Automatic Test Timing Assignment for RAMs Using Linear Programming

WEN-JER WU^{a, b} and CHUAN YI TANG^{a, *}

^a Department of Computer Science, National Tsing-Hua University, Hsin-Chu, Taiwan 300, R. O. C.;

^b Vate Technology Co., Ltd., 9 Li-Hsin Rd. V, Science-Based Industrial Park, Hsin-Chu, Taiwan 300, R. O. C.

(Received 23 June 1997; In final form 24 July 1998)

In this paper, an automatic technique for test timing assignment is proposed which is comprehensive enough to take the test objective (*e.g.*, strictness of selected AC timing parameters) and the constraints from both RAM specification and tester into consideration. Since test timing assignment problem could only be solved manually before, therefore, our work can significantly reduce the efforts and costs on developing and maintaining timing modules of RAM test programs. In the proposed technique, the test timing assignment problem is transformed into a linear programming (LP) model, which can be automatically solved. Examples of building LP models for an asynchronous DRAM are given to show feasibility of the proposed technique.

Keywords: Test timing assignment, AC timing parametric testing, memory testing, linear programming, ATE software development

1. INTRODUCTION

Random- Access Memory (RAM) test program usually consists of several test items, which are designed to test each particular functionality of device under test (DUT) respectively. Each test item contains a specific combination of operations, timing and voltage conditions. The goal of a test program is to cover all aspects of functionality under the requirements of voltage, current, and AC timing in agreement with the specification of the device.

Automatic Test Equipment (ATE), or tester, drives stimulus signals to DUT and check if the DUT can yield the expected responses as defined in the test program, which is developed according to the specification of DUT. The formation of electronic signals are defined by voltage conditions, waveform formats, and the occurrence of timing edges. Where voltage conditions determine the amplitude of signals, waveform formats determine the encoding of 0/1 data, and timing edges determine the time of signal transitions. More specifically, a typical RAM test program usually consists

*Corresponding author. e-mail: {wjwu, cytang}@cs.nthu.edu.tw

of the following six major modules: (1) *test plan module* for coordinating the whole process of testing, (2) *pin condition module* for setting input voltage level, output voltage comparing level, and loading electronics, (3) *socket module* for mapping signal channels of tester to pins of DUT, (4) *de-scramble module* for decoding address or data scramble defined in DUT, (5) *timing module* for test timing clock generation for the transition (*i.e.*, rising or falling) of test signals, and (6) *test pattern module* (or *data pattern module*) for the generation of binary data for control, address, and data patterns.

Test pattern (6 in the above) is the most critical part of the whole test program since it influences the fault coverage of the truth-table behaviors. Due to the impossibility of exhaustive testing, many fault models are therefore proposed to reduce the generation of test patterns. Many discussions on fault models and well-developed test patterns with proven fault coverage can be found in [1–3]. Moreover, due to the theoretical supports, many test patterns were, therefore, standard in practical testing and were built as standard program development library.

However, during the practical test program development, test timing assignment for developing timing patterns, among all tasks of developing a RAM test program, is the most troublesome. This is because that it usually requires a large number of different timing patterns for testing a single product. For example, at least twenty different timing patterns are required to test a typical asynchronous DRAM chip, which are applied for testing in different operation modes and the coverage of strictly defined AC timing parameters. Unlike the process of developing test patterns, the even worse problem for designing timing patterns is that they can not be reusable for different specifications. Therefore, to generate or just maintain such a great number of timing patterns is very difficult and tedious.

The main cause that makes the timing assignment problem such a complicated problem is that any timing point for signal transition is usually

bounded by many AC timing parameters simultaneously [2, 6] which are hard to track manually. For example, Figure 1 shows an example specification of asynchronous DRAM. Consider the case that “CAS b ” is adjusted, which stands for the falling transition point of “CAS” signal. “CAS b^{++} ” will also be changed due to the nature of transition. The following eight AC timing parameters that make use of “CAS b ” or “CAS b^{++} ” will also be influenced: T_{CAS} , T_{ASC} , T_{CAH} , T_{RCD} , T_{RSH} , T_{CAC} , T_{RCS} , and T_{CLZ} . Since each parameter has a defined legal range (see Fig. 1(b)), after “CAS b ” being adjusted, the strictness or even correctness of these parameters will also be changed.

