J Electron Test (2012) 28:121–135
DOI 10.1007/s10836-011-5244-5

Scheduling Tests for 3D Stacked Chips under Power Constraints

Breeta SenGupta · Urban Ingelsson · Erik Larsson

Received: 1 February 2011 / Accepted: 11 August 2011 / Published online: 10 September 2011

Abstract This paper addresses Test Application Time (TAT) reduction under power constraints for core-based 3D Stacked ICs (SICs) connected by Through Silicon Vias (TSVs). Unlike non-stacked chips, where the test flow is well defined by applying the same test schedule both at wafer sort and at package test, the test flow for 3D TSV-SICs is yet undefined. In this paper we present a cost model to find the optimal test flow. For the optimal test flow, we propose test scheduling algorithms that take the particulars of 3D TSV-SICs into account. A key challenge in testing 3D TSV-SICs is to reduce the TAT by co-optimizing the wafer sort and the package test while meeting power constraints. We consider a system of chips with cores that are accessed through an on-chip JTAG infrastructure and propose a test scheduling approach to reduce TAT while considering resource conflicts and meeting the power constraints. Depending on the test schedule, the JTAG interconnect lines that are required can be shared to test several cores. This is taken into account in experiments with an implementation of the proposed scheduling approach. The results show significant savings in TAT.

Responsible Editor: E.J. Marinissen

This work was supported by Swedish Research Council.

B. SenGupta (✉) · U. Ingelsson · E. Larsson
Department of Computer and Information Science,
Linköping University, 581 83 Linköping, Sweden
e-mail: breeta.sengupta@liu.se

U. Ingelsson
e-mail: urban.ingelsson@liu.se

E. Larsson
e-mail: erik.larsson@liu.se

Keywords Power constrained test scheduling · 3D integration

1 Introduction

Integrated circuits (ICs) with multiple chips (dies) stacked and bonded vertically, interconnected with Through-Silicon Vias (TSVs), so called 3D TSV Stacked ICs (TSV-SICs), have lately attracted a fair amount of research [7–10, 18]. Recent research have addressed test architecture design for 3D TSV-SICs [15], testing the TSVs [7–10, 15, 18] and 3D TSV-SIC specific defects [7, 10]. Due to imperfections in IC manufacturing, traditionally, for non-stacked ICs, each individual chip was tested twice [1, 14] in the following instances:

1. Wafer sort: Since IC packaging is costly [17], in this stage, the bare chip is tested, to avoid packaging of faulty chips. The chips which appear to be fault free during wafer sort are termed Known Good Dies (KGDs).
2. Package test: KGDs are packaged, and the test is applied to the complete packaged IC.

For non-stacked ICs the same test schedule is applied to both the bare chip and the packaged chip. However, for a 3D TSV-SIC the package test includes the test schedules for all the chips forming the stack after each chip has been tested in wafer sort. As will be illustrated in this paper, applying the optimized test schedule used for the individual chips during wafer sort to the stack of chips during package test for a 3D TSV-SIC may lead to sub-optimal test application time (TAT). Here, TAT is defined as the sum of the testing times for wafer

sort and package tests. It should be noted that TAT is a major part of the overall test cost [1]. Hence, it is important to schedule the tests for 3D TSV-SIC so that TAT is minimized, which is addressed in this paper.

Much work has addressed test scheduling for non-stacked chips with the objective of minimizing TAT [1, 14, 19]. For core-based systems where each core is to be tested, the most effective way of reducing TAT is to perform core tests concurrently. However, performing tests concurrently leads to higher power consumption than performing them sequentially. The test power consumption must be kept under control [1], to avoid false test positives due to voltage drop or damage due to overheating. For core-based systems, Chou et al. [1] proposed a method to schedule tests in sessions while taking resource conflicts and power consumption into account. A session is defined as a group of tests that start simultaneously and no other tests are initiated until all tests of the session are finished. The concept of sessions simplify test scheduling under power constraints, because once the tests have been allocated to sessions, that are within the power limit, the schedule is found by processing the sessions in a sequence. Muresan et al. [14] suggested an algorithm for the same problem as in [1]. While the studies in [1, 14, 19] address test scheduling for non-stacked chips under power constraints, no work has yet addressed test scheduling for 3D TSV-SICs under power constraints, which is the topic of this paper. In particular, we define a cost model that shows the benefit of testing individual bare chips at wafer sort, followed by testing the complete 3D TSV-SIC at package test, as compared to testing the complete stack directly before and after packaging or with intermediate testing instances. As the cost model shows that testing should be performed both for each individual bare chip and for the complete stacked 3D TSV-SIC, there are consequences for the test access infrastructure and the test schedule. The test access infrastructure must allow access to the cores of chip both through the wafer probe in wafer sort and through the TSVs in the package test. While there are a few different approaches to 3D TSV-SIC manufacturing and packaging, typically a chip in the middle of a stack has no wire-bonds to package pins, but are accessed through the top or bottom chip in the stack and the TSVs of chip layers in-between. In this paper, we consider a 3D TSV-SIC design with layers of core-based chips, where each core is accessible from package pins through a JTAG architecture, with a JTAG test access port (TAP) on each chip layer to enable wafer sort.

As mentioned above, the fact that tests are to be performed at both wafer sort and package test affects test scheduling. In contrast to the traditional test of non-stacked chips, where wafer sort and package test could be the same, the package test of a 3D TSV-SIC includes testing all cores of all chips in the stack, along with the TSVs. Performing tests on all cores concurrently would consume a lot of power and risk false positives from voltage-drop and damage due to over-heating. On the other hand, performing the same test schedules as in wafer sort in package test but over one chip at a time would take an unnecessarily long time, as will be shown in this paper. Therefore, we propose a power constrained test scheduling approach for 3D TSV-SICs.

This paper proceeds with some related work on non-stacked test scheduling, test access architecture and test cost analysis for various test flows in Section 2, followed by background on the manufacturing and testing process of 3D TSV-SICs in Section 3. A cost model, analyzing the various possible test schemes for 3D TSV-SICs, is described in Section 4 and the test architecture considered in this paper is discussed in Section 5. The test scheduling problem is motivated in Section 6 leading to an approach in Section 7. The experimental results are in Section 8 and the conclusions are in Section 9.

2 Related Work

In this section we first discuss previous works on scheduling tests in sessions under power constraints for non-stacked chips, followed by a brief discussion on chip architecture and concluding by discussing an economic test flow.

The problem of test scheduling for non-stacked chips has been addressed previously [1, 14, 19]. In [19], Zorian has discussed power constrained scheduling of tests for built-in-self-tested (BISTed) cores in a chip using sessions. In [1], Chou et al. proposes a solution for the same problem but also considers constraints.

Muresan et al., in [14], has developed an algorithm to schedule tests in sessions, while reducing TAT for non-stacked chips under power constraints. The algorithm is described as follows:

- All core tests are sorted in descending order of their test times.
- The longest test is considered first, which forms the first session.
- While descending through the list of sorted core tests, each test is compared for power compatibility with the existing sessions.
- The test is included in the first (longest) session which is compatible in terms of power. If no prior

power compatible sessions exist, the test forms a new session.
- Each test is considered in descending order of their length until all tests of the list are exhausted.

The test scheduling approaches discussed in [1, 14, 19] perform well for 2D ICs, where the same schedule is applied both at wafer sort and package test. However, no work has yet been visible for scheduling tests under power constraints for 3D TSV-SICs. In case of 3D TSV-SICs, the package test involves testing of all the chips in the stack together. Therefore, tests scheduled for individual chips during wafer sort using the algorithm in [1, 14] do not perform well during package test, as will be seen in Section 6.

Various test architectures have been proposed in [3–5, 11]. Iyengar et al. [3–5] proposes methods of optimizing test wrapper and test access mechanism (TAM) for multiple core based non-stacked ICs, based on the rectangular packing problem [2]. In [3, 5], Iyengar et al. use their proposed methods to reduce tester data volume. Although the applicability of those approaches are limited to multiple cores on a single non-stacked chip, they form the basis of the test architectures defined for 3D TSV-SICs. In [13], Marinissen et al. have proposed a test access architecture for 3D TSV-SICs which is based on [12]. The test architecture is based on IEEE 1149.1, commonly known as JTAG, and supports both wafer sort and package testing, using a modular approach. It is described by using an IEEE 1149.1 in the bottom chip and is scalable for any number of chips forming the stack. The test access architecture proposed in [13] uses few TSVs and supports both pre-bond and post-bond testing, but scheduling of tests has not been considered. In this paper we propose a test architecture based on JTAG, which is described in Section 5.

