Combined Test Data Selection and Scheduling for Test Quality Optimization under ATE Memory Depth Constraint[1]

Erik Larsson and Stina Edbom

Embedded Systems Laboratory

Department of Computer and Information Science

Linköpings Universitet

Sweden

contact: erila@ida.liu.se

Abstract - Testing is used to ensure high quality chip production. High test quality implies the application of high quality test data; however, the technology development has lead to a need of an increasing test data volume to ensure high test quality. The problem is that the test data volume has to fit the limited memory of the ATE (Automatic Test Equipment). In this paper, we propose a test data truncation scheme that for a modular core-based SOC (System-on-Chip) selects test data volume in such a way that the test quality is maximized while the selected test data is guaranteed to met the ATE memory constraint. We define, for each core as well as for the system, a test quality metric that is based on fault coverage, defect probability and number of applied test vectors. The proposed test data truncation scheme selects the appropriate number of test vectors for each individual core based on the test quality metric, and schedules the transportation of the selected test data volume on the Test Access Mechanism such that the system's test quality is maximized and the test data fits the ATE's memory. We have implemented the proposed technique and the experimental results, produced at reasonable CPU times, on several ITC'02 benchmarks show that high test quality can be achieved by a careful selection of test data. The results indicate that the test data volume (test application time) can be reduced to about 50% while keeping a high test quality.

Keywords: Test quality, System-on-Chip, Test data truncation, Test scheduling

[1]Preliminary versions of this paper have been presented at Asian Test Symposium (ATS'04) 4 and at VLSI SOC 22.

Introduction

The technology development has made it possible to develop chips where a complete system with an enormous number of transistors, which are clocked at an immense frequency and partitioned into a number of clock-domains, is placed on a single die. As the technology development makes it possible to design these highly advanced system chips or SOC (system-on-chip), the EDA (Electronic Design Automation) tools are aiming at keeping up the productivity, making it possible to design a highly advanced system with a reasonable effort in a reasonable time. New design methodologies are under constant development. At the moment, a modular design approach where modules are integrated to a system is promising. The advantage with such an approach is that pre-designed and pre-verified modules, blocks of logic or cores, with technology specific details, can at a reasonable time and effort be integrated to a system. The core provider designs the cores and the system integrator selects the appropriate cores for the system where the cores may origin from previous in-house designs or from different core vendors (companies). The cores can be delivered in various formats. They can in general be classified as soft cores, firm cores, and hard cores. Soft cores are general high-level specifications where the system integrator can, if necessary, apply modifications. Hard cores are gate-level specifications where only minor, if any, modifications are possible. Firm cores are somewhere between soft cores and hard cores. Soft cores allow more flexibility compared to hard cores. The advantage is that the system integrator can modify a soft core. On the other hand, hard cores can be made highly protected by the core provider, which often is desirable by the core provider.

A produced chip is tested to determine if it is faulty or not. In the test process, a number of test vectors, stored in an ATE (Automatic Test Equipment), are applied to the chip under test. If the produced test response from the applied vectors corresponds to the expected response, the chip is considered to be fault-free and can be shipped. However, testing these complex chips is becoming a problem, and one major problem is the increasing test data volume that has to be stored in the ATE. Currently, the test data volume increases faster than the number of transistors in a design 21. The increasing test data volume is due to (1) high number of fault sites because of the high amount of transistors, (2) new defect types introduced with nanometer process technologies, and (3) faults related to timing and delay since systems have higher performance and make use of multiple-clock domains 21.

The high test data volume is a problem. It is known that the purchase of a new ATE with higher memory capabilities is costly; hence, it is desirable to make use of the existing ATE instead of investing in a new. Vranken *et al.* 21discuss three alternatives to make the test data fit the ATE; (1) *test memory reload*, where the test data is divided into several partitions, is possible but not practical due to the high time involved, (2) *test data truncation*, the ATE is filled as much as possible and the test data that does not fit the ATE is simply not applied, leads to reduced test quality, and (3) *test data compression*, the test stimuli is compressed, however, it does not guarantee that the test data will fit the ATE. As, test memory reload is not practical, the alternatives are test data truncation and test data compression. This paper focuses on test data truncation where the aim is a technique that selects test data for each core such that the test quality is maximized for the system while making sure the test data volume fits the ATE memory.

The test data must be organized or scheduled in the ATE. A recent industrial study showed that by using test scheduling the test data was made to fit the ATE 5. The study demonstrated that the ATE memory limitation is a real and critical problem. The basic idea in test scheduling is to reduce the amount of idle bits to be stored in the ATE, and therefore scheduling must be considered in combination with the test data truncation scheme. Further, when discussing memory limitations, the ATE memory depth in bits is equal to the maximal test application time for the system in clock cycles 11. Hence, the memory constraint must be seen as a time constraint.

In this paper, we explore test data truncation. The aim is a technique that maximizes test quality while making sure that the selected test data fits the ATE. We assume that given is a core-based design and for each core the defect probability, the maximal fault coverage when all its test vectors have been applied, and the size of the test set (the number of test vectors) are given. We define for a core, a CTQ (core test quality) metric, and for the system, a STQ (system test quality) metric. The CTQ metric reflects that test data should be selected for a core (1) with high probability of having a defect, and (2) where it is possible to detect a fault using a minimal number of test vectors. For the fault coverage function we make use of an estimation function. Fault simulation can be used to extract the fault coverage at each test vector, however, it is a time consuming process and also it might not be applicable for all cores due to IP (Intellectual Property)-protection, for instance.

The test vectors in a test set can be applied in any order. However, regardless of the order, it is well-known in the test community that the first test vectors detects a higher number of faults compared to the last applied test vectors, and that the function fault coverage versus number of test vectors has an exponential/logarithmic behavior. We therefore assume that the fault coverage over time (number of applied test vectors) for a core can be approximated to an exponential function.

We make use of CTQ metric to select test data volume for each core in such a way that the test quality for the system is maximized (STQ), and we integrate the test data selection with test scheduling in order to verify that the selected test data actually fits the ATE memory. We have implemented our technique and we have made experiments on several ITC’02 benchmarks to demonstrate that high test quality can be achieved by applying only a sub-set of the test stimuli. The results indicate that the test data volume and the test application time can be reduced to 50% while the test quality remains high. Furthermore, it is possible to turn the problem (and our solution), and view it as: for a given test quality, which test data should be selected to minimize the test application time.

The advantage with our technique is that given a core-based system, a test set per core, a number on maximal fault coverage, and defect probability per core, we can select test data for the system and schedule the selected test data in such a way that the test quality is maximized and the selected test data fits the ATE memory. In the paper, we assume a single test per core. However, the technique can easily be extended to allow multiple tests per core by introducing constraint considerations in the scheme.

The rest of the paper is organized as follows. In Section 2 we present related work, and in Section 3 the problem definition is given. The test quality metric is defined in Section □4 and our test data selection and scheduling approach is described in Section Figure 1. 5. The experiments are presented in Section 6 and the paper is concluded in Section 7.

2. Related Work

Test scheduling and test data compression are examples of approaches proposed to reduce the high test data volumes that must be stored in the ATE in order to test SOCs. The basic principle in test scheduling is to organize the test bits in the ATE in such a way that the number of introduced so called idle bits (not useful bits) is minimized. The gain is reduced test application time and a reduced test data volume. A scheduling approach depends on the test architecture such as the AMBA test bus 6, the test bus 19 and the TestRail 16.

