Adaptive Group Testing
- a DUT Response Technique applied in Semiconductor Test

Dr. Gunther Karner, Dr. Ivica Rogina

g.karner@optimise.de, i.rogina@optimise.de
optimiSE GmbH
Haid-und-Neu-Str. 7, 76131 Karlsruhe, Germany

Abstract

DUTs failing in a certain test often announce their deficiency by ‘abnormal’ test results earlier in their test flow. The observed regularity of those patterns provides the foundation for a decision algorithm, able to switch on and off selected test groups on a die-by-die granularity. Thereby, switch-decisions are based on forecasted potential fails in the groups, which in turn are predicted by a real-time analysis of ‘Signature Tests’. Those tests are executed earlier in the flow and display high ‘Knowledge Correlations’ to the respective test groups.

A tester-based decision automaton, operating on these principles and implemented in a production test environment for Dialog Semiconductor, considerably reduces test time while preserving the original product quality.

1. Real-Time Adaptive Test

To start with, a quote characterizing the general subject of this presentation [1]:

Adaptive Test is concerned with making predictions about the behavior of the DUTs from the statistical distributions of their measurements.

(For more information on the topic we refer to [2].)

More precisely, our focus is on real-time monitoring of test results and, consequently, ‘very short-term’ predictions in the sense that test flows are dynamically changed by adding or removing selected test groups as a function of prior Signature Test readings on the same device.

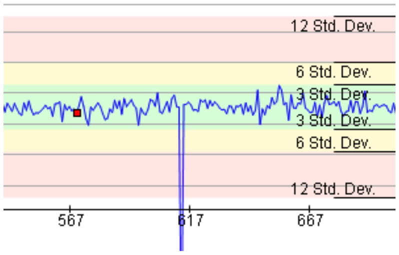
2. From Baseline C_{pk} to DUT Response

The most common form of flow adaptation is Sampling, which is based on the assumption that measurements in ‘statistical control’ need not be executed on every part, in order to verify that all parts comply with quality requirements.

No need to stress that the crux of the matter lies in the implementation of the term ‘statistical control’!

There are, besides mandatory and redundant tests, which can be handled differently [3], so-called ‘ugly’ tests. Those are almost always in a 6-sigma strip, often highly correlated within their particular test group, yet without warning they fail on a part while are ‘back to normal’ on the next one.

A proper example of that kind is



At a sampling ratio of 10, the C_{pk} over the last 10 fully tested parts before the fail equals 4.71! Thus, chances to predict that defect by a baseline C_{pk} -type of monitoring are negligible and another, more subtle strategy has to be applied!

In the sequel, we sketch a technique which is based on the observation that defective DUTs often display out-of-the-way results in certain trials before actually failing in another test further down the flow. The existence of this particular type of defect correlation has been confirmed in numerous analyses of large data sets [4]. In cases where those exceptional readings manifest themselves as statistically reliable entities, they can be used as predictors of potential fails.

The quantification of such patterns conducts to the DUT Response Algorithm outlined below, which is able to prevent escapes of defective parts to the field even in situations where ‘ugly’ fails occur. That ‘dynamic Quality Management’ is basically achieved by turning on and off corresponding test groups at the right time and the right device.

The technique employs both historical and real-time analyses of test results for pattern learning, respectively prediction purposes.

3. Historical Data Analyses prepare the Grounds

As far as logged data are concerned, the present approach proceeds towards the above stated goal of a DUT Response Algorithm as follows:

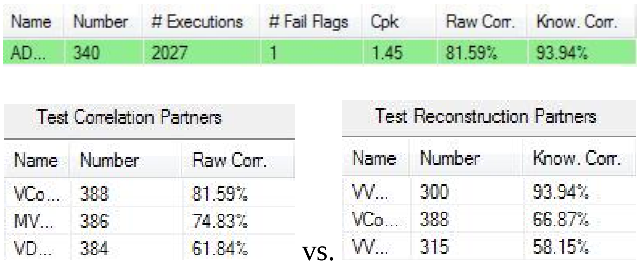
(i) Introduce so-called Knowledge Correlations, which are computed similar to classical (or ‘raw’) correlations, but accentuate critical values such as Bin1-outliers and fails due to a specific pre-processing of the actual measurements.

