Test time reduction for scan-designed circuits by sliding compatibility

J.-S. Chang
C.-S. Lin

Indexing terms: Logic testing, Scan design, Stuck-at faults, Test time reduction

Abstract: A postgeneration method for test time reduction of scan-designed circuits is developed. The maximum overlapping condition between consecutive applied patterns is identified. The application of the condition facilitated with the developed active sliding compatibility process significantly reduces the number of test clocks. It is demonstrated that the test clocks can be reduced by 50% on average from given test sets. Further evaluation shows that, for parity scan, the test clocks required by the authors' method are only 41% of those elsewhere.

1 Introduction

To increase the fault coverage and the test quality of sequential circuits, the now popular scan design had been proposed [13]. The main feature of scan design is that, by scanning the flipflops, the complicated sequential test problem is transformed into the much simpler combinational one, to achieve the desirable test quality. However, the necessity to shift test patterns and responses in the long scan chain may incur significant increase in test application time as well as test cost. To minimise the additional testing time while retaining test quality of scan design, various scan clock reduction methods have been proposed [1–5, 7].

These previous works of test time reduction for scan designed circuits can be divided into the following two general approaches: reduction during test generation or after test generation. To reduce the test application time during test generation, one way is to generate as compact a combinational test set as possible. Several effective techniques have been developed to generate such compacted test sets [6, 8, 9]. However, as pointed out in Reference 3, test compaction alone is not enough to reduce test application time because further reductions can always be obtained by carefully rearranging the test patterns. Alternatively [4, 7], the test generation is made to switch between scan mode and nonscan mode, and the scan mode is used only to achieve the required fault coverage. However, for circuits with many sequential hard-to-be-detected faults, scan operations will occur frequently and result in test time worse than that of full scan [4]. Moreover, complicated sequential test generation is involved in the process with the consequence of prolonged generation time.

Paper 1520E (C2, E10), received 31st March 1994

The authors are with the Department of Electrical Engineering, National Taiwan University, Taipei, Taiwan, Republic of China

The postgeneration approach is characterised by reducing scan requirements from a given test set. In References 1 and 2, based on the assumption that scan-in and scan-out flipflops 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 after scanning in only a segment but not necessarily a complete pattern. Thus some faults originally detected by the later patterns may become detectable by these segmented patterns, and the test time can be reduced due to pattern removal. More elaborate pattern overlapping techniques have also been developed [7]. The overlapping of successive patterns are obtained by precisely controlling the scan operation through the utilisation of don't care bits in the scanned flipflops. However, for a given test set with few don't care bits, the overlapping often collapses owing to the minor difference between successive patterns. In summary, the effectiveness of these previous works is heavily restricted by the characteristics of the given test set.

Recently, a novel idea, parity-scan design, has been proposed [3], in which an extra parity output is introduced to the scanned flipflops to enhance the observability. With this simple parity output, the scan operation can be eliminated if the fault effects can be easily observed, and the content of flipflops can be reused. To exploit the enhanced facility, direct test time reduction has been considered [3] by generating a pattern with part of the previous pattern as the constraint so that a test set with highly overlapping patterns can be obtained. As shown in [3], the test time is reduced by 30% on average with the modified FAN [12]. However, the high overlapping among patterns is obtained at the expense of a much larger test set and, as a result, even with the parity output, the reduction in test time is often inferior to that by compacting the test set for pure scan design.

The above observations suggest that, to obtain more consistent good results in test application time reduction, the postgeneration approach with a compact test set is preferable. Given a compact test set for the full scan circuit, the subsequent reduction procedures can then be proceeded based on these information embedded in the test set. One obvious advantage is that the given test set can be served as a bound of test time to ensure the subsequent procedures generate even superior results. Another advantage is that the test set provides a global information to guide the priority decision in the reduction procedures. Furthermore, prior to the test time reduction, the test generation flow is the same as that without special consideration on test time reduction. Namely, the decision of performing test time reduction can be made after a test set is obtained. Then test engineers can decide whether further reduction on test application time is necessary. This reduction-on-demand feature

is not possible in the reduction during test generation approach.

We are going to show in this paper that more significant reduction of test clocks in scan-designed circuits can be achieved by the post generation approach. Namely, the reduction is performed after an initial test set is generated. We identify the condition which allows the maximum overlapping between successive applied patterns and then actively modify these patterns to exploit the overlopping without sacrificing fault coverage. The experimental results on 22 ISCAS89 benchmark circuits show that, for the test sets generated by PODEM, 50% test clocks in pure full-scan can be reduced on average. Furthermore, for the parity scan design proposed in Reference 3, our method needs only 52% test clock cycles of those generated in Reference 3. When the given test set is already compact, the improvement over Reference 3 is even more significant, only 41% test clocks are required. The reduction is achieved strictly on the domain of combinational circuit.

