Testing High-Speed SoCs Using Low-Speed ATEs

Mehrdad Nourani
Center for Intergrated Circuits and Systems
The Univ. of Texas at Dallas
Richardson, TX 75083
nourani@utdallas.edu

James Chin
Texas Instruments, Inc.
Dallas, TX 75243
jchin@ti.com

Abstract

We present a test methodology to allow testing high-speed circuits with low-speed ATEs. The basic strategy is adding an interface circuit to partially supply test data, coordinate sending the test patterns and collecting the signatures. An ILP formulation is presented to globally optimize such coordination in terms of the overall test time and the hardware cost.

1 Introduction

■ Motivation

Rapid increase in the speed of new VLSI circuits and systems brought with itself many challenges for their testing. One of these challenges is at-speed testing to assure the performance of the circuit. This requires specifically test pattern transfer at its intended operating speed. On the other hand, the high cost of testers, also called ATEs (automatic test equipments), made it impossible to follow the design speed. In general, the cost of ATE depends on speed, memory and pin count.

The speed of systems doubles every 18 months on the average but the life time of a mid-range ATE is considered 2 to 4 years [1]. Therefore, at a given time, many designers have access to ATEs whose speeds is behind the CUT (circuit under test) speed. A CUT can be as small as an ASIC (application specific integrated circuit) or as large as a complicated SoC (system-on-chip). As a result of this speed difference, the high-speed circuits are currently tested at clock rates that are much slower than their specification [2]. This difficulty motivated us to work on a test scheme and architecture to alleviate the problem of speed mismatch during test.

■ Background

There have been some works in the literature to address concerns about the ATE limitations and usage techniques. Built-in self-test (BIST) techniques, in general, does not suffer from the low-speed ATE. In this method, the main circuitry for test pattern generation (TPG) and signature analysis (SA) are located within the chip and thus none or very little interaction with an external ATE is needed [3]. Unfortunately, due to the random nature of the generated patterns and the lossy response compaction the method does not provide an adequate diagnosis or debug capability [4]. Moreover, at-speed BIST is very expensive and consequently it is not widely used.

Various compression techniques have been used to reduce the volume of test data needed to be traded between ATE and CUT. Some examples are using Huffman encoding to reduce memory requirement [5], statistical coding for scan vector compression/decompression [6] and Golomb codes for test data compression [7].

Double strobe flip-flop, explained in [8], is a technique to test a scan circuit at low speed and yet verify its high speed performance. Using electro-optic techniques, authors in [9] have described how to use ATE to test very high-speed CUT. Authors in [10] used the idea of pin multiplexing. In this scheme, some of the tester pins are multiplexed by a high-speed circuit to generate clock frequencies that are up to four times higher than the clock produced by the tester itself. The works presented in [11] and [12] used an external shift register to store a limited length bit stream sent by ATE and an external comparator for response analysis. For small test sequences, the authors in [13] managed to test devices at 500 MHz using a 100 MHz ATE by a technique that basically repeats the test when the tester limitation is confronted.

Using an alternative strategy, the test is conducted using a slow-speed test mode, reported in [14]. The authors reduced supply voltage to slow down the circuit in order to test by a slow-speed ATE. The main difficulty of the lowering voltage technique is the increased sensitivity to noise and the unscaled portion of delay (e.g. the routing delay that remains almost unchanged). Another idea of reducing speed of CUT was suggested in [15] where a controllable delay was introduced in the timing paths during test. With the added delay, the maximum operating frequency is lowered to a rate which is within the capability of the ATE. For testing memories by slow ATEs, the use of an on-chip interface has been suggested in [16].

The speed difference between ATE and CUT becomes more important for delay fault testing. The test generation for delay fault testing is closely tied to the test application. Thus, prior to generating tests we need to know how the test vectors will be applied to the circuit. Two common strategies are slow-fast-slow [17][18] and at-speed [19][20]. Authors in [21] discussed a scheme for at-speed test on a low-speed tester for which the test generation is dependent on the speed of tester. Other important issues, related to signal characteristics, in using low-speed ATEs to test high-speed circuits include waveform fidelity, calibration and characterization [][].

The main contribution of our work is a test methodology to make a low-speed ATE capable of testing high-speed CUT. The main engine that matches the ATE's low and the CUT's

high frequencies is a hardware interface, called TIC (test interface circuit). Using TIC, the average test frequency will be equal or very close to the CUT frequency. TIC is an external circuitry that can be easily adjusted when various test architectures are used (e.g. for different cores) inside CUT. An integer linear programming (ILP) formulation is also presented that takes information about ATE/CUT and test patterns/signatures into account and optimizes the TIC circuitry to achieve minimum test overhead or minimum overall test time.

