DBpia

매칭에 기반한 발전된 고장 진단 방법

정회원 임 요 섭*, 강 성 호**

Matching-based Advanced Integrated Diagnosis Method

Yoseop Lim*, Sungho Kang** ***Regular Members***

요 약

본 논문에서는 효율적인 다중 고착 고장 진단 알고리듬을 제안하겠다. 제안하는 고장 진단 알고리듬은 완전일치공통부분을 고장 진단의 중요한 기준으로 사용함으로써 단일 고착 고장 시뮬레이터 환경에서도 다중 고착 고장을 진단할 수 있다. 또한 각 고장간의 식별성을 높여 다중 고착 고장을 진단함에도 불구하고, 고장 후보의 수를 획기적으로 줄일 수 있었다. 이를 위하여 출력단의 수에 따른 가중치 개념과 가산, 감산 연산을 사용하였다. 이 알고리듬은 ISCAS85회로와 완전 주사 스캔이 삽입된 ISCAS89회로에서 실험하여 성능을 입증하였다.

Key Words : Matching Algorithm, Diagnosis, Fault Simulation, Multiple Stuck-At Faults

ABSTRACT

In this paper, we propose an efficient diagnosis algorithm for multiple stuck-at faults. Because of using vectorwise intersections as an important metric of diagnosis, the proposed diagnosis algorithm can diagnose multiple defects in single stuck-at fault simulator. In spite of multiple fault diagnosis, the number of candidate faults is drastically reduced. For identifying faults, the variable weight, positive calculations and negative calculations are used for the matching algorithm. To verify our algorithm, experiments were performed for ISCAS85 and full-scan version of ISCAS89 benchmark circuits.

Ⅰ. 서 론

공정기술의 발달로 선폭이 좁아지고 한 개의 칩에 집적되는 회로의 크기가 커지면서 생산 공정 중에 결함이 나타날 가능성이 높아지고 있다. 시장 경쟁력을 높이기 위해서는 수율 향상시키는 것이 필수적이므로 결함이 생성되는 공정을 개선하는 작업이 필요하다. 고장 진단이란 동작의 오류를 일으키는 결함의 위치와 종류를 추론해내는 과정을 말한다. 정확한 고장 진단으로 설계와 공정의 오류를 찾아내어 수정할 수 있게 해준다. 따라서 칩의 품질을 높이고 생산 비용을 절감하기 위한 효율적인 고장 진단 방법론을 개발하는 것은 매우 중요하다[1].

고장 진단을 수행하기 위해서는 몇 가지의 필수적인 사항이 필요하다. 우선 고장검출률이 높은 테스트 패턴이 필요하다. 회로에 존재하는 결함을 출력단에서 확인할 수 있도록 하는 입력 패턴을 테스트 패턴이라 한다. 우선 오류가 출력단에서 검출되도록 하는 테스트 패턴을 사용하여야만 그 원인이 되는 고장을 찾아낼 수 있기 때문이다. 또한 고장 모델과 고장 시뮬레이터가 필요하다. 고장 모델이란 실제 회로에 존재하는 결함을 논리적 도메인으로 모델링한 것을 말한다. 실제 결함과 유사하게 고장 모델을 모델링하고 다양한 고장 모델을 사용한다면

※ 본 연구는 산업자원부 SYSTEMIC2010 과제 지원 및 한국반도체연구조합 관리로 수행되었습니다.
* 연세대학교 전기전자공학과 컴퓨터시스템 및 관련 SoC 연구실 (yoseop@soc.yonsei.ac.kr)
** 연세대학교 전기전자공학과 컴퓨터시스템 및 관련 SoC 연구실 (shkang@yonsei.ac.kr)
논문번호 : KICS2006-11-507, 접수일자 : 2006년 11월 24일, 최종논문접수일자 : 2007년 3월 19일

www.dbpia.co.kr

정확한 고장 진단을 수행할 수 있으나 복잡한 고장 모델에 대한 고장 시뮬레이터의 수행 시간이 급격히 증가되게 된다.

한 개의 고착 고장이 존재한다고 가정하는 단일 고착 고장 모델이 테스트 패턴을 생성하기 쉽고, 시뮬레이터 구현도 단순하며 시뮬레이터의 속도도 빠르기 때문에 가장 널리 사용되고 있다. 그러나 단일 고착 모델만을 사용하면, 다른 형태의 고장인 합선 고장과 다중 고착 고장을 진단할 수 없기 때문에 이러한 고장들의 진단을 목표로 하는 많은 고장 진단 방법론이 제안되었다[2,3]. [2]에서는 다중 고착 고장을 진단하기 위하여 다중 고착 시뮬레이터를 사용하였다. 또한 합선고장을 진단하기 위하여 [3]에서는 합선 고장 시뮬레이터를 사용하였다.

