DOI: 10.3745/KIPSTA.2009.16-A.4.235

확장 ASR 기법을 이용한 임무지향 컴퓨터의 설계 및 신뢰도 분석

신 진 범[†] · 김 상 하[††]

요 약

대공방어용 임무지향 컴퓨터는 장시간의 대공방어 임무를 성공적으로 완수하기 위하여 고장감내 기능이 필수적으로 요구되며, 고장감내 기법으로는 지상 장비인 임무지향 컴퓨터에 적합한 저비용의 기법이 요구된다. 현재 기본형의 임무지향 컴퓨터에는 저비용의 ASR 고장 감내 기법이 적용되어 있으나, ASR 기법은 TMR 기법을 적용한 컴퓨터에 비해 낮은 임무 신뢰도를 제공한다. 그러므로 본 논문에서는 TMR 기법을 적용한 컴퓨터 보다 적은 수량의 프로세서 보드를 사용하여 우수한 신뢰도를 제공하는 확장형 ASR(EASR) 고장 감내 기법을 제안하였으며, 제안된 EASR 기법 을 적용한 임무지향 컴퓨터의 고장 감내 성능을 입증하였다. EASR 기법의 임무지향 컴퓨터는 고장 감내 성능과 저비용성 측면에서 대공방어 시스템의 컴퓨터에 적합하였다.

키워드 : 능동 예비 이중화, 확장 능동 예비 이중화, 고장감내, 고장탐지, 고장복구, 임무지향 컴퓨터, 임무 신뢰도

The Design and Reliability Analysis of A Mission-Critical Computer Using Extended Active Sparing Redundancy

Shin, Jin Beom[†] · Kim, Sang Ha[††]

ABSTRACT

The mission-critical computer for air defense has to maintain its operation without any fault for a long mission time and is required to implement at low cost. Now the reliability of the mission critical-computer using Active Sparing Redundancy fault-tolerant technique is inferior to that of the computer using TMR technique. So in this paper are proposed Extended ASR(EASR) technique that provides higher reliability than that of the computer using TMR technique. The fault-tolerant performance of the implemented mission-critical computer is proven through reliability analysis and numbers of fault recovery test. Also, the reliability of the mission-critical computer using EASR technique is compared with those of computer using ASR and TMR techniques. EASR technique is very suitable to the mission-critical computer.

Keywords : Active Sparing Redundancy, Extended Active Sparing Redundancy, Fault Tolerant, Fault Detection, Fault Recovery, Mission-Critical Computer, Mission Reliability

1. 서 론

대공방어 임무의 수행중에 발생하는 임무지향 컴퓨터의 고장은 대공방어 시스템의 마비를 초래하여 아군에 심각한 피해를 초래할 수 있다. 이런 기능의 임무지향(mission-critical) 컴퓨터에는 고장감내 기능[1, 2, 3]과 안정성[4, 5]이 요구되며, 지상에서 운용되는 장비인 임무지향 컴퓨터의 용도에 알맞은 저비용의 구현이 요구된다. 항공기 또는 우주장비의 경우 치명적인 고장에 대해서도 복구가 가능한 TMR(Triple Modular Redundancy)과 같은 기법이 요구되지만[6], TMR 기법은 고비용이므로 지상장비에는 부적합하다.

기본형의 임무지향 컴퓨터에는 소프트웨어 고장감내 기법으로써 저비용 구현이 가능한 ASR(Active Sparing Redundancy) 기법이 적용되어있다[7]. ASR 기법은 주 소프트웨어 모듈에서 발생하는 고장을 능동적으로 동작하는 예비 모듈로 복구하는 기법으로써 하드웨어의 추가가 불필요하므로 저비용의 구현이 가능하다. ASR 기법의 기본형 임무지향 컴퓨터는 4장의 프로세서 보드로 구성되었으며, TMR 기법을 적용하면

† 정 회 원 : 국방과학연구소 책임연구원
†† 종신회원 : 충남대학교 전기정보통신공학부 교수
논문접수 : 2009년 2월 17일
수 정 일 : 1차 2009년 4월 28일, 2차 2009년 5월 25일, 3차 2009년 6월 18일
심사완료 : 2009년 6월 24일

임무지향 컴퓨터는 6 장의 프로세서 보드로 구성될 수 있다. ASR 기법은 비용측면에서는 우수하지만, TMR 기법의 컴퓨터 보다 낮은 신뢰도를 가진다. 본 논문에서는 저비용으로 TMR 기법보다 높은 신뢰도를 제공하는 확장형 ASR (Extended ASR, 이하 EASR 이라함) 고장감내 기법을 제안하였다. 구현된 EASR 기법의 임무지향 컴퓨터는 TMR 기법의 컴퓨터 보다 적은 수량의 프로세서 보드를 사용하여 TMR 기법보다 우수한 신뢰도를 제공한다. 군용 프로세서 보드는 가격이 수천만원 정도이므로 임무지향 컴퓨터의 총 배치 수량을 고려하면 저비용의 구현은 필수적이다.

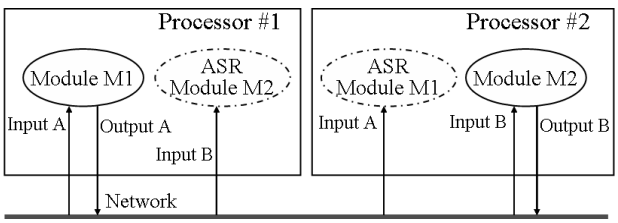
본 논문에서는 제안된 EASR 고장감내 기법을 적용한 임무지향 컴퓨터를 구현하여 고장감내 성능을 입증하고, 신뢰도 분석을 통하여 EASR 기법이 임무지향 컴퓨터의 고장감내 아키텍처에 매우 적합하다는 것을 입증하였다.

2. ASR, EASR 및 TMR 고장감내 기법

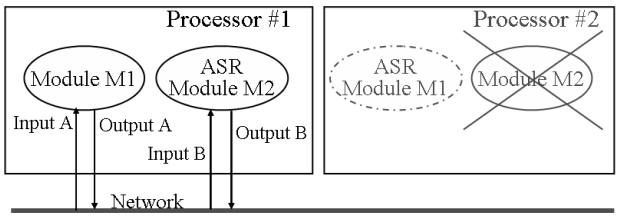
2.1 기본 ASR 고장감내 기법

기본형의 임무 컴퓨터에는 복잡한 교전통제 임무 소프트웨어들이 내장되며[8], 동일한 두개의 소프트웨어로 고장감내를 수행하는 2-Version 프로그래밍 기법[9]와 유사한 ASR 기법이 적용되어 있다. 기본형 임무 컴퓨터는 이산사건 중심으로 동작하는 실시간내장형 시스템이다[10].

