論文2002-39SD-9-6

Test-per-clock 스캔 방식을 위한 효율적인 테스트 데이터 압축 기법에 관한 연구

(A Study on Efficient Test Data Compression Method for Test-per-clock Scan)

朴 在 興 *, 梁 善 雄 *, 張 勳 **

(Jae-Heung Park, Sun-Woong Yang, and Hoon Chang)

요 약

본 논문에서는 SOC의 내장된 코어를 테스트하기 위한 새로운 DFT 방법인 순차적 테스트 데이터 압축 방법을 제안한다. 순차적 테스트 데이터 압축 방법은 테스트 데이터양을 줄이기 위하여 공유 비트 압축과 고장 무검출 패턴 압축 방법을 이용하였다. 그리고 순차적 테스트 데이터 압축 방법을 이용하는 회로는 스캔 DFT 방법을 기반으로 하고 있으며, test-per-clock 방법을 적용하여 매 클럭마다 테스트 할 수 있는 구조를 가지고 있다. 제안된 압축 방법의 실험을 위하여 벤치마크 회로인 ISCASS85와 ISCASS89 완전 스캔 버전을 이용하였으며, ATPG와 고장 시뮬레이션을 위하여 ATALANTA를 사용하였다. 실험 결과 순차적 테스트 데이터 압축 방법의 테스트 데이터의 양이 스캔 DFT를 적용한 회로에 비해 최대 98% 까지 줄어듦을 확인하였다.

Abstract

This paper proposes serial test data compression, a novel DFT scheme for embedded cores in SOC. To reduce test data amounts, share bit compression and fault undetectable fault pattern compression techniques was used. A Circuits using serial test data compression method are derived from a scan DFT method including a test-per-clock technique. For an experiment of the proposed compression method, full scan versions of ISCASS85 and ISCASS89 were used. ATALANTA has been used for ATPG and fault simulation. The amount of test data has been reduced by maximum 98% comparing with original data.

Key Words : scan, test-per-clock, DFT, data compression, SOC

Ⅰ. 서 론

최근 하나의 칩 안에 2천만 개 이상의 트랜지스터를 집적할 수 있는 초고집적 반도체 설계 및 제조가 가능하게 되었다. 또한 이미 설계되어 검증된 회로와 사용자가 디자인한 회로를 이용하여 SOC(System On Chip) 설계가 가능하게 되었다. SOC는 재사용 가능한 회로를 사용하여 설계하기 때문에 새로운 시스템을 개발하는 시간을 많이 줄일 수 있는 장점이 있다. 하지만 SOC의 복잡도가 증가함에 따라, 칩을 테스트하기 위한 비용도 급격하게 증가하게 된다.[1,2]

그림 1은 ATE(Automatic Test Equipment)와 SOC 사이의 연결 상태를 보여주고 있다. SOC를 테스트하기

* 正會員, 崇實大學校 컴퓨터學科
(Department of Computing, Graduate School, Soongsil University)
** 正會員, 崇實大學校 컴퓨터學部
(School of Computing, Soongsil University)
接受日字:2001年11月19日, 수정완료일:2002年8月19日

위해서는 칩에 내장되어 있는 모든 코어들의 테스트 데이터를 ATE의 메모리에 저장하고 있어야 하며 테스트를 수행하는 동안 테스트 데이터를 칩에 인가하고, 테스트 결과를 ATE에 저장하여야 한다.

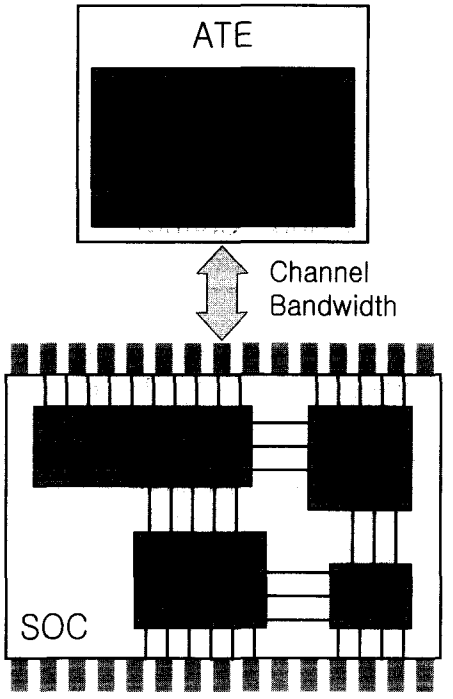


그림 1. SOC와 ATE의 연결
Fig. 1. SOC and ATE.

테스트 비용은 테스트 데이터의 양, 데이터를 SOC에 빠르게 전송할 수 있는 ATE의 데이터 전송 속도, ATE의 메모리 크기 등에 의해 결정되어 진다. 하지만 ATE의 동작 속도, 메모리 크기, 데이터 전송 속도 등의 증가에 따라 급격하게 증가하므로 ATE의 성능을 향상시킴으로써 테스트 비용을 줄이는 데는 한계가 있다[3]. 따라서 테스트 비용에 직접적으로 영향을 미치는 테스트 시간을 줄이기 위하여 테스트 데이터양을 줄이는 방법이 많이 연구되고 있다.[4~7]

