論 文
51D-4-3

결함내성형 이중 마이크로콘트롤러 구조

A Fault-Tolerant Duplex Microcontroller Architecture

金 炳 辰* · 白 承 洙** · 李 寅 煥*** · 林 東 進§
(Byungjin Kim · Seung-Soo Baek · Inhwan Lee · Dong-Jin Lim)

Abstract - This paper presents a fault-tolerant duplex architecture to build a high-reliability microcontroller using commercial VLSI processors. The architecture supports fail-silence under all single-failure situations and facilitates recovery from transient failures. The paper implements the duplex architecture using two Motorola MC68360 processors and evaluates its fault tolerance in a real application environment.

Key Words : Microcontroller, Fault Tolerance, Duplex, Lock-Step

1. 서 론

산업용 분산제어 시스템, 항공기 시스템, 의료기기 등의 다양한 분야에서 마이크로콘트롤러가 널리 사용되고 있다. 그리고 이러한 시스템에 대한 우리의 의존도가 높아짐에 따라 이들에 대한 신뢰도 요구조건 또한 엄격해 지고 있으며, 따라서 높은 신뢰도를 갖는 마이크로콘트롤러가 요구된다. 이러한 요구를 수용하기 위하여 인텔은 1980년경 당시의 기술을 종합하여 인텔 432 결함내성형 시스템을 설계하였으나, 그 설계상의 복잡도로 인해 상업적인 성공을 거두지 못하였다 [1]. 반면에 자체적인 결함내성을 갖지 않은 다수의 콘트롤러를 사용하여 하나의 결함내성형 콘트롤러를 실현하고자 하는 n-Plex 형태의 설계는 매우 성공적으로 사용되어 왔다. 구체적으로 이중구조, 삼중구조 및 기타 다중 구조를 갖는 내장형 시스템이 항공기와 우주탐사선 제어에 사용될 뿐 아니라 [1-4], 범용의 상용 컴퓨터에도 다중 구조가 사용되고 있다 [1,5].

지금까지 결함내성형 시스템의 개발은 연구 개발의 난이도와 특성상 주로 대형, 고가의 시스템을 대상으로 각각의 응용 환경에 따라 개별적으로 이루어져 왔다. 그리고 이러한 결함내성형 구조에 대한 개념적인 내용은 공개되었으나, 구체적인 설계 방법에 관한 기술적인 사항은 대부분 문헌에 공개되지 않고 있다. 이러한 이유로 인해, 그리고 실제 결함내성형 시스템의 설계가 전문적인 지식과 많은 경험을 필요로 하기 때문에, 다양한 응용 분야에서 필요로 하는 높은 신뢰도를 갖는 결함내성형 마이크로콘트롤러의 설계에서는 흔히 많은 시행착오와 실패를 경험할 수밖에 없다. 본 논문에서는 범용 VLSI 프로세서를 이용하여 결함내성형 마이크로콘트롤러를 실현하기 위한 일반적인 이중 구조와 구체적인 설계 방법을 제안함으로써 다양한 응용 분야에서 고신뢰도 마이크로콘트롤러 응용 시스템의 구축이 용이하도록 한다.

구체적으로 본 논문에서는 결함내성을 갖지 않은 두 개의 상용 VLSI 프로세서를 이용하여 고장정지 및 일시적인 고장으로부터의 회복기능을 갖는 하나의 마이크로콘트롤러를 실현하기 위한 이중 구조와 그 구체적인 설계 방법을 제안한다. 그리고 제안한 구조를 모토롤라 MC68360 마이크로프로세서를 이용하여 구현하고 실제 응용 시스템에 적용하여 그 기능을 검증한다. 다음절에서는 우선 이중 마이크로콘트롤러 구조의 개념과 전체적인 동작을 정립한다. 그리고 3절, 4절, 5절 및 6절에서는 제안한 구조를 실현하기 위한 핵심적인 설계 요소 즉, 두 프로세서의 동기화, 고장의 발견, 고장의 처리 및 클럭의 결함내성을 각각 다룬다. 마지막으로 7절에서는 제안한 구조의 구현 및 검증을 보인다.

2. 이중 마이크로콘트롤러 구조

그림 1은 제안한 이중 구조의 개념을 나타낸다. 이 구조에서는 두 프로세서가 같은 작업을 수행하고, 작업의 결과를 외부로 출력하기에 앞서 서로의 출력을 비교한다. 따라서 어느 하나의 프로세서가 고장으로 인해 잘못된 결과를 출력하는 경우 비교기가 이를 발견할 수 있다. 본 논문에서는 이러한 상황을 고장이 발견되었다고 한다. 일단 고장이 발견되면 스위치 회로는 즉시 외부로의 출력을 차단한다. 그리고 각 프로세서는 자체 진단을 통해 스스로의 상태를 판단하여, 영구적인 고장일 경우에는 자체적으로 동작을 정지하고 그렇지 않으면 재시작 과정을 통해 정상 동작으로 복귀한다. 따라서 이러한 이중 구조는 하나의 프로세서에 고장이 발생하더라도 전체적으로 고장정지(fail-silence) 특성을 제공하며 또한 일

* 非 會 員 : 三星電子 DS總括 SystemLSI 硏究員 · 工碩
** 非 會 員 : 三星電子 DS總括 AMLCD 先任硏究員 · 工碩
*** 正 會 員 : 漢陽大學 電子電氣컴퓨터工學部 助敎授 · 工博
§ 正 會 員 : 漢陽大學 電子컴퓨터工學部 副敎授 · 工博
接受日字 : 2001年 4月 27日
最終完了 : 2002年 2月 18日

시적인 고장(transient failure)으로부터의 회복을 가능하게 한다.

이러한 이중 구조에 대한 개념적인 원리는 크게 복잡하지 않으며 또한 문헌에서 다루어졌다. 그러나 지난 30여년간의 이 분야의 연구 개발의 경험은 이러한 결함내성형 구조를 실제로 실현하는 것이 대단히 어려운 일이라는 것을 잘 보여준다. 따라서 결함내성형 마이크로콘트롤러 연구의 핵심은 개념적인 구조의 제시에 있는 것이 아니라, 주어진 결함내성을 실현하기 위한 구체적인 설계 방법을 제시할 수 있는가에 있다. 본 논문에서는 결함내성을 갖지 않은 두 개의 상용 VLSI 프로세서를 이용하여 앞서 기술한 바와 같은 결함내성형 마이크로콘트롤러를 실현하기 위한 이중 구조와 그 구체적인 설계 방법을 제안한다.

