論文2002-39SD-12-6

System-On-a-Chip(SOC)에 대한 효율적인 테스트 데이터 압축 및 저전력 스캔 테스트

(Low Power Scan Testing and Test Data Compression for System-On-a-Chip)

鄭 俊 模 *, 鄭 正 和 **

(Jun Mo Jung and Jong Wha Chong)

요 약

System-On-a-Chip(SOC)에 대하여 테스트 데이터 압축 및 저전력 스캔테스팅에 대한 새로운 알고리즘을 제안하였다. 스캔벡터내의 don't care 입력들을 저전력이 되도록 적절하게 값을 할당하였고 높은 압축율을 갖도록 적응적 인코딩을 적용하였다. 또한 스캔체인에 입력되는 동안 소모되는 scan-in 전력소모를 최소화하도록 스캔벡터의 입력 방향을 결정하였다. ISCAS 89 벤치마크 회로에 대하여 실험한 결과는 평균전력 소모는 약 12% 감소되었고 압축율은 약 60%가 향상됨을 보였다.

Abstract

We present a new low power scan testing and test data compression method for System-On-a-Chip (SOC). The don't cares in unspecified scan vectors are mapped to binary values for low power and encoded by adaptive encoding method for higher compression. Also, the scan-in direction of scan vectors is determined for low power. Experimental results for full-scanned versions of ISCAS 89 benchmark circuits show that the proposed method has both low power and higher compression.

Keyword : 테스트 데이터 압축, 저전력, 스캔테스트

Ⅰ. 서 론

System-On-a-Chip(SOC)는 여러개의 기설계되어진 코어(core)들로 구성된다. 따라서 SOC를 테스트 하기 위해서는 각 코어들을 효율적으로 테스트하기 위한 전략이 필요하다. 각 코어를 테스트 하기 위한 테스트 데이터들이 해당 코어의 입력을 통해 인가되고 SOC 출력을 통해 테스트 출력이 관찰될 수 있도록 해야 한다 <그림 1>. 또한 Intellectual Property(IP)코어들을 효과적으로 테스트하기 위해서는 IP 코어용 테스트 데이터들이 충분히 인가되어야 하므로 코어 개발자에 의해서 제공되는 테스트 데이터의 양은 많아진다. 따라서 많은 IP 코어로 구성된 SOC를 테스트 하는데 필요한 테스트 데이터의 양도 커지게 된다.

테스터 장비(ATE:Automatic Test Equipment)의 테스트 데이터 전송속도, 채널용량 및 메모리는 한계가

* 正會員, 金浦大學 電子情報系列
(Kimpo College Dept. of Electronic Information)
* 正會員, 漢陽大學校 情報通信大學 情報通信學部
(Hanyang University College of Information and Communication. Dept. of Information and Communication)

※ 이 논문은 2002학년도 김포대학의 연구비지원에 의하여 연구되었음
接受日字:2002年9月18日, 수정완료일:2002年11月28日

있기 때문에 테스트 데이터의 양이 너무 많으면 테스트 시간이 길어진다. SOC의 테스트 시간은 테스트 데이터의 양, 테스트 데이터를 코어에 전송하는 시간, 테스트 데이터의 전송률에 따라 다르고 또한 코어가 스캔(scan)회로로 구성되어 있다면 최대 스캔 체인길이(chain length)에 의해서도 많이 좌우된다.

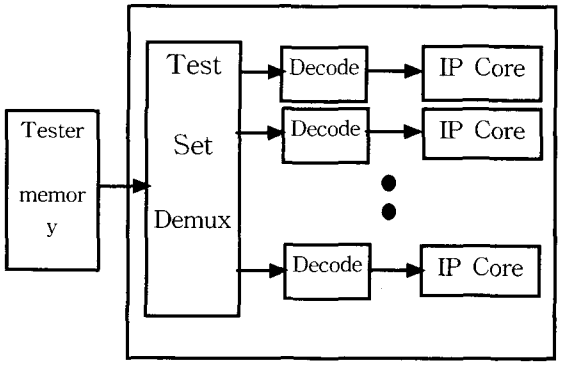

그림 1. SOC에 대한 테스트 아키텍처
Fig. 1. Test Architecture for SOC testing.

테스트 시간은 테스트 비용과 밀접하게 연관되어 있기 때문에 테스트 시간이 증가하면 테스트 비용이 늘어나고 칩 단가에도 영향을 준다. 그러므로 테스트 시간 및 테스터의 메모리를 줄이기 위해서는 테스트 데이터 양을 줄일 필요가 있다.

테스트 데이터의 양을 줄이는 문제에 대한 여러 가지 해결방법이 제안되었는데 그중 하나가 Built-In Self-Test(BIST)이다. 이것은 온칩(On-Chip) 하드웨어를 이용하여 코어를 테스트하므로 테스트 시간이 단축된다. 그러나 기존의 IP 코어들은 대부분 BIST로 구현되지 않았기 때문에 이 방법을 적용하기 위해서는 회로를 많이 수정해야 한다.

테스트 시간을 줄이는 또 다른 방법으로 테스트 데이터를 압축하는 것이다. IP 코어에 대한 테스트 데이터를 보다 작은 테스트 데이터로 압축함으로서 데이터의 양을 줄인다. 압축된 테스트 데이터는 IP 코어에 있는 디코더에 의해서 원래의 데이터로 디코딩(Decoding)되어 IP 코어에 인가된다. 먼저, 통계학적 코딩(statistical coding)을 사용하여 압축하는 방법으로서 non-scan 회로에 대한 테스트 데이터 압축[1,2] 및 full-scan 회로에 대한 테스트 데이터 압축[3] 방법등이 제안되었다. [1,2]의 방법은 플립플롭수가 많고 상대적으로 적은 주입력(primary input)을 갖는 순서회로에 적합하다. [3]의 방법은 통계학적 코딩이 Full-scan 회로에 대한 표준 ATPG 압축 방법들[4,5]보다 더 많은 압축을 얻을 수 있다는 것을 보여주진 못했다.

