論文94-31A-2-13

논리값 제약을 갖는 스캔 설계 회로에서의 자동 시험 패턴 생성

(A Method to Generate Test Patterns for Scan Designed Logic Circuits under Logic Value Constraints)

朴 恩 世*

(Eun Sei Park)

要 約

스캔 방식으로 설계된 디지탈 논리 회로의 테스팅은 설계와 검증의 다양한 요구 조건에 의해 회로의 일부 주입력에 대한 논리값 제약을 받을 수 있다. 본 논문에서는 논리값 제약 상태에서의 자동 시험 패턴 생성을 위한 taboo logic value 시스템을 제안한다. Taboo logic value는 주입력에서의 논리값 제약조건을 심볼화하고 taboo logic calculus를 사용한 implication을 통하여 부수적인 논리값 제약을 효율적으로 추출한다. 추출된 논리 제약 조건들은 자동 시험 패턴 생성 과정에서 불필요한 탐색 과정을 감소시키고 redundant한 고장을 효과적으로 검출하게 된다. 아울러 ISCAS85 회로들에 대한 실험으로 taboo logic value 시스템의 효율성이 입증된다.

Abstract

In testing for practical scan designed logic circuits, there may exist logic value constraints on some part of primary inputs due to various requirements on design and test. This paper presents a logic value system called taboo logic values which targets the test pattern generation of logic circuits under logic value constraints. The taboo logic system represents the logic value constraints and identifies additional logic value constraints through the implication of the taboo logic values using a taboo logic calculus. Those identified logic value constraints will guide the search during the test pattern generation to avoid the unfruitful searches and to identify redundant faults due to the logic value constraints very quickly. Finally, experimental results on ISCAS85 benchmark circuits will demonstrate the efficiency of the taboo logic values.

* 正會員, 韓國電子通信研究所
(Electronics and Telecommunications Research Institute (E.T.R.I))
接受日字 : 1993年 1月 21日

I. 서 론

최근 VLSI 설계 기술의 급격한 발전으로 회로의 고집적화 및 고속화가 가능해 졌으나, 그에 상응하는 생산 기술의 미비로 불량품의 출하 위험이 증가되어

왔다. 이러한 불량품은 상위 단계의 제품의 불량을 초래하게 되어 전체적인 설계 및 제작 비용을 증가시키므로 VLSI 테스트의 중요성이 더욱 부각되고 있다.

임의의 순차 회로 (sequential circuit)의 테스트는 그동안의 많은 연구 노력에도 불구하고 높은 고장 검출율 (fault coverage)의 보장이 어려워 일반적인 적용을 하지 못하는 실정이었으나, 1980년대 이래 LSSD (level sensitive scan design) [1], scan path [2] 등의 스캔 설계 방식의 적용으로 single stuck-at 고장 모델의 경우 높은 고장 검출율을 이룰 수 있었다. 스캔 설계 방식은 기본적으로 동기식 순차 회로를 테스트 동작시에 flip-flop등의 기억소자를 shift register로 변환시켜 각 flip-flop의 논리값을 외부에서 입출력시킴으로써 전체 순차 회로의 테스팅을 몇개의 분할된 조합 회로 (combinational circuit)의 테스팅으로 바꾸어 지게하는 설계방식이다. 이 방식은 테스트 생성을 자동화시킬수 있고 양질의 테스트 패턴의 생성을 보장할 수 있는 장점이 있는 반면 부수적인 실리콘 면적의 증가와 회로의 동작 속도 감소등의 단점이 있을 수 있다. 또한 스캔 설계는 비교적 엄격한 설계 법칙이 준수되어야 하는 부담이 있어 설계자의 창의력을 제한하는 경우가 종종 있다.

스캔 설계 법칙을 위배하는 경우, 실제 디지탈 논리 회로의 테스팅은 설계와 검증의 다양한 요구조건에 의해 회로의 일부 주입력에 대한 논리값의 제약을 받을 수 있다. 예를 들면, 대규모의 논리 회로 설계는 여러개의 기능에 따른 소규모의 논리 회로들로 구성되는데, 테스트는 주로 각 기능별로 하게 된다. 이때, 각 분할된 소규모의 회로간의 연결 신호들이 스캔 flip-flop에서 출발하지 않거나 또는 주입력 신호

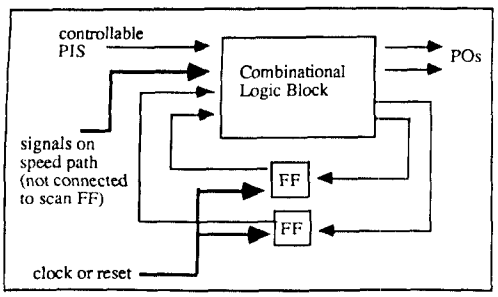


그림 1. 주입력단에 논리값 제약을 받는 스캔 설계 회로

Fig. 1. A scan-designed logic circuit with input constraints.

가 아니면 그 신호들을 논리값 0이나 1로 조절하지 못하는 경우가 많다. 특히, 이와 같은 상황은 연결 신호들이 임계 경로상에 존재할 때 두드러진다. 그림 1은 일반적인 스캔 방식의 설계에서 주입력에 논리값 제한을 갖는 예이다.

