논문 2005-42SD-2-8

SRAM 이중-포트를 위한 내장된 메모리 BIST IP 자동생성 시스템 개발

(The Development on Embedded Memory BIST IP Automatic Generation System for the Dual-Port of SRAM)

심 은 성*, 이 정 민*, 이 찬 영*, 장 훈**

(Eun-Sung Shim, Jung-Min Lee, Chan-Young Lee, and Hoon Chang)

요 약

본 논문에서는 내장된 메모리의 테스트를 편리하게 하기 위하여 간단한 사용자 설정에 의해 자동으로 BIST IP를 생성해 내는 범용 CAD 툴을 개발 하였다. 기존의 툴들은 널리 사용되고 있는 알고리즘에 국한되어 있어 메모리의 모델이 변하게 되면 다시 메모리 모델에 따라 BIST IP를 설계해야 하는 번거로움이 있었다. 하지만 본 논문에서는 사용자가 원하는 메모리 모델에 따라 알고리즘을 적용해 자동으로 BIST IP를 생성해 주는 툴을 개발하였다. 내장된 메모리로는 리프레쉬가 필요 없는 다중-포트 비동기식 SRAM이 가장 많이 사용되며, 본 연구에서는 이중-포트 SRAM에 대하여 연구 하였다.

Abstract

In this paper, we develop the common CAD tool that creates the automatically BIST IP by user settings for the convenient test of embedded memory. Previous tools have defect that when memory model is changed, BIST IP must re-designed depending on memory model because existing tools is limited the widely used algorithms. We develop the tool that is created automatic BIST IP. It applies the algorithm according to the memory model which user requests We usually use the multi-port asynchronous SRAM needless to refresh as the embedded memory. Howerve, This work researches on the dual-port SRAM.

Keywords : BIST, Dual-Port Memory, SRAM, Embedded Memory

I. 서 론

반도체 공정기술이 발달하고 VLSI가 고성능화됨에 따라 점점 더 많은 수의 코어들이 SoC(System on Chip)화 되고 있다. 그 중 내장 메모리는 전체 SoC 트랜지스터 수의 80%~90%를 차지하고 있으며 SoC 개발 시 내장 메모리의 테스트가 새로운 문제로 대두되고 있다. 현재 내장 메모리를 테스트하기 위하여 가장 널리 사용되는 방법은 메모리 BIST(Built-In Self Test) 기법이다[1-2]. 메모리 BIST 기법은 칩의 내부에 테스트 회로를 내장하여 자체적으로 테스트를 수행하는 기법으로써 부수적인 면적의 증가와 같은 오버헤드를 갖게 되지만, 각 모듈별로 자체적인 테스트가 수행되므로 전체

* 학생회원, ** 정회원 숭실대학교 컴퓨터학과
(Department of Computing, Soongsil University)
※ 본 연구는 정보통신부의 출연금으로 수행한 IT-SOC 핵심설계 인력양성 사업의 결과물입니다.
접수일자: 2004년7월29일, 수정완료일: 2005년1월24일

시스템의 테스트에 있어서 테스트의 복잡도가 크게 줄어들고, 고가의 외부 테스트 장비를 사용하지 않고도 빠른 시간에 테스트를 수행할 수 있다는 장점 때문에 내장된 메모리의 테스트를 위한 BIST 기법의 사용이 보편화 되어 있다. 더욱이 내장된 메모리의 크기가 점차 커져감에 따라 내장된 자체 테스트 회로의 단점인 면적 오버헤드가 상대적으로 크게 감소하게 되므로 그 장점이 더욱 부각되고 있다.

IEEE 1149.1은 VLSI급의 칩들이 다수 포함되어 있는 기판 수준의 테스트를 효율적으로 하기 위하여 제안되었으며[3-4], 현재 대부분의 반도체 회사들이 이를 칩의 구현에 실제 적용하고 있다[5-7]. IEEE 1149.1을 적용할 경우 기판상의 칩 상호간 연결 상태를 쉽게 테스트할 수 있으며, 기판상의 칩들에 적용된 테스트 관련 회로의 표준화된 인터페이스를 제공하고, 테스트 관련 입출력 단자 수를 줄일 수 있다. 또한 칩의 디버깅을 시스템 수준에서 할 수 있다는 장점을 가지고 있다.

메모리 BIST 역시 IEEE 1149.1을 이용하여 제어된다[8-9]. IEEE 1149.1을 통하여 메모리 BIST를 동작시키고, 메모리 테스트가 종료된 후 테스트 결과를 칩 외부로 출력한다. 이러한 BIST는 부수적인 면적의 증가 등과 같은 오버헤드를 갖게 되지만, 각 모듈별로 자체적인 테스트가 수행되므로 전체 시스템의 테스트에 있어서 테스트의 복잡도가 크게 줄어들고, 고가의 외부 테스트 장비를 사용하지 않고도 빠른 시간에 테스트를 수행할 수 있다는 장점 때문에 최근 그 사용이 확산되고 있는 추세다.

