VLSI 회로용 범용 자동 패턴 생성기의 설계 및 구현 기법

장 종 권†

요 약

본 논문에서는 VLSI 회로망의 테스트 패턴 생성에 적합한 범용 자동 테스트 패턴 생성기(UATPG)의 설계 및 구현 기법을 기술하고자 한다. UATPG는 기존 ATPG의 용량을 확장하고 CAD 사용자에게 편리한 설계 환경을 제공하는데 초점을 맞추어 구현되었다. 테스트 패턴 생성시에 함수적 게이트의 신호선 논리값확인 및 고장효과전달을 효과적으로 수행하기 위하여 경험적인 기법을 고안하여 적용하였다. 또한, 테스트 용이화 설계(design for testability)에 사용되는 기억소자(flip-flop)가 의사 입출력으로 이용되어 VLSI 회로망의 시험성을 한층 높여 주었다. 그 결과, UATPG는 사용의 용이성과 성능면에서 좋은 성과를 보여 주었다.

On a Design and Implementation Technique of a Universal ATPG for VLSI Circuits

Jong Kwon Chang†

ABSTRACT

In this paper we propose a design and implementation technique of a universal automatic test pattern generator(UATPG) which is well suited for VLSI digital circuits. UATPG is designed to extend the capabilities of the existing ATPG and to provide a convenient environment to computer-aided design(CAD) users. We employ heuristic techniques in line justification and fault propagation for functional gates during test pattern generation for a target fault. In addition, the flip-flops associated with design for testability (DFT) are exploited for pseudo PIs and pseudo POs to enhance the testabilities of VLSI circuits. As a result, UATPG shows a good enhancement in convenient usage and performance.

1. 서 론

VLSI 설계 및 공정 기술은 칩(chip)의 고집적화와 고속화를 가져왔다. 하지만, 칩 설계의 대규모화와 공정 기술의 복잡성으로 인하여 칩 내부에 여러 가지 형태의 고장 요소가 포함될 확률도 한층 커졌다. 제품 생산 초기 단계에서 불량 요소를 포함한 칩을 철저히 색출하기 위하여 최근 칩 테스트의 중요성이 크게 부각되고 있다.

1960년대 칩 테스트용 테스트 패턴 생성 알고리듬이 개발된 이후, 기존의 자동 테스트 패턴 생성기(ATPG) 기법은 효율성 향상을 위하여, 주입력(PI : primary input) 선택을 위한 백트레이싱(backtracing)과 고장효과전달면에서 보다 향상된 경험적(heuristic) 기법을 채택하고 탐색 영역(search space)을 줄이는데 초점을 맞추어 개발되어 왔다[1, 2, 3, 4, 5]. 그리고 기존의 ATPG 기법은 기본게이트(AND, OR, NAND, NOR, NOT 등)로 구성된 회로의 네트리스트(netlist)에만 적용되어 왔다. 하지만, 실제 회로 설계자는 VLSI용 디지털 회로망 설계시에 여러 형태의 함수게이트(예, decoder, MUX 등)를 기본게이트와 함께 사용할 뿐만 아니라 설계용 소프트웨어 패키지에서 제공하는 다양한 형태의 회로 도면 메뉴(schematic menu)도 최대한 활용하고 싶어한다. 하지만, 이러한 함수게이트는 테스트 패턴 생성에 필요한 게이트 수준의 정보가 갖추어져 있지 않다. 따라서, 테스트 패턴 생성시 백트레이싱과 고장효과전달이 성립되기 위하여

· 본 연구는 '93 학술진흥재단 연구비 지원에 의해 수행되었음.
† 정 회 원 : 울산대학교 컴퓨터공학과 부교수
논문접수 : 1994년 11월 23일, 심사완료 : 1995년 3월 4일

이 함수게이트를 특별히 취급하여야 한다. 본 논문에서는 각각의 함수게이트에 대하여 확장 table-driven 함수를 작성하여 테스트 패턴 생성시에 이용하였다.

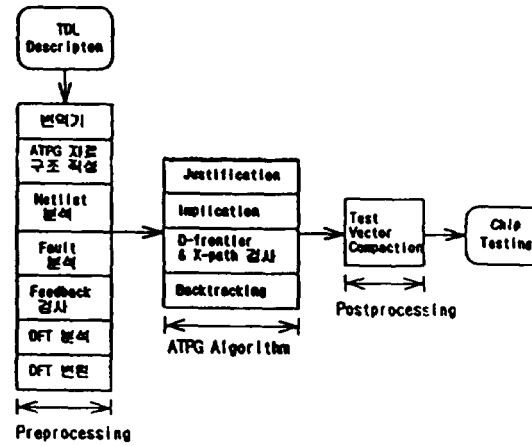
VLSI 디지털 회로망의 크기와 회로 복잡도가 증가하면서 회로의 제어성(controllability) 및 관측성(observability) 향상이 매우 어렵게 되어 테스트 패턴 생성 문제가 심각하게 대두되었다[6, 7, 8]. 주입력(PI) 및 주출력(PO: primary output)의 개수가 일반적으로 VLSI 칩 크기에 비례하여 증가하지 않는다.

