Functional Level Testability Measure Analysis in Digital Networks

SHIGERU TAKASAKI* and SHIGEHIRO FUNATSU*

This paper first defines the controllability/observability measures and the testability measure at the functional level, based on the gate level testability measure. Next, it describes an effective calculation method for the proposed functional level testability measure and offers an example of a typical functional block calculation. The method is based on a functional block description of a given network, and is easily applicable to various network categories. Finally, the relationships between the proposed functional level testability measure and the test data fault coverage are analyzed using the results of the D-algorithm ATPG program execution for a certain time. It is shown that the functional level testability measure can be used as an effective guideline to improve the testability bottlenecks of digital networks.

1. Introduction

1.1 Testing/Testability Measure Analysis Background

In order to manufacture highly reliable and cost-effective digital systems, LSI/VLSI technology has become an indispensable technological core. Though the technology has brought about great progress in the development of high level performance systems, it has also introduced several probelms into the menufacturing field. Testing is one of these problems. This testing will be repeated at several manufacturing levels, such as at the chip level, printed wiring board level and unit/system level. Therefore, testing efficiency and accuracy are key factors for reducing digital system manufacturing cost. A major problem in testing is test pattern generation. As logic networks become larger, automatic test pattern generation (ATPG) becomes more and more difficult.

This tendency is quite pronounced in the VLSI era. Experience shows that the ATPG procedural complexity is proportional to the size of the logic network, to the power of three.

Although there are many ATPG programs, none of them works well for large and highly sequential logic networks. Therefore, chip/logic designers have to pay careful attention to the networks before their designs have been completely drawn. This is called "Design for Testability" and has become one of the essential features governing digital network designs. However, it is quite difficult to predict quantitative complexity for ATPGs. This fact points to the earnest need for an effective testability yardstick in the early stage of logic designs [1], [2].

Up to now, several papers on the testability measure have been published. They are TMEAS by Grason and Stephanson, SCOAP by Goldstein, TESTSCREEN by Kovijanic, and CAMELOT by Bennetts [4]-[10]. However, some of them seem to be impractical for application to real networks. This is because they are too complicated to apply to large networks or they treat only gate level testability measures.

1.2 Testability Measure Analysis Objective

When designing various kinds of logic networks, there is an urgent need to determine the functional level testability measure. That is, it is necessary to know which functional part of a logic network is hard to test. The reason is that logic networks are generally composed of functional blocks. LSIs are designed with macroblocks, for example: selector, latch block and so on. In this case, functional blocks mean selector macro block, latch macro block and so on. On the other hand, Multi Chip Packages are designed with LSIs, for example: ALU, register file, decoder and so on. In this case, functional blocks mean ALU, register file, decoder and so on. Logic designers design their LSIs/Packages by using functional blocks. Therefore, if the designers find hard-to-test functional blocks after their initial network designs, it is quite convenient for them to carry out design modifications. In addition to the functional level testability measure, it is also quite important to be able to use this kind of testability measurement tool easily. The objective is to satisfy these requirements. That is, if the proposed testability measure can indicate the parts of a logic network which would be difficult to test, logic designers can easily modify their designs.

In Section **2**, the testability measure is defined at the functional level, after summarizing the gate level controllability and observability. Section **3** presents the functional block effective calculation method to determine controllability and observability at the functional level. A functional block example is given using the pro-

*CAD Engineering Dept., Computer Eng. Div., NEC Corporation., 10, 1-chome, Nisshin-cho, Fuchu City, Tokyo 183 Japan.

posed method. Section **4** shows the experimental results of the proposed testability measure at the functional level. The relationships between the proposed testability measure and test data fault coverage in testability measure ranges are expounded. Section **5** discusses an application of the proposed functional level testability measure. The measure can be applicable to testability bottleneck modifications in the initial network design.

2. Testability Measure at the Functional Level

This section summarizes the testability of the digital network and Goldstein's testability measure. Then, controllability/observability and the testability measure at the functional level are proposed.

Generally speaking, testability can be considered to be a function of controllability and observability. They are defined as follows [3]:

Controllability
Controllability is the ease with which test input patterns can be applied to the inputs of a subcircuit, by exercising the primary inputs of the circuit.

Observability
Observability is the ease with which the responses of a subcircuit can be determined by observing the primary outputs of the circuits.

For example, high testability logic networks have the following features.

A) The network can be easily initialized with simple manipulation (controllability).
B) The internal state of the network can be easily controlled with small test vector sequences (controllability).
C) The internal state of the network can be uniquely and easily identified through the primary outputs or the special test points (observability).
D) There are few correlations of the internal signals, i.e., reconvergent circuits, which are too complicated to test the network (controllability).

It is obvious that the combinational network offers the ideal amount of controllability and observability, because it has no internal state and its primary outputs can be directly controlled by an input vector on the primary inputs. On the other hand, a highly sequential network, for example, the binary counter, is subject to adverse effects from controllability and observability, because it has internal states and these states can be controlled only by its own control signals, including the clock [13]. However, these controllability and observability definitions are only a qualitative measure of the testability.

2.1 Testability Measure at the Gate Level

In order to evaluate the digital network testability, it is necessary to have a quantitative measure. Goldstein defined this kind of measure for gate level testability in his testability program [4]. These measures can be rephrased as follows:

Definitions:

0-Controllability
The minimum number of elements (gates) which should be set at a known value in order to obtain a logical 0 at a specified element (gate).

1-Controllability
The minimum number of elements (gates) which should be set at a known value in order to obtain a logical 1 at a specified element (gate).

Observability
The minimum number of elements (gates) which should be set at a known values for propagating the logical value of a specified element (gate) to primary outputs.

From the definitions, 0/1-controllability and observability measures values are integers. The smaller each measure becomes, the higher testability the network exhibits. Details of the procedure necessary for calculating the gate level testability measure are described in Ref. [5] and are omitted here.

2.2 Testability Measure at the Functional Level

As stated previously, it is quite important to identify whether or not the functional blocks are easy to test in the early digital network design stages. Consider detecting faults test generation difficulty in a functional block. For simplicity, assume a functional block, FB, including AND, NAND and OR gates and assume a stuck-at-0 fault, f_1, as depicted in Fig. 1. To detect this fault, it is necessary to set the following logic states: AND1 is a logic 1, AND2 is a logic 0 and AND3 is a logic 0. To realize these logic states, for example, all inputs must be set (1, 2, 3, 4, 5, 6, 7, 8, 9)=(1, 1, 1, 1, 0, 1, 0, 1, 1) as depicted in Fig. 1. This test also detects a stuck-at-0 fault, f_0, for line 1. Therefore, from test generation and fault detection points of view, it is possible to consider that fault f_1 is represented by fault f_0.

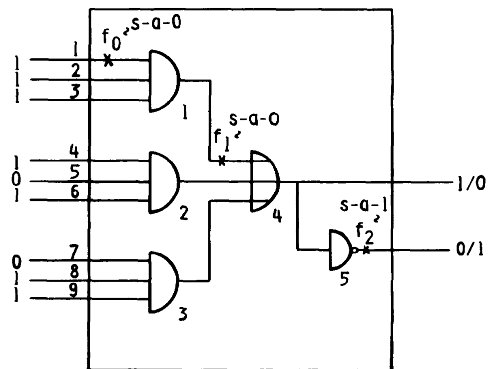


Fig. 1 A functional block FB including AND, NAND and OR gates.

