論文
2-6-6

HDL을 이용한 고장안전(Fail-Safe) 인터페이스 설계 (The Design of Fail-Safe Comparator by HDL)

양 성 현* 백 순 흠**
(Sung-Hyun Yang) (Soon-Huem Paik)

요 약

본 논문에서는 결함허용(Fault Tolerant) 시스템에 의해 발생된 2진 신호를 고장안전(Fail-Safe) 신호로 변환하는 SFS(Strongly Fail-Safe) 인터페이스를 설계하였다.

SFS 특성은 자체검사(Self-Checking) 기법을 근거로 만족할 수 있었으며, 이러한 인터페이스는 개별 부품에 의한 기존의 고장안전 인터페이스 설계와는 달리 집적화 가능성을 보여 주었다. 또한 고장안전 시스템 설계에 대한 정의를 함으로써 새로운 시스템 구현 방법을 제시하였다.

ABSTRACT

This paper presents the design of strongly fail-safe interface which transform binary signals, generated by fault-tolerant system into fail-safe signals.

The strongly fail-safe property is achived by means of self-checking techniques. It can be shown for this interface to be integreated while the conventional fail-safe interface require using discrete components. This paper also presents the new implementation methods by the definitions for fail-safe system.

1. 서론

컴퓨터 시스템은 임무를 수행하는 중 물리적인 구성성분(하드웨어 부품/소프트웨어 모듈)에서의 결함 또는 설계 오류에 의한 원인으로 인해 시스템이 고장을 일으킬 수 있다. 이러한 경우 신뢰성과 안전성이 중요시되는 시스템은 최소한 위의 두 가지 원인으로부터 시스템 고장이 발생되지 않도록 해야 하며, 시스템 고장 발생 시에 최소한의 피해로 국한할 수 있는 기술이 필요하다. 이러한 기술중의 하나인 고장안전(fail-safe) 기술의 특성은 시스템의 안전설계 원리 중의 하나로 Fool proof와 Temper proof등과 함께 중요한 요소가 된다. 즉 시스템에서 고장이 발생한 경우에도 그로 인한 피해가 확대되지 않고 단순고장으로 마무리 되도록 하는 설계개념으로 우리 주변에서 흔히 볼 수 있는 과부하 차단기, 퓨즈, 철도 신호기 등이 이와 같은 특성을 만족하기 위하여 사용되고 있다[1].

물리적인 부품에서 발생 가능한 결함을 취급하는 대표적인 기술로는 여분(Redundancy)과 투표기(Voting)를 사용하는 방법이 있으며, 이러한 기술을 적용한 시스템에 대한 신뢰성 평가는 마코브(Markov) 모델링을 근거로 한다.[2,3,4]

설계 오류 문제는 취급하기가 더욱 복잡하며, 현재 사용하고 있는 설계오류 취급 기술에 대해 과학적으로 정당하게 답변할 수 있는 이론이 존재하지 않는다. 단지 설계 단계에서 발생한 오류를 줄이기 위해서 검사(Testing)과정, 설계의 다양화(소프트웨어

* 정회원 : 광운대학교 전자공학부 교수
** 정회원 :
논문접수 : 2001. 5. 30.
심사완료 : 2001. 6. 11.
※ 본 논문은 2000년 광운대학교 교내 연구비 지원에 의한 것임.

결함허용: N-버젼 프로그래밍, Recovery Block 등), 결함회피(Fault-Avoidance: 정형 사양/검증, 자동화 프로그램 합성, 재사용 모듈)를 근거로한 방법적인 것들이 주장되고 있다.

위의 방법들에 따른 문제점으로는 첫째 시험을 수행하는데 있어서 시험기간이 실행 불가능할 정도로 길다는 것이다. 예를 들어 1시간동안의 시스템이 임무를 수행하는데 있어 고장확률이 10^{-9}라는 확률을 측정하기 위해서 114,000년 이상을 시험해야 한다.

이러한 시험 방법을 극복하기 위한 방법으로 설계의 다중화(Diversity)가 주장되고 있으며, 이방법의 기본적인 생각은 같은 명세서(Specification)로 부터 분리된 설계/구현 팀을 이용하여 다중 버전을 만들자는 것이다. 그리고 하나의 버전에서 발생하는 설계오류의 영향을 마스크하기 위해서 임계 투표기(Threshold Voter)를 사용하는 것이다[5,6,7].

이에 반해 VLSI 설계에서 오류 검출 구현을 위해서 논리 연산과 제어선 검사에 알맞은 중복과 부합(Duplication and Matching) 방법을 이용하여 중복된 기능 회로가 병렬로 그 기능을 수행하고 각 회로 출력을 비교하므로써 오류를 검출한다. 이때 두 개의 회로 중 최소한 하나 이상의 회로에서 결함이 발생한다면 오류는 검출되어 서로 다른 출력을 발생하게 한다. 이 방법의 장점은 결점이 어떻게 모듈에 영향을 미치는 가를 정확하게 예측할 필요가 없고, 설계, 설계 검증, 칩을 시험하는데 과정의 복잡도가 증가하지 않는다[8].

중복과 부합 체계에서 임계적인 요소는 두 개의 기능 모듈로부터 출력을 비교하는 회로이다. 이러한 비교기는 비교기 자체의 결함으로 인해 기능 모듈들의 고장(출력 불일치)를 마스크 하게 된다. 따라서 비교기는 정상동작 동안에 비교기 자체의 물리적 결점 발생시 오류 발생 지시와 함께 안전측(Safe Side)으로 동작할 수 있는 고장안전 특성을 가져야 한다.

본 논문은 정형 시스템인 VHDL을 이용하여 고장안전(Fail-Safe) 특성 비교기를 설계하였으며, 설계상의 오류를 검증하기 위해서 정형언어인 VHDL을 이용하여 설계하였다.

2. 결함(Fault) 모델

자체시험 특성 회로를 설계하고 구현하기 위해서 이용되는 특정 기술에서 발생 가능한 물리적 결점을 고려할 필요가 있다. 일반적으로 디지털 회로 설계는 전압, 전류 레벨에서 보다 부울 논리 레벨에서 수행하기 때문에 물리적 결점이 파악되면 이러한 결점이 논리 레벨에서 회로의 동작에 미치는 영향을 결정하는 것이 바람직하다.

이러한 영향을 수학적으로 표현한 것이 결함 모델이다.

