A Test Clock Reduction Method for Scan-Designed Circuits †

Jau-Shien Chang Chen-Shang Lin

Department of Electrical Engineering
National Taiwan University
Taipei, Taiwan, R.O.C.

Abstract

In this paper, a novel test clock reduction method is proposed to generate a compact test scheme for scan-designed sequential circuits. The method comprises of two phases. First, from a given compact combinational test set, sequential fault propagation is performed after each scan-in operation to propagate the activated faults and simultaneously detect other undetected faults as many as possible. In the second phase, two active overlapping techniques are developed to maximize the overlap between successive scan-in patterns in pure scan mode. The experimental results show that the number of test clocks are reduced to half of full-scan. Furthermore, in comparison with the mix-mode test generator, TARF[4] requires 54% more test clocks than ours.

1 Introduction

Scan design is one of the most popular technique of design for testability. The main advantage of scan design is to convert the complicated sequential test problem into the simpler combinational one, and thereby the desirable test quality can be easier to be achieved. On the other hand, significantly lengthened test application time may be incurred by the shifting operations in the long scan chain. Therefore, in the past years, various methods have been proposed to reduce test application time of scan-designed circuits[1-9,13]. In this paper, we mainly concern the test time reduction of full scan-designed circuits.

Test time reduction of full scan-designed circuits can be performed either after test generation or during test generation. Test time reduction after test generation has been intensively investigated in [1-3,5,7]. In [1,2], based on the assumption that scan-in and scan-out flip-flops are disjoint, each of the scan-in patterns is divided into several segments first. Then, the application of a test pattern to the CUT is invoked when a segment but not a pattern is completely scanned in. Thus, the detection of some faults in response to the later patterns may be occasionally detected by these mixed (overlapping) patterns. And, the test time can be reduced due to the removal of these latter patterns. In [3], the test set is rearranged such that the current contents of scanned flip-flops can be completely reused by its next pattern. However, the method may not achieve reduction when the test patterns have little common parts. More complicate overlapping techniques are developed in [7]. The overlapping of successive patterns are obtained by precisely controlling the scan in-out operation through the utilization of the don't care bits. But, for a given test set with few don't care bits, the overlapping will be always corrupted by a few different bits in successive patterns. In summary, the effectiveness of these previous works is heavily restricted by the characteristics of the given test set.

To reduce the test application time during test generation for scan-designed circuits, the combinational test set can be generated as compact as possible. For test set compaction, several effective techniques have been developed to generate compacted test sets[6,8-9]. However in test application, further reductions on test time can be obtained by carefully rearranging the test patterns. In [3], *test generation for test application* has been considered to generate a test set with highly overlapping patterns when the scan chain topology is known a priori. However, the penalty of high overlap is obtained with a larger test set. Unfortunately, sacrificing the compactness of a test set to achieve the overlapping is not always successful in test time reduction. In [4,13], for sequential circuits with scan capability, a compromising method, *mix-mode test generation*, is proposed to generate a compact test application sequence. The advantage of mix-mode test generation is capable of dynamically switching between scan

† This work was supported in part by the National Science Council under contract number NSC-83-0404-E-002-055

0-7803-2102-2/94 $4.00

and non-scan modes during test generation. Thus, by means of proper mode switching, 100% fault coverage can be obtained through a shorter test sequence. But, for circuits with sequential hard-to-detected or redundant faults, the scan operations will occur frequently in test generation and worse results than full-scan may be obtained due to the lengthy fault propagation sequences in non-scan mode. In [13], some sophisticated and time-consuming criterions for mode switching have been proposed. However, because generating a compact combinational test set is not considered in non-scan mode, their results are even worse than those of [4].

In this paper, to generate a compact test application scheme for full scan-designed circuits, a novel test clock reduction method is proposed. The proposed method comprises of two phases. First, based on a compact combinational test set, sequential fault propagation is performed after each scan-in operation to propagate these activated faults and simultaneously detect the other undetected faults as many as possible in a shorter test sequence than scan-out. When the effectiveness of the above process has decreased below a pre-determined threshold, the next phase will be then invoked. In the second phase, from the compact combinational test set for these remaining faults of phase one, two *active* overlapping techniques, *maximum overlapping* and *make-compatible*, are developed to maximize the overlapping between the successive scan-in patterns. Different from the previous overlapping techniques, the test patterns to be scanned will be adaptly modified to increase the chance of overlapping without sacrificing the overall fault coverage.

The effectiveness of the proposed method are demonstrated by the experimental results on 22 ISCAS'89 benchmark circuits. On an average, our test clocks are only half of those for full-scan. Furthermore, in comparison with the mix-mode test generator, TARF[4] requires 54% more test clocks than our reduction result.

The remaining part of this paper is organized as follows. The features of the proposed test clock reduction method and some notations are described in Section 2. Sections 3 introduces the details of maximum overlapping. In section 4, the make-compatible operation will be described. The experimental results are given in section 5. Section 6 gives the conclusions.

