TESTABILITY ENHANCEMENT OF VLSI USING CIRCUIT STRUCTURES*

V. G. Oklobdzija* and M. D. Ercegovac

University of California, Los Angeles
Computer Science Department
Los Angeles, CA.90024

ABSTRACT

This paper presents an efficient circuit-level approach for enhancing the testability of VLSI structures. It is based on the concept of a controllable gate (C-gate) and its variations. By changing certain gates within a given logic network into C-gates, controllability and observability of the network are improved and overall testing complexity reduced. The C-gate approach can also be used for testability enhancement of sequential networks.

1. INTRODUCTION

To improve observability and controllability in digital logic, input and output test points are added [11]. However, this may not be convenient for the LSI/VLSI circuits due to the pin limitation and added wiring complexity. Other methods such as LSSD suffer from the number of shift-in/shift-out sequences which contribute substantially to the testing time and make the testing method inherently dependent on the external tester.

The technique proposed in [7] claims to eliminate test pattern generation and fault-modeling. In this technique, multiplexers are added to allow the chip to be partitioned into manageable partitions. These can be tested exhaustively so that fault-modeling can be eliminated. Even though a considerable saving in test generation is achieved, the price to pay is rather high. The added multiplexing circuitry introduces an additional delay in the signal propagation path and a considerable hardware overhead. In the example of [7],[9] the hardware overhead amounts to 30%, and additional delay is 8/6 times the delay of the unmodified circuit.

The solution proposed here is to provide the desired testability features by modifying the existing circuits. The "controllable gate" is introduced as a mechanism of forcing a propagation d-cube [1] onto the regular gate in the ***test mode*** of operation. This can be achieved by introducing an extra MOS device at a particular place in the combinational logic. The overhead in extra circuitry and the decrease in performance are found to be small.

2. CONTROLLABLE GATE

Concept:

A controllable gate (C-gate) is a gate with a ***control*** input C added to it in such a way that the gate output **y** is:

$$y = \begin{cases} f(x_{n-1},\ldots,x_0) & \text{if } C=K \\ g(x_p) & \text{if } C=K' \end{cases}$$

where **f** is the function of the gate in the *normal mode*, **g** is a function of the *priority input* x_p, $x_p \in \{x_{n-1},\ldots,x_0\}$ and K is a Boolean constant. Usually, **g** is a simple function such as the identity or complement. The priority and control inputs are marked by arrows.

The purpose of a C-gate is to transmit its priority input (direct or complemented) to the output thus making the gate "transparent". That is, the output becomes independent of all but one selected input.

The controllable NAND and NOR gates (C-NAND and C-NOR) are illustrated in Fig.1. The constant K=0 for C-NAND and K=1 for C-NOR. Note that in the test mode a propagation **d**-cube [1] is forced upon the gate.

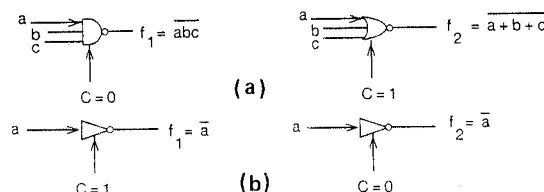


Fig.1: C-NAND and NOR: (a) normal operation (b) test mode

The concept of a controllable gate can be applied to other types of gates and different technologies. For brevity, we limit our discussion here to C-NAND and C-NOR gates implemented in n-MOS technology.

Implementation:

An n-MOS circuit implementation of controllable C-NAND and C-NOR gates is shown in Fig.2. Transistor Q_5 and one input have been added in order to obtain controllable gates.

In the following paragraphs we consider the effect of the added transistor on the propagation delay and the gate area.

Performance Degradation:

The added transistor Q_5 affects the propagation delay of the gate due to capacitances added to node 5. A SPICE [10] simulation has been performed using layouts of C-NAND and C-NOR gates implemented using a 2.5μ geometry shown in Fig.3.

