TABLE OF CONTENTS

INTRODUCTION

This document describes the implementation and operation of Part Average Testing (PAT) on the Cassini ATE platform.

WHAT IS PAT

PAT is an acronym for “Part Average Testing.” The goal of PAT as described by the AEC (Automotive Electronics Council) is to identify individual parts within a lot that are behaving differently from all the other parts within that lot. These parts, referred to as Outliers, represent some portion of the lot that has an underlying flaw or deficiency usually manifesting in shorter life expectancy even though they pass static program test limits. The idea is that traditional test limits cannot not accurately screen and/or characterize this failure type from the mean population.

PAT uses statistics to determine the mean and standard deviation of each test result and then uses those statistics to set limits on how far a test result may deviate from the general population before the part is considered an outlier. Generally accepted methods of determining the statistics are implemented with Static PAT (SPAT) and Dynamic PAT (DPAT).

SPAT

Static PAT uses the history of a test based on many lots previous to the current lot being tested. The SPAT test limits are calculated before the lot is run and are applied throughout the testing of the lot. Because of the amount of data required by SPAT, test limits are only changed very infrequently and usually by some kind of ECO procedure to code the limits into the test program itself.

DPAT

Dynamic PAT uses test results from the current lot to calculate the statistics. The DPAT limits are then calculated and reapplied to the parts already tested. DPAT is usually implemented at wafer probe because of the ease with which the entire wafer can be tested and then, after the new PAT limits are calculated, the wafer can simply be remapped using the new test limits. Once implemented, the test program does not need to be changed to change the PAT test limits. They are calculated dynamically as each lot is tested.

IPAT

iPAT stands for “Inline Part Average Testing.” The word “Inline” denotes that the calculation of PAT statistics and implementation of the computed test limits are concurrent with testing of the lot. With iPAT, the limits start changing as soon as testing begins and continue to change throughout the entire lot. PAT test limits for any given part are based on the data from a specified number of previous parts in the lot. As testing of the lot proceeds, old data is discarded in favor of more recent data. The amount of data that contributes to the limits calculations is referred to as the Window Size.

PAT AT FINAL TEST

The main advantage of iPAT over DPAT is its ability to provide dynamic PAT testing without retesting the parts simply to apply the new PAT test limits. At final test, this re-running of parts would be time consuming and would require special handling that our manufacturing systems are both unwilling and unable to accommodate. Additionally, for many of our products, there are customer-imposed limits to the number of socket insertions allowed at final test.

WHAT IS DRIFT PAT

Drift PAT, sometimes abbreviated as “dPAT”, is a dynamic PAT implementation that uses drift files storing data from a previous pass to perform PAT calculations and binning in the current pass. In addition to being applicable only to multi-pass products, it is restricted to serialized parts.

HOW IPAT TEST LIMITS ARE CALCULATED

PAT test limits are calculated as follows:

$$PAT_{UpperLimit} = \mu + N \cdot \sigma$$

$$PAT_{LowerLimit} = \mu - N \cdot \sigma$$

where

μ = *mean or average of PAT dataset*

σ = *standard deviation of PAT dataset*

N = *user specifed number (typically 6 for automotive products)*

ROBUST STATISTICS

iPAT uses Robust statistics as opposed to Gaussian statistics. This is because Robust statistics tends to be more immune to the inclusion of outlier data into the dataset. Whereas Gaussian statistics are calculated by mathematically combining a set of data to arrive at the Mean (average) and Sigma (standard deviation) for that data, Robust statistics use a data sorting and quartile selection algorithm.

To calculate the Robust Mean of a dataset, the data is first sorted from smallest to largest values. Then the value in the middle of the dataset (the Median, or second quartile) is selected as the Mean.

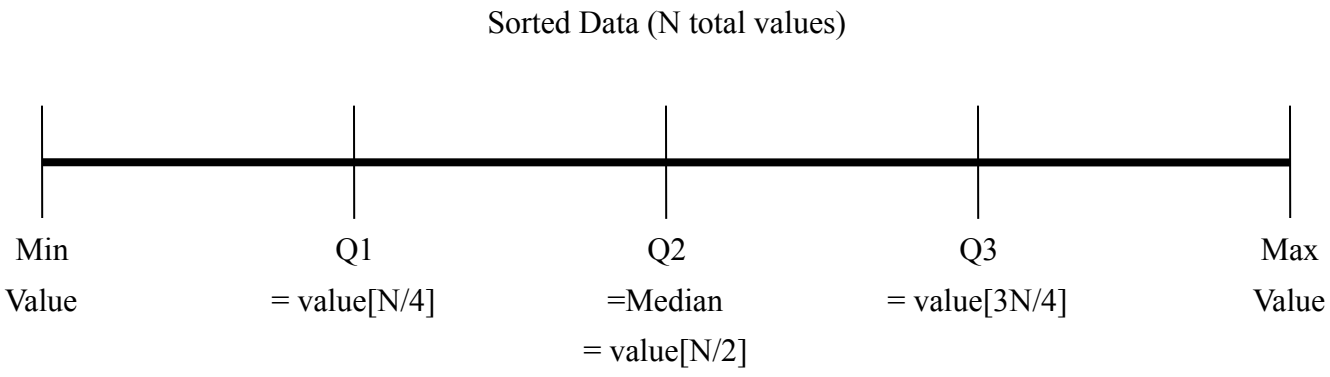
$$RobustMean = Q2$$

To calculate the Robust Sigma of a dataset, the data is sorted (as for the mean above) and the sigma value is calculated as the difference between the third and first quartiles divided by 1.35.

