Optimization of Test/Diagnosis/Rework Location(s) and Characteristics in Electronic System Assembly

Zhen Shi and Peter Sandborn
CALCE Electronic Products and Systems Center
Department of Mechanical Engineering
University of Maryland, College Park MD 20742
shizhen@calce.umd.edu
sandborn@calce.umd.edu

Abstract. In this paper, an optimization methodology is used to select the locations and characteristics of test, diagnosis and rework operations in electronic systems assembly processes. Real-coded genetic algorithms are used to perform a multi-variable optimization that minimizes the yielded cost of products resulting from an assembly process that includes test/diagnosis/rework operations characterized by costs, yields fault coverage, and rework attempts. A general complex process flow is analyzed using the algorithms proposed in this paper, and a multichip module assembly process flow is used to demonstrate that the methodology can identify optimum test and rework solutions that result in a reduction in yielded cost.

Keywords: test economics, optimization, yielded cost, electronic assembly, test strategies

1. Introduction

The cost of testing during electronic system manufacturing and assembly can be a significant contributor to the final system cost. Performing tradeoffs associated with where in a process to test and what level of test, diagnosis and rework to perform are key to optimizing the cost and yield of an electronic systems' assembly.

Previous research efforts have treated the economics of test for electronic systems. Test economics modeling and methodology development ranges from classical relations between fault coverage and yield [1], to the development of economic models for analyzing the financial aspects of test investments [2] and the tradeoffs associated with design for test [3]. Many modeling efforts have focused on the development of test step models for inclusion in process flow based cost analyses (a process flow is a set of process steps in an application-specific sequence), e.g., [4]-[8]. Ambler *et al.* [4] have proposed a test planning system that compares different test strategies for a given design based on circuit properties, intended market and company resources. Tegethoff *et al*. [5] developed a simulator for the manufactuing, test and rework of boards that can be used to select test characteristics and locations (but does not provide a structure to perform their optimization). Abdir *et al.* [6] analyze the economics of various test strategies for multichip design at an early stage of the design cycle. Driels *et al.* [8] analyzed the rework strategies used in assembly manufacturing system. The Trichy *et al.* [7] model is discussed in Section 3. The models in [4]-[8] vary in complexity, but in general they use incoming costs and yields (from upstream process steps) and determine the cost and yield of the product after testing to a specified fault coverage has been performed. In most cases, some type of diagnosis and rework can also be modeled.

Previous efforts using Test/Diagnosis/Rework (TDR) models have been confined to evaluating the impacts on product manufacturing when the locations of the test operations (relative to the manufacturing process steps) are predetermined either by the manufacturing process or based on experience. Optimization of the characteristics of testing (i.e., fault coverage versus test cost) concurrent with a search to determine the optimum location(s) for TDR operations in an application-specific process is not known to have been addressed in previous work. The paper discusses an optimization methodology for addressing the following questions on an application-specific basis:

1. How much testing should be done, i.e., how thorough should the test be? How much fault coverage should be required?
2. Where (in a process) should test operations be used?
3. After which test operations should rework be performed?
4. How many rework attempts should be made before scrapping the assembly?

The analysis described in this paper attempts to make the best choice between various possible versions of tests and reworks, and to suggest the values of the characteristics of testing that minimize an objective function associated with the cost and yield of products that result from the process flow.

2. Solution Strategy

The assumed starting point for the optimization problem considered in this paper is a manufacturing process for a product (described as a process flow comprised of a sequence of process steps). The goal of the algorithm is to determine where the TDR operations should be placed within the process and the characteristics of the specific test, diagnosis and rework operations.

The paper is organized as follows: Section 3 reviews the cost model used for the computations of a single TDR operation in the process flow. Section 4 describes a multi-variable objective function in which characteristics of all possible TDR operations are included. In Section 5, real-coded genetic algorithms are used to implement the optimization of the TDR operations and the parameters associated with them in the process flow. A test case is also provided in Section 5 to demonstrate the applicability of the model. Section 6 applies the optimization methodology to a multichip module (MCM) assembly process and the results from the optimization methodology developed in this paper are compared to the performance of published test characteristics for an MCM assembly process.

The framework (Fig. 1) of the methodology developed in this paper includes:

1) Insert a TDR operation (a combination of test/diagnosis/rework operations, see Section 3) at every possible location in the process flow. The possible TDR operations are located between

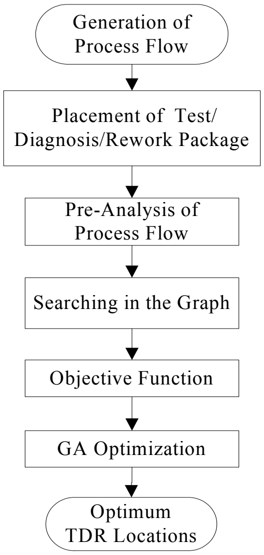

Generation of Process Flow
Placement of Test/ Diagnosis/Rework Package
Pre-Analysis of Process Flow
Searching in the Graph
Objective Function
GA Optimization
Optimum TDR Locations

Fig. 1. The framework for optimization of TDR operation location(s) and characteristics in a process flow. The elements shown in this figure are discussed throughout Sections 2-4. GA – Genetic Algorithms.

each pair of adjacent process steps, before initial steps and after the final step (for an example see Fig. 5 in Section 5). The number of test locations that need to be optimized depends on how many separate process steps are included in the process flow.

2) Vary the fault coverage of all test steps continuously from 0% to 100%. The recurring test costs are computed as a function of fault coverage using a variable test cost model (Section 3).

   a) A fault coverage threshold is embedded in the optimization algorithm. If the value of the fault coverage associated with a specific test operation falls below a predefined threshold — the minimum nonzero fault coverage that we can practically purchase, the test operation will be removed from the process flow, i.e., there is no test present in the location. The key to the methodology is that instead of separately finding test locations and fault coverage, the optimization algorithm merges the two into a single problem by defining the fault coverage threshold.

   b) Inclusion of rework is considered concurrent with fault coverage. The rework operation is associated with the test operation when the fault coverage of test is above the threshold, i.e., the rework cannot be done without a test present at the location. However, you can have a test without a rework.