[6]에서는 테스트 패턴을 인가할 때 연속적인 테스트 패턴(pattern)들이 오직 작은 수의 비트들만 다르다는 것을 이용하였다. 이웃한 테스트 패턴으로부터 얻어진 차(差)패턴(Difference Pattern)을 Run-length Coding을 기반으로 압축하였다. 이 방법은 Variable-to-fixed length code로서 일반적인 Variable-to-variable length codes[7,8]보다는 효과가 덜하다. 또 다른 방법으로 Golomb code를 이용하는 방법이 제안되었다.[9] 이 방법은 Difference pattern내의 0's run(연속적인 0)을 가변길이(variable-length) codeword로 변환하여 보다 큰 압축을 얻었다. 또한 새로운 종류의 Variable-to-variable length 압축 code[10]가 제안되었는데 이것은 0's run의 분포를 이용하는 방법으로 Frequency-directed Run-length(FDR) Code라 불리며 Golomb Code보다 좋은 압축효율을 갖는다.

테스트 시간에 대한 문제점 외에 SOC 테스트에 있어서 다른 중요한 사항은 전력소모이다.
테스트 모드에서는 정상동작 모드에 비하여 전력소모가 많이 증가[11~13]하는데 테스트 모드에서는 가능한 많은 신호선들이 더 많이 스위칭 하도록 하여 한번에 많은 고장을 검출하기 때문이다. 이 전력소모 문제를 해결하기 위하여 전력제한 스케쥴링(Power Constrained Test Scheduling)방식[14]이 제안되었고 연속적인 테스트 패턴간의 천이(Transition)가 적도록 테스트 패턴을 생성해주는 새로운 ATPG도 제안되었다.[15]

또한 full-scan 회로에 대한 전력소모 감소를 위한 새로운 방법들이 제안되었다.[16]

지금까지 기술한 바와 같이 테스트 데이터의 양과 전력소모는 매우 중요한 문제로 대두되고 있는데 SOC 테스트, 그 중에서도 full-scan 구조의 테스트에 대해서도 많은 해결방법이 제안되었다. 그러나 테스트 데이터 및 전력소모를 동시에 줄이지는 못하였다. 스캔벡터를 저 전력용으로 생성하면 테스트 데이터의 양이 많이 증가하고,[15] 정적 압축방식으로 스캔벡터를 압축하여 생성하면 전력소모가 크게 증가하였다.[16] 또한 don't care 입력을 갖는 스캔벡터에 대한 저전력 기법이 제안[17]되었으나 테스트 데이터의 양이 크게 증가하였다.

본 논문에서는 SOC내의 스캔 테스트에 있어서, 전력소모와 스캔 벡터의 양을 동시에 줄이는 새로운 기법

을 제안하였다. 전력소모는 스캔벡터내의 천이 수와 스캔벡터가 스캔체인으로 입력하는 동안 발생되는 천이 수에 비례한다. 스캔 벡터에 있는 don't care 입력을 효율적으로 0 또는 1로 매핑(Mapping)하여 스캔 벡터내의 천이 수를 줄였고 또한 스캔 체인으로 스캔벡터가 입력할 때 소모하는 scan-in 전력소모를 줄이기 위해 스캔 입력하는 방향을 구분하여 발생되는 천이 수를 최소화하였다.

저전력 매핑된 스캔벡터를 압축하기 위하여 0's run을 및 1's run을 적응적으로 이용하여 인코딩하였다.

본 논문의 구성은 다음과 같다. 먼저 II장에서는 기존의 압축코딩 방법과 전력소모를 정의하고 III장에서는 저전력 매핑 및 적응적 인코딩을 기술하였고 IV장에서는 적용된 알고리즘을 설명하였다. V장에서는 실험결과를 나타내었다.

II. 기존의 압축 코딩 방법과 scan-in 전력소모의 정의

1. 전력소모

CMOS 회로의 전력소모는 크게 정적 및 동적 전력소모로 분류할 수 있다. 정적 전력소모는 동적(Dynamic) 전력소모에 비하여 양이 작기 때문에 무시한다. 논리게이트 출력의 스위칭에 의해서 발생하는 동적 전력소모는 스위칭 빈도 수, 즉 천이 수에 비례한다. 일반적으로 CMOS 디지털 회로의 동적 전력소모는 다음과 같이 정의된다.

$$P_d = 0.5 C_{load} V_{dd}^2 F_c N_g$$

여기에서 Cload는 게이트 출력의 Load Capacitance, Vdd는 공급전압, Fc는 인가되는 클록주파수 그리고 Ng는 게이트 출력단의 천이 개수(transition number)이다. 이 천이개수는 전력소모의 중요변수이다. full-scan 회로를 테스트하는 동안에 소모되는 전력은 스캔벡터가 스캔체인에 입력하는 동안 발생되는 스캔셀의 Ng에 비례한다. 따라서 스캔테스트 하는 동안의 전력소모를 천이개수로 모델링할 수 있으며 3.1절에서 자세히 기술하였다.

