Applying the Model-View-Controller Paradigm to Adaptive Test

Nathan Kupp
Yale University

Yiorgos Makris
The University of Texas at Dallas

Editor's note:
Adaptive Testing has been a focus area for IC testing in the last few years. The "Model-View-Controller" (MVC) architecture has the potential to improve engineering productivity for analysis and application of Adaptive Testing. The vision of MVC is that it will be used to share Adaptive Test applications and analysis across industry and academia.
—*Anne Gattiker and Phil Nigh, IBM*

Despite the growing complexity of semiconductor devices, adaptive test deployments for such devices remain elusive. The high cost and difficulties of developing a custom adaptive test solution are likely causes for this sparsity. Indeed, there are many requirements that an adaptive test solution should meet. A good adaptive test system should dynamically adjust to process variation statistics that may not be stationary. It should modulate the test program at high granularities (die level or even subdie level), have low adaptation latency to achieve real-time adaptivity, and comfortably handle learning from terabyte-scale historical test data. Moreover, it should achieve these goals without violating stringent industrial requirements on permissible test error. Finally, many of the specific details of these requirements are likely to differ across product lines, e.g., consumer products certainly have very different comfort levels for permissible test error than automotive products.

However, the advantages of a successful adaptive test deployment are clear: test time reduction, test quality improvement, improved data analysis ability, and acceleration of yield learning. Therefore, substantial industrial interest exists for making adaptive test a reality in the near term. This interest is driving groups from a broad swath of industry to collaborate on adaptive test development. The Test & Test Equipment team within the International Technology Roadmap for Semiconductors (ITRS) has created a subgroup specifically targeted at the adaptive test problem [1].

The requirements that an adaptive test system must meet can be broken down to three discrete sets of challenges. First, there is the problem of data handling: device, wafer, and lot measurements should be linked and traceable throughout the fabrication process, data should be standardized and easy to exchange between adaptive test programs and throughout the organization, and large data sets should be readily accessible. Second, innovation within the adaptive test community is rapidly increasing the number of adaptive test techniques that have been demonstrated in literature. Thus, an adaptive test system should simplify integration of new adaptive test techniques as they become available. Third, test engineers should be provided with a convenient means of obtaining deep visibility into the inner workings of an adaptive test system. For

Digital Object Identifier 10.1109/MDT.2011.2179370
Date of publications: 15 December 2011; date of current version: 30 April 2012.

example, the system should provide real-time statistics and information on the adaptive test techniques which have been deployed as well as a detailed history of how the test flow has been adapted to process conditions over time.

As it turns out, developers creating modern web applications have already largely solved these challenges. The model-view-controller (MVC) software architecture, originally described in [2], is a paradigm that prescribes specific patterns for partitioning and solving software engineering problems. In this work, we argue that MVC is also a particularly suitable approach to adaptive test for semiconductor devices, as it provides a means to decouple the challenges of adaptive test analysis from data representation and presentation. Moreover, solving these challenges within an industry-standardized MVC framework is clearly superior to deploying custom solutions across the industry. In this work, we lay the groundwork for applying MVC design patterns to adaptive test, in particular, for analog and radio-frequency (RF) devices, although the principles developed are applicable across the broader semiconductor test community. Analog and RF test is a particularly rich domain given the high availability of parametric test data and high cost of test.

We envision that broad adoption of MVC as the path forward for adaptive test will lead to an "application ecosystem," where university researchers or third party vendors can release adaptive test applications as MVC controller modules that can be readily integrated into existing adaptive test systems without significant customization. Thus, by adopting a standardized MVC platform, novel open-source or commercial adaptive test techniques can be released, tested, and deployed with near-zero custom code; fragmentation of industry solutions is avoided; and broader industrial adoption can be achieved.