스캔 설계된 회로에서의 테스트 패턴의 생성은 주입력들로 이루어진 패턴이 가질 수 있는 모든 논리값의 집합중에서 주어진 고장을 검출할 수 있는 패턴의 부분 집합을 찾아내기 위한 탐색 과정이라 하겠다. 탐색 과정은 주로 고장의 유발 (fault excitation)과 그 고장을 관측할 수 있는 주출력 단자까지의 전달 (fault propagation)로 구성되는데, 각 단계별로 objective를 세우고 그 objective를 만족시키는 주입력의 논리값을 구하는 line justification 과정과 구해진 주입력의 논리값을 회로에 적용하는 implication 과정등이 있다. 그러나 주입력에서 가질 수 있는 논리값에 제약이 있는 경우, 종래의 조합 회로에 사용되는 PODEM [3] 또는 SOCRATES [4] 알고리즘등을 적용하기 어렵다. [5] 그 이유는 논리값 제약이 있을 때 line justification에서 논리 소자의 출력단의 논리값이 정하여지지 않은 상태에서 같은 소자의 입력단의 논리값이 정하여져 있거나 제약을 받는 경우가 발생하는 모순 (conflict)이 있을 수 있기 때문이다. PODEM과 같은 종래의 알고리즘에서는 implication 또는 고장 경로 검사만으로는 논리상의 모순을 발견할 수 없으므로 논리값 제약에 의한 모순을 발견하기 위해서는 탐색 알고리즘상의 근본적인 수정이 불가피하게 된다.

본 논문에서는 논리값 제약을 갖는 스캔 방식의 회로에서의 자동 시험 패턴 생성을 위한 새로운 논리값 시스템인 taboo logic value 시스템을 제안한다. Taboo logic value는 주입력단의 논리값 제약 상태를 심볼화하고 taboo logic calculus를 사용한 implication 과정에 의해 부수적인 회로상의 논리 제약들을 효율적으로 찾아내게 된다. Taboo logic 시스템에 의해 검출된 논리값 제약 조건의 사용으로 line justification시 불가능한 목표의 설정을 방지하고 테스트 과정을 가속화시키게 되며 논리값 제약에 의한 redundant 고장들을 조기에 발견할 수 있다.

본 논문은 다음과 같이 구성되어 있다. Ⅱ 장에서는 taboo logic value와 taboo logic calculus에 대한 소개가 있으며, Ⅲ 장에서는 taboo logic 시스템을 사용한 새로운 자동 시험 패턴 생성 시스템인 TABOO-ATPG에 대한 설명을 하고 마지막으로 Ⅳ 장에서는 ISCAS85 회로들에 대한 실험으로 taboo logic value 시스템의 효율성을 보여준다.

Ⅱ. Taboo Logic Value 시스템

1. Taboo logic value와 taboo logic calculus

주입력단에 논리값 제약을 갖는 회로의 자동 시험 패턴 생성에서는 제약된 주입력에 불합리한 논리값을 배정하지 않도록 주의하여야 하므로 논리값의 제약은 테스트 생성 시 탐색 과정을 어렵게 한다. 예를 들어, 그림 2에서 주입력 B는 시스템의 clock 신호이므로 테스트시에 임의의 값으로 조절이 안된다. 따라서 스캔 설계 방식에서는 clock 신호는 조합 회로 부분과 분리되어야 함에도 그림 2와 같은 스캔 설계 방식이 위배되는 경우가 설계상의 특별한 요구에 의해 발생할 수 있다.

그러한 경우, 종래의 PODEM등의 알고리즘에서는 스캔 법칙의 위배이므로 테스트 생성 과정에 들어 갈 수 없다. 테스트 생성을 위해서는 입력 B의 논리값을 정하지 않은 상태, 즉 unknown 논리값 X로 남아있어야 한다. 이때, G1의 출력단에 논리값 1을 justification하여야 한다고 가정하여 보자. G1의 출력단에 1을 얻기 위해서는 두 입력단 A와 B 동시에 논리값 1이 필요하다. 따라서, B의 논리값이 unknown 상태이어야 하는 조건에 모순이 생기게 되므로 결국 G1의 출력단에 논리값 1을 만들수 없음을 알수 있다. 그러나, G1의 출력단은 입력 B의 값에 관계없이 입력 A에 논리값 0을 가하여 논리값 0을 가질 수 있다. 비슷한 방법으로 G2의 출력단에 입력 B의 논리값 제약으로 논리값 0을 가질 수 없음을 알 수 있다. 이와 같은 방식으로, G3의 출력단에 논리값 1을 가할 수 없음을 알게 된다. 또한, G3의 출력단에 논리값 0을 가하려면 주입력 A의 논리값이 0이어야만 한다는 것을 알수 있다. 한편, G4의 출력단은 주입력 B의 논리값 제약에 무관하게 아무런 제약을 받지 않으므로 논리값 0 또는 1로 justification 될 수 있음을 알게 된다. 이와 같이, 논리값이 제약을 받는 경우, 그 영향이 회로의 일정 부분에 전달되어 테스트 생성시에 달성될 수 없는 objective의 justification에 의한 시간 낭비를 하게 된다. 그러므로 회로내의 논리 제약 정보를 preprocessing을 통해 미리 추출하여 테스트 생성시에 이용하면 시간을 절감할 수 있고 제약에 의한 redundant 고장들을 조기에 검출할 수 있다.

위의 관찰에서, 주입력 B의 논리값 제약이 회로내에서 주출력단을 향하여 전파됨을 알게 되는데, 논리값 제약의 implication은 전체 회로를 다음과 같은 4부분으로 분할한다.