따라서, ATPG 기법의 효율성 제고는 그 한계성이 있다. 본 논문에서는 테스트 용이화 설계(design for testability) 기법에 사용되는 플립플롭 소자를 pseudo PI 및 pseudo PO로 활용하여 VLSI 디지털 회로망용 ATPG 기법의 효율성을 크게 향상하였다. 이 Ad-hoc 기법은 시스템 클락 펄스를 이용하여 DFT 플립플롭(테스트 용이화 설계용 플립플롭)의 제어성을 쉽게 실현할 수 있지만 시스템 클락 분석과 하나의 목표 고장에 대하여 두개 이상의 테스트 패턴을 필요로 한다. 하지만, pseudo PI 또는 pseudo PO를 ATPG 탐색영역에 추가하여 ATPG 효율성을 크게 향상시킬 수 있었다.

본 논문은 6 장으로 구성되어 있고, 제 2 장에서는 본 논문에서 제안하는 범용 ATPG (UATPG:universal ATPG) 시스템의 구성에 관하여 기술하고 제 3 장은 UATPG 시스템에서 함수게이트의 테스트 패턴 생성을 위해 table-driven 형식의 함수가 백트레이싱 및 고장효과전달시에 어떻게 이용되는가를 보여주고 제 4 장은 순서회로망의 시험성을 향상시키기 위하여 DFT 플립플롭을 pseudo PI 및 pseudo PO로 활용하는 Ad-hoc 기법을 제안한다. 제 5장에서는 본 논문에서 제안하는 UATPG 시스템의 성능 평가를 여러 형태의 회로에 적용하여 살펴보았다. 마지막으로, 제 6 장에서는 UATPG 시스템의 기법을 종합하여 결론을 맺었다.

2. UATPG 시스템

본 연구에서 제안한 UATPG 시스템은 VLSI 회로망의 네트리스트(netlist)를 전처리하여 자료구조를 구축한 후 고정고장(stuck-at fault) 점검용 테스트 패턴을 생성하는 CAD 툴(tool)이다. VLSI 회로망은 설계 언어(design language)와 도면 메뉴를 사용하여 설계할 수 있고 기본게이트, 함수게이트 및 DFT 플립플롭 소자를 포함하고 있다. UATPG 시스템은 (그림 1)과 같이 크게 전처리, ATPG 알고리듬, 후처리로 나뉜다. 전처리는 테스트 대상 회로망(CUT:circuit under test)의 설계 언어(TDL: TEGAS description language)를 TCF(TEGAS circuit file) 형식으로 변환하는 번역기[9], TCF를 토대로 ATPG 알고리듬 적용을 위한 자료구조 생성기, 고장등가(fault equivalence) 규칙을 적용하여 목표 고장을 줄이는 고장 콜랩싱(fault collapsing)부, 순차회로망의 풀 스캔 규칙 검사 및 풀 스캔 자동형성부, 그리고 구조적 테스트 용이화 설계 기법을 적용할 수 없는 스캔 경로(scan path)를 제외한 random logic의 귀환신호선 검출부로 구성되어 있다. ATPG 알고리듬은 본 연구의 핵심으로 PODEM 알고리듬을 기초로 하여 설계되었다. ATPG 알고리듬은 목표 고장을 설정한 뒤 주입력(PI)의 논리값을 설정하는 신호선 논리 값 확인, 고장효과를 주출력(PI)으로 유도 전달하는 고장효과해석(implication), 고장효과해석을 위한 D-경계(D-frontier)와 X-경로(X-path)의 검사 및 선택, 그리고 신호선 논리값확인시에, 고장효과해석중에