2 Scan design

For a 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

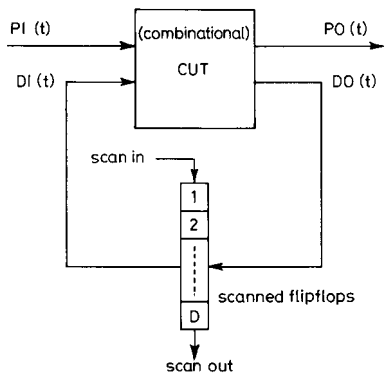

Fig. 1 *Scan-designed circuit*

other part for the FFs (flipflops) 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 resonse 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.

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 flipflops), then total testing time t_T is

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

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

$$t_T = ((|T| + 1) * D + |T|) * S \tag{2}$$

The total test clocks of T for a scan-designed circuit TTC is then

$$TTC = (|T| + 1) * D + |T| \tag{3}$$

It can be seen that TTC is dominated by $|T|$ and D. Therefore, to reduce the total test clocks, we can either compact the test set T as small as possible or shorten the number of shifts for each pattern. In Reference 3, the goal of test generation concentrates on how to increase the overlapping parts among patterns without giving due consideration to the test set size. As a result, the test clocks are not necessarily reduced because test set size $|T|$ is sacrificed for smaller effective D. Therefore, in our proposed postgeneration method, test clock reduction will start from a given set for CUT, preferably a compact one, then further test clock reduction on effective D can be carried out. There already exist effective test set compaction tools such as COMPACTEST [9] and TSR [6]. At present, TSR has been adopted in our work. Therefore, in this paper, we will aim at how to reduce the number of test clocks on D.

3 Maximum overlapping

Given a test set, the primary step of test clock reduction is to identify the maximum overlapping possible between two consecutive patterns without sacrificing fault coverage. In this section, the condition for maximum overlapping will be discussed.

Since the purpose of reduction is to use the current content of FFs as part of the next scan-in pattern, two types of overlapping are possible, either overlapping with the response of the previous pattern or the previous pattern itself. Both need some elaboration. In the first case, one must be sure that the content of FFs is not contaminated by the fault effect of all possible faults under consideration. And, in the second case, the fault effect should be observed directly from primary outputs and the FFs retain the previous pattern by skipping the load response operation.

Two special cases for pattern overlapping have been proposed in Reference 3. Under the constraint that all the activated faults of the current applied pattern t_i can be observed at POs, the following two cases are accepted for reducing the test clocks of the next pattern t_{i+1}.

Case a: Complete DO-reuse: $DO(t_i) \equiv DI(t_{i+1})$

Case b: Complete DI-reuse: $DI(t_i) \equiv DI(t_{i+1})$

The '$\equiv$' above is the compatible relation. Two vectors are said to be compatible if all the corresponding bits are either of 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 $v1 \equiv v2$. When one of the above two cases is satisfied, the current content of FFs can be completely reused as part of t_{i+1} and the scan-in operation of t_{i+1} can be eliminated. However, the conditions used in Reference 3 are clearly restrictive. The general conditions for maximum overlapping will be described next.

To allow maximum overlapping, we will first examine the conditions by which the content of FFs can be safely reused. As discussed in the beginning of this Section, to reuse the DO-part, one must be sure that no fault effect will appear in FFs. In other words, the content of FFs must be the fault-free response to allow safely reuse. This is by no means trivial since the fault effect of some faults can only be observed after scanning out the content of FFs. Nevertheless, we generally need not scan out the whole chain to determine whether or not it is the fault-free response. This is stated in the following observation.

3.1 Observation 1 (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, 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.

The implication of Observation 1 is that, after examining a few possible fault effect bits in $DO(t)$, the remaining fault-free response may provide an overlapping chance with the next pattern. From the observation, the number of bits to be shifted out after applying a pattern is then MAX(MINBIT(f)) for each detectable fault f by the pattern. This number can be easily determined by the fault simulation before actual test application. After shifting out this predetermined number of bits, it can be decided whether or not the remaining bits are fault-free and can be safely reused. For the example shown in Table 1, suppose three faults $\{f1, f2, f3\}$ are detected by applying pattern t and the fault-free $DO(t) = (1001010)$, by means of possible fault-effect bits computation in $DO(t)$, the faulty bit nearest to the scan-out pin is selected as MINBIT(f). Since the maximum of MINBIT is found to be 4, $DO(t)$ must be shifted out 4 bits before reusage.

