Generating Scan Patterns for Scan Chains with Intermittent Timing Problems

Richard Illman

Dialog Semiconductor
Edinburgh, UK

www.dialog-semiconductor.com

ABSTRACT

The presence of intermittent timing problems within scan chains, for example race conditions, can cause major problems when trying to diagnose the location. The diagnostic process can involve capturing large amounts of fail data, which may exceed the tester limits. The results may also be less accurate than for consistent failures. This paper describes a new technique for generating diagnostic patterns which solves these problems. It is also possible to generate production test patterns which can achieve some level of scan test even in the presence of such timing problems. The techniques were developed on a real design, where a small digital counter block had two scan chains. Both chains had intermittent race conditions.

Table of Contents

Table of Figures

1. Introduction

Diagnosing timing problems in the capture paths of a scan-tested design (i.e. paths which are active when scan-enable is inactive) is relatively straight forward. The gate causing the timing issue normally fans out to only a limited number of sink flops. For each failing scan pattern just these flops, or a subset, will have failures. This limits the size of the fail data that must be captured on the tester. Fault simulation of the possible candidate faults within the fan-in cone of the failing flops will usually identify the location of the fail.

For timing problems within the scan chains themselves the problems are more complex. If there is a simple race condition between two flops then even the basic continuity test of shifting 0011 (repeated) through the scan chain will produce a fail on every second output cycle. For diagnostics purposes many failing patterns must be logged and the size of the data can quickly exceed the tester fail memory. It is also intrinsically difficult to locate the failure because a single timing problem may corrupt the full chain. Data between the failure point and the scan-out pin may be corrupted during the scan load operation. Data between the failure point and the scan-in pin may corrupted during the scan unload operation.

Debug of this failure type is made more difficult if the failures are intermittent.

2. Target Design

We developed the techniques described on a small asic design which had a main digital core and also a small counter block embedded within the analogue parts of the design.

The counter block has around 180 scan flops. These were split into two scan sub-chains of 90 bits each. In scan compression mode each of these formed 2 of the overall 32 scan chains. The design is implemented to ensure that in compression mode the main digital core and the audio block does not share any scan chains. This simplifies the debug process and minimises the impact of any problems in the counter block. In main (legacy) scan mode there are four scan chains. The two counter chains are embedded in separate chains.

During initial debug of the scan patterns the main digital block chains worked correctly. The counter chains both showed intermittent race conditions.

Typically race conditions are setup/hold violations on the path between the Q(or QN) output of a scan flop and the scan-in (SI) input of the next flop in the chain. The race conditions occur because of excessive clock skew or insufficient delay buffering between the two flops. They have the effect of making a scan chain appear to be one bit shorter than its true length.

Because of the separate chains for the counter block it was still possible to achieve full test coverage on the main audio block by using XX constraints on the counter chains during ATPG. Compression had been implemented with high X-tolerance so this was not a problem.

The first step in analysing the problem was to run basic scan continuity tests. These consist of shifting a simple repeated pattern such as 00110011 through the chain. This test checks that the logic elements in the scan chain work correctly. It also checks the timing, if there is a race condition the “expected” pattern will appear one cycle early at the scan chain output. It is possible have a race condition which only occurs on one clock edge in which case the output data stream will be LHHHLHHH or LLLHLLLH instead of the expected LLHHLLHH.

