Proc 2nd Testing Conf. C[illegible]

1983

TESTING COMPUTER HARDWARE*
THROUGH DATA COMPRESSION IN SPACE AND TIME

Kewal K. Saluja,
Department of Electrical and
Computer Engineering,
University of Newcastle,
New South Wales, 2308
AUSTRALIA.

M. Karpovsky[†]
Computer Science Department,
State University of New York,
Binghamton,
New York, 13901
U.S.A.

Abstract

In this paper we introduce a new type of data compressor, called space compressor (SC). Such an SC can be used either to reduce the number of pins which need to be monitored by a tester for a given digital device or alternatively it can be used to reduce the width of response vectors stored in a tester. A detailed analysis and design of such SCs is discussed. Analogous results for linear feedback shift registers (LFSR) are discussed briefly.

Introduction

Perennial problem of testing logic circuits has been further complicated with the introduction of VLSI. As a result, considerable effort is being devoted to efficient test generation methods and design for testability. However, researchers have also started to realize, no matter how efficient and small the test sets are, test sets often require a very large data space. Thus considerable effort is being devoted to data compression techniques for testing digital devices.

All compression techniques which have been studies in literature for testing digital circuits, essentially follow the structure of Figure 1. A Circuit Under Test (CUT) receives its inputs for a Test Generator (TG) and output streams from CUT are fed to a Response Compressor (RC) which converts the output streams into signature(s) of the CUT. Typical compressors which have been studied in literature are Transition count [1] Syndrome (1's count)[2], Linear Feedback Shift Register (LFSR) [3], Multiple Input Linear Feedback Shift Register (MISR) [4], Spectral methods [5,6], etc. [7]. In all these schemes, at every instant a new input (test vector) is applied, the output of CUT is observed by RC and based on its own state and the value obtained from CUT, the RC transitions into a new state. This process continues in time till all the required inputs have been applied to the CUT. The state of the compressor, known as signature, is then observed and compared with expected (fault-free) signature to determine if the CUT is fault-free or faulty.

Clearly, this method achieves a substantial data compression. We call this method of data compression as data compression in time. In the following sections we give a new formulation of the data compression problem. We shall propose and study a new type of data compression called space compression.

Problem Formulation

Let us consider a CUT with n inputs and m outputs. Let us further consider the observation of m outputs in the presence of a fault and on application of a test input. It will be quite reasonable to assume that not all m outputs will be erroneous for a given test input. In a simple case if we assume that all m outputs of CUT are realized independantly (i.e. no two outputs of CUT share inputs or logic within the CUT), then any single fault in the CUT will cause no more than one output to be erroneous at any instant. In general number of erroneous outputs for a given test input, at any instant, will depend on: (i) fanouts and shared logic, and (ii) number of faults. This clearly suggests the possibility of monitoring less than m outputs to detect fault(s) in a CUT. However, lack of apriory knowledge of fault sight also implies that all m outputs must be monitored. We can achieve these apparently conflicting goals by using additional logic circuit, called Space Compressor (SC), as shown in Figure 2. The SC of Figure 2 accepts m outputs of CUT as inputs and provides reduced number of outputs for monitoring while testing a CUT. For the most simple case, in which no more than one output is erroneous at any instant, SC is a simple parity checker. Such a parity checker can be implemented by a casecode or tree of Exclusive-ORs(EOR). Thus, under test mode, we would need to monitor only the output of such an SC. A reduction in the test data is thus evident. We now give a more general statement of the problem as follows:

Problem 1: For a CUT with m outputs, if no more than ℓ_e outputs can be erroneous for any test

[†] Professor Karpovsky is now a visiting professor at Boston University.

* This work is supported in part by the Division of Mathematical and Computer Science of NSF under Grant No. MCS 8021262, Computer Research Board and Radio Research Board Australia and IRAC of the University of Newcastle, Australia.

input at any instant due to fault(s) in CUT, can one design an SC with m inputs and r outputs such that we need only observe the r outputs of SC to detect fault(s) in CUT?

Clearly, a trivial answer to the above problem is when $r=m$ and SC a trivial circuit with m inputs and m outputs. Thus, our objective is to have $r \leq m$. Furthermore, as r is a function of m and ℓ_s we denote it by $r_s(m,\ell_s)$ and state the following problems.

Problem 1.1: Give a design of SC such that $r_s(m,\ell_s)$ is small.