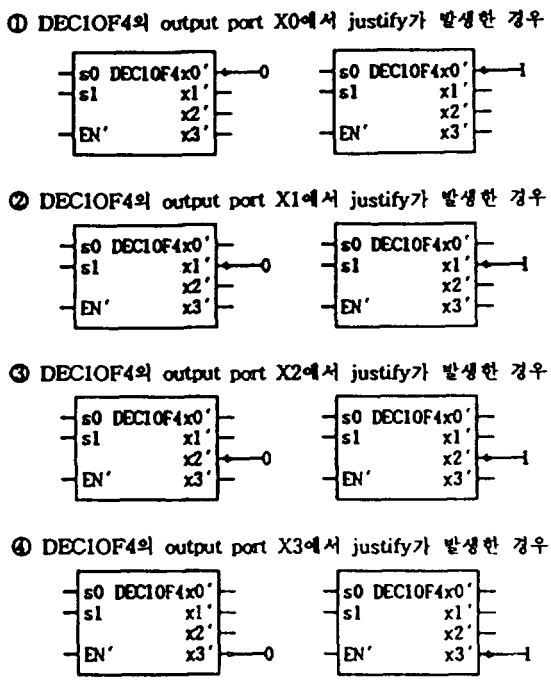


(그림 1) UATPG 시스템 구성도
(Fig. 1) UATPG system configuration

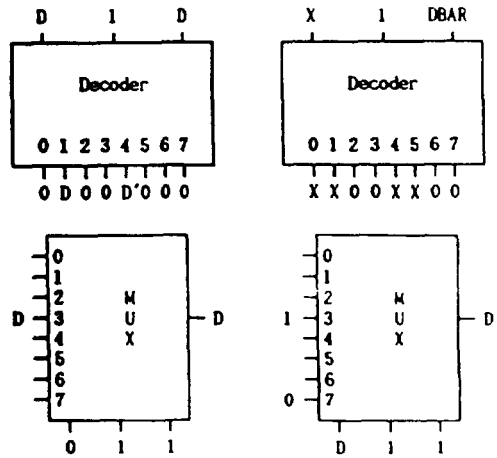
지정되는 신호선 논리값과 백트레이싱중 설정된 논리값이 일치하지 않을 때 이 문제점을 해결하기 위하여 ATPG 알고리듬을 재시도시키는 벡트래킹(backtracking)으로 구성되 어 있다. 후처리는 ATPG 알고리듬에서 생산된 테스트 패턴의 don't care 상태를 검사하여 상호 포함 관계가 있는 입력 패턴을 합하여 테스트 패턴을 압축한다.

3. 함수게이트의 테스트 패턴 생성

실제 VLSI 회로망은 다양한 형태의 함수게이트로 설계된다. 이러한 함수게이트는 기본게이트와 달리 다중 출력과 다양한 기능 핀을 가지기 때문에, 기존의 ATPG가 함수게이트를 가진 회로망을 테스트하기 위해서는 함수게이트를 기본게이트 수준으로 변환해야만 테스트 패턴을 생성할 수 있었다. 하지만, 변환된 게이트 수준의 회로망은 테스트 패턴 생성시에 메모리 소요가 클 뿐만 아니라 기능 테스트(functional test)에서 고려할 필요가 없는 함수게이트 내부 고장(internal fault)도 함께 처리되어 테스트 패턴 생성 시간이 길어지는 단점이 있다. 따라서, 본 연구에서는 함수게이트의 내부 게이트 연결 정보 없이 특성함수(characteristics function)만으로 회로망의 테스트 패턴을 효율적으로 생성할 수 있는 함수게이트용 경험적 기법을 고안하였다. 다양한 형태의 함수게이트는 논리값확인과 고장효과해석을 위해 전처리 단계에서 함수게이트의 이름, 종류, 입출력핀 정보 등이 UATPG용 자료구조로 저장된다. 함수게이트의 신호선 논리값확인을 효율적으로 수행하기 위하여 경험적 기법을 채택하였다. 이 경험적 기법은 백트레이스되어야 할 함수게이트 출력의 논리값, 현재 함수게이트 입력에 인가된 논리값, 그리고 함수게이트의 특성함수에 기초하여 신호선 논리값확인을 수행한다. 함수게이트의 다양한 기능과 다중 입출력핀 때문에, 게이트 수준의 정보를 갖지 않은 함수게이트의 신호선 논리값확인에 상당한 시간을 소비하게 되지만, 함수게이트에 알맞은 경험적 기법을 채택함으로써 쉽게 해결할 본 연구는 93' 한국학술진흥재단의 연구비 지원에 의해 수행되었음수 있다. 예를 들면, 디코더의 초기 신호선 논리값확인은 ENB 핀을 low로 하여야 한다. 즉, 디코더의 ENB 핀은 다른 입력핀의 논리값확인에 우선하여 논리값 0으로 결정하여야 한다. 기타 신호선 논리값확인은 함수게이트의 출력 및 입력 논리값에 의하여 결정된다. (그림 2)는 2×4 디코더의



(그림 2) 2×4 디코더의 논리가확인
(Fig. 2) Justification of 2×4 decoder



(그림 3) Decoder와 MUX의 고장효과해석
(Fig. 3) Implication of decoder & MUX

신호선 논리값확인의 8 가지 경우를 나타낸다. 신호선 논리가확인시 디코더 입력핀 선택은 백트레이싱되어야 할 출력핀의 위치 및 논리값에 의하여 결정된다.

