초고집적 반도체기술(차세대기억소자) 공동개발사업
년차 연구보고서
(1 차년도)

Built in Self Test가 용이한 CMOS 회로 설계

Design of CMOS Circuits for Built in Self Test

연구수행기관 : 서울시립대학교 공대 전자공학과

한국전자통신연구소

제 출 문

한국전자통신연구소장 귀하

본 보고서를 초고집적 반도체기술(차세대 기억소자) 공동개발사업 Built in Self Test가 용이한 CMOS회로 설계에 관안 과제의 (1)차 년도 년차 연구보고서로 제출합니다.

1990 년 3 월 15일

연구수행기관 : 서울시립대학교
과제 책임자 : 노 승 용
참여 연구원 : 홍 완 희 이 양 희
정 원 섭 김 희 식
김 영 우 김 영 호
한 기 주 박 기 현
민 병 현

요 약 문

1. 제 목

초고집적 반도체기술 공동개발사업
Built in Self Test가 용이한 CMOS 회로 설계의 개발 과제

2. 연구개발의 목적 및 중요성

가. 국내외의 기술개발 현황

지난 15년 동안 Test분야의 BIST(Built in Self Test)및 Scan Design기술은 어느 정도 확립되었으나 고집적화에 따른 Testability를 개선하기 위해 앞으로 계속 발전될 전망이다. 그리고 최근 저전력 소자로 대표적인 CMOS회로의 기술이 계속 발전하고 있지만 이들의 Test방법도 용이하지 않다. 따라서 고집적화와 CMOS 특유의 고장현상에 따른 Test 문제를 해결하기 위해, 각 국의 반도체 생산회사의 연구소 및 대학 연구소에서는 CMOS 회로의 Test 개념정립을 위한 연구를 현재 계속 전쟁 중이며, 국내의 반도체 생산회사들도 주문형 IC를 비롯한 고집적 메모리 소자를 개발함에 따라 이에 따르는 Test방법과 설계에 많은 관심과 연구를 기울이고 있다.

나. 연구개발의 목적

본 연구의 목적은 대규모 CMOS기억소자를 용이하게 테스트하는 새로운 Built In Self Test 방식을 개발하기 위한 그의 첫 단계로 CMOS의 Stuck open 고장검출을 가능하게 하는 테스트 패턴 발생방법을 연구한다.

다. 연구개발의 중요성

VLSI 기술의 발전과 더불어 설계와 제작기술상의 새로운 문제점들이 나타나고 있으며, 이런 문제점들을 해결하기 위한 보다 향상된 테스트 또는 새로운 개념의 테스트 장비들의 요구가 끊임없이 발생하고 있다. 또한 저전력 소자로 대표적인 CMOS회로의 기술이 계속 발전하고 있지만 이들의 Test방법도 용이하지 않다. 따라서 CMOS 특유의 고장현상에 대한 Test 문제는 시급히 해결해야할 중요한 문제이다.

3. 연구개발의 내용 및 범위

본 연구의 진행은 4 단계로 구분하여 다음과 같이 년차별로 달성한다.

가. 1 차 년도는 CMOS 회로의 Test를 위한 새로운 Test Pattern 발생기 설계와 Signature Analysis를 개발하고, NLFSR과 LFSR로 구성한 BILBO를 설계한다.

나. 2 차 년도는 CMOS 회로의 Test를 위한 TPG와 PSA를 설계하고, Sequential 논리회로의 상태 벡터 특성을 연구한다. 또한 Test Simulation이 가능한 CAD Tool도 개발한다.

다. 3 차 년도는 Sequential CMOS Logic 회로의 Test를 위한 TPG의 설계와 PAL, PEEL 소자로 BILBO를 개발한다. 또한 Convolution Code를 이용한 새로운 Built In Self Test방식을 연구한다.

라. 4 차 년도는 Built-In Self Test 방식을 채용한 CMOS Memory의 설계와 최종 Mask 제작을 한다.

4. 연구개발 결과 및 활용에 대한 건의

가. 1 차년도 연구개발 결과

(1) CMOS의 Stuck open 고장을 검출하기 위해 경로 활성화 방법을 이용하여 최소의 테스트 패턴을 구하였다.

(2) 다기능 BILBO를 설계하여 비선형적 (NLFSR)으로는 테스트 패턴을 생성하고 선형적 (LFSR)으로는 signature analysis를 할 수 있도록 하였다.
그 결과, 최소의 하드웨어를 가지고 매우 적은 테스트 패턴을 발생시킬 수 있었다.

(3) 다기능 BILBO의 입력과 출력은 3-State 버퍼를 이용하여 동일 선로에서 처리하였다.. 그 결과, BILBO의 외부 배선 수는 감소되어 chip 배선을 용이하게 하였다.

(4) 개발된 다기능 BILBO를 PLA회로에 적용한 결과, 정상동작시 종래의 단점인 데이타 신호의 전파지연을 일으키지 않도록 하였으며 테스트 동작시 테스트 피턴 시이퀀스는 종래의 pr-TPG보다 조금 감소되었다.

나. 연구개발의 활용에 대한 건의

반도체의 품질 향상은 근본적으로 막대한 설비투자를 필요로 하고 있다. 지난 4-5년 동안 국내 반도체 생산업체가 품질향상을 위해 투하한 액수만도 수천억 원에 육박하고 있으며 생후 3-4년 간 새로운 VLSI 소자의 개발 및 생산성을 위하여 또 다른 수천억 상당의 투하를 예상할 수 있다. 이는 VLSI 의 기술발전과 더불어 항상 설계와 제작기술상의 새로운 문제점들이 나타나고 있기 때문이다. 따라서 제품의 테스트 비용은 그 제조비용보다 증가하고 있으며 향상된 테스트 방식은 주로 Test시간, Test 부가회로의 실현성 및 높은 고장검출 등에 의존하고 Test 방식의 특징에 따라 결정되어진다. 따라서 차세대 기억소자를 개발하기 위한 본 연구의 결과는 다음과 같은 이점을 갖으면서 활용된다.

(1) 높은 고장검출을 갖는 Built In Self Test방식을 CMOS회로에 활용가능
(2) 부가회로의 축소로 인한 Chip의 배치설계가 간단하다.
(3) Test Pattern의 축소로 인한 Test 실행속도의 향상과 제품비용 절감.

5. 기대 효과

미국과 일본의 반도체 생산회사들이 Memory 및 주문형 IC 내에 BIST 방식을 이용하여 VLSI 고장검출을 용이하게 할 수 있는 논리설계 방식을 일부 채용하고 있지만 이의 기술을 공개하지 않고 지적보호를 하고 있으며, 특히 CMOS회로의 Test를 위한 설계는 미국을 비롯 일본과 서독에서 인재 활발하게 연구개발이 진행 중이다. 따라서 새로운 기술의 도입은 불가능하므로 국내에서도 Test분야의 기술축적은 물론 이와 같은 설계기술이 이식된 CMOS회로에 의한 기억소자의 생산이 가능하도록 이론과 실험적 자료를 우리 스스로 구하는데 기대 된다.

SUMMARY

Ⅰ. Subject

Design of CMOS Circuits for Built in Self Test

Ⅱ. Object and Importance of the Project

The technology, of BIST and scan design has been developed and stabilized to a certain degree for last 15 years. However, the technology is expected to make continuous advance to improve the testability of VLSI. The technology of CMOS circuits, typical components of low power dissipation, continues to develop recently and the test method of the devices is not supposed to be easy.

Accordingly, each country's semiconductor laboratories and universities research centers are in the middle of making extensive research for setting down the test concept for CMOS circuits. Domestic semiconductor companies, also pay much attention to the research and development for the test method and circuit design to keep up with a fast development of VLSI memory devices and custom IC.

(a) the Purpose of the Research

The purpose of the research is to develop a new Built In Self Test method to testify large scaled CMOS memory devices with ease. A research for test pattern generator, which enables to detect the CMOS stuck open trouble.

(b) The Importance of the Research

With an advance of VLSI technology, nev problems in terms of design and fabrication technique continues to emerge. In this regard advanced test method to solve such problems and test equipments of nev concept are required in accordance with the advancement of VLSI technology. It is noted that the test method for typical CMOS circuits as low power dissipation components is not so easy even if the CMOS technology developes in a stead3' and fast fashion. Therefore. test method for unique troubles of CMOS is an important thing to be solved urgently.

Ⅲ. The Contents and Scope of the Research

The procedure of the research will be divided into four steps as the following:

(a) In first year, we develop the design of the nev test pattern generator for CMOS circuit test and the signature analysis. We also design a BILBO composed of NLFSR and LFSR.

(b) Second year, we design TPG and PSA for the test of CMOS circuits and perform a research on the status vector characteristic of sequential logic circuits. And CAD tool for test simulation will be developed.

(c) Third year, we design TPG for a test of sequential CMOS logic circuits and develop BILBO with PAL and PEEL devices. We also make a research, an nev Built In Self Test based on convolution codes.

(d) Fourth year, we fabricate final mask pattern and chip with the design of CMOS memory employ Built In Self Test metod.

CONTENTS

목 차

제 1 장 서 론

반도체 공정기술이 급속히 발전함에 따라 고밀도 집적회로의 설계기술도 종래 방식과는 다르게 전개되어 발전되고 있으며, 이에 따른 VLSI 테스트 문제는 심각하게 대두되고 있다. 또한 집적도가 증가함에 따라 과다한 전력소비는 VLSI 설계를 어렵게 하고 있는데 이의 대응책으로 비교적 낮은 전력소비와 늪은 집적도의 특성을 갖는 CMOS가 VLSI기술의 중요한 구성소자로 사용되고 있다. 그러나 CMOS를 이용한 논리회로는 Stuck-at 고장과 같은 고전적인 고장모델 외에도 회로의 특성에 따라 Stuck-open과 Stuck-on 이라는 고장모델이 존재한다.