Problem 1.2: Determine lower and upper bounds on $r_s(m,\ell_s)$.

Problem 1.3: Determine the minimal complexity of an SC with m inputs and $r_s(m,\ell_s)$ outputs.

An analogous formulation of the problem exists in the case of Time Compressors. For analogous formulation we consider a CUT with n inputs and single output. A stream of N test vectors are applied to such a circuit and depending on the fault and a test input at any instant, the output may or may not be faulty. The objective is to compress the N bit long output stream from CUT into an r bit long stream by a TC while retaining the fault detections properties of the test set. Clearly it is desirable to have $r \leq N$. We now give a statement of the problem and its associated problems below:

Problem 2: For a CUT with single output, if a fault(s) causes the output to be erroneous for no more than ℓ_t instants, when an input test sequence of length N is applied (one test vector is applied every instant), can one design a TC with a single input and r bit memory (r-state variables) such that we need only observe the state of TC at the end of the test experiment to detect fault(s) in the CUT.

Furthermore as r is a function of ℓ_t and N we denote it as $r_t(N,\ell_t)$. Thus we have:

Problem 2.1: Give a design of TC such that $r_t(N,\ell_t)$ is small.

Problem 2.2: Determine lower and upper bounds on $r_t(N,\ell_t)$.

Problem 2.3: Determine the minimal complexity of a TC with single input and $r_t(N,\ell_t)$ state variables.

Further sets of problems, analogous to the above two sets of problems can be stated in the case of Space-Time Compressors (STC) and Time-Space Compressors (TSC). These problems are studied in more detail elsewhere [8]. Problems 2.1 and 2.2 have also been studied by Smith [9] for a class of TCs which are realized by linear feedback shift registers (LFSR). In the remainder of this paper we shall concentrate on Problem 1. We will observe that our results are also applic-

Notation

In this section we develope notation to study Space Compression. A general notation to study SC, TC, STC and TSC is rather complex. We will study SCs in detail and rely on readers understanding of this material while stating results for other compressors. Furthermore, we shall assume inputs and outputs of CUT to be binary vectors. Let $T_i = (t_1^i, t_2^i, \ldots, t_n^i)$ denote an input test vector to the CUT and the corresponding output vectors of CUT and SC are denoted by $Y_i = (y_1^i, y_2^i, \ldots, y_m^i)$ $Z_i = (z_1^i, z_2^i, \ldots, z_r^i)$ respectively. In the presence of a fault(s) in CUT we denote the outputs of CUT and SC as $\hat{Y}_i$ and $\hat{Z}_i$ respectively. Clearly, the effect of a fault can only be observed as an error in the output vector Y_i. A fault is said to be active with respect to a test vector T_i if on application of T_i to the CUT we have $Y_i \neq \hat{Y}_i$, fault is said to be inactive otherwise. If we denote number of 1's in a vector Y by $\omega t(Y)$, then $\omega t(Y_i \oplus \hat{Y}_i)$, where $\oplus$ denotes EOR, is number of errors in the output of a CUT on application of a test input T_i in the presence of a fault(s) in the circuit. Clearly, $\omega t(\hat{Y}_i \oplus Y_i) > 0$ if and only if a fault is active with respect to a test T_i. We can extrapolate these notions while considering a test set $T = \{T_1, T_2, \ldots, T_N\}$ for a CUT. For any fault f in the CUT, let the true and faulty output sets corresponding to test set T be $\{Y_i\}$ and $\{\hat{Y}_i\}$. Let $E_i = Y_i \oplus \hat{Y}_i$, then error set for a fault f with respect to a test set T is $\{E_i/i=1, \ldots, N\}$.

Definition 1: Space multiplicity of error with respect to a test set T for a fault f is defined as $\max\{wt(E_i/i=1, \ldots, N\}$. We denote this by $\ell_s(f)$.

We can extend the above definition for a fault set $\{F = f_1, f_2, \ldots, f_p\}$ as follows.

Definition 2: Space multiplicity of error with respect to T for F is defined as $\ell_s = \max\{\ell_s(f_i)/f_i \in F\}$.

Computation of exact value of ℓ_s may be quite complex in general. However, often a bound on ℓ_s may be established by simple analysis of CUT. For example, we have argued in the previous section that if all outputs of a CUT are realized independantly, i.e. do not share logic or inputs than for any single fault in such a CUT $\ell_s \leq 1$ with respect to any test set T. More generally, if a CUT has only one fanout of α than $\ell_s \leq \alpha$ for any T. Simple methods based on intution can be developed to determine a bound on ℓ_s for a CUT, but such methods will not be discussed here.

