Rafati, A. Z. Charaki, G. R. Chaji, S. M. Fakhraie, K. C. Smith**
ECE Department, University of Tehran, Tehran, Iran
**ECE Department, University of Toronto, Toronto, Canada*

ABSTRACT

In this paper, a new family of dynamic logic gates called Dual-rail Data-Driven Dynamic Logic (D^{4}L) is introduced. In this logic family, the synchronization clock signal has been eliminated and correct precharge and evaluation sequencing is maintained by appropriate use of data instances. The methodology and characteristics of D^{4}L are demonstrated in the design of a CLA 32-b adder and a 17-b high-speed multiplier. Based on VHDL simulations, the D^{4}L implemented 32-b adder has 23% less switching-activity than a comparable Domino adder and for D^{4}L multiplier switching-activity is 14.5% less than its Domino rival. HSPICE simulation in a 0.6μm CMOS process shows that D^{4}L has a 17% power saving over Domino in a 32-b CLA adder design and a 10% saving in a 17-b multiplier design while a D^{4}L adder has 8% less delay than a Domino one.

1. INTRODUCTION

We can divide most digital systems into control and data-path units. Generally speaking, the data path is regular and the speed of the system mainly depends on its delay. Therefore high-speed architectures can use dynamic logic [1] for their data path to increase speed and reduce critical-path delay. However power dissipation of clock signal is one of the drawbacks of dynamic logic styles. Data-Driven Dynamic Logic (D^{3}L) style [2,3] is one solution to reduce the severity of this problem. The main purpose of D^{3}L is to use input signals instead of one global clock signal for correct precharge and evaluation sequencing. Correspondingly, clock-buffering and clock-distribution problems can be eliminated from dynamic gates, implying less switching activity and less power dissipation. Furthermore, by replacing the clock signal with input signals, the "foot" transistor can be eliminated without causing a short-circuit problem. Accordingly, this replacement reduces power consumption and increases speed by the elimination of foot transistor.

2. DYNAMIC AND D^{3}L LOGIC STYLE

A dynamic logic style such as Domino logic normally operates in two phases: precharge and evaluation. In the precharge phase when the clock is low, outputs are charged to a fixed value. In the evaluation phase, when the clock is high, either outputs remain at their precharge level, or are changed to the opposite value. In D^{3}L, we can use a collection of input signals instead of a global clock signal for precharging dynamic gates. As long as the values of those precharging inputs have not changed, the gate remains in the precharge state. At the onset of a change on one of the precharging signals, the evaluation phase will start and the final output value would depend on the state of the other inputs.

Fig. 1 shows a Domino-logic implementation of an *AND* gate along with its D^{3}L implementation. In the precharge phase, both of the inputs become low through previously precharged dynamic gates. This turns on P_A and the Q output becomes low as well. During the evaluation phase, if A remains low, the output would be low regardless the value of B. If A changes to high, the final value of Q depends on the evaluated value of B.

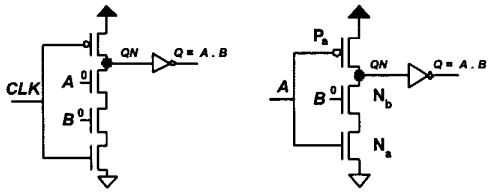

Fig. 1. (a) Domino *AND* gate, (b) D^{3}L *AND* gate.

3. DUAL-RAIL DOMINO AND D^{4}L LOGIC STYLES

A typical dynamic logic, for example Domino logic, has superior performance in the implementation of high-speed circuits. However, in contrast to their efficiency in non-inverting structures, they have inherent shortcomings in the implementation of inverting logics such as XOR. The XOR function is a basic building block in implementing adders and multipliers and its efficient implementation has a considerable effect in overall performance. The dual-rail implementation is one solution for implementing such gates in dynamic logic form. Dual-rail logic is a type of DCVSL logic family [1], which overcomes the non-inverting characteristic of single-rail domino logic. A dual-rail domino two-input XOR gate is shown in Fig.2. In precharge phase when clock is low, both outputs (X.H, X.L) are set low while the input signals are held low from previous dynamic gates at the same time. In the evaluation phase depending on the input values only one of (X.H, X.L) outputs is set to high value.