$$RobustSigma = \frac{Q3 - Q1}{1.35}$$

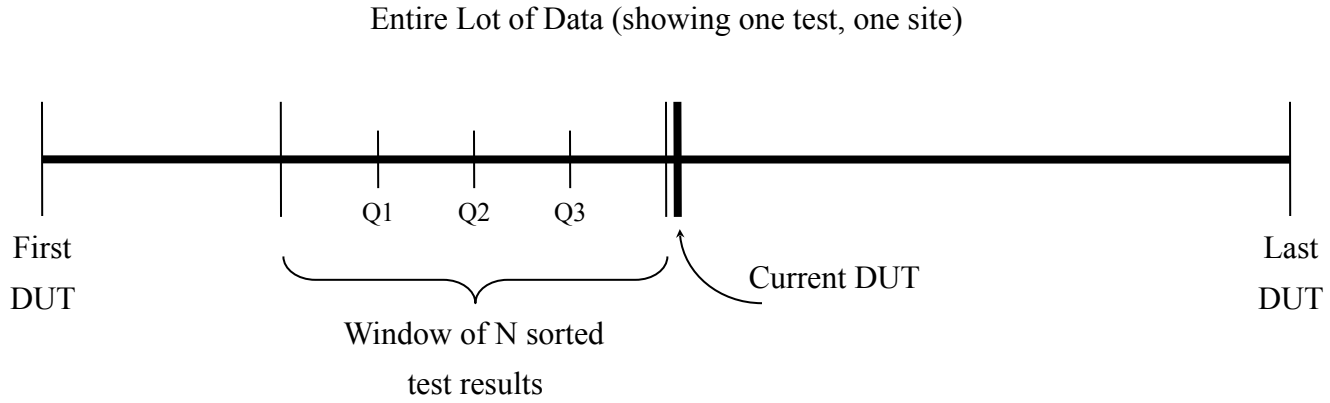
Note: If the data is perfectly Gaussian, the Robust Mean (Median) and Sigma values will be the same as the Gaussian Mean and Standard Deviation.

The following diagram shows how the quartile values are obtained:



ROLLING WINDOW

iPat uses a rolling window of data to calculate the Robust Mean and Sigma values. The window size is user-defined but is typically set to 100. That is to say the 100 most recent prior test results will be used to calculate the Mean and Sigma for the current device under test.



WHAT DATA GOES INTO THE WINDOW?

A test does not have to pass in order for its data to be added into the iPAT Rolling Window. It just has to be “good enough”. Good enough for iPAT is if the data is within 20% of the program limits.

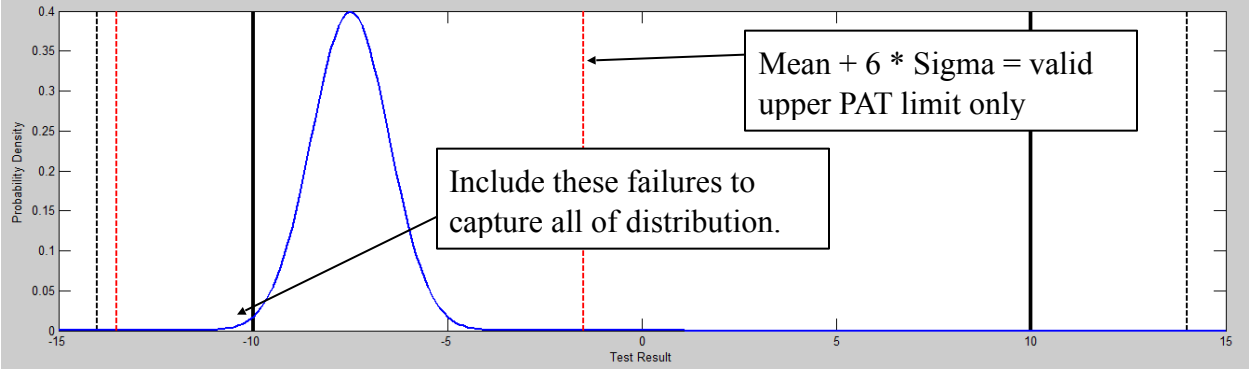
Example:

$$LowerProgramLimit = -10$$
$$UpperProgramLimit = +10$$
$$LimitWidth = UpperProgramLimit - LowerProgramLimit = 20$$
$$LimitExtension = 20\% * LimitWidth = 4$$
$$ExtendedLowerLimit = LowerProgramLimit - LimitExtension = -14$$
$$ExtendedUpperLimit = UpperProgramLimit + LimitExtension = +14$$

Therefore, in the above example, a test result within the extended limits of -14 to +14 will have its value added to the rolling iPAT window, even if it is a failing test result (outside the range of -10 to +10).