Table 1: Example of MAX(MINBIT(f)) computation

	$DO(t)$ 7654321	MINBIT(f)
ff	1001010	—
$f1$	d001d10	3
$f2$	1d0d010	4
$f3$	1d0d0d0	2

ff: fault free response
d: faulty bit

The Di-reuse condition is the same as described earlier and formally given as follows.

3.2 Observation 2 (DI-reuse)

The current pattern in the FFs, $DI(t_i)$, may be part of $DI(t_{i+1})$ only if all the detected faults of t_i can be observed at POs alone.

It can be seen that when fault effects must be observed from FFs, the DI-reuse can not be applied. The problem can be significantly alleviated if extra hardware is used to enhance observability such as that in parity scan designed circuits [3].

The above two observations are the necessary conditions to reuse the current content of FFs. The following observation provides a sufficient condition for overlapping the next pattern with the current content of FFs, so that the test clocks can be reduced.

3.3 Observation 3 (maximum overlapping)

The current content of FFs, $SFF(t_i)$($DI(t_i)$ or $DO(t_i)$, may constitute part of $DI(t_{i+1})$ without sacrificing fault coverage if the conditions of Observations 1 or 2 are satisfied and TAIL($SFF(t_i) \equiv$ HEAD($DI(t_{i+1})$), where TAIL and HEAD are the trailing and leading parts during scan, respectively.

The last condition of Observation 3 requires only that the trailing part of $SFF(t_i)$ is compatible with the leading part of $DI(t_{i+1})$ without explicitly specifying the size of parts. The condition is named sliding compatibility of $SFF(t_i)$ and $DI(t_{i+1})$. Sliding compatibility is the generalisation of complete compatibility for test clock reduction.

The extent of overlapping which can be actually achieved from the above observation depends on the ordering of scan chain. Although it is theoretically possible to rearrange scan path to obtain maximum reduction, physical constraints discourage such arbitrary rearrangement. Hence the aspect of rearrangement is not discussed and, in our experiment, the natural ordering of scan path is used.

Based on Observation 3, the process for maximum overlapping can be performed as follow. In the previous example in Table 1, after shifting four bits out of $DO(t) = (1001010)$, there are still three bits can be overlapped with the next pattern. In other words, three scan clocks can be reduced if the next pattern is compatible with ($xxxx$100). To take fullest advantage of this situation, the next pattern can be chosen accordingly. When this is not possible, the next choice can be the one that is sliding compatible with ($xxxx$100) (i.e. compatible with ($xxxxx$10) and then with ($xxxxxx$1)). Note that, with sliding compatibility, the extent of test clock reduction decreases step by step. In the extreme case, not even sliding compatibility is possible, and regular scan operation must be performed. A more effective implementation of this process will be described in the next Section.

4 Active sliding compatibility

The goal of maximum overlapping is to optimally reuse the current content of scanned FFs in the next scan-in operation. However, for a test set lack of sliding compatibility between patterns, the reduction on test clock will not be significant. It means that the effectiveness of maximum overlapping could be restricted by the nature of the given test set. Therefore, to further enhance the reuse in maximum overlapping, we propose the process active sliding compatibility, which actively modifies the test set for sliding compatibility and increases the chance of reuse while keeping the overall fault coverage intact.

The basic idea of active sliding compatibility is to take advantage of the overspecification property of the given test set. Given a fault set F and a set of patterns T, the detectable faults of T in F is denoted as $DET(T, F)$. The faults considered in this paper are single stuck-at faults. For the purpose of detecting all the faults in $DET(T, F)$, it is usually not necessary to preserve all bits of each pattern in T specified. For example, for a pattern $t = (01111)$ in T, $DET(T, F)$ may still keep intact when t is changed to $t' = (0x1x1)$. Note that the difference between t and t' is only that some specified bits ('0' or '1') in t are modified to don't care ('x'). In this case, t is said to be overspecified in T and these bits changed from '0' or '1' to 'x' are said to be raised. Comparing t with t', t' obviously has more chances to be compatible with any arbitrary vector than t. Thus, when trying to reuse the

current contents of FFs for the scan-in of t, modifying t to t' would be a better choice.

In active sliding compatibility, for the next pattern to be scanned in, those sliding incompatible bits with the current contents of FFs will be modified (raised) to 'x' for re-usage. However, during raising these incompatible bits, it is essential to preserve the overall fault coverage $DET(T, F)$. The following observation allows us to perform the raising operation while keeping the fault coverage intact [6]. Define $ESS_T(t, F)$, the essential faults of t, as the set of faults in F that can only be detected by t but not others in T.