본 논문에서는 테스트 데이터양을 줄이기 위하여 새로운 DFT(Design For Test) 방법인 순차적 테스트 데이터 압축 방법을 제안한다. 제안된 압축 방법은 데이터의 양을 줄이기 위하여 공유 비트 압축 방법과 고장 무검출 패턴 압축 방법을 사용하였다. 공유 비트 압축은 테스트 회로에 인가하는 테스트 패턴의 순서를 재배열함으로써 데이터의 양을 줄이는 방법이고, 고장 무검출 패턴 압축 방법은 테스트 패턴의 고장 검출(fault detection) 유무를 이용하여 테스트 데이터의 양을 줄인다. 그리고 순차적 테스트 데이터 압축 방법을 이용하는 회로는 스캔 DFT 방법을 기반으로 하고 있다. 따라서 회로의 입출력 포트, 테스트 방법, ATE와의 연결 등 모든 것이 스캔 DFT를 적용한 회로와 동일하며, 스캔 DFT를 적용한 회로를 테스트하는 방법 그대로 적용할 수 있다. 다만, 제안된 압축 방법을 이용하는 회로는 test-per-clock 방법을 적용하여 매 클럭마다 테스트를 수행 할 수 있으며, 스캔 DFT를 적용한 회로의 테스트 데이터보다 양이 적다는 것이 차이점이다. 즉, 제안된 압축 방법을 이용하는 회로는 같은 테스트 방법으로 스캔 DFT를 적용한 회로에 비하여 빠른 시간 안에 테스트를 수행할 있다.

본 논문의 구성은 다음과 같다. 2장에서는 제안된 순차적 테스트 데이터 압축 방법을 이용하는 회로의 구조를 설명하고 3장에서는 순차적 테스트 데이터 압축 방법에 대하여 기술한다. 4장에서는 실험 결과를 보여주고, 결론을 5장에 기술하였다.

Ⅱ. 제안된 압축 방법을 이용하는 회로 구조

순차적 테스트 데이터 압축 방법을 이용하는 회로의 구조가 그림 2에 나와 있다. 회로는 CUT(Circuit Under Test)에 테스트 패턴을 인가하기 위하여 스캔 체인을 사용하고, 테스트 결과를 저장하기 위하여 MISR(Multiple Input Signature Register)을 CUT의 출력에 연결하였다. 즉, 스캔 체인은 테스트 패턴을 인가하는데 사용하며, MISR은 테스트 결과를 저장하는데 사용한다. 그리고 MISR는 Scan-out과 연결되어 테스트 결과를 스캔 동작에 의하여 칩 외부로 출력하도록 되어 있다. 따라서 칩 외부에서는 회로를 스캔 DFT 회로와 같은 동작으로 테스트를 수행할 수 있다.

회로의 테스트는 test-per-clock 방법을 사용하고 있다. 따라서 결정적(deterministic) 테스트 패턴을 회로의 스캔 체인에 인가하는 동안 매 클럭마다 무작위 테스트 패턴이 인가된다. 그림 3은 매 클럭마다 스캔 체인에 한 비트씩 데이터가 입력되면서 스캔 체인이 회로에 인가하는 값을 나타내고 있다. 테스트 패턴의 크기는 5 비트이고, t_1, t_2, ..., t_7은 패턴의 인가 시간을 나타낸다. t_1 시간에 결정적 테스트 패턴이 인가된 후 다

음 결정적 테스트 패턴이 인가되는 시간은 t_6 이다. 일반적인 스캔 DFT 방법에서는 t_1에서 결정적 테스트 패턴1을 사용하여 회로를 테스트 한 후 t_2 시간에서 t_5 시간까지는 결정적 테스트 패턴2를 스캔 체인에 인가하기 위하여 테스트가 수행되지 않는다. 그리고 t_7 시간에 결정적 테스트 패턴2를 이용하여 회로를 테스트하게 된다. 하지만 제안된 회로에서는 t_2 - t_5 시간에 스캔 체인이 가지고 있는 값을 무작위(random) 테스트 패턴처럼 이용하여 매 클럭마다 테스트를 수행한다. 즉, t_1 시간에 결정적 테스트 패턴1로 테스트를 수행한 후 t_2 시간에서는 무작위 테스트 패턴1을 이용하여 회로 테스트를 수행하게 된다. 마찬가지로 t_3, t_4, t_5 시간에서는 무작위 테스트 패턴 2, 3, 4로 회로를 테스트한다. 그리고 매 클럭마다 수행된 테스트 결과는 MISR에 저장된다. 테스트가 끝난 후 테스트 결과인 MISR의 값은 Scan-out을 통하여 칩 외부로 내보내게 된다. 결과적으로 t_1 - t_6 시간 동안 스캔 DFT 방법은 두 개만으로 테스트를 수행하지만, 본 논문에서 제안하는 회로 구조는 총 6개의 패턴으로 테스트를 수행하게 된다.

