

# LOW OVERHEAD DELAY TESTING OF ASICS

Pamela Gillis, Kevin McCauley\*, Francis Woytowich, and Andrew Ferko

IBM Corporation Essex Junction, VT 05465 USA  
\*Cadence Design Systems Endicott, NY 13760 USA

## Abstract

*Delay testing has become increasingly essential as chip geometries shrink [1,2,3]. Low overhead or cost effective delay test methodology is successful when it results in a minimal number of effective tests and eases the demands on an already burdened IC design and test staff. This paper describes one successful method in use by IBM ASICs that resulted in a slight total test pattern increase, generally ranging between 10 and 90%. Example ICs showed a pattern increase of as little as 14% from the stuck-at fault baseline with a transition fault coverage of 89%. In an ASIC business, a large number of ICs are processed, which does not allow for the personnel to understand how to test each individual IC design in detail. Instead, design automation software that is timing and testability aware ensures effective and efficient tests. The resultant tests detect random spot timing delay defects. These types of defects are time zero related failures and not reliability wearout mechanisms.*

## 1. Introduction

Some random spot defects that slow the transition of a signal were relatively benign in slower technologies with larger device and wiring features, but are delay defects which impact the circuit function in .18 micron and smaller technologies. Early studies on CMOS ASIC products and the initial attempts at delay test techniques within IBM in the late 1980's did show defect reduction benefits [4]. This work guided CMOS metal wiring and other process improvements, continuous quality monitoring actions, and delay test methodology development. Integrated circuit (IC) defects include wire and interconnect via opens (complete and resistive), bridging between wires, and parametric defects [5,6]. As the device sizes have decreased, there has been an increase in the number of devices with greater than 100 million transistors and in the amount of wire, which is now measured in miles [7]. Just the increased amount of wire, layers of metal, and vias has increased the likelihood of resistive open type defects. This trend will continue as new technologies are developed [3,5].

Delay faults are defects that slow down the circuit. Static faults, which cause the circuit to not function properly at any speed, are represented by stuck faults, as shown in Fig. 1, also called stuck-at faults.

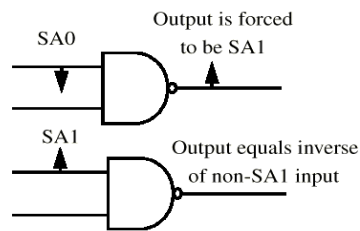


Figure 1. Stuck Faults on NAND Gate

Delay test patterns are based on the transition fault model [8]. A transition fault is a slow-to-rise or slow-to-fall fault on the input or output of a primitive logic gate (AND, OR, INV, etc.), as shown in Fig. 2. It causes an extra delay such that a system logic path will fail.

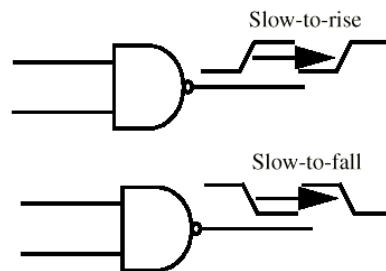


Figure 2. Transition Faults on NAND Gate Output

After waiting some extra time the logic transition will get to its desired destination. It is harder to do test generation for transition faults than stuck faults since a transition that causes the logic gate to switch has to be sensitized and observed at downstream latches. This requires at least one additional scan latch bit to be preconditioned and therefore reduces the detection

probability compared to an equivalent stuck fault by at least one-half. There are also approximately 50% more transition faults in a logic design than stuck faults, since slow-to-rise and slow-to-fall faults are on all inputs of primitive logic gates whereas stuck faults are only modeled on non-controlling inputs.

Achieving high transition fault coverage can require more patterns than stuck-at testing. With IBM ASICs, we have developed a low-cost static test strategy using reduced pin count testing, multiple well-balanced scan chains, and compact pattern sets [9]. Delay testing causes additional patterns, but the additional test time must be kept to a minimum. Test costs need to continue to be a very small percentage of the overall manufacturing cost of the ASICs.

Delay tests applied to IBM server custom microprocessor and support chips have utilized on-chip clock generation design for test (DFT) techniques [10].

## **2. Delay Test on ASICs**

### **2.1 IBM ASIC Test**

IBM ASICs are designed with full scan, conforming to the level sensitive scan design (LSSD) rules [11]. This allows a full structural test of the logic. The delay tests described in this paper use the LSSD structure and LSSD clocks, which are level sensitive and can be controlled from chip inputs. The advantages and disadvantages of LSSD are listed below.

