

An Economic Analysis and ROI Model for Nanometer Test

Brion Keller, Mick Tegethoff, Thomas Bartenstein, and Vivek Chickermane

Cadence Design Systems, Inc

Endicott, NY

Abstract

This paper describes an Economic and Return-on-Investment (RoI) model for a test methodology that ensures product quality for logic devices that are in the 130 nm technology node and below. We describe the key components of the nanometer test methodology (NTM) and how it drives the model. In addition to ensuring product quality we address the cost of test and time to volume and how both factors can be improved. Examples from realistic scenarios are provided to illustrate the net savings from the proposed NTM using this model.

1. Introduction

Test methodologies for digital circuits have evolved with technology over the years. The first big change was when the industry went from functional test (based on simulation vectors) at the chip I/O to scan based methods. Functional tests were unable to effectively detect manufacturing defects as geometries got smaller and chips got larger. That is, functional tests typically have low stuck-at test coverage. It is an accepted fact that high stuck-at test coverage is required to ensure product quality [1]. In addition to stuck-at tests, IDDQ became an effective way to ensure product quality [2]. Up to 130nm Stuck-at tests and IDDQ were generally regarded as sufficient to maintain outgoing product quality. This is because the defects seen in prior technology node were mostly static in nature and static leakage current was low enough that IDDQ was effective for detecting many of the non-static defects.

However, at 130nm things change. At these geometries, the wiring density, signal integrity and high frequency requirements reach critical mass. Defects that once behaved as static faults are no longer static but become delay defects. In many cases, the nodes will get to their correct logic levels, but slower than required by the device clock rate. Delay defects are difficult to detect and diagnose, exposing the limitations of traditional test methodologies. Also, at 130nm the IDDQ tests have questionable effectiveness for small delay defects since the leakage current of the design goes up limiting the test resolution. Both stuck-at tests and IDDQ will still be used but they need to be complemented by test methods that can effectively detect delay defects.

As a result, the semiconductor industry will have to embrace test methods that are effective in detecting delay defects, the next standard in test methodologies. In this paper we define this methodology as nanometer test and define both the technical and economic aspects of this methodology. We create an economic model and show its predictions for a typical large nanometer design under a few different testing scenarios.

2. Nanometer test methodology

The nanometer test methodology has to ensure product quality for devices that are in the 130 nm technology node and below. In addition to ensuring product quality, it also needs to address time to volume, cost of test and yield management challenges.

The fundamental component of the nanometer test methodology is the addition of scan based delay test to detect delay defects. Delay test is also known as AC test, dynamic test or at-speed test. There are two fundamental types of delay tests – transition fault and path delay. Transition faults are an extension of the stuck-at fault model, checking for slow to rise and slow to fall defects on every gate input and output in the circuit [3]. Transition fault testing is the workhorse of delay testing for manufacturing defects. Path delay is typically used to characterize the speed of a design by testing selected critical paths at functional frequency, or at multiple frequencies for speed-binning. For the remainder of this paper, the term “delay test” will be referring to transition fault delay test, unless noted otherwise.

Delay tests normally require more test patterns than stuck-at tests, and as such require more time to apply them on the tester. This means that adding (or switching to) delay testing increases manufacturing costs compared with just static testing. This increased test time (and test data volume) drives the need for test data compression to optimize the cost of test.

In addition, nanometer technology drives the need for automated diagnostics capability to accelerate time to yield. For instance, if defect density stays constant while feature size shrinks, then yield goes down, and cost goes up. To stay on Moore’s law, we must reduce defect density. Hence, improved failure analysis is required. Also, improved diagnostics is required, and not just to

locate the more difficult delay defects; all defects, both static and dynamic, are demonstrating more complex behavior for the smaller technologies. Traditional stuck-fault diagnostics is no longer as effective as it was for the larger technology nodes.

2.1 Delay defect behavior

In order to illustrate the need for delay test please consider the following example. Engineers at Intel [4] analyzed the behavior of the circuit in Fig. 1 for different values of a resistive bridge. We will focus on the behavior of the output voltage in node “v” as a function of the resistive bridge “Res” between nodes “j” and “k”.

