This document is an author-formatted work. The definitive version for citation appears as:

R. F. DeMara, A. Kejriwal, and J. R. Seeber, "Feedback Techniques for Dual-Rail Self-Timed Circuits," in *Proceedings of the 2004 International Conference on VLSI (VLSI-04)*, pp. 458 – 464, Las Vegas, Nevada, U.S.A., June 21 – 24, 2004.

Feedback Techniques for Dual-rail Self-timed Circuits

Ronald F. DeMara Amit Kejriwal Jude Seeber

Department of Electrical and Computer Engineering
University of Central Florida
Box 162450, Orlando, FL 32816-2450
Tel: 407-823-5916 E-mail: demara@mail.ucf.edu

Keywords: Asynchronous Circuit, Dual-rail Logic, NCL, Timer Register, Toggle Element

Abstract

Design techniques for time and space optimization of NCL feedback circuits are developed and compared. First, a 3-Register Stage method is employed to circulate DATA and NULL wavefronts required to realize feedback in 4-bit binary timer case-study. While modular and adaptable, this approach requires a significant gate count to realize the feedback circuit that comprises 75% of the total gates required. The second methodology aims at reducing the feedback overhead by using the state-maintaining capacity inherent with each threshold logic gate's built-in hysteresis behavior. This methodology employs two characteristics: the circuit preserves its present state; and it keeps track of the number of requests for DATA so that it can determine the appropriate next state. This embedded approach can reduce the feedback gate count by nearly 50% and also DATA-to-DATA cycle time by 31% depending on the feedback scheme used. Finally, the above-explained methodologies are assessed in terms of their design tradeoffs.

1. Introduction

NULL Convention Logic (NCL) [1] offers a self-timed logic paradigm where control is inherent with each datum. NCL follows the so-called "weak conditions" of Seitz's delay-insensitive signaling scheme [2]. As with other self-timed logic methods discussed herein, the NCL paradigm assumes that forks in wires are *isochronic* [3, 4]. The origins of various aspects of the paradigm, including the NULL (or spacer) logic state from which NCL derives its name, can be traced back to Muller's work on speed-independent circuits in the 1950s and 1960s [5]. NCL differs from the above-mentioned methods in that they only utilize one type of state-holding gate, the *C-element* [5]. On the other hand, all NCL gates are state-holding as described below.

As shown in Figure 1, a dual-rail signal in NCL consists of two wires, Rail_0 and Rail_1, which may assume any value from the set {DATA0, DATA1, NULL}. The DATA0 state (Rail_0 = 1, Rail_1 = 0) corresponds to a Boolean logic 0, the DATA1 state (Rail_0 = 0, Rail_1 = 1) corresponds to a Boolean logic 1, and the NULL state (Rail_0 = 0, Rail_1 = 0) meaning that the value is not yet available. The two rails are mutually exclusive, so that both rails can never be asserted simultaneously; this state is defined as an illegal state.

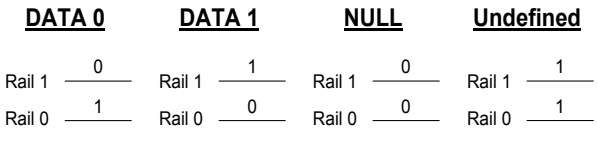


Figure 1: NCL DATA and NULL states.

NCL circuits can be composed using a methodology that utilizes NCL *threshold gates* with *hysteresis* [6,7]. One type of threshold gate is the *THmn gate*, where $1 \le m \le n$, as depicted in Figure 2. A THmn gate corresponds to an operator with at least m signals asserted as its set condition and all signals de-asserted as its reset condition. THmn gates have *n* inputs. At least *m* of the *n* inputs must be asserted before the output will become asserted. Because threshold gates are designed with hysteresis, all asserted inputs must be de-asserted before the output will

be de-asserted. Hysteresis is used to provide a means for monotonic transitions and a complete transition of multi-rail inputs back to a NULL state before asserting the output associated with the next wavefront of input data. In a THmn gate, each of the *n* inputs is connected to the rounded portion of the gate. The output emanates from the pointed end of the gate. The gate's threshold value, *m*, is written inside of the gate.