본 연구에서 제안한 경험적 기법은 (그림 3)과 같이 고장효과해석 또한 효율적으로 수행할 수 있다.

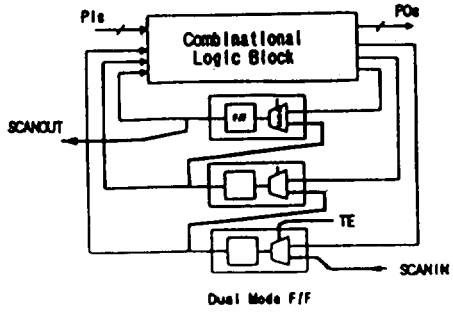
경험적 기법은 고장효과해석시에 함수게이트의 다중 입출력을 통해 고장효과전달을 효율적으로 수행하기 위하여 어떤 출력핀을 먼저 선택해야 하는지를 결정한다. 예를 들면, 2×4 디코더의 고장효과해석은 특성함수의 5가 논리(0, 1, X, D, DBAR) 연산을 수행한 다음 각각의 출력에 대하여 고장효과전달을 위한 출력을 결정한다.

4. DFT 플립플롭의 테스트 생성

4.1 스캔 기법

순서회로망은 조합회로망과는 달리, 테스트시에 다음과 같은 문제점을 갖고 있다. 첫째, 일반적으로, 회로망내의 초기상태는 결정되어 있지 않다. 따라서, 회로망의 초기 상태를 결정해 주어야 한다. 즉, 제어성(controllability)이 현재 검사중인 회로망에 요구된다. 둘째, 고장효과는 회로망의 적어도 하나의 출력으로 전달되어야 한다. 만약 고장효과가 회로망의 최종 상태를 변경하여 오동작 하더라도, 회로망에 인가한 테스트 입력으로 회로망의 최종상태를 관측할 수 있어야 한다. 즉, 관측성(observability)이 현재 검사중인 회로망에 요구된다. 일반적으로, 제어성은 입력수가 많을수록 증가하고, 관측성은 출력수가 많을수록 증가한다. 하지만, 순서회로망의 제어성과 관측성은 시간에 종속적인 동작 특성을 가진 플립플롭 때문에, 순서회로망 내의 플립플롭의 상태를 쉽게 제어 및 관측할 수 없다. 따라서, 최근의 대용량 VLSI 회로망의 테스트 추세는 VLSI 설계시에 회로망의 제어성과 관측성 향상을 위해 테스트 용이화 설계 기법을 필수적으로 채택하고 있다.

본 연구의 UATPG는 회로망의 시험성(testability) 향상을 위해 구조적 테스트 용이화 설계 기법중의 하나인 풀 스캔 기법을 지원한다. 스캔 기법은 순서회로망의 제어성과 관측성 향상을 위해 이중 모드(dual mode) 플립플롭을 사용하여, 순서회로망내의 플립플롭 상태를 직접 제어 관측할 수 있도록 순서회로망에 시험성을 고려한 구조적 설계 기법이다[8]. 이중 모드 플립플롭은 정상(normal) 동작과 테스트(test) 동작을 독립적으로 수행한다. (그림 4)는 스캔 기법으로 설계된 회로망의 테스트 절차를 나타내고 있다. 우선, 스캔 경로상에 있는 플립플롭의 고장을 테스트하기 위하여 flush 테스트와 shift 테스트를 수행한다. Flush 테스트는 플립플롭의 상태를 0으로 초기화한 다음에 1로 초기화하여 플립플롭의 상태가 0이나 1에 고정되는지 검사한다. Shift 테스트는 스캔 입력(SCANIN)에 00110011…의 패턴을 인가하여 동일한 입력패턴이 스캔 출력(SCAN-OUT)에서 나오는 지를 검사한다. 다음으로, 스캔 입력상의 플립플롭에 테스트 패턴을 인가한 뒤, 조합회로망의 입력에 테스트 패턴을 인가하여 고장을 검사한다.