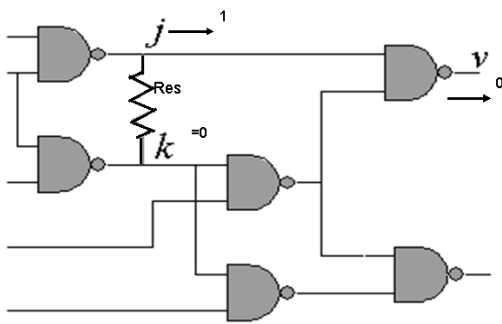


Fig . 1 - Sample circuit with resistive bridge

The output voltage “v” for different values of bridge resistance “Res” is shown in Fig 2. Note that for high resistance values the circuit is fault free, for low resistance the circuit has a true logic failure. Of interest are the resistive bridges that cause the signal to cross the speed failure zone.

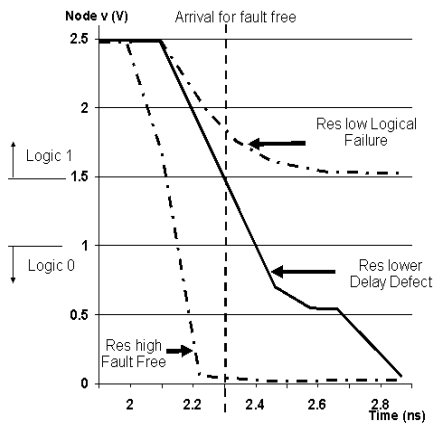


Fig . 2 - Output voltage for different resistances on “j – k” bridge

2.2 Detecting delay defects

We now use the same resistive bridge example to demonstrate the need for delay test. Fig 3 shows how different test methods perform as far as detecting delay defects. Traditional stuck-at static test is well equipped to deal with circuits that are either fault free or have logical failures. A traditional stuck-at test would be able to accurately detect the logic failure, but it may not see the timing failure unless the additional delay is substantial (a gross delay defect, longer than about 20ns in this example). In order to detect the timing failures that are not gross in nature, one needs to use a delay test.

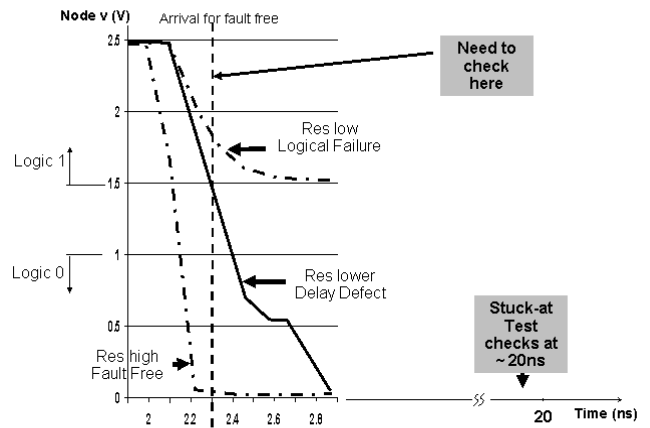


Fig . 3 - Resistive bridge response and testability

However, not all delay tests will have the same efficiency. Some delay test methods assume nominal delays. These tests will detect some of the delay defects, but since they do not have the accurate timing of the paths, they may miss many of the smaller delay defects. One needs to ensure that the delay test is effective, that it actually tests for what its coverage measure reports.

2.3 Effective delay test

There are three major success factors in using delay test: high transition fault coverage; compact test patterns; and post wiring timing. Transition fault test coverage is different than stuck-at coverage. Stuck-at can easily get to 99.5% and higher with good DFT techniques. Transition fault ATPG tests require three steps - the first is to set up an initial value on the faulty node(s), the second is to launch a transition on the faulty node(s), and the third is to observe that the faulty node(s) did not transition. The additional value assignments needing justification for delay testing drive the increased pattern counts compared with static fault ATPG. In addition, in order to achieve high coverage for delay test, one needs to have access and control to the system clock. This requirement makes the

coverage very design dependent, with coverage values often ranging from 60% to the high 90% range. In order to achieve high transition fault test coverage, the ATPG needs to support several techniques, including scan launch, broadside launch, multi-cycle paths, multiple clock domains, etc...

