2004년도 대한전기학회 하계학술대회 논문집 2004.7.14-16

시스템의 신뢰도를 높이기 위한 교환제어모듈 설계

정의국, 장경배, 심일주, 박귀태
고려대학교 전기공학과

DSCM(DualSwitchingControlModule)DesigntoHeightenReliabilityofSystem

Euikuk Jeong, Kyung-Bae Chang, Il-Joo Shim, Gwi-Tae Park
Electric engineering, Korea University

Abstract - As computer system has developed, the break down rate of system has been increased. So, engineers have been interested in reliability of system. General method that is used to heighten reliability of system is Redundancy of System by dual switching control. In this paper, we propose DSCM which can give Redundancy structure in existent PC(Microsoft Window 2000 OS) base system And we experiment redundancy structure that is applied DSCM to currently using train control in IFC(Interface Computer) system.

1. 서 론

컴퓨터 산업이 발전하면서 컴퓨터는 다양하고 복잡한 형태로 급격히 개발되었다. 하지만 시스템의 성능 향상에 비해 시스템의 안정성은 고려되지 못하여 시스템의 신뢰도는 많이 저하되었다. 이런 상태로 적용되는 분야는 확대되고, 산업 및 사회 전반에 컴퓨터에 의존도가 높아졌다. 만약 적용된 PC기반의 시스템들이 낮은 신뢰도의 시스템에 의해서 오동작이 발생하면 막대한 경제적 손실을 가져올 뿐만 아니라, 인명피해까지도 날 수 있다. 이에 많은 시스템 개발자들은 시스템의 신뢰성에 많은 관심을 갖게 되었다.

시스템의 신뢰성을 높이기 위해 사용되는 방법으로는 고품질의 소자를 사용하여, 소자의 고장 자체를 줄이는 고장회피(fault avoidance) 방법과 소자에서의 고장이 발생하더라도 시스템의 성능에 영향을 주지 않도록 하는 내고장성제어(fault tolerant control) 방법이 있다[1][2]. 고장회피 방법으로는 시스템이 복잡해지고 커지면서 원인을 알 수 없는 운영환경의 변화에 의한 고장 및 소프트웨어적인 버그 등에 의한 고장은 대처할 수 없으므로 현재는 내고장성제어 방법이 많이 연구되고 있다. 이 내고장성제어방법 중에서 가장 많이 사용되고 연구되고 있는 방법이 교환제어(Dual switching control)에 의한 시스템의 이중화(Redundancy)이다. 이 방법은 같은 두개의 시스템을 중복구조로 구성하여, 동작 중 에러가 발생하면 다른 준비된 시스템이 역할을 이어받아 정상동작을 하는 방법이다.

본 논문에서는 Fig.1. Concept of DSCM와 같이 기존의Microsoft Window OS기반의 PC 시스템을 교환제어방법을 이용한 시스템의 이중화가 가능하게 하는 모듈을 제시하고 구현했다. 이 모듈은 두 개의 같은 시스템이 각자의 상태를 모니터링하고, 그 정보를 서로 교환해야만 하므로 본 논문은 Microsoft Window OS기반의 시스템의 프로세스를 감시하기 위한 방법과 두 대의 PC가 서로 효율적으로 정보를 교환하기 위한 방법을 제시한다.

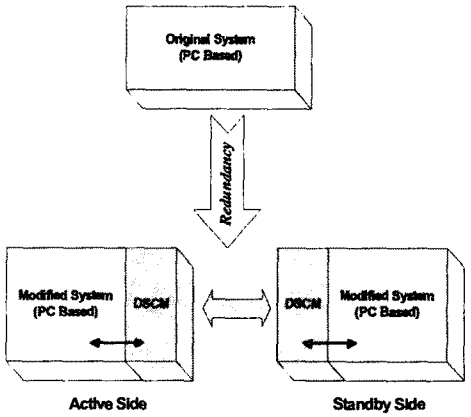

그림 1. Concept of DSCM

개발한 모듈을 현재 철도차량제어에 사용되고 있는 IFC(Interface Computer) 시스템에 적용을 하여 이중화시스템의 성능을 평가하였다.