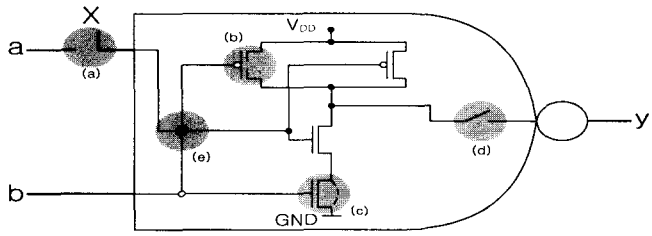
본 장에서는 일반적인 NMOS와 CMOS VLSI회로에 대한 결함 모델을 제시하고 그것들을 이용하여 PLA에 대한 세부적인 결함 모델을 개발한다.

2.1 고착(Stuck-at)결함 모델

가장 간단하고 일반적으로 많이 사용되는 결함 모델은 고착 결함 모델이다. 이 결함은 특정한 신호선, 즉 게이트의 입력이나 출력이 논리적으로 변하지 않고 어떤 논리적인 일정한 값(1 또는 0)을 계속 유지하며, 논리 값이 0이냐, 1이냐에 따라서 고착 0 결함과 고착-1 결함으로 나누어진다. 그림의 (a)부분에서 입력 a는 고착-X 결함이 발생한 상태이다. 따라서 입력 a는 X에 따라서 0이나 1의 값을 유지한다. 입력 a에 대해서 고착 0 결함일 경우에는 $\varphi 1$으로, 고착 1 결함일 경우에는 $\varphi 2$로 두면, NAND게이트의 논리 함수 $F_{\varphi 1}(a,b)=1$과 $F_{\varphi 2}(a,b)=\overline{b}$로 표현할 수 있다. 대부분의 물리적인 결함들은 고착 결함 모델로 표현할 수 있지만, 이 모델로 표현치 못하는 결함들도 있다.

2.2 Stuck-open과 Stuck-on 결함 모델

MOS 트랜지스터에서의 일반적인 결함 모델은 Stuck-Open과 Stuck-On 결함 모델이다. 보통의 Stuck-Open결함과 Stuck-On 결함은 동시에 발생하지만, 이 두 가지의 결함을 검출하는 것과 모델링하는 방법은 서로 다르다. [그림 1]의 NAND 게이트는 각각 두개의 n,p트랜지스터로 이루어진 구조를 가진다. 만약 임의의 트랜지스터에 Stuck-Open 결함이 발생



[그림 1] CMOS NAND 게이트에서의 여러 가지 결함

[Fig. 1] Different Faults in a CMOS NAND Gate

하면 이 트랜지스터는 게이트의 입력 논리값에 상관없이 현재의 값을 유지하게 된다. 따라서, 전원이나 그라운드방향으로 전류의 흐름이 발생하지 않게 되는데, 이러한 경우를 고 임피던스(High Impendence) 상태라고 한다. 이 경우에 출력 y는 이전의 논리 값을 유지하게 된다.

아래 <표 1>에 Stuck-Open 결함이 발생하였을 때의 NAND 게이트의 진리표를 보였다. 표에서의 $F_{\varphi 2}(a,b)$는 결함이 발생하였을 때의 논리값이고, M은 이전 상태의 논리 값을 가지고 있다는 뜻이다. Stuck-Open 결함은 이전 상태의 값을 가지게 때문에 모델링시 시간 t를 고려할 필요가 있다.

<표 1> 결함이 발생하였을 때의 NAND gate의 출력 논리표

<Table 1> Faulty Behavior of the NAND Gate

a	b	F(a,b)	$F_{\varphi 3}(a,b)$
0	0	0	1
0	1	1	1
1	0	1	M
1	1	0	0

따라서, 결함이 발생하지 않았을 때의 논리 함수는 식(2-1), 결함이 발생하였을 때의 논리 함수는 식(2-2)로 표현된다.

$$F(a,b;t)=\overline{a(t)\cdot b(t)} \tag{2-1}$$

$$F_{\varphi_3}(a,b;t)=\overline{a(t)\cdot b(t)}\cdot(\bar{a}(t)+b(t))+F_{\varphi_3}(a,b;t-1)\cdot a(t)\bar{b}(t) \tag{2-2}$$

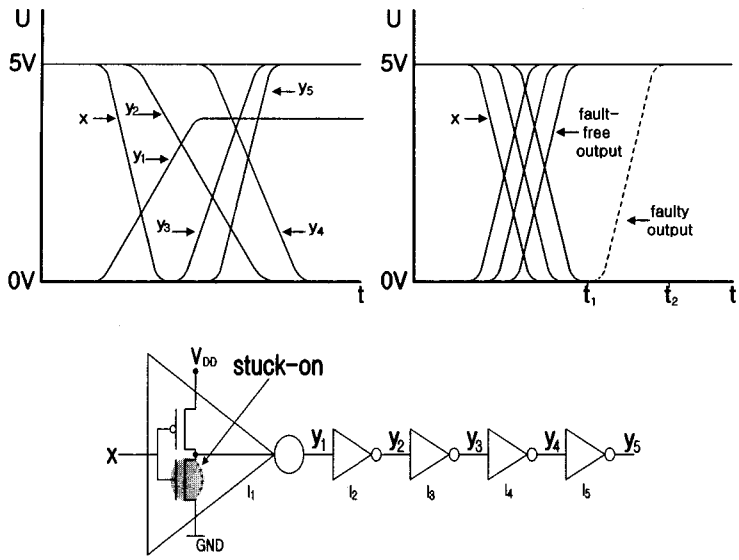
식(2-2)에서 이전 상태를 고려치 않으면, 다음과 같이 표현할 수 있다.

$$F'_{\varphi_3}(a,b)=\bar{a} \tag{2-3}$$

만약 NAND 게이트의 n-트랜지스터에 Stuck-Open 결함이 발생하였다고 하면, 출력 y는 0값을 가지지 못하기 때문에 고착 1 결함이 발생한 경우와 같다고 볼 수 있다. Stuck-On 결함은 [그림 1]의 (c)부분과 같이 트랜지스터에 단락이 발생한 경우이다. 따라서, 트랜지스터를 영구적으로 동작하는 것처럼 만든다. Stuck-On 결함이 발생한 경우를 $\varphi 4$라고 하면, 논리함수는 $F'_{\varphi_4}(a,b)=F(a,b)$이다. Stuck-On 결함은 신호에서의 지연을 가져오는데 이것을 [그림2]에 나타내었다. 그림에서 보듯이 NOT게이트가 5개가 연결되어 있는데, 이중 첫 번째 게이트의 n 트랜지스터에서 Stuck-On결함이 발생하였다고 하자. 결함이 발생하지 않았다고 하면, 입력 x가 0이면 최종 단의 출력 y5는 1을 출력할 것이다. 하지만, Stuck-On 결함으로 인해 출력이 왼쪽의 그래프에서와 같이 정확히 입·출력되는 것이 아니고, 오른쪽의 그래프와 같이 지연을 가지고 입·출력된다. 지연된 신호가 시간 t2에서는 정확한 값으로 인식되지만, 시간 t1에서는 정확한 값을 가지지 못한다.