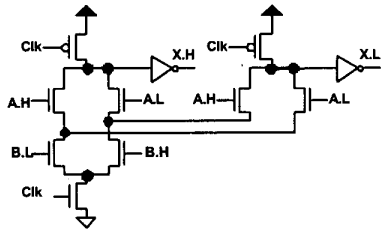

Fig. 2. Two-input Domino XOR gate.

In D^{4}L logic, our challenge is to eliminate the global clock signal and to replace it with a suitable input combination in PUN. One such two-input D^{4}L XOR is shown in Fig. 3. In this structure, we have used the signal pair (B.H, B.L) for precharging

corresponding gate, instead of a clock signal. When the precharging wave reaches the inputs of D⁴L XOR, set them to low and precharge the outputs to high. In the evaluation phase, one of the rails in (B.H, B.L) and (A.H, A.L) is set to high and prevent short circuit between Vdd and ground in this phase.

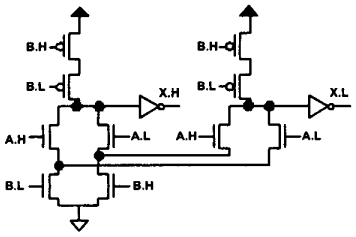

Fig. 3. Two-input D⁴L XOR gate.

4. MULTIPLIER ARCHITECTURE

In order to show the applicability of D⁴L design, we have implemented a 17b by 17b multiplier as a case study for our simulation experiments. The block diagram of the designed multiplier is shown in Fig. 4. It is composed of four major parts: Booth Encoder (BE), Partial-Product-Generators (PPG), Wallace-Tree of 42 compressors and a high-speed 32b CLA adder for final addition [4]. Each BE encodes three bits of multiplier input and sends the results to an array of PPGs. PPG blocks decode this information and generate the appropriate PPs from the multiplicand inputs. To increase the speed of the multiplier, a Wallace-Tree of 4-2 compressors is used which can sum up four PPs concurrently. Correspondingly, in the critical path only 3 stages of compressors needed to add nine rows of PPs.

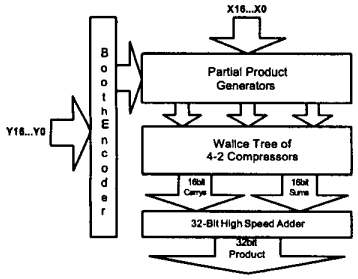

Fig. 4. Multiplier architecture.

For final addition, a high-speed 32-b CLA adder is used. In this design, we have made a tradeoff between speed and power, the two important factors in digital design. For this reason, we have used dynamic implementation of the Manchester Carry Chains (MCCs) as our basic building block in the 32b CLA adder.

5. MULTIPLIER DESIGN IN DOMINO AND D⁴L LOGIC STYLES

5.1. Booth Encoder

In Fig. 5, the logic circuit of a Booth Encoder (BE) is shown. We implement the BE with three logic cells, two XORs and one complex logic cell. Since each BE's outputs has to drive 18 PPG blocks, proper transistor sizing and buffer insertion must be employed.

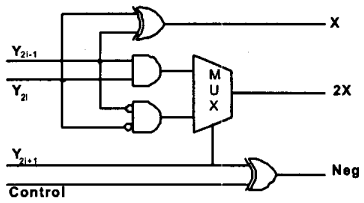

Fig. 5. Booth Encoder.

In D⁴L implementation, the clock must be replaced by a suitable combination of input data. Since at the first stage, none of the inputs has a predefined value, we employ the usual Domino implementation to force the BE's outputs to a predetermined value. In other words, this initiates a precharge wave that is essential for correct operation of a D⁴L design. This wave traverses all of the blocks from the PPGs to the final adder output and puts all the D⁴L gates into the precharge state. For the above reason, BEs have been implemented in conventional dual-rail Domino logic for this D⁴L implementation. These Domino-implemented BE are responsible for initiating the precharge and evaluation phases for the succeeding D⁴L gates.