- Group 1: 논리값이 결정되지 않는 부분, 즉 테스트 생성시 논리값이 don't-care 상태이어야 하는 부분
- Group 2: 논리값 0을 가질 수 없는 부분
- Group 3: 논리값 1을 가질 수 없는 부분
- Group 4: 논리값의 제약을 받지 않는 부분, 즉 논리값 0 또는 1을 제약없이 가질 수 있는 부분

Taboo Logic 시스템은 논리값 제약의 전달 문제를 체계적이며 효율적으로 해결하기 위하여 착안되었다. Taboo logic value는 논리값 제약에 의해 회로의 각 node가 어떠한 논리값을 가질 수 없는 가를 나타내는데, 즉 각 node가 주어진 주입력 제약하에 어떠한 논리값이 justification될 수 없는 가를 나타내게 된다. Taboo logic value 시스템은 4-valued algebra X (unknown), T0 (taboo 0), T1 (taboo 1)과 TX (taboo X)에 의해 정의된다. 위의 group 1에 속한 node들은 TX (or taboo-X)의 값을 갖게 되며 테스트 생성시 논리값이 정해지지 않는 상태, 즉 논리값 0이나 1을 가질 수 없는 unknown 상태이어야 함을 나타낸다. Group 2에 속한 node들은 T0 (taboo-0)의 값을 그리고 Group 3에 속한 node들은 T1 (taboo-1) 값이 주어지는데, 각각 논리값 0과 1을 가질 수 없음을 나타낸다. 마지막으로, Group 4에 속한 node들은 X, 즉 논리값 0 또는 1

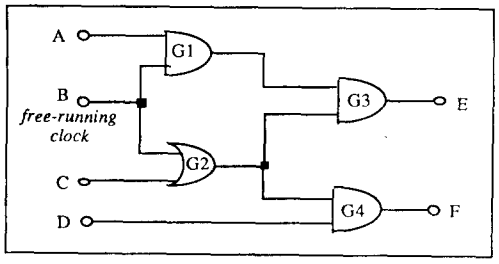


그림 2. 주입력의 논리값 제약의 전파
Fig. 2. Propagation of input constraints.

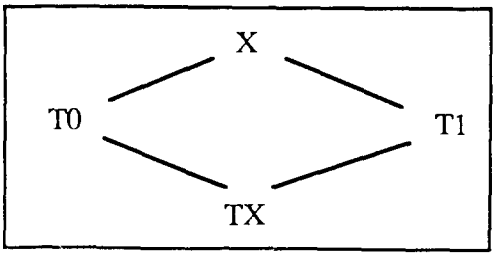


그림 3. Taboo logic value 의 Hasse diagram
Fig. 3. A Hasse diagram for taboo logic values.

을 제약없이 가질 수 있음을 나타낸다. Taboo logic value들의 Hasse diagram은 그림 3과 같다. 또한, taboo logic calculus에서 binary operation인 AND와 OR, 그리고 unary operation인 NOT이 표 1, 표 2 및 표 3과 같이 정의된다.

표 1. AND 소자 논리표
Table 1. AND gate calculus.

	X	T0	T1	TX
X	X	X	T1	T1
T0	X	T0	T1	TX
T1	T1	T1	T1	T1
TX	T1	TX	T1	TX

표 2. OR 소자 논리표
Table 2. OR gate calculus.

	X	T0	T1	TX
X	X	T0	X	T0
T0	T0	T0	T0	T0
T1	X	T0	T1	X
TX	T0	T0	TX	TX

표 3. NOT 소자 논리표
Table 3. NOT gate calculus.

X	T0	T1	TX
X	T1	T0	TX

이제 그림 2의 회로에 taboo logic 시스템을 적용하기로 한다. 주입력 B에 TX를 가한 후 논리 implication을 하면 그림 4와 같이 논리 소자 G1, G2, G3의 출력단들에 각각 taboo logic value인 T1, T0 및 T1으로 값이 정해지며 이 taboo logic value들로 부터 테스트 생성 과정시에 justification 될 수 없는 값들을 즉시 알게 된다. 여기서 한가지 주목할 점은 taboo logic value를 가진 node들의 실제값이 아직은 정해지지 않은 unknown 값이라는 점이다. 따라서, 그림 4에서 G1의 출력값인 T1은 논리값 0과는 의미가 다르다. T1은 주입력에 논리값 제약으로 인하여 논리값 1을 가질 수 없고 0은 가질 수도 있으나 아직은 justification이 되지 않은 상태이므로 실제값은 unknown 상태이다.

Taboo logic value 시스템은 종래의 논리값 시스템과 모순없이 함께 사용할 수 있다. 예를 들면, 종

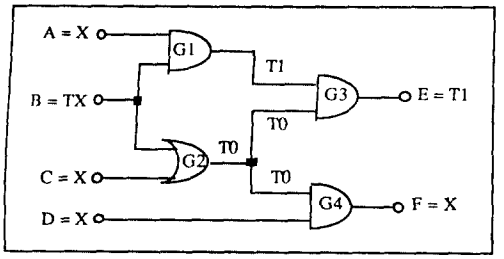


그림 4. Taboo logic value를 사용한 논리값 제약의 전파
Fig. 4. Propagation of constraints using taboo logic values.