(ii) Employ a neural network to reconstruct results of ugly tests from (an) other test(s) high in Knowledge Correlation. The best predictors form the set of potential fail indicators (aka Signature Tests).

(iii) Determine quantitative switch thresholds from the observed qualitative defect correlations in order to catalogue the effectiveness of the Signature Test candidates in predicting potential fails.

To illustrate items (i)-(iii), take the following display as an example:

When examined over a certain wafer, a Test #340 shows one fail and very high Knowledge Correlation with another test, namely Test #300:



Name	Number	# Executions	# Fail Flags	Cpk	Raw Corr.	Know. Corr.
AD...	340	2027	1	1.45	81.59%	93.94%

Test Correlation Partners

Name	Number	Raw Corr.
VCo...	388	81.59%
MV...	386	74.83%
VD...	384	61.84%

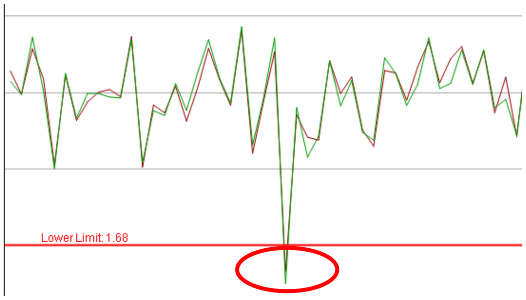
vs.

Test Reconstruction Partners

Name	Number	Know. Corr.
W...	300	93.94%
VCo...	388	66.87%
W...	315	58.15%

With #300 as input, the neural network reproduces the respective values of #340 very well, when trained on some other subset of data from the same wafer.

In particular, the net is able to predict the fail:



(Where the predicted trend plot is in red and the actual measurements is green.) Note that, on the other hand, the reconstruction by the highest raw-correlated test (#388) entirely misses the fail:

Hence, #300 is a possible Signature Test for #340, since it is executed earlier in the test flow.

To determine efficient ‘switch-on/-off’ conditions for the test groups of interest, the performance of the selected Signature Tests in the vicinities of fails in those groups is examined and quantified.

For reasons of competition, we cannot display the detailed formulas and calculations. Nevertheless, a sketch of the procedure will help:

Test #300 displays a distinct spike at the part failing in Test #340, when compared to its measurements on the devices tested just before. (By the way, that explains why the Knowledge Correlation #300 <-> #340 is almost perfect.)

In order to provide a numerical indicator for future potential fails in #340, which can be checked in real-time, the graph suggests that the result of #300 at the fail device is evaluated against a certain ‘local average region’ of #300. The latter is calculated from its readings at a collection of parts tested just before the defective one.

The number of free parameters in this calculation is just two, namely the weight of the ‘average’ against the spike and the size of the device window before the fail.

Consequently, any future out-of-the-local-average-conduct of #300 will be understood as an indication of a potential fail in #340.

Evidently, to indeed qualify Test #300 for the task of being a Signature Test to #340, verification of their defect correlation is required on a much larger data set, which in particular contains sufficient fails in #340.

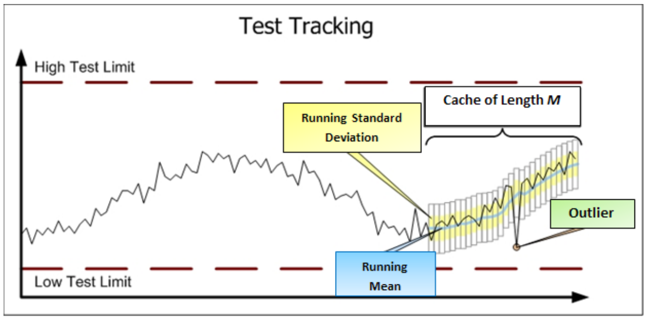
Naturally, in a real-world application (cf. Section 5) the product specialist has to confirm the proposed Signature Tests from a technical point of view, which concerns the applicability of the modified test program.

4. Real-Time Decisions keep Escapes away

Having determined a set of Signature Tests to an Adaptive Group (i.e., a group to be switched on/off dynamically) in an analysis of a large set of historical data (in the order of 10^6 parts) with our proprietary data analysis software tool, the next important question concerns the proper execution of the forecast and group switching processes in real-time.

