A Differential Design for C-elements and NCL Gates

Steven Yancey and Scott C. Smith (Senior Member IEEE)
Department of Electrical Engineering
University of Arkansas
Fayetteville, Arkansas, USA
swyancey@uark.edu and smithsco@uark.edu

***Abstract* — This paper demonstrates the performance, area and supply voltage scaling advantages of a Differential Cascode Voltage-Switch Logic (DCVSL)-like design over previous methods for designing C-elements. The DCVSL-like method is then applied to the design of arbitrary NULL Convention Logic (NCL) gates, which have hysteresis state-holding capability.**

Keywords – NULL Convention Logic (NCL); Differential Cascode Voltage-Switch Logic (DCVSL); Threshold Gate; C-element

I. INTRODUCTION

Asynchronous techniques such as NULL Convention Logic (NCL) [1] enjoy many advantages due to the elimination of clock trees and most timing constraints, but the tradeoffs include area and performance penalties associated with dual-rail signaling, completion detection, and handshaking between stages.

NCL circuits are comprised of 27 fundamental gates [2], which constitute the set of all functions consisting of four or fewer variables. The primary type of NCL gate is the threshold gate, denoted as TH*mn*, where $1 \le m \le n$. TH*mn* gates have *n* inputs. At least *m* of the *n* inputs must be asserted before the output will become asserted. NCL threshold gates are designed with *hysteresis* state-holding capability such that all asserted inputs must be deasserted before the output will be deasserted. Therefore, a TH*nn* gate is equivalent to an n-input C-element [3] and a TH1*n* gate is equivalent to an n-input OR gate.

Each NCL gate has a distinct Boolean SET function, which determines when the gate transitions from 0→1 (e.g., $A \cdot B$ for a 2-input C-element), but all gates with the same number of inputs have the same RESET function (i.e., all inputs are 0), which determines when the gate transitions from 1→0 (e.g., $\overline{A} \cdot \overline{B}$ for a 2-input C-element). Thus, the RESET function is not the complement of the SET function (except for TH1*n* gates), such that hysteresis functionality is required to hold the current state of the gate when neither SET nor RESET are active.

Large C-elements, needed for completion detection, can be implemented as trees of smaller C-elements, such as TH22, TH33, or TH44 gates [4]. The size and performance of large C-elements have a significant impact on asynchronous designs, which has led to several efforts to optimize C-elements [5-9].

Section II presents the previous work on C-element design, while Section III develops a differential design method for implementing C-elements, and then expands the method to the design of arbitrary NCL gates. Section IV compares the differential C-element design method to the previous work [5-9] and arbitrary differential NCL gates to their static and semi-static CMOS implementations. Section V presents conclusions and areas for future work.

II. PREVIOUS WORK

A. Semi-Static CMOS C-element

A semi-static CMOS C-element [8] implements the RESET function as a pull-up network (PUN) and the SET function as a pull-down network (PDN), which is then inverted to produce the output. A weak feedback inverter is used to achieve the hysteresis state-holding functionality. Fig. 1 depicts a 2-input semi-static C-element.

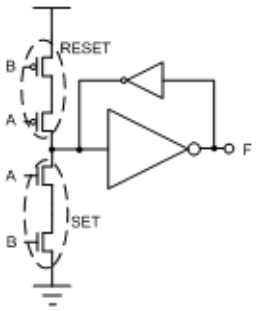

Figure 1. Semi-Static CMOS TH22 Gate.

Correct sizing is necessary for this type of gate to function correctly – the PUN and PDN must be strong enough to overcome the contention current from the weak feedback inverter. Another undesirable characteristic of the semi-static gate is that as supply voltage is reduced, it will cease to operate correctly sooner than the static design due to the relative strengths of the PUN, PDN, and weak inverter being affected by the reduced supply voltage.

B. Static CMOS C-element

A static CMOS C-element [7] utilizes the same SET and RESET circuitry as its semi-static counterpart, but replaces the weak feedback inverter with a second PUN and PDN, which include feedback from the output, to provide the Hold0 and Hold1 functionality, respectively. Note that $Hold0 = \overline{SET}$ and $Hold1 = \overline{RESET}$. Fig. 2 depicts a 2-input static C-element.

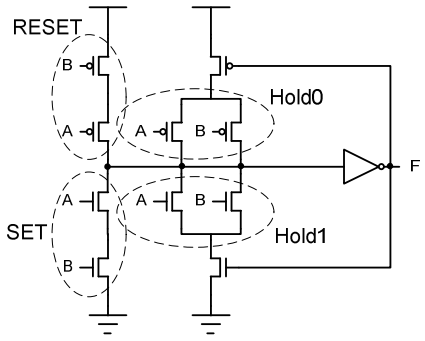

Figure 2. Static CMOS TH22 Gate.