We end this section with a few words about analogous notions for TC. In TC a sequence of test inputs is applied and a sequence of output stream is obtained. Time multiplicity of error, ℓ_t, can then be defined as the number of places faulty and fault-free sequences differ from each other. To a certain degree we can assert that ℓ_t is dual to ℓ_s by using the following arguments.

realizations may need lesser number of gates than others. However, while realizing linear SCs one must also consider their testability properties [11].

Thus, it appears natural to define the complexity of a linear SC in terms of number of two input EOR gates used in its realization. Following is a formal definition of complexity and Theorem 3 gives an upper bound on the complexity. For proof of Theorem 3 reader is referred to [8].

Definition 3: Complexity of a linear SC, denoted as $L_s(m,\ell_s)$ for a CUT with m outputs and ℓ_s space multiplicity of error, is the minimum number of two input EOR gates required to realize the linear SC.

Theorem 3

$$L_s(m,\ell_s) \lesssim \frac{m\, r_s(m,\ell_s)}{\log M}$$

For a linear SC

$$L_s(m,\ell_s) \le \left\lceil \frac{1}{2}[(m-r_s(m,\ell_s)r_s(m,\ell_s))+(m-1)] \right\rceil$$

Space Compressors - Application and Comments

From Table 1 one can work out the compression ratio of SCs for different values of ℓ_s and m. We demonstrate this by way of an example. Let us consider an SC corresponding to a [23,12,7] Golay code. Such an SC will have 23 inputs and 11 outputs. Thus a tester making use of such an SC would need to monitor only 11 outputs as opposed to 23. If test results are stored in memory, then for a CUT with 23 outputs and N test patterns, the storage space required for output stream will be 23N bits, whereas use of SC will reduce the storage space to 11N bits. In general the storage space is reduced from mN bits to $r_s(m,\ell_s)N$ bits. Obviously, such a reduction is obtained at a cost, being (i) SC introduces delay, thus effecting the speed of testing, and (ii) only those faults which have space multiplicity of ℓ_s or less or detected.

Another question, one may ask, as to where an SC must reside. There are two possibilities.

1. SC as a part of tester: This scheme is shown in Figure 5 and is self explanatory.

2. SC on a Chip: This scheme is shown in Figure 6. Part of the outputs from CUT can be multiplexed. An extra input is required for test and normal mode of operation. In test mode multiplexer is activated to provide the outputs of SC on the output lines. Thus only the outputs of multiplexers would need to be monitored. This process will not detect any faults on unmonitored output pins, but clearly faults on output pins can be detected with ease. For example in case of stuck type faults, such unmonitored outputs can be tested by at most $(k+1)$ tests, where k is the number of outputs.

In both the above cases, problem of testing SC remains to be solved. To this end we suggest the use of linear SCs, as such circuits are easy to test [11] often with a fixed test set of small size. The choice of realization may be dictated by the need of minimum complexity (maximum shared logic) or ease of testability (independant cascades of EORs).

In certain cases use of other types of SCs can offer distinct advantage as shown by the following example.

Example 1

Any single fault in a parallel adder, with arbitrary number of outputs, can be detected by an SC which realizes residue mode 3 operation. Clearly, such an SC has only two outputs. Thus, with a scheme like this test data storage space can be reduced considerably.

Above example only demonstrates that in some cases SCs based on non-binary codes may offer additional advantages.

Time Compression

The results of the previous sections also apply to TCs. For example we can restate Theorem 1 as follows [8].

Theorem 4

$$r_t(N,\ell_t) \ge \left\lceil \log_2 \sum_{i=0}^{\left\lfloor \frac{\ell_t}{2} \right\rfloor} \binom{N}{i} + \varepsilon \right\rceil$$

where

$$\varepsilon = 0 \quad \text{for } \ell_t \text{ even}$$

$$= \binom{N-1}{\left\lfloor \frac{\ell_t}{2} \right\rfloor} \quad \text{for } \ell_t \text{ odd.}$$

To state results analogous to Theorems 2 etc. we need to define irreducible and primitive polynomials which give rise to cyclic codes [10]. Once again for space limitation we will not state these results. However, following theorem is derived from the results in Table 1. This result is also stated in [8,9].