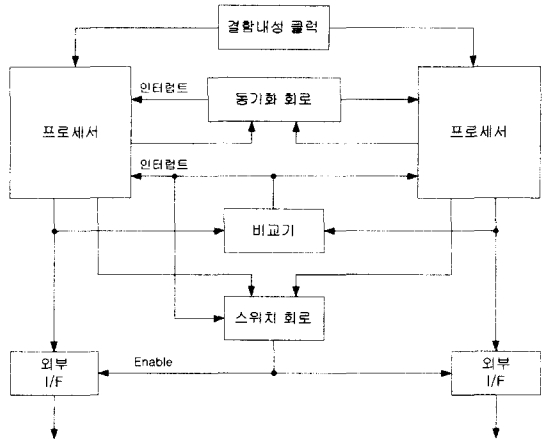


그림 1 이중 마이크로콘트롤러 구조
Fig. 1 Dual microcontroller architecture

그림 1과 같은 이중 구조는 두 프로세서가 시간적으로 서로 독립되어 동작하는 형태(loose coupling)와 클럭 사이클 단위로 일치되어 락스텝(lock-step)으로 동작하는 형태(tight coupling)로 구분할 수 있다. 전자의 구조에서는 두 프로세서가 각각 다른 클럭을 사용하므로 클럭의 결함내성을 위한 별도의 노력이 필요하지 않으나, 비교에 앞서 두 프로세서의 동작을 시간적으로 일치시켜야 하므로 비교기 부분이 상당히 복잡해진다. 따라서 본 연구에서는 자체적인 결함내성을 갖는 공통의 클럭을 사용하는 락스텝 구조를 선택한다. 그러고 프로세서 초기화 과정에서 동기화 회로가 두 프로세서의 동작을 클럭 사이클 단위로 일치시킨다. 다음절부터는 그림 1의 이중 구조를 구성하는 각 모듈의 설계에 있어서의 핵심적인 이슈와 구체적인 설계 방법을 제시한다.

3. 두 프로세서의 동기화

이중 구조에서 공통의 클럭을 사용하더라도 두 프로세서가 반드시 같은 사이클에 같은 동작을 수행하지는 않는다. 그 이유는 전원이 처음 인가될 때, 두 프로세서의 외부 리셋 회로를 구성하는 아날로그 소자의 값의 차이로 인해, 두 프로세서에 리셋 신호가 인가되는 시간이 서로 다를 수 있기 때문이다. 이 외에도 비교기에서 고장이 발견된 후 자체 진단을 통해 영구적인 고장이 없음을 확인하고 재시작 과정을 밟을

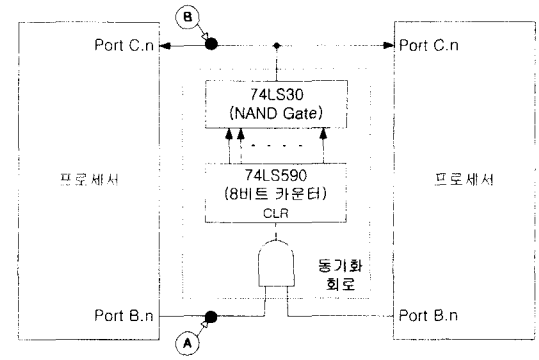


그림 2 동기화 회로
Fig. 2 Synchronization circuit

경우, 이미 두 프로세서의 동작에 시간적인 차이가 발생하여 정상 동작으로 복귀하더라도 두 프로세서는 락스텝으로 동작하지 않을 수 있다. 따라서 이러한 경우에 두 프로세서의 락스텝 동작을 보장하기 위한 추가적인 동기화가 필요하다.

이러한 동기화를 위해 본 구조에서는 두 프로세서가 동작을 시작할 준비가 완료되면 일단 정지 상태(모토롤라 계열 CPU의 STOP 모드 [6], 인텔 계열 CPU의 IDLE 모드 [7])로 들어가는 명령어를 실행하도록 한다. 그리고 그 후에 외부

표 1 동기화 회로의 고장에 대한 대응
Table 1 Tolerating defects in synchronization circuit

고장 위치	고장 형태	마이크로콘트롤러의 동작
동기화 회로의 입력 (그림 2의 Ⓐ)	영구적으로 "0"으로 고착	프로세서 초기화 과정의 자체진단에서 고착을 발견함; 동작 중에 고착이 발생하면 다른 정상적인 동기화 회로만 인터럽트를 발생하므로 이 후의 자체진단에서 이 고장을 발견함 (고장정지)
	영구적으로 "1"로 고착	프로세서 초기화 과정의 자체진단에서 고착을 발견함; 동작 중에 고착이 발생하면 동기화 인터럽트 이 후에 정상 값 "0"으로 환원되지 않아 이 고장 발견됨 (고장정지)
	일시적으로 "0" 또는 "1"로 변화	동기화 과정 이전에는 동작에 영향을 주지 않고, 동기화 과정 중에는 카운터가 일시적으로 클리어 되었다가 다시 카운팅을 시작하여 정상적으로 동기화 인터럽트 신호를 발생함 (정상 동작)
동기화 회로의 출력 (그림 2의 Ⓑ)	영구적으로 "0" 또는 "1"로 고착	프로세서 초기화 과정의 자체진단에서 고착을 발견함; 동작 중에 고착이 발생하면 다른 정상적인 동기화 회로만 인터럽트를 발생하므로 이 후의 자체진단에서 이 고장을 발견함 (고장정지)
	일시적으로 "0" 또는 "1"로 변화	최악의 경우, 동기화가 이루어지지 않은 채 동작하나 비교기를 통해 바로 고장을 발견하고 재시작함 (정상동작으로 복귀)

인터럽트를 두 프로세서에 동시에 발생시켜 이들이 같은 사이클에 동작을 시작하도록 한다. 그림 2는 하나의 카운터와 NAND 게이트로 구성된 동기화 회로이다. 동기화 과정에서 각 프로세서는 병렬 포트 C.n의 동기화 인터럽트를 활성화(enable)시킨다. 그리고 포트 B.n으로 "1"을 출력하여 동기화 회로를 활성화시키고, STOP 명령어를 실행하여 정지상태에 들어간다. 이때 두 프로세서가 완전히 STOP 모드로 들어가기 전에 동기화 인터럽트가 발생되지 않도록 카운터를 사용하여 일정 시간이 지난 후에 포트 C.n에 인터럽트 신호가 발생되도록 한다. 인터럽트가 발생하면 두 프로세서는 동시에 STOP 모드에서 빠져나와 동기화 인터럽트를 마스킹 한 후, 동기화 회로의 카운터를 정지시킨다.

