Hot Standby 고장 감내 구조를 지원하는 교환 제어시스템의 가동률 분석

송 광 석† 여 환 근† 한 창 호†† 문 태 수††
유 충 렬†† 이 광 배†† 김 현 욱†† 윤 충 화†††

요 약

본 논문에서는 높은 시스템 가용성을 제공할 뿐만 아니라 고장 발생시 데이타 손실이 없는 두 가지 hot standby 구조를 제안한다. 제안된 hot standby 구조의 성능을 평가하기 위해서 hot standby 구조로부터 동기 unit 만을 제거한 warm standby 구조를 고려하였으며, 각 구조에 대한 시스템 불가동률을 마르코프 상태도를 이용하여 구한 후 서로 비교 평가하였다. 본 연구 결과, 대부분의 경우 hot standby 구조가 warm standby 구조보다 훨씬 높은 가동률을 제공하였으며, 외부 동기 unit로 구성된 hot standby 구조가 내부 동기 unit로 구성된 hot standby 구조보다 항상 약간 높은 시스템 가동률을 유지하였다. 각 구조에 대한 active set time과 personnel recovery rate는 시스템 가동률에 거의 영향을 미치지 않았다. 그러나 data recovery time이 임계치 이상으로 증가하는 경우에는 hot standby 구조와 warm standby 구조 모두 시스템 가동률이 급격히 저하되었고 특히 hot standby 구조의 성능 저하가 심하여 궁극적으로 warm standby 구조보다 시스템 가동률이 떨어지는 현상을 보였다.

An Availability Analysis Of Switching Control System with Hot Standby Fault Tolerant Architecture

Kwangsuk Song† Hwangeun Yeo† Changho Han†† Taesoo Moon††
Choongyeol You†† Kwangbae Lee†† Hyenug Kim†† and Chunghwa Yoon†††

ABSTRACT

In this paper, we propose two hot standby architectures which not only provide high system availability but also lose little data on fault occurrence. In order to evaluate the performance of the proposed hot standby architectures, the warm standby architecture which is made from the hot standby architecture by eliminating its synchronization unit is considered. After system unavailability for each architecture is computed by using the corresponding Markov state diagram, the results are compared and evaluated. As the results, in most cases, hot standby architectures have higher availability than warm standby architecture. Also, hot standby architecture with external synchronization unit always maintains a little higher availability than hot standby architecture with internal synchronization unit. Active set time and personnel recovery rate for each architecture have little effect on system availability. However, in the case that data recovery time is too long, system availabilities of hot standby architectures and warm standby architecture degrade rapidly. In this case, the performance degradation of hot standby architectures is severe, and system availabilities of hot standby architectures eventually become lower than system availability of warm standby architecture.

1. 서 론

오래 전부터 시스템 개발의 주요 관심사가 되었었던 신뢰도는 반도체 소자가 개발되고 그 집적도가 높아지면서 단위 기능당 신뢰도가 향상되었다. 그러나, 오늘날 향상된 기능에 대한 요구에 부응하기 위한 시스템에서의 복잡도의 증가가

†정 회 원 : 한국전자통신연구소
††정 회 원 : 명지대학교 전자공학과
†††정 회 원 : 명지대학교 컴퓨터공학과
논문접수 : 1995년 8월 29일, 심사완료 : 1995년 11월 2일

거의 단위 기능당 신뢰도의 증가 속도에 이르므로, 전체 시스템 차원에서 신뢰도 문제는 여전히 문제가 될 수 밖에 없다. 또한, 예전에는 시스템의 고장이 경제나 사람의 생명에 치명적인 타격을 줄 수 있는 군사, 산업, 우주 항공 등과 같은 특수한 분야에서만 높은 신뢰도가 요구되었으나, 요즘은 상용분야에 까지 높은 신뢰도를 지닌 시스템이 요구되고 있는 실정이다. 이러한 높은 신뢰도에 대한 요구를 만족시키기 위한 것이 바로 고장 감내형 시스템이다. 고장 감내형 시스템이라 함은 그것을 구성하는 하드웨어나 소프트웨어에서 고장이 발생하더라도 계속해서 주어진 작업을 올바로 수행하도록 설계된 시스템을 의미한다[1~7].

오늘날은 기존의 협대역 ISDN이 BISDN으로 진화하면서 초고속 통신망에서의 통신방식이 등장하게 되었다. 이러한 BISDN은 광대역 전송 및 교환기술을 토대로 집중 또는 이산되어 있는 가입자 및 서비스 제공자들을 연결하여 수 bps에서 수백 Mbps에 이르는 폭넓은 대역 분포를 갖는 각종 서비스들을 종합적으로 제공하는 디지탈 통신망을 의미한다. 이러한 방식의 기본 취지는 성숙한 고속전송, 교환, 신호 처리, 컴퓨터, 소프트웨어, 소자 기술들을 활용하여, 각종 광대역 서비스들을 통합해서 제공할 수 있는 디지탈 망을 구축하는 데 있다. BISDN의 기본 취지를 실현하기 위한 방안으로서 비동기식 전송 모드를 사용하여야 하는데, 비동기식 전송 모드란 각종 서비스들을 잘라서 똑같은 크기의 비동기식 전송 모드의 셀에 매핑시킨 후, 비동기식 시분할 다중화를 통해서 전달하는 통신방식을 의미한다. 이러한 고속이면서도 대용량인 실시간 전송을 하는 과정에서 시스템에서의 약간의 에러나 고장은 엄청난 손실을 가져오는 결과를 발생시킬 수 있으므로 약간의 에러시에도 시스템의 작동을 중지시키지 않고 올바른 동작을 계속 수행할 수 있는 고장 감내형 교환 시스템의 개발이 요구되고 있다[9].

