# Testing of Asynchronous NULL Conventional Logic (NCL) Circuits

Sindhu Kakarla and Waleed K. Al-Assadi, senior Member, IEEE
Department of Electrical and Computer Engineering
Missouri University of Science and Technology
Rolla, MO-65409

*Abstract*- **Due to the absence of a global clock and presence of more state holding elements that synchronize the control and data paths, Conventional Automatic Test Pattern Generation (ATPG) algorithms would fail when applied to asynchronous circuits, leading to poor fault coverage. This paper focuses on design for test (DFT) techniques aimed at making asynchronous NCL designs testable using existing DFT CAD tools with reasonable gate overhead, by enhancing controllability of feedback nets and Observability for fault sites that are flagged unobservable. The proposed approach performs scan and test points insertion on NCL designs using custom ATPG library. The approach has been automated, which is essential for large systems; and are fully compatible with industry standard tools.**

**Index- ATPG, Design for Test, CAD, Asynchronous, Null Convention Logic (NCL), Scan**

## I. Introduction

The digital world has been dominated by the growth of synchronous techniques for nearly four decades due to their ease of design. Also, CAD tools for synchronous designs have become more advanced and sophisticated allowing total automation of several stages of the design process. However, with clock speeds nearing the GHz range and CMOS technology reaching the deep submicron range, serious concerns have been raised over the suitability of synchronous designs for next-generation devices due to clock synchronization, power consumption, and noise issues [1].

Designers are looking at asynchronous circuits as a potential solution to these problems as they are modular and do not require clock synchronization. Some of the possible benefits of asynchronous techniques include low power, less EMI, less noise, increased robustness, and design-reuse [2-4]. Such an operator consists of a set condition and a reset condition that the environment must ensure are not both satisfied at the same time. If neither condition satisfied then the operator maintains its current state.

Asynchronous circuits fall into two main categories: delay-insensitive and bounded-delay models [5]. Paradigms, like NCL, assume delays in both logic elements and interconnect to be unbounded, although they assume that wire forks are isochronic [6]. NCL circuits often outperform other self-timed methods since they target a wider range of logical operators as opposed to others targeting standard, restricted sets [2].

Testing asynchronous circuits has been a major challenge [7]. In order to compete with their synchronous counterparts, asynchronous schemes must be capable of producing VLSI circuits that are at least as readily testable as synchronous circuits. Asynchronous NCL designs present a complex test case to the tester/DFT CAD tools. Testability can be strengthened by making design modifications that are dormant under normal circuit operation, and only come into play during test mode. NCL uses a delay-insensitive, self-timed paradigm to achieve synchronization by means of handshaking, leading to the presence of many feedback paths, which in turn pose a serious problem for the DFT tools. Conventional Boolean ATPG libraries cannot be used for NCL circuits, since NCL circuits are comprised of threshold gates, each with hysteresis state-holding functionality. Hence, a custom NCL ATPG library is needed to use commercial DFT tools for testing NCL circuits.

This paper is organized as follows: Section II overviews the NCL paradigm; Section III reviews the previous work in testing NCL designs; Section IV details the proposed DFT implementation, automated procedure and results; and Section V provides conclusions.

## II. NCL Overview

NCL provides an asynchronous design methodology by incorporating data and control information into one mixed path, so there is no need for worse-case delay analysis and control path delay matching [1].

Table I
Dual-Rail Encoding

| D | $D^0$ | $D^1$ |
|---|---|---|
| DATA0 | 1 | 0 |
| DATA1 | 0 | 1 |
| NULL | 0 | 0 |

NCL relies on symbolic completeness of expression to achieve self-timed behavior. Traditional Boolean logic is not symbolically complete, since the output of a Boolean gate is only valid when referenced with time. NCL eliminates this problem of time-reference by employing dual-rail or quad-rail signals. A dual-rail signal, D, consists of two mutually exclusive wires, $D^0$ and $D^1$, which may assume any value from the set {DATA0, DATA1, NULL}, as shown in Table I.