여기서 중요한 것은 동기화 회로 자체의 고장 가능성을 고려해야 한다는 것이다. 본 연구에서는 동기화 회로 또한 이중으로 구성하고(그림 8), 이들의 동작을 비교함으로써 동기화 회로상의 하나의 입력 또는 출력 신호가 영구적으로 "0" 또는 "1"로 고착되는 고장(permanent stuck-at fault)과 일시적으로 "0" 또는 "1"로 변하는 고장에 대한 내성을 제공한다. 표 1은 동기화 회로에서 하나의 입력 또는 출력 신호에 고장이 발생할 경우, 전체 마이크로콘트롤러가 어떻게 고장정지 또는 일시적 고장으로부터의 회복을 제공하는지 보인다. 예를 들면, 하나의 동기화 회로의 출력이 "0"으로 영구적으로 고착된 경우, 이 동기화 회로는 (에지 트리거) 인터럽트 신호를 발생하지 못한다. 따라서 다른 정상적인 동기화 회로의 인터럽트에 의해 동기화 되는 과정에서 두 프로세서가 인터럽트 대기 비트를 검사할 때 하나의 동기화 회로에서만 인터럽트 신호가 발생한 것을 발견하게 된다. 그러면 두 프로세서는 자체 진단으로 들어가서 이 고장을 발견하고 모두 동작을 정지하여 고장정지 특성을 만족시킨다.

4. 고장의 발견

본 절에서는 고장을 발견하기 위한 수단으로서의 비교기와 고장을 발견한 경우 외부로의 출력을 차단하기 위한 스위치 회로를 다룬다.

4.1 두 출력의 비교

비교기는 락스텝으로 동작하는 두 프로세서의 출력을 비교하고, 두 출력이 서로 다를 경우 이를 다른 모듈에 알려 준다. 일반적인 비교기로는 투레일 체커(two-rail checker)를 생각할 수 있고 [8], 투레일 체커를 사용하면 비교기 내부의 고장을 자체적으로 발견할 수 있는 이점이 있으나, 투레일 체커는 정상 동작 중에 "01" 또는 "10"의 값을 임의로 출력하기 때문에 이를 프로세서의 인터럽트 신호로 사용하기 위해서는 자체적인 결함내성을 갖는 로직 회로가 추가로 필요하다. 본 구조에서는 평범한 XOR 형태의 비교기를 사용하고, 비교기 또한 이중으로 구성하여 비교기 자체의 고장에 대비한다.

비교기의 설계에서는 두 프로세서의 출력 신호의 타이밍이 완벽하게 일치하지 않음을 고려하여야 한다. 이러한 미세한 타이밍의 차이로 인해 두 프로세서가 모두 정상적으로 동작하더라도 비교기의 출력에 글리치 형태의 잘못된 일시적인 고장 신호가 발생할 수 있다. 그림 3은 XNOR 게이트와 글리치를 제거하는 F/F으로 구성된 비교기를 보인다. (그림 3은 프로세서 출력의 한 비트에 대한 비교기이다; 병렬 버스의 경우 이러한 비교기 여러 개를 병렬로 사용할 수 있다.) 이러한 구조에서는 비교기 출력의 글리치가 F/F의 출력에 반영되지 않는다.

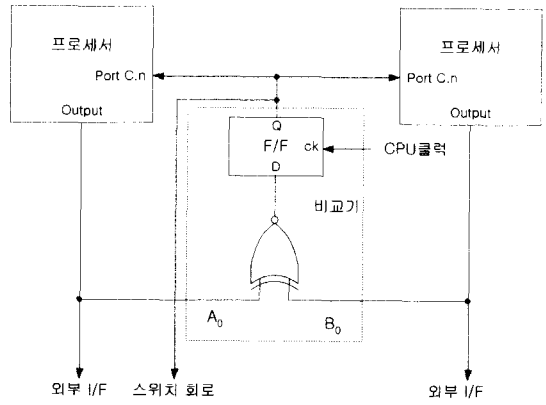


그림 3 비교기
Fig. 3 Comparator

그림 3의 비교기는 두 프로세서의 출력이 같으면 "1"을 출력한다. 그러나 두 프로세서의 출력이 다르면 비교기는 "1"에서 "0"으로 변하는 고장 신호를 출력하여 두 프로세서에 인

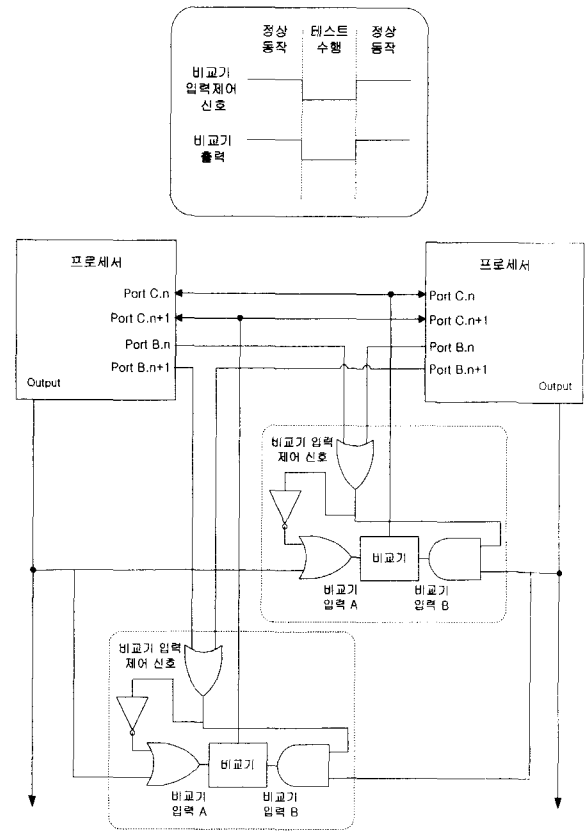


그림 4 비교기 자체의 고장에 대한 대응
Fig. 4 Tolerating defects in comparator

터럽트를 발생시킨다. 이러한 인터럽트는 프로세서가 출력을 외부로 내보내는 동안만 활성화된다. 비교기가 고장을 발견하면 이는 또한 스위치 회로에 전달되고, 스위치 회로는 즉시 외부로의 출력을 차단한다. 따라서 이러한 이중 구조는 두 프로세서에 같은 형태의 고장이 발생하지 않는 한 전체적으로 올바르게 동작한다. 비교기 자체에도 고장이 발생할 수 있으므로, 비교기 역시 앞의 동기화 회로와 같이 이중으로 구성하여 비교기의 고장에 대한 내성을 부가한다 (그림 4).

