RON-BEAM DEBUG AND FAILURE ANALYSIS OF INTEGRATED CIRCUITS

D. CONARD, B. COURTOIS, J. LAURENT AND M. MARZOUKI

Computer Architecture Group - IMAG/TIM3 Laboratory
Grenoble, France

Received: July 2, 1990.

Abstract

A current research project at IMAG/TIM3 Laboratory aims at an integrated test system combining the use of the Scanning Electron Microscope (SEM), used in voltage contrast mode, with a new high-level approach of fault location in complex VLSI circuits, in order to reach a complete automated diagnosis process.

Two research themes are induced by this project, which are: *prototype validation* of known circuits, on which CAD information is available, and *failure analysis* of unknown circuits, which are compared to reference circuits.

For prototype validation, a knowledge-based approach to fault location is used.

Concerning failure analysis, automatic image comparison based on pattern recognition techniques is performed.

The purpose of the paper is to present these two methodologies, focusing on the SEM-based data acquisition process.

Keywords: Electron-beam testing, image comparison, failure analysis, knowledge-based system, prototype validation.

Introduction

The tendency towards the large scale integration is leading to the 'testability' question of these circuits: debugging of new circuits, final tests for manufacturing, incoming inspection and failure analysis. The problem is particularly acute for debugging and failure analysis because of the low number of access points (pins). This number can be increased using mechanical microprobes allowing the measurement of the voltage at the metallic connections located on the top of the circuit. But the use of these microprobes is now questionable because of the circuit dimensions (mechanical restrictions).

Contactless probing seems to be a solution for these problems, essentially for debugging and failure analysis, thanks to the voltage contrast phenomenon (FEUERBAUM, 1982; WOLFGANG, 1986; MENZEL and KUBALEK, 1981; URA and FUJIOKA, 1989). Utilising this electro-optical method, the potential distribution in the circuit under test can be estimated from the

SEM image. Hence, this electronic microprobe allows to analyze internal connections without any mechanical damage or electrical disturbance, or with significantly less extent than with mechanical microprobes.

Our project of designing a fully automated, integrated, diagnostic test system is based on a combined use of the SEM with a new high-level fault location method for complex VLSI circuits. The two research themes induced by this project are: *prototype validation* of known circuits, on which CAD information is available, and on the other hand *failure analysis* of unknown circuits, which are compared to reference circuits.

Test Equipment

The test equipment used is composed of an E-beam tester in voltage contrast mode for observing circuits, and an exerciser for simulating the operating environment of the circuit under test. This equipment is controlled by a main computer, in order to form a complete diagnostic station.

E-Beam Tester

The tester CAMECA ST15 (*Fig. 1*) is a microscope, specially designed for integrated circuit test purposes (BOURGEON, 1987). Its principal features are the following:

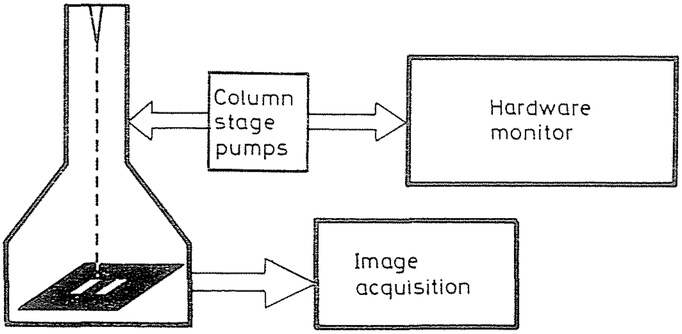

Fig. 1. CAMECA ST15 tester

- an electronic column designed to give optimum performances at low voltage (1 keV),
- a stroboscopic system to work in dynamic mode,

- a chamber with large dimensions that can incorporate an exerciser board,
- a motorized stage with a positioning accuracy of 1 micrometer,
- an electronic spectrometer,
- logic interface to the workstation.

Exerciser

We use an exerciser board for the special purpose of failure analysis of MC68000 microprocessors, developed in cooperation with the Polytechnical University of Catalunya, Spain. This environment allows the microprocessor to normally execute a program, stored in the memory.

As test sequences, loops are used. Synchronization with the beam blanking is achieved by decoding one of the addresses generated by the microprocessor when executing the sequence.

The large dimensions of the chamber allow to integrate all the electronic environment needed for controlling both reference and tested circuits. The main advantages of such a configuration are:

- since environmental conditions are identical for both circuits, disturbance effects in the data acquisition caused by changes in these conditions (e. g. vacuum break) are eliminated,
- improvement of the circuit signal quality and reduction of noise leading to better images than the ones obtained using an external exerciser (VELAZCO et al, 1989).

Control Workstation

The control and data processing is realized with a SUN workstation. This computer realises general functions, as well as specialized functions for debug and failure analysis. This workstation is equipped with a fast data acquisition system which digitizes the working images on a 512×512 pixels × 8 bits depth. A mass storage of 300 MBytes has been added and allows to store about 1200 current images.

Two main applications have been developed in the framework of this project. The first one is *failure analysis* of unknown circuits, which is detailed in the next section, and the second one is *prototype validation* of known circuits, detailed in the second next one.

Failure Analysis

Overall Methodology

The effective utilization of the very interesting possibilities of SEM requires specific test methods for testing/diagnosis. Moreover, these test methods should be different if the logical or electrical scheme of the analyzed circuit is known or if no such information is available. In this section (failure analysis), only the case of unknown circuits is investigated.

To locate a defect in a circuit of unknown internal structure, the following test principle is suggested. Suppose that some functional tests that manifest the defect (a single one) are known. The circuit can be either combinational or sequential. The SEM observation (*Fig. 2*) is giving an image of the voltage at the points in the circuit (a rectangular grid for example), for some input vector. An observation of the same area (some grid) for a fault-free (reference) circuit and for the analyzed circuit allows to note all points for which there is a discrepancy. These points include the location of the defect and locations of errors manifested by the defect (i. e. locations for which the defect has been propagated).