978-1-4244-2077-3/08/$25.00 ©2008 IEEE.

Similarly, a quad-rail signal, Q, consists of four mutually exclusive wires, $Q^0$, $Q^1$, $Q^2$, and $Q^3$, which may assume any value from the set {DATA0, DATA1, DATA2, DATA3, NULL}.NCL uses threshold gates with hysteresis for its composable logic elements. Such an operator consists of a set condition and a reset condition that the environment must ensure are not both satisfied at the condition is satisfied, then the operator maintains its current state.

One type of threshold gate is the THmn gate, where $1 \leq m \leq n$ as depicted in Fig. 1. THmn gates have n inputs. At least m of the n inputs must be asserted before the output will be asserted, which is the gate's set condition. Because NCL threshold gates are designed with hysteresis, all asserted inputs must be de-asserted before the output will be de-asserted, which is the reset condition [5]. Thus, any threshold gate can be represented in terms of its set and reset condition: $Z = f + (g \bullet Z^*)$, where f is the set condition, g is the complement of the reset condition, and $Z^*$ is the previous value of the output Z. Most threshold gates employ gate internal feedback paths (GIFs) in order to satisfy the hysteresis condition, represented by $g \bullet Z^*$ in the above equation. As an example, consider the TH23 gate whose output Z is asserted when at least two of its three inputs (i.e., A, B, C) are asserted, and remains asserted until all inputs are de-asserted. The TH23 gate is represented by $Z = AB + BC + AC + ((A + B + C) \bullet Z^*)$, and is depicted in Fig. 2.

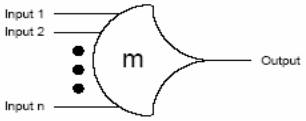

Fig. 1. THmn gate.

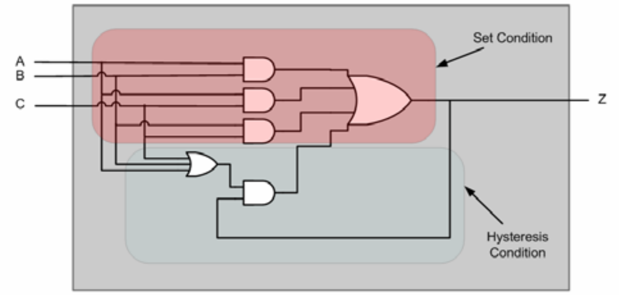

Fig. 2. Gate-level model of TH23 gate

NCL pipelines can be categorized as cyclic or acyclic based on the presence of feedback in the data path. A cyclic pipeline has a feedback loop in its data path, whereas an acyclic pipeline does not have data path feedback.

Both cyclic and acyclic NCL pipelines employ feedback in their handshaking completion paths. As seen in Fig. 3, each stage in a pipelined NCL system consists of three components: combinational logic, registration, and completion logic, all consisting of threshold gates. In an NCL system, the DATA wavefront and NULL wavefront are applied alternately [1]. The NCL registers interact with one another using handshaking signals to ensure that successive DATA wavefronts are separated by a NULL wavefront. When the register output is DATA (i.e., not NULL), request for NULL (rfn or logic 0) is generated on its $K_o$ output; and vice versa, when the register output is NULL, request for DATA (rfd or logic 1) is generated on its $K_o$ output. These handshaking signals constitute the global feedback paths (GFPs) that exist between registration stages.

## III. Testing NCL Designs

DFT methods collectively refer to the design practices used to modify the existing designs in order to make them easily testable using Automatic Test Pattern Generator (ATPG) [6]. Several DFT methods for asynchronous delay-insensitive circuits have been reported. Kang et al. [7] proposed a new scan design with low overhead for asynchronous micropipeline circuits to efficiently detect stuck-at and delay faults. A partial-scan technique for targeting delay faults for clockless systems was demonstrated in [8]. Kondratyev et al. [3] focused on test methodologies for acyclic and cyclic NCL pipelines.