4.1 Observation 4

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 $DET(\{t'\}, F) \supseteq ESS_T(t, F)$, $T' = T - \{t\} + \{t'\}$ has at least the same amount of fault coverage as T.

The correctness of Observation 4 can be demonstrated as follows. Note that

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

where '$-$' and '$+$' are the set difference and set union, respectively. Then

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

if $DET(\{t'\}, F) \supseteq ESS_T(t, F)$. Thus, $T' = T - \{t\} + \{t'\}$ has at least the same amount of fault coverage as T.

The implication of Observation 4 is that active sliding compatibility can be performed by rasing some bits of t while monitoring the detectability of $ESS_T(t, F)$. And, the detectability of $ESS_T(t, F)$ can easily be verified by fault simulation.

With Observation 4, the brief procedure for maximum overlapping described at the end of Section 3 can be refined to the active sliding compatibility process for achieving more reductions on test clocks. The process is illustrated by the following example. As shown in Table 2, for a test set $T = \{t_1, t_2, t_3\}$, after applying t_1, active sliding compatibility is performed to select the next applied pattern. Only DI-reuse is considered in this example for simplicity. Initially, $DI(t_1)$, $DI(t_2)$ and $DI(t_3)$ are mutually incompatible. In Step 1, after the raising operation on these incompatible bits, the DI parts of the three patterns are still incompatible. Therefore, one shift out operation is performed on $DI(t_1)$. In Step 2, after the $DI(t_1)$ is shifted out 1 bit, active operations for compatibility is performed on $DI(t_2)$ and $DI(t_3)$ again. Unfortuntely, it also fails. In Step 3, after the content of $DI(t_1)$ is shifted out one more bit, the DI parts of t_1 and t_3 become compatible after raising. Thus, 4 scan clocks can be saved for the scan-in of t_3. Without these active raising operations, only one scan clock (choosing t_2 as the next pattern) can be reduced in this example.

Table 2: Example of active sliding compatibility

Step	DI	Contents 654321	Operation	$\equiv DI(t_1)$?
Initial	$DI(t_1)$	101000	—	—
	$DI(t_2)$	010011	—	no
	$DI(t_3)$	111110	—	no
1	$DI(t_1)$	101000	—	—
	$DI(t_2)$	x1x010	bit raising	no
	$DI(t_3)$	1x1110	bit raising	no
2	$DI(t_1)$	x10100	shift out 1 bits	—
	$DI(t_2)$	010xx1	bit raising	no
	$DI(t_3)$	11x110	bit raising	no
3	$DI(t_1)$	xx1010	shift out 2 bits	—
	$DI(t_2)$	01x011	bit raising	no
	$DI(t_3)$	111x10	bit raising	yes

The detail of the complete algorithm based on the proposed active techniques for maximum overlapping is shown in Fig. 2. The algorithm consists of the main routine Active_Clock_Reduction and the subroutine maximum_overlapping(). In Active_Clock_Reduction(), first, the test pattern t with maximum fault coverage is chosen as the first applied pattern.

Then, to select the next pattern of the current applied pattern t, the number of preshifts of t is obtained by computing the maximum fault-effect bit, MAX_MINBIT(t). Then, the maximum saved clocks by DO-reuse and the corresponding next pattern are obtained by calling maximum_overlapping(). If the number of preshift is zero, the reuse of $DI(t)$ becomes possible, and the saved clocks are also computed to compare with that of DO-reuse. The one with more saved clocks by DI-reuse and DO-reuse is then chosen as the test scheme of t. The process is continued until the given test set T is empty. In the subroutine, maximum_overlapping(), the successive pattern t' of the current applied pattern t is obtained from the remaining patterns of T with the most saved clocks. The searching process is performed in a greedy way. To find the next pattern t', 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 by raising operations. If found, maximum_overlapping() terminates. Otherwise, the reuse_vector is shifted out one more bit and the above make-compatible process is repeated. The process continues until all bits of reuse_vector have been shifted out. If no clock reduction is possible, the one with the largest fault coverage among the remaining patterns in T will be chosen as the next pattern.