(그림 4) 스캔 회로망의 테스트
(Fig. 4) Test of scanned circuit

순서회로망내의 모든 플립플롭을 스캔 경로로 엮어 설계하는 기법을 풀 스캔(full scan) 기법이라 하고, 부분적으로만 엮어 설계하는 기법을 부분 스캔(partial scan) 기법이라 한다. 풀 스캔 기법은 순서회로망의 시험성을 상당히 향상시키지만, 칩의 면적(area)과 회로망의 지연(delay) 문제를 야기시킨다. 따라서, 스캔 경로에 연결해야 할 플립플롭의 수를 줄여 면적과 지연 문제를 해결하기 위해 부분 스캔 기법이 제안되었다[10]. 하지만, 부분 스캔 기법을 채택한 순서회로망은 완전한 조합회로망이 아니라 의사(pseudo) 조합회로망으로 변환된다. 왜냐하면, 부분 스캔 기법

을 채택한 회로망은 플립플롭 입출력에 귀환신호선만 없을 뿐 내부에는 플립플롭을 갖고 있기 때문이다.

4.2 테스트 패턴 생성

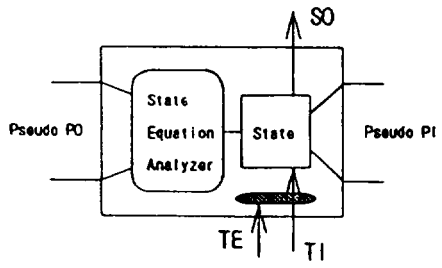
UATPG는 테스트 용이화 설계 기법중 풀 스캔 기법을 지원한다. UATPG는 순서회로망에 내장된 플립플롭의 테스트를 다음과 같은 과정으로 수행한다. 우선, 회로망의 플립플롭을 제외한 random logic 부분에 귀환신호선 유무를 검사한다. 이러한 귀환신호선을 가진 회로망에 적용할 수 있는 구조적(structured) 설계 기법이 존재하지 않기 때문에, 귀환신호선 문제는 테스트 기술자(test engineer)에게 맡기고 UATPG는 회로망 내에 존재하는 모든 귀환신호선의 위치를 제공하여 테스트 기술자가 귀환신호선을 찾는 수고를 덜어주는데 역점을 두었다. 귀환신호선 검사 다음에 회로망의 모든 플립플롭의 종류(type), 입력단에서 플립플롭까지의 레벨(level), 플립플롭의 입출력에 연결된 모든 게이트 ID 번호, DFT 플립플롭의 유무를 검사하고 변환되어야 할 DFT 플립플롭의 형 정보를 생성하는 스캔 형 검사를 수행한다. 스캔 형 검사로 생성된 여러 가지 정보를 이용해 DFT 플립플롭을 배열(scan ordering)한다. 마지막으로, 스캔 형 검사와 스캔 플립플롭 배열을 마친 다음 풀 스캔 규칙에 맞게 회로망의 네트리스트를 변경하여 스캔 경로를 완성한다.

스캔 경로 생성을 마친 테스트 용이화 기법으로 설계된 회로망의 플립플롭 테스트를 위해 본 연구에서는 Ad-hoc 기법인 pseudo PI PO 기법을 제안하였다. (그림 5)는 pseudo PI PO 기법을 채택한 플립플롭의 고장 해석 모델을 나타낸다.

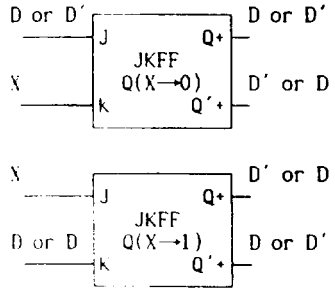
Pseudo PI PO 기법은 플립플롭의 입력을 pseudo PO로 간주한다. 고장효과해석중에 pseudo PO에 고장효과가 도달하면, PO에 고장 효과가 도달한 것으로 간주하고 테스트 생성을 중지한다. Pseudo PO에 나타난 고장은 스캔 경로에 쉬프트 클락을 인가하여 SO(scan out)로 관측이 가능하다. 또한, Pseudo PI PO 기법은 플립플롭의 출력을 pseudo PI로 간주한다. pseudo PI의 논리값을 결정 트리(decision tree)에 추가하여 백트래킹을 수행한다[2]. 따라서, 테스트 패턴은 PI 패턴과 pseudo PI 패턴으로 이루어진다. 상태해석부(SEAs:tate equation analyzer)는 pseudo PI에 인가된 논리값을 해석하여 플립플롭의 초기 상태를 결정한다. 초기 상태는 스캔 경로를 통하여 인가된다. 테스트 용이화 기법으로 설계된 회로망에 pseudo PI PO 기법을 채택함으로써 회로망의 시험성을 향상시킬 수 있다. 실제 pseudo PI PO 기법은 회로망의 입출력을 증가시키므로 테스트 패턴 생성시에 백트래킹 횟수를 상당히 감소시켜 준다. 따라서, pseudo PI PO 기법은 백트래킹 횟수가 주어진 한계를 초과하는 고장의 테스트 패턴 생성에 더욱 효율적이다. 〈표 1〉은 대표적인