래의 3-valued 논리 시스템은 논리값 0, 1, 그리고 U (unknown 또는 unspecified)를 포함하는데, 이때 U 대신 4-valued taboo logic 시스템을 사용하여 전체적으로 6-valued logic 시스템 (0, 1, T0, T1, TX, X)을 구성할 수 있다. 이와 같은 6-valued logic calculus는 4-valued taboo logic calculus에서 쉽게 도출할 수 있다.

2. Taboo Logic 시스템의 문제점

Taboo logic 시스템은 기본적으로 unknown 상태의 논리값에 대한 개념 확장이므로 종래의 3-valued 논리값 시스템 (0, 1, U)의 문제점인 loss of information[7], 즉 U-Uc (Uc는 U의 complement) 문제가 발생할 수 있다. 그러나 아직까지, loss of information을 완전히 해결하는 방법은 없으며, 여러가지의 부정확한 결과들이 테스팅 분야와 논리 검증 분야에서 노출되고 있다.[7, 8]

현재까지의 연구 결과, taboo logic 시스템을 사용할 경우 2가지의 문제, 즉 critical pessimism과 non-critical optimism이 노출되었다. Critical pessimism은 taboo logic 시스템에서 논리값 제약을 받는 입력단이 3-valued 논리값의 U-Uc 문제를 유발하여 비관적인 논리값 제약을 일으키는 경우인

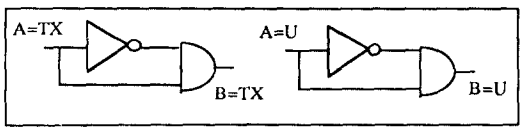


그림 5. 논리값 시스템에서의 정보 손실
Fig. 5. Loss of information in taboo logic system.

데, 그림 5.(a)와 같이 제약을 받는 주입력 A의 논리값에 상관없이 주출력 B의 논리값이 0이 되는 경우이다. 이 경우 B의 stuck-at 1 고장에 대해 비관적으로 테스트가 없음으로 선언된다. 그러나, 이 경우 3-valued 논리값을 사용하는 고장 진단 simulator를 사용하더라도 B에 위치한 stuck-at-1 고장은 검출할 수 없게 된다. 그 이유는 논리값 제약을 받는 주입력 A에 unknown 논리값인 U가 배정되어야 하므로 그림 5.(b)와 같이 U-Uc 현상에 의해 주출력단 B에 unknown 논리값인 U로 정해지기 때문이다.

Non-critical optimism 현상은 taboo logic calculus를 사용하여 추출된 논리값 제약이 충분하지 못하여 제약 조건을 검출하지 못하는 경우이다. 그림 6.(a)에서, exclusive-OR 소자의 입력단 A에 unknown 값인 U로 제약을 받을 때 (taboo value인 TX로 배정됨), exclusive-OR의 출력단 역시 unknown 논리값으로 제약을 받게 되어 TX 값으로 정해진다. 그러나, exclusive-OR 소자의 구현 방법에 따라, 그림 6.(b)와 (c)에서와 같이, 계산된 논리값 제약이 U-Uc 현상과 연동되어 loss of information을 일으켜 부분적인 논리값 제약만을 추출하게 된다. 일반적으로, 위와 같은 non-critical optimism은 테스트시에 taboo logic value의 implication을 통하여 해결될 수 있으며 결국에는 정확한 논리값 제약을 찾아내게 된다.

위에서 언급한 taboo logic 시스템의 제약에도 불구하고 생성된 테스트 패턴들은 주어진 논리값 제약하에서 항상 타당한 결과를 도출할 수 있으므로

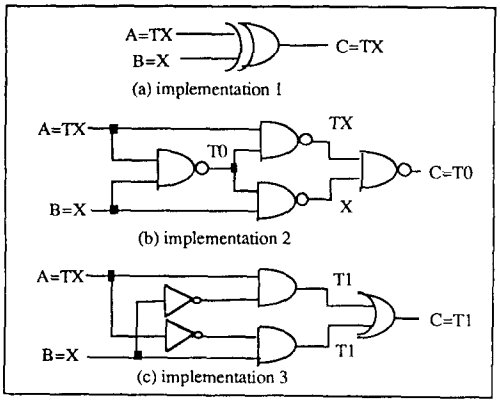


그림 6. Taboo llogic calculus에서의 non-critical optimism

Fig. 6. Non-critical optimism in taboo logic calculus.

```
constraints_propagation(list of constraints) {
  for each constrained primary input
      implication (constrained taboo value);
  for each signal with the logic value X
      find_additional_constraints();
}

find_additional_constraints() {
  implication (logic value 0);
  save_constraints(logic value 0);
  implication(logic value 1);
  save_constraints(logic value 1);
  analyze_constraints_and_accumulate();
}
```

그림 7. 논리값 제약 조건 전달 알고리즘

Fig. 7. Constraints propagation procedure.

taboo logic을 이용한 테스트 생성은 최소한의 안전한 fault coverage를 보장한다. 본 논문의 후반부에서 실험을 통하여 critical pessimism에 의해 untestable 고장으로 선언되는 경우가 매우 드물다는 사실을 보여준다.

한가지 여기서 언급할 필요가 있는 것은 최근의 symbolic method 연구의 진전인데, 특히 OBDD (Ordered Binary Decision Diagram) [9]과 제한된 symbolic evaluation 방법 [10] 등이 앞에서 토론된 taboo logic의 비관적인 요소를 감소시키는데 유용할 것으로 생각된다. 본 논문에서는 non-critical optimism을 줄이기위해 그림 7과 같은 특수한 제약조건 propagation 방법을 사용하였다. 이 방법은 SOCRATES [4]에서 사용된 learning 방식과 비슷하다.