5.2. Partial-Product Generator (PPG)

The PPG logic circuit is shown in Fig. 6. Based on BE-generated codes, the PPG selects one of X_i or X_{i-1}, or complements them based on the *Neg* value.

In the PPG block shown in Fig.6, there is one AO22 gate and one XOR gate. The XOR gate complements the OR output based on the value of *Neg* signal. To reduce delay in the dynamic implementation, all parts of the above-mentioned circuit are implemented in one complex logic block as shown in Fig. 7.

For D⁴L implementation, we use the *Neg* signal for *CLK* replacement since it has less delay than the other BE outputs (Fig. 8). This choice reduces the capacitance load on the other inputs on the critical path, to the same value as found for the Domino implementation.

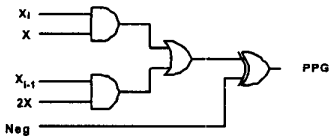

Fig. 6. Partial-Product Generator.

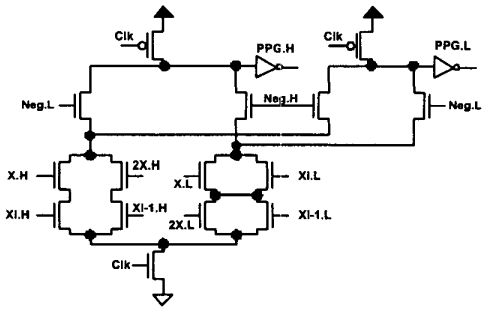

Fig. 7. PPG implementation in dual-rail Domino.

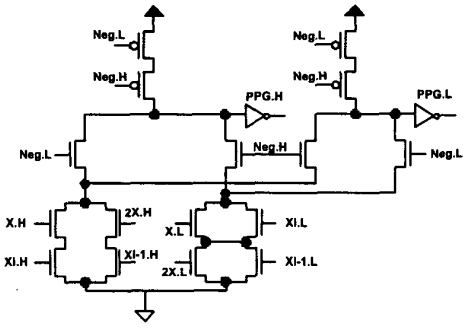

Fig. 8. PPG implementation in D^{4}L.

5.3 Wallace Tree and 4-2 Compressors

For summing up the PPs, 4-2 compressors are used in the Wallace Tree structure. As shown in Fig. 9, a 4-2 compressor has five inputs (I_0, I_1, I_2, I_3, C_{in}) and three outputs (Sum, $C_{out\,1}$, $C_{out\,2}$) and is composed of two serially-connected Full Adders.

To generate the summation signal in one compressor, a dynamic gate is used for each of the three-inputs XOR. A two-input version of the dynamic XOR gate shown in Fig. 2 can be easily enhanced to a three-input one.

In D^{4}L implementation, we replace the clock signal with selected inputs of the 4-2 compressor. In a typical compressor some of the inputs come from PPGs while the others come from a previous compressor stage. In finding a replacement for the clock, it is better to select the inputs that directly come from PPGs, since these signals do not lie in the critical path of the multiplier.

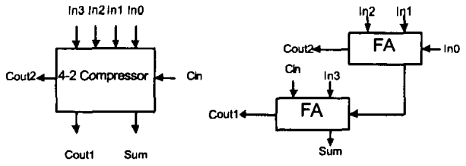

Fig. 9. 4-2 Compressor.

5.4 High-Speed 32-b CLA Adder

In the last stage of the multiplier, a fast 32-b adder is required to produce the final results. The architecture of our adder is based on carry-look-ahead addition. The basic building block of this design is a 4b dynamic MCC (Manchester Carry Chain) [1]. MCC is implemented in a way similar to that used Multiple Output Domino Logic (MODL [5]) while the intermediate nodes are precharged to Vdd. These MCCs are used for carry calculation and carry propagation in 4-b groups. Two-input XOR gates are employed in an adder structures for generating sum outputs (Fig. 10).