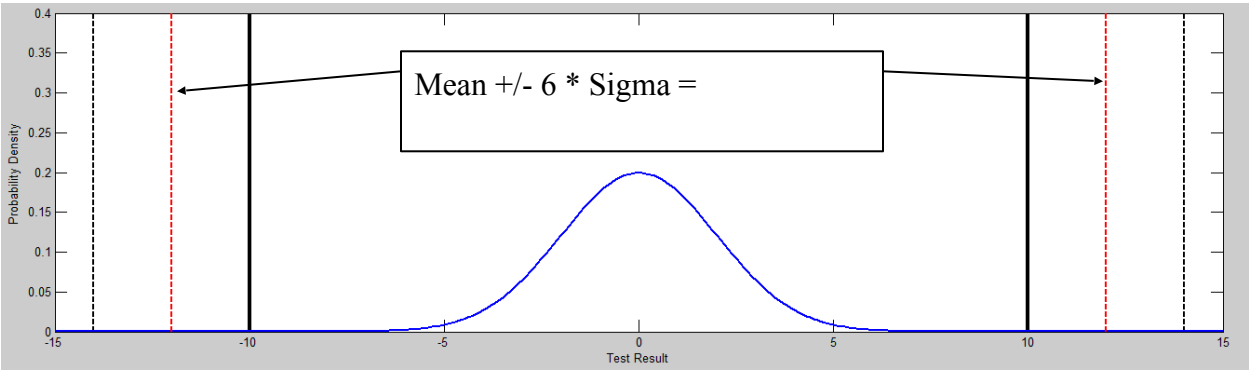
This technique is used mainly to cover the case where the test has a low Cpk. A low Cpk is an indicator that one side (or both) of the test’s Gaussian distribution is outside of the test program limits. When calculating iPAT limits, it is desireable to estimate the entire distribution of the test, not just that portion that is within the program limits. Note the following typical circumstances:

If the Cpk is low because of poor centering within the program limits, having a good estimate of the distribution will improve outlier detection on the opposite side of the limit range without getting too many false positives (falsely detecting an outlier).



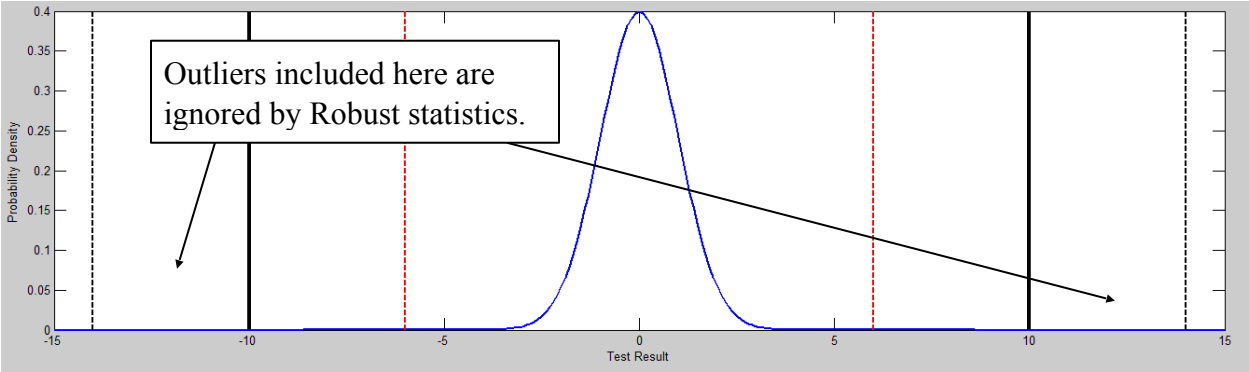
Mean = -7.5, Sigma = 1.0, Cpk = 0.833

1. If Cpk is low because the distribution is very wide within the program limits (i.e. the full distribution extends beyond both lower and upper program limits), the test will fail due to the program limits before it fails due to iPAT limits. This type of test is a poor candidate for outlier detection using iPAT in the first place, and iPAT should not be applied here. See “Selecting a Good iPAT Test Candidate” below.



Mean = 0.0, Sigma = 2.0, Cpk = 1.667

2. If the Cpk is high, including a failing test result in the window of data may very well be including an outlier. This is where using Robust statistics helps greatly. These included outliers will be sorted all the way out to the ends of the data window, and will be ignored during the quartile selection process. These “bad” data points will continue to be ignored by robust statistics until at least 25% of the data is represented by outliers.



Mean = 0.0, Sigma = 1.0, Cpk = 3.333

EARLY PAT LIMITS

When testing the early part of a lot (when the data window isn't full yet), the window size is necessarily shortened. The Quartile values are selected based on the smaller window size until the window size reaches N, at which point the oldest data points are discarded to make room for results from the most recent test results.

When the window size is very small, some special cases apply to the mathematics. If the window contains only one or two test results, the Sigma value cannot be calculated and so defaults to zero. This causes the PAT upper and lower limits to take on special behaviors based on a user-selected program parameter. See the section "Start-At-Zero vs. Start-At-Limits" below for further details.

If the window contains between 3 and 14 test results, the statistics are considered "suspect" and so the limits calculated based on those statistics are weighted.

If the window contains between 15 and N test results, the statistics are no longer considered "early" and their values are fully applied to calculating the PAT test limits as described above.

EARLY LIMITS ARE WEIGHTED

For early test results (between 3 and 14 values) weighting factors are calculated (per site) and only a portion of the standard PAT test limit calculations are applied to the actual PAT test limits. The weighting factor is:

$$WeightingFactor = \frac{NumPartsInWindow}{15}$$

How the weighting factor is used depends on whether the PAT limits are set to "start at zero" or "start at program limits".

