International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 3 Issue12, December-2014

High Performance Logic Style with Constant Delay characteristics and Self-Reset Circuitry

V. Divya Vani,
Pg Student (M.Tech-Embedded Systems); Rajiv Gandhi Memorial College Of Engineering And Technology, Nandyal, Kurnool (Dist.)

Dr. T. Jayachandra Prasad,
Professor & Principal; Rajiv Gandhi Memorial College Of Engineering And Technology, Nandyal, Kurnool (Dist.)

Abstract:- **The concept of Constant Delay Logic has already been established that it's performance is better compared to Feed Through Logic, Source Coupled Logic, Static Logic and Dynamic Logic, etc. The concept of Constant Delay Logic is extended to SR Latch and 8-bit Comparator in this paper. These two circuits have been simulated in various technologies like 65nm, 45nm and 32nm. The performance of SR Latch and 8-bit Comparator have been compared with Static and Dynamic Logics in the above technologies and found that the SR Latch and 8-bit Comparator implemented in CD Logic style out performs the other logics. The comparison in different technologies for Static, Dynamic and CD logics have been made in the metrics like delay, power, power-delay product and energy-delay product. It is found that in 32nm technology CD Logic delay is less by 96% and 68% in comparison with Static and Dynamic logics respectively. It is also found that in 32nm technology SR Logic and 8-bit Comparator implemented in CD Logic style is faster by 52% and 46% with respect to Static and Dynamic logics.**

INTRODUCTION

High-performance energy-efficient logic style has always been a well-liked search topic in the field of VLSI circuits because of the continuous demands of ever-increasing circuit operating frequency. The invention of domino logic allows designers to implement high-performance circuit blocks, at an operating frequency that traditional static and pass transistor CMOS logic styles find

difficult to achieve [2]. Though, the performance development comes with several costs, including a reduced noise margin, a predicament of charge-sharing and higher power dissipation due to a superior data activity. Several variations of the dynamic domino logic, namely NP domino (NORA domino) [3], zipper domino [4], and data-driven dynamic logic (*D*3L) [5], [6], have been projected but they are never pervasive in the VLSI industry [7], [8].

Compound domino logic (CDL), where dynamic and static gates flashing between each other, has become the most trendy logic style in high-performance circuit blocks, i.e., 64-bit adder [9]-[12], in modern CPUs. In this design, the output inverter is replaced with a more difficult inverting static gate, i.e., NAND, such that the monotonicity necessity is satisfied while conducting complex logic operations without wasting the one inverter delay [13]. Moreover, all the dynamic stages except the first stage can be footless in CDL. This implementation, however, comes at the expense of increased power consumption due to the possible direct path current during the pre-charge period and a reduced noise margin as a result of unprotected dynamic domino logic's outputs.

A considerable research attempt has been committed to exploring new logic styles that go beyond dynamic domino logic and CDL. In particular, source-coupled logic (SCL) [14] has shown finer performances that are difficult to achieve by any other logic styles. However, it suffers from high power dissipation due to a constant current draw, and its differential nature requires complementary signals. Pseudo-nMOS logic, which uses a single pull-up pMOS transistor, provides both high speed and low transistor count at the expense of high static power consumption as well as reduced output voltage swing. Output prediction logic (OPL) [15] has also shown superior performance in high-speed adders [16]. Nevertheless, OPL requires the generation and distribution of multiphase clock signals with small timing separations and low skews, which are difficult to achieve. While numerous high-speed logic styles have been proposed, dynamic and CDL still remain the most attractive choices when performance is the primary concern.

In recent years, a new way of logic operation, also known as feed-through logic (FTL) [17], [18], has been proposed, which has demonstrated its high-performance capability. Consider dynamic domino logic, the critical path consists of nMOS logic transistors. In FTL however, the roles of the clock and logic transistors are interchanged and the clock transistor is now the critical path. The first

generation of FTL exhibits many shortcomings, including excessive power dissipation, and reduced noise margin. To mitigate these problems, we propose a new high-performance logic, which we call "constant delay" (CD) logic. CD logic provides a local window technique and a self-reset circuit which enable robust logic operation with minimized power consumption while maintaining FTL's speed advantage. The most distinct characteristic of CD logic from previously proposed logic styles is that the delay is, on a first-order approximation, not affected by the logic expression. Unlike SCL, CD logic does not require complementary signals and can be easily integrated with static and dynamic domino logics. Also, CD logic does not have the problem of constant static power dissipation similar to pseudo-nMOS. Furthermore, the clock timing requirement of CD logic is not as stringent as OPL. CD logic can achieve robust operation with optimal performance as long as clock signals arrive earlier than the input signals.