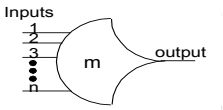


Figure 2: Threshold Gate with Hysteresis

Karl Fant [1] demonstrated an NCL-based binary-encoded up-counter design based on Ring Oscillator circuits. For instance, Figure 3 depicts an arrangement of *w* Ring Oscillators to generate the required bit sequences for a *w*-bit-wide count value *C* where *w*=3. The calculation of each bit C_i is computed independently upon assertion of up-count control. Specifically, whenever the value of *C* is incremented, the required bit values for the corresponding sequence of C_i for $1 \le i \le w$ are re-circulated within their respective ring. This approach generates the output sequence for each bit C_i independently without requiring input from any elements that generate C_j for $j \neq i$.

Although Figure 3 realizes the bit sequence for a binary-encoded up-counter, this approach is readily generalized to arbitrary sequences using the corresponding arrangement of feedback shift registers. However, when such circuits employing feedback are designed in NCL, a primary concern is maintaining the proper flow of DATA and NULL wavefronts such that neither wavefront is prematurely overwritten.

A straightforward solution is to insert register stages that delineate the completion boundaries of the asynchronous subcircuits. However, the inclusion of additional register stages also increases the transistor count and the DATA-to-DATA cycle time. Thus, this paper develops alternative methods to provide NCL space and time optimizations in the presence of feedback while maintaining delay-insensitivity.

This paper is organized into 6 sections. In Section 2, a baseline NCL 4-bit timer design using a *3-Register Stage* feedback circuit is presented. Section 3 defines the characteristics to be optimized over the baseline design. The consideration of these characteristics leads to development of a self-contained *toggle element* using NCL threshold gates in Section 4. Section 5 utilizes the toggle element in the 3 alternative design methods which are simulated and compared using a case study where *w*=4. Applications requiring *w*>4 can be accommodated by extending the proposed techniques or cascading 4-bit stages. Section 6 identifies conclusions and potential implementation considerations.

2. Three-Register Stage Feedback

2.1 Design Approach

The conventional 3-Register stage approach for controlling feedback includes the required up-counter combinational logic and 3 register stages. This feedback network is shown in Figure 4. The use of three stages accommodates re-circulation of the DATA and NULL wavefronts while prohibiting improper overwriting or deadlock scenarios possible with less than 3 register stages [1].

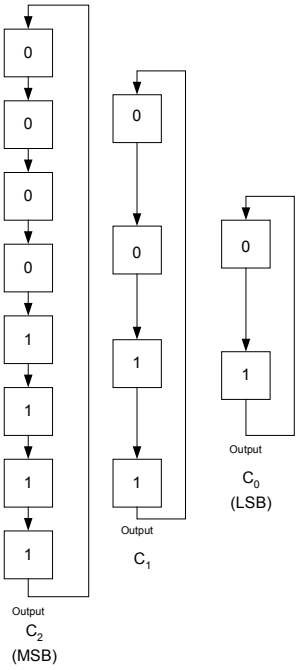


Figure 3: Ring Oscillator 3-bit Counter/Timer

The output of the combinational circuit is fed to the input of the first register while the output of the third register is fedback to the combinational circuit. Initially, the register stages R1, R2, and R3 are initialized to DATA, NULL, and NULL, respectively. R1 will request for NULL, R2 will request for DATA, and R3 will also request for DATA. This does not create the possibility of a lockup scenario since the required inputs are available for R1, R2, and R3. Table 1 lists the flow of wavefronts and signaling sequences for this configuration.

		Initial Operands	Transmit Condition (if Input = Request)	Register Operands	Transmit Condition (if Input = Request)	Register Operands	Transmit Condition (if Input = Request)	Register Operands
R1	Input	NULL	Hold state	NULL	Transmit State	NULL	Transmit State	DATA
	Output	DATA		DATA		NULL		NULL
	Request	DATA		NULL		NULL		NULL
R2	Input	DATA	Transmit State	DATA	Transmit State	NULL	Hold state	NULL
	Output	NULL		DATA		DATA		DATA
	Request	DATA		DATA		DATA		NULL
R3	Input	NULL	Transmit State	DATA	Hold state	DATA	Transmit State	DATA
	Output	NULL		NULL		NULL		DATA
	Request	NULL		NULL		DATA		DATA