그러나 복잡한 고장 모델을 위해 시뮬레이터를 사용하면 시뮬레이터의 수행시간이 길어지기 때문에 전체적인 고장 진단 시간이 길어진다. 예를 들면 단일 고착 고장 시뮬레이터의 수행 시간은 회로의 크기에 비례에서 증가하나 합선 고장 시뮬레이터의 수행 시간은 회로의 크기에 제곱에 비례해서 증가하므로 회로가 커진다면 실행 불가능할 수도 있게 되었다. 수행 시간이 너무 많이 소요되어 실행하지 못하는 문제점을 예방하기 위하여 제안하는 고장 진단 알고리듬은 단일 고착 모델과 단일 고착 고장 시뮬레이터를 기반으로 하였다.

또한 최종 고장 후보를 결정하기 위하여 점수화 매칭 알고리듬을 사용하였다. 점수화 매칭 방법론은 시뮬레이션 된 고장의 응답이 실제 테스터의 응답과 유사할수록 고장의 위치가 실제 결함의 위치와 가깝다는 가정을 기반으로 하고 있다.

점수화 매칭 알고리듬의 가장 기본적인 방법은 특정 테스트 벡터에서 오류가 관찰될 경우에 후보 고장의 출력단 응답에서 오류가 있을 경우, 그 후보 고장의 점수를 2점 증가 시킨다. 만일 미지의 값 X가 관찰될 경우는, 잠재적으로 오류가 검출되었다고 생각하여 1점을 증가 시킨다[4].

더욱 발전된 방법으로는 테스터의 응답과 후보 고장의 응답간의 일치, 부분 일치, 불일치의 정도에 따라 점수를 증가시킬 뿐만 아니라 감소시키기도 하여 보다 세밀하게 고장 후보와 실제 고장간의 유사도를 점수화 시킬 수 있다[5].

가장 발전된 형태의 POIROT 알고리듬[6]에서는 완전일치공통부분(vectorwise intersection), 공통부분(intersection), 잘못된 예측(misprediction), 예측실패(nonprediction)의 측정 기준으로 점수를 계산한다. 공통부분은 테스터의 응답과 고장 후보의 시뮬레이션 응답에 모두 오류가 나타날 때를 말한다. 이중 출력단의 시그니쳐가 완전히 동일할 경우를 완전일치공통부분이라고 말한다. 잘못된 예측은 테스터의 응답에서는 오류가 나타나지 않으나 고장 후보의 시뮬레이션 응답에서는 오류가 나타날 경우를 말한다. 반대로 예측실패는 테스터의 응답에서는 오류가 나타나나 고장 후보의 시뮬레이션 응답에서는 오류가 나타나지 않는 경우를 말한다. 완전일치공통부분의 수가 가장 큰 영향력을 가지며, 이어서 공통부분, 잘못된 예측, 예측실패의 순으로 영향력을 가지게 된다.

일반적으로 사용되는 점수화 매칭 알고리듬은 고장이 한 개만 존재한다고 가정하므로 고장 진단 과정을 단순화 시켜주지만 다중 고착 고장이나 복잡한 형태의 고장을 진단하기 어렵다. 본 논문에서는 점수화 매칭 알고리듬과 완전일치공통부분 테이블을 사용하여 단일 고착 고장 시뮬레이터 환경에서도 다중 고착 고장을 진단할 수 있는 매칭 알고리듬을 제안하였다. 또한 출력단의 수에 따른 가중치 개념을 도입한 점수화 매칭 알고리듬으로 고장간의 식별성을 높여 진단한 고장 후보 집합의 수를 줄이도록 하였다. 제안한 알고리듬을 사용하여 MAID (Matching-based Advanced Integrated Diagnosis)라고 명명한 툴을 제작하여 성능을 검증하였다.

Ⅱ. 고장 진단의 전체 과정

고장 진단의 전체 과정은 그림 1과 같다. 고장 진단을 수행하기 위해서는 회로의 설계 파일과 테스트 패턴이 필요하고, 오류가 발생한 칩의 테스터 응답 결과가 필요하다. 가장 먼저 고장 리스트 생성 과정을 거치게 된다. 고장 리스트 생성 과정에서는 회로의 설계 파일로부터 회로에 존재할 수 있는 논리 고장들을 결정한 다음에, 등가 고장 중첩과 임계경로 추적법[7]을 사용하여 시뮬레이션을 수행할 고장의 수를 줄이게 된다.

이렇게 결정된 고장 리스트로 고장 시뮬레이션을 진행하게 된다. 단일 고착 고장 시뮬레이션 과정 중에 후보 고장의 점수를 계산하고 완전일치공통부분 테이블을 생성한다. 고장 시뮬레이션 이후에 각 후보 고장의 점수와 완전일치공통부분의 테이블 결과로 최종 고장 후보 집합을 결정하여 최종 고장 진단 결과를 출력하게 된다.

www.dbpia.co.kr

정확하고 효율적인 고장 진단을 수행하는데 핵심적인 역할을 하는 후보 고장의 점수 계산, 완전일치공통부분 테이블 생성, 최종고장 후보 집합의 결정 과정은 Ⅲ~Ⅴ장에서 자세하게 설명하겠다.