2.3 Open line과 Bridging 결함 모델

Open line 또는 Open 결함은 회로내부의 도선이 단선되었을 경우이다. ([그림 1]에서의 (d)) nMOS 게이트에서의 Open line 결함은 트랜지스트에서의 Stuck-Open 결함과 같이 고착 X 결함으로 모델링할 수 있다.



[그림 2] Stuck-On 결함에 의한 신호의 지연

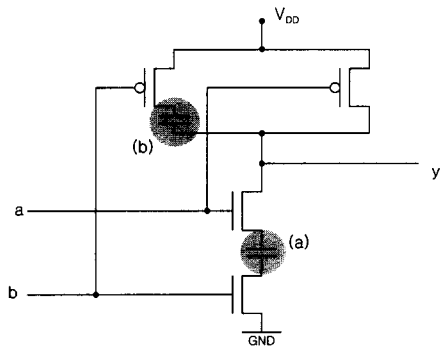
[Fig. 2] Influence of a Stuck-on Fault on the Signal Propagation

트랜지스터에서의 Stuck-Open 결함과 마찬가지로 CMOS 게이트에서의Open Line 결함은 [그림 3]과 같이 표현할 수 있다. a 부분에서 도선이 단선된다면, 출력 y는 논리 값 0을 가지지 못한다. 따라서 출력 y는 고착 1 결함을 가진다고 볼 수 있다. 그러므로, Open Line 결함의 모델링 방법은 고착 결함, Stuck open 결함을 모델링하는 방법과 비슷하다고 할 수 있다.

Bridging결함은 회로 내에서 임의의 도선이 다른 도선과 단락이 된 경우이다. Bridging 결함은 다음의 3가지 경우로 나눌 수 있다.

(a) 게이트내에서의 Bridging 결함
(b) 폐루프(feedback)를 가지지 않는 두 개의 게이트에서의 Bridging 결함
(c) 폐루프(feedback)를 가지는 두 개의 게이트에서의 Bridging 결함

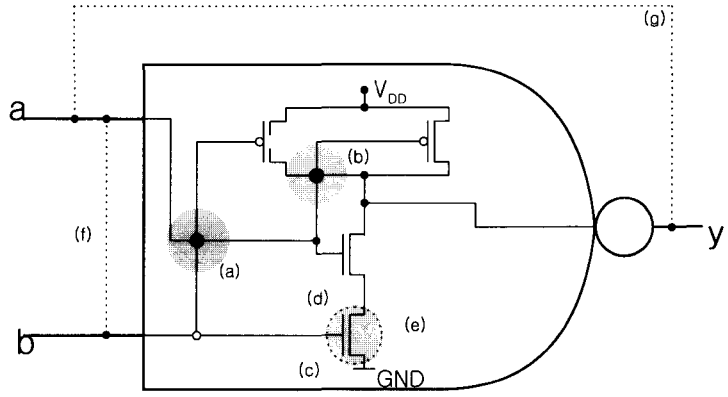
[그림 4]의 (a)부분과 같이 두 입력 사이에 발생한 결함은 게이트 외부에서 입력간에 Bridging 결함이 발생한 (f)부분과 같다고 볼 수 있다. 이 결함은 두 개의 입력이 논리합(Wired-OR)이나 논리곱(Wired-AND)상태로 되어서 NAND 게이트로 입력된다고 볼 수 있다. 이러한 결함을 $\varphi 5$라 두면, 논리곱의 경우는 $F'_{\varphi 5}(a,b)=F(a,b)$로 논리곱의 경우는, $F'_{\varphi 5}(a,b)=\overline{a+b}$로 표현할 수 있다.



[그림 3] NAND게이트에서의 Open line 결함

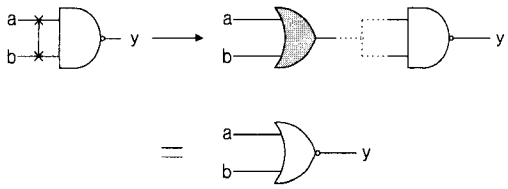
[Fig. 3] Open Line Fault in the NAND Gate

[그림 4]의 (b)경우($\varphi 6$)는 (g)경우와 같은 Feedback Bridging 결함이라 할 수 있다. 이 결함은 논리합이나 논리곱으로 표현할 수 있다. 이러한 결함은 어떤 입력 조합에 대해서도 출력 y가 발진이 일어나도록 만든다.



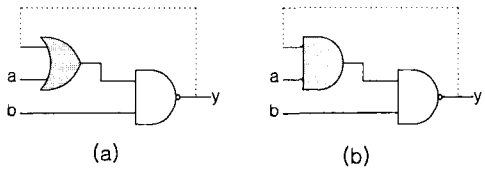
[그림 4] NAND게이트에서의 Bridging 결함

[Fig. 4] Bridging Faults in the NAND Gate



[그림 5] Bridging 결함이 발생하였을 때의 등가 회로

[Fig. 5] Equivalent Circuit when occurred Bridging Fault



[그림 6] 폐루프 Bridging 결함이 발생하였을 때의 등가 회로

[Fig. 6] Equivalent Circuit when occurred Feedback Bridging Fault

나머지 (c), (d), (e)경우를 $\varphi 7, \varphi 8, \varphi 9$라고 두고, <표 2>에 진리표를 나타내었다.

<표 2>에서 u의 의미는 발진 상태라는 의미이고, R의 의미는 내부 트랜지스터의 저항에 따라 출력 y가 변한다는 의미이다.

