VFSIM: Vectorized fault simulator using a reduction technique excluding temporarily unobservable faults

Takaharu Nagumo, Masahiko Nagai, Takao Nishida, Masayuki Miyoshi
and Shunsuke Miyamoto

General Purpose Computer Division, Hitachi Ltd.
1 Horiyamashita Hadano-shi, Kanagawa-ken, 259-13 Japan

ABSTRACT—A new fault simulator (VFSIM) for synchronous sequential circuits has been developed and applied to random access scan circuits of several hundred LSIs in mainframe computers. The results show that VFSIM is one or two orders of magnitude faster than a conventional fault simulator designed for random access scan circuits.

A vectorized pattern parallel event-driven method is introduced for accelerating the fault simulator for synchronous sequential circuits. Utilizing the concept of unobservable regions (UOR), in which simulation of temporarily unobservable faults is avoided, contributes greatly to further acceleration, especially for random access scan circuits.

I. INTRODUCTION

The rapid increase in the complexity and density of logic circuits has made it necessary to devote more computer power to generating test data for diagnosis. To meet this necessity, many testable design methodologies [1] have been proposed. The scan design technique [2] is one of them, and is now widely used. Using this technique, the complicated test generation problem for sequential circuits can be separated into two simple problems for combinational circuits and scan circuits. Many efficient test generation and fault simulation algorithms have been proposed and utilized for combinational circuits [3][4]. For scan circuits, functional test patterns can be easily generated, but fault simulation is time-consuming because of the flip-flops that are contained.

Various types of scan design are used depending upon trade-offs between hardware overhead and test time [5]-[7]. The random access scan circuit contains address decoders, which occupy relatively large chip area and external edges, and the scan operations onto each flip-flop can be independently performed. Therefore, the random access scan design enables the observation of the real-time signal waveform of a selected flip-flop and the injection of a pseudo-failure at the flip-flop, contributing to efficient system debugging and diagnosis.

Usually circuits utilizing a scan design are tested using two steps: '0-cycle test' and '1-cycle test'[8]. First, during the 0-cycle test, the scan circuit itself is tested to see whether it is functioning correctly. Next, during the 1-cycle test, combinational circuits surrounded by scannable flip-flops are tested using the scan circuit. In the beginning of LSI production, an inconstant manufacturing process might often produce 'bad' chips containing failures in the scan circuit itself. Therefore, many chips will fail at the 0-cycle test before the 1-cycle test, so that locating faults in the scan circuit is necessary to reduce the reject rate. It is thus important to analyze scan circuit test results for both the 0-cycle and 1-cycle tests; however, manual analysis is troublesome for highly dense circuits. Especially in the case of a random access scan, address decoder failures will generate complicated symptoms, which makes manual analysis more difficult. A fault dictionary is known to be an efficient tool for fault analysis and is generated through time-consuming fault simulations. Because scan circuits consist of synchronous sequential circuits controlled by scan-clocks, a fast fault simulator for synchronous circuits is needed [9][10].

In this paper, an accelerated fault simulator for synchronous sequential circuits is addressed. Fault simulator repeats the calculation of all the gate outputs to all the faults and test patterns. Two methods are essential in order to shorten the processing time of such repetition [11]. One is to increase the processing speed for unit operation and the other is to decrease the repetition of unit operations. Our approach is to utilize vector processor for obtaining high-speed processing, and to exclude temporarily unobservable faults from simulation while fully utilizing the characteristics of random access scan design. These two algorithms introduced in our new fault simulator VFSIM are described in Sections II and III. Implementations for these algorithms are outlined in Section IV. Finally, performance evaluation and detailed analysis are summarized in Section V.

II. HIGH-SPEED EVENT PROCESSING BY VECTOR PROCESSOR

Several types of hardware simulation engines have been used in order to increase processing speed [12]-[22]. Using a supercomputer to accelerate fault simulation [17],[18],[28] is one efficient way, because many other applications which involve extensive computations such as device simulation and circuit simulation are available. VFSIM has also been developed for obtaining high performance on a supercomputer.