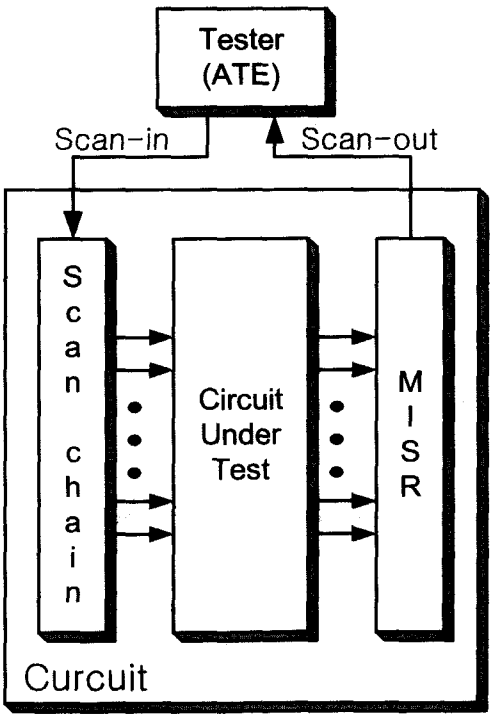


그림 2. 순차적 테스트 데이터 압축 방법을 이용하는 회로 구조
Fig. 2. A circuit architecture using the serial test data compression method.

t_1	**11110**	**결정적 테스트 패턴1**
t_2	11101	무작위 테스트 패턴1
t_3	11010	무작위 테스트 패턴2
t_4	10100	무작위 테스트 패턴3
t_5	01001	무작위 테스트 패턴4
t_6	**10010**	**결정적 테스트 패턴2**
t_7	00101	무작위 테스트 패턴5

그림 3. 스캔 체인에 인가된 테스트 패턴
Fig. 3. Test pattern applied to scan chain.

Ⅲ. 순차적 테스트 패턴 압축 알고리즘

순차적 테스트 데이터 압축 알고리즘은 크게 두 단계로 나누어진다. 첫 번째는 테스트 패턴의 인가 순서를 재배열하여 테스트 패턴간의 중복되는 비트를 제거하는 공유 비트 압축 단계이고, 두 번째는 고장을 검출하지 못하는 패턴의 비트를 제거하는 고장 무검출 패턴 압축 단계이다.

1. 공유 비트 압축

공유 비트 압축은 ATPG(Automatic Test Pattern Generator)를 이용하여 결정적 테스트 패턴을 생성한 후, 생성된 결정적 테스트 패턴의 인가 순서를 재배열함으로써 압축을 수행한다. 테스트 패턴의 재배열은 인접한 두 테스트 패턴 간에 공유 비트가 최대가 되도록 한다.

(1) 패턴 재배열

그림 4는 인접한 두 테스트 패턴의 순서에 따른 공유 비트의 변화를 보여 주고 있다. 두 패턴 p_1(00110), p_2(10001)가 있다고 할 때, 일반적인 스캔 DFT에서 두 패턴은 0011010001 순서로 스캔에 인가하여야 한다. 하지만 제안된 회로에서는 클럭 당 테스트를 수행하기 때문에 p_1, p_2 순서로 패턴이 인가될 경우, p_1 패턴의 마지막 2 비트(10)와 p_2의 처음 2 비트가 같으므로 그림 4의 (a)와 같이 p_1 패턴의 뒷부분 비트와 p_2의 앞부분 비트를 공유하여 00110001로 인가할 수 있다. 즉, 두 패턴의 공유 비트에 의하여 2 비트를 줄일 수 있다. 이렇게 제안된 회로가 test-per-clock 방법을 이용하기 때문에 인접한 패턴 간에 같은 비트를 공유함으로써 테스트 데이터의 양을 줄일 수 있다.

여기에서, 그림 4의 (b)와 같이 p_2, p_1 순서로 패턴을

인가하는 경우를 생각해 보자. p_2, p_1 순서로 패턴이 인가되는 경우, p_2 패턴의 뒷부분 3 비트(001)가 p_1의 앞부분 3 비트와 같기 때문에 1000110으로 인가할 수 있다. 따라서 p_2, p_1 순서로 패턴을 인가할 경우 3 비트를 줄일 수 있다. 즉, p_2, p_1의 순서보다는 p_2, p_1 순서로 테스트 패턴을 재배열하여 인가하는 것이 더 많은 테스트 패턴의 양을 줄일 수 있음을 알 수 있다. 물론, 두 결정적 테스트 패턴이 모두 인가되기 때문에 패턴의 인가 순서에 상관없이 고장 검출률(fault coverage)은 같다.

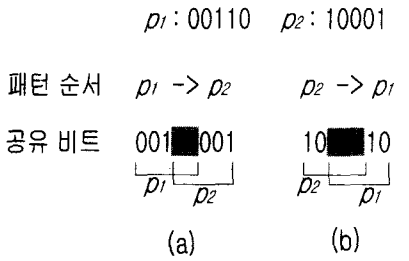


그림 4. 테스트 패턴의 인가 순서에 따른 공유 비트 변화

Fig. 4. Sharing bit change for test pattern input sequence.

(2) 공유 비트 압축 구현

인접한 테스트 패턴간의 공유 비트가 최대가 되도록 패턴의 순서를 재배열하는 문제는 최소의 비용으로 모든 도시를 한번씩만 순회하는 순회 외판원 문제(Travelling Salesman Problem)와 같다. 순회 외판원 문제에 적용하기 위하여 테스트 패턴의 재배열 문제는 무게달린 방향 그래프(weighted digraph) G = (V, E, W)로 표현할 수 있다.

V = {v_i | v_i는 테스트 패턴}

E = {e_{ij} | v_i에서 v_j로 연결되는 방향 간선(edge)}

W = {w_{ij} | e_{ij} 간선의 비용}

여기서, w_{ij}는 아래와 같이 구해진다.

w_{ij} = 패턴의 비트 수−v_i와 v_j 사이의 공유 비트 수

그림 5는 6개의 테스트 패턴들을 무게달린 방향 그래프로 표현한 것을 보여주고 있다.

