Design for Testability Implementation Of Dual Rail Half Adder Based on Level Sensitive Scan Cell Design

M.S.Kavitha

Department Of ECE, Srinivasan Engineering College

Abstract— Design for testability (DFT) refers to hardware design styles or it is an added hardware that reduces test generation complexity and test cost, also increases test quality. Sleep Convention Logic (SCL) is an asynchronous logic style which is based on Null Convention Logic (NCL). In the SCL the combinational blocks are made of threshold gates. SCL utilizes power gating method to further reduce the power consumption by incorporating the sleep signal in every single gate. There are currently no DFT methodologies existing for SCL. But in the current NCL, specific DFT methods cannot be directly used due to the sleep mechanism for power gating. The aim of this paper is to analyze various stuck-at-faults within SCL pipelines. Hence by using scan based testing methodology the SCL circuit is analyzed at the cost of usual area overhead. The proposed methodology is based on fault analysis.

Keywords— sleep convention logic, null convention logic, dual rail, power gating technique, Design for testability.

I. INTRODUCTION

Design for testability (DFT) consists of IC design techniques that add testability features to a hardware product design. The tests are generally driven by test programs that execute using automatic test equipment (ATE) or, in the case of system maintenance, inside the assembled system itself. The diagnostic information can be used to locate the source of the failure. The automatic test equipment is an instrument used to apply test patterns to device-under-test (DUT), analyze the responses from the DUT, and mark the DUT as good or bad. The DUTis also called as the circuit-under-test (CUT).

Sleep convention logic (SCL) is a self-timed asynchronous pipeline style that offers inherent power-gating, resulting in ultra-low power consumption while idle. SCL combines the ideas of NCL with early completion and MTCMOS power-gating. Sleep convention logic (SCL), is also known as a variant of NULL convention logic (NCL) [1], [2] that takes the advantage of the MTCMOS power-gating technique [3], [4] to further reduce the power consumption. Most of these advantages are the direct result of applying the sleep mechanism to the circuit. The first obvious advantage is reducing the static power consumption due to power-gating through high-Vth transistors.

The application of MTCMOS to the NCL circuits comes with interesting architectural changes that ultimately results in area and performance advantages as well. The aim of this paper is to analyze the various stuck-at faults within an SCL pipeline and propose a comprehensive scan-based testing methodology that provides for high test coverage by introducing the scan chain. Level Sensitive Scan Design (LSSD) is the DFT method used to test the sleep convention logic. The proposed DFT methodology is based on scan chain design, which is very popular in industry, and it provides for high test coverage.
II. SLEEP CONVENTION LOGIC

2.1. Overview
SCL is an asynchronous logic style [5] based on the NCL. SCL was originally developed in [6]. SCL combines the idea of the NCL with early completion [7] and fine-grained MTCMOS power-gating [8]. During normal operation, each pipeline stage alternates between set and reset phases. In the set phase, data change from a spacer (called NULL) to a proper codeword (called DATA), and in the reset phase it changes back to NULL. SCL uses delay-insensitive encoded data for data communication. The most popular delay-insensitive encoding is dual rail. A dual-rail encoded signal \( D \) consists of two wires, \( D_0 \) and \( D_1 \). \( D \) is logic 1 (DATA1) when \( D_1 = 1 \) and \( D_0 = 0 \), is logic 0 (DATA0) when \( D_0 = 1 \) and \( D_1 = 0 \), and is NULL when both \( D_0 \) and \( D_1 \) are 0. Combinational logic blocks in the SCL are made of threshold gates [8] and implement unate functions where no logic inversions are allowed. An SCL gate is generally denoted as \( \text{TH}_{mnWw_1,...,wn} \) where \( n \) is the number of inputs, \( m \) is the threshold of the gate, and \( w_1, w_2, \ldots, wn \) are the weights of inputs when the weights are > 1.

![Fig.1 Threshold Gate](image)

The SCL framework is shown in Fig. 1. Similar to the NCL, each pipeline stage contains a combinational logic function block \( (F_i) \), a register block \( (R_i) \), and a completion detector block \( (C_{Di}) \).