ASR 고장감내 모델[7]은 (그림 1)과 같다. ASR 기법은 쌍으로 구성되는 프로세서 보드 1, 2에 대해 프로세서 보드 2에서 동작하는 주 소프트웨어 모듈 M2의 복사본을 프로세서 보드 1에서 M1과 함께 동작시켜 프로세서 보드 2의 고장을 복사본의 ASR 모듈 M2가 복구하는 기법이다. 프로세서 보드 1에서 동작하는 예비 모듈 M2는 프로세서 보드 2에서 동작하는 주모듈 M2와 동일한 입력을 수신받아 처리하며, 프로세서 보드 2의 고장이 진단되면 결과를 출력하여 고장을 복구한다. 각 프로세서 보드에 탑재되는 고장탐지 모듈은 프로세서 보드의 주 모듈로 입출력되는 정보의 패러티 오류와 heartbeat를 검사하여 프로세서 보드의 고장을 탐지하며, 탐지결과를 쌍으로 구성된 프로세서 보드로 전달한다. 탐지결과를 전달받은 프로세서 보드의 고장진단 프로그램은 수신받은 탐지결과에 고장이 있을 경우, 예비 소프트웨어 모듈로 고장을 복구한다. (그림 2)는 프로세서 보드 2의 고장을 프로세서 보드 1의 ASR 모듈 M2가 복구한 결과를 보여준다.



(그림 1) ASR 고장감내 모델



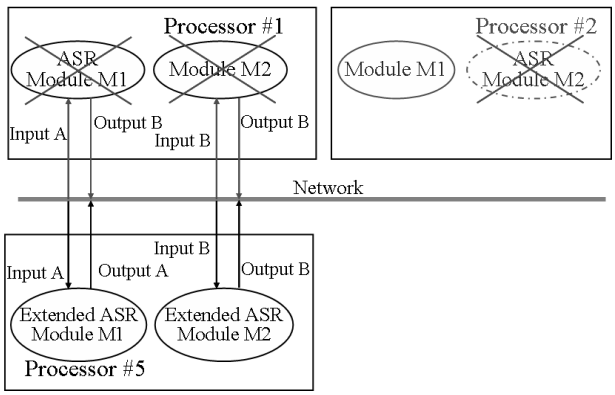
(그림 2) ASR 기법의 고장복구

2.2 EASR 고장감내 기법

제안된 EASR 기법은 ASR 기법을 확장하여 컴퓨터의 신뢰도를 개선하는 기법이다. EASR 고장감내 기법의 동작 모델은 (그림 3)과 같다. 프로세서 보드 2의 고장을 ASR 기법으로 복구한 프로세서 보드 1에 고장이 발생하였을 때, 확장 프로세서 보드 5가 EASR 모듈로 고장을 복구한다. EASR 모듈 M1과 M2는 ASR 기법의 프로세서 보드 1과 2에 탑재된 주 임무 소프트웨어 모듈 M1과 M2의 복사본이다.

확장 프로세서 보드(프로세서 5)의 고장복구 절차는 ASR 기법의 절차와 유사하다. EASR 모듈이 내장된 확장 프로세서 보드 5는 ASR 기법으로 동작하고 있는 프로세서 보드 1(또는 2)의 고장탐지 결과를 수신받아서, 고장으로 진단되면 두 개의 EASR 모듈로 프로세서 보드 1(또는 2)의 고장을 복구한다.

교전통제 임무 소프트웨어는 복잡한 대규모의 소프트웨어[8]이며, 임무 특성상 안정성[4, 5]과 50% 이상의 리소스 예비율이 요구된다. 이런 임무지향 컴퓨터에는 안정성을 위하여 분산 구조의 실시간 내장형의 시스템 설계가 요구되며[11], 리소스 예비율 최적화를 위하여 프로세서 보드들 간에 최적의 대역폭 분산이 요구되므로[12] 임무 용도와 네트워크 점유율에 따라 소프트웨어를 M1, M2, M3 및 M4 모듈로 나누고, 이들을 4 장의 프로세서 보드로 분산하였다[7]. ASR 기법은 M1, M2, M3 및 M4가 탑재되는 4 장의 프로세서 보드를 2 장씩 짝을 지어(그림 1 참조) 고장감내를 구현하며, EASR 기법은 4장의 보드로 구성되는 ASR 기법의 컴퓨터에 한 장의 확장 ASR 프로세서 보드를 추가하여 고장감내를 구현한다. (그림 3)은 2 장의 보드로 구성된 ASR 기법

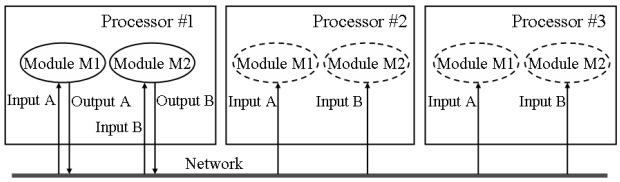


(그림 3) EASR 기법 동작 모델

의 컴퓨터에 EASR 기법을 적용한 예를 설명하였다. 2 장의 프로세서 보드로 구성된 컴퓨터에 EASR 기법을 적용하면 2.3에 기술하는 TMR 컴퓨터와 동일하게 3 장의 프로세서 보드가 필요하게 된다.

2.3 TMR 고장감내 기법

M1 및 M2 소프트웨어를 탑재한 한 장의 프로세서 보드에 대해 TMR 고장감내 기법을 적용한 컴퓨터 모델은 (그림 4)와 같다. TMR 기법의 컴퓨터는 ASR 기법보다 신뢰도는 우수하지만 한 장의 프로세서 보드가 추가적으로 사용되므로 비용이 증가한다.



(그림 4) TMR 고장감내 모델

3. EASR 고장감내 메커니즘

3.1 기본 가정

EASR 고장감내 기법을 임무 컴퓨터에 적용하기 위한 기본 가정은 다음과 같다.

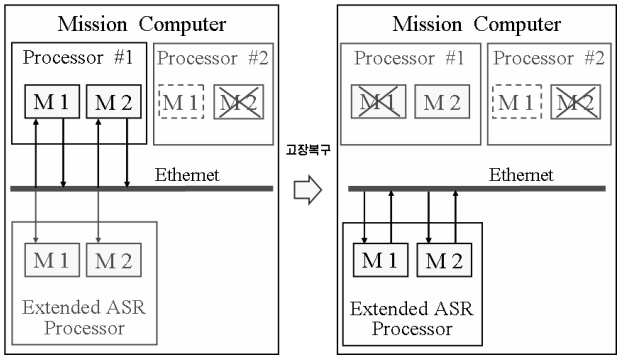
- 소프트웨어 코드에는 오류가 없다. 따라서 소프트웨어에서 탐지되는 고장은 하드웨어 고장이다.
- 외부 네트워크 장비에는 오류가 없다. 전달 정보의 오류는 프로세서 보드의 하드웨어 고장이다.
- 임무 컴퓨터는 프로세서 보드와 보드들이 장착되는 모체배선기판 및 연결케이블로 구성되며, 프로세서 보드를 제외한 장치에는 고장이 없다. 프로세서 보드(이하 프로세서라 함)는 SBC(Single Board Computer) 보드와 PMC(PCI Mezzanine Card)형 이더넷 네트워크 카드로 구성된다.