Cost is an advantage, as the existing LSSD test structures are being reused so little additional logic is required and methodologies need not be changed.

Fewer patterns are required since many tests can be setup by setting opposing values in the master and slave latches. This reduces the patterns since fewer care bits are required. This can be done with edge-triggered mux-scan products, but it requires switching an intervening scan-enable.

Since separate clocks are used to launch and capture, the tests are not limited by the tester's frequency and do not require special processing such as multiplexing tester pins.

A disadvantage to be overcome is the launch clock typically is a non-functional clock and without special consideration can degrade the test.

The effectiveness of the tests is reported by the coverage of the transition faults, though the effectiveness must

also take the speed of the tests into account. To achieve more effective delay testing, the two LSSD clocks are pulsed in quick succession; the speed is determined by test generation system. The clock speed for the transition fault test vectors is computed based on the post-layout design delays calculated for the test environment.

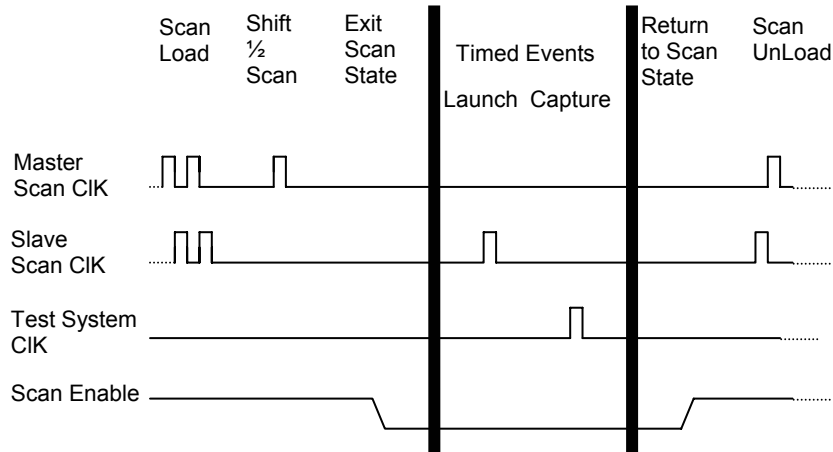
Chip logic gate/register array transition fault coverages can be over 90% for the bulk of the chip logic. Chip inputs and outputs can be delay tested using a wrap structure through uncontacted bidirectional inputs/outputs at wafer [12]. SRAMs are delay tested by a separate process that is not described here.

This testing differs in several aspects from the ASIC delay tests described in [1]. The IBM ASIC test methodology typically achieves higher transition fault coverage. The patterns can be applied at the tester without modification because the test generator understands the clocking and calculates the timings as well as the full pattern sequences, including timed launch and capture clock pulses.

A timed test typically launches from an L2 (slave) latch through logic gates to a system data input of an L1 (master) latch observation point. The clock network is tested as well as combinational logic gates. The test is a high coverage structural gate level test rather than a functional test. The timed portion of the test uses LSSD test clocks and can use any combination of clocks, with chip input changes as required before or between the clocks to handle clock gating. If long paths exist, these can be masked to allow tighter timings. There is coverage of clock slow-to-fall defects, as well as path delays and clock slow-to-rise problems.

Most chips have a large number of clock domains of differing sizes, and greater than 25 domains are not uncommon. The commonly used industry methods test either the whole chip at the slowest speed or separately create tests for each domain; this requires specific knowledge of the design and requires running multiple test generation processes. These practices require the manual entry of the time at which each clock event must occur in the pattern and verification by a separate static timing analysis process [1]. Also, when clock domains are not concurrently tested, additional patterns are required to achieve a desired defect detection capability.

Instead, we can handle different domains by grouping logic by their frequency so that tests are generated for multiple domains simultaneously. This includes logic within embedded digital cores. So we don't just test all



**Figure 3. LSSD Clock Sequence Timing Diagram**

the logic based on the slowest clock domain. The process is first to test the faster logic while ignoring the slower logic, masking out measures on the slower latches or disabling long paths feeding them. In a series of steps, the speed is decreased and as a result the percentage of logic to be targeted for transition fault test generation increases.

Alternately, if test time needs to be low, we can pick the best timing for each clock sequence such that the longest paths are ignored or masked out, but the rest of the logic is tested at one speed. Using the physical design data, as described in 2.3, the software can make this choice.

