자동열차제어장치의 Fault-tolerant 설계안
Fault-tolerant Design Concept of Safety Critical System for Automatic Train Control System

황 종 규*, 이 종 우*, 오 석 문*, 김 영 훈*
Jong-Gyu Hwang, Jong-Woo Lee, Suk-Moon Oh, Young-Hun Kim

ABSTRACT

The μ-processor based-controlled system is widely used in railway signaling system. The railway signaling systems are highly required safety and reliability. It is necessary to have a fault-tolerant and fail safe concept in μ-processor based railway signaling system. In this paper, several architectures and circuits of fault-tolerant computer system is reviewed. The basic concept of the fault-tolerant computer system will be adapted total self checking, strong fail safe, fault display circuit, logic testing circuit and system switching concepts.

1. 서론

철도 신호시스템은 과거에는 계전기 및 아날로그 회로를 이용하여 구성하였으며, 현재의 신호시스템은 컴퓨터 기술의 발달에 힘입어 마이크로프로세서를 기반으로 한 시스템이 널리 활용되는 추세에 있다. 신호시스템의 자동열차제어장치, 연동처리장치는 고장 안전측으로 작동이 되어야 하며, 동시에 열차집중제어장치와 같은 결합허용 특성이 요구된다. 현재 건설되고 있는 경부고속철도 자동열차제어장치와 연동장치의 안전도는 대재해에 해당하는 에러는 5,000년에 한번이며, 치명적인 고장은 200년에 한번으로 정의되어 있는 등 신호제어시스템은 상당히 높은 안정성 및 신뢰성을 요구된다. 따라서 신호시스템의 제어장치는 Fault-tolerant 성능을 확보하는 것이 필수적이다.

Fault-tolerant 성능을 확보하기 위해서는 정보의 이중화, 하드웨어 및 소프트웨어의 이중화와 이 방법들을 합성하는 등 다양한 방법이 제안되어 왔다. 과거의 Fault-tolerant 기술은 각 부품에 충분한 설계여유를 계산하여 확보하였으나, 오늘날처럼 기능이 복잡해지고 제품의 life-cycle이 짧아지고 설계자가 따라 특정용도의 부품을 생산할 수 없다. 이에 따라 대부분의 부품은 아웃소싱을 통하여 조달되고 있어, 범용제품을 이용하여 Fault-tolerant 특성을 갖도록 하는 것이 필요하다. 컴퓨터를 이용한 제품들은 life cycle이 짧으며, 외부의 인자에 의해 쉽게 영향을 받기 때문에 여러 분야에서 즉시 적용할 수 있도록 하는 Fault-tolerant기술 개발이 필요하다[1]-[5].

---

* 한국철도기술연구원, 정회원

본 논문에서 일본철도총합연구소의 '나카무라'와 '다케시'가 제안한 Fault-tolerant 시스템 구성방법과 이를 자동열차제어장치 하드웨어에의 적용방법을 검토하였다. 검토된 내용은 하드웨어 내에서 데이터의 오류검지, 표시방법을 이용한 Fail-safe 및 가용성을 높이기 위한 유니트의 절체방법 등을 논하였다[1].

2. Fail-safe 제어시스템의 개발

2.1 자동열차장치의 기본 구성

자동열차제어 장치는 구배, 곡선반경 등의 선로조건과 궤도회로를 포함한 현장 신호기기들의 상태를 바탕으로 열차의 간격제어 및 속도제어를 하는 장치로서 높은 안정성과 신뢰성을 요구한다. 따라서 이 자동열차제어장치의 제어시스템은 안정성과 신뢰성의 높은 요구조건을 만족하기 위하여 Fault-tolerant 시스템으로 구성하여야 한다.