2 Test Clock Reduction

In this section, the proposed test clock reduction method will be introduced.

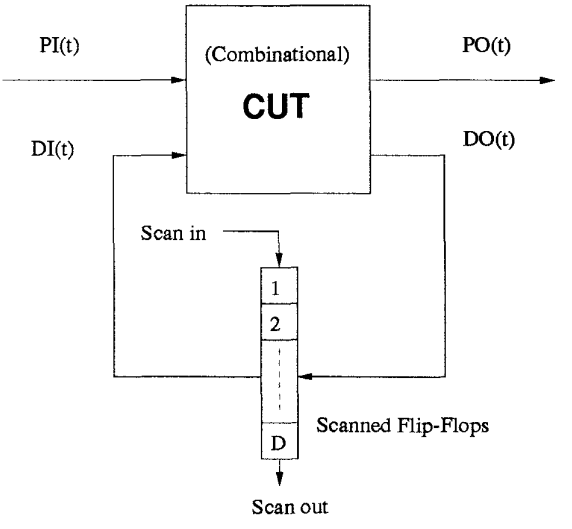

Fig. 1 A scan-designed sequential circuit

2.1 Total Test Clocks

For a full scan-designed circuit as shown in Fig. 1, each test pattern t for the CUT (Circuit Under Test) consists of two parts: the part applied to PI denoted as $PI(t)$ and the other part for the FFs (Flip-Flops) as $DI(t)$. In the test application of t, $DI(t)$ must first be shifted into the scan path which generally consists of all FFs in the circuit. Let $RESP(t)$ be the response of CUT after applying t. $RESP(t)$ can be similarly divided into $PO(t)$ and $DO(t)$, where $PO(t)$ is the response appearing at PO and $DO(t)$ is that to be loaded into FFs. $DO(t)$ must also be shifted out of the scan path for observation, which may overlap with the scan-in of the next pattern. Given a fault set F and a set of patterns P, the detectable faults of P in F is denoted as $DET(P, F)$. For a test set T of F and a pattern $t \in T$, the *essential faults* of t, denoted as $ESS_T(t, F)$, represents the set of faults in F that can only be detected by t but not by others in T.

Let T be the test set to be applied and D be the number of shifts for each pattern (in general, D is the number of scanned flip-flops), then total test time, t_T, is

$$t_T = (\mid T \mid +1) * D * S + \mid T \mid * C \tag{2.1}$$

where S and C are the periods of a shifting clock and system clock, respectively. Assuming $S = C$, t_T becomes

$$t_T = ((\mid T \mid +1) * D + \mid T \mid) * S \tag{2.2}$$

The total test clocks of T for a scan-designed cir-

cuit, TTC, is then

$$TTC = (\mid T \mid +1) * D + \mid T \mid \qquad (2.3)$$

It can be seen that TTC is dominated by $\mid T \mid$ and D. Therefore, to reduce the total test clock, we can either compact the test set T as small as possible or(and) shorten the number of shifts for each pattern. If $\mid T \mid \gg 1$ and $D \gg 1$, then $TTC \cong \mid T \mid * D$. In [3], because test generation for test application is considered, the goal of test generation concentrates on how to increase the overlapping parts among patterns but not to compact the test set. However, it can be seen from the above discussions that, if the test size, $\mid T \mid$, becomes much larger than the compacted one by a certain ratio, then D must be reduced at least by the same ratio to keep TTC intact. Unfortunately, this is hard to achieve especially for a CUT with a large number of scanned FFs. Therefore, in our works, the strategy for test clock reduction will start from a compact test set of the CUT, then further reduction processes are proceeded based on the compact test set. For test set compaction, currently an effective compaction tool, TSR[6], has been adopted.

2.2 Test Clock Reduction Method

The proposed test clock reduction method is introduced in this subsection. The method comprises of two phases. In Phase 1, *Scan and Propagation*, from a compact combinational test set, one scan-in operation followed by fault propagation process is performed iteratively to achieve a fault coverage as high as possible in a constrained length of test sequence. In Phase 2, *Scan and Overlapping*, two active overlapping techniques are used to arrange the test application in pure scan mode as compact as possible.