Similarly, f_0 is also a representative fault for a stuck-at-1 fault f_2. In the case of this kind of a combinational circuit block, all faults in the block are represented by its input signal lines' faults. Strictly speaking, it is difficult to represent all faults in a general block by its input signal line faults. However, internal faults in a block, which has a combinational circuit and tree configuration as in Fig. 1, are mostly represented by its input faults. If all faults in the block are represented by its input faults, it is sufficient for test generation to take into consideration the input faults only. This concept can apply to functional block level testability measure definitions. That is, functional block level 0/1-controllability is represented by its input signal line 0/1-controllability. Similarly, functional block level observability is represented by the input signal line observability. For example, if the functional block input signal lines are difficult to set at a logic 0, the total signal line 0-controllability becomes large. This leads to an idea that the functional block 0-controllabity is large. On the other hand, if all input signal lines are easy to set at a logic 0, the total signal lines 0-controllability is also small. As a result, functional block 0-controllability may be small.

From the above consideration, functional block FB's 0/1-controllability, controllability, observability and testability measures are defined as follows:

Functional block FB's 0/1-controllability measure is defined as a function of gate level 0/1-controllability for FB inputs, the number of logically available pins and a weighting factor for each pin.

A(FB): A functional block FB 0-controllability measure.

$$A(\mathrm{FB})=\frac{\sum_{i=1}^{P} W_{0i}C^0(i)}{P} \tag{1}$$

B(FB): A functional block FB 1-controllability measure.

$$B(\mathrm{FB})=\frac{\sum_{i=1}^{P} W_{1i}C^1(i)}{P} \tag{2}$$

where

$C^0(i)$: block input pin i 0-controllability measure for functional block FB,

$C^1(i)$: block input pin i 1-controllability measure for functional bolock FB,

P: number of logically available pins,

W_{0i}: 0-controllability weighting factor for pin i.

W_{1i}: 1-controllability weighting factor for pin i.

Functional block FB's controllability measure is defined as a function of FB's 0, 1-controllability and weighting factors.

C(FB): A functional block FB controllability measure.

$$C(\mathrm{FB})=\sqrt{(W_aA(\mathrm{FB}))^2+(W_bB(\mathrm{FB}))^2} \tag{3}$$

where

W_a: 0-controllability weighting factor for functional block FB,

W_b: 1-controllability weighting factor for functional block FB.

The functional block FB's observability is defined as a function of FB's input gate level observability, the number of logically available pins and a weighting factor for each pin.

O(FB): A functional block FB observability measure.

$$O(\mathrm{FB})=\frac{\sum_{i=1}^{P} W_{oi}C^0(i)}{P} \tag{4}$$

where

$CO(i)$: block input pin i observability measure for functional block FB,

W_{oi}: observability weighting factor for pin i.

The functional block FB's testability measure is defined as a function of FB's controllability, observability and a set of weighting factors.

T(FB): A functional block FB testability measure.

$$T(\mathrm{FB})=\sqrt{(W_cC(\mathrm{FB}))^2+(W_oO(\mathrm{FB}))^2} \tag{5}$$

where

W_c: controllability weighting factor functional block FB,

W_o: observability weighting factor for functional block FB.

[A functional block testability measure example]

Assume a selector functional block as depicted in Fig. 2. The selector functional block, FB, 0/1-controllability, controllability, observability and testability measures are obtained from Eqs. (1) to (5). They are as follow:

$$A(\mathrm{FB})=\frac{\sum_{i=1}^{4} W_{0Di}C^0(D_i)+\sum_{i=1}^{2} W_{0Ci}C^0(C_i)}{6} \tag{6}$$

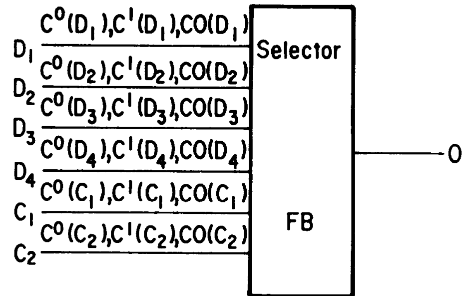


Fig. 2 A selector functional block example.

$$B(\mathrm{FB})=\frac{\sum_{i=1}^{4} W_{1Di}C^1(D_i)+\sum_{i=1}^{2} W_{1Ci}C^1(C_i)}{6} \quad (7)$$

$$C(\mathrm{FB})=\sqrt{(W_aA(\mathrm{FB}))^2+(W_bB(\mathrm{FB}))^2} \quad (8)$$

$$O(\mathrm{FB})=\frac{\sum_{i=1}^{4} W_{ODi}CO(D_i)+\sum_{i=1}^{2} W_{OCi}CO(C_i)}{6} \quad (9)$$

$$T(\mathrm{FB})=\sqrt{(W_cC(\mathrm{FB}))^2+(W_oO(\mathrm{FB}))^2} \quad (10)$$

where

W_{0Di}: data line i 0-controllability weighting factor,
W_{0Ci}: control line i 0-controllability weighting factor,
W_{1Di}: data line i 1-controllability weighting factor,
W_{1Ci}: control line i 1-controllability weighting factor,
W_{ODi}: data line i observability weighting factor,
W_{OCi}: control line i observability weighting factor,
W_a: 0-controllability weighting factor for FB,
W_b: 1-controllability weighting factor for FB,
W_c: controllability weighting factor for FB,
W_o: observability weighting factor for FB.

[Weighting factors]

To determine optimal weighting factors in Eqs. (1)–(5), it is necessary to consider the following factors.

i) Role of each functional block in the network.
ii) Functional block's attributes.
iii) Characteristics of the signal lines.

i) The role of each functional block means what kind of role each functional block plays in the network. For example, from the test generation view point, functional blocks used in a control path play more important roles than those of a data path. Therefore, functional block controllability and observability weighting factors W_c and W_o used in a control path have more weight than those for a data path. Furthermore, the role of each block in the network influences functional block 0, 1-controllability weighting factors W_a and W_b. They depend on how easily functional blocks' inputs are controlled to a logic 0 and 1. When some functional block inputs are quite difficult to set at logic 0 and easy to set at logic 1 from a circuit configuration, these functional block 0-controllability weighting factors are heavier than the 1-controllability weighting factors.

ii) The functional block's attributes also influence W_a, W_b, W_c and W_o. For example, sequential functional blocks, like registers and counters, have heavier weighting factors than combinational functrional blocks, like selectors and decoders. The combinational circuits can be directly controlled by the block input. On the other hand, sequential circuits are subject to adverse effects from controllability and observability. This is because they have internal states and these states can be controlled only by their own control signals.

iii) The signal lines' characteristics mainly determine each pin's weighting factors W_{0i}, W_{1i} and W_{oi}. Assume a 4-to-1 selector. This selector has 4 data lines and 2 control lines. The control lines play more important roles than the data lines. If the control lines don't have 4 logic states, i.e., (0,0), (0,1), (1,0), (1,1), all faults on the data lines cannot be found. Therefore, the control lines have heavier weighting factors than the data lines.