![Fig.2 SCL Pipeline](image)

SCL requires an extra gate to synchronize between DATA and NULL phases. This extra gate is a simple resettable \( C \)-element with inverted output, which will be called the completion \( C \)-element \( (C_i) \) hereafter. SCL utilizes fine-grained power-gating by incorporating a sleep signal, \( S \), in every single gate. Similar to the NCL gates [9], each SCL gate is made of a set block and a hold0 block (denoted as set). In the SCL circuits, however, since all the gates within the combinational blocks are forced to reset by asserting the sleep signal, input-completeness with respect to NULL is inherently ensured and NULL wave front propagation is no longer needed. Transistor-level implementations of rail-0 of a single-bit dual-rail SCL register. This is made of two inverters with a feedback path and can be considered as an unconventional latch. The implementation for rail-1 is exactly the same. Once the sleep signal is asserted, both outputs of the register (i.e., \( O0 \) and \( O1 \)) go to low, producing a NULL value. The outputs remain low until a new DATA value arrives and the sleep signal is deasserted. The register outputs will then get asserted according to the DATA value and remain asserted even if the inputs of the register are all deasserted. This latching behavior is special because once an output is asserted by an input it cannot be deasserted by the same input; it only gets deasserted by the sleep signal.
In the SCL completion detector the first logic level is made of the SCL TH12 gates that are essentially similar to Boolean OR gates; and the subsequent logic levels are made of the SCL TH\(n\) gates (where \(2 \leq n \leq 4\)) that are essentially similar to \(n\)-input Boolean AND gates. Each TH12 gate checks for a DATA value on a single bit of a signal and the TH\(n\) gates consolidate the results of a single-bit DATA checks to check for a complete DATA set on the entire signal. Finally, each stage of the SCL pipeline requires a completion \(C\)-element. This resettable \(C\)-element with inverted output is required to synchronize sleep signals and is critical for a safe DATA/NULL phase alternation; therefore, it must always be active during pipeline operation and it cannot be put to sleep. In fact, these are the only gates in the SCL pipeline that never sleep.

2.2. Endowments

SCL circuits have several advantages over traditional NCL circuits. These advantages are the direct result of applying the sleep mechanism to the circuit. The first obvious advantage is reducing the static power consumption due to power-gating through high-Vth transistors. Since the NULL phase is now forced through the sleep signal rather than waiting for the NULL wave front to propagate through the circuit, the gates no longer need hysteresis, because input completeness with respect to NULL is inherently ensured by explicitly sleeping all the gates. Removing hysteresis from the NCL gates results in a significant amount of area saving. As a result, no extra logic is required to be added to a combinational block to make it input complete with respect to the DATA. Finally, observability in the SCL circuits is also ensured via the sleep mechanism since any potential orphan is explicitly cleared between two adjacent data phases by asserting the sleep signal.

In summary, the following contributions are made.

1) Stuck-at faults within various components in the SCL pipeline and how they impact the pipeline are analyzed.
2) A comprehensive scan-based DFT methodology is proposed based on the fault analysis.

2.3. Organization

The remaining part of this paper is organized as follows. The related DFT work for the NCL is discussed in Section III. Section IV analyzes various stuck-at faults within the SCL pipeline. Then a scan-based DFT methodology is proposed based on the analysis results. The proposed methodology is eventually validated by applying various testing metrics. Finally, the conclusions are drawn in Section VI.

III. RELATED WORK

3.1. NCL Specific DFT Methodologies

Current ATPG tools do not support asynchronous circuit styles such as the NCL due to asynchronous feedback paths and absence of a clock signal. There are mainly two approaches in the literature to make the NCL circuits testable: limited insertion of control/observation points to increase fault coverage [10], [11], and synchronous modeling of NCL pipelines to make them compatible with synchronous ATPG tools and using scan chain technique [12].
3.1.1 Limited Insertion of Control/Observation Points:

The first approach to make the NCL circuits testable focuses on finding nodes that are not easily controllable or observable and then inserts additional control signals or observation points to improve testability. For example, the output of each completion detector in the NCL pipeline is connected to the register of the previous stage. This creates an asynchronous feedback path that is not easily controllable by synchronous ATPG tools especially when the signal is buried in a deep pipeline. Breaking the feedback path and inserting an XOR gate controlled by a primary input provides a test point that improves controllability. Moreover, in the NCL pipeline, some nodes may not be easily observable at primary outputs. These nodes themselves can be made primary outputs to increase observability, but this is not feasible if there are many unobservable nodes. Alternatively, several unobservable nodes can be consolidated to a single primary output via an XOR tree. The XOR tree, however, requires a lot of space, especially for more complex designs with several pipeline stages, and also increases the number of primary outputs significantly, so this is not very practical. In order to cope with low fault coverage, Satagopan et al. [10] then proposed to break the internal feedback path inside every NCL gate and insert a latch. This technique is shown to significantly improve testability (almost 100% fault coverage), but it requires substantial area overhead, which makes it impractical.

