RTL 회로를 위한 테스트 용이도 기반 비주사 설계 기법

(A Non-Scan Design-For-Test Technique for RTL Controllers/Datapaths based on Testability Analysis)

김 성 일 † 양 선 웅 † 김 문 준 † 박 재 흥 †
(Sung Ill Kim) (Sun Woong Yang) (Moon Joon Kim) (Jae Heung Park)

김 석 윤 †† 장 훈 †††
(Seok Yoon Kim) (Hoon Chang)

요 약 본 논문에서는 RTL 회로에 대한 테스트 용이도 분석방식과 테스트 용이화 설계 방식을 제안한다. RTL 회로에 대하여 제어도와 관측도를 분석하고 테스트 용이도를 높이기 위하여 테스트용 멀티플렉서의 삽입 위치를 결정한다. 그리고 삽입해야 할 테스트용 멀티플렉서의 우선순위를 결정하여 우선순위가 높은 몇 개의 테스트용 멀티플렉서만을 삽입한다. 제안하는 테스트 용이화 설계 방식은 우선순위가 높은 멀티플렉서만을 삽입함으로써 면적 오버헤드를 최소할 수 있다. 실험을 통해 주사 방식을 적용했을 때보다 적은 면적 증가율을 보이며, 높은 고장 검출율과 테스트 패턴의 효율을 얻을 수 있다. 그리고 주사 방식에 비해 테스트 패턴을 삽입하는데 필요한 시간이 적음을 확인하였다.

키워드 : 테스트 용이도

Abstract Abstract This paper proposes a design for testability (DFT) and testability analysis method for register-transfer level (RTL) circuits. The proposed method executes testability analysis controllability and observability on the RTL circuit and determines the insertion points to enhance the testability. Then with the associated priority based on the testability, we insert only a few of the test multiplexers resulting in minimized area overhead. Experimental results shows a higher fault coverage and a shorter test generation time than the scan method. Also, the proposed method takes a shorter test application time required.

Key words : RTL, DFT, Testability, Testability

1. 서 론

칩의 집적도가 날로 커짐에 따라서 칩에 대한 높은 신뢰도 보장은 더욱 중요한 문제로 대두되고 있다. 또한, 칩의 전체 제조비용에서 테스팅이 차지하는 비율이 상대적으로 커지게 되었다. 이러한 칩의 신뢰도 보장 및 테스트 비용 감소를 위해서 테스트 접점 삽입(test point insertion)방식, 내장된 자체 테스트(BIST) 방식, 주사(scan) 방식과 같은 많은 테스팅을 고려한 설계(DFT) 방식들이 제안되었다. 그러나 이러한 방식들은 게이트 수준의 회로에 적용하기 위한 방식들이었기 때문에 회로에 집적되는 게이트 수가 커짐에 따라 위의 방식들을 적용하는데 걸리는 시간과 컴퓨터 자원이 방대해졌다.

이러한 문제점을 해결하기 위해서 최근에 게이트 수

† 비 회 원 : 숭실대학교 컴퓨터학부
kimsi72@samsung.co.kr
swyang@watt.ssu.ac.kr
mjkim@watt.ssu.ac.kr
jhpark@watt.ssu.ac.kr
†† 정 회 원 : 숭실대학교 컴퓨터학부 교수
ksy@computing.ssu.ac.kr
††† 비 회 원 : 숭실대학교 컴퓨터학부 교수
hoon@computing.ssu.ac.kr
논문접수 : 2001년 2월 5일
심사완료 : 2002년 11월 18일

준이 아닌 RTL(Register Transfer Level) 회로에 대해서 문제에 접근하려는 많은 연구들이 진행되어 왔다. RTL 회로는 그 회로의 게이트 수준보다 구성 모듈들의 숫자가 훨씬 적기 때문에 적은 시간과 비용으로 테스트 용이도를 분석할 수 있다는 장점이 있다. RTL 회로에 대한 테스팅을 고려한 설계방식으로서 내장된 자체 테스트 방식[1], 주사방식[2,3]과 같은 방식들이 제안되었다. RTL 회로에 대한 내장된 자체 테스트 방식의 경우 컨트롤러에서 데이타의 흐름에 대한 정보를 얻어내어 회로에 대한 테스트 용이도를 분석하였다. 그리고 테스트 패턴을 라이브러리(technology library)에 면적(area)과 지연(delay)과 같은 정보와 함께 저장한 후 RTL 회로를 테스트할 때 라이브러리에서 테스트 패턴을 불러들여 테스트를 수행하였다. 그러므로 회로의 신호폭(bit-width)이 증가하더라도 순차회로(sequential circuit)나 조합회로 combinational circuit)에 대해서 ATPG(Automatic Test Pattern Generation)를 수행할 때보다 적은 시간이 소요되었다. 그러나 라이브러리를 생성할 때 ATPG를 수행하여 각 RTL 모듈에 대한 테스트 패턴을 미리 생성해야 하기 때문에 라이브러리를 만드는 시간이 많이 필요하고 테스트 패턴까지 저장하고 있는 라이브러리가 매우 크다는 단점이 있었다. 또한, 주사 방식은 순차회로에 대한 ATPG에 비해서 테스트 패턴을 생성하는데 걸리는 시간이 적다는 장점이 있으나 동작속도 테스팅(at-speed testing)이 불가능하고 주사방식을 적용한 회로의 면적 오버헤드가 크며 테스트 패턴을 회로에 인가하는데 시간이 많이 걸리는 단점을 갖고 있다. 이외에도 약성 테스트 용이도(weakly testable)를 가지는 경로를 통해서 회로를 테스트하는 방식도 제안되었다[4]. 그러나 이 방식은 전체 RTL 회로가 구현됐을 때 반드시 추가되는 컨트롤러는 고려하지 않고 데이타패스만을 고려한 방식이었다. 그러므로 데이타패스에 대한 테스트를 수행할 때 컨트롤러로부터 데이타패스로 인가되는 제어신호들을 완벽하게 제어할 수 있다고 가정하였다. 그러나 이러한 가정은 전체 RTL 회로가 구현됐을 경우 불가능하게 되므로 제안된 방식을 적용할 수 없다. 한편, RTL 회로에 주사 방식을 적용하지 않고 테스트용 멀티플렉서를 삽입하여 테스트 용이도를 높이는 방식도 제안되었다[5]. 그러나 이 방식 역시 RTL 회로의 컨트롤러를 제외하고 데이타패스만을 고려한 방식이었다.