EVOLUTION OF CD LOGIC

CD Logic:

To mitigate the above-mentioned problems, CD logic is proposed with a schematic shown in Fig. 1(a).

1. Timing block (TB): TB creates an adjustable window period to condense the static power dissipation [1].
2. Logic Block (LB): LB helps to moderate the avoidable glitch and also makes cascading CD logic realizable. A buffer implemented in CD logic with blocks of TB and LB is shown in Fig. 1(b) [1].

CD Logic Operation:

Fig. 2 depicts the resultant CD logic timing diagram and flowchart. For candour, we imagine that IN come from dynamic domino logic gates. While CLK is high, CD logic pre-discharges both *X* and *Y* to GND. As CLK is low, CD logic enters the evaluation period and three scenarios can obtain: in particular, the contention, C–Q delay, and D–Q delay modes. The contention mode happens when CLK is low as IN remain at logic "1." In this case, *X* is at a nonzero voltage level which causes Out to incident a impermanent glitch. The time taken of this glitch is stubborn by the local window width, which is firm by the delay among CLK and CLK_d. When CLK_d becomes high, and if *X* rest low, then *Y* rises to logic "1," and turns off *M1*. So the contention period is ended, and the temporary glitch at Out is eliminated. C–Q delay mode takes places when IN make a shift from high to low earlier to CLK becomes low. When CLK becomes low, *X* rises to logic "1" and *Y* residue at logic "0" for the complete evaluation cycle. The delay is calculated by the falling edge of both CLK and Out: so the name C–Q delay. D–Q delay mode utilizes the pre-evaluated feature of CD logic to make possible high-performance operations. In this mode, CLK falls from high to low further of IN transit, thus *X* chiefly rises to a nonzero voltage level. One time IN turn into logic "0," while *Y* is still low, then *X* quickly rises to logic "1." A race condition exists in this case among *X* and *Y*. If CLK_d rises much preceding than *X* and *Y* will go to logic "1," turn off *M1*, and outcome in a false logic estimation. If CLK_d rises to some extent slower than *X*, then *Y* will initially rise (thus to some extent turns off *M1*) but finally reconcile back to logic "0." CD logic can still carry out the accurate logic operation in this case, though, its performance is ruined because of M1's compact current drivability [1].

As a result, it is considerable to save a enough window width below process–voltage–temperature (PVT) variations. Compared to FTL, where the contention lasts for the complete

