Warm standby sharing을 이용한 프로세서 이중화의 설계

구중두
호원대학교 컴퓨터게임학부
e-mail:ygslee@sunny.howon.ac.kr

Design of Processor Duplication using Extend Warm standby sharing

Jung Du Goo
Division of Computer Games, Howon University

요 약

이동통신시스템에서 RNC의 MCP는 호 처리를 담당하는 부분으로, 신뢰도와 실시간성이 요구된다. MCP는 높은 견고성을 갖도록 구현되지만 다소간의 오류 율(Fault late)은 존재할 수밖에 없으므로 프로세서를 이중화하여 활성화된 프로세서가 장애를 일으키더라도 대기중인 프로세서가 연속적인 서비스를 제공할 수 있어야 한다. Warm standby sharing에 비하여 Hot standby sharing은 데이터 손실이 없고 오류 데이터가 확산되지 않는 등의 다수의 장점을 갖지만 동기화 문제로 인하여 이를 시스템에 실제로 구현하는 것은 어렵다. 따라서 본 연구에서는 동기화의 장점에 데이터 손실 및 거짓 데이터의 확산 문제를 개선 함으로서, 실제 구현의 용이성 및 성능 향상이라는 결과를 얻으려 하였다.

1. 서론

임의의 한 모듈에서 결함이 발생하면, 결함 모듈을 시스템으로부터 제거하고 수행 중인 일을 계속 진행시킬 수 있으므로 결함 감지로부터 시스템 정상 기능 재가동까지 걸리는 시간이 극히 짧다. 또한 단일 시스템 결함으로부터의 데이터 손실이 발생하지 않는 결함 감내 구조의 설계가 가능하다. 반면에 Warm Standby에 비하여 정상 동작 중에 결함 감내 시스템 모듈간의 동기화 유지가 어렵고 결함에서 복구된 시스템 모듈의 재정상 가동 등의 구현의 어려움이 해결해야 할 큰 문제점으로 지적되고 있다 [1,3].

Warm standby 구조는 시스템 구현이 Hot standby 구조에 비해 훨씬 용이하나 결함의 종류와 정도에 따라서 다소의 데이터 손실이 발생하며 현재 동작중인 모듈상에 감지되지 못한 오류 데이터(Fault data)가 궁극적으로 전체 시스템으로 확산되는 문제점을 지니고 있다. 그러나 현재 개발되어 사용되고 있는 AT&T사의 ESS 5 교환기, ETRI의 TDX 10 등의 대부분의 교환기들은 결함 감내 구조로서 Warm standby sharing 기법을 적용하고 있다. 그 이유는 결함 감내 구조로서 Hot standby Sharing 기법을 사용하는 교환 시스템은 데이터 무손실 등 많은 장점을 가지고 있으나, 동기화의 복잡성으로 인하여 실시간 교환 시스템으로의 구현 시에 이론적인 성능을 얻는 것이 어렵기 때문이다[4,7].

본 문서에서는 기존의 Warm standby 구조가 갖는 데이터 손실 및 오류 데이터의 확산 문제를 해결함으로서 구현의 용이성과 구현 비용 그리고 성능의 향상이라는 결과를 얻고자 하였다. 실제로 시스템에 결함 포용을 실현함에 있어서, Hot standby가 갖는 이론상의 장점에서 벗어나 시스템으로의 구현이라는 문제에 직면하여 안전도와 신뢰도라는 측면 그리고 구현의 용이성이라는 측면을 비교하지 않을 수 없었다. 본 연구에서는 이중화 구조를 갖는 RNC에서 확장된 Warm standby sharing 결함 감내 구조를 구현함으로써 고성능 및 고가용성을 유지하는 시스템 개발에 기여하는 것을 목표로 하고 있다.

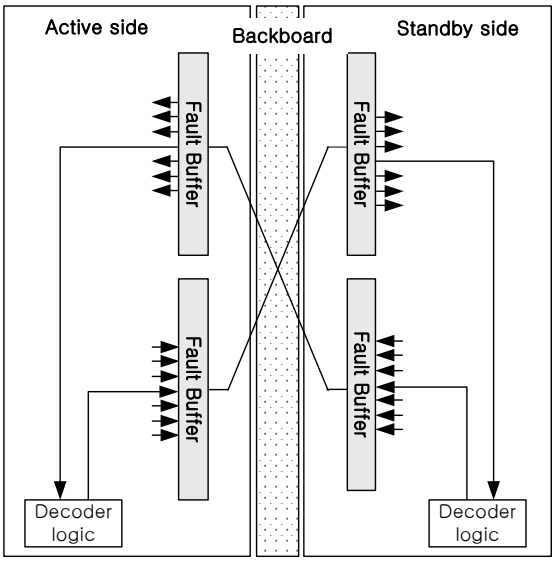
2. 확장된 Warm standby sharing

본 연구에서는 하드웨어적 측면과 소프트웨어적 측면에서 데이터 손실과 오류 데이터의 확산 문제를 해결하려 하였다. 하드웨어적 측면에서는 새로운 오

류 탐지 구조 제시하고 소프트웨어적 측면에서는 새로운 메시지 처리 구조를 제시한다.