2. Golomb Code를 이용한 인코딩

Golomb Codes를 이용하여 스캔벡터를 인코딩하는 방법은 [9]에서 제안되었다. 이 방법은 기존의 run-length 인코딩[6]보다 더 좋은 압축율을 갖는다. [6]에서는 연속적인 0의 길이(run-length)를 고정된 비트로 표현하는 방법이다. 예를 들어 "000000"는 0의 run-length가 6이므로 만약 3비트로 표현한다면 "110"이 된다. 이 경우 원래 데이터보다 3비트가 줄어든다. Golomb Codes는 0의 run-length를 고정된 비트로 표현하지않고 가변길이 코드로 표현하는 방법이다. 인코딩 하려는 데이터를 D라 하면 인코딩 과정은 다음과 같다. D에 있는 임의의 run-length는 group prefix 와 tail이라는 항으로 조합된 코드워드로 변환된다.

group의 크기를 m 이라 하면 D에 있는 0의 run-length들이 m의 크기를 갖는 여러 개의 group으로 나뉜다. 예를 들어 A1 group은 {0,1,,,m-1}의 run-length로 구성되고 A2 group은 {m, m+1, m+2,,, 2m-1}의 run-length로 구성된다. 즉, AK group은 {(k-1)m, (k-1)m+1, (k-1)m+2,,,km-1}의 run-length로 구성된다. Group Prefix는 이 group들을 구분하기 위한 코드이고 또한 group내에서 각 run-length를 구분하기 위한 코드가 tail이다. <그림 2>에 m이 4인 경우의 Golomb codes를 이용한 인코딩이다.

Group	Run-length	Group Prefix	Tail	Codeword (G.P & Tail)
A1	0	0	00	000
	1		01	001
	2		10	010
	3		11	011
A2	4	10	00	1000
	5		01	1001
	6		10	1010
	7		11	1011
A3	8	110	00	11000
	9		01	11001
...	...	...	...	...

그림 2. Golomb codes
Fig. 2. Golomb Codes.

3. FDR(Frequency-Directed Run-length) 인코딩

FDR codes는 0의 run-length의 분포에 따라 group의 크기를 조절한 것이다. <그림 3>은 FDR Codes의 예를 나타내었다.

Golomb Codes와는 달리 각 group에 속한 run-

length의 길이를 가변시켰다. 즉, 자주 발생하는 run-length에는 보다 짧은 코드워드를 할당함으로써 압축효율을 높였다. 이 방식은 Golomb Codes를 이용한 경우보다 평균 3%정도 좋은 압축율을 갖는다.

Group	Run-length	Group Prefix	Tail	Codeword (Group & Tail)
A1	0	0	0	00
	1		1	01
A2	2	10	00	1000
	3		01	1001
	4		10	1010
	5		11	1011
A3	6	110	00	11000
	7		01	11001
	8		10	11010
	9		11	11011
...	...	...	...	...

그림 3. FDR Codes
Fig. 3. FDR Codes.

Ⅲ. 제안된 저전력 매핑(Mapping) 및 적응적 인코딩

1. 스캔 벡터내의 don't care 입력들에 대한 저전력 매핑

본 절에서는 full scan 회로에 대하여 scan-in 테스트를 하는 동안에 소모되는 scan-in 전력소모 모델 및 스캔벡터내에 존재하는 don't care 입력을 효과적으로 0 또는 1로 매핑하는 방법을 기술한다.

먼저 scan-in 전력소모 모델에 대하여 기술한다.

scan-in 테스트의 전력소모는 스캔벡터가 스캔체인으로 입력하는 동안 체인에서 발생하는 천이로 인한 동적 전력소모(scan-in 전력소모)와 테스트 응답을 출력하는 동안 체인에서 발생하는 전력소모(scan-out 전력소모)로 나눌수 있다. 그러나 스캔벡터로부터 직접 scan-out 전력소모를 계산하는 것이 어렵기 때문에 본 논문에서는 scan-in 전력소모만을 고려하였다.

scan-in 전력소모는 [16]에서 제안한 Weighted transitions metric(WTM: 가중 천이 메트릭)모델을 이용하여 측정하였다. 이것은 스캔벡터의 천이위치를 알면 scan-in 하는 동안 발생하는 천이 수를 측정할 수 있다. 예를 들어 스캔 벡터 S1S2S3S4S5=10101이고 스캔체인 길이가 5인 경우를 고려하자. 가장 왼쪽에 있는 S1이 스캔체인에 먼저 입력된다고 가정하면, S1과 S2에서 발생된 천이(1→0)는 스캔체인을 통과하면서 총 4번의 천이가 발생한다. 왜냐하면 S1과 S2에서 발생된 천이는 클록이 발생할때마다 (스캔체인의 크기 - 1) 만큼 이동하기 때문이다. 또한 S2와 S3에서 발생한 천이(0→1)는 3번의 천이를 발생한다. 이와같이 SJ, SJ+1에서 발생한 천이는 (스캔체인의 길이 - J) 만큼의 천이를 발생시킨다.

스캔 체인의 길이를 K라 하면, 각 스캔벡터 SV는S1, S2 , , , SK로 구성된다.

이때 각 스캔벡터가 스캔체인에 입력되는 동안 소모되는 전력소모를 WTM으로 표현하면 다음과 같다.

$$WTM(SV) = \sum_{j=1}^{k-1}(S_j \oplus S_{(j+1)})(k-j) \qquad (1)$$

또한 테스트에 이용되는 스캔벡터의 집합 SV_set = { SV1, SV2, , , SVn} 이라하면, SV_set이 모두 스캔체인에 입력하는 동안 소모되는 전력소모는 다음과 같다.

$$\text{WTM(SV\_set)} = \sum_{i=1}^{n} WTM(SV_i) \qquad (2)$$

식 (2)는 스캔벡터들을 scan-in 하는 동안 소모되는 전체 전력소모이다. 따라서 평균 전력소모 및 최대치 전력소모는 식 (3)과 같이 나타낼 수 있다.

$$\begin{aligned} Pavg &= \text{WTM(SV\_set)}/n \\ Ppeak &= \text{Maximum(SVj)} \end{aligned} \qquad (3)$$

지금부터 스캔벡터내에 존재하는 don't care 입력의 효율적 매핑(Mapping) 기법에 대하여 기술한다. 기존의 방법에서는 높은 압축율을 얻기 위하여 don't care 입력을 0만으로 매핑하였다.[9] 그러나 이 경우에는 천이수가 증가하여 전력소모가 증가하는 문제점이 있었다. 또한 전력소모를 최소화하기 위하여 don't care 입력의 이웃한 값들을 복사하여 매핑[17]하는 방법이 제안되었다. 이웃한 비트와 동일한 값으로 복사되므로 천이가 발생하지 않아 전력소모는 감소한다.