본 논문에서는 RTL 회로에 대해서 내장된 자체테스트 방식이나 주사 방식을 적용하지 않고 테스트 용이도를 향상시키는 방안을 제안한다. 즉 데이타패스에 대한 테스트 용이도를 분석하여 테스트용 멀티플렉서의 삽입 위치와 우선순위를 결정하며, 이때 컨트롤러에 대한 정보 없이 데이타패스만을 가지고 테스트 용이도를 분석한다. 테스트 용이도를 분석할 때 필요한 정보은 데이타패스와 그 데이타패스에서 어떠한 모듈이 기능 모듈(functional module)인가에 대한 것이다. 또한, 컨트롤러의 테스트 용이도를 향상시키기 위한 방안을 제안한다. 이를 통하여 컨트롤러와 데이타패스를 모두 포함한 전체 RTL 회로의 테스트 용이도를 향상시킬 수 있다. 따라서 데이타패스에 대한 테스트 용이도를 분석할 때 컨트롤러로부터 인가되는 제어신호를 완벽하게 제어할 수 있다고 가정하더라도 전체 RTL 회로에 대해서 원하는 결과를 얻을 수 있다. 그리고 본 논문에서 제안한 방식의 결과는 테스트용 멀티플렉서가 데이타패스 내에 삽입될 위치와 멀티플렉서의 삽입 우선순위(priority)가 주어진다. 삽입 우선순위가 높은 테스트용 멀티플렉서를 삽입하여 적은 면적 오버헤드를 가지면서 테스트 용이도가 높은 회로를 구현할 수 있다.

본 논문의 2장에서는 RTL 회로에 대해서 소개하며, 3장은 RTL 회로의 데이타패스를 위한 테스트용 멀티플렉서의 삽입 방식에 대해 설명하고 있다. 4장에서는 본 논문에서 제안하고 있는 데이타패스 내에서 테스트용 멀티플렉서의 삽입위치와 우선순위를 결정하는 방식을 적용한 회로에 대한 설명을 하고 있다. 5장은 본 논문에서 제안한 방식의 실험결과를 보여주고 있으며 결론은 6장에서 기술하였다.

2. RTL 회로의 구성

RTL 회로는 크게 컨트롤러와 데이타패스로 구성되어 있으며 컨트롤러는 데이타패스에서 필요한 제어신호들을 생성해 내는 역할을 한다. 그림 1은 RTL 회로의 구조를 보여주고 있다. 그림 1에서 보는 바와 같이 데이타패스를 제외한 나머지 부분이 컨트롤러를 구성하는 모듈들이며 컨트롤러는 다음상태를 만들어내는 조합회로와 현재상태를 저장하는 레지스터, 그리고 각 상태의 제어신호를 생성해내는 디코더로 구성된다.

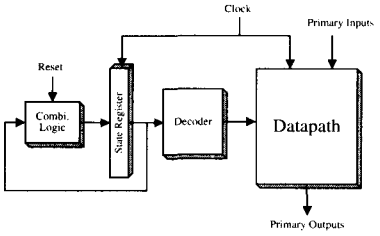

그림 1 RTL 회로의 구조

RTL 회로의 데이타패스는 주입력, 주출력, 기능모듈(덧셈기, 뺄셈기, 곱셈기 등), 멀티플렉서, 그리고 메모리 모듈(latch, register)로 구성된다. 즉, RTL 회로는 주입력의 집합 I, 주출력의 집합 O, 기능모듈의 집합 F, 메모리 모듈의 집합 M, 그리고 멀티플렉서의 집합 X로 구성된다. RTL회로에서 N을 주입력의 수, M을 주출력의 수, A를 기능모듈의 수, B를 메모리모듈의 수, 그리고 C를 멀티플렉서의 수라 할 때 각각의 집합 I, O, F, M, X는,

$I=\{ i_n \mid i_n$은 주입력, $1\le n\le N\}$ (N: 주입력의 수)

$O=\{ o_m \mid o_m$은 주출력, $1\le m\le M\}$ (M: 주출력의 수)

$F=\{ f_a \mid f_a$은 기능모듈, $1\le a\le A\}$ (A: 기능모듈의 수) (1)

$M=\{ m_b \mid m_b$은 메모리모듈, $1\le b\le B\}$

(B: 메모리 모듈의 수)

$X=\{ x_c \mid x_c$은 멀티플렉서, $1\le c\le C\}$

(C: 멀티플렉서의 수)

가 된다. 그리고, 기능모듈 f_a의 입력의 수를 K, 출력의 수를 P라고 할 때 f_a의 입력과 출력은 각각

$f_{a,i}=\{ i_{k,f_a} \mid i_{k,f_a}$는 f_a의 입력, $1\le k\le K\}$

(K: f_a의 입력의 수)

$f_{a,o}=\{ o_{p,f_a} \mid o_{p,f_a}$는 f_a의 출력, $1\le p\le P\}$

(P: f_a의 출력의 수) (2)

의 집합으로 나타낼 수 있다. 또한 RTL 회로에서 모든 기능모듈들의 입력과 출력의 집합은 각각

$$F_i=\bigcup_{a=1}^{A} f_{a,i}$$

$$F_o=\bigcup_{a=1}^{A} f_{a,o} \quad (3)$$

로 나타낼 수 있다. 내부 연결선(interconnection line)을 제외한 RTL 회로 전체를 집합 U라고 할 때 집합 U는

$U=\{u_j \mid u_j \in I\cup O\cup F_i\cup F_o\cup M\cup X,$

$1\le j\le (N+M+\sum_{a=1}^{A}|f_{a,i}|+\sum_{a=1}^{A}|f_{a,o}|+B+C)\}$ (4)

가 된다.