Domino and D^{4}L implementation of a 4-b MCC are shown in Fig. 11 and Fig. 12 respectively. In D^{4}L implementation of a 4-b MCC, G_i and P_i signals are used instead of the global clock signal. In the first stage, block *A* generates 32 Propagate (P_i), 32 Generate (G_i), 32 Kill (K_i) and 8 Group-Propagate (GP) signals.

$$P_i = A_i \oplus B_i, \qquad i = 1\ldots32$$

$$G_i = A_i.H \bullet B_i.H \qquad i = 1\ldots32$$

$$K_i = A_i.L \bullet B_i.L \qquad i = 1\ldots32$$

Notice that G_i and K_i are single-rail signals while *B* blocks in the second stage are 4b MCC which generate 8 Group Generate (GG) and 8 Group Kill (GK) from 32-b P_i, G_i and K_i signals. In the third stage, an 8 bits MCC uses these group signals (GG_i, GK_i, GP_i) for generating C_4, C_8 ... C_{28}. 4-b MCCs at second stage use these carry signals as inputs for calculating all carry output signals. The final sums are produced from these carry signals and previously-generated P_i signals from *A* blocks.

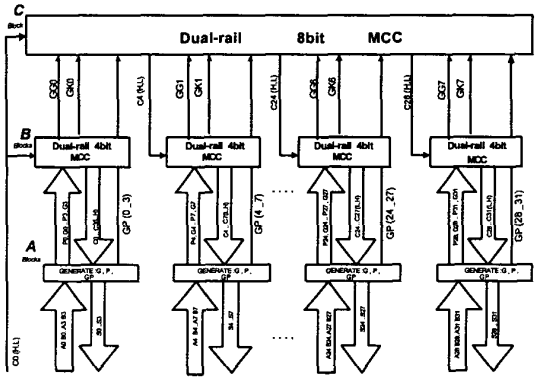

Fig. 10. 32-b CLA adder architecture.

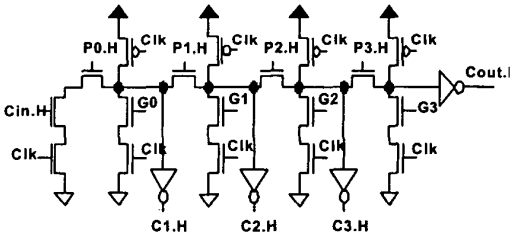

Fig. 11. Domino implementation of the 4-b MCC.

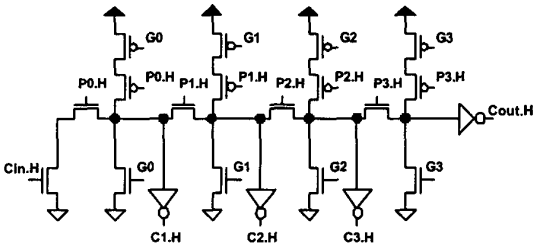

Fig. 12. D^{4}L implementation of the 4-b MCC.

6. SIMULATION RESULTS

For the estimation of power consumption, we have defined a shared variable in VHDL, called Event-Count (EC). An event at any input increments the EC value by an appropriate number. EC incrementing is weighted by consideration of the total capacitive load on each signal. For example, in a two-input Domino XOR gate (Fig. 2) every rising edge of the clock signal, results in an EC increase by 5 units. The reason is that there are two PMOS's and one NMOS transistor that should be driven by the clock signal. If we assume the width of a PMOS transistor to be twice that of an NMOS, then the EC value must be incremented by 5 for each rising edge of the clock. As shown in Table I, VHDL simulation for a 2-input XOR gate shows that D^{4}L design has 10% less switching activity than its Domino equivalent. In the same way, HSPICE simulation shows a 12% power reduction for D^{4}L gates over domino ones. Table II shows the aggregated EC value after applying 1000 random test vectors to Domino and D^{4}L implementations of the multiplier. As seen in this table, the D^{4}L implementation has about 14.5% less switching activity than the Domino one, and the D^{4}L 32-b adder implemented has 23% power saving over Domino. Basically the power saving of D^{4}L over Domino is the result of two factors: one is the elimination of foot transistors in D^{4}L, and the second is that substituted signals instead of a global signal generally has a switching activity less than one (i.e. $\alpha < 1$). By using the dual-rail logic style, we can have the advantage of only the first factor. If we assume *A.h* and *A.l* are precharging a dynamic gate, then at least one of them causes an event in the evaluation phase. Therefore resulting switching activity is equal to that in one clock signal in a usual dynamic implementation. However in the 32-b adder design, because of the existence of two single-rail signals (G_i , K_i) with low switching activity that take part in clock signal substitution, D^{4}L power saving over Domino reaches 23%.