따라을 2장에서는 교환제어모듈의 구조와 방법을 자세히 설명하고, 3장에서는 성능 비교를 위해 작성한 임의의 프로세스와 현재 철도차량제어에 사용되고 있는 IFC 시스템을 이중화하여 교환제어모듈을 실험 할 것이다. 마지막 4장에서는 결론을 짓고 앞으로의 향후 과제에 대해서 기술한다.

2. DSCM 교환제어모듈

우리는 2장에서 DSCM에 대한 설명을 한다. DSCM은 기존의 PC기반 시스템을 이중화하기 위한 프로세스로 기존의 시스템에 DSCM모듈을 추가하여 기존의 동작 프로세스의 이상 여부를 감시하고 이상이 발견되면 상대방에게 정보를 넘겨주어 절체를 할 수 있도록 한다. 이중화 기능을 구현하기 위해서는 먼저 기존 동작 프로세스를 감시하는 루틴과 상대방과 정보를 교환하는 루틴, 또 모은 정보를 이용하여 활동(Active)할 것인지 대기(Standby)할 것인지 결정하는 루틴이 필요하다. 우리는 DSCM의 구조와 자신의 프로세스를 감시하는 방법, 상대와 정보를 교환하는 방법, 또 동작 모드를 결정하는 방법과 마지막으로 기존 프로세스가 수정되어야 할 부분에 대해서 설명한다.

2.1 DSCM의 구조

Hot standby형태의 이중화 구조를 갖기 위해서는 같은 PC 시스템이 두 대가 필요하다[4]. 이 두 개의 시스템은 같은 입력과 출력을 제공하며 우선순위에 따라 결정된 Active Side의 시스템이 출력을 낸다. 또 DSCM은 모듈 형태로 기존의 동작 프로세서와 병렬형태로 같은

PC내에서 실행되며, 두 대의 시스템에서 모두 실행한다. Fig2.는 DSCM을 이용한 시스템의 구조를 보여준다.

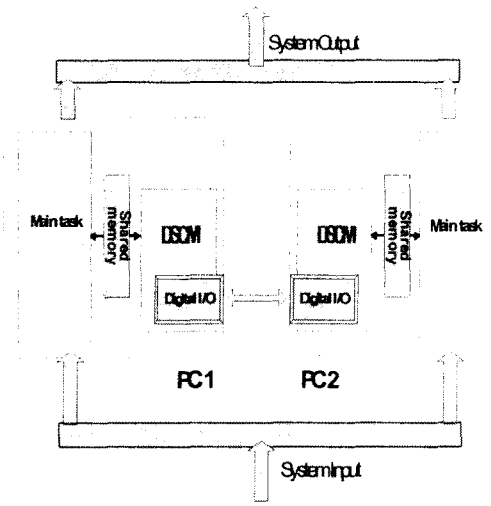

그림 2. DSCM의 구조

Fig2.에서 Main Task와 DSCM은 Microsoft사의 Windows OS상에서 구동되는 프로세서이고 여기서 Main task는 기존의 동작 프로세서를 DSCM의 적용이 가능하도록 수정을 한 것이다. 그리고 Main Task나 상대를 인식하기 위한 중간 매개로 Shared memory와 Digital I/O가 사용되었다. DSCS는 Main Task의 상태를 감시하기 위해서 Main Task의 일정 데이터를 계속 억세스한다.

Windows OS상에서 별개의 프로세스가 데이터를 공유하는 방법으로는 Windows가 제공하는 MAPI (Messaging Application Programming Interface) 중에서 IPC (Interprocess Communication)를 이용한다.[5] IPC에는 Clipboard, COM, WM_COPYDATA, File Mapping 등의 메카니즘을 사용하는 데 DSCM은 일정간격마다 계속적으로 데이터를 공유하므로 하나의 물리적인 메모리를 이용하는 File Mapping 메커니즘을 이용하는 것이 유리하다. 그래서 DSCM은 File Mapping 메커니즘을 이용하는 Shared Memory를 이용하였다.