2 ATE-TIC-CUT Architectural Model

In our model, we assume an ATE with working frequency f_{ATE} is used to test a CUT with operational frequency f_{CUT} such that $f_{CUT} > f_{ATE}$. In this paper, sometimes we refer to the ATE and CUT clock periods as ATE-cycle and CUT-cycle, respectively. To be able to link the actual frequencies and the scheduling steps, we define the frequency mismatch factor p:

$$p \geq \lceil \frac{f_{CUT}}{f_{ATE}} \rceil \tag{1}$$

In Section 5 we will discuss experimental results as to how p affects test time and overhead. Furthermore, to test a CUT we assume overall l patterns, each with bitwidth of w, need to be applied and the corresponding signatures need to be analyzed. The objective is to design a test interface circuit (TIC) that coordinates test data transfer (patterns and signatures) between ATE and CUT to allow at-speed testing. This model is shown in Figure 1. A circuit inside TIC multiplies the clock generated in ATE to match the speed of CUT, for example using phase-locked loop [22] or a frequency multiplier [21])

To take the ATE limitations (i.e. speed and pins) into account, in our model only part of the test patterns are sent from ATE to TIC and the rest are stored in the TIC's internal memory. These two pieces are assembled and forwarded to CUT for testing. Similarly, the signatures are collected by TIC. Part of signature is sent to ATE (with frequency f_{ATE}) for accurate analysis and the rest will be compared (with frequency f_{CUT}) in TIC. Note that the signature delivery to TIC or ATE does not need to be at speed in general.

From implementation point of view, TIC is made of memory (ROM/RAM), a small finite state machine (controller/sequencer) and some inexpensive glue logic (e.g. multiplexors) which need to be customized for a given CUT. When programmable devices, such as high-speed FPGAs, are available implementation of TIC becomes mostly automatic using the appropriate CAD tools.

2.1 Test Data Partitioning

We focus on $ATE \rightarrow TIC \rightarrow CUT$ path for optimization due to its flexibility in terms of time and memory requirement. We will elaborate on the readout architecture in a separate subsection. Assuming deterministic test patterns are available to test a CUT by an external ATE, we need to find an appropriate split of test data for the part that ATE sends to TIC and the part that is stored in the TIC's internal memory. This is quite important in terms of matching the speed of ATE and CUT. We

Figure 1. ATE-TIC-CUT architectural model.

assume overall $w \times l$ bits form the test pattern set which is partitioned into m groups with length of $l_1, l_2, \cdots, l_m$ such that $\sum_{i=1}^{m} l_i = l$. This partitioning is graphically shown in Figure 2(a) for a small example. For each partition i, we consider two parts: 1) *C-part* or common part (shaded area in the figure) that is made of multiple copies of an identical piece of data in partition i and 2) *D-part* or distinct part that are different for each vector. Partitioning the patterns set and identifying the C- and D- parts within each partition is beyond the scope of this paper. We used a simple partitioning algorithm based on the bit-similarity metric and allowed reordering across, but not within, partitions. However, any other partitioning approaches that can identify similarities among the test vectors and group them to maximize the common part can be employed.

As shown in Figure 2(a), the overall number of bits representing C- and D- parts are c_i and d_i, respectively. Note carefully that d_i is the bitwidth of D-part that needs to be sent to the CUT. However, c_i is the bitwidth of the *representative* portion of data whose repetition makes the C-part. Inside TIC, the representative portion (c_i bits) will be replicated appropriately to construct the entire C-part portion of test data. For example, in Figure 2(a) the representative portion of C-part for partition 1 and 3 are "111100" ($c_1 = 6$,$d_1 = 28$) and "1100" ($c_1 = 4$,$d_1 = 30$), respectively. Note also that Figure 2(a) only shows the idea of partitioning. The C-part is not necessarily located at the end (high significant bits) of the test vectors. The partitioning algorithm decides where exactly the C-parts of partitions are located.