CMOS 회로에서 Stuck-open 고장이 발생할 경우, 회로의 출력은 charge 축적기능에 의하여 바로 전 상태 값을 유지하게 되므로 조합논리회로가 순서논리회로와 같이 동작한다. 이와 같은 고장은 단일 테스트 패턴으로는 검출 할 수 없다. CMOS의 Stuck-open 고장을 검출하기 위해서는 테스트 대상회로의 CMOS를 초기화하는 테스트 패턴 입력과 이 소자를 테스트하는 테스트 패턴 입력으로 구성된 테스트 벡터를 인가해야 한다. 따라서 일정한 순서를 갖는 두개의 연속적인 테스트 패턴들이 필요하게 된다.

본 연구에서는 CMOS 회로의 고장을 용이하게 검출할 수 있도록 Built In Self Test의 설계방식을 개발하였다. 초기화 패턴과 테스트 패턴을 연속적으로 발생하기 위하여 pattern merging 과 재배열 알고리즘을 사용하여 NLFSR(Nonlinear Feedback Shift. Register)를 설계하고, 이것과 함께 다기능 BILBO를 구성함으로써 다기능 BILBO의 장점을 이용하여 Built-In Self Test가 용이하도록 하였다. 또한 규칙적인 구조를 갖는 PLA에 Built-In Self Test방식을 적용하기 위하여 LFSR과 NLFSR을 이용한 다기능 BILBO를 부가하여 Built-In Self Test가 용이하도록 설계하였다.

제 2 장 CMOS 회로의 Stuck-Open 고장

CMOS 회로는 s-a-1 (Stuck-at-1), s-a-0 (Stuck-at-0)와 같은 고전적인 고장과 Stuck-open과 같은 CMOS 특유의 고장이 있다. CMOS 회로에서 "Stuck-open" 고장은 부하회로 혹은 구동회로의 고장으로 인하여 발생하는 것으로서 소자 출력 측에 높은 임피던스 상태를 말한다. 그림 1은 2 입력 CMOS NOR 회로를 나타냈다. 이 회로에서 Stuck-open고장들은 NOR 회로의 각 입력 a, b와 연결된 NMOS 소자들의 개방 또는 missing에 의한 Stuck-open 고장, 그리고 NOR 회로의 공급전원(VDD)의 open에 의한 Stuck-open 고장들이 있다. 표 1은 2입력 CMOS NOR 회로에서 고장 및 고장이 없는 상태에 대한 진리표이다. 여기서 f(0), f(1)은 각 각 NOR 회로의 출력단위 s-a-0, s-a-1고장, f(a), f(b)는 NOR회로의 각 입력 a 및 b의 s-a-0고장, f(a)sop, f(b)sop는 NOR회로의 각 입력 a 및 b와 연결된 NMOS 소자들의 Stuck-open 고장, f(VDD)sop은 NOR회로의 VDD에서의 Stuck-open 고장을 나타냈다. "Qn"는 Stuck-open 고장이 발생하였을 경우 출력이 바로 전 상태의 값을 유지함을 나타내는 것으로서 Stuck-open 고장이 발생하면 조합논리회로가 순서논리회로와 같이 동작하는 것을 의미한다.

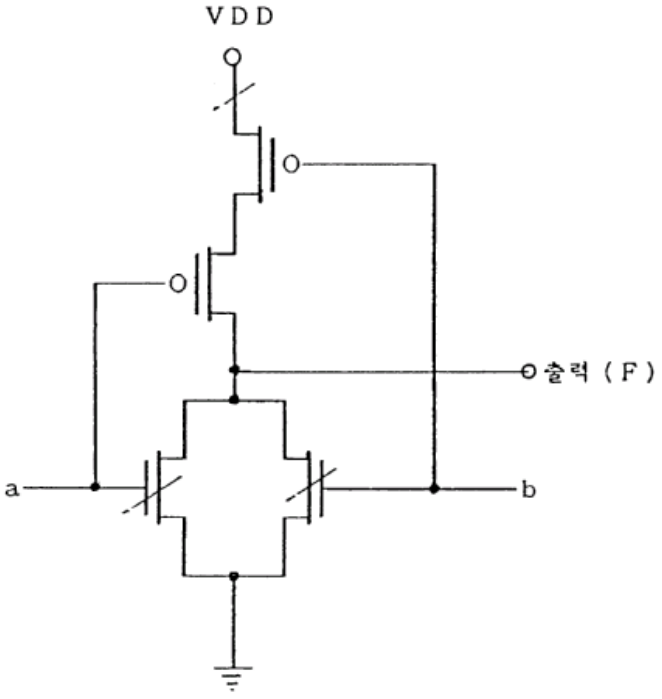


그림 1. 2입력 CMOS NOR 게이트

표 1. 2 입력 CMOS NOR 회로의 진리표

입력		출력	Stuck at 고장				Stuck Open 고장		
A	B	f	f(0)	f(1)	f(a)	f(b)	f(a)sop	f(b)sop	f(VDD)sop
0	0	1	0	1	1	1	1	1	Qn
0	1	0	0	1	0	1	0	Qn	0
1	0	0	0	1	1	0	Qn	0	0
1	1	0	0	1	0	0	0	0	0

따라서 2 입력 NOR회로의 고장을 검출하기 위하여 표 1의 입력을 순서대로 인가하면 NOR 회로에서의 f(a)sop와 f(VDD)sop 고장은 검출할 수 없다. 즉,NOR 회로에서 f(a)sop 고장이 발생할 경우 입력패턴을 "10"로 인가하면 출력은 바로 전 상태의 값인 "0"을 유지하게 되며, 고장이 없는 경우에도 "10"을 인가하면 출력은 "0"이 되므로 f(a)sop은 검출할 수 없게 된다. 그러므로 이와 같은 고장들을 검출하기 위해서는 일정한 순서를 갖는 두 개의 연속적인 테스트 패턴 즉, 초기화 패턴과 테스트 패턴(T)이 인가되어야 한다.

제 3 장 경로활성화에 의한 테스트 패턴 생성

제 1 절 CMOS의 Stuck-open 고장검출

게이트 레벨에서 고전적 고장인 Stuck-at고장을 검출하는 방법으로는 D-알고리즘, PODEH (Path Oriented Decision Making)등이 있다. Stuck-at 고장을 검출하기 위한 위 방법들은 검출하려는 고장의 장소를 지나는 경로, 즉 primary input에서 primary out put까지의 경로를 활성화(sensitizing)시키는 방법을 이용한다. 경로를 활성화시키기 위해서는 활성화시키고자 하는 경로 이와의 노드 중 경로 활성화에 관계하는 노드들의 값이 규정되어야 하며. 만일 경로가 활성화 월 수 있다면 그 경로 내의 모든 Stuck-at 고장은 검출할 수 있다. 그러나 CMOS 회로내의 여러 장소에 다중 Stuck-Open 고장이 존재하는 경우, 두 개의 테스트 패턴으로는 이 고장을 검출할 수 없다. 다중 Stuck-Open 고장을 검출하기 위해서는 3개 이상의 테스트 패턴이 필요하다. 본 연구에서 개발한 방법은 보다 효율적으로 다중 Stuck-Open 고장을 검출할 수 있는 테스트 패턴을 구하였다. 또한 테스트 패턴을 구하기 위해서는 테스트 대상회로 내의 모든 경로가 활성화 될 수 있어야 하지만 플립플롭과 같이 Feedback, Fan-Out 등이 존재하는 순서회로는 활성화시킬 수 있는 경로를 갖고 있으므로 이들 경로를 활성화시킬 수 있도록 설계되어야 한다. 본 연구에서는 경로 활성화를 위해서 정상 모드에서는 정상 Feedback 신호나 Fan-Out 신호를 통과시키고, 테스트 모드에서는 테스트 입력을 통과시키는 부가회로를 CMOS 소자로 설계하였다. 또한 완전한 테스트를 위해 사용된 부가회로도 테스트되어지는데 부가회로의 테스트는 함수적 레벨에서 수행하였다. 그림 2의 2입력 CMOS NAND 회로에서 PMOS 소자의 초기화 입력은 그 단의 출력을 논리 0로 하는 입력이고, 테스트 입력은 그 단의 출력을 논리 1로 하는 입력이다. 또한 NMOS 소자의 초기화 입력은 그 단의 출력을 논리 1로 하는 입력이고, 테스트 입력은 r 단의 축력을 0로 하는 입력이다.

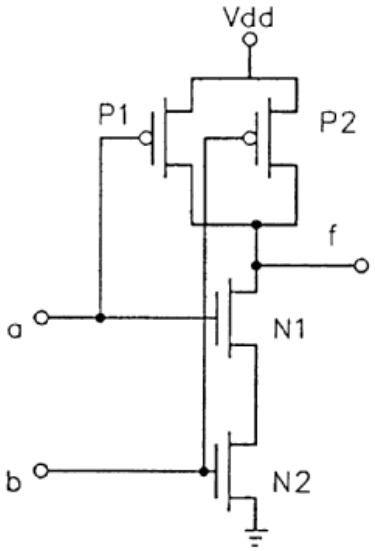


그림 2. 2입력 CMOS NAND 게이트

이 때 테스트 대상 소자에 직접 연결되지 않은 다른 입력들은 테스트를 방해하지 않는 규정된 값을 가져야 하는데 NAND 게이트에서는 논리값 "1"을 가져야 한다.
표 2에 그림 2의 모든 소자를 테스트할 수 있는 테스트 패턴을 보였다. 표 2에서 2입력 CMOS NAND 게이트의 모든 단일 Stuck-Open고장을 검출하기 위해서는 8개의 테스트 입력이 필요함을 알 수 있다. 그러나 한 개의 경로 상에 있는 PMOS 소자의 NMOS 소자의 테스트를 동시에 수행한다는 관점에서 고찰해보면 이 테스트 패턴 수는 줄일 수 있다. 그림 2의 회로에서 P1을 테스트하기 위한 입력 <a, b>=<0, 1>은 N1의 초기화 입력이다. 따라서 primary input-a 에서 primary output-f 에 이르는 경로 상에 <1, 1>, <0, 1>, <1, 1>로 이어지는 3개의 테스트 벡터를 인가하면 P1과 N1의 Stuck-Open고장을 검출할 수 있다. 같은 방식으로 Primary input-b에서 Primary output-f까지의 경로 상에 <1, 1>, <1, 0>, <1, 1>로 이어지는 3개의 테스트 벡터를 적용하면 P2와 N2의 Stuck-Open 고장을 검출 할 수 있다.