C. Symmetrical CMOS C-element

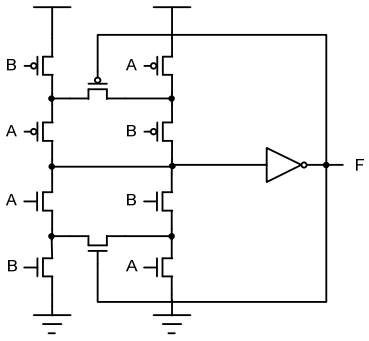

Figure 3. Symmetrical CMOS TH22 Gate.

A symmetrical C-element [9] has the best performance amongst 2-input single-ended CMOS implementations [6], but the topology is limited to 2 inputs.

D. NAND/NOR C-element

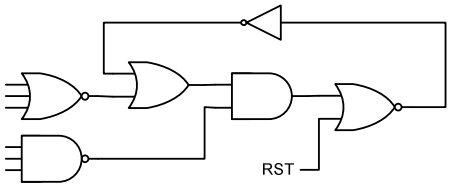

Figure 4. NAND/NOR C-element Structure.

This NAND/NOR C-element structure [5] is used for large C-elements as an alternative to trees of smaller C-elements. In comparison to trees of C-elements, savings result from removing the state-holding function from all but the root node (final gate) of the tree. The leading NAND and NOR networks are expanded to whatever number of inputs are required.

III. Differential C-element and NCL Gate Design

The differential design of state-holding gates developed herein is most similar to Differential Cascode Voltage-Switch Logic (DCVSL) [10]. Differential logics use two output rails so that a function and its complement are both available. In these dual-rail designs, the PUN that normally complements a PDN in a single-rail CMOS gate is implemented instead as a PDN on the second rail. Whichever PDN is active pulls down its own rail directly, and pulls up the complementary rail through a single PMOS transistor. The main advantage of differential designs is that PMOS area is minimized and all functions are realized in NMOS, which results in both a faster and smaller design.

In the case of DCVSL, both PDNs are grounded and all inputs are applied to gates in the PDNs. There is often overlap in the F and $\overline{F}$ networks, so they are commonly merged to further reduce transistor count and the PDNs are usually shown as a single N-network, as in Fig. 5.

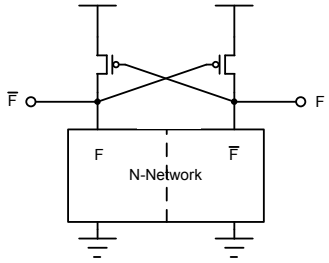

Figure 5. DCVSL Gate Architecture.

Adding hysteresis state-holding functionality to the basic DCVSL gate only requires the addition of two NMOS hold transistors. Once hysteresis has been added, the two PDNs no longer need to be complements of each other, so they may now be the SET and RESET functions of an NCL gate. Since any NCL SET function is composed only of uncomplemented inputs, and the RESET function is composed of all complemented inputs, there can never be any overlap of the two and they are better shown as two separate PDNs. This new Differential NCL (DNCL) gate architecture is shown in Fig. 6.

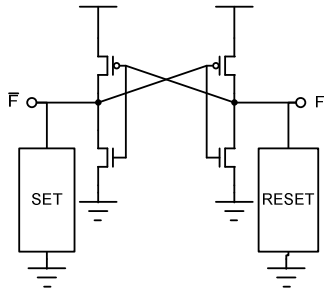

Figure 6. Differential NCL (DNCL) Gate Architecture.

Note that by forming a cross-coupled inverter pair, this gate design bears some resemblance to the semi-static gate; however, the DNCL inverters are balanced rather than one being weak. This also results in the DNCL gate working well past the point that the semi-static gate fails as the supply voltage is reduced. The low-voltage performance of the DNCL gate is comparable to that of the static CMOS gate.

Another characteristic of DNCL gates is that whenever a gate switches, the falling rail always switches first, so the outputs always transition 10→00→01→00→10. In a fully DNCL design this guarantees that input switching will never cause one rail to become asserted without first fully deasserting the other rail. This minimizes contention current during switching and improves average propagation delay throughout the gate network.

Fig. 7 shows a DNCL TH22 gate. The TH44 gate is identical except each PDN is a 4-transistor chain.