최소의 비용을 가지는 순회 외판원 문제는 NP-complete 문제로 알려져 있다.[8] 따라서 테스트 패턴

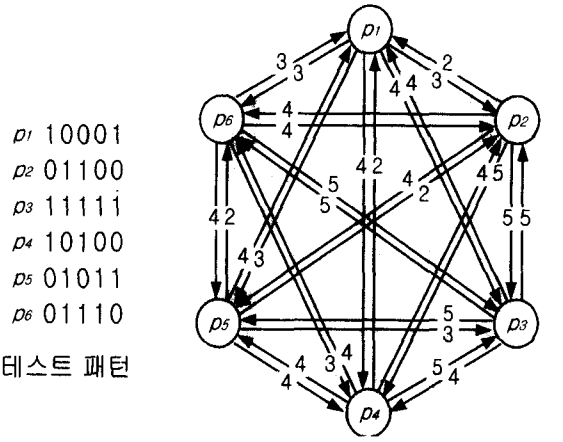


그림 5. 테스트 패턴의 무게달린 방향 그래프

Fig. 5. Weighted digraph of test pattern.

재배열 문제를 해결하기 위하여 구현이 간단한 kruskal 알고리즘을 이용하였다.[9] Kruskal 알고리즘은 최소-비용 신장 트리(minimum spanning tree)를 구하는 greedy 방법 중의 하나로 무게달린 방향 그래프에서 비용이 가장 적은 간선부터 선택하여 트리를 구성한다. 그림 6은 테스트 패턴을 재배열하기 위하여 그림 5의 그래프를 kruskal 알고리즘을 이용하여 최소-비용 신장 트리로 생성하는 것을 보여 주고 있다. 그림 6에서와 보는 바와 같이 테스트 패턴은 P_3, p_5, p_2, p_1, p_6, p_4 순서로 재배열된다.

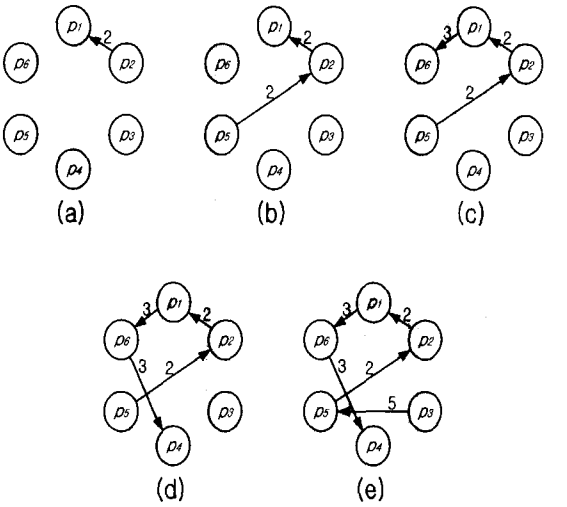


그림 6. Kruskal 알고리즘을 이용한 최소-비용 신장 트리 생성

Fig. 6. Generation of minimum spanning tree using kruskal algorithm.

2. 고장 무검출 패턴 압축

고장 무검출 패턴 압축은 공유 비트 압축 방법에 의해 생성된 테스트 패턴이 회로의 스캔 체인에 한 비트씩 인가되면서 생성되는 테스트 패턴에 대하여 고장 시뮬레이션을 수행하고, 고장을 검출하지 못하는 패턴의 비트를 제거하는 방법이다.

그림 7은 그림 6과 같이 재배열되어 공유 비트 압축이 이루어진 패턴이 실제 회로의 스캔 체인에 한 비트씩 인가됨에 따라 테스트를 수행하는 테스트 패턴을 보여 주고 있다. t_1, t_2, ... t_{19}, t_{20}은 회로의 스캔 체인에 Scan-in 패턴이 한 비트씩 인가되는 시간을 나타낸다. 테스트 패턴 p_3, rp_1, ... rp_{10}, p_4는 회로를 테스트하는 실제 테스트 패턴이다. 그리고 p_1, p_2, p_3, p_4, p_5, p_6은 결정적 테스트 패턴을, rp_1, rp_2, rp_3, rp_4, rp_5, rp_6, rp_7, rp_8, rp_9, rp_{10}은 무작위 테스트 패턴을 나타낸다. 즉, 공유 비트 압축에 의해 생성된 패턴이 회로의 스캔 체인에 한 비트씩 인가되면서 6개의 결정적 테스트 패턴과 10개의 무작위 테스트 패턴을 만들게 된다. 그리고 이렇게 만들어지는 패턴 중 고장을 검출하지 못하는 패턴을 찾기 위하여 그림 7의 테스트 패턴에 대하여 고장 시뮬레이션을 수행한다. 표 1은 그림 7 테스트 패턴에 대한 고장 시뮬레이션 결과를 보여주고 있다. 고장 시뮬레이션 결과 p_3, rp_1, rp_2, rp_7, rp_8, rp_6, rp_9는 고장을 검출하는 패턴이고, 나머지 패턴은 고장을 검출하지 못하는 패턴임을 알 수 있다.

표 1. 고장 시뮬레이션 결과
Table 1. Fault simulation result.