Sometimes it is not easy to determine appropriate weighting factors. Therefore, from the practical computation point of view, each weighting factor is set to one. This means that signal lines and functional blocks have the same priority in networks. These weighting factors make functional block testability computations much simpler. The experimental results discussed later show that the same priority weighting factors are sufficient to find testability bottleneck blocks. That is, from a practical point, it is sufficient for weighting factors to be assigned a value of one. More appropriate weighting factors depend on further research.

3. Testability Measure Calculation Method at the Functional Level

This section describes a testability measure calculation method at the functional level. Then, it offers a typical functional block example. The testability measure calculation method described in this paper is based on the SCOAP procedure [4].

3.1 Testability Measure Calculations

The basic testability measure calculation procedure consists of the following three steps.

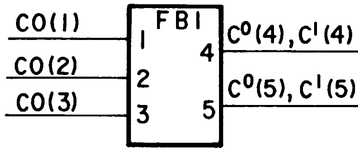
S–1: Determine the controllability and observability calculating equations for each functional block in the network. The resulting equations are registered as subroutines, that are used in the following step S–2.
S–2: Calculate controllability and observability for each functional block input/output pin in a given network.
S–3: Calculate the testability measure (Equation(5)) for each functional block in the network.

[S–1]

The controllability and observability equations in S–1 take into consideration the gate connections in the functional block. Fig. 3 shows an example of the S–1 equations. The Functional Block (FB1) has three inputs (1, 2 and 3) and two outputs (4 and 5). The controllability (C^0=0-controllability, C^1=1-controllability) and observability (CO) equations for each output/input pin are determined as a function of its inputs and outputs.

$$\begin{aligned} C^0(4)&=f_{11}(1, 2, 3)\\ C^1(4)&=f_{21}(1, 2, 3)\\ C^0(5)&=f_{12}(1, 2, 3)\\ C^1(5)&=f_{22}(1, 2, 3) \end{aligned} \quad (11)$$

$$\begin{aligned} CO(1)&=f_{31}(2, 3, 4, 5)\\ CO(2)&=f_{32}(1, 3, 4, 5)\\ CO(3)&=f_{33}(1, 2, 4, 5) \end{aligned} \quad (12)$$



$$C^0(4) = f_{11}(1, 2, 3)$$
$$C^1(4) = f_{21}(1, 2, 3)$$
$$C^0(5) = f_{12}(1, 2, 3)$$
$$C^1(5) = f_{22}(1, 2, 3)$$
$$CO(1) = f_{31}(2, 3, 4, 5)$$
$$CO(2) = f_{32}(1, 3, 4, 5)$$
$$CO(3) = f_{33}(1, 2, 4, 5)$$

Fig. 3 An example of S-1.

Once functional block calculating equations are established, they are registered as subroutines in a library. If the same functional block is used in other networks, these subroutines are called from the library.

[S-2]

The S-2 calculation is basically similar to the SCOAP procedures. To facilitate understanding, only the combinational measure procedures are shown. The sequential measure can be handled with procedures similar to that for the combinational measure. Functional block controllability calculations are as follows:

P-1: Initializations.
For each primary input (PI) which connects with the functional block,
$C^0(\text{PI}) = C^1(\text{PI}) = 1$
other inputs (IN) for the functional blocks
$C^0(\text{IN}) = C^1(\text{IN}) = \infty$

P-2: Calculation for each functional block.
The functional block measure calculations proceed from primary inputs to primary outputs. Each functional block output is determined from the functional block equations, as in Eq. (11). If there is a feedback line within a certain block, the calculation is repeated after determining the line value. The calculations are reiterated until the functional block values are stable.

The following theorem concerns the 0/1-controllability measure value stabilization. The value is an integer from the 0/1-controllability measures definitions.

Theorem 1:
If a network is logically stable (i.e., logic states in a network are stable after applying an input vector) each functional block's input/ouput 0/1-controllability measure values are also stable.

Proof:
i) A combinational network.
From the 0/1-controllability measure definitions, it is evident that each element's 0/1-controllability measure values are stable. Therefore, each functional block's input/ouput 0/1-controllability values are also stable.
ii) A network including loop circuits.
If a primitive element's 0/1-controllability measure values are not stabilized after a certain time, it is clear that the minimum number of elements for obtaining the 0/1-controllability measure are changing from the definitions. This leads to the network not being stabilized logically; for example, logic oscillations occur. Therefore, if the logic states in the network are stable, each functional block's input/output 0/1-controllability measure values are stable.

Q.E.D.

Functional block observability calculations are as follow:

P-1: Initializations.
For each primary output (PO), which connects with the functional block,
$CO(\text{PO}) = 0$
other outputs (OU) for the functional blocks
$CO(\text{OU}) = \infty$

P-2: Calculations for each functional block.
The functional block measure calculations are performed one after the other from primary outputs to primary inputs. Each functional block input is determined from the functional block equations, as in Eq. (12).

The observability measure values are also stable.

Theorem 2:
If a network is logically stable, each functional block's input/output observability measure values are also stable.

Proof:
From theorem 1 and the observability equation (see Appendix), it is evident that each functional block's input/output observability values are stable.

Q.E.D.

[S-3]

The testability measure for each functional block (S-3) is calculated through the following procedures, after each weighting factor is established according to the functional block attributes and signal line characteristics. Standard weighting factors are set to one.

P-1: Determine the controllability measure (Eq.

(3)) for a given functional block.

P-2: Determine the observability measure (Eq. (4)) for a given functional block.

P-3: Determine the testability measure (Eq. (5)) for a given functional block.

3.2 Functional Block Circuit Example

As a functional block circuit example, let's select LS258 2-to-1 selectors. Fig. 4 shows the LS258 circuit diagram. LS258 is a part of a large network.

[S-1] Controllability and observability equations.
The controllability equations for each pin are as follows:
(Controllability equations)

$$
\begin{aligned}
C^0(04) &= \mathrm{Min}\{C^0(02),\ C^0(01)\} \\
&\quad + \mathrm{Min}\{C^0(03),\ C^1(01)+a\}+2a \\
C^1(04) &= \mathrm{Min}\{C^1(02)+C^1(01),\ C^1(03)+C^0(01) \\
&\quad +a\}+2a \\
&\ \vdots \\
C^0(12) &= \mathrm{Min}\{C^0(14),\ C^0(01)\} \\
&\quad + \mathrm{Min}\{C^0(13),\ C^1(01)+a\}+2a \\
C^1(12) &= \mathrm{Min}\{C^1(14)+C^1(01),\ C^1(13) \\
&\quad +C^0(01)+a\}+2a
\end{aligned}
\tag{13}
$$

where a is a measure of gate/flip-flop circuit complexity, e.g., $a=1$ in Ref. [5].
In this paper, the primitive AND and OR gates are $a=1$. A D-type flip-flop is $a=4$. This is because it is composed of a few primitive gates and can be considered to have about four times the complexity compared to primitive gates. However, a scan path flip-flop is considered as a primary input/output. Input test data can be set into the scan path flip-flops and output test data can be observed from them. Primitive AND and OR gates' controllability/observability equations are described in the Appendix. The observability equations for each pin are as follows:
(Observability equations)

$$
\begin{aligned}
CO(02) &= CO(04)+\mathrm{Min}\ \{C^0(03),\ C^1(01)+a\} \\
&\quad +C^1(01)+2a \\
CO(03) &= CO(04)+\mathrm{Min}\ \{C^0(02),\ C^1(01)\} \\
&\quad +C^0(01)+3a \\
&\ \vdots \\
CO(01) &= \mathrm{Min}\ \{CO(04)+\mathrm{Min}\ \{C^0(03),\ C^1(01)+a\} \\
&\quad +C^1(02)+2a, \\
&\qquad CO(04)+\mathrm{Min}\ \{C^0(02),\ C^0(01)\} \\
&\qquad +C^1(03)+3a \\
&\qquad \vdots \\
&\qquad CO(12)+\mathrm{Min}\ \{C^0(13),\ C^1(01)+a\} \\
&\qquad +C^1(14)+2a, \\
&\qquad CO(12)+\mathrm{Min}\ \{C^0(14),\ C^0(01)\} \\
&\qquad +C^1(13)+3a\}
\end{aligned}
\tag{14}
$$