3.2 EASR 고장감내 기법의 동작

EASR 고장감내 기법은 ASR 기법이 적용된 프로세서의 고장을 EASR 모듈이 탑재된 확장 프로세서로 복구하여 신뢰도를 개선하는 기법이다. (그림 5)는 EASR 모듈의 고장복구 동작을 보여준다. 프로세서 보드 2의 고장을 프로세서 1에서 동작하는 ASR 모듈이 복구한 후에, 다시 프로세서 1에서 발생하는 고장을 확장 프로세서의 EASR 모듈이 복구한다.

3.3 고장탐지 및 진단

EASR 기법의 고장탐지 및 진단 절차는 ASR 기법의 절차를 확장한다. 앞선 (그림 5)의 확장 프로세서 보드에서 동

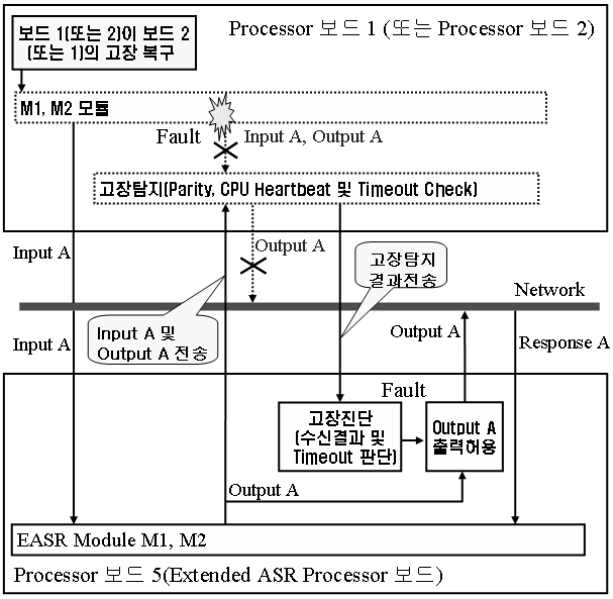


(그림 5) EASR 기법의 고장복구 동작

작하는 예비 모듈 M1, M2는 프로세서 보드 1과 2의 고장을 EASR 기법으로 복구한다. M1 및 M2 모듈들은 리소스를 관리하는 네트워크 미들웨어[13, 14]에 의해 publisher-and-subscriber 모드로 입출력 정보를 주고받는다.

(그림 2)와 같이 ASR 기법으로 동작하는 보드 1과 2는 고장탐지 결과를 상호 송수신하여 고장을 진단하며, 고장탐지 내용을 확장 보드 5로는 전송하지 않는다. 이때 보드 1(또는 2)에서 고장이 발생하면 보드 2(또는 1)가 ASR 기법으로 고장을 복구한 후에 확장 보드와 함께 EASR 기법으로 동작한다. 그러므로 확장 보드에서 보드 1(또는 2)의 고장이 진단된 경우는 이미 보드 1과 2에 고장이 발생한 상태이다. 프로세서 보드 1(또는 2)은 ASR 기법으로 보드 2(또는 1)의 고장을 복구한 후부터 고장탐지 결과를 확장 보드에 송신한다. M1과 M2가 동작하고 있는(처리 결과를 네트워크로 출력하지는 않음) 확장 보드는 수신받은 보드 2(또는 1)의 고장을 진단하기 시작한다. EASR 기법의 고장탐지 및 진단 절차는 (그림 6)과 같다. 확장 보드에서 동작하는 M1 및 M2 예비 모듈과 보드 1(또는 2)에 내장된 M1 및 M2 모듈은 이더넷 네트워크로부터 동일한 입력을 받아서 임무를 수행하며, 확장 프로세서의 M1 및 M2 모듈들은 출력의 계산 후에 입력값과 함께 출력값을 보드 1(또는 2)의 고장탐지 모듈로 전송한다. 보드 1(또는 2)의 고장탐지 모듈은 자신이 계산한 패러티 오류를 이 수신값과 비교하여 재차 확인하고, 패러티 오류와 함께 CPU heartbeat 및 timeout을 포함한 탐지결과를 확장 보드에 전달한다. 보드 1(또는 2)의 timeout 오류는 보드 1(또는 2)의 고장탐지 모듈이 확장 보드로부터 입출력 값을 수신한 후 일정시간이 지나도 자체의 입출력이 생성되지 않으면 발생된다. 보드 1(또는 2)은 고장이 탐지되면 결과를 외부로 출력하지 않으며, 확장 보드에서 보드 1(또는 2)의 고장이 진단되면 예비 모듈 M1 및 M2의 결과들이 네트워크로 출력되어 실시간적인 고장복구가 수행된다. 보드 1(또는 2)의 치명적 결함으로 탐지결과의 전송이 불가한 경우에 확장 보드는 이를 timeout 오류로 처리하여 고장을 복구한다.

EASR 기법은 M3과 M4 모듈이 내장되어 ASR 기법으로 동작하는 2 장의 추가 보드에도 적용된다. 4 개의 M1, M2,



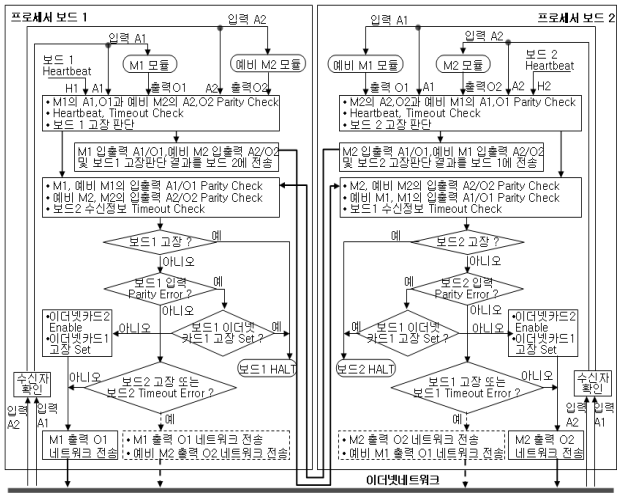
(그림 6) EASR 기법의 고장탐지, 진단 및 복구

M3 및 M4 모듈이 4 장의 프로세서 보드에 분산되는 임무 컴퓨터에서 M1과 M2가 탑재된 2 장의 보드중 한 장에 고장이 발생하면 다른 한 장이 ASR 기법으로 고장을 복구한 후에, 확장 보드와 EASR 기법으로 동작한다. 이는 M3와 M4 탑재 보드의 경우도 동일하다. 그러나 확장 보드는 두 그룹의 ASR 보드들중 먼저 탐지된 하나의 고장만을 복구하며, 그 후에 탐지된 고장은 복구하지 않고 치명 고장으로 처리한다. 한 장의 확장 보드에서 4 개의 M1, M2, M3 및 M4 모듈의 처리 결과를 동일한 네트워크로 출력하면서 임무를 수행하는 것은 통신대역폭도 부족과 관련하여 성능 저하를 야기한다. 그러므로 한 장의 확장 보드는 먼저 발생한 고장만을 복구하며, 그 후에 발생하는 고장은 복구하지 않는다. 따라서 EASR 기법으로 고장을 복구한 확장 보드에서 4개의 M1, M2, M3 및 M4 모듈은 입력은 모두 받지만, 두 개의 모듈만(M1과 M2 또는 M3와 M4)이 네트워크로 출력을 수행한다.

