한국반도체장비회지 제2권 제4호(2003년 12월)
Journal of the Korean Society of Semiconductor Equipment Technology, Vol. 2, No. 4. December 2003.

논리결함 검사를 위한 Pattern Generator의 PLD 회로 설계

김준식† · 노영동*
†호서대학교 전기정보통신공학부, *호서대학교대학원 전자공학과

The PLD Circuit Design of Pattern Generator for the Logical Inspection of Logical Defection

Joon Seek Kim† and Young Dong Roh*
†Dept. of Electronic Information Telecommunication Engineering, Hoseo University
*Dept. of Electronic Engineering, Hoseo University

ABSTRACT

In this paper, we design the pattern generator circuits using PLDs(Programmable Logic Devices). The pattern generator is the circuit which generates the test pattern signal for the inspection of logical defects of semiconductor products. The proposed circuits are designed by the PLD design tool(MAX+ II of ALTERA). Also the designed circuits are simulated for the verification of the designed ones. The simulation results have a good performance.

Key Words : Pattern Generator, PLD, Semiconductor, Memory Test

1. 서 론

우리나라의 반도체 산업은 1983년 메모리 공정 사업에 국내 기업이 본격적으로 참여하여 급속한 발전을 이룩해 왔다. 발전에 따른 반도체 소자의 고집적화로 인해 테스트 소요시간이 증가하게 되었다. 이러한 고집적화 메모리에 대하여 검사를 할 경우 동일한 동작을 여러 가지 셀에 반복적으로 수행한다면 검사소요 시간이 증가함에 비례하여 생산공정에 따른 비용도 증가하게 된다. 이러한 문제점을 효과적으로 처리하기 위해선 일괄적인 패턴을 발생시켜 단시간에 논리적인 오류를 검사하여야 한다. 이에 사용되는 장비를 pattern generator(PG)라 한다[1,2].

본 연구에서는 패턴을 발생시키기 위한 PG회로의 전체적인 개념과 그에 따른 PLD회로설계/ 블록도와 시뮬레이션 결과를 통해 회로의 성능을 검증하였다.

2. PG 회로 설계

2.1. PG의 전체 블록도

PG의 주요작업은 주소와 데이터 생성이다. 일반적으로 주소 생성은 카운터(counter), LFSR (Linear Feedback Shift Register) 또는 마이크로프로세서를 이용하며, 데이터 생성은 주소 또는 유한상태기 (finite state machine)를 이용한다[2].

다음의 Fig. 1 은 pattern generator의 전체 블록도 이다. 컴퓨터에서의 데이터 입력을 받아들여 PC data bus를 통하여 PG의 main memory와 program counter로 데이터가 입력된다. Program counter로 들어간 데이터는 PG main memory의 어드레스를 발생시키고 PG의 main memory로 들어간 데이터는 program counter에서 발생된 어드레스 메모리 라인에 의해 할당된 어드레스로 데이터가 들어가 address part, data part, index part 그리고 control part로 입력될 memory file 신호를 발생하게 된다. 각각의 발생된 memory file은 address part에서는 테스트에 사용될 주소를 생성하여 순차적으로 제공하게 되고, data part에서는 address part에서 발생된 주소에 들어갈 데이터를 생성한다.

Index part에서는 memory의 출력을 제어하고, control part에서는 PG를 제어할 명령과 메모리 제어에 관련된 신호를 생성한다. 또 main memory에서 출력된 신호는 MOD와 MUTC 신호 발생 부분으로 들어가