패턴	고장 검출 개수	패턴	고장 검출 개수
p_3	9	rp_6	0
rp_1	5	p_1	0
rp_2	3	rp_7	1
rp_3	0	rp_8	3
rp_4	0	p_6	5
p_5	0	rp_9	2
rp_5	0	rp_{10}	0
p_2	0	p_4	0

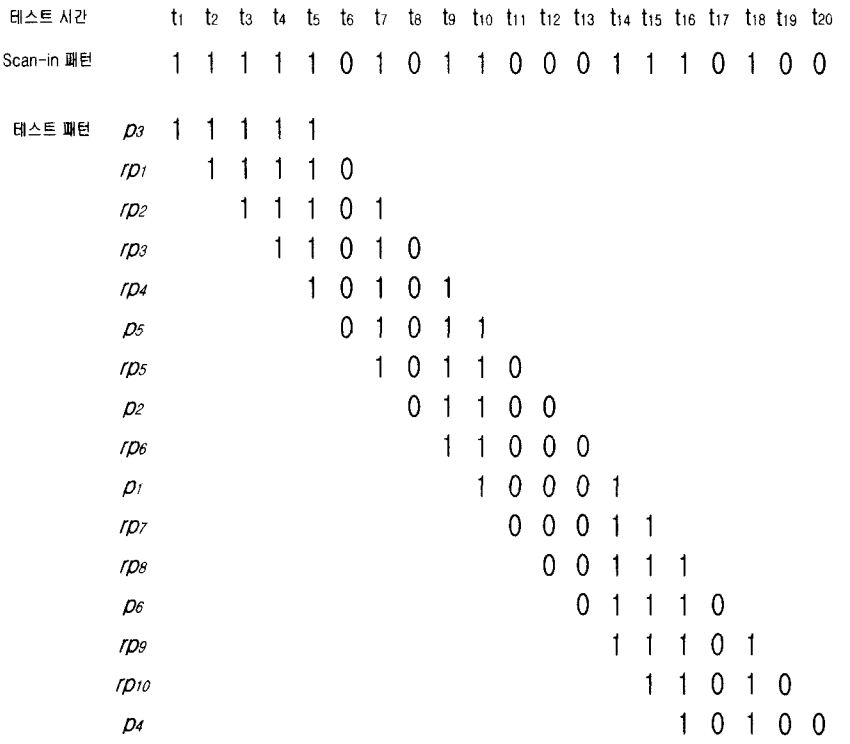


그림 7. 테스트 패턴
Fig. 7. Test pattern.

그림 8은 고장 무검출 패턴에 의하여 필요 없는 비트를 제거하는 것을 보여 주고 있다. Scan-in 패턴을 기준으로 윗부분의 패턴은 고장을 검출하지 못하는 패턴이고, 아랫부분은 고장을 검출하는 패턴이다. 여기에서, 고장을 검출하지 못하는 패턴일지라도 고장을 검출하는 패턴과 비트를 일부분 공유하고 있기 때문에 고장을 검출하지 못하는 패턴의 모든 비트를 제거할 수는 없다. rp_2와 rp_3 패턴을 보면 4 비트(1101)를 공유하고 있으므로 rp_3 패턴의 모든 비트를 제거할 수 없고 마지막 비트만을 제거 할 수 있다. 이런 방법으로 그림 8에서 보는 바와 같이 고장을 검출하는 패턴과 비트를 공유하지 않는 rp_2와 rp_7 사이의 3 비트와 rp_9 패턴 뒤의 2 비트, 총 5 비트를 제거할 수 있다. 그리고 그림 8의 Scan-in 패턴의 중간 부분에서 3 비트가 제거되었기 때문에 새로운 Scan-in 패턴이 생성된다. 새로운 Scan-in 패턴을 회로에 인가할 경우 그림 9에서와 같이 새로운 무작위 테스트 패턴 np_1, np_2, np_3, np_4 이 생성되고, 새롭게 생성된 패턴에 의하여 고장 시뮬레이션 결과가 달라질 수 있다. 그로 인하여 고장 무검출 패턴에 의한 비트 제거가 다시 발생할 수 있다. 따라서 고장 무검출 패턴에 의한 비트 제거가 발생하지 않을 때까지 고장 무검출 패턴 압축을 반복한다.

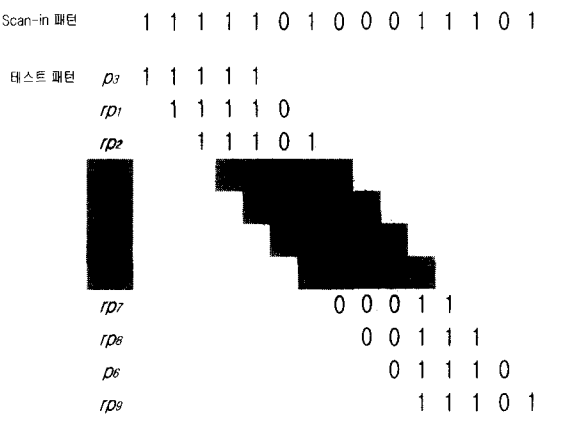


그림 9. 고장 무검출 패턴 압축에 의해 생성된 패턴
Fig. 9. Generated pattern by undetectable fault pattern compression.

Ⅳ. 실험 결과

순차적 테스트 데이터 압축 방법을 실험하기 위하여 벤치마크 회로는 ISCASS85[10]와 ISCASS89[11] 완전 스캔(full scan) 버전을 이용하였다. 결정적 테스트 패턴을 생성하는 ATPG와 고장 검출 유무를 확인하는 고장 시뮬레이션은 Virginia Ploytechinic Institute & State University의 하동삼 교수 연구실에서 개발한 ATALANTA[12]를 이용하였다.

