Integration and test strategies for semiconductor manufacturing equipment[1]

I.S.M. de Jong[†,‡], R. Boumen[‡], J.M. van de Mortel-Fronczak[‡], J.E. Rooda[‡],
[†]ASML, P.O. Box 324, 5500 AH Veldhoven, The Netherlands.
[‡]Systems Engineering Group, Department of Mechanical Engineering,
Eindhoven University of Technology, 5600 MB, Eindhoven, The Netherlands.
[†]ivo.de.jong@asml.com, [‡]{I.S.M.d.Jong, R.Boumen, J.M.v.d.Mortel, J.E.Rooda}@tue.nl

Copyright © 2006 by I.S.M. de Jong. Published and used by INCOSE with permission

Abstract. The complexity of semiconductor manufacturing equipment is growing. This growth results in a complexity increase of the integration and test phase of these systems. Simply adding more test resources is not possible anymore, because of the cost involved. A better design of an integration and test strategy can help to optimize this hectic phase. However, methods to design and evaluate integration and test strategies for multi-disciplinary systems are hardly available. In this paper, we present a method to design and compare integration and test strategies. Following this method, an optimal integration and test strategy can be chosen from a set of possible strategies. A case has been performed where a system is integrated and tested using three different integration and test strategies: a time-to-market-driven strategy, a quality-driven strategy and a combined quality and time-to-market strategy.

Introduction

The semiconductor manufacturing business has evolved from a niche to one of the main technology enablers in the last fifty years. The well known Moore's Law (Moore 1965) predicted that the number of transistors in an integrated circuit (IC) would double every 18 months. This has indeed happened and it resulted in the ability to use ICs in more complex systems for more complex tasks. The speed of innovation in this business is driven mainly by the capabilities of newly developed IC manufacturing equipment. Each new generation of manufacturing equipment leads to the ability to manufacture thinner IC-lines faster. Each new generation also leads to an increase of complexity, because of the tighter specifications. The number of components, interfaces and multi-disciplinary nature of these components result in an integration and test phase which is difficult to predict and control. This is the main reason for us to investigate the impact of integration and test strategies for an wafer scanner, developed and manufactured by ASML (ASML, 2005).

Nowadays, integration and test strategies are defined manually based on expert knowledge. Defining an integration and test strategy can be an easy task for small mono-disciplinary systems. Experts can manually adapt integration and test strategies such that the optimal parameters like time-to-market and cost are taken into account. Comparing strategies is done using known mono-disciplinary criteria, like the amount of code coverage for software systems and the number of tested inputs and outputs for electronic boards.

[1] This work has been carried out as part of the TANGRAM project (www.esi.nl/tangram) and partially supported by the Netherlands Ministry of Economic Affairs under grant TSIT2026.

Mono-disciplinary methods often cannot be used to integrate and test large multi-disciplinary systems. Therefore, integration and test strategies are still developed by experts. The main reason for the design of integration and test strategies by experts is that multi-disciplinary methods to design and evaluate these strategies hardly exist. In this paper, we present a method to describe, analyze and evaluate multi-disciplinary integration and test strategies.

The structure of this paper is as follows. In the section "*Integration and test strategies*" a definition of an integration and test strategy is given and the key performance indicators of such a strategy are defined. The next section "*Modeling and analysis of integration and test strategies*" describes a method to model and compare integration and test strategies. This method is applied in the following section on two cases: a time-to-market-driven strategy and a quality-driven strategy. This paper is finalized with conclusions and references.

Integration and test strategies

An integration and test strategy consists of the following elements:

- an integration and test sequence, consisting of one or more integration phases and one or more test phases,
- a test process configuration and
- the test stop criteria for each test phase.

Integration and test strategies should be compared using criteria applicable to a variety of systems. These high level criteria (key performance indicators) that characterize integration and test strategies are:

Φ *Total integration and test duration*, defined as the time from the start of the integration and test phase until the moment of meeting a stop criterion for stopping this phase.

C *Integration and test cost,* defined as the sum of the costs of all assembly actions, disassembly actions, actions to execute test cases, diagnosis actions, actions on developing fixes for observed faults and applications of these fixes during the test phase.

R_R *Remaining risk,* which is our measure for quality and is defined as the risk which remains in the system after the stop criterion is reached for the integration and test phase. The risk in the system can be determined by summing the risk which is in the system for each possible fault s. The risk for each possible fault s can be calculated by multiplying the probability that the fault exists in the system with the impact of that fault if it exists in the system: $R(s) = P(s)I(s)$.

The remaining risk can be determined at the end of an integration and test phase using the remaining possible faults in the system and the impact of these fault states. Consequently, the risk at any point in time can be calculated also.

These key performance indicators are illustrated by an example telephone system consisting of three modules: a handset, a cable, a device and the interfaces between handset and cable and cable and device. Figure 1 is a graphical overview of the telephone system.

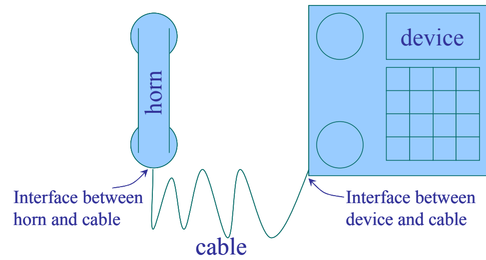

Figure 1 Overview of the telephone system

The first integration and test sequence, depicted by Figure 2, is an integration sequence where each assembly step is followed by a test step. The components themselves are not tested before integration. The stop criterion of each test phase is $R_R = 0$, so test until all remaining risk is removed. Assembly steps, A_i, are depicted with orange hexagonals and test steps, T_i, are depicted with green circles.