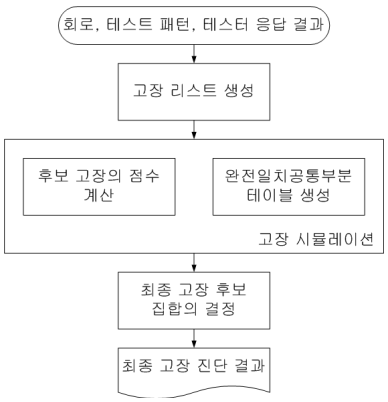

그림 1. 고장 진단의 전체 과정

Ⅲ. 후보 고장의 점수 계산

기존의 알고리듬은 단순히 완전일치공통부분, 공통부분, 잘못된 예측과 예측실패가 일어날 경우의 테스트 패턴 수만을 기준으로 점수를 계산하기 때문에 식별력이 떨어져 동순위 고장이 많이 나타날 수 있는 문제점을 가지고 있다. 또한 기존의 알고리듬은 무조건적으로 완전일치공통부분, 공통부분, 잘못된 예측, 예측실패의 순으로 영향력을 가진다. 다시 말하면, 공통부분, 잘못된 예측, 예측실패의 점수가 아무리 높더라도, 완전일치공통부분의 점수가 낮다면 낮은 고장 후보 순위를 가지게 된다. 이는 다중 고착 고장 등의 복잡한 형태의 고장을 진단할 경우에 문제점을 가지게 된다. 다중 고착 고장일 경우에는 테스터의 고장 응답의 형태가 두 고착 고장이 중첩된 형태로 나타나는 경향을 가지게 되므로 실제의 각 고장은 공통부분의 점수가 낮게 된다.

MAID 알고리듬에서는 고장간의 식별력을 높이기 위하여 후보 고장의 점수를 계산할 경우에 패턴의 수를 고려하지 않고, 응답을 보다 더 잘 반영할 수 있도록 출력단의 수를 고려하였다. 또한 다중 고착 고장과 같은 복잡한 고장의 진단을 위해서 완전일치공통부분, 공통부분, 잘못된 예측과 예측 실패의 경우에 앞의 것이 뒤의 것에 우선하는 영향력을 가진 기준으로 사용하지 않고 통합적으로 점수를 계산하도록 하였다. 또한 회로의 크기에 탄력적으로 대응할 수 있도록 완전일치공통부분의 가중치를 결정하였다.

고장 시뮬레이션은 고장 목록의 선택된 고장에 대하여 모든 패턴을 시뮬레이션을 진행하는 이중 루프의 형태로 진행되도록 한다. 한 패턴에 대한 시뮬레이션이 진행되면 매칭알고리듬을 통하여 점수를 계산하며, 모든 패턴에 대한 점수를 합산하여 선택된 고장의 점수를 결정한다.

고장 시뮬레이션은 고장 목록의 선택된 고장에 대하여 모든 패턴을 시뮬레이션을 진행하는 이중 루프의 형태로 진행되도록 한다. 한 패턴에 대한 시뮬레이션이 진행되면 매칭알고리듬을 통하여 점수를 계산하며, 모든 패턴에 대한 점수를 합산하여 선택된 고장의 점수를 결정한다.

제안하는 매칭 알고리듬은 고장의 경우를 다음과 같이 5가지 경우로 분류하고 점수의 계산방법을 결정하였다.

1) 완전일치공통부분 : 테스터의 응답과 시뮬레이션의 응답이 모두 오류가 나타나며 완전히 일치할 경우를 말한다. 이 경우는 모든 출력단의 수만큼 점수를 증가시킨다.
2) 공통부분 : 테스터의 응답과 시뮬레이션의 응답이 모두 오류가 나타나나 완전히 일치하지는 않을 경우를 말한다. 양쪽 모두 오류가 나타나는 출력단의 수만큼 점수를 증가시키며, 한쪽만 오류가 나타나는 출력단의 수만큼 점수를 감소시킨다.
3) 잘못된 예측 : 테스터의 응답에서는 오류가 나타나지 않으나 시뮬레이션의 응답에서는 오류가 나타날 경우를 말한다. 시뮬레이션 응답에서 오류가 나타나는 출력단의 수만큼 점수를 감소시킨다.
4) 예측실패 : 시뮬레이션의 응답에서는 오류가 나타나지 않으나 테스터의 응답에서 오류가 나타날 경우를 말한다. 테스터의 응답에서 오류가 나타나는 출력단의 수만큼 전수를 감소시킨다.
5) 무고장 영역 : 테스터의 응답과 시뮬레이션의 응답에서 모두 오류가 나타나지 않을 경우를 말한다. 이 경우는 점수에 변화를 주지 않는다.

