

ASIC Test

金 鎮 九

三星電子(株) 半導體部門
ASIC 개발그룹장

I. 서 론

ASIC(application specific intergrated circuit)은 수 많은 기본 소자들을 하나의 반도체 chip에 집적시킴으로써 원가를 절감하고, 성능을 향상시키며, 안정도를 증진시킨다는 장점을 갖는다. 이러한 장점을 유지하려면, ASIC을 구현하기 위한 설계(design), 제조(fabrication), 검사(test)등 제반 과정이 조화를 이루면서 효율적으로 진행되어야 함은 물론이다. 특히 AISC은 제조 공정상에서 open이나 short등의 물리적 결함(physical defects)을 내포할 가능성이 있으므로 이에 대한 철저한 검사는 매우 중요하다. 만일 물리적 결함을 내포하고 있는 불완전한 chip이 검출되지 않고 출하되어 board나 system에서 사용되었다면, 이에 대한 testing cost는 급격히 증가한다. 예를 들어서 chip에서 결함을 검출하기 위한 비용은 \$0.3이면, board에서 동일한 결함을 검출하기 위한 비용은 \$3이며, system에서는 그 비용이 \$30에 달하게 된다.^[1]

최근에 이르러 전자제품의 급격한 발달로 인하여 "Time to Market" 개념이 극히 중요한 요소로 대두되었다. 아무리 우수한 제품도 적절한 시기에 시장에 출하되지 못하면 상품가치는 급격히 하락하므로, 제품개발 기간의 단축은 시장에서의 성공여부에 결정적인 영향을 준다. 특히 ASIC은 이와 같은 시간의 개념이 가장 중요하게 적용되는 전자제품의 하나이다. 여기서 중요한 점은 time to market이란 제품의 구상부터 설계, 제조, 검사를 거쳐 시장에 출하되기까지의 모든 기간을 포함하여야 한다는 점이다. 과거 chip의 집적도가 낮았던 때는 검사에 소요되는 시간이 설계 및 제조과정에 비하여 상대적으로 작았으나, VLSI(very large scale

integration) 시대에 접어들면서 더 이상 무시할 수 없는 요인이 되고 있다. 설계와 제조기간이 혁신적으로 단축된다 하여도 검사에 많은 시간이 소요된다면, 종합적인 time to market 측면의 경쟁력은 약화될 것이다.

이와 같은 사실들을 종합해 볼 때, ASIC검사를 통하여 전체적인 testing cost를 절감하여야 하며, 또한 검사에 소요되는 시간을 단축함으로써 종합적인 time to market 측면의 경쟁력을 강화하여야 한다는 것이다. 이러한 목표를 달성하려면 과거와 같은 주먹구구식이 아닌 체계적인 접근이 필요하며, 특히 설계 초기부터 검사를 고려하여야 한다는 점이 중요하다. 그림 1은 이러한 개념을 반영한 ASIC 구현 순서도로 specification 작성부터 chip 생산까지 전 과정을 나타내고 있다. 여기서 특기할 사항은 test plan 작성을 chip 설계와 병행하여야 한다는 점이다. 즉, 검사에 사용할 장비 및 검사의 종류를 미리 고려하여 이를 설계에 반영시키므로써 높은 fault coverage를 갖는 test pattern 및 program의 작성등 검사에 관련되는 모든 과정을 효율화 하여야 한다. 논리 회로에 scan이나 BIST(built-in self test)등을 포함시킬지의 여부도 미리 결정하여야 함은 물론이다.

일반적으로 ASIC 검사는 일련의 test pattern을 chip에 인가하고, 출력을 측정하되, 예상치와 비교하는 3단계로 이루어진다. 예상치와 상이한 출력이 관측되는 chip은 물리적 결함을 내포하고 있는 것으로 간주되어 불량으로 취급된다. 이러한 과정에서 관건은 어떠한 test pattern을 인가하여야 하는가이다. 가장 완벽한 방법은 가능한 경우를 모두 고려하는 testing(exhaustive testing)일 것이나 이는 현실적으로 불가능하다. N개의 입력과 M개의 latch를 포함하는 회로의 경우, exhaustive testing을 하려면 $2^{(N+M)}$ 개의 test pattern이 필요

하다. 예를들어서, 1초에 1백만개의 test pattern을 인가할 수 있는 검사 장비를 사용하는 경우, 입력이 25개, latch가 50개인 회로를 검사하려면 무려 10억년 이상이 소요된다.^[1] 따라서 ASIC 검사의 핵심은 검사의 질을 유지함과 동시에 exhaustive testing의 문제점을 해결하는 것이다. 본 논문은 이에 대한 여러가지 접근방법에 대한 개요를 기술하고 있다. 또한 실제검사에서 사용될 test program 작성시 고려해야 할 사항도 설명하고 있다.