테스트 데이터를 줄이기 위한 공유 비트 압축과 고

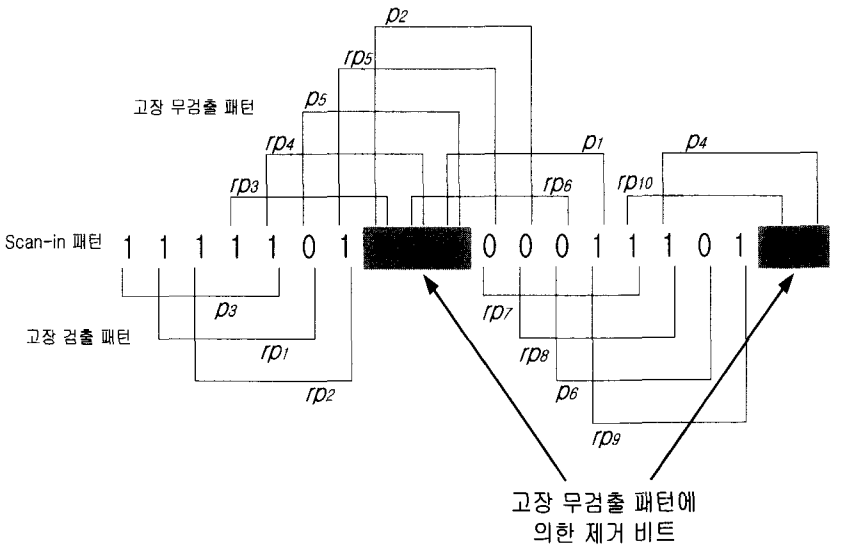


그림 8. 고장 무검출 패턴의 비트 제거
Fig. 8. Drop bit in undetectable fault pattern.

장 무검출 패턴 압축은 SUN Ultra 5 워크스테이션에서 수행하였다. n개의 테스트 패턴을 재배열하기 위하여 그래프로 표현하였을 경우 정점의 개수는 n이고, 간선의 개수는 n^2이다. 최소-비용 신장 트리를 구성할 경우, 모든 간선에서 최소의 무게를 가지는 간선을 선택해야 하고, 모든 정점이 연결되어야 하므로 구현한 공유 비트 압축 프로그램의 시간 복잡도는 $O(n^3)$이다. 그리고 Scan-in 패턴의 총 비트 수를 m이라 할 경우 테스트 패턴의 개수가 [m - (스캔 사이즈)] 이므로, 고장 무검출 패턴 압축 프로그램의 시간 복잡도는 $O(m)$이다. 표 2는 각 단계의 압축에 대한 수행 시간을 나타내고 있다. Scan Size는 회로의 스캔의 길이를 나타내고, Pattern Number는 스캔 DFT를 적용한 회로에 대하여 100%의 고장 검출률을 보장하는 테스트 패턴의 개수를 나타낸다. Test Data는 Scan Size x Pattern Number의 개수로써 스캔 DFT를 적용한 회로에서 필요한 테스트 데이터의 양을 나타낸다. 그리고 공유 비트 압축 수행 시간은 공유 비트 압축을 수행한 시간을 나타낸다. 고장 무검출 패턴 압축의 수행 시간은 고장 무검출 패턴 압축을 반복 수행한 총 시간을 나타내며, 반복 횟수는 고장 무검출 패턴 압축이 일어나지 않을 때까지 고장 무검출 패턴 압축을 수행한 횟수를 나타낸다. 총 압축 수행 시간은 공유 비트 압축 수행 시간과 고장 무검출 패턴 압축 수행 시간의 합이다.

표 3은 ISCASS85와 ISCASS89의 실험 결과를 보여주고 있다. Scan Size, Pattern Number, Test Data는 표 2 와 같은 값이고, Proposed Method의 Share bit Reduction은 공유 비트 압축에 의하여 줄어든 비트 수이고, Fault Drop Reduction은 고장 무검출 패턴 압축에 의하여 줄어드는 비트 수를 보여준다. Test Data

표 2. ISCASS85와 ISCASS89 STDC 수행 시간
Table 2. STDC operation time of ISCASS85 and ISCASS89.

Circuit Name	Scan Size	Pattern Number	Test Data (bits)	공유 비트 압축 수행 시간 (sec)	고장 무검출 패턴 압축		총 압축 수행 시간 (sec)
					수행 시간 (sec)	반복 횟수	
c880	60	52	3120	0.083208	0.090540	1	0.173748
c1355	41	86	3526	0.246888	0.129377	2	0.376265
c2670	233	105	24465	1.574936	5.519273	2	7.094209
c3540	50	150	7500	1.455082	0.253242	2	1.708324
c5315	178	115	20470	0.810389	2.954760	1	3.765149
c6288	32	29	928	0.016227	0.022497	1	0.038724
c7552	207	211	43677	4.934378	7.000719	2	11.935097
s382	24	35	840	0.021900	0.009834	1	0.031734
s400	24	33	792	0.024580	0.014897	1	0.039477
s510	25	59	1475	0.101929	0.026673	2	0.128602
s641	54	55	2970	0.062147	0.083802	1	0.145949
s820	23	118	2714	0.609294	1.070824	2	1.677088
s953	45	87	3915	0.253693	1.190972	2	1.444665
s1196	32	137	4384	2.582307	0.241051	3	2.823358
s1238	32	151	4832	2.420989	0.169562	1	2.590551
s1423	91	67	6097	0.095396	0.415523	1	0.510919
s5378	214	247	52858	7.961278	9.313128	1	17.274406
s9234	247	361	89167	25.564269	29.957097	2	53.521366
s13207	700	461	322700	43.350673	187.205792	1	203.556465
s35932	1763	65	114595	0.245900	127.573596	1	127.819496