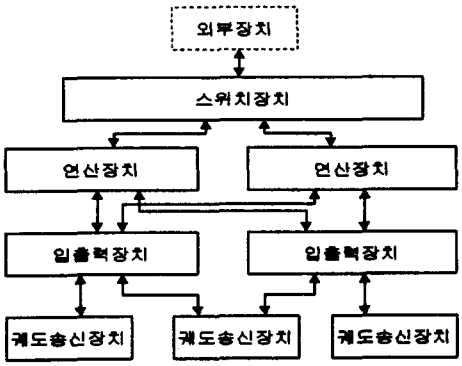

그림 1. 자동열차제어장치의 기본 구성

Fault-tolerant 기능을 갖는 자동열차제어 장치의 기본적인 하드웨어 구성은 그림1과 같이 이중의 연산장치가 스위치 장치를 통하여 2개 원격통신장치 및 입출력 장치에 연결되어 진다. 궤도회로 및 불연속 전송루프로 연속정보와 불연속 정보를 전송하기 위한 송신장치는 입출력 장치에 연결되어 진다. 연산장치와 입출력 장치간 등 각 장치간의 데이터 교환을 감시하여 Fault를 검지할 수 있도록 한다.

2.2 버스동기식 Fail-safe 컴퓨터 구성방법과 Self-checking 회로

Fail-safe 컴퓨터의 구성방법은 버스 동기식 이외에 소프트웨어의 Diversity에 맡기는 방법으로 하는 등 여러 가지 방법이 있다. 이들 중 버스동기식은 하드웨어의 Fault에 기인하는 오류는 하드웨어 자신이 담당하기 때문에 소프트웨어의 부담을 경감시킬 수 있고, 오류가 전파되기 전에 검출할 수 있기 때문에 Fail-safe 및 Fault-tolerant 기능을 쉽게 부여할 수 있다.

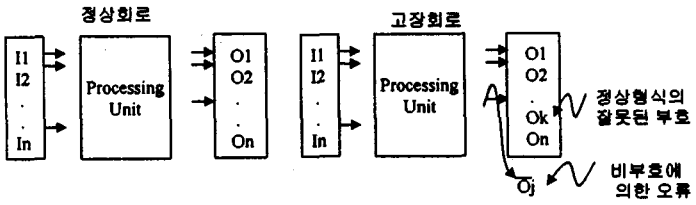

그림 2. 회로의 고장과 출력의 관계

이런 Fail-safe 및 Fault-tolerant 시스템의 구성은 버스동기식에 의한 Self-Checking

기법의 개념을 근거한다. 각 보드간의 입출력의 데이터 부호를 나타나는 회로에서 부호를 사전에 정하여 사용할 경우, 출력이 미리 정해진 결과와 틀린 비부호는 쉽게 구분할 수 있고 이로서 시스템 차원에서 해결책을 찾을 수 있다. 하지만 잘못된 입력에도 정해진 결과가 정상적으로 나오는 등의 경우 Fault를 검지 할 수 없는 경우가 발생할 수 있다.

이러한 회로의 동작시 오류의 검출을 위해서는 입력데이터가 입력이 되어야 하며, 입력에 대한 올바른 부호를 출력하는지의 여부를 확인할 수 있어야 한다. 잘못된 입력 데이터가 입력이 되어도 오류의 값을 출력하지 않는 것을 FS(Fault Secure)라 하고, 임의의 입력에 잘못된 출력을 검지 할 수 있는 것을 Self Checking이라 하며, 이 두 가지를 합쳐 전체자가검사(TSC : Total Self Checking)라 한다.

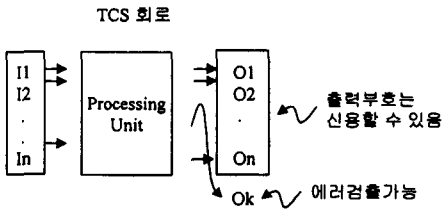

그림3. Total Self Checking 회로

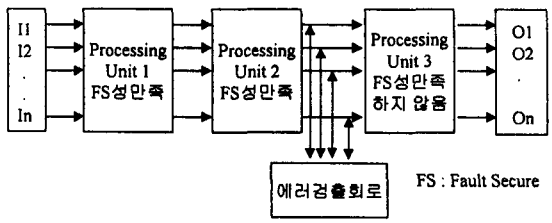

그림 4. TSC 회로를 합성한 SFS시스템의 개요

그러나, 고장이 검출될 때까지 이중의 고장이 발생한다면 이 특성도 보장되지 않는다. 따라서 실제에 있어서 고장이 발생되어도 검출될 때까지 출력이 보장되는 회로구성이 필요하다. 이와 같이 높은 신뢰성 및 안전성을 보장하는 것을 SFS(Strong Fail-safe)라 하며 이 성능을 위해서는 Self-checking 회로가 적절하다.