Ultimately, we see the MVC design pattern not only as a solution for adaptive test, but also as the bedrock of our vision for learning from semiconductor data. Ongoing research activity across the semiconductor industry is focused on topics such as yield learning, adaptive test, failure analysis, etc. We see all of these as symptomatic of a broader problem: the need for a unified semiconductor learning and data analysis solution. Adopting a standardized platform built on a time-tested design pattern such as MVC would assist in combating stratification of industry data sources and hasten learning by unifying data across semiconductor manufacturing. The proposed data analysis methodology, applied in this work to adaptive test, is readily extensible across all of semiconductor manufacturing.

Models, views, controllers

The MVC architecture comprises three components: models, views, and controllers. We detail the specifics of each component and link them to adaptive test challenges within this section.

Models

The "model" component of the MVC architecture describes a means of abstracting data into standardized structures, typically implemented as a layer positioned directly above low-level relational databases, as shown in Figure 1. These can effectively be considered as high-level database abstractions that simplify software engineering, while allowing the low-level databases to take care of database infrastructure, data storage, and data integrity problems independently. For example, a chip model may have $\{x, y\}$ coordinates, measurement data, and a wafer ID. A wafer model then would be treated as a collection of chip models, perhaps with a product ID linking it further up the hierarchy. Depending on the specific implementation, the test engineer may also desire to encapsulate data from different stages of fabrication and associate such data with each chip. A chip model would then be defined as containing collections of these models from throughout the fabrication process. Thus, once defined, such abstract model objects encapsulate key data segments

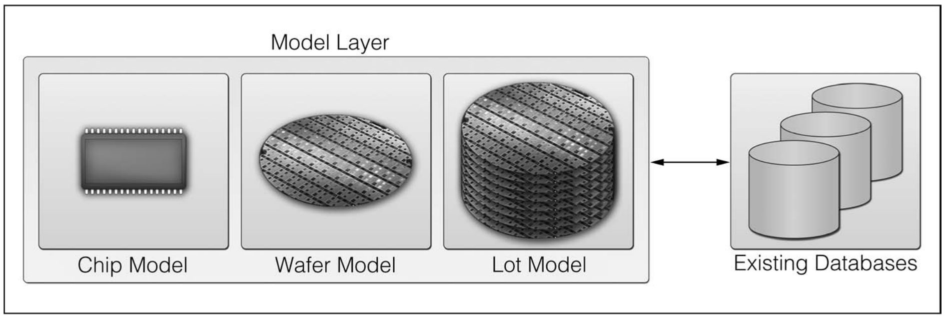


Figure 1. Model architecture example.

from throughout the entire semiconductor fabrication process, and form the atomic elements that are similarly transported and analyzed throughout the MVC architecture. In terms of adaptive test, industry already collects massive amounts of semiconductor test data in industrial information warehouses. Layering abstract MVC models on top of such information warehouse databases is a straightforward task, and once performed, enables complete standardization of analysis approaches across an organization and even interorganization across the industry by completely decoupling database-specific logic from analysis tasks.

Views

The "view" component of the MVC architecture provides a means for the test engineer to directly observe and interact with adaptive test analyses during execution. Views typically comprise graph, chart, or table-generating code, and collectively titled presentation logic. In the context of adaptive test, such views are designed to provide the test engineer with information on the adaptive test techniques in use and present related statistical data. The important feature of views is the total separation of presentation logic, which is encapsulated in the views, from database logic or business/analytic logic, which is located in models and controllers, respectively. Thus, generalized views can be defined which aggregate results from several controllers, and views can be developed and debugged independently from controllers.

Controllers

Last, the "controller" component of the MVC architecture combines models and inputs from the user (via the views), and uses these inputs to perform actions on the data encapsulated in the models. We expect that specific analysis tasks can live in the MVC system as "applications"; with sufficient development effort and attention from industry, we hope this could develop into a complete application ecosystem, with application-specific controller modules broadly available for use. Given this ecosystem of modular controller applications, custom analysis tasks can be realized with dramatically lower complexity by simply wiring together the appropriate controllers with custom code. At the same time, the test engineer's full control over the analysis is not sacrificed: all controllers can act as superclasses for user code allowing for complete customizability and overriding of default behaviors, if required. For example, consider the support vector machine (SVM) controller developed in the experimental demonstration. Because application-specific code is pushed into the models, the SVM controller is sufficiently abstracted to suit a broad range of applications. On the other hand, implementing a subclass is always possible if a particular application requires a unique SVM implementation.