†E-mail : joonskim@office.hoseo.ac.kr

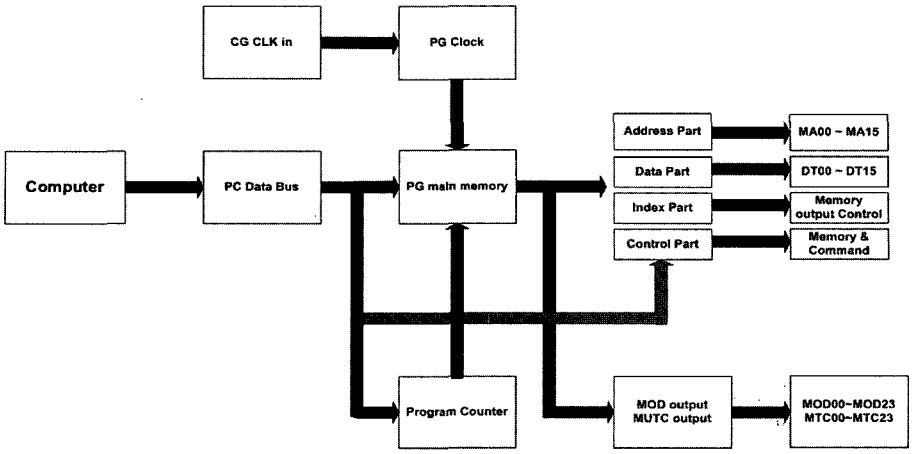

Fig. 1. Block diagram of pattern generator.

MOD 24개 신호와 MTC 24개 신호를 생성한다.

위의 언급한 메모리에서 발생되는 MF 신호와 data 신호가 PG의 다른 구성부분에서 어떠한 신호로서 입력이 되는지는 각각의 부분마다 설명을 할 것이다.

2.2. PG의 각 부분별 기능

위에서 전체 블록을 사용하여 전반적인 PG의 작동을 알아보았다. 그러면 다음 부분부터는 PG를 구성하고 있는 program counter, address part, index part, control part, data part에 대하여 각 블록과 설계된 회로를 제시하여 각 블록의 기능을 설명한다. 이 모든 블록은 PG main memory의 출력 신호를 사용한다. 그럼 먼저 PG에서 가장 중요한 부분인 program counter를 살펴보겠다.

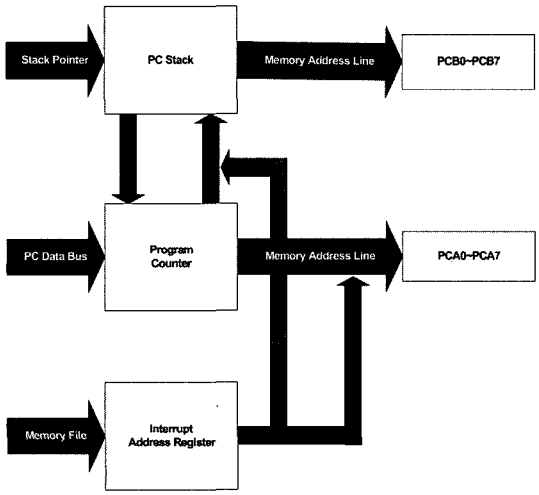

Fig. 2. Block diagram of program counter.

2.2.1. Program counter

Fig. 2의 블록은 PG에서 가장 중요한 역할을 하는 program counter에 대한 블록도 이다. 앞서 설명한 PG의 역할에서 PG는 주소와 데이터를 생성한다고 했는데, program counter는 PG의 메인 메모리에 주소를 할당하고 그 할당된 주소로 PC로부터 data가 입력되어 메인 메모리에서 발생되는 신호인 MF 신호와 data신호를 생성한다.

메인 메모리에서 발생된 memory file은 interrupt address register 회로에 들어가 메인 메모리의 메모리 어드레스 라인에 대하여 인터럽트 기능을 하는데, 이는 program counter에서 발생되는 메모리 어드레스 라인과 메인 메모리와의 충돌을 방지하기 위함이다.

Fig. 3은 Fig. 2에서 address line을 발생하는program counter 부분의 회로이다. Data 신호가 입력되면 counter와 adder를 거쳐서 memory address line 신호가 발생된다. Counter에서 출력된 신호는 MUX에서 data 신호를 쓸 것인지 counter에서 출력된 신호를 쓸 것인지

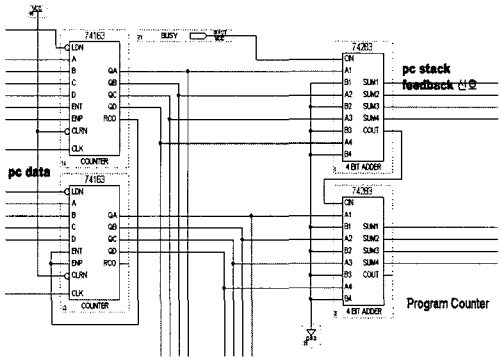

Fig. 3. Program counter circuit.