The controllability and observability equations are registered as an LS258 subroutine in a library.

[S-2] Functional block controllability and observability calculations.
First is the initializations for the controllability calculations. LS258 inputs are initialized as follows:

$$
\begin{aligned}
C^0(02) &= \infty \\
C^1(02) &= \infty \\
&\ \vdots \\
C^0(01) &= \infty \\
C^1(01) &= \infty
\end{aligned}
$$

Fig. 5 shows the initial states in the calculations. Other

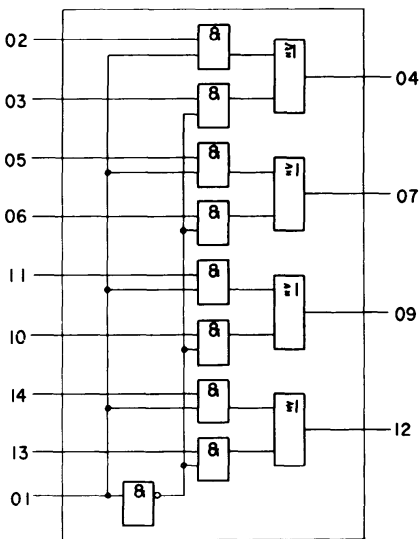


Fig. 4 LS258 circuit diagram.

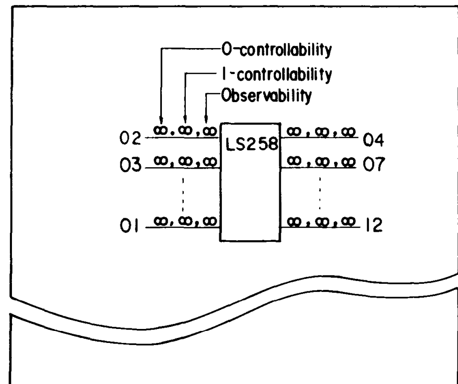


Fig. 5 Initial states of controllability and observability calculations.

functional blocks in the network are also initialized with a large number. Second is each functional block controllability calculation. After the controllability calculations are stable, LS258's input/output 0,1-controllability measures are as follow:

$$C^0(02)=50 \qquad C^0(04)=85$$
$$C^1(02)=75 \qquad C^1(04)=91$$
$$\vdots \qquad \vdots$$
$$C^0(01)=81 \qquad C^0(12)=60$$
$$C^1(01)=66 \qquad C^1(12)=109$$

Third is initializations for the observability calculations.

LS258 outputs are initialized as follows:

$$CO(04)=\infty$$
$$CO(07)=\infty$$
$$CO(09)=\infty$$
$$CO(12)=\infty$$

Other functional blocks are also initialized with a large number. Last is each functional block observability calculation. After the observability calculations are stable, LS258's input ouput observability measures are as follows:

$$CO(02)=152 \qquad CO(04)=140$$
$$CO(03)=132 \qquad CO(07)=120$$
$$\vdots \qquad \vdots$$
$$CO(01)=171 \qquad CO(12)=94$$

Fig. 6 shows the final states of the package level calculations.

[S-3] Functional block testability calculation.
LS258 is as follows:

$$\text{0-controllability: } A(\text{LS258})=\frac{549}{9}=61$$

$$\text{1-controllability: } B(\text{LS258})=\frac{828}{9}=92$$

$$\text{Controllability: } C(\text{LS258})=\sqrt{61^2+92^2}=102$$

$$\text{Observability: } O(\text{LS258})=\frac{1179}{9}=131$$

$$\text{Testability: } T(\text{LS258})=\sqrt{102^2+131^2}=181 \qquad (15)$$

where each weighting factor is 1.

The testability measure for LS258 is given in the LS258 block in Fig. 6. Fig. 7 shows an example of another network testability map. In Fig. 7, FB1 testability measure is 151. On the other hand, FB6, is 138. Therefore, testing FB1 should be more difficult than testing FB6. The testability measure calculations have been implemented as NETMAP (NEC Testability Measure Analysis Program) using an event driven simulation technique. NETMAP has a potential for faster calculation than the gate level method SCOAP. This is because it can handle functional blocks and adopts only

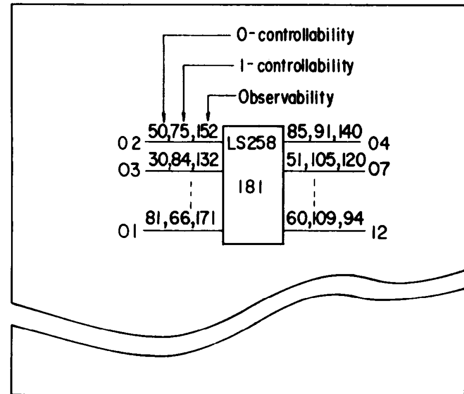


Fig. 6 Final states of controllability and observability calculations.

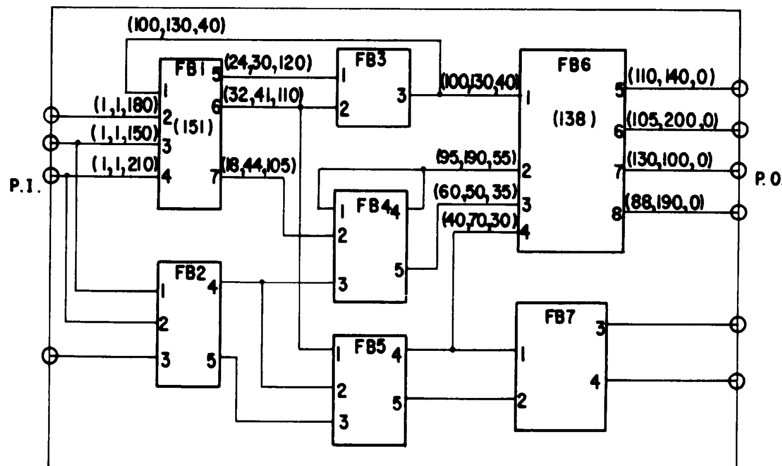


Fig. 7 An example of a testability map.

calculating equations and the event driven technique. NETMAP's performance is shown in Table 2.

3.3 Advantages of Functional Level Testability Measure/Calculations

The following advantages accrue when the functional

Table 1 Functional block testability measure execution report.