본 논문에서는 이러한 요구에 부응하기 위해서 높은 가용성을 제공할 뿐만 아니라 고장 발생시 데이타 손실을 최소화시킬 수 있는 교환 제어 시스템의 고장 감내 구조로서 hot standby 구조를 제안하고 그에 대한 신뢰도를 마르코프 상태 모델을 이용하여 비교 평가하였다.

본 논문의 2장에서는 기존 교환 시스템에서 사용되고 있는 기본 고장 감내 기법을 간단하게 살펴본 후 3장에서 제안된 hot standby 고장 감내 구조와 그 동작 방식을 설명한다. 4장에서는 hot standby 구조에 대한 마르코프 모델을 제시하고 그 모델에 근거하여 불가동률을 비교 평가한다. 마지막으로 5장에서는 결론 및 향후 연구 과제에 대하여 논한다.

2. 기존 교환 제어 시스템에서의 고장 감내 기법

교환 제어 시스템은 대부분의 고장 감내 컴퓨터 시스템이 요구하는 고성능 및 고신뢰성뿐만 아니라 시스템에 고장이 발생한 경우에도 시스템이 계속 올바른 동작을 수행하는 고가동률을 만족시켜야만 한다[15]. 그러한 고가동률 조건을 충족시키기 위해서 오늘날의 교환 제어 시스템은 standby sparing 기법을 기본 고장 감내 기법으로 채택하고 있다. 또한, 시스템의 비용 및 복잡도를 고려하여 대부분의 교환 제어 시스템은 이중화 보드 구조로서 이루어져 있다. standby sparing 기법은 크게 세가지 형태로 나눌 수 있으며 이는 다음과 같다. cold, warm 그리고 hot standby 기법.

cold standby sparing 기법에서, standby 보드는 active 보드상에 고장이 감지되었을 때 cold start한다. 즉, 평상시 standby 보드의 전원은 꺼져있다가 active 보드상의 고장 감지시 전원 공급과 더불어 standby 보드의 작동을 개시한다. 그러므로 이 기법은 고장 발생시에도 고가동률을 요구하는 교환 제어 시스템에는 적합하지 않다.

warm standby sparing 기법에서는 정상 동작시 active 보드만이 주어진 업무를 맡아 수행하며 standby 보드는 전원이 공급된 상태에서 고장 발생시 대체되기 위해서 대기상태에 있다. 이때, 고가동률 및 고신뢰성을 유지하기 위해서 active 보드와 standby 보드내의 메모리 내용은 서로 일치하도록 해야 한다. active 보드와 standby 보드의 메모리 내용을 정상 동작 중에

항상 상호 일치시키기 위해서, active 보드는 메모리 내용이 변경되어야만 할때 message passing 방법이나 concurrent writing 방법을 사용한다. message passing 방법에서는 active 보드가 변경될 데이타를 message 형태로 standby 보드에 전송하며, concurrent write 방법에서는 active 보드가 데이타 변경시 그 자신의 메모리와 standby 보드의 메모리를 동시에 변경시킨다. message passing 방법은 시스템 하드웨어에 부담을 주지 않는 대신 소프트웨어 오버헤드가 크다. 반면에 concurrent write 방법은 소프트웨어 오버헤드가 거의 없으나 하드웨어 부담이 존재한다. active 보드상에서의 고장 발생이 감지될 때 standby 보드는 필요한 조치를 취한 후 active 보드로서 동작하게 되며 이때 중지되었던 업무나 다음 업무를 계속 수행하게 된다.

hot standby sparing 기법에서는 단지 active 보드만이 I/O 채널을 통한 외부 요청에 응답하는 것 외에 active 보드와 standby 보드 모두가 동기를 맞추면서 일을 동일하게 수행해 나간다. 이 기법은 standby 보드도 active 보드와 동일한 동작을 수행하고 있기 때문에 고장 발생 감지 즉시 standby 보드가 active 보드로서 동작하게 함으로써 시스템 가동률을 높일 수 있으나 해결해야 할 두가지 중요한 문제점을 지니고 있다. 첫째는 active 보드와 standby 보드간에 동기 유지 문제이다. active 보드와 standby 보드간에 빈번한 동기화는 시스템 오버헤드로서 궁극적으로 심각한 시스템 성능 저하를 야기시킨다. 동기화는 micro-instruction, instruction, process 레벨 등에서 수행될 수 있으나 이중화 보드 구조를 갖는 교환 제어 시스템에서는 지나친 시스템 성능 저하를 피하기 위해서 process 레벨의 동기화가 사용되어야 한다. 두번째는 고장 발생시 고장 복구 동작과 고장 복구 완료시 다시 원래의 hot standby 정상 동작 상태로 전이시키는 문제이다. 본 논문에서는 두번째 문제의 해결 방안을 제시하였으며, 첫째 문제를 해결하기 위해서는 앞으로 많은 연구가 필요하다. 현재 국내 TDX 교환 시스템과 AT&T 사의 ESS 교환 시스템은 이중화 보드 구조상에서 warm standby sparing 기법의 concurrent write 방법을 채택하여 사용하고 있다[8~14].