Table 1: 3-Register Signaling Sequence

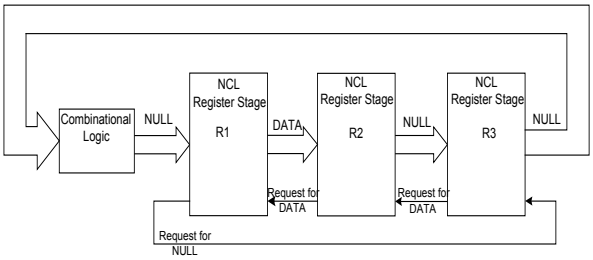


Figure 4: 3-Register Stage Feedback Initialization

2.2 Block Diagram

As shown in Figure 5, an NCL 4-bit timer/counter has the functionality of incrementing the present value of the Timer upon the arrival of each DATA cycle. The Timer Increment synchronization signal for the request for DATA or NULL is controlled externally by asserting the request input, K_i, of the register [1]. The convention used is that when K_i is asserted then it denotes a request for DATA. When it K_i is de-asserted then it denotes a request for NULL. The *w*-bit count value *C* indicates the current timer value and K_o is required to determine whether the DATA should be requested or NULL should be requested. Reset sets the timer to the initial condition and zeroes the count value.

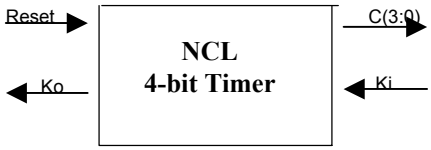


Figure 5: 4-bit Timer block diagram

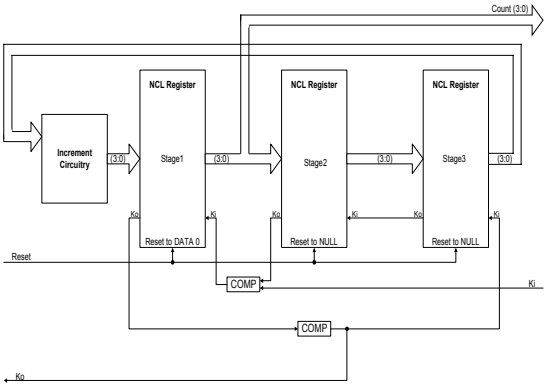


Figure 6: 3-Register Stage 4-Bit Timer

2.3 NCL Timer with 3-Register Stage Feedback

As shown in Figure 6, the Reset signal sets the registers to the initial values of DATA, NULL, NULL for register stage 1, register stage 2, register stage 3 respectively along with interconnections to the completion logic labeled COMP. The request, K_i, for DATA or NULL, is provided as the input to the circuit to regulate the timer operation. An AND gate is used to combine K_i with the request from register stage 2. Thus, unless they match, the request to register stage 1 will not change. This synchronizes the timer with the external control signal. The COMP logic detects the completion of the DATA or NULL cycle for all bits in the corresponding stage of the circuit. The output is taken from the first register, though it could be similarly obtained from the output of any register stage.

2.4 4-Bit Increment Circuit

Using a K-Map for both Rail1 and Rail0 individually, the increment-by-1 circuit can be reduced to the following equations:

$$Y(0).rail1 = X0.rail0$$
$$Y(0).rail0 = X0.rail1$$
$$Y(1).rail1 = X(1).rail0 \cdot X(0).rail1 + X(1).rail1 \cdot X(0).rail0$$
$$Y(1).rail0 = X(1).rail0 \cdot X(0).rail0 + X(1).rail1 \cdot X(0).rail1$$
$$Y(2).rail1 = X2.rail1\ (X1.rail0 + X0.rail0) + X2.rail0 \cdot X1.rail1 \cdot X0.rail1$$
$$Y(2).rail0 = X2.rail0\ (X1.rail0 + X0.rail0) + X2.rail1 \cdot X1.rail1 \cdot X0.rail1$$
$$Y(3).rail1 = X(3).rail1(X(1).rail0 + X(2).rail0 + X(0).rail0) + X(3).rail0 \cdot X(2).rail1 \cdot X(1).rail1 \cdot X(0).rail1$$
$$Y(3).rail0 = X(3).rail0\ (X(1).rail0 + X(2).rail0 + X(0).rail0) + X(3).rail1 \cdot X(2).rail1 \cdot X(1).rail1 \cdot X(0).rail1$$

The equations are realized using the circuit in Figure 7.