5 Experimental results

To show the effectiveness of our method (ACT), these test clock reduction techniques proposed in this paper have been implemented on SUN4-SPARC2 workstation and 22 ISCAS'89 benchmark circuits are evaluated. In Table 3, the statistics of these circuits are shown, including the number of primary inputs (PI), primary output (PO), scanned flipflops (SFF), gates (GATE) and the sequential fault coverage [11] (SEQ-FC%). In addition, two initial test sets of these circuits are evaluated for both full scan and parity scan version. One initial test set is generated by a simple PODEM-like ATPG and the other is its highly compacted set by TSR [6]. These two test sets can be regarded as two extremes for an arbitrarily given test set. The sizes of these four different test sets, two for full scan and two for parity-scan [3], are also included in Table 3. For the test sets in PDM-P and TSR-P columns, they are generated in a similar way to PDM and TSR test sets except a parity checking path for pseudo POs (i.e. scanned FFs) and an extra PARITY output are added during test generation. In Tables 4 and 5, the test clock reduction results of ACT on the PDM, TSR, PDM-P, TSR-P test sets are shown. The comparison of ACT with previous works are provided in Table 6.

In Table 4, the test clocks generated by ACT for PODEM and TSR test sets are given. The final results are shown in the ACT column and those only with sliding but without bit-raising operations are listed in the SLIDE column. The number after each datum is the normalised ratio with respect to pure full scan (FSCAN) of the given test set. For the PODEM test sets, by the

sliding operation (SLIDE), 29% test clock reduction can be achieved on average as shown in the last row. It can be seen that SLIDE is more effective for those circuits with less scanned flipflops such as s820, s832, s1488 and s1494. For those circuits with a large amount of scanned FFs such as s1494 and s5378, the reduction is less successful owing to the difficulty of overlapping between successive scanned patterns. However, when bit-raising is added to increase the chance of overlapping, as shown in the ACT column, up to 46% and 65% reduction on test clocks for s1423 and s5378 can be obtained, respectively. This shows the effectiveness of the proposed active sliding compatibility process. On average, by ACT, 50% of test clocks for FSCAN can be reduced. For the TSR test sets, although the test sets are much more compact than those of PODEM and hence there is less chance for overlapping 27% reduction can still be achieved by our methods. When comparing the total test clocks of FSCAN for TSR test sets with those for PODEM test sets in the $\sum$/AVG row, less than 1/3 (38268/118036) of the test clocks are required, a very significant reduction on test clocks and test cost. It demonstrates that using a compact test set as the starting point of test clock reduction is worthwhile, and the efforts on the additional test set compaction can be well justified.

```
/* 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 */
Active_Clock_Reduction() {
    choose a pattern t with maximum fault coverage from T;
    while (T is not empty) {
        T = T-{t} ;
        pre_shift_no = MAX_MINBIT(t) ;
        F = F-DET({t},F);
        /* Compute the saved clocks by DO-reuse */
        DO_save_clk = maximum_overlapping(T,DO(t),pre_shift_no) ;
        DO_next_t = next_t ;
        load_response = TRUE ; use_DO = TRUE ;
        /* Try DI-reuse */
        if (pre_shift_no == 0) { /* DET(t,F) can be observed at PO */
            /* Compute the saved clocks by DI-reuse */
            DI_save_clk = maximum_overlapping(T,DI(t),pre_shift_no) ;
            DI_next_t = next_t ;
            if (DI_save_clk > DO_save_clk) {
                load_response = FALSE ;
                push_test_scheme(t,pre_shift_no,load_response) ;
                use_DO = FALSE ;
                t = DI_next_t ;
            }
        }
        if (use_DO) {
            push_test_scheme(t,pre_shift_no,load_response) ;
            t = DO_next_t ;
        }
    }
}
maximum_overlapping(T,reuse_vector,pre_shift_no) {
    shift_no = pre_shift_no ;
    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') == TRUE ) {
                next_t = t' ;
                return(no_of_DFF-shift_no) ;
            }
        }
        shift_no++ ;
    }
    choose a pattern t with the maximum DET from T as next_t ;
    return(0) ;
}
```

Fig. 2 *Algorithm of active sliding compatibility*

For the parity scan designed circuits, the results of ACT on PODEM-P and TSR-P test sets are shown in Table 5. In comparison with Table 4, with the extra parity output, the average reduction ratio can be increased from 50% to 72% for PODEM test sets and from 27% to 53% for TSR test sets. This shows the positive effect of enhanced observability by the parity output. Note that, for these circuits with more scanned FFs such as s838, s1423 and s5378, the results of ACT on TSR and