그림 2는 본 논문에서 사용한 실험회로 중의 하나인 Tseng[1]의 데이타패스를 보여주고 있다. Tseng의 데이타패스의 주입력, 주출력, 기능모듈, 메모리모듈 그리고 멀티플렉서는 식(1)에 의해 집합

$I=\{i_1, i_2, i_3\}=\{INPORT1, INPORT2, INPORT3\}$

$O=\{o_1, o_2\}=\{OUTPORT1, OUTPORT2\}$

$F=\{f_1, f_2, f_3, f_4, f_5, f_6, f_7\}=\{ADD1, ADD2,$

$\{ADD3, SUB1, MULT1, OR1, AND1\}$

$M=\{m_1, m_2, m_3, m_4, m_5, m_6\}=\{REG1, REG2,$

$\{REG3, REG4, REG5, REG6\}$

$X=\{x_1, x_2, x_3, x_4, x_5, x_6, x_7\}$

로 나타낼 수 있다.

멀티플렉서를 위한 제어신호인 m1~m7과 메모리모듈을 위한 제어신호인 load1~load6은 컨트롤러로부터 데이타패스로 인가되는 신호이다.

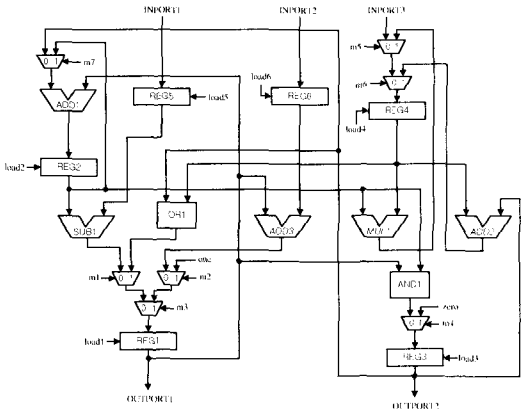

그림 2 Tseng의 데이타패스

3. 데이타패스를 위한 테스트용 멀티플렉서의 삽입 방식

그림 3은 본 논문에서 제안하는 테스트용 멀티플렉서의 삽입 방식을 보여주고 있다. 제안한 방식은 데이타패스의 구성모듈 중 기능모듈만을 기준으로 한다. 즉, 기능모듈에 대해서는 제어도와 관측도를 분석하지만 기능모듈에 비해 로직이 매우 간단한 메모리모듈과 멀티플렉서에 대해서는 제어도와 관측도 분석을 하지 않는다. 이는 테스트 용이도 향상을 위해서 삽입한 테스트용 멀티플렉서가 메모리모듈과 멀티플렉서의 테스트 용이도도 부가적으로 향상시키는 효과가 있기 때문이다.

기능모듈 f_a의 제어도와 관측도 분석은 RTL 회로를 구성하는 모듈 u_j들 간의 경로를 이용하며 RTL 회로에서 모듈 u_j들 간의 경로는 순서가 있는 n짝(ordered n-tuple)으로 나타낼 수 있다. 모듈 u_j들 간의 경로는 기능모듈 f_a에 대한 제어경로와 관측경로를 형성한다.

정의 3.1 기능모듈 f_a에 대한 제어경로(controllable path)는 집합 C_f이다.

$C_f=\{t_f \mid t_f$은 n짝$(n-tuple)$이고, t_f의 첫번째 원소$\in I,$

2번째부터 $n-1$번째원소$\in U-F$, n번째원소$\in f_{a,i}, 1\le l\le K\}$

정의 3.2 기능모듈 f_a에 대한 관측경로(observable

path)는 집합 O_f이다.

$O_f = \{t_g \mid t_g$는 n짝$(n-tuple)$이고, t_g의 첫번째 원소$\in f_{a,o}$, 2번째부터 $n-1$번째원소$\in U-F$, n번째원소$\in O$, $1 \le g \le P\}$

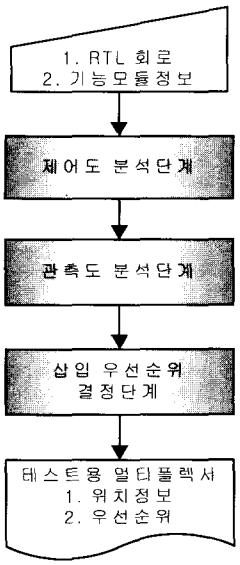

그림 3 데이타패스를 위한 테스트용 멀티플렉서 삽입방식

제어도와 관측도 분석을 통해 테스트 용이도가 낮다고 결정된 기능모듈들에 대해서는 테스트용 멀티플렉서를 삽입하여 제어경로와 관측경로를 만들어 주며 이렇게 함으로써 테스트 용이도를 높일 수 있다. 테스트용 멀티플렉서 삽입을 위한 방식을 적용하기 위해서는 다음의 두 가지 정보가 필요하다. 첫 번째로 필요한 정보는 테스트 용이도를 분석하고자 하는 RTL 회로의 데이타패스이다. 두 번째로 주어진 데이타패스에서 어떠한 모듈이 기능모듈인지에 대한 정보가 필요하다. 데이타패스에 대한 테스트 용이도 분석을 할 때 기능모듈의 입력단과 출력단이 각각 제어도와 관측도를 측정하는 기준이 되기 때문에 주어진 데이타패스에서 어떠한 모듈이 기능모듈인지를 우선 알고 있어야만 제어도와 관측도에 대한 분석을 수행할 수 있다.