The shmoo plots below show the results of running scan continuity tests on both “good” and “bad” devices. They show the pass (green) and fail (red) conditions when varying the shift frequency (x-axis) and capture cycle frequency (y-axis)

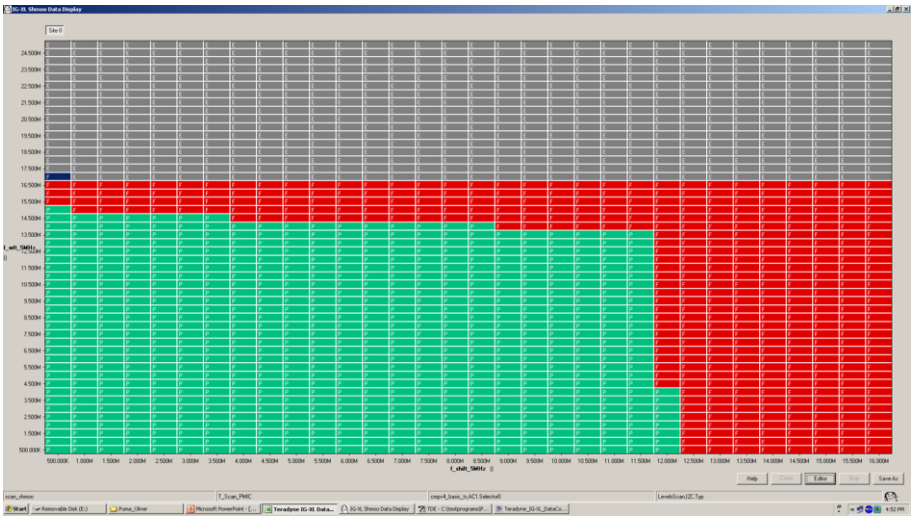


Figure 1 Good Device Scan Shift Shmoo Plot

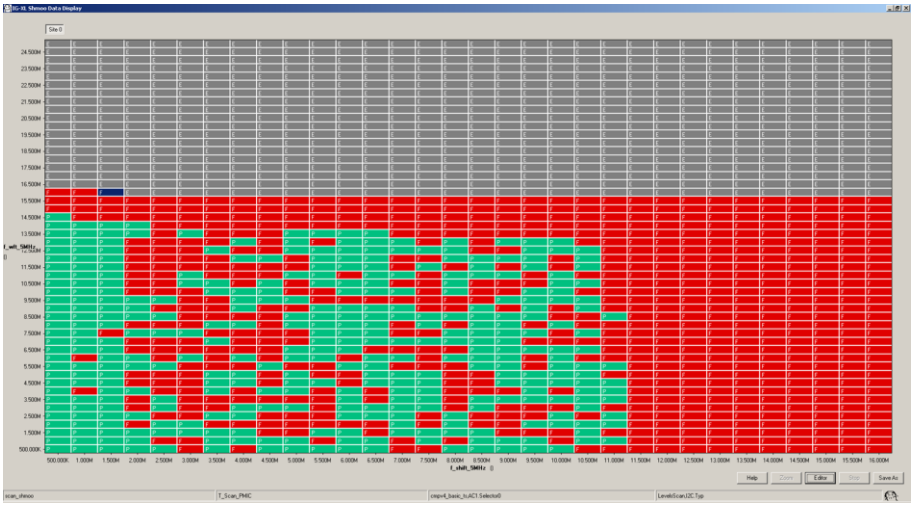


Figure 2 Bad Device Scan Shift Shmoo Plot

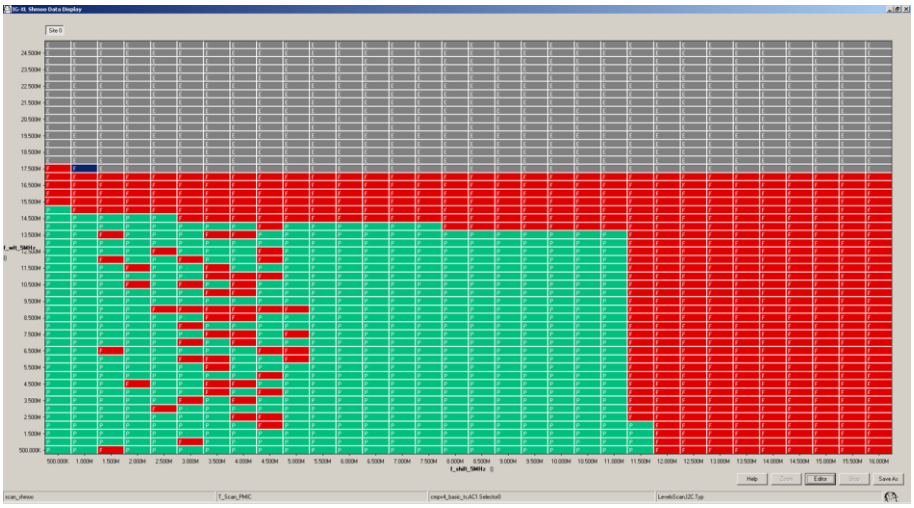


Figure 3 Bad Device Scan Shift Shmoo Plot with Bands

The random scattering of pass/fail conditions indicates that this is a random/intermittent condition. The appearance of "bands" on some of the shmoo plots is not easy to explain.

3. Existing diagnostic approaches

As mentioned earlier, one of the key problems is that the scan data can be corrupted during the load and unload operations. This problem can be overcome by only using load patterns which are insensitive to timing problems. This is supported within TetraMAX by using:

```
set_atpg -add_setreset_test
```

This causes ATPG to add special patterns:

Load 0s into the scan chain, pulse the set/reset clock, unload the scan chains

These improve the diagnosis accuracy for chain tests. Because the patterns loaded into the scan chains are unaffected by any timing issues and the capture cycle is also unaffected, then the timing problem can only affect data during the scan unload. So the location of the timing problem is known to be between the last good transition observed on the scan output and the first bad transition observed.

A simulation of this test is shown below.

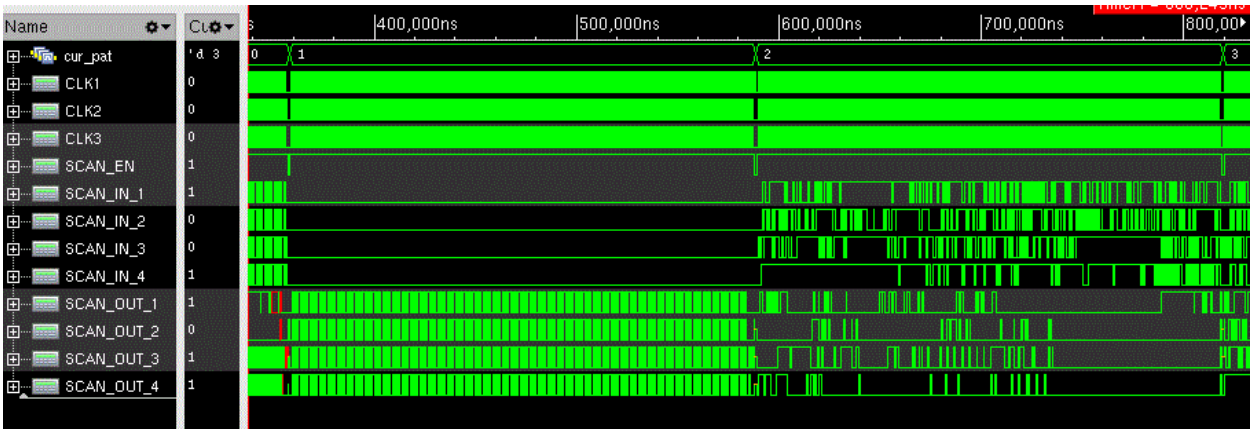


Figure 4 Simulation of "add_setreset_test" patterns

This technique is useful but its effectiveness depends upon the mix of flops with set, reset or neither operation during the scan reset pulse.