Theorem 5

For $\ell_t \le 2$ and $N = 2^i-1$, $r_t(N,\ell_t) = i$

The TC implied by Theorem 5 is a linear feedback shift register. Similar results can be stated for other LFSRs.

Time-Space and Space-Time Compressors

Having developed background and designs for SC a natural application which comes to mind is the use of LFSRs or a Multiple-input linear feedback shift register to time compress the outputs of SC. Detailed analysis of such scheme is given in [8].

Relation between error correction & (in my book [illegible]

study of such schemes, suggests that to retain e necessary fault detection properties, or more propriately error detection properties, we should ke use of compressors based on non-binary codes.

Conclusion

In this paper we have introduced a new com-ession technique called space compression. We ve related the space compression problem to a ll developed theory of error-correcting codes. is theory is then used as a tool to analyse and sign space compressors. We have also discussed plication of space compressors and their vantages and disadvantages.

The problem of time compression can be shown alogous to space compression problem and milar results can be derived using the theory error correcting codes.

References

] Hayes, J.P. "Transition Count Testing of Combinational Logic Circuits", IEEE Trans. Comp. Vol. C-25, June 1976, pp. 613-620.

] Savir, J. "Syndrome-Testable Design of Combinational Circuits", IEEE Trans. Computers, Vol. C-29, June 1980, pp. 442-451, and errata sheet to this paper in IEEE Trans. Comp. Vol. C-29, Nov. 1980, pp. 1012-1013. First version of the paper appears in Proc. 9th Int. Symp. on Fault Tolerant Computing, June 1979, pp. 137-140.

] Frohwerk, R.A. "Signature Analysis: A New Digital Field Service Method", Hewlett-Packard Journal, May 1977, pp. 2-8.

[4] Konemann, B., Mucha, J. and Zwiehoff, G., "Built-in Logic Block Observation Techniques", IEEE Int. Test Conference, Cherry Hill, New Jersey, 1979, pp. 37-41.

[5] Muzio, J.C. and Miller, J.M. "Spectral Techniques for Fault Detection", Proc. of 12th Int. Symposium on Fault Tolerant Computing, U.S.A., June 1982, pp. 297-302.

[6] Suskind, A.K. "Testing by Verifying Walsh Coefficients", Proc. of 11th Int. Symp. on Fault Tolerant Computing, U.S.A., June 1981, pp. 206-208.

[7] Saluja, K.K. and Su, S.Y.M. "VLSI Testing Through Data Compression: A Survey", Proc. IREECON International, Sydney, Australia, Sept. 5-9, 1983.

[8] Karpovsky, M.G. and Saluja, K.K. "Space and Time Data Compression for Testing of Computer Hardware and its Relation to Error Correcting Codes", paper under preparation.

[illegible] Karpovsky, Finite Orthogonal ser

[9] Smith, J.E. "Measures of Effectiveness of Fault Signature Analysis", IEEE Trans. on Computers, Vol. C-29, June 1980, pp. 510-514.

[10] MacWilliams, F.J. and Sloane, N.J.A., The Theory of Error Correcting Codes, North Holland, 1978.

[11] Saluja, K.K. and Reddy, S.M. "Fault Detecting Test Sets for Reed-Muller Canonic Networks", IEEE Trans. Computers, October 1975, pp. 995-998.

Table 1: Lower and Upper Bounds on $r_s(m,\ell_s)$

$r_s(m,\ell_s)$		Lower Bound	Upper Bound	Remark
ℓ_s	m			
1	m	1	1	SC is a simple Parity Checker
2	2^i-1	i	i	Corresponding to $[2^i-1, 2^i-1-i, 3]$ Hamming Code
2	m	$\lceil \log_2(m+1) \rceil$	$\lceil \log_2(m+1) \rceil$	Shortened Hamming Code
3	m	$1+\lceil \log_2 m \rceil$	$1+\lceil \log_2(m+1) \rceil$	Hamming Code with overall parity check
4	2^i-1	2i-1	2i	Double Error Correcting BCH Code
5	2^i-1	2i	2i+1	Double error correction BCH Code with overall parity check
6	23	11	11	Golay Code [23,12,7] perfect code
	. .	. .	. .	. .
m-1	m	m-1	m-1	Repetition code
m	m	m	m	