Putting it all together

In Figure 2, we show a high level diagram of the MVC paradigm as applied to semiconductor device fabrication. During semiconductor fabrication, an immense wealth of data is captured from wafers and devices. Our proposed MVC framework would overlay seamlessly on top of the existing flow. A platform controller handles basic functionality, and additional functionality is added to the system via MVC "applications," or additional controller modules. Finally, web-based views present the test engineer with live results and the ability to interact with the controllers in place.

One element of the infrastructure that is not presented

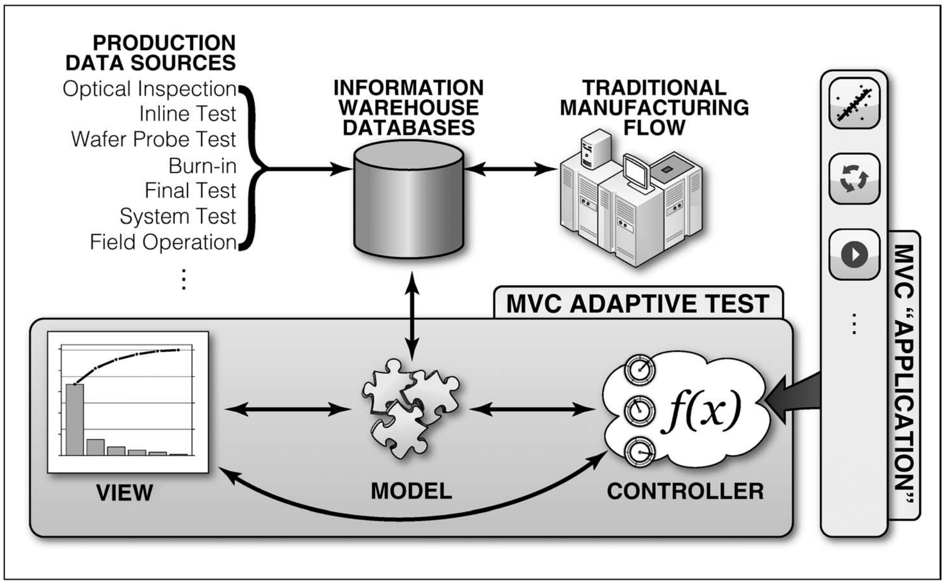


Figure 2. Overview of proposed MVC framework.

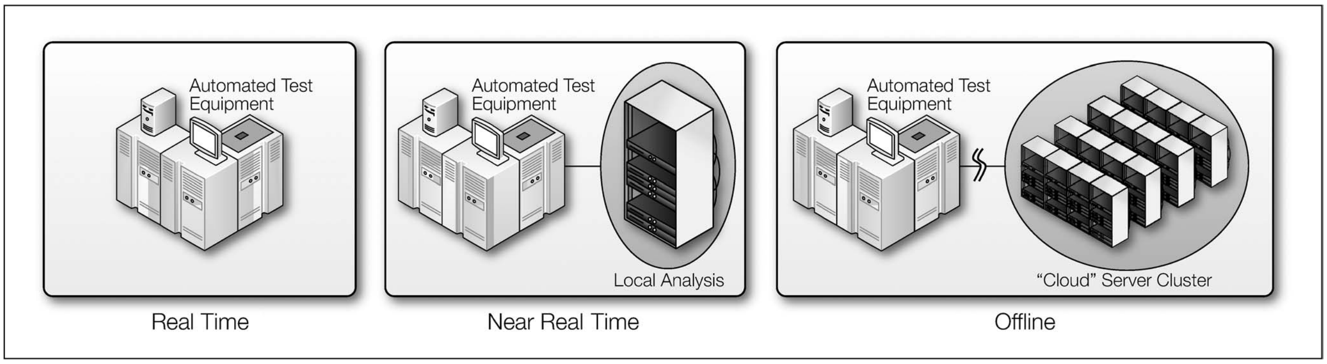


Figure 3. Temporal locality of analyses.