2.1 하드웨어에 기반한 제어 구조

그림 1은 데이터 손실 및 오류 데이터의 전체 시스템으로의 확산 문제를 해결하기 위하여 본 연구에서 제안한 이중화 결함 제어 연결도이다. 본 구조에서는 두 B'd가 자신의 상태뿐만 아니라 상대 B'd의 상태를 조사 함으로서 오류 발생을 탐지하지 못할 가능성을 획기적으로 줄였다. 즉 자기 B'd의 상태를 주기적으로 조사할 뿐만 아니라 Back Plane을 통해서 이중화 관련 Fault signals을 TTL로 Tx/Rx함으로서 하드웨어적으로 상대 B'd 오류를 Monitoring한다. 이를 통하여 자기 B'd 또는 상대 B'd가 Active 또는 Standby 인지를 확인하고 소프트웨어에 의해 절체하게 된다.



[그림 1] 이중화 오류 제어 연결도

이중화 절체는 B'd 자체적으로 검출되는 내부 상태 정보와 이중화로 구성된 상대 B'd로부터 보고되는 상태정보를 종합하여 활성 상태를 결정한다. 이를 위하여 Memory mapped register로서, 상대 B'd의 상태가 하드웨어적으로 setting되는 MFSR(Mate Fault State Register)과 자기 B'd의 상태가 하드웨어적으로 설정되는 MSSR(My Fault Status Register)을 둔다.

2.2 소프트웨어에 기반한 제어 구조

Data backup 과 Recovery 시 데이터 손실 및 오류 데이터의 확산을 막기 위한 방안으로, 메시지의 삭제 및 재 구성을 위한 메시지 처리 구조를 제안한다. 이 구조는 EIMBuf(External Input Message Buffer), OEIMBuf(Ordered External Input Message Buffer) 그리고 STBYBuf(Stnadby Buffer)로 구성된다. Active side와 동일하게 전달된 IPC message는 SSM의 EIMBuf에 연결하고, OEIMBuf를 구성한다. IPC에서 전달된 메시지를 저장하고 있는 buffer인 EIMBuf 를 두고 AP id를 key로 갖는 hash table로 관리한다. 각 EIMBuf는 AP id당 하나씩 할당되며, 해당 AP id를 목적지(destination)로 갖는 IPC message를 STBYbuf를 통해 저장한다. 이때, IPC message는 IPC로부터 할당 받은 buffer인 IPCBuf를 그대로 사용하며, EIMBuf는 IPC message의 포인터만을 관리한다. OEIMBuf는 AP로의 Input injection order를 저장하기 위해 사용되는 buffer로서 AP의 우선순위에 의하여 바뀌는 메시지의 처리 순서를 일치시키기 위하여 STBYbuf를 Active side의 처리 순으로 pointing함으로서 전체 메시지를 관리한다. STBYBuf는 Standby side로 전송되었지만 수행되지 않은 IPC message에 대해 순서적으로 포인터를 저장한다.

TCB(Task Control Block)가 오류 확산의 완충작용으로 확산 문제를 해결한다. 메시지 처리 결과로 생성된 Backup data의 저장을 Active side로부터 최종 데이터가 전송될 때 까지 지연한 후, Standby side에 최종 데이터가 도착하면 해당 AP 메모리에 데이터를 저장한다. 마지막 데이터가 수신될 때까지 TCB에서 이 Backup data를 관리하고, 마지막 데이터가 수신되기 전에 Active side에 오류가 발생한다면 해당 데이터를 TCB에서 삭제한다. 이를 위하여, Backup data의 관리는 AP별로 수행되며, TCB를 구축해 백업될 영역에 대한 Start address와 size를 관리하며, Standby side의 Garbage collection을 위해 최근의 External input message에 대한 정보를 저장한다. AP의 Backup message 와 Checkpoint에 의하여 TCB에 등록되어 있는 일련의 Backup data를 이중화 path로 전달하기 위해 ASM은 일단 패킷(packet)의 크기(1024 byte)에 맞게 데이터를 분리하고, 이들이 재조합 가능하도록 Sequence number를 붙인다. 그리고 COM으로 전달한다. 이중화 path를 통하여 Backup data를 수신한 SSM은 데이터를 재구성하기 위해 Active side로부터 전달된 패킷을 ASM에서와 같이 이중화 모듈의 TCB를 이용하여