〈표 1〉 UATPG 기법의 특성
〈Table 1〉 Characteristics of UATPG technique

시스템	기존 ATPG		UATPG
알고리듬	D	PODEM	PODEM 계열
탐색영역	전 노드	전 입력단	전 입력단 + 의사입출력단
적용대상회로	조합회로망	조합회로망	조합회로망 + DFT 순서회로망



(그림 5) 플립플롭의 고장 해석 모델
(Fig. 5) Fault analysis model of flip-flop



(그림 6) JK 플립플롭의 고장해석 모델
(Fig. 6) Fault analysis model of JK flip-flop

ATPG 기법과 본 논문의 UATPG 기법을 탐색영역 및 적용 대상 회로면에서 비교하였다.

UATPG의 탐색영역이 확장되고 적용 대상 회로가 DFT 기법으로 설계된 순차회로망까지 범위가 넓어 졌지만 테스트 패턴 적용시 테스트 인가 시간이 길어지는 것을 감수하여야만 한다.

또한, (그림 6)은 JK 플립플롭 고장해석 모델을 나타낸다. JK 플립플롭의 입력에 인가된 고장효과를 플립플롭의 출력단으로 전달할 수 있다. 따라서 J,K 입력에 고장효과가 전달되면 괄호안의 표기(Q(X 0))는 상태해석부(SEA)가 don't care인 이전상태(Q)를 논리값 0으로 초기화한다.

5. UATPG 시스템의 수행 결과

본 논문의 UATPG(Universal ATPG)는 UNIX 환경의 SUN 4 workstation상에서 15000 라인의 C 언어로 구현되었다. 〈표 2〉는 함수게이트 수준의 회로망과 기본게이트 수준의 회로망의 테스트 생성 결과 비교에 사용된 회로망의 특성을 나타내고 있다. 〈표 2〉에서 #PI, #PO, #fanout, #gates는 각각 PI, PO, fanout, 그리고 게이트의 개수를 나타낸다. #TF (target fault)는 테스트 대상 회로망의 목표 고장(target fault)을 나타낸다.

〈표 3〉은 〈표 2〉의 특성을 가진 회로망을 이용하여 UATPG의 테스트 생성을 수행한 결과를 나타내고 있다. 테스트 패턴 생성 시간이 ckt_d (MUX 포함)를 제외하고 게이트 수준보다 함수게이트 수준의 회로망에서 적은 것을 주목할 수 있다. 그 이유는 함수게이트 수준의 회로망이 백트레이싱 및 고장효과전달면에서 훨씬 효율적이기 때문이다. 〈표 3〉에서 #TF(target fault)는 목표 고장의 개수, #RF(redund- ant fault)는 테스트 대상 회로망의 목표 고장을 검출할 수 없는 고장의 개수, #AF(aborted fault)는 테스트 패턴 생성시에 최대 백트래킹 숫자 한계를 초과하여 테스트 패턴 생성을 중지한 고장의 개수를 나타낸다. time은 UATPG의 테스트 패턴 생성 시간을 나타낸다. TFC(tested fault coverage)는 RF를 제외한 테스트 패턴의 고장검출율을 나타낸다.

〈표 4〉는 기본게이트, 함수게이트, 그리고 플립플롭으로 구성된 회로망의 테스트 패턴 생성에 이용한 테스트 대상 회로망의 특성을 나타낸다. 〈표 2〉와는 달리 함수게이트와 일반게이트를 구분하지 않고 #gates 항목에 모두 표시하였다.