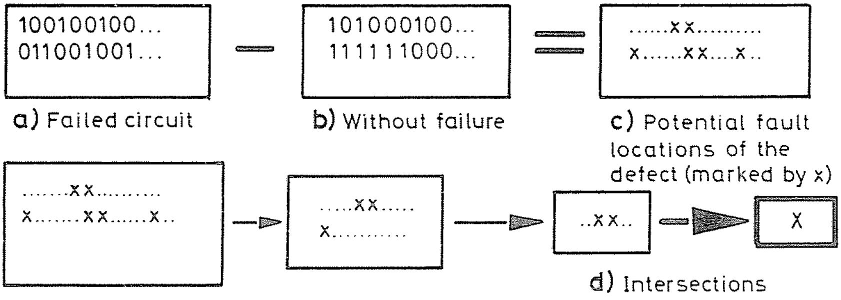

Fig. 2. Possible defect locations

Suppose now that we know another input, manifesting the same defect. Another image similar to *Fig. 2c* can be obtained, for which the marked points include the defect and the errors manifested by the second input. An ongoing process will restrict the possible locations of the defect, by intersecting the potential location sets, until ending with a single location, according to *Fig. 2d.*

The analysis of a sequential circuit is resolved by the technique of Dynamic Fault Imaging (DFI) (MAY et al, 1984). The description of this technique is outlined in *Fig. 3.* The E-beam tester is used in the stroboscopic voltage contrast mode which delivers, for both the reference circuit

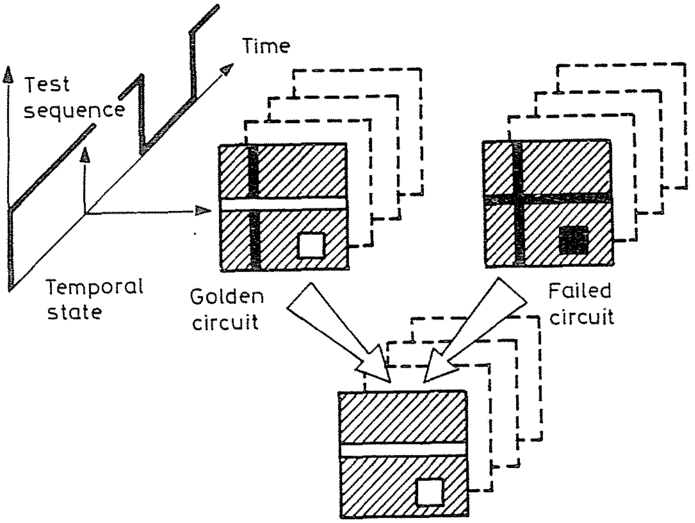

Fig. 3. Dynamic fault imaging

and the tested one frozen pictures (snapshots) of dynamic states. The test sequence is applied to each pixel of the picture for an acquisition phase which determines the temporal state. Then, the image subtraction gives the discrepancies due to the defect manifestation. The specific problem due to a sequential circuit is that errors can be propagated in time, creating many discrepancies, but without manifesting the defect location. In this case only the first image containing discrepancies must be retained among the series of images taken in a time sequence.

This global methodology has been implemented on an industrial tool. It essentially involves automatic image comparison, described in next the section.

Automatic Image Comparison

Goals and Problems

The process of comparison shows, by subtracting images, the differences in voltage contrast between the reference circuit and the faulty one. This operation is realized on a set of adjacent (neighbour) images cover-

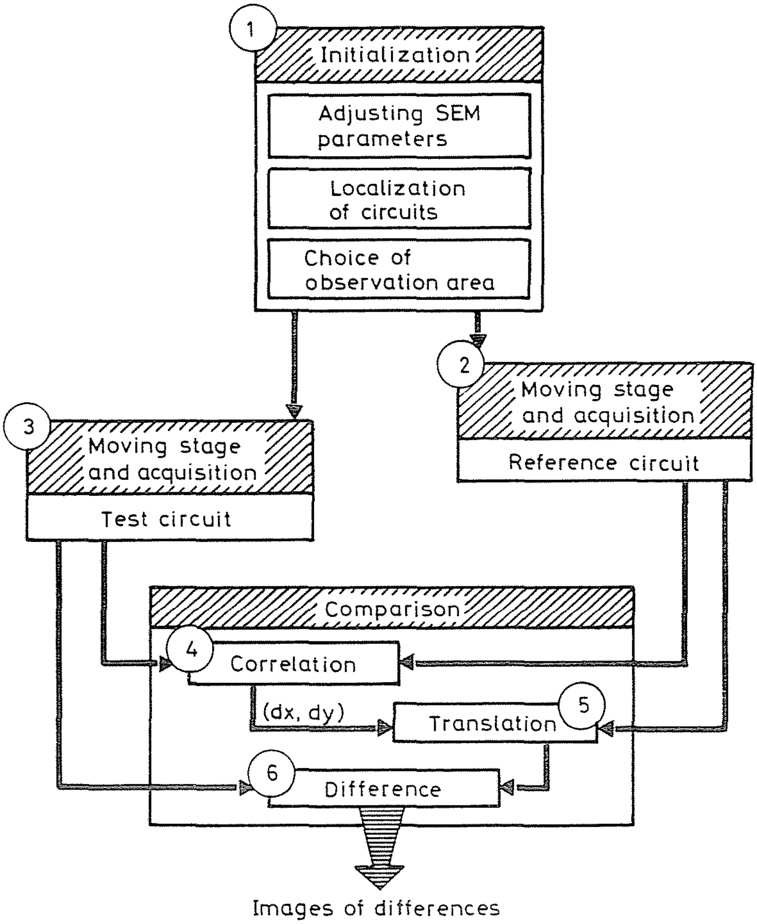

Fig. 4. Functional algorithm for failure analysis (step numbered according to the time ordering)

ing either the whole circuit or a suspected area. The determination of the suspected area requires a good knowledge of the device structure. In failure analysis, this level of knowledge is not available and then the number of images can be superior to one hundred. The MC68000, for instance, requires about 600 images for the whole circuit acquisition with 0.5 micrometer pixels. So, to cope with the great quantity of information, we must automate the elementary operations of image capture, moving stage and image subtraction. The good extraction of the voltage contrast differences requires comparable data at the temporal and spatial level. This requires two tasks to be performed: elimination of the rotation effect due to the position of circuits on the same stage and the translation discard due to the mechanical inaccuracy of the stage moving. We have developed an overall process of comparison which automatically solves these problems. This process is described in the following section.