explicitly in the diagram in Figure 2 is the high temporal specificity of adaptive test analysis tasks. Such tasks typically fall into three categories, as shown in Figure 3. *Real-time* tasks should be addressed directly on the testing platform to ensure precious tester time is not wasted. Obvious constraints exist on the complexity of analysis that is possible at this level, and code should be written to ensure very fast operation. *Near-real-time* tasks should ideally be accomplished on-tester or nearby, permit for slightly higher complexity of analysis, and generate results or actionable test choices in very short time frames, e.g., coinciding with wafer-level test adaptation instead of die-level test adaptation. Finally, *offline* analysis tasks do not immediately interact with the test plan, but instead affect test plans at time scales of a few seconds or minutes, at the wafer level or above. The advantage of performing analysis tasks offline is the orders-of-magnitude increase in complexity of analyses that can be performed offline: data from many wafers can be aggregated and analyzed jointly to provide deeper insight into the statistics of the fabrication process.

As the majority of machine-learning-based test research published to-date is grounded in the offline locale, this work presents a case study and develops indispensible components of an MVC framework targeted at offline adaptive test analysis. Specifically, we tackle the problem of early test metric estimation when deploying a low-cost test adaptive test system within the context of an MVC framework. We plan to extend the proposed MVC framework to address near-real-time and real-time adaptive tests, as these are natural extensions of the offline adaptive test MVC framework.

MVC implementation

The particular experiment presented in this paper is designed to showcase a specific use case for the proposed MVC framework by targeting the problem of low-cost testing for analog and RF devices. Low-cost testing is concerned with developing alternative test methods that reduce the cost of testing devices without sacrificing test quality. This is a longstanding area of interest in the test community, as traditional specification testing is extremely expensive by comparison. Anecdotally, test cost for analog and RF devices can reach as high as 50% of the manufacturing cost.

Specifically, the analysis presented herein addresses the challenge of cost-effectively evaluating candidate test methods. Some approaches to low-cost testing may introduce intolerable test metrics, while others may not achieve sufficient test cost reduction to merit implementation. Identifying these error rates correctly and efficiently early on can dramatically reduce the risk of bringing low-cost testing online in production. In this section, we outline the software components we architected to implement a low-cost test evaluation methodology within our MVC framework.

Models

We implemented several models for the analysis demonstrated in this work. Specifically, we required encapsulation of the specifications for each performance, data sets with device-level measurements, and within each wafer we wished to break out low-cost tests and specification performances into separate collections to facilitate the use of low-cost tests with a trained classifier, and derivation of pass/fail labels for the performances via the specifications.

Thus, we defined a specification class that encapsulates the device specifications and provides key-value access to upper and lower limit specifications for each performance. For each wafer data set, we defined a general hierarchical data model for our framework. The data model class is instantiated by passing a comma-separated value (CSV) filename to the constructor, which loads device data into a standardized data structure containing column names, device-level raw data (i.e., each row corresponds to a single device), a pass/fail matrix (which stores the result of comparing each cell in the raw data matrix to device specifications), and an overall pass/fail label column. This data structure is the atomic model unit of our MVC framework. Finally, we added convenience methods directly to the data structure for subsetting by column/row, joining by column/row, and a CSV export method.

However, as noted previously, it is often desirable to partition data into various subsets for many common analysis tasks. Thus, within our framework, we also implemented a higher level data set model (associated with each block of data from, say, a wafer) possessing a collection of DataStructs. As with the DataStruct, the data set model was defined with associated convenience methods: subsetting, joining, exporting to CSV, and computing the pass/fail matrices for all subordinate DataStructs.