그림 3의 테스트용 멀티플렉서 삽입을 위한 3단계를 마치면 삽입될 테스트용 멀티플렉서에 대하여 멀티플렉서들이 삽입될 위치와 테스트용 멀티플렉서들의 삽입 우선순위(priority)를 얻을 수 있다.

3.1 제어도 분석 단계

첫 번째 과정인 제어도 분석 단계는 RTL 회로의 주입력으로부터 각각의 기능모듈들의 모든 입력단으로 제어경로가 존재하는지를 분석한다. 즉, 집합 C_f의 원소인 n짝들의 마지막 원소들의 집합을 L_C라 할 때, 제어도가 낮은 기능모듈의 집합 B_f는

$$B_f = \{ f_a \mid f_a \in F \text{ and } f_{a,i} \notin L_C \} \quad (5)$$

로 나타낼 수 있으며 기능모듈 $f_a (\in B_f)$에 대해서는 제어도를 높이기 위해 테스트용 멀티플렉서를 삽입해야 한다.

제어경로에 대한 분석을 한 후 제어도 향상을 위해 추가할 테스트용 멀티플렉서를 RTL 회로의 어느 위치에 삽입할 것인가를 결정하는 것이다. 가장 간단한 방법은 제어도가 낮은 것으로 결정된 모든 기능모듈의 입력단에 테스트용 멀티플렉서를 위치시키는 것이다. 그러나 이 방법은 멀티플렉서의 삽입을 통해 향상되는 제어도의 효과를 극대화시킬 수 없게 된다. 그래서 다른 모듈들에 대한 영향도를 분석하여 삽입 우선순위가 높은 멀티플렉서를 삽입해야 한다. 멀티플렉서의 삽입 우선순위 결정에 대해서는 3.3절에서 설명한다.

3.2 관측도 분석 단계

두 번째 과정인 관측도 분석 단계에서는 기능모듈의 출력단으로부터 RTL 회로의 주출력으로의 관측경로가 존재하는지를 분석한다. 즉, 집합 O_f의 원소인 n짝들에서 첫 번째 원소들의 집합을 L_O라 할 때 관측도가 낮은 기능모듈들의 집합 B'_f는

$$B'_f = \{ f_a \mid f_a \in F \text{ and } f_{a,o} \notin L_O \} \quad (6)$$

로 나타낼 수 있으며 기능모듈 $f_a (\in B'_f)$에 대해서는 관측도를 높이기 위해서 테스트용 멀티플렉서를 삽입해야 한다.

제어도 분석 단계에서와 마찬가지로 관측도 분석 단계의 마지막 단계는 관측도 향상을 위해 추가할 테스트용 멀티플렉서의 위치를 결정하는 것이다.

3.3 테스트용 멀티플렉서의 삽입 우선순위 결정단계

제어도 분석 단계에 의해서 기능 모듈의 입력단에 추가될 테스트용 멀티플렉서들 중에서는 어떤 기능 모듈의 제어 경로를 형성하기 위해 추가된 멀티플렉서가 추가적으로 다른 기능 모듈의 제어 경로를 형성할 경우가 있다. 즉, 모든 추가되어야 할 멀티플렉서들 중에서 몇 개만을 추가함으로써 제어 경로가 존재하지 않는 모든 기능 모듈들의 경로를 형성할 수 있다. 관측 경로를 위한 멀티플렉서 역시 마찬가지로 우선순위에 의해 추가되어야 한다.

이 단계에서는 제어도 분석 단계와 관측도 분석 단계에서 제어도와 관측도 향상을 위해 추가하기로 결정한

테스트용 멀티플렉서들 중 데이타패스 내의 다른 모듈들에 대해서도 제어도나 관측도를 부가적으로 향상시키는 효과가 큰 순서로 삽입위치의 우선순위를 결정하게 된다.

제어도와 관측도 분석 단계에서 삽입 위치가 결정된 테스트용 멀티플렉서들의 집합을 T라 하면 집합 T는 $0<|T|\le(\sum_{a=1}|f_{a,i}|+\sum_{a=1}|f_{a,o}|)-(|L_C|+|L_O|)$를 만족한다. 여기에서 $|T|$는 집합 T의 원소의 개수를 나타낸다. 이 때 하나의 테스트용 멀티플렉서 $tx_h(\in T,\ 1\le h\le|T|$, 단 $|T|\neq 0)$의 추가로 새롭게 형성된 제어경로와 관측경로를 나타내는 n짝들의 원소들의 집합을 N_{tx_h}라 하면 tx_h의 삽입 우선순위 결정값은 정의 3.3에 의해 정의 3.4와 같이 정의된다. 이때 기능 모듈뿐만 아니라 멀티플렉서와 메모리모듈도 함께 고려하여야 한다. 이는 삽입된 테스트용 멀티플렉서가 기능 모듈뿐만 아니라 멀티플렉서와 메모리모듈에 대해서 부가적으로 테스트 용이도를 향상시키는 정도를 고려해야 하기 때문이다.

정의 3.3 n짝들을 원소로 갖는 어떤 집합 A에 대해서 $[A]$는 집합 A의 원소인 각각의 n짝들에 포함된 원소들의 합집합이다.