For ensuring conformity of DUT to its timing specification, the test timing patterns should be assigned according to the defined legal values of AC timing parameters. However, just keeping the values of AC timing parameters in their legal ranges is not enough for testing purpose. From the experiences of practical production test, a set of key AC timing parameters are required to be covered in their extreme value under some test condition. In the following discussion, the purpose of timing testing is formally defined as a test objective.

More specifically, the test timing assignment problem is to assign the value of timing transition points to satisfied the following three test timing requirements:

1. All AC timing parameters are assigned under their allowable values defined in the specification (see Fig. 1(b)),
2. Limitations enforced by the tester (*i.e.*, transition time, minimum separation of two transitions points, and *etc.*) are satisfied, and
3. Test objective is satisfied.

However, from our survey of literature and test developing tools from major ATE manufactures such as HP [10], Advantest [7], Teradyne [9] *etc.*, there was no automatic method or tool proposed yet. In practical test program development, timing assignment is usually solved manually with the assistance of some visual editing and verification tools. These tools can be used to edit a visual timing

Remarks: x and x⁺ represent the timing point that transition begins, and completes, respectively. The transition time, or elapsed time between x and x⁺ is determined by the source of driving signals.

(a) Signal waveform of a read operation

Timing Parameter	Definition (by Equation)	Value	
		Min	Max
T _{RC}	Access cycle time (given)	130	
T _{RP}	T _{RC} - RASc ⁺ + RASb	50	
T _{RAS}	RASc - RASb ⁺	70	10000
T _{CAS}	CASc - CASb ⁺	18	10000
T _{ASR}	RASb - ADDRb1 ⁺	0	
T _{RAH}	ADDRc1 - RASb ⁺	10	
T _{ASC}	CASb - ADDRb2 ⁺	0	
T _{CAH}	ADDRc2 - CASb ⁺	15	
T _{RCD}	CASb ⁺ - RASb ⁺	20	50
T _{RAD}	ADDRb2 ⁺ - RASb ⁺	15	35
T _{RSR}	RASc - CASb ⁺	18	
T _{CSH}	CASc - RASb ⁺	70	
T _{CRP}	T _{RC} + RASb - CASc ⁺	5	
T _{RAL}	RASc - ADDRb2 ⁺	35	
T _{CAL}	CASc - ADDRb2 ⁺	35	
T _{RAC}	DQo2 ⁺ - RASb ⁺	70	
T _{CAC}	DQo2 ⁺ - CASb ⁺	20	
T _{AA}	DQo2 ⁺ - ADDRb2 ⁺	35	
T _{OEA}	DQo2 ⁺ - OEb ⁺	18	
T _{RCS}	CASb - WEb ⁺	0	
T _{RCH}	WEc - CASc ⁺	0	
T _{RRH}	WEc - RASc ⁺	10	
T _{CLZ}	DQo1 ⁺ - CASb ⁺	0	
T _{OH}	DQo3 - CASc	3	
T _{OHO}	DQo3 - OEc	3	
T _{OFF}	DQo3 - CASc ⁺	0	18
T _{OEZ}	DQo3 - OEc ⁺	0	18

(b) Specification of AC timing parameters

FIGURE 1 An example specification of asynchronous DRAM.

waveform and verify the resultant timing parameters. If any parameters were violated (or not strict enough) after verification, some adjustments are

required to be made manually to correct or to ensure strictness. Since the change of any timing points may possibly alter several AC parameters or test objective simultaneously, in our experiences, it usually takes a large number of trial-and-error to achieve a final satisfied timing pattern.

In this paper, we will propose a technique for transforming timing assignment problems into a linear programming (LP) model [5]. The LP model, in turn, can be solved automatically by many existing LP tools, such as Lindo or lp_solve. (A lot of discussions about LP materials or tools can be found in the newsgroup "sci.op-research".) Based on the proposed technique, optimization of some critical AC timing parameters while satisfying timing requirement defined in the specification as well as constraints from tester can be solved automatically. Therefore, many efforts and much time in developing and maintaining timing patterns can be saved. Examples of timing assignment for asynchronous DRAM are given to show feasibility of the proposed technique.