No.	FB Name	Loca-tion	FB *C0*	FB *C1*	FB *CO*	FB *TM*	Defined Faults	Detected Faults	Undetected Faults	Fault Coverage
1	C 062	03E	1	1		4	5	5	0	100.0
2	Q 081	00G	1	1	13	13	257	174	83	67.7
3	Q 081	03C	1	1	13	13	257	230	27	89.5
4	Q 081	03D	1	1	13	13	257	216	41	84.1
5	Q 081	03F	1	1	13	13	253	165	88	65.2
6	Q 081	00F	2	2	14	14	253	215	38	85.0
7	Q 081	01C	1	1	14	14	253	231	22	91.3
8	Q 081	01D	1	1	14	14	257	234	23	91.1
9	Q 081	05E	1	1	14	14	257	205	52	79.8
10	Q 081	06E	1	1	14	14	257	204	53	79.4
11	Q 081	02G	2	2	15	15	257	206	51	80.2
12	Q 081	06D	2	2	16	16	257	205	52	79.8
13	Q 081	07D	2	2	16	16	257	225	32	87.6
14	Q 081	03B	10	11	15	21	257	160	97	62.3
15	B 006	03H	1	1	22	22	67	59	8	88.1
16	B 006	07A	1	1	23	23	66	64	2	97.0
17	B 088	00H	1	2	27	27	31	31	0	100.0
18	B 006	01H	1	1	27	27	59	51	8	86.4
19	Q 081	03A	18	17	16	29	257	195	62	75.9
20	Q 081	02D	19	23	17	34	253	191	62	75.5
21	Q 081	01E	21	23	17	35	257	159	98	61.9
22	B 088	01A	3	5	37	37	30	30	0	100.0
23	G 023	04B	4	4	42	42	352	238	114	67.6
24	B 088	00A	1	1	43	43	29	18	11	62.1
25	G 023	05B	4	5	47	47	337	239	98	70.9
26	G 023	00E	10	9	51	52	258	200	58	77.5
27	B 006	06A	1	1	52	52	66	59	7	89.4
28	B 089	02A	7	10	52	53	41	38	3	92.7
29	G 023	02E	7	9	56	57	337	240	97	71.2
30	G 024	04H	6	6	57	57	152	115	37	75.7
31	D 055	06B	1	1	76	76	94	84	10	89.4
32	G 024	04C	4	4	93	93	397	231	166	58.2
33	D 055	00C	1	1	103	103	96	75	21	78.1
34	G 024	04G	7	7	105	105	171	69	102	40.4
35	D 055	04F	1	1	108	108	80	55	25	68.8
36	G 024	00D	6	4	117	117	367	248	119	67.6
37	D 055	03G	1	1	119	119	94	67	27	71.3
38	D 055	00B	1	1	122	122	80	69	11	86.3
39	D 055	02H	1	1	126	126	94	71	23	75.5
40	D 055	06F	2	1	131	131	94	50	44	53.2
41	G 024	02B	4	5	136	136	374	241	133	64.4
42	B 089	07F	33	31	130	137	34	23	11	67.7
43	D 055	01F	1	1	148	148	94	79	15	84.0
44	D 055	05G	2	1	151	151	92	36	56	39.1
45	Q 081	07F	151	19	15	152	158	154	4	97.5
46	G 024	04E	6	7	157	157	146	39	107	26.7
47	G 024	06C	5	4	157	157	374	242	132	64.7
48	D 055	04A	1	7	162	162	94	63	31	67.0
49	F 022	05A	9	57	169	178	48	10	38	20.8
50	D 055	07H	2	1	178	178	94	27	67	28.7
51	F 022	01G	9	48	189	195	48	10	38	20.8
52	G 024	06H	6	6	207	207	171	0	171	0.0
53	D 055	07C	2	1	242	242	94	31	63	33.0
54	B 089	04D	29	97	242	262	32	18	14	56.3
55	F 022	05H	12	72	277	286	45	13	32	28.9
56	E 040	06G	5	7	297	297	231	109	122	47.2
57	E 040	01B	5	7	299	299	223	138	85	61.9
58	E 040	05D	6	7	314	314	227	71	156	31.3
59	E 040	07G	7	7	349	349	231	79	152	34.2
60	B 089	07B	32	70	395	402	39	11	28	28.2

level testability measure/calculations are used in a digital network testability analysis.

A) Handling hierarchy logic designs,
B) Determining functional level controllability, observability and testability measures quickly,
C) Achieving correspondence between block testability measures and functional blocks.

Recently, hierarchy design methodology has been widely used. LSIs are designed with macro blocks. Multi Chip Package are designed with MSIs and LSIs. In these circumstances, it is convenient to obtain the controllability, observability and testability measures at each stage without breaking down primitive gates. If functional level calculations are carried out using the proposed method, expansions from functional blocks to primitive gates and gate level calculations are not necessary. Consider that LSIs are composed of macro blocks. If macro block equations, like Eqs.(13) and (14), are prepared, the testability calculations can be executed at the macro block level without expanding primitive gates.

The network described in Table 1 is a circuit which compares read addresses with store addresses for pipeline operations in execution processing units. Though the network size includes about 2,950 gates, only ten kinds of functional blocks are used. They are C062 (decoder), Q081 (shift register), B006 (buffer), B088 (buffer), G023 (binary adder), B089 (buffer), G024 (binary adder-parity), F022 (look ahead carry) and E040 (parity). This network's controllability, observability, and testability measures can be calculated, if ten kinds of block equations are prepared. Therefore, if functional block equations are registered as subroutines in a library, it is possible to handle hierarchy logic designs and, as a result, determine the controllability, observability and testability measures quickly.

The important point in the functional level testability measure is the correspondence between the block testability measures and the functional blocks. Logic networks are generally composed of functional blocks. After finishing the network designs, logic designers are eager to find which functional block is hard to test in the networks. If the proposed functional level testability measure can indicate testing difficulty for blocks, the designers can easily apply these testability analysis results to their design modifications. This is because design modifications are easy at this stage. This will be discussed later as an application of the proposed measure.

4. Testability Measure Evaluation

In this section, first testability measure evaluation processes are described. Then, experimental results obtained by the proposed testability measures are shown. Finally, relationships between the gate level testability measure and the functional level testability measure are discussed.

4.1 Evaluation Process

In order to evaluate the proposed testability measure effectiveness, an ATPG program is run for a certain time after testability measure calculations. A system configuration for the testability measure evaluation is depicted in Fig. 8.

Irn Fig. 8, Circuit File (①) is an original circuit connection file, for example, block to block connections and gate to gate connections. NETMAP (②) is NEC Testability Measure Analysis Program. Circuit File (③) is the circuit connection file with functional block, 0, 1-controllability, observability and testability measures. ATPG (④) is an automatic test pattern generation program using the D-algorithm. The ATPG system includes a parallel fault simulator. That is, this system conducts a fault simulation with test patterns generated by the ATPG. Fault selections for test generation are at random. List (⑤) has correlation reports of numbers of undetected faults, block fault coverage and testability measure after running the ATPG program. This is sorted by testability measure values. The evaluation system clarifies relationships between the functional block fault coverage and the functional block testability measure under the ATPG program execution for a certain time. The functional block fault coverage is the ratio of block undetected faults to block total defined faults. If there are 150 block total defined faults, for example, and 120 faults are detected by the ATPG, the functional block fault coverage is 80%.