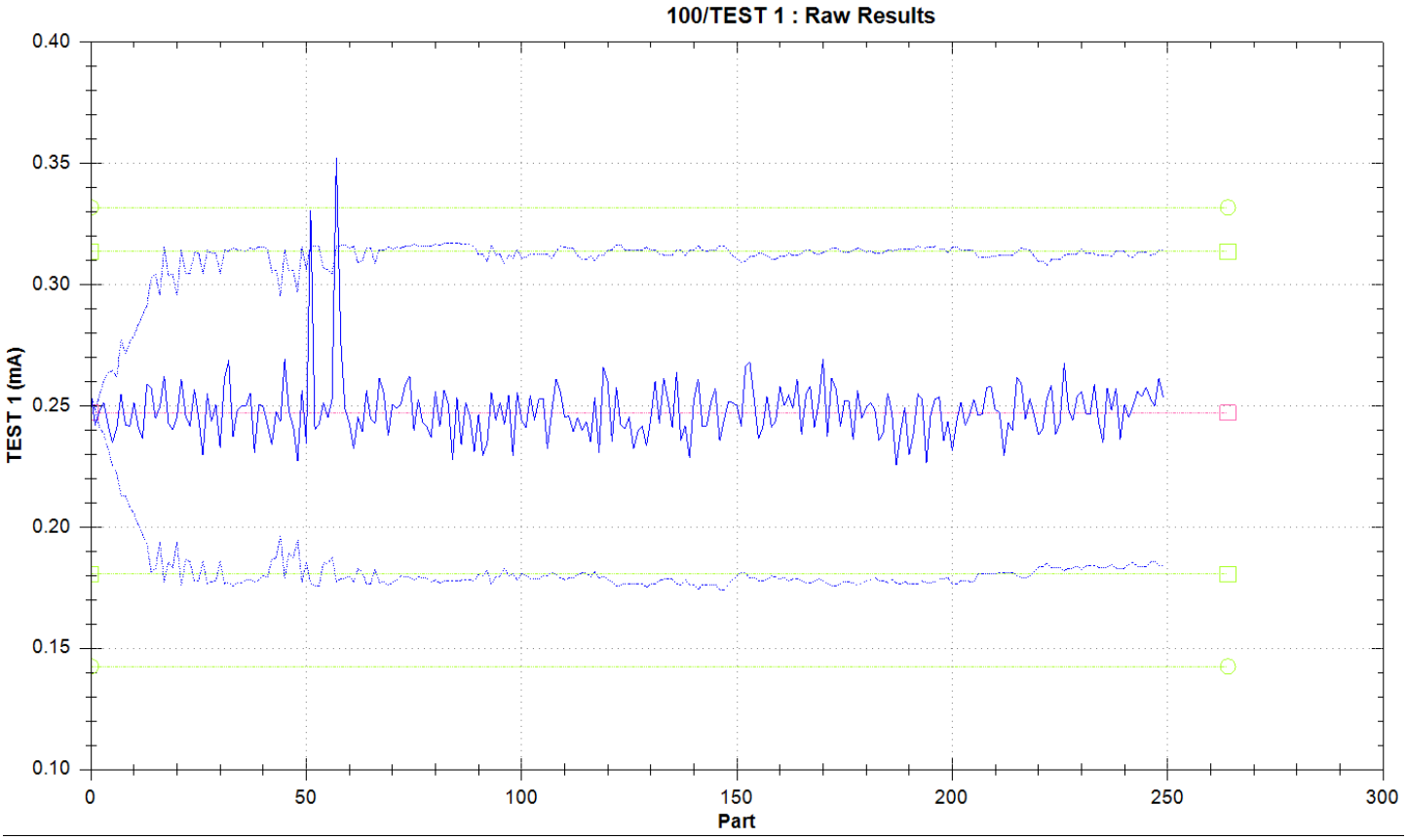
START-AT-ZERO VS. START-AT-LIMITS

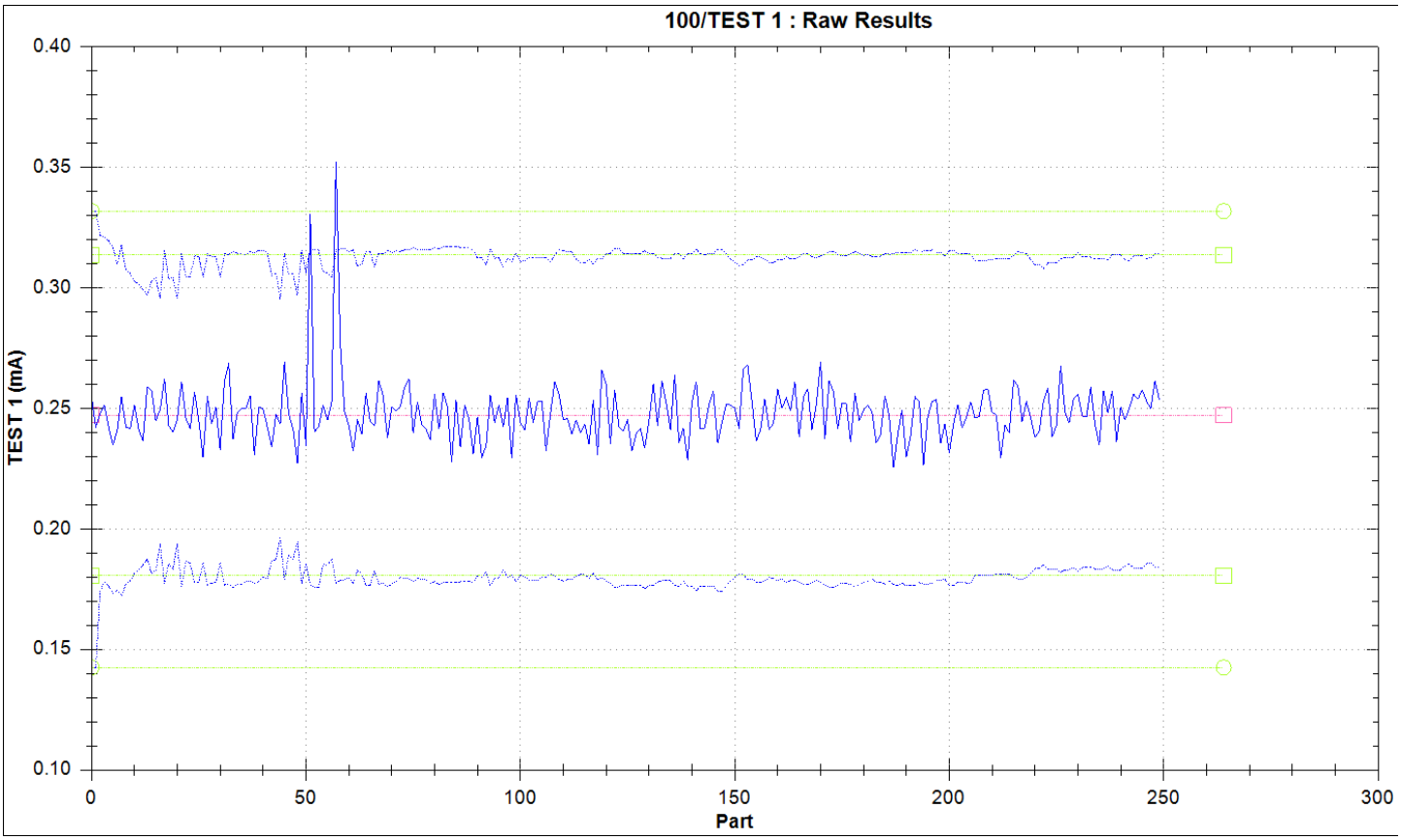
When the PAT limits are set to "start at zero", they will start out being identically the same, equal to the Mean. The first two parts (on each site) will always fail, then the limits will bloom outward toward their final (average) values until the

window contains 15 or more values, at which point the limits are calculated dynamically based on all the data in the window using the standard PAT test limits formula above.

When the PAT limits are set to “start at program limits”, they will start out being the same as (or near) the program limits (depending on which program limit is closer to the mean). Then they will shrink inward toward their final (average) values until the window contains 15 or more values, at which point the limits are calculated dynamically based on all the data in the window using the standard PAT test limits formula above.

The two graphs below show results for these two iPAT modes executed on identical data. The outer green lines are the program limits and the inner green lines are the average iPAT limits (equal to what the DPAT limits would have been). The red line is the mean for the entire lot (not to be confused with the iPAT mean which varies throughout the lot).





CONTINUATION LOTS

The “Continuation Lot” feature allows iPAT statistics to carry over from one lot to the next. This is to support splitting lots in production without incurring the yield hit caused by early PAT failures. Usually, the first two parts (on each site) in a lot are guaranteed to fail. If a continuation lot is detected, this will not happen.