Transition fault pattern counts may range from 2X to 5X the count of stuck-at tests. The compaction capabilities are ATPG algorithm dependent and will vary from one vendor to another. Another factor to consider is how well a transition test handles stuck-at faults. The stuck-at coverage of transition tests will impact the overall (Stuck-at + delay) test data volume if high stuck-at coverage requires adding static tests to the delay tests so as to bring the static fault coverage up to its maximum.

Having high transition fault coverage is not sufficient to guarantee product quality. It is important to understand the effectiveness of the test in terms of timing. That is, the test needs to check for the presence of the transitioned signal at the tightest possible point in time.

A delay test methodology can use either a timing-independent model or a timing-oriented model. In a timing-independent model the ATPG generates gate transitions or path transitions with no knowledge of actual circuit delays. When test vectors are applied at a tester the transition launch and capture constraints have to be applied very conservatively. The clock timings can be iteratively tightened to locate a good, “nominal” part timing and then relaxed somewhat for testing margins, but this is a manual process relying on the experience and knowledge of the test engineer. It relies on somehow knowing that the part samples being use to empirically derive the timings do in fact perform “nominally”. It also exposes the tests to being run at timings much slower than functional if there are multi-cycle and/or non-functional paths being exercised by the tests. An alternative approach first developed at IBM uses realistic timing and delay models that are derived from synthesis or timing analysis tools. These tools accept as inputs:

- Technology cell pin-to-pin delay models
- An interconnect delay calculator
- Design Timing constraints
- Operating conditions such as temperature and voltage

They then produce a gate-level delay estimate typically used for gate-level delay simulation in a format called Standard Delay Format (SDF). Using an SDF allows the delay ATPG to select the optimal path to propagate a

transition and also automatically adjust the launch and capture constraints.

2.4 Data compression

As we have discussed, transition fault pattern counts can range from 2X to 5X the count of stuck-at tests. As the number of patterns increase, it takes more tester buffer space to hold the complete test set and it takes longer to execute the test set in manufacturing. In some instances, the tests will not fit in the buffer memory of the target tester. In either case, the result is a significant increase to the cost of testing the devices. The industry has recognized the test data volume/time/cost issues and progress has been made both in the design side, with the use of compression techniques and on the ATE side, with lower priced DFT testers.

In terms of background in test data compression, we would note that compression may or may not leverage BIST techniques, but it typically uses ATPG deterministic vectors as input, not (purely) pseudo-random data. It is also important to note that compression rate is often limited by how compact the (non-compressed) vectors are initially, as measured by the average “care bit” density. There is also an optimal level of cost of test savings. The goal is to fit on the least expensive tester available with the minimum amount of impact in the logical and physical design. It has been our experience that simpler solutions coupled with correlation intelligent ATPG consistently exceed required compression levels.

In a typical test data compression method, the ATE stores compressed scan test vectors. The compressed test vectors are supplied to the device under test using a small number of scan channels typically 16 to 64. Internal to the chip a decompressor distributes uncompressed test stimuli to a large number of internal scan channels; typically 10-100x the number of ATE scan channels. The test responses are scanned into a compressor such as an LFSR-based MISR or XOR trees. The compressor feeds the small amount of response data back to the ATE using the small number of tester scan output channels or, in the case of a MISR, using non-scan output channels.

The criteria one should consider in selecting a compression solution includes:

- achieve good compression for a given design to allow tests to fit in the tester buffers and to keep test time reasonable,
- have minimum impact on the physical design and
- ensure that the solution works well with diagnostics.

2.5 Failure diagnostics

Having delay test capability to ensure product quality and implementing test data compression to minimize test costs are key components of the nanometer test methodology. However, unless one is able to ship the product on schedule and with reasonable profit margins the problem is not solved. This is where the diagnostic techniques that automate and speed up the defect identification flow become a key component of nanometer testing. Diagnostic tools are used in two major modes. When chips fail at a tester and one is able to determine why they fail, the manufacturing line may be able to be tuned to avoid that defect in the future, improving yields. Diagnostics tools can also be used during product bring-up to significantly reduce time to volume.

The time it takes to reach insight as to why a chip fails directly impacts time to market and time to profit. In the extreme, a product never makes it to market or never has sufficient yield to generate a profit. Improvements to the diagnostics and failure analysis loop are an urgent need for the industry [5]. Typical industry experience shows that it can take weeks to get to a root cause defect for test failures. This is not a sustainable scenario. The time to insight in root cause analysis has to be reduced from weeks to a few days. This will not be feasible unless there is an accurate, automated method to move from a test failure to a physical location in the chip where the defect is actually (or likely) located. As we go to nanometer technologies substantially more defects' behavior diverges from stuck-at fault behavior.