Overall Process of Comparison

As shown in *Fig. 4*, our process can be divided into 3 steps:

(i) Initialization

This manual operation characterizes the physical parameters of the microscope and the geometrical description of the circuits. All parameters are stored in a file for further use in the stage moving and acquisition steps. Each circuit is defined by a common origin and rotation angle with respect to the stage (see *Fig. 5*). We will subsequently use this angle to correct the discards in rotation.

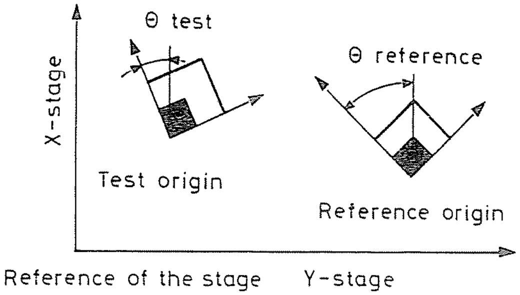

Fig. 5. Localization of circuits

Finally, we choose the observation area with the help of 2 opposite points on the reference circuit. The number of images overlapping the area of observation depends on the pixel size.

This one is given as follows: in theory 3 pixels per basic pattern are necessary for the extraction of contour lines but in practice at least 4 points are required for noise reduction. The area of observation is defined by the following parameters (*Fig. 6*):

- the coordinates of the first image: Xstart, Ystart,
- the discard between 2 contiguous images: D, computed with the magnification,
- the number of images on both X and Y directions: NBX and NBY, in order to cover the entire zone.

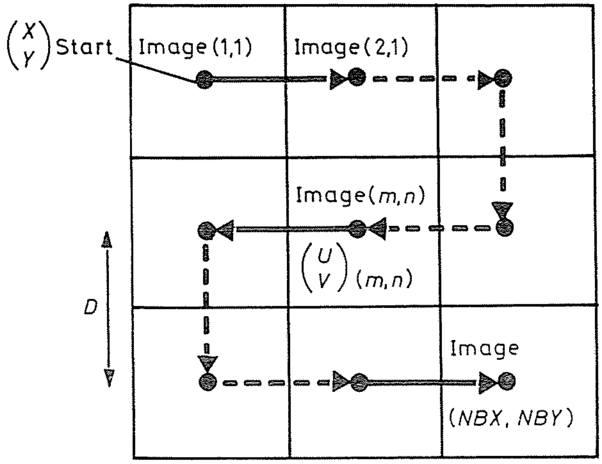
$\begin{pmatrix} X \\ Y \end{pmatrix}$ Start
Image (1,1)
Image (2,1)
Image(m,n)
$\begin{pmatrix} U \\ V \end{pmatrix}$ (m,n)
D
Image
(NBX, NBY)

Fig. 6. Description of moving stage and acquisition

(ii) Acquisition sequence (automatic step)

The moving stage is realized in a streamer mode. This organization allows us to capture all images with a minimum displacement which guarantees the best precision for the positioning of the stage. The controls of the moving stage and the acquisition are depicted in *Fig. 6.* The positions are computed to take the differences of orientation into account. In practice, we change the theoretical coordinates (U, V) of the image (m, n) by a correction with the angle of each reference circuit defined by the initialization step:

$$\begin{pmatrix} U \\ V \end{pmatrix}_{(m,n)} = \begin{pmatrix} mD \\ nD \end{pmatrix} \cdot \begin{pmatrix} x \\ Y \end{pmatrix}_{start} . \tag{1}$$

The real coordinates are given by the following transformation:

$$\begin{pmatrix} X \\ Y \end{pmatrix}_{(m,n)} = \begin{pmatrix} \cos(\vartheta) \sin(\vartheta) \\ -\sin(\vartheta) \cos(\vartheta) \end{pmatrix} \cdot \begin{pmatrix} U \\ V \end{pmatrix}_{(m,n)} . \tag{2}$$

In fact, this transformation is insufficient because it only allows the positioning of the image centre but it does not ensure the same orientation for both images (reference and test) on the screen display.

This problem is solved by electronic beam rotation which allows to turn the image with regards to its centre. For an image resolution of 512×512 pixels, an accuracy of 0.1 degree is necessary to ensure a discard

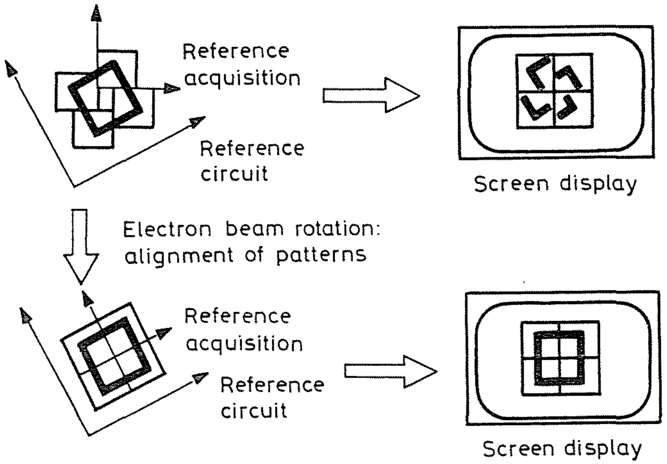

Fig. 7. Pattern alignment

between the patterns of the contiguous images of less than one pixel (see *Fig. 7*).

In practice, this operation consists of aligning an horizontal (or vertical) pattern on the screen of visualization. The beam rotation angles are estimated for each circuit and stored with the parameters of the initialization step. The images will be turned at the beginning of the acquisition phase. It should be noted that in the second step, the discards rotation are automatically eliminated without data processing. The novelty of in this solution lies in the reduction of the correlation process to a simple translation.