In [3], acyclic pipelines are converted into combinational logic by removing the registers and completion detection through a process of fault grading. The stuck-at faults in the completion circuitry are easily tested, and can therefore be ignored. Similarly, the faults in the registration stages are eliminated by fault collapsing using dominance. Every threshold gate in the remaining combinational logic is then replaced by equivalent Boolean gates implementing the same logic function. This method yielded a good correlation between the actual and the equivalent designs, since the actual designs were found to be 100% testable in most cases. Cyclic pipelines are more complex to test. A partial-scan methodology wherein the designer specifies the points where the scan latches are to be inserted was proposed to test cyclic pipelines.

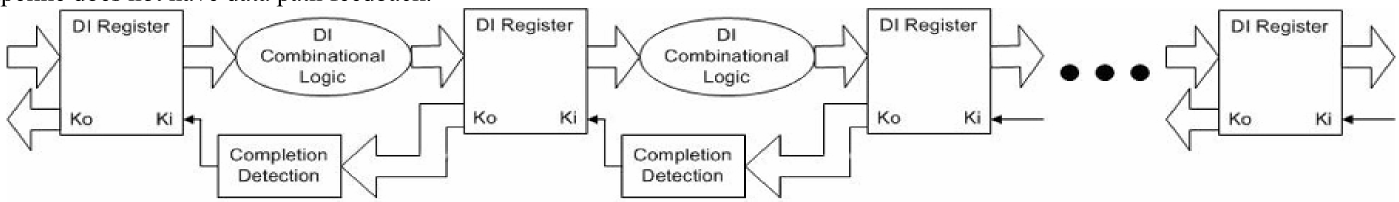

Fig.3. Pipelined NCL system

Table II
Fault Coverage for several NCL designs

| DESIGN NAME | Fault coverage | Total number of faults | Total number of Untestable faults | #Untestable Faults due to GIFs | # Untestable Faults due to GFPs |
|---|---|---|---|---|---|
| Half Adder | 50% | 48 | 24 | 24 | 0 |
| Full Adder | 28.57% | 56 | 40 | 16 | 0 |
| Dual-rail non-pipe lined multiplier | 3.14% | 2440 | 2406 | 2281 | 125 |
| Dual-rail bit wise multiplier | 4.12% | 2538 | 2442 | 2245 | 197 |
| Quad-rail bit wise multiplier | 3.99% | 2370 | 2286 | 2151 | 135 |

This method targets the Level Sensitive Scan Design (LSSD) style clocking with two phased non-overlapping clocks. A single register in an acyclic pipeline, identified as a scan candidate by the designer, would be replaced by its equivalent scan version. This technique was tested on circuits by using conventional ATPG tools to yield high test coverage [3].

While the work by Kondratyev et al. [3] presents proof for the supposition that an NCL gate's reset condition is always 100% testable, and hence can be excluded while running testability analysis, it would be very useful to be able to determine the fault coverage of the circuit as a whole, using conventional ATPG tools, rather than only the set condition. This would also eliminate patterns with NULL patterns to test the original pipeline. Furthermore, a stuck-at fault in a gate internal feedback path could result in: a) premature gate transitions that do not cause the pipeline to stall [9], b) undetected pipeline faults, or c) the static gate acting as a dynamic gate. These stuck-at faults within gates internal feedback paths have been addressed in, GIF scan technique where the controllability and Observability of the primitive gates are increased by breaking the local feedback path with a D-latch [10]. While the GIF scan technique provides good fault coverage for most of the NCL benchmark circuits, it has high gate overhead (due to the insertion of latch in internal feedback of the primitive THmn gates) which is the motivation for the proposed NCL ADIF methodology developed herein.