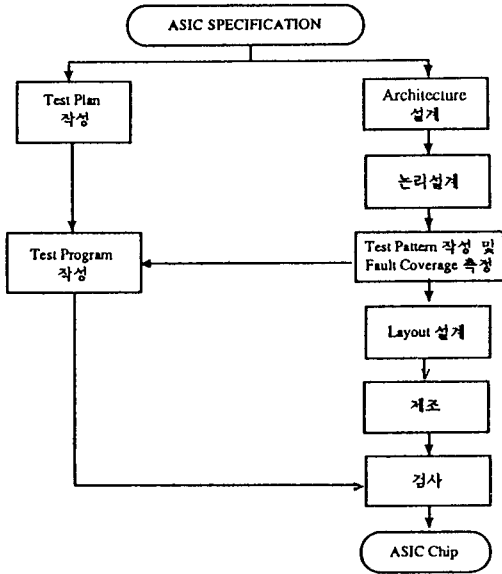


그림 1. ASIC 구현을 위한 순서도

Test에 있어서 fault coverage가 무엇인지를 자동차의 조립후 test하는 예를 들어서 설명해 보겠다. 조립공장에서 굴러나온 자동차를 걸로 대략 살펴보는 visual inspection 후에 test driver가 그차를 직접 몰고 운동을 한바퀴 돌아보았다고 하자. 그후 그차가 소비자에게 전달되었을때 과연 차가 후진은 하는가? Radio가 동작하는가? Headlight 및 windshield wiper가 동작하는가 등등 문제는 얼마든지 있을 수 있으며 이것을 다 test 해보는 exhaustive testing 방법을 적용할 수는 없을 것이다. 가장 통계적으로 문제가 있을 수 있는 부분을 test 해보는 방법을 취할 수 밖에 없으며 그 정도가 곧 falt coverage인 것이다.

II. Fault Modeling

물리적인 소자인 ASIC은 본질적으로 물리적 결함을 내포할 가능성이 있다. 물리적 결함은 gate oxide pin hole, line open 및 short 등 그 종류가 극도로 다양하다. 따라서 이들을 분석하여 검출하는 기법을 개발하려면 추상적인 computational model이 필요하다. 이러한 fault model은 물리적 결함의 특성을 효과적으로 반영하여야 한다.

현재 가장 보편적으로 사용되는 gate-level single line stuck-at fault model은 gate 사이에 line이 logical 1이나 0으로 고정되었으며, 이와 같은 fault가 전체 회로 상에서 오직 1개만 존재한다고 가정하는 것이다. 이러한 model을 사용하였을 때의 장점은 첫째 gate level에서의 model이므로 공정 기술에 무관하며, 둘째 model이 단순하여 software algorithm에 효율적으로 적용 가능하고, 셋째 물리적 결함을 비교적 잘 반영한다는 점이다. 실제로 사용되고 있는 대부분의 fault 관련 상용 software는 이 stuck-at fault model을 근거로 하고 있다. 그러나 COMS transistor로 구성되는 VLSI시대에 접어들면서, stuck-at fault model에 대한 반론이 제기되고 있다.^[2] 즉, CMOS transistor가 open이나 short되는 물리적 gate level에서의 stuck-at fault model로는 정확하게 표현할 수 없다는 것이다. 예를 들어서 그림 2에 나타나 있는 2-input CMOS NOR gate에서 PMOS transistor, P1이 open 되었다고 가정하자. 이를 검출하기 위해서는 2개의 test pattern(10, 00)이 필요하다. 이러한 현상은 전통적인 gate level에서의 fault model로는 표현이 불가능하므로, transistor level에서의 modeling이 필요하게 된다. 그러나 이러한 model은 정확도가 높은 반면, computational complexity를 급격히 증가시키므로 아직은 연구 단계에 머무르고 있다.

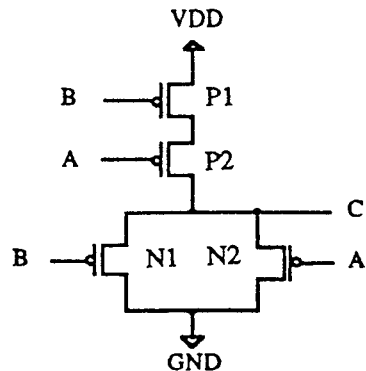


그림 2. 2-input CMOS NOR gate

III. Fault Simulation

Test quality 기준으로 불량비율인 reject ratio에 대한 정확한 data를 얻기 어려우므로 간접적인 방법으로 fault modeling을 통한 fault coverage를 사용하게 되었다. 그림 3은 fault simulator의 한 예로서 실질적으로 fault simulator는 임의의 digital circuit에 대하여 주어진 test pattern에 의해 어떤 fault가 발견되는지를 찾아 fault coverage를 구하는 것 이외에 test pattern의 quality 측정과 test generation process의 시간을 단축시키는 역할도 한다. 현재까지 가장 널리 알려진 gate-level single line stuck-at fault model에 대한 fault simulation의 algorithm을 크게 세가지로 나눌 수 있다.