(iii) Automatic comparison

Before comparing images, it is necessary to have exactly the same position for both images. A correlation must be performed to automatically superimpose translation in each pair of images. The principle of this correlation consists of extracting the contour lines in order to apply a fast algorithm for corner pattern localization. This processing is achieved in two phases. The preprocessing phase isolates the relevant information on each image with the following operations:

- smoothing: this filter gives a blurred image that allows us to obtain more homogeneous areas on the threshold images. In addition, it offers a good compromise between image improvement and execution time.

- automatic thresholding: the value of the threshold is chosen to extract the logic level '0'. To do that, we examine the histogram curve for all the minima and we take only the one with the lowest frequency (*Fig. 8*).

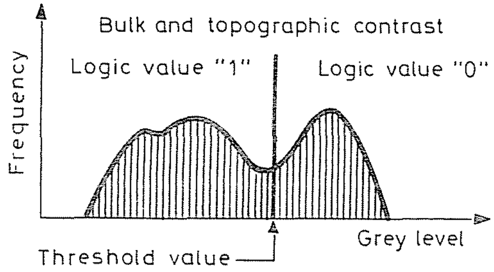

Fig. 8. Histogram in voltage contrast and cross section curve

- elimination of artifacts: this second filter eliminates the wrong information due to the substrate and to the topographic contrast. Moreover it gives straighter edges on which the localization of corners is easier. This operation is realized by two independent functions which eliminate all segments of a width less than a basic pattern in both X and Y directions.
- extraction of contour lines: this function is applied to a thresholded image. It allows the quick tracing of the contour because it does not make a derivation like a Sobel gradient. In this algorithm we keep a white pixel only if at least one of its neighbours is black. Hence we get the closed contour line with a one pixel width.

As a second phase a search and analysis of the corners position is performed, based on the localization of corners on the contour lines. It automatically determines the translation discards if they are less than the basic pattern size on a chip (named elementary segment on the image). Of course, this restriction depends only on the mechanical accuracy of the stage. It requires, for the present technology, an accuracy on each direction of 0.5 to 1.5 micrometers (SIMON and ROSENFELD, 1976).

At first, we search the corners defined by 2 orthogonal elementary segments lined up on both X and Y directions. Generally, these segments are never aligned on the horizontals or the verticals and we must search for more complex elements composed of sets of contiguous pixels. Moreover, a shifting notion with regards to both X and Y directions has been introduced to ensure a good orthogonality.

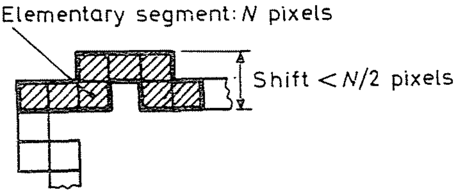

Fig. 9. Definition of corners

Thus, the tolerance for the elementary segments of N pixels, is less than $N/2$ shifting pixels (see *Fig. 9*). The algorithm distinguishes 8 corner patterns according to the segments orientation and their connectivity. We have introduced this distinction in order to compare corners which have similar positions (see *Fig. 10*).

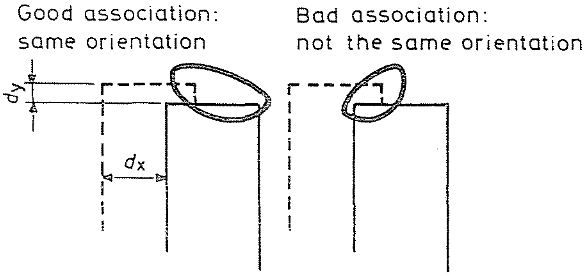

Fig. 10. Association of corners

As a second step, the corners of each image are associated in the following conditions:

- the position discards should be less than the size of an elementary segment,
- the corners must have the same orientation.

Finally, the translation discards are calculated in pixels, as the average of the position discards for each pair of corners.

After this spatial correlation step, we make an histogram equalization in order to ensure a grey level distribution normalization (COLLIN and PELLISSIER, 1987). Then, the reference image is translated and the subtraction is realized on all grey levels. The difference image will give clear areas for discrepancies and dark areas for the coincidences.

Case Study and Some Results

A set of single-faulty 68000 microprocessors has been obtained by cutting aluminium or polysilicon tracks in good 68000 circuits, using an optical LASER. This technique has provided us with a sample of circuits where faults were randomly distributed (the X-Y coordinates of the LASER were randomly generated for each circuit). Moreover, the location of these types of defects does not entail extra chemical processes to confirm the final diagnosis. Obviously, this technique allows only the injection of a limited set of faults. However, it must be noted that our goal was the validation of an automatic localization tool which provides the starting point of the failure analysis phase. In this section, we show an example of the automatic analysis.

Technical Features

(i) Test sequence timing

- sequence loop:

```
$label      movea.w data1, A1
movea.w     data2, A2
jmp         $label
```

- clock frequency: 2 MHz
- sequence period: 13 μs
- delay of observation phase: 1 μs

(ii) E-beam testing

- beam energy: 1 KeV
- beam current: 1 nA
- E-beam pulse width: 400 ns
- E-beam pulse period: 13 μs
- duty cycle: 31E-3
- magnification: 366

(iii) Image processing

- resolution: 512×512 pixels
- pixel depth: 8 bits
- frame time: 3 s
- averaged images: 10
- acquisition time: 30 s

(iv) Image comparison

- maximal observation area: 200 images
- basic pattern: 6 pixels
- execution time (preprocessing + corner study): 1 min.

Comments

The observation of fault manifestation has necessitated 3 comparison steps covering the following chip areas: operative part, control interface and ALU control. As the first discrepancies have been observed in the control interface, the remaining circuit portions (PLA, ROM,...) have not been covered. The operative part of the microprocessor, evaluated as an area of 15 mm^2, has taken 4 hours of analysis time (120 images were evaluated), while the evaluation of the ALU control and control interface parts (total area of 7.5 mm^2), required 2 hours analysis time (60 observed images).