The functional block testability measure is the measure defined in Eq. (5). Furthermore, the system shows relationships between fault coverage in some testability measure range and a testability measure range. The fault coverage in some testability measure range means the ratio of total functional blocks detected faults to functional block total defined faults

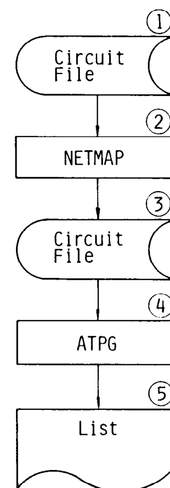


Fig. 8 A system configuration for the testability measure evaluation.

in some testability measure range. The testability measure range is the testability measure value range, for example, from 100 to 200 or from 500 to 700.

Table 1 shows the functional block testability measure execution report.

This circuit includes about 2,950 elements (gates). The ATPG execution requires two hours (on a 1 MIPS Machine) and the fault coverage is 70%. In Table 1, the FB name is the functional block name used in the network. FB C^0, C^1, CO and TM are the functional block 0, 1-controllability, observability and testability measures. Defined faults, Detected faults, Undetected faults and Fault coverage are the block defined faults, detected faults, undetected faults and fault coverage, respectively. For example, No. 3, FB name Q081, has $C^0=1$, $C^1=1$, $CO=13$, $TM=13$, defined faults=257, detected faults=230, undetected faults=27 and fault coverage=89.49%.

4.2 Experimental Results

Several practical networks were investigated based on the proposed testability measure. Table 2 shows the evaluated network attributes, which are number of elements (gates), number of functional blocks, number of inputs, number of outputs, number of scan flip-flops, number of signal lines, number of faults, the NETMAP execution time (seconds), the ATPG execution time (hours) at 1MIPS, and the circuit fault coverage.

Since these networks include scan path flip-flops, they are considered as combinational circuits.

Table 3 shows functional level testability measure evaluation results. In Table 3, 1*, 2* and 3* are the following ratios.

$$1^*: \frac{\text{No. of FBs in each testability range}}{\text{No. of total FBs}} (\%) \tag{16}$$

$$2^*: \frac{\text{No. of defined faults in each testability range}}{\text{No. of total defined faults}} (\%) \tag{17}$$

$$3^*: \frac{\text{No. of detected faults in each testability range}}{\text{No. of defined faults in each testability range}} (\%) \tag{18}$$

where

FBs are functional blocks. Testability range is the testability measure range discussed previously.

Fig. 9 shows relationships between the functional block fault coverage and testability measure range in Table 3. From Fig. 9, as a whole, the following tendency becomes apparent:

- As the testability measure increases, the functional block fault coverage in the range becomes lower.

4.3 Relationships between Gate Level Testability Measure and Functional Level Testability Measure

Table 3 and Fig. 9 have shown that the proposed functional level testability measure has a good correlation with the fault coverage in functional block testability ranges. This implies that primitive gates in functional blocks also have the same correlation in terms of relationships between a gate level testability measure and fault coverage in some testability range. The next step is to define the gate level testability measure and evaluate the functional level testability measure and the gate level testability measure. The gate measures are defined in a pattern similar to that for from Eqs.(1) to (5).

Definitions:

A(G): A gate G 0-controllability measure.

Table 3 Relationships between functional block fault coverage and testability measure range.

Circuit Name	Testability range	0 ~ 99	100 ~ 199	200 ~ 299	300 ~ 399	400 ~ 499	500 ~ 599	600 ~ 699	700 ~ 799
A	1*	41.4	34.4	12.1	6.9	1.7	1.7	—	1.7
	2*	23.7	46.5	16.4	6.4	1.7	3.2	—	2.1
	3*	94.9	86.6	84.4	40.5	51.8	30.9	—	30.3
B	1	53.3	30	11.7	3.3	1.7			
	2	60.7	25.8	8.5	4.6	0.4			
	3	72.9	60.2	40.8	41.7	23			
C	1	71.7	28.3						
	2	81	1.9						
	3	91.4	68.3						
D	1	61	16.9	10.2	6.8	1.7	1.7	—	1.6
	2	41	33	13	6.2	1.7	3.1	—	2
	3	92.6	85.2	80.6	40.5	55.5	30.9	—	30.3
E	1	54.2	31.7	10	3.3	1.7			
	2	62	25.4	7.4	4.4	0.8			
	3	76.6	61.8	38.8	32.8	28.2			
F	1	94.9	5.1						
	2	96.8	3.2						
	3	55	34						

Table 2 Networks evaluated by proposed testability measure.

Circuit Name	Element #	FB #	Input #	Output #	Scan F/F #	Signal line #	Fault #	NETMAP (seconds) 1 MIPS	ATPG (hours) 1 MIPS	Coverage (%)
A	2472	58	118	59	15	6761	9163	583s	2H	84.6
B	2912	60	117	12	170	6067	9249	414s	2H	68.2
C	2256	60	108	88	120	4265	6209	297s	2H	87.8
D	2503	59	118	69	15	6793	9204	589s	2H	84.2
E	2930	60	117	12	170	6064	9241	435s	2H	69.4
F	3238	59	135	94	115	6174	8100	451s	2H	54.5

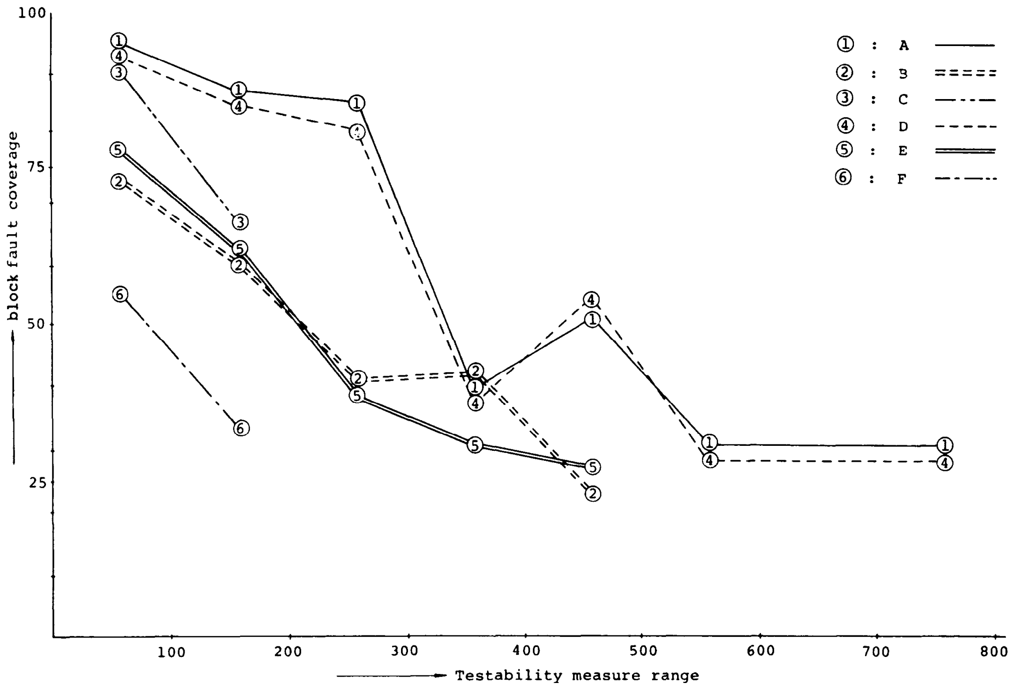


Fig. 9 Relationships between the functional block fault coverage and the testability measure range.

