Journal of the Korean Institute of Illuminating and Electrical Installation Engineers (2015) 29(1) : 113~120
http://dx.doi.org/10.5207/JIEIE.2015.29.1.113 ISSN 1229-4691(Print) ISSN 2287-5034(Online)

논문 29-1-15

데이터 충돌을 고려한 네트워크형 분산 제어 시스템

(Network Type Distributed Control System with Considering Data Collision)

최군호*
(Goon-Ho Choi)

Abstract

Network type distributed control system uses a communication line which is named the BUS to exchange a data among the sub-systems. Usually, on the bus, only one data must be exited at one time, so the control algorithm to prevent collision or to manage a priority of data is important. Including CAN Protocol, many kind of FieldBus which are used for distributed control system, prevent data collision by controlling transmission time. But, a system which have to make a control signal or get a data from a sensor at fixed time will be met a problem when it is composed by using a network type distributed control structure. In this paper, some of these cases will be discussed and solutions be proposed for preventing a data collision. Also, using Arago Disk System which have a structure for inner loop control, the validity of the proposed methods will be verified.

Key Words : Network Type Distributed Control, CAN(Controller Area Network), Data Collision

1. 서 론

다양한 형태의 자동화된 생산 설비를 구축함에 있어서 각 요소 기기 간의 정보 교환이나 제어 신호 연결 등의 목적 때문에 다양한 형태의 수 많은 데이터의 교환이 필수적이고 이 때문에 각 시스템 사이의 전기적 또는 회로적인 연결 관계가 복잡할 수밖에 없다. 이러한 문제를 해결하기 위하여 버스를 이용한 통신 시스템이 발달하게 되었고 최근에는 버스 네트워크와 제어를 연계한 분산형 제어 시스템이 많이 쓰이고 있다. 이러한 연구는 [1-3] 등을 통해 보다 다양한 형태로 나타나고 있으며 최근에는 내부 궤환 제어 구조를 가지고 있는 시스템에 대한 연구도 보이고 있다[4].

버스 구조를 이용한 네트워크형 시스템은 기존에 물리적인 선로의 수를 획기적으로 줄일 수 있어서 설비 비용 절감, 설비 구축 시간 단축 및 유지 보수의 편이성 등 많은 장점을 가지고 있다. 하지만 버스의 특성상 단일 선로를 사용하고 있기 때문에 데이터의 충돌을 근본적으로 피할 수 있는 방법은 없다. 다만 충돌이 발생하였을 때 분산 제어용으로 이용되는 다양한 Fieldbus 들은 각각의 선로 점유 알고리즘을 이용하여 이 문제를 해결하고 있다. 이 방식은 첫째, CAN의 형태와 같이 버스를 우선 점유하는 노드에서 먼저 데이

* Main author : KyungSung Univ. Associate Prof.
Dept. of Electrical Eng.
Tel : 051-663-4798, Fax : 051-624-5980
E-mail : goonho@ks.ac.kr
Received : November 7, 2014
Accepted : November 25, 2014

터를 전송하고 다른 노드는 앞선 데이터 송신이 끝날 때까지 기다리고 있다가 버스가 비워지면 데이터를 전송하는 방식을 사용하는 경우, 둘째, 데이터의 충돌이 발생할 경우 두 데이터를 모두 손실 처리하고 일정 시간이 지난 후 다시 데이터를 전송하도록 하는 경우, 셋째, 우선 순위가 높은 노드가 지정되어 있어서 다른 노드에서 데이터가 전송 중이더라도 해당 노드의 데이터 전송을 무시(취소)하고 데이터를 전송하는 경우로 크게 나눌 수 있다[5-6].

이 중에서 가장 많이 사용되는 방법이 첫 번째로 언급한 "WAITING" 방식이라는 것이다. 이 방식은 일반적인 시스템의 경우 비교적 효과적인 방법이지만, 타이밍이 중요한 시스템의 경우, 특히 각 서브 시스템 사이의 데이터 교환이 빈번할 경우 문제가 발생할 가능성이 많다.

본 논문에서는 내부 궤환 제어기(Inner Loop Controller)를 가지는 제어 시스템이 네트워크형 분산 제어 구조로 구성될 때 이러한 문제가 발생할 가능성이 많다고 보고 이 경우 발생할 수 있는 문제를 분석하고 아울러 해결 방안을 제시하여 보고자 한다.