## IV. Proposed Dft Techniques For Ncl Designs

Testing asynchronous circuits has been a major challenge [11,12]. In order to compete with its synchronous counterparts, asynchronous schemes must be capable of producing VLSI circuits that are at least as readily testable as synchronous circuits. NCL uses a delay-insensitive, self-timed paradigm to achieve synchronization by means of handshaking, leading to the presence of many feedback paths, which in turn pose a serious problem for the ATPG programs. To test for a fault, two vectors <$t_1$, $t_2$> are required, where $t_1$ is the initialization vector and $t_2$ is the test vector. For small circuits, this could be sufficient, but for complex circuits, it could result in large computation, making this option unfeasible. Analysis of the fault coverage for several NCL circuits using conventional ATPG has revealed two important causes for fault degradation – 1) untestable faults in the feedback paths 2) unobserved faults in paths propagating through many logic levels. Such untestable or unobservable faults occur due to poor controllability and observability [13]. Results of fault coverage applying conventional ATPG programs for several NCL circuits are given in Table II. Results indicate that majority of untestable faults are due to the GIFs of NCL THmn primitive gates. Conventional ATPG programs use conventional Boolean primitive gates library. This library can only model the set condition of the THmn gate, but not the hysteresis condition. Therefore, THmn gates are represented as pure combinational circuits, and as such, faults in GIFs are not targeted. High testability for NCL designs utilizing conventional scan-based ATPG programs can be achieved by enhancing the controllability and observability of the feedback paths in NCL circuits. This in turn requires accurate modeling of NCL THmn primitive gates for ATPG that preserve the asynchronous nature of NCL designs.

In this work, the proposed DFT approach consists of two parts; 1) Modeling of NCL THmn primitive gates for ATPG , and 2) Insertion of exclusive-or gates controlled by latched test-enable in GFPs and test points (TPs) with Scannable observation latches (SOLs) in faults sites that are flagged untestable due to lack observability or controllability.

### *A. Custom ATPG library for NCL THmn primitive gates*

The NCL THmn primitive gates are modeled for ATPG to give better representation of the asynchronous NCL function of each gate. To enhance controllability and observability for faults in GIFs, the Test-enable, an external signal controlled from a primary input (PI), is applied. During the functional mode, Test-enable is set to "1", while during the test mode, it is controlled by the tester and can be set to any value. In this modeling, faults that are blocked because of the feedback are testable.

Fig. 4 shows the ATPG modeling for the TH23 gate. Faults in the Test-enable line are not included. As Test-enable is an external PI signal to the NCL design, it could be fed by the surrounding logic if the NCL design is embedded in a synchronous-based design.

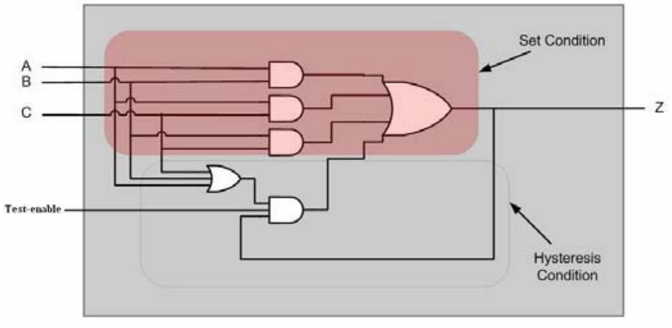

Fig. 4 Inserting Test-enable to TH23 gate

Table III
Fault coverage using developed custom ATPG library

| Test Circuit | Fault coverage including test-enable faults | Fault coverage ignoring faults on test-enable signal | Total number of faults |
|---|---|---|---|
| Half adder | 85.71% | 100% | 56 |
| Full adder | 88.64% | 100% | 66 |
| Dual-rail Full-word non-pipelined Multiplier | 21.06% | 33.19% | 2972 |
| Dual-rail bit wise pipelined multiplier | 20.74% | 34.37% | 3026 |
| Quad-rail bit wise pipelined multiplier | 18.53% | 30.49% | 3519 |

Therefore, faults in Test-enable faults can be tested when scan-based ATPG is applied for the whole system. Table III shows results of applying ATPG using the developed custom ATPG library.