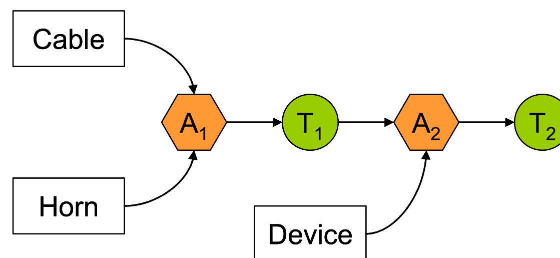

Figure 2 Example integration and test sequence 1

The key performance indicators for this integration and test strategy can be derived as follows:

$$\Phi = \Phi_{A_1} + \Phi_{T_1} + \Phi_{A_2} + \Phi_{T_2}$$
$$C = C_{A_1} + C_{T_1} + C_{A_2} + C_{T_2}$$
$$R_R = 0$$

The duration and cost of developing or manufacturing the individual components is not taken into account. The assumption is that all components are available when integration and testing is started. A *risk-profile* of the sequence of Figure 2 is depicted in Figure 3. This risk profile depicts the risk as function of time. Risk increases at t=1 with 1 risk unit per developed module. At t=2 additional risk is introduced by the assembly of the cable and the handset. The reason for

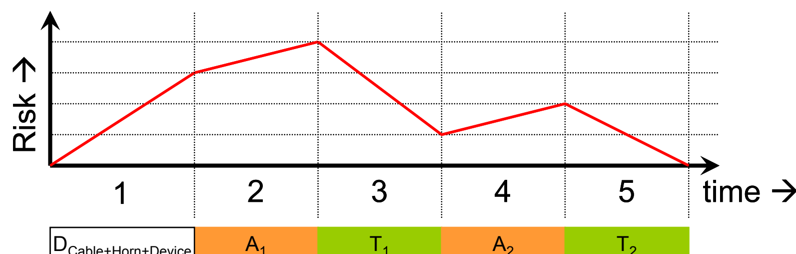

Figure 3 Risk profile of the integration sequence of Figure 2

this risk is the existence of interface faults, which are introduced by the assembly of the Handset and Cable. Testing at t=3 reduces the risk of the Cable, Handset and interface between the Cable and Handset to 0. The remaining risk in the system at that point is 1 risk unit from the Device. At t=4 risk is again introduced due to the interfaces between Device and rest of the phone. This risk is reduced by the test phase at t=5. The remaining risk after test phase T2 is $R_R = 0$.

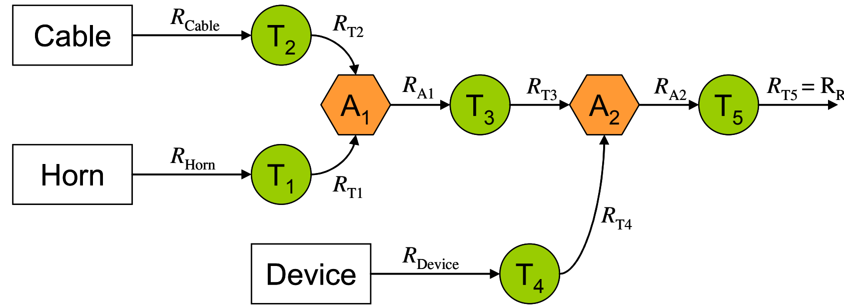

Figure 4 Example integration and test sequence 2

In a second integration and test sequence, each module is tested first. Every assembly is tested also. The stop criterion for each test phase is now at $R_R = 20\%$ of the initial risk of the test phase. So 80% of the risk is reduced with each test phase. Figure 4 depicts this strategy. The risk after each phase is denoted on the edges between the phases. Now the key performance indicators integration and test phase 2 can be derived as follows:

$$
\begin{aligned}
\Phi &= \max((\max(\Phi_{T_1}, \Phi_{T_2}) + \Phi_{A_1} + \Phi_{T_3}), \Phi_{T_4}) + \Phi_{A_2} + \Phi_{T_5} \\
C &= C_{T_1} + C_{T_2} + C_{A_1} + C_{T_3} + C_{T_4} + C_{A_2} + C_{T_5} \\
R_R &= R_{T_5} \\
&= 0.8 R_{A_2} \\
&= 0.8(R_{T_3} + R_{T_4}) \\
&= 0.64 R_{A_1} + 0.64 R_{Device} \\
&= 0.64(R_{T_1} + R_{T_2}) + 0.64 R_{Device} \\
&= 0.64(0.8 R_{Horn} + 0.8 R_{Cable}) + 0.64 R_{Device} \\
&= 0.512 R_{Horn} + 0.512 R_{Cable} + 0.64 R_{Device}
\end{aligned}
$$

The corresponding risk-profile is depicted in Figure 5.