Table 3: Statistics of ISCAS89 benchmark circuits

Ckts	PI	PO	SFF	GATE	SEQ-FC%	PDM	TSR	PDM-P	TSR-P
s208	11	2	8	96	63.72	50	27	50	27
s298	3	6	14	119	85.71	51	25	51	24
s344	9	11	15	160	96.20	33	14	33	15
s349	9	11	15	161	95.71	33	14	33	15
s382	3	6	21	158	90.98	50	25	49	25
s386	7	7	6	159	81.77	97	63	92	63
s420	19	2	16	196	41.63	88	44	88	44
s444	3	6	21	181	89.45	47	25	47	25
s510	19	7	6	211	0.00	79	57	76	56
s526	3	6	21	193	75.32	98	50	97	50
s641	35	24	19	379	86.30	88	24	98	24
s713	35	23	19	393	80.90	88	24	85	24
s820	18	19	5	289	81.88	188	96	183	96
s832	18	19	5	287	81.38	180	97	182	96
s838	35	2	32	390	29.64	155	76	152	76
s953	16	23	6	395	7.78	109	76	115	77
s1196	14	14	17	529	99.76	222	124	232	126
s1238	14	14	17	508	94.69	230	133	234	129
s1423	17	5	74	657	67.39	126	32	118	32
s1488	8	19	6	653	92.60	176	104	183	106
s1494	8	19	6	647	92.10	185	102	175	103
s5378	35	49	179	2779	74.02	425	109	415	111

SFF = number of scanned flipflops.
PDM = test size generated by PODEM-like ATPG.
TSR = test size of compacting PDM by TSR [6].
PDM-P = test size of PDM with PARITY output.
TSR-P = test size of TSR with PARITY output.
* number of total flipflops.

TSR-P test sets are quite close ((1163, 1013), (2285, 2086), (14252, 14351)). The reason is as follows. Due to a large number of FFs and compacted test sets, the fault effect of each fault in these circuits are more likely to reach many FFs and scan operations for each pattern generally can not be avoided. However, from Observation 1, we are able to observe the existence of these faults through a few shift-out operations almost as effective as adding an extra parity output. It suggests that, in those circuits with large number of flipflops, it is more advantageous to use the proposed ACT starting from the compacted test set, since it can accomplish almost the same amount of test clock reduction as the additional parity chain, albeit without any hardware overhead.

The comparison of ACT with the previous work, parity scan [3], is shown in Table 6. The test clocks by [3] are listed in the PTY-SCAN column. The results of ACT with the parity output are shown in the ACT-PDM(P) and ACT-TSR(P) column, respectively, which are identical to Table 5. The number after each datum is the normalised ratio with respect to PTY-SCAN. Comparing ACT-PDM(P) and PTY-SCAN, except s1196 and s1238, our results are far superior to those of [3]. On average, the test clocks of ACT-PDM(P) are only 52% of those of PTY-SCAN. It shows the effectiveness of the proposed active test clock reduction. In particular, for s1423 and s5378, 61% and 75% reduction can be achieved, even though the test set size used in [3] are

Table 4: Results of test clock reduction by ACT

Ckts	PODEM test sets			TSR test sets		
	FSCAN	SLIDE	ACT	FSCAN	SLIDE	ACT
s208	458	301/0.66	220/0.48	251	198/0.79	182/0.73
s298	779	654/0.84	478/0.61	389	354/0.91	317/0.81
s344	543	471/0.87	360/0.66	239	225/0.94	224/0.94
s349	543	471/0.87	360/0.66	239	229/0.96	227/0.95
s382	1121	1001/0.89	728/0.65	571	535/0.94	504/0.88
s386	685	418/0.61	405/0.59	447	341/0.76	341/0.76
s420	1512	1110/0.73	582/0.38	764	638/0.84	431/0.56
s444	1055	937/0.89	638/0.60	571	545/0.95	448/0.78
s510	559	333/0.60	320/0.57	405	273/0.67	276/0.68
s526	2177	1905/0.88	1293/0.59	1121	1039/0.93	947/0.84
s641	1779	1277/0.72	645/0.36	499	458/0.92	370/0.74
s713	1779	1293/0.73	624/0.35	499	475/0.95	367/0.74
s820	1133	680/0.60	633/0.56	581	474/0.82	477/0.82
s832	1085	657/0.61	623/0.57	587	479/0.82	493/0.84
s838	5147	3667/0.71	1872/0.36	2540	2101/0.83	1163/0.46
s953	769	446/0.58	428/0.56	538	373/0.69	377/0.70
s1196	4013	1610/0.40	726/0.18	2249	1045/0.46	576/0.26
s1238	4157	1552/0.37	745/0.18	2411	1161/0.48	676/0.28
s1423	9524	8850/0.93	5109/0.54	2474	2429/0.98	2285/0.92
s1488	1238	747/0.60	711/0.57	734	587/0.80	587/0.80
s1494	1301	770/0.59	743/0.57	720	591/0.82	591/0.82
s5378	76679	69792/0.91	26535/0.35	19799	19618/0.99	14252/0.73
$\sum$/AVG	118036/1.00	98942/0.71	44778/0.50	38628/1.00	34168/0.83	26111/0.73

FSCAN = test clocks of pure full scan.
SLIDE = ACT without bit-raising.
ACT = active sliding compatibility technique.