3) Use overall "yielded cost" as the objective function to be minimized. The overall yielded cost is the accumulated cost divided by the final yield of the whole process. The characteristics of all possible TDR operations included in the process flow are treated as the variables that need to be optimized.

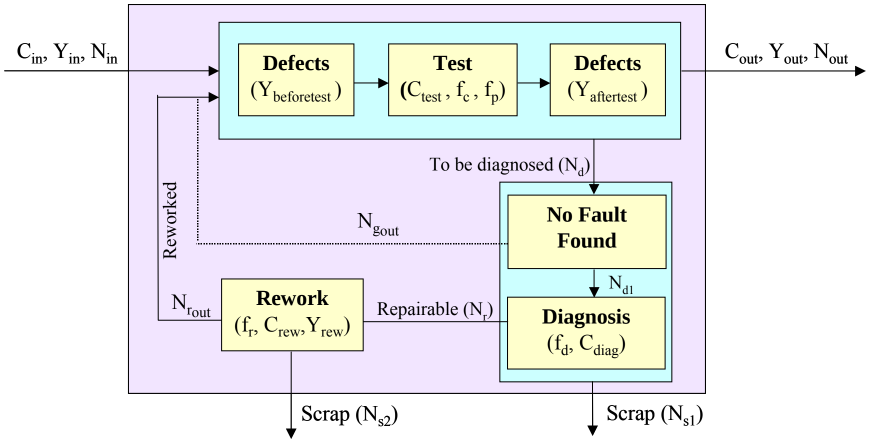
4) A Waiting Sequential Search (WSS) algorithm is used to determine the order for evaluating process steps and controlling the objective function calculation, [9]. This graph-based search algorithm traverses all the process steps in a direct graph representation of the process flow and performs computations of the yield and cost of products. The search process performs the sequential calculations from the start to the end of the process flow, and the single TDR model (Section 3) is used when the search encounters a TDR operation. WSS essentially means that one process step can be searched only after all the inputs to it have been visited.

5) Real-coded Genetic Algorithms (RCGAs) are used to manage the optimization.

The solution strategy developed in this section is not specific to any particular TDR model formulation, however, we have used the model from Trichy *et al.* [7] (see Section 3). The solution strategy outlined in the section also has several limitations: 1) it assumes the cost of actually placing a TDR is the same independent of the location (this is not necessarily true); 2) the strategy assumes that if rework is performed, it is performed immediately after a test; 3) the methodology does not directly exploit the fact that changing the manufacturing process steps (and possibly the product topology) could reduce test cost further (i.e., the methodology optimizes a given process flow only); and 4) the test operation modeled only accounts for a single fault type as opposed to a portfolio of fault types.

3. The Test/Diagnosis/Rework Model

To involve all the effects of the characteristic parameters of a TDR operation (e.g., fault coverage, test cost, rework attempts, rework cost, etc.) in a general process flow and how they relate to the resulting cost and yield, a comprehensive model was developed by Trichy *et al.* [7]. This model provides a detailed formulation of the characteristic parameters for a single TDR operation. Fig. 2 shows the content of the model and detailed formulations are given in [7].[1]



C_{in}: Incoming cost
C_{diag}: Cost of diagnosis operation
C_{out}: Outgoing cost
C_{rew}: Cost of rework operation
C_{test}: Cost of test operation
f_c: fault coverage
f_d: Fraction of assemblies determined to be reworkable
f_p: False positives fraction
f_r: Fraction of assemblies actually reworked
N_d: Number of assemblies to be diagnosed
N_{gout}: Number of no fault found
N_{in}: Number of assemblies entering TDR operation
N_{d1}: N_d-N_{gout}
N_{out}: Number of assemblies exiting TDR operation
N_r: Number of assemblies to be reworked
N_{rout}: Number of assemblies actually reworked
N_{s1}: Number of assemblies scrapped by diagnosis operation
N_{s2}: Number of assemblies scrapped during rework
$Y_{aftertest}$: Yield of processes that occur exiting the test
$Y_{beforetest}$: Yield of processes that occur entering the test
Y_{in}: Incoming yield
Y_{out}: Outgoing yield
Y_{rew}: Yield of rework operaton

Fig. 2. The model of a TDR operation used in this work, [7].

[1] Note, several typographical errors should be corrected in [7]: In (2) and (3), the maximum of the summation should be n-1 instead of n, and (4a) can be used for either definition of f_p with $N_{d_{n+1}}$ changed to N_{d1_n}. In (13), the subscript of N_r should be i-1 instead of i when i>0.

This model provides the computation of the parameters: C_{out}, Y_{out}, and N_{out} for a single TDR operation in a process flow in which the variables shown in Fig. 2 are defined. For real processes, the variables defining the test, rework and diagnosis processes are not constants. In particular C_{test} is not a constant and instead is related to the fault coverage of the test. The rework yield (rework success rate) is also not a constant, it depends on the specific rework actions taken. For use in this work, the Trichy *et al.* model has been extended by defining general forms of the relationships among the parameters, e.g., the costs of test and rework in terms of fault coverage and rework yield respectively.

A relationship between the cost of test (proportional to test time or number of tests performed) and fault coverage has been suggested by Goel [10]. Empirical data shows that the test process can be divided in two phases. T_I (number of tests in Phase I, see Fig. 3) provides a fault coverage ranging from 65% to 85% for most combinational logic circuits [10]. For Phase II of the test generation, the number of additional tests required is approximately a linear function of the number of untested faults remaining at the end of Phase I. Unlike Phase I, in Phase II each generated test tends to detect fewer faults than the one before it and the average cost per detected fault increases.

An exponential function can be used to approximately simulate Phase I and a linear function in Phase II on the assumption that Phase II cannot reach a full coverage (100% fault coverage) in practical testing.[2] Assuming that the test cost is proportional to the number of tests, a relationship between test cost and fault coverage can be derived,

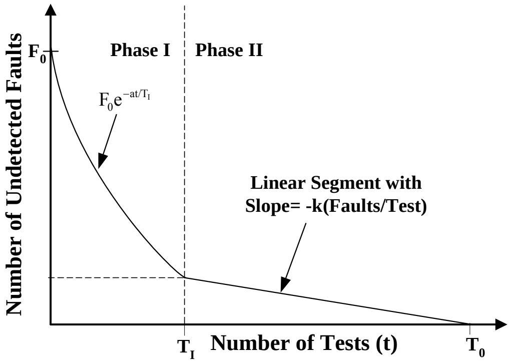


Fig. 3. Typical curve of untested faults versus the number of tests, [10].