Taboo logic value로 나타내진 논리값 제약을 taboo logic calculus를 사용하여 회로내에 전달시킨 후 부수적인 논리값 제약을 find_additional_constraints 루틴을 사용하여 검출한다. find_additional_constraints에서는 논리값이 X인 신호에 논리값 0 또는 1을 각각 배정하여 implication을 한 후 회로내의 모든 신호에 대한 implication 결과를 기록한다. 그 다음, 회로내의 각 신호에 대하여 구해진 두개의 implication 결과를 분석한 뒤, 새로운 논리값 제약이 발견되면 축적시켜 사용한다. 예를 들어, 회로내의 어느 신호에 대한 두개의 implication 결과가 T0와 1이라 할 때, 여기서 알 수 있는 것은 그 신호가 절대로 논리값 1을 가질 수 없다는 사실이다. 따라서 새로운 논리값 제약 T0가 추론된다. 표 4는 새로운 제약 조건을 추론하기 위해 사용된다. find_additional_contsraints 루틴은 특히 reconvergent fanout stem이 많은 회로에서 유용하다. 예를 들어, 그림 6.(b)와 (c)에서 주입력 B의

표 4. 새로운 논리값 제약을 추론 방법
Table 4. A table to deduce new constraints.

0/1 implication	0	1	X	T0	T1	TX
0	0	X	X	X	T1	T1
1	X	1	X	T0	X	T0
X	X	X	X	X	X	X
T0	X	T0	X	T0	X	T0
T1	T1	X	X	X	T1	T1
TX	T1	T0	X	T0	T1	TX

논리값에 관계없이 출력단 C가 taboo 논리값인 TX를 가지게 됨을 알게 된다.

Ⅲ. TABOO-ATPG 알고리즘

이 장에서는 taboo 논리값을 사용한 테스트 패턴 자동 생성 시스템인 TABOO-ATPG에 대하여 설명한다. TABOO-ATPG 시스템은 기본적인 구조는 PODEM의 implicit enumeration에 기초한 branch-and-bound search 알고리즘을 사용하였다. TABOO-ATPG에서는 6개의 논리값을 갖는 taboo logic value 시스템을 사용하였는데 논리값은 Ⅱ 장에서 설명한 바와 같이 0, 1, X, T0, T1, 그리고 TX들이다. 또한 taboo logic value의 효능을 극대화하기 위해 split circuit model [11] 에서와 같이 good circuit과 faulty circuit의 논리값을 각각 따로 표현하였다. TABOO-ATPG의 최상위 차원의 구조는 그림 8과 같다.

우선, preprocessing 단계에서 주입력단의 제약된 논리값의 전파를 통해 찾아진 taboo logic value로 전체 회로를 초기화된다. 그다음, PODEM과 같은 테스트 자동 생성 알고리즘에서와 같은 방식으로 objective를 한번에 하나씩 선정하여 논리값 justification과 implication을 통한 테스트 과정이 시작된다. 초기의 objective는 고장 발생지의 good circuit value인데, 이는 고장을 유발시키는 fault excitation 과정에 해당된다. 일단, 고장이 유발되면 그다음 objective는 good circuit과 faulty circuit에서의 논리값 차이점을 관찰할 수 있는 주출력단이나 scan flip-flop 입력단까지 전파시키는 fault propagation을 위한 신호값 선정을 위한 것이다. TABOO-ATPG 시스템은 종래의 테스트 생성 알고리즘과 비교하여 크게 세가지의 장점이 있다.

1. 진보된 untestable fault 검출

대부분의 untestable 고장들이 초기 objective 선정시에 검출된다. 예를 들면, preprocessing 단계에서 T1 값을 갖는 신호들의 경우 stuck-at-0 고장이 즉각적으로 untestable임을 알게된다. 그 이유는 고장 발생지에 고장을 유발시키는 논리값을 만족시키지 못하기 때문이다. 또한, 테스트 생성 과정에서 논리값 제약 조건들이 동적으로 전파되므로 대부분의 untestable fault가 조기에 검출된다.

```
TABOO-ATPG (input: fault under test, output: test)
{
  for each signal, initialize with the taboo logic value
        determined in the constraint propagation.

  while (status != TEST_FOUND && status != TEST_ABORTED)
  {
    objective = get_objective();  /* objective becomes empty */
          /* if a conflict occurs due to input constraints */
    if objective is empty,  status = backtrack();
    else {
      pi_value = enhanced_justification(objective);
      implication(pi_value);
      if((status = enhanced_x_path_check()) == BLOCKED)
          status = backtrack(); /* fault propagation is */
                /* not possible with current PI value */
       else if(status == FAULT_PROPAGATED_TO_PO)
          status = TEST_FOUND;
    }
  }
}   /* end of test generation */
```

그림 8. TABOO-ATPG 알고리즘의 최상위 구조
Fig. 8. Top-level description of the TABOO-ATPG algorithm.