A continuation lot is detected by entering the same base lot ID as the previous lot. For example, if the first lot ID is 123456.1 and the second lot ID is 123456.2, this second lot will be detected as a continuation lot and the statistics will not be reset for the new lot.

Note that this only applies for a single run of the test program. If the test program is ended and then restarted, there will be no continuation lot detected.

The continuation lot feature is not supported by all iPAT tester platforms. See the Test Program Implementation section below.

HOW IPAT TEST RESULTS ARE CALCULATED

For every test that the user sets up as a PAT test, the iPAT software will create (up to) four additional tests that are executed and logged after all other user-defined tests (assuming that the DUT is still passing up to that point).

Note: A failing DUT can still have its test results added to the iPAT statistics (see "What Data Goes Into the Window" above) but if it fails, the PAT tests will not be executed or logged.

If it is determined that any PAT test should be executed (i.e. the DUT is still passing) then ALL PAT tests will be executed regardless of the state of any STOP_ON_FAIL flag.

AUTOMATICALLY ADDED PAT TESTS

Each base PAT test has the following four tests added automatically:

(PATUL)

This is the PAT upper limit as described above. The units for the test are the same as the units for the base (non-PAT) test. There are no upper or lower limits for this test. It is used simply as a way to log the PAT upper limit.

(PATLL)

This is the PAT lower limit as described above. The units for the test are the same as the units for the base (non-PAT) test. There are no upper or lower limits for this test. It is used simply as a way to log the PAT lower limit.