제안한 방법에서는 스캔벡터의 패턴을 참고하여 가능한 천이수가 적고 run-length가 길도록 효율적인 매핑을 한다. 이 방법의 대 전제는 기존의 0's run 뿐만 아니라 1's run을 이용하여 인코딩 한다는 점이다. 즉 0의 run-length뿐 아니라 1의 run-length를 이용하여

인코딩한다는 점을 충분히 활용한 방법이다.

스캔벡터내의 don't care 입력을 X라 하고 don't care 입력에 매핑되어야 할 로직 값을 A∈{0,1}라 하고 A의 보수를 A'라 하자.

다음과 같은 2가지 경우를 고려하여 don't care X에 값을 할당할수 있다.

경우 1 : 스캔 벡터가 "XXXA---" 또는 "---AXXX" 인 경우

이 경우에는 X들은 A로 매핑되어진다. 예를 들어 스캔벡터가 "XXX1000"인 경우(A=1)에 X는 1로 매핑되어 1111000이 된다. 이 경우 벡터내의 천이 수는 1이다. 만약 X에 0이 할당된다면 0001000이 되어 천이 수는 2가 되어진다.

경우 2 : 스캔벡터내의 연속적인 비트 열의 구조가 (A의 run-length가 m)(X의 run-length가 p)(A'의 run-length가 n) 으로 구성된 경우,

1) (m+p) ≥ (p+n) 이면 X들은 A로 할당

2) (m+p) 〈 (p+n) 이면 X들은 A'로 할당

이 경우는 don't care X 좌우에 서로 다른 run-length를 갖는 값으로 구성된 경우이다. 이런 경우에는 X가 어떤 값으로 할당이 되어도 천이 수는 동일하다. 하지만 압축률을 높이기 위해서는 가능한 run-length의 길이가 길도록 X를 할당해야 하기 때문에 좌, 우의 run-length를 고려해야 한다. 예를 들어 스캔벡터가 "000XX100"인 경우, A는 0, m은 5, p는 2, A'는 1이고 n은 1이다. 따라서 m+p = 5 이고 p+n은 3이기 때문에 X를 0으로 매핑하여 00000100이 되고 0의 run-length는 5가 된다. 이와 같은 방법으로 매핑하면 run-length가 길어지게 되므로 높은 압축률을 얻을 수 있다.

2. scan-in 방향을 고려한 저 전력 기법

3.1절에서 기술한 바와 같이 스캔벡터의 scan-in 전력소모는 스캔벡터가 스캔체인에 입력하면서 발생되는 총 천이의 개수에 비례한다. 또한 총 천이개수는 각 스캔벡터에서 발생된 천이의 위치와 밀접하게 관련되어 있다.

스캔벡터가 scan-in 되는 방향을 기준으로, 앞부분에서 발생된 천이는 뒤에서 발생된 천이에 비하여 scan-in 하는 동안 더 많은 천이를 발생시킨다. 따라서 각 스캔벡터들에 대하여 scan-in 하는 동안 발생되는 총 천이개수가 최소가 되도록 scan-in 방향을 결정하면 scan-in 전력소모가 줄어들게 된다.

스캔체인에 입력되는 방향을 기준으로, 스캔벡터내의 n번째 비트위치에서 발생된 천이는 스캔체인의 길이가 K인 스캔체인에 입력하는 동안 K-n의 천이를 발생시킨다.

총 천이개수를 줄이기 위해서는 n이 커야 하며 이것은 천이의 발생위치가 입력되는 방향에서 가능한 뒤부분에 위치함을 의미한다.

<그림 4>는 스캔벡터가 입력하는 방향에 따라 총 천이개수가 달라짐을 나타낸 것이다. 예를 들어 임의의 스캔 벡터를 S1S2S3S4S5 = 10000 라 가정하자.

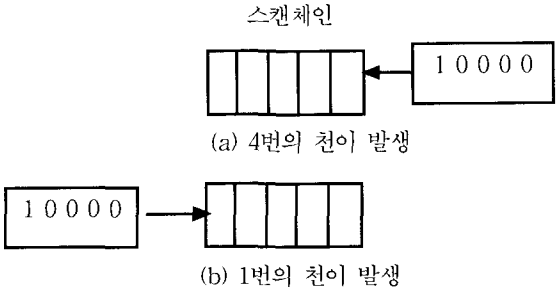

그림 4. 스캔 입력하는 방향에 따른 총 천이 개수
Fig. 4. Number of transitions according to the direction of scan-in.

스캔벡터의 S1이 먼저 입력되고 그 다음 S2가 입력되는 경우<그림 4(a)>, S1과 S2에서 발생한 천이(1 → 0)는 스캔 체인으로 입력하는 동안 4번의 천이를 발생시킨다. 그러나 S5가 먼저 입력되는 경우<그림 4(b)>에는 오직 1번의 천이만을 발생시킨다. 따라서 이 경우가 이전의 경우보다 전력소모가 작게된다.

이 방법을 적용하기 위해서 필요한 부가회로는 스캔체인 길이만큼의 2×1 멀티플렉서이다. 왜냐하면 양방향 스캔입력이 가능하기 위해서는 스캔 셀에서 각 입력을 방향에 따라 바꿔줘야 하기 때문이다. 멀티플렉서에서 추가로 소모하는 전력소모는 스캔 셀에서 천이로 인해 발생하는 전력소모에 비해 매우 작기 때문에 전체적인 전력소모에서는 줄어들게 된다.