A. Pattern-parallelism in a sequential circuit

A key point for obtaining high performance on a supercomputer is how to collect a sufficiently large set of data for each vector instruction. The size (which often refers to the number of the words contained) is called "vector length." Basically, acceleration improves with longer vector lengths. The dynamic two-dimensional parallel simulation technique [18] makes good use of the vector processor; however, it is not adequate for sequential circuits because of the difficulty in evaluating sequential events in multiple patterns and faults at a time. Events must be evaluated in the order of patterns. Through some experiments and estimations, although controlling the sequence of event evaluations must be implemented, it has been proved that pattern-parallelism accomplishes higher performance than fault-parallelism while both using the fault dropping properly.

Scan circuits have relatively uniform structure and simple functions. Feedback loops are not complicated, and sequential behavior is mainly a result of memory elements such as flip-flops. VFSIM controls the sequence of evaluations in the ways stated below.

(1) Sorting gates into ranks: The logic elements are divided and sorted into several ranks. Input edges belong to the lowest (first) rank, and output edges belong to the highest. The basic characteristics of a rank are as follows:

(i) There is no interconnection between logical elements belonging to the same rank.

(ii) The output of an element in a rank does not connect with any element in a lower rank.

(iii) If a rank contains flip-flops, no other gates belong to the rank, and it is assigned "FF rank" attribute. A rank which contains gates is assigned "GATE rank" attribute.

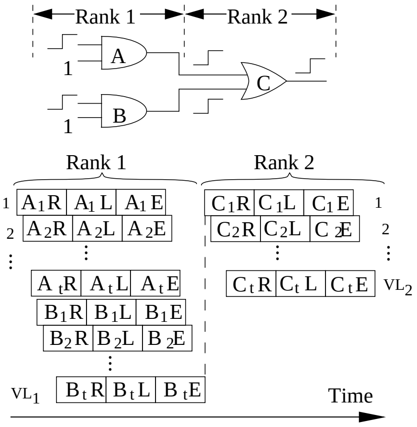
(iv) If the circuit contains a feedback loop, all the feedback signal lines are cut during the sorting. An element that has had its output cut is marked as a "loop exit." A rank which contains a "loop exit" is assigned "EXIT rank" attribute. An EXIT rank does not contain any element but marked as a "loop exit."

According to the definition of a rank, all the logical elements belonging to the same rank can be evaluated simultaneously in the same vector (Fig. 1). The simulation is accomplished by recurring evaluations of a rank in the rank order, from the lowest rank causing the initial events, toward the (highest) primary output rank, as long as events occur.

(2) Controlling feedback events: Evaluation of an EXIT rank might cause events in the lower ranks. In that case, the rank evaluation repeats from the lowest rank to which the events propagated. All the signals of "loop exits" become stable when all the events are exhausted in the EXIT and lower ranks. After this, events that occur in the next rank are propagated subsequently. It is estimated that a scan circuit with the usual feedback loops is only 20% slower than a same size circuit without loops.

(3) Evaluation of flip-flops: Given test patterns are divided into several sets of patterns. Each set is called "pattern-set." Events that occur in a pattern-set are evaluated at a time. Because a flip-flop is a sequential element, each event evaluation in a pattern might cause subsequent events after the pattern. Therefore, although a flip-flop might receive events in a small part of a pattern-set, VFSIM evaluates the flip-flop output values in the whole of the pattern-set. In comparison with the single fault propagation (SFP) technique in a scalar processor, there are unnecessary calculations at the patterns in which the flip-flop is not accepting any event; however, the ratio of unnecessary calculations must decrease when event occurrences are localized on a small part of flip-flops in a pattern-set. The validity of this vectorized evaluation technique is discussed in Section V.

For maximum processor performance, the vector length (which is the number of events to be evaluate) must sufficiently be longer than the vector register (which consists of 512 words for HITAC S-820/80). The larger pattern-set enables the longer vector length to be obtained. However, an excessively large pattern-set decreases effectiveness of fault dropping. Because detected faults can be dropped only at the end of each pattern-set process. As a result of experiments, 512 patterns are proved to be suitable in the context of vector length, memory usage, and fault dropping. The experimental results about the vector length are also discussed in Section V.