The general plan is to transfer C-part (c_i bits) from ATE to TIC with frequency f_{ATE}. Then, the C&D-part (concatenation of C- and D- parts) is transferred from TIC to CUT with frequency f_{CUT}. So, transferring the common part is almost p ($p \geq \lceil \frac{f_{CUT}}{f_{ATE}} \rceil$) times slower. On the other hand, if only N_{ATE} pins of the ATE are available, c_i bits can be transferred in $p_i = \lceil \frac{c_i}{N_{ATE}} \rceil$ ATE-cycles ($1/f_{ATE}$). In general, to transfer c_i bits of C-part from ATE to TIC, the time period equivalent to k_i CUT-cycles ($1/f_{CUT}$) for partition i is required:

$$k_i = p \cdot p_i = p \cdot \lceil \frac{c_i}{N_{ATE}} \rceil \tag{2}$$

2.2 Effect of Scheduling on Time and Cost

To clarify the role of parameters introduced above and also the effect of scheduling, consider the small example in Figure 2(a). This figure shows three partitions with length of $l_1 = 3, l_2 = 8, l_3 = 6$ and C-part bitwidths of $c_1 = 6, c_2 = 16, c_3 = 4$. For brevity, we have not shown the exact D-part of partitions. If the operational frequencies of the ATE and CUT were equal we would not need any TIC (test overhead of 0) and the overall test time was $\sum_{i=1}^{3} l_i = 17$ cycles. Let's assume the CUT is three times faster than ATE ($p = 3$) and the number of ATE pins available is $N_{ATE} = 8$. The C-parts for the three partitions can be transferred from ATE to TIC in $p_1 = 1, p_2 = 2, k_3 = 1$ ATE-cycles, which is equivalent to $k_1 = 3, k_2 = 6, k_3 = 3$ CUT-cycles. Figure 2(b) shows two possible scheduling. Note that the time scale is based on CUT-cycle ($\frac{1}{f_{CUT}}$) and the shaded rectangles of size 3 CUT-cycles show the slower C-part transfers by the ATE.

In general, reordering test patterns across, but not within, partitions is allowed. Reordering can be decided based on the nature of cores under test (e.g. combinational versus sequential), the way that partitioning is done (e.g. to define the C-part) and also the optimization phase (e.g. to find the test schedule). For example, we may keep all test patterns for a sequential core within one partition when their order is crucial for test quality. The transfer time is equal to the overall time of the schedule. The main part of test overhead cost is the memory units required in TIC which depend on the C- and D- part overlaps. After the transfer of one partition is complete, the memory can be reused to store data corresponding to the other partitions. For partition i, overall $c_i + l_i \cdot d_i$ bits of memory are needed. For simplicity, we did not differentiate between RAM and ROM cells (see Figure 1). The overall memory is the maximum of memory required for the overlapped partitions over all scheduling cycles:

$$Memory_Bits = \max_{1 \le t \le T_{max}} \{ \sum_{overlapped\ i} (c_i + l_i \cdot d_i) \} \quad (3)$$

Compared to the ideal at-speed test (i.e. $f_{ATE} = f_{CUT}$ with no TIC circuitry), the test transfer time of schedule 1 and 2 are $\frac{23-17}{17} = 35\%$ and $\frac{20-17}{17} = 18\%$ longer, respectively. This means $\frac{f_{test_sch1}}{f_{CUT}} = \frac{17}{23} = 74\%$ and $\frac{f_{test_sch2}}{f_{CUT}} = \frac{17}{20} = 85\%$ for these two schedules, where f_{test} is the average test transfer frequency for that schedule. Obviously, test schedule 2 is superior because it uses the same amount of memory (i.e. 328 bits as $c_2 + l_2 \cdot d_2 + c_3 + l_3 \cdot d_3$ indicates) but can test CUT with frequency $0.85 f_{CUT}$ as opposed to $0.74 f_{CUT}$ achieved by schedule 1.

2.3 Architecture to Readout Signatures

As the ATE-TIC-CUT model (Figure 1) shows, we assume that the C-part of signature is accumulated in a RAM and need to be sent to the ATE (at the ATE speed) for accurate (e.g. timing, glitch, etc.) analysis. The D-part of signature is compared to the expected signatures stored in an internal memory (e.g. ROM) as they are produced (at the CUT speed). Note that time for generating signature depends on the nature of CUT

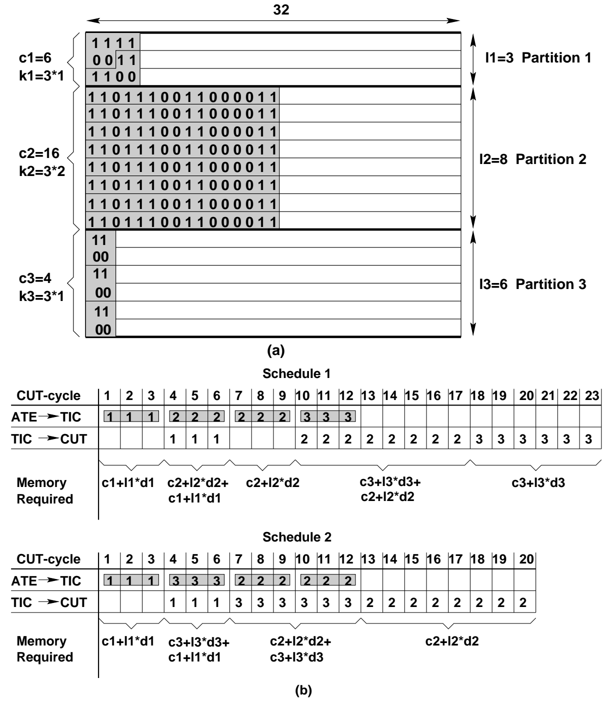

Figure 2. A small example: (a) Partitioning; (b) Scheduling.