### *B. Breaking GFPs*

Fault coverage is still poor due to GFPs connecting register stages via completion detection circuits in Fig. 3. Faults are still blocked because of GFPs due to poor observability and controllability of such nets. This is because those lines are deeply buried in the design that cannot be controlled easily by a PI, nor can be observed by a primary output (PO). In this approach, breaking the global feed-back paths with a latch and an Exclusive-or gate, insertion of test points (TPs) is proposed. To enhance controllability, an exclusive-or gate controlled by latched test-enable were inserted in the GFPs as shown in Fig. 5.

While this approach enhances controllability and observability of GFPs, undetected faults still occur on nets that are blocked from being observable at a PO. Making these nets POs themselves would improve observability, but would also lead to several undesirable effects, including increase in cost for adding PO pins and long wire connections leading to signal integrity problems. The solution is to insert (Test Points) TPs by grouping nets whose faults are flagged as unobservable (UO) based on SCOAP (Sandia Controllability and Observability Program) Fig.s and use a SOL (Scannable Observation Latch) as an observation point.

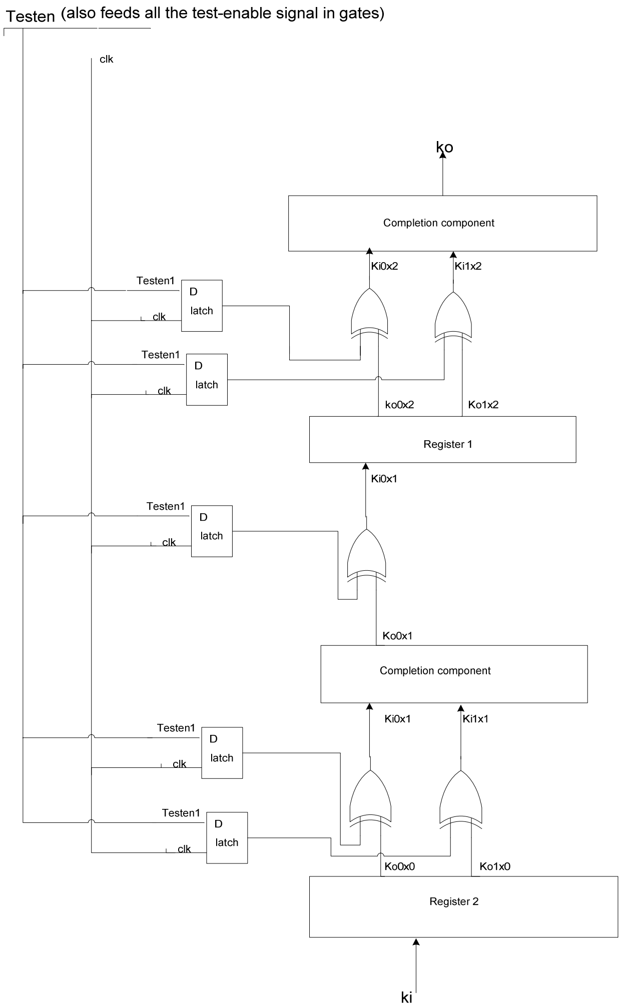

Fig. 5 Breaking GFP by inserting xor gate controlled by latched test-enable

### *C. Test point Insertion using SCOAP Fig.s*

Test points are inserted in NCL designs by grouping the nets flagged as UO considering the SCOAP Observability Fig.s. SCOAP is an algorithm to determine the difficulty of controlling (called controllability) and observing (called Observability) signals in digital circuits [6]. Fan-out factor of the gates for which nets are flagged as UO is also considered in proposed grouping strategy.