첫째는 명백한 fault 결과로부터의 접근이다. 한번에 1개의 fault simulation을 하여 n개의 fault에 대하여 (n+1)번 logic simulation하는 가장 간단하고 원시적인 방법인 serial, good circuit 및 각각의 fault circuit의 동작이 독립적이라는 점을 착안하여 bitwise operation을 이용 한번에 computer machine word size개의 fault simulation하는 parallel^[3], good circuit과 모든 fault circuit을 동시에 simulation하고 fault circuit의 경우 good circuit과 전번 test vector로 부터 fault evaluation list를 비교하여 다른 부분만 simulation하여 serial과 parallel simulation보다 빠르고 full timing delay로 simulation하여 가장 보편적으로 사용되는 concurrent^[4], concurrent와 같으나 gate evaluation 방법만 다른 deductive^[5], concurrent의 evaluation gate list의 searching과 sorting 및 dynamic memory management로 부과되는 시간과 memory size를 줄이기 위해 각각의 fault에 대한 simulation하기 전에 good circuit state를 복원하며 heuristic들의 적용할 수 있는 single fault propagation, single fault propagation의 변형으로 각각의 fault circuit simulation을 전에 simulation한 circuit으로부터 구하는 differential^[6], computer machine word size개의 test pattern들을 동시에 single fault propagation하는 parallel pattern single fault propagation(PPSFP)^[7] 등을 들 수 있다.

둘째는 분석적 접근이다. 종류로는 circuit structure를 분석하여 fault free simulation을 하고 backward processing하여 주어진 test pattern에 대하여 각각 net의 detectability를 결정하는 critical path tracing method (CPT)^[8], critical path tracing과 같지만 forward fault propagation에 필요한 stem수를 줄이는 fault informa-

tion tracing algorithm(FIT)^{[9][10]}, 검출할 수 없는 fault set을 구하는 non-critical path tracing algorithm^[11], simulation해야 되는 fault 수를 줄이는 inactive faults identification방법^{[12][13]}들이 있다.

세째는 통계적 접근 방법이 있다. Circuit size, 주입된 fault 수, 적용된 test pattern 수에 좌우되는 fault simulator의 computation complexity를 줄이는 방법으로 fault simulation을 하지 않고 good circuit에 대한 simulation이 수행될 때 node의 activity를 사용하여 node의 controllability와 observability로 fault coverage를 측정하는 STAFAN^[14]이 있다.

이상과 같이 각각의 algorithm은 장단점을 가지고 있어 이들을 상호보완한 새로운 fault simulator들이 나오리라 예상된다. 즉, Mentor의 Quickfault, Hilo의 Hifault, Verilog의 Verifault들은 concurrent algorithm을 사용한 fault simulator로 concurrent algorithm의 단점을 가지고 있어 이들을 보완한 simulator에 경쟁력을 상실하게 될 것이다.

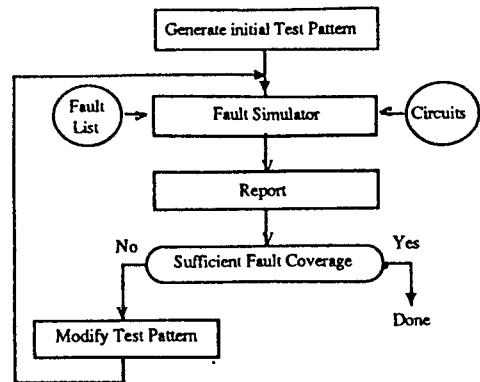


그림 3. Fault simulator의 사용

IV. Automatic Test Pattern Generation

Test generation의 목적은 주어진 circuit에 대하여 가장 빠른 시간내에 높은 fault coverage를 구할 수 있는 가장 적은 test pattern을 구하는 것이다. 이를 위하여 상세한 circuit에 대한 이해없이 single-stuck at fault들에 대하여 fault의 영향을 primary out에서 관측되는 input stimulus를 자동적으로 구하는 ATPG (automatic test pattern generations)에 대한 연구 및