표 2. 2 입력 CMOS NAND 게이트의 Stuck open 고장시 테스트 패턴

테스트 대상	초기화 입력		테스트 입력	
소 자	a	b	a	b
P1	1	1	0	1
N1	0	1	1	1
P2	1	1	1	0
N2	1	0	1	1

이 때 처음 경로에 대한 마지막 패턴은 두 번째 경로상 의 처음 패턴과 같으므로 그림 2의 모든 Stuck-0Pen 고장을 검출하는 데는 <1, 1>, <0, 1>, <1 ,1>, <1, 0>, <1, 1>로 이어지는 5개의 입력 패턴만이 필요하다. 즉, 정리 1과 정리 2가 성립한다.

(정리 1) 만일 Primary input에서 Primary output까지 하나의 경로가 활성화되고, 이 경로상의 Primary input에서 1, 0, 1의 신호가 연속해서 인가된다면 그 경로 상의 모든 Stuck-Open 고장은 이 경로상의 Primary 출력을 관측함으로서 검출된다.

<증명> 활성화된 경로의 각 NAND 게이트 단을 Primary input에서 Primary output까지 순서대로 G1, G2, G3, ----, Gn이라고 하면 Primary input에 인가되는 신호는 하나의 NAND 게이트 단을 지나면서 inversion된다. 또한 PMOS 소자에 대한 초기화 입력과 NMOS 소자에 대한 초기화 입력은 서로 반대의 논리값을 가지며, PMOS 소자라 NMOS 소자에 대한 테스트 벡터도 서로 반대의 논리값을 가지므로 한 stage의 PNOS소자의 테스트 입력은 그 다음 stage의 NNOS 소자에 대한 초기화 입력이 된다. 즉, "$Gn-_1$ " 단의 PNOS 소자를 테스트하기 위한 입력은 "$Gn-_1$ " 단의 NMOS소자를 초기화하며, "$Gn-_1$ " 단의 NHOS 소자를 테스트하기 위한 입력은 Gn단의 PNOS 소자를 초기화한다. 또한 Primary input에서 인가되는 신호는 계속 변하므로 Primary output에서의 논리값을 관측하면 그 경로내의 Stuck-Open 고장의 존재여부를 알 수 있다.

(정리 2) 만약 Primary input에서 Primary output까지 하나의 경로가 활성화되고, 이 경로상의 Primary input 에서 1, 0, 1의 신호가 연속해서 인가된다면 그 경로상의 모든 Stuck-Open고장은 다중 Stuck-Open 고장의 존재여부에 관계없이 경로상의 Primary output를 관측함으로서 검출된다.

제 2 절 multiple Stuck-Open고장의 영향

Multiple Stuck-Open고장의 영향을 나타내기 위해 그림 3에 출력한 수 f3=ab+ ac인 CMOS 회로를 보였다. 또한 Stuck-Open고장이 발생한 트랜지스터를 점선으로 나타내었다. 그림 3의 출력 f1을 갖는 게이트에서 NMOS N1의 고장을 검출하기 위한 2개의 테스트 패턴 <010, 110>과 입력 a에 연결된 PMOS P1의 고장을 검출하기 위해 2개의 테스트 패턴 <110, 010>을 primary 입력단에 인가하였을 때, 각 게이트의 출력 논리값을 표 3에 나타내었다. 표 3에서, 관측 가능한 primary출력 f3의 논리값이 고장회로와 정상회로가 모두 같음을 알 수 있다.

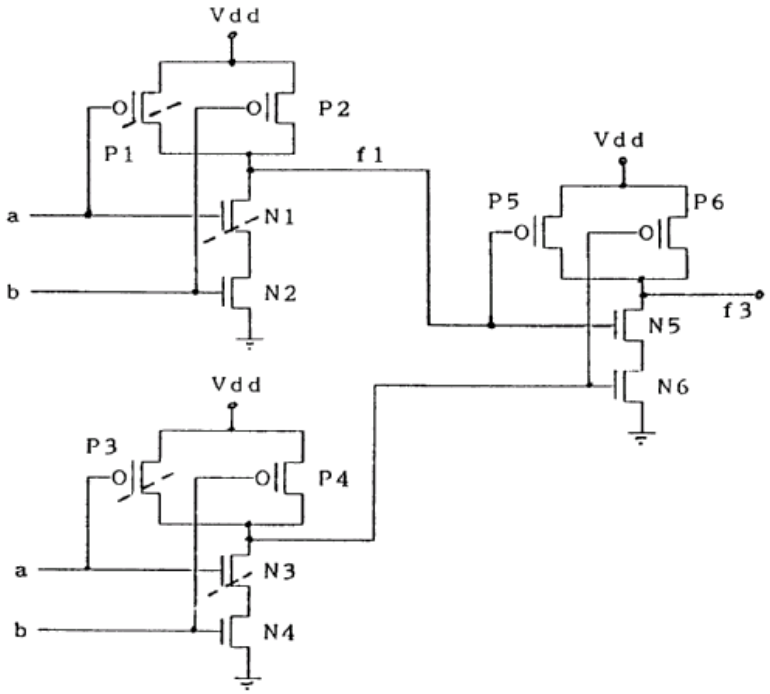


그림 3. 다중 Stuck-open 고장이 있는 CMOS 조합회로

표 3. 그림 3에서 Sensitize된 경로의 Stuck-open 고장을 검출하기 위한 테스트의 집합

Input			Fault-free			Faulty		
a	b	c	f_1	f_2	f_3	f_1	f_2	f_3
0	1	0	1	1	0	1	1	0
1	1	0	0	1	1	0	1	1
1	1	0	0	1	1	1	0	1
0	1	0	1	1	0	1	1	0

따라서 2개의 테스트 패턴으로는 다중 Stuck-Open고장을 검출할 수 없다. 이것은 다른 고장들에 의해서 검출하려는 고장이 Masking 되어지기 때문이다. 그러나 경로 활성화가 이루어진다면 sensitize된 경로의 primary입력의 논리값을 두 번 변하게 앞으로서 제한된 알고리즘으로 다른 모든 Stuck-Open고장을 검출 할 수 있다. 이 알고리즘은 다음과 같다.

- 알고리즘 -

(단계 1) primary input에서 primary output까지 path를 Sensitize한다.

(단계 2) Path sensitizing에 있는 첫째 단 PNOS의 초기화를 위해 초기화입력 "1"을 인가한다.
이 때, primary output의 논리값은 $f_{1A}(1)$이라 한다.

(단계 3) PNOS의 테스트를 위한 입력 "0"을 인가한다.
이 때, Primary output의 논리값은 $f_{1B}(0)$ 라고 한다.

(단계 4) $f_{1A}(1)$와 $f_{1B}(0)$의 논리값을 서로 비교한다.
이 때, $f_{1A}(1)$와 $f_{1B}(0)$의 값이 서로 다르면 이 경로상의 PNOS와 NNOS로 반복되는 CMOS들의 Stuck-Open고장은 없다. 또한 $f_{1A}(1)$와 $f_{1B}(0)$의 값이 서로 같으면 이 경로 상에는 적어도 하나 이상의 Stuck-Open 고장이 존재한다.

(단계 5) "단계 4"에서 $f_{1A}(1)$와 $f_{1B}(0)$의 값이 서로 다르면 경로 상에 있는 첫째 단 NMOS의 고장점술을 위한 테스트 입력 "1"을 인가하고 이 때의 출력값을 f1c(1)라고 한다.

(단계 6) $f_{1B}(0)$와 f1c(1)의 논리값을 비교한다.
이 때, $f_{1B}(0)$와 f1c(1)의 값이 서로 다르면 이 경로 상의 NNOS와 PNOS로 반복되는 CMOS들의 Stuck-Open고장은 없다. 또한 $f_{1B}(0)$와 f1c(1)의 값이 서로 같으면 이 경로 상에는 적어도 하나 이상의 Stuck-Open 고장이 존재한다.

(단계 7) "단계 1"에서 "단계 6"까지의 과정을 모든 경로 상에 적용시킨다.

이 방법은 하나의 경로를 활성화시킨 다음 경로 상에 있는 하나의 primary input만을 변화시킴으로 테스트 과정이 매우 간편하다. 그림 3에서 점선으로 표시된 트랜지스터가 Stuck-Open고장이라고 가정할 경우, 즉 다중 Stuck-Open 고장이 존재함에도 불구하고 굵은 선으로 표시된 경로를 테스트할 때, 위 알고리즘을 이용하여 그 경로상의 Stuck-Open 고장이 검출 될 수 있음을 보였다. 여기서 출력 f1을 갖는 게이트에서 테스트 입력이 인가되기 전 상태를 X1이라 하고. 출력 f2을 갖는 게이트에서 테스트 입력이 인가되기 전 상태를 Y1이라고 하면 표 4와 같이 하나의 경로를 테스트하기 전의 상태에 관계없이 관측 가능한 primary output f3에서의 논리값을 관찰함에 따라 회로내의 고장 유, 무를 용이하게 검출할 수 있다. 만약 그림 3의 입력단과 다르게 하나의 활성화된 경로를 테스트하기 위한 입력이 경로의의 다른 입력 게이트들에 영향을 미치지 않는다면, 그 경로를 활성화시키기 위해 다른 모든 게이트 입력들은 정해져 있으므로 관측되는 primary output은 고정된 값을 갖게 된다. 따라서 어떤 경우에는 언제나 활성화된 경로에 두개의 테스트 패턴만을 인가함으로서 고장 검출이 가능하다.