## 2.2 Clocking Sequences

Generally, LSSD delay test pattern sequences consist of pulsing two separate clocks. An example is shown in Fig. 3.

“Scan Load” plus an additional master scan (A) clock; this sets up opposite states in the master and slave latches which create the potential of many transition launches in parallel, the probability of which is higher since functional logic feeding the scan register does not take part in the launch of the transition phase.

“Exit Scan State” sets any other controls inputs needed for the test. This event can be done slowly. Thus the scan enable does not need to be fast.

Apply the delay test launch/capture time interval according the delays of the active logic paths as follows:

Pulse slave scan (B) clock feeding slave (L2) latches (shifts new values into slave latches)

Pulse test system (C) clock feeding L1 latches (captures data into master latches)

Scan out master latch values to see if transitions were received from the logic gates in time.

If the transition propagated too slowly, the correct value will not be clocked into the L1 master latch. The tester will measure a failure versus the expected latch value.

Only the launch and capture pulses are “timed”. The rest of the sequence events can be applied at any speed (the speed only affects test time, not detection of defects). For LSSD, any scan enable signals can either be switched to their non-scan value (which is done on most patterns), or left in their scan state, depending on what values are needed to detect particular transitions.

In order to get very high transition fault coverage, additional sequences are also used. It could be that the scan order does not allow a transition to be setup via scan [8]. Changing the L1 (master) latch via a C clock can load a different value in that couldn’t be scanned. This clock can be applied at a slower speed so that the timed or fast portion of the sequence looks just like the above sequence.

Another clocking sequence may switch chip inputs between the clock pulses. This typically is because there is clock gating and these gates (which may actually be functional clocks) must switch to allow the pulses through. This sequence times the B pulse, input switch, and C pulse events. The switching of chip inputs may be considered a detractor, but because the timing events are calculated automatically, the impact is limited.

Another clocking sequence uses two different timed C clocks (master latch clocks). Some memory arrays require system launch clock. These include register arrays in IBM ASICs. Our SRAMs and embedded DRAMs also have data output latches that are loaded via the SRAM/DRAM clock, which is controlled by an LSSD C clock.

### 2.3 Physical Design and Delay Calculation

Lower overhead also means we are not relying on the IC designer laying out the design to simplify delay testing, but use software including our physical design tools to provide an effective delay test. An improperly designed test clock tree can decrease our ability to provide a fast test. This is due to clock skew, which is the difference in time that the clock signal arrives at all the latches it feeds. We limit this by having place and route routines that optimize the LSSD test clocks trees for delay testing. This is done by balancing the trees (rather than building them using minimum amount of wiring) and by the use of multiple clocks. Because each design has its own peculiar logic, including clock gating and shaping that effect the LSSD clocks, and because balancing is never perfect, the test generation routines also take into account clock skew.

In order for the software to generate patterns that automatically handle clock skew, shaping, and multicycle paths, detailed post-layout delay information is used by the software. This information is provided by a Standard Delay Format (SDF) [13] file. The SDF's delay information is calculated for the tester temperature and voltage conditions.

Low V<sub>dd</sub> voltage conditions are used based on our experience of AC defects causing larger time delays than nominal conditions. This makes the defects more detectable with our AC test timings. The low V<sub>dd</sub> is selected such that it amplifies delay defects while not exposing us to yield losses.

Our ASIC delay rules [14] have multiple modes, or subrules, for more complex cells such as latches and cores. These special modes describe the paths and delay calculations used for test. The SDF contains the wire delays between all cells, the cell input to output delays, and setup, hold and pulse width requirements. For large ICs with digital cores, the delay data is handled hierarchically, but still accounts for paths crossing into or within the core to enable the correct masking of internal latches. For intellectual property reasons this is done without the core's internal information being exposed in the SDF.

### 2.4 Test Pattern Generation

The pattern generation uses the delay information provided in the SDF to optimize to the tightest time between the launch and capture events, utilizing improvements developed beyond previously described concepts [15]. There have been many enhancements, such as automatically ignoring logic that will not evaluate in the expected time and use of that information by the test generator.

The process used for the low-overhead testing is to reduce the number of clock sequences. Otherwise, a large number (>50) of the timings sequences might be created. Limiting the number of clock sequences is important since it reduces the overhead of verification of the timing for the patterns and eliminates special cases that were not previously expected in the delay rules. The test generation process first determines the best sequences, the launch-to-capture time intervals (including clock pulse widths), and the observation locations that will not be evaluated. This information is used by the timing calculation code, test generator, and the fault simulator to create and grade the transition fault tests produced.