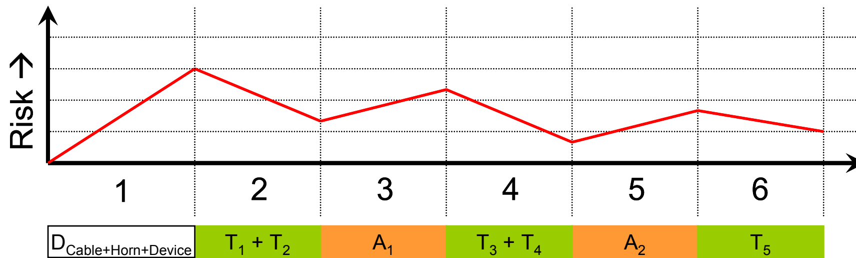

Figure 5 Risk profile of the integration sequence in Figure 4

Again this is a fairly straight forward derivation of the key performance indicators. The complexity of the derivation increases with the size of the system which needs to be integrated. Additionally, the assumption is that the values for each of the key performance indicators are deterministic, which is only sometimes the case for testing. Namely, the test process is a stochastic process, because of two reasons: the actual set of faults in the system under test can only be estimated, but is not known in advance (otherwise testing would not be necessary); the second reason is the configuration of the test process, which is described next.

A generic test, diagnosis and fix process, depicted in Figure 6, consists of six sub-processes: *select test sequence*, *dispatch test case*, *test execution*, *diagnose faults*, *fix faults* and *buffer the fixes* which are to be applied on the system under test.

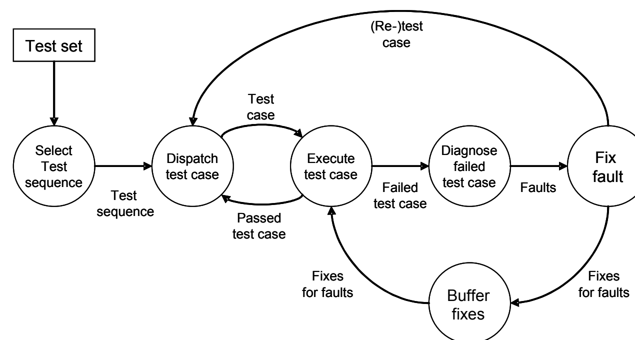

Figure 6 Graphical overview of a generic test execution, diagnosis and fix process

The sequence of test cases is selected first. A dispatcher dispatches the test sequence which is generated by the *select test sequence*-process. Test cases are dispatched one by one to a system under test for test execution. Test execution results in a verdict (*passed* or *failed* or *other*). A passed test case results in a new dispatched test case by the dispatcher. Failed test cases are diagnosed by the diagnosis process. Diagnosed test cases are fixed by the fix process. After fixing the fault, a retest of the original failing test case can be requested. After fixing, one or more fixes are buffered and applied on the system under test. Buffering of fixes is required because the system under test can still be executing test cases at the moment fixes are available. Testing is finished when the stop criterion is reached.

Two test process configurations are used for the cases in this paper. In the parallel, time-to-market-driven, test process configuration, parallel test execution and fixing takes place. In this strategy fixes are applied on the system under test while test execution is progressing. In the serial, quality-driven, test process configuration, diagnosis is started when all test cases are executed and all results are available. Fixes are applied after test execution with diagnosis is finished.

The configuration of the test process influences the duration of the total test phase. Parallel testing and fixing can lead to total test durations which are significantly lower than total test durations in non-parallel cases. The reason for this is that all test cases are always executed in the serial configuration, while in the parallel configuration testing can be stopped earlier. Testing can

be stopped because the failures have been fixed already (if the diagnosis and fix duration does not exceed the duration of the remaining test phase). Testing combined with fixing can, on the other hand, result in variability in test duration and test cost. Other sources of variability in the test process are the expected quality of the system under test and the chosen test sequence. Both sources of variability are now described. The individual variation of test execution, diagnosis and fix duration and cost also leads to variability in total test duration and cost, but is not taken into account here.

The expected quality of the system under test causes variability, because test cases and a test sequence are derived using these expectations. The exact moment of detecting a fault cannot be known at forehand, so the exact moment of diagnosing and fixing a fault is unknown too. The moment a fault has been fixed more test cases will pass and testing could stop earlier. Therefore the quality of the system under test causes variability. Some of this variability can be reduced by using a test sequence that tries to detect faults as early as possible in the test phase. The paper (Rothermel, Elbaum 2003) shows that the sequence of test cases influences the moment of fault detection. In this paper, different test sequencing techniques were applied on a known faulty software system and the *fault detection rate* was measured. A random test selection technique caused low fault detection rates. Another method to select test cases, called partition based test selection, performed better, in this study.

Two often used integration and test strategies are a *time-to-market-driven strategy* and a *quality-driven strategy*. The goal of a *time-to-market-driven strategy*, as often used at ASML for newly developed products, is to release a product as soon as possible. In this strategy test phases are stopped while risk is still remaining in the system and a parallel test execution, diagnosis and fix configuration is used. The goal of the other strategy, a *quality-driven strategy*, is to minimize risk in the system as soon as possible and completely with each test phase. All possible test phases are executed. Test phases which are executed later in the integration sequence are not influenced by lower level faults which are still present in the system. These lower level faults have been removed at that point in time. The analysis and comparison of both strategies require a single method to describe these strategies. This method will be explained in the next section, followed by a section describing the two cases: a time-to-market-driven strategy and a quality-driven strategy.