The criteria one considers in selecting a diagnostic solution should include accurate diagnostic callouts, automated execution, high callout percentage and high callout accuracy and resolution. In addition, the solution needs to support a wide range of fault modeling capabilities that allow diagnosis of delay defects, bridges and many other complicated failure mechanisms. It may also be important to be able to perform diagnostic test pattern generation, to help diagnose problems which the manufacturing patterns simply cannot resolve.

3. Economic Model

Having defined the nanometer test problem and outlined a three step methodology to manage the challenge, we now turn our attention to understanding the financial impact of nanometer test.

We have modeled four major behaviors in nanometer test: increase in the number of delay defects in a design; use of delay test methods to detect delay defects; use of compression to reduce the cost of test and the use of diagnostics to improve yields at a faster rate. Our model

uses industry accepted equations for yield, defect level and yield learning. Below we outline the key statistical relations used in the model.

Yield model [6] – the yield model estimates the yield for a process with defect density Do , a device with area A , and an empirical complexity factor No :

$$Y = \frac{1}{(1 + Do \times A)^{No}} \quad (1)$$

Yield time function- since we are interested in modeling how diagnostics impacts yield learning, we need to model yield as a function of time:

$$Y(t) = \frac{1}{(1 + Do(t) \times A)^{No}} \quad (2)$$

Improved defect density [7] - the key variable that can be improved over time is the defect density. The improved defect density Do' is defined as a function of the nominal defect density Do and of the yield learning factor ylf :

$$Do' = (1 - ylf) \times Do \quad (3)$$

Defect Density time function - we have chosen to model the defect density improvement as an exponential function, with an initial defect density Doi , and a rate of change Ro :

$$Do(t) = Doi - Ko \times (1 - e^{-Rot}) \quad (4)$$

In order to estimate the average cost per device over the life of the device we use the Average Yield:

$$Y_{avg} = \frac{1}{T} \times \int_0^T Y(t) dt \quad (5)$$

In order to estimate the cost of quality in modeling the delay defects and delay test we model Defect level [8]:

$$DL = 1 - y^{(1-TC)} \quad (6)$$

There are additional equations used in the model, but these are the key ones.

4. Model Results

The results shown here illustrate the dynamics of nanometer test and are not intended to predict the actual economics for any specific design. Consider, for example, a design with the assumptions described in Fig 4. Entries in italics are derived from (non-italic) model inputs. A key input to the model is the delay defect %, set at 30% in this

example. This implies that 30% of the defects in this chip will be delay-only defects.

Model Assumptions and Estimates			
Logic Gates =	15 Million	Stuck-at patterns =	5,000
Scan chains =	50 Chains	Stuck-at Test Cov =	99.0 %
Chip I/O =	500	Delay patterns =	15,000
Memory =	16 Mbits	Delay Test Cov =	90.0 %
Scan Frequency =	20 MHz	Mem Test Cov =	99.0 %
# scan cells =	1,455 K	IDDQ effect DC =	48.8 %
Tech node =	130 nm	IDDQ effect AC =	36.0 %
Do =	0.20 def / in ²	Gross DPW =	213
No =	16	Nominal Yield =	56.1 %
Delay defect =	30 %	Nominal DPW =	120
Wafer size =	8 in	Initial Yield =	44.8 %
Wafer Price =	3,000 \$	Yield learning factor =	0.20
Volume =	100 K/mo	Yield learning rate =	0.04
Device lifetime =	52 weeks	Encounter diag Factor =	3.00
Cost (nom yield) =	25.1 \$ / device	CoQ factor =	1.50 *Dev cost
Tester rate =	0.10 \$ / sec	CoQ =	37.6 \$ / def

Fig . 4 - Economic model assumptions

4.1 Test Data Volume

Fig 5 illustrates what happens to the test data volume and test time as we add delay test.