3. 적응적 인코딩/디코딩 방법

이 절에서는 저전력으로 매핑된 스캔벡터에 대하여 높은 압축효율을 얻을 수 있는 적응적 인코딩/디코딩

기법에 대하여 기술한다. 3.2 절에서 기술한바와 같이, 스캔벡터내의 천이가 적도록 don't care 입력에 값을 할당했기 때문에 연속된 0의 개수 및 연속된 1의 개수도 많아진다. 이러한 스캔벡터를 기존의 0's run만을 적용한경우, 연속된 1의 경우는 0의 run-length가 0인 것을 의미하며 이것을 인코딩하면 오히려 데이터 양이 늘어나게 된다. 왜냐하면 run-length가 0이면 FDR 인코딩의 경우 2 비트가 필요하기 때문이다. 즉, 데이터의 양이 2배가 된다. 따라서 기존의 방법에서는 0의 run-length가 0인 경우가 많이 나오면 나올수록 인코딩 효율은 떨어지게 된다.

본 논문에서는 0's run 뿐만 아니라 1's run을 고려한 새로운 인코딩 방법을 적용하였다. <그림 5>는 기존의 FDR 인코딩을 이용한 적응적 인코딩을 나타내었다. 적응적 인코딩은 기존의 인코딩 방법에 run_type이란 부가 비트를 추가하여 이것이 0인 경우에는 0's run 인코딩, 1인 경우에는 1's run 인코딩으로 구분한다. 따라서 최종 코드워드는 run_type , group prefix 그리고 tail을 조합하여 구성된다.

<그림 5>를 이용하여 적응적으로 인코딩한 예를 들어보면 다음과 같다.

스캔 벡터를 00000001111001 라 가정하자.

(a) 0's run만을 이용하여 인코딩한 경우

Group	Run-length	Type of pattern		run_type Prefix		Group Prefix	Tail	Codeword (run_type & group prefix & tail)	
		0's run	1's run	0's run	1's run			0's run	1's run
A1	0	1	0	0	1	0	0	000	100
	1	01	10				1	001	101
A2	2	001	110			10	00	01000	11000
	3	0001	1110				01	01001	11001
	4	00001	11110				10	01010	11010
	5	000001	111110				11	01011	11011
A3	6	0000001	1111110			110	000	01100000	11100000
	7	00000001	11111110				001	01100001	11100001
	...	...	...				...	...	...
...	...	...	...	...	...	...	...	...	...

그림 5. 적응적 인코딩
Fig. 5. Adaptive Encoding.

스캔벡터 00000001111001는 아래의 빗금친 것과 같은 5개의 패턴형태로 분류되어 인코딩 된다. 이때 인코딩된 비트수는 17 비트이다.

00000001 1 1 1 001 → (110001) (00) (00) (00) (1000)

(b) 적응적으로 인코딩한 경우

스캔벡터는 빗금친 것과 같이 3개의 패턴형태로 분류되어 인코딩 된다.

00000001 1110 01 →(0110001) (11001) (001)

(0's run) (1's run) (0's run)

첫 번째 패턴은 0's run 형태로 인코딩하고 2번째 패턴은 1's run, 3번째 패턴은 0's run형태로 인코딩하면 총 인코딩 비트수는 15비트가 되므로 (a)의 경우보다 압축효율이 좋아진다.

Ⅳ. 저전력 및 높은 압축률을 갖는 인코딩/디코딩 알고리즘

본 절에서는 이전 절에서 제안하였던 여러 가지 저전력 및 인코딩 방법을 적용하여 스캔벡터를 효율적으로 인코딩/디코딩 하는 알고리즘에 대해 기술한다.

1. 인코딩 과정

```
Encoding()
{ read(scan vectors);
 Num_sv = number of scan vectors
 Num_right=Num_left=1;
 for i=1 to Num_sv
 { X_map(SVi); /** 저전력 매핑 **/
  calculate left_first_power(SVi); /** Grouping **/
  calculate right_first_power(SVi);
  if(left_first_power >= right_first_power)
    right_first_group[Num_right] = SVi; Num_right ++;
  else
    left_first_group[Num_left] = SVi; Num_left ++;
 }
 Ordering(right_first_group[]);
 Adapative_encoding(right_first_group[]);
 insert Escape_codes;
 Ordering(left_first_group[]);
 Adapative_encoding(left_first_group[]);
} End_encoding;
```

그림 6. 제안된 알고리즘
Fig. 6. Proposed Algorithm.

<그림 6>은 인코딩 알고리즘을 나타내고 인코딩의 각 과정은 다음과 같다.

(a) 저전력 및 높은 압축률을 갖는 don't care 매핑

(b) scan-in 입력 방향에 따라 각 스캔벡터들을 분류(좌방향(left_first), 우방향(right_first))

(c) 각 그룹에 대하여 높은 압축률을 위한 스캔벡터의 정렬(Ordering)

(d) 각 그룹에 대하여 0's run/1's run을 적용한 적응적 인코딩(Adaptive_encoding)

자세한 설명은 다음과 같다.

(a) 저전력 및 높은 압축률을 갖는 don't care 매핑

이 방법에 대한 자세한 설명은 3.1절에 기술되었다.