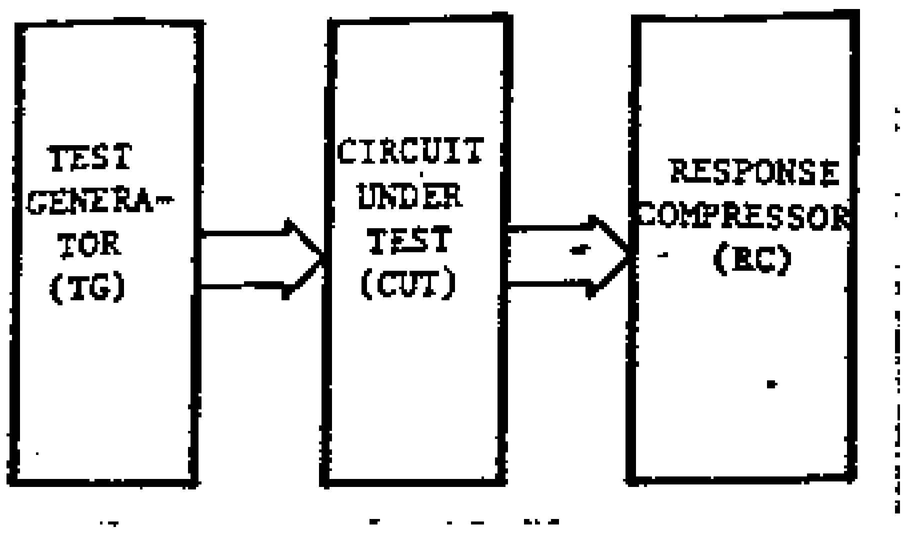


Figure 1: A Test Structure