*At present with IBM Research, P.O.Box 218, Yorktown Heights, NY 10598

*Supported by the Office of Naval Research Contract No. N00014-79-C-0866

Reprinted from IEEE INTERNATIONAL CONFERENCE ON CIRCUITS AND COMPUTERS ICCC '82, September-October 1982

CH1813-5/82/0000/0198$00.75 © 1982 IEEE

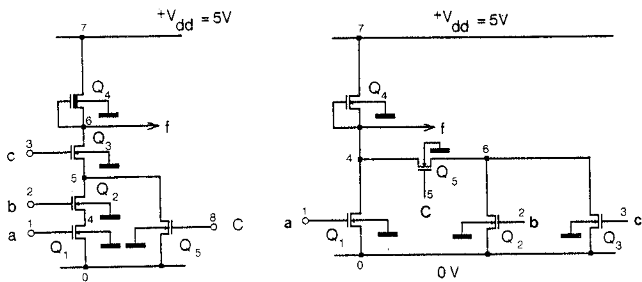


Fig.2: nMOS implementation of C-NAND and C-NOR gates

All parasitics have been taken into account. We define the propagation delay as $t_{\Delta} = \max\{t_{ON}, t_{OFF}\}$ where t_{ON} is the rising edge time required for the saturated gate output to reach V_{TOFF}, and t_{OFF} is the falling edge time required to reach the threshold V_{TON} from the nonconducting output. To determine the extra delay t_{δ} caused by Q_5, simulation is performed with Q_5 included to determine $t_{\Delta 1}$, and without Q_5 to obtain $t_{\Delta 2}$. As indicated in Fig.4, t_{δ} is about 10% for C-NAND. In a similar manner, it was determined that t_{δ} for C-NOR is about 20% of the NOR gate delay [8].

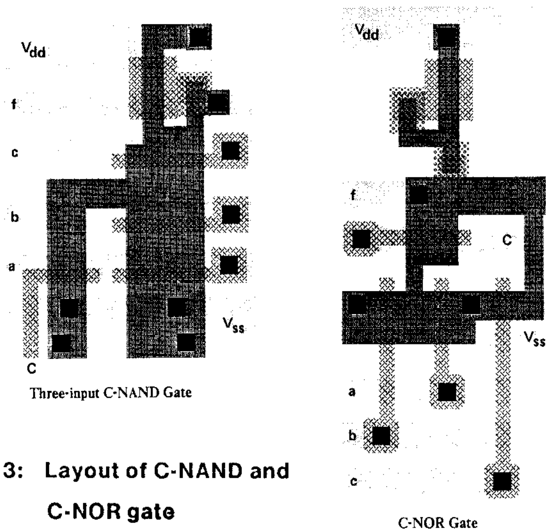


Fig.3: Layout of C-NAND and C-NOR gate

A conservative estimate of the area overhead of a controllable gate is about 20%. The total area increase for a given system depends on the number of gates replaced by C-gates and should be considerably smaller.

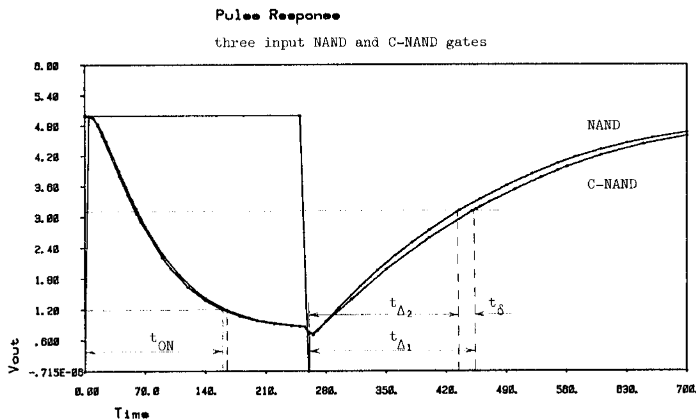


Fig.4: Timing diagram

3. MULTIPLEXING WITH CONTROLLABLE GATES

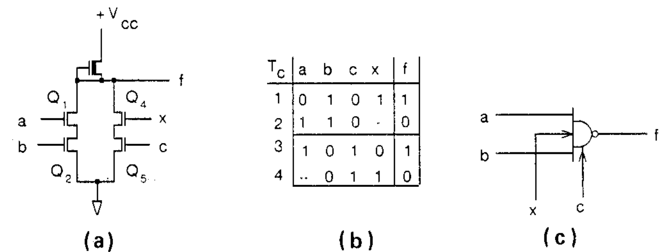
In order to increase observability of a given VLSI logic network, it is convenient to place low-cost multiplexers at specific points. Low-cost multiplexers allow the use of the existing logic, without a significant loss in performance, in the sharing of primary inputs/outputs in the test mode. We propose two controllable gates as low-cost multiplexers.

