A Genetic Algorithm Based Approach for System-on-Chip test Scheduling using Dual Speed TAM with Power Constraint*

CHANDAN GIRI [1] DILIP KUMAR REDDY TIPPARTHI[2] and SANTANU CHATTOPADHYAY [3]
Department of Electronics and Electrical Communication Engineering
Indian Institute of Technology Kharagpur
Kharagpur -721 302, West Bengal
INDIA
{[1]chandan, [3]santanu}@ece.iitkgp.ernet.in, [2]dilipreddytipparthi@gmail.com

Abstract: - Increasing complexity of System-on-Chip (SOC) has encouraged the engineers to design versatile automated test equipments (ATE) that can drive simultaneously different data channels at different data rates so that overall test cost can be reduced. Devices like Agilent 93000 series tester and Tiger system from Teradyne provide such flexibility to drive different channels at different data rates. Number of tester channels with higher data rate is limited due to different constraints like power rating of the SOCs, limitation of scan frequency, complexity of ATE etc. Hence proper utilization of the tester channels to reduce test time and thereby test cost is important. In this paper we provide a Genetic algorithm based approach for SOC-level TAM architecture optimization that minimizes testing time considering two different data rates for ATE channels. Our approach achieves better results than those reported in the literature. Experimental results show that the maximum improvement of 40.99% in testing time can be achieved. We have also addressed the issue of power constrained testing.

Key-Words: - System-On-Chip test, Test scheduling, Test access mechanism, Dual speed, Power constraints, TAM, Power-constrained testing.

1 Introduction

In order to meet the demand of low time-to-market, core-based System-on-Chip (SOC) design has become a widely used paradigm for integrated circuit design. An SOC typically contains several heterogeneous modules called cores of digital logic, embedded memories and analog modules. To minimize the development cycle different pre-designed and pre-verified intellectual property cores like CPUs, DSPs, memories etc. are used. Integrating reusable cores into an SOC involves complicated design and test issues. In today's scenario, different cores of an SOC may operate at different scan clock frequencies. Hence to test the cores with higher scan frequency, test data need to be transported at higher data rate along some selected tester channels of the ATE. To provide such flexibility, ATE vendors manufactured a class of tester devices that can drive simultaneously different tester channels with different data rates [1]. Agilent 93000 series tester based on port scalability and test-processor-per-pin architecture [2] and the Tiger system from Teradyne [3] are such tester devices that are widely used. In case of Teradyne technology based testers, test data rate can be increased for selected pin groups to match the SOC requirement. But, due to different constraints like ATE resource limitations, SOC power ratings, limitations of scan clock frequencies etc., it is needed to efficiently use the ATE channels with higher data rates such that test time can be reduced which in turn reduces the test cost.

The test methodology follows modular design process where embedded cores are isolated from the surrounding logic by using a shell-like structure, called test wrapper. The Test Access Mechanism (TAM) is used to transport test data to and from cores. One of the important issues of core-based SOC testing is to design an effective and efficient TAM. A number of solutions exist for accessing the embedded cores from chip I/Os. Bhatia et al. [34] proposed a grid-based *CoreTest* methodology that uses test-points like storage elements, embedded cells and observation points. Direct parallel access is

* This work is supported in part by the Dept. of Science & Technology, Govt. of India, under grant SR/S3/EECE/19/2003-SERC-ENGG, dated 05-05-2004

made through the "soft" netlist to these test points via SOC I/O pins. In this method test logic and wiring can be shared, thus reducing the hardware overhead. In [35] Whetsel presented a test access architecture that utilizes the IEEE 1149.1 test access port and a novel TAM linking module for the overall SOC test control. Aerts and Marinissen [36] proposed test access architecture based on dedicated test buses. Three basic types of test access architectures were introduced by them. These are Multiplexing, Daisy Chain and Distribution architectures. Besides these architectures, two other test architectures have been introduced. These are *TestBus*[37] and *TestRail*[38] architectures. In [30] also a TestRail architecture based test solution has been proposed. Tseng et al [40] has been proposed self-test structure for crosstalk fault test in SOC buses.

Modular testing also supports the use of port scalable testers for dual-speed testing of SOCs. For port scalable testers, the group of TAM wires connected to a particular port can be configured to operate at the same scan data rate. Hence depending on the availability of high-speed and low-speed TAM wires, ports can be configured accordingly to operate at on-demand frequency.

The problem of designing an optimized TAM architecture and determine a test schedule to minimize the SOC testing time has been shown in the literature to be NP-hard [4]. Several heuristics have been used to solve this problem. Several recent works considered various aspects of the TAM architecture optimization with test scheduling problem. Earlier works propose methods to solve wrapper design and test scheduling as separate problems. [4] and [5] proposed an integer linear programming based solution for co-optimization of wrapper design and test scheduling for SOCs. Huang et al [6] formulated the problem of SOC pin allocation to cores and test scheduling using 2-D bin-packing or rectangle packing approach. Same authors also proposed 3-D bin-packing approach [7] considering power constraints. Several other works [8-13] also consider SOC power dissipation constraints during scheduling. A heuristic approach using the sequence pair representation for test scheduling has been considered in [14]. Zou et al [15] proposed test scheduling algorithm based on simulated annealing (SA) using the sequence pair representation. A B*-tree based approach has been proposed in [16] to get the test schedule. In [17] SOC test scheduling with reconfigurable core wrappers has been proposed. Ant colony optimization (ACO) based approach [18] considers the rectangle packing for test scheduling solution. A two-stage genetic algorithm (GA) based approach has been proposed in [19] where each solution is represented by a sequence pair. A GA based design and optimization of SOC test solution has been proposed by Sakthivel et al.[31] and Giri et al [39].

However, in all of these proposed schemes, it is assumed that at any instant of time, test data are transferred from the ATE at a single data rate. It can be noted that all the above mentioned approaches assumes static TAM assignments, that is TAM width is fixed during test. Koranne [32] first proposed a reconfigurable core wrapper approach that allows a dynamic change in the TAM width during the execution of core test, so that it may lead to more efficient test schedule. In [33] SOC test scheduling with reconfigurable core wrappers has been used. Although reconfigurable wrappers lead to efficient test schedules, more gate and routing overheads are imposed. It also increases the complexity of control mechanisms of the wrappers. Sehgal et al. [20] proposed a scheme that matches the ATE channels with higher data rate with core scan chain frequency using virtual TAMs considering the availability of dual-speed ATEs. But main drawback is the need for higher number of TAM wires on the SOC and the extra frequency division hardware for bandwidth matching. Same authors in [21] also proposed a TAM architecture optimization with test scheduling scheme based on rectangle packing approach considering two available data rates for the ATE channels. But it does not need any extra hardware. In [29] again same authors proposed multi-frequency TAM architecture optimization and TAM scheduling using TAM width partitioning so that port-scalable testers can be used efficiently.