Views

In offline analyses, the most important information to provide in a view is 1) a running log of the analysis, and 2) charts which describe the results once the run has terminated. Typically, user interaction during the run is not required for offline analyses. Thus, our views heavily relied on the model view convenience classes to output model information during the analysis run, along with several helper methods that generated plots from the outcome of the analysis. The information generated by these views was used in collecting the experimental results presented in the subsequent "Results" section.

Controllers

Last, we implemented three controllers within the context of our MVC framework. The first controller implements nonparametric kernel density estimation, the second implements Laplacian score feature selection, and the third implements an SVM. Details of the implementation of each of these controllers, and rationalization of their necessity for the early test metric estimation adaptive test analysis task undertaken in this work, can be found in [5]. Each controller operates on a DataStruct argument and parameters required to complete the designed task. Note that the controllers are defined to be as general as possible, and are not specific to the analysis at hand, maximizing code reusability. The MVC framework we have implemented is shown in Figure 4, showing all models, views, and controllers; a distinction is made between general modules constructed as part of the framework and analysis-specific modules required for the example analysis demonstrated in this work.

Experimental demonstration

As a first deployment of our proposed MVC framework for adaptive test, we analyzed a data set

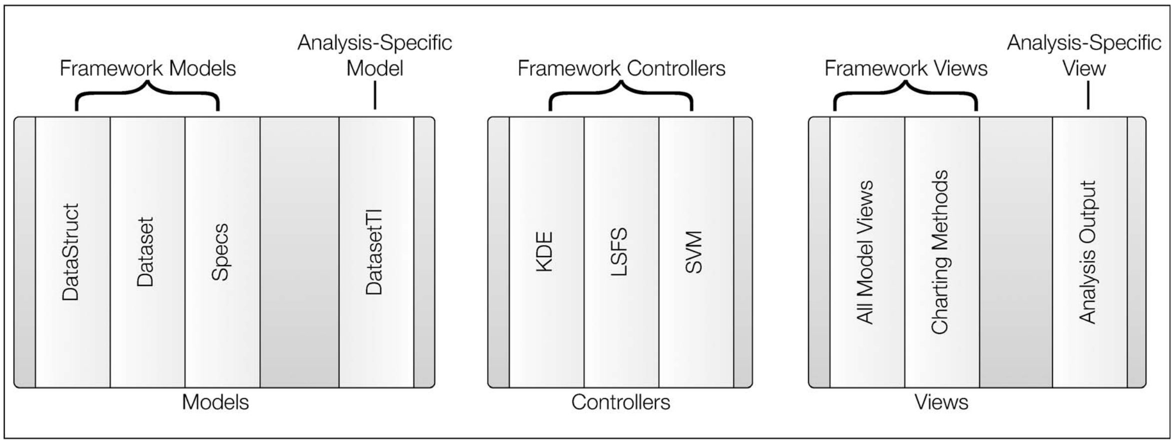
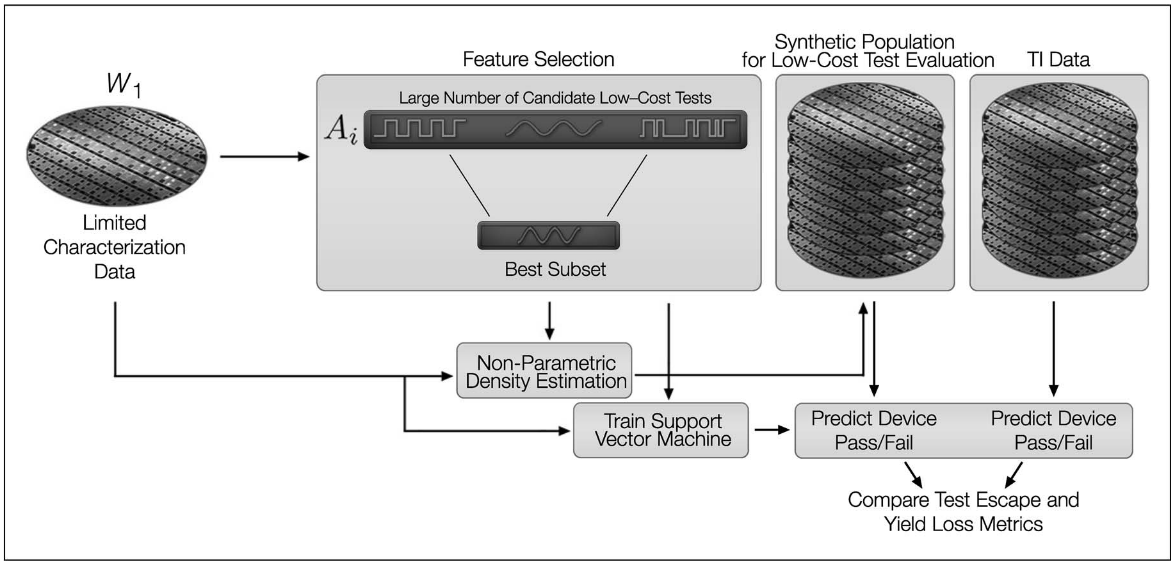