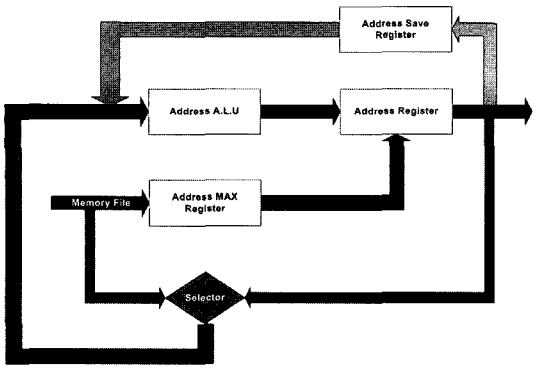

Fig. 4. Block diagram of address part.

를 결정하게 된다. PC stack은 출력된 memory address line을 feedback 시켜 다른 address 신호를 발생한다. 이런 과정으로 출력되는 memory address line 신호는 main memory에서 memory address line을 출력할 것인지 아니면 이전 신호를 유지할 것인지를 interrupt address register에 MF 신호를 보내어 결정하게 된다.

2.2.2 Address part

Fig. 4는 address part의 구성도이다. 이 부분에서는 검사할 소자에 데이터가 입력될 어드레스를 발생하게 된다. Address part에서는 메모리에서 생성된 memory file이 입력으로 들어가 전반적인 신호를 구성하게 되는데, 입력인 MF 신호는 selector를 통하여 새로운 MF 신호를 사용할 것인지 아니면 이전 신호의 구성을 사용할지를 결정하게 된다. 선택 되어진 신호는 address A.L.U에서 정의되는 연산을 하고 나오는 출력값과 address MAX register에서 나오는 출력값, 즉 MF신호들과 AND 연산을 하여 그 조합된 신호의 출력이 address register를 통과하여 address save register와 selector로 신호가 들어가게 된다. Selector로 들어간 신호는 MF신호를 쓸 것인지 아니면 address register에서 들어온 신호를 사용할 것인지를 결정하게 되고, 그 결정된 신호는 다시 address A.L.U의 입력으로 들어가게 된다. 그리고 address save register를 통과하고, 다시 address A.L.U로 들어가 selector에서 들어간 입력과 조합되어 정의하는 연산을 하게 되어 위의 일련의 과정을 수행하게 되며 최종적으로 나오는 16개의 신호는 buffer를 거쳐 감사할 디바이스에 도달하게 된다.

Fig. 5는 Fig. 4에서 어드레스를 발생시키기 위해서 산술 연산을 하는 Address A.L.U의 회로이다. 초기의 입력 MF 신호에 의해 MUX에서 선택 되어진 신호가 A.L.U에 입력되어지고, 출력된 신호와 MF 신호와 AND 연산이 이루어진다. Buffer를 통과하여 MUX에

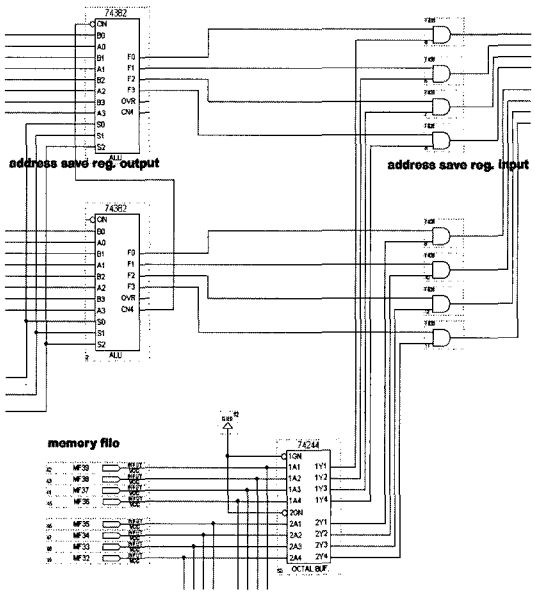

Fig. 5. Address A.L.U circuit.