In this work we concentrate on the problem of designing an optimized dual-speed TAM architecture using the port scalability of the ATE. The main contributions of the paper are following:

1. We describe a Genetic Algorithm (GA) based heuristic approach for TAM width partitioning, that provides test solution better than rectangle packing approach as presented in [21]. With a partitioned approach, the test controller also becomes simpler.
2. Studied the effect of power constraints on the overall test time.

The rest of the paper is organized as follows. Section 2 defines the test scheduling problem for optimizing the usage of dual-speed ATE channels. Section 3 determines the lower bound of the proposed solution. Section 4 describes the GA method used for solving the concerned problem.

The impact of dual-speed TAM on test power is discussed in Section 5. Section 6 describes the experimental results obtained and Section 7 draws the conclusion.

2 Optimization with dual-speed ATE channels

This section describes the main techniques employed in our TAM optimization algorithm. In essence, the test scheduling algorithm is developed based on genetic algorithm. The general integrated wrapper/TAM co-optimization and test scheduling problem that we address is as in [21].

$P_{dual\text{-}speed}$: Given the test data parameters for the embedded cores, total SOC-level TAM width W, a total of V available high-speed ATE channels (V <W), and the ratio f of the high-speed data transfer rate to the low-speed data transfer rate, determine the wrapper design, TAM width, and test data rate for each core, and the SOC test schedule such that a) the total number of TAM wires utilized at any moment does not exceed W, b) the number of TAM wires driven at the high data rate does not exceed V, and c) the SOC testing time is minimized.

The test set parameters for each core include the number of primary inputs, primary outputs, bidirectional I/Os, test patterns, scan chains, and scan chain lengths. In our work, the assumptions we made are same as those in [21], such as, cores are hard cores and for a given TAM width and wrapper design for a core, we assume that its testing time at the high data rate is *f* times lesser than its testing time at the low data rate. It is also assumed that a core cannot be connected to both high and low data rate lines at a time. If it takes $T_{ih}(w_i)$ seconds to test Core *i* at the high data rate with a TAM width w_i, the time taken to test it at the low data rate is $T_{il}(w_i)$ seconds, where $T_{il}(w_i) = f \times T_{ih}(w_i)$. The optimization procedure described here can be utilized to determine an efficient TAM architecture for given values of *f* and *V*.

The *Design_Wrapper* algorithm from [4] has been used to design a wrapper and determine the testing time of a core for several possible TAM widths. These testing times include the cases of both the high data rate and low data rate channels. These pre-calculated testing times are subsequently used in the TAM optimization procedure. We formulate the dual-speed TAM optimization problem as follows: Given two sets of TAM widths (with total TAM width *W*), one representing the high data rate TAM lines (V) and the remaining representing the low data rate lines (W-V), partition the TAMs of high data rate and low data rate and assign each core to an appropriate partition such that

i. the maximum TAM width is not exceeded at any time
ii. each core is assigned to either a high speed or a low speed TAM partition
iii. total test time is minimized and
iv. power constraint is honoured.

It is to be noted that a core vendor can specify an upper limit on the scan test frequency for a core. Hence, if this upper limit is lower than the higher data rate of ATE channel then the core can only be tested with lower data rate. Otherwise it can be tested at higher data rate also.

3 Lower Bound on Test Time

In this paper TAM architecture optimization problem is solved based on TAM width partitioning approach. To derive the lower bound on the overall system testing time, we have borrowed the concept from rectangle packing approach [28]. We assume that there can be overlap of the scan-out operation for the last test pattern of a core with the scan-in operation for the first pattern of the next core on the same TAM wire.

From wrapper design algorithm, for a range of TAM widths, the testing times of a core can be represented using a set of rectangles. For each core *i* ($1 \le i \le N$) of an SOC, a set R_i of rectangles is obtained. Now according to the optimization algorithm one rectangle from each R_i ($1 \le i \le N$) has to be selected and packed into a bin of fixed height *W* (maximum available TAM width) such that overall testing time can be minimized and no two rectangles overlap.

If a core *i* is being tested at TAM width *w*, then the area of the rectangle is given by $R_i(w)=T_i(w) \times w$, where $T_i(w)$ is the testing time of the core *i.*

Let $R_i^{min} \in R_i$ be the minimum area rectangle for a core *i,* that is, $R_i^{min}=min_i\{R_i(w)\}$, $1 \le w \le W$. Now if this core is tested at a frequency *f* then the test time of the core will be $T_i(w,f)=T_i(w)/f$. Hence area occupied by this core is given by $R_i(w,f) = T_i(w,f) \times w$, where $0 \le f \le f_i^{max}$. f_i^{max} is the maximum frequency of the core *i.*

The testing time of a core *i* on a TAM partition of width *w* can be expressed as [28]

$$T_i(w) = \left\lceil (\max\{si_i, so_i\} \times p_i + \min\{si_i, so_i\})/w \right\rceil + p_i \quad \text{.....(1)}$$

Where, p_i is the number of test patterns for Core *i* and $si_i(so_i)$ is the length of the longest wrapper scan-in(scan-out) chain obtained from the wrapper design algorithm presented in [4].

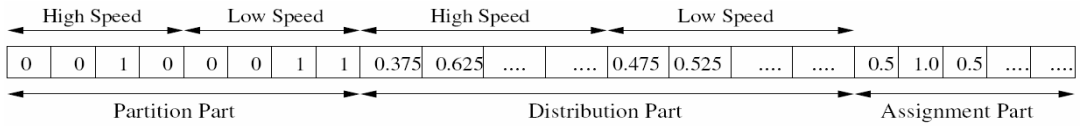

Figure 1: Chromosome Structure

So a lower bound on testing time $T_i(w,f)$ can be expressed as

$$T_i(w,f) \geq \left\lceil (\max\{si_i, so_i\} \times p_i + \min\{si_i, so_i\})/w \right\rceil + p_i) \times 1/f \quad \text{.....(2)}$$

Hence, $R_i(w,f) = T_i(w,f) \times w$

$$= (\left\lceil (\max\{ si_i, so_i\} \times p_i + \min\{ si_i, so_i\}) / w \right\rceil + p_i) \times w / f$$

$$= \lceil (v/w + p_i) \rceil \times w/f \quad \text{.....(3)}$$

Where *v* is the total test data volume to be applied to the core and is independent of number of TAM wires or operating frequency of TAM wires.