Xi denotes process for gate X at pattern i.
R : Event Reception
L : Logic Evaluation
E : Event Entry
VL_n : Vector Length in rank n

Fig.1 Vectorized event-driven technique

B. Definition and evaluation of an event

(1) Logic evaluation in gate rank: An 'event' in the fault simulation is identified by a pair of the element number and the pattern number in which the element has different input (or output) values between the faulty circuit and the fault free circuit. A fault is provoked by changing the connections of the faulty element in the circuit descriptions. For example, a connection of a faulty input pin of a gate stuck at '1' is changed to a special gate of which output value is fixed to '1'. The faulty gate is evaluated in the whole of a pattern-set and event occurrence is checked for. Those events in a rank are

stored in a queue, and are evaluated as vector data simultaneously.

On the contrary, the event-driven method is not adopted in the fault free simulation. Output values of all the logic elements are calculated in all the patterns without handling any event queue.

(2) Logic evaluation in flip-flop rank: An event of a flip-flop is identified by the logic element number of a flip-flop which accepts at least one event in a set of patterns. An event queue stocks only the element numbers, and they are processed one by one. The output values of each flip-flop are calculated in all the patterns within the pattern-set. In the fault free simulation, similar to the gate rank evaluation, all the flip-flops belonging to a rank are evaluated.

The logic evaluation for a flip-flop is achieved by utilizing the standard vector instruction of the first order iteration (VITR). The function of VITR is represented as in

$$Q_i = Q_{i-1} \cdot A_i + B_i \ .$$

Q, A, and B are the vector operands, and the subscript i means the i-th word of the vector. The i-th word stores the data about the i-th pattern of a pattern-set. During a flip-flop evaluation, in the case of referencing to the previous state (Q_{i-1}) of a flip-flop, '1' and '0' are stored in A_i and B_i respectively. When the i-th output value is defined (O_i) independently of the previous state, '0' and the value (O_i) are stored in A_i and B_i respectively.

III. UOR (UNOBSERVABLE REGION) EXTRACTION

In this section, a new reduction technique is discussed, which is efficient especially for random access scan circuits.

A. Concepts of UOR

A UOR is the region in which no signal changes affect any observable output edge. It depends on a circuit structure and a pattern (or patterns) assigned to input edges.

A schematic diagram of the random access scan circuit is shown in Fig. 2. In the random access scan design, each flip-flop has a unique scan address. Scan in/out is effective only on the flip-flop selected via address decoders. The other flip-flops not selected are stable. Their output values will not change, so that faults related to these flip-flops are unobservable from the scan out edges.

Our test pattern generator for scan circuits groups test patterns into several "pattern groups." Each pattern group accesses only a group of flip-flops within some range of scan addresses. Therefore, the faults in the region related to the other flip-flops are unobservable in the pattern group. The exclusion of these faults from the simulation process in the pattern group decreases the number of faults to be simulated.

During the '0-cycle test,' the system clock is fixed in disable state to avoid interference in the scan procedure, and only the scan output is observable. Consequently, except for the scan circuit, the whole of the combinational circuits connected to the output edge "O1" and the input "D" of flip-flops is unobservable. The scan circuit is easily extracted according to the flip-flops disabled by the fixed clock lines.