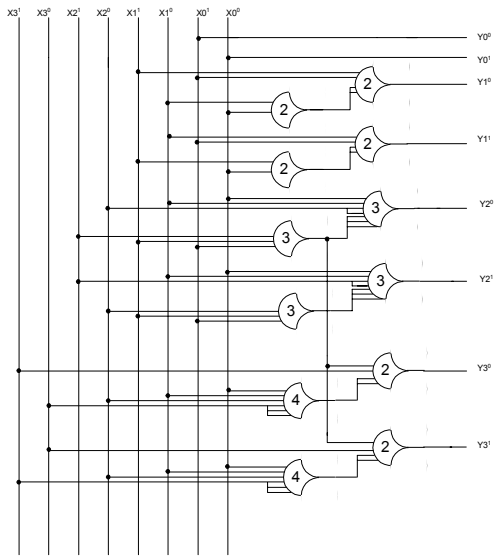


Figure 7: 4-Bit Timer Increment Circuit

Table 2: Simulation Results for 3-Register Timer

Gate Count			Transistor Count		T_{DD}
Increment	Register	Total	Increment	Total	
12 TH22x0 (2) TH33x0 (2) TH23w2x0 (4) TH34w32x0 (2) TH44w3x0 (2)	41 Threshold Gates Used for the Circuit	53	152	608	5.05 ns

Table 2 shows design parameters and simulation results for the 3-Register stage 4-bit timer. Results are based on simulations performed using SPICE in a Cadence design environment with a 0.18µ technology library at 2.3 Volts. T_{DD} denotes the DATA-to-DATA cycle time, which for this configuration was 5.05 ns. Observe that the gate count of the registers is 3.41-fold larger than the gate count of increment circuit itself. Thus, just to achieve the feedback regulating the DATA and NULL wavefronts, 41 overhead gates are required in addition to the gates needed by the increment circuit. The overhead not only increases the cost but also increases T_{DD}. These are the primary motivation for the alternative designs developed in the remainder of the paper.

3. NCL Timer Without Feedback Registers

Instead of inserting register stages for controlling feedback, an increment circuit is developed with embedded feedback using the hysteresis in the NCL threshold gates themselves. Embedded registration within the increment circuit requires that the increment circuit:

1. retains its present state, and
2. keeps the track of the number of request for DATA to determine the appropriate next state.

These two objectives can both be achieved if each bit C_i is generated by an appropriate state machine. The state machine that generates C_i has 2^i distinct states of which half will output a single bit with a value of 0 and the other half will output a single bit with a value of 1. In particular, the state machine outputs a string 0's that is 2^i long and then a string 1's that is 2^i long. The state machine then resets and repeats its cycle, just as in the case of the Ring Oscillator design in Figure 3.

Without using Feedback Registers, additional gates are used to detect when the output from C_i transitions from 1 → 0 to perform the reset of the state machine generating C_i. It simultaneously triggers the transition to the next state in the state machine that generates C_{i+1}. By cascading these state machines, the complete sequence for all bits of C are generated while maintaining the overall integrity of the DATA and NULL wavefronts.

4. Toggle Elements for Embedded Registration

The basic building block for embedded registration in the NCL timer is the state machine that generates C_0. This circuit generates the sequence 0 → 1 → 0 → 1 → 0 … and thus acts as a Toggle Element. The NCL toggle element must consider both the NULL wavefront and the DATA wavefronts. In particular, it will generate a NULL → DATA0 → NULL → DATA1 cycle. Instead of feedback using registers, if the present state is retained internally along with the count of requests for DATA that are received, then the required behavior for a binary toggle element is achieved.

4.1 Toggle Element with Reset

The circuit shown in Figure 8 can store the present state and also provide a mechanism to maintain the number of requests for DATA. The sequence of timing signals is listed in Table 3. After the cycle is complete, the circuit is required to begin a new cycle. Since the last output is a DATA output, the next is the request for NULL, which indicates K_i=0. The reset circuitry in Figure 8 will be asserted since R_0^1=1 and K_i=0.