〈표 2〉 테스트 대상 회로망의 특성
〈Table 2〉 Characteristics of circuit under test

CKT	primitive gate / functional gate	#PI	#PO	#fanout	#gates	#TF
ckt_a	gate level	14	8	43	102	338
	ADDER	14	8	34	63	266
ckt_b	gate level	14	8	32	69	254
	DECODER	14	8	28	63	246
ckt_c	gate level	14	8	49	94	297
	ENCODER	14	8	34	63	260
ckt_d	gate level	14	8	41	77	292
	MUX	14	8	35	63	274

〈표 3〉 회로망 〈표 2〉의 테스트 패턴 생성 결과
〈Table 3〉 Test pattern generation of 〈Table 2〉

CKT	primitive gate / functional gate	#TF	#RF	#AF	#TB	time	TFC(%)
ckt_a	gate level	338	4	1	326	4.100	99.704
	ADDER	266	0	0	92	2.267	100
ckt_b	gate level	254	0	0	22	1.650	100
	DECODER	246	0	0	26	1.500	100
ckt_c	gate level	297	49	0	586	4.367	100
	ENCODER	260	2	0	245	3.300	100
ckt_d	gate level	292	11	0	118	2.467	100
	MUX	274	0	4	526	3.400	98.54

〈표 4〉 테스트 대상 회로망의 특성
〈Table 4〉 Characteristics of circuit under test

CKT	#PI	#PO	#fanout	#gates	#FF	#TF
ckt1	5	2	3	16	0	22
ckt2	6	5	16	44	11	64
ckt3	8	2	24	64	12	92
ckt4	14	2	15	59	8	142
ckt5	55	9	17	165	9	202
ckt6	14	8	31	116	0	247
ckt7	36	7	89	290	0	524
ckt8	12	72	199	699	81	1182
ckt9	52	25	347	1671	216	1466
ckt10	37	11	718	1863	537	2079

〈표 5〉 회로망 〈표 4〉의 테스트 패턴 생성 결과
〈Table 5〉 Test pattern generation of 〈Table 4〉

CKT	#TF	#RF	#AF	#TB	time(s)	TFC(%)	FC(%)
ckt1	22	0	0	0	0.016	100	100
ckt2	64	0	0	33	0.1	100	100
ckt3	92	64	0	739	1.433	100	30.435
ckt4	142	1	0	174	0.633	100	99.296
ckt5	202	68	4	3792	6.2	98.02	64.356
ckt6	247	0	0	21	1.583	100	100
ckt7	524	0	20	2711	23.966	96.183	96.183
ckt8	1182	21	0	416	57.750	100	98.223
ckt9	1466	34	0	226	43.166	100	97.681
ckt10	2079	30	0	21471	118.866	100	98.557

〈표 5〉는 〈표 4〉의 특성을 가진 회로망을 이용하여 UATPG의 테스트 생성을 수행한 결과를 나타내고 있다. TFC(tested fault coverage)는 RF를 제외한 테스트 패턴의 고장검출율 나타내고 FC(fault coverage)는 테스트 패턴의 고장 검출율을 나타낸다. 예외적으로, ckt3와 ckt5는 RF가 많이 포함되어 있어 FC가 낮게 도출되었다.

〈표 6〉은 ISCAS89 benchmark 회로망의 특성을 나타낸다[11]. #PI, #PO, 그리고 #fanout 항목은 UATPG 시스템이 테스트 패턴 생성을 위해 입력 회로망을 풀 스캔 기법으로 변환된 회로망의 PI, PO, 그리고 fanout의 개수를 나타내고, 변환전의 입력 회로망의 PI, PO, 그리고 fanout 개수는 괄호 속에 표시하였다. 〈표 7〉은 〈표 6〉을 입력 회로망으로 하여 테스트 패턴 생성을 수행한 결과이다.

6. 결 론

본 논문에서 제안한 UATPG는 경험적 기법과 Ad-hoc 기법을 기초로 설계되었으며 사용자에게 편리한 환경을 제공하기 위하여 기존 ATPG의 단점을 보완하고 기능을 확장하여 구현되었다. UATPG 사용자는 설계 언어 또는 도면 메뉴를 사용하여 회로망의 네트리스트를 작성할 수 있다. 테스트 대상 회로망은 기본게이트, 함수게이트 및 DFT 플립플롭을 사용하여 설계될 수 있고 DFT 기법으로 설계되지 않은 순차 회로망은 자동적으로 풀 스캔 기법을 적용하여 시험 가능한 DFT 순서회로망으로 변환하여 준다. 이상과 같이 UATPG는 사용자에게 편리한 환경을 제공하여 준다.

한편, UATPG는 함수게이트의 백트레이싱과 고장효과전달시에 경험적 기법을 사용하였고 DFT 플립플롭을 의사 PI 및 의사 PO로 활용하여 VLSI 회로망의 시험성을 크게 향상시켰다. 하지만, 이러한 편리성과 효율성은 테스트 패턴의 양이 증가되는 단점을 가져 왔다. 전체적인 VLSI 회로망 설계 공정에서 보면 테스트 패턴인가 시간은 길어졌지만 테스트 패턴 생성 시간은 크게 짧아졌고 고장 검출 범위는 매우 향상되었다.