3.1.2 Synchronous Modeling of NCL Pipeline:
The second approach to make the NCL circuits testable, as described in [12], starts with modeling the NCL pipeline and how stuck-at faults impact its behavior. It first proves that in an acyclic NCL pipeline with M stages, the faults in completion detectors can be checked by applying at most \( \frac{M}{2} + 1 \) pairs of \{DATA, NULL\} to the pipeline. If there is a stuck-at fault at any node inside the completion detectors, the pipeline is guaranteed to stall. The value of DATA is not important and, in fact, the same DATA value can be repeated. It is further shown that faults in the registers can be eliminated by fault collapsing based on fault dominance. Consequently, both completion detectors and registers can be dropped from stuck-at fault checking. The original NCL pipeline can then be considered as a purely NCL combinational logic, after removing the register and completion detector circuitry then proving that irredundant NCL combinational circuits are fully testable for all stuck-at faults.

VI. PROPOSED LEVEL SENSITIVE SCAN CELL DESIGN

As discussed before, each stage of the SCL pipeline is made of four separate blocks: combinational logic function \((F_i)\), completion detector \((CD_i)\), register \((R_i)\), and completion C-element \((C_i)\). Since the stuck-at faults in each block can impact the SCL pipeline in different ways, each block should be analyzed separately. For the analysis in Section IV-A, it is assumed that the sleep signals are fault-free. The effect of stuck-at faults on sleep signals is analyzed later in Sec. IV-B.

4.1 Fault Analysis

4.1.1 Faults on Completion C-Element Signals:

A C-element works as follows: the output is asserted when all inputs are asserted; the output remains asserted until all inputs are deasserted (hysteresis behavior). In the SCL, however, this C-element’s output is inverted. Since in each DATA/NULL phase all inputs and consequently the output of a completion C-element must make a transition for the corresponding DATA/NULL set to propagate through the pipeline. In the SCL pipeline, all stuck-at faults on the inputs and output of all completion C-elements can be detected by allowing a single \{DATA, NULL\} pair to propagate through the pipeline from primary inputs to primary outputs. Therefore, a complete propagation of a \{DATA, NULL\} pair ensures that there is no stuck-at-0 or stuck-at-1 fault on the inputs and output of completion C-elements.
4.1.2. Faults in Completion Detector:
The stuck-at faults in a completion detector may not necessarily result in a pipeline stall. In NULL phase, even when a gate’s output is stuck-at-1, the completion detector will still produce a 0 at its output once the sleep signal is asserted, as long as the output of the last gate in the completion detector is not stuck-at-1. This is in fact a consequence of using the sleep signal to force the completion detector to get cleared rather than requiring the propagation of a NULL wave front to clear it. Note that if the output of the last gate in the completion detector is stuck-at-1, the pipeline will stall after a while. Stuck-at-0 faults always result in a deadlock, so detecting them is easy. This is due to the fact that all gates within the completion detector must be asserted in the DATA phase to assert the output of the completion detector. Therefore, if even a single transition does not happen due to a stuck-at-0 fault, the output of the completion detector cannot be asserted, which eventually results in deadlock, since the output of the completion detector is an input of the completion C-element. Hence the faults are analyzed by propagating DATA<NULL pair through the pipeline.

4.1.3. Faults in Combinational Logic:
Combinational logic blocks in SCL are unate. In the DATA phase, gates within a combinational block can only make low-to-high transitions; and in the NULL phase, they can only make high-to-low transitions. This might imply that an approach similar to can be used to detect stuck-at faults in the combinational logic blocks as discussed earlier; unfortunately, this is not possible for two reasons. The first is that the SCL combinational logic is not input-complete; so, in contrast to the NCL, a stuck-at-0 fault on a signal may not necessarily stop it from producing a valid output DATA set, and hence the pipeline may not stall. The second reason is that a stuck-at-1 fault on the output of a gate may be hidden by the gates at its fanout if those gates can be properly put to sleep. Each combinational logic block in the SCL pipeline behaves exactly like a traditional Boolean combinational logic block when its sleep signal is disabled. Therefore, traditional synchronous combinational ATPG techniques can be used to detect its stuck-at faults.