In [16], a test cost analysis has been performed for 3D TSV-SICs, with upto six chips in a packaged stack. The yield of each die is assumed to be within a range of 60% to 90%, while the stack yield and TSV yield are assumed to be constant, 93% and 99%, throughout the paper. Taouil et al., in [16] compares the test flow of non-stacked ICs with 3D TSV-SICs. Case studies show that including wafer sort in the test flow results in reduction of the overall chip cost. In addition, it is concluded that fewer number of tests may not reduce the overall 3D TSV-SIC cost and the test cost and waste also depends on the test yields of the intermediate partial stacks and the final stack before and after packaging. In Section 4, we perform a test cost analysis for 3D TSV-SICs to arrive at an economic test scheme for 3D TSV-SICs. Different from [16] who makes their analysis for chip yield values in the range of 60%–90%, a constant stacked yield of 93%, and a constant TSV yield of 99%, our analysis includes any yield values for chips and TSVs and reasonable yield values (> 0.9) of the stacked chips. Also, our cost analysis shows that a test flow testing partial stacks leads to higher test cost. We perform the analysis over a range of yield values shown in Table 1. However, we describe the test schemes using a specific set of values, thus arriving at the conclusion to perform the package test on the stacked and packaged KGDs obtained after wafer sort. Finally in Section 7 we propose an algorithm to schedule tests for reducing TAT which supports the test scheme.

Table 1 Test cost for individual parts of 3D TSV-SIC

Component	Test time	Yield
Chip 1	3000 t.u.	0.9
Chip 2	20000 t.u.	0.92
Chip 3	100000 t.u.	0.87
TSV	500 t.u.	0.95
Package	500 t.u.	0.95
Yield per stacking step	0.95	

3 Background

To continue according to Moore's law, having more functionality into smaller form factors, reducing power dissipation and cost while enhancing the performance, integrated circuits (ICs) with multiple chips (dies), called 3D TSV-SICs have been developed. Earlier versions of high integration in non-stacked multiple chip ICs include:

- Printed Circuit Boards (PCBs), with multiple ICs on the same board
- System-on-Chip (SoC), with multiple cores in a chip
- Multi-Chip-Package (MCP), where multiple dies are integrated in a single package [10].

Eventually multi-chip ICs were stacked vertically, but not bonded with TSV interconnects, or elevators, which include:

- System-in-Package (SiP), where dies are vertically stacked within a package, interconnected by wire-bonds to the substrate
- Package-on-Package (PoP), where multiple chips are vertically stacked

Although 3D TSV-SICs has its advantages in terms of performance or power requirements, the manufacturing process introduces new challenges in terms of

achieving high yield, testing and power constraints [7, 10].

Since 3D TSV-SICs are unlike any other ICs due to the presence of TSVs, also known as elevators, among the layers of the stack, the manufacturing process for 3D TSV-SICs is different. 3D TSV-SICs can be obtained by three stacking processes, viz, Die-to-Die (D2D), Wafer-to-Wafer (W2W) and Die-to-Wafer (D2W). In W2W stacking, complete wafers are stacked over one another, resulting in exponentially decreasing yields with increasing number of layers in the stack [15]. Therefore, this paper considers D2W and D2D stacking [15].

While stacking, the orientation of the stacked chips has to be considered. There are three possible variations in this regard: face-to-face, back-to-back and face-to-back. In this context, the face of a chip is the side of the transistors and the metal interconnect layers and the back is the silicon substrate layer. Among the three possibilities, only face-to-back bonding is scalable to stacks of more than two chips [10]. Hence, only face-to-back bonding is applicable for this paper.

The test flow model as discussed in [10] is shown in Fig. 1. A traditional non-stacked chip is tested twice at the two levels (Fig. 1a), viz. (i) wafer sort and (ii) package test. Wafer sort is motivated by the fact that packaging the faulty products is more expensive than the test itself. By testing, unnecessary packing of faulty chips is avoided. For non-stacked chips, the only possible introduction of faults after wafer sort might occur while packaging the same IC. Therefore, the test performed at wafer sort is repeated at the package test.

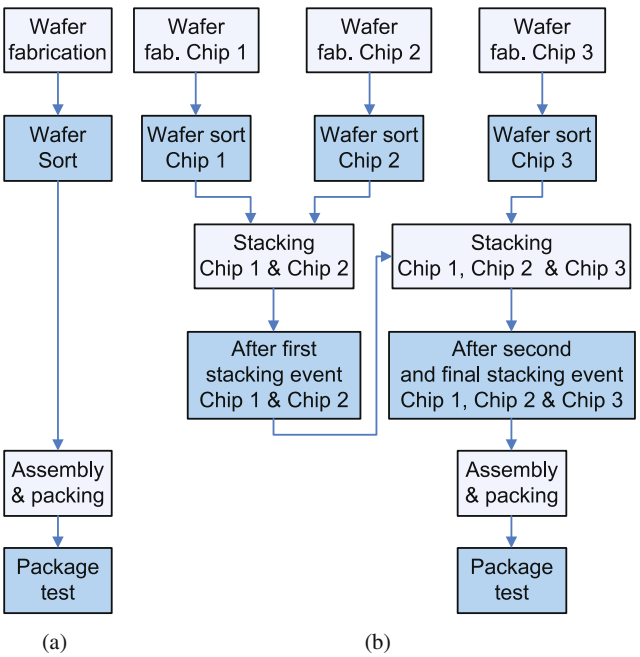


Fig. 1 **a** 2D test flow [10], **b** 3D test flow (for 3D TSV-SICs) [10]

In case of 3D TSV-SIC, as seen in Fig. 1b, there are four steps in the stacking process when faults can be introduced to any individual chip of the stack: (i) die fabrication, (ii) when the bottom of the chip is bonded to the stack, (iii) when another chip is bonded to the top of the chip, (iv) packaging. Based on these steps, several test runs can be considered, one for each step that can introduce faults. For a three-chip stack, these test runs can be referred to as wafer sort, test after the first stacking event (for the two chips that are first stacked together), test after the second stacking event and package test, as shown in Fig. 1b. It should be noted that testing after a stacking event or package test includes testing the TSVs.

Chip-specific test schedules that are optimized for wafer sort do not consider testing of other chips in the stack. Similarly, test schedules that are optimized for the package test are not necessarily optimal for wafer sort. Thus, it can be seen that a complete view of test scheduling from wafer sort through to package test is required in order to arrive at a minimal TAT.

4 Cost Model

A major part of the chip cost accounts for the testing of the chips [10]. In the example of Fig. 1b, it can be seen that stacking three chips to make a 3D TSV-SIC can lead to testing the same chip four times. That is twice the test cost per chip, as compared to traditional non-stacked testing.

To arrive at an efficient low-cost test scheme, we develop a cost model considering the test flow graph in Marinissen and Zorian [10]. In [16], Taouil et al. defined and employed a detailed cost model for testing 3D TSV-SICs to conclude that inclusion of wafer sort in the test flow results in reduction of the overall cost. Furthermore, they observed benefits from TSV tests in partial stacks. In this paper we develop a test scheduling approach in Section 7, which applies to the most economic test flow scheme among the three schemes described below:

- A: Wafer sort test followed by testing the TSVs after each stacking event, and package test.
  In Fig. 1b, the second level of events from the top implemented; in the fourth level, only TSV tests are performed and then the lowermost level, package test is performed.
- B: Complete stack test after all stacking events, and package test.

Only the rightmost event of the fourth level from the top, and the package test, in Fig. 1b are implemented.

- C: Wafer sort followed by partial stack tests after each stacking event, and package test.
  Here all the events in the second, fourth and sixth levels from the top in Fig. 1b are performed. All chips of the stack are tested in each event, and the TSVs are tested in all stages, but the topmost one.