(1) Phase 1: Scan and Propagation

In mix-mode test generation, by means of switching in non-scan and scan mode, an ideal resultant test sequence is shown as Fig. 2(a). In this sequence, some timely scan operations will be incorporated to enrich the fault coverage of each scan segment and quickly improve the overall fault coverage. A *scan segment* is the clock interval between the first scan-in clocks of a scan-in operation and the next scan-in operation. However, for a CUT with some sequentially hard-to-detected or redundant faults, a test sequence shown as Fig. 2(b) with a lot of scan operations and lengthy non-scan sequences could occur. In this case, the number of test clocks will be increased significantly and even greater than that by pure scan mode only. Therefore, the key idea of Phase 1 is to devise a scheme which retains the nice feature of mix-mode

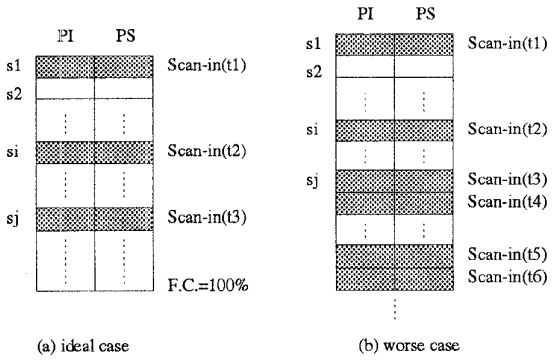

Fig. 2 Test sequence of mix-mode test generation

test generation without incurring excessively lengthy sequence.

Given a compact combinational test set of the CUT, in Phase 1, the pattern with the largest fault coverage is chosen for scan-in. After applying this pattern, in addition to some faults detected at POs, there are other faults whose effects have been propagated to FFs. For these faults, a propagation process to POs is performed. The process can be any fault propagation algorithm in sequential ATPG. In this work, a CONTEST[11]-like sequential test generator is used for fault propagation and simultaneously detect the other undetected faults as many as possible during propagation. To avoid lengthy propagation sequences for some hardly detected faults, the following criterion is used to decide when to scan in the next pattern.

Scan-In Criterion: *The scan-in of a new pattern occurs when the size of of the last non-scan segment in the test sequence is greater than the number of scanned FFs and no fault is detected in this segment.*

The last segment without detecting any faults can then be discarded because it does not contribute to fault detection. When the fault coverage increases gradually in Phase 1, in general, the scan-in operations will occur more frequently. However, in a test sequence, too many scan-in operations will cause the previous sequential fault propagation sequences redundant and meaningless. To avoid such a situation, the following Stop Criterion is used to decide when to stop Phase 1.

Stop Criterion: *Let C_i be the number of clocks of the last scan segment S_i and DET_{Si} be the number of the detected faults in S_i. If*

$$DET_{S_i} < (C_i/D) * (\mid F \mid / \mid T \mid) \qquad (2.4)$$

then the mix-mode test generation will be stopped,

where | F | is the size of combinational detectable faults of the CUT, | T | is the size of the test set for F and D is the number of the scanned FFs.

Condition (2.4) implies that the effectiveness of fault detection in non-scan mode has fallen below the average detectability of scan mode. In this case, Phase 1 should be stopped and proceeds to Phase 2 which employs scan operation more efficiently.

(2) Phase 2: Scan and Overlapping

In Phase 2, the pure scan mode is adopted to detect the remaining faults in Phase 1. Starting from a compact test set of these remaining faults, the pattern overlapping techniques are used to arrange the test set such that there are as much overlapping as possible between successive pattern pairs. In this phase, two active overlapping techniques, maximum overlapping and make-compatible, have been developed. In maximum overlapping, the current contents of scanned FFs are pre-shifted out some *signature faulty bits* in order to safely reuse the remaining bits in FFs. The signature faulty bits are the bits at which a given fault set, single stuck-at faults in our case, would manifest the fault effect nearest to the scan-out pins when applying the test pattern. Then, successive shifting operations are proceeded to maximally overlap these remaining bits with the next scan-in pattern. Make-compatible is to enhance the effectiveness of maximum overlapping. The blocking bits of the next scan-in pattern for overlapping are modified such that as many bits as possible in FFs can be reused. These two techniques will be described detailedly in the following section. Note that, in this phase, we assume the scan chain topology has been determined.

3 Maximum Overlapping

In this section, the first proposed overlapping technique, *Maximum Overlapping*, will be introduced.

To reduce the test time, the goal of maximum overlapping is to optimally reuse the current contents of FFs in the next scan-in operation, i.e. to overlap the scan-in and scan-out clocks of two consecutive patterns as many as possible. In [3], two special cases for pattern overlapping have been proposed and described as follow. For a pattern t_i and its successive pattern t_{i+1}, if *the fault effects of each detectable fault of* t_i *can be observed at PO* and one of the two cases listed below is satisfied, then the scan-in operation of t_{i+1} can be completely removed due to the re-use of the current contents of FFs.

case (a) *complete DO-reuse*: $DO(t_i) \equiv DI(t_{i+1})$,

case (b) *complete DI-reuse*: $DI(t_i) \equiv DI(t_{i+1})$,

where $'\equiv'$ is the *compatible* operator. Two vectors are said to be compatible if all the corresponding bits are either with the same logic value or one of them is $'x'$ (don't care). For example, the two vectors, $v1 = (0x01)$ and $v2 = (010x)$, are compatible and denoted as $v1 \equiv v2$.