Table 3: Sequence of Toggle Element with Reset

	initial state	state1	state2	state3	state4
T1	Null	1	1	1	reset
T2	Null	0	1	1	reset
T3	Null	0	0	1	reset
T5(Reset)	Null	0	0	0	1
R00	Null	1	0	0	0
R01	Null	0	0	1	1
Ki	DATA	DATA	NULL	DATA	NULL

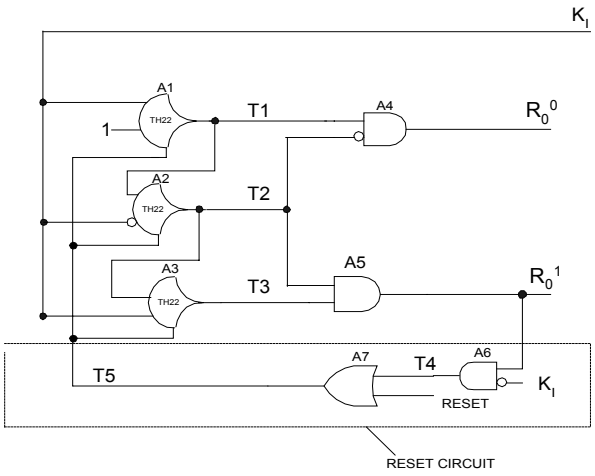


Figure 8: Toggle Element with Reset

This circuit has an exposure to the case that the reset signal to the A2 gate is delayed while the NULL cycle is complete and the DATA wavefront is arriving. Thus, this circuit should be designed as a module and needs to take into consideration the wire and gate propagation delays associated with the reset circuitry.

4.2 Toggle Element without Reset

Instead of using the above-explained methodology for the Toggle element, an alternative methodology shown in Figure 9 can be used. Instead of always keeping one of the inputs asserted, the compliment of one of the output rails is feedback using the bubble inverted input on the upper TH22 gate labeled A1 in Figure 9.

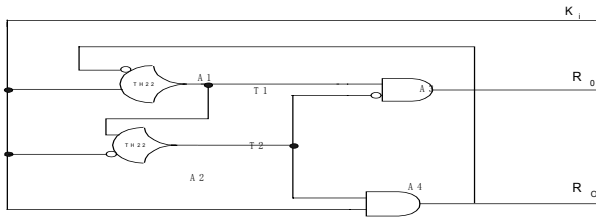


Figure 9: Toggle Element without Reset

This methodology eliminates the race condition involved with generating the reset signal to set the circuit back to its initial value once it completes each cycle. The TH22 gates A1 and A2 are initialized to NULL and will follow the sequence listed in Table 4.

Table 4: Sequence of Toggle Element without Reset

	initial state	state1	state2	state3	state4
T1	Null	1	1	1	reset to NULL
T2	Null	0	1	1	reset to NULL
R00	Null	1	0	0	0
R01	Null	0	0	1	1
Ki	DATA	DATA	NULL	DATA	NULL

After each cycle is complete the circuit is required to start a new cycle. Since last output was a DATA output, next is the request for NULL, which has K_i=0. At this time both the inputs to gate A1 are de-asserted, as R_0^1=1 so its compliment is equal to 0 and K_i=0, which caused the gate A3 to be de-asserted and the NULL output is achieved.

However, this circuit is not entirely delay-insensitive because NCL circuits utilizing feedback without intervening asynchronous registers are susceptible to a *feedback-lagging* race condition. In this case, the time at which the feedback to the A1 threshold gate might arrive being later than the request for NULL should be considered.

5. NCL Timer with Embedded Registration

5.1 Version A1: Reset-equipped Gates with Combinational Rollover-Detect

Bit C_0 is generated by the Toggle Element directly. Bit Bit C_1 is generated in Figure 10 according to Table 5. The information that C_0 is starting a new cycle is stored in the TH22 threshold gate with one input always asserted. Once the reset signal of C_0 is asserted the output of TH22 is also asserted otherwise it remains de-asserted.