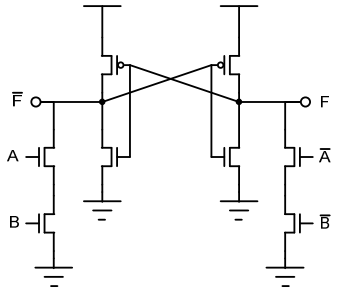

Figure 7. DNCL TH22 Gate.

IV. COMPARISONS

The DNCL gate design was compared to the static and semi-static CMOS designs in a number of simulations. Low-V_t transistor models from the IBM CMRF8SF library (130nm MTCMOS) were used. In the first set of tests, two versions of each of the static and semi-static CMOS TH22 designs were compared to the DNCL TH22 design at the nominal operating voltage of 1.2V. "Small" versions of the two CMOS designs were minimized for area, while "fast" versions used adjusted transistor sizings to improve performance. In the case of the small static CMOS design this meant all minimum-width (unit) transistors. Because of the sizing requirements of the semi-static design, the small version used 3x-width transistors in the PUN and a 4x-length NMOS transistor for the weak inverter. The fast static CMOS gate used 3x for the PUN, 2x for the PDN, and 2x for all other PMOS transistors. The fast semi-static gate used 4x for the PUN, 2x for the PDN, and 2x-length for the weak inverter NMOS transistor. The DNCL gate was sized with 2x-width transistors for both PDNs and unit transistors for the cross-coupled inverters.

Each gate variant was tested individually and in a 2-deep and a 3-deep tree. An input pattern tested for the fastest-case single-input-to-output switching (A-input tree-edge switched last), slowest-case single-input-to-output (B-input tree-edge switched last), and the worst-case (all inputs switched simultaneously). Fig. 8 illustrates the test tree method.

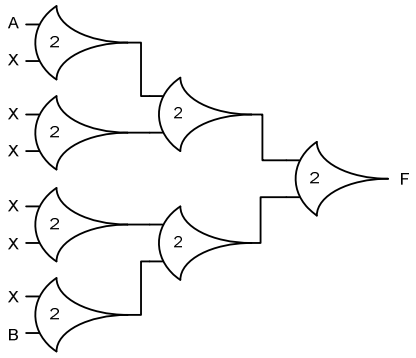

Figure 8. 3-Deep test tree with fast(A) and slow(B) edges.

The second set of tests compared fast versions of static and semi-static TH44 gates to the DNCL TH44 gate. The static gate used 6x for the PUN, 3x for the PDN, and unit transistors otherwise. The semi-static gate used progressive sizing on the PUN and PDN (660-480nm and 240-160nm respectively). The DNCL TH44 gate used 1-2-2-3 sizing on the PDNs.

The third set of tests repeated the testing of TH22 gates, but at reduced operating voltages. At 400mV, the semi-static gates no longer worked, but the static and DNCL gates both functioned correctly. When the supply voltage was further reduced, the fast static gate worked correctly down to 110mV, the small static gate down to 90mV, and the DNCL gate down to 80mV.

Table 1 shows the simulation results for these three sets of tests. Area is in terms of number of unit transistors, and propagation delay is in picoseconds.

V_{DD}	Gate	Logic	Area	t_{pr}(fast)	t_{pf}(fast)	t_{pr}(slow)	t_{pf}(slow)	t_{pr}(all)	t_{pf}(all)	t_p(avg)
1.2V	TH22	Static(small)	12	92	117	92	123	91	106	104
1.2V	TH22	Static(fast)	22	65	81	64	85	71	87	75
1.2V	TH22	Semi-Static(small)	15	139	134	129	138	151	145	139
1.2V	TH22	Semi-Static(fast)	19	67	122	64	130	73	135	98
1.2V	TH22	DNCL	12	68	68	71	70	76	76	71
1.2V	TH44	Static(fast)	49	85	107	99	147	167	215	136
1.2V	TH44	Semi-Static(fast)	24	137	87	170	151	270	175	165
1.2V	TH44	DNCL	20	90	88	110	109	113	114	104
0.4V	TH22	Static(small)	12	1767	2765	1867	2890	1764	2565	2270
0.4V	TH22	Static(fast)	22	1146	1540	1205	1685	1293	1670	1423
0.4V	TH22	DNCL	12	1457	1445	1592	1440	1626	1615	1529

Table 1. Simulation Results

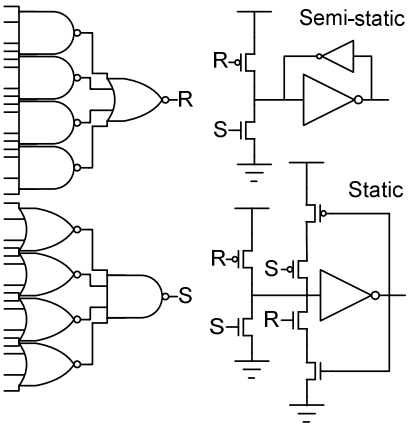

Figure 9. Semi-Static and Static NAND-NOR C-elements.