C'-gate:

For simplicity, consider a 2-input NAND gate, modified to perform the following function:

$$f = \begin{cases} \text{NAND}(a,b) & \text{if } C=0 \\ x & \text{if } C=1 \text{ and } a \bullet b = 0 \end{cases}$$

i.e. the C-NAND multiplexes the normal NAND output with output x. The circuit, the symbol and the test are shown in Fig.5.



T_C	a	b	c	x	f
1	0	1	0	1	1
2	1	1	0	-	0
3	1	0	1	0	1
4	..	0	1	1	0

Fig.5: C'-gate: (a) Circuit diagram (b) test and (c) logic symbol for C'-gate

It can be generalized to allow multiplexing of x_1, x_2, ...x_p under control of C_1, C_2, ...C_p respectively. This type of controllable gate can be implemented with any other functions.

When the output of the C'-gate is directly connected to the primary output, the propagation mode allows for the node connected to line x to be directly observable. This has a favorable impact on testing: by gaining access to the nodes which are not easily observable, the length of the test can be reduced. The C'-gate provides the means for inserting the low-cost multiplexers in the logic. The overhead is not significant since only two extra devices are being used per access point. However, due to the additional capacitance introduced, there will be some increase in the propagation delay of the C' gate compared to the NAND gate. This increase is comparable to the additional delay introduced by C-NAND, and is significantly smaller than the increase resulting from the addition of an extra gate. It is found that the degradation in delay amounts to approximately 10%.

The condition for setting one of the remaining inputs to a 0 is not difficult to meet since in the propagation mode the gate is used only to propagate information from some other part of the logic under test. Here, we assume that the C'-gate is chosen in such a way that the test to be propagated has no effect on the status of the logic containing C'-gate in particular. The gate itself is tested with the gate in the normal mode when the portion of the logic containing C'-gate is tested.

C"-gate:

A multiplexer, slightly more complex to implement but easier to use than the previous one, is defined as C"-gate. Again, we

illustrate the approach using a 2-input NAND gate. The output of C"-NAND is:

$$f = \begin{cases} \text{NAND}(a,b) & \text{if } C=0 \\ x & \text{if } C=1 \end{cases}$$

A circuit is shown in Fig.6.

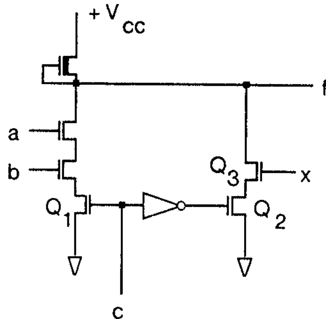


Fig.6: Circuit Diagram for the C"-NAND

In this case, there are three extra devices Q_1 , Q_2 , Q_3 and one extra inverter. This amounts to approximately two extra gates per control point C. The time penalty for introducing this type of multiplexer amounts to 10%. It should be noted that this is due to the implementation of the multiplexer structure where the devices Q_1 and Q_2 are connected to the ground side of the gate. The extra capacitance introduced by devices Q_1 and Q_2 is minimized because they have virtually been grounded. The only effect shown is by the ON resistance of device Q_1 (which can be made small by properly dimensioning Q_1) and the capacitance introduced by device Q_3. With careful circuit design, the time-area overhead caused by introducing such a multiplexer can be minimized.

4. DESIGN FOR TESTABILITY IMPROVEMENT

The controllable circuit structures, described above, are useful in partitioning combinational logic into relatively small partitions. These partitions can be tested exhaustively or with a modest effort in generating test patterns because of increased controllability and observability. The elimination of test pattern generation has beneficial effects on testing since

(a) the cost of testing is reduced by eliminating the high cost associated with generation of test patterns;
(b) the storage requirement for the test patterns is eliminated; and
(c) it is possible to generate the test on the chip itself, a significant step towards achieving a self-testing chip.

However, if test generation is required, partitioning of the logic helps to simplify the problem of generating the test by reducing the time needed for test generation.