From the above description, to reuse the DO-part or DI-part of t_i, the pre-condition must be satisfied first, i.e., all the detectable faults of t can be observed at PO. Thus, even only one detectable fault fails to be observed at PO, there is no chance of reuse. This is clearly too restrictive. In fact, for the case of DO-reuse, although some fault effects can not be detected at PO, if all the signature faulty bits have been pre-shifted out, these remaining bits in FFs can be safely reused. This fact is stated more clear in the following observation and provides a chance to *maximally* but not *completely* reuse the current content of FFs.

Observation 1: (Maximum DO-reuse) *Under single fault assumption, after applying a pattern t to the CUT and loading its response DO(t) into the scanned FFs, the presence of a detectable fault f of t can be determined from either POs or its first fault effect bit in DO(t). Furthermore, if POs and the first fault effect bits of all detectable faults of t in DO(t) have been observed to be fault-free, then these faults are not present and DO(t) is the fault-free response.*

The above observation can be easily justified. For a given detectable fault of a pattern t, its presence in a circuit can be observed from POs or $DO(t)$ in which one fault effect bit will be enough. In addition, for all detectable faults, their presence can also be determined from POs and one fault effect bit in $DO(t)$ for each fault. If any of these faults is determined to be present in a circuit, then the circuit is declared faulty and, for testing purpose, no further test application will be needed. Otherwise, none of the detectable faults of t is present and the response $DO(t)$ must be the fault-free response. The single fault assumption ensures that no fault masking effect can occur.

Based on Observation 1, the possible fault effects of each scanned pattern will be analyzed accurately to pre-shift out a certain bits before reuse. To reuse the DO-part of a pattern t, for each fault f detected by t, the possible faulty bit nearest to the scan-

out pin is recorded, $MINBIT(f)$. Then, find the largest $MINBIT(f)$ for all the detected faults of t to be the $MAX(MINBIT(f))$. In such way, even when the pre-condition is not satisfied, the $DO(t)$ can be safely reused by shifting out $MAX(MINBIT(f))$ bits first. For the example shown in Table 1, suppose $DET(\{t\}, F) = \{f1, f2, f3\}$, $DO(t) = (1001010)$ and the $MINBIT$ of each fault as listed in $MINBIT(f)$ column, the largest $MINBIT(f)$ is found to be 4. Therefore, $MAX(MINBIT(f))$ is set to 4 and $DO(t)$ will be shifted out 4 bits before reuse.

	FFs 7654321	$MINBIT(f)$
ff	1001010	-
$f1$	d001d10	3
$f2$	1d0d010	4
$f3$	1d0d0d0	2

ff: Fault Free response
d: denote the fault effect

Table 1. $MAX(MINBIT(f))$ computation

In the above example, after shifting out $DO(t) = (1001010)$ by 4 bits, there are still 3 bits remained in the scanned FFs and $DO(t)$ becomes ($xxxx$100). To reuse the DO(t) after pre-shifting, patterns not yet applied and compatible with $DO(t)$ will be chosen as the next scan-in pattern. But, if no pattern is compatible with $DO(t)$, then, a *shift-and-compare* process is tried to reuse these remaining bits in $DO(t)$ as many as possible. First, $DO(t)$ is shifted out one bit and becomes ($xxxxx$10). Then, the comparison of compatibility between $DO(t)$ with these remaining patterns will be performed to choose the next scan-in pattern. If it fails, $DO(t)$ is shifted out again and the compatibility comparison is performed iteratively. The process will be stopped when a pattern is chosen or all the specified bits in $DO(t)$ have been shifted out. For the latter case, the pattern not yet applied and with the highest fault coverage is chosen as the next scan-in pattern.

For DI-reuse, when the pre-condition is not satisfied, the load operation of all FFs must be performed. In other words, no partial overlapping with pre-shifting is possible. However, if extra hardware is permitted, the problem of satisfying the pre-condition can be alleviated by improving the observability of CUT such as adding an extra parity output[3].

4 Make-Compatible

In this section, to enhance the effectiveness of maximum overlapping, the second proposed technique, *Make-Compatible*, will be introduced.

4.1 Principle

Recall that, maximum overlapping is performed to optimally reuse the current contents of scanned FFs in the next scan-in operation. However, for a test set lacking compatibility between patterns, the reduction on test clock will be insignificant. In other words, the effectiveness of maximum overlapping depends heavily on the characteristics of the given test set. Therefore, to further enhance the reuse in maximum overlapping, we propose the *make-compatible* technique which is able to actively modify the test set for compatibility and increase the chance of reusage while keeping the overall fault coverage intact.