The global study including circuit preparation, E-beam configuration, test sequence preparation and discrepancies analysis took about 2 days.

Prototype Validation Using E-Beam

Purpose and Means

Prototype validation using E-beam constitutes, like failure analysis, an internal test of circuits, that is, a premanufacture debugging. The aim of the prototype validation process is to localize residual design or prototyping faults. These faults are not easily modelled, thus methods based on the use of fault dictionaries (KUJI and TAMAMA, 1986; MELGARA et al, 1988) are not really suitable for a topological fault localization. The consideration of the most general fault model (existence of a discrepancy between the expected and observed behaviour of a circuit) is then necessary, allowing to take any kind of potential fault into account. In the case of prototype validation, the reference source is information from external CAD tools (circuit structure, functional descriptions and simulation results). This information is compared with the SEM observation results (grey level images of the device under test). After comparison the list of observed discrepancies indicates error occurrence places and dates.

From this list of discrepancies, our objective consists to develop of a completely automatic, robust and precise diagnosis method, relatively to the different levels of abstraction used for the representation of the device under test. To achieve that, we have decided to develop a knowledge-based system (KBS) for automatic fault localization (MARZOUKI and COURTOIS, 1989). This system, named PESTICIDE ('a Prolog-written Expert System as a Tool for Integrated CIrcuits DEbugging') is a second generation KBS (MARZOUKI et al, 1989), in that it makes a joint use of heuristic — shallow and deep — knowledge, which allows reasoning from the structure, function and behaviour of the device under test (DAVIS and SHROBE, 1983; DAVIS, 1984). The device model is based on hierarchical partitioning, and allows to make a clear distinction between structural, functional, and behavioural knowledge.

The global architecture of PESTICIDE is depicted in *Fig. 11.* Its main characteristics are the following:

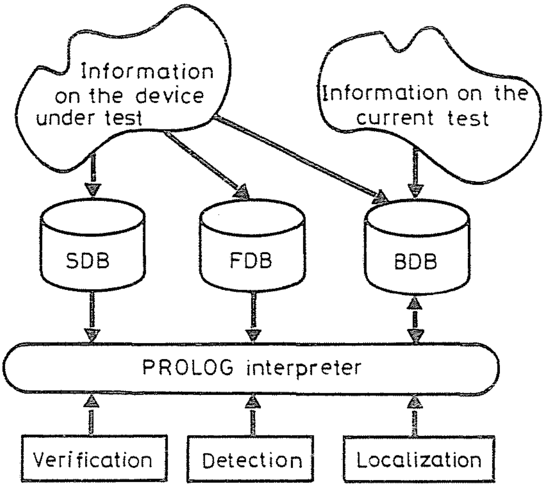
Information on the device under test
Information on the current test
SDB
FDB
BDB
PROLOG interpreter
Verification
Detection
Localization

Fig. 11. PESTICIDE global architecture

- three knowledge bases are used: structural (SDB) and functional (FDB) knowledge bases, containing information on the device under test, and behavioural (BDB) knowledge base, containing information on the current test session,
- three control modules exploit this knowledge: a verification module ensures data consistency in the knowledge bases, a detection module identifies faults on wires and a localization module searches for the faulty blocks, in the interconnected block model of the device under test.
- all the components of PESTICIDE are controlled and managed using a PROLOG interpreter.

The following subsection details the components of PESTICIDE and the diagnosis methodology.

Knowledge Representation: Model and Knowledge Bases

In this model, the device is hierarchically decomposed into interconnected blocks. Three basic components describe the circuit: the block, the wire (connection) and the block interface (see *Fig. 12*).

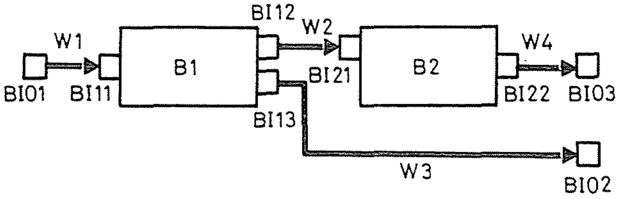

Fig. 12. Device model for debugging

Assumption:

A circuit is considered to be a sequential one if, at a given step of the test process, an the interconnections in it its partition into loops can appear. By this assumption, the proposed model facilitates to represent sequential circuits as easily as combinational ones, by means of associating to each component the relevant parameters.

Modelling the Circuit

(i) The block parameters are its name and a list of block interfaces, consisting of input, output or bidirectional block interfaces connecting the block to its external environment.

(ii) The block interface allows to connect the blocks between themselves and to the pads of the device. The major reason for introducing this notion of block interface is to allow fault detection on wires and in sequential circuits as well, thanks to time parameters.

Parameters associated to a block interface are its name, its state (erroneous or not, during the current test session), its logical value, the measuring time instant of this value its type (input or output of a block or bidirectional), a delay, measured in clock cycles, representing the time after a block interface value will be valid. This delay is evaluated as the propagation delay through the block plus the stabilization time of the block interface value (if type = output) or the propagation time of the wire (if type = input). The last parameter is the hold time of the block interface.

(iii) The wire has also a name and an associated list of block interfaces, connected either upstream or downstream in the signal flow to the wire.

Databases

The three components described above form a basis of the databases. The word 'database' is inherited from early expert systems (BARR and FEIGENBAUM, 1982), but in the case of PESTICIDE, the database is not a collection of pieces of associative knowledge in the form: 'observed symptoms ⇒ possible deductions' but rather a set of specifications of device structure, behaviour and function. In order to distinguish between these different types of specifications, we have defined three distinct databases: the 'structural' database, the 'functional' database, and the 'behavioural' one.

The *'structural' database* contains all the available information onthe device under test. These data consist of:

- The definition of the blocks, with their two parameters, as described in the previous subsection,
- The definition of the block interfaces, restricted to its 'structural' parameters (name, type, delay and the hold time),
- The definition of the wires, with their two parameters.