4.2 비교기 출력의 "1"로의 고착

비교기의 출력이 두 프로세서의 정상적인 동작을 나타내는 "1"로 영구적으로 고착되면 재미있는 문제가 발생한다. 즉, 하나의 비교기의 출력이 "1"로 고착된 이후에 추가적인 고장이 발생하지 않으면 이중 구조의 마이크로콘트롤러는 정상적으로 동작할 수 있다. 그러나 이러한 고착을 방치할 경우 만일 다른 비교기에도 유사한 고장이 발생하면, 두 프로세서의 출력이 다르더라도 두 비교기 중 어느 것도 고장신호를 발생하지 않아 잘못된 결과가 외부로 전달된다. 따라서 비교기의 출력이 "1"로 고착될 경우 이를 빨리 발견할 수 있어야 한다.

이러한 고착을 발견하기 위해서는 인위적으로 비교기의 출력이 "0"이 되는 조건을 만들어 주어야 한다. 이를 위해 우선 소프트웨어적인 방법으로서 두 프로세서에 서로 다른 출력 패턴을 저장하고, 정상 동작 중에 이를 출력함으로써 비교기에서 고장신호가 발생하도록 할 수 있다. 그러나 그럴 경우 두 프로세서의 실행 코드가 달라지는 문제가 있으므로, 본 연구에서는 비교기에 서로 다른 입력을 인가할 수 있는 하드웨어를 추가하여 비교기의 출력이 "1"로 고착되었는지를 검사한다.

그림 4는 이러한 기능을 포함한 비교기를 보인다. 두 프로세서는 평상시 포트 B.n을 "1"로 유지하고 이때 각 프로세서의 출력은 그대로 비교기에 전달된다. 비교기의 출력이 "1"로 고착되었는지를 검사할 때, 두 프로세서는 비교기에 의한 인터럽트를 마스킹 한 후 포트 B.n으로 "0"을 출력한다. 그러면 비교기 입력 제어 신호가 "0"이 되어 비교기 입력 전단의 OR 게이트로부터는 "1"이, 그리고 AND 게이트로부터는 "0"이 각각 비교기에 입력되어 비교기(XNOR)의 출력은 "0"이 되어야 정상이다. 따라서 이때 포트 C.n의 값을 읽음으로써, 비교기의 출력이 "1"로 고착되었는지를 검사할 수 있다. 검사가 끝난 후 프로세서는 인터럽트 대기 비트를 지우고, 포트 B.n으로 "1"을 출력한 후 비교기 인터럽트를 활성화한다. 이러한 검사는 프로세서가 외부로 데이터를 출력하기 전에 수행할 수 있다.

4.3 외부로의 출력의 제어

비교기에서 고장을 발견하면 스위치 회로는 즉시 외부로의 출력을 차단하여 잘못된 결과가 외부로 전달되는 것을 방지한다. 이러한 외부 출력의 차단은 자체진단 결과 두 프로세서가 모두 정상이라고 판단되지 않는 한 계속된다.

그림 5는 클럭의 하강 에지에서 트리거 되는 F/F을 사용한 스위치 회로를 나타낸다. 이중 비교기의 두 출력은 AND 게이트를 통해 F/F의 리셋 단에 연결된다. 여기서 어느 하나

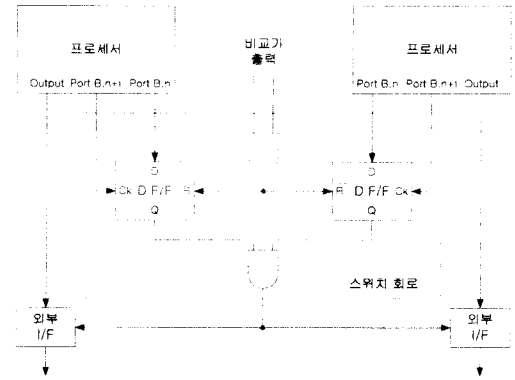


그림 5 스위치 회로
Fig. 5 Switch circuit

의 비교기라도 고장을 발견하면 F/F이 리셋되고 따라서 스위치 회로의 출력은 "0"으로 바뀌어 외부로의 출력이 차단된다. 자체진단 결과 내부에 영구적인 고장이 없다고 판단되면 두 프로세서는 F/F의 데이터 입력(포트 B.n)으로 "1"을 출력하고, F/F의 클럭 입력(포트 B.n+1)으로 "0"과 "1"을 교대로 출력함으로써, F/F의 출력을 "1"로 설정하여 외부로의 출력을 재개한다.

스위치 회로 역시 이중으로 구성하여 스위치 회로 자체의 고장에 대한 내성을 부여한다 (그림 8). 스위치 회로의 출력이 두 프로세서의 정상적인 동작을 나타내는 값 즉 "1"로 영구적으로 고착되는 경우 여기서도 문제가 되며, 이에 대한 검사는 비교기 출력이 "1"로 고착되었는지를 검사할 때 같이 수행한다. 즉 비교기의 출력에 인위적으로 고장 신호를 발생시킬 때 스위치 회로의 출력이 차단 상태로 바뀌는가를 확인함으로써, 스위치 회로의 출력이 "1"로 고착되었는지를 판단한다.

5. 고장 진단 및 회복

이중 구조의 마이크로콘트롤러는 비교기에서 고장이 발견되면 내부의 고장 유무를 판단하기 위한 고장 진단을 수행한다. 이 과정에서는 프로세서, 프로그램/데이터 메모리, 그리고 모든 외부 회로(비교기, 동기화 회로, 스위치 회로)의 고장 여부를 검사한다. 이러한 자체진단에서 영구적인 고장이 발견되면 마이크로콘트롤러는 즉시 동작을 정지한다. 이러한 고장정지 특성을 갖는 마이크로콘트롤러는 높은 신뢰도를 요구하는 응용 시스템을 구성하는데 필수적인 요소이다. 자체진단에서 영구적인 고장이 확인되지 않을 경우 마이크로콘트롤러는 재시작 과정을 거쳐 정상 동작으로 복귀한다. 실제 디지털 시스템에서 일어나는 고장의 대부분은 일시적인 결함에 의한 것으로 알려져 있으므로 이러한 일시적 고장으로부터의 회복 능력은 마이크로콘트롤러의 신뢰도를 크게 향상시킬 수 있다 [9].