The basic idea of make-compatible is based on the over-specification property of the test set to be scanned. For a test set T of F, it is usually unnecessary to preserve all bits of each pattern in T specified for keeping $DET(T, F)$ intact. For example, for a CUT with one PI and four FFs, if $T = \{t1 = (10111), t2 = (01111), t3 = (10011)\}$ is a test set of F. For the test set T, if $t1$ is the first applied pattern and $DO(t1) = (0101)$, whatever $t2$ or $t3$ is chosen as the next pattern, it is not possible to reuse the whole content of FFs. However, if $t2$ can be modified to $t2' = (0x1x1)$ without reducing the overall fault coverage, then $t2'$ can be chosen as the next applied pattern without any shift-in operation because $DO(t1) \equiv DI(t2')$. The modification on $t2'$ is said to *make compatible* with $DO(t1)$. Thus, when choosing the next applied pattern, make-compatible can be performed to increase the chance of reusage by actively modifying these different bits of the next pattern.

In make-compatible, the modification on a test pattern is to change some bits from '1' or '0' to 'x'. For a specified bit of a pattern t in a test set T, the bit is said to be *raised* if its value is changed to 'x' while keeping the fault coverage of T intact. The preservation of fault coverage is evidently essential. Therefore, to keep the overall fault coverage intact, the following observation is used to guide the raising operation in make-compatible[6].

Observation 2: *Given a fault set F and a test set T of F, for a pattern $t \in T$, if t is substituted by t' such that $DET(\{t'\}, F) \supseteq ESS_T(t, F)$, then $T' = T - \{t\} + \{t'\}$ has at least the same fault coverage as T.*

The correctness of Observation 2 can be demonstrated as follow. We have known that

$$DET(T,F) = DET(T,F) - DET(\{t\},F) + ESS_T(t,F),$$

where '$-$' and '$+$' represents the set difference and set union. Therefore, if $ESS_T(t,F) \subseteq DET(\{t'\},F)$, then

$$DET(T,F) \subseteq DET(T,F) - DET(\{t\},F) + DET(\{t'\},F)$$

Thus, $T' = T - \{t\} + \{t'\}$ at least has the same fault coverage as T. To find t' such that $DET(\{t'\},F) \supseteq ESS_T(t,F)$ is not trivial. Fortunately, in our application, t' is expected to be obtained from raising some specified bits in t to 'x'. Thus, make-compatible can be performed by unspecifying some bits of t while monitoring the detectability of $ESS_T(t,F)$ by fault simulation.

Step	DI-part		Operation	$\equiv DI(t_1)$?
Init	t_1	101000	-	-
	t_2	010011	-	No
	t_3	111110	-	No
1	t_1	101000	-	-
	t_2	x1x010	make compatible	No
	t_3	1x1110	make compatible	No
2	t_1	x10100	shift out 1 bits	-
	t_2	010xx1	make compatible	No
	t_3	11x110	make compatible	No
3	t_1	xx1010	shift out 2 bits	-
	t_2	01x011	make compatible	No
	t_3	111x10	make compatible	Yes

Table 2. Example of shift-and-make-compatible

Based on Observation 2, the shift-and-compare process described in maximum overlapping can be refined to a *shift-and-make-compatible* process for achieving more reductions on test clocks. The new process is described by the following example. As shown in Table 2, for a test set $T = \{t_1, t_2, t_3\}$, after applying t_1, shift-and-make-compatible is performed to select the next applied pattern. Only DI-reuse is considered in the example for simplicity. Initially, $DI(t_1)$, $DI(t_2)$ and $DI(t_3)$ are mutually incompatible. In Step 1, after the make-compatible operation on these incompatible bits, the DI-parts of the three patterns are still incompatible. Therefore, a shift out operation is performed on $DI(t_1)$. In step 2, the $DI(t_1)$ is shifted out one bit first, then make-compatible is performed for $DI(t_2)$ and $DI(t_3)$ again. Unfortunately, it also fails. However, in Step 3, after the content of $DI(t_1)$ is shifted out two times, the DI-parts of t_1 and t_3 become compatible through make-compatible. Thus, 4 scan clocks can be saved for the scan-in of t_3. If only shift-and-compare process is used, there is at most one scan clock which can be saved in this example.

4.2 Phase-2 Algorithm

The detail algorithm of the second phase of test clock reduction with maximum overlapping and make-compatible, is shown in Fig. 3. The algorithm consists of the main routine $TCR\_Ph2()$ and the subroutine *max_overlap*().

In $TCR\_Ph2()$, first, the test pattern t with maximum fault coverage is chosen as the first applied pattern. To choose the successive pattern of the current applied pattern t, the number of pre-shifts of t is computed by the subroutine $MAX\_MINBIT(t)$. Then, the maximum saved clocks by DO-reuse and the corresponding successive pattern are obtained by calling *max_overlap*(). However, if the number of pre-shift is zero, the reuse of $DI(t)$ becomes possible and the saved clocks are also included to compare with that of DO-reuse. At last, the pattern with the maximal saved clocks with DI-reuse or DO-reuse is chosen as the successive pattern of t. The process is iterated until the given test set T is exhausted.