The remainder of this paper is organized as following. In Section 2, we will discuss the requirements for test timing assignment. In Section 3, we will propose the method of transforming test timing requirements into LP model. Section 4 shows several LP models of timing assignment problems for the example DRAM, and their resulting timing assignment. Section 5 is our conclusion.

2. TEST TIMING REQUIREMENTS

AC timing parameters of RAM define the relative separation time between the transitions of two input signal, or the delay time between the transition of input signal and the transition of output signals. Due to variation or flaws arisen from the manufacturing process, some declination in AC timing parameters can be found.

The degeneration on AC timing parameters may not necessarily lead to faults in functionality. Commonly, several speed grades of the same products are classified or sorted for different

applications. Each speed grade has a different set of allowable values of AC timing parameters. The test process to distinguish between different speed grades of RAM is called *speed sort* [6, 8].

During production test, characterizing or measuring AC timing parameters of DUT for speed sort is time-consuming and hence impractical. Instead, Go/No Go test is performed using various timing patterns, where each is created for a distinguished speed specification. According to the result of applying each timing pattern, speed grade of a DUT can be distinguished.

For example, consider a type of DRAM chip has three different speed grades, and labeled as -50, -60 and -70 (-70 has the slowest access speed). For incorporating speed sort into production test program, three test sections are created, where each timing specification is applied in a test section respectively. The sequences of these sections are ordered as -70, -60 and -50. The sorting process follows a simple rule:

- DUT that can't pass -70 timing specification is classified as faulty, since it falls under the slowest allowable speed,
- DUT that passes -70 and can't pass -60 specification is classified as -70 timing,
- DUT that passes -70 and -60 and can't pass -50 timing specification is classified as -60 timing, and

- DUT that passes the whole test program is classified as -50 timing.

The test timing pattern is defined for the clock edge generation, which is used for the transition (*i.e.*, rising or falling) of input signals, or the timing for strobing output signal of DUT for comparing. Accompanied with the definition of waveform format, voltage level, and data pattern, a complete input waveform can be generated.

For example, the detail definition of test signal generation and comparison for the example asynchronous DRAM is summarized in Figure 2. According to the table, "RAS", "CAS" and "OE" signals are generated by corresponding transition timing point "b" and "c" with Return-To-One (RTO) waveform. WE signal is generated by timing point "b" and "c" but with Return-To-Zero (RTZ) waveform. Address signals are generated by combining the edges of two clocks (or called *double clock*), where "a1", "b1" and "c1" are defined for the first clock, and "a2", "b2", and "c2" are defined for the second clock, and Surround-By-Complement (SBC) waveforms are applied. As the case of "DQ", since it is defined as outputting, the timing points "o1", "o2⁺" and "o3" are defined to be the ending of High-Z status, the starting of valid output data, and the ending of valid output data, respectively. For testing purpose, the duration "o2⁺" through "o3" are usually compared, which may

Signal Name	Type	Transition timing points	Waveform Format	Voltage level
RAS	Driving	b, c	RTO (Return-To-One)	2.4/0.4
CAS	Driving	b, c	RTO	2.4/0.4
Address	Driving	a1, b1, c1, a2, b2, c2 (combining edges of two clocks)	SBC (Surround - By - Complement)	2.4/0.4
WE	Driving	b, c	RTZ (Return-To-Zero)	2.4/0.4
OE	Driving	b, c	RTO	2.4/0.4
DQ	Comparing output data with expected data	o2 ⁺ , o3	two strobes (or windows strobes) between the two timing points o2 ⁺ and o3	2.4/0.4

FIGURE 2 The definition of test signal generation and comparison for the example asynchronous DRAM.

be completed by strobing both the two ends for comparing, or comparing continuously (or called *window strobe*).

Test objective is defined for a distinguished purpose of timing testing. Formally, a *test objective* consists of (1) a subset of key AC timing parameters and (2) a test condition. The key AC timing parameters are a subset of all AC timing parameters, selected to be tested under their extreme value. The test condition defines a set of parameters to be optimized (maximized or minimized) after the previous subset of key AC timing parameters being set at their extreme values. Therefore, an instance of timing assignment is said to satisfy the given test objective if all key AC timing parameters included in the subset are assigned at their strict values as well as the test condition is also optimized.