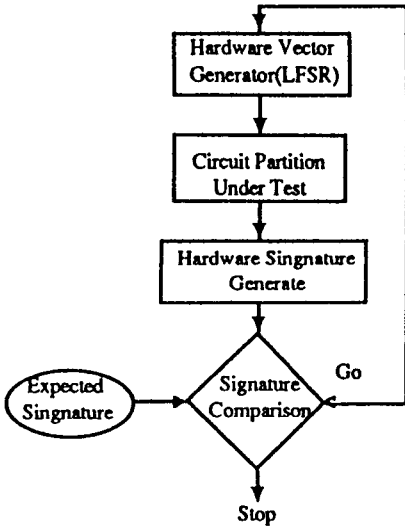


그림 4. Exhaustive test generation

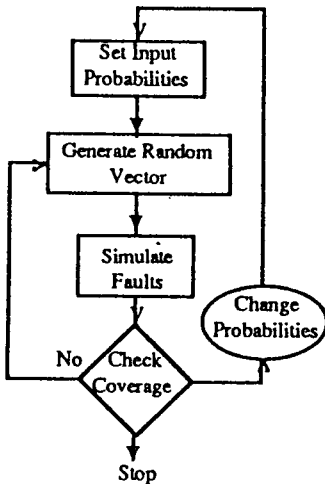


그림 5. Random test generation

개발은 1960년대 이래 계속되어 왔으나, circuit size가 클수록 효율이 떨어지고, sequential circuit에 대해서는 아직 완벽한 해결책을 제시하지 못하고 있다. 따라서 ATPG 사용뿐 아니라 designer가 circuit으로부터 구한 functional test pattern이나 high level에서 device의 구조와 동작을 알 수 있는 emulator^[15]등을 이용하는 방법들이 제시되었다. ATPG를 구조에 따라 네가지로

나누어 보면, 첫째는 exhaustive test를 위한 exhaustive test generation이다. 그림 4는 한 예이다. LFSR (linear feedback shift register) 등의 hardware나 software를 이용하며 큰 circuit에 대하여 부분 circuit 들을 나누어 사용할 수 있다. 둘째로는 circuit의 topology에 무관한 random pattern으로 test pattern을 만드는 random test generation이 있다. 그림 5는 random test generation의 한 예이다. Random pattern generation은 test vector들을 만들기는 쉬우나 같은 fault에 대하여 deterministic test generation보다 test set 크기가 커서 test pattern의 quality가 떨어진다. 셋째는 algorithm이 적용된 deterministic test generation이다. 종류로는 path sensitization을 이용 D-frontier나 J-frontier를 구하여 input vector를 결정하며 exclusive-OR gate에 대하여 backtracing 문제를 갖고 있는 D-algorithm^{[16][17]}, brach와 bound algorithm을 통하여 모든 가능한 input 조합을 고려하는 PODEM (path oriented decision making)^[18], 불필요한 backtracing을 primary input까지 하지 않게 하고 multiple backtracing을 동시에 하는 FAN(fanout-oriented test generation)^[19] 등이 있다. 이들은 combinational circuit에 잘 적용되지만 아직 sequential circuit에는 많은 문제점을 가지고 있다. 넷째로는 둘째와 셋째의 test generation들의 장점을 살리는 algorithm이다. 이들은 random critical path를 이용한 RAPS(random path sensitization)^[20]과 random test generation에 critical path tracing을 적용한 SMART(sensitizing method for algorithm random testing)^[21] 등이 있다. 앞에서 언급한 바와 같이 각각의 test generator는 각각의 문제점을 가지고 있으며 이를 해결하기 위하여 새로운 algorithm 개발뿐 아니라 sequential circuit을 combination화 하여 사용하는 방법이나 switch-level test generation이 개발되고 있다. 또한, 현재의 tool들의 장단점을 고려한 ATPG system 측면에서도 활발한 연구가 계속되고 있다.^[22]

V. Testability 분석

앞에서 설명한 test pattern generation이나 fault simulation을 하기전에 testing에 관련된 fault simulation의 cost, test pattern generation의 cost, test equipment의 cost등에 영향을 주는 design 요소들을 분석하여 circuit을 test할 때 예상되는 문제점들을 초기에 진단하는

방법이 있다. 이러한 분석 방법은 정확도가 떨어지지만 circuit의 test하기 힘든 부분을 찾거나 test 개발을 위한 노력의 정도를 측정하거나 test vector의 크기를 측정하는 데 매우 효과적이다.