2. 진보된 objective justification

TABOO-ATPG 시스템에서의 objective justification은 항상 논리값의 제약을 피해서 아직 논리값이 정해지지 않은 주입력단을 찾는 것이 보장되기 때문에 종래의 테스트 생성 알고리즘과는 달리 탐색 과정에서 모순이 발생하지 않는다. 이러한 특징은 잘못된 선택에 대한 backtrack 과정을 크게 감소시킬 수 있어 테스트 시간을 크게 절약할 뿐아니라, 고장 검출율도 크게 증진시키는 효과가 있다. 예를 들어, AND 소자의 출력단에 논리값 0을 justification 한다고 가정할때, 현재 출력단의 논리값은 unknown X 또는 T1이며 입력중에는 반드시 한개 이상의 입력에서의 논리값이 X 또는 T1이어야 한다. 그렇지 않을 경우, 그 이전의 테스트 과정에서 모순점이 발견되게 된다. 따라서, 논리값 0을 만족시키기 위한 주입력단으로의 path가 반드시 존재한다.

3. 진보된 고장 전달 경로 검색

주입력단에 새로운 논리값을 지정한 후, 논리값 implication을 시행한다. 그 다음에 고장 발생지에서부터 주출력단까지의 고장 전달 경로의 유무에 대한 검색이 실시되는데, PODEM 알고리즘에서는 이를 x-path check이라 부르며 고장 전달 경로의 신호값들이 아직 지정되지 않은 상태 (blocking되지 않은 상태) 임을 확인한다. 만약 path가 blocking되었으면 그전에 정해진 결정이 잘못되었으므로 시정하는 작업인 backtrack을 하게 된다. 종래의 테스트 알고리즘에서는 신호선이 good 0/faulty 0 또는 good 1/faulty 1의 경우에만 path blocking을 검출할 수 있지만, TABOO-ATPG 알고리즘에서는 고장 경로 검색이 더욱 확장된 개념으로 시행되어 조기에 blocking된 고장 전달 경로를 검출하게 된다. 예를 들면, 고장 전달 경로 상의 신호의 논리값이 good 1과 faulty T0이라고 가정하면 비록 T0가 unknown인 논리값이지만 T0의 의미가 논리값 0을 가질 수 없음을 뜻하므로 good circuit 논리값과 faulty circuit 논리값이 다를 수가 없게 되며 따라서 고장 효과가 그 path로는 전달될 수 없음을 알게 된다. 표 5는 path blocking 조건을 보여주고 있다.

표 5. Path blocking을 판단하는 조건
Table 5. A criteria to determine blocked paths.

G\F	0	1	X	T0	T1	TX
0	B	-	-	-	B	B
1	-	B	-	B	-	B
X	-	-	-	-	-	B
T0	-	B	-	B	-	B
T1	B	-	-	-	B	B
TX	B	B	B	B	B	B

앞에서 언급한 세가지 장점은 taboo logic value 시스템을 도입함으로써 얻을 수 있으며 특히 논리값 제약을 갖는 회로의 자동 시험 패턴 생성 문제에 효과적이다. 이밖에도 TABOO-ATPG 시스템에는 기존에 발표된 바 있는 learning, unique path sensitization 및 dominator등과 같은 효율적인 heuristics들이 적용되었다. [8] [13]

Ⅳ. Experimental Results

TABOO-ATPG 시스템은 C 언어로 구현되었으며 여러가지 실험들이 ISCAS85 benchmark 조합 회로에 대하여 Sparc-II workstation에서 시행되었다. 비록 ISCAS85 회로들이 논리값 제약을 갖는 실제 회로는 아니지만 주입력단에 논리값 제약을 가하기위해 임의의 주입력을 선택하여 그 입력을 clock 신호선으로 가정하여 테스트 동작시에 조절이 불가능한 입력, 즉 논리값을 배정할 수 없는 입력으로 사용하였다.

테스트 생성에 앞서서, fault list를 fault equivalence 개념을 사용한 fault collapsing 방법으로 준비하고, 각 fault에 대한 테스트 생성 과정에서 backtrack의 수를 최대 10으로 정하였다. 또한, TABOO-ATPG 시스템은 fault simulator로 PPSFP (parallel pattern single fault propagation) 방식 [12] 을 사용하였으며 논리값 제약을 받는 주입력에는 unknown 논리값 X로 고정시켜 수행하였다. 표 6은 ISCAS85 회로에 대한 테스트 생성 결과를 요약한 것이다. 고장 검출율 계산에서 논리값 제약으로 인한 untestable fault는 제외하였는데, 이는 그러한 fault들이 redundant fault가 아니고 단순히 회로내의 제약 조건에 의해 테스트가 불가능하기 때문이다. 따라서, TABOO-ATPG 시스템의 성능을 정확히 분석하기 위해 같은 ISCAS85 회로에서 제약 조건이 없는 경우의 테스트 결과도 괄호안에 기록하였다. TABOO-ATPG의 성능은 논리값 제약 조건의 유무의 차이에도 CPU 시간의 차이가 적으며 또한 논리값 제약에 따른 untestable fault를 매우 신속하게 검출하고 있다. C6288 회로를 제외하고는 모든

표 6. 논리값 제약을 받는 회로의 테스트 생성 결과
Table 6. Test generation results with constraints.
(Note:PDUF:pessimistically declared untestable faults.)