In the subroutine, *max_overlap*(), the successive pattern t' of the current applied pattern t is obtained by searching in the remaining patterns of T such that the saved clocks is largest. The searching process is performed in a greedy way. To find the next pattern t' with largest saved clocks, each time *reuse_vector*, $DI(t)$ or $DO(t)$, is shifted out one bit, and all remaining patterns in T are tried to make compatible with *reuse_vector*. If found, then *max_overlap*() terminates and the pattern and its saved clocks are recorded. Otherwise, the *reuse_vector* is shifted out one more bit and the make-compatible process is repeated. The shifting process will continue until all bits of *reuse_vector* have been shifted out. If there is no saved clock possible, these remaining patterns in T with the largest fault coverage will be chosen as the successive pattern.

5 Experimental Results

The proposed two-phase test clock reduction method (ACTIVE) equipped with active overlapping techniques, maximum overlapping and make-compatible, has been implemented on SUN4/SPARC2 workstation. To show the effectiveness of our method, 22 benchmark circuits of ISCAS'89 are used as test examples. In the evaluation, the topology of scan-chain is assumed to be known and is in the natural ordering.

```
/* T: the given test set applied to CUT */
/* F: the fault list of CUT */
/* next_t: global variable to store the next scan-in pattern
*/
TCR_Ph2() {
    choose a t ∈ T with maximum fault coverage;
    while (T is not empty) {
        T = T−{t} ;
        pre_shifts = MAX_MINBIT(t) ;
        F = F−DET({t},F);
        /* Compute the saved clocks by DO-reuse */
        DO_save_clk =
            max_overlap(T,DO(t),pre_shifts) ;
        DO_next_t = next_t ;
        load_response = TRUE ; use_DO = TRUE ;
        /* Try DI-reuse */
        if (pre_shifts == 0) {
            /* DET(t) can be observed at PO */
            /* Compute the saved clocks by DI-reuse */
            DI_save_clk =
                max_overlap(T,DI(t),pre_shifts) ;
            DI_next_t = next_t ;
            if (DI_save_clk > DO_save_clk) {
                load_response = FALSE ;
                push_test_seq(t,pre_shifts,load_response) ;
                use_DO = FALSE ;
                t = DI_next_t ;
            }
        }
        if (use_DO) {
            push_test_seq(t,pre_shifts,load_response) ;
            t = DO_next_t ;
        }
    }
}

max_overlap(T,reuse_vector,pre_shifts) {
    shift_no = pre_shifts ;
    while (shift_no ≤ number_of_DFF) {
        D_content = reuse_vector ;
        shift_out(D_content,shift_no) ;
        foreach pattern t' in T {
            if (make_compatible(D_content,DI(t'),t')) {
                next_t = t' ;
                return(no_of_DFF-shift_no) ;
            }
        }
        shift_no++ ;
    }
    choose a t ∈ T with maximum DET as next_t ;
    return(0) ;
}
```

Fig. 3 Algorithm of proposed overlapping techniques

Ckts	PI	PO	SFF	SEQ-FC%	#P
s208	11	2	8	63.72	27
s298	3	6	14	85.71	25
s344	9	11	15	96.20	14
s349	9	11	15	95.71	14
s382	3	6	21	90.98	25
s386	7	7	6	81.77	63
s420	19	2	16	41.63	44
s444	3	6	21	89.45	25
s510	19	7	6	0.00	57
s526	3	6	21	75.32	50
s641	35	24	19(35*)	86.30	24
s713	35	23	19(35)	80.90	24
s820	18	19	5	81.88	96
s832	18	19	5	81.38	97
s838	35	2	32	29.64	76
s953	16	23	6(29)	7.78	76
s1196	14	14	17(18)	99.76	124
s1238	14	14	17(18)	94.69	133
s1423	17	5	74	67.39	32
s1488	8	19	6	92.60	104
s1494	8	19	6	92.10	102
s5378	35	49	179	74.02	109

SFF: number of scanned flip-flops
SEQ-FC%: sequential fault coverage
#P: size of the combinational test set
* number of total flip-flops

Table 3. Statistics of ISCAS'89 benchmark circuits

In Table 3, the statistics of these benchmark circuits are shown, including the number of primary input (PI), primary output(PO), scanned FFs (SFF), sequential fault coverage (SEQ-FC%) and the combinational test size (#P). The sequential fault coverage is obtained from [12]. These combinational test sets are generated by a PODEM-like ATPG and compacted by the compaction tool, TSR[6]. Note that, for some circuits in ISCAS'89 benchmark, part of flip-flops are only the output buffers and can be removed from the scanned chain. It results in the difference between the number of actually scanned FFs and total FFs for some circuits in Table 3. The results of test clock reduction by ACTIVE are shown in Table 4 and the comparison with previous works is provided in Table 5.