[2] In the real electronic testing and manufacturing, the 100% fault converge cannot be achieved without significant increase on the testing cost. The assumption is made in this paper that the full fault coverage can't be obtained.

$$C_{test} = p_t[b_t \ln(1 - f_c) - r_t] + C_{ft}, f_c \in (0,1) \quad (1)$$

where p_t is the cost coefficient; b_t is the coefficient of test characteristic, r_t is the fault ratio, f_c is the fault coverage of test, C_{ft} is the fixed cost of test, which accounts for fixed costs associated with testing, i.e., there is a minimum fixed cost for having even a small fault coverage. Fig. 4 shows a plot of (1) using the values in Table 1.

Table 1. Example values of factors in (1) and (2).

p_t	b_t	r_t	C_{ft}	p_r	b_r	r_r	C_{fr}
0.02	-288	8.2	1	0.02	-300	10	1

There is similar relationship between rework yield (y_r) and rework cost (C_{rew}) for the first rework attempt,

$$C_{rew} = p_r[b_r \ln(1 - y_r) - r_r] + C_{fr}, y_r \in (0,1) \quad (2)$$

An alternative model for calculating the rework cost has been proposed in [8], which relates C_{rew} with the cost of rework equipment and rework time.

In (1) a small cost of test will be predicted for a small fault coverage. Actually, depending on the type of the test operation, there is a minimum test cost that is independent of fault coverage at very small fault coverage, i.e., the capital expense of the test equipment and the fixturing (tooling) cost. A threshold is defined as the minimum nonzero fault coverage that we can practically purchase. When the desired fault

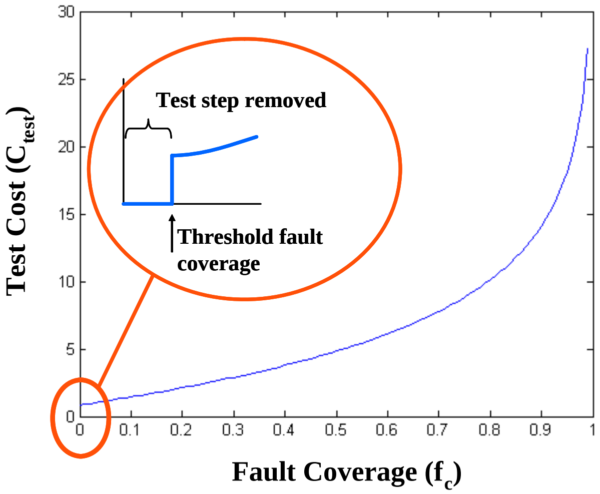


Fig. 4. Example relationship between the test cost and fault coverage showing the definition of the threshold fault coverage.

coverage is below the threshold value, the test would be removed to avoid the fixed testing cost, i.e., the fault coverage close to zero doesn't mean the testing cost is near to zero. An example functional relationship between test cost and fault coverage showing the threshold fault coverage appears in Fig. 4, in this case the test step will be removed, i.e., no test operation is placed at the location, if the fault coverage is below the threshold.

Functional relationships between fault coverage and test cost obviously depend on the type of system being considered. The relationships in (1) and (2) were used in this paper as examples only. The methodology that is the subject of this paper, will work successfully with alternative models.

After the single TDR operation is resolved, the graph-based search algorithm described in [9] is used to cumulatively evaluate the objective function of characteristics of a general process flow that includes multiple TDR operations with the computation performed according to the sequence of process steps.

4. Multi-Variable Optimization Function

The objective of optimizing TDR location(s) and characteristics is to minimize the "yielded cost" of the entire process flow. The yielded cost we are interested in is the final cost per product instance (after the final processing step and/or TDR operation) divided by the final product yield (see [11] for a discussion of "yielded cost"). This yielded cost gives a measure of the effective cost per good product instance after all the manufacturing and TDR operations are completed. The final cost per product instance and yield are determined by accumulating (sum or product) the individual process step costs and yields and the TDR operation costs and yields in the appropriate sequence through the process.

The objective function in which characteristics of all possible TDR operations are considered can be derived from a sequential cumulative computation from the first step in all the process flow branches to the single final step in the process flow. When the search algorithm traverses the entire process flow, a cumulative function is computed for use as the objective function that will be minimized in the optimization. The optimization problem becomes,

$$\min_{x \in X} C_Y(x_1, x_2, \cdots x_m) \tag{3}$$

Equation (3) also can be written as,

$$\min_{x \in X} \frac{C_{out}(x_1, x_2, \cdots x_m)}{Y_{out}(x_1, x_2, \cdots x_m)} \tag{4}$$

where,

m = number of characteristics to be optimized;

C_Y = yielded cost of the process flow, cumulative cost (C_{out}) divided by final yield (Y_{out}).

Equation (3) is one possible objective function. The form of the objective function could be changed to satisfy the specific manufacturing process requirements, e.g., some locations in the process don't allow insertion of a test. Because of the objective-independent property of Genetic Algorithms, the objective function can be modified without changing the optimization algorithm.

5. Optimization with Real-Coded Genetic Algorithms (RCGAs)

To optimize characteristics of possible TDR operations in order to find the global minimum of yielded cost of the process flow, genetic algorithms (GAs) are used. GAs are stochastic, directed search algorithms that have proven to be useful in finding global optima in both static and changing environments [12]. To handle the complex process flows in electronic systems assembly with hundreds of process steps, RCGAs, using a floating-point representation of parameters [13, 14], are applied to the optimization methodology according to the framework in Fig. 1. Traditional binary-coded GA evenly discretizes a real variable space and results in discrepancy between the binary representation space and the actual problem space in the continuous domain. The real-coded GAs are robust, accurate, and efficient because the floating point representation is conceptually closest to the real variable space, and moreover, the string length reduces to the number of variables. The system generates suggestions of where the TDR operations should be inserted and not inserted, and what the optimal values of characteristics of testing and rework should be based on the minimization of the objective function.