내장된 메모리로는 리프레쉬가 필요 없는 다중-포트 비동기식(multi-port asynchronous) SRAM이 가장 많이 사용되고 있다[10], 대부분의 다중-포트 메모리 테스트를 위한 알고리즘들은 다중-포트 메모리를 여러 개의 단일-포트 메모리로 간주하고 각각의 포트에 대해 단일-포트 메모리 테스트 알고리즘을 적용하는 방식을 사용하고 있다[11].

이와 같이 여러 종류의 메모리가 생산되고 거기에 따라 새로운 알고리즘이 도입되면서 메모리 코어별로 메모리 BIST 모듈을 설계하는 것은 매우 어렵게 되었다. 따라서 이 과정을 자동화하기위한 툴이 필수적이다. 현재 상용화 툴은 소수의 보편적인 메모리 테스트 알고리즘 중에서만 선택을 해야 하고 사용자가 원하는 새로운 알고리즘의 구현이 불가능하다. 본 논문에서는 이중-포트 메모리 테스트를 위한 메모리 BIST IP의 개발 및 구현과 메모리 BIST IP를 자동 생성해 주는 범용 GenMBC(Generator Memory BIST Circuit)를 제안하고자 한다.

Ⅱ. 이중-포트 메모리의 고장 모델

1. 이중-포트 고장 모델

이중-포트 고장(2PFs)은 하나의 셀(2PF1s) 혹은 두 개의 셀(2PF2s)에 동시에 접근할 때 생기는 고장으로 약고장(weak fault)이 발생하여 하나의 강고장(strong fault)으로 나타나는 경우이다. 그림 1에서 와같이, 하나의 셀 고장은 두 개의 포트가 하나의 셀에 동시에 접근할 때 생기는 고장을 말하며, 두 개의 셀 고장은 세 가지로 나누는데, 첫 째는 하나의 셀이 특정 값을 가지고 있으면서 다른 셀에 두 포트가 동시에 접근할 때 생기는 고장(2PF2a), 둘째는 두 개의 포트를 통하여 동시에 하나의 셀에 접근할 때 다른 셀에 고장이 발생하는 경우의 고장 (2PF2v), 마지막으로 각각의 포트를 통해 하나의 셀에는 쓰기 동작을 하고, 다른 하나의 포트로는 다른 셀에 읽기 동작을 했을 때, 읽기 동작의 셀이 바뀌게 되는 고장(2PF2av)을 말한다.

두 개의 포트가 하나의 셀에 접근할 때 발생하는 고장 종류에는 wDRDF&wDRDF와 wRdf&wRDF가 있다. wDRDF&wDRDF는 두개의 포트로 하나의 셀을 동시에 읽을 때 정상적인 값을 출력하지만, 그 셀은 값이 바뀌는 고장을 말하며 <r0(r1) : r0(r1) / ↑(↓) / 0(1)>게 표기한다. r0(1)은 0(1)을 읽는 동작을 말하고, ↑(↓)는 0에서 1로 (1에서 0으로) 셀의 값이 바뀌는 동작을 표시하며 ":"는 포트가 동시에 동작하는 것을 나타낸다. wRDF&wRDF은 두 개의 포트로 하나의 셀을 동시에 읽을 때 셀의 값이 반전되어 출력하는 고장을 말하며 <r0(r1) : r0(r1) / ↑(↓) / 1(0)>게 표기한다.

두 개의 포트가 두 개의 셀에 대해 동시에 접근하여

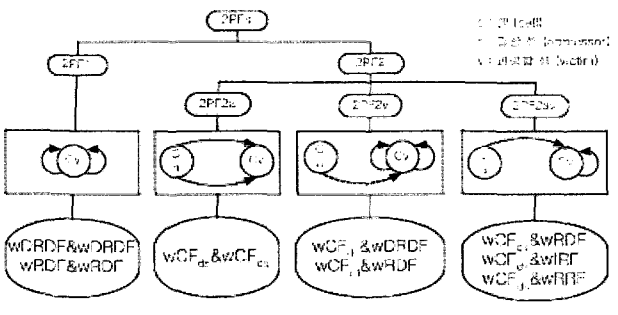

그림 1. 이중-포트 분류
Fig. 1. Dual-port classification.