In contrast to [16], we assume that the tests are perfect in the sense that all faults are correctly detected. The parameters of our cost model are: The manufacturing yield Y_C and the test time T_C for each chip C. Similarly, the test yield and test times for TSV testing are Y_{TSV} and T_{TSV} respectively. We assume each stacking step and packaging may damage the chip, with the yield Y_D.

We illustrate our cost model with an example of a three chip 3D TSV-SIC, where the test times and yield values for each component are shown in Table 1. In this table we arrive at 100 good packaged chips applying the three mentioned schemes A, B and C. The total time spent in testing all the components that result in 100 good packaged chips is calculated. The cost is related to the required number of chips to arrive at 100 good packaged chips.

For each step in the test scheme (wafer sort, after first stacking event, after second stacking event, package test) we calculate the test time which depends on the number of stacked components to test. Furthermore, we calculate the time spent testing faulty components and components that end in a faulty stack, which we term as waste.

The number of components to be tested in a given step is calculated using Eq. 1.

$$Quantity = \frac{\text{Desired output quantity}}{\prod Y_{\text{untested components and steps}}} \tag{1}$$

Equation 1 expresses that in order to manufacture 100 good packaged 3D TSV-SICs, it may be necessary to test more than 100 components, due to yield loss. This yield corresponds to the components that are yet untested and the yield of subsequent stacking steps.

The time taken to test the given number of components is as Eq. 2:

$$TestTime = \sum_{\text{tested components}} (Quantity \cdot T_C) \tag{2}$$

For a given step in the test scheme, where a number of components are tested for the first time, the time spent in testing faulty components or components that end up in a faulty stack is given by Eq. 3:

$$\begin{aligned} Waste = {} & Quantity \cdot \\ & \cdot \left(\sum \text{Test time for stacked components}\right) \cdot \\ & \cdot \left(1 - \prod Y_{\text{components tested for the first time}}\right) \end{aligned} \tag{3}$$

With Eqs. 1, 2 and 3, it is possible to express the following: To get N good packaged chips, where the chip design has yield Y_C and takes T_C time units, while the package has yield Y_p and takes T_p time units, it may be necessary to test $\frac{N}{Y_C \cdot Y_p}$ chips in wafer sort and $\frac{N}{Y_p}$ packaged chips in package test. Thus, wafer sort will take $T_C \cdot \frac{N}{Y_C \cdot Y_p}$ time units and package test will take $(T_p + T_C) \cdot \frac{N}{Y_p}$ time units. The waste (time spent testing faulty chips or faulty packages) amounts to $(1 - Y_C) \cdot \frac{N}{Y_C \cdot Y_p} \cdot T_C$ and $(1 - Y_p) \cdot \frac{N}{Y_p} \cdot (2 \cdot T_C + T_p)$ for wafer sort and package tests respectively. It should be noted that T_C is counted twice in the calculation of the waste from the package test, because at that point, the chips are tested for the second time.

In Table 1, from the left, the first column lists the components to be tested. It should be noted that there are two instances of TSVs in the stack, between Chip 1 and Chip 2, and between Chip 2 and Chip 3. The second and third columns show the test time required and the respective yield values for each component. The final row of Table 1 shows the yield of a stacking step. Here, we use 0.95 yield to express that when Chip 2 is stacked upon Chip 1, five out of a hundred partial stacks are damaged. Similarly when Chip 3 is stacked on top of Chip 1 and Chip 2. Also for packaging, it is assumed that five out of a hundred stacks are damaged.

The example of the cost model is carried through in Table 2. The four testing events, *viz.*, wafer sort, after first stacking event, after second stacking event and package test are analyzed. In a group of four subcolumns for each testing event, listed are the components that are tested (chips and/or TSVs), the total number of components tested under quantity, the total time taken for the testing event as test time and waste is the time spent on testing products that do not pass the testing event.

For test scheme A, 133 Chip 1, 131 Chip 2 and 138 Chip 3 are tested in wafer sort to obtain KGDs. The sum 402 is given in column 3. Wafer sort takes in total 16819000 time units as in column 4. Because of the yield of the three chips, 2043500 time units are wasted on faulty chips as detailed in column 5. Wafer sort results in 120 good chips of each type. That means that 120 partial stacks of Chip 1 and Chip 2 are manufactured and testing the TSVs in the partial stacks takes 60000

Table 2 Test schemes

Scheme	Components	Quantity	Test time	Waste	Components	Quantity	Test time	Waste
	Wafer sort				**After first stacking event**			
A	Chips 1,2&3	402	16819000	2043500	TSV layer 1-2	120	60000	143850
B	n/a	n/a	n/a	n/a	n/a	n/a	n/a	n/a
C	Chips 1,2&3	402	16819000	2043500	Chips 1&2, TSV 1-2	120	2820000	412950
	After second stacking event				**Package test**			
A	TSV layer 2-3	113	56500	703284	All	106	13197000	1317050
B	Chips 1,2&3, TSV 1-2,2-3	166	20584000	8506590	All	106	13197000	1317050
C	Chips 2&3, TSV 2-3	113	13616500	2155334	All	106	13197000	2074950
Total		100	30132500	4207684				
		100	33864000	9823640				
		100	46452500	6686734				

time units. From this test it is revealed that 143850 time units are wasted on testing components that will never be a part of a 3D TSV-SIC. The process goes on in test scheme A where 113 good partial stacks, consisting of Chip 1 and Chip 2, are combined with Chip 3 and another TSV test is applied, after the second stacking event. Before the package test, there are 106 stacks, which because of the yield of packaging and the risk of damage end up as 100 good 3D TSV-SICs. Similar observations can be made about test scheme B and test scheme C. In particular, test scheme B, which tests the chips of the stack for the first time, after all the chips have been stacked, requires 166 chips of each type to ensure that there will be 100 good TSV-SICs. Our conclusion from the cost model is that test scheme A has the lowest cost in terms of test time and the number of required chips. Furthermore, test scheme A spends the least amount of time on testing components that will not be used in a good 3D TSV-SIC.

Similar applications of the cost model as in the example above has been repeated for various yield and test time values. We have seen that the observations made regarding the benefits of test scheme A hold for reasonable yield values (> 0.9).

Hence, from the above analysis of various test cost schemes, it can be concluded that scheme A is the most economic in terms of test time and waste. Therefore, for the rest of the paper we will assume scheme A, i.e., two steps in the test flow: wafer sort (each individual chip), followed by packaging (complete packaged stack).

5 Test Architecture

In this paper we consider that each chip is equipped with a BIST engine that can be accessed through IEEE 1149.1, commonly known as JTAG, as shown in Fig. 2. Besides the cores of two chips, Chip 1 and Chip 2, Fig. 2 shows how the cores are accessed through a JTAG TAP on each chip. The use of JTAG for accessing the cores of the individual chips has three main benefits.

1. It adheres to an existing standard.
2. JTAG can be used for test access both in wafer sort and package test of the 3D TSV-SIC. Typically, after stacking, test access is only possible through the bottommost chip. Therefore, dedicated test infrastructure TSVs are required to access cores of chips that are not at the bottommost chip. When testing the chips individually, as in wafer sort, the JTAG TAP enables test access. This corresponds to the interfaces used for wafer sort as shown in Fig. 2. Figure 2 shows how the JTAG TAPs of different chips in the stack can be connected in series to enable test access to all chips in the stack (interface for package test).

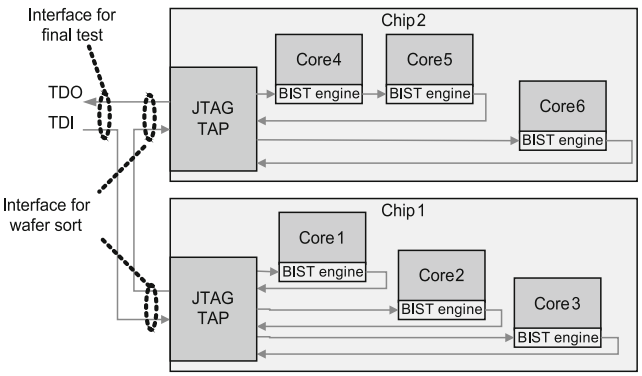


Fig. 2 Test architecture of 3D TSV-SIC with JTAG

3. Independently, on each core or each chip of the stack, upto five TSVs are required for dedicated test infrastructure, which correspond to the five terminals of JTAG TAP, namely Test Data Input (TDI), Test Data Output (TDO), Test Mode Select (TMS), Test Clock (TCK) and an optional Test Reset (TRST). It should be noted that in Fig. 2 only the TDI and TDO pins are shown.