Heartbeat는 CPU 버스의 입출력 계통이 정상인 가를 100 msec의 주기로 점검하는 기능으로 프로세서 보드가 외부의 입출력장치들과 점검 정보를 송수신하여 프로세서 보드와 관련된 고장을 탐지한다. Heartbeat는 CPU 내부와 메모리 및 모든 입출력 장치를 포함한 세부 점검을 수행하지는 않으나, 임무 소프트웨어 운용에 치명적인 프로세서 보드의 고장이나 외부장치들의 고장을 탐지한다. 외부 장치를 포함한 프로세서 보드의 세부 점검은 컴퓨터에 전원이 인가되는 초기에 POST(Power On Self Test) 점검에 의해 수행되며, POST 점검에 의해 탐지되는 고장은 프로세서 보드를 교체하거나 외부 장치를 수리하여 제거된다.

3.4 고장복구

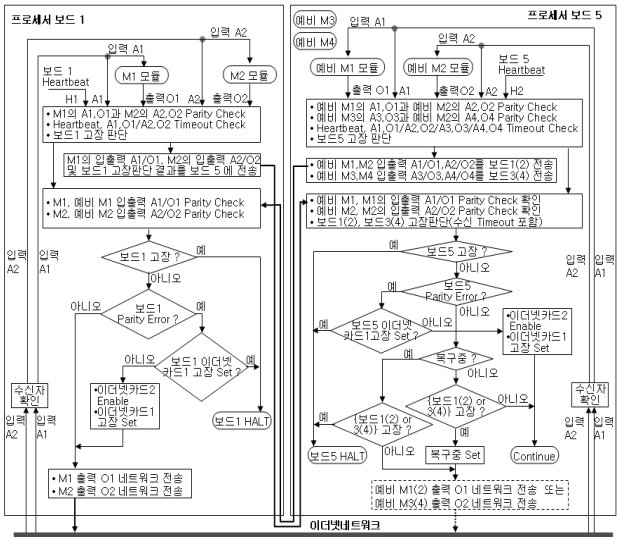
앞의 (그림 6)과 같이 EASR 기법의 확장 프로세서의 고장진단 소프트웨어에서 고장이 진단되면 확장 프로세서에 탑재된 예비 모듈들의 출력이 네트워크로 전송되게 하여 임무 컴퓨터의 고장을 복구한다. ASR 기법으로 동작하는 보드 1과 2 또는 3과 4의 고장탐지, 진단 및 복구 알고리즘은 (그림 7)과 같다. 보드에서 Heartbeat와 입출력 값의 패러티로부터 고장을 판단하는 기준은 <표 1>과 같으며, 입출력에 대한 Timeout은 프로세서 보드의 고장으로 판단한다. 보드에서 첫 번째 네트워크 카드의 고장은 두 번째 네트워크 카드로 복구되며, 두 번째 네트워크 카드의 고장은 프로세서 보드의 고장으로 판단한다. 각 보드는 보드의 고장 유무와 입출력 정보(패러티 포함)를 상대 보드로 전송하며, 이후에 자신의 보드에서 동작하는 소프트웨어 모듈의 입출력 정보와 상대에서 보내온 입출력 정보를 재확인하고(Timeout 점검 포함), 복구를 수행한다. (그림 7)과 같이 보드 1, 2, 3 및 4 의 고장을 복구한 보드들은 프로세서 보드 5 와 EASR 기법으로 동작하며, 이때의 고장탐지, 진단 및 복구 알고리즘은 (그림 8)과 같다. 프로세서 보드 5는 프로세서 보드 1(2) 또는 3(4)로 입출력 정보는 전송하지만 보드의 고장 유



(그림 7) ASR 기법의 고장탐지 및 복구 알고리즘

〈표 1〉 프로세서 보드의 고장판단 기준

Heartbeat Error	Output Error	Input Error	고장 판단
X	X	X	고장 없음
X	X	○	네트워크카드 고장
X	○	X	프로세서보드 고장
X	○	○	프로세서보드 고장
○	X	X	프로세서보드 고장
○	X	○	프로세서보드 고장
○	○	X	프로세서보드 고장
○	○	○	프로세서보드 고장



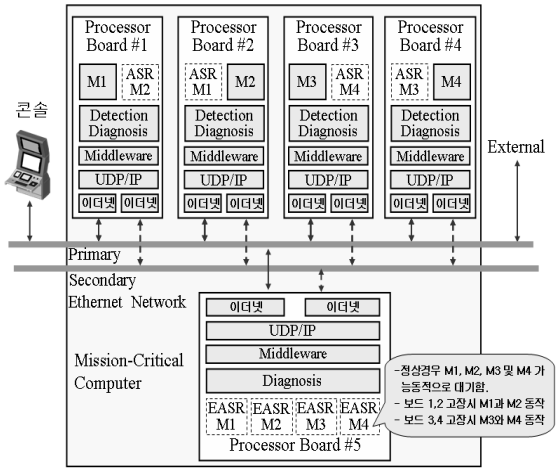
(그림 8) EASR 기법의 고장탐지 및 복구 알고리즘

무는 전송하지 않는다. 이는 보드 1(2)은 보드 5의 고장을 복구할 필요가 없기 때문이다. 보드 5 는 보드 1(또는 2) 또는 보드 3(또는 4)에서 먼저 탐지된 고장을 복구하며, 두 번째 고장은 치명 고장으로 처리한다.

ASR 기법과 동일하게 EASR 기법의 예비 모듈들은 프로세서 1의 모듈들과 동기를 지속적으로 유지하므로 복구 시점의 결정을 위한 별도의 절차는 필요하지 않다.