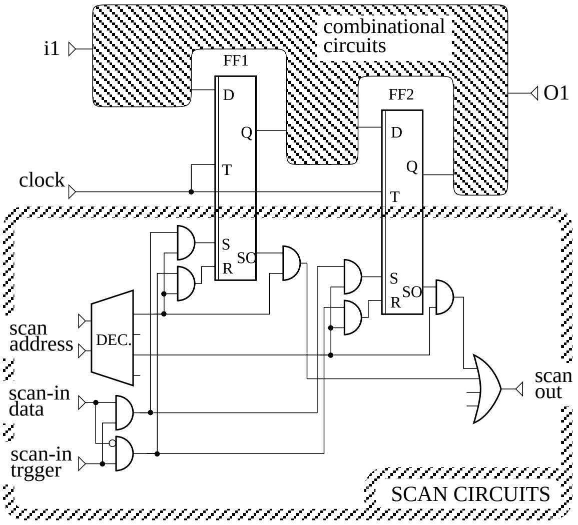
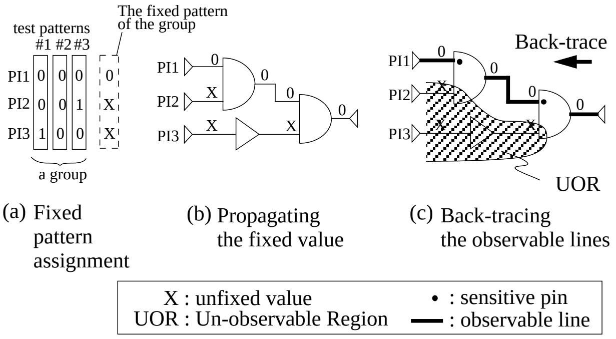


Fig. 2 Random Access Scan Circuits



(a) Fixed pattern assignment (b) Propagating the fixed value (c) Back-tracing the observable lines

Fig. 3 UOR extraction

B. Processing flow of UOR extraction

Fig. 3 shows the processing flow of UOR extraction. There is a set of input edge pins whose logic value does not change during a test period of a pattern group, such as some scan address pins. The fixed values of input edge pins are propagated until the circuit becomes stable. Backward tracing is executed to detect observable lines: Starting from the output edge pins, the observable lines are marked in sequence. Sensitivity [11] is calculated for all input pins of logic elements whose outputs are connected to newly marked observable lines. Source lines of the sensitive pins are marked as observable components. Finally, the logic elements connected to the lines not yet marked are regarded as UOR components.

If multiple input pins have the same control value ('0' for AND gate, '1' for OR gate), these should be assumed as sensitive pins in order to avoid excluding detectable reconvergent faults [11]. This case is called "multiple path sensitization." Critical Path Tracing [23] stops backtracing at the gate at which input pins have the same control value, and then a provoked fault on the source pin of this reconvergent path is regarded as undetectable. Since VFSIM addresses the accurate fault simulation, backtracing must be continued. This rule will bring slight degradation in processing performance because of the undetectable non-reconvergent faults that are included in the observable region. For combinational circuits, this problem is almost solved by precomputing the "control vector" using a symbolic simulation with a small additional overhead [24]. An extension to sequential circuits is planned in the future.

C. *Effects obtained by UOR recognition*

All faults within the UOR of a pattern group are blocked by a fixed control value, and cannot be observed at any output edge. The exclusion of these faults from fault simulation during pattern group has no effect upon the final fault dictionary. Therefore, UOR recognition is efficient for reducing the number of faults to be simulated. The other merit of UOR recognition is reducing the area of fault propagation. In the case of faults on a non-fixed value line, fault propagation can be stopped at the boundary from the observable region to the UOR. However, provoked faults on a fixed value line which might be detected accidentally must be propagated across the UOR because fault effects might change the UOR into an observable region. Violating this rule would result in an underestimation or overestimation of fault coverage and an incorrect fault dictionary, because some detectable faults would be regarded as undetectable or vice versa.

This reduction technique is widely applicable to various circuits not limited only to the random access scan circuit. There are trade-offs between the effects gained by UOR extraction and overhead required for UOR extraction. Optimal division of the test pattern groups is a key point for obtaining the maximum effect. Critical path tracing [23], Star-Algorithm, etc. [24]-[26] are extreme cases, where the test pattern group always consists of only one test pattern. These techniques may be useful to reduce the number of faults to be simulated for each test pattern group. Detailed consideration is left as a future topic of study to balance the additional overhead consumed by a reduction process with the saving acquired by avoiding meaningless SFP.

IV. Implementation

The processing flow of VFSIM is shown in Fig. 4. Processing is divided into two major phases; pre-processing and simulation.

First of all, the scan circuit is extracted from a total sequential circuit by considering the condition that system clocks are always off when a scan in/out is performed. Next, fault collapsing and rank-sort are performed for the extracted scan circuits.