The BIST engines are connected to the JTAG TAP as Test Data Registers (TDRs). Only one TDR can be accessed at a time. Figure 2 illustrates the TDRs as loops that start from a JTAG TAP, proceeds through one or more BIST engines and returns back to the JTAG TAP. Thus, if tests for more than one core of a chip are to run concurrently, in a session, these cores are connected in series on the JTAG interface, forming a single TDR. It should be noted that this enforces the session concept that was introduced in Section 1. In this paper we assume, for a single chip, only cores that are in the same TDR can be tested concurrently. Furthermore, if two cores are to be tested in sequence, in different sessions, they cannot be connected in the same TDR. On the other hand, in practice, a session of tests (corresponding to a TDR) from a chip can be performed concurrently with a session of tests from another chip by selecting the TDRs in the TAPs of the two chips. While testing concurrently can lead to power dissipation above the safe power limit $P_{\max}$, scheduling must take power dissipation into account.

The test process is conducted as follows. The BIST engine is started by using JTAG to shift in configuration data (possibly including a seed for an LFSR) into a register within the BIST engine. After the completion time of the core test, which is assumed to be known for each core, JTAG is used to shift out the test response in the form of a signature from a register within the BIST engine. Typically the time required for shifting in configuration data and for shifting out signatures is negligibly small compared to the time the BIST engine is running to conduct the test. Therefore, only the BIST engine test time is considered in this paper.

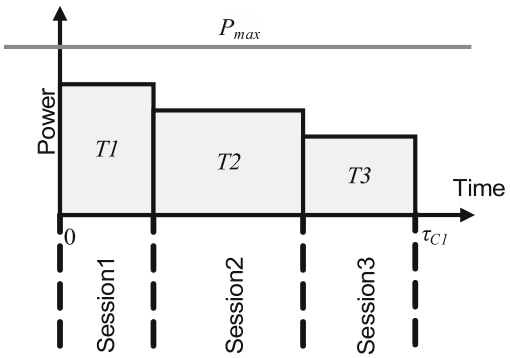
The TSV interconnect between two chips may be tested by using a special JTAG TDR called the boundary scan register, which connects all input/output pins and TSVs with special scan cells forming a shift register. The scan cells are transparent when the 3D TSV-SIC is in functional mode, but it test mode, the scan cells are control points and observation points. Boundary scan registers are implemented on both the chips and both are used in TSV interconnect test. Test stimuli are applied on out-going TSVs and test responses are captured on in-coming TSVs. Since the boundary scan register is a separate TDR, testing of TSVs cannot be performed concurrently with any other test.

It should be noted that the TSV interconnect tests will contribute with a constant term to TAT and could not be scheduled with any other core tests, due to the JTAG. Therefore, TSV interconnect tests will not be regarded when addressing test scheduling in the remainder of the paper.

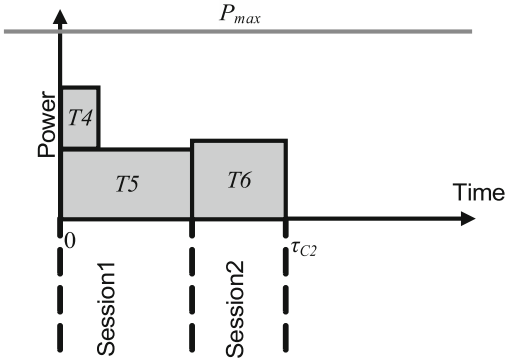
6 Problem Analysis

From Section 4, we concluded from the test cost analysis that, for a wide range of yield values it is more economic to have only the wafer sort and the package test, in comparison to the other schemes which either included intermediate steps, or did not include wafer sort. In other words, testing individual chips before stacking and hence stacking only KGDs, followed by the package test of the complete stack requires the least test time and the time spent on faulty parts is also lowest.

Figure 2 shows a chip, Chip 1, with three cores where each core is tested by its BIST engine. Two parameters are associated with each test: test time (τ) and power consumption (P). The test controller, which in this case accesses the cores through JTAG (as in Section 5), determines when the test for each core is initiated. Figure 3a shows a test schedule for the tests (light shade) of the three cores of Chip 1 (Fig. 2), which have been scheduled as per [14] where the TAT is minimized and the power consumption at any moment is less than the maximal allowed power consumption $P_{\max}$, which is indicated by a horizontal line. The test schedules are illustrated by a rectangle corresponding to each core test, where the height represents power consumption, while the time taken by the test is represented by the width. The horizontal axis shows the time taken to perform the tests, and the vertical axis marks the power consumption. Two types of constraints are considered for the test schedule. The first constraint type is a resource constraint, as has been discussed in [14], which expresses that tests of cores which share some common resource cannot run at the same time. The second constraint type is a constraint regarding the maximum power consumption, $P_{\max}$, which cannot be exceeded. The test schedule contains three sessions: Session1, Session2 and Session3, as marked in Fig. 3a. Considering only Chip 1, this chip is a single-chip IC, so the same test schedule is applied at wafer sort and package test. $TAT = \tau_{C1} + \tau_{C1}$, as the same test schedule is run twice.



(a) Chip 1



(b) Chip 2

Fig. 3 Wafer sort schedules

Figure 4 shows a 3D TSV-SIC where Chip 2 (from Fig. 2) is stacked on top of Chip 1. The testing of the 3D TSV-SIC requires wafer sort of Chip 1 and Chip 2 and a package test of the stacked chip including tests for the cores in Chip 1 and Chip 2 as well as tests for the TSVs. The test durations and power consumption values for each core tests are provided in Table 3. The power constraint value is $P_{\max} = 20$ units.

Prior to stacking chips into a 3D TSV-SIC, each chip can be considered as individual non-stacked chips and the methods in [1, 14] apply for generating the wafer sort schedules. Figure 3 shows examples of the wafer sort schedules for the two chips, Chip 1 and Chip 2,

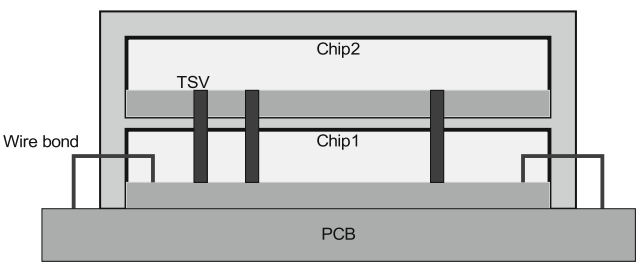


Fig. 4 3D TSV-SIC of Chip 1 and Chip 2

Table 3 Test time and power consumption for core tests in Chip 1 and Chip 2

Chip	Core test	Test time (τ)	Power consumption (P)
Chip 1	$T1$	5	15
Chip 1	$T2$	8	12
Chip 1	$T3$	6	9
Chip 2	$T4$	2	7
Chip 2	$T5$	7	8
Chip 2	$T6$	5	9

from Table 3. The test schedule for Chip 1 contains three sessions (Session1, Session2 and Session3) and the test schedule for Chip 2 contains two sessions (Session4 and Session5) as shown in the figure. The test time for the schedules as obtained by [14] are τ_{C1} and τ_{C2} for Chip 1 and Chip 2, respectively.

Once the chips have been stacked, package test will test the chips again. We define three different approaches for test scheduling depending on the available knowledge from wafer sort. In this paper, the three approaches are called Serial Processing (SP), Partial Overlapping (PO) and ReScheduling (RS).

In case the only knowledge of the wafer sort schedules consist of the test time for the schedules and the fact that the wafer sort schedules are within the power constraint, the limited knowledge available restricts the test schedules that are possible. In this case the package test is scheduled by Serial Processing, which is illustrated in Fig. 5. With Serial Processing we mean that the test schedules of individual chips are run serially during package test. It should be noted that no tests from different chips are performed concurrently, because otherwise we would risk exceeding the power limit from lack of information of the actual power consumption. For Serial Processing, the time taken to run the package test schedule is equal to the sum of the time taken to test the individual chips. For the schedule in Fig. 5, $TAT_{SP} = \tau_{C1} + \tau_{C1} + \tau_{C2} + \tau_{C2}$.