Iyengar *et al.* 9 proposed a technique to partition the set of scan chain elements (internal scan chains and wrapper cells) at each core into wrapper scan chains, which are connected to TAM wires in such a way that the total test time is minimized. Goel *et al.* 5 showed that ATE memory limitation is a critical problem. On an industrial design they showed that by using an effective test scheduling technique the test data can be made to fit the ATE.

There has also been scheduling techniques that make use of an abort-on-fail strategy that is the testing is terminated as soon as a fault is detected. The idea is that as soon as a fault is present, the chip is faulty and the testing can be terminated. Koranne minimizes the average-completion time by scheduling short tests early 13. Other techniques have taken the defect probability for each testable unit into account 7,12,14. Huss and Gyurcsik proposed a sequential technique making use of a dynamic programming algorithm for ordering the tests 7, while Milor and Sangiovanni-Vincentelli present a sequential technique based on selection and ordering of test sets 18. Jiang and Vinnakota proposed a sequential technique, where the information about the fault coverage provided by the tests is extracted from the manufacturing line 12. For SOC designs, Larsson *et al.* proposed a technique based on ordering of tests, considering different test bus structures, scheduling approaches (sequential vs. concurrent) and test set assumptions (fixed test time vs. flexible test time) 14. The technique takes defect probability into account; however, the probability of detecting a fault remains constant through the application of a test.

Several compression schemes have been used to compress the test data. For instance, Ichihara *et al.* used statistical codes 8, Chandra and Chakrabarty made use of Golomb codes 1, Iyengar *et al.* explored the use of run-length codes 10, Chandra and Chakrabarty tried Frequency-directed run-length codes 2, and Volkerink *et al.* have investigated the use of Packet-based codes 20.

All approaches above (test scheduling and test data compression techniques) reduce the ATE memory requirement. In the case of test scheduling, the effective organization means that both the test time and the needed test data volume are reduced, and in the case of test data compression, less test data is required to be stored in the ATE. The main advantage with these two approaches is that the highest possible test quality is reached since the whole test data volume is applied. However, the main disadvantage is that these techniques do not guarantee that the test data volume fits the ATE. Hence, they might not be applicable in practice. It means that there is a need for a technique that in a systematic way defines the test data volume for a system in such a way that the test quality is maximized while the test data is guaranteed to fit the ATE memory.

3. Problem Formulation

We assume that given is a core-based architecture with *n* cores denoted by *i,* and for each core *i* in the system, the following is given:

- $sc_{ij}=\{sc_{i1}, sc_{i2},..., sc_{im}\}$ - the length of the scanned elements at core *i* are given where *m* is the number of scanned elements,
- wi_i - the number of input wrapper cells,
- wo_i - the number of output wrapper cells,
- wb_i - the number of bidirectional wrapper cells,
- tv_i - the number of test vectors,
- fc_i - the fault coverage reached when all the tv_i test vectors are applied.
- pp_i - the pass probability per core and,
- dp_i - the defect probability per core (given as $1\text{-}pp_i$).

For the system, a maximal TAM bandwidth W_{tam}, a maximal number of *k* TAMs, and a upper-bound memory constraint M_{max} on the memory depth in the ATE are given.

The TAM bandwidth W_{tam} is to be partitioned into a set of *k* TAMs denoted by *j* each of width $W_{tam}=\{w_1, w_2, ...,w_k\}$ in such a way that:

$$W_{tam} = \sum_{j=1}^{k} w_j \tag{0.1}$$

and on each TAM, one core can be tested at a time.

Since the memory depth in the ATE (in bits) is equal to the test application time for the system (in clock cycles) 11, the memory constraint is actually a time constraint τ_{max}:

$$M_{max} = \tau_{max} \tag{0.2}$$

Our problem is to:

- for each core *i* select the number of test vectors (stv_i),
- partition the given TAM width W_{tam} into no more than *k* TAMs,
- determine the width of each TAM (w_j), j=1..*k*,
- assign each core to one TAM, and
- assign a start time for the testing of each core.

The selection of test data (stv_i for each core i) and the test scheduling should be done in such a way that the test quality of the system (defined in Section 4) is maximized while the memory constraint (M_{max}) (time constraint τ_{max}) is met.

4. Test Quality Metric

For the truncation scheme we need a test quality metric to (1) select test data for each core and (2) to measure the final system test quality. In this section we describe the metric where we for a core i take the following parameters into account to measure test quality:

- defect probability (dp_i),
- fault coverage (fc_i), and
- number of applied test vectors (stv_i).

The defect probability, the probability that a core has a defect, can be collected from the production line or set by experience. Defect probability has for a test quality metric to be taken into account since it is better to select test data for a core with a high defect probability than to select test data for a core with a low defect probability as the core with high defect probability it is more likely to hold a defect.

The possibility to detect faults depends on the fault coverage versus the number of applied test vectors; hence the fault coverage and the number of applied test vectors also have to be taken into account. Fault simulation can be used to extract which fault each test vector detects. However, in a complex core-based design with a high number of cores, fault simulation for each core is, if possible due to IP-protection, highly time consuming. A core provider may want to protect the core, which makes fault simulation impossible. We therefore make use of an estimation technique. It is known that the fault coverage does not increase linearly over the number of applied test vectors. For instance, Figure 1. shows the fault coverage for a set of ISCAS benchmarks. The following observation can be made: the curves have an exponential/logarithmic behavior as in Figure 2. We, therefore, assume that the fault coverage after applying stv_i test vectors for core i can be estimated to (Figure2(b)):

$$fc_i(stv_i) = \frac{\log(stv_i + 1)}{slopeConst} \qquad (0.3)$$

where the $slopeConst$ is given as follows:

$$slopeConst = \frac{\log(tv_i + 1)}{fc_i} \qquad (0.4)$$

and the +1 is used to adjust the curve to passes the origin.
For a system we assume that the test quality can be estimated to:

$$P(we\_find\_a\_defect \mid we\_have\_a\_defect\_in\_the\_SOC) \qquad (0.5)$$

The test quality defines the probability of finding a defect when we have the condition that the SOC has one defect. By introducing this probability, we find a way to measure the probability of finding a defect if a defect exist in the SOC and hence the test quality. However, it is important to note that our metric only describes the test quality and hence we are not introducing any assumptions about the number of defects in the SOC.

In order to derive an equation for the test quality using information about defect probability, fault coverage and the number of test vectors, we make use of definitions from basic probability theory 3:

Definition 1. If A and B are independent events => $P(A \cap B) = P(A)P(B)$

Definition 2. If $A \cap B$ is the empty set $\varnothing \Rightarrow P(A \cup B) = P(A) + P(B)$

Definition 3. $P(A \cap B) = P(A)P(B \mid A)$, where P(B|A) is the probability of B conditioned on A.

Furthermore, we assume (Section 3) that the quality of a test set (a set of test vectors) for a core i is composed by the following:

- fault coverage fc_i and
- defect probability dp_i.

Since the number of applied test vectors indirectly has an impact on the fault coverage, we define for each core i:

- stv_i - selected number of test vectors, and
- $fc_i(stv_i)$ - fault coverage after stv_i test vectors have been applied.

We do the following assumption:

- dp_i and fc_i are independent events.

Since we assume one defect in the system when we introduced test quality (Equation (0.5)), we can only have one defect in a core at a time in the system. Therefore we can say:

- The intersection of any of the events dp_i is the empty set .