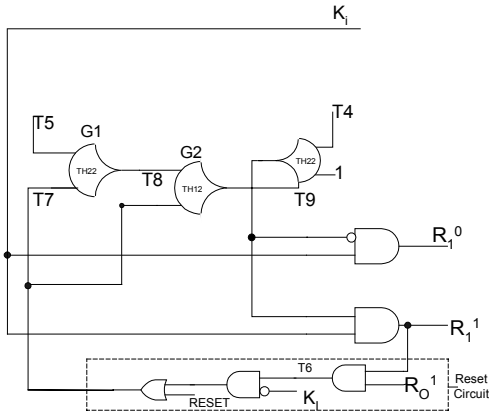


Figure 10: C_1 circuit for Version A1 Timer

Table 5: C_1 sequence for Version A1 Timer

	initial state	state1	state2	state3	state4
T4	1		1	1	reset to NULL
T7(Reset)	Null				1
A1	Null	0	1	1	reset to NULL
R10	Null	1	0	0	0
R11	Null	0	0	1	0
R01	Null	0	0	1	1

Once the circuit has completed its cycle of NULL and DATA wavefronts, it is necessary to reset the C_1 state machine to its initial condition. Reset will be asserted when RAIL1 of C_0 and C_1, i.e. R_0^1 and R_1^1 respectively, and complement of K_i are asserted. This is only possible when both C_0 and C_1 start a new cycle. Succeeding bits are generated similarly as described below.

5.2 Version A2: Reset-equipped Embedded Register

The version A1 circuit can be developed using more threshold gates to replace all the non-hysteresis gates used in the reset circuitry except the non-hysteresis AND gate. This reduces the gate count from 44 gates to 38 gates, although the transistor count remains unchanged since the weighted gates used each require more transistors individually. The complete circuit for Version A2 is shown in Figure 11 for an NCL timer with w=4.

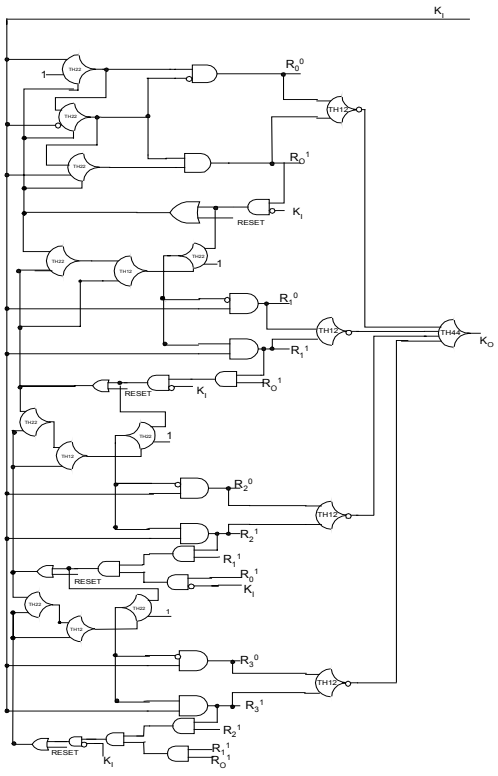


Figure 11: Version A2 Timer

5.3 Version B: Without Reset-equipped Gates

The timer without embedded registration and without reset uses the complement of RAIL1 as a controlling input. Observe in Figure 10 that the threshold gate G1 always has one input asserted so there is a need to generate a reset signal to de-assert it when needed. As shown in Figure 12, the S1 threshold gate in the chain of the Self-Reset circuitry is a Th33 gate to which one input is the output of RAIL1 of C_0, the second input is the complement of the output of gate S3. The third input to first threshold gate, x0_0, to the S1 gate of the Self-Reset circuit is provided so that the first threshold gate can assert at the proper wavefront. If this input is not present then RAIL1 will start following K_i prematurely.

The S1 gate will assert when RAIL1 of C_0 is asserted which will allow RAIL1 of C_1 to follow K_i as it has in Versions A1 and A2. The output of the S3 gate will remain de-asserted until the time C_0 completes its cycle twice. This is possible because the S2 gate in the Self-Reset circuit is asserted when the S1 gate and RAIL0 for C_0 is asserted. In the second cycle, RAIL1 of C_0 will be asserted and the third threshold gate will assert its output. At that time, the complement of the output of the S3 gate, which is the input to S1 gate, will be de-asserted.