To this end, the computation sketched in Section 3 is repeated in a die-by-die rhythm. That is, the Signature Tests are executed all time and the set of their most recent values, say [1, *N*-1], are kept in memory. Those readings enter the determinations of the 'local average regions', against which the results of the Signature Tests at Part *N* are assessed. Depending on the 'being inside or outside' scores of the latter, a potential fail in a defect-correlated Adaptive Test Group at the Part *N* is predicted.

On Device *N+1* the procedure repeats itself, now with the set [2, *N*] of the Signature Test results as entry of the 'local average region' calculation. And it keeps on going that way… Schematically, the tracking procedure looks as follows:



An important plus-factor of that practice is its ability to adapt to fluctuations in product quality stemming from fab and/or hardware variations.

Given a sufficiently broad historical analysis, even sporadic fails like the ones discussed in the preceding sections can be handled in almost all cases.

In addition, as the number of free computation parameters is just two, the actual escape risk of a defective part 'leaking' to the next test regime or into the field can be easily calculated in simulations and adjusted to the current product range under test.

Remark that the method is independent from the actual test regime, i.e., it applies both in Wafer Probe and Final Test. Moreover, in a case study at one of the leading EMS providers, it has been exerted to In-Circuit and Functional Tests of Printed-Circuit-Boards [5], with principally the same results as far as benefit and risk are concerned.

5. The tester-based Automaton

Employing the DUT Response technique as described above, the so-called *Tester Driver* or *Adaptive Group Test Controller* (AGTC) instructs the test machine to execute or skip Adaptive Groups on the actual DUTs, respectively plunges in multi-site testing.

That tester-based unit is an ultra-fast decision-making automaton, which is linked to the test program and receives real-time results from the Signature Tests.

No additional hardware is required to run the AGTC, only a minor one-time adaption of the test program (see the example below).

At the beginning of the test program, the AGTC is initialized by reading a configuration file, which contains all the necessary information on Signature Tests, data window size, etc. Subsequent changes in the decision algorithm can be made by updating this file without the need for modifying the test program itself.

Of course, the employed tester infrastructure has to allow handshakes of that kind. In particular, it has to enable every Signature Test to permanently pass in 'no time' its results to the AGTC in a specific function call such as *oatcLog*(oatc,300,result).

Why entire test groups rather than individual tests? The example from a Catalyst test program explains:

A test group can be easily handled inside an "if" statement (1=execute, 0=skip), where *oatcDo* is the automaton call:

```
if (oatcDo(oatc,340)==1)
seq Adaptive Group()
{
TestNr(340)…
}
```

In this way, the test time of the entire group can be saved, whereas turning on and off individual tests is technically more elaborate and, in most cases, provides only minor test time gains since group set-ups, etc., have to be performed in that case anyway.

The final concern is about possible test time overhead generated by the AGTC. Here we can give the all-clear, since the measured overhead per automaton decision on a standard Catalyst tester is approximately 0.3 ms. Typically, two to four Signature Tests are necessary to control an Adaptive Group. As the AGTC preferably acts on groups of sufficiently long duration, the additional few milliseconds do not hurt at all.

A first implementation of our (patented) Adaptive Group Test Controller on Catalyst testers, used in production testing for Dialog Semiconductor, displays a steady test time reduction in the double digits. No quality problems have been reported so far while the number of parts tested is already in the several millions.

3. References

[1] P. M. O'Neill, "Adaptive Test", *http://icdt.ece.pdx.edu/~icdt/cgi-bin/adaptive.cgi/AdaptiveTest*

[2] Various Authors, "ITRS Adaptive Test Section", *http://icdt.ece.pdx.edu/~icdt/cgi-bin/adaptive.cgi/ITRSWorkingGroup*

[3] I. Rogina, H. M. v. Staudt, G. Karner, "Controlling Test Plans by Information-Content-Based Redundancy Analysis", Proc. IMSTW 2005

[4] G. Karner, "Confidential Data Analyses for Clients", 2010 ff.

[5] O. Bachlechner, G. Karner, M. Kuschnig, "Adaptive Testing in Electronics Manufacturing Services – A Case Study", 12th European Manufacturing and Test Conference (EMTC), Dresden 2010.