두 대의 PC시스템이 데이터를 정보를 교환하기 위한 인터페이스 방법은 정말 다양하다. 그 중에서 Printer Port(SPP, EPP, ECP), Serial Port(RS-232)와 LAN(Ethernet)을 이용한 방법은 최근 가장 많이 사용되고 있는 인터페이스들이다. 하지만 이런 인터페이스들은 모두 데이터 전송을 위한 컨트롤러와 드라이버가 필요하기 때문에 데이터전송속도가 느리고, 컨트롤러와 드라이버 자체의 에러가 발생 할 수 있으므로 DSCM에서는 가장 간단하면서도 안정성이 높은 인터페이스인Digital I/O Board를 이용한다.

2.2 동작프로세스를 감시하는 방법

DSCM은 Shared memory를 이용하여 Main Task와 정보를 교환하고 상태를 감시한다. 먼저 Main Task의 상태를 정의하면 Table 1.과 같다.

Main Task의 상태는 두 가지 방법을 이용하여 결정한다. 첫 번째 방법은 일정 주기마다 Window OS의 API를 이용하여 실행 중인 프로세스의 이름과 핸들을 가져와서 동작여부를 확인하는 것이다. 두 번째 방법은 Shared Memory에 데이터 구조체를 번갈아가며 읽고 쓰면서 정보를 교환합니다. 이때 사용하는 구조체는 Table2.와 같다.

Table 1. Main Task의 상태 정의

Normal	정상 상태로 이상 없이 동작중인 상태	0x04
Warning	경고 상태로 불안정한 상태, 신호를 한번 놓친 경우	0x02
NoResponse	응답이 없는 상태, Shared memory에 전혀 응답을 하지 않는 상태	0x03
Error	에러 상태 Main task 내부에서 자체진단해서 동작이 불가능하다고 판단되었을 경우	0x01

Table2. Shared Memory에서 공유할 데이터 구조

Handle	Shared Memory의 핸들값으로 메모리가 생성한 때 값을 씌운다	Char
Code	서로의 정보값이 들어간다	Char
SamplingTime	동작주기값이 들어간다	Char
RW	0x01: DSCM이 데이터를 쓴 경우 0x02:MasterTask가 데이터를 쓴 경우	Char
CheckSum	Check Sum	Char

데이터 구조체에서 RW멤버는 Main Task와 DSCM의 정보 업데이트 여부를 판단하는 데이터이고, Code멤버는 서로에게 전달할 정보를 갖고 있다. 또 CheckSum 멤버는 메모리쓰기 중의 오류나 잘못된 연산에 의한 오류를 찾아낸다. 또 Sampling time멤버는 서로 정보를 교류할 주기시간 값을 전달하여 최적의 시간을 설정하는 데 사용된다.

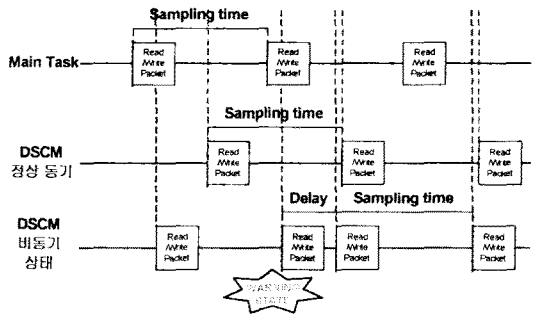

그림 3. 동기화

두 개의 프로세스는 독립적으로 동작을 하기 때문에 정보를 한번씩 읽고 쓰는 데에는 동기를 맞춰주어야 하는 문제가 있다. 이 문제는 Fig 3.과 같이 한번 신호를 놓치면 Sampling time의 2/5값만큼의 Delay를 주어 동기를 맞추려 시도를 한다. 여기에서도 응답이 없으면 Fig 4.과 같이 NoResponse State로 인식한다.