시스템을 SFS 성능을 갖도록 구성하려면 FS성을 만족하지 않는 유니트 앞에 에러 검지회로를 삽입시킴으로서 고장이 발생하여도 이 검사회로에서 검출됨으로 인해 출력의 신뢰성을 확보할 수 있다. 그러나 이처럼 FS성을 만족하지 않는 유니트가 여러 개일 경우 에러 검지회로가 다수 필요하고 유니트가 각각 틀린 유니트들에 대해 SFS특성의 회로설계를 해야하므로 범용적으로 이용할 수 없다. 또한 이 검사회로가 필요한 부분이 CPU, ALU 등 단위 유니트별 필요로 하여 범용소자의 사용이 불가능하다. 따라서 인터페이스되는 버스상의 FS를 만족하기 위해서는 비부호 정보의 확산방지가 필요하며, 오류가 전파되어 어디에서든지 버스상 오류가 검출될 수 있도록 하여야 한다. 이에 따라 FS성이 보증되지 않는 부분에만 검출회로를 삽입하는 것으로도 SFS 시스템을 구성할 수 있다.

버스동기식 FS 컴퓨터는 CPU, 메모리 등을 각각 구성요소들 사이의 데이터 및 어드레스 버스를 에러검출을 위한 인터페이스로 보고 비부호화 검출을 위해 검지회로를 삽입시키는 방법이다. 프로세서 내부 혹은 메모리 등의 구성요소에 고장이 발생하여도, 버스로 에러가 전파되지 않는 경우에 안전하다고 볼 수 있으며, 버스 상에 전파된 경우에는 검사회로로서 검출할 수 있다. 이 결과 CPU 내부에 검사회로를 삽입하지 않고서도 SFS시스템을 구성할 수 있다.

2.3 검사회로에 요구되는 성능

버스에 삽입하는 검사회로는 회로자신이 TSC 특성을 갖고있는 것과 FS성을 만족하는 것이 있다. 검사회로의 한 예로서 자기진단 '3-out-of-6 code' 검사회선의 논리적인 실행에 대해서 설명한 것이다[2]. 각 하위그룹에 의해 생성되어 실행할 수 있는 출력에 대한 논리적인 표현은 다음과 같다.

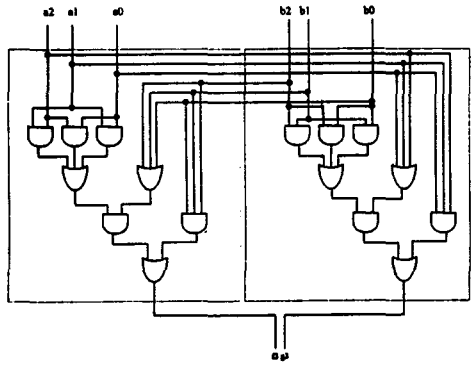

그림 5. 3 out of 6의 TSC 회로

㉠ *A Partition*(a_2, a_1, a_0):

$$[\geq 1_a(1)] = a_0 + a_1 + a_2$$

$$[\geq 2_a(1)] = a_0\,a_1 + a_0\,a_2 + a_1\,a_2$$

$$[\geq 3_a(1)] = a_0\,a_1\,a_2$$

㉡ *B Partition*(b_2, b_1, b_0):

$$[\geq 1_b(1)] = b_0 + b_1 + b_2$$

$$[\geq 2_b(1)] = b_0\,b_1 + b_0\,b_2 + b_1\,b_2$$

$$[\geq 3_b(1)] = b_0\,b_1\,b_2$$

$[\geq 2_a(1)]$의 의미는 입력 데이터 단어의 *A Partition*(a_2, a_1, a_0) 의 하위그룹으로서 읽혀지는 것이고, 그것에서는 비트의 상태가 1인 수가 2개 이상인 경우이다. 입력 데이터는 (a_2, a_1, a_0, b_2, b_1, b_0)와 같이 일련의 2진수로 주어진다. 2개의 하위그룹(f_3 와 g_3)에 의해 생성된 실행결과에 대한 논리적인 표현은 다음과 같다.