Reduction은 스캔 DFT를 적용한 회로의 테스트 데이터가 순차적 테스트 데이터 압축에 의하여 줄어든 양을 퍼센트로 나타내고 있다. Test Data Reduction의 값은 아래의 식에 의해서 구해진다.

$$\text{Test Data Reduction} = \frac{\text{Share bit Reduction + Fault Drop Reduction}}{\text{Scan DFT Test Data}} \times 100$$

Test Data Reduction 값을 보면 스캔 DFT에 비하여 최소 38 %에서 최대 98 %까지 테스트 데이터의 양이 줄어듦을 확인할 수 있다.

표 4는 본 논문에서 제안한 방법과 MinTest[13]의 실험 결과를 비교하여 보여주고 있다. MinTest[13]는 무해 벡터 제거와 기본 고장 제거 방법을 이용하여 조합 회로에 대한 최소의 압축된 테스트 패턴을 생성해 준다. 표 4의 Test Data Reduction을 보면 테스트 데이터가 최대 75%까지 줄어듬을 확인 할 수 있다. 그리고, c880, c2670, c7552, s1423, s9234의 경우 MinTest[13] 방법이 더 적은 테스트 데이터를 생성하지만 수행 시간에서는 본 논문에서 제안한 방법이 월등히 빠름을 알 수 있다.

순차적 테스트 데이터 압축 방법을 이용한 회로의 면적 오버헤드 스캔 DFT 회로의 면적과 비교하여 추가되는 MISR의 면적만큼만 증가한다. 따라서 각 테스트 회로에 대하여 플립플롭(flip-flop) 면적 x 출력 포

표 3. 실험 결과
Table 3. Experimental result.

Circuit Name	Scan Size	Scan DFT		Proposed Method		Test Data Reduction (%)
		Pattern Number	Test Data (bits)	Share bit Reduction (bits)	Fault Drop Reduction (bits)	
c880	60	52	3120	260	1754	64.551
c1355	41	86	3526	564	2105	75.695
c2670	233	105	24465	598	8708	38.038
c3540	50	150	7500	1085	4287	71.627
c5315	178	115	20470	811	17648	90.176
c6288	32	29	928	160	666	89.009
c7552	207	211	43677	1671	21119	52.178
s382	24	35	840	218	395	72.976
s400	24	33	792	184	382	71.465
s510	25	59	1475	363	745	75.119
s641	54	55	2970	310	1459	59.562
s820	23	118	2714	783	748	56.411
s953	45	87	3915	576	1565	54.687
s1196	32	137	4384	920	1451	54.083
s1238	32	151	4832	1049	1481	52.359
s1423	91	67	6097	418	3562	65.278
s5378	214	247	52858	1975	38567	76.700
s9234	247	361	89167	2577	40752	48.593
s13207	700	461	322700	4205	261329	82.285
s35932	1763	65	114595	442	112181	98.279

표 4. 제안한 방법과 MinTest[13]의 비교
Table 4. Comparison of proposed method and MinTest.

Circuit Name	MinTest [13]		Proposed Method		Test Data Reduction (%)
	Test Data (bits)	수행시간 (sec)	Test Data (bits)	수행 시간 (sec)	
c880	960	**50.9**	1106	**0.173748**	**115.2**
c1355	3444	29.4	857	0.376265	75.1
c2670	6908	73.3	15159	**7.094209**	**219.4**
c3540	4200	1372.9	2128	1.708324	49.3
c7552	15038	**794.7**	20887	11.935097	138.9
s820	2139	34.1	1183	1.677088	44.7
s953	3420	30.3	1774	1.444665	48.1
s1196	3616	43.6	2013	2.823358	44.3
s1238	3872	127.4	2302	2.590551	40.5
s1423	2002	**205.3**	2117	**0.510919**	**105.7**
s5378	20758	131.5	12316	17.274406	40.7
s9234	25935	**3157.1**	45838	**53.521366**	**176.7**
s13207	163100	1178.4	57166	203.556465	65

트의 개수만큼 추가적인 면적이 필요하다. 하지만, 테스트 회로가 여러 개의 스캔 체인을 가지고 있고, 스캔 체인별로 회로를 분할하여 테스트를 수행한다면 별도의 MISR을 삽입하지 않고, 데이터의 적재(scan)와 테스트 결과 압축(MISR)을 모두 수행할 수 있도록 회로에 포함되어 있는 스캔 체인에 MISR 기능을 추가할 수 있다. 이 경우 스캔 체인이 MISR 기능을 수행하기 위한 몇 개의 exclusive-OR만이 면적 오버헤드가 된다.

V. 결 론

초고집적 반도체 설계 및 제조가 가능하게 되면서 SOC의 복잡도가 증가하고 있다. 그에 따라 테스트 비용도 크게 증가하고 있으며, 테스트 비용을 줄이기 위하여 많은 연구가 진행되고 있다.