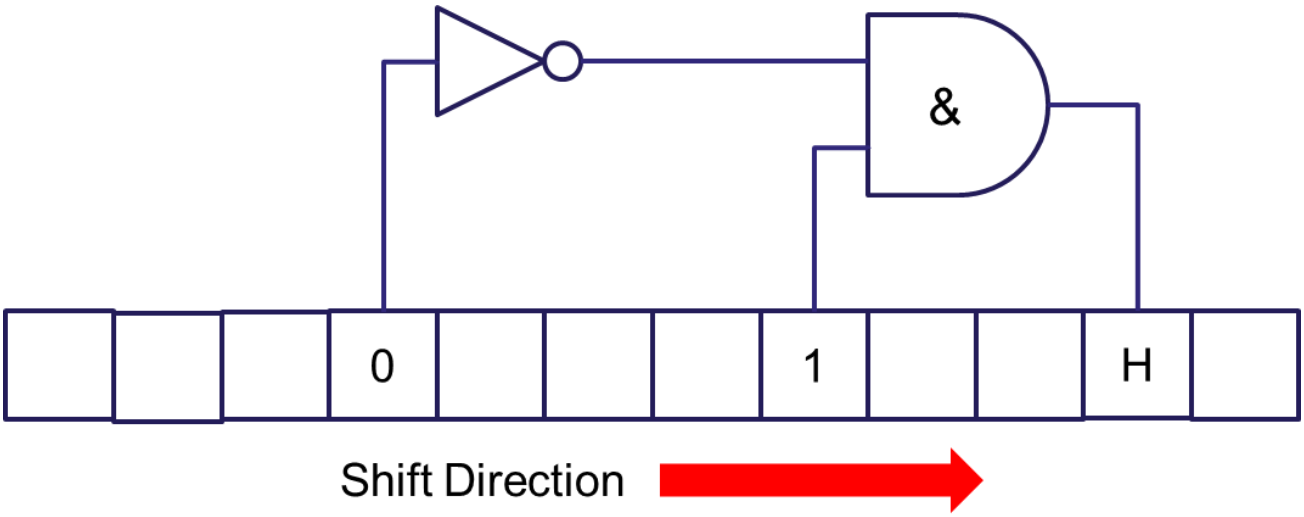
If all flops have asynchronous reset than the expected data is all zeros. This pattern is unaffected by timing issues. The same occurs if they all have asynchronous sets. The best result is achieved if each bit in the chain has a different operation (set/reset/none) from its neighbours. However, the use of set/reset values for each flop is normally fixed by functional requirements of the design. It would be possible to reorder the scan chain during synthesis/layout to try to meet this requirement as closely as possible but this could have serious negative implications on the physical layout, skew fixing etc.

TetraMAX describes this test as the `setreset_test` but **its most important characteristic is loading a constant value pattern which is unaffected by timing issues**. It is possible to generate similar patterns using `add_cell_constraints` to force the load of a constant value and then permitting ATPG to use other clock combinations to generate patterns. Normally this would be done with a fault set limit to only faults which can be detected by flops in the scan chain under diagnosis. However, ATPG normally can generate only a very limited set of patterns.