and its delivery to TIC/ATE does need to be at circuit speed. The signature readout and testing is an inflexible part of ATE-TIC-CUT interaction and it adds a constant time and memory overhead to the overall cost. Therefore, the readout time and cost solely depend on the specifications and the user's choice, e.g. as to which signals need accurate analysis. In the continuation of this paper, we will focus on the test data transfer and optimization which is the flexible part of the interface (TIC).

3 The Preliminary ILP Formulation

■ **Variables**

Assuming a set of test data is split into m partitions such that partition i ($i = 1, 2, \cdots, m$) has C- and D-parts of c_i and d_i bits as explained before, we define two set of integer variables:

$$x_{i,t} = \begin{cases} 1 & \text{if ATE sends C-part of partition } i \text{ at time } t \\ 0 & \text{otherwise} \end{cases}$$

$$y_{i,t} = \begin{cases} 1 & \text{if TIC sends C\&D-part of partition } i \text{ at time } t \\ 0 & \text{otherwise} \end{cases}$$

where $1 \le i \le m$ and $1 \le t \le T_{max}$. T_{max} reflects an upper bound for the time steps required for test scheduling and can be estimated as $\sum_{i=1}^{m}(k_i + l_i)$. Ideally, we have $f_{ATE} = f_{CUT}$ and there is no need for TIC. In such ideal scenario at-speed test can be done in $\sum_{i=1}^{m} l_i$ CUT-cycles. However, when $f_{ATE} < f_{CUT}$ at-speed test may be too expensive or even impossible in some cases due to the ATE speed and pin limitations. The goal, then, should be trying to maximize the average test frequency while satisfying the constraints.

■ **Constraints**

At each time step, only one partition can be transferred. Thus we need the following constraints:

$$\begin{cases} \sum_{i=1}^{m} x_{i,t} \leq 1 & \forall t \\ \sum_{i=1}^{m} y_{i,t} \leq 1 & \forall t \end{cases} \tag{4}$$

To assure that the C- and D- parts are entirely transferred to add these constraints:

$$\begin{cases} \sum_{t=1}^{T_{max}} x_{i,t} = k_i & \forall i \\ \sum_{t=1}^{T_{max}} y_{i,t} = l_i & \forall i \end{cases} \tag{5}$$

where k_i is given by Equation 2.

Based on definition of $x_{i,t}$ ($y_{i,t}$), $t \cdot x_{i,t}$ ($t \cdot y_{i,t}$) gives the time index when time step t is dedicated to transfer C-part (D-part) of partition i. Therefore, $\sum_{i=1}^{T_{max}} t \cdot x_{i,t}$ ($\sum_{i=1}^{T_{max}} t \cdot y_{i,t}$) shows the sum of all time indices for k_i (l_i) cycles. This is a constant number equal to $\frac{k_i \cdot (k_i - 1)}{2}$ ($\frac{l_i \cdot (l_i - 1)}{2}$). New variables can be defined now for C- and D- parts of each partition to show the start (sx_i and sy_i) and end (ex_i and sy_i) time indices:

$$\begin{cases} \sum_{t=1}^{T_{max}} t \cdot x_{i,t} = k_i \cdot sx_i + \frac{k_i \cdot (k_i - 1)}{2} & \forall i \\ \sum_{t=1}^{T_{max}} t \cdot x_{i,t} = k_i \cdot ex_i - \frac{k_i \cdot (k_i - 1)}{2} & \forall i \end{cases} \tag{6}$$

$$\begin{cases} \sum_{t=1}^{T_{max}} t \cdot y_{i,t} = l_i \cdot sy_i + \frac{l_i \cdot (l_i - 1)}{2} & \forall i \\ \sum_{t=1}^{T_{max}} t \cdot y_{i,t} = l_i \cdot ey_i - \frac{l_i \cdot (l_i - 1)}{2} & \forall i \end{cases} \tag{7}$$