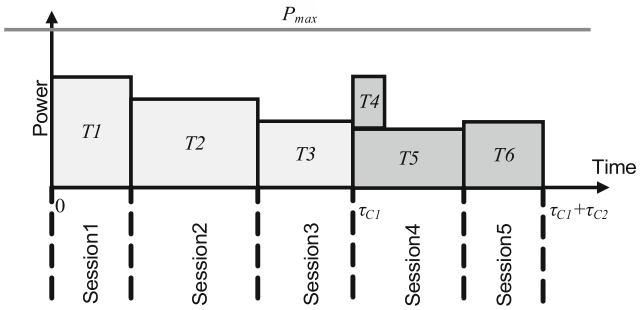


Fig. 5 Serial processing

If the maximum power reached by individual sessions and the test time for the sessions are known, package test scheduling by Partial Overlapping is possible. In Partial Overlapping, we utilize the knowledge of the test sessions to determine the power compatible test sessions of different chips that can be performed concurrently without exceeding the power constraint. Figure 6 shows the Partial Overlapping test schedule. In the schedule for package test, test $T3$ of Chip 1 and test $T6$ of Chip 2 are performed concurrently because they are power compatible. The wafer sort schedule of the chips remain unchanged, but there is a reduction in the TAT equal to the length of test $T6$.

When full knowledge is available concerning individual tests and sessions of the wafer sort schedules, ReScheduling of the existing schedules can be performed. In the ReScheduling approach, knowledge of the wafer sort schedules is utilized to create a package test schedule to reduce test time. ReScheduling may cause changes to the wafer sort schedules. In this context, changing the wafer sort schedule means to split a session and replace it with two new sessions. This means that the corresponding chips are redesigned so that a TDR is replaced by two TDRs in the test architecture (see Section 5). The benefit of splitting a session is that the two new sessions can be scheduled concurrently with sessions of the other chip during package test, if that reduces TAT. Figure 7 depicts the result of the ReScheduling approach. In the original wafer sort schedule (Fig. 5), Session 4 consisted of tests $T4$ and $T5$. In the wafer sort schedule, after rescheduling (Fig. 7b), test $T4$ is performed in sequence with the other tests, while test $T5$ is performed together with test $T2$. This results in a reduction of test time for the package test equal to the duration of test $T5$. ReScheduling results in splitting Session4 and renumbering the sessions, as shown in Fig. 7a, Session4 is test $T5$, Session5 is test $T6$ and Session6 is test $T4$. But because of the splitting of the original Session4, there is an increase in test time for Chip 2 wafer sort from τ_{C2} to $\tau_{C2} + \tau_{T4}$. The

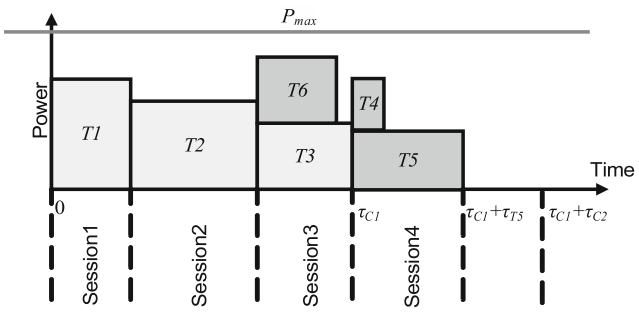


Fig. 6 Partial overlap

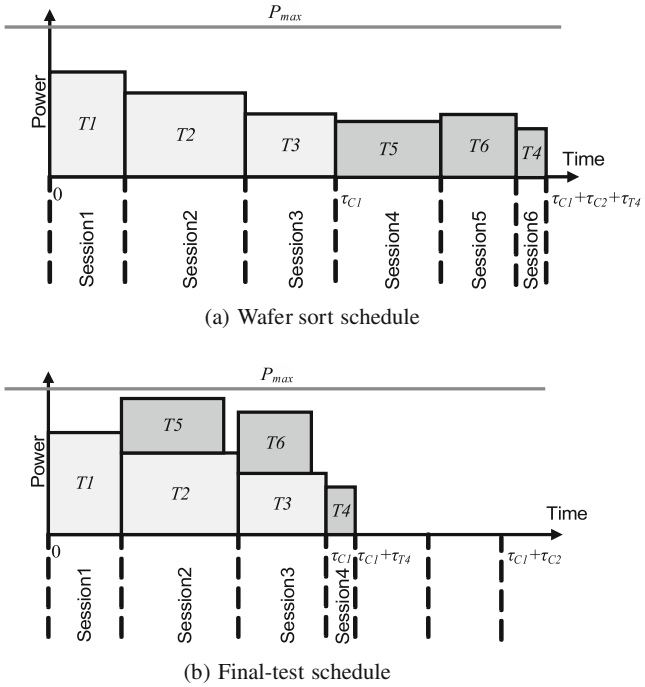


Fig. 7 ReScheduling

increase is equal to the duration of test $T4$, which is now performed serially with test $T5$. Compared to SP, the reduction in TAT is equal to the sum of the durations of tests $T5$ and $T6$, minus the duration of test $T4$. From the above example, it can be seen that ReScheduling required a lower TAT as compared to Serial Processing and Partial Overlapping, as is shown in Fig. 7. However, in contrast to Serial Processing and Partial Overlapping, ReScheduling can lead to an increase in the routing of JTAG interconnect lines, as a result of splitting sessions, which means additional TDRs (see Section 5). Therefore, in the following section we describe an approach for ReScheduling, while taking into account the additional routing that results from splitting sessions.

7 Proposed Approaches

In this section we first detail the two approaches, Partial Overlapping (PO), and ReScheduling (RS), considering a stack of two chips. Subsequently, we explain how to generalize the approaches to stacks with > 2 chips. Finally, we discuss the complexity of the approaches.

The PO approach considers only the knowledge of individual sessions and can be considered as a special case of RS. The PO approach can be implemented with the same algorithm as RS by considering each session to contain only a single test, as an abstraction from the

actual number of tests in the session, which would in the case of PO be unknown.

7.1 ReScheduling for a 2-Chip Stack

The following describes the RS approach which consists of two phases. Before the first phase, the wafer sort for each chip is generated by the heuristic from [14]. Each session of the wafer sort schedules is given a unique number.

7.1.1 Phase 1

In Phase 1, a table is created such as Table 4, where the columns are described by the sessions of Chip 1 and the rows are described by the sessions of Chip 2. The entries in the table describe how much reduction in TAT that is possible by rescheduling a session from Chip 1 (a specific column) with a session from Chip 2 (a specific row). The idea of creating the table is that a schedule can be defined by selecting a unique entry in each column and/or each row. These selected entries correspond to sessions that are rescheduled in the defined schedule. Sessions for which no entry was selected will be added to the defined schedule unchanged. Table 4 will be further described later in this section, but for Phase 1, the aim is to create that table.

Figure 8 shows a 14 step process for Phase 1 of the RS approach. The key idea is to group the tests of two wafer sort sessions from different chips in two sessions for package test such that the long tests are grouped together and the short tests are grouped together. This way, there will be one long test session and one short test session, instead of the previous two long sessions. In step 2 of Fig. 8, we consider two sessions, $S1$ and $S2$, from the original wafer sort schedules of two different chips, Chip 1 and Chip 2, respectively. In step 3 and step 4, the tests of $S1$ and $S2$ are arranged in descending order of length in a list called M. A session for package test, Sa, is produced in step 5, along with two sessions $Sa1$ and $Sa2$ which will eventually replace the existing wafer sort sessions $S1$ and $S2$. Starting from the first test in M, i.e. the test with the longest test time, tests are moved from M to Sa, as shown in step 6 and step 7. In step 7 there is a check to see if the total power of Sa is within the power constraint. Step 6 and step 7 are iterated as long as the power constraint is met. As soon as $P_{\max}$ is exceeded as a result of moving a test from M to Sa, that test is moved back to M (step 8).