발생하는 고장의 타입에는 2PF2a, 2PF2v, 2PF2av가 있다. 2PF2a고장 종류에 속하는 고장 모델은 wCF_{ds}&wCF_{ds}가 있다. wCF_{ds}&wCF_{ds} 고장 특징은 피결합셀에 특정 값을 가지고 있고, 두 포트를 통해 다른 셀들을 접근할 때 피결합셀에 나타나는 고장이며, < w0 : rx ; 0(1) / ↑(↓) / - >, < w1 : rx ; 0(1) / ↑(↓) / - >로 표기하고 x는 0 또는 1을 나타내며, "-"는 출력 값을 적용할 수 없는 경우를 말한다. w0(1)는 메모리 셀에 0(1)값을 쓰는 동작을 표시한다.

2PF2v 고장 타입에 속하는 고장 종류는 wCF_{rd}&wRDF 이며, wCF_{rd}&wRDF는 두 개의 포트로 두 개의 셀을 동시에 읽었을 때 다른 셀의 값이 바뀌는 고장을 말한다. < 0 ; r0(r1) : r0(r1) / ↑(↓) / 1(0) >, < 1 ; r0(r1) : r0(r1) / ↑(↓) / 1(0) > 표시할 수 있다.

2PF2av 고장 타입에 속하는 고장 종류는 세 가지가 있으며 특징은 하나의 포트로 하나의 셀에 대해 읽기 동작을 하고, 다른 하나의 포트로 다른 셀에 쓰기 동작을 했을 때 읽기 동작을 한 셀에 나타나는 상태나 출력 값에 오류를 발생하는 고장이다. wCF_{ds}&wRDF는 < w0 : r0(r1) / ↑(↓) / 1(0) >, < w1 : r0(r1) / ↑(↓) / 1(0) >이고, wCF_{ds}&wIRF는 < w0 : r0(r1) / 0(1) / 1(0) >, < w1 : r0(r1) / 0(1) / 1(0) > 이며, wCF_{ds}&wRRF는 < w0 : r0(r1) / 0(1) / ? >, < w1 : r0(r1) / 0(1) / ? >로 나타내며 "?"는 무작위 논리 값이나 정의 되지 않은 논리 값을 나타낸다[12 13]

2. 이중-포트 고장 검출을 위한 March 테스트

단일-포트를 위한 March 테스트 알고리즘은 MAT+, MAT++, March C- 등이 소개가 되었으며, 이중-포트 메모리를 위한 March 테스트는 기존에 소개된 단일-포트 메모리 고장 검출을 위한 March 테스트가 아닌 이중-포트 고장 검출을 위한 March 테스트를 사용하여 고장을 검출을 해야 한다. 2PF1s, 2PF2a, 2PF2v, 2PF2av를 검출 할 수 있는 March 테스트는 표 1을 통해 확인 할 수 있다.

표 1에 표기된 ⇳ 기호는 메모리 주소 가장 낮은 주소에서 가장 높은 주소 혹은 가장 높은 주소에서 가장 낮은 주소 중 하나의 방향으로 메모리의 동작을 수행하는 것을 말하며, "-"는 어떤 동작이든 관계없음을 나타내고, "n"는 동작하지 않음을 표시한다[12, 14]. 표 1의 2PF1의 알고리즘을 살펴보면, 처음에 메모리를 "0"으로 초기화를 하고, 두 번째 동작으로 'w1:r0'는 < r0 : w↑ / 0 / - >의 고장을 검출 할 수 있다. 계속되는 March 동작으로 < r1 : r1 / ↓ / 1 >, < r1 : r1 / ↓ / 0 > 고장을 모두 검출 할 수 있다[12]. 표 1에서의 $\Updownarrow_{i=0}^{R-1}$는 주소의 순서가 0부터 R-1까지를 나타내며, $r1_{i,i+1}$은 i 번째 열, i+1번째 행에서의 1값을 읽는 동작을 말한다.

표 1. 이중-포트 고장 검출을 위한 March 알고리즘
Table 1. March algorithm for dual-port fault detection.