SCOAP Observability Fig.s ranges between 0 and ∞. In the proposed grouping strategy with a tree-structure, faulty nets with

(i) Observability greater than 60% are grouped using 4-input exclusive or gates
(ii) Observability greater than 30% and less than 60% are grouped using 3-input exclusive or gates

(iii) Observability less than 30% are grouped using 2-input exclusive or gates

The later stages of tree-structure are grouped using 4-input exclusive or gates. The concept is illustrated in Fig.  6.

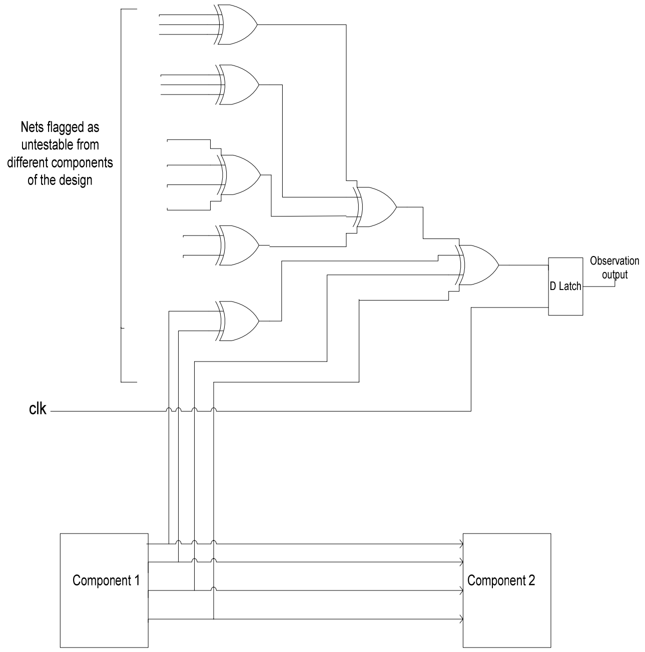

Fig: 6 Insertion of TPs using grouping strategy in UO fault sites

The added gates and SOLs do not affect the functional behavior of the design; however, careful design considerations should be taken since adding gates will change the electrical strengths of the original nets. In case of nets from the primitive gates with fan-out greater than one and same SCOAP Observability Fig., source of the net is identified and used in grouping strategy in order to decrease the gate overhead. In this approach, inserted exclusive-or gate with latched test-enable in the GFPs guaranteed controllability on these nets, while inserted TPs and SOL enhances Observability for inner nets. All inserted SOLs become part of the system's scan chain when applying scan-based ATPG.

The procedure of grouping the nets flagged as UO based on SCOAP Fig.s is automated using a PERL script. Fault list along with their SCOAP Observability Fig.s and the corresponding VHDL net list of the NCL design are the inputs to the script. First, the script reads the VHDL design and identifies the fault nets. Next, it checks whether the fault nets are same as in the fault list. Since the strategy uses exclusive-or components, 4-input, 3-input, 2-input exclusive –or components are inserted in the design netlist. Script also checks for source of the gate fan-out for fault nets. It is followed by grouping of the fault nets using Observability Fig.s. The script outputs a structural VHDL netlist with TPs and SOLs inserted.

A SOL is also inserted at the primary input test-enable in order to target the stuck-at faults at test-enable signal. Stuck-at faults at the fan-out of the test-enable for all the NCL primitives can be equivalent to the stuck-at faults at the primary input test-enable signal of the whole NCL design as they are physically same net.