For a system with n cores, we can now derive STQ (system test quality) from Equation (0.5) by using Definition 1, Definition 2 and Definition 3:

$$STQ = P(defect\_detected\_in\_the\_SOC \mid defect\_in\_the\_SOC) \Rightarrow$$

$$\frac{P(defect\_detected\_in\_the\_SOC \cap defect\_in\_the\_SOC)}{P(defect\_in\_the\_SOC)} \Rightarrow$$

$$\frac{\sum_{i=1}^{n} P(defect\_detected\_in\_core\_i \cap defect\_in\_core\_i)}{P(defect\_in\_the\_SOC)} \Rightarrow$$

$$\frac{\sum_{i=1}^{n} P(defect\_detected\_in\_core\_i)P(defect\_in\_core\_i)}{P(defect\_in\_the\_SOC)} \Rightarrow$$

$$\frac{\sum_{i=1}^{n} dp_i \times fc_i}{\sum_{i=1}^{n} dp_i} \qquad (0.6)$$

And for a single core, the CTQ (core test quality) is:

$$CTQ = \sum_{i=1}^{n} dpv_i \times fc_i(stv_i) \qquad (0.7)$$

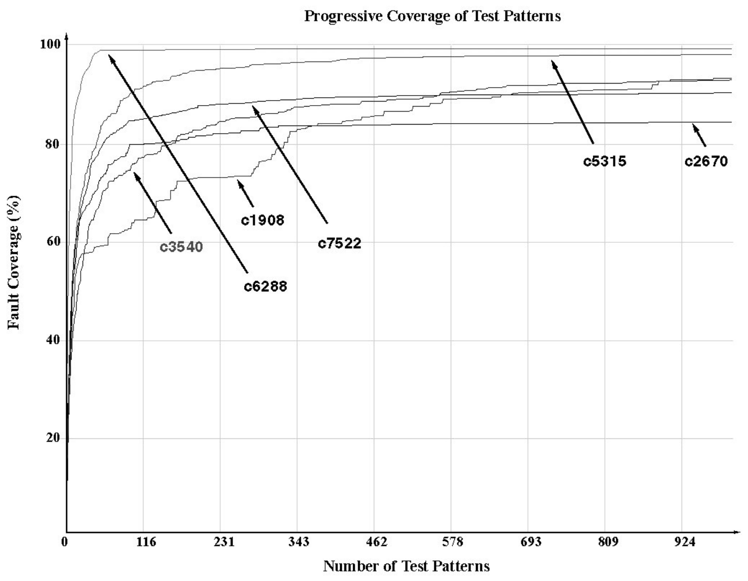

Figure 1. Fault coverage versus the number of test vectors for a set of ISCAS designs.

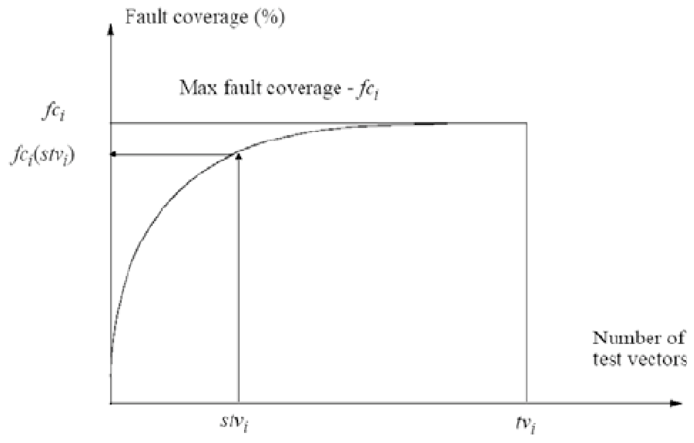

Figure 2. Fault coverage versus the number of test vectors estimated as an exponential/logarithmic function.

5. Test Scheduling and Test Vector Selection

In this section we describe our technique to optimize test quality by selecting test vectors for each core and schedule the selected vectors for an SOC under the time constraint given by the ATE memory depth (see Equation (0.2) and 11). We assume that given is a system as described in Section 3 and we assume an architecture where the TAM wires can be grouped into several TAMs and the cores connected to the same TAM are tested sequentially one after the other 19. We make use of the test quality metric defined in Section 4.

The scanned elements (scan-chains, input cells, output cells and bidirectional cells) at a core has to be configured into a set of wrapper chains, which are to be connected to a corresponding number of TAM wires. The wrapper scan chains, which are to be connected to the TAM wires w_j, should be as balanced as possible and we make use of the *Design_wrapper* algorithm proposed by Iyengar *et al*. 9. For a wrapper chain configuration at a core i where si_i is the longest wrapper scan-in chain and so_i is the longest wrapper scan-out chain, the test time for core i is given by 9:

$$\tau_i(w,tv) = (1+\max(si_i(w)))\times tv + \min(si_i(w), so_i(w)) \quad (0.8)$$

where tv is the number of applied test vectors for core i and w is the TAM width.

We need a technique to partition the given TAM width W_{tam} into a number of TAMs k and to determine which TAM a core should be assigned to. The number of different ways we can assign n cores to k TAMs grows with k^n, and therefore the number of possible alternatives will be huge. We need a technique to guide the assignment of cores to the TAMs. We make use of the fact that Iyengar *et al*. 9 made use of, which is that balancing the wrapper scan-in chain and wrapper scan-out chain introduces different number of ATE idle bits as the TAM bandwidth varies. We define *TWU*TWU_i (TAM width utilization) for a core *i* at a TAM of width *w* as:

$$TWU_i(w) = \max(si_i(w), so_i(w)) \times w \qquad (0.9)$$

and we make use of a single wrapper-chain (one TAM wire) as a reference point to introduce *WDC* (wrapper design cost) that measure the imbalance (introduced number of idle bits) for a TAM width *w* relative to TAM width 1:

$$WDC_i = TWU_i(w) - TWU_i(1) \qquad (0.10)$$

For illustration of the variations in the number of ATE idle bits, we plot in Figure 3. the value of *WDC* for different TAM widths (number of wrapper chains), obtained by using core 1 of the ITC'02 benchmark p93791. We also plot the maximum value of the scan-in and scan-out lengths at various TAM widths for the previous design in Figure 4. In Figure 4. several TAM widths have the same test time. For a set of TAM widths with the same test time, a Pareto-optimal point is the one with lowest TAM 9. We notice, we can notice that the TAM widths having a low value of the WDC, and hence a small number of idle bits, corresponds to the Pareto-optimal points. Hence, we make use of WDC to guide the selection of wrapper chains at a core.

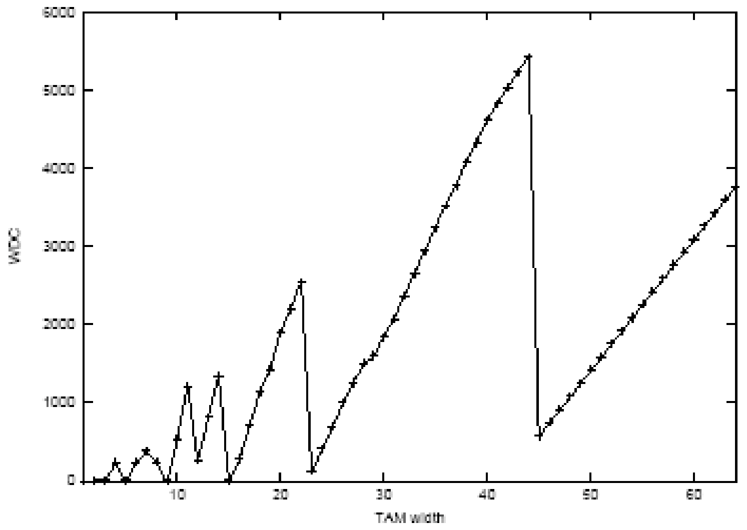

Figure 3. WDC over the TAM width. for core 1 in P93791.