$$A(G)=\frac{\sum_{m=1}^{gp} C^0(m)}{gp} \tag{19}$$

$B(G)$: A gate G 1-controllability measure.

$$B(G)=\frac{\sum_{m=1}^{gp} C^1(m)}{gp} \tag{20}$$

$C(G)$: A gate G controllability measure.

$$C(G)=\sqrt{A(G)^2+B(G)^2} \tag{21}$$

$O(G)$: A gate G observability measure.

$$O(G)=\frac{\sum_{m=1}^{gp} CO(m)}{gp} \tag{22}$$

$T(G)$: A gate G testability measure.

$$T(G)=\sqrt{C(G)^2+O(G)^2} \tag{23}$$

In Eqs. (19)-(22),

$C^0(m)$: input pin m 0-controllability measure for gate G,

$C^1(m)$: input pin m 1-controllability measure for gate G,

$CO(m)$: input pin m observability measure for gate G,

gp: number of available gate input pins.

Let's choose network A, B and E in Table 2 and evaluate relationships between the gate level testability measure and fault coverage in the testability range under the same condition. These results are described in Table 4 and shown in Fig. 10.

Table 4 Relationships between gete fault coverage and testability measure range.

Circuit Name \ Testability range		0 ∫ 99	100 ∫ 199	200 ∫ 299	300 ∫ 399	400 ∫ 499	500 ∫ 599	600 ∫ 699	700 ∫ 799	800 ∫ 899	900 ∫
A	1*	63.2	7.4	9.7	5.4	5.8	2.1	3.5	2.5	0.3	
	2*	55	5.8	26	3.8	4	1.6	2.5	1.3	0.1	
	3*	92.4	77.9	89.2	54.3	57.6	43.9	58.8	48	15.4	
B	1	65.8	17.1	8.6	8	0.2	—	—	0.06	0.06	0.18
	2	64.1	18.8	8.5	8.3	0.1	—	—	0.06	0.06	0.1
	3	76.5	61.1	49.8	55.3	41.7	—	—	33.3	33.3	30
F	1	88.3	7.6	4	0.1						
	2	86	9.7	4.2	0.1						
	3	65.3	24.2	25.8	12.5						

Though there are a few ups/downs in Fig. 10, as a whole the gate level testability measures show a similar tendency.

By comparing Fig. 9 with Fig. 10, it is shown that each netowork has a similar trend. For example, network B's curve in Fig. 9 nearly the same trend as network B's curve in Fig. 10. This means that there is a good correlation between the gate level testability measure and the functional level testability measure.

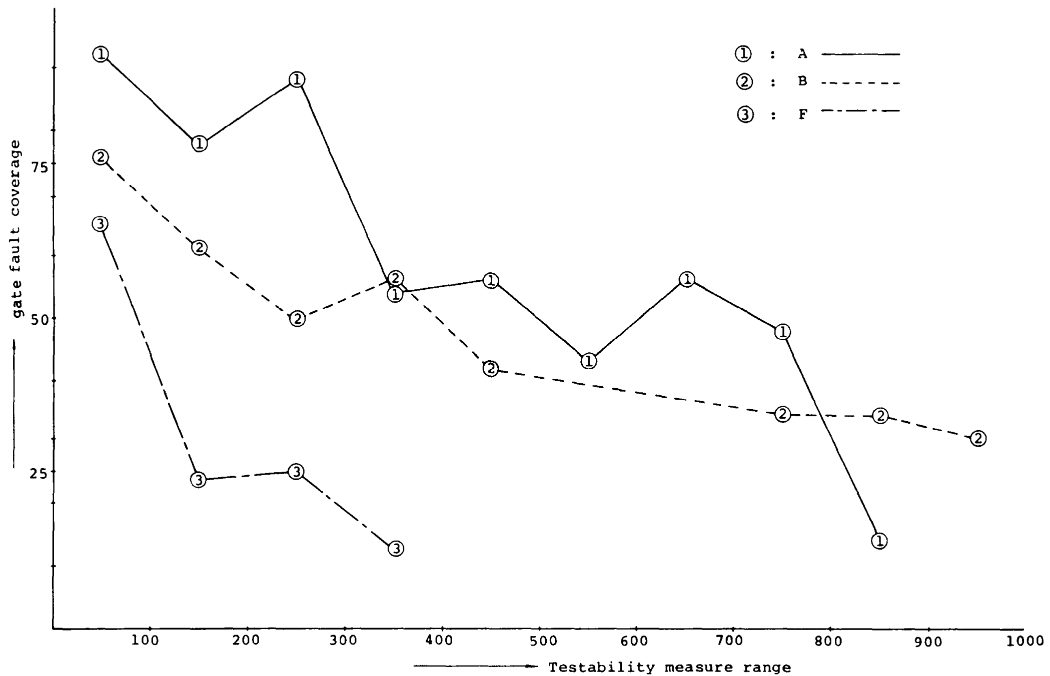


Fig. 10 Relationships between the gate fault coverage and the testability measure range.

Therefore, the proposed measure can be applicable to both the gate and the functional level testability analyses. The functional level testability measure is a much clearer tendency, in terms of fault coverage in testability ranges, than the gate level testability measure. This implies that functional blocks are more suitable for testability bottleneck analysis. A few ups/downs in Fig. 10 can be considered as the difficulty in carrying out a gate level testability measure estimation. It is difficult to predict that a specific gate having a large testability measure value will always have bad fault coverage. A gate circuit, which has a large/small testability measure and good/bad coverage, may be shown, for example, in reconvergent networks.

However, if the gate/functional block fault coverage is considered in the testability range, as depicted in Fig. 9 and 10, it has a good correlation with testability measure ranges. This indicates the proposed testability measure's effectiveness.

5. Discussion

In this paper, the functional level testability measure was defined and evaluated in terms of the relationships between functional testability measure regions and block fault coverage. As a result, the proposed functional testability measure had a good correlation with the fault coverage in block testability ranges. That is, there was a tendency wherein, the larger the functional block testability measure, the worse its fault coverage. Based on these results, the proposed testability measure can be applicable to the Design for Testability.

Application: Design for Testability

If designers can find hard-to-test blocks after detail logic designs, they can easily apply these testability analysis results to their design modifications. Hard-to-test parts detection is accomplished as an application of the proposed functional level testability measure.

Fig. 11 shows an easily testable network design process. The first step is a detail logic design for a network.

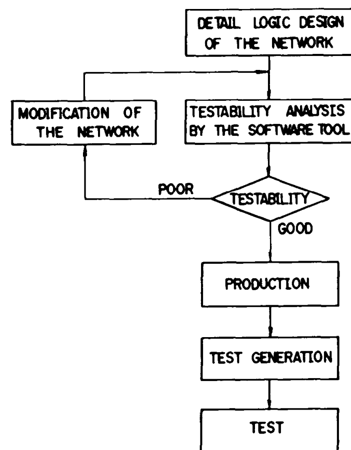


Fig. 11 Easily testable network design process.