Now $R_i(1, f_i^{\max}) = (v + p_i) \times 1/f_i^{\max}$ ……(4)

From the equations of (2), (3) and (4) it can be obtained that for $w>1$ and $f<f_i^{max}$,

$\lceil v/w \rceil \times w/f > v/f^{\max}$ and $p_i \times w/f > p_i/f_i^{\max}$

Hence $R_i(w,f) > R_i(1, f_i^{\max})$, for $w \geq 1$ and $f < f_i^{\max}$.

If *T* is the total testing time and *W* is the total assigned TAM wires to the SOC, total rectangle area is $T \times W$. But, if we pack the representative rectangles with minimum area for each core then definitely there will be some unfilled area in the bin. Hence we can easily write

$$T \times W \geq R_1(1, f_1^{\max}) + R_2(1, f_2^{\max}) + ... + R_N(1, f_N^{\max}) + \partial$$

, where ∂ being the unfilled area.

$$\Rightarrow T \times W \geq R_1(1, f_1^{\max}) + R_2(1, f_2^{\max}) + ... + R_N(1, f_N^{\max})$$

$$\Rightarrow T \geq \sum_{i=1}^{N} R_i(1, f_i^{\max})/W$$

Hence lower bound $LB_1 = \sum_{i=1}^{N} R_i(1, f_i^{max})/W$. This lower bound is more accurate for smaller *W* values. For higher *W* values we have utilized another lower bound from [29] $LB_2 = max_j\{min_i\ T_i\ (w, f_i^{max})\}$, $1 \leq i \leq N$ and $1 \leq j \leq N_B$, where N_B is the number of TAM partitions. Hence overall lower bound $LB_T = max\ \{LB_1, LB_2\}$. The lower bound values calculated for different SOCs and varifying TAM width have been noted in Table 2.

4 Test Scheduling Using Genetic Algorithm

Genetic algorithms (GA) [23] are stochastic optimization search algorithms based on the mechanics of natural selection and natural genetics. The genetic formulation of our problem involves the careful and efficient choice of the following.

a. A proper encoding of the solutions to form chromosomes.
b. To decide upon a crossover operator.
c. To identify a proper mutation operator.

Cost function for measuring the fitness of the chromosome in a population.

4.1 Solution representation

A chromosome for the given problem conceptually consists of three parts. But first two parts basically consist of again two parts each with same kind of information having different sizes. Hence total number of parts in the chromosome is five as shown in Fig. 1.

First part named as *partition part* is the number of TAM partitions. The sub-part 'High Speed' of it is the number of partitions for the high-speed TAM of width V. Since the total TAM width V for high-speed can be encoded with a binary string of $(log_2V + 1)$ bits, this partition part is an array of size $(log_2V + 1)$. Second sub-part is the number of TAM partitions for the low-speed TAM channels of width (W-V), where W is the maximum allowable TAM width (including high-speed and low-speed) for an SOC. Since the total low-speed TAM width (W-V) can be encoded with a binary string of $(log_2(W-V) + 1)$ bits, this part is an array of size $(log_2(W-V) + 1)$.

The *distribution part* gives the width of each partition of the high-speed and low-speed TAMs. It consists of two components – 'High Speed' and 'Low Speed'. The 'High Speed' part is an array of real numbers between *0* and *1*, where the j^{th} entry of this array multiplied by the high-speed TAM width (V) represents the width of the bus *j*. The array size is equal to the decimal value of the first part. However to keep the chromosome length fixed, we have used W entries here, only the first few are actually used (depending upon the value in partition part). Similarly fourth part (also called the *distribution part* for low-speed TAM) gives the partition of the width (W-V) among the low-speed

TAMs. This is also an array of real numbers between *0* and *1*, where the j^{th} entry of this array multiplied by the TAM width (W-V) represents the width of the bus *j*. The array size is equal to the decimal value of the second part.

Fifth part is the *assignment part.* This is also an array of real numbers between 0 and 1, where the j^{th} entry of the array multiplied by the decimal value of the sum of high speed and low speed components of a partition part represents the bus assigned to the j^{th} core. The array size is equal to the total number of cores in the SOC. These five parts of the chromosome form a solution to the given problem.

Example:

For example, in Fig. 1 number of bits in partition part of high speed TAM is 4 and number of partitions is *2 (0010)*. So, in the distribution part of high speed TAM, first two entries will be considered and the respective TAM widths are *(0.375×V)* and *(0.625×V).* Similarly, number of bits in the partition part for low speed TAM is 4 and number of partitions is 3 *(0011).* So, in the distribution part of low speed TAM first three entries will be considered. Fifth part is the assignment part. Size of this part is equal to the number of cores in the SOC. As the total number of partitions i.e., summation of partition part of high speed and low speed is 5 and first value in the assignment part is 0.5, hence TAM number *1 (0.5×2=1)* of width *0.375×V* is assigned to core 1 and so on.

4.2 Genetic operators

Two genetic operators, crossover and mutation have been used to evolve new generations.

4.2.1 Crossover

Our GA formulation has been biased towards selecting the chromosomes with better fitness to participate in crossover. For this purpose, the whole population is sorted according to their fitness values. A certain percentage of population with better fitness value is defined to be the "Best Class". To select a chromosome participating in crossover, first a uniform random number between 0 and 1 is generated. If the number is greater than 0.5, a chromosome from the "Best Class" is selected randomly. Otherwise, a chromosome from the entire population is selected. After selecting two chromosomes to participate in crossover, a single point crossover is applied on each of the *partition part*, *distribution part* and the *assignment part* of the chromosome.

4.2.1 Mutation

The mutation operator brings more effective variations into the chromosomes introducing newer search options. To select a chromosome participating in mutation, a uniform random number between 0 and 1 is generated. If the number is greater than 0.5, a chromosome from the "Best Class" is selected randomly. Otherwise, a chromosome from the entire population is selected. Then we select a random point in each of the five fields of the chromosome and change its value. For the first field, we complement a randomly selected bit among the least significant *log W* bits, W being the total TAM width. For the second and third fields we replace with randomly generated values in the range 0 to 1. For the distribution part, normalization is carried out to ensure the unity sum requirement.