The pattern generation must be done to maximize the detection of defects while not adversely affecting the test yield. One way yield loss is avoided is to add margin into the arrival time calculation by separately accounting for the clock, data and gating paths. The data and clock arrival times are calculated for signals launched by the events in the timed portions of the test patterns. The calculations take the SDF delays and guaranteed steady logic values into account. The result is timing checks (i.e., setup and holds) are ensured to not be violated or, if so, the logic they control will be ignored. The calculations use early and late modes which are a combination of the best, typical, and worst case delays. For hold checks, the late clocks are compared to the early data arrivals; conversely, for setup checks, the late data propagational times are compared to the early clock events. Also, to check boundary conditions, such as when short data paths and long clock paths are compared, the best and worst cases are maintained and compared.

If the pulse width, setup, or hold time checks can not be maintained without increasing the pulse widths or launch-to-capture time interval, the effectiveness of the tests is reduced. Instead, the locations which fail this check will be ignored by placing an X measure in the scan bits fed by these locations. For designs that cannot tolerate observing X states because the scanouts are compressed in a linear feedback shift register, such as a multiple input shift register (MISR), the test generator and simulator ensure the unstable latches do not observe

transitions. The transitions are either blocked at the observing latch or prevented from being launched at the origin of the long path. The test generator ensures the transition constraints do not switch as it creates the tests to detect the target fault. Additionally, the fault simulator checks that this is accomplished; and, if they do not comply, the vectors are removed. Otherwise, the observation latches would need an X state. Transitions that are protected from occurring may be fed by long paths or caused by other situations. Long paths gating clocks, overlapped clocks, or pulses that cannot be maintained without wide pulses at the chip inputs can be encountered.

Multiple test generation runs can be concatenated, with the first ones using aggressive launch-to-capture times and pulse widths; the later ones relax these constraints to achieve tests for the slower of the logic.

The target tester timing capabilities are also input to the test generation system via a tester description rule. These include minimum pulse width, pin-to-pin accuracy, timing resolution, and minimum/maximum tester cycle. These capabilities are used along with the chip timing data to determine the optimal timings for each delay test sequence type. Also, logic values that are held steady are used, as these sometimes block longer paths. Having unique timings for each of the sequences types helps catch smaller delay defects than would a generic timing for the whole chip or for the chip clock domain.

If accurate timing information is present and complete in the SDF for all cells and cores in the design, the test timings calculated by the delay test automatic test pattern generation (ATPG) code is directly used in applying manufacturing tests. However, if data is not available for every cell or core instance, default algorithms can be used to allow test timing computation. For this case, the timings can be learned at the tester via characterization using device samples covering the process specification window. Wafer batches specially processed to vary transistor channel length (L effective) and voltage threshold (VT) have been used to create such hardware. Metal wiring resistance tolerance is another factor, as wiring levels and interconnects have increased in importance in the chip path delay timings. This learning is important to maintain an effective balance between real delay defect detection and avoidance of false yield rejection. The ATPG delay test output files are modified as part of the production test program implementation.

The methodology described here is not limited to working with LSSD parts, but can also be used with

parts containing edge-triggered mux-scan or a combination of the two scan types. Only LSSD parts were used by for the work described in this paper.

## 3. Experience

### 3.1 Keeping Cost Low

Clearly, keeping pattern count and tester cost low is vital to any ASIC test, as emphasized in [1]. IBM ASIC delay tests have several elements that work towards this goal. First, we make use of boundary scan and only use a limited number of full-function tester channels, but in a manner that allows multiple scan chains [10]. Next, there is no requirement for high frequency pin electronic channels on the production automatic test equipment (ATE); racing the timing edges of the launch excitation pulse relative to the capture observation clock pulse can be accomplished on most commercially available ATE. This allows us to continue using older testers. We have even been able to continue to get high quality tests from the IBM tester described in [16], which has been in use since 1989. The only major enhancement to that tester has been increased pattern buffer memory.

The method used to generate the IBM ASIC delay test data is a combination of stuck-at and transition fault test generation. To start the procedure, a few static (DC stuck fault only) patterns (a DC “warm-up”) are created. The timed delay test patterns are generated next. For this step, both stuck-at and transition faults are simulated. The completion phase is a static DC test generation “cleanup” to enable achievement of the normal very high (>99%) stuck fault coverage.