Moreover, due to the nature that test signals are driven and compared by tester, limitations in terms of timing generation exhibited in the tester must also be followed for guaranteeing the correctness of testing activities. Therefore, the restrictions enforced by tester must be taken into consideration in the test timing assignment process.

In summary, test timing is assigned to meet the following three requirements: (1) The definition of AC timing parameters and their legal values in the specification of DUT, (2) test objectives, and (3) limitation of testers. In the following, we will discuss these three requirements in more detail.

2.1. Timing Requirement in Specification

Timing requirement is usually described as two forms in the DUT specification, that is, (1) timing relationship of AC timing parameters, and (2) allowable values.

Before timing parameters are defined, the transition relationship between “ x ” and “ x^+ ” must first be considered. The input transition time is determined by the tester, and the output transition time is determined by both the input transition time and the characteristic of DUT. The two transition times may be different, and hence required to be defined separately. In this paper, “ it ”

and “ ot ” are defined for input transition time and output transition time, respectively. The timing point “ x^+ ” is simply the timing point “ x ” plus the corresponding transition time. For example, the equation “ $RASb^+ = RASb + it$ ” defines the fall transition of the timing point “ b ” of RAS signal from tester, and the equation “ $DQo2^+ = DQo2 + ot$ ” defines the transition of output timing point “ $o2$ ” of “DQ” signal from DUT.

Timing relationship of AC timing parameter defines the relative separation of two timing edges, which can be defined as a simple subtraction relationship of two timing points. For example, the timing parameter T_{ASR} define the setup time of row address, and is defined to be the time between address data being stable and that RAS going low, that is, the timing point “ b ” of RAS through “ b^+ ” of Address. The relationship is commonly depicted in the timing waveform (see Fig. 1(a)), and the formal definition can be: “ $T_{ASR} = RASb - ADDRb^+$ ” (see Fig. 1(b)).

The allowable AC parameter values are commonly specified as a range of legal value, *i.e.*, minimum, maximum, or both values. Values within the defined ranges are considered as legal. Figure 1(b) also shows the legal AC parameter value for normal read operation of the example asynchronous DRAM. Consider the timing parameter T_{RAS} , which is defined to be the duration of “RAS” being low, and the legal value for T_{RAS} is between 70 ns to 10,000 ns according to the specification shown in Figure 1(b).

2.2. Test Objectives

As defined in the previous, a test objective consists of a subset of key AC timing parameters and a test condition. The key AC timing parameters are a subset of all AC timing parameters, representative and critical to the operation at the specified speed. Therefore, they are tested under their extreme values. The set is usually selected based on some design concerns, test experiences, and even customer complains. Formally, the set of key AC timing parameters can be denoted as KATP, and each

member is represented as 2-tuple where the first element is the name of AC timing parameter, and the second element is the type of strict condition.

For example, in the case of asynchronous DRAM, cycle time, access time and the parameters that influence the access time (*e.g.*, T_{RCD}) and functionality (*e.g.*, maximum T_{RAS} and T_{CAS}) are commonly listed as key AC timing parameters, which can be defined as: $KATP = \{(T_{RC}, \text{min}), (T_{AA}, \text{min}), (T_{RAC}, \text{min}), (T_{CAC}, \text{min}), (T_{OEA}, \text{min}), (T_{RCD}, \text{min}), (T_{RCD}, \text{max}), (T_{RAS}, \text{max}), (T_{CAS}, \text{max})\}$.

However, some conflicts may be observed if we want to satisfy all members in KATP simultaneously within a timing pattern. For example, minimum data access time from column address (T_{AA}) and the minimum “RAS” and “CAS” delay (or separation) time (T_{RCD}) can not both be put at their extreme values simultaneously. For ensuring that all strict values of these key AC parameters are tested, some combinations of strict parameters must be made. That is, within a specific timing pattern, a subset of strict key parameters is exhibited while keeping all other parameters in their legal range. The subset of KATP, which is suitable for a test objective is denoted as SKATP.