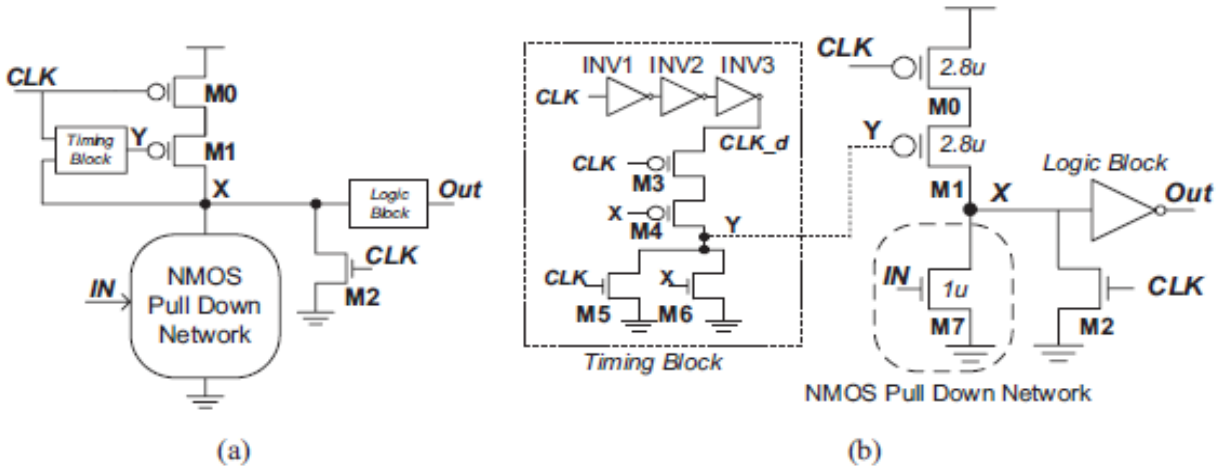


Figure 1: CD Logic (a) block diagram and (b) buffer

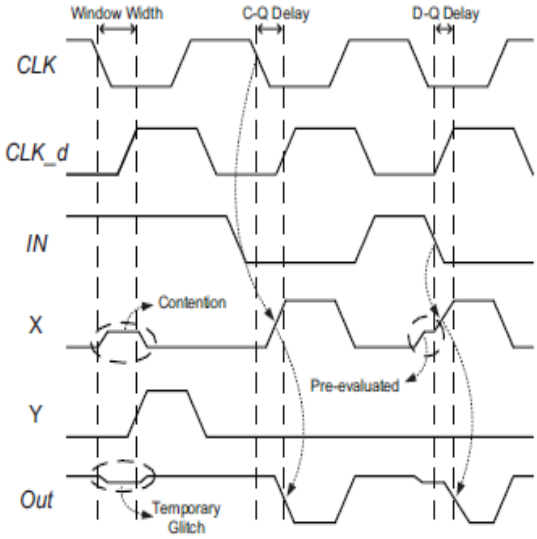


Figure 2: Timing diagram and flowchart of the proposed CD Logic

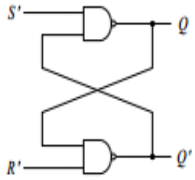
evaluation period, TB powerfully reduces CD logic's power consumption for the duration of the contention mode. The local window system in the considered CD gate allows designers to change the window width for unlike logic expressions to acquire negligible power dissipation while not sacrificing the performance [1]. For example, a multiple input NAND gate will necessitate a longer window width than a NOR gate as of the larger internal capacitance due to the stacked nMOS transistors. An added development of CD logic is that the internal node (*X*) is forever connected to either VDD or GND, so making the vitality of CD logic corresponding to static logic, apart from all over the contention mode. CD logic

eliminates the difficulty of false logic evaluation linked with cascaded FTL. Consider a cascaded CD logic system, in which the inputs to nMOS PDN are all the time at logic "1" when original arriving the evaluation period, as *X* and Out are always pre-discharged and pre-charged to logic "0" and "1," respectively. Consequently, when CLK is low, CD gates will everlastingly 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, but, as there is no declaration that the inputs will always be at logic "1." In other words, designers need to declare that the input signals to the first CD gate get there preceding than the clock signal, i.e., function in C–Q delay mode only [1].

PERFORMANCE ANALYSIS

SR-Latch:

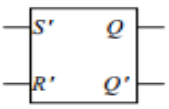
The bi-stable element is able to memorize or store up one bit of information. But, since it does not have any inputs, we cannot alter the information bit that is stored in it. With the purpose of modify the information bit, we require to add inputs to the circuit. The easiest way to add inputs is to change the two inverters with two NAND gates as shown in Figure 3(a). This circuit is called a SR latch. Besides to the two outputs Q and Q', there are two inputs S' and R' for set and reset respectively. Following the convention, the prime in S and R denotes that these inputs are active low. The SR latch can be in one of two states: a set state when Q = 1, or a reset state when Q = 0. To make the SR latch go to the set state, we simply assert the S' input by setting it to 0. Keep in mind that 0 NAND no matter which gives a 1, hence Q = 1 and the latch is set.



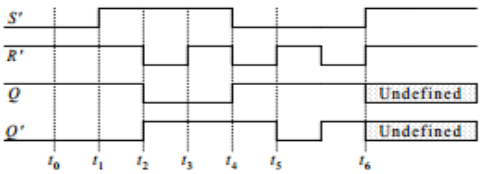
(a)

S	R	Q	Q'
1	0	0	1
1	1	0	1 (after S=1, R=0)
0	1	1	0
1	1	1	0 (after S=0, R=1)
0	0	1	1

(b)



(c)

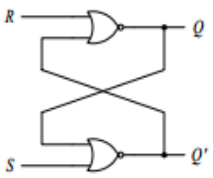


(d)

Figure 3: SR Latch: (a) circuit using NAND gates; (b) truth table; (c) logic symbol (d) timing diagram

If R' is not asserted (R' = 1), then the output of the bottom NAND gate will give a 0, and so Q' = 0. This state is shown in Figure 3 (d) at time t0. If we de-assert S' so that S' = R' = 1, the latch will stay at the set state because Q', the second input to the top NAND gate, is 0 which will keep Q = 1 as shown at time t1. At time t2 we reset the latch by considering R' = 0. Now, Q' goes to 1 and this will oblige Q to go to a 0. If we de-assert R' so that once more we have S' = R' = 1, this time the latch will wait at the reset state as shown at time t3. Observe the two times (at t1 and t3) when both S' and R' are de-asserted. At t1, Q is at a 1, while, at t3, Q is at a 0. When both inputs are de-asserted, the SR latch maintains its previous state. Preceding to t1, Q has the value 1, so at t1, Q ruins at a 1. Correspondingly, prior to t3, Q has the value 0, so at t3, Q remnants at a 0.