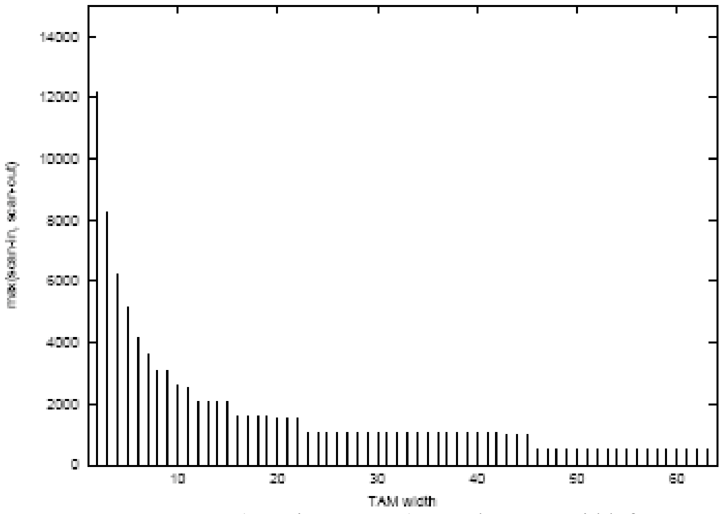

Figure 4. Max (scan-in, scan-out) over the TAM width for core 1 in P93791.

```
1. Given:
   $\tau_{max}$ - the upper bound on test time limit for the system
   $W_{tam}$ - number of TAM wires - distributed over $k$ TAMs $w_1$,
   $w_2$, ..., $w_k$ in such a way that Eq.
2. Variables:
   $stv_i = 0$   //selected number of test vectors for core $i$
   $TAT = 0$   // test application time of the system
3. Compute $WDC_i$ for all cores at all $k$ TAMs
4. Select best TAM for each core based on $WDC_i$
5. **while** $TAT < \tau_{max}$ **at any TAM begin**
6.   **for** $i$=1 to $n$ **begin** // For all cores
7.     Compute $\tau(w_j,1)$ (Eq. 8)
8.     Compute $STQ_i$ assuming $stv_i=stv_i+1$
9.   **end**
10.  **for core** with highest $STQ/\tau(w_j,1)$ and $stv_i<tv_i$
11.    $stv_i=stv_i+1$
12.  **for all** cores where $stv_i>0$ **begin**// selected vectors
13.    Assign core to an available TAM with minimal $WDC_i$
14.    **if** a TAM is full ($<\tau_{max}$) - mark TAM as unavailable.
15.  **end**
16. Compute and return $STQ$
17. **end**
```

Figure 5. Test vector selection and test scheduling algorithm.

The algorithm for our test truncation scheme is outlined in Figure 5. Given is a system, the upper bound on the test time (τ_{max}) and the TAM width (W_{tam}). Initially no test vectors are selected for any core (stv_i=0 for all i) and the test time for the test schedule is zero (TAT=0). The test vector that contributes most to improving STQ is selected, assigned to a TAM where WDC is minimal and scheduled on the selected TAM in order to make sure that the τ_{max} is not violated. Additional vectors are selected one by one in such a way that STQ is maximized, and after each selection the schedule is created to verify that the time constraint (ATE memory depth constraint) is not violated. Note that the test vectors for a core might not be selected in order. For instance, in a system with two cores A and B, the first vector can be selected from core A, the second from core B, and the third from core A. However, at the scheduling, the test vectors for each core are grouped and scheduled as a single set. The algorithm (Figure 5.) assumes a fixed TAM partition (number of TAMs and their width). We have therefore added an outer loop that makes sure that we explore all possible TAM configurations.

5.1. Illustrative Example

To illustrate the proposed technique for test scheduling and test vector selection, we make use of an example where the time constraint is set to 5% of the maximal test application time (the time when all available test vectors are applied). For the example, we make use of the ITC'02 benchmark 17 d695 with the data presented in TABLE I. As the maximal fault coverage for a core when all test vectors are applied and the pass probability per core are not given in the ITC'02 benchmarks, we have added these numbers. In order to show the importance of combining test scheduling and test vector selection, we compare our proposed technique to a naive approach where we order the tests and assign test vectors according to the initial sorted order until the time limit (ATE memory depth) is reached. For this naive approach we consider three different techniques.

	Core										
	0	1	2	3	4	5	6	7	8	9	10
Scan-chains	0	0	0	1	4	32	16	16	4	32	32
Inputs w_i	0	32	207	34	36	38	62	77	35	35	28
Outputs w_o	0	32	108	1	39	304	152	150	49	320	106
Test vectors tv_i	0	12	73	75	105	110	234	95	97	12	68
Pass probability pp_i	97	98	99	95	92	99	94	90	92	98	94
Max fault coverage fc_i (%)	95	93	99	98	96	96	99	94	99	95	96

TABLE I DATA FOR BENCHMARK D695.

1. Sorting when not considering defect probability and fault coverage (Technique 1).
2. Sorting when considering defect probability but not fault coverage. The cores are sorted in descending order according to defect probability (Technique 2).
3. Sorting when considering defect probability in combination with fault coverage. In this technique, we make use of the *STQ* (Equation(0.6)) equation to find a value of the test quality for each core. The cores are then sorted in descending order according to test quality per clock cycle. The sorting constant is described in Equation (0.11) (Technique 3).

$$sortConst = \frac{dp_i \times fc_i(tv_i)}{\tau(w,tv_i) \times \sum_{i=1}^{n} dp_i} \tag{0.11}$$

For our test vector selection and test scheduling technique, we consider three cases where we divide the TAM into one (Technique 4), two (Technique 5) or three test buses (Technique 6). The selected test data volume per core for each of the six scheduling techniques is reported in Table TABLE II and the test schedules with the corresponding *STQ* are presented in Figure 6. Figure 6. (a) illustrates the case when no information about defect probability and fault coverage is used in the test ordering. As seen in the figure, such technique produces a schedule with an extremely low system test quality (*STQ*). By making use of the information on defect probability (Figure 6. (b)), respective defect probability and fault coverage (Figure 6. (c)) in the ordering, we can improve the test quality significantly. Although it is possible to increase the *STQ* by using an efficient sorting technique, we are still not exploiting the fact that the first test vectors in a test set detect more faults than the last test vectors. In (Figure 6. (d) - (f)), we make use this information as we are using our proposed technique for test scheduling and test vector selection. We note that it is possible to further improve the *STQ* by dividing the TAM into several test buses (Figure 6. (e) - (f)).

Technique	Selected test data for each core (%)										
	0	1	2	3	4	5	6	7	8	9	10
Technique 1	0	0	100	0	0	20	0	0	0	0	0
Technique 2	0	0	0	0	0	0	0	54.7	0	0	0
Technique 3	100	0	0	0	0	0	0	52.6	0	0	0
Technique 4	0	100	9.6	6.7	4.8	0	1.7	10.5	6.2	8.3	4.4
Technique 5	0	100	9.6	16.0	10.5	0	3.8	21.1	13.4	8.3	4.4
Technique 6	0	100	9.6	17.3	11.4	0	2.6	13.7	17.5	33.3	14.7

TABLE II SELECTED TEST VECTORS (%) FOR THE CORES IN D695 CONSIDERING DIFFERENT SCHEDULING TECHNIQUES.