$$f_3 = [\geq 0_a(1)][\geq 3_b(1)] + [\geq 2_a(1)][\geq 1_b(1)]$$

$$= b_0\,b_1\,b_2 + (a_0\,a_1 + a_0\,a_2 + a_1\,a_2)(b_0 + b_1 + b_2)$$

$$g_3 = [\geq 1_a(1)][\geq 2_b(1)] + [\geq 3_a(1)][\geq 0_b(1)]$$

$$= a_0\,a_1\,a_2 + (b_0\,b_1 + b_0\,b_2 + b_1\,b_2)(a_0 + a_1 + a_2)$$

단 f_3 와 g_3의 표기는 출력기능이 두 개의 하위그룹에서 3개 이상의 bit가 '1'인 경우에만 동작되는 것을 의미한다. 만약 일반적인 m-out-of-2m 코드를 사용한다면, f_m 와 g_m 출력에 대한 논리적인 표현은 아래와 같이 되어진다.

$$f_m = \sum_{i=0}^{m} [\geq i_a(1)][\geq (m-i)_b(1)],\ \text{단 I = 0, 2, 4, } \cdots\cdots,\ \text{짝수}$$

$$g_m = \sum_{i=0}^{m} [\geq i_a(1)][\geq (m-i)_b(1)],\ \text{단 I = 1, 3, 5, } \cdots\cdots,\ \text{홀수}$$

자기검사회로는 2개로 분리된 독립적인 하위회로에서 서로 독립적으로 분할된 검사회로에 의해 실행되어진다. 각 하위회로는 단독으로 출력을 발생한다. 그러한 출력 값은 통상적인 m-out-of-2m 입력코드에 대해 상호 보완적이다. 입력 부호어 또는 검사 로직에서 발생하는 고장에 대해서 두 개의 출력 값은 같다(f_g출력 값은 '00' 혹은 '11' 이다). 전체적인 자기 진단 검사회로는 주기적인 검사 없이도 논리회로에서 발생하는 오류에 대해서 즉각적인 탐색이 가능하다는 장점이 있다. 과거에는 m-out-of-n에 대한 진단회로는 소스

입력 부호어를 재생시키기 위해 암호화된 것을 해독하는 출력 형태로 되어있다. 재생된 코드워드와 소스코드워드를 비교하는 방법은 두 코드워드에서 각각의 비트를 비교하여 수행하게 된다. 단순 비교는 XOR기법을 사용하는 것이다. 그러나 n-bit에 대한 XOR 기법에 의한 비교는 자기진단 설계가 아니다. XOR기법은 적절한 기능을 보장하기 위해 주기적인 검사가 필요로 하며, 자기진단 비교는 XOR기법을 사용하는 논리적인 실현보다 더욱 더 복잡하다.

그러나 이 검사 회로는 비부호 언어가 주어진 경우에는 비부호어의 출력을 보장하는 것이기 때문에, 다른 부호어가 입력되어 진다면 부호어를 출력한다. SFS 시스템에서는 일단 비부호어의 출력을 행한 후의 출력 값에 대해 보장하지 않는다. 따라서 2중의 고장으로 천이하지 않는 동안에 고장발생을 확실히 알아내고, 시스템 차원에서 해결책을 사용하는 것이 중요하게 된다. 이러한 것을 위한 것이 비부호 언어의 발생을 기억하는 회로가 '에러표시회로'가 있다.

2.4 에러표시회로

셀프체크 회로에 대해 오류표시 회로의 필요가 있는 곳에서는 그림6과 같은 회로가 사용될 수 있다[5]. 그림6은 입력이 (1,0) 혹은 (0,1)의 정상부호가 주어지게 되면 FF1, FF2가 각각 '0', '1'로 되어 LED가 점등되고, 불일치가 된다면 XOR의 출력이 '0'이 되고, 각 FF의 출력이 반전하기 때문에 소등되어 에러를 표시하게 되는 것이다. 그러나 XOR의 출력은 동력학적인 출력이 없기 때문에, 정상 시에 출력이 하나의 고장이 발생하면은 검지할 수 없다. 즉 두개의 XOR가 고장일 경우 그것의 입력에 에러 발생하여도 점등하여 계속되고 에러표시와 이에 따른 Fail-safe 기능도 확보할 수 없다.