Table 1 shows structural data for our example chips, all of which are real ASIC chips. Most are not IBM designs, but all are manufactured by IBM. The tests were often generated without any knowledge of the design function. The table shows gate count in millions, total number of shift register latches (SRLs) in thousands, and the number of stuck-at and transition faults, both in millions. The SRLs are in random logic, in register arrays, in memory BIST, and in cores. The SRLs are arranged in typically 26 shift registers or scan chains. For most of these chips, these 26 scan chains are reconfigured into 54 scan channels that feed an on-product MISR. The 54 scan inputs and outputs of the 26-chain configuration are used as 54 scan inputs and 54 MISR observe points. The delay tests are performed in this configuration. Chip F does not have the on-product MISR. Chip H has a unique configuration to test half the logic using an on-product MISR and the other half without the MISR. The reason for this configuration is beyond the scope of this paper. We have included this

chip since it is the largest IBM ASIC that we have tested to date.

**Table 1. Example Chip Size Statistics**

Chip	Gates (M)	SRLs (k)	Stuck Faults (M)	Transition Faults (M)
A	0.2	17	0.93	0.96
B	1.9	217	8.76	9.13
C	2.7	264	11.98	12.8
D	2.9	116	8.75	12.27
E	3.2	280	14.36	15.23
F	5	386	20.95	22.93
G	7.5	763	39.42	41.74
H	22.6	1854	98.45	108.16

The pattern coverages and counts for these chips are shown in Table 2. The last column is the percent increase in pattern count in moving from a pure stuck-at fault pattern set (DC-only) to the stuck-at plus transition fault pattern set (AC+DC). The DC-only and the AC plus DC patterns sets achieve the same stuck-at (DC) fault coverage. As can be seen in the table, the transition fault coverages vary quite a bit. We did not have the same goals for all of these chips and so ran more transition patterns for some of them. However, high transition fault coverage is not as uniformly easy to achieve as stuck fault coverage. It is much more design dependent. Some chips have a lot of logic that includes long paths, such as delay multiplexers that choose a varying delay to compensate for signals coming from off chip, and this needs to be held in a fast path state to achieve good timings.

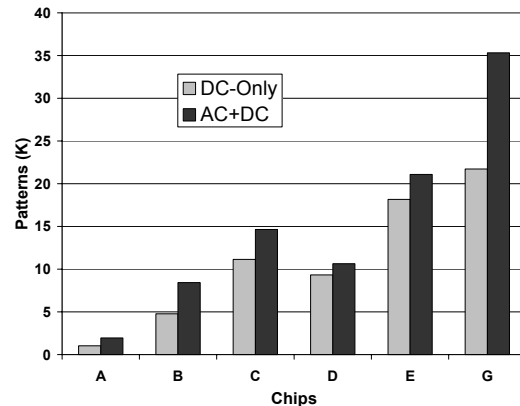
**Table 2. Example Chip Pattern Statistics**

Chip	Gates (M)	DC %	AC %	DC-only patterns	AC+DC patterns	% delta
A	0.2	99.7	76.3	1022	1942	90
B	1.9	99.7	84.3	4774	8424	76
C	2.7	99.8	77.4	11129	14649	31
D	2.9	99.8	89.5	9331	10639	14
E	3.2	99.9	82.2	18160	21078	16
F	5	99.8	94.6	3424	8554	150
G	7.5	99.9	80.7	21715	35326	63
H	22.6	99.9	93.1	4407	12410	180

Chip D shows the smallest pattern increase, only 14%, and achieves a very high transition fault coverage of 89.5%. For this chip and most of the others, we were

aiming for 75% or better transition fault coverage. For chip F, we had a high transition fault goal (>90%) and could not take stuck fault credit for the delay patterns, so the pattern increase is much higher. For chip H, the coverage goal was also very high. This chip was readily testable for stuck faults, requiring a small pattern count even though it is very large. It was not quite as easily tested for transition faults, so the pattern increase is a larger percentage. All of the transition fault coverages (AC %) are above 75%.

Figure 4 shows the pattern count data for our example chips that use our low-overhead methodology. They contain an on-product MISR and we aimed at modest transition fault coverage (75%), or greater if achievable with a moderate pattern count. Thus, chips F and H are excluded from this figure because they did not follow this methodology. The data shows that there is some increase in pattern count as the size of the chips increase; sometimes, small chips can require a large number of patterns to achieve acceptable transition fault coverage and, also, some very large chips can be tested with a very small number of patterns, such as chip H. What this figure makes most obvious is that the pattern count increase is modest with this low-overhead methodology.