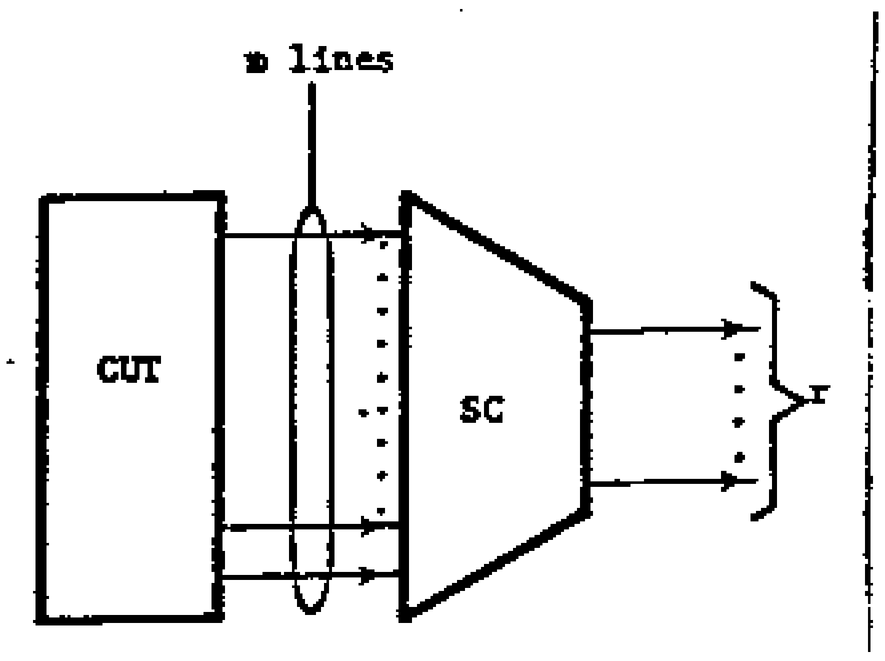
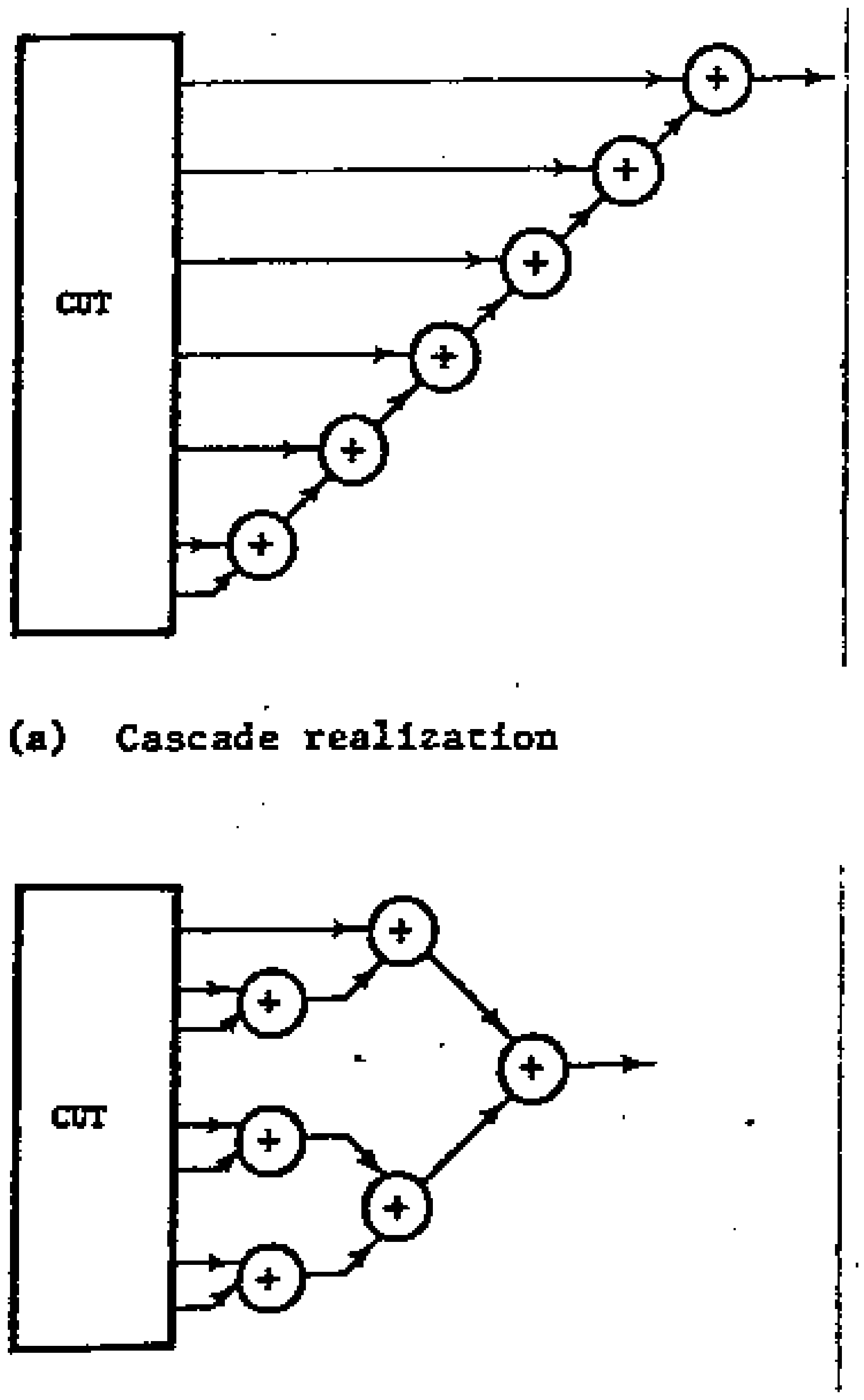