The objective of this method is to achieve the testability with a reduced overhead in the time and area since it is very important that the introduction of testability features has minimal or no effect on the performance. The proposed method enhances the testability of the combinational portion of the VLSI chip in the following ways

(a) In the test mode, the logic is partitioned into small manageable partitions. The way in which the logic is partitioned and the size of the partitions are decided during the design stage. Several circuit analysis methods could be used [3],[4],[6].

(b) Paths are created to and from the partition in order to control the inputs and observe the outputs.

(c) If the partition is untestable because of the exsistence of reconvergent fanout, then the reconvergent fanout is eliminated by blocking the reconvergent path in the test mode or the partition is subdivided into two testable partitions.

During the design phase the gates which are replaced by C-gates are determined in such a way that the blocking of certain paths and the creation of others facilitates the division of the logic into subblocks or partitions. When the logic does not physically consist of separable partitions or the paritions cannot be accessed using "path senzitation", C-gates are used.

In the test mode, a partition to be tested is chosen and the logical values on the control lines are selected in such a way that the partitioning is enforced and the access to it from (to) the primary inputs (outputs) is created. The test for this particular partition is applied from the primary inputs and the response is observed on the primary outputs. If it passes the test, another partition is chosen and the process is repeated until the entire logic is covered.

Since the number of control lines can be significant and prohibit the use of external paths, the control lines can be controlled from an internal register (C-register) whose content is loaded serially from the outside through an additional pin.

Each cell of the register has control over one or several C-gates. The content of the register is observed through the additional I/O pin in order to test the register. If the VLSI chip is designed with the self-test capabilities, the C-register can be treated as an internal register that can be loaded from the on-chip microcontrol memory. The test procedure is performed in the following steps:

(a) The partition to be tested is selected. The content of the C-register is loaded and the pointer is advanced to the starting location of the test patterns for the particular partition;

(b) The test patterns for the particular partition are applied;

(c) The resulting output, after applying each test vector, is compared with the expected output;

(d) If the test for a particular partition is completed, select the next partition, and repeat steps (a) through (c) until all partitions are tested.

The C-gate approach can be also applied for cost-effective testing of sequential networks [8].

5. AN EXAMPLE

The test method using C-gates is illustrated by an example of testing the popular ALU 74181 (Fig.7). This method is comparable to the method proposed in [9] which uses the same example and achieves the exhaustive test with 1056 test vectors with a cost of 33% in time and area penalties. Our approach is as follows:

1. The circuit is logically partitioned into three partitions α , β and γ.

Partition α propagates the inputs to partition β. The inputs into the partition α are primary inputs A_i , B_i and control inputs S_i (i=0,1,2,3). The outputs from partition α are L_i and H_i (i=0,1,2,3), which are the functions of the inputs A_i , B_i selected by the control inputs S_i (i=0,1,2,3).

2. The inputs to partition β are primary inputs C_n and M, and lines L_i , H_i. The outputs from partition β are primary outputs F_i $(i = 0,1,2,3)$.

3. Partition γ has no primary inputs. Inputs to γ are A_i , L_i which are outputs from α . The outputs from γ are P' , G' and C_{n+4} which are primary outputs of 74181.

Only four NAND gates are replaced with the C-NAND gates. This is done with the purpose of achieving an independent access to the β partitions from the primary inputs.

The inputs of each partition are cycled through all possible combinations. The only assumption is that ***no fault makes the partitions dependent on each other.*** Stuck-at fault model complies with this assumption but some other faults, like bridging, may invalidate it. To prevent this, care should be taken that all the partitions chosen are physically located in such a way that the probability of this type of fault is minimized [5].

Utilizing the inherent properties of the partitions, the ALU can be tested with a total of 448 test patterns, with no need for pattern generation. Estimated penalties for the introduction of C-gates are 1.6% in area and 1.6% in additional propagation delay.

A recent report [3] shows that with a proper redesign of partition γ, this ALU could be tested with a minimum number of 11 test vectors. The algorithm of [3] can be used to locate the gates to be replaced with C-gates, instead of redesigning the network. In this example, a two-input NAND gate with output C_{n+4} can be replaced with a C-NAND gate in order to satisfy the condition for test reduction. The significance of the results of [3] is in providing a useful algorithm for the C-gate approach.