고장내용	이중-포트 고장 검출 알고리즘
2PF1	$\Updownarrow(w0 : -)$; $\Updownarrow(w1 : r0, r1 : r1, r1 : -)$; $\Updownarrow(w0 : r1, r0 : r0, r0 : -)$;
2PF2a	$\Updownarrow(w0 : n)$; $\Updownarrow(r0 : -, w1 : r0, w0 : r1)$; $\Updownarrow(r0 : -, w1 : n)$; $\Updownarrow(r1 : -, w0 : r1, w1 : r0)$; $\Updownarrow(r1 : -)$;
2PF2v	$\Updownarrow(w0 : -)$; $\Updownarrow(r0 : r0, w1 : -, r1 : r1, w0 : -)$; $\Updownarrow(w1 : -)$; $\Updownarrow(r1 : r1, w0 : -, r0 : r0, w1 : -)$;
2PF2av	$\Updownarrow_{i=0}^{R-1}(w0_{i,i} : n, w0_{i,i+1} : n, w0_{i+1,i} : n)$; $\Updownarrow_{i=0}^{R-1}(w1_{i,i} : r0_{i+1,i}, w0_{i,i} : r0_{i+1,i})$; $\Updownarrow_{i=0}^{R-1}(w1_{i,i} : r0_{i,i+1}, w0_{i,i} : r0_{i,i+1})$; $\Updownarrow_{i=0}^{R-1}(w1_{i,i} : n, w1_{i,i+1} : n, w1_{i+1,i} : n)$; $\Updownarrow_{i=0}^{R-1}(w0_{i,i} : r1_{i+1,i}, w1_{i,i} : r1_{i+1,i})$; $\Updownarrow_{i=0}^{R-1}(w0_{i,i} : r1_{i,i+1}, w1_{i,i} : r1_{i,i+1})$;

III. 이중-포트 고장 검출을 위한 BIST IP 구조

GenMBC는 사용자에게 메모리 구조와 메모리 테스트 알고리즘의 정보를 입력받아 자동으로 BIST IP를 생성해 주는 CAD 툴이다. GenMBC는 테스트하고자 하는 메모리를 종류와 적용시킬 알고리즘을 선택해 입력하면 GenMBC는 사용자로부터 받은 정보를 전략적 적용을 통해 VerilogHDL 코드를 생성해 내며 그림 2을 통해 동작 순서를 볼 수 있다.

GenMBC의 설정파일 부분에서는 크게 세 부류로 구분된다. 첫 번째로 메모리 모델을 구분하는 부분으로서 이중-포트 메모리 모델을 설정하며, 메모리에 입력, 출력 핀의 이름과 핀의 크기를 정의하고 메모리의 읽기, 쓰기 시의 타이밍을 기술해 준다. 두 번째로 GenMBC는 여러개의 같은 사이즈의 메모리 BIST IP가 지원되므로, 테스트 하고자 하는 메모리 모델 개수를 입력한다. 마지막으로 적용할 알고리즘과 배경데이터, 컨트롤

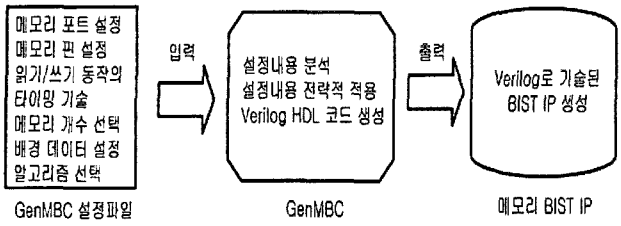

그림 2. GenMBC 구성
Fig. 2. GenMBC configuration.

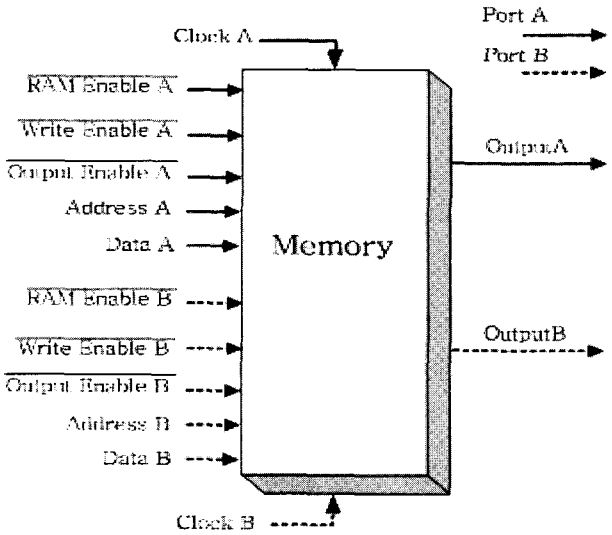

그림 3. 모든 신호 정의에 따른 이중-포트 메모리 구조
Fig. 3. Dual-port memory architecture for full signal.

표 2. 메모리 신호 설명
Table 2. Memory signal description.

신호	설명
RAM Enable A	A 포트를 동작 가능하게 하는 Enable 신호이며, 액티브 로우로 동작
Write Enable A	A 포트 쓰기 가능하게 하는 Enable 신호이며, 액티브 로우로 동작
Output Enable A	A 포트 읽기 가능하게 하는 Enable 신호이며, 액티브 로우로 동작
Address A	A 포트의 Address 입력 신호
Data A	A 포트의 Data 입력 신호
Clock A	A 포트 동작을 위한 클럭 사이클 신호
Output A	A 포트의 출력 신호

신호등이 입력되면 모든 설정내용을 전략적으로 적용시켜 Verilog HDL로 기술된 BIST IP를 생성해 낸다.