정의 3.4 테스트용 멀티플렉서 tx_h의 삽입 우선순위 결정값 P_{tx_h}는 $|[N_{tx_h}]|$이다.

높은 삽입 우선순위 결정값을 갖는 테스트용 멀티플렉서가 높은 삽입 우선순위를 갖게 되며 제어도와 관측도 향상을 위한 테스트용 멀티플렉서를 구분하지 않고 일괄적으로 우선순위를 결정한다. 이러한 일괄적인 방식을 채택할 경우 관측도를 향상시키기 위해 추가된 테스트용 멀티플렉서보다 제어도를 향상시키기 위해 추가된 테스트용 멀티플렉서에 높은 우선순위가 주어질 확률이 높게 된다. 이는 기능모듈을 기준으로 생각했을 때 관측도 향상을 위한 테스트용 멀티플렉서는 관측도가 낮은 기능모듈의 출력단의 관측도만을 높이지만 제어도를 높이기 위해 추가된 테스트용 멀티플렉서는 분기로 인해 제어도가 낮은 기능모듈의 입력단뿐만 아니라 메모리 모듈이나 멀티플렉서의 제어도까지 향상시키기 때문이다.

4. RTL 회로에 대한 테스트 용이도 분석방식의 적용

4.1 제어도(controllability) 향상을 위한 테스트 용이화 설계 기법

2장에서 언급했던 바와 같이 데이타패스을 구성하는 세 가지의 모듈 중에서 상대적으로 복잡한 로직을 갖고 있으며 따라서 많은 고장들이 집중되어 있는 모듈은 기능모듈이다. 따라서 테스트 용이도 분석은 데이타패스에 포함되어 있는 모든 모듈에 대해서 수행하지 않고 기능모듈에 대해서만 수행한다.

그림 2의 Tseng의 기능모듈 중 논리로직인 AND1과 OR1은 구성이 상대적으로 다른 기능모듈들에 비해서 간단하기 때문에 기능모듈이지만 테스트 용이도 분석을 위한 단위가 되지 않도록 하였다.

그림 4는 Tseng의 데이타패스의 주입력인 INPORT1과 INPORT2, 그리고 INPORT3으로부터 기능모듈들의 입력단으로 도달하는 8개의 제어경로 (i_1, m_5, i_{2,f_1}), $(i_2, m_6, i_{2,f})$, $(i_3, x_5, x_6, m_1, i_{2,f})$, $(i_3, x_5, x_6, m_1, f_6, x_1, x_3, m_1, i_{2,f_2})$, $(i_3, x_5, x_6, m_1, f_6, x_1, x_3, m_1, i_{1,f})$, $(i_3, x_5, x_6, m_1, i_{1,f})$, $(i_3, x_5, x_6, m_1, f_6, x_1, x_3, m_1, f_7, x_1, m_3, x_7, i_{1,f})$, $(i_3, x_5, x_6, m_1, f_6, x_1, x_3, m_1, f_7, x_1, m_3, i_{2,f_2})$를 보여주고 있다.

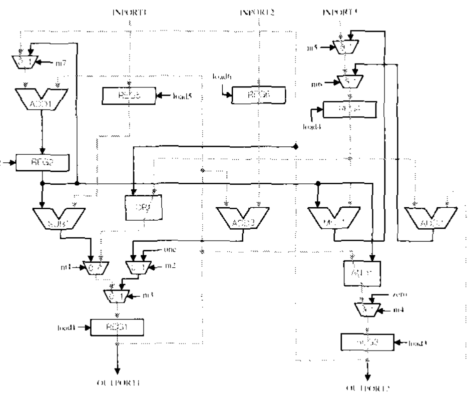

그림 4 주입력으로부터의 제어경로

만약 기능모듈의 모든 입력단에 RTL 회로의 주입력단으로부터의 경로가 존재한다면 그 기능모듈은 높은 제어도를 갖는다[6]. 기능모듈의 입력값을 주입력단으로부터 직접 제어할 수 있기 때문에 제어도 향상을 위해서 추가적으로 테스트용 멀티플렉서를 포함시킬 필요가 없다. 그러나 RTL 회로의 주입력단으로부터 기능모듈들의 입력단으로의 제어경로가 존재하지 않는다면 두 개의 입력단을 갖는 테스트용 멀티플렉서를 RTL 회로에 포함시켜야 할 필요가 있다. 이렇게 추가된 테스트용 멀티플렉서의 한쪽 입력단에는 기존에 존재하는 경로가 연결되고 나머지 한쪽 입력단에는 RTL 회로의 주입력단 중 하나가 연결된다. 그림 4를 보면 $B_i=\{f_1, f_2\}=\{SUB1, MUL1\}$는 제어경로가 존재하지 않는다. 따라서, 입력 i_{1,f_1}, i_{1,f_2}의 제어도 향상을 위한 테스트용 멀티플렉

서를 삽입해야 한다.

4.2 관측도 향상을 위한 테스트 용이화 설계 기법

그림 5은 Tseng의 데이타패스에서 뺄셈기 SUB1과 덧셈기 ADD3의 출력단으로부터 주출력단인 OUTPORT1과 OUTPORT2로의 2개의 관측경로 $(o_{1,f_1}, x_1, x_3, m_1, o_1)$, $(o_{1,f_1}, x_2, x_3, m_1, o_1)$를 보여주고 있다.