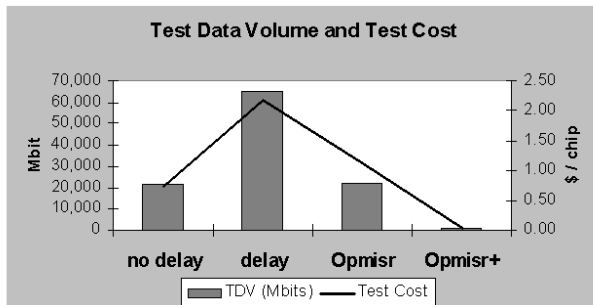


Fig . 5 - Test data volume with delay test and compression

The first behavior we want to illustrate is how to manage the 30% of defects that will behave as delay defects. The solution is to add Delay test. In this example, we define Test cost = Scan test time * Tester rate. Note that as we add delay tests, the test data volume and test time increase significantly. In order to mitigate this effect we use test data compression. In this example we modeled the effects of using output only compression (OPMISR), and input/output compression (OPMISR+). These solutions have been described in the literature [9, 10] as well as other solutions that achieve similar results [12, 13]. We note that compression is very effective, with OPMISR

bringing the test costs back to the level of stuck-at tests and OPMISR+ reducing it significantly further.

4.2 Net Savings from Cost of Quality and Cost of Test

The purpose of adding delay test is to improve the product quality. We can model this as a savings from the cost of quality. Cost of quality is defined as a factor times the part cost for each escaped defect. Once one adds delay test, the number of delay defects that escape the test will be reduced resulting on a cost of quality savings. There will also be an increase (or decrease) of the cost of test depending on the compression strategy. The graph in Fig. 6 shows the net savings dynamics for the example of Fig.4.

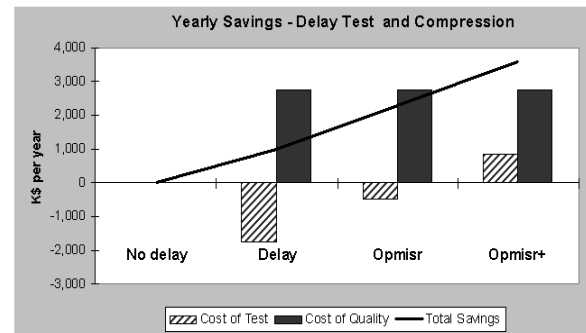


Fig . 6 - Model results - net savings delay and compression

We use the leftmost category (No delay / No compression) as the baseline. As we add Delay test, we see savings resulting from the cost of quality offset somewhat by the additional cost of performing delay testing. However, the offsetting (increased) cost of test due to delay testing is reduced when we add simple compression (OPMISR). With further compression (OPMISR+), the cost for delay testing is reduced below the original cost of just static testing, so there is an actual savings in test cost from the use of compression, which makes the net savings even greater. The graph shows the total (net) savings with a line. The NET savings is positive for delay testing without compression, more positive for delay testing with simple (OPMISR) compression, and even more positive for delay testing with OPMISR+. Of most interest is when both delay test and high efficiency compression (OPMISR+) are used. For this example, with the volume and price assumptions from Fig 4 and Fig 6, the model estimates a yearly savings of \$3.5 M\$.

4.3 Net Savings from Accelerated Diagnostics

In order to understand the effect of diagnostics in yield management we assume an exponential yield learning behavior. This is consistent with other experiences in quality improvement. The model has a yield improvement factor and rate. In the traditional case, the yield learning rate is limited by the typical multi week failure analysis loop. We assume a faster learning rate for the automated and accurate diagnostics. We have developed diagnostic algorithms that are more accurate and effective than traditional cause-effect analysis [11]. It is our experience that a 3X improvement in the yield learning rate is a reasonable assumption based on feedback from customers. Fig 7 shows the results for 52 weeks of production. Note that the yield curve with automated diagnostics has a much steeper rise and achieves a higher average yield.