In order to verify the optimization methodology (verify that the methodology is finding global optimums), several simple test cases were formulated that can be validated with closed-form computations, for examples of these validation tests see [9]. The following example demonstrates the operation of the methodology on a complex multi-branched process flow.

5.1. Optimization in a Complex Process Flow

For use as a general demonstration, the process flow shown in Fig. 5 has been used.

The algorithm begins by automatically placing TDR operations in all possible locations: numbered 14-25 in Fig. 5. The algorithm then determines the fault coverage and rework requirements of each TDR that minimize the final yielded cost. Fig. 6 shows that results from the optimization for different assumptions about the fixed costs associated with the test and rework (values of the remaining parameters are given in Table 1). Fig. 6(a) has low (inexpensive) test and rework - C_{ft} and C_{fr} are both small; as a result, 9 of the 12 possible locations for tests are present (i.e., TDRs in these locations have fault coverages above the threshold for testing, which is a fault coverage of 0.1 in this demonstration case). Because rework is also inexpensive in this case, rework is being done at all the actual test locations. The Fig. 6(a) result is intuitive, if test and rework are inexpensive, then test and rework will be done after nearly every process step. Fig. 6(b) has inexpensive testing (same as the case in Fig. 6(a)), but the rework is expensive. As a result, significantly fewer rework opportunities are actually exercised. Notice also that the optimum test locations

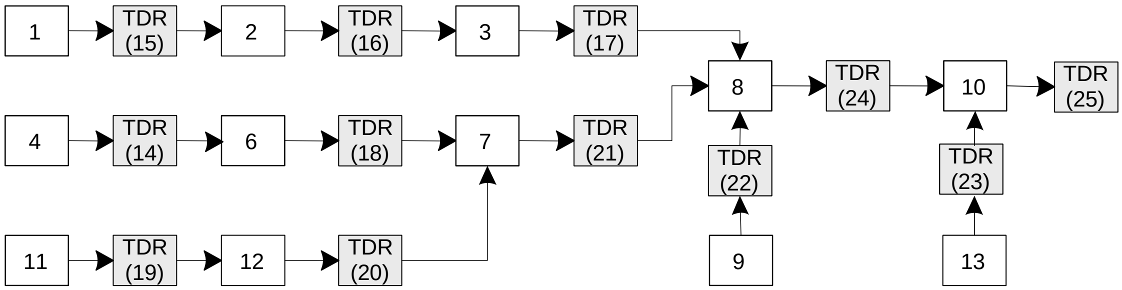
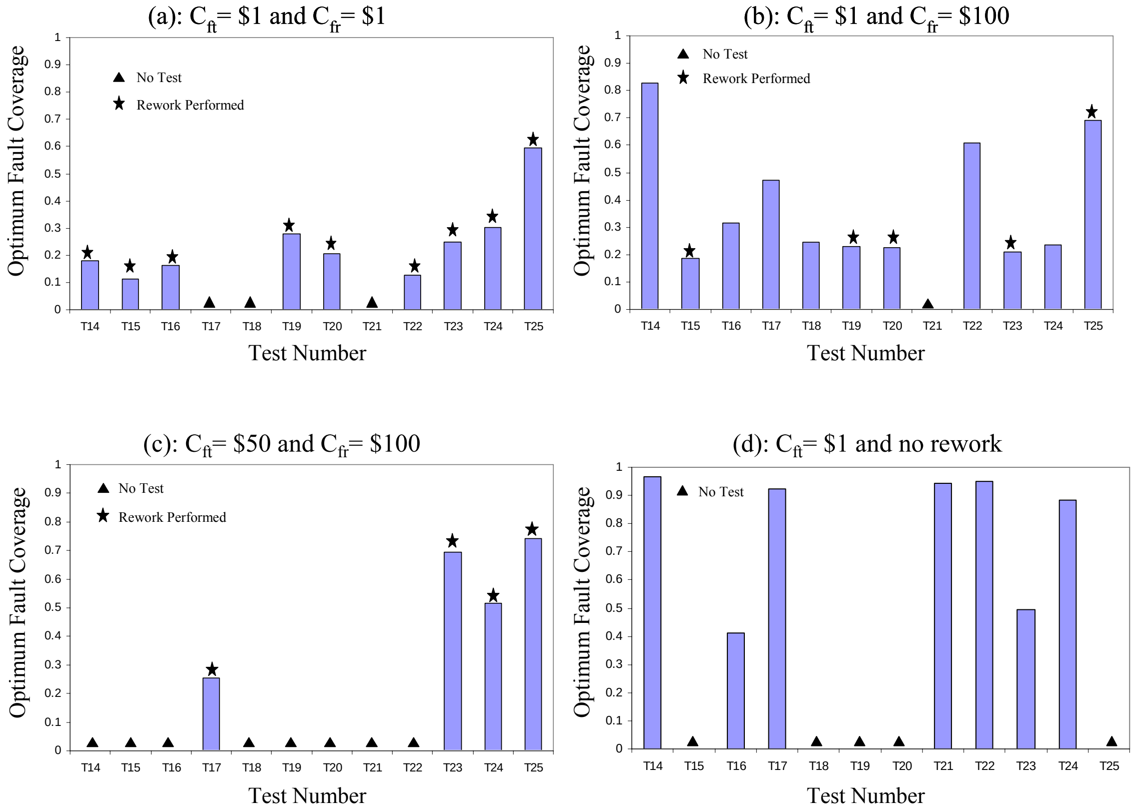


Fig.5. A general process flow with all possible TDR operations. Step characteristics are given in Table 2.

Table 2. Characteristics of and inputs to the steps in process flow in Fig. 5.

step	$cost_{in}$ ($)	$yield_{in}$	cost ($)	yield
1	41	0.91	21.1	0.95
2	62.1	0.92	12.3	0.88
3			14	0.89
4	60	0.36	23.8	0.94
6			45	0.96
7			11.3	0.86
8			33.8	0.92
9	37	0.42	13	0.90
10			15	0.92
11	9	0.95	60	0.95
12			78	0.91
13	75	0.96	54	0.94



(a): C_{ft}= \$1 and C_{fr}= \$1

(b): C_{ft}= \$1 and C_{fr}= \$100

(c): C_{ft}= \$50 and C_{fr}= \$100

(d): C_{ft}= \$1 and no rework

Fig. 6. Computed optimum parameter values for TDR operations in the process flow shown in Fig. 5. Various fixed test and rework costs used in (1) and (2) are assumed.