완전일치공통부분에 대하여 출력단의 수만큼 가중치를 주게 되므로 회로의 크기에 탄력적으로 대응할 수 있도록 되었다. 그리고 테스터의 응답과 시뮬레이션의 응답이 일치하더라도 고장의 유무에 따

www.dbpia.co.kr

라 완전일치공통부분과 무고장 영역으로 구별함에 따라 고장의 전파가 적게 되는 고장이 무고장 영역에서 높은 점수를 얻어 높은 순위를 가지는 문제점을 방지할 수 있게 되었다.

그림 2의 예제에서 점수가 계산되는 방식을 자세하게 설명하도록 하겠다. 그림 1에서 나타난 값들은 의도하였던 값과 같으면 0, 오류가 나타나면 1을 나타내는 시그니쳐 값이다. 테스트 패턴 1에 대하여 테스터 응답과 후보 고장 1은 둘 다 오류가 나타나고 모든 출력단에 대하여 동일한 형태를 가지고 있으므로 완전일치공통부분으로 연산된다. 5개의 주출력단을 가지고 있으므로 후보 고장 1은 5점을 증가시키게 된다. 테스트 패턴1에 대하여 후보 고장 2는 테스터 응답은 오류가 있으나, 후보 고장의 시뮬레이션 응답에는 오류가 없으므로 예측실패로 연산된다. 이 경우는 테스터 응답에서 2회 오류가 나타났으므로 2만큼 점수를 감소시킨다. 테스트 패턴1에 대하서 후보 고장 3은 테스터 응답과 후보 고장의 시뮬레이션 응답 모두 오류가 나타나지만 완전히 일치하지 않으므로 공통부분으로 연산한다. 같은 출력단에 오류가 나타나는 경우가 1회이므로 1점을 증가시키고 둘 중에 하나만 고장이 나타난 경우가 2회이므로 2점을 감소시켜 총 1점을 감소시킨다. 이러한 식으로 모든 패턴에 대하여 총합을 구하면 후보 고장 1은 총 5점을 가지며, 후보 고장 2는 -6점이고 후보 고장 3은 -4점으로 총점이 가장 높은 후보 고장 1이 실제 고장과 가장 유사하다고 판단할 수 있다.

	테스트 패턴 1	테스트 패턴 2	테스트 패턴 3
테스터 응답	00110	00000	00110
후보 고장 1	00110	00000	00010
후보 고장 2	00000	11000	10011
후보 고장 3	01100	00001	00000

그림 2. 후보 고장의 점수 계산 예제

Ⅳ. 완전일치공통부분 테이블

일반적인 점수화 매칭 알고리듬으로는 다중 고착 고장을 진단할 때 어려움이 따른다. 모든 고장이 최상위 점수를 가지지 않는 경우가 많기 때문이다.

이를 보완하기 위한 방법을 찾기 위하여 다중 고착 고장 회로의 응답 결과를 분석하였다. 그 결과, 조합 회로와 완전 주사 회로의 경우에는 다중 고착 고장 회로의 응답은 다중 고착 고장 회로에 삽입된 고장들을 한 개씩 삽입한 회로의 오류들이 중첩된 형태로 나타나는 경우가 매우 많음을 알게 되었다. 테스트 패턴이 한 개의 고장만을 활성화 시킬 경우에는 그 고장만 삽입된 회로의 응답과 동일하였고, 여러 고장을 활성화 시킬 경우에는 오류가 중첩되거나 전혀 다른 응답 결과를 나타내게 되었다. 만일 테스트 패턴이 한 개의 고장만을 활성화 시킨다고 가정한다면, 다중 고착 고장의 응답은 단일 고착 고장들의 완전일치공통부분의 중첩된 형태로 나타난다고 할 수 있다. SLAT 논문에서도 한 개의 고장만을 활성화 하는 패턴이 대부분을 차지한다는 실험결과를 보여주었다[8]. 이러한 가정을 다중 고착 고장을 진단 시에 사용하기 위하여 고장 시뮬레이션 과정에서 완전일치공통부분이 나타날 경우에 각 고장별로 완전일치공통부분이 나타났던 패턴들을 저장해 두며, 이를 완전일치공통부분 테이블이라고 부르도록 하겠다. 이 테이블은 고장 점수와 함께 최종 고장 후보를 결정할 경우에 사용하게 된다.

Ⅴ. 최종 후보 고장 집합의 결정

고장 시뮬레이션이 종료된 이후에는 각 고장 후보에 대하여 테스터의 응답과 유사도를 반영한 점수와 완전일치공통부분이 나타났던 고장과 테스트 패턴을 저장한 완전일치공통부분 테이블을 사용할 수 있다. 두 가지 기준을 사용하여 최종 고장 후보를 결정하게 된다. 최종 고장 후보 집합을 결정하는 방법은 다음과 같은 단계를 거치게 된다.