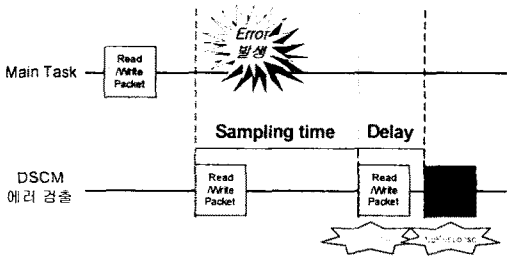

그림 4. 동기화 실패

2.3 상대방과 정보를 교환하는 방법

DSCM은 자신과 완벽하게 똑같은 시스템과 이중화를 구현한다. 이 두 대의 똑같은 시스템은 서로의 정보를 공유하는 데 이 때 DIO Board를 이용한다고 미리 언급을 했다. 이중화를 위해서 서로 공유되어야 할 정보는 시스템의 동작 가능 여부와 현재 동작하고 있는 모드 두 가지 밖에 없으며, 또 빠른 절체를 위해서 DSCM은 간단한 이진의 데이터를 이용해서 정보를 공유한다.

먼저 동작 가능 여부는 Heart Response Signal로 DIO Period 시간마다 하나의 스텝 신호를 보내준다. 이 때 Timer Period동안 응답이 없는 경우에 DSCM은 상대 시스템을 동작 불가능한 상태로 인식한다. 또 현재 동작 모드는 Current Mode Signal로 'H'상태면 Active 모드, 'L'상태면 Standby or Error 모드를 나타낸다.

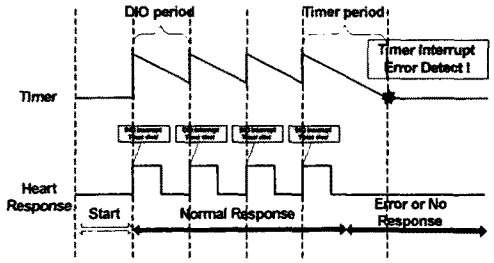

그림 5. Heart Response Signal

2.4 DSCM의 동작 모드 결정

DSCM은 Main Task의 상태와 상대 시스템의 상태를 보고 세 가지 모드를 정의하여 동작을 한다. 먼저 Active 모드는 동작모드로 Active Side쪽 시스템의 동작 모드이다. Standby 모드는 동작 가능모드로 Standby Side 쪽 시스템의 동작모드이다. 마지막으로 Emergency 모드는 자체 진단으로 오류를 하여 동작이 불가능한 상태이다. 만약 Emergency모드로 변경되면 Heart Response Signal을 출력하지 않음으로써 상대 시스템에게 우선권을 넘겨준다.

표 3. 동작 모드

Active	Active Side로 동작 중인 모드	0x01
Standby	Standby Side로 시스템 동작이 가능한 모드	0x02
Emergency	DSCM이 자체 에러라고 판단하여 시스템 동작이 불가능한 모드	0x03

시스템을 이중화로 구현하였을 때 두 대의 시스템은 같은 우선순위를 가지므로 동시에 Normal State가 되면 어느 시스템이 Active Side가 되어야 할지 결정해야 한다. DSCM에서는 두 가지 방법을 제공한다. 어느 한 시스템에게 우선권을 부여하는 Master/Slave Method와 랜덤하게 우선권을 부여하는 Random Select Method이다.

Master/Slave 방법은 DSCM 프로세스에 Master와 Slave를 입력하여 Master가 입력된 시스템이 항상 높은 우선순위를 갖는 방법이다.

또 Random Select 방법은 두 시스템이 같은 모드로 동작중일 때 먼저 동작 중이던 시스템에게 우선순위가 부여되며 만약 동시에 같은 모드로 변환을 하면 Time_Slot(1~10) * DIO_Period [msec]사이에 중 랜덤 값을 취해 그 시간 후에 Active Mode로 동작하는 방법이다. 즉 랜덤 값이 작게 나온 시스템이 먼저 우선권을 갖게 되는 데 만약 같은 랜덤 값이 결정되어 또 같은 모드로 변환되면 다시 한번 랜덤 값으로 우선권을 결정한다.