표 4. 그림 3에서 Multiple Stuck-Open 고장 검출

State	a	b	c	$f_{1A}(1)$	$f_{1B}(0)$	$f_{1C}(1)$
$X_1=1,\ Y_1=1$	1	1	0	1	1	1
	0	1	0	1	1	0
$X_1=0,\ Y_1=1$	1	1	0	0	1	1
	0	1	0	0	1	1
$X_1=1,\ Y_1=0$	1	1	0	1	0	1
	0	1	0	1	1	0
	1	1	0	1	1	0
$X_1=0,\ Y_1=0$	1	1	0	0	0	1
	0	1	0	0	1	1

제 4 장 CMOS회로의 Built-In Self Test

제 1 절 Built-In Self Test

Built-In Self Test방법의 기본형태는 그림 4와 같이 CUT(Circuit Under Test)입력 측과 출력 측에 테스트 패턴 생성과 테스트 응답 처리를 위한 부가회로를 각각 삽입시키는 것이다. 이 부가회로는 BILBO(Built-In Logic Block Observer)라고 하며, BILBO의 출력은 TPG (Test Pattern Generator)이고, BILBO입력은 Signature Analyzer이다. 즉, TPG는 테스트 패턴을 생성하여 CUT에 입력시키고, 그 응답을 압축하고 해석하는 기능은 Signature Analyzer에서 한다. 또한 테스트 분석 결과는 Signature Analyzer의 Signature Registor와 제어회로 그리고 비교기, 디코더로 구성된 부가회로에서 GO/NO GO형태로 표시한다.
CMOS회로의 Stuck-open 고장을 검출하기 위하여 초기화 패턴과 테스트 패턴은 연속적으로 인가되어야 하므로 확장된 길이를 갖는 레지스터를 사용한다.

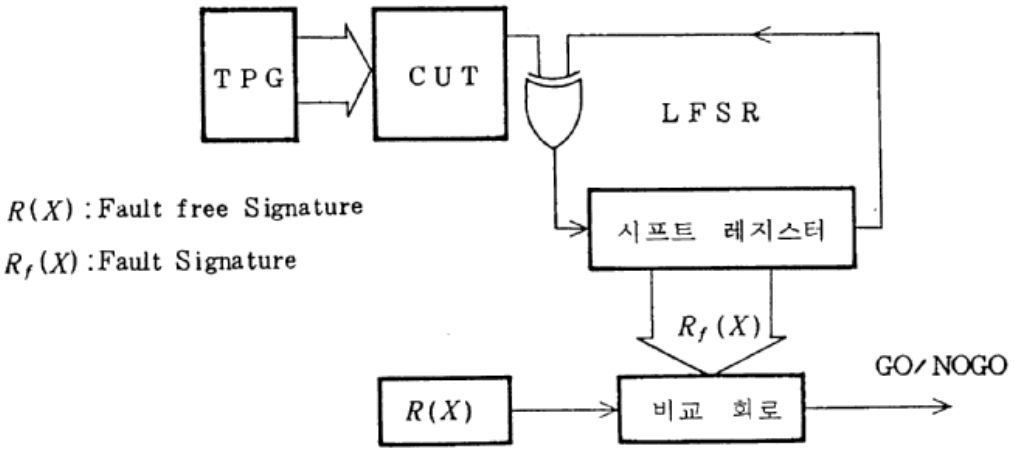


그림 4. Built In Self Test의 기본구조

그림 5에 각각 m비트인 초기화 패턴과 테스트 패턴을 결합하여 테스트 벡터 sequence를 발생할 수 있는 TPG의 구성을 나타냈다. 여기서 시프트 레지스터의 출력은 매 두 번째 플립플롭으로부터 나오므로 m개의 입력을 갖는 CMOS회로에 대한 TPG의 시프트 레지스터 길이는 n=2m이 된다. CMOS의 Stuck-open고장을 테스트하기 위해 3 장에서 서술한 경로 활성화 방법에 따라 얻은 테스트 패턴은 merging sequence로 변환시키고, 이 패턴의 발생을 위하여 CMOS회로에 대한 TPG를 설계한다. TPG의 설계는 다음과 같은 단계로 실행한다.

1. 초기화 패턴과 테스트 패턴들은 시프트 레지스터의 단일 벡터로 "pattern merging" 한다.
2. 시프트 레지스터의 상태 패턴들을 재배열한다.
3. 레지스터의 길이와 궤한 함수를 계산한다.

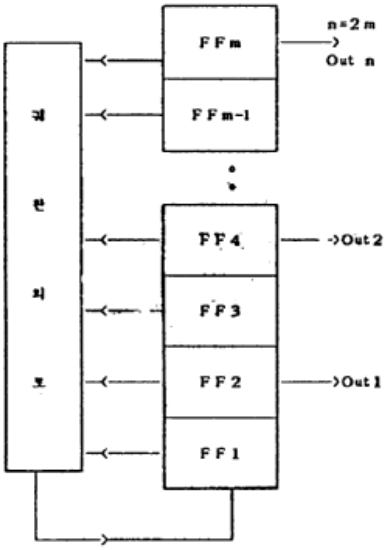


그림 5. CMOS회로의 Test를 위한 TPG 구성

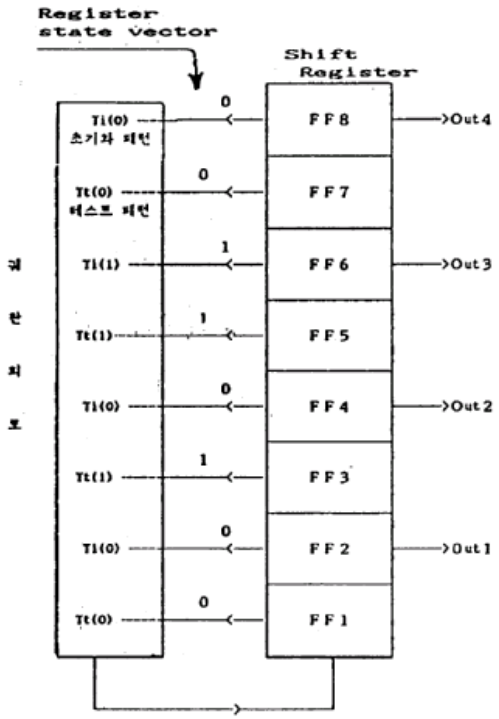


그림 6. Test Pattern의 Nerging

이와 같은 단계에 의하여 설계한 TPG는 시프트 레지스터의 매 클럭마다 초기화 패턴과 테스트 패턴을 연속적으로 생성된다.

제 2 절 테스트 패턴의 Merging과 재배열

일반적으로 조합회로의 테스트를 위해 Built-In Self Test 방식을 적용할 경우 임의의 순서로 테스트 패턴들을 인가하여도 고전적인 고장들은 검출할 수 있으나 CMOS회로의 Stuck-open고장은 시프트레지스터로부터 초기화 패턴과 테스트 패턴을 순서대로 얻어야 하므로 다음과 같은 정의를 도입한다.

(정의 1) 테스트 패턴과 초기화 패턴들을 각각 1비트씩 교대로 배열하여 하나의 패턴으로 작성하는 것을 "pattern merging"이라고 한다. 이 패 패턴의 길이는 두 배로 확장되며 배열 순서는 테스트 패턴의 비트를 먼저 배일하고 그 다음에 초기화패턴의 비트를 배열한다.

그림 6은 결합된 시이퀀스들의 pattern merging과정을 나타낸 것으로 Ti, Tt은 각각 초기화 패턴과 테스트 패턴이다. 이들 패턴들을 연속적으로 생성하기 위해서 두 개의 패턴들은 시프트 레지스터의 단일 상태벡터로 merging한다. Shift-in절차가 끝나면 초개화 패턴은 매 두 번째 플립플롭 즉, 레지스터 출력에 할당되고 한 번 더 시프트하면 테스트 패턴이 레지스터의 출력에 탈당되어 고장검출을 위한 결합된 시이퀀스가 연속적으로 생성된다. 이것은 연속된 두 클럭 내에 초기화 패턴 Ti와 테스트 패턴 Tt가 연속적으로 발생하여 Stuck-open 고장의 테스트를 가능하게 할 수 있음을 알 수 있다. pattern merging이 끝난 후 시프트 레지스터의 단일 벡터들은 그 순서를 바꾸어도 CMOS회로의 Stuck-Open고장을 검출할 수 있다. 따라서 시프트 레지스터 상태벡터들은 적절한 순서로 바꾸면서 NLFSR을 설계한다. 이것을 재배열이라고 한다. 즉, 시프트 레지스터의 궤한에 의해서 레지스터의 상태벡터가 바뀌므로 시프트 레지스터의 상태벡터들은 집합 Tm(최종 테스트 패턴들)이 재배열되는 것임을 의미한다. CMOS회로의 Stuck-open고장들을 테스트하기 위해 pattern merging한 벡터들의 집합 Tm을 생성하는 NLFSR의 설계는 다음과 같은 재배열 알고리즘 이용하였다.

- 테스트 패턴 집합 Tm의 재배열 알고리즘 -

(단계 1) 테스트 패턴 집합 Tm에서 임의의 테스트 패턴을 선택한다.

(단계 2) 선택된 테스트 패턴을 "sequence file T"에 전송하고 집합 Tm에서 제거한다.

(단계 3) 만약 Tm에 테스트 패턴이 공집합이면 (단계 12)로 간다.

(단계 4) 차수 K를 1로 정한다.

(단계 5) 테스트 패턴 집합 Tm에서 k차 계승자를 찾는다.

(단계 6) 만약 k차-계승자가 존재하면 "sequence file T"의 하위패턴 다음에 순서적으로 링크 패턴을 작성하고, 만약 존재하지 않으면 (단계 8)로 간다.

(단계 7) (단계 6)에서 검출된 계승자를 Tm에서 제거하고 (단계 3)으로 간다.

(단계 8) 테스트 패턴 집합 Tm에서 k차-선임자를 찾는다.

(단계 9) 만약 k차-선임자가 존재하면 "sequence file T"의 상위패턴 앞에 순서적으로 링크 패턴을 작성하고, 만약 존재하지 않으면 (단계11)로 간다.

(단계10) (단계 9)에서 검출된 선임자를 Tm에서 제거하고 (단계 3)으로 간다.

(단계11) 차수 k를 1씩 증가시키고 (단계 5)로 간다.