More specifically, if $SKATP_1, SKATP_2, \dots$, and $SKATP_n$ are defined as test objectives for generating timing patterns of a test program. The conditions for them are that (1) $SKATP_i \subseteq KATP$ for every i , (2) $SKATP_i \not\subseteq SKATP_j$, $SKATP_j \not\subseteq SKATP_i$ and $SKATP_i \neq SKATP_j$ for every $i \neq j$, and (3) $\bigcup_i SKATP_i = KATP$. Moreover, from the viewpoint of practical application, a minimum number of SKATP is preferred since it will generate the test program with minimum test items.

Furthermore, for increasing observability of timing testing, some set of timing parameters are also required to approach their extreme value as closely as possible when the above set of key parameters, *i.e.*, SKATP, are at their extreme values. Similarly, the set of test condition can be formally denoted as TC, and each member is also represented as 2-tuple where the first element is the name of AC timing parameter or the expression of timing points, and the second element is the type of optimization condition.

For example, Figure 3 lists the major test condition in the case of testing DRAM. Base on the table, the set of test condition can be defined as: $TC = \{(T_{AA}, \text{min}), (T_{RAC}, \text{min}), (T_{CAC}, \text{min}), (T_{OEA}, \text{min}), (T_{ASR}, \text{min}), (T_{ASC}, \text{min}), (T_{RAH},$

Parameter Types	Involved Parameters	Optimization Types	Descriptions
Access time	$T_{AA}, T_{RAC}, T_{CAC}, T_{OEA}$	Minimize	For testing if the DUT can generate valid data output as early as required.
Setup time	T_{ASR}, T_{ASC}	Minimize	For testing if the DUT suffer from race condition between control and address signals under the minimum setup/hold time.
Hold time	T_{RAH}, T_{CAH}		
R/W duration time	$WEc - WEb$	Minimize	For testing if the DUT can make read correctly under the minimum WE duration.
Output enable duration time	$OEc - OEb$	Minimize	For testing if the DUT can make output properly under the minimum OE duration.
Valid data output duration	$DQo3 - DQo2^+$	Maximize	For testing if the DUT can retain the valid data output as long as required.

FIGURE 3 The definition of test condition (TC) for the example DRAM specification.

min), $(T_{CAH}, \min)$, $(WEc - WEb, \min)$, $(OEc - OEb, \min)$, $(DQo3 - DQo2^+, \max)\}$.

2.3. Limitation of Tester

Due to the limitation of techniques applied in testers, some kinds of constraints can be observed in testers. Generally, the restrictions can be summarized as the following items:

1. The minimum test cycle time,
2. The maximum number of transitions within a test cycle,
3. The maximum number of strobes (comparison) within a test cycle,
4. The minimum separation between two transitions, and
5. The signal transition time.

Basically, these limitations must meet the requirements as defined in the specification of DUT before the tester is selected and applied for testing. If they can't meet the test requirement, some feature of DUT can't be checked. However, items 4 and 5 can further influence the resultant timing assignment. Hence, they must be considered as constraints for test timing assignment.

3. LINEAR PROGRAMMING MODEL FOR TIMING ASSIGNMENT

Linear programming (LP) is an effective technique for modeling optimization problem consisting of an object function and a number of constraints. As the name suggests, the object function and constraints must be specified as linear combinations of variables. Object function is the function to be optimized, *i.e.*, minimized or maximized, by assigning appropriate values for all variables. The assignment of variables can not violate any defined constraints. The main advantages of LP models are that they can be easily formatted, and can be solved automatically by computer.

Since the test timing requirement and test objective can be specified as linear functions. For

any given timing specification of the DUT and test objective, we can generate an LP model as following:

- Timing points and AC timing parameters are defined as variables. Note that the variable name for timing point " x^+ " is specified as " xp " since "+" is not allowed in the variable name.
- Timing requirements are transformed into constraints, where timing relationships and legal range of timing parameters are specified as equations and inequalities, respectively. The strictly-defined key AC timing parameters in the test objective (*i.e.*, SKATP) can also be specified as equations for constraints.
- Test condition in the test objective (*i.e.*, TC) is specified as the object function. Note that minimization form is commonly used since most LP solvers require that object function should be specified as a minimization form.