4.3 Fitness measure

Fitness of the chromosome is measured in terms of the cost of a solution, which is the total time required to test all cores in the system and is explained as follows:

Consider an SOC consisting of N cores with B_h *and* B_l *being the number of partitions for high-speed and low-speed TAMs respectively. Hence* B (= B_h + B_l) *is the total number of TAM partitions.*

Different TAMs can be used simultaneously for delivering test data to the cores while the cores assigned to the same TAM are tested sequentially. It is to be noted that a core cannot be assigned to both high-speed and low-speed TAMs. So, total core testing time for a TAM is the sum of the testing times for all the cores assigned to it and the total time to test all the cores in the SOC is the maximum of the times taken among the TAMs. The testing time $T_{ih}(w_j)$ for a core *i* assigned to a high-speed TAM of width w_j is calculated using (1).

To get the test time needed to test all the cores on TAM *j* we use binary variables x_{ij} (where $1 \le i \le N$ and $1 \le j \le B_h$) to determine the assignment of cores to high-speed TAMs in the SOC. Let variable x_{ij} be a *0* or *1*defined as follows:

$x_{ij} = 1$, if *Core i* is assigned to high-speed TAM *j*

$= 0$, otherwise.

Hence, the test time required to test all cores on TAM *j* is given by $\sum_{i=1}^{N} T_{ih}(w_j) \times x_{ij}$.

Similarly for low-speed TAMs the time required to test all cores on TAM k is given by $\sum_{i=1}^{N} T_{il}(w_k) \times y_{ik}$, where y_{ik} ($1 \le i \le N$ and $1 \le k \le B_l$) is the binary variable that determines the assignment of cores to low-speed TAMs in the SOC and be defined as

$y_{ik} = 1$, if *Core i* is assigned to low-speed TAM *k*

$= 0$, otherwise.

Since all the TAMs can be used simultaneously for testing, the system testing time equals

$$T = \max_{\{1\le j\le B_h, 1\le k\le B_l\}} \left\{ \sum_{i=1}^{N} T_{ih}(w_j) \times x_{ij}, \sum_{i=1}^{N} T_{il}(w_k) \times y_{ik} \right\} \quad \text{.....(5)}$$

4.4 Evolution process

Initial population of size 3000 is taken. First 20 % of the chromosomes with less test time are taken as the "best class chromosomes". We copy the best class chromosomes directly and select 80% chromosomes by crossover. This new population is sorted according to their fitness values. Now we copy 80% directly to the next generation and select 20% via mutation as the population for the next generation, keeping the population size fixed. The resulting population is sorted according to their fitness values. The population thus obtained by operating both the genetic operators is termed as new generation. This process is repeated up to certain predefined number of generations. For our experimentation we took 500 generations to obtain the results.

5 Impact of Dual-Speed TAM on Test Power

It is known that power consumption of a circuit plays a vital role for normal operation of the circuit. During testing, power consumption is much higher than that during the normal operation. Hence excessive power consumption in an SOC during scan testing can cause overheating, which may lead to the damage of the chip [24]. It is important to find out the effect of the TAM design on power consumption during test and also at the same time it is required to reduce the testing time. In this section we studied the impact of dual-speed TAM architecture on overall SOC test power. As power consumption is directly proportional to the operating frequency of the circuit [25], the use of high-speed TAM though reduces the test time significantly can cause an increase in power consumption during scan testing. We studied the impact of using high-frequency TAM lines on test power for one of the ITC '02 benchmark circuits. Since power estimation models for the ISCAS benchmark circuits have been published in the literature, we utilize one of the SOC benchmarks that consist of ISCAS circuits.

Though during test, power consumption varies over time [11], to simplify, we assume peak power value attached to each test. Although these power values are for functional patterns, we use these values for scan test power due to the lack of any additional power information for these circuits. The power of a core scheduled on the high-frequency TAM is calculated as the frequency factor *f* times its power for the low-frequency TAM. The peak power is measured as peak switching frequency (PSF) per node [26], [7] and the average power is measured in mW [27]. Table 1 shows the peak power data that is considered for each core in d695. The following power schedule algorithm (*Algorithm 1)* is used to estimate the test time of the SOCs under power constraints. This has been utilized to evaluate the fitness of individual chromosomes in the population of GA.

Algorithm 1

Power_schedule()

Input:

1. A chromosome identifying TAM assignments to cores.
2. Power values of each core.
3. Maximum power budget (P_{max}).

Output:

Final test time and complete schedule after placing all cores in the list satisfying power constraint.

Data Structure:

time_instants

/* Global list of tuples like <*time_instant, power_value*>. The entry *time_instant* represents a particular instant of time and *power_value* represents the power consumed by the cores scheduled for testing at that particular instant.*/

Begin

time_instants = NULL.
Schedule first core as specified by the chromosome;
time_instants = time_instants ∪
{<0,core_test_power>} ∪ {<core_test_time,0>};
While (*list of cores not null*)
  R = Take next core from the list;
  TAM_i = TAM assigned to R (depicted by the chromosome);
  For each free time interval T_{gap} on TAM_i do
    If $T_{gap} \geq$ Test time of *R* then
      Check for power constraints at each entry in time_instants list in the range of T_{gap}.
      If power constraint met then
        Update time_instants list;
        Schedule the core *R* at T_{gap};
        Update the list of cores scheduled on TAM_i;
        Break from loop;
  If no such time gap exists
    Determine the next available time instant after the last core scheduled so far on TAM_i, such that power constraint is met.
    Update time_instants list;

Schedule the core *R* from available time instant.

Update the list of cores scheduled on TAM_i; Return the maximum time amongst all TAMs.
End

Table 1: Peak Power Data for ISCAS-85 cores in d695

Core	Peak power (PSF)
c432	660
c499	602
c880	823
c1355	275
c1908	690
c2670	354
c3540	530
c5315	753
c6288	641
c7552	1144

6 Experimental Results

In this section, we present experimental results on test scheduling for the three largest SOCs (in terms of the number of cores) from the ITC '02 SOC Test Benchmarks [22]. All the tests were conducted on a 2.66 GHz Pentium IV machine with 512 MB memory. The proposed algorithm is implemented in C++. The testing time is calculated in μs, the low-speed TAM lines are assumed to be driven at 20 MHz, and the high-speed TAM lines are assumed to be driven at 20*f* MHz for different values of *f*. The following sections show the test time results with and without power constraints.