앞으로의 연구방향은 테스트 패턴 양을 최소화할 수 있는 스캔 체인 배열(scan chain ordering) 기법 개발에 초점이 맞춰져야 한다고 생각된다.

〈표 6〉 테스트 대상 회로망의 특성
〈Table 6〉 Characteristics of circuit under test

CKT	#PI	#PO	#fanout	#gates	#FF	#TF
s27	7(5)	2(1)	9(5)	10	3	34
s298	6(4)	7(6)	37(35)	119	14	286
s382	6(4)	7(6)	57(50)	158	21	373
s444	6(4)	7(6)	73(66)	181	21	448
s526	6(4)	7(6)	57(55)	192	21	519
s713	38(36)	24(23)	101(81)	414	19	585
s953	19(17)	24(23)	183(159)	395	29	1071
s1196	17(15)	15(14)	169(156)	529	18	1234
s1423	20(18)	6(5)	194(181)	657	74	1395
s5378	38(36)	50(49)	915(856)	2779	179	4229

〈표 7〉 회로망 〈표 6〉의 테스트 패턴 생성 결과
〈Table 7〉 Test pattern generation of 〈Table 6〉

CKT	#TF	#RF	#AF	#TB	time(s)	TFC(%)	FC(%)
s27	34	0	0	13	0.05	100	100
s298	286	33	0	110	3.45	100	100
s382	373	15	0	20	4.633	100	95.978
s444	448	33	0	428	9.766	100	92.634
s526	519	39	0	119	11.050	92.485	100
s713	585	23	21	2996	31.767	96.410	92.478
s953	1071	54	2	1147	79.750	99.813	94.771
s1196	1234	0	2	1046	55.083	99.838	99.838
s1423	1395	32	15	1773	161.233	98.925	96.631
s5378	4229	117	6	1675	1177.383	99.858	97.092

참 고 문 헌

[1] J. P. Roth, "Diagnosis of Automata Failures:A Calculus and a Method," IBM Journal of Research and Development, Vol. 10, No. 4, pp. 278-291, July 1966.

[2] P. Goel, "An Implicit Enumeration Algorithm to Generate Tests for Combinational Logic Circuits," IEEE Trans. on Computers, Vol. C-30, No. 3, pp. 215-222, March 1981.

[3] H. Fujiwara and T. Shimono, "On the Acceleration of Test Generation Algorithms," IEEE Trans. on Computers, Vol. C-32, No. 12, pp. 1137-1144, December 1983.

[4] T. Kirkland and M. Ray Mercer, "A Topological Algorithm For ATPG," 24th Design Automation Conference, pp. 502-508, 1987.

[5] M. H. Schulz and E. Auth, "Advanced Au-

tomatic Test Pattern Generation and Redundancy Identification Techniques," Proceedings of the 18th Symposium Fault-Tolerant Computing, pp. 30-35, June 1988.

[6] E. B. Eichelberger and T. W. Williams, "A Logic Design Structure for LSI Testability," Journal Design Automation & Fault Tolerant Computing, Vol. 2, No. 2, pp. 165-178, May 1978.

[7] L. M. Goldstein and E. L. Thigen, "SCOAP : Sandia Controllability / Observability Analysis Program," Proc. 17th Design Automation Conf., pp. 190-196, June 1980.

[8] M.Abramovici, M. Breuer, and A. D. Friedman, Digital Systems Testing and Testable Design, Computer Science Press, Chapter 9, pp. 343-408, 1990.

[9] E. W. Thomsonand Szygenda, "Digital Logic Simulation in a Time-Based, Table-Driven Environment-Part2. Parallel Fault Simulation," Computer, Vol. 8, No. 3, pp.34-49, March 1975.

[10] K. T. Cheng and V. D. Agrawal, "A Partial Scan Method for Sequential Circuits with Feedback," IEEE Trans. on Computers, Vol. 39, No. 4, pp. 544-548, April 1990.

[11] F. Brglez et al., "Combinational Profile of Sequential Benchmark Circuits," Proc. IEEE ISCAS, pp. 1929-1934, 1989.

장 종 권

1973년 서울대학교 전기공학과 졸업 (공학사)
1985년 University of Texas at Austin 전기공학과 졸업(공학석사)
1990년 University of Texas at Austin 전기공학과 졸업(공학박사)
1991년~현재 울산대학교 컴퓨터공학과 부교수
관심 분야 : VLSI Testing, Fault Simulation, Logic Synthesis 등임.