1) 고장 점수 순으로 후보 고장을 정렬한다.
2) 가장 점수가 높은 후보 고장들 중에서 가장 완전일치공통부분이 많이 나타나는 후보 고장을 선택하여 최종 후보 고장 집합에 포함시킨다.
3) 테스트 응답에서는 오류가 나타났지만 이 패턴에 대하여 완전일치공통부분이 나타나는 후보 고장이 최종 후보 고장 집합에 없을 경우에, 이 패턴에 대해서 완전일치공통부분이 나타나며 그 중 가장 점수가 높은 후보 고장을 최종 후보 고장 집합에 포함시킨다.
4) 오류가 발생한 모든 테스트 응답에 대해 완전일치공통부분으로 설명하는 고장들이 최종 후보 고장 집합에 포함될 때까지 단계 3을 반복한다. 테스트 응답에 대하여 완전일치공통부분을 가진 후보 고장이 없을 경우에는 넘어간다.

www.dbpia.co.kr

	점수	패턴 1	패턴 2	패턴 3	패턴 4
고장 1	102	▣	▣		▣
고장 2	98	▣	▣		
고장 3	87	▣			
고장 4	50			▣	
고장 5	23		▣		

그림 3. 최종 후보 고장 집합의 결정 예제

이 과정을 그림 3에서 예를 들어 설명하겠다. 예제의 회로는 총 후보 고장이 5개이며, 각 고장의 점수는 Ⅲ장에서 설명한 과정을 거쳐 그림 3와 같이 결정되었다. 그리고 ▣ 표시는 완전일치공통부분을 나타낸 것이다. 여기서 주의해야할 점은 완전일치공통부분은 테스터 응답과 시뮬레이션 응답이 모두 오류가 있어야 하기 때문에, 완전일치공통부분에 저장되는 패턴들은 모두 테스터의 응답에 오류가 나타나게 하는 패턴들로만 구성되어야 한다는 점이다. 단계 1을 거쳐 고장 점수 순으로 정렬하면 그림 3과 같은 순서로 정렬이 된다. 단계 2에서 가장 높은 점수를 가진 고장들 중에서 완전일치공통부분이 가장 많은 고장을 선택한다. 예제에서는 가장 점수가 높은 고장이 후보 고장 1이므로 하나만 선택하면 되나, 다수가 있을 경우에는 완전일치공통부분이 가장 많은 고장을 선택하면 되고 이 수도 동일하다면 모두 다 최종 후보 고장 집합에 포함시킨다. 다음 단계 3에서는 패턴 3에 대해서 완전일치공통부분이 나타나지 않으므로 패턴 3에서 완전일치공통부분이 나타나는 고장 4를 최종 후보 고장 집합에 포함시킨다. 최종 후보 고장 집합에는 고장 1과 고장 4가 포함되고 이를 고장 진단 결과로 출력한다.

단순히 점수가 높은 고장을 선택하는 방식으로는 다중 고착 고장을 진단할 수 없다. 한 개의 고장은 1순위로 진단되더라고, 다른 고장이 대부분의 경우에 낮은 순위에 위치하기 때문이다. 이를 보완하기 위하여 완전일치공통부분 테이블을 사용하여, 낮은 순위에 있더라도 고장 후보 집합의 고장들로 설명하지 못하는 패턴에 대해 완전일치공통부분이 나타난다면 최종 고장 후보 집합에 포함될 수 있도록 하였다. 이러한 방법은 다중 고착 고장 진단 시에 유용함을 Ⅵ장의 실험을 통해서 입증하였다. 또한 고장들을 세밀하게 식별할 수 있는 매칭 알고리듬을 사용함으로써 SLAT 알고리듬[9]이 가지고 있는 후보 고장의 수가 커지는 문제점을 해결할 수 있다.

Ⅵ. 실험 결과

Ⅱ장에서 설명한 전체 고장 진단 과정을 툴로 작성하여 고장 진단 알고리듬의 성능을 확인해 보았다. 고장 진단 툴의 이름을 MAID(Matching Based Advanced Integrated Diagnosis)라 명명하였다. 모든 실험은 SUN BLADE 2000 워크스테이션에서 수행하였다. 실험을 위하여 ISCAS85, ISCAS89 벤치마크 회로들을 삼성 STD150 라이브러리를 사용하여 합성하였으며, ISCAS89 회로는 전체 주사 회로가 되도록 합성하였다. 테스트 패턴은 Synopsys 사의 TetraMAX를 사용하여 생성하였다. 30개의 회로에 임의로 고장을 두 개씩 삽입한 경우와 세 개씩 삽입한 경우에 대해서 실험을 하였다. 이렇게 고장을 삽입한 회로를 논리 시뮬레이션을 수행한 결과를 테스터의 응답이라 가정하고, 이 결과로부터 삽입한 고장을 진단해내는가의 여부로 고장 진단 알고리듬을 검증하였다.