여기서 유의할 것은 상용 VLSI 프로세서는 그 복잡성에 비해 상당히 제한된 자체 점검만을 지원하므로 이러한 자체진단에서 모든 고장을 완벽하게 진단할 수는 없다는 것이다. 즉 실제로 마이크로콘트롤러에 영구적인 결함이 있더라도 이

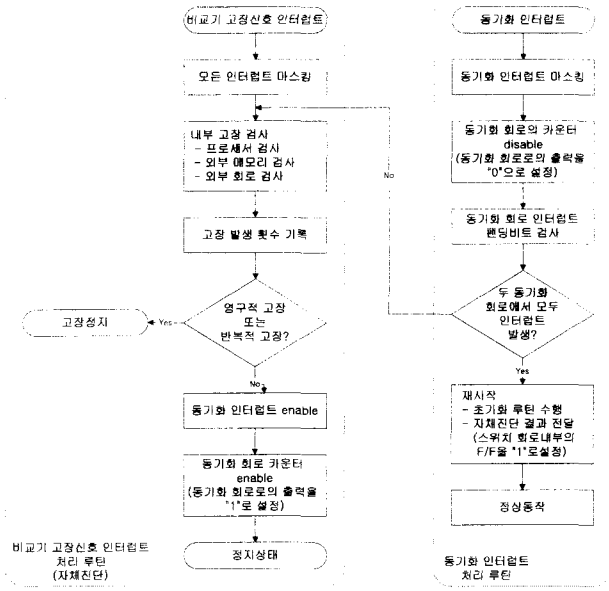


그림 6 자체진단과 재시작 과정

Fig. 6 Self diagnosis and restart

결함이 자체진단에서 발견되지 않고 동작 중에 다시 고장을 일으킬 수 있다. 따라서 제안한 구조에서는 자체진단에서 정상으로 판단하였더라도 이 후 반복적으로 고장이 발견되는 경우, 자체진단이 가능하지 않은 영구적인 고장이 있는 것으로 판단하여 동작을 정지한다.

구체적으로, 비교기의 고장 발견에 의한 자체 진단 과정에서 각 프로세서는 내부의 "고장카운터"의 값을 1만큼 증가시킨다. 그리고 영구적인 고장의 발견되지 않아 동작을 재개한 경우, 매번 출력이 성공적으로 이루어질 때마다 "출력카운터"의 값을 1씩 증가시킨다. 만약 동작을 재개한 후, 100번의 출력이 성공적으로 이루어질 때까지 추가로 고장이 발견되지 않으면, 두 카운터의 값은 클리어 되고, 과거에 발견된 고장은 일시적인 결함에 의한 것으로 간주된다. 그러나 100번의 출력이 성공적으로 이루어지기 전에 추가로 고장이 발견되면, 자체 진단 과정에서 고장카운터의 값이 다시 1만큼 증가된다. 만약 고장카운터의 값이 4가 되면, 마이크로콘트롤러는 내부에 자체 진단이 가능하지 않은 영구적인 고장이 있는 것으로 판단하고 스스로 동작을 정지한다. 그림 6은 자체진단부터 고장 회복까지의 과정을 나타낸다.

6. 클럭의 결함내성

두 프로세서는 공통 클럭을 사용하므로 이 클럭에 고장이 발생하면 두 프로세서 모두 잘못된 동작을 수행할 수 있다. 따라서 클럭의 신뢰도를 높이는 것이 매우 중요하다. 이와 관련하여 스스로가 정상인지 아닌지를 판단할 수 있는 자체 점검형 클럭(self-checking clock)의 구현 방법이 제시되었으나 [10], 이러한 클럭을 여러 개 갖추고, 현재 사용 중인 클럭에 고장이 발생하면 다른 정상적인 클럭으로 전환하고자 하는 경우, 클럭을 전환하는 과정에서 프로세서를 연속적으로 동작시키기 어렵다. 따라서 제안한 구조에서는 결함내성을 갖지 않은 여러 개의 오실레이터를 사용하고 이들의 동작을 비교하여 전체적으로 하나의 결함내성형 클럭을 구현한다.

일반적인 TMR (tripple modular redundancy) 구조에서는 세 개의 동작 모듈과 다수결 보터(majority voter)를 사용하여 하나의 동작 모듈에 고장이 발생하더라도 전체적으로 정상적인 동작을 보장한다. 그러나 이러한 TMR 구조를 그대로 클럭의 설계에 적용할 경우 각 오실레이터의 출력은 동기가 되어 있지 않기 때문에 다수결 보팅이 그 의미를 상실한다. 따라서 네 개의 오실레이터를 사용하고, 이들의 출력에서 다수 값이 존재 할 경우에는 그 값을 선택하고 그렇지 않은 경우에는 과거 값을 선택하는 소위 히스테리시스 보터를 사용한 결함내성형 클럭 구조가 제안되었다 [11,12]. 본 연구에서는 이 방법의 구현에 초점을 맞춘다.

이러한 결함내성형 클럭은 네 개의 PLL(phase-locked loop)과 네 개의 히스테리시스 보터로 구성한다 (그림 7). 하나의 PLL은 VCO (voltage controlled oscillator), 위상 검출기, RC 저역통과 필터로 구성된다. 각 히스테리시스 보터는 네 개의 PLL의 출력을 이용하여 공통 클럭 신호를 만든다. 각 PLL은 위상 검출기를 통해 히스테리시스 보터의 출력과 자체 VCO 출력간의 위상 차를 검출하고, RC 저역통과 필터