The S2 gate introduced in the Self-Reset circuit has the purpose of becoming asserted when a unique sequence arises before RAIL1 of C_1 can be asserted. Thus, the Self-Reset circuit remembers the status of the previous bit and upon every other cycle resets the circuit.

For C_2 and C_3, the same logic is used except in C_2, the S1 gate in the Self-Reset circuit is asserted only when RAIL1 for bit0 and RAIL1 for bit1 are asserted. For generating K_o the same logic is required as for the completion circuit in Version A1. The complete circuit for Version B is shown in Figure 13.

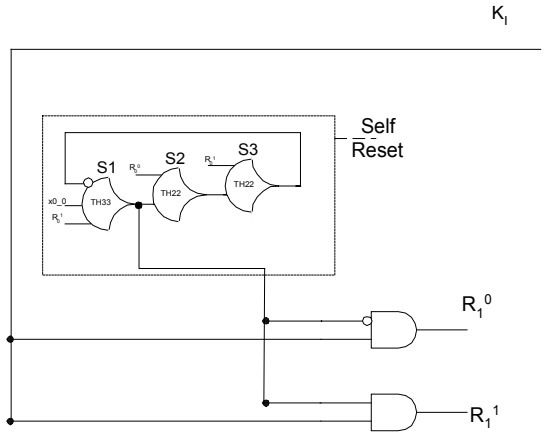


Figure 12: C_1 Circuit Diagram for Version B

5.4 Simulation Results

Table 6 lists the Gate Count, Transistor Count and Data-to-Data cycle time for all the methodologies discussed in the paper. Results are based on simulations performed using SPICE in a Cadence design environment with a 0.18µ technology library at 2.3 Volts.

Storage Mechanism	Design	Gate Count	Transistor Count	T_{DD}
Approach 1: 3-Stage Asynchronous registration	Conventional design	53	608	5.05 ns
Approach 2: 2-Stage Asynchronous registration	Collapsed REQUEST control	46	614	6.3 ns
Approach 3: Embedded registration				
Version A1:	Reset-equipped gates with combinational rollover-detect gates	44	304	4.35 ns
Version A2:	Reset-equipped gates with TH rollover-detect gates	38	305	4.35 ns
Version B:	Without Reset-equipped Gates	33	274	3.5 ns