4. EASR 임무지향 컴퓨터의 설계 및 구현

EASR 임무지향 컴퓨터는 (그림 9)와 같이 5 장의 프로세서 보드로 구현되었으며, 프로세서 보드 1에서 4 까지의 구성은 ASR 기법의 임무 컴퓨터와 동일하다[7]. 보드에는 네트워크 미들웨어인 DDS(Data Distribution Service)가 탑재되어 publisher-and-subscriber 모드의 입출력 정보 교환을 이중화 네트워크 상에서 지원한다. 프로세서 보드는



(그림 9) 설계 구현된 EASR 임무지향 컴퓨터

SBC 보드와 2 장의 PMC 이더넷 카드로 구성되며, SBC 보드는 VME-183(Pentium 1.2GHz, Curtiss Wright)[15]이고 이더넷 카드는 GNET/PMC(1G-byte, CCII)[16]이다. 2 장의 이더넷 카드는 네트워크 이중화 고장감내를 지원한다. 확장 보드 5에는 M1, M2, M3 및 M4 예비 모듈들과 보드 1, 2, 3 및 4의 고장을 진단하는 프로그램이 탑재된다. 쌍을 이루어 ASR 기법으로 동작하는 보드 1 과 2, 그리고 보드 3 과 4 에 고장이 없으면 확장보드의 M1, M2, M3 및 M4 예비 모듈들은 네트워크의 입력 정보의 처리 결과를 네트워크로 출력하지 않는다. 보드 2에서 처음 고장이 발생하면 보드 1은 ASR 기법으로 고장을 복구하며, 다시 보드 1에 고장이 발생하면(보드 2는 이미 고장임), 확장 보드의 M1과 M2가 출력을 네트워크로 전달하여 고장을 복구한다. 만약 보드 3에 고장이 발생하면(보드 4는 이미 고장임), 확장 보드의 M3와 M4가 출력을 네트워크로 전달하여 고장을 복구한다. 확장 보드 5에서 보드 1(또는 보드 2)의 고장을 탐지한 경우는 보드 1과 2 모두에 고장이 있는 것이며, 보드 3(또는 보드 4)의 고장은 보드 3과 4에 고장이 있다는 것이다. 이 두가지 경우(보드 1과 보드 3의 고장)에 대해 확장 보드 5는 먼저 탐지된 고장을 복구하며, 두 번째 고장은 복구하지 않고 치명 고장으로 처리한다. 그러므로 확장 보드에서 동작하는 4 개의 모듈중 두 개의 모듈(M1과 M2 또는 M3와 M4) 만이 네트워크로 출력을 수행하여 복구에 사용되며, 나머지 두 개는 복구에 사용되지 않는다.

본 논문 2.2에 기술된 50% 이상의 리소스 예비율 요구조건에 따라 임무 컴퓨터에서 프로세서 보드 1, 2, 3, 4는 주모듈과 예비모듈이 동작하는 상태에서 50%이상의 리소스 예비율을 보장한다. 임무 소프트웨어의 출력 통신량과 ASR 및 EASR 기법으로 동작할 때의 통신량은 <표 2>와 같다. 소프트웨어 모듈들은 publisher-and-subscriber 모드로 통신하므로 모듈의 출력이 기본 통신량이다. 보드 1이 ASR 모드로 동작시 (M1 출력 + M1/M2 고장탐지결과 출력)의 통신량을 가지며, 약 22.2 Mbps 이다. (M1/M2 고장탐지결과 출력)은 M1과 M2 모듈의 입력과 출력, 그리고 보드 진단결과의 합이며, 보드 진단결과보는 정보량이 매우 적으므로 무시하였다. 보드 1(2)이 보드 2(1)의 고장을 복구하여 보드 5와 EASR 모드로 동작시에는 (M1/M2

〈표 2〉 임무 모듈 통신량 및 보드 통신량

구분	임무 모듈	모듈통신량(Mbps)		보드통신량(Mbps)	
		출력	입력	ASR	EASR
보드 1	M1	5.7	4.2	22.2	26.5
보드 2	M2	4.3	2.3	20.8	
보드 3	M3	3.2	2.5	17.8	23.0
보드 4	M4	5.2	3.7	19.8	
보드 5	M1/M2 또는 M3/M4	10.0 8.4	12.7	X	31.1 또는 24.6/24.9

출력 + M1/M2 고장탐지결과 출력) 통신량을 가지며, 이는 26.5 Mbps 정도이다. EASR 기법으로 동작하는 보드 5는 보드 1(2) 또는 보드 3(4)에 고장이 없을 경우에는 M1/M2/M3/M4의 고장탐지 결과 만이 전송되며(31.1 Mbps), 고장의 복구후에는 복구 보드에 따라 24.6 또는 24.9 Mbps의 통신량을 가진다. <표 2>에서 모듈 간의 임무 수행을 위해 출력되는 M1, M2, M3 및 M4 기본 정보량(18.4 Mbps)은 미들웨어 상에서 응용 프로그램의 송수신자 제어와 네트워크 대역폭 제어(QoS)와 같은 복잡한 과정을 거치므로 처리시간이 소요되며, 이는 프로세서의 성능에 큰 영향을 미친다. 그러므로 보드 5는 보드 1, 2, 3 및 4 의 모든 고장을 복구하여 입출력을 수행하기에는 성능에 제한이 있다. 그러므로 보드 4장의 모든 고장은 복구하지 않으며, 보드 1(또는 2) 또는 3(또는 4)에서 먼저 발생한 하나의 고장만을 복구한다. 보드 상호간에 교환되는 고장탐지 결과 정보는 미들웨어를 거치지 않고 간단하게 송신되므로 보드 5에서 고장탐지 결과의 정보량은 임무를 수행하는 프로세서의 성능에 큰 영향을 주지 않는다. 향후에 한 장의 확장 보드로 4 장의 보드의 고장을 복구하는 문제는 리소스 사용율과 통신점유율의 최적화 연구와 함께 향후 지속적인 연구가 필요하다고 판단된다.

5. 분 석

5.1 고장탐지, 진단 및 고장복구 분석

앞의 (그림 9)에서 프로세서 1, 2, 3 및 4의 고장은 쌍으로 구성된 프로세서에서 복구된다(예로, 프로세서 2의 고장은 프로세서 1이 복구하며, 프로세서 4의 고장은 프로세서 3이 복구함). 그 시점 이후부터 확장 프로세서 5가 EASR 기법으로 프로세서 1(또는 3)의 고장탐지 결과를 진단하여 고장을 복구한다. 프로세서 1(또는 3)의 고장탐지 결과는 다음과 같으며, 고장탐지 결과는 확장 프로세서로 전달된다.