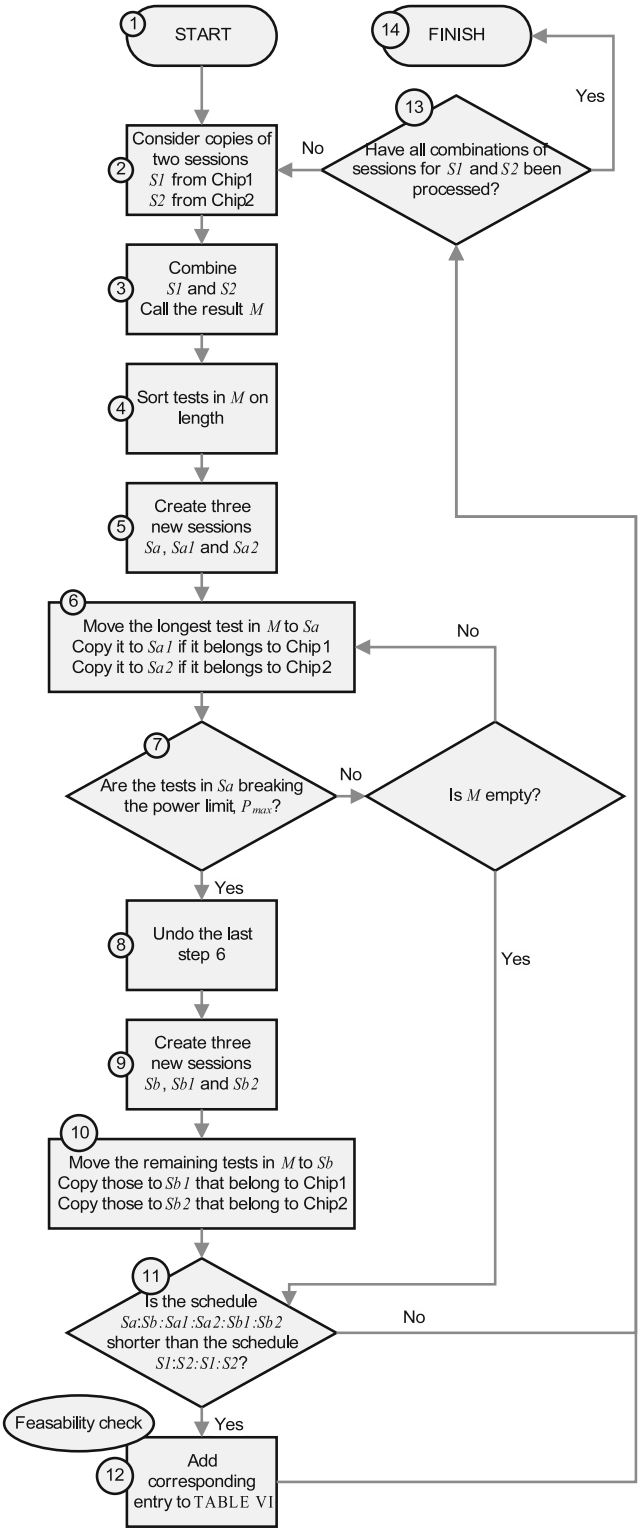


Fig. 8 Flow diagram

Table 4 Maximum possible time reduction of sessions

		Chip 1		
		Session 1	Session 2	Session 3
Chip 2	Session 4	0	(3)	2
	Session 5	0	0	(5)

It should be noted that if all tests are be moved from M to Sa without exceeding the power limit, the process continues directly to step 11 without any action in steps 8–10. In step 9, a new final-test session, Sb, is created with the remaining tests of list M, which is shown in step 9 and step 10. Final-test sessions Sa and Sb are disjoint sets of tests that together contain all the tests from M. Through step 6 and step 10, the wafer sort sessions $S1$ and $S2$ are replaced with test sessions ($Sa1$, $Sa2$, $Sb1$ and $Sb2$) according to how the tests were allocated in Sa and Sb. It should be noted that some of the sessions Sb, $Sa2$, $Sb1$ and $Sb2$ may become empty of tests and can be disregarded. The modified TAT is calculated in step 11. If the new test schedule (wafer sort and package test) is shorter than the test schedule for SP, the value is included in Table 4 as in step 12, as the entry for session $S1$ and session $S1$. Otherwise, if there is no reduction the value is set to be zero. While adding values to Table 4, extra considerations are required if both chips are of the same design. Then, if a session is split, it should affect both chips, because sessions correspond directly to JTAG TDRs which are specific to the chip design. If the chips are of the same design and the session pair for rescheduling indicates a schedule that requires two different designs, rescheduling that session pair is infeasible and the entry in Table 4 is set to zero.

The process described above is repeated for all possible combinations of two sessions from the wafer sort schedules of the two chips, as is shown in step 13 of Fig. 8.

7.1.2 Phase 2

In Phase 2, a schedule is defined with the maximum reduction in TAT compared to SP by considering the table that is created in Phase 1. Table 4 shows the possible reduction in TAT as a result of rescheduling a session of Chip 1, as denoted by the column number, with a session of Chip 2 of the corresponding row number. Given a table such as Table 4, a schedule is generated by rescheduling each session of one chip with different sessions of the other chip, such that every session is considered only once. The sessions that are not rescheduled are added to the final schedule without any modification. For example, in Table 4, two session pairs are selected, namely Session 2 with Session 4 and Session 3 with Session 5. Session 1 is not included in any of the selected pairs and is added to the final schedule unmodified.

A key observation regarding the table that is created in Phase 1 is that pairs of sessions can be handled independently. If combining a pair of sessions as described by step 2 to step 12 leads to a reduction in TAT compared to the test schedule in SP, a new test schedule can be constructed by combining several independent session pairs. The total reduction in TAT can be summed up from the reductions in test time when all session pairs have been considered, while each session has been taken into account only once.

The objective is to find the combination of rescheduled session pairs, which would give the minimum TAT. For example, with respect to Fig. 5, considering Session 2 from Chip 1 and Session 4 from Chip 2 results in a reduction of 3 time units on rescheduling, compared to the time required to perform the original Session 2 of Chip 1 and Session 4 of Chip 2 sequentially, as in SP. In case of PO, where no sessions are split, the values in the table would either be zero (when the sessions are not power compatible), or equal to the length of the smaller session. For example, it was not possible to reduce TAT by combining Session 1 with Session 4 as marked by 0 in Table 4. The test schedule and the total reduction in TAT are obtained by rescheduling each session of Chip 1, with sessions of Chip 2. As discussed before, tests from Session 2 of Chip 1 and tests from Session 4 of Chip 2 upon rescheduling, result in a reduction of 3 time units (marked with (3) in Table 4). Similarly, rescheduling Session 5 of Chip 2 with Session 3 of Chip 1 give a reduction of 5 time units. The sessions that result from the marked session pairs are included in the final-test schedule with the summed total of test time reduction adding up to $3 + 5 = 8$ time units. The test time of the rescheduled session pairs are added to the remaining sessions to give TAT. Thus, the final-test schedule has Session 1 in series with the combination of Session 2 with Session 4 and Session 3 with Session 5. The result is $TAT_{RS} = 54$ time units. For the SP approach, $TAT_{SP} = 62$, corresponding to time units in both wafer sort and package test. From this it can be seen that RS results in a reduction in TAT by 8 time units, as was predicted by selecting entries worth 5 and 3 time units in Table 4.

As will be shown in Section 7.3, finding the schedule with the lowest TAT is a complex task. Existing heuristics can be applied to obtain a schedule with low TAT (but not necessarily lowest) from a table from Phase 1 such as Table 4. The heuristic that has been used selects the table element with the highest value and continues to select the table element with the next highest value, while restricting the selection to columns and rows that are not corresponding to a previous selection. The heuristic ensures that only independent session pairs are selected. The process continues until all rows are exhausted. The sum of all the values corresponding to selected session pairs give the net reduction in test time that is achieved by the rescheduling the selected

session pairs. Sessions that were not joined with other sessions are added to the list of session pairs to form the schedule. The particular combination of session pairs that lead to the schedule correspond directly to the wafer sort and package test schedules for the 3D TSV-SIC. The combination of session pairs that gives the largest reduction in terms of TAT corresponds to a candidate for the schedule.