<표 2> Bridging 결함이 발생하였을 때의 진리표

<Table 2> Truth Table when occuring Bridging faults

a	b		$F_{\varphi 6}^{OR}$	$F_{\varphi 6}^{AND}$	$F_{\varphi 7}^{OR}$	$F_{\varphi 7}^{AND}$	$F_{\varphi 7}$	$F_{\varphi 8}^{OR}$	$F_{\varphi 8}^{AND}$	$F_{\varphi 9}$
0	0	1	1	1	1	1	1	1	1	1
0	1	1	0	1	u	1	1	1	1	1
1	0	1	0	1	1	1	1	u	1	R
1	1	0	0	0	u	u	1	0	0	0

2.4 지연 결함(Delay Fault) 모델

지연 결함(Delay Fault)는 신호가 임의의 회로상의 노드에서 정해진 범위의 시간에 도달하지 않을 경우이다. 이 결함을 모델링하기 위해 가장 많이 쓰이는 모델은 단일 게이트 지연(Single Gate Delay) 결함 모델과 경로 지연(Path Delay) 결함모델이다. 결함 모델을 모델링하기 위해 임의의 회로내의 노드에 신호가 지연을 가지고 입력된다고 하면, 이 노드에서는 현재 이전의 논리값이 유지된다고 볼 수 있다. 따라서, 이러한 지연 결함은 게이트내의 트랜지스터의 Stuck-Open결함과 유사하다고 볼 수 있고, Stuck-Open결함을 지연 결함에서 지연시간이 무한대인 경우라고 볼 수 있다.

3. 고장 안전(Fail-Safe) 이론

3.1 고장 안전(Fail-Safe) 회로

Fail-Safe 회로에 대한 특성은 Mine and Koga에 의해 다음과 같이 처음 (정의) 되었다.

함수 $F(x,h)$에 대한 오동작(Failed Operating)함수 $F(x,h')$가 정상동작 함수의 값과 정확하게 일치하거나 언제나 "0"(또는 "1")을 취할 수 있다면 이때 함수 $F(x,h)$는"0"(또는 "1")에 대해서 fail-safe 하다. 이때 x_0, x_1는 $x_0 \in \{x \mid F(x,h^i) \neq F(x,h^i)\}, x_1 \neq x_0$)

위의 정의를 자체검사 회로(Self-checking Circuit)의 이론에서 사용하는 기호로 사용 하면 $\forall x \in X$, $\forall f \in F$: $G(x,f) \in G(x,0)$또는$G(x,f)=0$(또는 "1")이면 단일 출력회로 G는 입력공간 X, 결함 집합 F, "0" (또는 "1")에 대해서 고장안전 하다로 표현할 수 있다.

따라서 함수 $F(x,h)$가 고장안전하기 위해서는 $F(x,h)$가 단조 증가(감소) 함수(Monotonic Increasing(Decreasing) Function)가 되어야 한다.

그러나 기존의 고장안전을 위한 해결방법은 VLSI로 구현 불가능하기 때문에 본 논문에서는 고장 안전 기술과 자체검사(Self-checking)기술을 결합하여 새로운 방법을 다음과 같이 (정의)한다.

(정의 1) $\forall f \in F$, $\forall x \in X$: G(x,f)=G(x,0) or G(x,f)∈ Os(G의 safe output space)이면 G는 결함 집합 F에 대해서 고장 안전하다. 여기에서 G는 회로의 입력공간, Os는 G의 출력공간이다.

(정의 1)을 만족하는 고장안전 시스템은 결함 검출 능력을 갖지 않는다. 따라서 오류가능성 있는 출력들의 발생을 검출하지 못하고 정상동작 출력 공간에 포함할 수 있는 경우가 발생한다.. 이러한 상황은 중대한 문제점을 야기하기 때문에 결함 검출 메커니즘으로 자체 시험(Self-Testing) 기능을 추가한다.

3.2 TFS (Totally Fail-Safe) 회로

발생 결함에 대해 다음과 같은 전제 조건을 만족하는 결함에 대해서 첫 번째 결함 검출 때까지 모든 오류 가능성 있는 출력은 안전 출력 공간(safe output space)에 속하게 하는 회로가 TFS 회로이다.

1) 결함은 한번에 하나씩 발생한다.
2) 두 개의 결함 발생사이에는 충분한 시간이 흐르고, 회로의 동작모드 동안 회로는 모든 검출 가능한 결함들을 검출할 수 있는 입력 집합에 의해 여기된다.

이러한 TFS 회로중 가장 큰 부류는 SFSC 이다.

3.3 TSC(Totally Self-Checking)와 SFS (Strongly Fail-Safe) 회로

회로 L이 Fault-Secure 하다면 TSC(Totally Self-Checking)이다. 이때 Fault secure의 정의는 다음과 같다.

$$Y(x,f) = Y(x,\theta) \text{ or } Y(x,f) \notin S$$

이러한 TSC와 TFS 회로를 근거로 SFSC 회로를 다음과 같이 정의한다.

i) 회로 G가 TFS(Totally Fail-Safe) 하거나

ii) 결함 집합 F 중 새로운 결함이 발생할 때 회로 G가 고장안전 하다면 다중 결함에 대해서 회로 G는 SFS 회로이다.

3.4 UFS(Ultimate Fail-Safe) 회로

시스템이 정상동작 동안 모든 오류 가능성 있는 출력들은 안전 출력공간에 속해야 한다. 이를 위해서 결함검출 메커니즘에 의해서 발생된 첫 번째 오류 검출지시가 시스템을 안전상태로 Locking 할 수 있는 Fail-safe 메커니즘을 동작 시켜야 한다.

3.4.1 UFS의 구현 방법

오류지시(EI)를 변환하는(Fail-safe 오류지시 신호로 변환함) 변환기를 사용함으로써 실현가능하며, 일반적으로 2-rail 코드를 이용한다.

이러한 사상은 시스템의 전원을 단절하기 위하여 이 신호의 안전값을 사용하고, 안전상태로 시스템을 locking 하는 것이다. 그러나 이러한 안전 값은 시스템이 안전상태에 들어가기 위해서 일정한 시간동안 유지되어야 한다. 이것은 2-rail 오류 지시 신호들이 일정한 시간동안 오류검출 지시 상태를 유지해야 함을 의미하며, Self-checking 시스템의 이론을 근거로 한다.

3.5 오류지시 변환기 (Error Indication Translator)

UFS 목적을 성취하기 위해서 요구되는 오류지시 변환기 특성은 Safe-Disjoint로 다음과 같이 정의된다.