The simulation phase is repeated for all test pattern groups. Before simulation, the UOR is extracted by using fixed input values as explained in Section III. Fault free simulation and fault simulation are performed for each of the 511 patterns. The scan circuit is a synchronous sequential circuit; therefore, Four values ('0', '1', 'X': unknown, 'Z': high-impedance) and zero delay simulation are sufficient, because it is not necessary to detect a hazard.

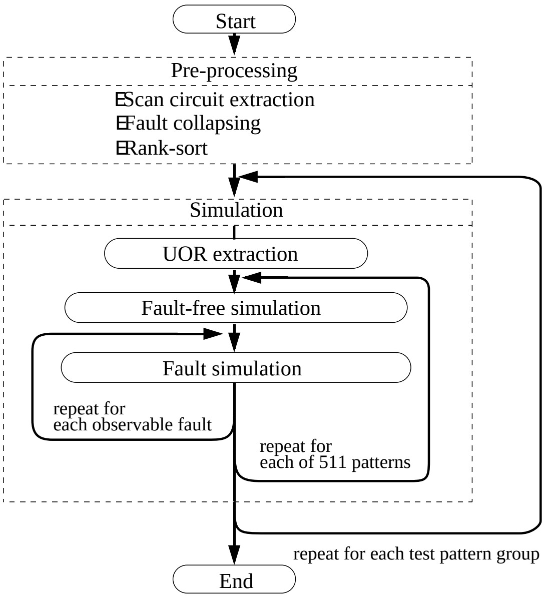


Fig. 4 Processing flow of VFSIM

V. Results and analysis

The performance of VFSIM was evaluated by using it on several hundred LSIs of mainframe computers. Logic size ranged from 3k to 40k gates. All flip-flops are scannable by the random access scan technique. Each fault coverage of all the 0-cycle tests is up to 100%. A performance comparison was made between VFSIM and a conventional fault simulator that was designed for random access scan circuits and run on a scalar processor using four values ('0', '1', 'X', 'Z'), rank-sort, and concurrent techniques[27].

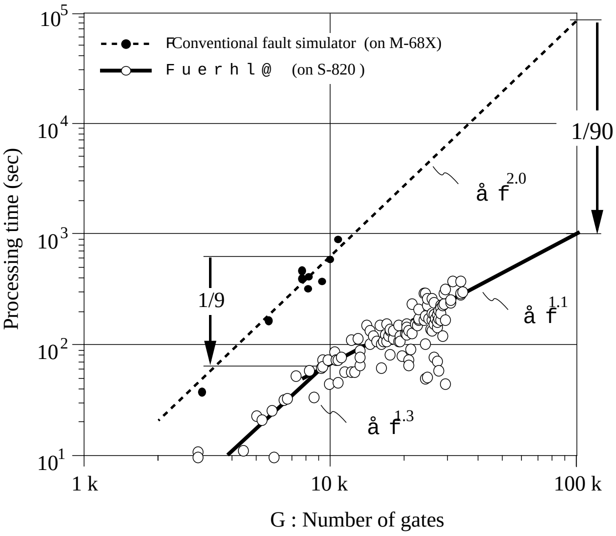


Fig. 5 Performance comparison

In Fig. 5 the bold line shows VFSIM performance. Note that the line slope changes at the 10k gate point. This discontinuity is due to the effect of UOR extraction. In the region below 10k gates, test patterns consist of only one pattern group, so the acceleration factor of 9 is solely due to the use of a vector processor. In the region above 10k gates, test patterns consist of multiple pattern groups. The number of groups increases with the circuit size. The size of a UOR increases with the number of groups; therefore, simulation time is almost proportional to the circuit size. Extrapolation shows that the acceleration factor may be about 90 at the 100k gate point. Thus, the acceleration factor solely due to UOR extraction for each pattern group is about 2 at 20k gates, and about 10 at 100k gates, demonstrating that VFSIM is especially advantageous in simulating large scale circuits.