**Figure 4. Pattern Count, Static vs. Transition & Static**

### 3.2 Detecting Defects

As shown by other research [5,12,17,18], as well as our own experience, most delay defects are easiest to detect at low voltages. The amount of delay caused by the defects is amplified compared with the functional slowdown of the good circuits. This lessens the dependence on the ATE pin electronic timing requirements since the chip is now operating slower. As a result, the tester is better able to achieve delay tests representative of the functional chip speed. Hence, we apply our tests at the lowest of the manufacturing voltage specification conditions, at least 10% below functional operating voltage, and often lower than that. It is important that all circuits on the chip function at these

voltages, or false yield loss will occur. Analog cores on the chip can also limit the application voltage.

We have collected data on the effect of voltage on delay defects. In 2001, we collected data on 8 fails in 0.25 micron technology. This data is shown in Table 3. The product voltage was 2.5 V nominal, +/-0.2 V, and our test conditions were 2.2 V. We collected this data to see what would happen if the test voltage were lowered. The defects did become more prominent as the voltage was lowered, though we don't have the data on a reference chip to see how much slower it would become.

**Table 3. Effect of Voltage on Delay Defects, ns, 0.25 micron Technology**

Chip #	2.2V	2.1V	2.0V	1.9V	1.8V	1.7V
1	0.6	0.6	1.1	1.9	2.5	
2	5	6	7	8	9	11
3	5	7	9.5	12	16.5	
4	3.8	4.4	5.4	6.4	7.4	8.6
5	4.2	8	11	16	22	30
6	7	8	10.5	13	18.5	21
7	8	10	12	15	19	21
8	9	11	>37			

We very recently did a similar experiment on 13 delay fails on a 0.13 micron technology. This is chip F in the previous table. For this experiment, we started at a higher voltage and stepped the voltage down to below where our tests are done. The chip nominal voltage is 1.5 V. The delays measured are versus the timing conditions at the manufacturing delay test voltage, 1.05 V, at the first passing timing point. Thus some of the numbers are negative, meaning that the chip was faster than the manufacturing test timings. We also collected the same data for two reference chips. The highly negative values for the reference chips indicate that these chips are not at the slow end of the process window. R1 is slightly faster than nominal and R2 is slightly slower than nominal. The defective chips are either nominal or somewhat fast. However, we could not obtain the process data on chip F9.

The failing chips failed for different pattern vectors. For each pattern, the data was gathered on the reference chips. Table 4 shows data for one fail and the two reference chips. This is a small defect and would not show up as a fail at higher voltages. The defect becomes noticeable only as the voltage is lowered. Table 5 shows data for another fail and the two reference chips. This is a much larger defect. At low voltage it was less predictable, being measured at greater than one microsecond at 1.08V, and then 154 ns at 0.98 V.

**Table 4. Effect of Voltage on Delay Defect, ns, 0.13 micron Technology, First Defective Chip**

#	1.68 V	1.58 V	1.48 V	1.38 V	1.28 V	1.18 V	1.08 V	0.98 V
F1	-7	-6.8	-6.2	-5.8	-5.2	-3.8	-1.6	1
R1	-7.2	-7	-6.8	-6.6	-6	-5.2	-3.8	-2
R2	-6.8	-6.6	-6.4	-6	-5.4	-4.2	-2.8	-0.8

**Table 5. Effect of Voltage on Delay Defect, ns, 0.13 micron Technology, Second Defective Chip**

#	1.68 V	1.58 V	1.48 V	1.38 V	1.28 V	1.18 V	1.08 V	0.98 V
F2	2.8	4.8	6	8.8	13.2	14		154
R1	-5	-5	-5	-4.8	-4.6	-4.4	-3.8	-3.2
R2	-5	-4.8	-4.8	-4.6	-4.2	-3.8	-3.2	-2.2

Table 6 shows the voltage effect for the bulk of the fails, which failed at a very early timed pattern vector. These show a variety of effects, but all of them get slower as voltage is dropped and, at a significantly more rapid rate than the reference chips. F6, F7, F8 and F9 are particularly interesting, as they behave like the reference chips at higher voltage, but then degrade into clear defects as the voltage is lowered. If these chips were tested at their functional voltage, which typically would be between 1.2 and 1.5 V, they likely would not fail this test. However, they do have defects which could show up on other paths.