현재까지 널리 알려진 testability 분석 방법을 살펴보면, 첫째로 단순히 circuit의 각 node에서 primary input으로 부터 이 node를 통제할 수 있는 정도인 controllability와 primary output에서 이 node의 상태를 아는 정도인 observability를 구하여 testability를 직접 계산하는 방법이다. Gate level에서 gate들의 control vector들을 이용하고 signal의 상호관계를 무시하여 complexity가 node수에 선형적으로 비례하는 SCOAP (sandia controllability/observability analysis program)^[23]가 이에 속한다. 이는 각 node의 combinational zero controllability, combinational one controllability, combinational observability, sequential zero controllability, sequential one controllability, sequential observability들을 구하는 방법이다. 그림 6은 AND gate의 b input에 대한 observability를 line a를 1로 하고 line c의 observability와 primary input부터 먼 정도를 나타내는 depth factor로 나타낸 것이다. 또한, good simulation의 data로 부터 controllability와 observability를 구하는 STAFAN이 있다. 둘째는, SCOAP가 제시하지 못하는 각각의 circuit에 대한 test 할 수 있는 정도나 fanout의 문제점들을 보완한 circuit node의 signal probabilities를 구하여 detection probability를 구하는 방법이다^[24]. 방법으로는 각 gate마다 직접적으로 계산하거나 signal probability의 최대값과 최소값을 구하는 cuttingalgorithm^[25]등을 사용할 수 있다. 그림 7은 line g에서 stack-at 0의 signal probability를 cutting algorithm으로 구한 것이다.

이상과 같이 chip size가 커지고 fault simulation이나 test generation에 부과되는 complexity가 커지면서 효과적인 많은 testability 분석 방법들이 제시되고 있다.

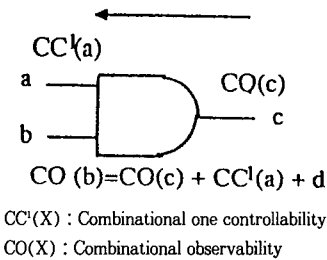


그림 6. AND gate of SCOAP

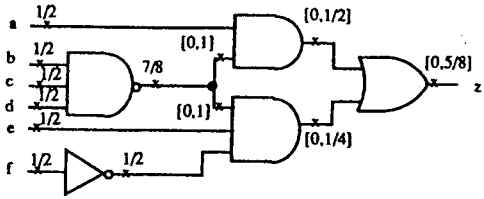


그림 7. Cutting algorithm의 예

VI. Testability를 고려한 설계

Chip size가 커지면서 test의 software tool 개발의 한 계성이 드러남에 따라 test generation cost와 testing cost를 줄이고 compter-aided설계 tool들을 효과적으로 사용하기 위한 circuit의 설계방법들이 제안 되었다. 이들을 크게 두 종류로 나누어 살펴보면, 첫째는 design guide line에 따른 접근이다. Test point, initialization, monostable, multivibrators, oscillator와 clocks, counter와 shift register, partitioning large circuits, logical redundancy, breaking global feed-back path등 각각에 대해 고려하는 방법으로 Ad Hoc설계^[26]가 이에 속한다. 둘째는 구조적 접근 방법이다. Scan-cell을 이용하여 scan-in과 scan-out의 path를 circuit에 부가하여 내부 signal의 controllability와 observability를 구하는 scan path^[27]가 있다. 그림8은 scan-cell의 한 예이고 그림 10은 그림9의 circuit에 scan-cell을 적용한 full scan design 예이다. T=0일때 정상동작을 하며 T=1이면 test동작을 한다. 또한 IBM사에서 개발 사용하는 level-sensitive scan design(LSSD)^[28]은 polarity-hold, hazard-free, level-sensitive한 latch를 사용하며 clock pin이 동기할때 input data를 level sensitive한다. 따라서, clock edge가 rising과 falling time에 무관하며 network이 test generation에 관련된 combinational logic 이고, 모든 hazard와 race가 제거되어 test generation 이나 testing하기 쉽다. 한 chip의 I/O pin을 따라서 scan chain을 형성 board level에 사용되는 boundary scan design 방법도 있다. Board는 많은 다른 vendor의 chip들로 구성 될 수 있기 때문에 설계의 표준화가 중요하여 IEEE는 1990년 Joint Test Action Group (JTAG)의 제안으로 IEEE STD. 1149.1을 도입하여 boundary scan design의 응용에 박차를 가했다. 그림10은 normal mode, scan mode, capture mode, updata