In Table I, performance figures for VFSIM are summarized using average values. About 30k faults assumed in the scan circuit are almost all detected with the exception of 6 redundant faults. About 13 patterns are generated per flip-flop in order to detect faults including those around address decoders.

TABLE I
VFSIM performance

item	average
number of flip-flops	604
number of assumed faults in a scan circuit	30,270
number of redundant faults	6
number of undetected faults	0
number of test patterns	7,808
number of test pattern groups	2
VFSIM processing time (s)	150 (on S-820)

(1) Breakdown of simulation time : Details of the processing time in the simulation phase are shown in Table II for the same LSI circuit. UOR extraction overhead is only 1.3% of total processing time. Most of the time is devoted to SFP, which is executed in accordance with the vectorized event-driven technique, hence processing time mainly depends on event statistics shown in the next paragraph.

TABLE II
Processing time of the simulation phase (30k gates)

Process	time (s)	(%)
UOR extraction	4.0	1.3
Fault-free simulation	12.9	4.4
Fault simulation	278.9	94.3
Fault injection	15.4	5.2
Single fault propagation	238.8	80.7
Fault dropping, Logic value recovering	14.7	5.0
Fault dictionary	10.0	3.4
Total	295.8	-

(2) Event statistics : Processing time per event is plotted in Fig. 6. The small squares show the speed of SFP for gate rank at each vector (event queue) length. Maximum simulation speed (1.5M event evaluations/sec) is obtained in the region exceeding 500 in vector length. Simulation speed is decreased in the region below 100 in vector length.

In Fig. 6, the number of events processed at each event queue length in the gate rank is also plotted with solid circles. Most events are processed as vector data with vector length greater than 100. It is concluded that the vectorizing strategy introduced into VFSIM is adequate for fault simulation. The simulation speed (1.5M events/sec) is obviously larger than the speed (≈0.5 M events/sec) in the fault simulation hardware reported in [19], and larger than or comparable to the event-driven logic simulator prototypes in [15],[28].

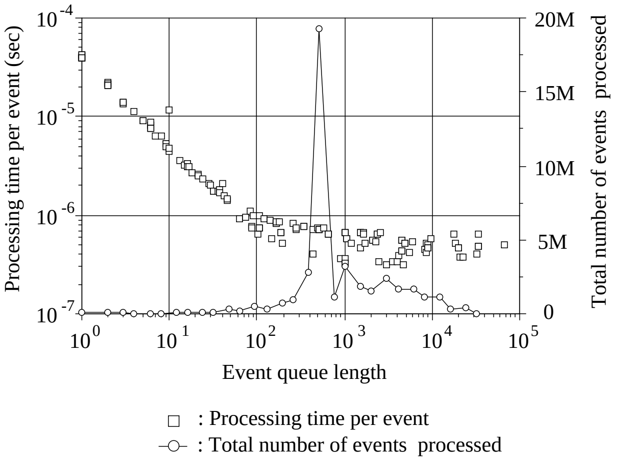


Fig. 6 Event statistics

As a result of several experiments on 20k - 30k gate LSIs, the average probabilities of an event occurring in an observable region of a GATE rank (P_n) and that in a FF rank (P_f) were 1.3x10^{-4} and 2x10^{-3}, respectively.

If events occur at random, the probability of an event being accepted in a pattern by a flip-flop equals approximately P_n. Thus, P_f is estimated to be P_{fc}, as follows:

$$P_{fc} = 1-(1-P_n)^s .$$

Here, s is the size of the pattern-set. In the case of s=511, P_{fc} is up to 0.064. P_f is much smaller than P_{fc}. This experimental result shows that the events do not occur at random, and are localized around several flip-flops in each pattern-set. Therefore, unnecessary calculation for the patterns in which a flip-flop accepts no events is negligible.

VFSIM evaluates 1.5M gate events/sec, which is equivalent to 68G gate evaluations/sec with non-selective trace method, which is several times faster than the method reported in [17],[28].