6.1 Test scheduling without power consideration

In Table 2, we present the testing time for various values of TAM widths W and a range of values for V. Number of TAM wires driven by the high-speed ATE channels is represented in terms of n, where n is the % of total TAM lines considered as high speed. We also assume in this set of experiments that the high-speed data rate is twice that of the low-speed data rate, i.e., $f = 2$. Table 2 also gives a comparison with the testing time of [21]. The values corresponding to LB are the lower-bound values as described in Section 3. The maximum improvement in test application time was found to be 40.99 % for p93791 with a TAM width of W=32 among which 25% of the TAM lines are high speed. In some cases it can be observed from Table 2 that the test time produced by our method is a bit higher as compared to the one in [21]. For example, for core p22810 with *W=48, n=25%*, test time of our approach is *0.46%* higher than [21]. This happens due to the small number of TAM partitions for low speed TAMS. For this case out of 36 (for n=25%) low speed TAM lines only 2 partitions have been made. These are of widths 5 and 31. Similarly, for cores p34392 with *W=32, n=100%* and *n=0%* the results are *0.43%* poorer for both cases. However, on an average, the partitioning approach based on GA performs better than the bin-packing approach of [21]. This is depicted in Table 3.

Table 3 depicts the % improvement in test time over [21] (rows marked with [21]) and also with lower bound results (rows marked with `LB'). From Table 3(a) it can be noted that the maximum average improvement of 9.64% in test time is obtained when *n=25%.* But in this case also test time results are on an average 15% higher than the lower bound results. So, better heuristic method can be applied to minimize the difference between the lower-bound and SOC test time. From Table 3(b) it can be noted that SOC test time results are close to the lower-bound values. From W=40 onwards for almost all of the cases test time results and lower-bound values are same. Again from Table 3(c) it is noted that maximum average improvement of 24.39% is achieved for *n=25%.* For this SOC (p93791) also test time results are closer to the lower-bound values (within the limit of 6%). Results of Table 3 have been summarized in Figure 2, that represents the % average improvement in test time over [21] for the SOCs and on an average how far from lower bound the results are.

6.2 Test scheduling with power consideration

In this section, we present results for power-constrained test scheduling for d695. The results are presented for power limits of 2,500 PSF and 3,000 PSF for *n = 25%* and *n = 50%* and with no high speed lines (*n=0%*). In Table 4, we compare the test time (TP_1 and TP_2 for power limits of 2,500 PSF and 3,000 PSF, respectively) of the power-constrained test schedule for different values of *W* to that of unconstrained test schedule TP_0. It is important to note that, in the case of power-constrained test scheduling, we do not attempt to optimize the test schedule for test power. Rather the goal here is to minimize the test time under power limits. It is seen from the table that for most of the TAM widths our method provides better results than the heuristic approach in [21]. It is also to be noted from Table 4 that for *n=25%* and *n=50%* improvement in test time is more than the case of *n=0%.* Maximum average improvement of *40.05%* in test time is obtained for *n=50%* with power limit of 2500 PSF.

Table 2
Testing Time Results (Flat core case) for (a) p22810 (f = 2) (b) p34392 (f = 2) (c) p93791 (f = 2)

p22810										
T_n	n=100%		n=75%		n=50%		n=25%		n=0%	
	Method [21]	Ours	Method [21]	Ours	Method [21]	Ours	Method [21]	Ours	Method [21]	Ours
W=16	11315.97	**10960.92**	14652.25	**12755.62**	17236.85	**14404.20**	25329.25	**17617.50**	22631.95	**21870.00**
LB	**10313.47**		**11835.15**		**13696.46**		**16339.55**		**20626.95**	
W=24	7694.45	**7559.625**	9987.25	**8685.00**	10838.75	**10290.60**	14807.95	**12316.60**	15389.00	**15258.90**
LB	**6966.75**		**7864.74**		**9391.17**		**10984.85**		**13932.05**	
W=32	6153.75	**5844.925**	6966.00	**6789.60**	8641.20	**7666.10**	10437.95	**9781.30**	12307.50	**11582.25**
LB	**5156.74**		**5775.39**		**7270.85**		**8179.46**		**10313.47**	
W=40	4932.37	**4865.65**	5890.00	**5444.325**	6966.00	**6714.60**	7914.70	**7776.10**	9864.65	**9617.3**
LB	**4125.39**		**5150.57**		**5805.78**		**7270.43**		**8250.78**	
W=48	4181.35	**3976.70**	5159.30	**4833.425**	6069.95	**5626.90**	6691.10	6722.15	8362.80	8553.20
LB	**3635.40**		**3962.49**		**5155.12**		**5399.65**		**7270.85**	
W=56	3635.40	3635.40	4393.70	**4219.20**	5150.55	**5148.25**	6424.55	**5944.35**	7270.85	7270.85
LB	**3635.40**		**3635.40**		**5148.25**		**4617.65**		**7270.85**	
W=64	3423.45	**3415.30**	3985.75	**3864.37**	4915.35	**4175.55**	5531.70	**5257.90**	6847.05	**6833.20**
LB	**3339.67**		**3635.40**		**3649.05**		**4085.38**		**6679.35**	

(a)

p34392										
T_n	n=100%		n=75%		n=50%		n=25%		n=0%	
	Method [21]	Ours	Method [21]	Ours	Method [21]	Ours	Method [21]	Ours	Method [21]	Ours
W=16	25592.45	**24969.62**	33798.55	**28117.95**	42318.60	**32301.78**	51660.45	**39973.95**	51191.00	**49939.25**
LB	**23422.03**		**28117.95**		**32371.78**		**39152.02**		**46844.06**	
W=24	18021.35	**16911.95**	22469.20	**18962.40**	25830.20	**23444.55**	31902.15	**28646.70**	37971.35	**33826.35**
LB	**16579.82**		**18745.90**		**22118.87**		**25587.78**		**31229.37**	
W=32	13614.45	13673.45	18985.65	**14931.725**	18404.10	**17655.75**	21813.80	**20775.15**	27228.95	27346.90
LB	**13614.45**		**14690.22**		**16579.82**		**18620.49**		23422.03	
W=40	13614.45	13614.47	13614.45	13614.45	14703.20	14703.20	17008.75	**16689.30**	27228.95	27228.95
LB	**13614.45**		**13614.45**		**14703.20**		**15879.45**		**27228.95**	
W=48	13614.45	13614.45	13614.45	13614.45	14703.20	14703.20	14889.85	**14703.20**	27228.95	27228.95
LB	**13614.45**		**13614.45**		**14703.20**		**14703.20**		**27228.95**	
W=56	13614.45	13614.45	13614.45	13614.45	13614.45	13614.45	14703.20	14703.20	27228.95	27228.95
LB	**13614.45**		**13614.45**		**13614.45**		**14703.20**		**27228.95**	
W=64	13614.45	13614.45	13614.45	13614.45	13614.45	13614.45	14703.20	14703.20	27228.95	27228.95
LB	**13614.45**		**13614.45**		**13614.45**		**14703.20**		**27228.95**	