ISCAS85회로에 대한 두 개의 고장이 삽입되었을 경우의 실험 결과는 표 1에 정리하였다. 진단 고장수가 나타내는 의미는 고장이 2개 삽입된 회로에서 평균적으로 몇 개의 고장이 진단되었는지를 나타내는 것이다. 후보 고장 수는 최종적으로 진단된 후보 고장의 수를 나타낸다. 비교한 SLAT 알고리듬[9]이 원래의 논문과 다른 점은 원래의 논문에서는 고착 고장의 극성에 상관없이 위치만으로 진단하였으나, 비교 실험을 위해서 고착 고장을 기준으로 실험을 하였다. 그리고 SLAT 알고리듬은 multiplet이라고 부르는 후보 고장 집합을 여러 개

표 1. 두 개의 고장이 삽입된 ISCAS85회로의 고장 진단 결과

회로	SLAT[9]		MAID	
	진단 고장수	후보 고장수	진단 고장수	후보 고장수
c432	1.67	9.77	1.73	4.20
c499	1.23	31.27	1.37	2.73
c880	1.90	9.57	1.93	2.67
c1355	1.13	19.93	1.13	3.40
c1908	1.20	10.07	1.63	2.30
c2670	1.83	21.97	1.80	2.60
c3540	1.77	9.53	1.83	3.17
c5315	1.83	14.03	1.83	3.03
c6288	1.70	27.23	1.70	3.53
c7552	1.83	37.83	1.83	3.13
평균	1.61	19.12	1.68	3.08

www.dbpia.co.kr

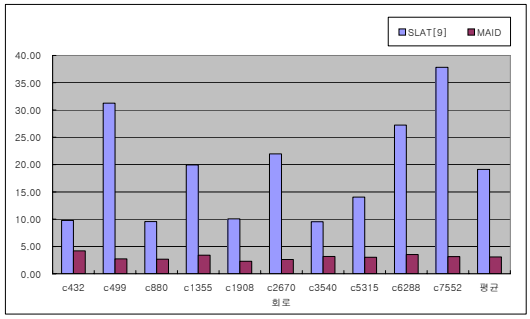

그림 4. 두 개의 고장이 삽입된 ISCAS85회로에 대한 최종 고장 후보 수

표 2. 두 개의 고장이 삽입된 ISCAS89회로의 고장 진단 결과

회로	SLAT[9]		MAID	
	진단 고장수	고장 후보수	진단 고장수	고장 후보수
s1196	1.77	11.53	1.93	2.93
s1238	1.77	13.87	2.00	3.20
s1488	1.70	12.00	1.80	3.47
s1494	1.77	12.47	1.73	3.73
s5378	1.90	8.67	1.90	2.90
s9234	1.93	11.97	1.93	3.87
s13207	1.83	15.63	1.83	4.40
s15850	1.97	10.80	1.97	3.03
s35932	1.93	11.80	1.93	3.47
s38594	1.97	9.30	1.97	2.67
평균	1.85	11.80	1.90	3.37

생성하기 때문에 정확한 비교를 위해서 이들 후보 고장 집합들에 포함된 고장의 수를 중복되지 않게 세어 정리하였다.

표 1을 보면 진단된 고장의 수는 SLAT 알고리듬에서 평균 1.61, MAID에서 평균 1.68로 고장 진단의 정확도는 거의 차이 없거나 MAID 알고리듬이 조금 더 좋다고 말할 수 있다. 그러나 후보 고장 집합의 수는 평균 19.12와 평균 3.08로 SLAT 알고리듬의 1/6 밖에 되지 않음을 알 수 있다. 이는 MAID 알고리듬의 결과가 훨씬 좁게 고장의 범위를 줄이므로 효율적이라고 말할 수 있다. 이러한 차이점은 그림 4에서 좀 더 두드러지게 관찰 할 수 있다.

MAID 알고리듬은 조합회로뿐만 아니라 보다 크기가 큰 전체 주사 회로에서도 뛰어난 성능을 보여준다. 표 2에 이러한 결과를 정리하였다. 진단된 고장의 수는 SLAT 알고리듬에서 평균 1.85, MAID 알고리듬에서 평균 1.90으로 정확도는 조합회로와 마찬가지로 MAID 알고리듬이 비슷하거나 조금 더 뛰어나다고 말할 수 있다.

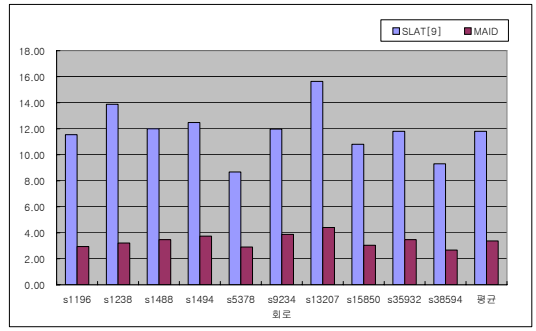

그림 5. 두 개의 고장이 삽입된 ISCAS89회로에 대한 최종 고장 후보 수

전체 주사 회로에 대한 최종 후보 고장 집합의 수는 SLAT 알고리듬에서는 평균 11.80, MAID 알고리듬에서는 평균 3.37로 제안한 알고리듬이 SLAT 알고리듬에 비해 1/4정도인 것을 관찰할 수 있다. 이러한 차이는 그림 5를 통하여 보다 쉽게 관찰할 수 있다.