4.1.4. Faults in Register:
The test patterns generated by traditional ATPG tools are applied to each combinational block through a scan chain design similar to a synchronous approach. This implies that the SCL registers must be augmented to have functionalities similar to a traditional scan cell.

4.2. Sleep Signal Fault Analysis

The analysis performed in Section IV-A was based on the assumption that the sleep signals are fault-free. But in reality, the sleep signals are also prone to stuck-at faults. In this section, the effects of stuck-at faults on sleep signals are analyzed. In the SCL pipeline, as shown in Fig. 1, each sleep signal generated by the output of a completion C-element is forked to a register block, a combinational logic block, a completion detector block, and the subsequent completion C-element.

4.2.1. Sleep Signal Fork to Registers:
A sleep signal that forks to a register block can be either stuck-at-0 or stuck-at-1. In the case of a stuck-at-0 fault on a sleep signal, the register outputs will never return to NULL once they are set to DATA. When the outputs of registers do not get properly reset by the sleep signal, it will cause the registers to output an illegal value. If the new DATA set generates a different output than the previous DATA set then the propagation of illegal values through the pipeline can then be interpreted as a sign of a stuck-at-0 fault on the sleep signal.
4.2.2. Sleep Signal Fork to Completion Detectors:
The sleep signal forks to a completion detector are automatically tested for stuck-at-1 faults at the
time of testing the completion C-elements and are untestable for stuck-at-0 faults due to redundancy.

4.2.3. Sleep Signal Fork to Combinational Logic Blocks:
The stuck-at faults on the sleep signal fork within combinational blocks are either untestable or it can
be ignored due to fault collapsing.

4.2.4. Fault Analysis Summary:
By allowing a single \{DATA, NULL\} pair to propagate through the SCL pipeline, all stuck-
at faults on the inputs and output of all completion C-elements can be detected.

By disabling the sleep signal, the SCL combinational logic block becomes a normal Boolean
circuit that can then be checked for stuck-at faults using the traditional combinational ATPG tools.
The stuck-at faults on the sleep signal forks within a combinational logic block are either
untestable (stuck-at-0 faults) or can be ignored through fault collapsing (stuck-at-1 faults).

The stuck-at faults on the sleep signal forks within a completion detector block are either
untestable (stuck-at-0 faults) or can be detected during the test of the completion C-elements (stuck-at-1 faults).

The stuck-at faults on the sleep signal forks within a register block are best tested through a
scan chain design to be discussed.

4.3. Test Procedure
After analyzing different fault scenarios and how they impact the SCL pipeline, we can now
devise a methodology to perform testing.

- Replacing Registers With Scan Cells:
Similar to a synchronous scan-based testing approach, the SCL registers need to be replaced with
scan cells in order to shift in the test. Fig. 5 shows the interface of our proposed SCL scan cell.

![Scan Cell Diagram]

In dual-rail encoding, each register bit is made of two scan cells, one for each rail. \(D_{in}\) is the
main input, which could be either rail of a dual-rail input signal. \(S_{in}\) is the scan input, and \(D_{out}\) is the
output of the scan cell. In a scan chain configuration, \(D_{out}\) of each scan cell is connected to \(S_{in}\) of
the next scan cell. \(M\) is the test mode selection signal. When \(M = 0\), the scan cell is in normal mode;
but when \(M = 1\), the scan cell enters test mode.

In normal mode, the scan cell operates exactly like the SCL register; but in test mode, it
behaves like a traditional LSSD-type scan cell, where data can be shifted from \(S_{in}\) to \(D_{out}\) through
the non overlapping clock signals \(CL_0\) and \(CL_1\).

In test mode, once the test patterns are applied to a combinational logic block and a sufficient
amount of time has passed, the outputs of the combinational logic block can be loaded into scan cells
using signal \(L\). Finally, \(S\) is the sleep signal that puts the register in sleep mode when the scan cell is
in normal mode.
4.4. Proposed Scan Chain DFT Methodology

SCL pipeline with two primary inputs (A and B) and two primary outputs (Y and Z), shown in Fig. 8, when registers are replaced with scan cells. Similar to a traditional scan chain design, the scan cells form a long shift register in test mode so that the test vectors can be shifted in, and the captured results can be shifted out. There are, however, two major differences compared to the original SCL pipeline in Fig. 1.