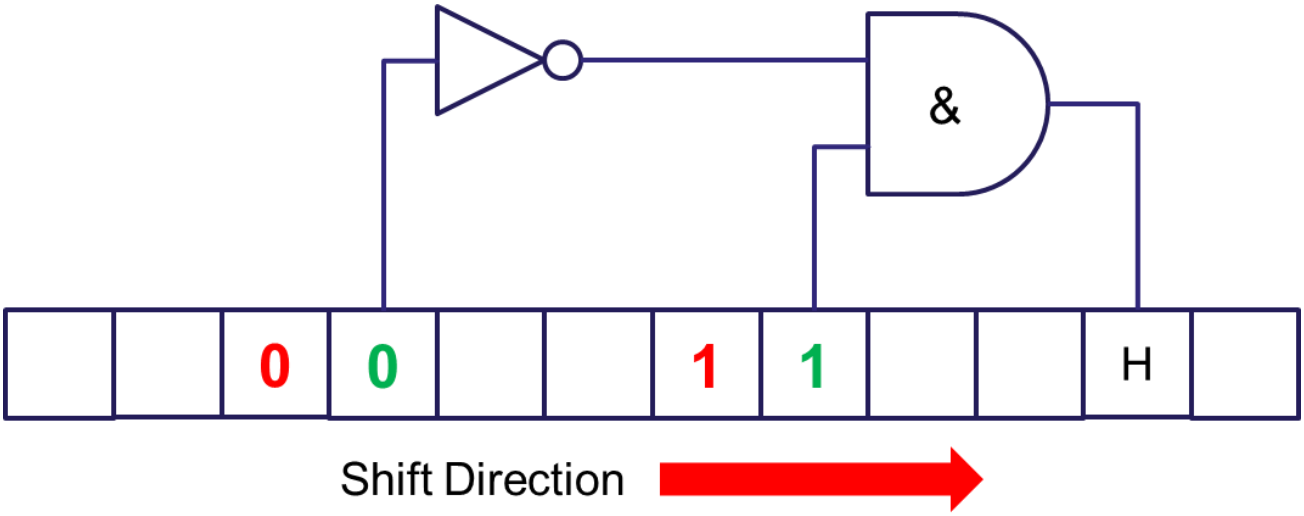
4. New Diagnostic Approach

As described earlier, the most important characteristic of the `-add_setreset_test` atpg option is that it generates patterns with load values of "all-0" or "all-1". These patterns are unaffected by timing issues during the scan load. It might appear that these are the only two possible patterns with this characteristic. However, there are in fact large numbers of such possible patterns.

If we consider the AND gate short in the figure below and the test for inputs/output SA0 it might appear that it is not possible to test the gate because the load pattern required is neither "all-0" nor "all-1".

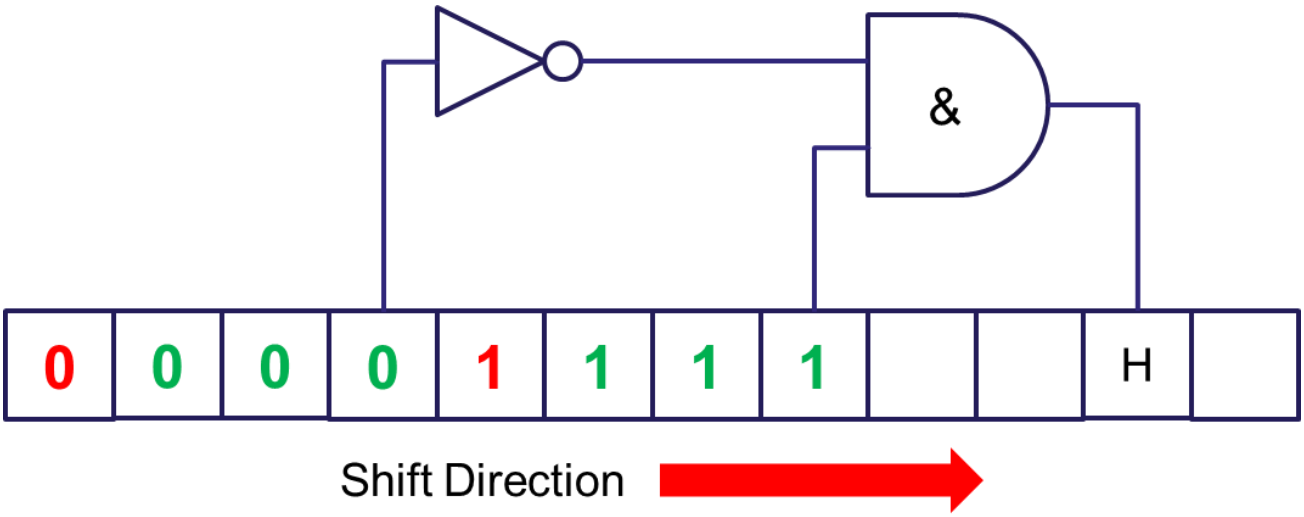


However, it is possible to apply the correct input values if the “fill” values between the two “care” values are carefully controlled.