- 입력값의 패러티 오류 : 프로세서의 네트워크 카드 계통에 고장이 있음을 탐지
- 출력값의 패러티 오류 : CPU 계통의 비정상에 의한 임무 소프트웨어의 동작 오류를 탐지
- CPU Heartbeat 오류 : CPU와 메모리 계통의 하드웨어 오류를 탐지
- Timeout 오류 : 입력값과 출력값 및 Heartbeat 가 생성되지 못하는 하드웨어 고장을 탐지

구현된 임무지향 컴퓨터에서 프로세서에 탑재된 고장탐지 및 고장진단 소프트웨어는 모의로 구현된 고장들에 대해 EASR 기법의 고장복구를 잘 수행하였다. 확장 프로세서의 고장진단 소프트웨어는 프로세서 1(또는 3)에서 전달된 고장탐지 결과를 진단하여 컴퓨터의 고장을 복구하였다. 또한 일정시간 내에 고장탐지 결과가 수신되지 않으면 timeout 오류로 프로세서 1(또는 3)의 고장을 복구하였다.

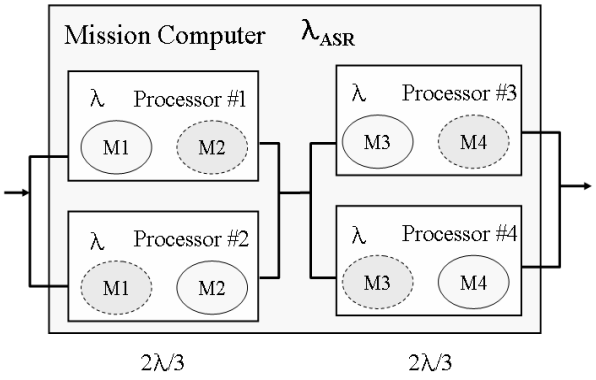
5.2 고장복구 시간 분석

마하 1.5로 기동하는 항공기의 경우 1,500 msec 동안에 약 510 m를 이동한다. 대공방어 시스템에서 고속의 적 항공기를 연속적으로 추적하여 교전을 수행하기 위해서는 1,500 msec 이하의 추적정보 갱신시간이 요구되므로, 컴퓨터의 고장은 1,500 msec 이내에 복구되어야 한다.

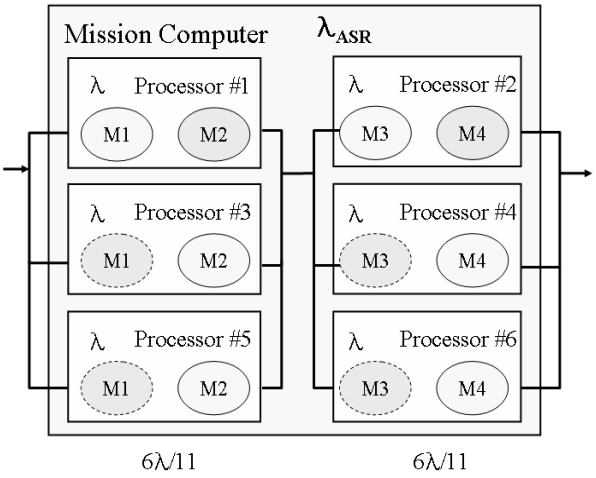
EASR 기법의 고장감내 성능시험은 ASR 기법의 시험과 동일한 방법으로 수행되었으며, 모의 오류에 의해 시험되었다. 임무 소프트웨어의 입출력부에 패러티 오류 생성부를 추가하고, heartbeat 프로그램에 오류 생성부를 추가하여 모의 오류를 발생하였으며, 프로세서가 전혀 동작하지 않는 치명적 고장에 대해서도 timeout 모의 오류를 생성하였다. 랜덤하게 발생시킨 10,000 회 이상의 모의 고장에 대해 고장탐지와 진단을 포함하여 고장이 복구되기까지의 시간은 1,300 msec 이하로 측정되었으며, 이는 대공방어 시스템의 임무지향 컴퓨터에 요구되는 실시간적 고장복구 시간을 만족하였다.

5.3 MTBF 분석

ASR 기법이 적용된 임무지향 컴퓨터의 신뢰도 모델[7]은 (그림 10)과 같고, 프로세서 별로 TMR 기법을 적용한 임무 컴퓨터의 신뢰도 모델은 (그림 11)과 같다. 신뢰도가



(그림 10) ASR 컴퓨터의 신뢰도 모델



(그림 11) TMR 컴퓨터의 신뢰도 모델

〈표 3〉 TMR 컴퓨터 보드 1,3,5의 정상동작 확률

Processor1	Processor3	Processor5	*정상확률P(n)*
○	○	○	$r1^3$
○	○	X	$r1^2(1-r1)$
○	X	○	$r1^2(1-r1)$
○	X	X	$r1(1-r1)^2$
X	○	○	$r1^2(1-r1)$
X	○	X	$r1(1-r1)^2$
X	X	○	$r1(1-r1)^2$
X	X	X	-

R(t)인 컴퓨터의 MTBF는 식 (1)과 같이 표현[17]된다. TMR 컴퓨터에서 보드 1, 3, 5가 정상으로 동작할 경우의 수에 대한 확률은 <표 3>과 같으며, 이때 R(t)는 다음과 같이 표현된다.

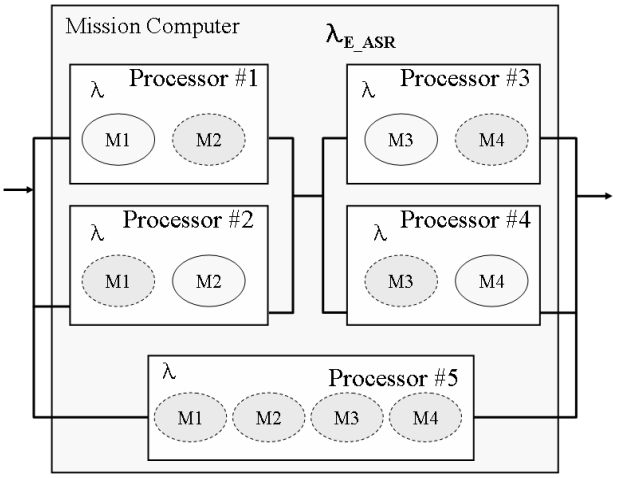
$$MTBF = \int_0^\infty R(t)\,dt \qquad (1)$$
$$R(t) = \sum_n P(n)$$

보드 1, 3, 5에 대한 R(t)와 MTBF는 아래와 같이 계산되며, <표 3>에서 고장율이 λ인 보드 한 장의 신뢰도는 r1이다(고장 확률은 1-r1). 그러므로 보드 6 장에 대한 TMR 컴퓨터의 MTBF는 식 2와 같으며, ASR 컴퓨터의 MTBF도 동일한 방법으로 식 (3)과 같이 계산된다.