circuit name	# total faults	# redundant faults	# untestable faults	# aborted faults	fault coverage	CPU (seconds)	# PDUF
C432	524	2 (2)	44 (0)	2 (2)	91.2 (99.6)	0.7 (0.6)	1
C499	758	8 (8)	504 (0)	0 (0)	33.5 (100.0)	0.8 (0.6)	1
C880	942	0 (0)	60 (0)	0 (0)	93.6 (100.0)	1.9 (1.2)	0
C1355	1574	8 (8)	1228 (0)	0 (0)	22.0 (100.0)	8.2 (3.8)	0
C1908	1879	9 (9)	49 (0)	0 (0)	97.4 (100.0)	12.0 (5.2)	0
C2670	2747	117 (117)	100 (0)	0 (0)	96.4 (100.0)	17.9 (8.3)	36
C3540	3428	137 (137)	29 (0)	0 (0)	99.2 (100.0)	36.7 (22.4)	0
C5315	5350	59 (59)	50 (0)	0 (0)	99.1 (100.0)	18.6 (8.2)	0
C6288	7744	34 (34)	189 (0)	83 (0)	96.5 (100.0)	165.9 (18.2)	1
C7552	7550	131 (131)	27 (0)	0 (0)	99.6 (100.0)	35.2 (34.1)	0

회로들에서 Sparc-II workstation에서 40초 미만에 100% fault coverage (논리값 제약 조건이 없는 경우) 또는 모든 논리값 제약으로 인한 untestable fault들의 검출이 가능하다. 표 6에서 논리값 제약이 없는 상태에서의 자동 시험 패턴 생성 결과는 [4]와 [13] 에서 발표된 결과와 비교하여 고장 검출율 및 CPU 시간면에서 대등하거나 우월하였다. 한편, 논리값 제약이 있는 경우, 일부 회로 (C2670, C6288등)에서 논리값 제약이 없는 경우에 비해 많은 CPU 시간이 소모되었는데, 그 이유는 taboo logic calculus를 이용하여 불합리한 objective의 설정을 대부분 방지하였으나 II장에서 설명한 바와 같은 non-critical optimism 현상이 발생하여 테스트 생성 과정에서 정확한 논리값 제약을 찾아내기 위해 많은 시간이 소모되었기 때문이다.

표 6에는 taboo logic value의 loss of information에 의한 critical-pessimism을 분석하기 위해 untestable fault가 아님에도 비관적으로 untestable로 선언된 fault의 수를 수록하였다. 이를 위해, 논리값 제약이 하나의 주입력에만 있을 경우를 위한, symbolic test 생성 알고리즘을 구현하였다. Symbolic 테스트 알고리즘에서는 논리값 0, 1, U외에 2개의 새로운 논리값 S와 Sc가 사용되어 논리값 제약을 받는 주입력단의 unknown 논리값을 나타내게 되는데, 논리 관계식은 S 〈AND〉 Sc = 0, S 〈OR〉 Sc = 1, Sc = 〈NOT〉 S 이다. 또한, S와 Sc 논리값과 다른 논리값인 0, 1, U와의 논리식은 일반 3-valued 논리식과 같다. 이들 symbolic 논리값을 사용한 테스트 생성은 우선 제약받는 주입력에 논리값 S를 배정하고 나머지 과정은 종래의 테스트 생성 과정과 동일하다. 이들 두개의 논리값의 도입으로 논리값 제약에 따른 pessimism, 즉 loss of information을 방지하여 정확한 untestable fault를 검출할 수 있다. 그러나, 논리값 제약을 받는 주입력이 많을 경우, 각각 다른 symbolic 논리값을 사용하여야 하므로 실용적이지 못하나 [7], 여기서는 TABOO-ATPG와 비교할 목적으로 논리값 제약을 받는 주입력이 한개일 경우에만 사용되는 간단한 symbolic 테스트 알고리즘을 구현하였다. 표 6의 실험 결과를 분석하면 taboo logic value에 의한 pessimism이 거의 없다는 점이다. 반면, 이 symbolic 테스트 알고리즘은 같은 고장 검출율을 얻기위해 TABOO-ATPG에 비해 100배에서 10000배 이상의 CPU 시간이 걸렸다. C499, C1355 및 C6288 회로들의 경우, 최대 backtrack 허용치가 1000만번까지 걸렸다. 위의 실험으로 TABOO-ATPG의 성능이 우수함이 간접적으로 입증되었다.

두번째 실험으로 논리값 제약을 받는 주입력의 수

표 7. 제약된 주입력단수의 변화에 따른 테스트 결과

Table 7. C1908 test results with various number of constraints.

# of constrained inputs	# redundant faults	# untestable faults	# aborted faults	fault cover-age	CPU seconds
0	9	0	0	100.0	5.1
1	9	49	0	97.4	12.0
2	9	768	4	58.9	19.7
3	9	893	0	50.1	16.2
4	9	1035	0	42.1	19.2

표 8. find_additional_constraints의 효과

Table 8. The impact of find_additional_constraints.

experiment	# of redundant faults	# of untestable faults	# of aborted faults	# of backtracks	CPU seconds
I	8	1128	0	0	8.2
II	8	144	1080	11388	637.9

를 변화시켰을 때의 TABOO-ATPG의 성능을 측정하였다. 이 실험에서는 C1908 회로의 제약받는 주입력수를 0에서 4까지 변화시켜 untestable fault 수의 변화 및 논리값 제약이 회로의 testability에 미치는 영향을 분석하여 표 7에 수록하였다.