In Table 4, the results of ACTIVE on the test clock reduction are shown. The second column, FSCAN, lists the number of test clocks of pure full-scan for the original compacted test sets in Table 3. The ACTIVE-1 column shows the results obtained by the first phase of ACTIVE only, with the Stop Criterion for Phase 1 turned off. In ACTIVE-2 column, the results by the second phase of ACTIVE only are shown, i.e., by maximum overlapping and make-compatible. The complete results of ACTIVE are shown in the last column. The ratio after each data is the normalized result with

respect to the result of ACTIVE.

Ckts	FSCAN	ACTIVE-1	ACTIVE-2	ACTIVE
s208	251/1.43	229/1.31	182/1.04	*175
s298	389/1.63	289/1.21	317/1.33	*238
s344	239/1.51	*148/0.94	224/1.42	158
s349	239/1.50	181/1.14	227/1.43	159
s382	571/1.13	572/1.13	*504/0.99	506
s386	447/1.65	340/1.25	341/1.26	*271
s420	764/1.58	904/1.86	*431/0.89	485
s444	571/1.36	565/1.34	448/1.06	*421
s510	405/2.50	*159/0.98	276/1.70	162
s526	1121/1.42	1114/1.41	947/1.20	*789
s641	499/1.59	321/1.02	370/1.18	*314
s713	499/1.63	354/1.15	367/1.20	*307
s820	581/1.55	448/1.19	477/1.27	*375
s832	587/1.66	449/1.27	493/1.39	*354
s838	2540/2.06	2847/2.31	*1163/0.94	1235
s953	538/2.16	327/1.31	377/1.50	*249
s1196	2249/5.45	687/1.66	576/1.39	*413
s1238	2411/6.13	685/1.74	676/1.72	*393
s1423	2474/1.42	2578/1.48	2285/1.31	*1746
s1488	734/2.05	372/1.04	587/1.64	*358
s1494	720/1.97	382/1.05	591/1.62	*365
s5378	19799/1.49	27998/2.11	14252/1.07	*13262
AVG	2.04	1.36	1.30	1.0

FSCAN: test clocks needed for pure Full Scan
ACTIVE-1: mix-mode test generation based on CONTEST
ACTIVE-2: maximum overlapping+make-compatible
ACTIVE: complete results of proposed two-phase method
AVG: average ratio w.r.t. the results of ACTIVE
*: best results

Table 4. Results of test clock reduction

From Table 4, it can be seen that, the average test clocks of ACTIVE are only one half of those for pure full-scan (FSCAN) with well-compacted test sets. The resultant test clocks of ACTIVE-1 and ACTIVE-2 are about 30% larger than ACTIVE but still 70% less than FSCAN. The performance on individual circuit however shows the discrepancy of ACTIVE-1 and ACTIVE-2 while ACTIVE is able to obtain best results in most cases. In Table 4, the best results of 17 out of 22 circuits, marked by '*', are obtained by ACTIVE, three by ACTIVE-2 and only two by ACTIVE-1. For these circuits with more scanned FFs and lower sequential fault coverage such as s838, s1423 and s5378, the results of ACTIVE are far superior to that of ACTIVE-1. This clearly demonstrates the improvements of ACTIVE on mix-mode test generation and shows the effectiveness of Stop Criterion described in Section 2. On the other hand, for those circuits, which are sequentially easily testable and have high sequential fault coverage such as s1196, s1238, s1488 and s1494, ACTIVE clearly outperforms. The fact shows the necessity of combining sequential fault propagation with scan operations.

The comparisons of ACTIVE with TARF[4], a mix-mode test generator, are shown in Table 5. It can be seen that, except s349, our results are significantly better than those of TARF. On the average, TARF needs 54% more test clocks than ACTIVE. Similar to ACTIVE-1, ACTIVE is especially successful than TARF in those circuits with many scanned FFs and low sequential fault coverage, such as s838 and s1423. Since TARF did not depend on the topology of scan-chain, the comparison indicates the capability of ACTIVE to exploit the additional information on scan-chain.

	Test Clocks		CPU-time(ACTIVE)		
Ckts	TARF[4]	ACTIVE	Ph-1	Ph-2	Total
s208	-	175	1.7	1.2	2.9
s298	376/1.58	238	3.6	0.9	4.5
s344	166/1.05	158	2.2	1.2	3.4
s349	125/0.79	159	2.5	1.2	3.7
s382	680/1.34	506	14.2	1.5	15.7
s386	376/1.39	271	4.4	2.8	7.2
s420	737/1.52	485	8.3	5.9	14.2
s444	788/1.87	421	16.2	1.8	18.0
s510	192/1.19	162	5.3	1.4	6.7
s526	1551/2.02	768	17.3	4.0	21.3
s641	-	314	13.6	7.6	21.2
s713	-	307	17.1	8.2	25.3
s820	617/1.65	375	18.0	10.2	28.2
s832	589/1.66	354	18.7	8.3	27.0
s838	3263/2.64	1235	54.2	32.6	86.8
s953*	924/3.71	249	14.9	6.5	21.4
s1196	465/1.13	413	36.6	11.8	48.4
s1238	448/1.14	393	46.1	15.6	61.7
s1423	3222/1.85	1746	200.3	21.5	221.8
s1488	641/1.79	358	32.2	11.2	43.4
s1494	582/1.59	365	33.7	11.0	44.7
s5378	-	13262	1652.0	242.6	1894.6
AVG	1.54	1.0	-	-	-