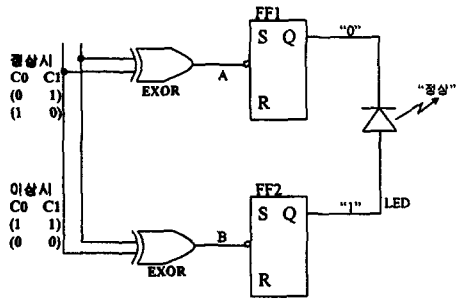

그림 6. 'Usas'가 제안한 에러표시회로

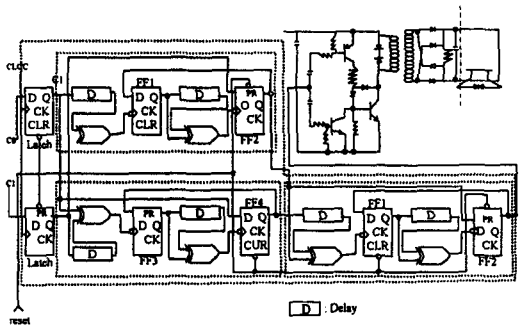

그림 7. '나카무라'가 제안한 에러표시장치

이에 따라 모든 경우에도 에러표시 및 Fail-safe 특성을 갖는 표시회로가 제안되었다[1]. 또한 제안된 에러표시 회로는 정규부호 입력이 계속되는 경우에 '1', '0'이 반복되게 출력되고 비부호가 입력되면 '1' 또는 '0' 출력이 고정되어 출력되어 그 상태를 유지하는 것으로 되어있다. 물론 회로자신은 Fail-safe가 있고, 회로 내부 내에 고장이 있다면 교번출력은 멈추게 된다.

검사회로와 에러 표시회로를 조합한 것을 회로는 버스동기 방식 시스템의 성능에 적당

한 별도의 논리회로가 필요하다. 이 논리회로에서는 비교처리가 'read/write' 중 하나의 기능이 고장인 입력이 연속해서 주어질 수 있는 경우에 회로고장과 식별하는 것을 고려하여 다음의 2개의 기능 부가가 필요하다.

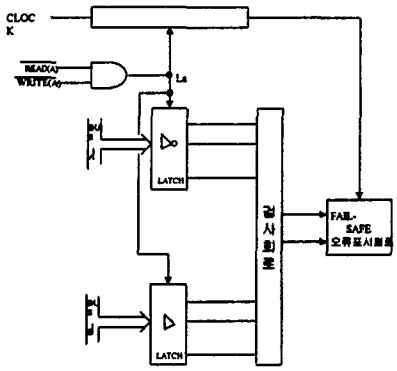

그림 8 에러가 발생해도 체크할 수 없을 수도 있음

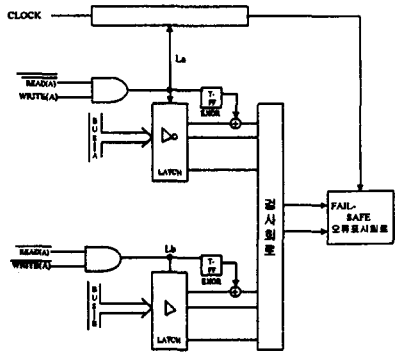

그림 9 Self-checking 장치를 사용한 회로

• 비교처리를 못하는 것을 방지하는 검증기능

그림8에서 버스데이터 인가신호 La에 의해 비교처리가 시행된다[1]. 여기서 고장에 의해 La가 'read(write)'로 되어졌다면 'write(read)'시에 비교처리가 시행되지 않기 때문에 버스 상에 에러를 검출하지 못할 우려가 있다. 이와 같은 고장을 검출하기 위해 그림9와 같이 양쪽의 인가신호 La, Lb를 이용하여 비교 데이터를 취합함과 동시에 T형식의 FF을 반전시키고, 이것의 출력과 비교 데이터 중의 하나의 비교 비트를 XOR로 하고, 하나의 트의 비교 비트에 대한 것을 한 주기마다 반전시키는 것으로 한다. 만일 한쪽의 인가신호에 위의 간헐적인 고장이 발생하는 경우에, T-FF와 출력 값이 불일치가 되기 때문에 XOR 연산으로 나타난 비트에 대해 갖추어지지 않아 데이터 불일치가 발생한다. 이 결과 에러 표시회로의 교번출력이 정지하게 된다.