(b)

p93791										
	n=100%		n=75%		n=50%		n=25%		n=0%	
T_n	Method [21]	Ours	Method [21]	Ours	Method [21]	Ours	Method [21]	Ours	Method [21]	Ours
W=16	46278.15	**43747.72**	60845.35	**49966.07**	85952.05	**58301.275**	102033.40	**69828.85**	92556.75	**87189.20**
LB	**42677.38**		**48838.55**		**57130.59**		**67939.70**		**85354.75**	
W=24	31135.45	**29681.70**	43795.65	**33806.80**	60795.50	**39269.975**	76184.00	**47122.50**	62439.75	**59471.50**
LB	**28451.58**		**32585.88**		**37792.67**		**45486.94**		**56903.17**	
W=32	24375.40	**21972.725**	31135.45	**25221.825**	37761.05	**29723.97**	59911.05	**35351.45**	48750.80	**43947.20**
LB	**21338.69**		**24189.69**		**28411.75**		**34319.71**		**42677.38**	
W=40	19850.50	**17927.80**	23184.00	**20253.55**	28588.90	**24647.075**	33035.45	**28541.80**	39701.00	**36225.30**
LB	**17070.93**		**19378.59**		**22550.58**		**27424.185**		**34141.90**	
W=48	15698.10	**15007.10**	20691.35	**17185.15**	23665.90	**20079.57**	28436.15	**24078.70**	31396.70	**29833.15**
LB	**14225.79**		**16201.07**		**18984.05**		**22799.43**		**28451.58**	
W=56	14210.90	**12965.05**	17915.55	**15041.12**	20383.35	**17710.075**	24223.55	**21710.50**	28421.80	**25895.95**
LB	**12193.54**		**13704.15**		**15766.08**		**19700.77**		**24387.07**	
W=64	12782.15	**11483.875**	15727.15	**12979.50**	16806.55	**15488.625**	22794.75	**18061.30**	25564.30	**22968.05**
LB	**10669.34**		**12120.14**		**14066.125**		**17240.74**		**21338.69**	

(c)

Table 3

% Average improvement in test time results over [21] and lower bound values

W	Comparison with	n=100%	n=75%	n=50%	n=25%	N=0%
16	**[21]**	3.13	12.94	16.43	30.45	3.37
	LB	-5.90	-7.21	-4.91	-7.25	-5.68
24	**[21]**	1.75	13.04	5.05	16.82	0.84
	LB	-7.84	-9.45	-8.74	-10.81	-8.69
32	**[21]**	5.02	2.53	11.28	6.29	5.89
	LB	-11.77	-14.94	-5.16	-16.37	-10.95
40	**[21]**	1.35	7.57	3.60	1.75	2.50
	LB	-15.21	-5.39	-13.53	-6.50	-14.21
48	**[21]**	4.89	6.32	13.29	-0.46	2.89
	LB	-8.58	-18.01	-8.38	-19.67	-10.47
56	**[21]**	0	3.97	0.04	7.47	0
	LB	0	-13.84	0	-22.32	0
64	**[21]**	0.23	3.04	15.05	4.94	0.20
	LB	-2.21	-5.93	-12.61	-22.3	-2.25
Avg. Impr. over	**[21]**	2.34	7.06	9.25	9.64	2.24
	LB	-7.36	-10.68	-7.62	-15.03	-7.46

(a) p22810

W	Comparison with	n=100%	n=75%	n=50%	n=25%	N=0%
16	**[21]**	2.43	16.8	23.67	22.62	2.44
	LB	-6.19	0	0	-2.06	-6.19
24	**[21]**	2.02	15.60	9.23	9.24	7.26
	LB	-1.96	-1.94	-5.65	-10.68	-7.68
32	**[21]**	-0.43	21.35	4.07	4.85	-0.43
	LB	-0.43	-1.62	-6.09	-10.37	-0.43
40	**[21]**	0	0	0	1.87	0
	LB	0	0	0	-4.85	0
48	**[21]**	0	0	0	1.25	0
	LB	0	0	0	0	0
56	**[21]**	0	0	0	0	0
	LB	0	0	0	0	0
64	**[21]**	0	0	0	0	0
	LB	0	0	0	0	0
Avg. Impr. over	**[21]**	0.57	7.68	5.28	5.69	1.32
	LB	-1.23	-0.51	-1.68	-3.99	-2.04

(b) p34392

W	Comparison with	n=100%	n=75%	n=50%	n=25%	N=0%
16	**[21]**	5.46	17.88	32.17	31.56	5.80
	LB	-2.45	-2.26	-2.01	-2.70	-2.10
24	**[21]**	4.67	22.81	35.41	38.15	4.75
	LB	-4.14	-3.61	-3.76	-3.47	-4.32
32	**[21]**	9.85	18.99	21.28	40.99	9.85
	LB	-2.89	-3.72	-4.41	-2.92	-2.89
40	**[21]**	9.68	12.63	13.79	13.60	8.75
	LB	-4.78	-4.32	-8.50	-3.92	-5.75
48	**[21]**	4.40	16.94	15.15	15.32	4.82
	LB	-5.21	-5.73	-5.46	-5.31	-4.63
56	**[21]**	8.77	16.04	13.11	10.37	8.89
	LB	-5.95	-8.89	-10.97	-9.26	-5.83
64	**[21]**	10.16	17.47	7.84	20.76	10.15
	LB	-7.09	-6.62	-9.18	-4.54	-7.09
Avg. Impr. over	**[21]**	7.57	17.53	19.82	24.39	7.57
	LB	-4.64	-5.02	-6.33	-4.59	-4.66