Second is a testability analysis using a software tool (NETMAP). Third is a testability evaluation, that is, determining whether or not the network is easy to test. The proposed testability measure is used to identify testability bottleneck blocks in this stage. The testability measure boundary indicating testing ease/difficulty depends on test generation algorithms and execution time. From the experimental results, as an indication, blocks for which the testability measure is more than 300 imply hard-to-test parts, i,e., testing difficulty for parts under the condition of the D-algorithm ATPG program execution for two hours. A more appropriate measure boundary depends on future research. If the testability is good, proceed to production, test generation and the final test. These are the same as conventional test methods. If the testability is poor, it is necessary for logic designers to modify the network and repeat the same process.

It is clear that modification just after the detail logic design stage is mor useful and cheaper than that after test generation. Sometines, it is quite difficult to modify logic designs after assembling devices/packages. Furthermore, scan path flip-flops/test points implementation, based on the testability measure, is one useful technique for reducing testing difficulty [16].

Fig. 12 shows an application of the proposed method. In this example, 74LS163 was hard to test. On the other hand, 74LS251 was easy to test. Therefore, it is necessary to modify the logic design including 74LS163. In this network, each IC is a functional block.

Many large networks have been produced due to improved circuit technology and user demands. On the other hand, as logic networks become larger, the automatic test pattern generation becomes more and more difficult. In these environments, it is difficult to spend enough computer resources for each network test generation. However, highly accurate fault coverage is necessary to guarantee design qualifications. Therefore, it is important to pinpoint testability bottleneck blocks under the restricted condition.

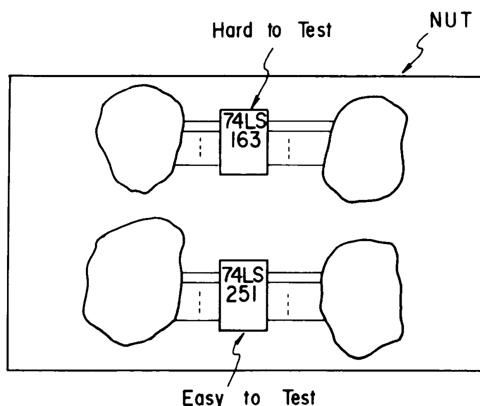


Fig. 12 An application of the proposed method.

Useful testability analysis tools must meet two feasibility requirements.:

1) Accurate and consistent test generation difficulty correspondence,
2) Use of much less computer resources than test generation.

The proposed testability measure can satisfy these requirements.

6. Conclusion

The controllability/observability measures and testability measure at the functional level have been defined. An effective calculation method for the proposed measures has been presented. Experimental results indicate that the functional block testability measure closely relates to the fault coverage for each block, so it can be used as an effective guideline for improving the testability of a given network in the early stages of design.

Acknowledgement

The authors would like to thank Mr. Kitamura, Dr, Saito, Mr. Murano, Dr. Yamada and Mr. Kuwata for their helpful guidance and recommendations. The authors would also like to thank their colleagues for providing many valuable suggestions and comments.

References

1. AKERS, S. B. Test Generation Techniques, *Computer* (March 1980), 9-15.
2. WILLIAMS, T. W. and PARKER, K. P. Design for Testability—A Survey, *IEEE Trans. Comput.* **C-31**, (Jan. 1982), 2-15.
3. HAYES, J. P. and MCCLSKEY, E. J. Testability Considerations in Microprocessor—Based Design, *Computer* (March 1980), 17-26.
4. GOLDSTEIN, L. H. and THIGPEN, E. L. SCOAP: SANDIA Controllability/Observability Analysis Program, *Proc. 17th Des. Automat. Conf.* (June 1980), 190-196.
5. GOLDSTEIN, L. H. Controllability/Observability Analysis of Digital Circuit, *IEEE Trans. Circuits and Sys.* **CAS-26**, (Sep. 1979), 685-693.
6. STEPHENSON, J. E. and GRASON, J. A Testability Measure for Register Transfer Level Digital Circuits, *Proc. 6th Fault-Tolerant Comput. Sysmp.* (June 1976), 101-107.
7. GRASON, J. TMEAS: A Testability Measurement Program, *Proc, 16th Des. Automat. Conf.,* (June 1979), 156-161.
8. KOVIJANIC, P. G. Testability Analysis, *Dig IEEE 1979 Test Conf. (Nov. 1979), 310-316.*
9. KOVIJANIC, P. G. Single Testability Figure of Merit, *Dig. IEEE 1981 Test Conf.* (Oct. 1981), 521-529.
10. BENNETTS, R. G. MAUNDER, C. M. and ROBINSON, G. D. CAMELOT: A Computer Aided Measure of Logic Testability, *Proc. IEEE Int. Conf. on Circuits and Computers, (1980), 1162-1165.*
11. TAKASAKI, S. KAWAI, M. FUNATSU, S. and YAMADA, A. A Calculus of Testability Measure at the Functional Level, *Dig. IEEE 1981 Test Conf.* (Oct. 1981) 95-101.
12. TAKASAKI, S. KAWAI, M. FUNATSU, S. and YAMADA, A. Testability Measure Analysis at the Functional Level, Presented at 5th Annual IEEE Design for Testability Workship, (April 1982).
13. FUNATSU, S. WAKATSUKI, N. and YAMADA, A. Easily Testable Design of Large Digital Circuits, *NEC Res. Develop.* (July 1979), 49-55.
14. AGRAWAL, V. D. and MERCER, M. R. Testability Measures—what do they tell us?, *Dig. IEEE Test Conf.* (Oct. 1982), 391-396.
15. RATIU, I. M. VINCENTELLI, A. S. and PEDERSON, D. O. VICTOR: A Fast VLSI Testability Analysis Program, *Dig. IEEE Test*

Conf. (Oct. 1982), 397-401.
16. TRISCHLER, E. Testability Analysis and Incomplete Scan Path, *Dig. ICCAD83* (Sep. 1983), 38-39.

Appendix

The primitive AND and OR gate equations can be expressed as follows:
AND

$$C^0(O_1)=\text{Min}\,\{(C^0(I_1),\, C^0(I_2),\, \ldots,\, C^0(I_n)\}+a$$

$$C^1(O_1)=\sum_{i=1}^{n} C^1(I_i)+a$$

$$\underset{1\leqq j\leqq n}{CO(I_j)}=\sum_{\substack{i=1\\ i\neq j}}^{n} C^1(I_i)+CO(O_1)+a$$

OR

$$C^0(O_1)=\sum_{i=1}^{n} C^0(I_i)+a$$

$$C^1(O_1)=\text{Min}\,\{(C^1(I_1),\, C^1(I_2),\, \ldots,\, C^1(I_n)\}+a$$

$$\underset{1\leqq j\leqq n}{CO(I_j)}=\sum_{\substack{i=1\\ i\neq j}}^{n} C^0(I_i)+CO(O_1)+a$$

where
a is a measure of circuit complexity ex. $a=1$.

(Received July 2, 1984; revised Octorber 18, 1984)