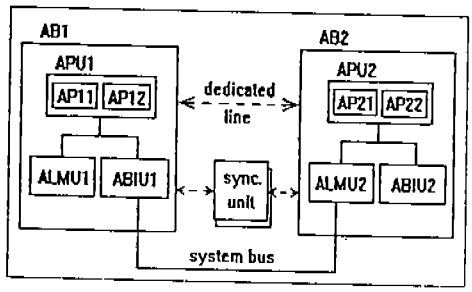
3. 제안 hot standby 고장 감내 시스템

본 연구에서 제시하는 시스템 구조들에 대해서 프로세서(processor)는 오류 체크(check) 기능을 내장하고 있지 않으며, 메모리(memory)와 버스 인터페이스(bus interface)는 오류 체크(check) 기능을 내장하고 있고, 버스 라인(bus line)은 에러가 발생하지 않는다고 가정하였다.

시스템의 고장(fail)은 하드웨어 고장과 소프트웨어 고장 및 사용자 작동 미숙으로 인한 고장으로 이루어진다. 오늘날 반도체 기술 및 제반 기술의 발달로 인해 하드웨어 고장 발생률이 다른 고장 발생률보다 상대적으로 작으나 본 논문에서는 하드웨어 고장에 대해서만 언급하기로 한다.

3.1 외부 동기 unit로 구성된 hot standby 시스템

외부 동기 unit로 구성된 hot standby 구조는 (그림 1)에서 보는 바와 같이 이중화 동기 unit이 외부에 존재하며 이중화 보드 AB1과 AB2가 정상 동작시 모두 active하게 동작한다. 또한 두 보드간에 시스템 버스와 전용 line은 정상 상태에서 사용되지 않는다. 또한 이 구조에서는 두 보드가 모두 active하게 동작하므로 보드내의 component가 고장 발생 감지시 이 사실을 상대



AB : active board
AP : processor
ABIU : bus interface unit
APU : processor unit
ALMU : local memory unit

(그림 1) 외부 동기 Unit로 구성된 Hot Standby 구조
(Fig. 1) Hot Standby Architecture with External Sync. Unit

편 보드에게 전용 line을 통하여 알려주는 동작만이 필요하다. 즉 고장 발생시 필요한 정보를 상대편 보드에 전송할 필요가 없다. 그러므로 고장 감지를 위해 APU는 두개의 프로세서를 필요로 하며 ALMU와 ABIU는 고장 감지 기능을 내포하고 있으므로 단일 unit 만을 사용하면 된다. 이 구조에서 power 고장이 감지되는 시기는 동기 동작 중에 감지될 수 있으므로 warm stand-by 구조에서와 같이 power 고장을 감지하기 위해 전용 line을 통한 정기적인 정상 상태 확인 동작을 수행할 필요가 없다.

이 시스템은 정상 동작 중에 데이타 읽기(read) 동작시 AB1의 APU1(AP11과 AP12 모두)은 ALMU1으로 부터 데이타를 읽으며 AB2의 APU2(AP21 과 AP22 모두)는 ALMU2로부터 데이타를 읽는다. 데이타 쓰기(write) 동작시 APU1의 AP11은 ALMU1으로, APU2의 AP21은 ALMU2로 변경될 데이타를 각각 보낸다.

AB1 보드와 AB2 보드는 정상 동작시 서로 똑같이 동작하여야 하므로 두 보드간에 동기 작업은 필수적으로 요구된다. 두 보드들 간에 동기화는 micro-instruction 레벨, instruction 레벨, process 레벨 단위로 수행될 수 있는데 micro-instruction 레벨과 instruction 레벨 단위의 동기화는 빈번한 동기화 동작으로 인해 시스템의 성능을 매우 저하시킨다. 그러므로 process 레벨 단위의 동기화가 본 연구에서 제안한 시스템에서 사용된다.

외부 시스템으로부터의 데이타 입력이 발생할 경우 AB1 과 AB2 보드는 현재 수행하던 일을 일시 중단하고 외부 입력 데이타를 받아들이기 위해 동기 unit에 각각 동기 생성 신호를 보낸다. 이때 동기 unit은 두 보드로 부터의 동기 생성 신호가 모두 도달할 때까지 대기하다가 두 동기 생성 신호를 모두 받으면 동기 응답 신호를 두 보드에 동시에 보내며, 동기 응답 신호를 받은 두 보드는 각각 대기중인 외부 입력 데이타를 그 자신의 보드로 읽어들인다. 두 동기 생성 신호가 동기 unit에 도달하는 시간의 격차가 임계치보다 큰 경우 동기 unit는 아직 동기 생성 신호를 보내지 않은 보드에서 power 고장이 발생한 것으로 간주하여 그 에러 신호를 두 보드에 동시에 보내게 된다. 외부 시스템으로의 데이타 출력시에도 데이타 입력시와 동일하게 동작한다.

power 고장 발생은 동기 unit 상에 한 보드로부터 동기 생성 신호가 도달했으나 다른 보드로부터 임계시간이 지나도록 동기 생성 신호가 도달하지 못한 경우에 감지된다. 이때 동기 unit는 두 보드에게 power 고장의 발생 사실을 통보한다. 두 보드에게 power 고장 사실을 통보하는 이유는 이때 발생한 에러가 power 고장뿐만 아니라 동기 unit로 동기 생성 신호를 보내는 회로의 고장에 기인할 수도 있기 때문이다. component 고장시 시스템내의 각 보드의 component는 고장 감지 기능을 내포하고 있기 때문에 고장 발생 즉시 그와 관련된 에러 신호가 즉시 전용 line을 통해 상대편 보드에 전송된다. 이때 고장이 발생하지 않은 보드는 동기화 작업을 중지시키고 외부 시스템에 고장 발생 사실을 통보한 후 single 보드와 같이 동작하게 된다. 고장 발생시 고장나지 않은 보드도 active하게 동작 중이었으므로 이때 데이타 손실은 발생하지 않는다.