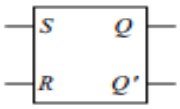
If both S' and R' are asserted, then both Q and Q' are identical to 1 as shown at time t4. If one of the input signals is de-asserted in advance than the other, the latch will end up in the state forced by the signal that was de-asserted later as shown at time t5. At t5, R' is de-asserted first, so the latch goes into the usual set state with Q = 1 and Q' = 0. A difficulty exists if both S' and R' are de-asserted at accurately the similar time as shown at time t6. If both gates have accurately the same delay then they will both output a 0 at accurately the same time. Feeding the zeros reverse to the gate input will turn out a 1, yet again at exactly the similar time, which again will produce a 0, and so on and on. This oscillating performance, called the dangerous event, will carry on evermore. If the two gates do not have precisely the same delay then the condition is like de-asserting one input before the other, and so the latch will go into one situation or the other. Though, as we do not know which is the faster gate, so, we do not know which situation the latch will go into. As a result, the latch's next state is indeterminate.



(a)

S	R	Q	Q'
1	0	1	0
0	0	1	0 (after S=1, R=0)
0	1	0	1
0	0	0	1 (after S=0, R=1)
1	1	0	0

(b)



(c)

Figure 4: SR Latch: (a) circuit using NOR gates; (b) truth table; (c) logic symbol.

In order to avoid this in-deterministic performance, we have to put together that the two inputs are not at all de-asserted at the similar time. Note that both of them can be de-asserted, but now not at the identical time. Actually, this is assured by not having both of them asserted. One more reason why we do not want both inputs to be asserted is that when they are both asserted, Q is equal to Q', but we typically want Q to be the inverse of Q'.

From the above examination, we get the truth table in Figure 3(b) for the NAND implementation of the SR latch. Q is the present state or the current content of the latch and Q' is the value to be simplified in the next state. Figure 3(c) shows the logic symbol for the SR latch. The implementation of SR Latch and it's simulation results using CD logic in 32nm technology is shown in figure 5, 6 and it's performance comparison is shown in table 2.

The SR latch can also be implemented using NOR gates as shown in Figure 4(a). The truth table for this implementation is shown in Figure 4(b). From the truth table, we see that the main variation among their implementation and the NAND implementation is that for the NOR implementation, the S and R inputs are active high, so that setting S to 1 will set the latch and setting R

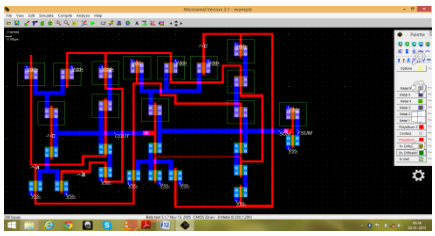


Figure 5:Layout diagram for SR-Latch using CD Logic.

to 1 will reset the latch. Nevertheless, just like the NAND realization, the latch is set when Q = 1 and reset when Q = 0. The latch remembers its prior condition when S = R = 0. When S = R = 1, both Q and Q' are 0. The logic symbol for the SR latch using NOR implementation is shown in Figure 4(c).

Figure 6:Simulation Results for SR-Latch using CD Logic.

8-bit Comparator:

A comparator is a unique combinational circuit intended mainly to compare the relative magnitude of two binary numbers shown in below figure. It receives two n-bit numbers A and B as inputs and outputs are A > B, A = B, A < B. Depending upon the relative magnitudes of the two number, one of the will be high. The truth table of comparator is shown in table 1. The logic diagram and it's implementation of 8-bit Comparator, it's simulation results using CD logic in 32nm technology is shown in figure 7, 8, 9 and it's performance comparison is shown in table 3.