(PAT)

This is the main PAT test result. It is reported in units of Sigma. The limits for this test are +/-Num_Sigmas as specified by the user for that PAT test (typically +/-6.0). This test result is calculated by subtracting the current Robust Mean from the base test result, then dividing by the current Robust Sigma.

$$PAT\_Result = \frac{BaseTestResult - RobustMean}{RobustSigma}$$

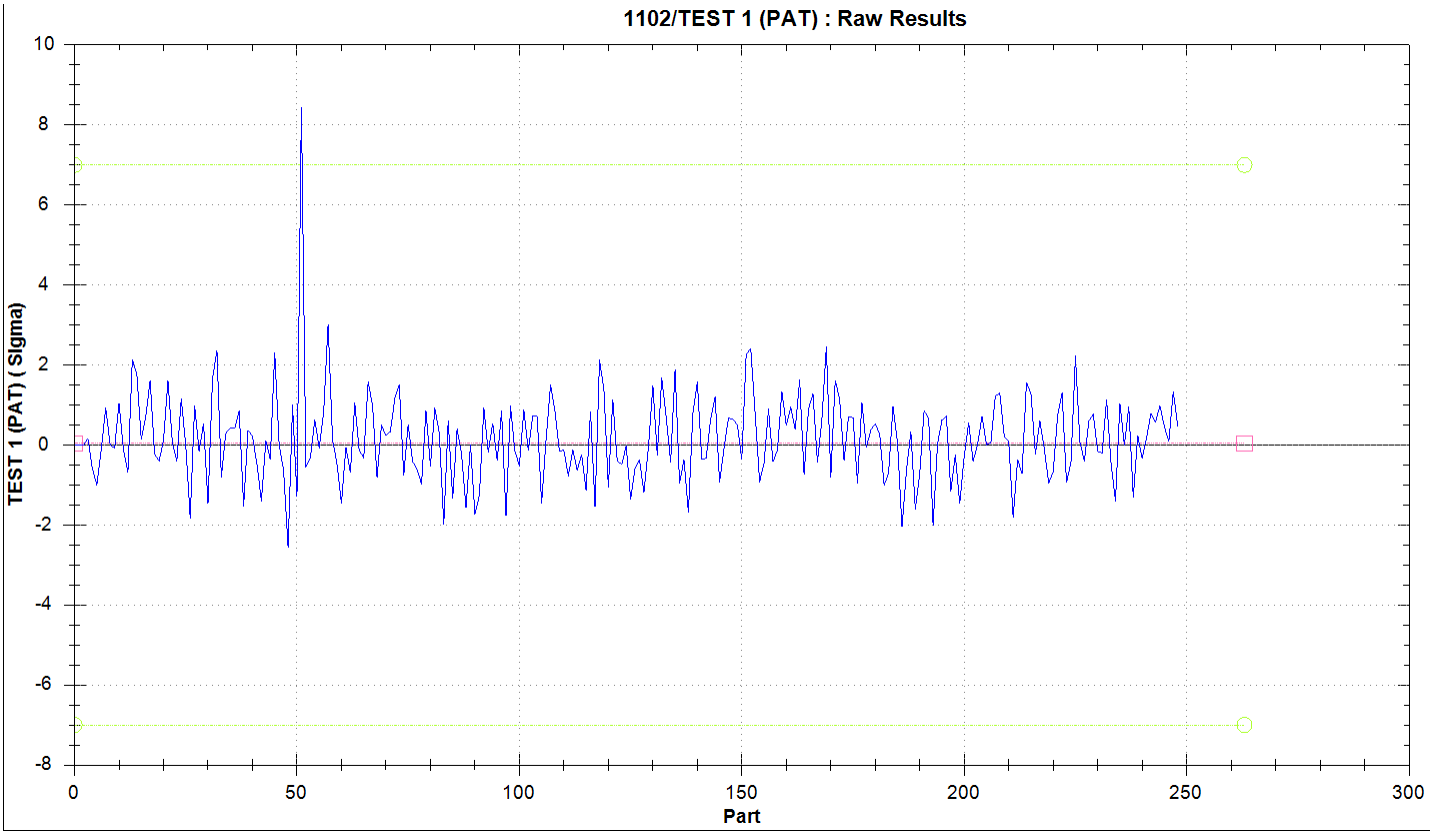
This test is therefore a measure of how many Sigmas the base test result is away from the Mean. If the test falls outside of the PAT test limits, the DUT is considered an outlier. When this test fails, the DUT will fail to the PAT fail bin (e.g. _IPAT_DQ_BIN on 40-series testers). This bin should be mapped to a unique hardware bin so that the parts can be isolated as iPAT failures for further evaluation. If a part fails this test it should NOT be considered for retest.

(PATEF)

This is the PAT Early Failure test. It will only be executed if the iPAT Rolling Window has less than 15 values in it. Otherwise, this test result is identical to the (PAT) test described above. The main purpose for this test is to isolate parts that have failed iPAT tests during the early part of the lot, where statistics are questionable due to a small number of data points. This test will fail to a different bin than the (PAT) test, the "early" PAT fail bin (e.g. _IPAT_EARLY_DQ_BIN on 40-series testers). If a part fails this test, it may be considered for retest since the PAT failure may not reliably indicate that the part is an outlier.

TYPICAL IPAT TEST RESULTS

The below graph shows the PAT test results for the same data as the PAT test limits graphs above. Note the similar “shape” of the data except that a scaling has taken place using the varying Mean and Sigma from the iPAT Rolling Window. Note also that only one large spike shows up in this graph where the base test result data has two. This is because the second spike in the graphs above has failed the base test limits and so that part did not have the PAT tests executed.