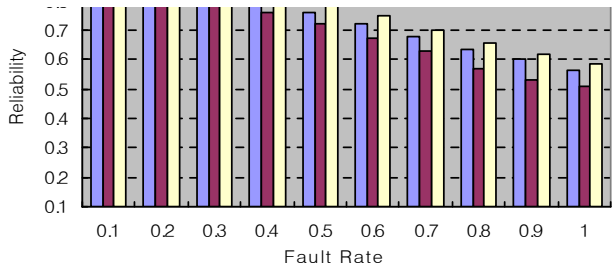
관리한다. 즉, SSM에서 데이터가 재구성되기 전까지 Active side로부터 전달된 패킷을 TCB에 매달아 놓는다. 그리고, 데이터의 마지막 패킷이 도착하게 되면, 해당 Backup data를 Start address와 동일한 메모리 공간에 저장 함으로서 동기를 유지한다. 만약, 마지막 패킷을 전달 받기 전에 절체가 된다면, 이 불완전한 데이터를 삭제 함으로서 오류 데이터의 확산을 방지한다. 이를 위하여 Data backup 과 Recovery 시 Data loss 및 Fault data의 확산을 막기 위한 방안이다.

3. 성능평가

본 시스템은 결함 검출과 recovery 절차에 있어서 상대 B'd 결함 검출 확률과 자기 B'd 결함 검출 확률의 합에 의존한다. 성능 평가를 간략화 하기 위하여 다음의 가정을 고려한다.

1) 소프트웨어 오류는 고려하지 않는다.
2) 시스템 내에 감지되지 않는 어떠한 에러도 존재하지 않는다.
3) 시스템이 두 side의 결함에 의하여 강등되었을 때, 시스템은 두 side를 동시에 회복하고 그 때 정상 상태로 복귀된다.
4) 전자적인 오류율은 Gate Level에서 입증된 것으로 고려하지 않는다.

그림 2는 기존 방안과 확장된 Warm standby의 신뢰도를 비교한 것이다. 각 방안의 운영 특성상, 주 모듈이 작업을 수행하고 있을 때, 대기하거나 진단 등의 예비 작업을 수행하는 Cold Standby의 시스템 신뢰도가 가장 우수함을 알 수 있다. 그리고 확장된 Warm standby는 오류로부터 시스템이 완전하게 동작 가능한 확률이 Hot Standby에 비하여 우수하므로 신뢰도가 더 높음을 알 수 있다.



[그림 2] 오류율에 따른 신뢰도 비교

4. 결 론

본 연구에서는 Warm standby의 동기화의 단순함에 따른 구현의 용이성이라는 측면에 기존 Warm standby의 단점을 개선함으로서 효율적인 이중화 시스템을 구축하였다. H/W 적인 측면에서는, 각 B'd가 자신 및 상대 B'd의 상태를 실시간으로 조사 함으로서, 오류 탐지 확률을 높이고 오류가 발생하면 H/W적으로 상태 비트를 설정하여 데이터 전송을 원천적으로 차단하였다. S/W적인 측면에서는, TCB와 EIMBuf, OEIMBuf를 이용한 메시지 처리 구조를 제시하여 데이터 손실 및 오류 데이터의 확산 문제를 해결하였다. 이를 통하여 자기진단계산에만 의존하고 오류로부터 완전하게 동작할 확률이 낮은 Hot standby에 비하여 확장된 Warm standby의 신뢰도 및 안전도가 우수함을 알 수 있었다. 향후에는 H/W 탐지 기능에 오류가 발생했을 경우 등의 부가 기능을 위한 S/W 오류 탐지 기능을 제시해야 한다.

참고 문헌

1) L. Young, "Software Fault Tolerance at the Operating System Level," Proc.of COMPSAC '95, pp.393, 1995
2) M. Russinovich, and Z. Segall, "Fault Tolerance for Off The Shelf Applications and Hardware," Proc.of FTCS 25, pp.67 71, Jun. 1995
3) B.J. Min, S.S. Shin and K. W. Rim, "Design and Analysis for a Multiprocessor System with Extended Fault Tolerance," IEEE FTDCS., pp. 301 307, Aug. 1995.
4) "부하분담 방식의 이중화 및 절체 방법과 이를 수행하기 위한 시스템," 현대전자 특허 96 42408, Dec. 1996.
5) 양승민외 3인, "BSC용 제어 프로세서의 실시간 미들웨어에 관한 연구," 한국전자통신연구원 위탁보고서, 숭실대학교, Dec. 1999.
6) 김상하외 4인, "Fault Tolerance(duplex) 방식 연구," 한국전자통신연구원 위탁보고서, 충남대학교, Dec. 1999.
7) 박동선외 6인, "프로세서 시스템의 고가용성 및 고신뢰성 구현에 관한 연구," 한국전자통신연구원 위탁보고서, 전북대학교, Nov, 1999.