For the final test, the static and semi-static 16-input C-elements shown in Fig. 9 were compared with a 2-deep tree of DNCL TH44 gates. These two larger C-elements are implemented using NAND and NOR tree structures and a single hysteresis element in a method similar to that proposed by Wuu [5]. Speed-optimized 4-input NOR and NAND gates were used to build a NOR-NAND tree driving the pull-up transistor and a NAND-NOR tree driving the pull-down transistor.

Both 16-input C-elements were faster than the DNCL TH44 tree, with an average propagation delay of 144ps for the static gate, 152ps for the semi-static gate, and 208ps for the TH44 tree. This speed advantage was at the cost of more than double the area though (235 unit transistors for the semi-static gate vs. 100 for the TH44 tree). In subthreshold operation, the DNCL circuit functions correctly down to 90mV, while the static gate fails below 140mV and the semi-static gate fails below 490mV.

V. Conclusions and Future Work

The simulation results for TH22 gates show that at 1.2V, the DNCL design has the lowest average propagation delay (71ps) while being tied for least area (12 unit transistors) with the much-slower static CMOS gate. As the operating voltage is reduced, the DNCL gate slows down slightly more than the speed-optimized static CMOS gate; however, in subthreshold operation the DNCL gate still functions correctly down to 80mV after all CMOS gate designs have failed.

For TH44 gates, the DNCL implementation is also both faster and smaller than the two CMOS designs. Additionally, a single DNCL TH44 gate is faster and smaller than a tree of three DNCL TH22 gates (107ps vs. 142ps, and 20 vs. 36 unit transistors); so, it would be preferable to use DNCL TH44 gates as much as possible for large completion trees.

The DNCL TH44 tree was slower than the large NAND-NOR C-elements; however, the tradeoff for this increased speed was more than double the area. The tests that were run at reduced operating voltages indicate that DNCL gates may be particularly well suited for subthreshold operation; however, the designs tested here were not optimized specifically for low-voltage performance, so further evaluation in this area is warranted.

While this work focused on comparing DNCL C-elements and C-element structures used in completion detection, future work is needed to compare larger combinational logic circuits composed of DNCL gates vs. static and semi-static gates.

References

[1] K. M. Fant, S. A. Brandt, "NULL Convention Logic: A Complete and Consistent Logic for Asynchronous Digital Circuit Synthesis," *International Conference on Application Specific Systems, Architectures, and Processors*, pp. 261–273, 1996.

[2] Gerald E. Sobelman and Karl M. Fant, "CMOS Circuit Design of Threshold Gates with Hysteresis," *IEEE International Symposium on Circuits and Systems (II)*, 1998, pp. 61-65.

[3] D. E. Muller, "Asynchronous Logics and Application to Information Processing," in *Switching Theory in Space Technology*, Stanford University Press, 1963, pp. 289-297.

[4] S. C. Smith, R. F. DeMara, J. S. Yuan, M. Hagedorn, and D. Ferguson, "Delay-Insensitive Gate-Level Pipelining," *Integration, the VLSI Journal*, vol. 30, no. 2, pp. 103-131, 2001.

[5] T. -Y Wuu and S. B. K. Vrudhula, "A design of a fast and area efficient multi-input Muller C-element," *IEEE Transactions on VLSI Systems*, vol. 1, no. 2, pp. 215-219, 1993.

[6] M. Shams, J. C. Ebergen, and M. I. Elmasry, "Modeling and comparing CMOS implementations of the C-element," *IEEE Transactions on VLSI Systems*, vol. 6, no. 4, pp. 563-567, 1998.

[7] I. E. Sutherland, "Micropipelines," *Communications of the ACM*, vol. 32, no. 6, pp. 720-738, 1989.

[8] A. J. Martin, "Formal progress transformations for VLSI circuit synthesis," in *Formal Development of Programs and Proofs*, Addison-Wesley, 1989, pp. 59–80.

[9] K. Van Berkel, "Beware the Isochronic Fork," *Integration, the VLSI Journal*, vol. 13, no. 2, pp. 103-128, 1992.

[10] L.G. Heller, W.R. Griffin, J.W. Davis, and N.G. Thoma, "Cascode Voltage Switch Logic: A Differential CMOS Logic Family," *ISSCC Dig. Tech. Papers*, pp. 16-17, 1984.