또한 이러한 조건을 만족하는 아라고 원판 시스템 제작하고 이 시스템을 CAN 프로토콜을 이용하여 제시된 방법으로 제어하여 봄으로써 본 논문의 유효성과 실용 가능성을 판단하여 보고자 한다.

2. 배경이론

2.1 내부 궤환 제어기 구조를 갖는 네트워크형 분산 제어 시스템

일반적으로 내부 궤환 제어기는 시스템의 제어 대상 플랜트를 여러 개의 서브 플랜트로 나눌 수 있고 그 중에서 특정한 일부분의 서브 플랜트의 안정도를 만족하도록 하거나 성능 개선을 위하여 해당 서브 플랜트에만 적용하는 특정한 제어기를 말한다(그림 1)[4].

이러한 형태의 제어 구조는 시스템 전체를 고려하여 제어 안정도와 성능 개선을 구현하는 제어기를 설계하는 것에 비하여 시스템에서 요구하는 제어 성능 중에서 중요한 부분이나 또는 제어가 어려운 부분만을 먼저 고려할 수 있다는 장점이 있다. 반면에 제어기 설계가 보다 어려워지고 또한 이와 같은 시스템을 실제로 구현할 경우 전체 제어 구조가 복잡해짐으로써 회로를 구성하거나 회로 선로를 연결하는 데에 복잡도가 증가하는 등의 문제점을 가지게 된다[7-9].

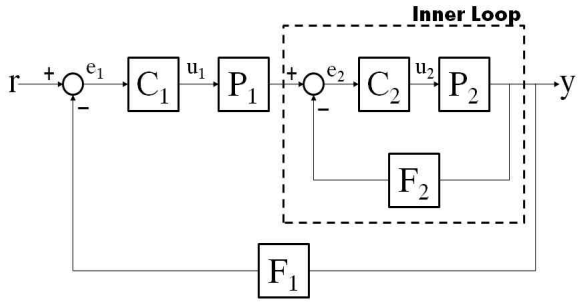

그림 1. 내부 궤환 제어 구조를 포함한 제어 시스템의 블록선도
Fig. 1. A block diagram of a control system with inner loop control structure

네트워크형 분산 제어 시스템은 각 플랜트들을 분리할 수 있고 이 플랜트 사이의 정보의 취사 선택 여부는 각 네트워크 제어기의 내부 운용 프로그램에 따라 좌우 되므로 시스템의 H/W적인 변화 없이 제어 구조의 변형이 가능하다[4]. 그림 2에서 보는 바와 같이 기본 H/W 구성은 (a)와 (b)가 동일하지만, 제어 형태에 따라 (a)에서는 C_1 제어기가 P_1 플랜트를 제어하고, C_2 제어기는 P_2 플랜트를 제어하는 형태로 서로 독립되어 있는 반면, (b)의 경우 C_2의 제어기의 역할은 같지만 C_1의 제어기가 P_1, P_2 시스템 전체를 관장하는 주 제어기라는 점, 이에 따라 C_2 제어기는 C_1 제어기의 내부 궤환 제어기로서 종속적인 관계를 갖는다는 것이 다르다. 이와 같이 네트워크형 분산 제어 시스템은 설치, 유지, 및 보수의 편리성과 기존 시스템에 비해 추가 요구 자원이 적어 수정 보완하여 사용하는 데에도 크게 문제가 발생하지 않는다는 등의 장점이 있지만, 단일 통신 선로를 사용함으로써 발생하는 신호의 충돌로 인한 데이터의 손실을 막을 수 있어야 한다는 단점이 있다[2,4,10].

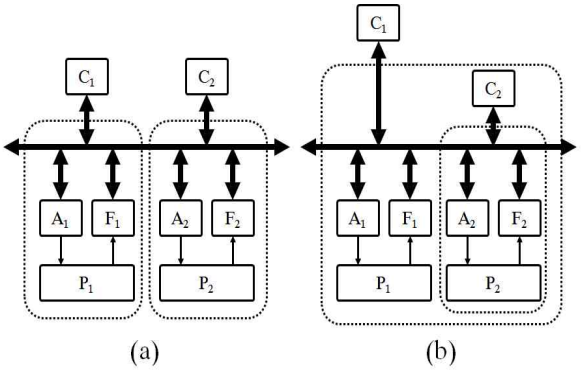

그림 2. (a) 일반적인 네트워크형 분산제어, (b) 내부 궤환 제어가 적용된 분산 시스템
Fig. 2. (a) Typical network type distributed control systems, (b) In case of, with a Inner Loop Control