(and fault coverages) change due to the inclusion or exclusion of rework possibilities, even though the C_{ft} has not changed. Fig. 6(d) shows the same test costs as Fig. 6(a) and 6(b), but no rework is allowed – the optimum test locations and fault coverages again differ from cases 6(a) and 6(b), and the average value of fault coverage increases to compensate for the loss of rework capabilities (the optimization algorithm is attempting to maximize the final yield in order to minimize cost divided by yield). Fig. 6(c) has expensive test and expensive rework. As a result, only 4 of 12 possible test locations are used (Test 17, 23, 24 and 25 only), however, rework is included with all four of these tests. In this case, the majority of the testing is focused on test operations near the end of the process indicating that if test is expensive and defects introduced in the process are spread over the entire process (as opposed to being focused at a single process step), the optimum test location is near the end of the process, which is intuitive.

In the analysis demonstrated in this section, we have assumed that all the locations between adjacent process steps are practical locations to consider performing test and rework. In reality, if particular locations within the process flow are not practical for consideration for testing or rework they can be omitted from consideration in the optimization process.

The next section will apply the optimization algorithms to a multichip module assembly process.

6. Optimization ExerciseOptimization of TDR in a Multichip Module (MCM) Assembly

Testing an MCM is a particularly challenging problem because bare die generally have lower yields than packaged die and tradeoffs between wafer-level and die-level testing, and reworking the module after assembly are very relevant.

Abadir [15] has performed work on test strategies for MCMs. They analyzed the test, diagnosis and rework process for an MCM using a simple MCM manufacturing and test model, and calculated the cost of an MCM after TDR assuming various test fault coverages, which are applied to die of different incoming yield. Abadir compares the results from the various test cases (different fault coverage and test cost), but does not determine the fault coverage of the test process necessary to minimize the yielded cost of the MCM. The following example explores the optimization of a TDR process for the MCM described in Abdir's paper and show that the optimized test coverage leads to a lower yielded cost than the calculation from the conventional scenarios of specified test coverages considered in Abdir's paper.

6.1. The TDR of an MCM

A generic assembly, test and rework simulation model (Fig. 7) was used to analyze the relationship among all the parameters of the test process and how they impact the final cost and quality of the MCM.

In this section we use the example described in Abdir's paper as the tested MCM and compare it with the optimization results generated using the methodology described in this paper. The problem to be treated is an MCM containing 50 identical die. These die can be procured (purchased or made internally) at different cost and quality levels. For example, a die that costs \$2 has an 80% yield (i.e., 20% of those die are defective), while a version of the same die can be bought for \$9 with 99.9% yield. Note that this is the yield of die before assembly onto the MCM. This is not an unrealistic situation. Bare die may be purchasable as wafers from the semiconductor manufacturer, or in a tested (i.e., Known Good Die) form

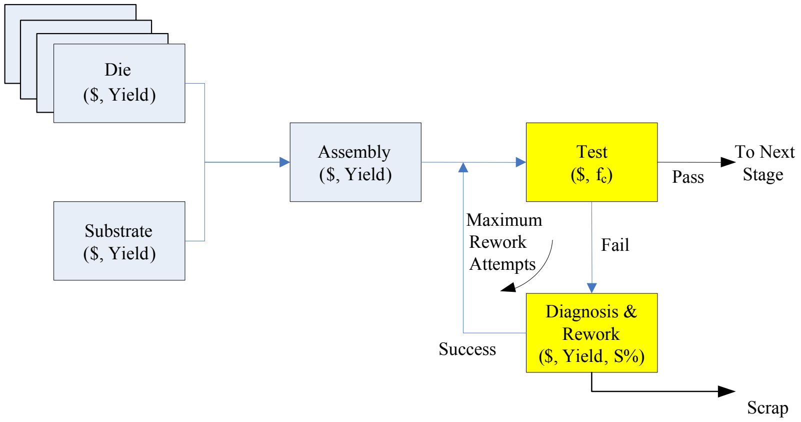


Fig. 7. A MCM manufacturing and test model [15].

from an aftermarket supplier who has performed bare die test (and burn-in) to pre-sort the good (non-defective) die from the bad (defective) die.

The test case scenarios considered in Abdir's paper are representative of the conventional approach to determining test locations within a system in which a tradeoff analysis of test strategies is based on enumerating the different cases. The following three different test strategies were applied by Abdir:

- Case 1: Fault coverage on MCM testing =95%, Test Cost = $100;
- Case 2: Fault coverage on MCM testing =99%, Test Cost = $150;
- Case 3: Fault coverage on MCM testing 99.9%, Test Cost = $200.

6.2. Optimization of Fault Coverage in the MCM Assembly Process Flow

This section will apply the optimization method proposed in this paper to the MCM manufacturing and test process flow discussed by Abdir. The optimum fault coverages to minimize the yielded cost of the MCM for specific input die yield are generated by RCGAs without enumerating all the combinations of possible test strategies. In order to enable the optimization methodology, a functional relationship between test cost and fault coverage used in the optimization algorithms needs to be determined.

Using a nonlinear regression analysis technique, (5) gives the relationship between test cost and fault coverage that was assumed in Abdir and is used in the RCGAs optimization of the MCM assembly process,[3]

[3] Equation (5) fits the relationship between fault coverage and test cost using the test cost versus die yield data reported in [15] with a function of the form given by (1) with a 10% fault coverage threshold imposed.

$$C_{test} = \begin{cases} -21.68\ln(1-f_c)+25 & (0.1 < f_c < 1) \\ 0 & (0 \le f_c \le 0.1) \end{cases} \quad (5)$$

Fig. 8 shows how the cost of the MCM after the TDR operation varies with the yield of the MCM before testing. Yield of the MCM (before TDR) refers to the yield of the MCM after all die have been assembled onto the substrate before any test or rework is performed, i.e., the incoming yield to the test operation. The yield of the MCM before testing is calculated by taking the input die yield to the power of number of die (50 in this example). It is obvious that higher fault coverage tests always lead to higher MCM cost.