SELECTING A GOOD IPAT TEST CANDIDATE

The purpose for adding PAT tests to a test suite is to identify statistical outliers in a given lot of parts. Typically, this means identifying parts that are deviating from what the bulk properties of the wafer say the part should be doing. Therefore tests whos results have good correlations with bulk property changes usually make good PAT tests. Good PAT test candidates therefore include most current tests (supply current, leakage, etc.). Resistance measurements can also be good indicators of bulk properties. Either direct resistance measurements or indirect measurements (like gain or attenuation) can be used.

Regardless of the physical characteristics of the test, it should always have good mathematical characteristics as well. A statistical outlier should be very obvious to “see” when one is found.

Some test results may be tightly coupled or correlated with each other. Only one of these coupled tests (the one with the better distribution) should be selected in order to minimize the number of PAT tests added.

As a side note, two well correlated tests could become candidates for ratio testing. This is a method of testing whether tests that should be correlated deviate from that expectation. It may be as simple as dividing one test result by another and applying iPAT to that new “ratio” test.

GAUSSIAN DISTRIBUTION

Good iPAT test candidates should have a Gaussian distribution. Tests that are not Gaussian in nature (for example bi-modal distributions) tend to be less able to identify outliers with iPAT. One reason is that values of Robust Sigma tend to increase when the data is not Gaussian.

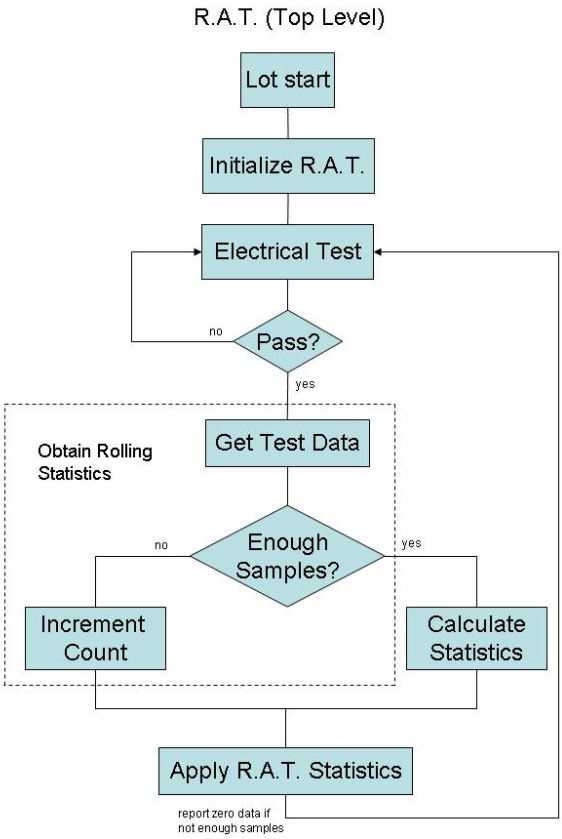
There are a number of methods for determining the distribution of a dataset. Several methods include calculating Skew and kurtosis as well as using a “Histogram Density” test and Regression Analysis. The Histogram Density test is simply a way to determine how many bins in the data’s histogram actually have values in them. If you histogram your data, and you have too many empty histogram bins, the distribution is usually non-Gaussian. The Regression test also uses the data’s histogram, but in this case the shape of the histogram is compared to the “ideal histogram” for that data set using least squares linear regression. The “ideal histogram” here is created using the simple mean and standard deviation of the original data.

GOOD CPK

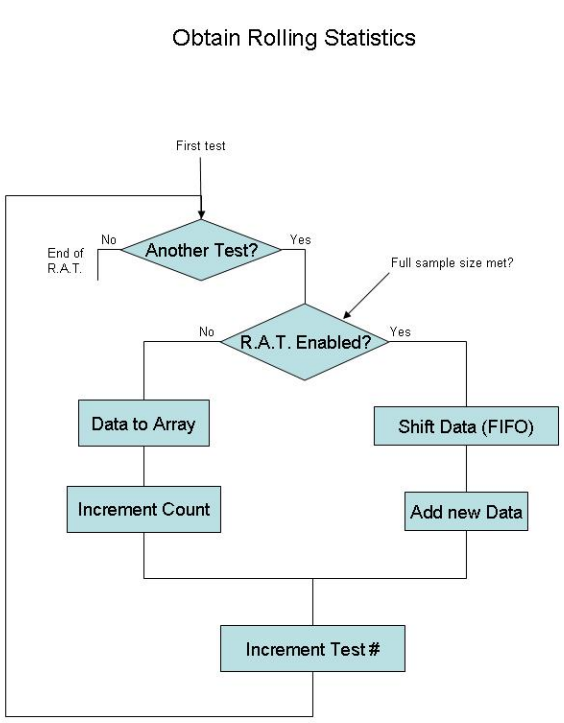
A good test candidate should have a high Cpk (should have a small Standard Deviation and be well centered within the program test limits). iPAT calculates the PAT limits by multiplying Sigma by a fairly large constant (6 or 7 or even larger) and adding and subtracting that from the Mean. If these PAT limits fall outside the program limits on a regular basis, the test is useless as a PAT test because the parts will fail the base test before it can be identified as an oulier. It is in fact NOT an outlier; it’s just a failed part. By our definition an outlier is a part that has passed regular testing but is still not the same as other parts in the lot. So iPAT limits should usually, if not always, fall within the program limits.