To arrive to the schedule, the heuristic is iterated K times, where K is the sum of the number of rows and columns, with different session pairs (not necessarily the element with the highest value) as starting point to produce a number of solutions that can be evaluated by the designer of the 3D TSV-SIC with regard to the acceptable amount of JTAG interconnect line routing. This results in Table 5 for the considered example. Schedule 1 is the result of combining Session2 with Session4 as well as Session3 with Session5. Schedule 2 is the result of combining Session3 with Session4.

ReScheduling of sessions resulting in a reduction of TAT can lead to a corresponding increase in the number of TDRs due to splitting of sessions, and consequently more routing of JTAG interconnect lines (see Section 5). Table 5 shows an example providing the reduction in TAT and the number of additional TDRs for five of the test schedules produced by the proposed RS approach.

7.2 Generalization to Stacks with More Than Two Chips

To perform power-constrained test scheduling for 3D TSV-SICs with more than two chips in the stack using the approach described in Section 7.1, the following generalizing step is applied.

By using the approach described in Section 7.1 for the first two chips (say Chip 1 and Chip 2), a package test schedule is defined. By abstracting from the fact that it is the final-test schedule is for two chips, it can be considered as a wafer sort schedule for a single chip, Chip 1*, that contains the cores of Chip 1 and Chip 2. The procedure is illustrated in Fig. 9. The same approach (described in Section 7.1) can again be applied to add another chip, Chip 3, to the test scheduling process. By applying the abstraction, the table created in Phase 1 of the approach from Section 7.1 will remain

Table 5 TAT reduction vs. increase in number of additional TDRs

Schedule no.	1	2	3	4	5
TAT reduction	8	2	3	5	0
Increase in TDRs	1	1	1	0	0

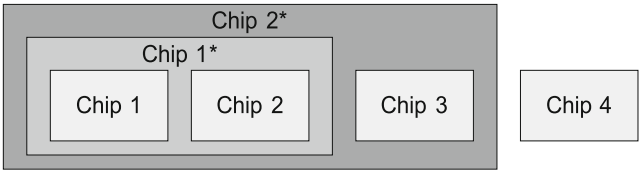


Fig. 9 Generalization process by abstracting from already processed chips

two-dimensional. The process can continue by adding chip after chip until all the chips of the stack are included, as shown in Fig. 9 for four chips. The package test schedule for the 3D TSV-SIC consists of the sessions that are defined when the last chip is processed. The wafer sort schedules for the individual chips (now without the abstraction) are found by removing all tests but the ones belonging to the considered chip from the final-test schedule (sessions that become empty while removing tests are disregarded).

7.3 Complexity of the RS Approach

In this section, we study the complexity of the proposed RS approach. The RS approach consists of two phases, Phase 1 and Phase 2, as described in Section 7.1.

In Phase 1 of the RS approach, the tests from two sessions are initially sorted, using quick-sort, by their test durations and stored in the list M. The average time complexity for quick-sort is $O(N \cdot \log N)$ for N tests.

The combination of session pairs that give the minimum TAT on rescheduling could be found by comparing of all possible combinations of session pairs in the table created in Phase 1. However, the number of possible combinations can be prohibitively large. To arrive at the complexity of exploring all possible schedules from Table 4, say Chip 1 and Chip 2 have x and y number of sessions respectively, and that $x \geq y$. Then there are x columns and y rows. The first row contains x values to choose from. Once a value is chosen, the row and column to which the value belongs are ignored and there remains $x - 1$ values to choose from in the second row. Thus, as we process each row, the number of choices decreases by one. This accounts for a factorial function that describes the number of possible sets of session pairs. But, when $y - 1$ rows have been traversed the last value can be chosen from the remaining $x - y + 1$ columns. Thus, the total number of ways, N, in which values can be selected from Table 4, with each value from a unique row or column, is given by $N = \frac{x-y+1}{y!}$. Hence, for a total number of ten sessions each in two chips, N becomes as large as 3628800. Thus, it can be seen that the problem of selecting session pairs from Table 4 to explore all possible test schedules is difficult.

Phase 2 of the RS approach involves obtaining the maximum sum of individual entries in the table created in Phase 1, taking one entry from each row or column. As discussed, the number of possible combinations of session is large, so a heuristic has been applied, which has an average time complexity of $O(T \cdot \log T)$, where T is the number of entries in the table.

Thus the overall complexity of the RS approach, assuming the number of entries in Table 4, T, in Phase 2 is considerably greater than the number of tests N, becomes $O(T \cdot \log T)$.

8 Experimental Results

To demonstrate the benefits of the proposed test scheduling approach, this section describes an experiment to compare TAT achieved by Partial Overlapping (PO) and ReScheduling (RS) with TAT achieved by the straight forward Serial Processing (SP) approach, which is used as baseline. Table 6 gives the results. In the experiment, the power constraint is met and the number of TDRs required by different test schedules is taken into account. As the RS approach yields a table such as Table 5 with several different test schedule solutions where the acceptable number of TDRs determines the test schedule selection, the experiment is performed with the test schedule that results in the largest TAT reduction (8 time units in the case of Table 5). The initial wafer sort schedules were generated by the approach in [14] and our approaches were applied for generating the package test schedule. The approach proposed in Section 7 was used to find the maximum reductions in TAT while considering the number of TDRs as the number of sessions in the example designs were in a reasonable range.

In the Table 6, the following notation is used:

- Z : ASIC Z
- L : SYSTEM L
- M : Muresan's design
- SP : Serial Processing
- PO : Partial Overlapping
- RS : ReScheduling
- $R = \frac{T_{SP} - T_{RS}}{T_{SP}}$: Reduction

The experiments are performed with the circuits ASIC Z [19], System L [6] and Muresan [14] (marked by Z, L and M respectively in Table 6 and Table 6) and these circuits were used to create 3D TSV-SICs. These designs are seen as single-die chips that have 9 [19], 14 [6] and 10 [14] cores, respectively. To enable experiments, the Muresan design and System L were scaled to have the same power limit $P_{\max}$ as ASIC Z.

Table 6 Maximum possible reduction in time with increase in number of TDRs using stacks of three or four chips

No. of chips	Design	Time taken by wafer sort				Time taken for package test				TAT Wafer sort + package test				Increase in TDRs	
		T_{SP}	T_{PO}	T_{RS}	R (%)	T_{SP}	T_{PO}	T_{RS}	R (%)	T_{SP}	T_{PO}	T_{RS}	R (%)	%	(orig)
2	ZZ	600	600	600	0	600	562	562	6.3	1200	1162	1162	3.2	0	(6)
	LL	2050	2050	2224	−8.5	2050	1412	1199	41.5	4100	3462	3343	18.5	3.8	(26)
	MM	3900	3900	3900	0	3900	3900	3300	15.4	7800	7800	7200	7.7	0	(10)
	ZL	1325	1325	1325	0	1325	958	958	27.7	2650	2283	2283	13.9	0	(16)
	LM	2975	2975	2975	0	2975	2530	2530	15.0	5950	5505	5505	7.5	0	(18)
	MZ	2250	2250	2250	0	2250	2212	2212	1.7	4500	4462	4462	0.8	0	(8)
3	ZZZ	900	900	900	0	900	862	862	4.2	1200	1160	1160	3.0	0	(9)
	LLL	3075	3075	3597	−17.1	3075	1799	1199	61.0	6150	4874	4796	22.1	5.1	(39)
	ZML	3275	3275	3275	0	3275	2724	2724	16.8	6550	5999	5999	8.4	0	(21)
4	ZMLZ	3575	3575	3575	0	3575	2896	2896	19.0	7150	6471	6471	9.5	0	(24)
	ZMLM	5225	5225	2966	0	5225	4634	4634	11.3	10450	9859	9859	5.7	0	(26)
	ZMLL	4300	4300	4474	−4.0	4300	3624	3411	20.7	8600	7924	7885	8.3	2.9	(34)

The test durations and power consumptions were scaled with the same factor.