Figure 4. MVC framework implementation.

from Texas Instruments including more than 1.1 million Bluetooth/wireless local area network (WLAN) RF devices and more than 1000 measurements per device. Our objective was to construct a tool running on the MVC platform that provides early estimates of test escapes (T_E) and yield loss (Y_L) test metrics incurred by low-cost testing [6], [10]. This tool was based on the approach described in [7], where an early sample of devices is used to evaluate a candidate set of low-cost tests. To date, most low-cost test implementations have provided test metric estimates based on a limited data set, e.g., from one wafer. Given such small test sets, it is uncertain whether the test metric estimates are reliable at the parts-per-million levels required by industry. Here, we present an MVC framework-based general technique to obtain reliable parts-per-million early test metric estimates of candidate low-cost test techniques without incurring the cost of explicitly testing one million devices. A more involved discussion of this methodology can be found in [5], and we refer the reader to [8] and [9] for further background on the state-of-the-art in machine-learning-based low-cost testing.

To accomplish the early test metric estimation objective within the MVC framework context, we implemented all of the models, views, and controllers of Figure 4. The total framework lines-of-code count is approximately 1500, with about 150 additional lines-of-code for the particular analysis presented herein. The first production wafer is used along with the Laplacian score feature selection controller to reduce the number of candidate low-cost tests from more than 700 to 10. Nonparametric density estimation was used to generate a synthetic population of one million devices based on the measurements collected on the first wafer. An SVM controller was trained on the first wafer data, and T_E and Y_L estimates were obtained by applying the trained classifier to the synthetic population. Given the large data set from Texas Instruments, we were then able to verify the efficacy of the proposed approach by similarly predicting on the complete true data set and comparing the test metric estimates with the true test metrics. The complete analysis flow is shown in Figure 5. A complete experiment run took approximately 4 h on a 2010 2.4-GHz Core i5 processor.

Finally, we present the results of the analysis in Figure 6, where we show realized test metrics (in percent) against each wafer as the difference with respect to the long-term mean. The test escape and yield loss estimates are presented as the solid horizontal lines. As can be observed, test escape is slightly overestimated, and yield loss is slightly underestimated. Specifically, the early estimates differ from the true long-term means by ΔT_E 0.491% and ΔY_L 0.517%. We remind that the objective of this analysis is not to propose a state-of-the-art alternative test technique, but to evaluate a candidate



W_1
Limited Characterization Data
Feature Selection
Large Number of Candidate Low–Cost Tests
A_i
Best Subset
Synthetic Population for Low-Cost Test Evaluation
TI Data
Non-Parametric Density Estimation
Train Support Vector Machine
Predict Device Pass/Fail
Predict Device Pass/Fail
Compare Test Escape and Yield Loss Metrics

Figure 5. MVC framework implementation.