The logic values shown in red can be thought of as “guard values”. They exist to make sure that after the load the correct “care” values (shown in green) are applied to the gates being tested. The “care” values will be correct if there is a race condition anywhere in the chain or if there is no race condition. The guard values (shown in red) are “unsafe” and cannot be used for any fault detection or logic simulation.

It is possible to extend the fill process as shown.



All the “safe” (green) values can be used for logic simulation to generate expect values and to credit detected faults.

It is possible to use any pattern which does not have consecutive “care bits” when the pattern is generated.

TetraMAX does not generate these patterns automatically so a special sequence of commands and external scripts has to be used. This flow is only possible in full (legacy) scan mode. It cannot be used with DFTMAX or DFTMAX Ultra because the “set_atpg -fill” command does allow the use of X fill. So the chip must be configured into a legacy scan mode. However, the results of scan chain diagnostics from compression mode may be used to identify which regions of the chains contain the timing problem.

- Normal ATPG runs fill any unspecified bits with random values to get additional fault detection. This option must be turned off so that only the care bits are generated.

```
set_atpg -fill X
```

- There may be two faults, each of which can independently be tested without consecutive care bits, but when tested together this requirement cannot be met. So individual patterns must be generated for each fault.

```
set_atpg -merge off
```

- There may be more than one possible pattern to test a particular fault. One meets the requirement, others do not. It is not possible to control this selection directly but it is possible to generate multiple different patterns to try to find possible candidates.

```
set_atpg -decision random

run_atpg -ndetect d
```

Typically a value of 8 would be used for ndetect.

Using these settings ATPG is run. Typically a fault list containing only faults which can be captured in the chain under diagnosis will be used. Flops in all other chains have capture masks added.

After the ATPG run the patterns are written as STIL and then processed through a number of steps.

1. Any pattern which has consecutive care bits in the load data for the chain under diagnosis is removed from the pattern set.

2. Any X states in the unload data for the patterns are replaced with L or H. These do not need to be logically the correct patterns.

3. In the load data the "safe" values are used to replace X values where possible. The unsafe "guard" values are NOT added.

4. The patterns are simulation with the option: **`run_simulation -override_differences`** this fills in the correct expect values in the pattern.

5. The patterns are fault simulated against the fault list with the option: **`detected_pattern_storage`** and then the fault list is written to a file. For each detected fault the file will contain the number of first pattern that detects it. A list of the effective patterns can be generated and the final pattern set cut down to the minimum by deleting any ineffective faults

6. The patterns are written as a STIL file and then the "guard bits" are added to the load data. The patterns can then be used on the ATE and diagnostics run on the resulting fail data in exactly the same way as for any conventional patterns.

Using this technique the scan chain issues can be diagnosed with fewer patterns and much smaller fail logs then with existing techniques.

If the location of the failing section of the scan chain can be approximately identified, then the use of "guard values" is only needed between the first possible fail bit and the scan-out pin. Any bits between the scan-in pin and the first possible fail location are by definition "safe" for scan load operations.