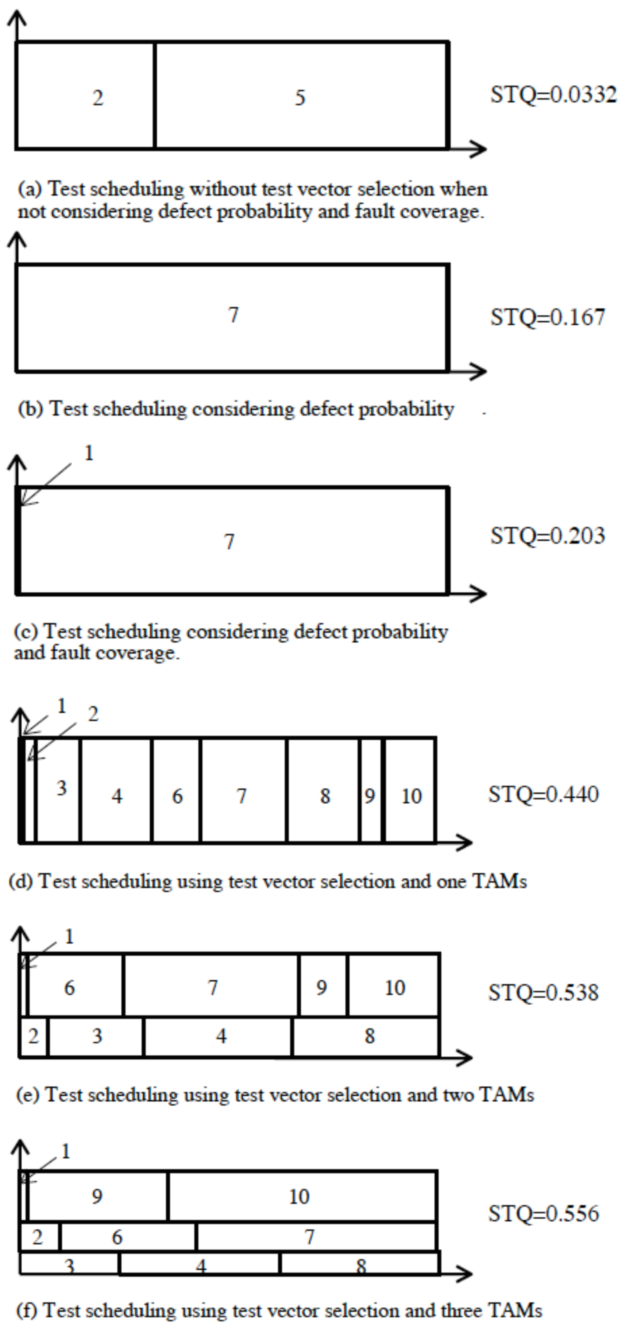
2
5
STQ=0.0332
(a) Test scheduling without test vector selection when not considering defect probability and fault coverage.
7
STQ=0.167
(b) Test scheduling considering defect probability
1
7
STQ=0.203
(c) Test scheduling considering defect probability and fault coverage.
1
2
3
4
6
7
8
9
10
STQ=0.440
(d) Test scheduling using test vector selection and one TAMs
1
6
7
9
10
2
3
4
8
STQ=0.538
(e) Test scheduling using test vector selection and two TAMs
1
9
10
2
6
7
3
4
8
STQ=0.556
(f) Test scheduling using test vector selection and three TAMs

Figure 6. Test quality (STQ) results for different test scheduling techniques.

5.2. Optimal Solution For Single TAM

The algorithm above can easily be improved to produce an optimal solution in the case of a single TAM. The algorithm above aborts the assignment of test vectors immediately when the time constraint (memory constraint) is reached - a selected test vector cannot be assigned since it violates the constraint. However, test vectors from other cores (not from the core that violates the time constraint) could have been selected while making sure that they do not violate the ATE constraint.

Note, that the selection of test vectors is based on a monotonically decreasing function. The test vector that contributes most to the test quality is first selected. That process continuous on an updated list until the constraint is reached. In the case of a single TAM, the scheme is optimal.

6. Experimental Results

The aim with the experiments is to demonstrate that the test quality can be kept high by using the proposed ATE memory constrained test data truncation scheme. We have implemented the proposed technique described above, and we have in the experiments made use of five ITC'02 benchmarks 17, d281, d695, p22810, p34392, and p93791. It is given for each core in these benchmarks, the number of test vectors, the number of scanned elements (number and length of the scan-chains), the number of input pins, bidirectional pins and output pins. The netlists for the ITC'02 benchmarks are not publicly available, and therefore we have, in order to perform experiments, added for each core a pass probability and a maximal fault coverage number when all its test vectors are applied (TABLE III).

In order to have a memory (time) constraint from the ATE, we performed for each design a schedule where all vectors are applied and that test application time reefers to 100%. We have performed experiments at various ATE memory depths constraints (equal to time constraints (see Equation (0.2) and 11)) and these constraints are set as a percentage of the time it would take to apply all test vectors.

We identify six techniques:

1. Test scheduling when not considering defect probability nor fault coverage and testing is aborted at τ_{max} - technique 1.
2. Test scheduling when considering defect probability but not fault coverage and testing is aborted at τ_{max} - technique 2.
3. Test scheduling when considering defect probability as well as fault coverage and testing is aborted at τ_{max} - technique 3.
4. Test scheduling and test vector selection when considering defect probability and fault coverage, using one TAM - technique 4.
5. Test scheduling and test vector selection when considering defect probability and fault coverage, using up to two TAMs - technique 5.
6. Test scheduling and test vector selection when considering defect probability and fault coverage, using up to three TAMs - technique 6.

		Core																																
		0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32
d281	pp	98	98	99	95	92	99	94	90	92																								
	fc	93	98	97	95	98	98	96	99	97																								
d695	pp	97	98	99	95	92	99	94	90	92	98	94																						
	fc	95	93	99	98	96	96	99	94	99	95	96																						
p22810	pp	98	98	97	93	91	92	99	96	96	95	93	91	92	93	99	99	99	95	96	97	93	99	96	98	99	92	91	91	93				
	fc	95	99	97	98	94	99	99	97	95	97	97	99	99	94	97	94	99	98	94	95	99	99	95	98	95	99	99	97	98				
p34392	pp	98	98	97	91	95	94	94	93	99	99	91	91	90	95	94	96	96	97	92	90													
	fc	97	97	99	98	99	99	97	98	94	96	98	98	99	94	97	95	98	98	95	95													
p93791	pp	99	99	99	97	90	91	92	98	96	91	94	93	91	91	90	99	98	97	99	99	99	90	99	90	98	92	96	95	91	90	96	99	99
	fc	99	99	95	98	98	99	97	99	95	96	97	99	99	94	98	94	97	97	95	95	99	98	96	98	94	99	99	98	99	97	98	99	94

TABLE III THE PASS PROBABILITY AND MAXIMAL FAULT COVERAGE NUMBERS FOR THE CORES IN THE SOCS (%).