For more accurate estimation of total power and delay of the multiplier, we have used HSPICE models of a 0.6μ CMOS process with a 5v V_{DD} supply for our simulations. The worst-case delay and average power of four main parts of the multiplier have been simulated in HSPICE and are shown in Table III. However, in this simulation the power consumption of the clock distribution network in dynamic logic is not considered. Finally, we add up the average power consumption of all blocks to compute the total power consumption of the multiplier.

There are one BE, one PPG, three compressors and one 32-b adder in the critical-path of the multiplier. A summary of HSPICE results for delay is shown in Table IV.

7. CONCLUSION

A new dual-rail dynamic logic called Dual-rail Data-Driven-Dynamic Logic or D^{4}L has been introduced. D^{4}L is a type of dual-rail dynamic logic from which the global clock has been eliminated and correct precharge and evaluation sequences are performed under control of the input data instead of by a global clock. We have employed both D^{4}L and Domino logic in designing a high-speed 17-b multiplier. VHDL simulation shows that switching activity of the D^{4}L multiplier is 14.5% less than that of the Domino one. Moreover, HSPICE simulation shows a 10% power saving for the D^{4}L multiplier over the Domino one. Simulation of the entire 32-b adder shows that D^{4}L consumes 17% less power than Domino one, in spite of the fact that the power consumption of the clock distribution network needed in Domino has not been considered yet.

Table. I VHDL and HSPICE simulation for a 2-input XOR gate.

	Domino	D^{4}L	D^{4}L/ Domino
Event count in VHDL	21995	19996	0.9
Power (mw) in HSPICE	0.707	0.62	0.88

$W_P = 2W_N$ @ 250MHz

Table. II. Event count for individual blocks of the multiplier.

Multiplier Blocks	Domino (EC)	D^{4}L (EC)	D^{4}L / Domino
BE	665,829	665,829	1
PPG	4,536,000	4,218,480	93%
Compressor	8,106,957	6,848,254	84%
32-b Adder	2,836,018	2,175,884	77%
Total	16,144,804	13,908,447	86%

Table. III. HSPICE simulation for average power consumption @ 125 MHz.

Multiplier Blocks	Domino (mW)	D^{4}L (mW)	D^{4}L / Domino
BE	2.02	2.02	1
PPG	1.18	1.07	91%
Compressor	2.83	2.56	90%
32-b Adder	38.5	32	83%
Total (mW)	241	217	90%

Table. IV. HSPICE simulation results for worst-case delay.

Multiplier Blocks	Domino		D^{4}L	
	Pre-charge (ps)	Evaluate (ps)	Pre-charge (ps)	Evaluate (ps)
BE	353	477	353	477
PPG	329	304	449	374
Compressor	307	577	488	603
32-b Adder	607	2500	1007	2300
Multiplier Delay	607	5012	2297	4960

8. REFRENCES

[1] J. M. Rabaey, *Digital Integrated Circuits,* Prentice Hall, 1996.

[2] R. Rafati, S. M. Fakhraie, K. C. Smith, "Low-Power Data Driven Dynamic Logic," *ISCAS'* 2000, vol. 1, pp. 752-755.

[3] R. Rafati, A. Z. Charaki, S.M. Fakhraie, K. C. Smith, "Data Driven Dynamic Logic versus NP-CMOS Logic a Comparison," in Proc of *ICM-2000*, Oct. 2000, Iran-Tehran.

[4] Muhmmod S.Elrabaa, Issan S.Abu-Khater, Mohamad I.Elmasry, "Advanced Low-power Digital Circuit Techniques," Kluwer Accademic Publisher, 1997.

[5] Z. Wang, G. A. Jullien, W. C. Miller, J. Wng, S. Bizzan,"Fast Adder Using Enhanced Multiple-Output Domino Logic", *IEEE JSSC*, vol. 32, pp.206-214, Feb. 1997.