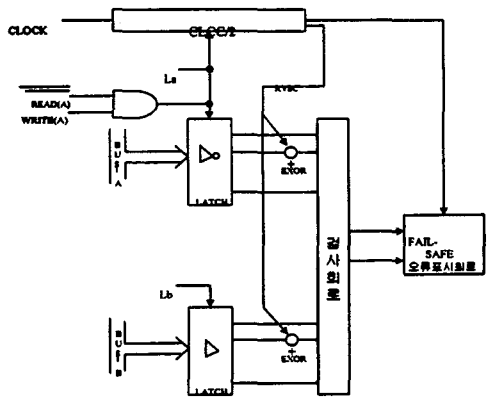

그림 9 연속된 동일입력에서도 교번출력을 발생시키는 회로

• 동일입력의 반복에 대한 식별회로기법

부호검출기 회로의 입력과 출력으로 예시한 것과 같이 데이터 검사회로의 부호 출력은 입력 비트에 대한 값에 의해 (1,0) 혹은 (0,1)이 된다. 예를 들어 (1,0)의 출력을 채용하는 입력이 연속하여 주어진 경우는 출력이 정지하기 때문에 비교 클럭의 고장과 같은 것이 된다. 이러한 문제를 해결하기 위해 1회 비교사이클 중 (1,0), (0,1)의 출력이 얻는 것과 같이 입력 비트 대를 변화시켜, 두 번 비교를 행하는 것으로 하였다. 출력을 변화시키는 입력 비트에 대한 생성은 각각 데이터의 하나의 비트에 대하여 XOR게이트를 이용하여 반전시키는 것

에 의해 얻을 수 있다.

2.5 고장검출시 안전측 제어

비교회로의 교번출력정지는 연산, 입출력, 스위치 장치 등의 프로세스를 구성하는 기능에서 고장이 발생한 것을 의미하기 때문에 출력을 안전측으로 제어할 필요가 있다[1]. 비교회로의 교번출력은 정상적으로 릴레이를 작동시키기고 교번작동이 정지하는 때에는 릴레이가 무여자로 되어 복귀된다. 정상상태의 릴레이 접점은 출력 전원 선에 삽입시키고 이것이 개방에 의해 출력전원이 끊기어 모든 출력은 안전측으로 동작된다. 더욱이, 비교회로의 교번출력을 Fail-safe 출력회로의 클럭신호로서 사용되며 이 장치에서 교번출력의 정지는 출력을 안전측으로 바뀌게 된다.

3. 자동열차제어장치의 Fault Tolerance

자동열차제어장치의 시스템은 FS 컴퓨터에서 시스템 버스로 결합하여, 각 장치간에 matching 프로세서로서 구성할 수 있다. 대부분의 신호보안장치에서는 하나로 된 독자 시스템 제어장치와 matching 프로세서 구성을 채용하기도 하고, 시스템 버스는 전용의 2중계 Fault-controller 버스를 개발하여 사용하기도 한다. 따라서, 범용의 프로세서와 독자의 버스인터페이스를 갖춘 별도의 제어보드 등이 있다. 3중계 다수결 FS컴퓨터는 버스인터페이스 유니트를 구성하여 시스템을 결합 시켜서 사용하기도 한다.

3.1 처리 연속성을 유지를 위한 방법

열차추적 처리 등에서는 시스템의 절체시에 동작이 중단될 경우 에러를 발생시키므로 이를 보상하기 위한 프로세스가 복잡해진다. 따라서, 최대한으로 절체시간을 줄여 동작의 연속성을 유지하면서 절체되어야 하는 것이 높은 신뢰성을 요구하는 제어시스템의 기본적인 요구사항이다. 이것의 일반적인 실현방법은 두 개의 시스템이 각각 대응하는 프로세서들을 연결시켜 대기상태의 유니트가 활성화 상태인 유니트의 데이터를 상시 파악할 수 있도록 하는 것이 있다.