The *'functional' database* contains two kinds of additional information concerning the block interfaces:

- A condition for each bidirectional block interface, determining the direction of signal flow (input or output),

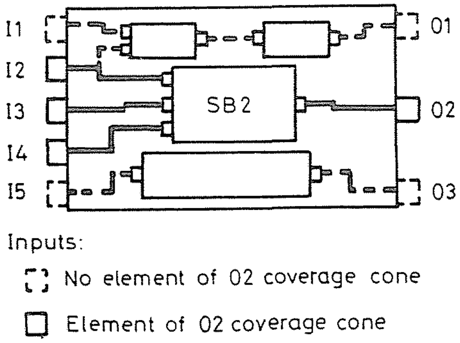

Fig. 13. Usefulness of coverage cone notion

- A coverage cone (*Figs. 13 and 14*) for each block interface, expressing the functional dependency (McCluskey, 1984) between an output-type and the input-type block interfaces.

Although not necessary, this notion of coverage cone allows to improve the computing performance of the expert system by reducing the number of alternatives (datapaths) to be explored. *Fig. 13* illustrates the coverage cone notion. In this Figure, the input block interfaces that constitute the coverage cone of output O2 are I2, I3, and I4, since by the circuit topology the values on O2 at least potentially depend on the values I2, I3, I4, and on the function of the sub-block SB2. *Fig. 14* illustrates the advantage of this coverage cone notion. Let us suppose that errors appear at output O2 and inputs I3 and I5 as well. If all the inputs of the block are indistinguishable, then the expert system can activate propagation rules on both the paths 'O2-I3' and 'O2-I5', while input I5 and output O2 are functionally independent. Such a superfluous activation of a propagation rule can be avoided by the use of the coverage cone of each block output, saving time by activating only the relevant propagation rules.

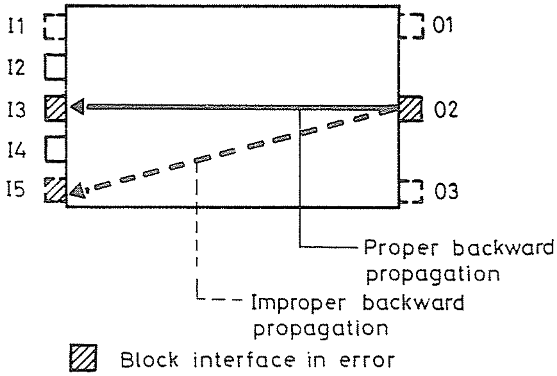
I1
I2
I3
I4
I5
01
02
03
Proper backward propagation
Improper backward propagation
▨ Block interface in error

Fig. 14. Illustration of the coverage cone notion

The *'behavioural' database* contains data about the current test, and consists of the following information:

- Instances of block interfaces, qualified by the name of the concerned block interface, and its 'behavioural' attributes, which are its state (erroneous or not), its logic value and its direction (input or output), at the given measuring time instant.
- The fault model hypothesis for the current test session. This hypothesis can be one of the followings: 'single combinational' fault (single

fault model, restricted to combinational circuits), 'multiple combinational' fault (multiple fault model, for the same type of circuits), 'single sequential' fault or 'multiple sequential' fault, for sequential circuits. This hypothesis is not a restriction of the automation of the diagnosis: in fact, it was only introduced in order to simplify the diagnosis process, and, consequently, to shorten the response time of PESTICIDE. If no hypothesis is formulated, PESTICIDE will work using the most general mypothesis, that is, multiple fault case for sequential circuits.

Of course, the information contained in the three databases must respect some data consistency constraints. These constraints are due to the application domain of the expert system, and are verified by this system. The detailed description of data consistency verification methodology can be found in (MARZOUKI et al, 1989).

Fault Localization Process

The strategy for fault localization within blocks is adapted according to the user-defined fault hypothesis for the test session. Before presenting the four strategies, some important remarks must be made:

Remark 1:

Fault localization within blocks necessitates to have the circuit under test functioning, even if it is an off-line test, so bidirectional-type block interfaces will not be mentioned any more in this section. Indeed, all of the block interface values will be sampled during the functioning of the circuit. Particularly sampled direction-control signals of bidirectional-type block interfaces will be applied to the direction selection (input or output).

Remark 2:

Sequential circuits are considered, but neither special feedback loops elimination nor transformation procedures are used. This fact can be easily explained: considering different instances of each block interface makes it possible to have the device debugged in a step-by-step functioning mode Therefore, it is possible to have at any given moment an instantaneous image of the circuit under test, and consequently, any differentiation between external inputs and internal states of a sequential system of blocks is totally useless.

Remark 3:

When pure combinational devices are considered, the following parameters are ignored:

- delay and hold time of a block interface,
- measuring instant of block interface instance.

This is possible since we can consider in this case a zero delay and infinite hold time and, accordingly, the measuring instant is not interesting.

Remark 4:

Preceding the fault localization process, PESTICIDE executes a preprocessing step in order to facilitate the use of the information shared among the three databases. The result of this step is to establish the connections between blocks and wires, using the block interface notion. This is made by reversing the relations between blocks and block interfaces on one hand, and wires and block interfaces on the other hand (for more details, see MARZOUKI and COURTOIS, 1989).

The different strategies can now be described: they are the expression of classical rules for error propagation within a given device.

(S1) 'Single combinational' fault localization strategy: when this hypothesis is used, the expert system searches first all of the block interfaces which state parameter is declared as erroneous, and then, for each of them, apply the backtracing strategy:

- examine the block which has this block interface as an output,
- if none of the members of the coverage cone (inputs) of this block interface is in error, then this block is the faulty one, and a DIRECT-type error is manifested at its primary output(s).
- if one of the inputs in the coverage cone is in error, go upstream and examine other blocks, until reaching the faulty block that manifests a PROPAGATED-type error on the current block interface.

At the end of the process, the system provides as many results as the number of erroneous block interfaces .