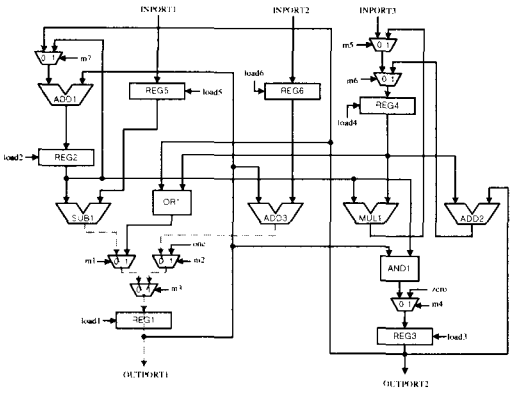

그림 5 주출력으로의 관측경로

그림 5에서 $f_{1,o} \equiv L_{O_1}$인 뺄셈기 SUB1과 $f_{5,o} \equiv L_{O_2}$인 덧셈기 ADD3는 데이타패스의 주출력으로의 관측경로가 존재하기 때문에 관측도 향상을 위한 테스트용 멀티플렉서가 필요하지 않다. SUB1과 ADD3를 제외한 나머지 기능모듈들 중 MUL1과 ADD2의 경우 이들 기능모듈들의 출력단으로부터 데이타패스의 주출력으로의 관측경로가 존재하지만 관측경로가 제어경로와 중복되기 때문에 관측경로를 형성하지 못한다. 이는 제어경로와 관측경로가 중복될 경우 기능모듈 내의 고장의 효과를 데이타패스의 주출력으로 전파하기 위해 필요한 멀티플렉서들의 제어신호가 중복된 값을 가져야 하기 때문에 원하는 패턴을 생성할 수 없게 되기 때문이다. ADD1의 경우 MUL1과 ADD2와 마찬가지로 제어경로와 관측경로가 중복되지만 경로형성을 위해 필요한 제어신호들이 동일한 값을 갖게 되므로 경로의 중복으로 인한 테스트용 멀티플렉서는 필요하지 않은 것으로 결정된다. 그러나 제어도 향상을 위해 추가된 테스트용 멀티플렉서의 삽입위치 때문에 관측도 향상을 위한 테스트용 멀티플렉서가 필요하게 된다. 그러므로 관측도가 낮은 기능모듈들의 집합은 $B_f = \{f_1, f_2, f_3\} = \{ADD1,\ \{ADD2, MUL1\}$로 나타낼 수 있으며 이 집합에 속하는 기능모듈들의 출력단인 $o_{1,f}$, $o_{1,f}$, o_{1,f_1}의 관측도를 향상시키기 위해서 테스트용 멀티플렉서를 삽입하여야 한다.

그림 6은 Tseng에서 제어도와 관측도가 낮은 기능모듈인 SUB1, MUL1, ADD1, ADD2를 위해 데이타패스에 삽입된 테스트용 멀티플렉서를 보여주고 있다. OR1과 AND1의 경우 기능모듈이지만 곱셈기나 덧셈기 등과 같은 기능모듈에 비해 간단한 로직을 갖고 있기 때문에 멀티플렉서나 메모리 모듈처럼 경로를 형성할 수 있는 모듈로 간주하였다.

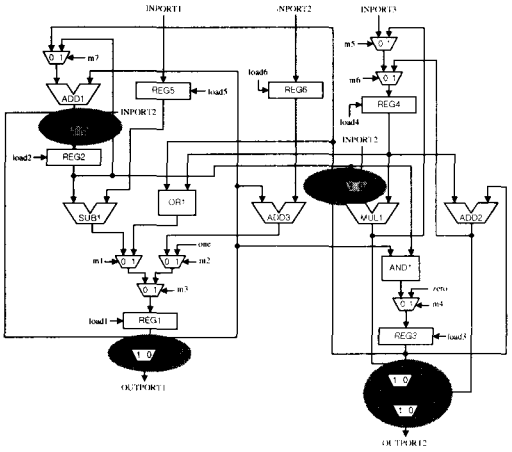

그림 6 SUB1, ADD1, ADD2, MUL1를 위해 추가된 테스트용 멀티플렉서

4.3 테스트용 멀티플렉서의 삽입 우선순위 결정

제어도와 관측도 분석 단계를 마치면 제어도와 관측도 향상을 위해 삽입해야 할 테스트용 멀티플렉서의 위치를 알게 된다. 마지막으로 삽입 위치가 결정된 테스트용 멀티플렉서 tx_k에 대한 우선순위 결정값 P_{tx_k}를 구하여 높은 우선순위 결정값를 갖는 테스트용 멀티플렉서를 데이타패스에 우선적으로 삽입한다.

그림 10에서 보여주고 있는 Tseng의 테스트 용이도 향상을 위해 삽입할 수 있는 테스트용 멀티플렉서 $T = \{tx_1, tx_2, tx_3, tx_4, tx_5\}$의 우선순위 결정값은 각각 $P_{tx_1} = 12$, $P_{tx_2} = 3$, $P_{tx_3} = 3$, $P_{tx_4} = 3$, $P_{tx_5} = 2$이며 i_{1,f_1}의 제어도 향상을 위해 삽입된 테스트용 멀티플렉서 tx_1이 가장 높은 삽입 우선순위 결정값을 갖는다.

5. 실험결과

본 논문을 위한 실험결과는 UltraSparc IIi 350MHz 프로세서와 512MByte의 메모리를 탑재한 Sun 마이크

로 시스템의 워크스테이션에서 게이트수준의 sequential ATPG tool인 Syntest의 asicgen[7]를 사용하여 얻어 내었다.