본 논문에서는 SOC의 테스트 비용을 줄이기 위한 새로운 DFT 방법인 순차적 테스트 데이터 압축 방법을 제안한다. 제안된 압축 방법은 테스트 데이터의 양을 줄이기 위하여 공유 비트 압축 방법과 고장 무검출 패턴 압축 방법을 사용하였다. 공유 비트 압축은 테스트 패턴의 순서를 재배열하여 인접한 패턴들 간의 공유 비트를 제거함으로써 데이터의 양을 줄이는 방법이고, 고장 무검출 패턴 압축 방법은 고장 시뮬레이션에 의하여 고장을 검출하지 못하는 패턴을 제거함으로써 압축이 이루어진다. 그리고 제안된 압축 방법을 이용하는 회로는 스캔 DFT 구조에 MISR를 추가하여 test-per-clock 방법을 적용하였으며, 스캔 DFT 구조를 기반으로 하고 있어 스캔 DFT 방법을 적용한 회로를 테스트하는 방법과 동일하게 테스트를 할 수 있다.

벤치마크 회로인 ISCASS85와 ISCASS89 완전 스캔 버전을 이용하여 본 논문에서 제안된 압축 방법을 실험하였으며, ATPG와 고장 시뮬레이션을 위하여 Virginia Ploytechinic Institute & State University의 하동삼 교수 연구실에서 개발한 ATALANTA를 사용하였다. 벤치마크 회로에 대한 실험 결과 순차적 테스트 압축 방법은 스캔 DFT를 적용한 회로의 테스트 데이터를 최소 38%에서 최대 98%까지 줄일 수 있었다.

참 고 문 헌

[1] Y. Zorian, "Test Requirements for Embedded Core-based Systems and IEEE P1500,"

Proceedings of International Test Conference, pp. 191~199, 1997.

[2] Y. Zorian, E.J. Marinissen, and S. Dey, "Testing Embedded-Core Based System Chips," *Proceedings of International Test Conference*, pp. 130~143, 1998.

[3] A. El-Maleh, S. al Zahir and E. Khan, "A Geometric-Primitives-Based Compression Scheme for Testing Systems-on-a-Chip," *Proceedings of VLSI Test Symposium*, pp. 54~59, 2001.

[4] K. Chakrabarty, B.T. Murray, J. Liu, and M. Zhu, "Test Width Compression for Built-In Self-Testing," *Proceedings of International Test Conference*, pp. 328~337, 1997.

[5] J. Rajski, J. Tyszer, and N. Zaccharia, "Test Data Decompression for Multiple Scan Designs with Boundary Scan," *IEEE Transactions on Computers*, Volume 47, Issue 11, pp. 1180~1200, 1998.

[6] M. Ishida, D. S. Ha, and T. Yamaguchi, "COMPACT: A Hybrid Method for Compression Test Data," *Proceedings of VLSI Test Symposium*, pp. 62~69, 1998.

[7] A. Chandra and K. Chakrabarty, "Test Data Compression for System-On-a-Chip using Golomb Codes," *Proceedings of VLSI Test Symposium*, pp. 113~120, 2000.

[8] R. Garey and S. Johnson, *Computers and Intractability, a guide to the theory of NP-completeness*, Freeman and Company, 1991.

[9] J. B. Kruskal, "On the shortest spanning subtree of a graph and the traveling salesman problem," *Proceedings of American Mathematical Society*, Volume 7, Number 1, pp. 48~50, 1956.

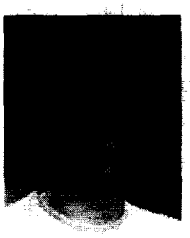
[10] http://cbl.ncsu.edu/CBL_Docs/iscas85.html.

[11] http://cbl.ncsu.edu/CBL_Docs/iscas89.html.

[12] http://www.ee.vt.edu/ha /

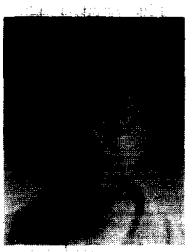
[13] I. Hamazaoglu, J. H. Patel, "Test Set Compaction Algorithms for Combinational Circuits," *Proceedings of International Conference on Computer-Aided Design*, November, pp. 283~289, 1998.

저 자 소 개



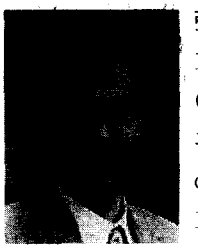
朴 在 興(正會員)

1999년 숭실대학교 컴퓨터학부 졸업(B.S). 2002년 숭실대학교 대학원 컴퓨터학과 졸업(MS). 2002년~숭실대학교 대학원 컴퓨터학과 박사과정. <주관심분야 : 컴퓨터 구조, CAD. VLSI 설계. VLSI 테스팅>



梁 善 雄(正會員)

1996년 숭실대학교 전자계산학과 졸업(B.S). 1998년 숭실대학교 대학원 전자계산학과 졸업(M.S). 1998년~숭실대학교 대학원 컴퓨터학과 박사과정. <주관심분야 : 컴퓨터 구조, VLSI 설계 및 테스팅, CAD>



張 勳(正會員)

1987년 서울대학교 전자공학과 졸업(B.S.). 1989년 서울대학교 전자공학과 졸업(M.S.). 1993년 University of Texas at Austin 박사학위 취득. 1991년 IBM Inc. 1993년 Motorola Inc. Senior Membger of Technocal Staff. 1994년~숭실대학교 컴퓨터학부 조교수. <주관심분야 : 컴퓨터 시스템, VLSI 설계, VLSI 테스팅>