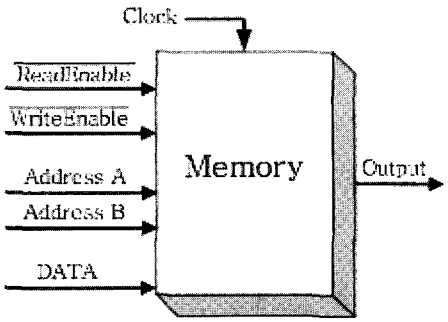

그림 4. 사용자 신호 정의에 따른 이중-포트 메모리 구조
Fig. 4. Dual-port memory architecture for user signal define.

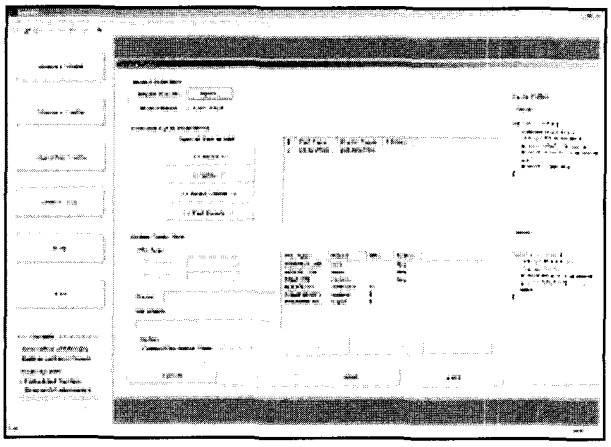

그림 5. 메모리 구현 기술
Fig. 5. Memory implementation description.

GenMBC를 통해 모델링된 이중-포트 메모리의 구조는 포트의 입력, 출력 신호에 특정 규정 없이 원하는 메모리 모델의 특징에 따라 입력, 출력 신호들을 정의할 수 있다. 그림 2는 입력, 출력 신호를 모두 정의 했을 때의 구조도이며 이 구조에서 사용자가 원하는 이중-포트 메모리 입력, 출력 신호를 택해 이중-포트 메모리를 설계할 수 있다.

그림 3의 신호들에 대한 설명은 표 2에서 설명되어 있다. 그림 4는 사용자가 원하는 신호만을 선택해 만든 이중-포트 메모리 구조의 하나의 예제이다.

그림 3에서 B 포트 또한 A 포트와 같은 기능을 하는 신호 들을 가진다. 그림 4는 사용자가 원하는 신호만을 선택해 만든 이중-포트 메모리구조의 하나의 예제이다.

그림 4의 사용자 신호 정의에 따른 이중-포트에서는 제어신호, 입력, 출력신호 뿐만 아니라 그 신호들의 비트너비, 읽기, 쓰기 동작 기술 또한 정의 할 수 있다. 이와 같이 사용자가 테스트하기 원하는 이중-포트 메모리

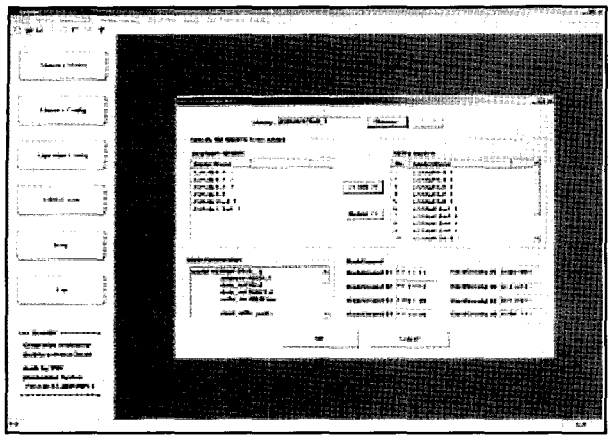

그림 6. 메모리 모델 등록
Fig. 6. Memory model registration.

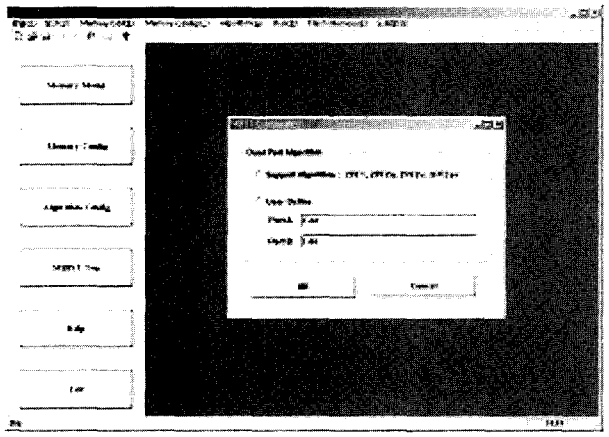

그림 7. 테스트 알고리즘 적용
Fig. 7. Apply an user define to an algorithm.