To make a 3D TSV-SIC, a number of the three single-die chips are combined to form a stack, which is denoted by the column marked No. of chips in Table 6. The column marked Design shows the single-die chips that form the stack. The group of four columns marked Time taken by wafer sort, shows the test times for SP, PO and RS for the wafer sort schedules of the stack. The fourth column in the group shows the relative reduction in wafer sort test time of RS compared to SP. It should be noted that a negative reduction is an increase. The next group of four columns marked Time taken for package test, shows the test times for the final-test schedules generated by SP, PO and RS, and gives the relative amount of package test time reduction achieved comparing the results for RS with the result for SP. The group of columns marked TAT includes the sum of the wafer sort times and package test times. The first three columns in the group show TAT for the SP, PO and RS approaches, respectively. The relative reduction in TAT is shown in the last column where RS is compared against SP. The right-most column of Table 6, shows the relative increase in the number of TDRs that result from splitting sessions in the RS approach. The number of TDRs for the SP approach is shown in parenthesis.

From Table 6, it can be seen that RS can achieve up to 41.5% reduction in the package test schedule time in comparison to SP in the case, when two chips of System L are stacked to form the 3D TSV-SIC. This result can be explained by a high power constraint, which enables a beneficial final-test schedule where many core tests are performed concurrently. In particular for the LL design, one session was split, resulting in an additional TDR and an increase in the wafer sort schedule duration. The reduction in TAT was 18.5%, while the amount of TDRs are increased by 3.8%. For three System L chips, LLL, the package test schedule time is 61.0% and the TAT reduction was 22.1%. It should be noted that other 3D TSV-SICs consisting of two identical chips (such as the pair of ASIC Z chips, denoted by ZZ) does not lead to the same result. For the 3D TSV-SIC design made up by a pair of ASIC Z chips, TAT was reduced by 3.2% and RS and PO achieved the same result. This corresponds to a case when it is not possible to reduce TAT by splitting sessions. Three experiments led to splitting of sessions, which increased the number of TDRs, as can be seen in the right-most column of Table 6. For the other experiments, the reduction in TAT was achieved without splitting sessions and the best result achieved without splitting sessions was 13.9% reduction in TAT for design ZL.

It can be calculated from Table 6, for the sub-column $R(\%)$ under TAT, that the average reduction for the overall TAT is 9.05% for the twelve considered stacks, while the average increase in the amount of TDRs is 0.98%.

9 Conclusion

In this paper, the problem of power-constrained test scheduling for 3D Stacked Integrated Circuits (SICs) with Through-Silicon-Vias (TSVs) has been addressed for the first time. It is shown that the test planning for 3D-SICs with TSVs is different, compared to the test planning for non-stacked ICs, and requires specific test scheduling solutions. Based on a proposed test cost model, the paper proposes two test scheduling approaches, Partial Overlapping and ReScheduling that minimize test application time while taking power-constraints and the need to route JTAG and Test Data Registers (TDRs) into account. Experiments done with the two scheduling approaches and a straight forward approach (Serial Processing) with several benchmarks show up to 22% reduction in test application time and an average reduction of 9% in test application time with less than 1% average increase in the amount of TDRs over the Serial Processing scheme.

References

1. Chou RM, Saluja KK, Agrawal VD (1997) Scheduling tests for VLSI systems under power constraints. IEEE Trans Very Large Scale Integr (VLSI) Syst 5(2):175–185
2. Hopper E, Turton BCH (2001) An empirical investigation of meta-heuristic and heuristic algorithms for a 2D packing problem. Eur J Oper Res 128:34–57
3. Iyengar V, Chakrabarty K, Marinissen EJ (2002) Wrapper/TAM co-optimization, constraint-driven test scheduling, and tester data volume reduction for SOCs, no 44.3, pp 685–690
4. Iyengar V, Chakrabarty K, Marinissen EJ (2002) Test wrapper and test access mechanism co-optimization for system-on-chip. J Electron Test: Theory Appl 18:213–230
5. Iyengar V, Chakrabarty K, Marinissen EJ (2003) Test access mechanism optimization, test scheduling, and tester data volume reduction for system-on-chip. IEEE Trans Comput 52(12):1619–1632

6. Larsson E, Peng Z (2002) An integrated framework for the design and optimization of soc test solutions. J Electron Test: Theory Appl 18(4):385–400 (Special Issue on Plug-and-Play Test Automation for System-on-a-Chip)
7. Lee H-HS, Chakrabarty K (2009) Test challenges for 3D integrated circuits. IEEE design and test of computers. Special Issue on 3D IC Design and Test, pp 26–35
8. Lee Y-J, Lim SK (2008) Co-optimization of signal, power, and thermal distribution networks for 3D ICs. In: Proceedings of electrical design of advanced packaging and systems symposium, pp 163–166
9. Lewis DL, Lee H-HS (2007) A scan-island based design enabling pre-bond testability in die-stacked microprocessors. In: Proceedings of the IEEE International Test Conference (ITC), pp 1–8
10. Marinissen EJ, Zorian Y (2009) Testing 3D chips containing through-silicon vias. In: Proceedings of the IEEE International Test Conference (ITC), pp 1–11
11. Marinissen EJ, Chakrabarty K, Iyengar V (2002) A set of benchmarks for modular testing of SOCs. In: International Test Conference (ITC), no 19.1, pp 519–528
12. Marinissen EJ, Kapur R, Lousberg M, McLaurin T, Richetti M, Zorian Y (2002) On IEEE P1500s standard for embedded core test. J Electron Test: Theory Appl 18:365–383
13. Marinissen EJ, Verbree J, Konijnenburg M (2010) A structured and scalable test access architecture for TSV-based 3D stacked ICs, pp 1–6
14. Muresan V, Wang X, Muresan V, Vladutiu M (2004) Greedy tree growing heuristics on block-test scheduling under power constraints. J Electron Test: Theory Appl 20:61–78
15. Noia B, Goel SK, Chakrabarty K, Marinissen EJ, Verbree J (2010) Test-architecture optimization for TSV-based 3D Stacked ICs. In: Proceedings of the IEEE European Test Symposium (ETS), pp 24–29
16. Taouil M, Hamdioui S, Beenakker K, Marinissen EJ (2010) Test cost analysis for 3D die-to-wafer stacking. In: Proceedings of the IEEE Asian Test Symposium (ATS), pp 435–441
17. Verbree J, Marinissen EJ, Roussel P, Velenis D (2010) On the cost-effectiveness of matching repositories of pre-tested wafers for wafer-to-wafer 3D chip stacking. In: Proceedings of the IEEE European Test Symposium (ETS), pp 36–41
18. Wu X, Falkenstern P, Xie Y (2007) Scan chain design for three-dimensional Integrated Circuits (3D ICs). In: Proceedings of the International Conference on Computer Design (ICCD), pp 208–214
19. Zorian Y (1993) A distributed BIST control scheme for complex VLSI devices. In: Proceedings of the IEEE VLSI test symposium, pp 6–11

Breeta SenGupta recieved her M.Sc. degree from Indian Institute of Technology, Kharagpur, India, in 2009. Currently she is pursuing her Ph.D. at the Department of Computer and Information Science, Linköping Universitet, Sweden. Her area of research includes 3D integration and testing. Breeta is a member of the IEEE.

Urban Ingelsson recieved the M.Sc. degree from Linköping University, Sweden, in 2005 and the Ph.D. degree from University of Southampton, UK, in 2009. Currently he holds a postdoctoral position at the Department of Computer and Information Science, Linköping Universitet, Sweden. His research interests include test, diagnosis and fault-tolerance of digital circuits. Urban is a member of the IEEE.

Erik Larsson received his M.Sc., Tech. Lic, and Ph.D. degrees from Linköping University in 1994, 1998, and 2000, respectively. From October 2001 to December 2002, he held a Japan Society for the Promotion of Science (JSPS) funded postdoctoral position at the Computer Design and Test Laboratory at the Nara Institute of Science and Technology (NAIST), Nara, Japan. Currently, he is an associate professor at the Department of Computer and Information Science, Linköping Universitet, Sweden. His current research interests include the development of tools and design for testability methodologies to facilitate the testing of complex digital systems. The main focuses are on SoC test scheduling and test infrastructure design. He is a member of the program committee of Design and Test Automation in Europe (DATE) 2004, 2005, 2006, the IEEE Workshop on Design and Diagnostics of Electronic Circuits and Systems (DDECS) 2004, 2005, 2006, and the Workshop on RTL ATPG and DFT (WRTLT) 2005, 2006. He is a senior member of the IEEE.