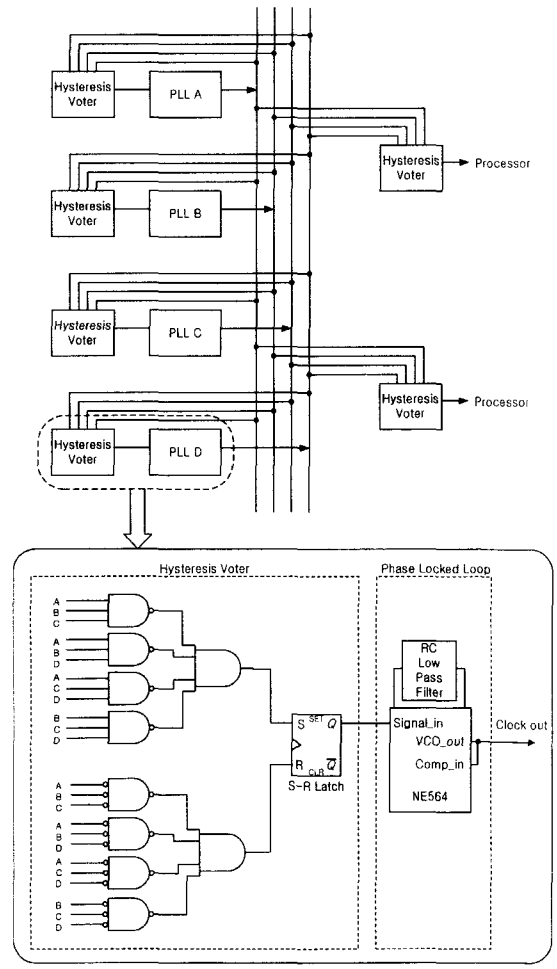


그림 7 결함내성형 클럭

Fig. 7 Fault-tolerant clock

를 사용하여 VCO의 출력 주파수를 조정하는 페이즈 락킹을 수행함으로써, 네 개의 PLL 출력의 주파수와 위상을 일치시킨다. 따라서 고장이 없을 경우 모든 PLL은 동일한 주파수와 위상으로 동작을 하고, 만일 하나의 PLL에서 고장이 발생하더라도 히스테리시스 보터에 의해 이 고장이 마스킹 되어 나머지 세 개의 오실레이터는 일치된 주파수와 위상으로 동작한다. 또한 각 프로세서는 네 개의 PLL의 출력을 보팅하여 클럭으로 사용한다.

그림 7의 클럭에서는 다수의 로직 게이트로 구성된 히스테리시스 보터의 신호 전달 지연으로 인해 PLL로 입력되는 클럭 신호에 왜곡이 발생하여 페이즈 락킹이 이루어지지 않을 수 있다. 따라서 고속 로직을 사용하여 보터 내부의 전달지연을 최소화하는 것이 중요하다. 본 연구에서 구현한 결함내성형 클럭은 65%의 duty ratio와 25MHz의 출력주파수로 동작한다. 이 클럭에서 하나의 보터 및 PLL 모듈에 고장이 발생하여 그 출력이 "0" 또는 "1"로 고착되거나 그 출력 주파수(4~31MHz) 및 duty ratio가 정상 범위를 크게 벗어나도 전체적으로 정상적인 클럭을 계속 공급하는 것을 인위적인 고장 주입 실험(fault injection experiment)을 통해 확인하였다.

7. 구현 및 검증

본 연구에서는 모토롤라의 MC68360 프로세서와 범용 로직 IC를 사용하여 제안한 이중 구조를 갖는 마이크로콘트롤러를 구현하고, 이를 필드버스 응용 시스템의 하나의 노드로 사용하며 그 동작과 기능을 평가하였다. 그림 8은 구현한 마이크로콘트롤러의 구조를 보인다. 그림의 짙은 색 부분은 MC68360의 SCC(Serial Communication Controller)에서 제공하는 직렬통신 기능, 필드버스와 연결되는 RS-485 통신부, 그리고 마이크로콘트롤러의 동작을 감시하는 PC와의 인터페이스를 위한 RS-232 통신부를 나타낸다. 필드버스 출력 단에는 라인버퍼를 추가하여 스위치 회로에 의해 RS-485 트랜시버의 DE(Driver Output Enable)와 라인버퍼가 모두 활성화될 때 버스로의 출력을 허용한다. 또한 버스로부터의 입력은 각각의 RS-485 트랜시버를 통하여 동일한 데이터를 동시에

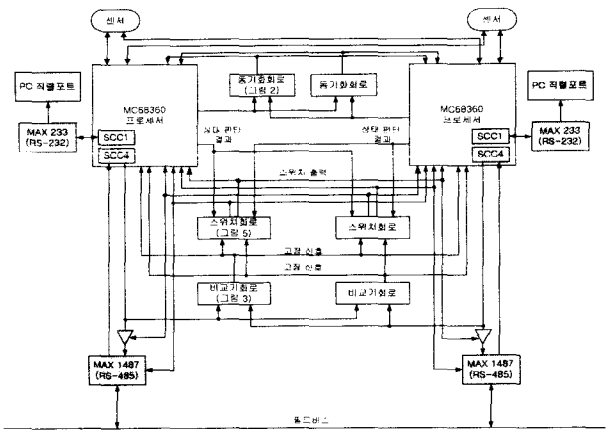


그림 8 이중 마이크로콘트롤러 구조의 구현

Fig. 8 Implementation of dual microcontroller architecture

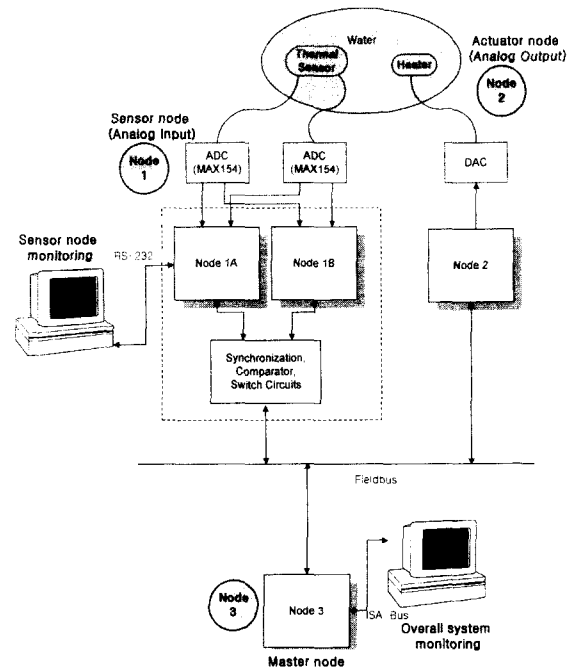


그림 9 응용 및 검증 환경

Fig. 9 Testbed

표 2 테스트 항목

Table 2 List of tests performed

테스트 항목	고장주입	확인방법
· 일시적 고장으로부터의 재시작	정상 동작 중 비교기 회로 출력에 일시적인 stuck-at-low 고장 주입	노드 모니터링 (노드 상태변화 출력)
· 반복적인 자체진단에 의한 고장정지	정상 동작 중 직렬 출력에 연속해서 일시적인 stuck-at-low 고장 주입	
· 동기화 회로의 결함내성 (영구적 고장에 의한 고장정지) · 자체진단에 의한 고장 발견	전원 인가 시 동기화 회로 출력에 영구적 stuck-at-low 고장 주입	
· 스위치 회로의 결함내성 (정상 값으로의 고착 발견)	정상 동작 중 스위치 회로 출력에 영구적 stuck-at-high 고장 주입	
· 비교기 회로의 결함내성 (정상 값으로의 고착 발견, 영구적 고장에 의한 고장정지)	정상 동작 중 비교기 회로 출력에 영구적 stuck-at-high(low) 고장 주입	
· ADC 회로의 결함내성 (영구적 고장에 의한 고장정지)	정상 동작 중 ADC 회로의 CS입력 단에 영구적 stuck-at-high 고장 주입	

수신한다.