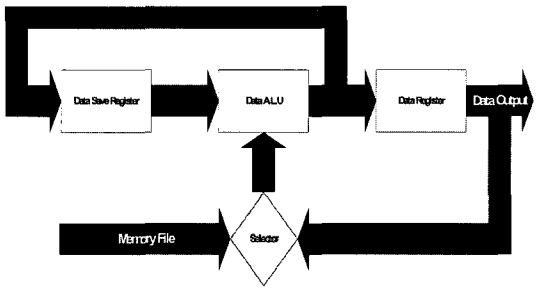

Fig. 6. Block diagram of data part.

서 이전신호를 A.L.U에 입력할 것인지 아니면 MF 신호를 입력할지를 결정하게 되고, 다시 위와 같이 일련의 과정을 수행하게 된다.

2.2.3. Data part

Fig. 6은 PG의 중요한 기능 중 하나인 어드레스에 들어갈 data를 출력하는 부분이다. 우선 memory file이 입력으로 들어간다. 들어간 신호는 data A.L.U에서 정의된 논리식에 의해 출력을 발생시키고 출력된 신호는 data save register와 data register로 입력 된다. 이전 신호를 유지해야 할 경우 data save register에 들어간 신호를 사용하던지 아니면 data register에서 출력된 신호를 사용한다. 이 신호는 selector에서 새로운 MF신호를 사용할 것인지 아니면 feedback 된 신호를 사용할 것인지를 결정하게 된다. 이와 같은 일련의 과정을 통해 출력 된 16 bit data 값은 buffer를 통해 신호를 보강하여 검사할 디바이스에 가해진다.

Fig. 7은 Fig. 6에서 data를 출력시키기 위한 연산을 하는 Data A.L.U의 회로이다. MF신호가 입력으로 주어지고, data A.L.U인 TP소자에서 사용자가 정의해 놓은 논리 연산을 통해 data출력이 이루어진다.

2.2.4. Index part

Fig. 8은 index part의 구성도 이다. Index part는 memory address line의 출력과 main memory output enable를 제어하여 PG에서의 입출력의 충돌을 제어하게 된다.

Index register는 MF신호를 입력으로 받아들여 index register jump 신호를 (이하 IRJP) 발생시킨다. 발생시키는 과정은 index register main memory 신호와 jump index register 신호에 의해 full adder에서의 carry가 발생 되면 IRJP 신호가 발생된다. 또 carry의 발생으로 인해서 index register main memory가 feedback 되어 들어온 신호를 유지하거나 아니면 새로운 데이터를 메모리에 입력 받게 된다. 그럼 IRJP 신호가 과연 어떤 역할을 하는지를 알아보겠다. 블록에는 출력되는 신호에 의해 memory OE 제어라고 나와 있다. 이는 IRJP 신호를 입력으로 받아들이는 PG의 구성요소에서 그 입력으로 발생되는 신호가 PG의 main memory와 SetPC에 들어갈 신호를 발생하게 된다. 우선 PG의 main memory 블록에 들어가는 신호는 memory output enable로 입력되게 되고, SetPC에 들어가는 신호는 program counter 블록으로 들어가 memory address line의 출력을 제어하게 된다. 이것은 출력되는 address와 data를 제어하기 위해 사용된다.

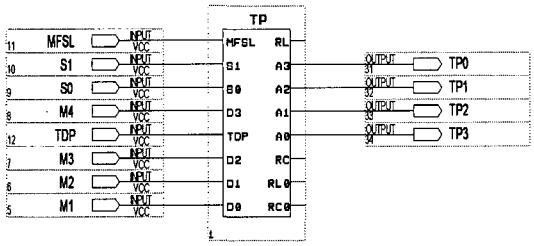

Fig. 7. Data A.LU circuit.