To guarantee that entire C- and D- parts are transferred in consecutive time steps, we add the following constraints:

$$\begin{cases} t \cdot x_{i,t} \leq ex_i & \forall i, t \\ t \cdot y_{i,t} \leq ey_i & \forall i, t \\ (T_{max} - t) \cdot x_{i,t} \leq T_{max} - sx_i & \forall i, t \\ (T_{max} - t) \cdot y_{i,t} \leq T_{max} - sy_i & \forall i, t \end{cases} \tag{8}$$

Finally, to guarantee that the C-part transfer is completed before C&D-parts are assembled and sent to CUT, we add this constraint set:

$$ex_i < sy_i \quad \forall i \tag{9}$$

■ **Objective Function**

The objective is to minimize the maximum time assigned to ey variables (i.e. showing the last time index that a partition is transferred), that is: $Minimize[\max_{1 \leq i \leq m}\{ey_i\}]$. This non-linear relation can be linearized by changing it to:

$$\begin{array}{ll} \text{Minimize} & total_time \\ \text{subject to:} & ey_i \leq total_time \quad \forall i \end{array} \tag{10}$$

■ **The Complete ILP Formulation**

The complete ILP formulation to minimize the test transfer time can be obtained by combining all constraints expressed by Equations 4 through 10. This is shown in Figure 3. Note that the order of m and T_{max}, which determines the number of constraints, makes this formulation of a manageable size such that almost any ILP package can solve it.

4 The Generalized ILP Formulation

The main idea of generalizing the formulation is to include the interface cost which is mainly due to the TIC memory requirement. To do this, all variables and constraints explained before are still necessary and we need to add additional variables and constraints. A new set of 0/1 variables are needed:

$$\begin{array}{lll} \text{Minimize} & total_time & \\ \text{subject to:} & & \\ \text{(i)} & \sum_{i=1}^{m} x_{i,t} \leq 1 & \forall t \\ & \sum_{i=1}^{m} y_{i,t} \leq 1 & \forall t \\ \text{(ii)} & \sum_{t=1}^{T_{max}} x_{i,t} = k_i & \forall i \\ & \sum_{t=1}^{T_{max}} y_{i,t} = l_i & \forall i \\ \text{(iii)} & \sum_{t=1}^{T_{max}} t \cdot x_{i,t} = k_i \cdot sx_i + \frac{k_i \cdot (k_i - 1)}{2} & \forall i \\ & \sum_{t=1}^{T_{max}} t \cdot x_{i,t} = k_i \cdot ex_i - \frac{k_i \cdot (k_i - 1)}{2} & \forall i \\ & \sum_{t=1}^{T_{max}} t \cdot y_{i,t} = l_i \cdot sy_i + \frac{l_i \cdot (l_i - 1)}{2} & \forall i \\ & \sum_{t=1}^{T_{max}} t \cdot y_{i,t} = l_i \cdot ey_i - \frac{l_i \cdot (l_i - 1)}{2} & \forall i \\ \text{(iv)} & t \cdot x_{i,t} \leq ex_i & \forall i, t \\ & t \cdot y_{i,t} \leq ey_i & \forall i, t \\ & (T_{max} - t) \cdot x_{i,t} \leq T_{max} - sx_i & \forall i, t \\ & (T_{max} - t) \cdot y_{i,t} \leq T_{max} - sy_i & \forall i, t \\ \text{(v)} & ex_i < sy_i & \forall i \\ \text{(vi)} & ey_i \leq total_time & \forall i \\ \text{(vii)} & x_{i,t}, y_{i,t} \in \{0, 1\} & \forall i, t \\ & Other\ Variables \in Z^+ & \end{array}$$

Figure 3. The complete ILP formulation.

$$z_{i,t} = \begin{cases} 1 & \text{if TIC needs to store C- and/or D- parts of partition } i \text{ at time } t \\ 0 & \text{otherwise} \end{cases}$$

Variable $z_{i,t}$ should be 1 only for those time steps between the time that C-part starts to transfer (ATE→TIC) and the end of transferring C&D parts (TIC→CUT). To guarantee this we add this set of constraints:

$$sx_i \leq t \cdot z_{i,t} \leq ey_i \quad \forall i, t$$

Similar to what we discussed for constraint set 8, they are equivalent to:

$$\begin{cases} t \cdot z_{i,t} \leq ey_i & \forall i, t \\ (T_{max} - t) \cdot z_{i,t} \leq T_{max} - sx_i & \forall i, t \\ \sum_{t=1}^{T_{max}} z_{i,t} = ey_i - sx_i + 1 & \forall i \end{cases} \tag{11}$$

The generalized objective function can include both time and memory factors:

$$(W_t \cdot total_time) + (W_m \cdot total_memory)$$

and a new set of constraints will be:

$$\sum_{i=1}^{m} [c_i + (l_i \cdot d_i)] \cdot z_{i,t} \leq total_memory \quad \forall t \tag{12}$$

Selecting normalized weights for W_t and W_t (e.g. $W_t = \frac{r_t}{r_t + r_m}$ and $W_m = \frac{r_m}{r_t + r_m}$ for arbitrary positive values of r_t and r_m) imply the importance of these factors in the optimization process from the user point of view. For example, (W_t, W_m) values of (1,0) and (0,1) correspond to time and memory optimization, respectively. $(W_t, W_m) = (0.5, 0.5)$ shows that both factors are seen equally important.