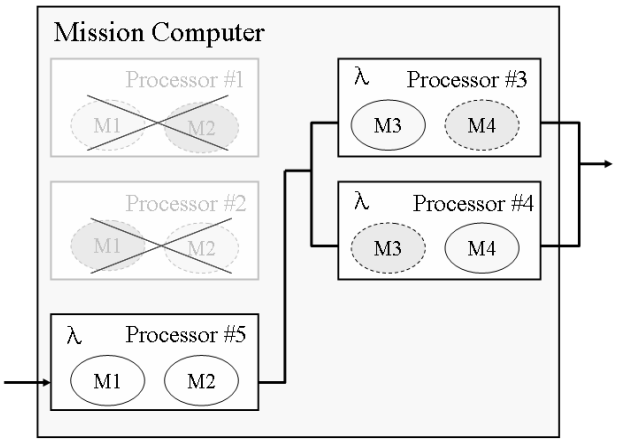
$$R(t) = r1^3 - 3r1^2 + 3r1$$
$$r1 = e^{-\lambda t}$$
$$MTBF_{TMR(\text{보드}1,3,5)} = \int_0^\infty R(t)\,dt = \frac{11}{6\lambda}$$
$$MTBF_{TMR} = \frac{11}{12\lambda} \qquad (2)$$

$$MTBF_{ASR} = \frac{3}{4\lambda} \qquad (3)$$

5 장의 프로세서로 구성된 EASR 컴퓨터의 신뢰도 모델은 (그림 12)와 같다. 확장 프로세서 5는 (그림 13)과 같이 프로세서 보드 1과 2(또는 3과 4)의 고장을 복구한다. (그림 12)에서 컴퓨터가 정상 동작할 경우의 확률은 <표 4>와 같이 표현되며, 신뢰도 R(t)는 이 확률들의 합으로 표현된다[18]. 그러므로 식 (1)에 따라 MTBFEASR는 식 (4)와 같이 계산된다. r1은 프로세서 보드 1과 2가 ASR 기법에서 정상 동작할 신뢰도이며(3과 4도 동일), (그림 10)에 나타낸 고장율 2λ/3 에 대한 신뢰도이다. r2는 고장율이 λ인 프로세서



(그림 12) EASR 컴퓨터의 신뢰도 모델



(그림 13) EASR 컴퓨터의 고장복구 모델

〈표 4〉 EASR 컴퓨터의 정상동작 확률

Processor 1 & 2	Processor 3 & 4	Processor 5	*정상확률 P(n)*
○	○	○	$r1^2r2$
○	○	X	$r1^2(1-r2)$
○	X	○	$r1(1-r1)r2$
○	X	X	-
X	○	○	$r1(1-r1)r2$
X	○	X	-
X	X	○	-
X	X	X	-

보드 5의 신뢰도이다. (그림 11)의 TMR 컴퓨터는 6 장의 보드에 M1과 M2 모듈(보드 1, 3, 5)과 M3와 M4 모듈(보드

2, 4, 6)을 탑재하여 고장을 복구하지만, EASR 컴퓨터는 ASR 기법으로 고장을 복구하는 4장의 보드에 M1과 M2 모듈(보드 1, 2)과 M3와 M4 모듈(보드 3, 4)을 탑재하여 각각의 고장을 복구하고, 추가적으로 EASR 기법의 보드 5에는 M1, M2, M3 및 M4 모듈을 탑재하여 M1과 M2 모듈 또는 M3와 M4 모듈로 보드의 고장을 복구한다. 그러므로 EASR 컴퓨터는 5 장의 프로세서로 6 장으로 구성되는 TMR 기법보다 우수한 임무 신뢰도를 제공한다.

$$R(t) = r1^2 + 2r1r2 - 2r1^2r2$$
$$r1 = e^{-\frac{2\lambda}{3}t},\ r2 = e^{-\lambda t}$$
$$MTBF_{EASR} = \frac{153}{140\,\lambda} \quad (4)$$

5.4 가용도 분석

임무 컴퓨터의 치명적인 고장은 전원차단 후에 고장 모듈을 교체하여 정비되며, 고장 모듈을 교체하는 평균 MTTR은 2 시간이다. 컴퓨터의 가용도는 식 (5)와 같으며[19], MTTR은 MTBF에 비하여 매우 적으므로 가용도에 영향을 거의 미치지 않는다.

$$\text{가용도} = \frac{MTBF}{MTBF + MTTR} \quad (5)$$

5.5 ASR 및 TMR 기법과의 MTBF 비교 분석

프로세서의 고장율(λ)은 CPU 보드[15]와 네트워크 카드[16]의 고장율로부터 2.08556 x 10-5 로 계산된다. 식 (2), (3) 및 (4) 로부터 계산된 MTBF와 식 (5)로부터 계산된 가용도는 <표 5>와 같으며, EASR 기법은 TMR 기법보다 우수한 MTBF를 제공한다.

〈표 5〉 MTBF 및 가용도

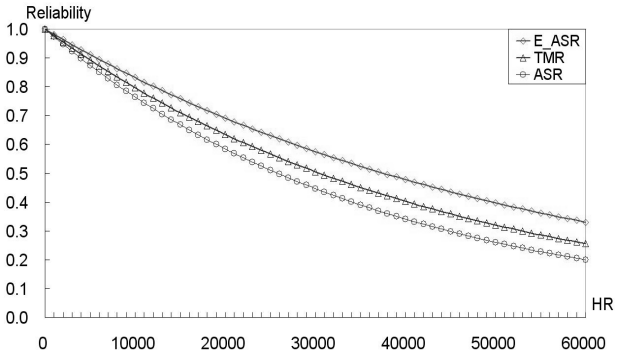
구분 / 고장감내 기법	MTBF(Hr)	가용도
ASR	35,960	0.99994
TMR	43,951	0.99995
EASR	52,399	0.99996

5.6 임무 신뢰도 분석

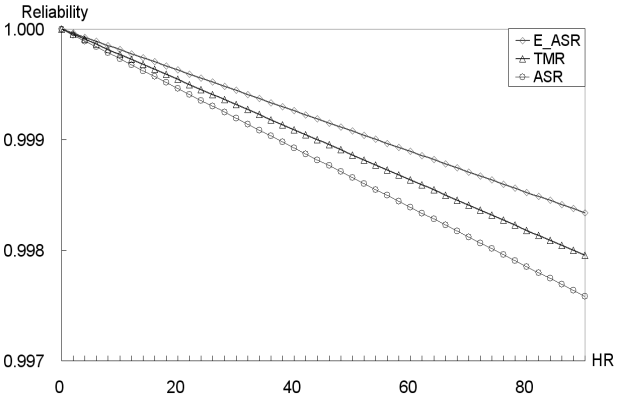
컴퓨터의 임무 신뢰도는 다음과 같이 표현된다.

$$R_{(t)} = e^{-\lambda t} = e^{-\frac{1}{MTBF}t}$$

임무 컴퓨터에 ASR 기법, TMR 기법 및 EASR 기법을 적용한 경우의 임무 신뢰도는 (그림 14)와 같으며, EASR 기법의 신뢰도가 가장 우수함을 알 수있다. 대공방어 시스템의 임무시간은 약 72 시간이므로 임무지향 컴퓨터의 100 시간까지의 임무신뢰도는 (그림 15)와 같다.