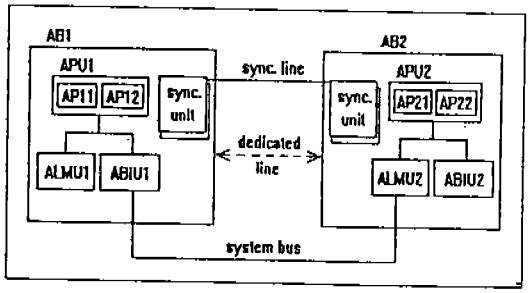
한편 고장난 보드는 상대편 보드가 single 보드로서 동작하게 된 후 고장 복구 동작에 들어간다. 먼저 고장난 하드웨어에 대한 자체 복구를 시도하고 불가능한 경우에는 수리 요원에게 복구를 요청한다. 고장난 보드는 하드웨어를 수리한 후 현재 active 하게 동작 중인 보드에게 전용 line을 통해 그 사실을 통보하고 그 자신의 메모리 내용을 상대편 보드의 메모리 내용과 일치시킬 준비를 갖춘다. 이때 현재 active하게 동작 중인 보드는 그 자신의 메모리 내용을 수리 완료된 보드에게 전송하며 그 동작은 다음과 같다. 현재 active하게 동작 중인 보드에게 주어진 일이 있는 경우에는 그 일을 먼저 수행하며, 만약 데이타 변경이 요구되는 경우에는 concurrent write 방법을 이용하여 그 자신의 메모리 내용과 상대편 보드의 메모리 내용을 동시에 변경시킨다. 이때 데이타 전송은 시스템 버스를 통해서 이루어진다. 만약 active하게 동작 중인 보드에게 주어진 일이 없다면 active 보드는 그 자신의 메모리 내용을 시스템 버스를 통해 상대편 보드의 메모리로 전송한다. 수리 완료 보드의 메모리

내용이 active 보드의 메모리 내용과 동일하게 된 후 두 보드간에 재동기화가 이루어지며, 이때 전체 시스템은 정상 hot standby 상태로 복귀된다.

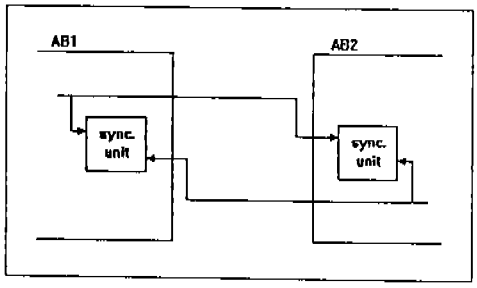
3.2 내부 동기 unit로 구성된 hot standby 시스템

앞에서 언급한 외부 동기 unit로 구성된 hot standby 구조는 이중화 보드 외부에 동기 unit이 존재하므로 교환 제어 시스템의 하드웨어 제작 및 운영상 부가적인 문제를 야기시킨다. 그러므로 기존의 교환 제어 시스템의 이중화 보드 구조를 유지시키기 위해서 (그림 2)에서 처럼 각 보드내에 동기 unit을 내장시키는 hot standby 구조를 제시하였다. 동기 unit과 관련된 좀 더 상세한 구조는 (그림 3)에 나타내었다. 이 구조에서는 두 보드간에 동기를 정확히 일치시키는 일이 앞에서 언급한 구조보다 다소 복잡하며, 하드웨어 소자가 증가하는 단점도 지니게 된다.

내부 동기 unit로 구성된 hot standby 구조에서의 정상 상태 동작과 고장 발생시 복구 동작은 외부 동기 unit로 구성된 hot standby 구조와 동일하나, 그 가동률에 대한 마르코프 상태 모델은 서로 다르다.



(그림 2) 내부 동기 Unit로 구성된 Hot Standby 구조
(Fig. 2) Hot Standby Architecture with Internal Sync. Unit



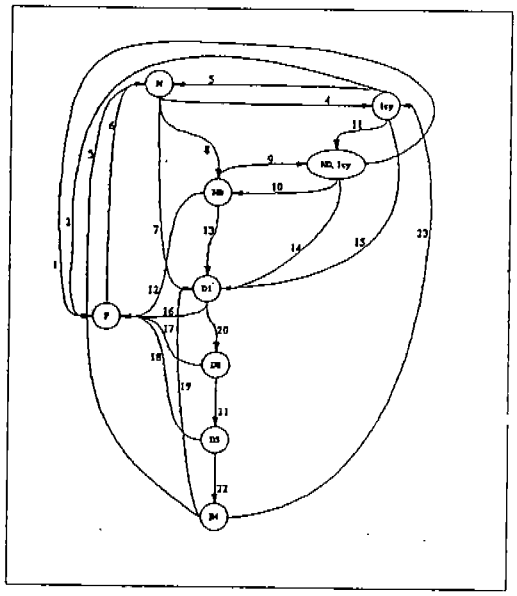
(그림 3) 동기 Unit과 동기 Line
(Fig. 3) Sync. Unit and Sync. Line

4. 성능 평가

4.1 시스템 불가동률

본 논문에서는 각 구조의 불가동률 계산시 마르코프 상태 모델을 이용하였으며, 이때 고장 감지 기능을 갖춘 하드웨어는 고장 발생 즉시 이를 감지할 능력이 있다고 가정하였다.