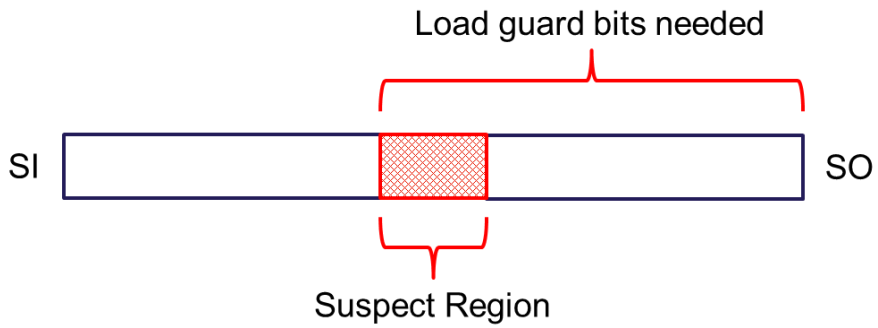


Figure 5 Diagnostic ATPG for Partially Diagnosed Scan Chain

It is important to note that this flow only works with un-compressed (legacy) scan architectures. They cannot be used with DFTMAX or DFTMAX Ultra because the “fill” options needed during ATPG are not supported.

5. Production Patterns

The techniques described earlier allow the generation of patterns for diagnosing the location of timing issues within scan chains.

If the location has been identified, it is possible to generate scan patterns to achieve some level of test coverage. These will work correctly even if there are permanent or intermittent timing problems.

If there is a consistent race condition, which occurs across the process spread and test conditions (voltage and temperature) it is possible to run ATPG on a netlist which has been modified to represent the race condition. Typically one flop is removed from the scan chain. However, if the race condition is not consistent this technique cannot be used.

Another technique is to apply cell constraints to the faulty scan chain during ATPG. All flops from the scan-input to the last suspected fail location have OX (observe X) constraints. The flops from the first suspected fail location to the scan-out have ”load 0” or “load 1” constraints. These techniques can be effective, especially where data can propagate between the faulty and fault free scan chains.

However, in our design there were two faulty scan chains, effectively isolated from the rest of the design and with the fail locations near the scan-out pin. This would effectively make all flops “observe X” and produce near-zero coverage.

The first technique that was used was a modified scan chain continuity pattern. This was generated with the options:

```
set_atpg -chain_test 00001111R
```

This produces an expected output stream in the STIL file of

LLLLHHHHLLLLHHHH repeated

This can be modified to give:

XLLXXHHXXLLXXHHX

This pattern will work even if there is a timing issue in the scan chain but still gives the same test coverage for stuck-at faults as the classic continuity test. Typically this will be around 20% which is clearly low but does offer a gross check of the flops, clocks etc. The "setreset" test can also be added to test the asynchronous sets/resets.

It is possible to generate higher coverage patterns using and extension of the concept of guard values described earlier.

After step 5 of the ATPG flow described above the scan patterns typically have load/unload data as shown below:

```
Load     0000X1X00X1111X0
Unload   LLXHHHXXLXHHHXLL
```

The "guard values" have not yet been added to the load data. In the unload data there will be some X states (due to the undefined guard values) or due to other X-generation logic in the design.

The patterns can be modified to make use of any "guard" values which exist in the unload data. If there are two or more repeated logic values then all except the last one are effectively safe. So the pattern can be modified.

```
Original Unload    LLXHHHXXLXHHHXLL
Modified Unload    XLXXHHXXXXXHHXXL
```

This means that the pattern is safe during the scan unload, even if a race condition occurs anywhere within the chain the output data will still match the expect values.

The technique can be verified by performing full Verilog serial simulation of the scan patterns against the original netlist and also against a netlist modified to model a race condition. This can be done with separate simulations for potential race conditions at the start, end and near the centre of the scan chain. The Verilog testbench should be generated in serial mode only:

```
stil2verilog -ser_only
```

When the unload data of the patterns has been modified they can be fault simulated to check the final test coverage. It is important to recognise that this is the coverage for devices which don't have the race condition. A method to measure the test coverage in the presence of the race condition has not yet been identified.

6. Partially Diagnosed Race Condition

Using the scan diagnostics it may be possible to identify the location of the timing hazard, possibly to a restricted subsection of the chain even if a single location cannot be found. This this case the process of generating patterns can be modified so that restrictions on the load pattern are not needed between the scan-in pin and the start of the suspected fail region. Observe constraints are not needed between the end of the suspect region and the scan-out pin as shown below.