(그림 14) 임무 신뢰도



(그림 15) 100 시간 임무 신뢰도

5.7 구현비용대 고장감내 성능 효과

본 논문에서 제안된 EASR 고장감내 기법이 적용된 대공방어용 임무지향 컴퓨터는 5 장의 프로세서로 구현되었다. ASR 기법의 컴퓨터는 4 장의 프로세서로 구현되었으며, TMR 기법 컴퓨터는 6 장의 프로세서가 요구된다. EASR 기법은 5 장의 프로세서를 사용하여 TMR 기법 보다 높은 신뢰도를 제공하였다. EASR 기법은 4장 이상의 프로세서 보드로 구성되는 임무 컴퓨터의 고장감내 기법으로 매우 적합하며, 2 장의 프로세서 보드로 구성되는 임무 컴퓨터에서는 TMR 기법의 컴퓨터(2.3 참조)와 동일한 3장의 프로세서 보드로 구성되므로 장점이 크게 없다.

임무지향 컴퓨터는 많은 대공방어 시스템에 배치되며, 군용 프로세서의 가격이 수천만원 정도이므로 구현비용 대 효과 측면에서 EASR 고장감내 기법은 대공방어용 임무지향 컴퓨터에 매우 적합하다고 판단된다.

6. 결 론

대공방어를 위한 임무지향 교전통제 컴퓨터에는 저비용의 고장감내 기능이 요구된다. 임무지향 컴퓨터에 적용된 EASR

고장감내 기법은 TMR 기법 보다 적은 수의 프로세서를 사용하여 TMR 컴퓨터 보다 높은 임무 신뢰도와 MTBF를 제공하므로 고장감내 성능 대 구현비용 측면에서 TMR 보다 우수하였다. EASR 기법은 복잡한 내장형 소프트웨어로 구성되는 지상장비용 임무지향 컴퓨터에 매우 적합한 고장감내 기법이므로 향후 유사한 군사용 임무지향 컴퓨터에 적용될 수 있을 것으로 사료된다.

향후 임무 컴퓨터의 소프트웨어 처리 부하량과 네트워크 사용 대역폭에 따른 리소스 예비율의 안정성에 대한 추가적인 연구와 한 장의 확장 프로세서 보드를 사용하여 모든 임무 프로세서 보드(4 장의 보드)의 고장을 복구하는 기법에 대한 추가적인 연구가 필요하다고 판단된다.

참 고 문 헌

[1] John Rushby, "Bus Architectures For Safety-Critical Embedded Systems," Lecture Notes in Computer Science, Vol.2211, pp.306-323, October, 2001.

[2] Gul N. Khan, "Fault-Tolerant Architectures for High Performance Embedded System Applicat-ions," in Proc. of *IEEE Conference on Comp-uter Design*, pp.384-389, October, 1998.

[3] K. H. (Kane) Kim, "Issues Insufficiently Reso-lved in Century 20 in the Fault Tolerance Distributed Computing Field," in Proc. of the IEEE CS *19th Symp. Reliable Distributed Systems (SRDS)*, pp.106-115, October, 2000.

[4] 'System Safety Management Guide', Departm-ent of the Army, Pamphlet 385-16, 1987.

[5] 'Risk Management', Department of the Army, Field Manual No.100-14, April, 1998.

[6] Dhiraj K. Pradhan, 'Fault-Tolerant Computer System Design', p.6-7, Prentice-Hall, Inc., New Jersey, 1996.

[7] 신진범, 김상하, "ASR 기법을 적용한 임무지향 교전통제 컴퓨터의 신뢰도 분석" *정보처리학회논문지*, 제15-A권, 제6호, pp.309-316, 2008. 12.

[8] 유명환, 배정일, 신진화, 조길석, "UML 2.0 모델 기반의 교전통제 소프트웨어 아키텍처 개발" *한국군사과학기술학회지*, 제10권, 제4호, pp.20-29, 2007. 12.

[9] Liming CHEN, Algirdas AVIZIENIS, "N-Vers-ion Programming : A Fault-Tolerant Approach To Reliability Of Software Operation," in Proc. of the *IEEE Fault Tolerant Computing Systems*, Vol.III, pp.113-119, 1996.

[10] Thomas Huining Feng, Edward A. Lee, "Real-Time Distributed Discrete-Event Execution with Fault Tolerance," *in Proc of IEEE Real-Time and Embedded Technology*, Volume 00, pp.205-214, April, 2008.

[11] Paul Rubel et al, "Fault Tolerant Approaches for Distributed Real-time and Embedded Sys-tems," *In: IEEE Military Communications Conference*, pp.1-8, October, 2007.

[12] Manel Velasco et al, "A control approach to bandwidth management in networked control systems," *In: 30th Annual Conference of IEEE Industrial Electronics Society*, Vol.3, pp.2343-2348, November, 2004.

[13] K. H.(Kane) Kim, "Toward Globally Optimal Resource Management in Large-Scale Real Time Distributed Computer Systems," in Proc. of the *IEEE CS 6th Workshop on Future Trends of Distributed Computing Systems*, pp.248-255, October, 1997.

[14] K. H.(Kane) Kim, "Middleware of Real-Time Object Based Fault Tolerance Distributed Computing Systems: Issues And Some Appr-oaches," in Proc. of the *Pacific Rim Int'l Symp. on De-pendable Computing*, pp.3-8, December, 2001.

[15] http://www.cwcembedded.com/p/5/3/75.html

[16] http://www.ccii.co.za/products/blp_pmc/pmc_fc-1gbps.html

[17] Norman B. Fuqua, 'Reliability Engineering for Electronic Design', pp.135-137, MARCEL DEKKER, INC., New York, 1987.

[18] Norman B. Fuqua, 'Reliability Engineering for Electronic Design', pp.133-135, MARCEL DEKKER, INC., New York, 1987.

[19] Dhiraj K. Pradhan, 'Fault-Tolerant Computer System Design', pp.70-73, Prentice-Hall, Inc., New Jersey, 1996.

신 진 범

e-mail : espinosa@hanafos.com
1987년 부산대학교 전자공학(석사)
1987년~현 재 국방과학연구소 책임연구원
관심분야 : 실시간시스템, 컴퓨터구조, 임베디드시스템 및 네트워크

김 상 하

e-mail : shkim@cnu.ac.kr
1980년 서울대학교(석사)
1986년 미국 휴스턴대학교(석사)
1989년 미국 휴스턴대학교(박사)
1990년~1992년 한국과학기술연구원 선임연구원
1990년~현 재 충남대학교 전기정보통신공학부 교수
관심분야 : 컴퓨터구조, 임베디드 시스템, 정보통신, 컴퓨터네트워크