Modeling and analyzing integration and test strategies

Integration and test strategies can be modeled using the elements as described above. The key performance indicators of a test sequence can be calculated analytically assuming that the duration and cost of a test phase are deterministic. However, the stochastic behavior and high variability of a test phase requires an analysis method which is able to deal with these aspects. Moreover, integration sequences and systems under test change often. An adapted test and integration sequence should give results quickly. The input models should be easy to maintain and change and the determination of the key performance indicators of a test phase should be fast.

The method described in this paper consists of the following steps:

1. Define the ***integration and test sequence***
2. Model the system under test, the ***system test model***
3. Determine the ***test process configuration***, ***test stop criteria*** and ***test sequence for each test phase***
4. Simulate numerous integration and test sequence executions
5. Analyze the simulation results

The remainder of this section describes the models and techniques which are used. First, models for simulation of the ***integration and test sequence*** are described. The ***system test model*** follows. Next, some techniques to determine ***test sequences*** and ***test stop criteria*** are defined. ***Test process configurations*** were already described in the previous section.

For the modeling of the ***integration and test process*** a parallel process paradigm has been used, supported by the χ modeling language (Rooda, Vervoort 2004). The modeling language is developed in the Systems Engineering group of the Eindhoven University of Technology for modeling, analysis and optimization of manufacturing lines. We use the modeling approach of manufacturing lines in the context of integration and test process modeling. In a sense, we see the integration and test process as a manufacturing line that produces tested systems. The χ simulator is able to simulate large numbers of integration and test sequences in a short time.

A ***system test model*** is used to describe the multi-disciplinary system under test and the test cases which can be performed on the system under test. The same model is utilized to simulate testing with a test process simulation model. A system test model D describes a system under test from the perspective of testing. This model can be derived from early failure mode effect analysis (FMEA), specifications, detailed designs, UML use-cases and known sets of test cases.

The model D used in this paper is inspired by the work of Pattipati et. al. (Pattipati et.al., 1991). The following elements are described in D:

- The set of test cases, T, with a number of properties for each test case: test cost T_C, test duration T_T and diagnose cost T_{DC} and diagnose duration T_{DT}. The cost and duration of diagnosis are utilized when the test case fails and the result has to be diagnosed.
- The set of possible fault states, S, with a number of properties for each fault state: probability of occurrence P, impact if not detected in the test phase I, cost and duration of fixing the fault state, S_C, S_T. The fix cost and duration are applied when a fault state was present and is fixed. The impact I can be quantified in dollars or as impact on performance of the system.
- The relation between tests and fault states, or in other words, the coverage of every test in T on a set of fault states from S: R_{TS}. A relation between a test and a fault state is indicated by *1*. No relation is indicated by *0*.

This system test model of Pattipati (Pattipati et.al., 1991) has been extended by Boumen et. al. (Boumen et.al., 2006). This extended model can be used for modeling system level test phases.

In addition, the extended model allows modeling of:

- Fault states that (re-)appear when a specific fix is applied
- Multiple outcomes, instead of only *Pass* and *Fail* as test outcomes
- The ability to model the relation between a test and a fault state other than *0* or *1*, so that partly sensitive tests can be modeled.

The telephone system from Figure 1 is used again as an example to illustrate a system test model. A more abstract representation of this telephone, related to the system test model, is given in Figure 7.

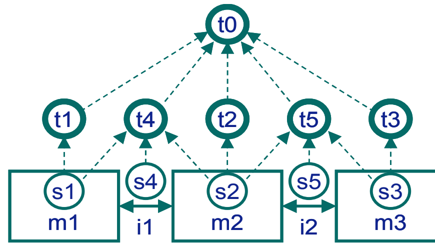

Figure 7 The telephone example with fault states and test cases

The handset, cable and device of the telephone are represented as m1, m2 and m3 in Figure 7. The interfaces between the modules are labeled i1 and i2. The possible fault states S are labeled as s1, s2, s3, s4 and s5. For example, s1 is a fault state which can be present in m1 and s4 is a fault state which can be present in the integrated m1 and m2. Finally, the set of tests T are depicted as t_0 to t_5. Test t_0 covers all fault states, while t_1 only covers fault state s_1. A convenient representation of the system test model is given in Table 1.

S \ T	t_0	t_1	t_2	t_3	t_4	T_5	P	I	S_C	S_T
s_1	1	1	0	0	1	0	10%	1	24	24
s_2	1	0	1	0	1	1	10%	1	24	24
s_3	1	0	0	1	0	1	10%	1	24	24
s_4	1	0	0	0	1	0	10%	1	24	24
s_5	1	0	0	0	0	1	10%	1	24	24
T_C	3	1	1	1	2	2				
T_T	1	1	1	1	1	1				
T_{DC}	18	6	6	6	12	12				
T_{DT}	9	3	3	3	6	6				

Table 1 System test model of a telephone system

Fault states can be derived from known faulty modules, FMEA/FMECA analysis, UML use cases, etc. The test set S is normally a set of known test cases to start with.

By using a system test model the modeler can statically analyze the system test phase. Coverage analysis reveals weak spots or even uncovered spots in the test phase. Double test cases can be detected easily, because both test cases cover the same set of fault states i.e. the *test signatures* of both tests are equal. Finally, these models can also be used for test sequencing (see below) or determination of the next best test case to be developed.