(그림 4)는 (그림 1)에서 도시한 외부 동기 unit로 구성된 hot standby 구조에 대한 마르코프 상태도를 보여준 것으로, 그림에서 N은 정상 상태, ND는 보드 상에 고장이 발생한 것을 감지하지 못한 상태, 1sy는 정상 상태에서 동기화 유닛에 고장이 발생한 상태, ND, 1sy는 보드 상에 고장이 발생한 것을 감지하지 못한 상태에서 동기화 유닛에 고장이 발생한 상태, D1은 보드 상에서의 고장을 감지한 상태, D2는 고장난 보드



(그림 4) 외부 동기 Unit로 구성된 Hot Standby 구조에 대한 마르코프 상태도
(Fig. 4) Markov State Diagram for Hot Standby Architecture with External Sync. Unit

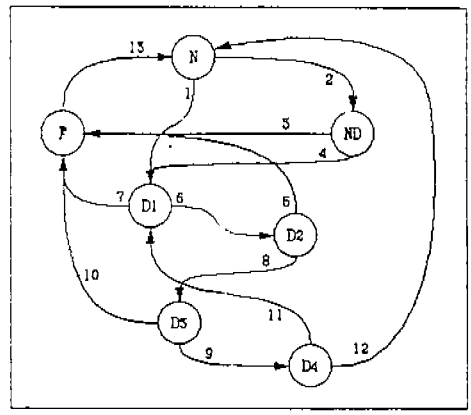
를 수리하고 정상 보드는 일중화 구조로서 동작하게 하는 상태, D3은 고장난 보드의 하드웨어가 완전 수리된 상태, D4는 hot standby 구조로 재가동 시키기 위해서 재동기하는 상태이며, F는 시스템이 완전히 고장난 상태를 의미한다. power 고장은 동기 동작시에 감지되므로 ND 상태로 전이하게 되며, 보드내의 그 이외의 고장은 즉시 감지되므로 시스템을 N 상태로부터 D1 상태로 전이시킨다.

〈표 1〉 그림 4에서의 Line 파라메터
〈Table 1〉 Line Parameter of Fig. 4

선 번호	파라메터
1	$\lambda h + \lambda sy$
2	λsy
3	$\mu h5$
4	$2\lambda sy$
5	μsy
6	$\mu h6$
7	$2\lambda h2$
8	$2\lambda h1$
9	$2\lambda sy$
10	μsy
11	$2\lambda h1$
12	λh
13	$\mu h1$
14	$\mu h1$
15	$2\lambda h2$
16	λh
17	λh
18	λh
19	λh
20	$\mu h2$
21	$\mu h3$
22	$\mu h4$
23	$\mu h5 \cdot 2\lambda sy$

〈표 1〉에서는 외부 동기 unit로 구성된 hot standby 구조에 대한 상태도를 보여주는 (그림 4)에서의 line 파라메터를 나타내었다. λh는 보드에서의 고장률이며 $\lambda h1$과 $\lambda h2$의 합을 의미하고, λsy는 동기화 유닛에서의 고장률, μh는 회복률을 의미한다.

(그림 5)는 (그림 2)에서 보여준 내부 동기 unit로 구성된 hot standby 이중화 보드 구조에 대한 마르코프 상태도를 보여준 것으로, 그림에서 N은 정상 상태, ND는 보드 상에 고장이 발



(그림 5) 내부 동기 Unit로 구성된 Hot Standby 구조에 대한 마르코프 상태도
(Fig. 5) Markov State Diagram for Hot Standby Architecture with Internal Sync. Unit

생한 것을 감지하지 못한 상태, D1은 보드 상에서의 고장을 감지한 상태, D2는 고장난 보드를 수리하고 정상 보드는 일중화 구조로서 동작하게 하는 상태, D3는 고장난 보드의 하드웨어가 완전 수리된 상태, D4는 hot standby 구조로 재가동 시키기 위해서 재동기하는 상태이며, F는 시스템이 완전히 고장난 상태를 의미한다. 이 경우에도 power 고장시에만 ND 상태로 전이하며, 그 이외의 보드 내부 고장은 시스템을 D1상태로 전이시킨다.

〈표 2〉 그림 5에서의 Line 파라메터
〈Table 2〉 Line Parameter of Fig. 5

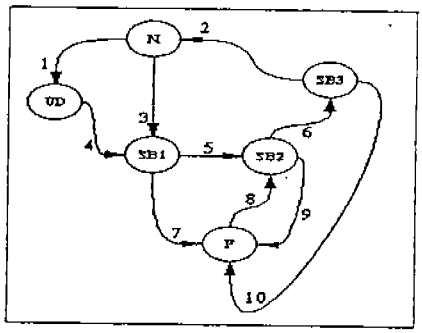
선 번호	파라메터
1	$2\lambda h2$
2	$2\lambda h1$
3	λh
4	$\mu h1$
5	λh
6	$\mu h2$
7	λh
8	$\mu h3$
9	$\mu h4$
10	λh
11	λh
12	$\mu h5$
13	$\mu h6$

〈표 2〉에서는 내부 동기 unit로 구성된 hot standby 이중화 보드구조에 대한 상태도를 보여

주는 (그림 5)에서의 line 파라메터를 나타내었다. λh는 보드에서의 고장률이며 $\lambda h1$과 $\lambda h2$의 합을 의미하고, μh는 회복률을 의미한다.