SOC	% of max test time	Technique 1	Technique 2	Technique 3	Technique 4	Technique 5	Technique 6
		STQ	STQ	STQ	STQ	STQ	STQ
p93791 TAM width 16	5	0.00542	0.118	0.560	0.719	0.720	0.720
	10	0.0248	0.235	0.618	0.793	0.796	0.796
	25	0.0507	0.458	0.747	0.884	0.885	0.885
	50	0.340	0.619	0.902	0.945	0.945	0.945
	75	0.588	0.927	0.958	0.969	0.969	0.969
	100	0.976	0.976	0.976	0.976	0.976	0.976
p93791 TAM width 32	5	0.00542	0.118	0.559	0.715	0.748	0.748
	10	0.0249	0.235	0.618	0.791	0.822	0.822
	25	0.0507	0.459	0.742	0.883	0.908	0.908
	50	0.340	0.619	0.902	0.945	0.960	0.960
	75	0.584	0.927	0.957	0.969	0.974	0.974
	100	0.976	0.976	0.976	0.976	0.976	0.976
p93791 TAM width 64	5	0.00535	0.118	0.499	0.703	0.752	0.752
	10	0.00606	0.235	0.567	0.780	0.827	0.827
	25	0.0356	0.461	0.739	0.878	0.918	0.918
	50	0.335	0.620	0.901	0.944	0.965	0.965
	75	0.566	0.927	0.961	0.969	0.975	0.975
	100	0.976	0.976	0.976	0.976	0.976	0.976

TABLE IV COMPARISON OF DIFFERENT TAM WIDTHS USING ITC'02 BENCHMARK P93791

In the first experiment, we analyze the importance of TAM width. We have made experiments on benchmark p93791 at TAM width 16, 32 and 64 at time constraint 5%, 10%, 25%, 50%, 75%, and 100% of the test application time if all test data is applied. The results are collected in Table TABLE IV and illustrated for technique 2, 4, and 6 in Figure 7. . The results show that the produced results (STQ) are at a given time constraint, rather similar at various TAM widths. Therefore, for the rest of the experiments we assume a TAM bandwidth W_{tam} of 32.

The results from the experiments on d281, d695, p22810, p34392, and p93791 are collected in TABLE V, and also plotted in Figure 13. . In column 1 the design name is given, in column 2 the percentage of the test time is given, and in column 3 to 8 the produced STQ is reported for each technique (1 to 6). The computational cost for every experiment is in the range of a few seconds to a few minutes.

From the experimental results collected in TABLE V and Figure 13. we learn that the STQ value increases with the time constraint (a larger ATE memory results in a higher STQ), which is obvious. It is also obvious that the STQ value for a design is the same at 100% test time, all test data is applied. From the results, we also see that test set selection improves the test quality when comparing STQ at the same test time limit. That is, technique 4, 5, 6 have significant higher STQ value compared to technique 1, 2 and 3. But also important, we note that we can achieve a high test quality at low testing times. Take design p93791, for example, where the STQ value (0.584) for technique 1 at 75% of the testing time is lower than the STQ value (0.748) at only 5% for technique 6. It means that it is possible, by integrating test set selection and test scheduling, to reduce the test application time while keeping the test quality high. Also, we have selected rather high pass probabilities and rather high fault coverage as these numbers are not publicly available for the ITC'02 designs. For designs with lower pass probabilities and lower fault coverage, and also, for designs where the variations in these numbers are higher, our technique becomes more important.

7. Conclusions

The technology development has made it possible to design extremely advanced chips where a complete system is placed on a single die. The requirement to test these system chips increases, and especially, the growing test data volume is becoming a problem. Several test scheduling techniques have been proposed to organize the test data in the ATE in such a way that the ATE memory limitation is not violated, and several test compression schemes have been proposed to reduce the test data volume. However, these techniques do not guarantee that the test data volume fits the ATE.

In this paper we have therefore proposed a test data truncation scheme that systematically selects test vectors and schedules the selected test vectors for each core in a core-based system in such a way that the test quality is maximized while the constraint on ATE memory depth is met.

SOC	% of max test time	Technique 1	Technique 2	Technique 3	Technique 4	Technique 5	Technique 6
		STQ	STQ	STQ	STQ	STQ	STQ
d281	5	0.0209	0.164	0.496	0.674	0.726	0.726
	10	0.0230	0.186	0.563	0.774	0.818	0.818
	25	0.198	0.215	0.834	0.879	0.905	0.912
	50	0.912	0.237	0.903	0.935	0.949	0.949
	75	0.956	0.870	0.923	0.960	0.968	0.968
	100	0.974	0.974	0.974	0.974	0.974	0.974
d695	5	0.0332	0.167	0.203	0.440	0.538	0.556
	10	0.0370	0.257	0.254	0.567	0.670	0.690
	25	0.208	0.405	0.510	0.743	0.849	0.863
	50	0.335	0.617	0.803	0.879	0.952	0.952
	75	0.602	0.821	0.937	0.946	0.965	0.965
	100	0.966	0.966	0.966	0.966	0.966	0.966
p22810	5	0.0333	0.174	0.450	0.659	0.691	0.759
	10	0.0347	0.186	0.608	0.764	0.796	0.856
	25	0.0544	0.398	0.769	0.885	0.900	0.940
	50	0.181	0.830	0.912	0.949	0.949	0.968
	75	0.600	0.916	0.964	0.969	0.969	0.973
	100	0.973	0.973	0.973	0.973	0.973	0.973
p34392	5	0.0307	0.312	0.683	0.798	0.843	0.859
	10	0.0341	0.331	0.766	0.857	0.893	0.898
	25	0.0602	0.470	0.846	0.919	0.940	0.942
	50	0.533	0.492	0.921	0.950	0.963	0.967
	75	0.547	0.906	0.943	0.965	0.972	0.972
	100	0.972	0.972	0.972	0.972	0.972	0.972
p93791	5	0.00542	0.118	0.559	0.715	0.748	0.748
	10	0.0249	0.235	0.618	0.791	0.822	0.822
	25	0.0507	0.459	0.742	0.883	0.908	0.908
	50	0.340	0.619	0.902	0.945	0.960	0.960
	75	0.584	0.927	0.957	0.969	0.974	0.974
	100	0.976	0.976	0.976	0.976	0.976	0.976

TABLE V EXPERIMENTAL RESULTS. TECHNIQUE 1 - ONLY TEST SCHEDULING, TECHNIQUE 2 - TEST SCHEDULING AND CONSIDERING DEFECT PROBABILITY (DP), TECHNIQUE 3 - TEST SCHEDULING CONSIDERING DP AND FAULT COVERAGE (FC), TECHNIQUE 4 - TEST VECTOR SELECTION AND TEST SCHEDULING CONSIDERING DP AND FC AT ONE TAM, TECHNIQUE 5 - AS IN TECHNIQUE 4 BUT TWO TAMS, TECHNIQUE 6 - AS IN TECHNIQUE 4 BUT THREE TAMS.

We have defined a test quality metric based on defect probability, fault coverage and the number of applied vectors that is used in the proposed test data selection scheme. We have implemented our technique and the experiments on several ITC'02 benchmarks at reasonable CPU times show that high test quality can be achieved by careful selection of test data. Further, our technique can be used to shorten the test application time for a given test quality value.

REFERENCES

1. Chandra and K. Chakrabarty, "System-on-a-Chip Test Data Compression and Decompression Architectures Based on Golomb Codes", *Transactions on CAD of IC and Systems*, pp. 355-367, Vol. 20, No. 3, 2001.

2. Chandra and K. Chakrabarty, "Frequency -Directed-Run-Length (FDR) Codes with Application to System-on-a-Chip Test Data Compression", *Proceedings of VLSI Test Symposium (VTS)*, pp. 42-47, 2001.