mode로 동작하는 boundary scan cell의 예이다. 또한 boundary scan cell은 extest mode, intest mode, sample mode, bypass mode등의 각각의 test 형태에도 사용할 수 있다. 그러나 scan design cell인 flip-flop이나 latch는 다른 cell들 보다 복잡하여 board와 silicon area에서 expensive하고 한개 이상의 I/O pin이 사용되며, test pattern을 shift해야 함으로 test time이 증가하고, scan design으로 하기 힘든 design들도 있다. 이밖에도 test하고자 하는 회로에 self-testing을 하기 위해 test pattern generation(TPG)와 test data evaluation(TDE)을 같은 chip에 내장시키는 방법인 built in self-test(BIST)가 있다. 그림11은 간단한 BIST의 구조이다. 이 방법은 따로 testing하기 위한 장비가 필요 없지만 BIST를 초기화 하거나 go/on-go정보를 읽는데 외부 pin이 필요하고, 기존의 register를 변환하는데 overhead가 요구되며 회로의 성능이 떨어진다는 단점들이 있다.

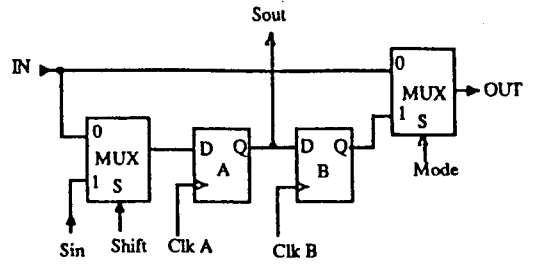


그림 11. A boundary scan cell

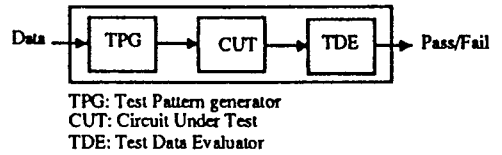


그림 12. BIST의 구조

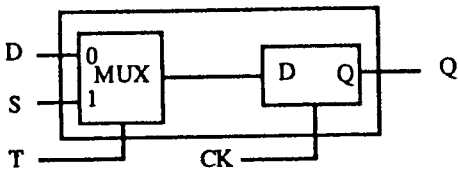


그림 8. Scan design cell

이상과 같이 DFT(design for testability)는 circuit modification을 필요로 하고, circuit의 area, I/O pin, delay들이 증가하기 때문에 절대적인 방법이 될 수 없고 yield를 고려하여 적절히 사용되어야한다.

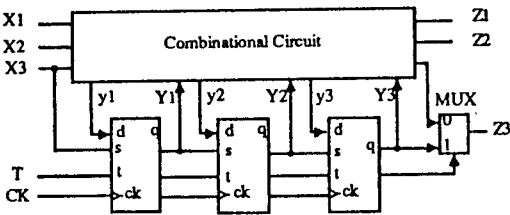


그림 9. Original circuit

Ⅶ. Test Program 작성

Testing의 형태를 크게 두종류로 나누면, 첫째는 parametric test이다. 각 chip에 대해 input과 output 단자의 전압 또는 전류 특성을 측정하는 short test, open test, maximum current test, leakage test, output drive current test, threshold levels test인 DC parametric test나 circuit내의 임계경로 test와 같은 propagation delay test, setup and hold test, functional speed test, access time test, refresh와 pause time test, rise와 fall time test등과 같은 AC parametric이 이에 속한다. 둘째는 chip의 내부 node를 test하여 design 동작을 검사하는 functional test가 있다. 이들 test로 부터 device의 불량품을 판정 할 수 있고 조립과정의 정보 및 design의 취약한 점을 알려주는 가장 널리 알려진 automatic test equipment(ATE)^[29]가 이에 속한다.