4.2 분석 및 고찰

본 연구에서는 제안한 두가지 hot standby 구조뿐만 아니라 hot standby 구조에서 동기 unit을 제거함으로써 구성된 warm standby 구조에 대한 불가동률을 마르코프 상태 모델을 이용하여 구한 후 서로 비교 평가하였다. 비교되는 warm standby 구조의 불가동률에 대한 마르코프 상태도와 선 파라메터는 (그림 6)과 〈표 3〉에 각각 나타내었다. (그림 6)에서 N은 전체 시스템의 정상 상태, UD는 active 보드상에 고장 발생을 감지 못한 상태, SB1은 active 보드상에 고장을 감지한 상태, SB2는 standby 보드가 active 보드로 전이된 상태, SB3는 고장난 보드의 하드웨어가 완전히 수리된 상태, F는 시스템이 완전히 고장난 상태를 의미한다. 〈표 3〉에서 λ는 시스템에서의 고장률로서 $\lambda 1$과 $\lambda 2$의 합이며, μ는 시스템 회복률을 의미한다.



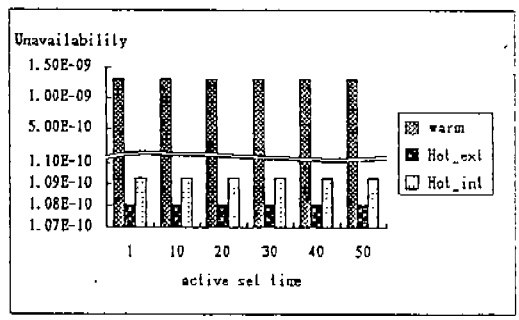
(그림 6) Warm Standby 구조에 대한 마르코프 상태도
(Fig. 6) Markov State Diagram for Warm Standby Architecture

〈표 3〉 그림 6의 Line파라메터
(Table 3) Line Parameter of Fig. 6

선 번호	파라메터
1	$\lambda 1$
2	$\mu 4$
3	$\lambda 2$
4	$\mu 1$
5	$\mu 2$
6	$\mu 3$
7	λ
8	$\mu 5$
9	λ
10	λ

현재 active하게 동작 중인 한 보드에서 고장이 발생한 경우, hot standby 구조에서는 상대편 보드도 독립적으로 active하게 동작 중이었으므로 고장 발생으로 인한 데이타 손실이 정상 동작 중인 보드에 발생하지 않는다. 그러므로, 이 경우에 hot standby 구조에서는 데이타 복구 동작이 필요없다. 반면에 warm standby 구조에서 active 보드에 고장이 발생한 경우, active 보드가 concurrent write 방식으로 standby 보드의 메모리 내용을 active 보드의 메모리 내용과 상호 일치시키는 중이었으므로 standby 보드의 상태는 고장이 발생한 active 보드에 의해 영향을 받는다. 즉, 프로세서 또는 local 메모리 고장시에는 거짓 데이타가 concurrent write 동작에 의해 standby 보드의 메모리에 저장될 수 있으며, 본 논문의 구조에서는 이러한 경우 active 보드의 프로세서 관련 정보를 standby 보드에 전송할 수 없으므로 recovery log file의 이전 check-point 위치까지의 데이타 복구 동작이 요구된다. 한편 warm standby 구조에서 active 보드상에 power 고장이 발생한 경우 교환 제어 시스템에서 요구하는 실시간 처리를 만족시킬 수 없으므로 power 고장시 수행하던 일은 손실된다.

(그림 7)은 현재 active하게 동작 중인 보드에 고장이 발생한 경우 상대편 보드가 데이타 복구 동작을 수행하는 데 걸리는 시간(active set time)의 변화에 대한 각 구조의 불가동률을 도시하였다.

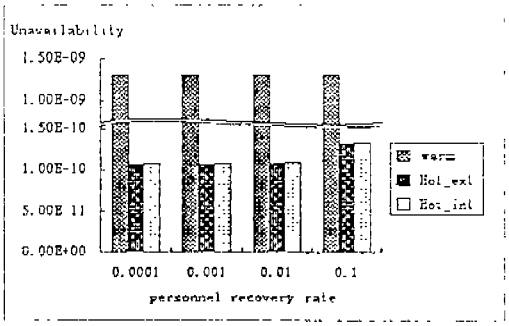


(그림 7) Active Set Time에 대한 불가동률
(Fig. 7) Unavailability vs Active Set Time

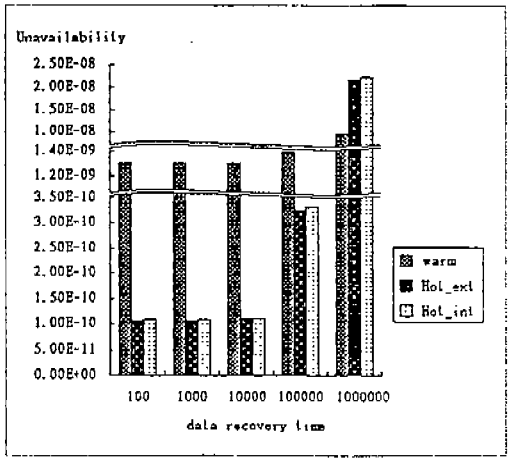
데이타 복구 동작은 warm standby 구조에서는 대부분의 경우 요구되나 hot standby 구조에서는 필요가 없으므로 그림에서 나타내었듯이 이러한 경우 hot standby 구조가 warm standby 구조보다 시스템 가동률이 훨씬 뛰어남을 알 수 있다. 또한, 동기 unit이 외부에 존재하는 hot standby 구조가 동기 unit이 내부에 존재하는 hot standby 구조보다 시스템 가동률이 약간 높았다.