(S2) 'Multiple combinational' fault localization strategy: when this hypothesis is used, the process searches first the faulty blocks which manifest a DIRECT-type error, in exactly the same way as for the single fault case. After that, the process searches the blocks that are assumed to be faulty, by applying a specific strategy to each of the faulty blocks. Some blocks are called 'assumed to be faulty' because it can happen that the knowledge the system has about them is insufficient to decide unambiguously.

Table 1
Experiments with 'Single combinational fault' and 'Multiple combinational fault' strategies (CPU times for SCF and MCF in seconds)

Example	Blocks	Block int.	Errors	SCF	MCF
2-bit adder	2	10	3	0.08	0.14
4-bit adder	4	20	5	0.16	0.20
8-bit adder	8	10	7	0.43	0.61
16-bit adder	16	80	10	1.02	1.27

Therefore, it will afterwards be necessary to apply to them specific methods. These blocks are such that they have an erroneous input. The strategy is as follows:

- for each of the faulty blocks, search its erroneous outputs,
- for each of these block interfaces, examine the block(s) which has (have) this block interface as an input,
- if this block is not already declared as a faulty one, assume it to be faulty,
- apply this strategy to all the downstream blocks of this one.

(S3) 'Single Sequential' fault localization strategy and (S4) 'Multiple Sequential fault localization strategy: they will not be detailed here because they are exactly the same as the S1 and S2 strategies, respectively except that a reference value or status is assumed to be a 'good' one, that is, a value or status captured at the relevant instant. This is done by using, at each step of the reasoning process, a strategy for evaluating the validity of a measuring instant detailed in (MARZOUKI and COURTOIS, 1989).

Note that the fault localization process is interactive: the user (a design or test engineer) asks questions form PESTICIDE. If the expert system has a sufficient knowledge about the circuit and the current test, it can then directly provide the answer, otherwise PESTICIDE asks the user for some additional information like the state of a still unobserved block interface. Further developments of PESTICIDE will allow it to ask for applying some given input pattern sequence as well.

Some Execution Results

Some experiments were performed using PESTICIDE. This was not done with a real circuit, because the link between PESTICIDE and the image acquisition and processing tool is not completely achieved, but work is ongoing on this subject.

Table 2
Experiments with 'Single sequential fault' and 'Multiple sequential fault' strategies
(CPU times for SSF and MSF in seconds)

Example	Blocks	Block int.	Instances	Errors	SSF	MSF
2-bit adder	2	10	10	3	0.50	0.41
4-bit adder	4	20	20	5	1.16	0.67
8-bit adder	8	40	40	7	2.80	1.38
16-bit adder	16	80	80	10	7.10	3.30
Master/slave *Dflip−flop*	6	18	72	15	10.70	2.20

In *Table 1* the CPU time measured on a BULL DPX- 5000 computer, using the C-prolog 1.5 interpreter under the UNIX System V operating system for fault localization in 2-bit, 4-bit, 8-bit and 16-bit adders, for the 'single combinational fault' (SCF) and 'multiple combinational fault' (MCF) case, with a varying number of erroneous block interfaces is shown. In all cases, each single-bit adder is modelled as a block.

Table 2 gives the CPU time measured for fault localization using the 'single sequential fault' (SSF) and the 'multiple sequential fault' (MSF) strategies. These latter experiments have been tried on the same series of adders as in *Table 1*, and on a master/slave D flip-flop, for which each NAND gate was modelled as a block.

In each table the measurement conditions are given. These conditions are the number of blocks, the total number of block interfaces, as well as the number of errors for the two strategies, plus the number of instances of block interfaces taken into account for the single and multiple sequential strategies.

Some comments can be made on these tables. The first remark is addressing the two types of devices (combinational or sequential circuits). The selected particular cases are rather simple, but this choice is not so restrictive since, up to now, PESTICIDE uses only topological propagation rules at a given hierarchical level. We are not really interested in what is a block, rather in the number of blocks, their interrelations and the number of erroneous block interfaces or instances of block interfaces are significant.

The second remark only concerns combinational circuits. *Table 1*, compared to the first four rows of *Table 2*, gives the following ratios:

SSF/SCF: 6.7
MSF/MCF: 2.7
SCF/MCF: 0.7
SSF/MSF: 1.7

This can be interpreted as follows:

- For either single or multiple fault case, the CPU time with 'combinational strategy' is less than the CPU time with 'sequential strategy' (SSF/SCF and MSF/MCF). This is due to the fact that, for 'sequential strategy', the formulas for evaluating the measuring instant validity must be computed. This remark leads us to conclude that it is really important to define four different strategies, rather than a general, unified strategy.
- Nevertheless, it can be noted that the ratio SSF/SCF is approximately 2.5 times greater than the ratio MSF/MCF, and SSF/MSF is 2.5 times greater than SCF/MCF). In other words, for single fault localization, the CPU time required by the 'sequential strategy', as opposed to the one required by the 'combinational strategy', has grown much more than for multiple fault localization. This is due to the larger search space of instances of block interfaces for the backward propagation used in the diagnosis of propagated-type errors (single fault case) than the search space for the forward propagation, used in the identification process of blocks assumed to be faulty (multiple fault case). In the former case, the search space is a tree, while in the latter case, the search space consists only of some straightforward paths.

Conclusions

Concerning the first research topic, it is known that failure analysis is a tricky and time consuming task, usually performed manually. The global strategy presented here allows to automatically point out discrepancies between a failed circuit and a reference circuit.

As microprocessors are among the most complex circuits, we have chosen them to validate the developed tools. Nevertheless, the same tools and techniques can be applied with minor adaptations to other circuit types. The experiments performed on a faulty 68000 show the feasibility of the automation of the first and most important step of the failure analysis: fault location at the chip surface.