Solution for the LP model is a correct timing assignment, since it satisfies the given test objective as well as both the timing specification of DUT and the restrictions of tester. In the following, we will illustrate the method of transforming timing assignment problem to constraints and object function of LP model in detail.

3.1. Constraints

As stated previously, constraints may possibly come from the constraints of both target specification and those of tester. In summary, they can be used to specify the following categories of constraints:

1. Timing constraints enforced by specification, *i.e.*, the definition of timing relationship and allowable value of each AC timing parameter,
2. Timing constraints enforced by tester, *e.g.*, input/output transition time specified as pre-defined value, transition relationship enforced by tester,
3. Strictly-defined key AC timing parameters (SKATP of test objective) specified as pre-defined value.

The constraints transformed from these three categories are summarized in the table of Figure 4. Figure 4(1) show the timing relationship defining input transition time, and the example defines the constraint based on the equation: “ $RASb^+ = RASb + it$ ”. Figure 4(2) show the timing relationship defining output transition time, and the example defines the constraint based on the equation: “ $DQo2^+ = DQo2 + ot$ ”.

Figure 4(3) shows the constraints of timing relationship by defining AC timing parameter as the duration between two timing points. In the case of T_{RAS} , which is the duration from “ $RASb^+$ ” to “ $RASc$ ”, and defined as the equation: “ $T_{RAS} = RASc - RASb^+$ ”.

Figures 4(4) and 4(5) show the constraints on the legal values of AC timing parameters, which are specified as inequalities to bind legal value. For example, the legal value of T_{RAS} can be specified as: “ $T_{RAS} \geq 70 \text{ ns}$ ” and “ $T_{RAS} \leq 10000 \text{ ns}$ ”.

As to the constraints of tester, Figures 4(6) and 4(7) define the transition time of input signals and output signals, respectively. Figure 4(8) shows the constraints of minimum separation of two transitions. In the case, for the correctness of driving

signals, the timing point “ $b1$ ” of address is only allowed after the transition of “ $a1$ ” is completed, and therefore, the constraints can be specified as: “ $ADDRb1 - ADDRa1^+ \geq 0$ ”.

Figure 4(9) shows the constraints arisen from SKATP defined in the test objective, which are specified as pre-defined values. For example, we want to perform our testing in the least cycle timing, and thus, we may define “ $T_{RC} = 130$ ” directly.

3.2. Object Function

Test condition requires to be optimized after a solution is found. Hence, it can be specified as an object function in the LP model.

Due to most LP solvers accept only minimization objection function, hence, several parameters with different optimization types defined in TC are required to integrate in a minimization objective function. Since to maximize a function is just equivalent to minimize it in the negative form, and *vice versa*, positive and negative sign is given to the parameter that will be minimized and maximized, respectively.

Category	Constraints	Example specification in LP
Specification of DUT	(1) timing relationship defining input transition	$rasbp = rasb + it$
	(2) timing relationship definition output transition	$dqo2p = dqo2 + ot$
	(3) timing relationship definition AC timing parameter	$tras = rasc - rasbp$
	(4) allowable minimum value for AC timing parameter	$tras \geq 70$
	(5) allowable maximum value for AC timing parameter	$tras \leq 10000$
Tester	(6) pre-defined input transition time	$it = 2$
	(7) pre-defined output transition time	$ot = 2$
	(8) the minimum separation time of two transitions	$addrb1 - addra1p \geq 0$
Test Objective	(9) pre-defined value of certain key AC timing parameters	$trc = 130$

FIGURE 4 The definition of constraints for the example DRAM specification.

For example, object function using the test condition expressed as TC can be specified as following:

$$\begin{aligned} \min: & \text{tasr} + \text{trah} + \text{tasc} + \text{tcch} + \text{taa} + \text{trac} + \text{tcac} + \text{toea} \\ & + \text{wec} - \text{web} + \text{oec} - \text{oeb} + \text{dco3} - \text{dco2p}. \end{aligned}$$

3.3. Developing LP Models

As described in the previous sections, members of KATP may possibly contradict with each other. Hence, in practical development process, SKATP is not easily decided. A reasonable solution to avoid conflicts arisen from the elements of SKATP is to generate SKATP adaptively. The following algorithm shows the procedure formally:

ALGORITHM Generate Timing Assignment with little prior knowledge of SKATP.