The complete ILP formulation to minimize a weighted function of both the test transfer time and memory cost can be obtained by combining all constraints expressed by Equations 4 through 12.

Table 1. Tradeoff in testing the MCS 8051 when $p = 2$, $N_{ATE} = 36$ and $(W_t, W_m) = (1, 0)$.

Core	*Metrics*			*ILP Results*		
	w	l	m	*Time*	*Memory*	$\frac{f_{test}}{f_{CUT}}\%$
Core 1	32	29	3	31	788	93.6
Core 2	32	12	4	14	396	85.7
Core 3	32	28	3	34	831	82.4
Core 4	29	18	3	20	587	90.0
Core 5	21	22	2	24	700	91.7
Core 6	26	22	3	24	668	91.7

Table 2. Tradeoff for Core 1 when $N_{ATE} = 36$.

Weights (W_t, W_m)	p	ILP Results		
		Time[cycle]	*Memory[bit]*	$\frac{f_{test}}{f_{CUT}}\%$
(1,0)	2	31	788	93.6
	3	39	884	74.4
	4	33	836	87.9
(0,1)	2	37	464	78.4
	3	53	528	54.7
	4	45	496	64.4
(0.5,0.5)	2	35	581	82.9
	3	42	708	69.1
	4	41	606	70.7

5 Experimental Results

We used the ILOG *CPLEX* package from ILOG S. A., Inc. [23] to solve the ILP formulation. We wrote a C program to translate the ATE/CUT constraints and pattern specifications to the ILP input format required by CPLEX.

To challenge a large example, we selected the Intel 8051 microcontroller. Excluding the oscillator circuitry and RAM in test mode, the six cores in 8051 microcontroller have overall 20 input ports (with total bitwidth of 114 bits) and 13 output ports (with total bitwidth of 82 bits). Six I/O terminals (overall bitwidth of 36) are connected to the ATE and will be used to send patterns to the cores. We assume that an internal test access mechanism (TAM) [24][25] is available and the core access itself is not a bottleneck in this experiment.

The ILP formulation has been applied to each of the six cores separately assuming $p = 2$, $N_{ATE} = 36$ and $(W_t, W_m) = (1, 0)$. The results are tabulated in Table 1. The ILP running time to obtain the optimized test schedule for each core is between 0.5 and 4 minutes wall clock time on a SUN Ultra 5 workstation with 128 Mbyte RAM. The transfer scheduling ime (in cycle) and memory estimate (in bit) overhead are shown in columns 5 and 6, respectively. The last column in Table 1 shows how close the overall effective test frequency (f_{test}) is, compared to the ideal at-speed test frequency (f_{CUT}). For example, when $f_{ATE} = f_{CUT}$ *Core 1* can be tested in $l = 29$ CUT-cycles assuming a large number of ATE pins (i.e. 36) are available. Without using TIC circuit the test speed is limited to the ATE working frequency. However, using TIC the test schedule for *Core 1* requires $Time = 31$ CUT-cycles, which means $\frac{f_{test}}{f_{CUT}} = \frac{l}{Time} = \frac{29}{31} = 93.6\%$. This is much better than $\frac{f_{ATE}}{f_{CUT}} = 50\%$ (when $p = 2$) that we can achieve without TIC to test that core. Using our optimization technique, the effective test frequency (average of numbers in the last column in Table 1) is about $\frac{f_{test}}{f_{CUT}} = 89.2\%$.

To show the effect of frequency mismatch factor ($p \geq \lceil \frac{f_{CUT}}{f_{ATE}} \rceil$) and N_{ATE} (number of available ATE pins), we analyzed one of the six cores in 8051 (e.g. core 1 that is the CPU core) for three choices of (W_t, W_m). For the first choice $(1, 0)$, the ILP formulation finds the fastest access schedule with no consideration of cost. For $(0, 1)$, the formulation finds the cheapest (i.e. minimum memory cost) transfer schedule with no effort to minimize time. Finally, choice of $(0.5, 0.5)$ implies that both factors are equally important. When both weights W_t and W_m take non-zero values, the overall time and memory values are often between the corresponding values obtained by $(1, 0)$ and $(0, 1)$ extremes. The CPLEX running time is longer by a factor 4 to 10 (about 2 to 25 minutes) when both weights take non-zero values (i.e. concurrent minimization of time and memory). These results are tabulated in Table 2 (for $N_{ATE} = 36$ and different values of p) and Table 3 (for $p = 2$ and different values of N_{ATE}).