Test program은 바로 test 장비를 동작시키기 위한

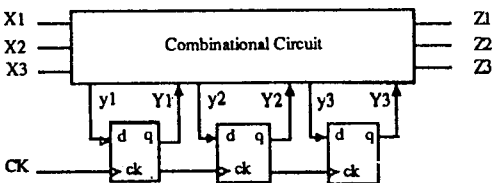


그림 10. Full scan design

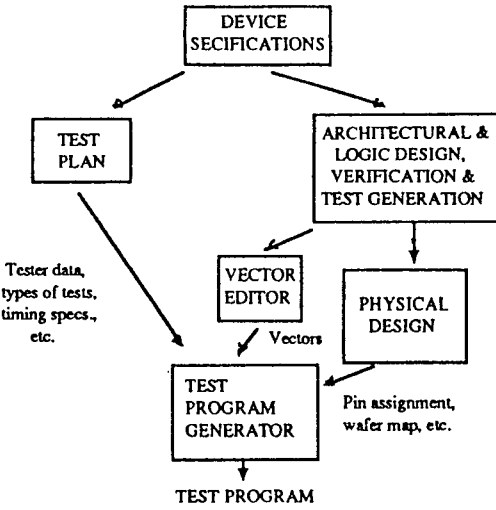


그림 13. Test program generation

명령 program이다. 그림 13은 test program generation의 한 예로 test plan부터 얻은 test 형태에 대한 정보와 test specification, layout부터 얻은 device의 물리적 자료, simulator부터 test vector나 signal의 timing 정보들로 test program을 만들어 준다. Vector editor는 design 검증을 위해 사용한 simulator vector를 test 장비에 맞게 변형시키는 tool로 test가 시작되기 전 device 내부의 모든 logic은 반드시 초기화 되어야 하는 device의 초기화 문제, tester의 hardware memory size 및 memorydepth에 따라 제한을 받는 test vector size, test frequency, timing violation, 각 pin은 전 test cycle을 통하여 동일한 format을 유지하며 전구간을 통하여 같은 delay 및 pulse width를 유지해야 하는 input waveform기술, deviceoutput data가 simulation 시 예측하였던 data와 동일한지 여부를 비교하기 위한 output strobe 지점 설정 등을 고려해야 한다. 또한, functional test pattern 작성시에는 device 기능을 완벽하게 test할 수 있는 pattern set작성, test 장비와 호환성 여부, critical path에 대한 spec등이 고려되어야 한다. Test 장비가 다양해지면서 호환성 있는 test program의 작성이 중요시 되고 있다^[30]. 또한, scan이나 BIST등과 같은 DFT(design for testability)방법이 test program generation의 사용에 제안되고 있다.


Ⅷ. 결 론

지금까지 “Time to Market”측면에서 ASIC 구현을 위한 효과적인 검사방법은 설계초기에 검사를 고려해야 한다고 역설하였으며 현재까지 널리 알려진 test방법 및 이들의 장단점들을 설명하였다.

결론적으로 보다 빠르고 정확한 각 분야에 따른 test tool 개발, design에 따른 test cost를 줄이기 위한 방법 설정, circuit testability로 test vector generation을 고려하는 design engineer, test 장비를 다루는 test engineer, fault coverage를 고려하는 quality engineer 등 각 능동적이고 협조적인 지원등 다각적인 노력이 ASIC test에 필요하다. 즉, fault modeling과 fault simulation 개발은 automatic test pattern generation 개발에 직접적인 영향을 주는 반면에 testability 분석과 testability를 고려한 설계는 design에 대한 이해를 필요로 한다. 따라서 이들 방법의 장점을 살리는 적절한 방법의 모색은 chip 개발 비용에 검사 비용이 점점 비중을 더해가는 현 시점에서 매우 중요하다.

參 考 文 獻

- [1] T.W.Williams and K.P.Parker, “Design for Testability-A Survey”, IEEE Transactions on Computers, vol. C-31, no.1, pp.2-15, Jan. 1982.
- [2] R.L.Wadsack, “Fault Modeling and Logic Simulation of CMOS and MOS Integrated Circuits”, Bell Syst. Tech. Jour., vol.57, pp. 1449-1474, May-June 1978.
- [3] E.W.Thompson and S.A.Szygenda, “Digital logic simulation in a time-based, table driven environment, part2, parallel fault simulation,” IEEE Computer, vol.8, no.3, pp.38-40, 1975.
- [4] E.G.Ulrich and T.Baker, “The Concurrent Simulation of Nearly Identical Digital Networks,” Proc. of 10th DA Workshop, vol.6, pp.145-150, 1973.
- [5] D.B.Armstrong, “A Deductive method for simulating faults in logic circuits,” IEEE Trans. Computer, vol.C-21, no.5, pp.464-471, 1972.
- [6] W.-T.Cheng and M.-L.Yu, “Differential Fault Simulation-A Fast Method Using Minimal Memory”, Proc. of 26th DAC, pp.424-428, June 1989.
- [7] J.A.Waicukauski, E.B.Eichelberger, D.O.Frolenza, E.Lindbloom, T.McCarthy, “Fault Simulation for Structured VLSI,” VLSI Systems Design, pp.20-