- Input** Timing constraints of specification, timing constraints of tester, KATP, and TC.
- Output** SKATP_{*i*} and the corresponding timing assignment TA_{*i*} ($i \geq 1$), where $\cup_i \text{SKATP}_i = \text{KATP}$; or an unfeasible LP model.
- Step 1* Set *i* to 1 and all members of KATP as unmark.
- Step 2* Select some consistent unmarked members from KATP, to form an LP model (LP_{*i*}).
- Step 3* Solve LP_{*i*} and if it is not feasible then output the unfeasible LP_{*i*} and stop.
- Step 4* Let the solution be a timing assignment TA_{*i*}. Check TA_{*i*}, and select the members of KATP that are reached in TA_{*i*}, mark them in KATP, and output it as SKATP_{*i*}, and output TA_{*i*}.
- Step 5* Add 1 to *i* and repeat Step 2 through Step 4 until all members of KATP are marked.

In Step 2, if the consistency relationship can be decided beforehand, we can get a minimum number of SKATP_{*i*}, and hence, the minimum number of timing assignment to cover the test objective KATP. Otherwise, the most conservative way is

to include members in KATP one by one for avoiding conflicts between members. In such a case, at most the number |KATP| of LP models and timing patterns can be derived.

After generating all constraints and an objection function using the guidelines provided in the previous sections, an LP model can be generated for solving the timing assignment problem. In some cases, LP tool may yield the response of “unfeasible” instead of giving a legal timing assignment. This is due to that some inconsistency between constraints is discovered, therefore, not all constraints can be satisfied simultaneously. In the following, we will list the three major types of constraints leading to inconsistency in the order of their occurrence probability. Some diagnostic and solving techniques are also presented.

1. The constraints of timing relationship enforced by tester. For example, SBC waveform enforce more constraints than RTO or RTZ (Return-To-Zero) waveform, which enforces more constraints than NRZ (Non-Return-Zero) waveform. Therefore, if the waveform constraints can not be satisfied, another waveform type should be selected.
2. The constraints of transition times. To diagnosis if the “unfeasible” problem is arisen from the transition times, we can set them to zero and send to LP solver again. If the LP model become feasible, then the problem is arisen from the constraints of transition times. Anyway, the transition times can not be altered unless tester is changed. Therefore, if the constraint of transition time can not be satisfied, the constraints of SKATP or timing relationship require to be revised.
3. The constraints of specifications. The timing specification may possibly contain errors.

4. RESULTS

In the following, we will present the timing patterns generation for the example specification shown in Figure 1. The constraints of tester are

shown in Figures 4(6)–(8), and test condition is defined as: $TC = \{(T_{AA}, \min), (T_{RAC}, \min), (T_{CAC}, \min), (T_{OEA}, \min), (T_{ASR}, \min), (T_{ASC}, \min), (T_{RAH}, \min), (T_{CAH}, \min), (WEC - WEB, \min), (OEC - OEB, \min), (DQ03 - DQ02^+, \max)\}$, and key AC timing parameters are defined as: $KATP = \{(T_{RC}, \min), (T_{AA}, \min), (T_{RAC}, \min), (T_{CAC}, \min), (T_{OEA}, \min), (T_{RCD}, \min), (T_{RCD}, \max), (T_{RAS}, \max), (T_{CAS}, \max)\}$.

Using the above algorithm with some manual assistance, three SKATP and timing assignments can be reached. Note that the bolded members are listed as constraints when generating the LP model (*i.e.*, added in Step 2), while the members in plain text are included in $SKATP_i$ after the LP model is solved (*i.e.*, added in Step 4).

- (1) $SKATP_1 = \{(T_{RC}, \min), (T_{RAC}, \min), (T_{RCD}, \max), (T_{AA}, \min), (T_{CAC}, \min), (T_{OEA}, \min)\}$.
- (2) $SKATP_2 = \{(T_{RC}, \min), (T_{RCD}, \min), (T_{RAC}, \min), (T_{OEA}, \min)\}$.
- (3) $SKATP_3 = \{(T_{RAS}, \max), (T_{CAS}, \max), (T_{RCD}, \max), (T_{AA}, \min), (T_{RAC}, \min), (T_{CAC}, \min), (T_{OEA}, \min)\}$.