Table 5: Results of test clock reduction by ACT with PARITY output

Ckts	PODEM-P test sets			TSR-P test sets		
	FSCAN	SLIDE	ACT	FSCAN	SLIDE	ACT
s208	458	296/0.65	222/0.48	251	198/0.79	183/0.73
s298	779	534/0.69	241/0.31	374	282/0.75	171/0.46
s344	543	409/0.75	247/0.45	255	213/0.84	191/0.75
s349	543	409/0.75	247/0.45	255	213/0.84	191/0.75
s382	1099	865/0.79	465/0.42	571	485/0.85	372/0.65
s386	650	198/0.30	180/0.28	447	152/0.34	153/0.34
s420	1512	982/0.65	415/0.27	764	601/0.79	366/0.48
s444	1055	872/0.83	381/0.36	571	495/0.87	344/0.60
s510	538	161/0.30	161/0.30	398	156/0.39	155/0.39
s526	2155	1757/0.82	556/0.26	1121	967/0.86	538/0.48
s641	1979	1168/0.59	329/0.17	499	406/0.81	265/0.53
s713	1719	1042/0.61	306/0.18	499	428/0.86	289/0.58
s820	1103	249/0.23	252/0.23	581	169/0.29	167/0.29
s832	1097	264/0.24	263/0.24	581	174/0.30	183/0.31
s838	5048	2918/0.58	1248/0.25	2540	1896/0.75	1013/0.40
s953	811	236/0.29	211/0.26	545	165/0.30	165/0.30
s1196	4193	1180/0.28	404/0.10	2285	575/0.25	278/0.12
s1238	4229	1170/0.28	408/0.10	2339	617/0.26	285/0.12
s1423	8924	7919/0.89	3041/0.34	2474	2324/0.94	2086/0.84
s1488	1287	313/0.24	296/0.23	748	221/0.30	221/0.30
s1494	1231	293/0.24	279/0.23	727	207/0.28	207/0.28
s5378	74879	60835/0.81	19563/0.26	20159	19363/0.96	14351/0.71
$\sum$/AVG	115832/1.00	84070/0.54	29715/0.28	38984/1.00	30307/0.62	22174/0.47

FSCAN = test clocks of pure Full Scan.
SLIDE = ACT without bit-raising.
ACT = active sliding compatibility technique.

Table 6: Comparison of ACT with PARITY-SCAN [3]

Ckts	PTY-SCAN [3]	ACT-PDM(P)	ACT-TSR(P)	Time, s
s208	279	222/0.80	183/0.66	2.0
s298	495	241/0.49	171/0.35	3.0
s344	461	247/0.54	191/0.41	3.0
s349	443	247/0.56	191/0.43	3.0
s382	828	465/0.56	372/0.45	4.0
s386	380	180/0.47	153/0.40	5.0
s420	1296	415/0.32	366/0.28	7.0
s444	936	381/0.41	344/0.37	5.0
s510	354	161/0.45	155/0.44	6.0
s526	1707	556/0.33	538/0.32	10.0
s641	579	329/0.57	265/0.46	12.8
s713	729	306/0.42	289/0.40	12.1
s820	487	252/0.52	167/0.34	20.1
s832	471	263/0.56	183/0.39	20.7
s838	5602	1248/0.22	1013/0.18	30.2
s953*	2391	211/0.09	165/0.07	16.3
s1196	359	404/1.13	278/0.77	40.1
s1238	351	408/1.16	285/0.81	44.7
s1423	7894	3041/0.39	2086/0.26	60.2
s1488	617	296/0.48	221/0.36	36.8
s1494	702	279/0.40	207/0.29	35.0
s5378	76739	19563/0.25	14351/0.19	1492.0
$\sum$/AVG	101709/1.0	29504/0.52	22009/0.41	—/—

ACT-PDM(P) = results of ACT on PDM-P test sets.
ACT-TSR(P) = results of ACT on TSR-P test sets.
TIME = CPU time of ACT-PDM(P) on SUN4-SPARC2.
* = removed from $\sum$/AVG due to different number of scanned FFs.

approximately equal to that of the PODEM-P test set shown in Table 3. The result demonstrates the merit of the postgeneration approach of test clock reduction. It is also interesting to compare PTY-SCAN with ACT-TSR(P). In each evaluated case, the result of ACT-TSR(P) is far better than that of PTY-SCAN. The average ratio of test clocks is only 41%. Recall that, in Reference 3, the test sets are generated by preserving as many common parts as possible among patterns to increase the chance of overlapping. The compactness of the generated test set is scarified. This comparison result shows that scarifying the compactness of the test set for overlapping is not worthwhile, and a compact test set is desired for test clock reduction in scanned design.