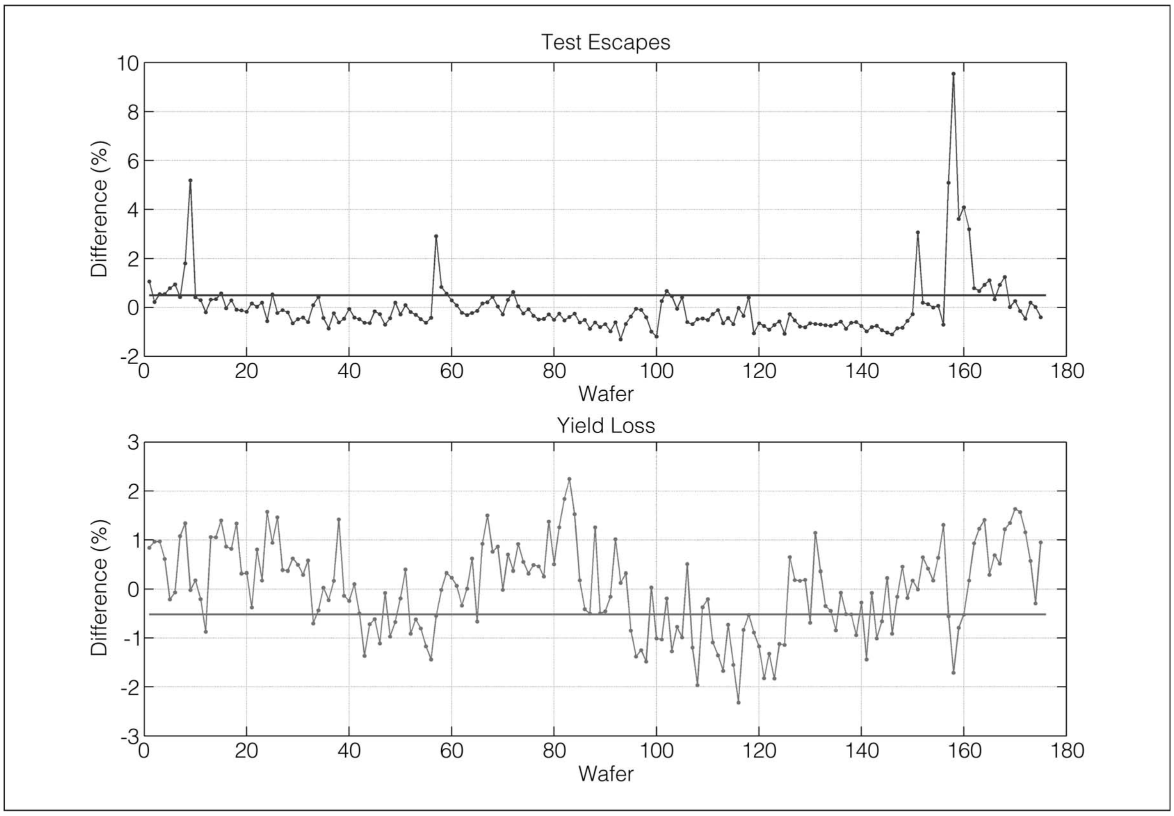


Figure 6. Prediction across all wafers: test escapes and yield loss.

technique at an early phase; that is, the key indicators of success in this case are ΔT_E and ΔY_L, and not the absolute values of the observed test metrics. Thus, via the proposed MVC framework, we have implemented an offline adaptive test analysis methodology that efficiently estimates viability of candidate low-cost test methods prior to full production.

In software engineering, there are a plethora of design patterns available. To some degree, the classification of software design patterns as good or bad is qualitative. Our litmus test for whether an effective design pattern has been found is when the time spent chasing marginal bugs dramatically decreases. Our group has been working on low-cost testing problems similar to the analysis presented herein for quite a long time, and every new analysis we undertook often required a great deal of custom code. With the move to MVC, our turnaround time from problem definition to meaningful results has improved significantly, and we have spent far less time fixing marginal bugs due to the effective abstractions MVC affords.