(그림 8)은 현재 active하게 동작 중인 보드에 고장이 발생한 경우 고장난 하드웨어를 수리 요원이 직접 수리할 확률(personnel recovery rate)에 대한 각 구조의 불가동률을 보여주고 있다.

warm standby 구조의 시스템 불가동률은



(그림 8) Personnel Recovery Rate에 대한 불가동률
(Fig. 8) Unavailability vs Personnel Recovery Rate



(그림 9) Data Recovery Time에 대한 불가동률
(Fig. 9) Unavailability vs Data Recovery Time

personnel recovery rate에 거의 영향을 받지 않으나 hot standby 구조에서는 personnel recovery rate가 임계치 이상 증가하면 시스템 가동률이 저하됨을 알 수 있다. 또한 이 경우에도 hot standby 구조가 warm standby 구조보다 시스템 가동률이 훨씬 높음을 알 수 있으며, 두 hot standby 구조간에 시스템 가동률은 거의 동일하다.

(그림 9)는 고장난 보드가 하드웨어 수리를 완료한 후 현재 active하게 동작 중인 보드의 메모리 내용과 그 자신의 메모리 내용을 상호 일치시키는 데 걸리는 시간(data recovery time)에 대한 각 구조의 시스템 불가동률을 보여주고 있다.

그림에서 알 수 있듯이 data recovery time이 10000 이하인 경우에는 warm standby 구조나 hot standby 구조에서의 불가동률이 거의 변화가 없으나, 10000 이상에서는 hot standby 구조에서의 가동률 저하가 warm standby 구조에서의 가동률 저하보다 훨씬 빠른 속도로 이루어지고 있다. 그리고 data recovery time이 1000000 이상인 경우에는 hot standby 구조의 가동률이 warm standby 구조의 가동률보다 낮아진다.

5. 결 론

본 논문에서는 높은 가용성을 제공할 뿐만아니라 고장 발생시 데이타 손실이 없는 두가지 hot standby 구조와 동작 방법을 제안하고, 마르코프 상태도를 이용하여 시스템 불가동률을 구한 후 그 성능을 비교 평가하였다. 제안된 외부 동기 unit로 구성된 hot standby 구조는 하드웨어적인 동기화가 비교적 간단하나 현재 상용화되어있는 이중화 보드 교환 제어 시스템의 하드웨어 제작과 운영상 많은 수정을 요구하므로 이 문제를 해결하기 위해서 동기 unit을 보드 내부에 포함시키는 hot standby 구조를 별도로 제안하였다. 그러나 동기 unit이 보드 내부에 포함되는 hot standby 구조는 하드웨어적인 동기화가 좀 더 복잡해지고 하드웨어 소자도 증가하는 단점을 지니고 있다.

시스템 성능 평가시 hot standby 구조의 가동

률을 평가하기 위해서 그 구조에서 동기 unit 만을 제거하여 구성한 warm standby 구조에 대한 가동률도 계산하였으며, 비교 결과는 다음과 같다. 대부분의 경우 hot standby 구조가 warm standby 구조보다 훨씬 높은 가동률을 제공하였으며, 외부 동기 unit로 구성된 hot standby 구조가 내부 동기 unit로 구성된 hot standby 구조보다 항상 약간 높은 가동률을 유지하였다. active set time은 언급된 구조들에 대한 가동률에 영향을 미치지 않았으나, personnel recovery rate가 임계치 이상 증가하면 hot standby 구조의 시스템 가동률이 저하됨을 알 수 있었다. data recovery time이 임계치 이상 증가하는 경우 hot standby 구조와 warm standby 구조의 가동률은 저하되며, 특히 hot standby 구조는 warm standby 구조보다 훨씬 급격한 가동률 저하 현상을 나타내었다. hot standby 구조에서 전체 시스템의 성능 향상을 위하여 데이타 캐쉬 메모리의 사용을 고려하는 경우, 캐쉬 메모리의 사용이 시스템 고장 발생시 data recovery time을 증가시키므로 캐쉬메모리를 사용하고자 하는 경우 시스템 가동률이 시스템 허용기준을 만족시키는 지 검토하여야 한다.

본 연구 결과, 가동률과 데이타 손실 및 시스템 운영 면에서 제안된 내부 동기 unit로 구성된 hot standby 구조가 향후 개발될 교환 제어 시스템에 대한 최적 후보라 사료되며, 향후의 연구 과제로서 빈번한 동기화로 인해 전체 시스템 속도가 저하되는 hot standby 구조의 문제점을 해결함으로써 미래의 초고속 통신망용 교환 제어 시스템 구조를 개발하고자 한다.

참 고 문 헌

[1] D. P. Siewiorek, "Architecture of fault-tolerant computers : an historical perspective," Proc. IEEE, Vol. 79, No. 12, pp. 1710~1734, Dec. 1991.

[2] P. Nelson and B.D. Carroll, Tutorial : Fault-Tolerant Computing, IEEE Computer Society Press, 1987.

[3] B. W. Johnson, Design and analysis of fault-tolerant digital systems, Addison-Wesley Pub. Co., 1989.