세 개의 고착 고장을 삽입한 회로에서 고장 진단 수행결과를 정리해 보았다. 삽입된 고장이 많아질수록 고장 간에 영향을 서로 미칠 가능성이 높아지고 단일 고장과의 출력단의 응답의 차이가 커지므로 고장 진단이 어려워진다. 정리한 실험 결과를 관찰해 보면 MAID 알고리듬은 세 개의 고장이 삽입된 경우에도 보다 정확하게 고장 진단을 수행함을 관찰할 수 있다.

조합 회로에 대하여 수행한 실험 결과는 다음과 같이 표 3에 정리하였다. SLAT 알고리듬은 평균 2.01이고 제안한 알고리듬 2.13이므로 MAID 알고리듬이 보다 더 정확한 고장진단을 수행한다고 말할 수 있다.

표 3. 세 개의 고장이 삽입된 ISCAS85회로의 고장 진단 결과

회로	SLAT[9]		MAID	
	진단 고장수	고장 후보수	진단 고장수	고장 후보수
c432	2.03	11.93	2.17	5.77
c499	1.30	19.33	1.40	3.83
c880	2.33	14.17	2.37	4.03
c1355	0.97	7.37	0.97	3.93
c1908	1.43	11.27	1.90	3.77
c2670	2.30	30.43	2.30	3.77
c3540	2.43	14.57	2.60	4.63
c5315	2.57	15.43	2.60	3.93
c6288	2.33	56.03	2.60	5.00
c7552	2.40	31.57	2.43	4.30
평균	2.01	21.21	2.13	4.30

www.dbpia.co.kr

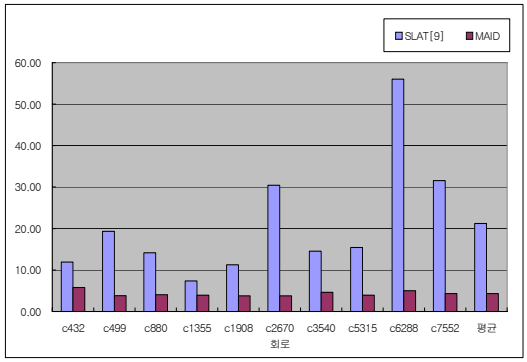

그림 6. 세 개의 고장이 삽입된 ISCAS85회로에 대한 최종 고장 후보 수

표 4. 세 개의 고장이 삽입된 ISCAS89회로의 고장 진단 결과

회로	SLAT[9]		MAID	
	진단 고장수	고장 후보수	진단 고장수	고장 후보수
s1196	2.23	13.63	2.83	4.37
s1238	2.30	17.00	2.87	5.20
s1488	2.27	16.17	2.47	5.23
s1494	2.27	14.50	2.47	6.77
s5378	2.73	14.23	2.77	4.17
s9234	2.73	17.83	2.77	5.27
s13207	2.72	12.41	2.72	4.24
s15850	2.77	14.10	2.77	4.33
s35932	2.53	12.67	2.53	3.73
s38594	2.87	13.90	2.90	4.30
평균	2.54	14.64	2.71	4.76

또한 SLAT 알고리듬이 평균 고장 후보 집합의 수가 21.21이고 제안한 알고리듬이 평균 4.30으로 약 1/5로 후보 고장의 수를 줄인 것을 확인 할 수 있다. 이 결과를 그림 6에서와 같이 막대그래프로 정리하면 보다 확연하게 관찰 할 수 있다.

마지막으로 수행한 실험은 전체 주사 회로 버전의 ISCAS89 10개의 회로에 대하여 세 개의 고착 고장을 삽입하여 실험을 수행하였다. 그 결과는 표 4에 정리하였다.

이번 실험에서도 전체적인 경향은 이전에 수행했던 실험들과 동일하다. MAID 알고리듬이 SLAT 알고리듬보다 보다 고장을 정확하게 진단하며 평균 고장 후보 집합의 수가 14.64에서 4.76로 1/3가량 감소된 것을 확인 할 수 있다. 평균 고장 후보 집합의 수를 막대 그래프로 나타낸 그림 7을 관찰하면 그 차이를 보다 확연하게 관찰 할 수 있다.

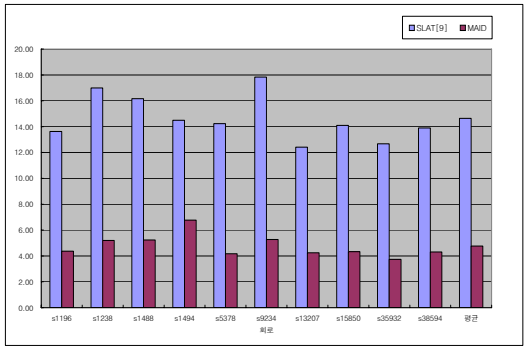

그림 7. 세 개의 고장이 삽입된 ISCAS89회로에 대한 최종 고장 후보 수

이상으로 실험에서 MAID 알고리듬은 기존의 알고리듬보다 정확도는 증가하며, 진단한 후보 고장의 수는 감소됨을 확인할 수 있었다. SLAT알고리듬의 오류 테이블만을 사용하여 다중 고착 고장을 정확히 진단하기는 어려우며 고장간의 식별성을 높이는 작업을 수행한 MAID 알고리듬에서 이러한 문제점을 보완하여 높은 성능을 보임을 확인할 수 있었다.