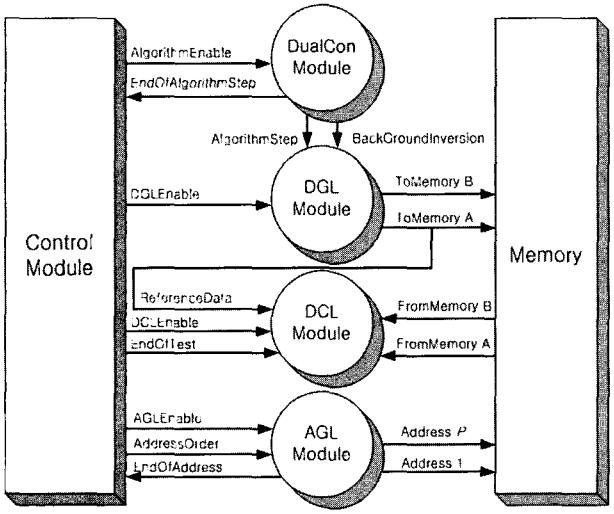

그림 8. 이중-포트 메모리 BIST IP
Fig. 8. Dual-port memory BIST IP.

모델을 정의 할 수 있고 정의한 모델을 용이하게 테스트 할 수 있도록 BIST IP를 제공한다.

본 연구에서 개발한 GenMBC(Generator of Memory Built-in-self-test Circuit)는 Windows OS환경에서 구현하였다. 주로 사용한 소프트웨어 개발 툴은 그래픽 사용자 인터페이스의 작성을 위하여 Microsoft 사의 Visual C++ Compiler를 사용하였다. 구현된 GenMBC, 메모리 BIST Generator는 시스템 온 칩에 내장된 메모리 테스트 RTL 모듈을 생성한다. 생성하기 위한 동작 과정은 다음의 4단계의 과정을 거친다.

첫 번째 단계에서는 메모리 모델을 기술하는 부분으로서 그림 5에서와 같이 볼 수 있다. 이 단계에서는 메모리 모델명, 포트 타입, 핀 타입, 핀의 크기 등의 메모리의 정보를 입력한다.

두 번째 단계에서는 생성한 메모리 모델을 종류와 개수에 상관없이 등록할 수 있다. 메모리 모델의 종류가 같으면 개수의 상관없이 BIST IP 하나가 생성되며, 배경 데이터를 입력할 수 있으며, 그림 6에서 볼 수 있다.

세 번째 단계는 그림 7에서와 같이 테스트하기 위한 알고리즘을 적용하는 부분으로 [12]에서 제안한 March 알고리즘을 기본적으로 제공하고 있으며 사용자 정의 또한 적용 시킬 수 있다. 위와 같은 단계를 통해 이중-포트 메모리를 테스트 할 수 있는 BIST IP가 생성된다.

BIST IP 구조는 그림 8이며 모든 신호를 담당하는 제어 모듈(CONTROL Module), 이중-포트에서 발생하는 고장을 검출하기 위해 적절한 알고리즘 처리를 위한 이중-포트 제어 모듈 (DualCon Module), 알고리즘에 따라 적절한 데이터 값을 메모리에 전달 해주는 데이터 생성 모듈 (DGL Module), 메모리의 값을 읽어와 테스트 패턴과 비교하는 데이터 비교 모듈 (DCL Module), 테스트 패턴을 메모리의 원하는 주소에 인가시키기 위한 주소 생성 모듈 (AGL Module)로 총 다섯 개의 모듈이 이중-포트 메모리 고장을 검출하기 위한 BIST IP를 구성한다.

각각의 모듈의 기능을 살펴보면, BIST 전체를 담당하는 제어 모듈(Control Module)은 테스트의 시작과 종료 시점을 판단하며, 각각의 모듈에 그림 7과 같이 서브 모듈이 동작하기 알맞은 테스트 동작을 진행되도록 구현하였다. 설정 파일에서 기술한 알고리즘을 수행 상태 머신과 각각의 메모리에 대하여 정의한 배경데이터 상태 머신이 존재한다. 제어 모듈에서 AGLEnable, DCLEnable, DGLEnable, AlgorithmEnable의 신호를 각각 인가해주면 주소 생성 모듈 (AGL Module), 데이터 비교 모듈 (DCL Module), 데이터 생성 모듈 (DGL Module), 알고리즘 생성 모듈 (AlgoGL Module)이 수행