In the last column of Table 6., the CPU-time for ACT-PDM(P) is shown. Except s5378, all the examples can be completed in one minute. For s5378, owing to its circuit size, large test set and scanned FFs, more CPU-time, albeit not prohibitively long, is required. For ACT-TSR(P), because the test sets are much more compact than those of ACT-PDM(P), the CPU-time is far less than that shown in the Table. For example, in ACT-TSR(P), the CPU-time for s5378 is only 448 seconds.

6 Conclusions

Test time reduction for scan designed circuits has been investigated in this paper. To reduce the lengthy shifting operations in the long scan path, post generation method to optimally reuse the contents of the scanned flipflops have been developed. Specifically, the condition for maximum overlapping between successive applied patterns has been identified. This sliding compatibility condition is the generalisation of complete compatibility and allows more reduction of scan clocks when scanning in the next pattern. To further enhance the effectiveness of maximum overlapping, active operations have also proposed to increase the sliding compatibility between successive patterns and to maximise the reuse of the FFs' contents. From the experimental results on 22 ISCAS89 benchmark circuits, up to 50% test clocks can be reduced on average by the proposed method. Furthermore, when the parity output is included in the scan designed CUT, the resultant test clocks by ACT have been only 41% of those in Reference 3.

It has been reported that by switching between scan mode and nonscan mode, further reduction on test clock cycles are possible in pure scan designed circuits. However, a sequential circuit test generation will then be required, and the test generation time will be substantially increased. To retain the advantage of simple combinational test generation of scan designed circuits, the mode-switching is not performed in this work.

7 References

1 ABADIR, M.S., and BREUER, M.A.: 'Scan path with look ahead shifting (SPLASH)'. Proceedings of 1986 international test conference, pp. 696–704

2 ABADIR, M.S.: 'Efficient scan path testing using sliding parity response compaction'. Proceedings of 1987 international test conference, pp. 332–335

3 FUJIWARA, H., and YAMAMOTO, A.: 'Parity-scan design to reduce the cost of test application', *IEEE Trans.*, 1993, **CAD-12**, (10), pp. 1604–1611

4 LEE, S.Y., and SALUJA, K.K.: 'An algorithm to reduce test application time in full scan designs'. Proceedings of 1992 ICCAD, pp. 17–20

5 POMERANZ, I., and REDDY, S.M.: 'A test application scheme for embedded full-scan circuits to reduce testing costs'. Proceedings of first Asian test symposium, 1992, pp. 206–221

6 CHANG, J.S., and LIN, C.S.: 'Test set compaction for combinational circuits'. Proceedings of first Asian test symposium, 1992, pp. 20–25

7 LAI, W.C., KUNG, C.P., and LIN, C.S.: 'Test time reduction in scan design circuits'. Proceedings of 1993 EDAC, pp. 489–498

8 TROMP, G.-J.: 'Minimal test sets for conmbinational circuits'. Proceedings of 1991 international test conference, 1991, pp. 204–209

9 POMERANZ, I., REDDY, L.N., and REDDY, S.M.: 'COMPACTEST: a method to generate compact test sets for combinational circuits'. Proceedings of 1991 international test conference, pp. 194–203

10 GOEL, P.: 'An implicit enumeration algorithm to generate tests for combinational logic circuits', *IEEE Trans.*, 1981, **C-30**, (3), pp. 215–222

11 CHENG, W.T., and DAVIDSON, S.: 'Sequential circuit test generator (STG) benchmark results'. Proceedings of international symp. of circuits and systems, 1989, pp. 1931–1941

12 FUJIWARA, H., and SHIMONO, T.: 'On the acceleration of test generation algorithms', *IEEE Trans.*, 1983, **C-32**, (12), pp. 1137–1144

13 EINCHELBERGER, E.B., and WILLIAMS, T.W.: 'A logic design structure for LSI testability'. Proceedings of design automation conference, 1977, pp. 462–468