전체 응용 환경은 그림 9와 같다. 이 응용 예에서 이중 구조의 마이크로콘트롤러는 센서 및 A/D 컨버터와 결합하여 필드버스 응용 시스템을 구성하는 하나의 지능형 센서 노드(Node 1)를 형성한다. 물론 제안한 이중 구조는 아날로그 출력을 위한 액추에이터 노드(Node 2)나 PC에 ISA 카드 형태로 장착된 마스터 노드(Node 3)에도 활용 가능하다. 이 세 노드로 구성된 응용 시스템은 하나의 온도제어 루프를 형성하고, 이 시스템의 전체적인 동작은 마스터 노드와 연결된 PC를 통해 통제된다. 센서 노드를 감시하는 PC는 센서 노드의 모든 상태 변화를 기록한다. 그림 9의 환경에서 필드버스 인터페이스 기능과 온도제어 시스템은 본 연구와 연계하여 진행된 다른 연구의 결과를 활용하였다.

이러한 환경을 이용하여 제안한 이중 구조의 결함내성을 인위적인 고장 주입 실험을 통해 검증하였다. 표 2는 이중 구조를 갖는 센서 노드의 결함내성 기능을 검증하기 위해 수행한 평가에서의 테스트 항목을 보인다. 주입한 결함의 형태는 신호의 영구적 고착 및 10μs 동안 틀린 값을 갖도록 하는 일시적 고장의 두 가지이다.

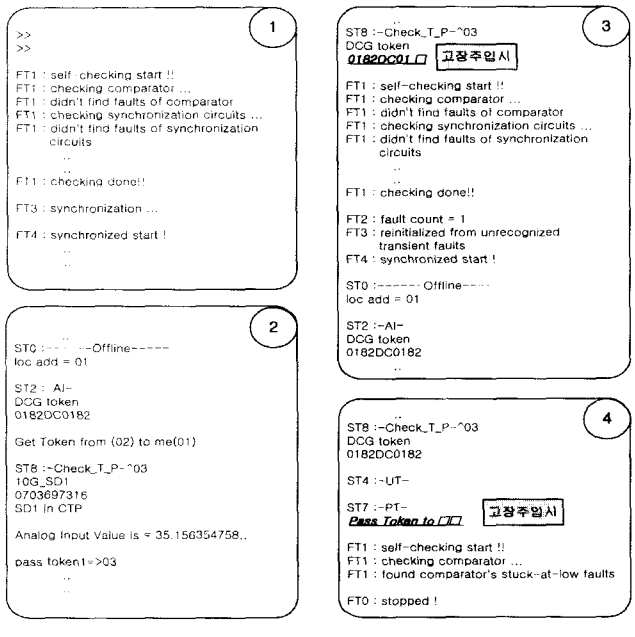


그림 10 고장 주입을 통한 검증

Fig. 10 Fault injection and verification

그림 10은 이중 구조의 동작과 고장 주입 실험의 실제 예를 센서 노드를 감시하는 PC의 화면을 통해 보인다. 그림에서 'FT'로 시작하는 메시지는 이중 구조의 결함내성 기능과 관련된 동작을 나타내고, 그 외의 메시지는 필드버스 프로토콜에 의한 네트워크 기능의 수행과 센서를 통해서 입력된 데이터의 값을 나타낸다. 그림 10의 ①번 화면은 센서 노드에 전원이 인가된 후 자체진단과 동기화 과정을 거쳐서 락스텝 동작이 시작되었음 나타낸다. 화면 ②에서는 1번 노드(센서 노드)가 2번 노드로부터 토큰을 수신하고 센서의 데이터를 처리하여 네트워크를 통해 전송한 후에 다시 3번 노드로 토큰을 넘겨주었음 보인다.

화면 ③ 및 ④는 센서 노드에 인위적인 고장을 주입했을 때의 동작을 나타낸다. 화면 ③은 비교기의 출력이 일시적으로 "0"이 되도록 고장을 주입한 경우이다. 이 고장은 토큰이 수신되는 도중에 주입된다("DCG token 0182DC01..."). 이 때 비교기가 두 프로세서의 출력이 다른 것을 발견하고, 이에 따라 RS-485 트랜시버의 출력이 차단되며, 비교기에 의한 인터럽트가 발생되어 자체진단이 시작된다("FT1 : self-checking ..."). 자체진단에서는 내부의 고장이 발견되지 않아 일시적인 고장으로 판단하고 고장 횟수를 기록한("FT2 : fault count = 1") 후에 동작을 재개한다("FT4 : synchronized start !").

화면 ④는 비교기의 출력이 "0"으로 고착되도록 고장을 주입한 경우이다. 이 고장은 센서 노드가 2번 노드에게 토큰을 보내려고 하는 도중에 주입되고("Pass Token to..."), 이에 따라 RS-485 트랜시버의 출력이 차단되고 동시에 비교기 인터럽트가 발생된다. 그 후 센서 노드는 자체진단을 통해 비교기의 영구적인 고장을 확인하고 동작을 정지한다("FT0 : stopped !").

8. 결 론

지금까지 결함내성형 시스템의 개발은 연구 개발의 난이도와 특성상 주로 대형, 고가의 시스템을 대상으로 각각의 응용 환경에 따라 개별적으로 이루어져 왔다. 그리고 결함내성형 구조에 대한 개념적인 내용은 공개되었으나, 구체적인 설계 방법에 관한 기술적인 사항은 대부분 문헌에 공개되지 않아, 높은 신뢰도를 갖는 다양한 마이크로콘트롤러 응용 시스템의 설계를 어렵게 하고 있다. 본 논문에서는 상용 VLSI 프로세서를 이용하여 높은 신뢰도를 갖는 마이크로콘트롤러를 구성하기 위한 이중 구조와 그 구체적인 설계 방법을 제안하였다. 제안한 이중 마이크로콘트롤러 구조는 어느 하나의 구성 모듈에 고장이 발생하더라도 전체적으로 고장정지 기능을 제공하며 또한 모든 일시적인 고장으로부터의 회복을 가능하게 한다. 이러한 고장정지 특성을 갖는 마이크로콘트롤러는 높은 신뢰도를 요구하는 응용 시스템을 구성하는데 필수적인 요소이다. 또한 디지털 시스템에서 발생하는 거의 대부분의 고장은 일시적인 결함에 의한 것이므로 일시적인 고장으로부터의 회복 능력은 마이크로콘트롤러의 신뢰도를 크게 향상시킬 수 있다.