Ⅶ. 결 론

본 논문에서는 효율적인 다중 고착 고장 알고리듬을 제안하였다. 제안한 MAID 알고리듬은 단일 고착 고장 시뮬레이터 환경에서도 다중 고착 고장을 진단할 수 있도록 한다. 다중 고착 고장 진단을 위하여 완전일치공통부분의 정보를 저장하여 사용하였다. 또한 각 고장간의 식별성을 높여 다중 고착 고장을 진단함에도 불구하고, 고장 후보의 수를 획기적으로 줄일 수 있었다. 이를 위하여 출력단의 수에 따른 가중치 개념과 가산, 감산 연산을 사용하였다. ISCAS85회로와 전체 주사 기법이 적용된 ISCAS89 회로에 대한 고장을 진단한 실험으로 MAID 알고리듬이 효율적이고 정확하게 고장 진단을 함을 확인할 수 있었다.

그러나 제안한 MAID 알고리듬도 한계를 가지고 있다. 완전일치공통부분 테이블은 한 개의 패턴이 고장을 활성화시키고 출력단으로 오류가 존재하는 값이 전파 되었을 때에 큰 의미를 지닌다. 이번에 실험을 진행하였던 조합 회로와 전체 주사 회로에서는 이러한 조건이 만족되므로 좋은 결과를 얻을 수 있었다. 그러나 순차 회로에서는 완전일치공통부분 테이블의 의미가 퇴색하므로 지금과 같은 좋은 결과가 나타나지 않을 거라고 예상된다. 순차 회로

www.dbpia.co.kr

에서도 다중 고착 고장을 진단하기 위해서는 보다 향상된 고장 진단 알고리듬이 제안되어야 할 것이다.

참 고 문 헌

〔1〕 Y. Takamatsu, T. Seiyama, H. Takahashi, Y. Higami and K. Yamazaki, "On the fault diagnosis in the presence of unknown fault models using pass/fail information," ISCAS 2005. IEEE International Symposium, pp. 2987 - 2990. 2005.

〔2〕 H. Takahashi, K.O. Boateng, K.K. Saluja and Y. Takamatsu, "On diagnosing multiple stuck-at faults using multiple and single fault simulation in combinational circuits," IEEE Transactions on CAD of Integrated Circuits and Systems, pp. 362 -368, 2002.

〔3〕 Wu Jue and E. M. Rudnick, "A diagnostic fault simulator for fast diagnosis of bridge faults," Proc. of VLSI Design, pp. 498 - 505, 1999.

〔4〕 S. Venkataraman, I. Hartanto and W. K. Fuchs, "Dynamic Diagnosis of Sequential Circuits Based on Stuck-at Faults," Proc. of VLSI Test Symposium, pp. 198-203, 1996.

〔5〕 V. Boppana and M. Fujita, "Modeling the unknown! Towards model-independent fault and error diagnosis," Proc. of International Test Conference, pp. 1094-1101, 1998.

〔6〕 S. Venkataraman and S. Drummonds, "Poirot : Applications of a Logic Fault Diagnosis Tool," IEEE Design & Test of Computers, pp. 19-30, 2001.

〔7〕 P. Goel, et al, "LSSD Fault Simulation Using Conjunctive Combinational and Sequential Methods", Proc. of International Test Conference, pp. 371-376, 1980.

〔8〕 T. Bartenstein, D. Heaberlin, L. Huisman and D. Sliwinski, "Diagnosing combinational logic designs using the single location at-a-time (SLAT) paradigm," Proc. of International Test Conference, pp. 287-296, 2001.

〔9〕 L. M. Huisman, "Diagnosing arbitrary defects in logic designs using single location at a time (SLAT)," IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, pp. 91-101, 2004.

임 요 섭 (Yoseop Lim) 정회원

2004년 2월 : 연세대학교 전기전자공학부 졸업

2006년 2월 : 연세대학교 전기전자공학과 석사

2006년 3월~현재 : 연세대학교 전기전자공학과 박사과정

<관심분야> Diagnosis, CAD, DFT

강 성 호 (Sungho Kang) 정회원

1986년 2월 : 서울대학교 제어계측공학과 졸업

1988년 6월 : The University of Texas, Austin 전기 및 컴퓨터공학과 석사

1992년 6월 : The University of Texas, Austin 전기 및 컴퓨터공학과 박사

1992년 미국 Schlumberger Inc. 연구원

1994년 Motorola Inc. 선임 연구원

2007년 현재 연세대학교 전기전자공학과 교수

<관심분야> SoC 설계 및 SoC 테스트

www.dbpia.co.kr