(3) Effect of UOR extraction : Actual effect of UOR extraction is shown in Fig. 7 for a 30k gate benchmark circuit. Number of master faults after fault collapsing is 20,057. Total number of test patterns applied is 13,194. There are three pattern groups: the first containing 6,100 patterns, the second containing 4,410 patterns, and the third containing 2,684 patterns. The number of processed faults for each test pattern group is 12316, 6982, and 3022, respectively. Among them 10607, 6428, and 3020 faults are detected, respectively (Fig. 7 area **A**1). Hence, 1709, 554 and 2 faults are processed wastefully for each test pattern group, respectively (Fig. 7 area **A**2). A major part of these undetected faults is related to the subset of provoked faults on the fixed value line shown in Fig. 7. Two faults remaining as undetected in the last pattern group are redundant. In the first and second test pattern groups, 7741 and 2468 faults in the UOR are excluded from the simulation, respectively (Fig. 7 area **B**). That is to say, 81.9%

and 81.7% of undetected faults for each test pattern group are excluded, respectively. The size of area **B** is almost the same as the size of area **A**1 and **A**2. Considering the reduction effect of the fault propagation path, it can be inferred that UOR extraction may accelerate this benchmark circuit by a factor of 3 or 4.

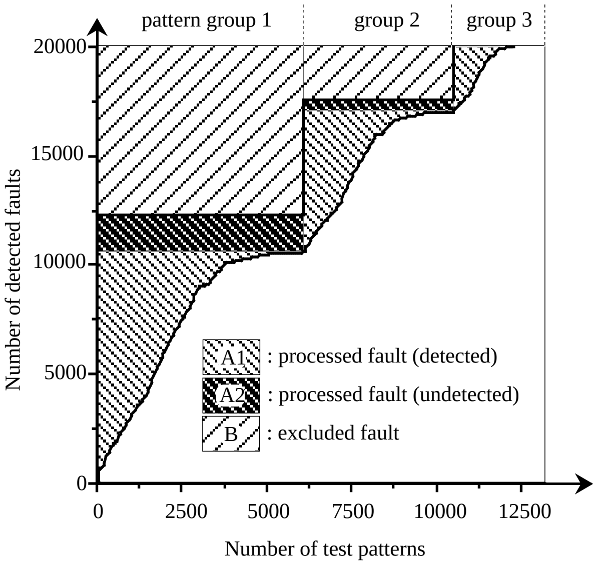


Fig. 7 Actual effect of UOR extraction

VI. Conclusions

A vectorized fault simulator (VFSIM), a fast fault simulator for synchronous sequential circuits, has been developed. VFSIM was applied to the random access scan circuit to generate a fault dictionary for 0-cycle test.

Two key techniques were introduced: vectorized event-driven method and unobservable region (UOR). The UOR is automatically extracted for each test pattern group using fixed input values, and the faults within the UOR are excluded from simulation during each test pattern group, resulting in an extreme reduction in the number of faults to be simulated. The faults not included in a UOR are propagated by the event-driven technique for multiple logic elements and for multiple test patterns simultaneously. For flip-flops, special iteration instructions are utilized to certify sequential behavior.

VFSIM was applied to several hundred LSI circuits of mainframe computers. All the LSIs were designed by random access scan technique. Logic size ranged from 3k to 40k gates. Application results show that VFSIM is one or two orders of magnitude faster than a conventional fault simulator designed for random access scan circuits. VFSIM is especially advantageous in simulating large scale circuits, depending on the effect of UOR extraction. VFSIM achieved more than 1M events/s performance.

Acceleration techniques provided in this paper are useful for a general class of synchronous sequential circuits not limited only to the scan circuit but also applicable to a combinational circuit. Optimal division into test pattern groups is a key point for maximizing the effect of UOR extraction.

Acknowledgment

The authors would like to acknowledge Y. Ohno and H. Horikoshi for the opportunity to develop this system, and S. Terai, Y. Tsuchiya and M. Hisamura for providing us with necessary support. Thanks are also extended to S. Ishiyama, E. Tsurumi, K. Moriwaki, and T. Numata, for their help in completing the total system.