In Table 2, when p increases it implies that we intentionally lower the working frequency of ATE. As the ILP optimization results show, this allows tradeoffs between TIC memory cost and scheduling time. Note carefully that although minimum p (i.e. $p = \lceil \frac{f_{CUT}}{f_{ATE}} \rceil$) always produces the highest test frequency (f_{test}), increasing p does not necessarily decrease test frequency monotonically. Also, our ILP formulation can find a valid solution for large values of p. However, achieving at-speed test may not be possible. The time-cost tradeoff is possible in our approach due to the fact that different lengths of the C- and D- parts combined with different p values create various situations for the ILP to overlap and schedule partitions into available time steps. We propose to use our ILP formulation to explore different options.

In Table 3, we fixed $p = 2$ and investigated how using less or more ATE pins affect the scheduling time and TIC memory. As expected, when we use more number of ATE pins, a larger size of data (i.e. C-part) can be sent. This reduces the time of the C-part transfer which ultimately reduces the overall test scheduling time. On the other hand, reducing number of ATE pins does not increase the time (decrease test frequency) linearly. For example, in Table 3, when number of ATE pins is reduced by almost 50% from 36 to 16 (or from 16 to 8) for $(W_t, W_m) = (1, 0)$ cases the time has increased by almost $\frac{39-31}{31} \approx 26\%$ (or $\frac{47-39}{39} \approx 21\%$). This is equivalent to $\frac{93.6-74.4}{93.6} \approx 21\%$ (or $\frac{74.4-61.7}{74.4} \approx 17\%$) reduction in the effective test frequency. Reducing number of pins affects only the C-part and depending on its size, length and the frequency mismatch factor p, the ILP decides on arranging the partitions and generating the schedule. The ILP looks at the scheduling and constraints as a whole and does not monotonically follow one individual factor. That is why sometimes adjusting ATE-CUT frequencies with the scheduling factors and constraints is not successful. In such cases poor results, as low as $\frac{f_{test}}{f_{CUT}} = 53.7\%$, are reported in Tables 2 and 3. Nonetheless,

Table 3. Tradeoff for Core 1 when $p = 2$.

N_{ATE}	Weights (W_t, W_m)	ILP Results		
		Time[cycle]	*Memory[bit]*	$\frac{f_{test}}{f_{CUT}}$%
8	(1,0)	47	935	61.7
	(0,1)	54	517	53.7
	(0.5,0.5)	50	712	58.0
16	(1,0)	39	764	74.4
	(0,1)	45	548	64.4
	(0.5,0.5)	44	628	65.9
36	(1,0)	31	788	93.6
	(0,1)	37	464	78.4
	(0.5,0.5)	35	581	82.9

Table 4. TIC's Area and Delay for Core 1.

N_{ATE}	*Time [Cycle]*	TIC Statistics	
		Area [NANDs]	*Delay [ns]*
8	47	963	4.62
	50	984	4.62
16	39	1397	4.96
	44	1427	4.96
36	31	2286	5.26
	35	2335	5.26

they are still better than $\frac{f_{ATE}}{f_{CUT}} = 50\%$ achieved without TIC.

Finally, to demonstrate the actual cost and performance, a complete TIC for *Core 1* has been designed under different assumptions. The circuits are described in VHDL and synthesized using SYNOPSYS design compiler toolset [26] using its generic library. The results are tabulated in Table 4. The total area cost is expressed in terms of 2-input NAND gates. The memory modules are implemented based on regular flip-flops and form 48% (when $N_{ATE} = 8$) to 60% (when $N_{ATE} = 36$) of the total area. When small number of ATE pins are available, the scheduling time and the possibility of sharing memory both increase. As tabulated in the last column, the maximum delay in TIC is not that much different for various options.

Acknowledgements

This work was supported in part by the National Science Foundation CAREER Award #CCR-0130513.

References

[1] Semiconductor Industry Association, *The International Technology Roadmap for Semiconductors*, Sematech, Inc. 1999.

[2] A. Flint, "Testing Multichip Modules," IEEE Spectrum, vol. 31, pp. 59-62, March 1994.

[3] M. Abramovici, M. Breuer and A. Friedman, *Digital Systems Testing and Testable design*, IEEE Press 1990.