(b) scan-in 입력 방향에 따라 각 스캔벡터들을 분류

3.2절에 기술한 바와 같이 각 스캔벡터에 대하여 저전력 스캔입력의 방향을 구한다. 스캔벡터의 가장 왼쪽 비트가 먼저 입력되는 경우를 좌방향(Left_first) 그룹, 가장 오른쪽 비트가 먼저 입력되는 경우를 우방향(Right_first)그룹으로 분류하였다. 먼저 좌방향으로 스캔입력하는 동안 발생한 총 천이개수(left_first_power)를 구하고, 우방향으로 입력하는 동안 총 천이개수(right_first_power)를 구하여 천이개수가 작은 것을 선택하여 분류한다. 즉 좌방향의 총 천이개수가 우방향의 경우보다 작으면 좌방향 그룹으로 분류하고 그렇지 않으면 우방향 그룹으로 분류한다.

(c) 높은 압축률을 얻기 위한 스캔벡터들의 정렬(Ordering)

압축률을 높이기 위해서는 run-length가 가능한 길어야 하므로 각 그룹에 대하여 가능한 run-length가 길어지도록 벡터들을 재정렬한다. 따라서 임의의 스캔벡터와 다음 스캔벡터가 연결될 때 연결되는 부분의 run-length가 최대한 길도록 정렬해야 한다. 좌방향 그룹의 예를 들어 설명하면, 임의의 스캔벡터 SVi 에 대하여 다음 스캔벡터는 가장 왼쪽 비트값이 SVi의 가장 오른쪽 비트와 값이 같고 가능한 run-length가 길어야 한다. 마찬가지로 좌방향 그룹의 경우에는 현재 입력하는 스캔벡터의 가장 왼쪽의 비트와 다음 스캔벡터의 가장 오른쪽 비트의 값이 같고 가능한 run 길이가 길어야 한다.

<그림 7>에 좌방향 그룹에 대한 정렬 예를 나타내었다. 4개의 스캔벡터 SV1, SV2, SV3, SV4가 초기상태에 <그림 7(a)>와 같다고 가정하자. SV1을 최초의 시작 벡터라 가정하면 다음 스캔벡터는 SV1의 가장 오른쪽 비트 값(0)과 동일하고 그것의 run-length가 가능한 길어야 한다. <그림 7(b)>에 나타낸 바와같이 SV3이 다음 스캔벡터가 된다. 위의 방법으로 정렬하면 <그림 7(c)>처럼 정렬이 된다.

(a)	(b)	(c)
SV1 : 11110000	SV1 : 11110000	SV1 : 11110000
SV2 : 00111111	SV3 : 00000011	SV3 : 00000011
SV3 : 00000011	SV2 : 00111111	SV4 : 11111000
SV4 : 11111000	SV4 : 11111000	SV2 : 00111111

그림 7. 정렬 예
Fig. 7. Ordering Examples.

(d) 0's run/1's run을 이용한 적응적 인코딩

정렬된 그룹에 대한 적응적 인코딩을 한다.

2. 디코딩 과정

인코딩되어 외부 메모리 혹은 테스터기에 저장된 테스트 벡터는 IP 에 인가되기 전에 디코딩 되어야 한다. 이전 절에서 제안된 인코딩 알고리즘은 기존의 인코딩 기법에 부가하여 적용될 수 있다. 따라서 디코딩 과정도 기존의 방법을 약간 수정하면 된다. 본 논문에서는 제안된 알고리즘을 기존의 FDR 인코딩 방법에 적용하였다. <그림 8>은 기존의 FDR 디코더에 간단한 회로를 부가하여 디코더를 구현하였다.

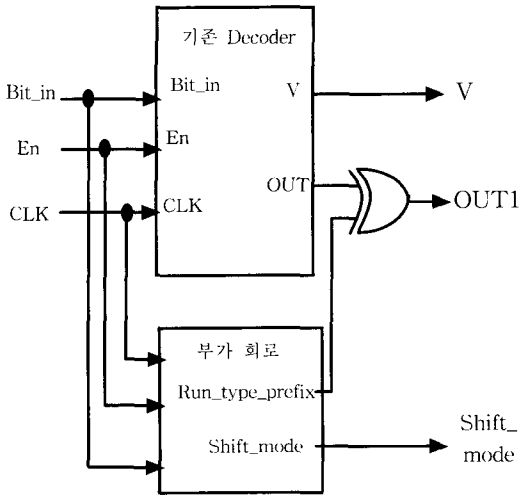

그림 8. 디코더 아키텍처
Fig. 8. Decoder Architecture.

run_type을 디코딩한 출력 run_type_prefix, 이동방향을 디코딩한 shift_mode 등은 새롭게 부가된 출력핀이다. bit_in은 인코딩된 스캔벡터의 입력단자이고 OUT은 디코더에서 디코딩된 스캔벡터, 유효신호단자인 V=1인 경우에만 OUT신호가 유효한 출력이 된다. 인코딩된 스캔벡터가 0's run으로 인코딩된 경우에는 run_type_prefix=0, 1's run으로 인코딩된 경우에는 run_type_prefix=1 이다.

기존 Decoder 블록은 0's run 만으로 인코딩된 스캔벡터를 디코딩하는 블록이다. 따라서 OUT 출력단자로 출력되는 데이터는 0's run으로 디코딩되어 출력된다. 1's run으로 인코딩된 경우에는 출력이 반전되어야 한다. 따라서 run_type_ prefix=1 인 경우에 OUT 출력을 반전시켜야 하므로 EXOR 게이트가 필요하다. 또한 디코딩된 출력은 shift_ mode 값에 따라 우방향 혹은 좌방향으로 스캔 체인에 입력된다.

V. 실험결과

이 절에서는 ISCAS89 벤치마크 회로를 대상으로 제안된 방법을 적용하여 얻은 실험결과에 대해 기술한다. 실험대상회로는 한 개의 스캔체인을 이용하여 full-scan 회로로 구성되었다고 가정한다.

MINTEST ATPG(Automatic Test Pattern Generation) 프로그램을 이용하여 dynamic compaction 방법[4]으로 생성된 스캔벡터들을 이용하여 실험하였다. 이 실험들은 SUN Ultra 10 웍스테이션상에서 수행되었다. <표 1>은 scan-in 전력소모를 측정한 실험결과이다.