각 코드워드 입력은 비안정(Nonsafe) 출력으로, 비코드워드 입력은 안전(Safe) 출력으로 (오류검출지시 →2 레일 코드 00, 11) 맵하게 하는 회로가 있다면, 그 회로 G는 safe-space disjoint (또는 safe-disjoint) 이다.

$\forall a \in A,\ G(a,\emptyset) \notin Os$

$\forall a \notin A,\ G(a,\emptyset) \in Os$

A는 입력 코드 공간 Os ; 회로 G의 출력 안전 공간

자체검사(Self-Checking) 이론에서 채택한 오류지시 신호는 정확한 동작에 대해서는 10과 01, 오류를 검출했을 때의 신호는 00과 11을 사용 하는 것으로 한다.

따라서 2 레일 코드를 이용한 Safe-disjoint 회로 기능은 다음과 같다.

변환기 입력 코드 공간 → 비안전 출력 공간
변환기 입력 비코드 공간 → 안전 출력 공간

3.6 변환기가 만족해야하는 특성들의 정의

변환기 자체 결함이 비코드워드 입력을 안전 출력으로 맵하는 능력을 상실 할 수 있기 때문에 변환기는 자신의 결함을 검출하기 위한 자체오류검출 메커니즘을 소유해야 하며, 이를 위해서는 다음에 정의하는 특성들을 만족해야 한다.

① Safe-locking : 결함 집합 F에서 변환기가 각 코드워드 입력을 안전출력으로 맵할 수 있다면, 이 변환기는 safe- locking이다.

② TFS(Totally Fail-Safe) : 모델화된 모든 결함에 대해서 변환기가 safe- disjoint하고, 자체시험(self-testing :결함을 검출할 수 있는 최소한 하나의 입력존재) 기능과 safe-locking을 만족하면 모델된 결함 집합에 대해서 변환기는 TFS하다.

③ Strongly Safe-disjoint (SSd) 정의 : 각 모델화된 결함 하에서, 변환기가 safe-disjoint이고 다음 ⓐ, ⓑ, ⓒ 조건 중 하나를 만족하며, 모델화된 결함이 새로이 발생할 때에도 조건 ⓐ, ⓑ, ⓒ 중 최소한 하나를 만족한다면 변환기는 SSd이다.

ⓐ 변환기가 self-testing(회로의 동작모드 중 최소한 한 동작에서 결함을 검출할 수 있는 적용입력이 존재)

ⓑ 변환기가 self -locking

ⓒ 변환기가 비코드 입력을 안전 출력으로 맵

④ SFSd(Strongly Fault-Safe-disjoint) 정의

모델화된 결함하에서 변환기가 Safe-disjoint 이고 다음 조건 ①, ②, ③ 중 하나를 만족하며, 새로 발

생하는 결함에 대해서도 이 조건을 만족할 때 변환기는 SFSd 하다.

① 변환기가 자체시험 특성을 갖고, 비코드 입력을 안전 출력으로 맵 하거나
② 자체시험 특성은 없지만 Safe-locking 할 수 있거나
③ 각 비코드 입력을 안전 출력으로 맵 한다.

본 연구에서 설계하는 인터페이스 회로 중 가장 핵심이되는 신호 변환기는 모델링된 결함에 대해서 위의 정의들을 만족해야 한다.

3.7 Fail-Safe 회로에 대한 필요충분 조건

정 리 : 결함 검출능력을 갖지 않은 시스템 G가 결함집합 F들로 구성된 다중결함 F*에 대해서 고장안전(Fail-safe)하면 시스템 G는 결함집합 F에 대해서 SFS(Strongly Fail-Safe) 하다.

증 명 : 결함 집합 F에 대해서 시스템이 SFS 하다면, F에 포함되는 결함이 발생할 때 정의 3.3 ii)는 성립된다.(이때 정의 i)은 시스템이 결함검출 능력이 없기 때문에 관계없다) 따라서 시스템은 다중 결함에 대해서 고장안전일 것이다. 이것은 F에 포함되는 결함들의 조합에 대해서도 성립되고 결과적으로 G는 F*에 대해서 고장안전이 된다. 만약 시스템이 F*에 대해서 고장안전이고 F에 포함되는 새로운 결함이 발생하면 정의 3.3 ii)는 성립되고 시스템은 결함 집합 F에 대해서 SFS이다.

4. 인터페이스 회로 설계

고장안전 시스템의 각 작동기는 고장안전 신호(정확하거나 또는 안전신호)에 의해서 제어되어야 한다. 자체검사 시스템은 코드화된 신호 그룹을 전달하기 때문에 고장안전 특성을 만족해야 하는 작동기를 구동하는데 알맞지 않다.(각 작동기는 개별적으로 Fail-Safe한 단일 신호에 의해서 제어되어야 하고, 이러한 요구 때문에 VLSI로 고장안전 시스템을 구현하는 것이 가능하지 않다) 그러나 자체검사 회로는 비코드워드에 대해서 결함동작을 나타냄으로써 결함검출 능력을 갖고 있다.(전용검사기가 결함 검출)

반면에 고장안전 회로는 정상행위와 결함행위 양쪽에서 안전과 비안전 상태가 발생하기 때문에 근본적인 결함검출 능력이 없다. 따라서 본 연구에서는 결함 검출능력이 부족한 고장안전 인터페이스 구현 가능성을 분석한다.

4.1 2 레일(Two-rail) 코드 검사기(Checker)

코드 검사기는 on-line 오류 검출기능을 갖는 컴퓨터 시스템에서 가장 중요한 부분이다. 본 장에서는 VLSI 회로로 자체 시험 특성 비교기를 구현하기에 앞서 2 레일 코드 검사기와 비교기 설계에 대해서 논한다. 두 개의 n 비트 벡터 $A=(a_{n-1}, a_{n-2}, \cdots, a_0)$와 $B=(b_{n-1}, b_{n-2}, \cdots, b_0)$가 비교된다고 할 때 $0 \le i \le n-1$인 모든 i에 대해서 $a_i = b_i$'이라면 A, B벡터를 조합한 2n 비트 벡터 $AB=(a_{n-1}, a_{n-2}, \cdots, a_0, b_{n-1}, b_{n-2}, \cdots, b_0)$는 2 레일 코드워드이다. $B'=(b_{n-1}', b_{n-2}', \cdots, b_0')$ 일 때 비트 벡터 AB'를 입력으로 갖는 2-레일 코드 검사기는 벡터 A,B의 비교기와 같다. 따라서 모든 입력 비트가 보수(complement)형과 정상(uncomplement)형으로 이용 가능하다는 가정 하에서 비교기 설계와 2-레일 검사기 설계의 차이점은 없다.