[4] V. Agrawal, C. Lin, P. Rutkowski, S. Wu and Y. Zorian, "Built-In Self-Test for Digital Integrated Circuits," AT&T Technical Journal, vol. 73, pp. 30-39, March 1994.

[5] V. Iyengar, K. Chakrabarty and B. Murray, "Huffman Encoding of Test Sets for Sequential Circuits," *IEEE Trans. on Instrumentation and Measurement* , vol. 47, no. 1, pp. 21-25, Feb. 1998.

[6] A. Jas, J. Ghosh-Dastidar and N. Touba, "Scan Vector Compression/Decompression Using Statistical Coding," in *Proc. VLSI Test Symposium (VTS-99)*, pp. 114-120, 1999.

[7] A. Chandra and K. Chakrabarty, "Test Data Compression for System-on-a-Chip Using Golomb Codes," in *Proc. VLSI Test Symposium (VTS-00)*, pp. 113-120, 2000.

[8] B. Dervisoglu and G. Strong, "Design for Testability: Using Scanpath Techniques for Path-Delay Test and Measurement," in Proceedings of *International Test Conf.*, pp. 365-374, 1991.

[9] S. Barton, "Characterization of High-Speed (Above 500 MHz) Devices Using Advanced ATE - Techniques, Results and Device Problems," in Proceedings of *International Test Conf.*, pp. 860-868, 1989.

[10] L. Ackner and M. Barber, "Frequency Enhancement of Digital VLSI Test Systems," in Proceedings of *International Test Conf.*, pp. 444-451, 1990.

[11] D. Keezer, "Multiplexing Test System Channels for Data Rates Above 1 Gb/s," in Proceedings of *International Test Conf.*, pp. 362-368, 1990.

[12] D. Keezer, "Real Time Data Comparison for Gigahertz Digital Test," in Proceedings of *International Test Conf.*, pp. 790-797, 1991.

[13] D. Wimmers, Sakaitani and B. West, "500 MHz Testing on a 100 MHz Tester," in Proceedings of *International Test Conf.*, pp. 273-278, 1994.

[14] H. Hao and E. McCluskey, "Very Low Voltage Testing for Weak CMOS Logic ICs," in Proceedings of *International Test Conf.*, pp. 275-284, 1993.

[15] V. Agrawal and T. Chakraborty, "High-Performance Circuit Testing with Low-Speed Testers," in Proceedings of *International Test Conf.*, pp. 302-310, 1995.

[16] J. Gasbarro and M. Horowitz, "Techniques for Characterizing DRAMs with a 500 MHz Interface," in Proceedings of *International Test Conf.*, pp. 516-525, 1994.

[17] P. Agrawal, V. Agrawal and S. Seth, "Generating Tests for Delay Faults in Nonscan Circuits," *Design & Test of Computers*, pp. 20-28, March 1993.

[18] T. Chakraborty, V. Agrawal and M. Bushnell, "Delay Fault Models and Test Generation of Random Logic Sequential Circuits," in Proceedings of *Design Automation Conf.*, pp. 453-457, 1993.

[19] I. Pomeranz and S. Reddy, "At-Speed Delay Testing for Sequential Circuits," in Proceedings of *Design Automation Conf.*, pp. 177-181, 1992.

[20] S. Bose, P. Agrawal and V. Agrawal, "A Rated-Clock Test Method for Path Delay Faults," *Trans. on VLSI*, vol. 6, no. 2, pp. 323-331, June 1998.

[21] A. Krstic, K. Cheng and S. Chakradhar, "Testing High Speed VLSI Devices Using Slower testers," in Proceedings of *VLSI Test Symposium*, pp. 16-21, 1999.

[22] P. Gray and R. Meyer, *Analysis and Design of Analog Integrated Circuits*, John Wiley & Sons, 2000.

[23] ILOG S. A., Inc., "The User's Manual for ILOG CPLEX 6.5," 1999.

[24] E. Marinissen, R. Arendsen, G. Bos, H. Dingemanse, M. Lousberg and C. Wouters, "A Structured and Scalable Mechanism for Test Access to Embedded Reusable Cores," in *Proc. Intern. Test Conf. (ITC-98)*, pp. 284-293, 1998.

[25] M. Nourani and C. Papachristou, "An ILP Formulation to Optimize Test Access Mechanism in SoC Testing," in *Proceedings of the International Test Conference (ITC)*, pp. 902-910, Oct. 2000.

[26] Synopsys Design Analyzer, "User Manuals for SYNOPSYS Toolset Version 2000.05-1," Synopsys, Inc., 2000.