ACTIVE: proposed two-phase method
*: removed from AVG due to different no. of scanned FFs
CPU-time: seconds on SUN4/SPARC2

Table 5. Comparison of test clocks of ACTIVE and TARF[4]

The CPU-time for ACTIVE is also shown in Table 5. Due to the complexity of sequential test generation, in comparison with Phase 2, Phase 1 contributes the dominant part of CPU-time. A practical compromise would be to employ Phase 2 only for test clock reduction on larger circuits such as s1423 and s5378.

6 Conclusions

To minimize test cost for full scan-designed circuits, a novel test clock reduction system, ACTIVE, has been developed. The proposed method comprises two phases. In the first phase of ACTIVE, based on a

compact combinational test set, fault effect propagation is performed in non-scan mode after each scan-in operation to detect these sequential easy-to-detected faults. To preserve the compactness of the test set, two criteria, Scan Criterion and Stop Criterion, have been used to decide when to scan in a new pattern and when to stop the phase. Phase 1 is proceeded until the effectiveness of fault detection has decreased below a pre-determined threshold. In the second phase, from the compact combinational test set for these remaining faults from phase one, two active overlapping techniques, maximum overlapping and make-compatible, have been developed to maximize the overlapping between successive scan-in patterns. With these two techniques, the dependency of test clock reduction on the characteristics of original test set can be greatly reduced.

The effectiveness of ACTIVE has been evaluated with 22 ISCAS'89 benchmark circuits. On an average, our test clocks are only half of those for full-scan. Furthermore, in comparison with the mix-mode test generator, TARF requires 54% more test clocks than ACTIVE.

References

[1] M. S. Abadir and M. A. Breuer, "Scan Path With Look Ahead Shifting (SPLASH)," *Proceedings of 1986 International Test Conference*, pp. 696-704.

[2] M. S. Abadir, "Efficient Scan Path Testing Using Sliding Parity Response Compaction," *Proceedings of 1987 International Test Conference*, pp. 332-335.

[3] H. Fujiwara and A. Yamamoto, "Parity-Scan Design to Reduce the Cost of Test Application," *Proceedings of 1992 International Test Conference*, pp. 283-292.

[4] S. Y. Lee and K. K. Saluja, "An Algorithm to Reduce Test Application Time in Full Scan Designs," *Proceedings of 1992 ICCAD*, pp. 17-20.

[5] I. Pomeranz and S. M. Reddy, "A Test Application Scheme for Embedded Full-Scan Circuits to Reduce Testing Costs," *Proceedings of First Asian Test Symposium*, pp. 206-211, 1992.

[6] J. S. Chang and C. S. Lin, "Test Set Compaction for Combinational Circuits," *Proceedings of First Asian Test Symposium*, pp. 20-25, 1992.

[7] W. C. Lai, C. P. Kung and C. S. Lin, "Test Time Reduction in Scan Design Circuits," *Proceedings of 1993 EDAC*, pp. 489-498.

[8] Gert-Jan Tromp, "Minimal Test Sets for Combinational Circuits," *Proceedings of 1991 International Test Conference*, pp. 204-209, 1991.

[9] I. Pomeranz, L. N. Reddy and S. M. Reddy, "COMPACTEST: A Method to Generate Compact Test Sets for Combinational Circuits," *Proceedings of 1991 International Test Conference*, pp. 194-203.

[10] P. Goel, "An Implicit Enumeration Algorithm to Generate Tests for Combinational Logic Circuits," *IEEE Trans. on Computers*, Vol. C-30, No. 3, Mar. 1981.

[11] V. D. Agrawal, K. T. Cheng and P. Agrawal, " CONTEST: A Concurrent Test Generator for Sequential Circuits," *Proceedings of 25th Design Automation Conference*, 1988, pp. 84-89.

[12] W. T. Cheng and S. Davidson, " Sequential Circuit Test Generator (STG) Benchmark Results," *Proceedings of Int. Symp. of Circuits & Systems*, 1989, pp. 1931-1941.

[13] D. K. Pradhan and J. Saxena, " A Design for Testability Scheme to Reduce Test Application Time in Full Scan," *Proceedings of IEEE VLSI Test Symposium*, 1992, pp. 55-60.