An Automatic DFT insertion flow (ADIF) algorithm is developed based on the steps of implementing the proposed DFT technique. It is detailed in the flow chart shown in Fig.7

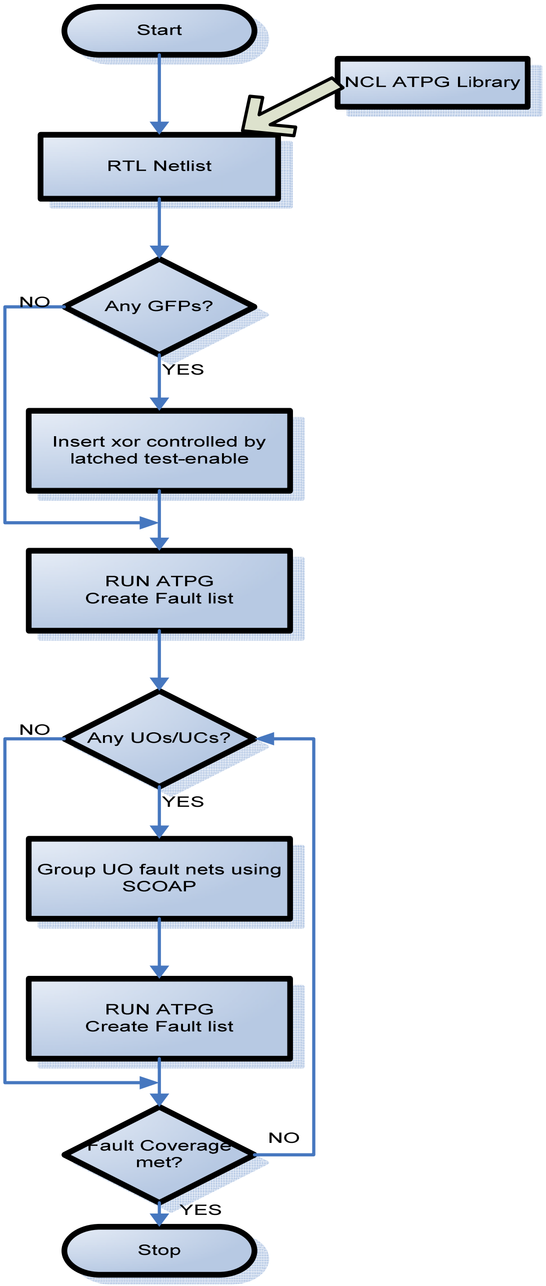
Start
NCL ATPG Library
RTL Netlist
Any GFPs?
NO
YES
Insert xor controlled by latched test-enable
RUN ATPG Create Fault list
Any UOs/UCs?
NO
YES
Group UO fault nets using SCOAP
RUN ATPG Create Fault list
Fault Coverage met?
NO
YES
Stop

Fig. 7 Proposed ADIF algorithm flowchart

The algorithm takes the structural RTL netlist of the design, identifies the GFPs, insert exclusive-or gates controlled by latched test-enable and generates ATPG netlist using the developed custom ATPG library.

Table IV
Fault statistics along with the gate over head for different NCL designs

| Circuit | Fault Coverage (%) | Gate Overhead (%) | Total Faults | Untestable Faults | CPU (Sec) |
|---|---|---|---|---|---|
| Half Adder | 100 | 0 | 56 | 0 | 1.4 |
| Full Adder | 100 | 0 | 66 | 0 | 1.5 |
| Dual-Rail Non-Pipelined Multiplier | 100 | 20.6 | 3064 | 0 | 4.52 |
| Dual-Rail Bit-Wise Pipelined Multiplier | 98.49 | 37.3 | 5464 | 62 | 31.23 |
| Dual-Rail Full-Word Pipelined Multiplier | 100 | 25.8 | 6442 | 0 | 64.90 |
| Quad-Rail Non-Pipelined Multiplier | 100 | 17.2 | 4516 | 0 | 7.13 |
| Quad-Rail Bit-Wise Pipelined Multiplier | 99.41 | 32.3 | 4992 | 21 | 17.54 |
| Quad-Rail Full-Word Pipelined Multiplier | 100 | 19.5 | 5694 | 0 | 46.54 |
| MAC | 100 | 28.5 | 380402 | 0 | 2981 |

The conventional scan- based ATPG is applied to the generated netlist. The faults that are flagged as UO are identified and TPs along with Scannable observation latches (SOLs) are inserted at faults sites as illustrated in Fig. 6. Once the target fault coverage is achieved, functional verification is performed as the final step. Fault statistics along with the gate over head for different NCL designs are given in Table IV.