First, the output of completion detector is also fed to scan cells. As discussed in Section III-A, stuck-at-0 faults on the output of gates within a completion detector can be easily detected since they cause the pipeline to stall. However, detecting stuck-at-1 faults is not as easy, since the sleep signal hides those faults. Therefore, in order to detect stuck-at-1 faults, a traditional ATPG method must be used similar to the case of combinational logic faults. Since the output of completion detector is not readily available for observation, adding an extra scan cell solves the problem. The second difference of the new SCL pipeline is adding an additional input signal rstL, to the completion C-element gates. This additional signal disables the sleep signals in test mode by forcing them to low. Signal rstH, however, does the same that rst does in the original pipeline, i.e., initializing the circuit to an all-NULL state by putting all the blocks into sleep mode.
Fig. 6 shows our proposed implementation of the SCL scan cell. The design is made of two D-latches, one of them being the original SCL register, as shown in Fig. 3, which is reconfigured by signals $M$ and $S$ to become a D-latch.

![Fig.6 SCL Scancell Design](image)

The modified version of the original SCL register for a single rail is shown in Fig. 10. This modified version makes use of three additional transistors to cut the feedback path and make the first half of the register look like an inverter when $M = 1$. For the second half of the register to look like an inverter, it is enough to just disable the sleep signal, i.e. $S = 0$.

![Fig.7 Modified Version Of SCL Register](image)

Performing the Test:

The testing procedure starts with testing the scan cells first. Similar to a synchronous scan chain, a shift test can be initially used to detect most stuck-at faults associated with scan cells and ensure the correctness of the shifting operation. For the shift test, the circuit is first placed in test mode by asserting signals $M$ and $rst_L$. This will disable the sleep signals and set the scan cells to a shift register configuration. A toggle sequence 00110011…, of length $N + 4$ is then shifted in and out, where $N$ is the number of scan cells. The toggle sequence generates various transitions on $S_{in}$ and $D_{out}$ signals to capture most of the faults associated with the scan cells. The shift test also detects stuck-at-1 faults on the sleep signal forks within the register blocks. Since every bit of the toggle sequence needs to pass all the scan cells, a stuck-at-1 fault on the sleep signal of even a single scan cell causes all the 1s in the toggle sequence to change to 0; therefore, the output sequence will be all 0s.

A similar approach can be used to detect all the stuck-at-0 faults on the sleep signal forks within the register blocks. This time an all-1 sequence, 1111…, is shifted into the scan chain. Then, a signal $rst_H$ is asserted temporarily followed by asserting $rst_L$ again. Asserting $rst_H$ causes the sleep signal of all the scan cells to be asserted, and asserting $rst_L$ returns the circuit to test mode. If there are no stuck-at-0 faults on the sleep signal forks, all the registers must then get cleared, the output sequence to be all 0s. The presence of even a single 1 in the output sequence indicates the existence of a stuck-at-0 fault on a sleep signal fork. In fact, the number of 1s in the output sequence shows how many of the sleep signal forks are stuck-at-0. The second testing step is to apply a single {DATA, NULL} pair to the SCL pipeline in normal mode and it propagate from primary inputs to primary outputs. This will detect all the stuck-at faults on the inputs and output of completion C-elements. Additionally, as discussed earlier, this also detects all stuck-at-0 faults on the output of all gates within the completion detector blocks. To detect stuck-at faults in the combinational logic blocks and stuck-at-1 faults on the output of gates within the completion detector blocks remain to be
tested. By disabling the sleep signal, the combinational logic blocks become normal Boolean circuits. Therefore, the traditional ATPG tools can be used to generate test patterns to detect the remaining faults.

V. EXPERIMENTAL RESULTS

**HALF ADDER SCL OUTPUT**

![Fig. 8 HA SCL output](image1)

![Fig. 9 SCL Scancell Output](image2)

![Fig. 10 Proposed DFT Method Output](image3)

![Fig. 11 True Response Analyzer Output](image4)
VI. CONCLUSION

The problem of testing SCL circuits for stuck-at faults was investigated. The faults were initially divided into two separate categories:
1) Faults on logic gates and
2) Faults on sleep signal forks.

The faults within each category were then analyzed separately, and the impact of the faults was discussed. Finally, the proposed DFT methodology was validated through experimental results. Future work consists of DFT testing for rijndael s-box SCL circuit. By using the fault injection techniques the fault coverage will be improved.

REFERENCES