본 논문에서는 스캔벡터의 가장 오른쪽 비트가 먼저 스캔체인에 입력된다고 가정하였다. 따라서 [17]에서 측정한 scan-in 전력소모는 그대로 인용이 가능하지만 [9,10]에서 측정한 인코딩 결과는 그대로 인용이 불가능하여 알고리즘을 구현한후에 재 측정하였다. 왜냐하면 [9,10]의 인코딩결과는 가장 왼쪽 비트가 먼저 입력된다고 가정한 결과값이기 때문이다.

<표 1>은 벤치마크 회로에 대하여 scan-in 전력소모의 평균전력 및 최고치 전력소모를 측정한 결과이다. 전력소모의 단위는 천이수이다.

<표 1(a)>는 평균 전력소모의 측정결과이다. 제안된 방법으로 측정한 전력소모는 Pavg_proposed이고 이전의 측정치는 Pavg_0 및 Pavg_prv이다. Pavg_0 는 don't care 비트들을 모두 0으로 할당하여 인코딩한 경우이고 P_{avg_prv}는 [17]에서 제안한 방법으로 인코딩된 경우이다.

표 1. 평균 전력소모의 측정 결과
Table 1. Experimental results on the average /peak power.

대상 회로	P_{avg_0}	P_{avg_prv}	제안된 방법의 평균 전력소모 및 감소율		
			$P_{avg\_proposed}$	감소율 (%) (P_{avg_0} 대비)	감소율 (%) (P_{avg_prv} 대비)
S5378	4300	3433	2407	44.1	29.9
S9234	6705	3957	3434	48.7	13.2
S13207	12317	7734	7284	40.9	5.8
S15850	19448	13513	11749	39.6	13
S38417	201940	117541	109148	45.9	7.14
S38584	134037	85655	84163	37.2	1.74
평균				42.7	11.8

(a) 평균 전력소모

대상회로	P_{peak_0}	P_{peak_prv}	제안된 방법의 최고치 전력소모 및 감소율		
			$P_{peak\_proposed}$	감소율 (%) (P_{peak_0} 대비)	감소율 (%) (P_{peak_prv} 대비)
S5378	12085	11519	11523	4.65	-
S9234	15395	14092	14116	8.30	-
S13207	110129	94879	94887	13.8	-
S15850	84360	70875	70919	15.9	-
S38417	501914	437884	437997	12.7	-
S38584	525417	481148	481190	8.4	-
평균				10.6	-

(b) 최고치 전력소모

S5378에 대하여 제안한 방법으로 인코딩하면 평균전력소모가 2407이며 P_{avg_0}에 대해서는 약 44%, P_{avg_prv}에 대해서는 약 30%가 적은 수치이다. 또한 S15850의 경우에는 제안된 방법의 전력소모가 11749이며 P_{avg_0}에 대해서는 약 40%, P_{avg_prv}에 대해서는 약 13%가 적은 양의 전력소모이다. 또한 제안한 방법을 적용한 scan-in 전력소모는 평균적으로 P_{avg_0}에 대해서는 약 43%, P_{avg_prv}에 대해서는 약 12%가 감소됨을 알수 있다.

<표 1(b)>는 최고치 전력소모의 측정결과이다.

제안된 방법의 결과($P_{peak\_proposed}$)는 기존의 방법[17]을 적용한 결과(P_{peak_prv})와 거의 차이가 없지만 P_{peak_0}와 비교하면 평균 10% 정도 감소됨을 알 수 있다. S15850의 경우에는 P_{peak_0}와 비교하여 약 16% 정도 감소하며 모든 벤치마크회로에 대하여 평균 약 11% 감소함을 보여준다.

<표 2>는 스캔벡터를 인코딩인 결과를 나타내었다.

표 2. 저전력 매핑된 스캔백터의 인코딩 실험 결과

Table 2. Results of encoding the scan vectors mapped for low power.

대상회로	저전력 매핑 (T_{map})	Golomb 인코딩		FDR 인코딩		제안된 인코딩	
		T_{en_golomb} (Bits)	압축율 (%)	T_{en_FDR} (Bits)	압축율 (%)	$T_{en\_proposed}$ (Bits)	압축율 (%)
S5378	23754	37240	-56.8	26668	-12.3	11922	49.8
S9234	39273	64297	-63.7	47398	-20.7	21784	44.5
S13207	165200	236075	-42.9	155074	6.12	31807	80.7
S15850	76986	130260	-69.2	90822	-17.9	26376	65.7
S38417	164736	284309	-72.6	198374	-20.4	67150	59.2
S38584	199104	308483	-54.9	216820	-8.9	78295	60.7
평균			-60		-12.3		60.1

Tmap은 don't care 비트들을 제안된 저전력 할당법으로 할당한 스캔벡터의 비트수이고 이것을 기존의 방법과 제안된 방법으로 인코딩하였다.

제안된 방법으로 인코딩한 결과(Ten_proposed)를 기존의 Golomb Codes로 인코딩한 결과(Ten_golomb) 및 FDR Codes로 인코딩한 결과(Ten_FDR)와 비교하였다. 여기서 압축률은 $\text{압축률} = \frac{(T_{map} - T_{en*})}{T_{map}} \times 100$로 나타낼수 있다. Tmap에서 인코딩된 비트수(Ten_*)을 감산하여 백분율로 나타낸 것이다. 기존의 방법으로 인코딩을 하면 압축률이 매우 떨어져서 오히려 비트수가 증가하게 된다. S5378의 경우 저전력으로 매핑을 하면 23575 비트가 되고 이것을 Golomb Codes로 인코딩을 하면 37240 비트로 되어 약 60%정도 비트수가 증가하게 된다. FDR Codes로 인코딩을 하면 12.3% 증가한다. 하지만 제안된 방법을 이용하여 인코딩하면 50%정도 비트수가 감소한다.