(단계12) 종 료

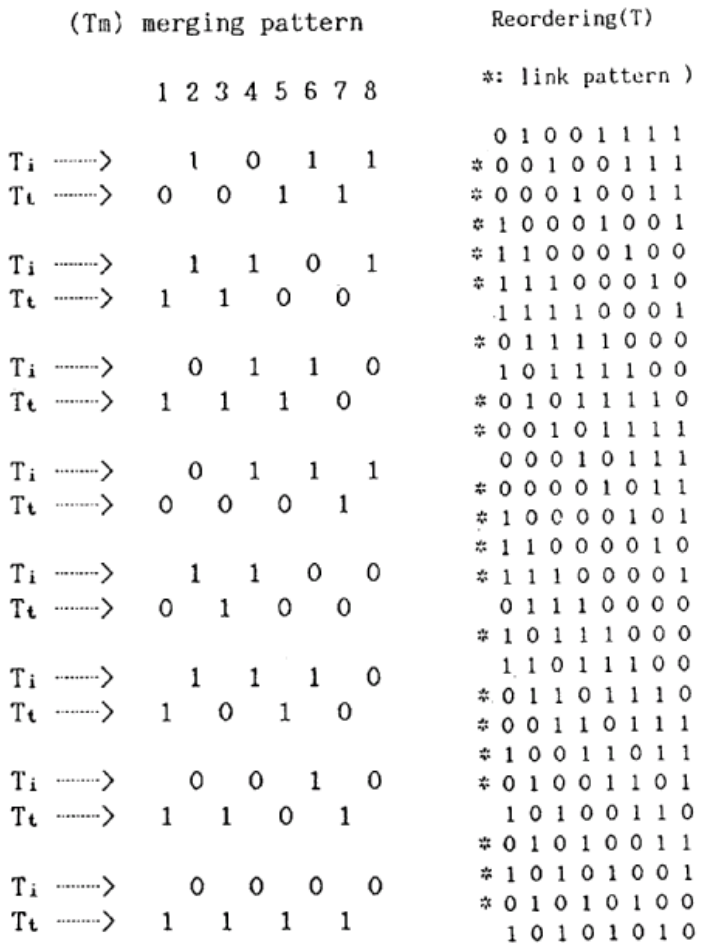


그림 7. Test pattern Merging과 재배열에 의한 단일 상태벡터의 생성과정

CMOS회로의 Stuck-open고장들을 테스트하기 위해 pattern merging한 벡터들의 집합 Tm을 그림 7에 나타내었다. 시프트 레지스터 길이의 최소화는 초기화 패턴(Ti)과 테스트 패턴(Tt)을 pattern merging한 상태 벡터 즉, 테스트 패턴 집합(Tm)들의 길이에 의하여 결정된다. 이 때 모든 상태 벡터들은 서로 다른 상태 즉, 상태벡터들이 유일성을 갖는 것을 전제로 한다. 따라서, 레지스터의 길이를 결정하기 위하여 재배열된 상태 벡터들이 서로 다른 계승자들을 갖고 두면 또는 그 이상 존재하는지 조사한다. 만약 두 번 이상 동일한 상태 벡터들이 존재하게 되면 상태 벡터들은 그 다음 벡터의 첫 번째 비트를 사용하여 확장한다. T의 마지막 벡터는 다음 벡터가 없으므로 don't care에 위하여 확장한다. 테스트를 위한 상태 패턴들은 레지스터 상대 벡터의 최종 n개 비트를 windowing 하여 얻는다.

이와 같은 절차는 어떠한 상태 벡터도 두 번 이상 나타나질 않을 때까지 실행한다. 따라서 레지스터의 길이는 확장되기 전의 상태 벡터 길이와 유효한 상태 벡터를 구하기 위하여 확장을 반복한 회수의 합으로 결정한다. NLFSR의 설계를 위한 마지막 단계로. NLFSR의 궤환 함수의 계산은 다음과 같은 방법에 위하여 쉽게 할 수 있다. 재배열된 상태벡터들은 시프트 레지스터의 시프트 작용에 의하여 생성되므로 현재의 상태 벡터가 다음 상태 벡터의 첫 번째 비트를 생성하게 된다. 따라서 이들은 일반적으로 불안전한 Boolean 함수로 표현되는데 Karnaugh Hap등에 위하여 간략화 할 수 있다.

제 3 절 CMOS회로의 테스트를 위한 NLFSR 설계

2 비트 CMOS 가산기의 고장들을 검출차기 위해 제 3 장에서 서술한 경로 활성화방법에 위하여 다음 그림 7과 같이 각각 8개의 초기화 패턴(Ti)과 테스트 패턴(Tt)들이 구해졌을 경우 설계 절차는.

1. 초기화 패턴(Ti)과 테스트 패턴(Tt)들을 단일 상태 벡터들로 pattern merging한다. 그림 7에 pattern=merging 과정을 나타냈다.
2. 단일 상태 벡터들을 시프트 레지스터 시이퀀스의 일부분이 되도록 재배열한 후 상태 벡터들의 집합 T를 구한다. 그림 7에서 집합 T는 새롭게 재배열한 상태 벡터들이다. "*" 표시의 패턴들은 Tm에는 없는 링크 패턴이다.
3. 절차 2에서 구해진 시프트 레지스터 시이퀀스들은 서로 다른 벡터들로 구성되어 있으므로 벡터 확장이 필요 없으며 벡터의 최초 또는 최종 어느 비트도 생략되지 않는다. 따라서 레지스터 길이는 8이다.

4. 궤환 함수들은 시프트 레지스터 시이퀀스 T의 시프트 작용에 따라 그 다음 벡터의 첫 번째 비트를 함수값으로 정하고 Karnaugh Map을 이용하여 간략화 한다.
5. 따라서 NLFSR에 의한 테스트 패턴 발생기는 그림 8과 같이 구성된다.

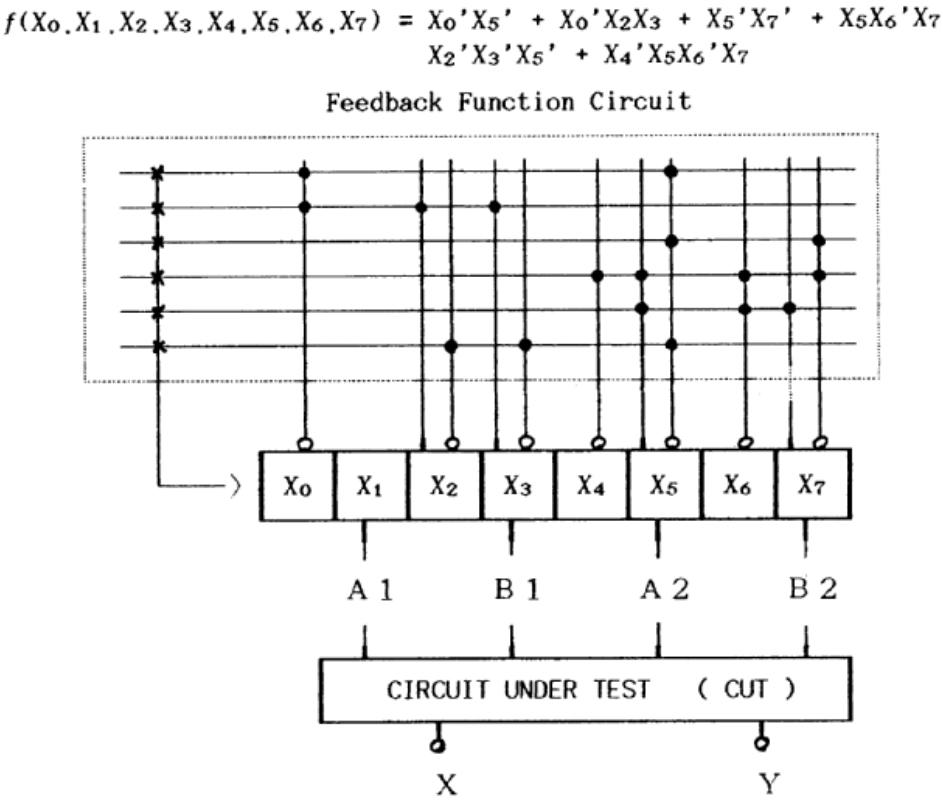


그림 8. 2비트 CMOS 가산기의 테스트를 위한 TPG

제 4 절 PLA의 테스트를 위한 BILBO의 구성

2 개의 NOR plane 또는 NAND plane으로 구성된 PLA 회로를 Built-In Self Test방식으로 테스트하기 위해 궤환 선택회로를 첨가한 BILBO를 구성하고 두개의 제어선에 따라 여러 가지 기능을 수행하도록 하였다. PLA 회로의 Built-In self Test를 위해 그림 9에 궤환 선택 회로를 부가한 다기능 BILBO를 나타냈다. 궤환 선택 회로는 제어 신호 C1이 0이면 테스트 패턴 생성을 위한 NLFSR을 선택하고 C1이 1이면 Signature Analysis를 위한 LFSR을 선택한다. 따라서 다기능 BILBO는 TPG와 LFSR의 기능이 서로 독립적으로 수행되도록 하였다. 또한 다기능 BILBO는 PLA와 같은 CUT의 외부회로에서 TPG의 출력선과 NISR의 입력선을 같은 선으로 구성할 수 있도록 Tri-State 버퍼를 이용하였다. 표 3에 다기능 BILBO의 기능을 나타냈다.

PLA 회로의 고장 검출을 위해 Built-In Self Test 방식 적용은 BILBO의 설계가 가장 큰 문제가 된다. 본 연구에서는 조합 회로의 고장 검출 방법에 따라 얻은 테스트벡터를 NLFSR에 의하여 발생하도록 하였고 NLFSR 과 LFSR로 BILBO를 구성함으로서 Built-In self Test가 가능하도록 하였다. 또한 3개의 BILBO를 그림 10과 같이 배치하여 PLA 회로의 테스트가 용이하도록 하였다.