We have outlined some of the key ideas forming the basis of an adaptive test infrastructure based on the MVC paradigm. Our MVC platform is in the very early stages of development; however, we are already able to demonstrate publishable adaptive test analysis tasks. We implemented kernel density estimation and an SVM as controller applications, and look forward to continuing development as much work remains to be done. Finally, in the interest of making the MVC platform for adaptive test an industry standard, we have open-sourced all of our code with a nonviral MIT license at https://github.com/trela/qikify. The software is decidedly in pre-alpha state, but we enthusiastically invite participation from the adaptive test community in developing a mature MVC framework to serve as an industry standard for adaptive test deployment. ■

Acknowledgment

This work was supported by the National Science Foundation (NSF CCF-0916803 and CCF-0916415) and the Semiconductor Research Corporation (SRC-1836.073). The work of N. Kupp was supported by an

IBM/GRC (Global Research Collaboration) graduate fellowship. The authors would like to thank E. de-Obaldia, J. Carulli, and K. Butler of Texas Instruments for their provision of and technical support for the data set.

■ References

[1] ITRS Adaptive Test Subgroup. [Online]. Available: http://icdt.ece.pdx.edu/~icdt/cgi-bin/adaptive.cgi/AdaptiveTest

[2] T. Reenskaug, "Thing-model-view-editor," Institutt for Informatikk, Univ. Oslo, Oslo, Norway, Tech. Rep., 1979.

[3] S. Ghemawat, H. Gobioff, and S.-T. Leung, "The Google file system," *SIGOPS Oper. Syst. Rev.,* vol. 37, no. 5, pp. 29–43, 2003.

[4] J. Dean and S. Ghemawat, "MapReduce: Simplified data processing on large clusters," *Commun. ACM,* vol. 51, no. 1, pp. 107–113, 2008.

[5] N. Kupp, H.-G. D. Stratigopoulos, P. Drineas, and Y. Makris, "PPM-accuracy error estimates for low-cost analog test: A case study," in *Proc. 17th Int. Mixed-Signals Sensors Syst. Test Workshop*, 2011, pp. 43–47.

[6] P. N. Variyam, S. Cherubal, and A. Chatterjee, "Prediction of analog performance parameters using fast transient testing," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.* vol. 21, no. 3, pp. 349–361, Mar. 2002.

[7] H.-G. D. Stratigopoulos, S. Mir, and A. Bounceur, "Evaluation of analog/RF test measurements at the design stage," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 28, no. 4, pp. 582–590, Apr. 2009.

[8] H.-G. D. Stratigopoulos and Y. Makris, "Error moderation in low-cost machine learning-based analog/RF testing," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 27, no. 2, pp. 339–351, Feb. 2008.

[9] N. Kupp, P. Drineas, M. Slamani, and Y. Makris, "On boosting the accuracy of non-RF to RF correlation-based specification test compaction," *J. Electron. Test.*, vol. 25, pp. 309–321, 2009.

[10] D. Mannath, D. Webster, V. Montano-martinez, D. Cohen, and S. Kush, "Structural approach for built-in tests in RF devices," presented at the Int. Test Conf., Austin, TX, Nov. 2010, Paper 14.1.

Nathan Kupp is pursuing a PhD in electrical engineering at Yale University. His research interests include applying machine learning and statistical learning theory to problems in analog and RF test. He has an MS in electrical engineering from Yale University. He is a student member of IEEE.

Yiorgos Makris is an associate professor of electrical engineering at The University of Texas at Dallas. His research interests include test and reliability of analog, digital, and asynchronous circuits and systems. He has a PhD in computer science and engineering from the University of California, San Diego. He is a senior member of IEEE.

■ Direct questions and comments about this article to Yiorgos Makris, Electrical Engineering Department, The University of Texas at Dallas, 800 W. Campbell Rd., ECSN 4.914, Richardson, TX 75080-3021, USA; yiorgos.makris@utdallas.edu.