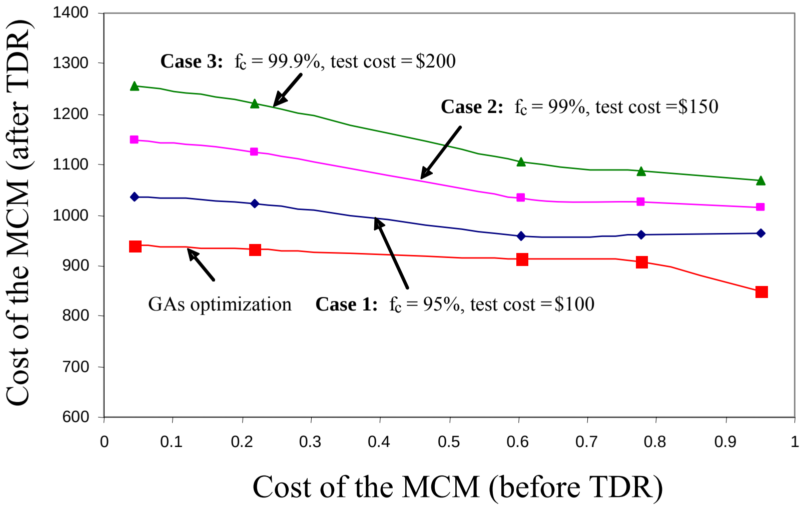


Fig. 8. Cost of the MCM after TDR versus incoming yield of the MCM for the three test scenarios in [15] plus the optimum solution.

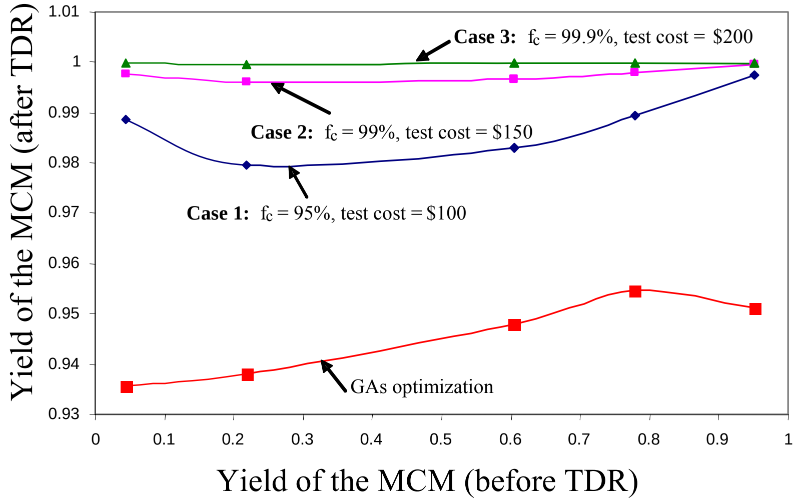


Fig. 9. Output yield of the MCM after TDR versus incoming yield of the MCM for the three test scenarios in [15] plus the optimum solution.

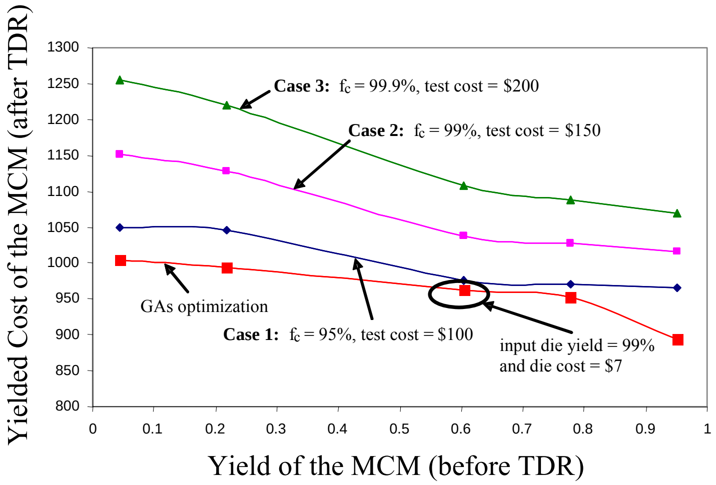


Fig. 10. Yielded cost of MCM after TDR versus incoming of MCM for the three test scenarios in [15] plus the optimum solution.

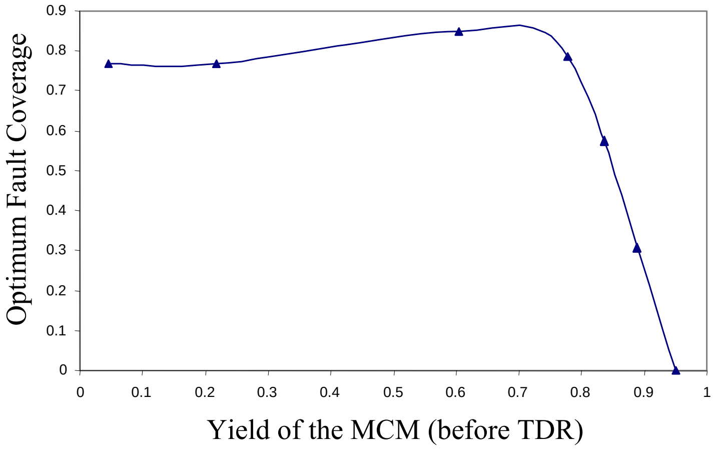


Fig. 11. GA-based determination of fault coverage (the fault coverage that minimizes yielded cost) versus incoming yield of MCM.

Comparing with the calculation results from the three test strategies discussed in Abdir, the cost of the MCM (after TDR) for each yield level of MCM (before testing) is lower, which is derived from the optimum test coverage generated by GAs optimization. This means choosing or buying a single value of fault coverage for the entire range of possible incoming die yield is not appropriate, i.e., a different fault coverage should be chosen (if possible) for different incoming die yields. The optimum fault coverage generated by the GAs varies with the different die yield to the MCM assembly process flow and therefore results in a lower MCM cost.