표 1은 본 논문에서 제안한 테스트용 멀티플렉서 삽입방식의 효과를 실험해보기 위해 선택한 RTL 회로들에 대한 특성을 나타내고 있다. 표 1의 두 번째 열에서 보는 바와 같이 chain_mult[8]를 제외한 나머지 회로들은 16 bits의 비트폭을 갖도록 구현하였다. 세 번째 열에 표시된 #Functional units는 각각의 회로에 포함돼 있는 기능모듈들의 수를 나타내고 있으며 네 번째 열에 표시된 #Test muxs는 추가된 테스트용 멀티플렉서의 수를 나타낸다. Paulin의 경우 하나의 테스트용 멀티플렉서만을 추가하였으며 Tseng의 경우 네 개의 테스트용 멀티플렉서를 추가했을 때를 실험했다. 괄호 안의 숫자는 본 논문에서 제안된 방식을 적용했을 때 삽입할 수 있는 테스트용 멀티플렉서의 최대수를 나타낸다. Chain_mult의 경우 최대 세 개의 테스트용 멀티플렉서의 삽입이 가능하며 실험에서는 각각 한 개와 두 개의 테스트용 멀티플렉서를 삽입했을 때의 실험결과를 도출하였다. Example2[9]의 경우 최대 두 개의 테스트용 멀티플렉서 중 한 개만을 추가하여 실험하였다.

표 1 실험 RTL 회로의 특징

Circuit	Characteristics			
	Bit width	# Functional Units	# Test mux2(possible choices)	
Paulin	16	4	1(7)	
Tseng	16	5	4(5)	
Chain_mult	8	2	1(3)	2(3)
Example2	16	2	1(2)	

표 2 고장 검출율과 테스트 패턴의 효율

Circuit	Fault Coverage(%)			Test Efficiency(%)		
	Original	Proposed (# of Test mux inserted)		Original	Proposed (# of Test mux inserted)	
Paulin	0	97.78(1)		24.24	99.91(1)	
Tseng	28.69	95.40(4)		35.04	99.85(4)	
Chain_mult	84.7	95.22(1)	95.47(2)	99.26	99.76(1)	99.84(2)
Example2	45.82	96.23(1)		56.73	99.42(1)	

표 2는 고장 검출율과 테스트패턴의 효율(test efficiency)을 나타내고 있다. DFT 방식을 적용하지 않았을 때의 고장 검출율과 테스트패턴의 효율은 Original 열에 각각 나타나 있으며 Proposed열은 본 논문에서 제안한 방식을 적용했을 때의 고장 검출율과 테스트패턴의 효율을 각각 나타내고 있다. 표 2에서 보는 바와 같이 삽입 가능한 모든 멀티플렉서를 삽입하지 않고 우선순위가 높은 몇 개의 멀티플렉서만을 삽입하여도 고장 검출율과 테스트패턴의 효율의 증가는 매우 크다는 것을 알 수 있다.

표 3은 본 논문에서 제안한 방식을 적용했을 때와 주사방식을 적용했을 때의 합성 후 면적 증가율을 보여주고 있다. 본 실험에서는 logic synthesis tool인 Synopsys의 design compiler[10]를 사용하여 실험회로들을 합성한 후 그 면적을 구했다. Partial scan과 Full scan은 각각 실험회로들에 부분 주사사슬(partial scan chain)과 완전 주사사슬(full scan chain)을 삽입했을 때의 면적 증가율을 보여주고 있다. 표 3의 마지막 열인 Proposed열은 본 논문에서 제안한 방식을 적용했을 때의 면적증가율을 나타내고 있다. 표 3에서 보여주듯이 주사 방식에 비해서 본 논문에서 제안한 방식의 면적 증가율이 매우 적음을 알 수 있다.

표 3 면적 증가율(%)

Circuit	Partial scan	Full scan	Proposed (# of Test mux inserted)	
Paulin	16.21	17	0.27(1)	
Tseng	11.39	16.75	4.9(4)	
Chain_mult	19.66	25.58	1.6(1)	3.42(2)
Example2	6.62	16.71	0.5(1)	

표 4 테스트 패턴 생성시간 (hour:minute:second)

Circuit	Original	Proposed (# of Test mux inserted)	
Paulin	01:09:17	00:06:00(1)	
Tseng	00:49:45	00:08:06(4)	
Chain_mult	00:00:22	00:00:35(1)	00:00:29(2)
Example2	00:01:11	00:01:44(1)	

표 4는 실험회로들에 대해서 Syntest를 이용하여 ATPG를 했을 때의 cpu time을 나타내고 있다. Original행은 본 논문에서 제안한 방식을 적용하지 않았을 때 실험회로에 대한 ATPG에 필요한 cpu time을 나타낸다. Proposed 행은 본 논문에서 제안한 방식을 적용했을 때 회로에 대한 ATPG에 필요한 cpu time을 나타낸다. 괄호 안은 실험회로에 삽입한 테스트용 멀티플렉서의 수를 나타낸다. Chain_mult의 경우 테스트용 멀티플렉서를 삽입했을 경우 오히려 ATPG를 위한 cpu time이 증가된 것을 알 수 있지만 표 2가 보여주는 바와 같이 고장 검출율과 테스트패턴의 효율은 많이 향상

되었다.

표 5는 실험회로들에 테스트패턴을 인가하는데 필요한 클럭 사이클의 수를 보여주고 있다. 표 5에서 보는 바와 같이 실험회로들에 주사방식을 적용했을 때에 비해서 본 논문에서 제안한 방식을 적용했을 때 테스트패턴을 인가하는데 필요한 클럭의 수가 적음을 알 수 있다.

표 5 테스트 패턴 인가시간 (# of clock cycles)

Circuit	Partial scan	Full scan	Proposed (# of Test mux inserted)	
Paulin	14873	8510	3900(1)	
Tseng	11752	10890	1272(4)	
Chain_mult	3520	1026	1950(1)	1386(2)
Example2	7385	3484	1182(1)	

6. 결 론

칩의 집적도가 커짐에 따라 칩의 신뢰도 보장과 테스트 비용 감소를 위하여 테스팅을 고려한 설계 방식들이 제안되었다.