표 5. 다기능 BILBO의 형식

C1	C2	전송 형식	기 능	클럭
~	~	scan mode	scan path	A,B
0	0	입력 mode	Reset	C,B
0	1	출력 mode	TPG(NLFSR)	C,B
1	0	입력 mode	정상동작	~ ~
1	1	입력 mode	MISR	C,B

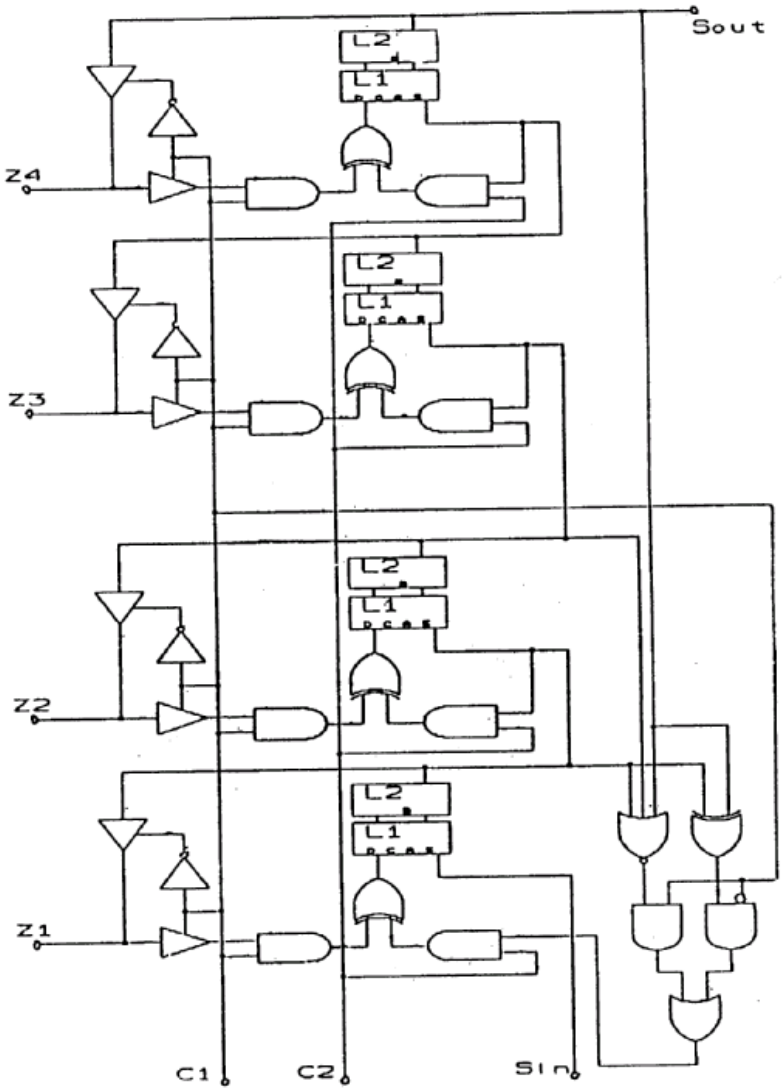
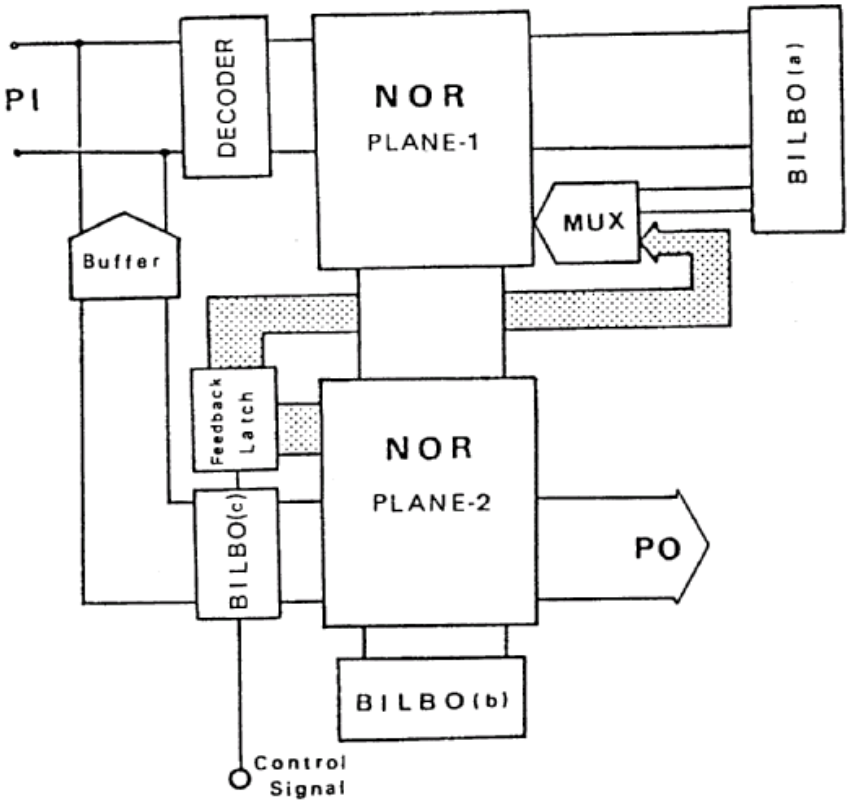


그림 9. 다기능 BILBO

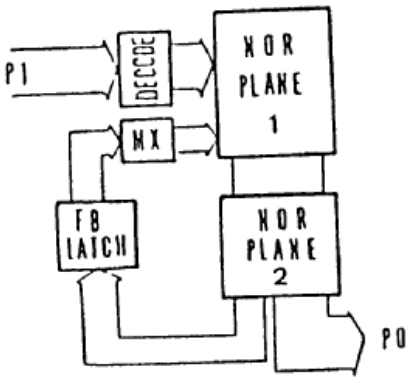


PI : Primary Input (원시 입력)

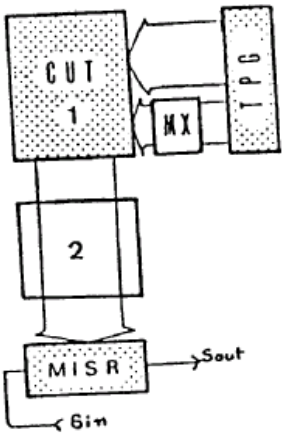
PO : Primary Output (원시 출력)

그림 10. CMOS PLA의 Test를 위한 Built In Self Test

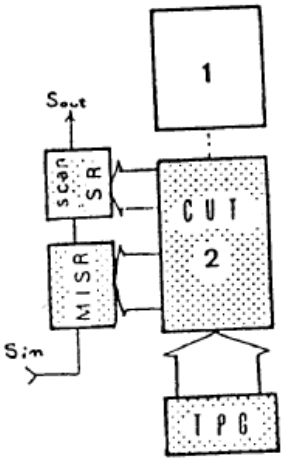
BILBO를 이용한 PLA 회로의 테스트 동작은 제어신호에 따라 BILBO(a)와 BILBO(b)를 한 쌍으로 NOR plane 1을 테스트하고, BILBO(b)과 BILBO(c)를 한 쌍으로 NOR plane 2를 테스트한다. 그림 11에 테스트 절차를 나타냈다. 특히 PLA 회로의 입출력의 신호를 같은 선으로 전달할 수 있도록 BILBO (a)와 BILBO(b)는 High Impedance 기능이 있는 3-State 버퍼를 사용하였다. 또한 궤한 선택회로에 의하여 NLFSR과 LFSR의 기능을 독립적으로 수행할 수 있도록 하였다. 그림 9의 다기능 BILBO를 이용한 그림 10의 구성을 검증차기 위하여 그림 11(b)와 같이 분할하여 시뮬레이션 실험을 실행하였다. 분할한 NOR plane을 VAX/750(UNIX) 컴퓨터상에서 C언어의 논리 레벨 시뮬레이션으로 실현하였다. 실험한 논리 레벨 시뮬레이션은 게이트 레벨로서 하드웨어 기술연어의 컴파일러는 게이트 구조, 연결 구조들로 데이타 구조를 생성하며, 시뮬레이션 알고리즘은 문헌(80)의 방법을 이용하여 구성하였다. 테스트를 위한 NOR-Plan 1의 회로구성은 그림 12와 같으며, 그림 13은 그림 12약 블럭들을 나타낸다. 그림 12의 시뮬레이션을 위하여 표 5와 같이 입력 데이타를 구성하고 그 결과는 표 6에 나타냈다. 회로의 고장은 NOR3, NOR5의 missing고장을 가정하였다. 표 5에 나타낸 입력 데이타를 참고하면서 표 6의 Signature를 비교하면 제시한 설계 방법이 유효함을 알 수 있다.