(c) p93791

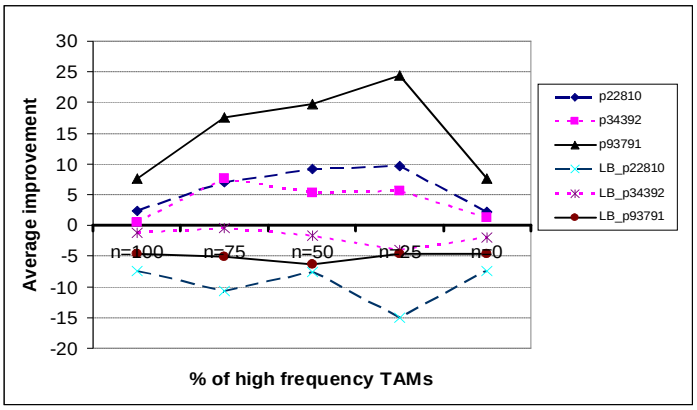

Figure 2: Comparison of average % improvement over [21] and lower bound (LB) results

Table 4
Testing Time for SOC d695 in µs, with Power Limits of 2,500 PSF and 3,000 PSF for (a) n=0% (b) n = 25% (c) n = 50%

n=0%						
W	TP_0		TP_1		TP_2	
	Method [21]	Ours	Method [21]	Ours	Method [21]	Ours
16	2146.95	**2113.40**	2146.95	**2121.40**	2146.95	**2113.40**
24	1463.55	**1414.60**	1463.55	**1419.45**	1463.55	**1414.60**
40	884.50	**883.85**	898.75	**892.80**	884.50	**883.85**
48	743.65	**733.30**	789.35	848.80	762.50	848.75
56	621.05	644.20	668.80	**647.05**	647.05	**646.75**
64	580.20	**521.70**	624.80	**548.75**	618.80	**521.70**
% Avg. impr.	**2.14**		**2.21**		**1.76**	

(a)

n=25%						
W	TP_0		TP_1		TP_2	
	Method [21]	Ours	Method [21]	Ours	Method [21]	Ours
16	2786.10	**1690.75**	3016.35	**1690.75**	2786.10	**1690.75**
24	1738.30	**1145.20**	1738.25	**1145.20**	1738.30	**1145.20**
32	987.85	**878.35**	1207.85	**878.35**	1110.60	**878.35**
40	881.20	**682.60**	1070.60	**693.40**	917.90	**682.60**
48	623.50	**582.25**	815.45	**588.65**	876.60	**582.25**
56	551.65	**503.45**	704.95	**517.35**	625.60	**503.45**
64	531.25	**449.5**	704.95	**470.40**	587.20	**449.5**
% Avg. impr.	**19.69**		**32.60**		**31.02**	

(b)

n=50%						
W	TP_0		TP_1		TP_2	
	Method [21]	Ours	Method [21]	Ours	Method [21]	Ours
16	1724.50	**1399.80**	2117.65	**1405.00**	1928.30	**1399.80**
24	1131.25	**958.25**	1466.30	**964.70**	1202.75	**958.25**
32	801.15	**733.50**	1145.20	**751.50**	894.60	**733.50**
40	672.55	**572.55**	1031.55	**587.55**	788.95	**572.55**
48	551.65	**493.45**	927.80	**501.35**	682.30	**493.45**
56	521.70	**475.85**	848.75	**493.45**	639.10	**493.45**
64	464.10	**374.20**	767.90	**405.00**	602.35	**374.20**
% Avg. impr.	**13.73**		**40.05**		**25.93**	

(c)

TP_0: Testing time with no power limit; TP_1: Testing time with power limit of 2500 PSF; TP_2: Testing time with power limit of 3000 PSF

7 Conclusion

In this paper we have presented a Genetic algorithm based test scheduling approach for minimizing testing time considering two different data rates for ATE channels. We use the SOC-level TAM architecture optimization by partitioning TAMs and then assigning cores to the TAMs. Experimental results have been presented for ITC '02 benchmark circuits and compared with the results of [21]. We also consider the impact of dual-speed TAMs on power consumption and the effect of power consumption on overall SOC test time. The future works include how this dual speed TAM access architecture can be used for the hierarchical cores. As the core based SOC design utilizes the reuse philosophy, the current generation SOCs can be embedded into next generation SOCs.

References:

[1] R. Dorsch et al. Adapting an SoC to ATE concurrent test capabilities. *Proc. Int. Test Conf.,*2002, pp.1169–1175.

[2] Agilent Technologies. Winning in the SOC market, available online 16243240485664 at: http://cp.literature.agilent.com/litweb/pdf/5988-7344EN.pdf.

[3] Teradyne Technologies. Tiger: Advanced digital with silicon germanium technology. http://www.teradyne.com/tiger/digital.html.

[4] V. Iyengar, K. Chakrabarty and E. J. Marinissen. Test wrapper and test access mechanism co-optimization for system-on-chip. *Journal of Electronic Testing: Theory and Applications,* vol.18, April, 2002 pp.213–230.

[5] K. Chakrabarty, *Test Scheduling for core-based systems using Mixed-Integer Linear Programming*, *IEEE Transaction on Computer Aided Design*, 2000, pp. 1163-1174.

[6] Y. Huang et al, *Resource allocation and test scheduling for concurrent test of core-based SOC design,* In *proc. Asian Test Symposium(ATS)*, 2001, pp:265-270.

[7] Y. Huang, S. M. Reddy, W.-T. Cheng and P. Reuter. Optimal core wrapper width selection and SOC test scheduling based on 3D-bin packing algorithm, In *Proc. Intl. Test Conference(ITC)*, Baltimore, 2002, pp:74-82.

[8] C. P. Ravikumar, A. Verma, and G. Chandra. A polynomial time algorithm for power constrained testing of core-based systems, In proc. *Asian Test Symposium(ATS)*, 1999, pp:107-112.

[9] Y. Xia, M. Chrzanowska-Jeske, B. Wang and M. Jeske, Using a Distributed Bin-Packing Approach for Core-based SOC Test Scheduling with Power Constraints, In *Proc. Intl. Conf. Computer Aided Design*, 2003, pp:100-105.

[10] C.-P. Su and C.-W Wu. A Graph-Based Approach to Power-Constrained SOC Test Scheduling, In *Journal of Electronic Testing: Theory and Applications*, pp. 45-60, 2004.

[11] J. Pouget, E. Larsson and Z. Peng, Multiple-Constraint Driven System-On-Chip Test Time Optimization, In *Journal of Electronic Testing: Theory and Applications*, Vol:21, 2005, pp: 599-611.