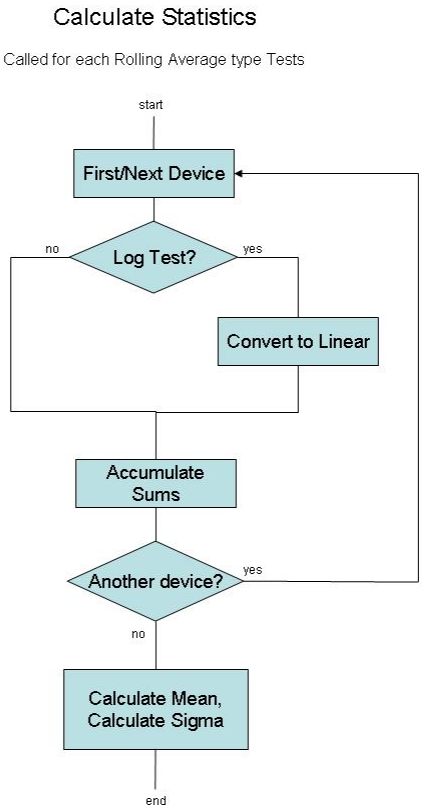
DYNAMIC OUTLIER REMOVAL

Test limits for production testing of integrated circuits require a margin for process variation to determine specifications that can be supported over time with acceptable yields. This is one scenario where test limits are often quite wide with respect to any single lot test distribution, hence making them unsuitable for detecting and removing outliers, devices that are significantly distant from the test mean. Outliers, which are statistical anomalies in a test distribution, have the potential to behave as unstable devices. Therefore it is highly undesirable to ship an outlier if avoidable. The Rolling Average Test (R.A.T.) algorithm was developed to dynamically characterize the manufacturing distribution, using this information to validate individual device performance in order to detect and remove outliers.

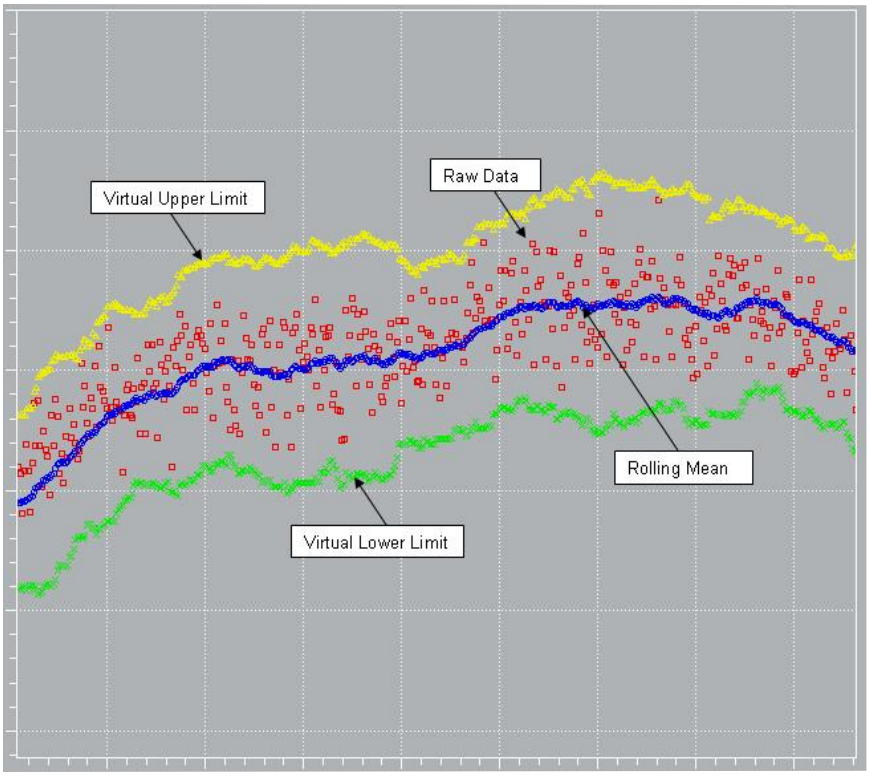


R.A.T. is a post-electrical testing algorithm. It accumulates test data from an in-line windowed user-definable sample, generating distribution statistics that are regionally precise. For each electrically passing device, test data is captured while the oldest sample point is discarded. This maintains a reasonably sized and local sample. Static and lot-based Part Averaged Testing (PAT) sample methods by definition capture lot and wafer based variations while the R.A.T. sample effectively eliminates these effects by its tracking nature and limited sample size. The rolling average approach allows a much more strict method to detect and remove outliers by removing process variations.





The primary test result of R.A.T. is a measure of the distance of a device test result from the mean in number of standard deviations. It allows test-specific control of the degree that outliers are removed. It adds no further electrical testing and is transparent to traditional test operations beyond the occasional outlier device that should be rejected. Therefore, the benefits of R.A.T. outweigh the marginal overhead in test time. Along with the critical result above, R.A.T. also reports test statistics (mean and standard deviation) and the virtual test limits with which it is effectively testing. This occurs on a per-part basis due to the rolling average nature of the sampling. R.A.T. is designed to work with linear and logarithmic types of test results.



The above plot illustrates the R.A.T. algorithm performed on a dataset.

AEC RECOMMENDATIONS

AEC-Q001 Guidelines for Part Average Testing, published by the Automotive Electronics Council gives a number of recommendations for selecting PAT tests. These recommendations are briefly summarized at the top of this section. You can download all of the AEC documents from this web site: http://www.aecouncil.com/

IMPLEMENTATION