오류검출을 위해 중복과 부합(Duplicate and Matching) 방법을 이용할 때 다음과 같은 가정에서 출발한다.

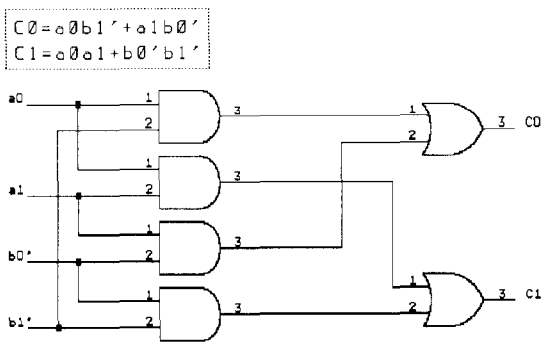
가. 영구 결함과 간헐 결함은 중복된 기능 모듈과 비교기에서 동시에 발생하지 않는다.
나. 기능 모듈 중 하나에서 첫 번째 결함이 발생하면, 그것이 회로 내에서 또는 회로의 상태를 영구적으로 변화시키는 원인이라면, 두 모듈로부터 출력은 불일치하고, 비교기에서 결함이 발생하기 전에 비교기로부터 비코드 출력으로 귀착된다.
다. 비교기나 기능 모듈에서 부가적인 결함이 발생 전에 비교기의 입력이 되는 코드워드의 집합은 비교기의 완전한 자체 시험 특성을 만족

한다. 만약 이기간 동안에 결함이 지속되고 비교기가 자체시험 특성을 만족한다면 비교기의 출력은 기능 모듈 중 하나의 모듈에서의 결함으로 인한 오류성 기능 출력을 유도하기 전에 오류 발생을 표시할 것이다.

이러한 가정을 근거로 2장에서 정의한 모든 단일 결함을 포함한 결함 모델에 대해서 비교기는 자체 시험 특성이 될 것을 필요로 한다.

본 논문에서는 Carter와 Schneider가 제시한 자체 시험 특성 2-레일 검사기를 근거로 비교기를 논한다.

[그림 7]에서 회로는 코드 입력에 대해서 (C1, C0)=(0,1)또는 (1,0) 비코드 입력에 대해서 (C1, C0) =(0, 0)또는 (1, 1)을 갖는 2개의 출력선을 갖는다.



[그림 7] 자체 시험 특성 2-레일 코드 검사기
[Fig 7] A self-testing 2-rail Code Checker

Carter와 Schneider의 검사기는 결함이 없을 때 최소 하나의 코드 입력에 대해서 회로의 모든선은 0이고, 또 하나는 1이다. 만약 하나의 선이 고착 0 이거나 고착 1 이라면 1이나 0로 생각되는 선에 대한 코드 입력은 출력 (0, 0)또는 (1, 1)이 된다.

Wang과 Avizienis에 의해 제안된 검사기는 식 (3-1)의 합의 곱(sum-of-products)으로 나타낼 수 있다.

임의의 정수 k 개의 대해서 Ik는 0과 k-1사이의 k개의 정수 집합이라 놓고, (Ik={0, 1, ···, k-2, k-1}) Q가 집합일 때 |Q|는 Q에서의 집합 원소의 수를 나타낸다.

$$C_0 = \sum_{\{Q|Q\subset In,|Q|\ even\}} \{(\Pi_{\{i|i\in Q\}} a_i)\ \ (\Pi_{\{i|i-Q\}} b_i')\}$$
$$C_1 = \sum_{\{Q|Q\subset In,|Q|\ odd\}} \{(\Pi_{\{i|i\in Q\}} a_i)\ \ (\Pi_{\{i|i-Q\}} b_i')\} \quad (3\text{-}1)$$

4.2 자체 시험 특성 검사기의 최적 설계

Wang과 Avizienis가 설계한 자체 시험 특성 검사기는 n 비트 벡터를 비교하는데 있어서 2^n개의 곱의 합을 필요로 했다. 그러나 코드 입력대 대해서 (0, 1) 또는 (1, 0)인 출력하고 비코드 입력에 대해서는 (1, 1)을 출력하는 비교기를 구현하는 것은 다음 방정식으로 가능하다.

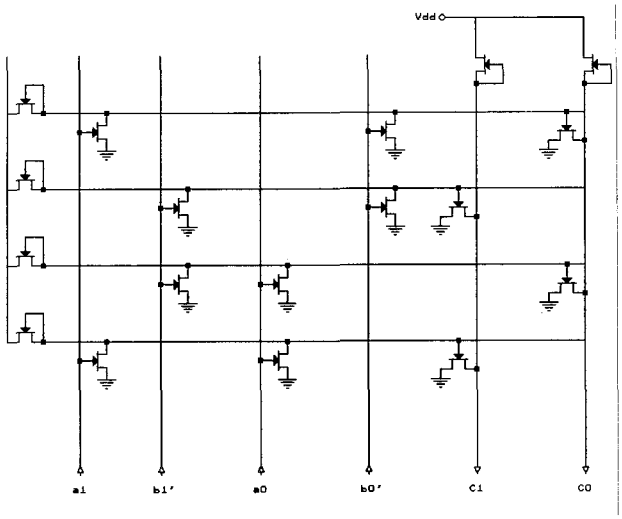
$$C_0 = a_0' + b_0' + \sum_{i=1}^{i=n-1} (a_i b_i' + a_i b_i)$$
$$C_1 = a_0 + b_0 + \sum_{i=1}^{i=n-1} (a_i b_i' + a_i b_i) \quad (3\text{-}2)$$

유사하게 NOR-NOR 형태에서는 식 (3-2)를 근거로 그 함수성을 성취할 수 있다.

$$C_0 = \underset{\{Q|Q\subset In,|Q|\ odd\}}{\text{NOR}} \{NOR(\{a_i|i\in Q\} \cup \{b_i'|j\in(i_n - Q)\})\}$$
$$C_1 = \underset{\{Q|Q\subset In,|Q|\ even\}}{\text{NOR}} \{NOR(\{a_i|i\in Q\} \cup \{b_i'|j\in(i_n - Q)\})\} \quad (3\text{-}3)$$

이 비교기는 입력선과 출력선의 결함에 대해서 단지 4n개의 곱의 합으로써 자체 시험 이 있다. 그러나 곱의 합을 만드는 선상에서의 고착 결함에 대해서는 자체시험 특성이 없다. 2^n 개의 입력 코드워드 중 각 하나에 대해서 그 코드워드에 의해 1로 선택되는 유일한 곱의 항이 존재한다. 벡터 A=(a_{n-1}, a_{n-2}, ···, a_0)의 패리티에 따라서 코드 입력 중 반은 출력 C_0을 선택하고 나머지 반은 출력 C_1을 선택한다.

n=2인 경우에 대해서 방정식 (3-3)을 근거로 구현한 NMOS PLA는 [그림 8]과 같다.