3. G. Blom, "Sannolikhetsteori och statistikteori med tillŠmpningar", Studentlitteratur, 1989.

4. S. Edbom and E. Larsson, "An Integrated Technique for Test Vector Selection and Test Scheduling under Test Time Constraint", Proceedings of Asian Test Symposium (ATS), pp. 254-257, 2004.

5. S. K. Goel, K. Chiu, E. J. Marinissen, T. Nguyen, and S. Oostdijk, "Test Infrastructure Design for the Nexperia^{TM}Home Platform PNX8550 System Chip", *Proceedings of Design, Automation and Test in Europe Conference (DATE)*, pp. 1530-1591, Paris, France, 2004.

6. P. Harrod, "Testing reusable IP-a case study", *Proceedings of International Test Conference (ITC)*, pp. 493-498, Atlantic City, NJ, USA, 1999.

7. S. D. Huss and R. S. Gyurcsik, "Optimal Ordering of Analog Integrated Circuit Tests to Minimize Test Time", *Proceedings of Design Automation Conference (DAC)*, pp. 494-499, 1991.

8. H. Ichihara, A. Ogawa, T. Inoue, and A. Tamura, "Dynamic Test Compression Using Statistical Coding", *Proceedings of Asian Test Symposium (ATS)*, pp. 143-148, Kyoto, Japan, November 2001.

9. V. Iyengar, K. Chakrabarty, and E. J. Marinissen, "Test wrapper and test access mechanism co-optimization for system-on-chip", *Proceedings of International Test Conference* (ITC), pp. 1023-1032, Baltimore, MD, USA, 2001.

10. V. Iyengar, K. Chakrabarty, and B. Murray, “Built-In Self-Testing of Sequential Corcuits Using Precomputed Test Sets”, *Proceedings of VLSI Test Symposium (VTS)*, pp. 418-423, 1998.

11. V. Iyengar, S. K. Goel, E. J. Marinissen, and K. Chakrabarty, “Test resource optimization for multi-site testing of SOCs under ATE memory depth constraints”, *Proceedings of International Test Conference (ITC)*, pp. 1159 - 1168, Baltimore, USA, October 2002.

12. W. J. Jiang and B. Vinnakota, “Defect-Oriented Test Scheduling”, *Transactions on Very-Large Scale Integration (VLSI) Systems, Vol. 9, No. 3*, pp. 427-438, June 2001.

13. S. Koranne, “On Test Scheduling for Core-Based SOCs”, *Proceedings of International Conference on VLSI Design (VLSID)*, pp. 505-510, Bangalore, India, January 2002.

14. E. Larsson, J. Pouget, and Z. Peng, “Defect-Aware SOC Test Scheduling“, *Proceedings of VLSI Test Symposium (VTS)*, Napa Valley, CA, USA, pp. 359-364, April 2004.

15. T.L. McLaurin and J.C. Potter, “On-the-Shelf Core Pattern Methodology for ColdFire(R) Microprocessor Cores”, *Proceedings of International Test Conference (ITC)*, pp. 1100-1107, 2000.

16. E. J. Marinissen, R. Arendsen, G. Bos, H. Dingemanse, M. Lousberg, and C. Wouters, “A structured and scalable mechanism for test access to embedded reusable cores”, *Proceedings of International Test Conference (ITC)*, pp. 284-293, Washington, DC, USA, October 1998.

17. E. J. Marinissen, V. Iyengar, and K. Chakrabarty, “A Set of Benchmarks for Modular Testing of SOCs”, *Proceedings of International Test Conference (ITC)*, pp. 519-528, Baltimore, MD, USA, October 2002.

18. L. Milor and A. L. Sangiovanni-Vincentelli, “Minimizing Production Test Time to Detect Faults in Analog Circuits”, *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems.,* Vo. 13, No. 6, pp 796-, June 1994.

19. P. Varma and S. Bhatia, “A Structured Test Re-Use Methodology for Core-based System Chips”, *Proceedings of International Test Conference (ITC)*, pp. 294-302, Washington, DC, USA, October 1998.

20. E. H. Volkerink, A. Khoche, and S. Mitra, “Packet-based Input Test Data Compression Techniques”, *Proceedings of International Test Conference (ITC)*, pp. 154-163, Baltimore, MD, USA, October 2002.

21. H. Vranken, F. Hapke, S. Rogge, D. Chindamo, and E. Volkrink, “ATPG Padding And ATE Vector Repeat Per Port For Reducing Test Data Volume”, *Proceedings of International Test Conference (ITC)*, pp. 1069-1078, Charlotte, NC, USA, 2003.

22. E. Larsson and S. Edbom, “Combined Test Data Selection and Scheduling for Test Quality Optimization under ATE Memory Depth Constraint”, pp. 429-434, IFIP VLSI-SOC 2005, Perth, Australia, October 17-19, 2005.

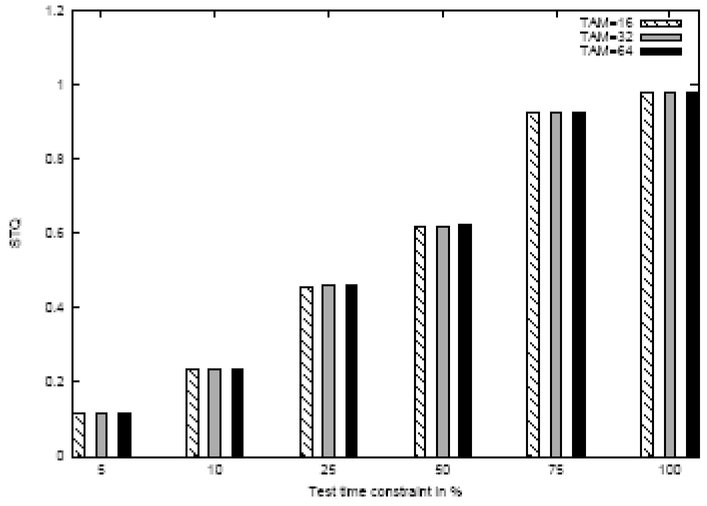

Figure 7. STQ comparison at TAM width 16, 32, and 64 for technique 2 on design p93791.

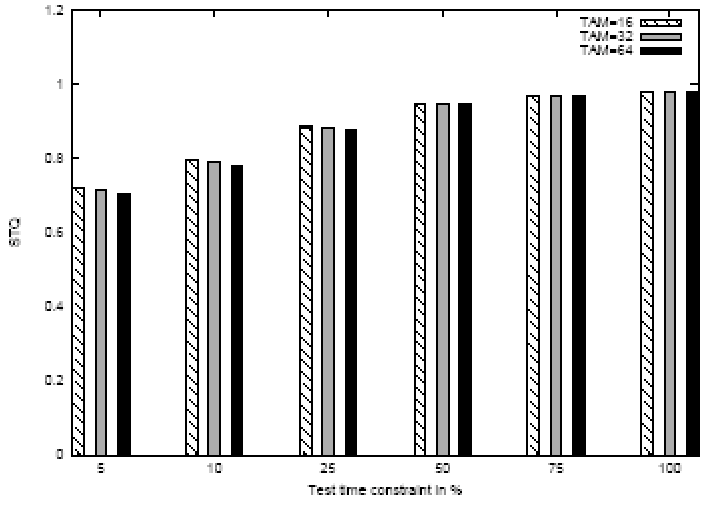

Figure 8. STQ comparison at TAM width 16, 32, and 64 for technique 4 on design p93791