F11 and F12 have extremely large defects that almost any test would detect, as long as it invoked a transition which was observed. F5 has quite a large defect, also. F5 and F11 are very sensitive to voltage, with F5 causing a delay delta over a micro second at 0.98 V.

Table 7 shows behavior for the last of the failing chips. Note that for this pattern sequence the reference chips actually had less delay at lower voltages than at higher voltages. However, the defect still showed itself as contributing more delay at lower voltages. This was a very large defect, that could have been detected with the right pattern at any voltage. Still, it was more than twice as slow at 0.98 V as at 1.68 V.

For IBM ASICs, our general practice has been to apply the delay test at the lowest of the manufacturing voltages normally used for full stuck-fault coverage, about 10-15% below the lowest voltage rating of the technology. With the most recent technologies that use copper, the prevalence of resistive vias and other delay-only defects has increased. Thus, we slightly lowered the voltage at

**Table 6. Effect of voltage on Delay Defects, ns, 0.13 micron Technology, 10 Defective Chips**

#	1.68V	1.58V	1.48V	1.38V	1.28V	1.18V	1.08V	0.98V
F3	-0.4	0.4	1.6	2.8	4.6	6.6	9.2	12.6
F4	-1.8	-0.6	1	2.6	4.6	7.6	10.8	15.2
F5	14.4	46.8	53.2	59	64.4	96.2	98.4	
F6	-5	-5	-4.8	-3.6	-1.6	1.2	3.6	7.2
F7	-5	-4.8	-3.4	-1.4	0.8	3.8	7.8	13.8
F8	-4.8	-4.2	-2.4	-0.4	1.8	5.2	9.4	16.2
F9	-4.8	-4.8	-4.8	-4.6	-4.2	-2.2	0.8	4.4
F10	-4	-3.2	-3.2	-1.8	-1.6	-1.6	0.2	3.2
F11	118	164	220	238	259	309	348	394
F12	213	218	222	228	233	239	248	256
R1	-5	-5	-5	-4.6	-4.2	-3.6	-2.2	-1.4
R2	-5	-5	-4.8	-4.6	-4.2	-3.8	-3.2	-1.6

which we apply delay test, but still keeping it well within the operational range of the technology. Our tester timings are calculated at the test voltage.

**Table 7. Effect of Voltage on Delay Defect, ns, 0.13 micron Technology, Thirteenth Defective Chip**

#	1.68 V	1.58 V	1.48 V	1.38 V	1.28 V	1.18 V	1.08 V	0.98 V
F 1 3	166	188	210	238	265	303	338	380
R 1	-1.6	-1.6	-1.8	-2	-2	-2.4	-2.6	-3
R 2	-1.8	-1.8	-2	-2.2	-2.4	-2.6	-2.4	-3.2

Test engineers can analyze excessive production delay test unique fallout to distinguish between overly aggressive low voltage and delay test timing conditions when necessary. Data analysis inspection for repeating failing patterns and failing observation latches is very important. Correlation to other device speed indicators such as ring oscillator frequencies, scan chain input to output delays, and inline process data is also considered. Any corrective actions are always implemented to protect the customer from receiving defective devices.

The defects which are being detected are all time-zero defects detectable during manufacturing test; they are not reliability wearout failures. We know this because the fails happen before the chips are subject to any stress testing or any system use.

#### 4. Limitations

While the IBM ASIC delay test methodology can achieve very high transition fault coverage and automate the ASIC delay test, some small delay defects are missed due to the test clocks often not matching the speed of the

functional clocks. Also, when the same test clock is used for multiple functional clock domains and if the tester memory to hold ignore-latch data is not large enough, the data for each delay test launch-to-capture timing sequence is often determined by the slower clock domain. Additionally, slow test logic such as IEEE 1149.1 and memory redundancy fuse logic can be hard to be removed from this test.

These limitations are part of the reason why testing at lower-than-functional voltage is so important.

#### 5. Discussion

The data we have shown above demonstrates that high transition fault coverage can be achieved for very large ASICs. The timings associated with these patterns vary for each clock sequence, being calculated according to what logic is active for that sequence. Current and future work involves determining simple design changes to make it easy to remove slower logic from the test and still use on-product signature collection (on-product MISR). These design changes will be inserted automatically, without involvement from the chip designer or impact to the function of the chip.