Table 1: Truth Table for Comparator

Inputs		Outputs		
B	A	A > B	A = B	A < B
0	0	0	1	0
0	1	1	0	0
1	0	0	0	1
1	1	0	1	0

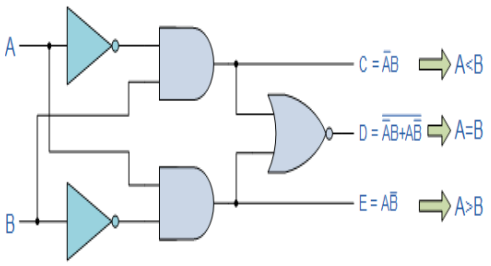


Figure 7: Logic diagram of Comparator.

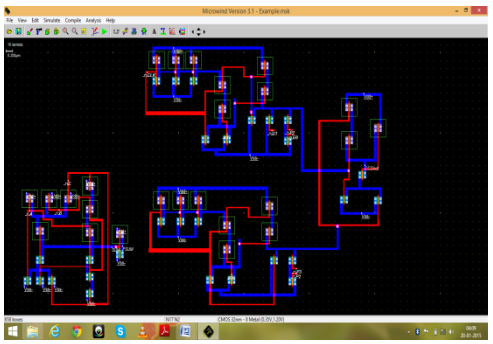


Figure 8:Layout diagram for 8-bit Comparator using CD Logic.

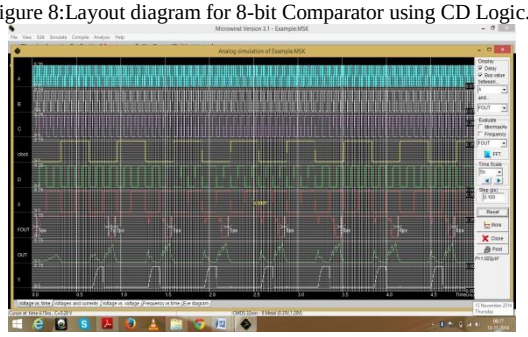


Figure 9: Simulation Results of 8-bit Comparator using CD Logic.

Table 2: SR-Latch Performance Comparison

	Static			Dynamic			CD		
Technology	65nm	45nm	32nm	65nm	45nm	32nm	65nm	45nm	32nm
Delay(ps)	63	58	43	63	55	31	27	14	10
Power(μW)	4.725	3.437	3.25	4.925	4.254	3.782	3.177	2.25	1.269
Power-delay product(PDP)(fJ)	0.297	0.199	0.139	0.310	0.233	0.117	0.085	0.031	0.012
Energy-delay product(EDP) (fJps)	2.97	1.90	1.39	3.10	2.33	1.17	0.85	0.31	0.12

Table 3: 8-bit Comparator Performance Comparison

	Static			Dynamic			CD		
Technology	65nm	45nm	32nm	65nm	45nm	32nm	65nm	45nm	32nm
Delay(ps)	148	78	65	78	61	49	37	28	15
Power(μW)	6.254	7.261	7.037	7.943	6.769	5.506	4.144	3.25	2.269
Power-delay product(PDP)(fJ)	0.925	0.566	0.457	0.619	0.412	0.269	0.153	0.091	0.034
Energy-delay product(EDP) (fJps)	9.25	5.66	4.57	6.19	14.12	2.69	1.53	0.91	0.34

CONCLUSION

A new high-performance logic style with CD characteristic and self-reset circuitry was explored to implement complex circuits. The pre-evaluated feature of CD logic makes it predominantly suitable in a circuit block where a exclusive critical path exists and performance is the primary concern. Using CD logic SR-Latch and 8-bit Comparator are designed in 32-nm general-purpose CMOS Technology and it is compared with different CMOS technologies like 65nm and 45nm. It is found that, in 32nm technology CD logic delay is less by 96% and 68% over static and dynamic logics, respectively. Simulation results of SR-Latch and 8-bit Comparator shows that 52% and 46% faster than the static and dynamic logics.