[그림 8] 자체 시험 NMOS 2- 레일 코드 검사기
[Fig 8] A Self-testing NMOS two-rail Code Checker

5. 실험 및 결과

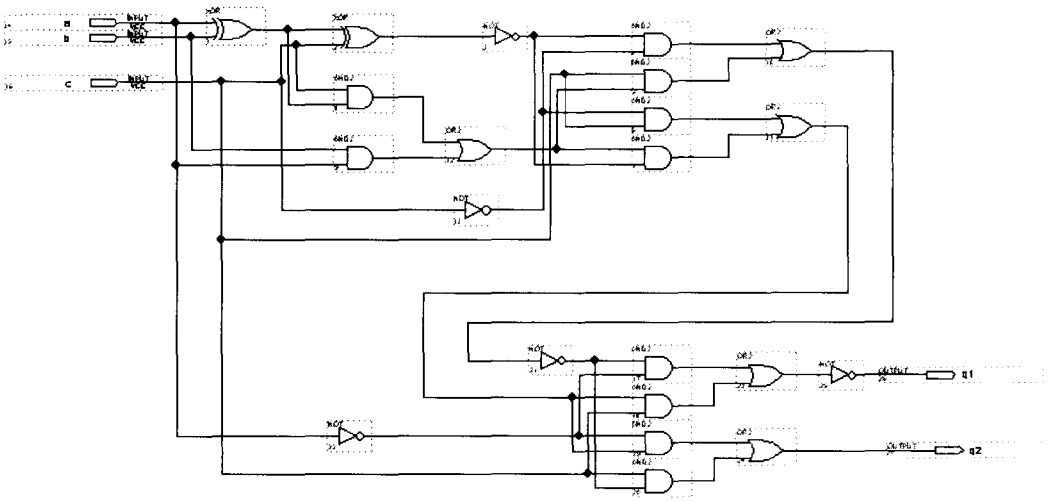
제안한 검사기를 가산기에 적용했을 경우에 대한 실험이다. 실험을 위한 툴은 ALTERA MAX +PLUS II를 이용하였다.

시험 벡터 abc_{000, 010, 101, 111}=:T는 32개의 단일 고착결함 중 30개의 고착결함을 검출 할 수 있다. 이러한 벡터들에 대해서 s=b이고 c+=a를 갖는다.

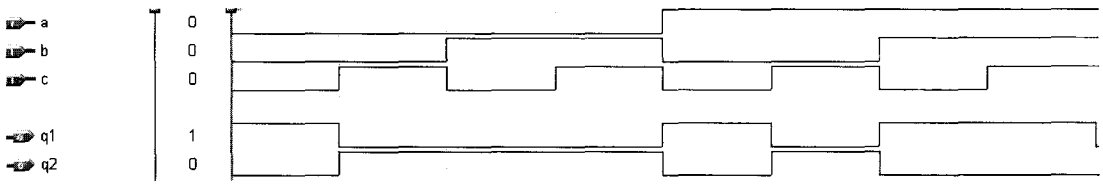
벡터 T에 대해서 모든 나머지 벡터 ac_=1이 성립되는 반면 역시 ac_=0이다. 따라서 a와 c_에 의해서 제어되는 자체검사 특성 검사기에 의해서 가산기의 기능을 검사 할 수 있었다.

벡터 T에 속하지 않는 벡터(a=c-)가 등가 검사기를 디스에이블(Disable)하게 하는 반면 벡터 T(a=c_)는 등가 검사기를 인에이블 한다. 실험을 위해서 가산 기능중 위에서 정의한 입력 벡터에 대해서만 검사하는 것으로 했으며, 이때 회로는 이 입력 벡터에 대해서만 동일한 기능을 하는 것으로 한다. 아래 VHDL 코드는 검사 기능이 있는 가산기[그림 9]에 대한 코드이며, [그림 10]은 무결함 시의 검사기 응답, [그림 11], [그림 12]는 입력 a와 b에서 고착결함이 발생했을 때의 결과를 보여 준다.

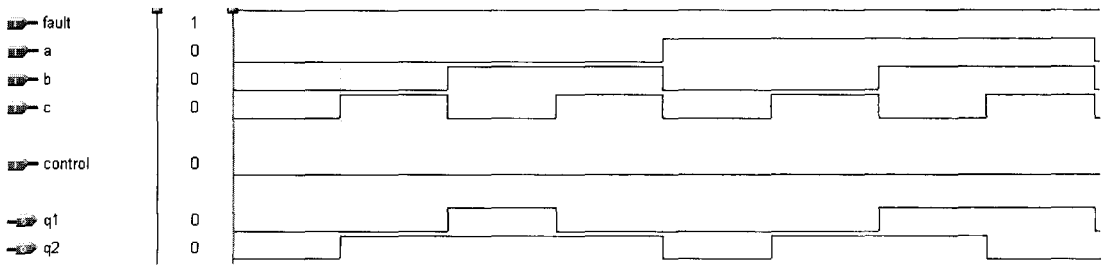
```
LIBRARY ieee;
USE ieee.std_logic_1164.all;

ENTITY checker IS

PORT (  a : in std_logic;
        b : in std_logic;
        c : in std_logic;
        q1 : out std_logic;
        q2 : out std_logic );

END checker;

ARCHITECTURE behave OF checker IS

signal d,e,f,g,h,k,l,n,o,m,p,r,u,s,w,x,y : std_logic ;

BEGIN

d<=a xor b;
e<=d xor c;
f<=d and c ;
g<=a and b ;
h<=f or g;
k<= not c and not e;
l<=b and h ;
n<=not c and b ;
o<= h and not e ;
m <= k or l;
p <= n or o;
r<= not m and not a;
u <= p and c;
s <= r or u;
w<=not a and p;
x<=not m and c;
y<=w or x ;

q1<=not h;
q2<=y;

END behave;
```