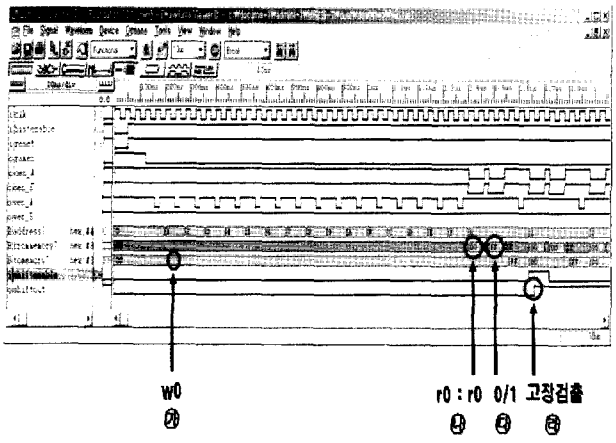

그림 9. 이중-포트 메모리 고장 검출 파형
Fig. 9. Dual-port memory fault detect waveform.

된다.

주소 생성 모듈(AGL Module)은 테스트 모드 시에 제어 회로로부터 주소 생성 신호를 받아 카운터 회로를 생성해 메모리의 주소 0번지부터 마지막 주소까지 증가, 감소가 쉽게 이루어지도록 하였고, 또 테스트 패턴을 정확한 주소에 읽고, 쓰기가 가능하도록 하였다. 메모리 주소가 증가 상태에서 감소 상태로 변할 때나 감소 상태에서 증가 상태로 변할 때 주소의 보수를 취해 원하는 주소를 생성할 수 있도록 하였으며, 여러 개의 메모리가 사용될 경우 현재 테스트가 진행 중인 메모리의 주소의 크기에 맞도록 마지막 메모리 주소를 조정할 수 있게 구현하였다.

데이터 비교 모듈 (DCL Module)은 메모리에서 읽혀진 값과 메모리에 쓰기 한 값을 비교해 고장의 여부를 판단한다. 고장의 유무는 데이터 비교 모듈 내에서 연산 후 쉬프트 연산에 의해 IEEE 1149.1 또는 IEEE P1500을 통해 외부로 최종 테스트 결과 값을 내보낸다.

데이터 생성 모듈 (DGL Module)은 이중-포트 제어 모듈(DualCon Module)에서 알고리즘 단계에 따라 배경 데이터를 생성해 현재 테스트 진행 중인 메모리에 전달하는 모듈이다. 이중-포트 제어 모듈 (DualCon Module)은 사용자에 의해 기술된 다중-포트 테스트 알고리즘을 분석해 각각의 알고리즘 단계에 맞게 데이터 신호를 인가하는 모듈이다.

Ⅳ. 검 증

본 연구에서 개발한 GenMBC를 검증하기 위해 다양한 구성 형태와 알고리즘을 사용하여 생성된 BIST IP를 Xilinx사의 Xilinx Foundation에서 제공하는 simulator를 사용하여 RTL 검증을 하였으며, GTL 검증은 Xilinx사의 Spartan XCS40PQ240 FPGA를 사용하여 합성 후 실제 동작을 검증하였다. [12]에서 제안한 알고리즘 중 2PF2av고장을 제외한 통합 알고리즘을 사용하였으며 그림 9는 wDRDF &wDRFF 고장을 검출하는 파형이다.

wDRDF&wDRFF 고장은 두 포트가 동시에 같은 셀에 읽기 동작을 했을 때, 출력은 정상 값을 내보내지만, 읽은 셀의 값은 변하는 고장이다. 메모리에 16진수 00h 값을 모두 쓰기 한 후(㉮), 메모리 주소 0h 부분에서 두 포트가 읽기 동작을 했을 때 00h값을 출력해 주고(㉯) 메모리의 같은 주소에 다시 읽기 동작을 했을 때 00h값을 가져야 하는데 FFh값을 메모리에서 읽어(㉰) 고장난 셀을 확인할 수 있다. shiftenable 신호가 액티브 하이 값을 다음 메모리 주소에서 인가함으로써 shiftout 신호를 통해 고장이 발생했음을 출력(㉱)해 준다.

Ⅴ. 결 론

최근에 SOC 환경이 급속히 늘어 가면서 상당 부분의 비중을 차지하고 있는 내장 메모리의 수율을 높이기 위해 BIST IP의 자동 생성의 필수 요건으로 자리 잡고 있다. 메모리의 구조와 메모리 테스트 알고리즘이 매우 다양하기 때문에 모든 경우에 맞추어 BIST IP를 개발하기는 어려운 일이다. 따라서 본 논문에서는 다양한 메모리 BIST IP를 자동으로 개발하는 범용 GenMBC (Generator of Memory Built-in self test Circuit)을 개발하였다. GenMBC를 통해서 BIST IP의 설계되는 시간과 다양한 메모리의 재구성 시간을 많이 줄 일 수 있었다.