제안한 이중 구조에서는 공통의 클럭을 사용하여 두 프로세서를 락스텝으로 동작시키고, 두 프로세서의 외부로의 출력을 비교함으로써 하나의 프로세서에 고장이 발생한 경우 이를 발견한다. 이를 위해 결함내성형 클럭을 사용할 뿐 아니라, 프로세서의 초기화 과정에서 프로세서의 정지모드를 이용하여 두 프로세서의 동작을 클럭 사이클 단위로 일치시킨다. 일단 두 프로세서의 출력에 차이가 발생하면, 우선 외부로의 출력을 차단한 후에 자체진단을 통해 스스로의 상태를 진단하며, 발견된 고장이 일시적인 고장으로 판단된 경우에만 동작을 재개한다. 제안한 이중 구조에서는 프로세서 뿐 아니라 비교기와 동기화를 위한 회로 등의 결함내성을 위해 부가되는 모든 구성 모듈을 이중화하여 이들의 자체적인 고장에 대한 내성을 부가한다. 그리고 이 구조는 비교기의 출력과 같은 신호가 정상 값으로 고착되는 경우 이를 발견할 수 있는 능력을 갖고 있다. 또한 상용 프로세서의 자체적인 고장 진

단 기능의 불완전함으로 인해 진단되지 않는 영구적인 결함이 있을 수 있고 이에 따른 반복적인 고장이 발생할 수 있으므로, 이러한 반복적인 고장이 발견될 경우 스스로 동작을 정지한다.

본 연구에서는 제안한 이중 구조를 모토롤라의 MC68360 프로세서와 범용 로직 IC를 이용하여 구현하였다. 그리고 구현한 마이크로콘트롤러와 온도 센서를 결합하여 필드버스 응용 시스템을 위한 센서 노드를 구성하고, 이러한 응용 환경에서 인위적인 고장 주입을 통하여 제안한 구조의 기능을 확인하였다. 제안한 이중 구조는 높은 하드웨어 신뢰도를 요구하는 다양한 마이크로콘트롤러 응용 시스템에 활용될 수 있을 것이다

감사의 글

본 연구는 한국과학재단 목적기초연구(과제번호: 97-0100-1101-3)지원으로 수행되었음.

참 고 문 헌

[1] D. P. Siewiorek and R. S. Schwartz, The Theory and Practice of Reliable System Design, Digital press, 1982.

[2] B. W. Johnson, Design and Analysis of Fault-Tolerant Digital Systems, Addison-Wesley, 1989.

[3] A. L. Hopkins Jr., T. B. Smith III and J. H. Lala, "FTMP-A Highly Reliable Fault-Tolerant Multiprocessor for Aircraft," Proceedings of the IEEE, Vol. 66, No. 10, pp. 1221-1239, Oct. 1978.

[4] A. Avižienis, G. C. Gilley, F. P. Mathur, D. A. Rennels, J. A. Rohr and D. K. Rubin, "The STAR(Self-Testing and Repairing) Computer: An Investigation on the Theory and Practice of Fault-Tolerant Computer Design," IEEE Transactions on Computers, Vol. C-20, No. 11, pp. 1312-1321, Oct. 1971.

[5] O. Serlin, "Fault-Tolerant Systems in Commercial Applications," IEEE Computer, Vol. 17, No. 8, pp. 19-30, Aug. 1984.

[6] MC68360 Quad Integrated Communications Controller User's Manual, Motorola, 1993.

[7] 8XC196Kx, 8XC196Jx, 87C196CA Microcontroller Family User's Manual, Intel, 1995.

[8] P. K. Lala, Fault-Tolerant and Fault-Testable Hardware Design, Prentice-Hall, 1985.

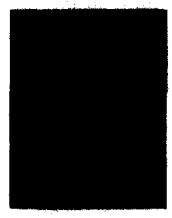
[9] J. Gray and A. Reuter, Transaction Processing: Concepts and Techniques, Morgan Kaufman, 1993.

[10] A. M. Usas, "A Totally Self-Checking Checker Design for the Detection of Errors in Periodic Signals," IEEE Transactions on Computers, Vol. C-24, No. 5, pp. 483-488, May 1975.

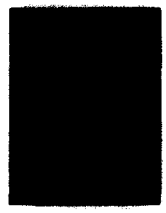
[11] W. M. Daly, A. L. Hopkins Jr. and J. F. McKenna Jr., "A Fault-Tolerant Clocking System," International Symposium on Fault-Tolerant Computing, 1973.

[12] C. M. Krishna, K. G. Shin and R. W. Butler, "Ensuring Fault Tolerance of Phase-Locked Clocks," IEEE Transactions on Computers, Vol. C-34, No. 8, pp. 752-756, Aug. 1985.

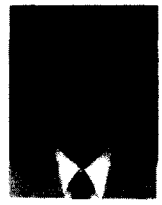
저 자 소 개



김 병 진 (金 炳 辰)
bj.kim@samsung.com
삼성전자 DSN총괄 SOC연구소 연구원
1999년 광운대학교 공학사
2001년 한양대학교 공학석사



백 승 수 (白 承 洙)
ssbaek@samsung.co.kr
삼성전자 DSN총괄 AMLCD사업부
선임연구원
1998년 한양대학교 공학사
2000년 한양대학교 공학석사



이 인 환 (李 寅 煥)
ihlee@hanyang.ac.kr
한양대학교 전자전기컴퓨터공학부 조교수
1979년 서울대학교 공학사
1985년 서울대학교 공학석사
1994년 University of Illinois
at Urbana-Champaign 공학박사

임 동 진 (林 東 進)
limdj@chollian.net
한양대학교 전자컴퓨터공학부 부교수
1979년 서울대학교 공학사
1981년 서울대학교 공학석사
1988년 University of Iowa 공학박사