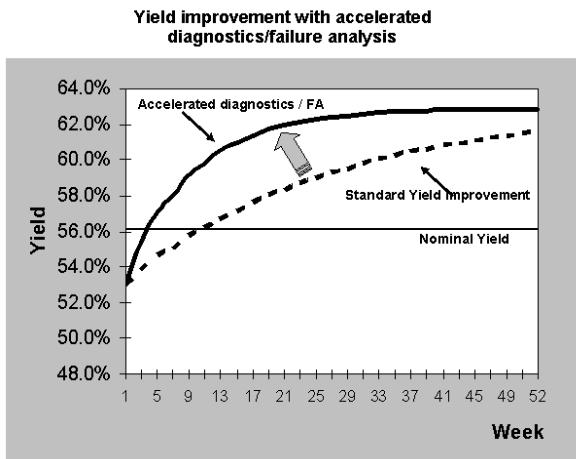


Fig . 7 - Model results – Yield curves

One can capture the savings in average yield by estimating the reduction of the average manufacturing cost per good device. The improved bar is the savings due to the traditional yield learning rate. The next two bars show the savings from using Encounter Test with a faster yield learning curve and the net savings.

Average yield and device cost for 52 weeks of life			
	Yield	DPW	\$/dev
Initial Yield	45%	95	31.43
Nominal Yield	56%	120	25.08
Avg Yield Improved	54%	115	25.98
Avg Yield Improved w/ faster learning from better diagnostics/FA	59%	126	23.77
Net Savings from Faster Yield Learning			
Device price Net Savings with Better Diag/FA = 2.21 \$			
Yearly net savings with Better Diag/FA = 2.65 M\$			

Fig . 8 - Model results – net savings from better diagnostics

Fig 8 shows the savings for one year based on a run rate of 100K devices per month.

5. Limitations

All models have limitations and the model shown here is no different. The model has many parameters, most of which are listed in fig 4. It is difficult to know *apriori* what the values of all these variables should be. The intent is to play “what if” scenarios and look at trends.

6. Discussion

Having a model to show what could happen under certain conditions can be enlightening. In some cases, the model output may appear to be counter-intuitive. For those who champion delay testing, it may come as a surprise that there are cases where the increase in quality is not enough to cover the increased cost of test. The use of test data compression helps to increase the range of cases where delay testing does provide a net savings.

7. Conclusions

This paper described the test challenges facing companies that design chips in nanometer technologies. We defined a nanometer test methodology as a combination of delay test, test compression and automated diagnostics. We further demonstrated the positive economic impact of the methodology as a function of cost of quality, cost of test, time to market and yield management.

8. Acknowledgements

We would like to thank our colleagues who helped review and suggest improvements to this work.

9. References

- [1] Abramovici, Breuer & Friedman, "Digital Systems Testing and Testable Design", Computer Science Press, 1990.
- [2] Gulati & Hawkins, "IDDQ Testing of VLSI circuits", Kluwer Academic Publishers 1993.
- [3] B. Koenemann et al, "Delay test: The next Frontier for LSSD Test Systems", Proc. International Test Conference, pp. 578–587, 1992.
- [4] Sanjay Sengupta et al, "Defect-Based Test: A Key Enabler for Successful Migration to Structural Test", Intel Technology Journal, 1st quarter 1999.
- [5] International Technology Roadmap for Semiconductors 2003 Edition.
- [6] Cheek et al, "Yield models in a Design for Manufacturing Environment: A Bibliography", SEMI Int'l Semiconductor Manufacturing Science Symposium, 1993.
- [7] Pranab Nag et al, Modeling the economics of testing: a DFT perspective, IEEE Design and Test of Computers, Jan-Feb 2002,
- [8] E.J. McCluskey et al, "IC Quality and Test Transparency", IEEE Transactions on Industrial Electronics, Vol. 36, NO.2, May 1989.
- [9] C. Barnhart et al, "OPMISR: The Foundation for Compressed ATPG Vectors," Proc. International Test Conference, pp. 748–757, 2001.
- [10] C. Barnhart et al, "Extending OPMISR beyond 10X Scan Test Efficiency", IEEE Design &Test of Computers Sept – Oct 2002.
- [11] T. Bartenstein et al., "Diagnosing Combinational Logic Designs Using the Single Location At-a-Time Paradigm", Proc. International Test Conference, pp. 287–296, 2001.
- [12] J. Rajski et al, "Embedded Deterministic Test for Low Cost Manufacturing Test," Proc. International Test Conference, pp. 301-310, 2002.
- [13] P. Wohl, J. Waicukauski, S. Patel and M. Amin, "X-Tolerant Compression and Application of Scan-ATPG Patterns in a BIST," Proc. International Test Conference, pp. 727-736, 2003