Determining a ***test sequence*** of a system can be done in many ways. An *undirected* test sequence is randomly ordered. A *directed* test sequence on the other hand uses a strategy to define the sequence of test execution. A common directed test sequencing technique is *risk-based* test sequencing (Rothermel, 1996 and Harrold, 2001). In this technique, the first test case in the sequence covers the highest risk, the second test case covers the second highest risk, and so on. Result-based test sequencing (Boumen et.al., 2006) is a novel directed test sequencing technique which takes into account the individual results of each test case and uses the same system test model as described above.

Directed test sequences can be defined before actual testing is started (offline test sequencing) and also during test execution (online test sequencing). New methods like exploratory testing (ET) (Bach, 2002) use online test sequencing methods to determine the next test case in the sequence. ET uses informal techniques to determine the next test case. An online risk-based test sequencing technique selects test cases while testing is progressing, using the online calculation of the risk covered by the available test cases. This formalizes informal online test sequencing techniques like ET and uses a system test model as input.

Finally the ***test stop criterion*** is defined. Boris Bezier (Bezier, 1983) noted on when to stop testing: "*There is no single, valid, rational criterion for stopping. Furthermore, given any set of applicable criteria, how each is weighed depends very much upon the product, the environment, the culture and the attitude to risk*". In other words, the stop criteria which are used in industry are based on combination of metrics like problems found and test progress. These metrics approach the required stop criterion as close as possible. In our approach, we define stop criteria on total test duration, quality (risk) and cost. Distances from meeting the stop criteria are known at any point during the simulation. The simulation is stopped immediately when the test stop criterion is satisfied. Stop criteria in terms of time and cost are measurable in an industrial context. A stop criterion based on quality (risk) is easy to define and measure in a simulation environment and not in an industrial case. The exact stop moment is important. However, the path, the risk profile, to this stop criterion gives insight in the quality of the stop decision.

Measuring and analysis of integration and test strategies

The result of a single simulation of an integration and test sequence is an *integration and test profile*. This profile consists of profiles for risk, as depicted before in Figure 3, and cost (not depicted) as function of the integration and test time. These profiles can be used to analyze a single integration and test execution. High peaks in risk are visualized and the integration and test sequence can be adjusted to minimize these high peaks if required.

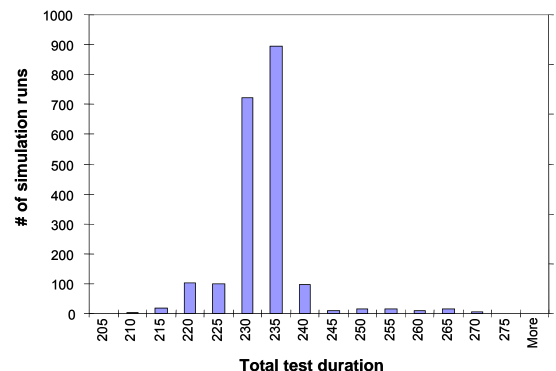

Figure 8 Variation in total test duration of an integration and test sequence

Another way of measuring and analyzing key performance parameters of an integration and test phase is by simulating numerous executions of an integration and test sequence. The actual faults in the system under test are randomly selected for

each simulation. Each simulation results in a total test duration, cost and risk, and an integration and test profile. A histogram of one of the key performance parameters, total test duration, is depicted in Figure 8. We will use the distribution of total test duration for the following cases. Similar graphs can be obtained for cost and remaining risk.

Cases: a time-to-market-driven and a quality-driven strategy

The methods described above are used to perform two cases for a simplified wafer scanner. In case 1, a time-to-market-driven integration and test strategy is used. In case 2, a quality-driven integration and test strategy is used. In both cases the same simplified wafer scanner is used to make sure that the system under test itself is not affecting the performance of the integration and test strategy. A graphical overview of the system under test is presented in Figure 9. The results in case 2 indicate that a combination of a time-to-market and quality driven strategy could be beneficial. This combination is investigated also.
The simplified wafer scanner consists of six sub-systems which all are connected to a base-frame. The sub-systems are: a reticle handler (RH), a wafer handler (WH), a reticle stage (RS), a wafer stage (WS), a projection lens (LE) and a laser system (LS). The system test model for the wafer scanner is presented in Table 2.

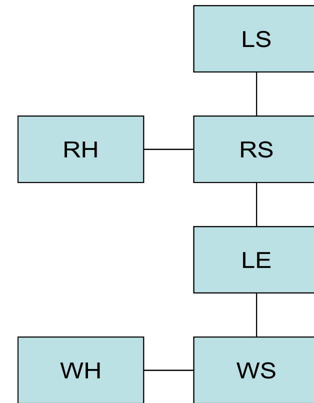

Figure 9 The simplified wafer scanner

A single test in Table 2 represents a set of test cases and a fault state represents a set of fault states. Average properties for test phases and fault sets are presented in this table. This table, therefore, gives an overview of the test model and is not the complete model. The sub-systems consist of 10 fault states and 10 test cases. The interfaces consist of 5 fault states each covered by a single test. Each interface test additionally covers a number of fault states of previous test sets.

The fault states of the sub-systems have different failure probabilities. The average failure probability per sub-system is presented in. As can be seen, the RH and WS have a slightly lower failure probability than the other subsystems. The average fault probability is fairly high, which corresponds to a newly developed system. The diagnose costs and durations for interfaces are higher than the diagnose cost and duration for sub-system fault states.