2.5 기존 시스템의 수정 사항

시스템을 이중화로 구현하였을 때 두 대의 시스템은 같은 우선순위를 가지므로 동시에 Normal State가 되면 어느 시스템이 Active Side가 되어야 할지 결정해야 한다. DSCM에서는 두 가지 방법을 제공한다. 어느 한 시스템에게 우선권을 부여하는 Master/Slave Method와 랜덤하게 우선권을 부여하는 Random Select Method이다.

Master/Slave 방법은 DSCM 프로세스에 Master와 Slave를 입력하여 Master가 입력된 시스템이 항상 높은 우선순위를 갖는 방법이다.

또 Random Select 방법은 두 시스템이 같은 모드로 동작중일 때 먼저 동작 중이던 시스템에게 우선순위가 부여되며 만약 동시에 같은 모드로 변환을 하면 Time_Slot(1~10) * DIO_Period [msec]사이에 중 랜덤 값을 취해 그 시간 후에 Active Mode로 동작하는 방법이다. 즉 랜덤 값이 작게 나온 시스템이 먼저 우선권을 갖게 되는 데 만약 같은 랜덤 값이 결정되어 또 같은 모드로 변환되면 다시 한번 랜덤 값으로 우선권을 결정한다.

표 4. DLL에서 제공하는 함수들

DSCMOn	DSCM과 통신하는 프로세스 동작
DSCMOff	DSCM과 통신하는 프로세스 멈춤
DSCMGetMode	현재 DSCM에서 판단한 동작 모드를 얻어 오는 함수
DSCMSetState	Main Task의 자체 진단 기능이 있는 경우 현재 자기 상태를 DSCM에 넘겨주는 함수

지금까지 우리는 DSCM의 구성과 실제 구현 방법에 대해서 논하였다.

3. 실 험

우리는 제안한 DSCM을 현재 철도제어시스템에서 사용되고 있는 IFC에 적용하여 실험을 한다. 이 실험의 목적은 절체되는 과정을 직접 확인하고, 시스템에서 발생한 에러에 대해 얼마나 빠른 처리가 가능한지 측정하여 시스템의 이중화시에 필요한 요구사항을 만족하는 지 확인한다.

방금 소개한 IFC시스템은 각 철도 연동역 및 차량기지 신호기계실에 설치되어, 연동장치와 상위시스템(사령실 CDTS, LCC, LMC, SSC)의 인터페이스를 수행하는 일종의 게이트웨이이다. 즉 연동장치 PROFIBUS 프로토콜과 상위시스템 Ethernet, TCP/IP 프로토콜의 서로 다른 프로토콜을 변환하여 줌으로써 정보를 송, 수신 할 수 있도록 한다.

IFC의 가장 중요한 역할은 연동장치와 연결되어 있는 Profibus 프로토콜의 데이터를 수집, 정리하여 TCP/IP 프로토콜로 변환하는 것이다. 연동장치로부터 들어오는 데이터들은 모두 500[msec] 주기로 들어오기 때문에 에러에 의한 시스템의 절체시간은 500[msec]보다 짧은 시간이어야 한다.

DSCM에서 프로세스의 수행 시간을 측정하는 루틴을 포함하여 시간을 측정하고 그 평균치를 계산하였다. 여기서 절체되는 시간은 Active Side의 Main Task에서 임의로 에러를 발생한 시각부터 절체가 완료되어 상대 시스템이 Active Mode로 동작하는 것을 Active Side의 DSCM이 확인한 시각까지의 시간을 100번 측정하여 파일로 저장한 다음 이 평균을 구하였다.

다음은 실험에 사용될 시스템의 사양이다.