Concerning the second research topic, the knowledge-based system presented in this paper constitutes a real progress in the state of the art of prototype validation of VLSI circuits using E-beam, since it implements an automated of the fault diagnosis process, while other researchs and products (CONCINA and RICHARDSON, 1987; CONCINA and LIU, 1987; GORLICH et al, 1987; KOMATSU et al, 1987; KUJI and TAMAMA, 1986; HENNING et al, 1987) have only achieved the automation of the link between an E-beam tester and CAD tools, leaving the diagnosis itself to the designer of the circuit. In spite of the fact that we have presented only here a feasibility study, we hope that this research work will draw wider attention, especially through the real-case experiments.

References

BARR, A. – FEIGENBAUM, E. (1982): The Handbook of Artificial Intelligence. London, Pitman.

BOURGEON, G. (1987): Electron Beam Tester, a Tool for VLSI Component Analysis. *The Microelectronic Engineering Journal (Special Issue on Electron and Optical Beam Testing of Integrated Circuits)*, Vol. 7, No. 2-4, pp. 327-332.

COLLIN, J. P. – PELLISSIER, J. L. (1987): An Industrial Experience of Integrated Circuit Failure Analysis Using E-Beam Testing. *The Microelectronic Engineering Journal (Special Issue on Electron and Optical Beam Testing of Integrated Circuits)*, Vol. 7, No. 2-4, pp. 385-394.

CONCINA, S. – LIU, G. (1987): Integrating Design Information for IC Diagnosis. *Design Automation Conference*, Florida, June 1987. pp. 251–257

CONCINA, S. – RICHARDSON, N. (1987): IDS 5000: An Integrated Diagnostic System for VLSI. *The Microelectronic Engineering Journal (Special Issue on Electron and Optical Beam Testing of Integrated Circuits)*, Vol. 7, Nos. 2-4, pp. 339-342.

DAVIS, R. (1984): Diagnostic Reasoning Based on Structure and Behaviour. *Artificial Intelligence Series*, No. 24, pp. 347-410 Amsterdam, Elsevier – North Holland Science Publishers.

DAVIS, R. – SHROBE, H. (1983): Representing Structure and Behaviour of Digital Hardware. *IEEE Computer Journal*, pp. 75-82, Oct. 1983.

FEUERBAUM, H. P. (1982): E-Beam Testing: Methods and Applications. Proc. Microcircuit Engineering: Microlithography. Grenoble, 5–8 Oct. 1982. pp 171–178.

GORLICH, S. – HARBECK, H. – KEBLER, P. – WOLFGANG, E. – ZIBERT, K. (1987): Integration of CAD, CAT and Electron-Beam Testing for IC-Internal Logic Verification. *International Test Conference*, Washington, DC, 1–3 Sept. 1987. pp. 566–574.

HENNING, S. S. – KNOWLES, W. R. – PLOWS, G. S. (1987): CAD System Interface for a Stand-Alone E-Beam Tester. *The Microelectronic Engineering Journal (Special Issue on Electron and Optical Beam Testing of Integrated Circuits)*, Vol. 7, Nos. 2-4, pp. 317-325.

KOMATSU, F. – MIYOSHI, M. – SANO, T. – OKUMURA, K. (1987): An Electron Beam Test System Linked with a CAD Database. *The Microelectronic Engineering Journal (Special Issue on Electron and Optical Beam Testing of Integrated Circuits)*, Vol. 7, Nos. 2-4, pp. 267-274.

KUJI, N. – TAMAMA, T. (1986): An Automated E-Beam Tester with CAD Interface: FINDER. *International Test Conference*, Washington, DC, 8–11 Sept. 1986. pp. 857–863.

MCCLUSKEY, E. J. (1984): Verification Testing - a Pseudo-Exhaustive Test Technique. *IEEE Transactions on Computers*, Vol. 33, No. 6, June 1984. pp. 541–546.

MARZOUKI, M. – COURTOIS, B. (1989): Debugging Integrated Circuits: AI Can Help! *European Test Conference*, Paris, France, 12–14 April 1989. pp. 184–191.

MARZOUKI, M. – LAURENT, J. – COURTOIS, B. (1989): A Unified Use of Deep and Shallow Knowledge in an Expert System for Prototype Validation of Integrated Circuits. *International Workshop on Expert Systems and their Applications*, Avignon, France, 29 May – 2 Jule 1989. pp. 55–69.

MAY, T. C. – SCOTT, G. L. – MEIERAN, E. S. – WINER, P. – RAO, V. R. (1984): Dynamic Fault Imaging of VLSI Random Logic Devices. *IEEE International Reliability Physics Symposium,* April 1984 pp 95–108.

MELGARA, M. – BATTU, M. – GARINO, P. – DOWE, J. – VERNAY, Y. J. – MARZOUKI, M. – BOLAND, F. (1988): Automatic Location of IC Design Errors Using an E-Beam System. *International Test Conference,* Washington, DC, 12–14 Sept. 1988. pp. 898–907.

MENZEL, K. – KUBALEK, E. (1981): Electron Beam Techniques for Integrated Circuits. *Proc. Scanning Electron Microscopy,* Chicago, 1981. pp. 305–322.

SIMON, J. – ROSENFELD, C., eds. (1976): Image Enhancement and Restoration Digital Image Processing and Analysis. NATO course, NATO Advanced Study Institute, FRANCE

URA, K. – FUJIOKA, H. (1989): E-Beam Testing. *Advances in Electronics and Electron Physics,* Vol. 73, pp. 233-317.

VELAZCO, R. – CONARD, D. – GUYOT, A. – ZIADE, H. (1989): Top-Down IC Failure Analysis Using an E-Beam System, Coupled to a Functional Tester. *2nd European Conference on Electron and Optical Beam Testing of Integrated Circuits,* Duisburg, FRG, 1–4 Oct. 1989. pp. 113–120.

WOLFGANG, E. (1986): Electron Beam Testing. *Proc. Automation'86: High technology computer conference,* Houston, TX, March 1986. Vol. 4, pp. 77–106.

Address:

D. CONARD, B. COURTOIS, J. LAURENT AND M. MARZOUKI
Computer Architecture Group – IMAG/TIM3 Laboratory
46 Avenue Felix Viallet
F–38031 Grenoble, Cedex
France

INDEX