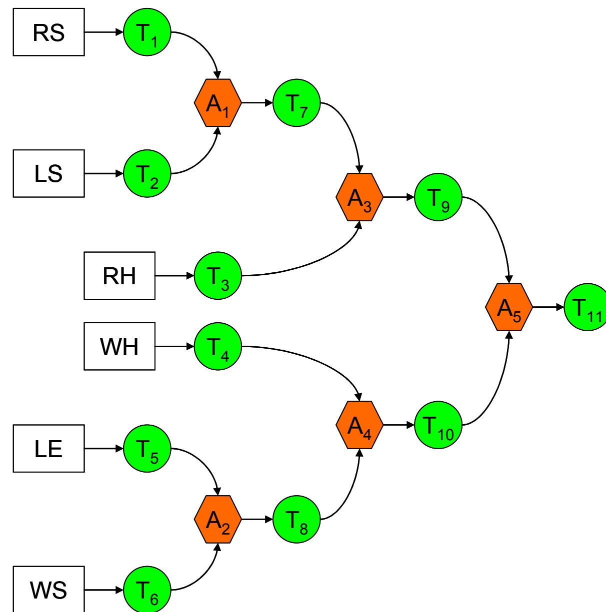

Figure 10 Parallel integration and test strategy for the simplified wafer scanner

S/T	Test phase LS	Test phase RS	Test phase RH	Test phase WH	Test phase LE	Test phase WS	Test phase LS-RS	Test phase LSRS-RH	Test phase LE-WS	Test phase LEWS-WH	Test phase LSRSRH-LEWSWH	$\overline{P}$	$\overline{I}$	$\overline{S_C}$	$\overline{S_T}$
LS	0.1	0	0	0	0	0	0.16	0.02	0	0	0.02	83%	1	1	1
RS	0	0.1	0	0	0	0	0.16	0.02	0	0	0.02	82%	1	9.5	9.5
RH	0	0	0.1	0	0	0	0	0.16	0	0	0.04	70%	1	1	1
WH	0	0	0	0.1	0	0	0	0	0.16	0.02	0.02	83%	1	1	1
LE	0	0	0	0	0.1	0	0	0	0.16	0.02	0.02	83%	1	1	1
WS	0	0	0	0	0	0.1	0	0	0	0.16	0.04	70%	1	9.5	9.5
LS-RS	0	0	0	0	0	0	0.2	0.12	0	0	0.08	80%	1	4.4	4.4
LSRS-RH	0	0	0	0	0	0	0	0.2	0	0.06	0.12	73%	1	11	11
LE-WS	0	0	0	0	0	0	0	0	0.2	0	0.2	80%	1	7.8	7.8
LEWS-WH	0	0	0	0	0	0	0	0	0	0.2	0.2	73%	1	11	11
LSRSRH-LEWSWH	0	0	0	0	0	0	0	0	0	0	0.2	83%	1	7.8	7.8
$\overline{T_C}$	2.0	2.0	2.0	2.0	2.0	2.0	2.0	2.0	2.0	2.0	2.0				
T_T	2.0	2.0	2.0	2.0	2.0	2.0	2.0	2.0	2.0	2.0	2.0				
$\overline{T_{DC}}$	9.0	9.0	9.0	9.0	9.0	9.0	37.8	32.4	37.8	32.4	48.6				
$\overline{T_{DT}}$	2.0	2.0	2.0	2.0	2.0	2.0	37.8	32.4	37.8	32.4	48.6				

Table 2 System test model for the simplified wafer scanner

Newly developed wafer scanners contain many unknown faults which are detected in a very late stage. These unknown fault states are still present in a late stage because of unknown physical interactions that exist on a nanometer scale and because of incomplete design and test phases.

Case 1: A time-to-market-driven integration and test strategy

Step 1: Define the integration and test sequence

The relations between sub-systems dictate possible integration or assembly steps. For this case it is assumed that all sub-systems are available at the start of integration. The integration sequence, presented in Figure 10, is maximally parallel.

Step 2: Model the system under test

A system test model has been created for each sub-system and each interface. An overview of the test model is depicted in Table 2.

Step 3: Determine the test process configuration, test stop criteria and test sequence

A time-to-market integration and test strategy for an ASML wafer scanner has the following characteristics. Testing, diagnosis and fixing are executed in parallel. The integration sequence is

executed as parallel as possible, and test phases are skipped or stopped while leaving fault states in the system (R_R>0). The remaining faults are covered in later test phases. Furthermore, the modules in the system have a high average fault probability.

The test stop criterion for test phases T_1, T_2, T_3, T_4, T_5, T_6, T_7 and T_8 is set at R_R=0. T_9 and T_{10} are skipped (the stop criteria are set to R_R=100% of the initial risk). The T_{11} test phase is stopped when all possible faults are excluded or removed from the system, i.e. $R_R = 0$. So testing is performed mainly at a sub-system level and at the final level. An offline generated undirected test sequence is used for each test phase in the system. In other words, test cases are randomly selected for execution.

Step 4: Simulate numerous integration and test sequences

The integration and test sequence has been simulated. The average total test duration stabilizes after 2000 simulation runs.