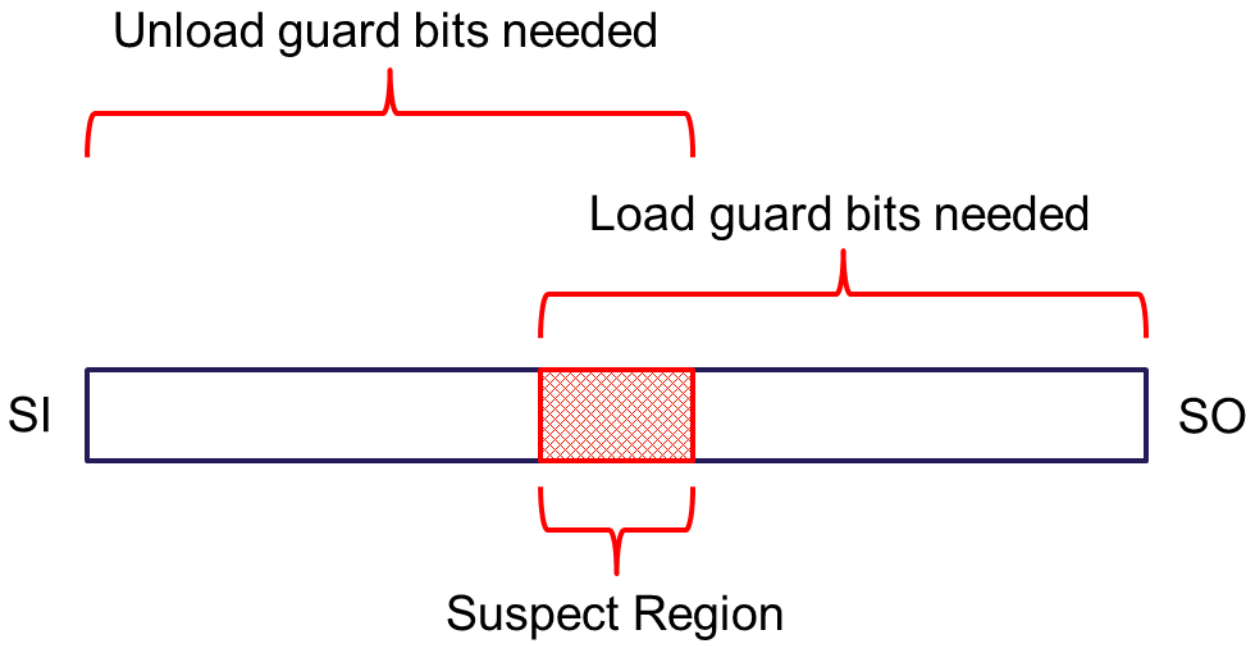


Figure 6 Production ATPG for a partially diagnosed scan chain

The same approach can be used iteratively when generating diagnostic patterns, without observe restrictions, to improve the accuracy of the patterns after an initial set has identified a failing region.

7. Results

In this design the two counter scan chains were each 90 bits long. For one chain the race condition was diagnosed as being between the scan-out pin and bit 20 (bit 0 being closest to scan-out). For the other chain the race was diagnosed as being between the scan-out pin and bit 8.

Using the techniques described it was possible to achieve 73% test coverage. This is below the usual target values but is a huge improvement over near-zero coverage which would normally be the limitation.

8. Conclusions

The techniques described improve the accuracy of diagnostics for scan chains with intermittent timing problems. They also reduce the volume of fail data which must be logged, which can be a limiting factor.

It is also possible to produce some production vectors for failing scan chains. The coverage will be lower than normally achieved but will be significantly better than if a complete scan chain is simply masked.

9. References

[1] TetraMAX ATPG User Guide

10. Acknowledgements

I would like to thank Franz Muehlegg, David Johnson and Frank Nolting of Synopsys for their help in preparing this paper.