Statistics from table IV shows that using the custom ATPG library with inserted test-enable signal, inserting exclusive-or gate controlled by latched test-enable signal and insertion of TPs based on grouping leads to good fault coverage with acceptable gate over head for most of the complex NCL designs.

## V. Conclusion

This paper proposes a methodology for testing asynchronous NCL designs that aims at targeting untestable faults due to the feedback paths (both global and gate internal) using the conventional scan-based ATPG programs. The proposed methodology consists of two parts; First includes the development of custom ATPG component library for NCL THmn primitive gates with the insertion of test-enable signal. Second includes breaking of global feedback paths by inserting exclusive or gates controlled by latched test-enable signal, identifying sites whose faults are flagged unobservable and inserting test points there using grouping strategy based on SCOAP Fig.s. The proposed methodology has shown a substantial improvement in fault coverage, with reasonable gate overhead. In addition it allows NCL designs to be embedded in scan-based architectures. The drawback of this method is the inclusion of external signal that is only used during test mode, and the insertion of scannable latches and test points gates.

## ACKNOWLEGMENT

This work was supported partly by the National Science Foundation under grant DUE 071776.

## References

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

[3] A. Kondratyev, L. Sorensen, A. Streich, "Testing of Asynchronous Designs by Inappropriate Means. Synchronous Approach," *8th International Symposium on Asynchronous Circuits and Systems*, pp. 171-180, 2002.

[4] W. K. Al-Assadi, Y. K. Malaiya, and A. P. Jayasumana, "Faulty Behavior of Storage Elements and its Effects on Sequential Circuits," *IEEE Transactions on VLSI Systems*, Vol. 1/4, pp. 446-452, 1993.

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

[6] M.L. Bushnell, V.D. Agrawal, "*Essentials of Electronic Testing for Digital, Memory, and Mixed-Signal VLSI Circuits*", Kluwer Academic Publishers, Nov 2000.

[7] Y-S. Kang, K-H. Huh, and S. Kang, "New Scan Design of Asynchronous Sequential Circuits," *1st IEEE Asia Pacific Conference on ASICs*, pp.355-358, 1999.

[8] M. Kishinevsky, A. Kondratyev, L. Lavagno, and A. Taubin, "Partial-Scan Delay Fault Testing of Asynchronous Circuits," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 17/11, pp. 1184-1199, 1998.

[9] S. Banerjee, S. T. Chakradhar, and R. K. Roy, "Synchronous Test Generation Model for Asynchronous Circuits," *9th International Conference on VLSI Design*, pp. 178-185, 1996.

[10] V. Satagopan, B. Bhaskaran, W. Al-Assadi, S. C. Smith and S.Kakarla, "DFT Techniques and Automation for NULL Convention Logic Circuits", *IEEE Transactions on VLSI Systems*:Special Issue on System chip Integration, 2007

[11] V. Satagopan, B. Bhaskaran, W. Al-Assadi, and S. C. Smith, "Automation in Design for Test for Asynchronous Null Conventional Logic (NCL) Circuits," *12th NASA Symposium on VLSI Design*, 2005.

[12] S. K. Bandapati, S. C. Smith, and M. Choi, "Design and Characterization of NULL Convention Self-Timed Multipliers," *IEEE Design and Test of Computers: Special Issue on Clockless VLSI Design*, vol. 30/6, pp. 26-36, 2003.

[13] S. C. Smith, "Development of a Large Word-Width High-Speed Asynchronous Multiply and Accumulate Unit," *Elsevier's Integration, the VLSI Journal*, vol. 39/1, pp. 12-28, 2005.