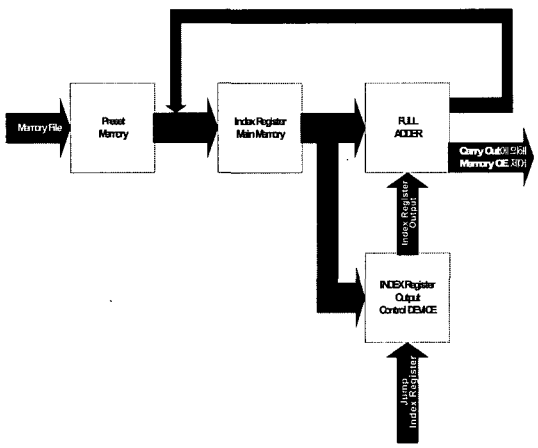

Fig. 8. Block diagram of index part.

Fig. 9는 Fig. 8의 index part 블록의 adder 회로이다. 이 회로는 adder에서 carry가 발생되어 IRJP신호가 발생되는 부분으로 MF 신호가 메모리에 입력되면 메모리에서 출력된 4 bit adder의 입력으로 들어가고, IRJP를 발생해야 되는지 아니면 이전신호를 유지 시켜야 되는지를 결정하게 된다. 0에서 15의 출력은 이전 신호의 유지에 관한 신호이며, 이 신호와 MF 신호에 의해 memory address line의 출력 상태와 memory 출력 상태를 제어하는 신호가 발생된다.

2.2.5. Control part

Fig. 10은 PG의 제어를 담당하는 control에 관련된 구성도이다. 입력으로 들어가는 데이터는 PC data bus,

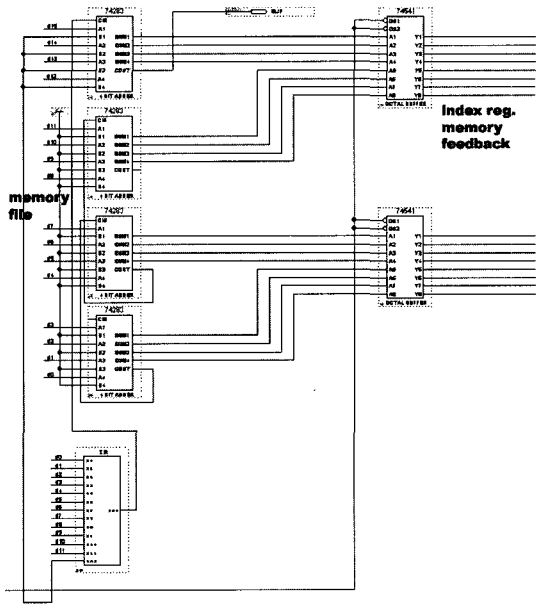

Fig. 9. Index part circuit.

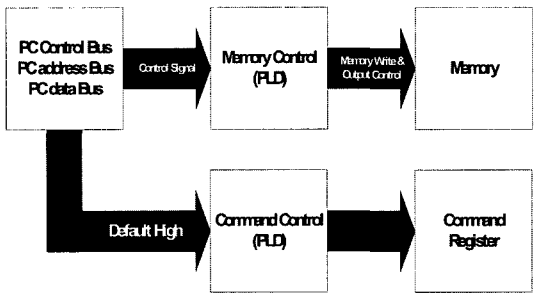

Fig. 10. Block diagram of control part.

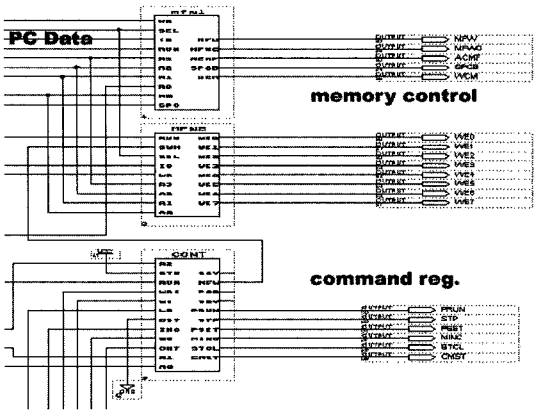

Fig. 11. PLD device circuit.