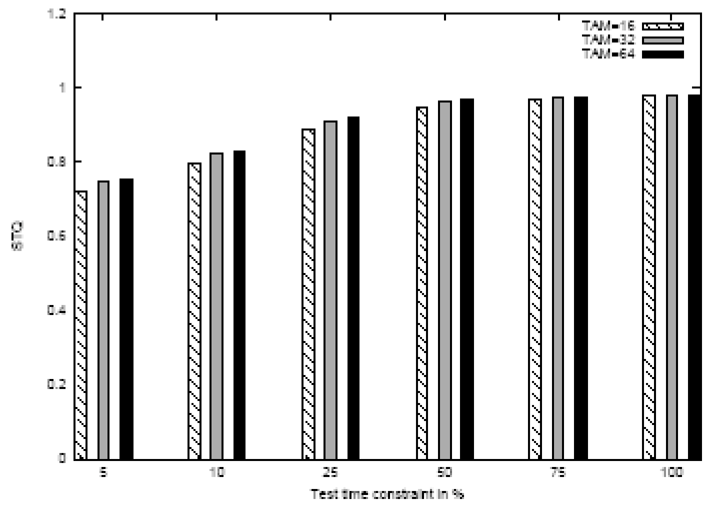

Figure 9. STQ comparison at TAM width 16, 32, and 64 for technique 6 on design p93791.

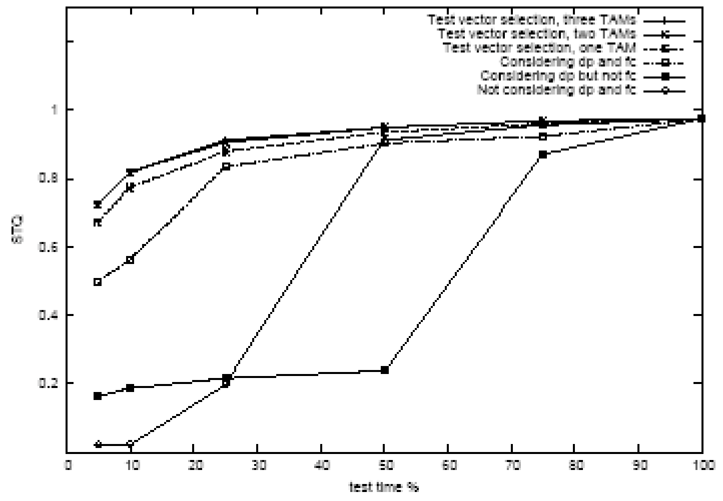

Figure 10. STQ at various test time limits for design d281.

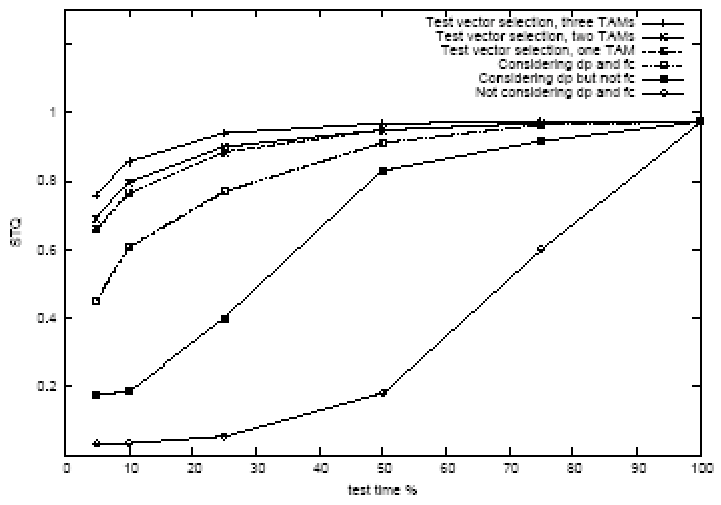

Figure 11. STQ at various test time limits for design d695.

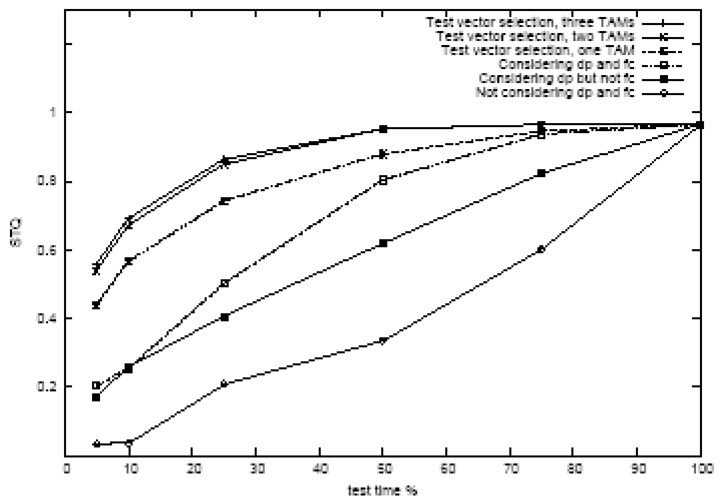

Figure 12. STQ at various test time limits for design p22810.

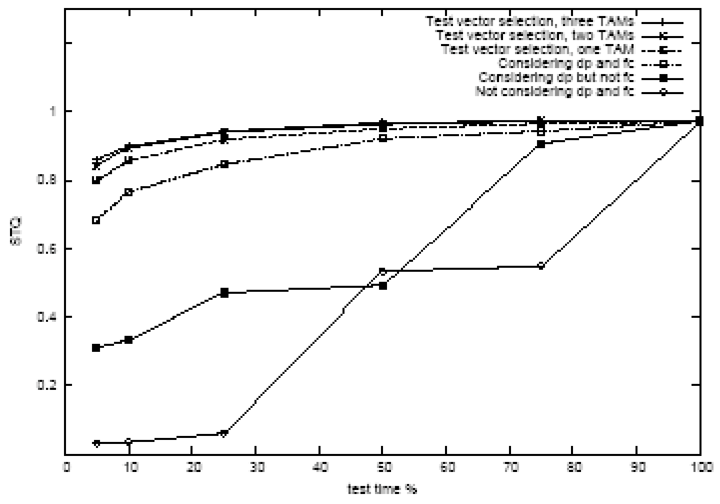

Figure 13. STQ at various test time limits for design d281.

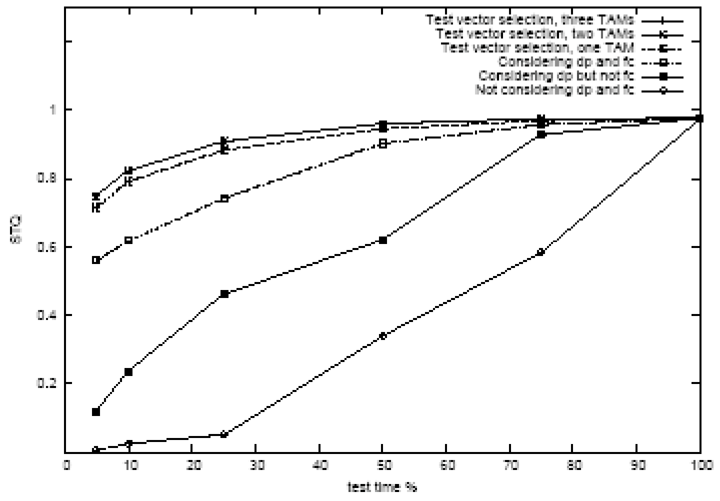

Figure 14. STQ at various test time limits for design p93791