참 고 문 헌

[1] Benso, A., Di Carlo, S., Di Natale, G., Prinetto, P., Lobetti Bodoni, M., "A programmable BIST architecture for clusters of multiple-port SRAMs," *International Test Conference*, pp. 557-566, 2000.

[2] A. Bommireddy, J. Khare, S. Shaikh, S. Su. "Test and debug of networking SoCs a case

study," Montreal, pp. 121-126, 2000.

[3] IEEE Stndard 1149.1-1990, *"IEEE Standards Test Access Port and boundary-scan Architec -ture,"* IEEE Standards Board, New York, 1990.

[4] Test Technology Standards Committee, "IEEE Standard Test Access Port and Boundary-Scan Architecture," *IEEE Computer Society Press,* 1993.

[5] Parulkar, I., Ziaja, T., Pendurkar, R., D'Souza, A. and Majumdar, A., "A scalable, low cost design-for-test architecture for UltraSPARC/spl trade/ chip multi-processors," *International Test Conference.* pp. 726-735, 2002.

[6] Braden, J., Lin, Q. and Smith, B., "Use of BIST in Sun FireTM servers," In Proc. *International Test Conference.* pp. 1017-1022, 2001.

[7] R. Raina, R. Bailey, D. Belete, V. Khosa, R. Molyneaux, J. Prado, A. Razdan, "DFT Advances in Motorola's Next-Generation 74xx PowerPCTM Microprocessor," *International Test Conference,* pp. 131-140, 2000.

[8] Daehan Youn, Ohyoung Song, Hoon Chang, "Design-for-testability of the FLOVA," *In Proceedings of the Second IEEE Asia Pacific Conference,* pp. 28-30, 2000.

[9] Appello, D., Fudoli, A., Tancorre, V., Corno, F., Rebaudengo, M., Sonza Reorda, M., "A BIST-based solution for the diagnosis of embedded memories adopting image processing techniques," *In Proceedings On-Line Testing Workshop,* pp. 206-210, 2002.

[10] *Memory BistCoreTM User's Reference Manual,* GeneSys TestWare, Revision 14, June, 1998.

[11] Yuejian Wu and Sanjay Gupta, "Built-In Self-Test for Multi-Port RAMs", I*International Test Conference,* 1997.

[12] Hamdioui, S., Rodgers, M., A. J. van de Goor, Eastwick, D., "March tests for realistic faults in two-port memories," *Memory Technology, Design and Testing,* pp. 73-78, 2000.

[13] Hamdioui, S., Rodgers, M., A. J. van de Goor, Eastwick, D., "Realistic Fault Models and Test Procedure for Multi-Port SRAMs," *Memory Technology, Design and Testing,* pp. 65-72, 2001.

[14] Hamdioui, S., A. J. van de Goor., "Efficient Tests for Realistic Faults in Dual-Port SRAMs," *IEEE Transactions on, Computers,* pp. 460-473, 2002.

저 자 소 개

심 은 성(학생회원)

2003년 한서대학교 컴퓨터정보학과 학사 졸업

2003년 3월 ~현재 숭실대학교 일반대학원 컴퓨터학과 석사 과정

<주관심분야 : VLSI 설계 및 테스트, 컴퓨터구조, VLSI CAD>

이 찬 영(학생회원)

2004년 숭실대학교 컴퓨터학부 학사 졸업

2004년 3월 ~현재 숭실대학교 일반대학원 컴퓨터학과 석사 과정

<주관심분야 : VLSI 설계 및 테스트, 컴퓨터구조, VLSI CAD>

이 정 민(학생회원)

2004년 숭실대학교 컴퓨터학부 학사 졸업

2004년 3월 ~현재 숭실대학교 일반대학원 컴퓨터학과 석사 과정

<주관심분야 : VLSI 설계 및 테스트, 컴퓨터구조, VLSI CAD>

장 훈(정회원)

1987년 서울대학교 공대 전자공학과 학사 졸업.

1989년 서울대학교 공대 전자공학과 석사 졸업.

1993년 University of Texas at Austin 졸업.

1991년 IBM Inc. Senior Member of Technical Staff.

1993년 Motorola Inc. Senior Member of Technical Staff.

1994년~현재 숭실대학교 컴퓨터학부 부교수.

<주관심분야 : 통신, 컴퓨터, 신호처리, 반도체>