PC control bus, PC address bus가 입력으로 들어가게 된다. Memory control 하는 PLD 소자를 통해 memory 부분의 양방향 버퍼를 제어해 이전 신호를 쓸 것인지 새로운 신호를 쓸 것인지를 결정하는 제어신호와 memory output enable를 결정하는 신호를 제어하는 신호, 그리고 memory write enable 제어 신호가 발생된다.

Command control 부분은 입력으로 control bus와 default high로 정해진 입력을 사용하여 command register에 입력될 신호를 발생하게 된다. 입력된 신호는 main memory의 output enable를 제어하고, program counter에서 발생되는 memory address line을 제어하게 된다.

Fig. 11는 control part 블록의 메모리 제어 신호와 command reg.를 제어하는 신호를 발생하는PLD 소자 회로이다. PC에서의 data신호와 control bus가 입력으로 들어가게 된다. Memory file write와 control 소자는 일련의 논리연산을 정의해 놓아 입력되는 신호와의 조합으로 control 신호를 출력하게 된다.

3. 실험결과 및 고찰

3.1. Program counter

Fig. 12는 program counter의 simulation 결과이다. 이 결과는 main memory에 할당할 address발생 부분에 중점을 두었다. 점선 부분은 각 조건에 의해 memory address line 인 PCA[7~0]와 PCB[7~0]의 출력을 나타낸다. 입력된 데이터 즉 counter에서 발생되는 신호에 의해 main memory의 address line을 발생하는 program counter는 main memory에서 발생되는 신호인 MF와 data 신호에 의해 interrupt address를 제어하며 program counter에서 발생하는 address line과 main memory와의 충돌을 방지하게 된다. Fig. 12의 simulation 조건은

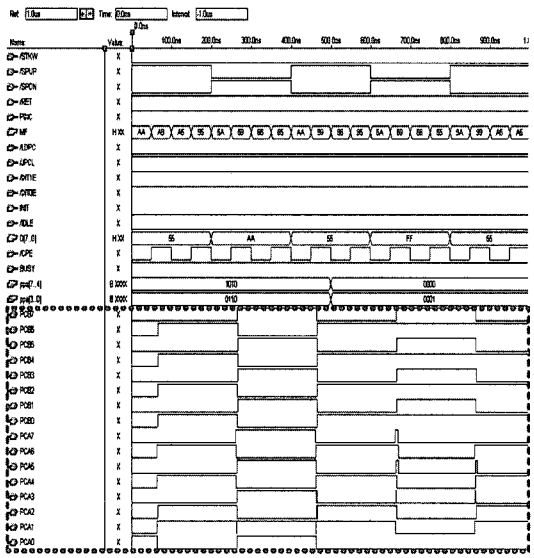

Fig. 12. The simulation results for program counter.

memory address line 이 발생할 수 있는 최적의 조건을 만족시키기 위한 입력을 사용하였다. 그리고 feedback signal을 구현하기 위해 PPA[7:0] 신호를 만들어 입력으로 사용하여 Fig. 12와 같은 결과를 얻을 수 있었다.

3.2. Address part

Fig. 13은 address part의 simulation 결과이다. 이 결과는 address A.L.U의 연산에 의한 출력과 MF신호와의 연산 후 address 발생에 중점을 두고 시뮬레이션 하였다. MFSL 신호에 의해 이전신호를 유지할 것인지 아니면 새로운 신호를 사용할 지를 결정하게 된다. 결

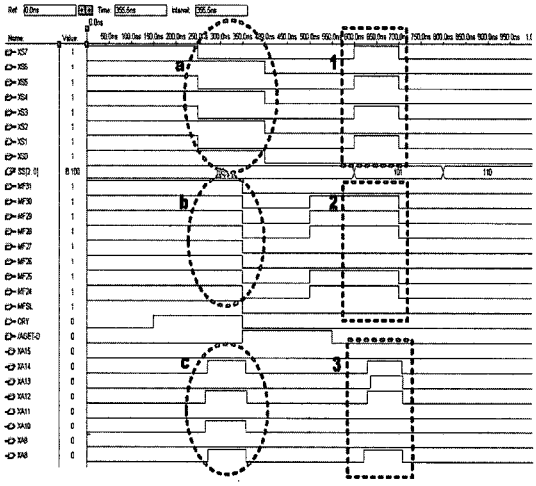

Fig. 13. The simulation results for address part.