본 논문에서는 RTL 수준 회로의 컨트롤러에 대한 정보 없이 데이타패스에 대한 정보만을 갖고 데이타패스에 대한 테스트 용이도를 향상시키는 방법을 제안한다. 테스트 용이도를 향상시키기 위하여 제어도 분석과 관측도 분석을 이용하여 제어도와 관측도가 낮은 기능모듈을 위해 테스트용 멀티플렉서를 삽입해야 할 위치를 찾아낸다. 그리고 삽입할 테스트용 멀티플렉서의 삽입 우선순위를 계산하여 우선순위에 의해서 멀티플렉서를 삽입하는 방식을 제시하였다. 멀티플렉서를 우선순위를 이용하여 모든 멀티플렉서를 삽입하지 않고 우선순위가 높은 몇 개의 멀티플렉서만을 삽입함으로써 면적 오버헤드를 줄일 수 있다.

실험결과를 통해 RTL 수준 회로의 데이타패스 정보만을 이용한 데이타패스의 테스트 용이도 분석을 통해서 적은 수의 멀티플렉서를 삽입함으로써 높은 고장 검출율을 얻을 수 있었다. 또한 테스트패턴의 효율, 짧은 테스트패턴 생성시간과 인가시간을 얻을 수 있었다. 그리고 주사방식에 비해서 본 논문에서 제안한 방식을 RTL 회로에 적용했을 때 합성 후 회로의 면적 오버헤드가 적음을 확인하였다.

참 고 문 헌

[1] I. Ghosh, N. K. Jha, S. hawmik, "A BIST Scheme for RTL Circuit Based on Symbolic Testability Analysis," IEEE Trans. on CAD, vol. 19, no.1, pp. 111-128, Jan. 2000.

[2] S. Bhattacharya, F.Brglez and S. Dey, "Transformations and Resynthesis for Testability of RTL Control-Data Path Specifications," IEEE Trans. VLSI Syst., vol. 1, pp. 304-318, Sept. 1993.

[3] S. Bhattacharya, S. Dey, "H_SCAN: A High Level Alternative to Full-Scan Testing with Reduced Area and Test Application Overheads," in Proc. VLSI Test Symp., pp. 74-80, 1996.

[4] S. Ohtake, M. Inoue, H. Fujiwara, "A Method of Test Generation for Weakly Testable Data Paths Using Test Knowledge Extracted from RTL Description," in Proc. Asian Test Symp., pp. 5-12, Dec. 1999.

[5] S. Dey, M. Potkonjak, "Non-Scan Design-For-Testability of RT-Level Data Paths," in Proc. Int. *Conf. on CAD, pp. 640-645, Nov.* 1994.

[6] I. Ghosh, A. Raghunathan, N. K. Jha, "A Design-for-Testability Technique for Register-Transfer Level Circuits Using Control/Data Flow Extraction," IEEE Trans. on CAD, vol. 17, pp. 706-723, Aug. 1998.

[7] SynTest User's Guide, Syntest, 1998.

[8] S. Ravi, I. Ghosh, R. K. Roy, S. Dey, "Controller Resynthesis for Testability Enhancement of RTL Controller/Data Path Circuits," in Proc. International Conference on VLSI Design, pp. 193-198, 1998.

[9] I. Ghosh, A. Raghunathan, N.K. Jha, "Design for Hierarchical Testability of RTL Circuits Obtained by Behavioral Synthesis," IEEE Trans. on CAD, vol. 16, no. 9, pp. 1001-1014, 1997.

[10] Design Compiler Tutorial, Synopsys, 1996.

김 성 일

2000년 숭실대학교 컴퓨터학부 학사. 2002년 숭실대학교 대학원 컴퓨터학과 석사. 현재 삼성전자 SOC 연구소 연구원. 관심분야는 컴퓨터구조, VLSI 설계 및 테스팅, CAD

양 선 웅

1996년 숭실대학교 전자계산학과 학사. 1998년 숭실대학교 대학원 전자계산학과 석사. 2002 숭실대학교 대학원 컴퓨터학과 박사. 관심분야는 컴퓨터 구조, VLSI 설계 및 테스팅, CAD

김 문 준

2000년 숭실대학교 컴퓨터학부 학사. 2002년 숭실대학교 대학원 컴퓨터학과 석사. 현재 숭실대학교 대학원 컴퓨터학과 박사 과정. 관심분야는 컴퓨터구조, VLSI 설계 및 테스팅, CAD

박 재 홍

1999년 숭실대학교 컴퓨터학부 학사. 2002년 숭실대학교 대학원 컴퓨터학과 석사. 2002년~숭실대학교 대학원 컴퓨터학과 박사과정. 관심분야는 컴퓨터 구조, CAD, VLSI 설계, VLSI 테스팅.

김 석 윤

1980년 서울대 공대 전기공학과 학사. 1990년 University of Texas at Austin 전기, 컴퓨터공학과 석사. 1993년 University of Texas at Austin 전기, 컴퓨터공학과 박사. 1982년~1987년 한국전자통신 연구소 연구원. 1993년~1995년 Motorola Inc., Senior Staff Engineer. 1995년~현재 숭실대학교 컴퓨터학부 교수. 주관심 분야는 설계 자동화, VLSI 회로해석 및 설계, 통신 시스템

장 훈

1987년 서울대학교 전자공학과 학사. 1989년 서울대학교 전자공학과 석사. 1993년 University of Texas at Austin 박사. 1991년 IBM Inc. 1993년 Motorola Inc. Senior Membger of Technocal Staff. 1994년 ~ 숭실대학교 컴퓨터학부 조교수. 관심분야는 컴퓨터 시스템, VLSI 설계, VLSI 테스팅