Table 6: Simulation Results for Alternative Designs

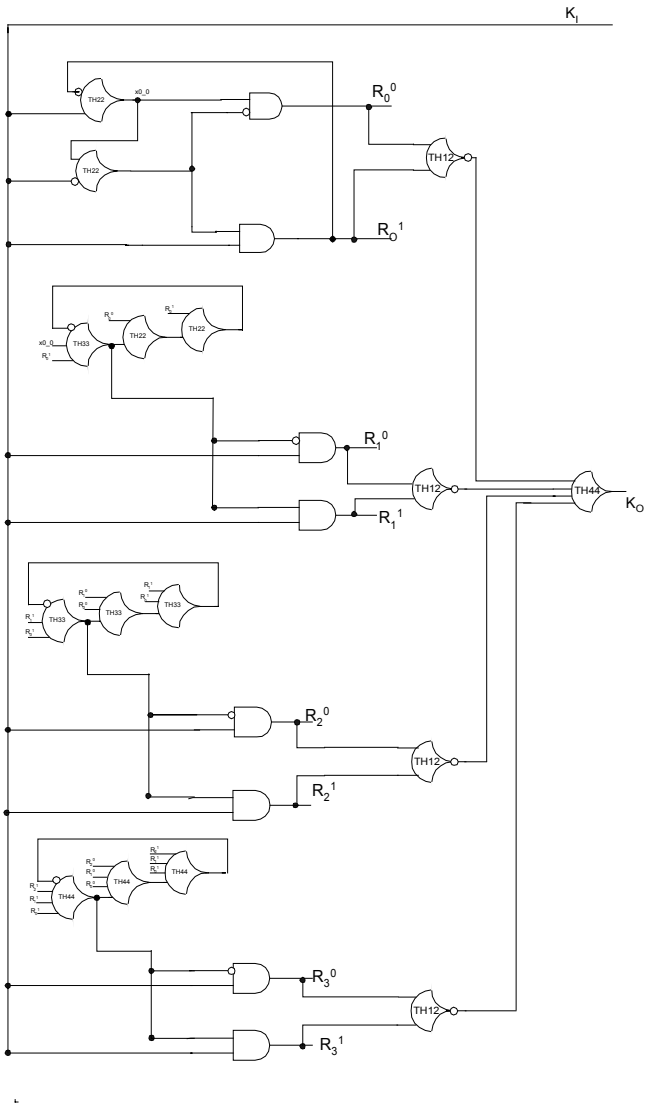


Figure 13: Version B Without Reset-Equipped Gates

6.0 Conclusion

Feedback in dual-rail self-timed circuits using the conventional method of 3-register stages has overhead three times larger than the Increment circuitry itself. The methodology developed herein reduces this overhead by embedding registration within state machines that realize an increment sequence on a bit-by-bit basis. For the test case of a 4-bit timer, the embedded approaches reduced the feedback gate count by nearly 50% and also DATA-to-DATA cycle time by 31% over 3-stage feedback.

Version A1 and A2 used feedback with embedded registration with reset-equipped gates. Version B further optimized the feedback process without using reset-equipped gates. Version B also produced the most space and time optimized circuits among these alternatives. It was also shown how dual-rail designs without explicit registration can suffer from the drawback of feedback-lagging race conditions that can only be mitigated at the physical design level.

All the methodologies developed herein will reduce overhead in terms of gate count and cycle time more significantly as the input width *w* of the circuit increases. In particular, the increases in the transistor count are almost linear with respect to *w* using methodologies for versions A1, A2, and B. A minor implementation limitation when w becomes large is the input width of the gates available, which is limited to 4 in current NCL libraries.

Acknowledgements:

The authors would like to acknowledge Heng Tan and Kening Zhang for their review and enhancements that have improved this paper.

References:

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

[2] C. L. Seitz, "System Timing," in *Introduction to VLSI Systems*, Addison-Wesley, 1980, pp 218-262

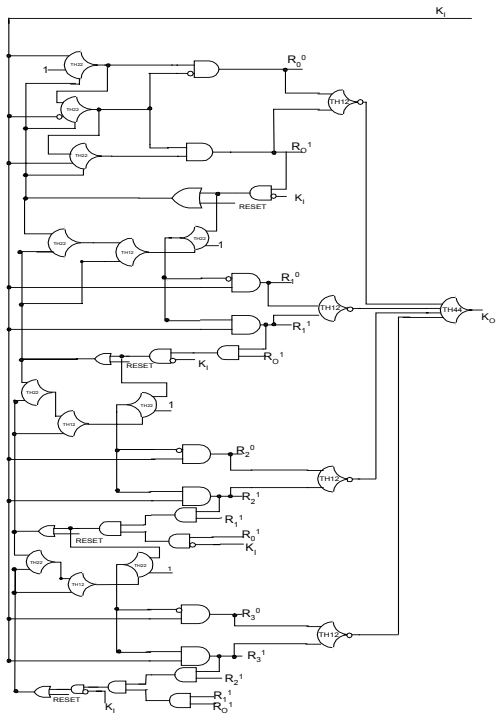
[3] A. J. Martin, "Programming in VLSI," in *Development in Concurrency and Communication*, Addison-Wesley, 1990, pp 1 - 64.

[4] K. Van Berkel, "Beware the Isochronic Fork," *Integration, The VLSI Journal*, Vol. 13, No. 2, 1992, pp 103-128.

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

[6] S. C. Smith, R. F. DeMara, J. S. Yuan, D. Ferguson, and D. Lamb, "Optimization of NULL Convention Self-timed Circuits," *Integration, The VLSI Journal*, 2004, in press.

[7] 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, November 2001, pp 103-131.


K_I
TH22
1
R_0^0
TH12
TH22
TH22
R_0^1
K_I
RESET
TH22
TH22
TH22
1
R_1^0
TH12
TH44
K_O
R_1^1
RESET
K_I
R_0^1
TH22
TH23
TH12
1
R_2^0
TH12
R_2^1
R_1^1
RESET
R_0^1
K_I
TH22
TH23
TH12
1
R_3^0
TH12
R_3^1
R_2^1
R_1^1
RESET
K_I
R_0^1