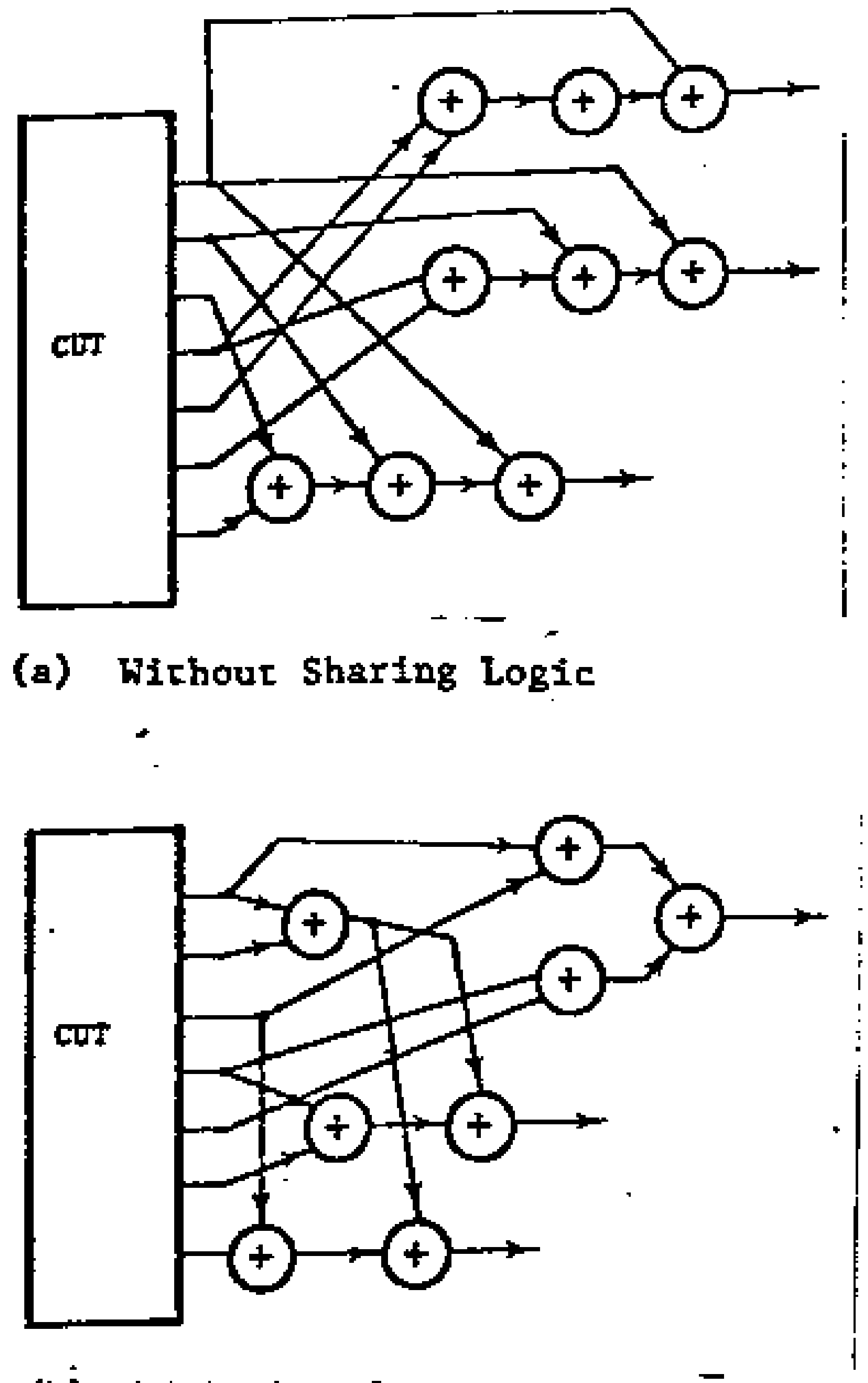
Figure 2: A Test Strucrure with Space Compressor