References

[1] T. W. Williams and K. P. Parker: "Design for testability - A survey, "*Proc. of IEEE*, pp. 98-112, 1982.
[2] E. B. Eichelberger and T. W. Williams: "A logic design structure for LSI testing, "*DA Conf.*, pp. 462-468, 1977.
[3] M. H. Schulz et al. : "SOCRATES:A highly efficient automatic test pattern generation system, "*IEEE Trans. on CAD*, vol. 7, pp. 126-137, 1988.
[4] J. A. Waicukauski et al. : "Fault simulator for structured VLSI, "*VLSI System Design*, Dec., pp. 20-32, 1985.
[5] S. Funatsu et al. : "Test generation system in Japan, "*DA Conference*, pp. 114-122, 1975.
[6] H. Ando: "Testing VLSI with random access scan, "*COMPCON*, pp. 50-52, 1980.
[7] R. Takagi and R. Yoshino: "Custom VLSI test system, "*Proc. of ITC*, pp. 431-436, 1985.
[8] Y. Umetani et al. : "Some results on the application of FLT generator to a large scale computer, "*FTCS*, 1973.
[9] T. Niermann et al. : "PROOFS:A fast, memory efficient sequential circuit fault simulator, "*DA Conference*, pp. 535-540, 1990.
[10] H. Lee et al. : "HOPE:An efficient parallel fault simulator for synchronous sequential circuits, "*DA Conference*, pp. 336-340, 1992.
[11] T. Nishida et al. : "RFSIM:Reduced fault simulator, "*IEEE Trans. on CAD*, vol. 6, pp. 392-402, 1987.
[12] M. Denneau et al. : "Design and implementation of a software simulation engine, "*Computer-Aided Design*, vol. 15, pp. 123-130, 1983.
[13] T. Sasaki et al. : "HAL:A block level hardware logic simulator, "*DA Conference*, pp. 150-156, 1983.
[14] T. Blank: "A Survey of hardware accelerators used in computer aided design, "*IEEE Design and Test Comput.*, pp. 21-39, 1984.
[15] N. Ishiura et al. : "High-speed logic simulation on vector processors, "*IEEE Trans. on CAD*, vol. 6, pp. 305-321, 1987.
[16] F. Hirose et al. : "A method to generate tests for combinational logic circuits using an ultrahigh-speed logic simulator, "*ITC*, pp. 102-107, 1988.
[17] F. Ozguner et al. : "Vectorized fault simulation on the Cray-XP supercomputer, "*ICCAD*, pp. 198-201, 1988.
[18] N. Ishiura et al. : "Dynamic two-dimensional parallel simulation technique for high-speed fault simulation on a vector processor, "*IEEE Trans. on CAD*, vol. 9, pp. 868-875, 1990.
[19] S. Bose et al. : "Concurrent fault simulation of logic gates and memory blocks on message passing multicomputers, "*DA Conference*, pp. 332-335, 1992.
[20] S. Nagashima et al. : "Hardware implementation of VELVET on the HITACHI S-810 supercomputer, "*ICCAD*, pp. 390-393, 1986.
[21] Y. Takamine et al. : "Clock event suppression algorithm and its application to S-820 development, "*DA Conference*, pp. 716-719, 1988.
[22] Y. Kazama et al. : "Algorithm for vectorizing logic simulation and evaluation of VELVET performance, "*DA Conference*, pp. 231-236, 1988.
[23] M. Abramovici et al. : "Critical path tracing - An alternative to fault simulation, "*DA Conference*, pp. 214-220, 1983.
[24] S. B. Akers et al. : "Why less information from logic simulation more useful in fault simulation?, "*ITC*, pp. 786-800, 1990
[25] E. M. Rudnick et al. : "Methods for reducing events in sequential circuits fault simulation, "*ICCAD*, pp. 546-549,1991
[26] F. Maamari and J. Rajski : "The dynamic reduction of fault simulation, "*ITC*, pp. 801-808,1990
[27] E. G. Ulrich, and T. Baker, "The concurrent simulation of nearly identical digital networks, "*DA Conference*, pp. 145-150,1973
[28] A. Bataineh et al. : "Parallel logic and fault simulation algorithms for shared memory vector machines, "*ICCAD*, pp. 369-372,1992