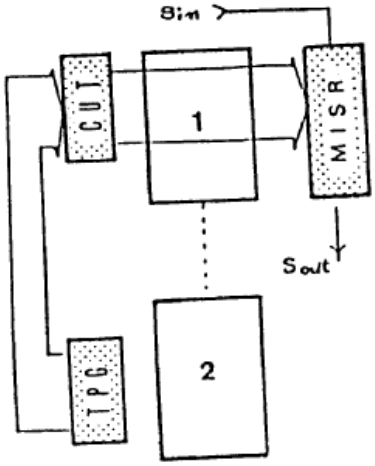
(a) 정상 동작



(b) NOR plane-1 테스트



(c) NOR plane-2 테스트



(d) Decoder 테스트

그림 11. 그림 10의 정상 동작과 테스트 동작

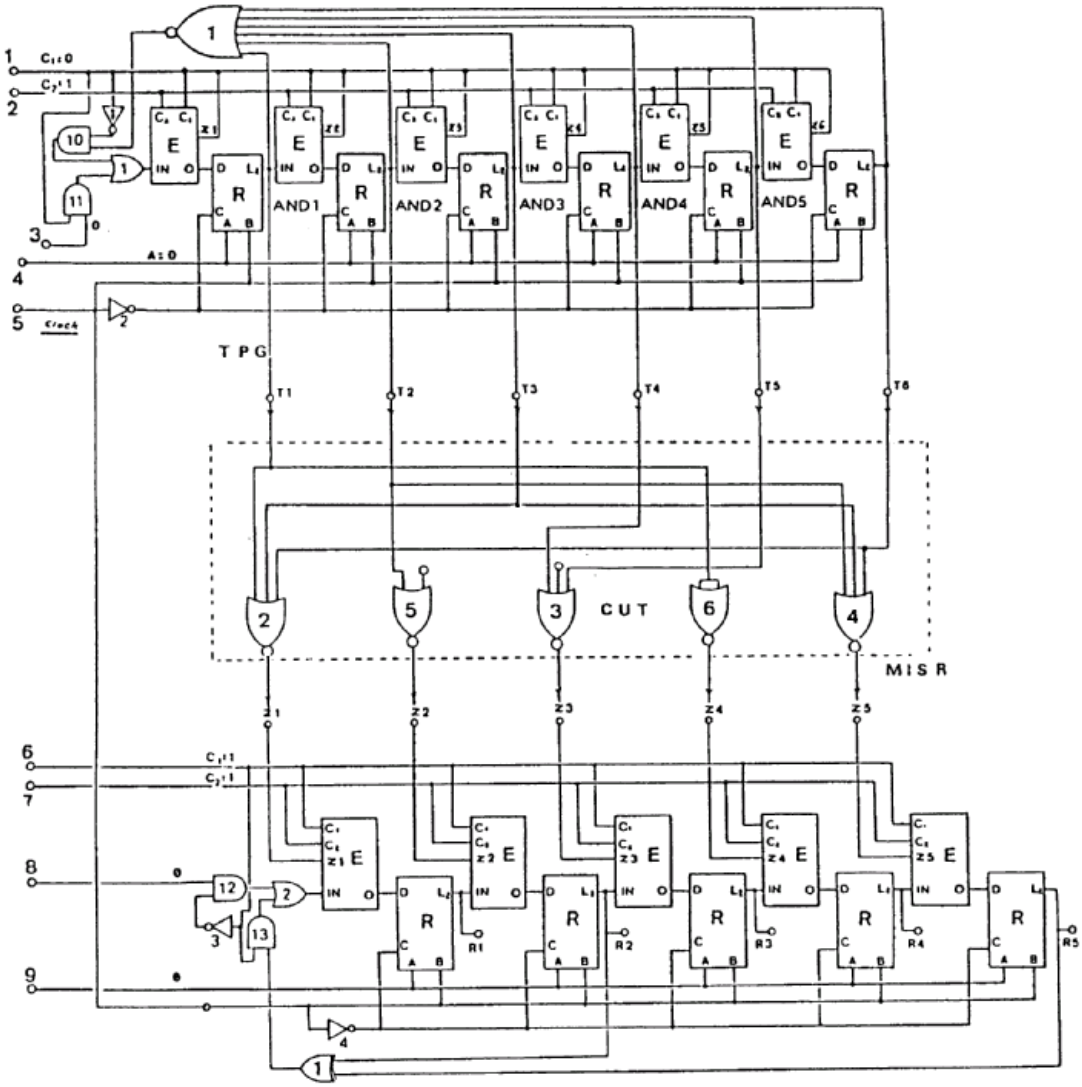


그림 12. 논리 시뮬레이션을 위한 실험회로

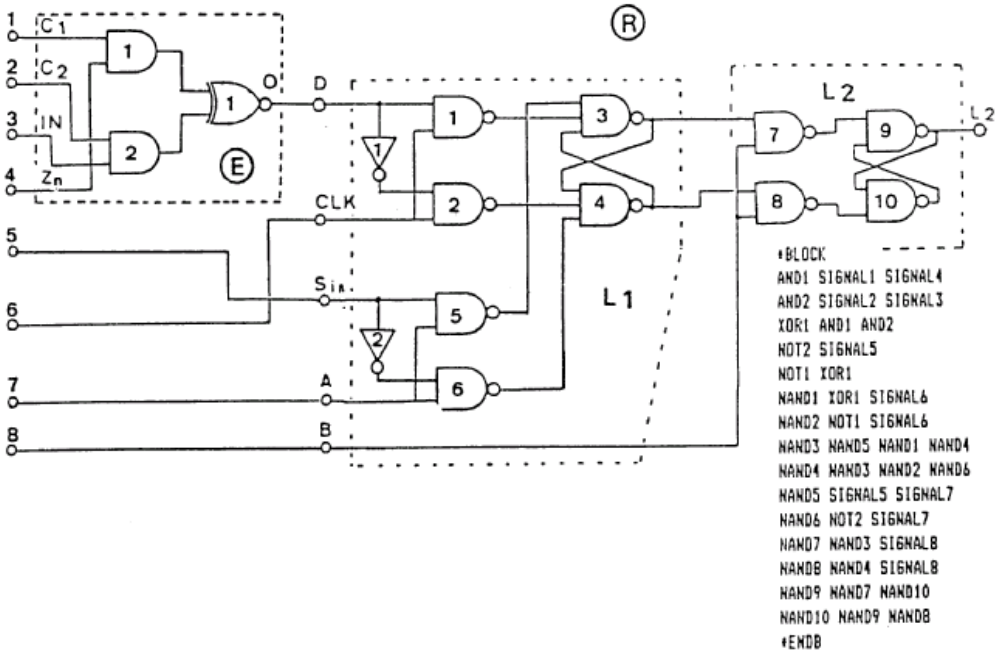


그림 13. 그림 12의 블럭회로와 블럭 시뮬레이션을 위한 입력 데이터

표 5. 그림 12의 실험을 위한 입력 데이타 (C, D점의 missing 고장)

```
*CIRCUIT
AND10 NOT1 NOR1
AND11 SIGNAL1 SIGNAL3
AND12 SIGNAL8 NOT3
AND13 SIGNAL6 XOR1
OR1 AND10 AND11
OR2 AND12 AND13
NOT1 SIGNAL1
NOT2 SIGNAL5
NOT3 SIGNAL6
NOT4 SIGNAL5
NOR1 AND1 AND2 AND3 AND4 AND5 BLOCK6 NAND9
NOR2 AND1 AND3 BLOCK6 NAND9
NOR3 AND4 AND5   ---------- ( missing device of C-point )
NOR4 AND2 AND3 BLOCK6 NAND9
NOR5 AND2 AND2    ---------- ( missing device of D-point )
NOR6 AND1 AND1
XOR1 AND7 BLOCK11 NAND9
AND1 BLOCK1 NAND9 BLOCK1 NAND9
AND2 BLOCK2 NAND9 BLOCK2 NAND9
AND3 BLOCK3 NAND9 BLOCK3 NAND9
AND4 BLOCK4 NAND9 BLOCK4 NAND9
AND5 BLOCK5 NAND9 BLOCK5 NAND9
AND6 BLOCK7 NAND9 BLOCK7 NAND9
AND7 BLOCK8 NAND9 BLOCK8 NAND9
AND8 BLOCK9 NAND9 BLOCK9 NAND9
AND9 BLOCK10 NAND9 BLOCK10 NAND9
BLOCK1 SIGNAL1 SIGNAL2 OR1 SIGNAL1 SIGNAL10 NOT2 SIGNAL4 SIGNAL5
BLOCK2 SIGNAL1 SIGNAL2 AND1 SIGNAL1 SIGNAL10 NOT2 SIGNAL4 SIGNAL5
BLOCK3 SIGNAL1 SIGNAL2 AND2 SIGNAL1 SIGNAL10 NOT2 SIGNAL4 SIGNAL5
BLOCK4 SIGNAL1 SIGNAL2 AND3 SIGNAL1 SIGNAL10 NOT2 SIGNAL4 SIGNAL5
BLOCK5 SIGNAL1 SIGNAL2 AND4 SIGNAL1 SIGNAL10 NOT2 SIGNAL4 SIGNAL5
BLOCK6 SIGNAL1 SIGNAL2 AND5 SIGNAL1 SIGNAL10 NOT2 SIGNAL4 SIGNAL5
BLOCK7 SIGNAL6 SIGNAL7 OR2 NOR2 SIGNAL11 NOT4 SIGNAL9 SIGNAL5
BLOCK8 SIGNAL6 SIGNAL7 AND6 NOR5 SIGNAL11 NOT4 SIGNAL9 SIGNAL5
BLOCK9 SIGNAL6 SIGNAL7 AND7 NOR3 SIGNAL11 NOT4 SIGNAL9 SIGNAL5
BLOCK10 SIGNAL6 SIGNAL7 AND8 NOR6 SIGNAL11 NOT4 SIGNAL9 SIGNAL5
BLOCK11 SIGNAL6 SIGNAL7 AND9 NOR4 SIGNAL11 NOT4 SIGNAL9 SIGNAL5
*END0
*OUTPUT
BLOCK1 NAND9 BLOCK2 NAND9 BLOCK3 NAND9
BLOCK4 NAND9 BLOCK5 NAND9 BLOCK6 NAND9
NOR2 NOR5 NOR3 NOR6 NOR4
BLOCK7 NAND9 BLOCK8 NAND9 BLOCK9 NAND9
BLOCK10 NAND9 BLOCK11 NAND9
*END0
```

표 6. 그림 12의 실험 결과 (C, D점의 missing 고장)