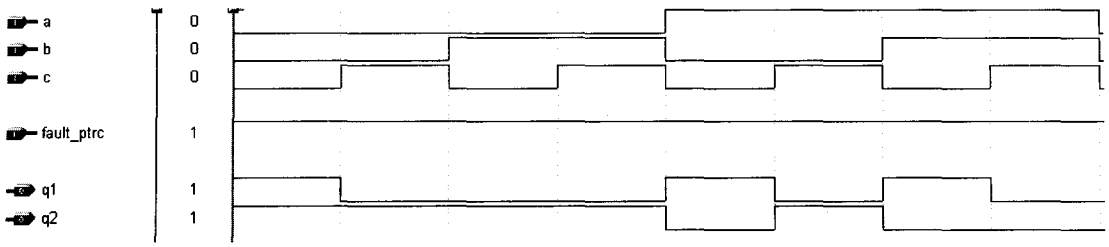
[그림 9] 검사기능이 있는 가산기 회로
[Fig. 9] Adder circuit with checking



[그림 10] 무결함 시 회로 응답
[Fig. 10] Circuit response for the fault free state



[그림 11] 입력 a에서 고착결함 1이 생긴 경우의 결과
[Fig. 11] Result of output in the case of the stuck at 1 fault in input a node



[그림 12] 입력 b에서 고착결함 1이 생긴 경우의 결과

[Fig. 12] Result of output in the case of the stuck at 1 fault in input b node

위의 실험 결과 설계한 자체검사 특성 검사기는 정의한 입력 벡터에 대해 결함 발생을 검출할 수 있었으며, 검사기 자체에 대한 결함 발생시 비코드워드를 발생함으로써 기능 및 회로자체에 대한 검사기능을 유지하며 고장안전 특성을 만족할 수 있음을 보여 주었다.

6. 결론

본 논문에서는 집적회로에서 발생 가능한 결함 형태에 대해 2 레일 코드를 이용한 검사기를 비교기에 적용함으로써 단일 결함에 대한 고장안전(Fail-Safe)특성을 갖는다는 것을 알 수 있었다. 이러한 비교기는 오류검출 뿐 아니라 비교기 자체 결함 발생시 고장안전 특성을 만족함으로써 기능 모듈 레벨에서 중복과 부합 방법을 이용한 오류검출 수행 시 입출력단의 인터페이스로 활용 가능하다. 특히 HDL을 이용한 회로설계에 의해 자체시험 특성의 검증과 해석 과정이 매우 간단했고 수정이 용이했으며 이는 마이크로 프로세서와 같은 VLSI 회로 상에서 결함 영향을 실질적으로 예측할 수 있는 확실한 방법이었음을 알 수 있었다.

※ 참고문헌

[1] Yanghua Min, Yutang Zhou, "Tuotorial : Formal Verificatio of Hardware Design," IEEE Computer Sdcity press, 1990

[2] 양성현, 이기서, "Fault-Tolerance를 위한 시스템 동작 방식에 대한 비교연구," 한국통신학회 논문지, 제17권, 제11호, pp..1297~1289, 1992

[3] 양성현, 이기서, "TMR 시스템의 설계 및 신뢰도 측정 알고리즘," 대한 전기 학회 논문지 제43권, 제3호, pp.515~527, 1994

[4] 김진수, 양성현, "시스템 신뢰도 평가를 위한 동적 결함 트리 알고리즘 연구," 한국통신학회 논문지, vol 24, no 10A, pp.1546-1554, 1999

[5] Rushby J. (1993). Formal Methods and the Certification of Critical Systems. Technical report CSL-93-7, SRI International, Menlo Park, CA. (Also issued as Formal Methods and Digital System Validation, NASA CR 4551)

[6] Butler, Ricky W.,: and Finelli, George B.: "The Infeasibility of Quantifying the Reliability of Life-Critical Real-Time Software. IEEE Transactions on Software Engineering, vol. 19, no. 1, Jan. 1993, pp. 3-12.

[7] Holloway, C. Michael: Holloway, C. Michael: Why Engineers Should Consider Formal Methods, Proceedings of the 16th AIAA/IEEE Digital Avionics Systems Conference, October 26-30, 1997, Irvine CA, Volume 1, pages 1.3-16-1.3-22.

[8] 양성현, 이상훈, "자체시험(Self-Testing) 특성을 갖는 비교기(Comparator) 설계," 한국컴퓨터산업교육학회 논문지, vol2, no2, pp.219-228, 2001

[9] Paray K. Lala, Fault Tolerant and Fault Testable Hardware Design, Prentic Hall International, Inc., London, 1985

[10] Dhiraj K. Pradhan, Fault-Tolerant Computer System Design, Prentic Hall PTR, 1996

[11] 양성현, 이기서, "제어 가능한 자체검사 특성 검사기 설계," 한국통신학회 논문지, 제23권, 제5호, pp1149-1159, 1998

[12] J.P. Khakbaz, "Totally Self-Checking Checker for 1 out of n Code Using Two-Rail Codes," IEEE Tran. on Computer, Vol. C-31, No. 7, pp. 677-681, July, 1982

[13] S. Kundu, S.M. Reddy, "Embeded Totally Self-Checking Checkers: A Practical Design," IEEE Design & Test of Computer, pp.5-12, August, 1990

[14] Wakerly, Error Detecting-Codes, Self-Checking Circuit and Applications, New York, North-Holland, 1978.

[15] Javad Khakhaz, "Totally Self-Checking Checker for 1-out-of-n Code using Two-Rail codes," IEEE Tran. on Computers, Vol.C-31, No.7, pp.677~681, July. 1982.

양 성 현

1958년 2월 1일 생
1983년 2월 광운대학교
전기과 졸업 (공학사)
1987년 8월 광운대학교 대학원
전기과 졸업 (공학석사)
1992년 2월 광운대학교 대학원
전기과 졸업 (공학박사)
1996-1998년 Boston University,
Reserch scientist
1991년 ~ 현재 광운대학교
전자공학부 교수
E-mail : leesh58@mail.gwu.ac.kr

백 순 흠

1981년 3월 육군사관학교 이학사
1985년 12월 국방대학원 석사
1995년 3월 연세대학교 대학원 박사
자이로 스코프의 신뢰성
예측모델에 관한 연구
주관심분야 : 시스템 신뢰성 분야,
체계개발관리