The final yield of the MCM after going through all the assembly and test process flow is shown in Fig. 9 in terms of the yield of the assembled the MCM before testing. The GAs optimization results in the lowest outgoing yield of the MCM (this is an artifact of the use of yielded cost as the objective function). It is always true that higher incoming yield leads to higher yield level of the output. Alternative objective functions might result in a different choice of fault coverages and thereby yields.

Neither Fig. 8 or Fig. 9 represent the quantity actually minimized in the analysis, i.e., the cost divided by the yield. Fig. 10 shows the yielded cost and implies that the optimization algorithm proposed in this paper does in fact generate the solution that minimizes the yielded cost of MCM after final assembly and test. Fig. 11 shows the fault coverage (determined by the GA optimization) for the range of values of incoming yield of the MCM module. For high incoming yield, the optimization algorithm chooses the lower fault coverage to decrease the cost of test. The optimization results prefer less than 100% fault coverage due to the high cost associated with the test process in Fig. 11.

After all the optimum fault coverages are generated by the GAs optimization, the results can be verified using the Trichy *et al.* model discussed in [7] and Section 3 by assessing a full range of fault coverages for a specific incoming die yield and comparing the GAs optimum fault coverage value with those calculated values. Fig. 12 describes the relationship between all possible test fault coverages for the MCM and the yielded cost of the MCM after the TDR is done for the cost of incoming die at $7 with a yield level of 99% (this point is indicated in Fig. 10). In Fig. 12, the expected minimum yielded cost should be lower than $964.14 with the possible range of fault coverage between 0.8 and 0.9, which validates the optimum result

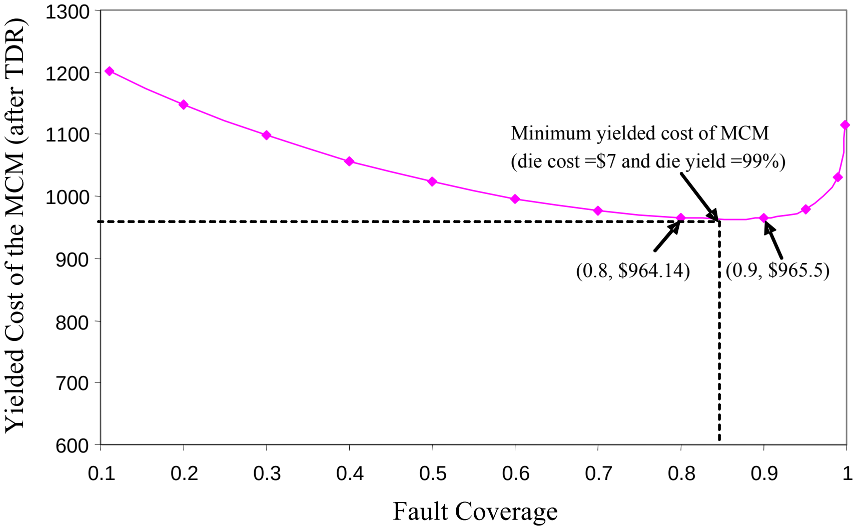


Fig. 12. Yielded cost of the MCM (after TDR) versus fault coverage (die cost = $7 and die yield = 99%).

from optimization algorithms where the minimized yielded cost is $962.28 and the optimum of fault coverage is set at 0.85.

This section applied the GAs optimization to minimize the yielded cost of an MCM during its assembly process with fault coverage optimized. The optimization results have been verified as the minimum for various combinations of die cost and yield. Compared with the test strategies discussed in Abdir, in which there is no optimization performed, the GAs based optimization successfully generates the lowest value of the objective function defined in Section 4 by selecting the optimum characteristics of the test. From Table 3, the GAs-based optimum yielded cost results in an average cost reduction efficiency[4] of 6.85% lower than that obtained when using test strategies based on selecting specific fault coverages in Abdir. Furthermore, in a complicated process flow, it would be impossible to enumerate all fault coverages without employing optimization techniques.

6.3. Optimization of Rework Attempts for MCM Assembly Process Flow

The maximum number of rework attempts is fixed at one for the optimization of the MCM assembly case in all of the discussion so far in this paper. One rework attempt means that the module is scrapped if it does not pass the test after a single rework attempt (no second attempt to rework the module is made). This section will include rework attempts as a variable to be optimized for finding the lowest yielded cost of systems.

Table 3. Cost reduction efficiency of minimum yielded cost with various input of die yield

Die cost ($)/Die yield		2/0.8	3/0.85	4/0.9	5/0.94	6/0.97	7/0.99	8/0.995	9/0.999	Average
Minimum yielded cost ($)	Test strategies in [15]	1046	1047	1046	1049	1045	975	971	965	1018.12
	GAs-based Optimization	896	921	958	1004	994	962	952	894	947.507
	Cost reduction efficiency	0.14	0.12	0.08	0.04	0.05	0.01	0.02	0.1	0.0685

According to the multi-variable objective function defined in Section 4, a sequence of variables could be optimized simultaneously during the GAs optimization process, i.e., the optimization algorithms can be extended to accommodate more than one variable that needs to be optimized in order to minimize the yielded cost. The difference between placing the rework attempts into the optimization loop and optimizing the fault coverage of the test is the range of values predefined in the analysis. For the fault coverage, the

[4] The cost reduction efficiency is defined as the ratio of cost reduction between the minimum yielded cost generated by GAs-based optimization and calculated using the test strategies in [15] divided by the latter value.

GAs generate candidates in the range from 0 to 1. The variable representing rework attempts should be the integer greater than or equal to 0.[5]

The MCM assembly process flow is used in this section for optimizing the number of rework attempts (the specific case considered is a die yield of 99% and die cost of $7 – indicated in Fig. 10). The rework cost is assumed to be $100/module for all rework attempts and the rework yield is also fixed at the minimum value of 95% and the die yield. The minimum yielded cost decreases from $962.28 (with a fixed maximum number of rework attempts set at one) to $902.68 (with the number of rework attempts optimized). The optimized number of rework attempts at the minimum yielded cost is 8. Assuming a fixed rework cost, the GAs optimization prefers more rework operations to improve the outgoing yield of the MCM. The fault coverage for the minimum yielded cost also changed when the number of rework attempts is optimized. The fault coverage when a maximum of only one rework attempt was allowed (0.85), increased to 0.94 when the number of rework attempts was optimized. The optimization result indicates that the GAs optimization is willing to pay for higher fault coverage in order to enable the test to fail more defective MCMs when more rework attempts are available.