REFERENCES

[1] Pierce Chuang, David Li, Manoj Sachdev, "Constant Delay Logic Style" in *IEEE Trans. VLSI Systems*, vol. 21, no.3, March 2013.
[2] R. Zimmermann and W. Fichtner, "Low-power logic styles: CMOS versus pass-transistor logic," *IEEE J. Solid-State Circuits*, vol. 32, no. 7, pp. 1079–1090, Jul. 1997.
[3] N. Goncalves and H. De Man, "NORA: A racefree dynamic CMOS technique for pipelined logic structures," *IEEE J. Solid-State Circuits*, vol. 18, no. 3, pp. 261–266, Jun. 1983.
[4] C. Lee and E. Szeto, "Zipper CMOS," *IEEE Circuits Syst. Mag.*, vol. 2, no. 3, pp. 10–16, May 1986.
[5] R. Rafati, S. Fakhraie, and K. Smith, "A 16-bit barrel-shifter implemented in data-driven dynamic logic (*D*3L)," *IEEE Trans. Circuits Syst I, Reg. Papers*, vol. 53, no. 10, pp. 2194–2202, Oct. 2006.
[6] F. Frustaci, M. Lanuzza, P. Zicari, S. Perri, and P. Corsonello, "Lowpower split-path data-driven dynamic logic," *Circuits Dev. Syst. IET*, vol. 3, no. 6, pp. 303–312, Dec. 2009.
[7] N. Weste and D. Harris, *CMOS VLSI Design: A Circuits and Systems Perspective*, 4th ed. Reading, MA: Addison Wesley, Mar. 2010.
[8] K. Bernstein, *High Speed CMOS Design Styles*, 1st ed. New York: Springer-Verlag, Aug. 1998.
[9] S. Mathew, R. K. Krishnamurthy, M. A. Anders, R. Rios, K. R. Mistry, and K. Soumyanath, "Sub-500-ps 64-b ALUs in 0.18-μm SOI/bulk CMOS: Design and scaling trends," *IEEE J. Solid-State Circuits*, vol. 36, no. 11, pp. 318–319, Nov. 2001.
[10] S. Mathew, M. Anders, R. Krishnamurthy, and S. Borkar, "A 4 GHz 130 nm address generation unit with 32-bit sparse-tree adder core," in *VLSI Circuits Dig. Tech. Papers Symp.*, 2002, pp. 126–127.
[11] S. K. Mathew, M. A. Anders, B. Bloechel, T. N. Krishnamurthy, and S. Borkar, "A 4-GHz 300-mW 64-bit integer execution ALU with dual supply voltages in 90-nm CMOS," *IEEE J. Solid-State Circuits*, vol. 40, no. 1, pp. 44–51, Jan. 2005.
[12] S. Wijeratne, N. Siddaiah, S. Mathew, M. Anders, R. Krishnamurthy, J. Anderson, S. Hwang, M. Ernest, and M. Nardin, "A 9 GHz 65 nm intel pentium 4 processor integer execution core," in *IEEE Int. Solid-State Circuits Conf. ISSCC Dig. Tech. Papers*, San Francisco, CA, Feb. 2006, pp. 353–365.
[13] *Effort: Designing Fast CMOS Circuits* [Online]. Available: http://amazon.com/o/ASIN/1558605576/
[14] S. Kiaei, S.-H. Chee, and D. Allstot, "CMOS source-coupled logic for mixed-mode VLSI," in *Proc. IEEE Int. Circuits Syst. Symp.*, New Orleans, LA, May 1990, pp. 1608–1611.

[15] L. McMurchie, S. Kio, G. Yee, T. Thorp, and C. Sechen, "Output prediction logic: A high-performance CMOS design technique," in *Proc. Comput. Des. Int. Conf.*, Austin, TX, 2000, pp. 247–254.
[16] K. H. Chong, L. McMurchie, and C. Sechen, "A 64 b adder using selfcalibrating differential output prediction logic," in *IEEE Int. Solid-State Circuits Conf. Dig. Tech. Papers*, San Francisco, CA, Feb. 2006, pp. 1745–1754.
[17] V. Navarro-Botello, J. A. Montiel-Nelson, and S. Nooshabadi, "Low power arithmetic circuit in feedthrough dyanmic CMOS logic," in *Proc. IEEE Int. 49th Midw. Symp. Circuits Syst.*, Aug. 2006, pp. 709–712.
[18] V. Navarro-Botello, J. A. Montiel-Nelson, and S. Nooshabadi, "Analysis of high-performance fast feedthrough logic families in CMOS," *IEEE Trans. Circuits Syst. II, Exp. Briefs*, vol. 54, no. 6, pp. 489–493, Jun. 2007.

V. Divya Vani received Bachelor's degree from Y.S.R. Engineering College of Yogi Vemana University, Proddatur, Kadapa (Dist), in the year 2012 and currently persuing M. Tech in Embedded Systems at RGM College of Engineering and Technology, Nandyal, Kurnool (Dist), Andhra Pradesh. Her areas of interests are Embedded System Design, Microcontrollers, Low Power VLSI and Digital System Designs.