(a) Cascade realization

(b) Tree Realization

Figure 3: Two Realizations of SC for m=7, ℓ_s=1



(a) Without Sharing Logic

(b) With Shared Logic

Figure 4: Two Realizations of SC for m=7, ℓ_s=2

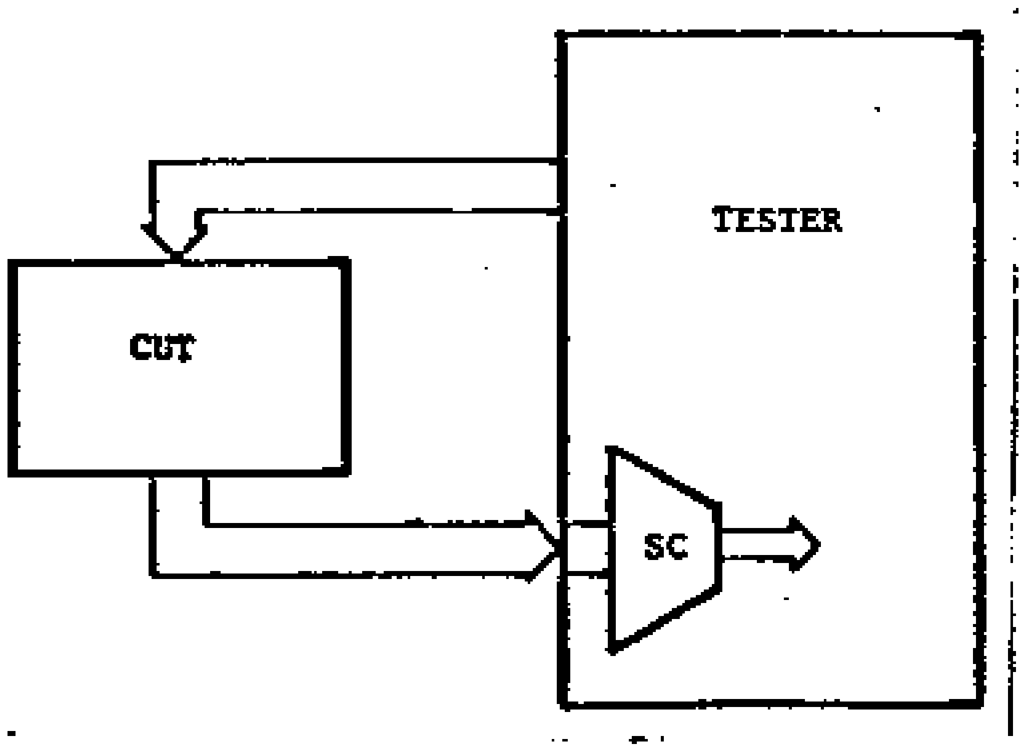


Figure 5: SC As Part of a Tester

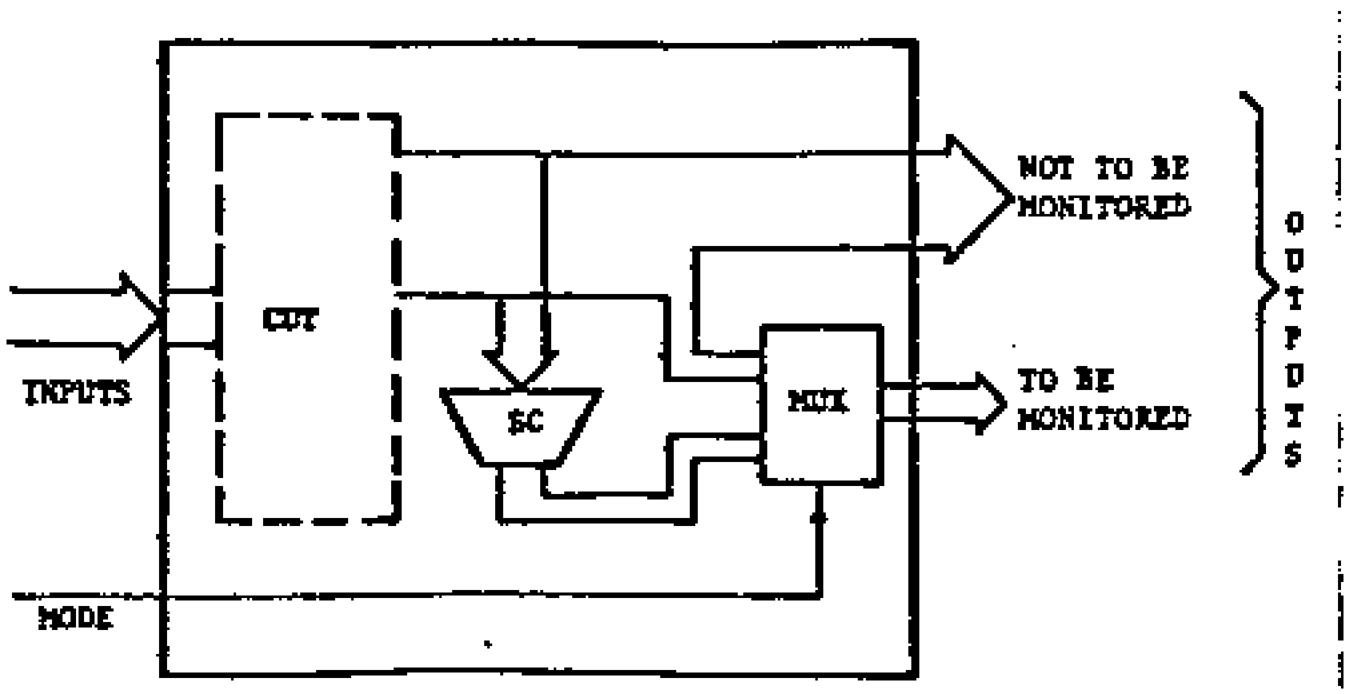


Figure 6: SC On A Chip