The LP formulations are solved by a public-domain LP tool, called `lp_solve` by Michel Berkelaar (Can be downloaded from ftp://ftp.es.ele.tue.nl/pub/lp_solve). Each model can be solved within 0.1 second using the SUN SPARC/20 workstation. The first timing assignment is depicted in Figure 5, which also shows the timing pattern for Advantest memory testers.

FIGURE 5 The resulting waveform of timing assignment (1) and the corresponding timing pattern for Advantest memory tester [7].

5. CONCLUSIONS

In this paper, a new technique for automatic testing timing assignment problem was proposed. Our technique transforms the problem into an LP model, which could therefore be solved automatically by many existing LP solvers.

Before using the proposed technique, the developers must first take a lot of time to be familiar with the detail device specification (*i.e.*, timing relationship and legal values), tester specification, and test objectives, and follow by a number of trial-and-error processes to get a final satisfied timing pattern. After using the proposed technique, the constraints of specification and tester can be generated mechanically, the understanding of the specification is up to design test objectives, therefore, a satisfied timing pattern can be developed in very short time.

From our experiences, the proposed technique can be applied to the following jobs for the best advantage, which are commonly found in the production test process: (1) Transform test program for different testers, (2) Develop test timing patterns for similar specification, (3) Develop test timing pattern for different speed grades.

Acknowledgements

The authors would like to thank the anonymous referees for many constructive suggestions for the presentation of this paper.

References

- [1] Cockburn, B. F. (1994). "Tutorial on semiconductor memory testing", *J. of Electronic Testing: Theory and Applications*, 5, 321–335.
- [2] van de Goor, A. J. (1991). *Testing Semiconductor Memories: Theory and Practice*, John Wiley & Sons.
- [3] Abramovici, M., Breuer, M. A. and Friedman, A. D. (1990). *Digital Systems Testing and Testable Design*, Computer Science Press.
- [4] Wu, W.-J., Tang, C. Y. and Lin, M. Y. (1996). "Methods for memory test time reduction", *Proc. of IEEE Int'l Workshop on Memory Technology, Design and Testing*.
- [5] Ignizio, J. P. and Cavalier, T. M. (1994). *Linear Programming*, Prentice Hall.
- [6] Prince, B. (1991). *Semiconductor Memories*, 2nd edition, John Wiley & Sons.
- [7] ADVANTEST Co. (1996). *T55xx Series Memory Test System Programming Reference Manual*.
- [8] Stevens, A. K. (1986). *Introduction to Component Testing*, Addison-Wesley.
- [9] Teradyne (1996). Inc., *Marlin-J996 Memory Test System Reference Manual*.
- [10] Hewlett-Packard Co., *Testing Memories with the HP83000*, July 1997.

Authors' Biographies

Wen-Jer Wu received B.Sc. degree from Department of Computer Science and Information Engineering, National Taiwan University, Taipei, Taiwan in 1988. From 1988 to 1990, he entered compulsory military service. From 1990 to 1992, he worked as a software engineer in Digital Equipment Co. (Taiwan branch). Then, he entered Department of Computer Science, National Tsing-Hua University, Hsin-Chu, Taiwan, and currently is a Ph.D. candidate. Since 1995, he also worked as a test engineer in Vate Technology Co., Ltd., Hsin-Chu, Taiwan, where he participated in several projects on automatically solving practical IC testing problems.

Chuan Yi Tang received B.S. and M.S. degrees in electrical engineering and computer science from National Tsing-Hua University, Hsin-Chu, Taiwan, in 1980 and 1982, and Ph.D. degree in computer engineering from National Chiao-Tung University, Hsin-Chu, Taiwan, in 1985, respectively. He is currently a professor of the Department of Computer Science, National Tsing-Hua University, Hsin-Chu, Taiwan. His research interests includes algorithm design, parallel algorithms, protocol testing, and computational biology.