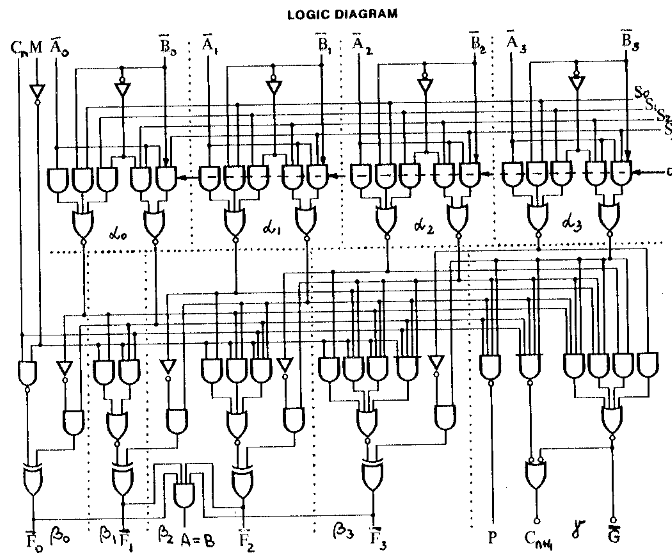


Fig.7: Example (74181 ALU)

6. CONCLUSION

This paper addresses the problem of VLSI testing and designing of testable structures. It presents an approach for improving the testability of VLSI systems by introducing the concept of controllable gates (C-gates) and a design technique for their use. The primary objective of the approach is to increase the access to internal points in the VLSI system without significantly changing the area and the time requirements of implementation. By replacing selected gates in the VLSI logic with C-gates, more testable structures are achieved in terms of the number of test patterns needed and the test generation time. Also, the properties of the C-gates in the test mode are utilized to create conveniently chosen partitions of the logic. By testing the partitions rather than the whole system, the problem of testing large structures can be simplified by using the "divide and conquer" approach.

REFERENCES

[1] J.P.Roth, Computer Logic Testing and Verification, Computer Science Press, 1980.

[2] H.Y.Chang,L.Manning,G.Metze, Fault Diagnosis of Digital Systems, Wiley-Interscience, 1970.

[3] S.B.Akers, B.Krishnamurty, A Test Counting Approach to Testability Analysis, presentation at the IEEE Design for Testability Workshop, Vail, Colorado, April 20-22, 1982.

[4] M. A. Leonard, Automatic Path Determination for Test Migration, presentation at the IEEE Design for Testability Workshop, Vail, Colorado, April 20-22, 1982.

[5] J. Galiay et. al., Physical Versus Logical Fault Models in MOS LSI Circuits, Impact on Their Testability, Proceedings of FTCS-9, Madison- Wisconsin, p. 195, June 1979.

[6] L.H.Goldstein, Controllability/Observability Analysis of Digital Circuits, IEEE Transaction on Circuits and Systems, Vol.CAS-26, No.9,p.685,September 1979.

[7] E.J.McCluskey, S.Bozurgui-Nesbat, Design for Autonomous Test , Digest of Papers 1980 Test Conference, Cherry Hill, New Jersey ,1980.

[8] V. G. Oklobdzija, Design for Testability of VLSI Structures through the Use of Circuit Techniques, PhD Dissertation, University of California Los Angeles, June 1982.

[9] E.J.McCluskey, S.Bozurgui-Nesbat, Design for Autonomous Test , IEEE Transaction on Computers, Special issue on Design for Testability, Vol. C-30, No.11,p.866,November 1981.

[10] SPICE-Users Manual, University of California Berkeley,1978.

[11] J.P.Hayes, A.D.Friedman, Test Point Placement to Simplify Fault Detection, IEEE Transaction on Computers, Vol.C-23, No.7,p.727, July 1974.

[12] M.J.Y.Williams, J.B.Angell, Enhancing Testability of Large Scale Integrated Circuits via Test Points and Additional Logic, IEEE Transactions on Computers, Vol.C-22,p.46,January 1973.

[13] E.B.Eichelberger, T.W.Williams, A Logic Design Structure for LSI Testing, Proc. 14th Design Automation Conference, p.462,June 1977,77CH1216-1C.