#### 6. Conclusions

Complex system on a chip designs with multiple cores and clock domains have challenged delay test techniques. Achieving a low cost effective delay defect detection test can be accomplished by utilizing the design's timing data, recognizing clock gating and other complex logic content (such as delay multiplexers), and maximizing ATE configuration. An effective ATPG algorithm has been developed that encompasses these critical drivers and is used in the production testing of IBM ASICs. The data presented for our recent ASIC technology shows that low voltage testing continues to

enhance the benefit of this constantly improving methodology.

## 7. Acknowledgements

The authors would like to acknowledge Ted Cayea and Michael Degregorio, both of IBM, for doing the experimental work to determine the effect of voltage on delay defects; Mark Haviland and Mark C. Johnson, both of IBM, for delay test timing margin characterization on the tester and timing/pattern response analysis software development; Prakash Venkitaraman and Kenneth Pichamuthu, both of IBM India, for running the delay test ATPG and contributing to the methodology; and Martin Amodeo, of Cadence Design Systems, for work on the delay test timing code.

## 8. References

- [1] J. Saxena, K. Butler, et. al., "Scan Based Transition Fault Testing - Implementation and Low cost Challenges", *Proceedings IEEE International Test Conference*, 2002, pp. 1120-1129
- [2] C-W Tseng & E. J. McCluskey, "Multiple Output Propagation Transition Fault Test", *Proceedings IEEE International Test Conference*, 2001, pp. 358-366
- [3] K. Tumin, C. Vargas, R. Patterson & C. Nappi, "Scan vs. Functional Testing – A Comparative Effectiveness Study on Motorola's MMC2107TM", *Proceedings IEEE International Test Conference*, 2001, pp. 443-450
- [4] F. Woytowich et al., "Gross delay defect evaluation for a CMOS logic design system product", *IBM J. Res. Develop.*, Vol. 34, No. 2/3, pp. 325-338, Mar/May 1990
- [5] C. Hawkins, et. al., "Defect Classes – An Overdue Paradigm for CMOS Testing", *Proceedings IEEE International Test Conference*, 1994, pp. 413-425
- [6] K. Baker et al., "Defect-Based Delay Testing of Resistive Vias-Contacts, A Critical Evaluation", *Proceedings IEEE International Test Conference*, 1999, pp. 467-476
- [7] Round Table Discussion, "IC Reliability and Test: What will Deep Submicron Bring?", *IEEE Design and Test of Computers*, 1999, pp. 84-91
- [8] J. A. Waicukauski et al., "Transition Fault Simulation", *IEEE Design and Test of Computers*, Vol. 4, No. 2, pp. 32-38, April 1987
- [9] R. Bassett et al., "Boundary-Scan Design Principles for Efficient LSSD ASIC Testing", *IBM J. Res. Develop.*, Vol. 34, No. 2/3, pp. 339-354, Mar/May 1990
- [10] M. P. Kusko, B. J. Robbins, T. J. Koprowski and W. V. Huott, "99% AC test coverage using only LBIST on the 1 GHz IBM S/390 zSeries 900 microprocessor," *Proceedings IEEE International Test Conference*, 2001, pp. 586-592
- [11] E. Eichelberger and T. Williams, "A Logic Design Structure for LSI Testability", *Proceedings 14<sup>th</sup> Design Automation Conference*, 1977, pp. 462-468
- [12] P. Gillis, F. Woytowich, K. McCauley, U. Baur, "Delay Test of Chip I/Os using LSSD Boundary Scan", *Proceedings IEEE International Test Conference*, 1998, pp. 83-90
- [13] IEEE Standard for Standard Delay Format (SDF) for Electronic Design Process, IEEE Std 1497-2001
- [14] IEEE Standard for Integrated Circuit (IC) Delay and Power Calculation System, IEEE Std 1481-1999
- [15] B. Konemann et al, "Delay Test: The Next Frontier for LSSD Test Systems", *Proceedings IEEE International Test Conference*, 1992, pp. 578-587
- [16] B. Butkus et al, "Low Cost Testing of High Density Logic Components", *Proceedings IEEE International Test Conference*, 1989, pp. 550-557
- [17] C-W Tseung, R. Chen, P. Nigh, E. J. McCluskey, "MINVDD Testing for Weak CMOS Ics", *Proceedings IEEE VLSI Test Symposium*, 2002, paper 2.3
- [18] T-Y Chang, E. J. McCluskey, "Quantitative Analysis of Very Low-Voltage Testing", *Proceedings IEEE VLSI Test Symposium*, 1996, paper 10.2