동일의 시스템 버스 상에 대기상태 유니트의 프로세서를 장치하면은 활성화 상태 유니트의 데이터를 획득하는 것이 용이하게 되어 있다. 따라서, 범용의 시스템버스는 Fault-tolerant 기능이 없기 때문에 하나의 유니트가 버스를 전유해 버리게 되므로 시스템 버스 자체에 고장이 발생한 경우에는 활성화 유니트의 프로세스도 작동을 할 수 없게 된다. 또한 시스템 버스를 분리시켜, 프로세서들을 통신회선으로 결합하는 방법이 사용될 수 있다. 이 경우 버스간의 양방향 통신을 위한 버스 어댑터와 프로세서 서로간의 모니터링 및 오류검지를 위한 알고리즘을 필요로 한다. 또 다른 방법은 두 개의 시스템을 접속하는 미러메모리라 하는 인터페이스 보드를 개발이 필요하다. 미러메모리는 시스템상의 각 프로세서에 이것까지 통하게 하는 처리환경을 보장하면서, 시스템 버스를 사용한 프로세서 상호교환 데이터를 타 시스템 버스에서 임의로 억세스가 가능하여야 한다.

3.2 미러메모리 방식의 개념

절체시에 처리의 연속성을 확보하기 위해, 액티브 유니트의 프로세서 상호간 교환되는 데이터를 대기중인 유니트가 모니터링 하여 유니트의 절체에 대비하여 준비처리를 상시 행하는 프로세서와 그렇지 않은 프로세서가 혼재하여 설치하는 것이 필요하다. 이 방식에는 다른 프로세서가 필요로 하는 데이터는 자기의 보드상의 쌍방향 메모리에 기입하고, 필요로 하는 프로세서가 시스템 버스를 억세스 할 수 있게 한다. 이것을 위해 미러메모리를 사용한 활성화 유니트의 데이터를 기록할 필요가 있다. 미러메모리의 개념은 다음 것이 필요하다.

- 2개의 시스템 버스에 대해서 미러메모리는 상호 접속되어 진다.
- 시스템 버스 상에서 프로세스는 다른 미러메모리 내용뿐만 아니라 자기의 미러메모리의 내용을 억세스 한다.
- 미러메모리 자체는 액티브 보드로 되어 중재자가 없다. 따라서, 다른 유니트의 프로세서에 의한 억세스 및 출력시는 상대의 버스 사이클과 완전히 독립되어 시행되지 않으면 안 된다.
- 활성화 유니트의 프로세서끼리 상호데이터 입·출력시에 이것의 전체 데이터는 미러데이터 상에서 운전된다. 이것을 위해 시스템버스의 읽기 신호에 의해 미러메모리 상에는 데이터 상에 기록되는 것과 같다.
- 시스템 버스상의 어드레스 공간을 미러메모리는 별도의 공간의 어드레스에 교환시켜, 대기 유니트에 의한 미러메모리의 읽기가 소프트웨어에서 별도의 어드레스에 의해 선택되는 것같이 한다.

4. 결론

앞에서 언급한 방법은 서론에서도 언급한 바와 같이 일본철도연구소의 '나카무라'와 '다케시'가 제안한 방법에 대해 검토하였다. 현재 고속전철 기술개발 사업에서 개발하고 있는 신호시스템의 제어부분은 위에서 검토한 부호어, 고장검사회로, 고장표시회로 등을 사용을 검토하고 있다. 하드웨어의 구성은 가능한 한 상용보드와 VME 버스 시스템을 이용하여 개발할 예정에 있으며, 고장표시에 필요한 논리회로는 FPGA를 이용하여 개발할 예정에 있다.

<참고문헌>

[1] 中村英夫, 武子 淳, '保安制御計算機システムのフォールトトレランス設計', RTRI Report vol 7, No.5, pp. 55-62, 1993.

[2] George D. KRAFT and et al., 'Microprogrammed Control and Reliable Design of Small Computer', Bell Telephone Laboratory Inc.

[3] Barry W. Johnson, 'Design and Analysis of Fault Tolerant Digital Systems Addison-Wesley', 1989.

[4] Robert SPENCE, 'Tolerance Design of Electronic Circuits', Addison-Wesley, 1988.

[5] A. M. Usas, 'A Totally Self-checking Checker Design for the Detection of Error in Periodic Signals', IEEE Trans. on Computers, Vol. 5, pp. 483-489, 1975.