7. Conclusions and Discussion

The goal of this paper is to develop a methodology for optimizing the location(s) and characteristics of TDR operations for complex process flows in order to minimize the yielded cost of electronic systems. It should be pointed out that although we have used a genetic algorithms approach to implement the methodology in this paper, other numerical optimization approaches may also be applicable (and potentially preferred over genetic algorithms).

For an example MCM assembly case, the optimum yielded cost resulting from the optimization methodology developed in this paper is an average of 6.85% lower than that obtained when using the specifically enumerated test strategies (fault coverages) from [15]. In a complicated process flow, it would be impossible to enumerate all the combinations of fault coverages with various inputs. The optimization methodology developed in this paper will effectively find the optimized solution in less time than

[5] For practical rework operations, the number of rework attempts is usually limited to less than 10. The range of values of the gene representing the rework attempts variable is defined between 0 and 10 in this paper. Zero rework attempts means there is no rework operation included.

traditional approaches and help manufacturers choose the best assembly process flow for achieving the goal of minimized yielded cost by enabling the identification of optimum test locations and their characteristics. With experience, we believe that the development of better test and rework application and placement heuristics could be an outcome of using a methodology like this one. The optimization results can also be used as the feedback to Design for Test (DFT) of electronic systems.

There are several simplifying assumptions that underlie the implementation of the optimization methodology demonstrated in this paper. First, the test operation modeled only accounts for a single fault type. Multiple faults could be modeled as parallel TDR operations with unique characteristics for each relevant fault type. Second, the rework operation is considered to happen immediately after the test, in reality, the rework could take place at a later point in the process. Finally, the methodology assumes that the cost of performing a TDR (for an equivalent fault coverage and rework yield) is the same independent of the location, which is not necessarily true.

In addition, the optimization algorithms developed in this paper could be extended to find the optimum fault coverages that are applicable to a range of inputs (e.g., a probability distribution) instead of being restricted to a single value and produce robust optimums that account for uncertainties in the input parameters. This can be done by embedding Monte Carlo analysis within the cost analysis portion of the calculation, which produces a distribution of yielded costs for each case considered in the optimization. The ranking of the cases by the optimizer is then based on some descriptive attribute of the distribution of yielded costs (e.g., the mean, or some parameter that folds the mean and the width of the resulting distribution together can be used to represent the fitness of the objective function). Preliminary robust optimization results for test/diagnosis/rework location and characteristics problems appear in [16].

References

1. T. W. Williams and N. C. Brown, "Defect Level as a Function of Fault Coverage," *IEEE Trans. on Computers,* vol. C-30, no. 12, pp. 987-988, December 1981.
2. E. H. Volkerink, A. Khoche, L. A. Kamas, J. Revoir, and H. G. Kerkhoff, "Tackling Test Trade-offs from Design, Manufacturing to Market Using Economic Modeling," *Proc. International Test Conference,* pp. 1098-1107, November 2001.

3. P. K. Nag, A. Gattiker, S. Wei, R. D. Blanton, and W. Maly, “Modeling the Economics of Testing: A DFT Perspective,” *IEEE Design & Test of Computers,* vol. 19, pp. 29-41, Jan/Feb 2002.

4. C. Dislis, J. H. Dick, I. D. Dear, I. N. Azu, and A. P. Ambler, Economics modeling for the determination of test strategies for complex VLSI boards, *Proc. of the Int. Test Conf.*, pp. 210-217, 1993.

5. M. Tegethoff, and T. Chen, “Defects, fault coverage, yield and cost, in board manufacturing,” *Proc. of the Int. Test Conf.*, pp. 539-547, 1994.

6. M. Abadir, A. Parikh, L. Bal, P. Sandborn, and C. Murphy, “High level test economics advisor,” *Journal of Electronic Testing: Theory and Applications*, vol. 5, no. 2&3, pp. 195-206, 1994.

7. T. Trichy, P. Sandborn, R. Raghavan, and S. Sahasrabudhe, “A New Test/Diagnosis/Rework Model for Use in Technical Cost Modeling of Electronic Systems Assembly,” *Proc. of Int. Test Conf.,* pp. 1108-1117, 2001.

8. M. Driels and J. S. Klegka, “Analysis of Alternative Rework Strategies for Printed Writing Assembly Manufacturing Systems,” *IEEE Transactions on Components, Hybrids, and Manufacturing Systems*, Vol. 14, No. 3, September 1991, pp. 637-644.

9. Z. Shi and P. Sandborn, “Optimization of Test/Diagnosis/Rework Location(s) and Characteristics in Electronic Systems Assembly Using Real-Coded Genetic Algorithms,” *Proc. International Test Conference,* pp. 937-946, 2003.

10. P. Goel, “Test Generation costs analysis and projections,” *Proc. of Design Automation Conference,* pp. 77-84, 1980.

11. D. Becker and P. Sandborn, “One the Use of Yielded cost in Modeling Electronic Assembly Processes,” *IEEE Trans. on Electronics Packaging Manufacturing*, vol. 24, pp. 195-202, July 2001.

12. R. L. Haupt, and S. E. Haupt, *Practical Genetic Algorithms,* Wiley, 1998.

13. C. Z. Janikow and Z. Michalewicz, “An Experimental Comparison of Binary and Floating Point Representations in Genetic Algorithms,” *Proc. of the 4th Int. Conference on Genetic Algorithms*, pp. 31-36, 1991.

14. A. H. Wright, “Genetic Algorithms for Real Parameter Optimization,” *Foundations of Genetic Algorithms*, Morgan Kaufmann Publishers, pp. 205-218, 1991.

15. M. Abadir, “Economics of Multichip Modules Testing Strategies,” *IEEE Transactions on Components, Packaging, and Manufacturing Technology,* Vol. 21, No. 4, pp. 360-370, November 1998.

16. Z. Shi, “Optimization of Test/Diagnosis/Rework Location(s) and Characteristics in Electronic Systems Assembly,” Ph.D. Dissertation, Department of Mechanical Engineering, University of Maryland, College Park, MD., 2004.