- 32, Dec. 1985.
- [8] M.Abramovici, P.R.Menon, and D.T.Miller, "Critical path tracing : An alternative to fault simulation", *IEEE D&T*, vol.1, no.1, pp.83-93, February 1984.
- [9] Y.Kitamura, "Exact Critical Path Tracing Fault Simulation on Massively Parallel Processor AAP2", Proc. of the ICCAD, pp.474-477, Nov. 1989.
- [10] Y.Kitamura, "Sequential Circuit Fault Simulation by Fault Information Tracing Algorithm : FIT", Proc. of 28th DAC, pp.151-154, 1991.
- [11] S.B.Akers, B.Krishnamurthy, S.Park and A. Swaminathan, "Why is Less Information From Logic Simulation More Useful in fault Simulation?", Proc. International Test Conf., pp.786-800, September 1990,
- [12] E.M.Rudnick, T.M.Niermann and J.H.Patel, "Methods for Reducing Events in Sequential Circuits Fault Simulation", to be published.
- [13] H.K.Lee and D.S.Ha, "An Efficient, Forward Fault Simulation Algorithm Based The Parallel Pattern Single Fault Propagation", Proc. of International Test Conf., pp.946-955, Oct.1991.
- [14] S.K.Jain and V.D.Agrawal, "Statistical fault analysis", *IEEE Design & Test of Computers*, vol. 2, pp.38-44, Feb. 1985.
- [15] W.E.Den Beste, "Using a Software Emulator to Generate and Edit VLSI Test Patterns," *Electronics Test*, pp.42-52, March 1984.
- [16] J.P.Roth, "Diagnosis of Automata failures: A calculus and a method," *IBM Journal of Research and Development*, vol.10, no.4, pp.278-291, July 1966.
- [17] J.P.Roth, W.G.Bouricius and P.R.Schneider, "Programmed algorithms to compute tests to detect and distinguish between failures in logic circuits," *IEEE Trans. on Electronic Computers*, vol. EC-16, no.10, pp.567-579, October 1967.
- [18] P.Goel, "An implicit enumeration algorithm to generate tests for combinational logic circuits," *IEEE Trans. on Computers*, vol.C-30, pp.215-222, March 1981.
- [19] H.Fujiwara and T.Shimono, "On the acceleration of test generation algorithms", *IEEE Trans. on Computers*, vol.C-32, no.12, pp.1137-1144, December 1983.
- [20] P.Goel, "RAPS test pattern generation," *IBM Technical Disclosure Bulletin*, vol.21, no.7, pp. 2787-2791, December 1978.
- [21] M.Abramovici, J.J.Kulikowski, P.R.Menon, and D.T.Miller, "SMART and FAST: Test generation for VLSI scan-design circuits," *IEEE Design & Test of Computer*, vol.3, no.4, pp.43-54, August 1986.
- [22] V.D.Agrawal and P.Agrawal, "An automatic test generation system for ILLIAC IV logic boards," *IEEE Trans. on Computers*, vol. C-21, no.9, pp. 1015-1017, September 1972.
- [23] L.H.Goldstein and E.L.Thigpen, "SCOPA: Sandia Controllability/Observability Analysis Program," Proc. 17th Des. Auto. Conf., Mineapolis, MN., pp. 190-196, June 1980.
- [24] K.P.Parker and E.J.McCluskey, "Analysis of logic circuits with fault using input signal probabilities," *IEEE Trans. on Computers*, vol.C-24, pp. 573-578, May 1975.
- [25] J.Savir, G.S.Ditlow and P.H.Bardell, "Random pattern testability," *IEEE Trans. on Computers*, vol. C-33, pp.79-90, January 1984.
- [26] R.P.Davidson, "Some Straightforward Guideline Help Improve Board Testability." *Electronic Design News*, pp.127-129, May 5, 1979.
- [27] S.Funatsu, N.Wakatsuki and A.Yamada, "Designing Digital Circuits with Easily Testable Consideration," Proc. Test Conf., pp.98-102, September 1978.
- [28] E.B.Eichelberger, T.W.Williams, "A Logic Design Structure for LSI Testing," Proc. 14th Design Automation Conf., pp.462-468, June 1977.
- [29] A.C.Stover, *ATE: Automatic Test Equipment*, Mc Graw-Hill, New York, 1984.
- [30] R.L.Hickling, "Tester Independent Problem Representation and Tester Dependent Program Generation," Proc. Int. Test Conf., Philadelphia, PA, pp. 476-482, October 1983. 

筆 者 紹 介


金 嶺 九

1950年 2月 17日生

1972年 서울대 전자공학과

1976年 미국 Rutgers University (석사)

1979年 미국 Rutgers University (박사)

(Electrical Engineering)

- 미국 : 1979年 ~ 1988年 RCA David Sarnoff Research Center
 GTE사, Senior Member Technical Staff
 Gould AMI사, Manager, Auto Place & Route Group
 Fairchild Research Center, Manager CAD Development
 California Device Inc., Manager Design Automation
- 한국 : 1988年 10月 삼성반도체 입사, ASIC 사업부장(직위 : 理事)
 현재 삼성전자(주) 반도체부문 ASIC 개발그룹장 (직위 : 理事)