Input Signal											Output Signal														
1	2	3	4	5	6	7	8	9	10	11	T1	T2	T3	T4	T5	T6	Z1	Z2	Z3	Z4	Z5	R1	R2	R3	R4	R5
0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	0	0	0	0	0
0	1	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	0	0	0	0	0
0	1	0	0	1	1	1	0	0	0	0	1	0	0	0	0	0	0	1	1	0	1	1	1	1	1	1
0	1	0	0	0	1	1	0	0	0	0	1	0	0	0	0	0	0	1	1	0	1	1	1	1	1	1
0	1	0	0	1	1	1	0	0	0	0	0	1	0	0	0	0	1	0	1	1	0	0	0	0	1	0
0	1	0	0	0	1	1	0	0	0	0	0	1	0	0	0	0	1	0	1	1	0	0	0	0	1	0
0	1	0	0	1	1	1	0	0	0	0	0	0	1	0	0	0	0	1	1	1	0	1	0	1	1	1
0	1	0	0	0	1	1	0	0	0	0	0	0	1	0	0	0	0	1	1	1	0	1	0	1	1	1
0	1	0	0	1	1	1	0	0	0	0	0	0	0	1	0	0	1	1	0	1	1	1	0	1	0	1
0	1	0	0	0	1	1	0	0	0	0	0	0	0	1	0	0	1	1	0	1	1	1	0	1	0	1
0	1	0	0	1	1	1	0	0	0	0	0	0	0	0	1	0	1	1	0	1	1	0	0	0	0	1
0	1	0	0	0	1	1	0	0	0	0	0	0	0	0	1	0	1	1	0	1	1	0	0	0	0	1
0	1	0	0	1	1	1	0	0	0	0	0	0	0	0	0	1	0	1	1	1	0	0	1	0	1	1
0	1	0	0	0	1	1	0	0	0	0	0	0	0	0	0	1	0	1	1	1	0	0	1	0	1	1
0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	0	1	0	1	1
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	0	1	0	1	1
0	1	0	0	1	1	1	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	0	0	0	0	0
0	1	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	0	0	0	0	0
0	1	0	0	1	1	1	0	0	0	0	1	0	0	0	0	0	0	1	1	0	1	1	1	1	1	1
0	1	0	0	0	1	1	0	0	0	0	1	0	0	0	0	0	0	1	1	0	1	1	1	1	1	1
0	1	0	0	1	1	1	0	0	0	0	0	1	0	0	0	0	1	0	0	1	0	0	0	0	1	0
0	1	0	0	0	1	1	0	0	0	0	0	1	0	0	0	0	1	0	0	1	0	0	0	0	1	0
0	1	0	0	1	1	1	0	0	0	0	0	0	1	0	0	0	0	1	1	1	0	1	0	0	1	1
0	1	0	0	0	1	1	0	0	0	0	0	0	1	0	0	0	0	1	1	1	0	1	0	0	1	1
0	1	0	0	1	1	1	0	0	0	0	0	0	0	1	0	0	1	1	0	1	1	1	0	1	1	1
0	1	0	0	0	1	1	0	0	0	0	0	0	0	1	0	0	1	1	0	1	1	1	0	1	1	1
0	1	0	0	1	1	1	0	0	0	0	0	0	0	0	1	0	1	0	0	1	1	0	0	0	0	0
0	1	0	0	0	1	1	0	0	0	0	0	0	0	0	1	0	1	0	0	1	1	0	0	0	0	0
0	1	0	0	1	1	1	0	0	0	0	0	0	0	0	0	1	0	1	1	1	0	1	0	0	1	1

Input Signal 1: Control Terminal C1 of TPG.
Input Signal 2: Control Terminal C2 of TPG.
Input Signal 3: Input Terminal of AND-11.
Input Signal 4: Clock Terminal A of TPG Block-R.
Input Signal 5: Clock Terminal of Circuit Fig.13.
Input Signal 6: Control Terminal C1 of MISR
Input Signal 7: Control Terminal C2 of MISR
Input Signal 8: Input Terminal of AND-12.
Input Signal 9: Clock Terminal A of MISR Block-R.
Input Signal 10: Scan Input Terminal of TPG Block-R.
Input Signal 11: Scan Input Terminal of MISR Block-R.

제 5 장 결 론

본 연구에서는 CMOS 회로의 고전적인 고장과 Stuck open 고장 검출을 용이하게 하기 위해 테스트 패턴 발생 방법을 개발하였다. CMOS회로의 Stuck-open 고장은 경로 활성화에 위하여 검출하고 그의 적합한 TPG의 설계는 결합 시이퀀스의 Merging과 재배열에 의하여 NLFSR을 설계함으로서 Built-In Self Test가 가능하도록 하였다. 또한 비선형적으로 테스트 패턴을 생성하고 선형적으로 Signature Analysis를 수행하도록 다기능 BILBO를 구성하였다. 개발된 설계방법의 특징을 요약하면 다음과 같다.

1. CMOS의 Stuck open고장을 검출하기 위해 경로 활성화 방법을 이용하여 최소의 테스트 패턴을 구하였다.

2. 다기능 BILBO를 설계하여 비선형적(NLFSR)으로는 테스트 패턴을 생성하고 선형적(LFSR)으로는 signature analysis를 알 수 있도록 하였다. 그 결과, 최소의 하드웨어를 가지고 매우 적은 테스트 패턴을 발생시킬 수 있었다.

3. 다기능 BILBO의 입력과 출력은 3-State 버퍼를 이용하여 동일 선로에서 처리하였다. 그 결과, BILBO의 외부 배선 수는 감소되어 chip 배선을 용이하게 하였다.

4. 개발된 다기능 BILBO를 PLA 회로에 적용한 결과, 정상 동작시 종래의 단점인 데이타 신호의 전파지연을 일으키지 않도록 하였으며 테스트 동작시 테스트 패턴 시이퀀스는 종래의 pr-TPG보다 조금 감소되었다.

참 고 문 헌

[1] K.J.Niraj, "Multiple Stuck-Open Fault Detection in CMOS logic circuits" IEEE Trans. Vol.37. No 4, pp.426-432, April 1988

[2] J.Paul Roth, "Diagnosis of Automata Failures : A Calculus and a method", IBM Journal, pp.278-291, July 1966

[3] Nadhukar K.Reddy and Sudhakar N.Reddy, "Detecting FET Stuck-Open Faults in CMOS Latches and Flip-Flops", IEEE Design & Test, pp.17-26, Oct. 1986

[4] Martin G.Ruehler and Michael V.Sievers, "Off-Line, Built-in Test Techniques for VLSI Circuits" IEEE Comput. pp.69-82, June. 1982

[5] Gopal Gupta, "A Universal Test Set for CMOS Circuit", IEEE Trans. Computer-Aided Design, Vol.7. No.5, MAY 1988

[6] Sudhakar M.Reddy, "Testable Realizations for FET Stuck-Open Faults in CMOS Combinational Logic Circuits", IEEE Trans. Compt. Vol.C-35, No.8, Aug. 1986

[7} E.J. McC1uskey, "Built-in Self Test, Techniqes", IEEE Design and Test, pp.29-36, April 1985

[8] P. S. Noritz and L.M.Thorsen, "CMOS circuit testablity" IEEE Custom Integrated Circuit Conf. pp.311-314 1985

[9] G.Gupta and N.K.Jha, "A Universal Test Set, for CMOS Circuits" IEEE Tcad. Vol. 7. no 5. NAY 1988

[10] Y.H.Elziq and R.J.C1outier, "Function-level Test Generation for Stuck-Open Faults in CMOS VLSI," IEEE Test Conf. pp.536-546 1981.
[11] Y.N.Elziq, "Automatic Test Generation for Stuck-Open Faults in CMOS VLS", in Proc. Design Automation Conf. pp.347-354 1981.
[12] S.K.Jain and V.D.Agraval, "Test genaration for MOS circuits using D-algorithem" in Proc .Design Automation Conf. pp.64-70 1983.
[13] S.N.Reddy and M.K.Reddy, "Testable realization for FET stuck open fault in CMOS combinational logic circuits", IEEE Trans. pp.742-754 1986.
[14] S.M.Reddy and H.K.Reddy, "On testable design for CMOS logic circuits", in Proc. Int. Test Conf. pp.435-445, 1983.
[15] S.R.Rho, I.C.Lim "Testable Design on the Built-in Test Method" J. of KIEE, Vol.24., No.3., pp.166-171, 1987
[16] S.K.Jain and V.D.Agrawal, "Test genaration for MOS circuits using D-algorithe", in Proc. Design Automation Conf. pp.64-70 1983.
[17] S.R.Rho,Y.M.Kim, B.H.Min "VLSI Testable Design for Autonomous Test" Seoul City Univ. Theses, Vol.22 pp.395-409, 1988

[18] R.Chandramouli, "On testing stuck open faults", in Proc. Int. Symp. Fault-Tolerant Comput. pp.258-265, 1983.
[19] B.KOnnemann,J.Nucha.G.Zviehoff, "Built-in Logic Block Observation Techniques", Proc. of 1979 IEEE Test Conf., pp.37-41, Oct. 1979.
[20} S.Bozorgui-Nesbat, E.J.McCluskey, "Structured Design for Test ability, to Eliminate Test Pattern Generation", Proc. of 10th Intern. Symposium Fault-Tolerant Computing, pp.158-163, 1980
[21] W.Daehn, J.Mucha, "A Hardware Approach to Self-Testing of Large Programmable Logic Arrays", IEEE Trans. Comput., Vol.C-30, No.11 pp 829-833, Nov.1981
[22] P.P.Fasang. "BIDCO : Built-In Digital Circuit Observer". Proc. of IEEE Test Conf.. pp.261-266, 1980
[23] T.W.Williams, K.P.Parker, "Design for Testability a Survey", in Proc. of the IEEE, Vol.71, No.1, Jan.1983
[24] D.K.Bhavsar, R.V.Heckelman, "Self Testing by Polynomial Division", proc. of 1981 IEEE Test Conf. pp.208-216, 1981
[25] D.Komonytsky, "LSI Self Test using Level Sensitive Scan Design and Signature Analysis", Proc. of IEEE Test Conf., pp.414-424, 1982

[26] E.B.Eichelberger, T.W.Williams, "A Logic Design Structure for LSI Testability", J. of Design Autom. Fault-Tolerant Computing, Vol.2, No.2, pp.165-178, 1978
[27] J.P.Hayes, "A NAND Model for Fault Diagnosis in Combinational Logic Networks", IEEE Trans. Comput., Vol.C-2O, No.12, pp.1496-1506, 1971
[28] B.Konnemann, J.Mucha, G.Zwiehoff, "Built-in Test for Complex Digital Integrated Circuits", IEEE J. of Solid Circuits, Vol.SC-15, No.3, pp.315-318, Jun. 1980
[29] E.J.McCluskey, S.Bozorgui-Nesbat, "Design for Autonomous Test", IEEE Trans. Comput., Vol.C-30, No.11, pp.866-875, 1981