Step 5: Analyze the simulation results

Figure 11 depicts the distribution of the total test duration of 2000 simulation results. The minimal duration is Φ_{min}=789, the maximal duration Φ_{max}=1468. On average 949 time units are used for this integration and test strategy. The variability of the total test duration is rather small for a system where each sub-system has been tested in three test phases. Every test phase adds variability to the total test duration. The configuration of the test process however can result in less variability on the other hand.

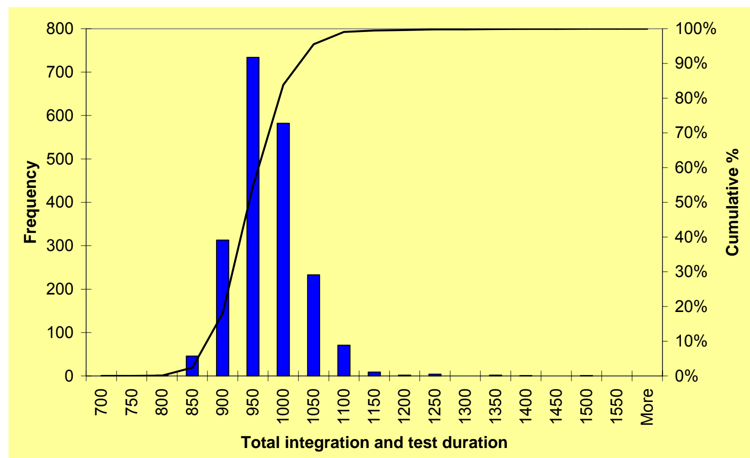

Figure 11 Total test duration for a time to market driven strategy

Case 2: A quality-driven integration and test strategy

Step 1: Define the integration and test sequence

The integration and test sequence that has been used for case 1 is used.

Step 2: Model the system under test

The system test model that has been used for case 1 is used.

Step 3: Determine the test process configuration, test stop criteria and test sequence

Diagnosing and fixing faults is performed for each test phase after all test cases have been performed. Testing is stopped in each test phase when all possible fault states are excluded, $R_R = 0$. All test phases are executed in this configuration. Test cases are, again, randomly selected for execution.

Step 4: Simulate numerous integration and test sequences

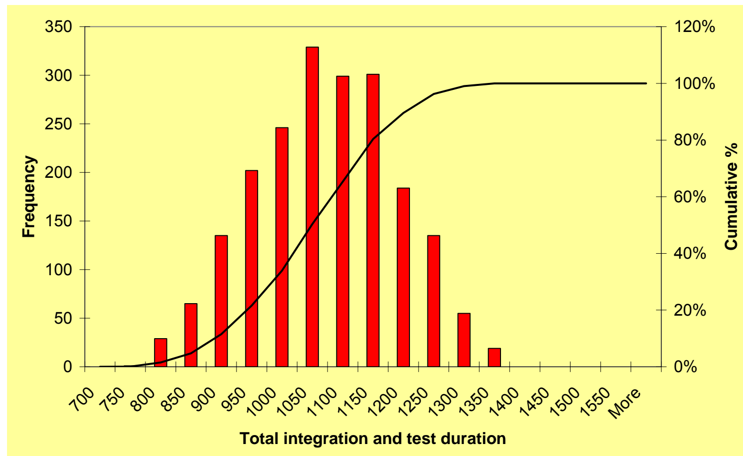

Figure 12 Total test duration for a quality-driven strategy

Figure 12 depicts the distribution of total test duration for 2000 simulation runs.

Step 5: Analyze the simulation results

The average duration of the quality-driven strategy, $\overline{\Phi} = 1024$, is almost equal to the average duration of the time-to-market strategy. However, the variability of the duration is much higher with a minimum of 738 and a maximum of 1348 time units. The standard deviations of the total duration of case 1 and case 2 are resp. $\sigma_{\Phi,\text{case } 1}=58$ and $\sigma_{\Phi,\text{case } 2}=116$. The main reason for the increased variability in this quality-driven strategy is the additional test phase that is performed on each sub-system. The test process configuration does not minimize the variability in this case.

Comparing both test strategies and the corresponding results leads to an additional integration and test sequence. In this sequence a combination of a time-to-market-driven and quality-driven strategy is used. This combined strategy should lead to a ‘best of both worlds’ in terms of total test duration. The only differences between the quality-driven strategy and the time to market driven strategy are the additional test phases T_9 and T_{10} in the quality-driven strategy and the configuration of the test process. Test phase T_9 and T_{10} are removed in the combined strategy, since more test phases lead to a higher variability. The test process configuration remains a serial test process, as used in the quality-driven strategy.

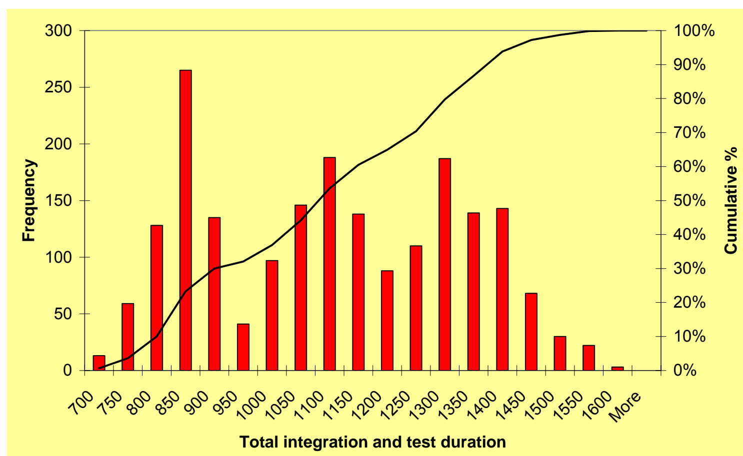

Figure 13 Total test duration for a mixed strategy

This strategy has been simulated and the results are depicted in Figure 13. With this combined strategy it is possible to obtain shorter total test durations. The minimal total test duration $\Phi_{min}=662$ now. However, shorter total test durations are only obtained in 8% of the cases and 13% of the simulation runs result in longer total test durations then in the quality-driven strategy. The variability of the distribution is much higher.