정된 신호는 address ALU device 에 입력되어 S0, S1, S2 신호에 의해 ALU device에서 정의해 놓은 연산을 하게 된다. 이 연산에 의해 출력되는 신호는 MF신호와 AND LOGIC 연산을 하여 최종 신호를 출력하게 된다. Fig. 13의 결과에서 이 부분에 대해 설명한다면, 점선부분에서 보인 것처럼 a(1)의 입력으로 발생되는 신호와 b(2)의 입력신호와 AND 연산되어 최종결과인 c(3)가 출력하게 된다.

3.3. Data part

Fig. 14는 data part의 simulation 결과이다. address part

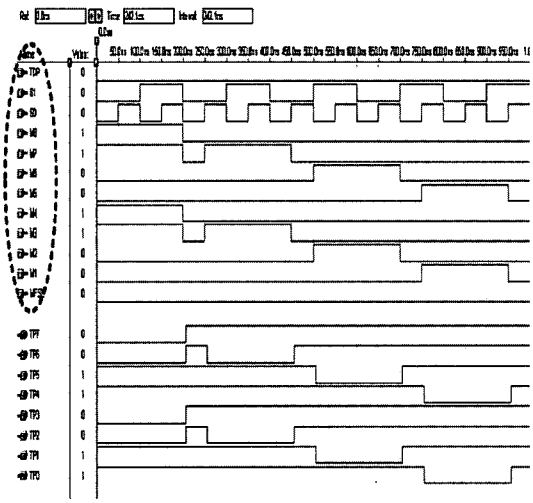

Fig. 14. The simulation results for data part.

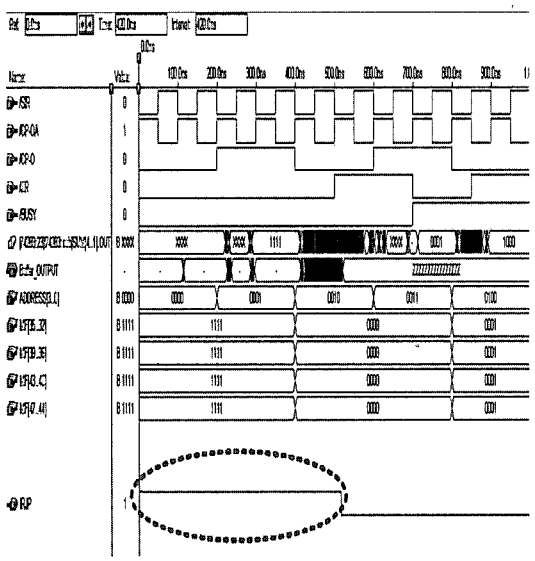

Fig. 15. The simulation results for index part.

에서 발생된 address에 들어갈 data를 출력하는 부분의 simulation 결과인데, data part 회로에서 명시된 TP 라는 PLD device에 입력되는 MF 신호와 그 외의 제어 신호들이 정의해 놓은 논리 연산에 의해서 TP7~TP0 신호를 발생하게 된다. 점선부분의 입력에 따라 출력이 결정된다.

3.4. Index part

Fig. 15는 index part의 simulation 결과이다. index part의 회로의 출력단(Fig. 15에서는 buffer output)은 feedback 신호로서 이전신호를 유지시키기 위해 사용하였다. 이 결과는 PG의 각 부분별 기능에서 설명한 IRJP 신호의 발생되는 과정을 중점으로 시뮬레이션 하였다. 입력단에서 /JIR과 /BUSY 신호가 LOW 값을 가질 때 IRO 신호가 HIGH 값을 가지도록 IR (PLD device)에 정의되어 있다. IRO에서 출력된 HIGH 값은 4 bit address의 Cin으로 들어가 다른 입력들과의 조합으로 IRJP를 발생하게 된다. 점선 부분은 IRJP가 HIGH 값을 가지는 상태로서 memory output enable과 program counter를 제어할 부분인 SETPC로 들어가 최종적인 신호를 발생 하게 되어, main memory 에서는 memory output enable 신호를, program counter 에서는 memory address line의 출력을 제어하는 신호를 발생하게 된다.