마지막으로 C1355 회로에서 find_additional_constraints의 영향을 분석하였다. C1355 회로는 C499와 동일한 회로로 단지 C499의 exclusive-OR 소자를 그림 6.(b)와 같은 4개의 NAND 소자로 대치한 것이다. 따라서, C1355 회로에서 계산된 논리값 제약은 non-critical pessimism에 해당한다. 테스트 생성 결과는 표 8에 요약되어 있는데, 실험 I은 find_additional_constraints을 사용한 경우이고 실험 II는 find_additional_constraints를 사용하지 않은 경우이다. 실험 결과에서 알 수 있듯이, find_additional_constraints의 사용은 backtrack 수 및 테스트 생성 시간의 현격한 감소 효과가 있다.

V. 결론

본 논문에서는 새로운 논리값 체계인 taboo logic value 시스템을 제안하였다. 또한, taboo logic value의 응용으로써 스캔 설계 방식의 위배로 주입력단에 논리값 제약을 갖는 회로에 대한 자동 시험 패턴 생성에 대하여 연구하였다. 이러한 taboo 논리값을 사용한 테스트 패턴 시스템인 TABOO-ATPG 시스템을 개발하여 스캔 설계된 회로에 적용하였다. ISCAS85 benchmark 조합 회로에 대한 실험 결과를 통해 스캔 설계 방식의 위배로 인한 주입력단의 논리값 제약이 전체 회로의 testability를 감소시킬 수 있음을 알 수 있었다. Taboo logic value 시스템의 장점은 테스트 생성 과정에서 달성될 수 없는 objective의 선택을 방지하고 논리상의 모순을 조기에 적발하여 전체 테스트 생성 시간의 단축과 높은 고장 검출율을 얻을 수 있다는 것이다.

Taboo logic value와 taboo logic calculus 개념은 비슷한 설계 제약 조건이 있는 다른 CAD 분야에도 유용하게 활용될 수 있을 것이다. 예를 들어, switch-level 또는 mixed-level 회로의 자동 시험 패턴 생성 문제에서도 본 논문에서 설명된 것과 같은 유사한 논리값 제약을 가질 수 있어 taboo logic value 시스템을 적용하여 높은 고장 검출율을 얻을 수 있다.[14]

參考文獻

[1] E. B. Eichelberger and T. W. Williams, "A Logic Design Structure for LSI Testability," *Proc.* 14th Design Automation Conf., pp.462-468, 1977.

[2] S. Funatsu, N. Wakatsuki, and T. Arima, "Test Generation Systems in Japan," *Proc.* Design Automation Symp., pp.114-122, 1975.

[3] P. Goel, "An Implicit Enumeration Algorithm to Generate Tests for Combinational Logic Circuits," *IEEE Trans.* Comput., vol.C-30, no.3, pp. 215-222, 1981.

[4] M. H. Schulz and E. Auth, "Improved Deterministic Test Pattern Generation with Applications to Redundancy Identification," *IEEE Trans.* on Computer-Aided Design, vol.8, no.7, pp.811-816, 1989.

[5] C. T. Glover, "Mixed-Mode ATPG under Input Constraints," *Proc. Int. Test* Conf., pp.142-151, 1990.

[6] F. Brglez and H. Fujiwara, "A Neutral Netlist of 10 Combinational Benchmark Circuits and a Target Translator in Fortran," *Proc. IEEE Int.* Symp. Circuits Syst., 1985.

[7] M. A. Breuer, "A Note on Three-Valued Logic Simulation," *IEEE Trans.*

Comput., vol.C-22, no.4, pp.399-402, 1972.

[8] K. T. Cheng, "On Removing Redundancy in Sequential Circuits," *Proc.* Design Automation Conf., pp.164-169, 1991.

[9] R. E. Bryant, "Graph-Based Algorithms for Boolean Function Manipulation," *IEEE Trans. Comput.*, vol.C-35, no.8, pp.677-691, 1986.

[10] J. L. Carter, B. K. Rosen, G. L. Smith, and V. Pitchumani, "Restricted Symbolic Evaluation is Fast and Useful," *Proc. Int. Conf.* on CAD, pp. 38-41,1989.

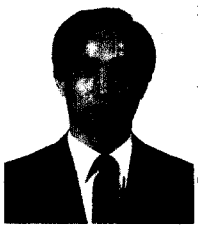
[11] W.-T. Cheng, "Split Circuit Model for Test Generation, "Proc. Design Automation Conf., pp.96-101, 1988.

[12] J. A. Waicukauski, E. B. Eichelberger, D. O. Forlenza, E. L. Lindbloom, and T. McCarthy, " Fault Simulation for Structured VLSI," VLSI Systems Design, pp.20-32, December 1985.

[13] J. A. Waicukauski et. al., "ATPG for Ultra Large Structured Designs," *Proc. Int. Test Conf.*, pp.44-51, 1990.

[14] E. S. Park and M. R. Mercer, "Switch-Level ATPG using Constraint-Guided Line Justification," *Proc. Int. Test Conf.*, pp. 616-625, 1993.

著者紹介



朴 恩 世(正會員)

1957年 11月 11日生. 1980年 2月 서울대학교 전기공학과 졸업 (공학사). 1982年 2月 한국과학기술원 전기 및 전자공학과 졸업 (공학석사). 1989年 8月 The University of Texas at Austin, 전기 및 컴퓨터 공학과 졸업 (공학박사). MCC 위촉연구원. Mentor Graphics사 연구원. Motorola사 연구원. 현재 한국전자통신연구소 책임연구원. 주관심 분야는 VLSI 설계, CAD 및 Testing 등임.