S13207의 경우에는 약 81%정도의 압축률을 보이고 모든 벤치마크 회로에 대한 평균 압축률은 60%에 달한다. 기존의 방법에 비하여 매우 높은 압축률을 갖게 된다.

위의 실험결과에 나타낸 바와같이 기존의 방법에서는 scan-in 전력소모를 감소시킬수는 있지만 인코딩효율은 매우 좋지 않았다. 또한 인코딩 효율을 높이면 scan-in 전력소모가 커지는 문제점이 있었지만 제안된 방법을 적용하면 scan-in 전력소모를 감소시킬수 있고 동시에 인코딩 효율이 좋아짐을 알 수 있다.

Ⅵ. 결 론

SOC 스캔 테스트에 있어서 저전력 및 높은 압축률을 갖는 새로운 인코딩/디코딩 방법을 제안하였다.

저전력 및 높은 압축률을 동시에 얻기위하여 효율적인 don't care 매핑을 하였고 높은 압축율을 얻기위하여 적응적 인코딩을 적용하였다. 또한 스캔벡터의 scan-in 방향을 저전력이 되도록 결정하였다.

ISCAS89 벤치마크 회로에 대하여 실험을 실시한 결과, 제안된 알고리즘은 저전력 및 높은 압축율을 나타내었다. 평균 전력소모는 don't care를 0으로 매핑한 경우보다 약 42% 감소하였고 기존의 저전력 매핑방법보다는 약 12% 감소하였다. 또한 압축율도 약 60%가 증가되었다.

참 고 문 헌

[1] V.Iyengar, K.Charabarty, and B.T.Murray, "Built-in self testing of sequential circuits using precomputed test sets," in Proc. IEEE VLSI Test Symp., pp. 418~423, May 1998.

[2] V.Iyengar, K.Charabarty, and B.T.Murray, "Deterministic built-in pattern generation for sequential circuits," J.Electron. Tet. Theory Applicat., Vol. 15, pp. 97~115, Aug./Oct. 1999.

[3] A.Jas, J. Ghosh-Dastidar, and N.A. Touba, "Scan vector compression/decompression using statistical coding," in Proc. IEEE VLSI Test Symp., pp. 114~120, May 1999.

[4] I.Hamzaoglu and J.H. Patel, "Test set compaction algorithms for combinational circuits," in Proc. Int. Conf. Computer-Aided

Design, pp. 283~289, Nov. 1998.

[5] S.Kajihara, I. Pomeranz, K. Kinoshita, and S. M. Reddy, "On compacting test sets by addition and removal of vectors," in Proc. IEEE VLSI Test Symp., pp. 202~207,May 1994.

[6] A.Jas, N.A. Touba, "Test vector decompression via cyclical scan chains and its application to tesing core-based design," in Proc. Int. Test Conf., pp. 458~464, Nov. 1998.

[7] S.W. Golomb, "Run-Length encoding," IEEE Transactions on Information Theory, Vol. IT-12, pp. 399~401, 1966.

[8] H.Kobayashi and L.R.Bahl, "Image data compression by predictive coding, Part I: Prediction Algorithm," IBM Journal of Research & Development, Vol. 18, pp. 164, 1974.

[9] A.Chandra and K. Chakrabarty, "System-on-a-chip test data compression and decompression architectures based on Golomb codes," IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, Vol. 20, No. 3, pp. 355~368, March 2001.

[10] A. Chandra and K. Chakrabarty, "Frequency-directed run-length (FDR) codes with application to system-on-a-chip test data compression," Proc. IEEE VLSI Test Symposium, pp. 42~47, 2001.

[11] H.J. Wunderlich and Y.Zorian, "Built-In Self Test(BIST): Synthesis of self-testable systems," Kluwer Academic Publishers, 1997.

[12] Y.Zorian, "A distributed BIST control scheme for complex VLSI devices," IEEE VLSI Test Symp., pp. 4~9, 1993.

[13] W.H. Debany, "Quiescent scan design for testing digital logic circuits," Dual-Use tech. & App., pp. 142~151, 1994.

[14] R.M. Chou, K.K. Saluja and V.D. Agrawal, "Scheduling tests for VLSI systems under power constraints," IEEE Transactions on VLSI, pp. 175~184, 1997.

[15] S. Wang and S.K. Gupta, "ATPG for heat dissipation minimization during test application," IEEE Transactions on Computers, pp. 256~262, 1998.

[16] R. Sankaralingam, R.R. Oruganti and N.A. Touba, "Static compaction techniques to control scan vector power dissipation," Proc. VTS, pp. 35~40, 2000.

[17] A. Chandra and K. Chakrabarty, "Combining low-power scan testing and test data compression for system-on-a-chip," Proc. IEEE/ACM Design Automation Conference (DAC), pp. 166~169, June 2001.

[18] M.Sugihara, H.Date and H.Yasuura, A novel test methodology for core-based system LSIs and a testing time minimization problem, Proc. ITC, pp. 465~472, 1998.

저 자 소 개

鄭 俊 模(正會員)

1965년 2월 20일생. 1987년 2월 : 한양대학교 전자공학과 졸업(학사). 1989년 2월 : 한양대학교 대학원 전자공학과 졸업(석사). 1996년~현재 : 한양대학교 전자전기공학부 박사과정. 1989년 3월~1996년 7월 : 삼성전자 종합연구소 ASIC 센타 근무. 1996년 8월~현재 : 김포대학 전자정보계열 조교수. <주관심분야 : ASIC Design, CAD for VLSI, Test and Testable Design>

鄭 正 和(正會員) 第39卷 SD編 第5號 參照