2.2 아라고 원판 시스템의 네트워크형 분산 제어

아라고 원판 시스템은 1822년 프랑스의 물리학자 Dominique Françoi Jean Arago에 발견되어 알려진 아라고 원판 현상을 응용한 시스템으로, 비자성체 원판에 직결되어 있는 직류 전동기의 회전 속도를 제어함으로써 이로 인해 발생하는 유기 기전력에 의한 자석 막대의 회전력과 추의 무게가 특정 각도에서의 균형이 이루어지도록 함을 목적으로 하고 있다. 구현된 시스템 구조는 그림 3과 같다[4,11-12].

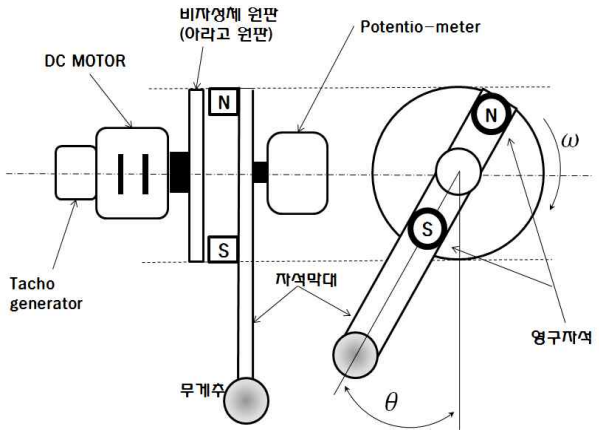

그림 3. 아라고 원판 시스템 구성도
Fig. 3. The configuration of Arago's Disk System

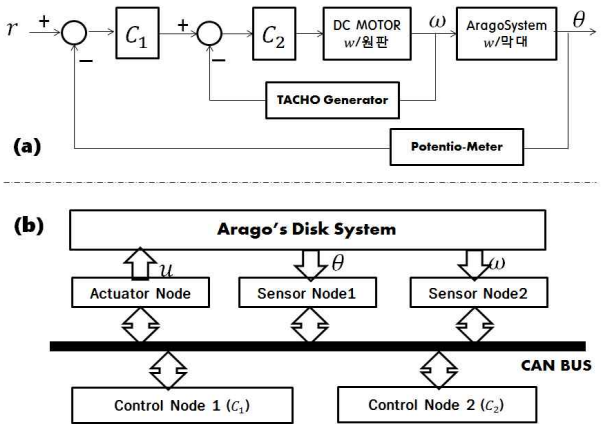

그림 4. (a) 일반적인 아라고 원판 시스템, (b) 네트워크형 아라고 원판 시스템
Fig. 4. (a) A normal type, (b) A network type Arago's Disk system

그림 4 (b)의 네트워크형 분산 제어 시스템이 적용된 경우를 보면, 기존의 방식(그림 4 (a))에 비해 간단한 구조를 가지는 것을 볼 수 있다. 직류 전동기의 속도 제어 부분을 담당하는 내부 궤환 제어기 부분은 SN2(Senser Node 2; Tachometer)를 통하여 현재 원판의 회전 속도값을 측정하고 이를 바탕으로 CN2(Control Node2; 직류 전동기 속도 제어기, C_2)에서 속도 지령을 계산하여 AN(Actuator Node, 전동기 구동 드라이버)로 내보내는 구조를 가지고 있다. 외부 궤환 제어 구조는 CN1(Control Node1; 주제어기, C_1)을 통해서 이루어지며, 이때 SN1(Sensor Node1; Potentiometer)을 통해 막대의 각도값을 읽어 이 값을 통해 원판의 회전 속도 지령값을 계산하고 이를 내부 궤환 제어기 부분(CN2)로 보냄으로써 전체 시스템을 제어하는 역할을 하게 된다.

3. 데이터 충돌을 고려한 네트워크형 분산 제어

본 논문에서 제시하고자 하는 알고리즘은 CAN 프로토콜에만 해당된 것이 아니며, CAN 방식과 같이 데이터가 충돌할 때, 기다렸다가 전송하는 방식의 다른 필드 버스들에도 동일하게 적용 가능함을 우선 설명하고자 한다.