- CPU : Intel Pentium IV 2.4G
- Memory : 256MB(DDR)
- OS : Microsoft Windows 2000PRO
- Main Task : IFC 운영 프로그램

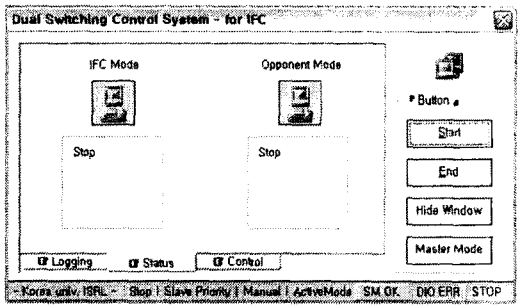

그림 6. DSCM 프로그램

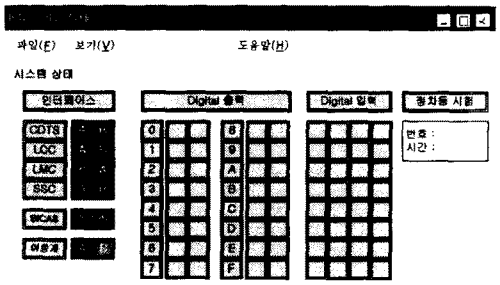

그림 7. IFC 운영 프로그램

파일로 저장된 데이터는 MatLab을 이용하여 그래프로 그리고, 평균값을 얻었다.

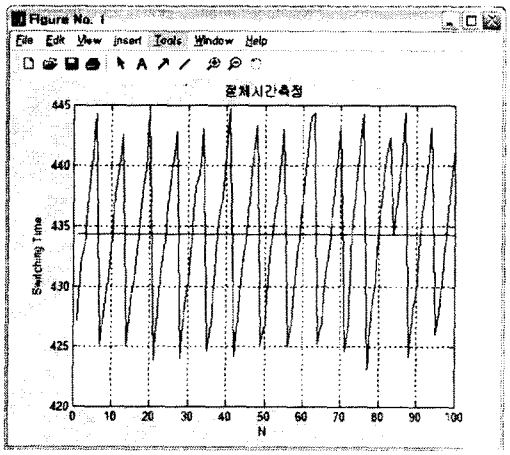

그림 8. IFC 절체 시간 측정 결과

Fig 8.은 절체 시간을 그래프로 나타낸 것이다. 모두 100번을 측정하는 데 최고 445[msec]에서 최저 423[msec]사이를 계속 왔다 갔다 하는 것을 볼 수 있다. 그림에서 붉은 선은 절체 시간의 평균값을 나타낸다. 이 값은 434.3622[msec]로 측정되었다.

실험결과 에러 발생시에 완벽하게 절체를 하였다. 또 시스템의 절체시간은 434.3622[msec]가 소요되어 IFC에 필요한 요구 조건 500[msec]를 만족시켰다.

4. 결 론

우리는 앞에서 PC 기반의 시스템이 이중화가 가능하도록 모듈 형태의 이중화 방법을 제시하였다. 제시한 방법은 단순하면서도 기존 시스템을 약간 수정함으로써 시스템의 이중화가 가능하였다. 실험을 통하여 이중화가 가능한 시스템의 성능을 알아보았다.

앞으로 우리는 완벽한 모듈 형태를 지원하기 위해서 별도의 수정 없이 동작 프로그램의 상태를 알아볼 수 있는 방법을 연구하고, 상대 시스템과의 정보 교류에 필요한 몇 가지 알고리즘을 더 추가, 수정 할 것이다.

[참 고 문 헌]

[1] Sae Hwa Park, "A Study on the Control system with Dual Structure to Enhance Its reliability", *Journal of Control. automation and Systems Engineering*, 1990. 10

[2] K. S. Song, H. K. Yeo, "Hot standby Computer Structure : AnAvailability Analysis Of Switching Control System with Hot Standby Fault Tolerant Architecture", *Jounal of Korea Information Processing Society*, 1995. 11

[3] Barry W. Johnson, "Reliability & Safety Analysis of a Fault-Tolerant Controller", *IEEE MICRO*, p22-p33, 1984

[4] Hwan Geun Yeo, "The Implementation Method of Synchronization Unit in Hot Standby Dual Switching Control System", *Jounal of Electronics and Telecommunications Research Institute*, 1995

[5] http://msdn.microsoft.com/