Detailed analysis of the integration and test profiles (not depicted) showed that the final test phase T_{11} can become very lengthy for this strategy. The reason is that the faults which are revealed in A_3, A_4 and A_5 have to be tested in T_{11}. The test process configuration of T_{11} is such that first all test cases are executed and then all faults are diagnosed and fixed. The diagnosis and fix duration of these high level system fault are higher then the lower level faults. The impact of this higher duration combined with the removal of T_9 and T_{10} can lead to long total integration and test duration. The actual duration depends on the fault state content of each generated system under test and therefore also short total integration and test durations are obtained.

Conclusions

In this paper, we presented a method to analyze and compare system level integration and test strategies. The key performance indicators of such integration and test strategies are defined as total integration and test duration Φ, cost C and remaining risk R_R. The method is supported with a set of techniques which enable designers to model and simulate integration and test strategies.

Following this method, two cases have been performed: the modeling and analysis of a time-to-market-driven strategy and the modeling and analysis of two quality-driven strategies. The strategies have been compared and the time-to-market-driven strategy resulted in the shortest average total test duration. The variability of this strategy was small compared to the other strategies.

A combination of both strategies resulted in the minimal possible total test duration. However, this minimal total test duration could only be obtained by 8% of the simulation runs. The total test duration was larger than the other two strategies in 13% of the cases. The combination of test process configuration and number of test phases determines the total test duration of a strategy. However, the total integration and test duration only becomes visible until the last test phase is progressing. It might already too late to decide to switch integration and test sequence or test process configuration in this last phase. Further research is required in the area of integration and test sequence optimization, such that optimal integration and test sequences can be obtained before integration and testing is started.

REFERENCES

ASML, http://www.asml.com, 2006

Bach, J., "*Exploratory testing explained*", 2002-2003, www.satisfice.com.

Bezier, B., "*Software testing techniques*", Van Nostrand Reinhold Company. ISBN 0-442-24592-0, 1983.

Boumen, R., de Jong I.S.M., van de Mortel-Fronczak, J.M., Rooda, J.E., "*Test sequencing in complex manufacturing systems*", Submitted at IEEE transactions on Systems, Man and Cybernetics – Part A: Systems and humans. To be published.

Harrold, M.J., Rosenblum, D., Rothermel, G., Weyuker, E., "*Emperical studies of a prediction model for regression test selection*", Transactions on software engineering, vol. 27, no. 3, 2001.

Moore, G., "*Cramming more components onto integrated circuits*", Electronics, Volume 38, 1965. http://www.intel.com/intel/intelis/museum/exhibits/hist_micro/hof/index.htm

Pattipati, K., et. al., "*Start: System testability analysis and reserach tool*", IEEE Aerosp.

Electron. Syst. Mag., January 1991.

Rothermel and Elbaum, "*Putting your best tests forward*", IEEE Software, September/October 2003.

Rooda, J., Vervoort, J., "*Analysis of Manufacturing Systems using χ0.8*", http://se.wtb.tue.nl, November 2004.

Rothermel, G., Harrold, M.J., "*Analyzing regression test selection techniques*", IEEE Transactions on software engineering, vol. 22, no. 8, 1996.

BIOGRAPHY

I.S.M. de Jong has a B.Sc. in Laboratory Informatics and Automation from Breda Polytechnic. He has been a software engineer in various companies in the USA and The Netherlands. Since 1996 he has worked with ASML in systems testing, integration, releasing and reliability projects. His specialization is in the field of integration and test strategies. Since 2003 he is an active member in the Tangram project.

R. Boumen received his M.Sc. degree in Mechanical Engineering from the Eindhoven University of Technology, the Netherlands, in 2004. During his work as a master student he worked in the field of supervisory machine control of lithographic machines. Since 2004 he is a Ph.D. student at the Eindhoven University of Technology. His research concerns test strategy within the Tangram project.

J.M. van de Mortel-Fronczak graduated in computer science at the AGH University of Science and Technology of Cracow, Poland, in 1982. In 1993, she received the Ph.D. degree in computer science from the Eindhoven University of Technology, the Netherlands. Since 1997 she works as an assistant professor at the Department of Mechanical Engineering, Eindhoven University of Technology. Her research interests include specification, design, analysis and verification of machine control systems.

J.E. Rooda received the M.S. degree from Wageningen University of Agriculture Engineering and the Ph.D. degree from Twente University of Technology, The Netherlands. Since 1985 he is Professor of (Manufacturing) Systems Engineering at the Department of Mechanical Engineering of Eindhoven University of Technology, The Netherlands. His research fields of interest are modeling and analysis of manufacturing systems. His interest is especially in control of manufacturing lines and in supervisory control of manufacturing machines.