[12] D. Zhao and S. Upadhyay. A Generic Resource Distribution and Test Scheduling Scheme for Embedded Core-Based SOCs, In *IEEE Trans.. Instrumentation and Measurement*, vol. 53, no. 2, 2004, pp. 318-329.

[13] V. Iyengar. and K. Chakrabarty. Precedence-Based, Preemptive, and Power constrained Test Scheduling for System-On-Chip, In *Proc. VLSI Test Symposium(VTS),* 2001, pp: 368-374.

[14] S. Koranne, and V. Iyengar. On the use of k-tuples for SOC test schedule representation, In *Proc. Intl. Test Conference(ITC)*, 2002, pp. 539-548.

[15] D. Zou, S. M. Reddy, I. Pomeranz and Y. Huang. *SOC Test Scheduling Using Simulated Annealing*, In *Proc. VLSI Test Symposium(VTS)*, 2003, pp. 325-330.

[16] J. –Yi. Wuu, T.-C. Chen and Y.-W. Chang, SOC Test Scheduling Using B*-tree Based

Floor planning Technique, In *Proc. Asia South Pacific- Design Automation Conference (ASP-DAC)*, 2005, pp. 1188-1191.

[17] E. Larsson, and H. Fujiwara, System-On-Chip Test Scheduling with reconfigurable core wrappers, In *IEEE Trans. VLSI Systems*, Vol. 14, No. 3, March 2006, pp:305-309.

[18] J.-Ho Ahn and S. Kang. SoC Test Scheduling Algorithm Using ACO-Based Rectangle Packing, In *Proc. International Conference on Intelligent Computing(ICIC)*, August 2006, pp. 655-660.

[19] Y. Yu, X. Y Peng, and Y. Peng, A Test Scheduling Algorithm Based on Two-Stage GA, In *Proc. International Symposium on Instrumentation Science and Technology*, 2006, pp: 658-662.

[20] A. Sehgal et al. Test cost reduction for SOCs using virtual TAMs and Lagrange Multipliers, *Proc. Design Automation Conference(DAC)*, 2003, pp:738-743.

[21] A. Sehgal and K. Chakrabarty, Optimization of Dual-Speed TAM Architectures for efficient Modular Testing of SOCs, *In IEEE Trans. on Computers*, Vol. 56, No. 1, January 2007, pp:120-133.

[22] E. J. Marinissen, V. Iyengar, and K. Chakrabarty. A set of benchmarks for modular testing of SOCs, In *Proc. Int. Test Conference(ITC).*, 2002, pp.519–528.

[23] Melanie Mitchell, An Introduction to Genetic Algorithm, Prentice Hall India.

[24] Y. Bonhomme, P. Girard, C. Landrault, and S. Pravossoudovitch, Test Power: A Big Issue in Large SOC designs. *Proc. DELTA Workshop*, 2002, pp. 447-449.

[25] N. Nicolici and B.M. Al-Hashimi, Power-Constrained Testing of VLSI Circuits, *Kluwer Academic*, 2003.

[26] M.S. Hsiao, E.M. Rudnick, and J.H. Patel, Effects of Delay Models on Peak Power Estimation of VLSI Sequential Circuits, *Proc. Int'l Conf. on Computer-Aided Design (ICCAD),* 1997, pp. 45-51.

[27] R. Burch, F. Najm, P. Yang, and T. Trick, A Monte Carlo Approach for Power Estimation, *IEEE Trans. VLSI Systems*, vol. 1, Mar. 1993, pp. 63-71.

[28] A. Sehgal and K. Chakrabarty, Test Planning for the Effective Utilization of Port-Scalable Testers for Heterogeneous Core-Based SOCs, in *Proc. Int'l Conf. on Computer-Aided Design (ICCAD)* , 2005, pp:88-93.

[29] K. Chakrabarty, Optimal Test access Architecture for system-on-a-chip, *ACM Trans. Design Automation of Electronic Systems*, vol. 6, January 2001, pp: 26-49.

[30] J.-c. Rau, W.-t. Huang and C.-l. Chien, The optimal Test-Rail Architecture for Core-based SOC Testing, *WSEAS Trans. On Circuits and Systems,*3(4), May 2004.

[31] P. Sakthivel, P. Narayanasamy, Genetic Algorithm-based Design and Optimization of SOC Test Solutions, *WSEAS Trans.On Circuits and Systems,* 8(5), August 2006, pp:1212-1219.

[32] S. Koranne, A novel Reconfigurable Wrapper for Testing of Embedded Core-Based SOCs and its Scheduling Algorithm, *Journal of Electronic Testing: Theory and Applications,* 18(4/5), August 2002, pp:415-434.

[33] E. Larssen and H. Fujiwara, System-On-Chip Test Scheduling with Reconfigurable Core Wrappers, *IEEE Trans. On Very Large Scale Integration (VLSI) Systems,* 14(3), March 2006, pp:305-309.

[34] S. Bhatia, T. Gheewala and P. Varma, A Unifying Methodology of Intellectual Property and Custom Logic Testing, *In Proc. International Test Conference,* October 1996, pp:639-648.

[35] L. Whetsel, An IEEE 1149.1 Based Test Access Architecture for ICs with Embedded Cores, *In Proc. International Test Conference,* November 1997, pp:69-78.

[36] J. Aerts and E. J. Marinissen, Scan chain Design for Test Time reduction in Core-based ICs. *In Proc. International Test Conference,* October 1998, pp:448-457.

[37] P. Varma and S. Bhatia, A Structured Test Re-use Methodology for Core Based System Chips, *In Proc. International Test Conference,* October 1998, pp:294-302.

[38] E. J. Marinissen, R. Arendsen, G. Bos, M. L. Dingemanse, and C. Wouters, A Structured and Scalable Mechanism for Test Access to Embedded Reusable Cores. *In Proc. International Test Conference,* October 1998, pp:284-293.

[39] C. Giri, S. Sarkar, and S. Chattopadhyay, Test Scheduling for Core-Based SOCs Using Genetic Algorithm Based Heuristic Approach, *Proc. Intl. Conference on Intelligent Computing*, 2007, pp:1032-1041

[40] W.-D. Tseng, T.-F. Chien, A Self-Test Structure for Crosstalk Fault Test in SOC buses, *WSEAS Trans. On Circuits and Systems,* 4(6), April 2007.