[4] Randell, B., "System Structure for Software fault tolerance," IEEE Trans. on Software Engr., pp. 220~232, June 1975.

[5] Laprie, J-C, et al., "Definition and Analysis of Hardware and Software Fault-Tolerant Architectures," Computer, pp. 39~51, July 1990.

[6] Shem-Tov Levi and Ashok K. Agrawala, Fault Tolerant System Design, McGraw-Hill, 1994.

[7] A. L. Hopkins, T. B. Smith, and J. H. Lala, "FTMP-A highly reliable fault-tolerant multiprocessor for aircraft," Proc. IEEE, Vol. 66, pp. 1221~1239, Oct. 1978.

[8] W. N. Toy, "Fault-Tolerant Design of Local ESS Processors," Proceedings of the IEEE, pp. 1126~1145, Oct. 1978.

[9] Daniel P. Siewoirek and Robert S. Swarz, Reliable Computer Systems, Digital Press, 1992.

[10] R. M. Wolfe and N. A. Martellotto, "Telecommunications Data Base Application with the 3BTM20 Processor," ISS '84 Florence, 7-11 May 1984..

[11] Frank, R. J., Siegel, El H. Jr. and Watters, R. J., "Attached Processor for 4ESSTM and 1A ESSTM Switches," ISS ' 84 Florence, 7-11 May 1984.

[12] Jan-Erik Andersson, Per-Erik Gustafsson and Lennart Soderberg, "New Operation and Mainternace Functions in AXE10," Erricsson Rev. No. 3, pp. 104~111, 1986.

[13] Thomas Edsbacker, "New Regional Processor for AXE10," Ericsson Rev. No. 3, pp. 112~118, 1986.

[14] Ingmar Jonsson, "Control System for AXE10," Ericsson Rev. No. 4, pp. 146~155, 1984.

[15] G. Fantauzzi, Digital Switching Control Architectures, Artech House, Inc., 1990.

송 광 석

- 1979년 고려대학교 전자공학과 졸업(학사)
- 1981년 고려대학교 대학원 전자공학과(석사)
- 1992년 고려대학교 대학원 전자 공학과(박사)
- 1992년 Georgia Institute of Technology 객원연구원
- 1982년~현재 한국전자통신연구소 제어시스템연구실장
- 관심분야 : Fault Tolerant Control System, Computer Architecture, ATM Switching System

여 환 근

1981년 경북대학교 전자공학과 졸업(학사)
1983년 경북대학교 대학원 전자공학과(석사)
1992년~현재 경북대학교 대학원 컴퓨터공학과 박사과정
1993년~94년 U. of Maryland SRI 객원 연구원
1983년~현재 한국전자통신연구소 제어시스템연구실 선임연구원
관심분야 : Fault Tolerant Computing System, Distributed Computer Architecture, ATM Switching System

이 광 배

1979년 고려대학교 전자공학과 졸업(공학사)
1979년~81년 고려대학교 전자공학과 졸업(공학석사)
1981년~82년 삼성반도체연구소
1982년~83년 금성 연구소
1984년~86년 Univ.4of Southern Califonia, Computer Engineering전공(공학석사)
1986년~91년 Arizona State Univ., Electronical Engineering전공(공학박사)
1992년~현재 명지대학교 전자공학과 조교수
1994년~현재 대한전자공학회 논문편집위원
관심분야 : 멀티미디어(영상 및 음성 신호처리), 병렬처리 및 고속 컴퓨터(prolog 방식), Communication System(고장 감내형)

한 창 호

1994년 명지대학교 전자공학과 졸업(공학사)
1994년~현재 명지대학교 전자공학과 석사 재학
관심분야 : ATM통신, 고장감내형 시스템 설계, 멀티미디어(영상 처리)

문 태 수

1989년 명지대학교 전자공학과 졸업(공학사)
1994년 명지대학교 전자공학과 졸업(공학석사)
1994년~현재 명지대학교 전자공학과 박사 재학
관심분야 : 컴퓨터 시스템, 네트워크 구조 및 설계, 데이타 통신

유 충 열

1989년 명지대학교 전자공학과 졸업(공학사)
1991년 명지대학교 전자공학과 졸업(공학석사)
1991년~현재 명지대학교 전자공학과 박사 재학
관심분야 : 컴퓨터 구조, 멀티미디어, 프로세서 설계, ASIC

김 현 욱

1978년 고려대학교 전자공학과 졸업(공학사)
1978년~80년 고려대학교 전자공학과 졸업(공학석사)
1980년~87년 고려대학교 전자공학과 졸업(공학박사)
1980년~81년 동양공업전문대학 전자과 전임강사
1981년~88년 명지대학교 전자공학과 전임강사, 조교수, 부교수
1988년~90년 Arizona State Univ., Computer Science전공(Adijunct Faulty)
1990년~현재 명지대학교 전자공학과 교수
1994년~현재 대한전자공학회 논문편집위원
관심분야 : 멀티미디어(영상 및 음성 신호처리), 병렬처리 및 고속 컴퓨터(prolog 방식), Communication System(고장 감내형)

윤 충 화

- 1979년 서울대학교 수학과 졸업(학사)
- 1984년 U. of Texas at Austin 전산학과졸업(석사)
- 1989년 루이지에나 주립대학 전산학과 졸업(박사)
- 1990년~현재 명지대학교 컴퓨터공학과 조교수
- 관심분야 : 신경회로망, 전문가 시스템, 성능평가