3.5. Control part

Fig. 16은 control part의 simulation 결과이다. Data

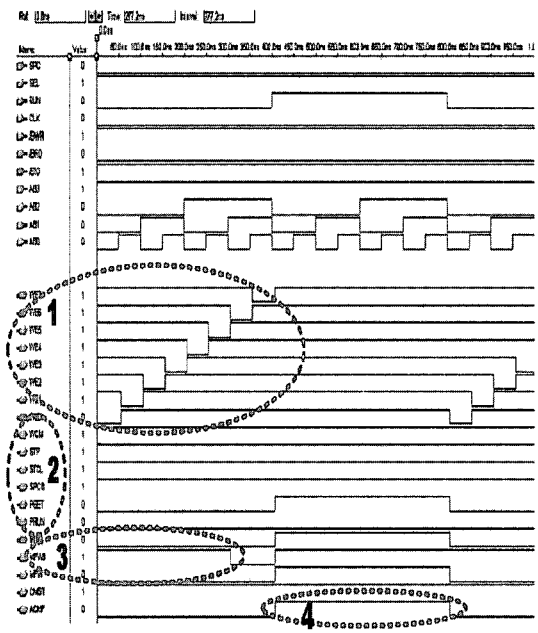

Fig. 16. The simulation results for control part

가 입력되면 PLD device에 정의해 놓은 연산에 의해 출력 ①, ②, ③, ④를 발생하게 된다.

①은 memory write enable 신호에 대한 출력인데, 이 신호는 memory file write PLD device에서 정의된 논리 연산에 의해서 발생된다. ③, ④에서 발생되는 신호 또한 이 소자를 통하여 발생된다. ②의 출력은 command register 의 제어 신호이다. 이 신호는 control PLD device 에 정의된 논리 연산을 통하여 발생하게 된다. ③은 main memory part의 버퍼 방향을 결정하는 신호선 이고, ④는 memory output enable 제어 신호 발생부분이다.

4. 결 론

고집적화 되고 기능이 복잡해진 반도체 소자 정상 동작 여부를 효율적으로 테스트하기 위해서는 테스트 패턴 생성기를 효율적으로 설계하여 빠른 시간 안에 테스트가 이루어지도록 하는 것이 중요하다.

본 연구에서는 pattern generator를 연구하여 반도체 칩의 생산라인에서의 생산성 증대 및 품질 향상에 중점을 두었다. functional test를 빠른 시간에 수행하기 위해 검사 할 소자의 특성에 따라 검사 패턴을 발생하게 된다. 이 연구에서는 알고리즘 패턴 적용을 위해 counter를 사용하여 address를 발생 시켰다. 또, 전체적인 구성에서 각 부분이 어떤 기능을 하는지에 중점을 두었다. control part의 command part에 의해 전체적인 PG의 제어가 이루어짐을 확인 했으며, 각 부분별 출력되는 신호가 어떤 과정을 거쳐 발생되는지를 알 수 있었다. 64 Mb 메모리에 모두 0을 쓴다고 하고 한 개의 셀에 0을 쓰는 동작에 4줄의 프로그램이 필요하다면 2억 5600만 라인의 길이를 갖는 프로그램이 필요하다. 이 pattern generator를 사용하면 이러한 과정의 효율성을 증대 시킬 수 있으며 보다 정확한 오류를 검출 할 수 있게 된다. 이로 인해 테스트 소요시간과 생산 비용이 감소되어 산업화의 경쟁력이 될 것이다.

감사의 글

본 논문은 한국과학재단지정 호서대학교 RRC의 연구지원으로 이루어진 것임.

참고문헌

1. A.Stevens, "Introduction to Component Testing" Addison-Wesley Publishing Company, 1986
2. 강성호, 김규철, 소병세, 홍성제 공저 메모리 테스트.