3.1 내부 궤환 제어기를 가질 경우, 내부 및 외부 제어 주기 설정시 고려사항

내부 제어기를 가지는 시스템의 경우, 내부 제어기의 연산 시간이 외부 제어기 보다 느릴 경우 외부 제어기가 동작하기 위해 사용하는 결과 값이 실제로는 제어 결과가 반영되기 이전의 결과 값이 된다. 특히 디지털 제어기로 구현할 경우 외부 제어기의 출력값(내부 제어기의 입력)이 내부 제어기의 출력값에 반영이 되지 않은 상태에서 다시 외부 제어기가 동작할 경우 사실상 외부 제어기가 동작하기 위하여 센싱한 결과 값이 자신의 제어 출력이 반영되지 않은 값이 되므로 제어기의 동작에 오류를 발생시킬 가능성이 존재한다. 따라서 내부 루프 제어기를 포함하는 경우 내부 루프 제어기의 제어 주기(샘플링 시간)는 주 제어기의 제어 주기보다 빠르도록 구성해야 하며, 이러한 요건은 분산형 제어기 구조에서도 반드시 반영이 되어야 한다[4].

2.2절에서 제어기 C_1과 C_2는 각각 시스템의 주 제어기(외부 제어기)와 내부 제어기이다. 디지털 제어 시스템에서는 내부 제어기의 제어 주기를 결정하고, 이 결과를 고려하여 외부 제어기의 제어 주기를 결정한다. 이 비율을 정수배로 할 경우 일반적인 제어 구조에서는 각 서브시스템 사이의 통신 선로를 별도로 사용하기 때문에 이와 같은 설정이 전혀 문제가 되지 않는다. 그러나 BUS 형태의 통신 선로를 사용할 경우 데이터는 하나의 선로를 통해 전달되어야 하고, 이것은 내부 제어기의 데이터가 주 제어기의 데이터 교환이 필요한 시점이 같을 경우 충돌이 발생한다는 것을 의미한다. 물론 CAN 프로토콜은 이렇게 동시에 데이터의 교환이 필요할 경우 제어기의 우선 순위에 대한 사전 설정이 없다면 기본적으로 먼저 선로를 점유한 데이터를 처리하고 난 후 다음 데이터를 전송하는 방식을 사용함으로써 데이터의 손실을 방지한다. 하지만 이러한 선로 우선 점유로 인한 대기 시간의 발생은 제어 주기가 일정해야 하는 시스템의 경우 문제를 일으킬 가능성이 있다. 때문에 일반적으로는 선로가 비어있는 시간을 최대한 길게 하거나 주 제어기와 내부 제어기의 제어 주기를 정수배가 아닌 경우를 가지게 함으로써 최대한 데이터의 충돌이 적어지도록 하는 방법을 사용하여 왔다[4]. 하지만 이 역시 정밀하고 빠른 제어 시스템을 구성할 경우 최적의 제어 주기에 대한 조건을 갖출 수 없게 됨으로써 어느 정도의 성능 저하를 발생시킬 가능성을 남겨 두게 된다.

3.2 내부 및 외부 시작 시점에 따른 제어 주기 불일치

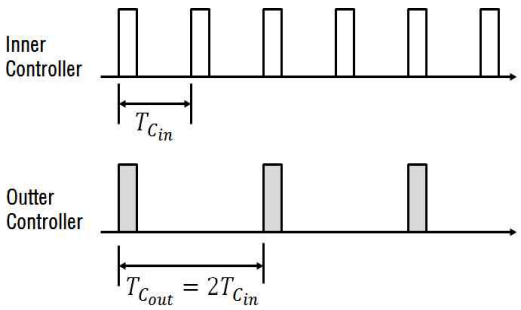

그림 5. 내부 및 외부 제어기의 제어 타이밍도
Fig. 5. Control timing diagram of a inner and a outer controller

그림 5는 내부 제어기와 외부 제어기의 제어 주기를 1 : 2로 설정한 경우이다. 이 경우 앞서 설명한 바와 같이 주 제어기의 제어 동작이 일어나는 시점에서 선로 상의 데이터의 충돌이 발생하게 된다. 이때 두 제어기의 선로 점유 시점에 따라 그림 6 (a), (b)와 같은 두 가지 형태가 나타난다.

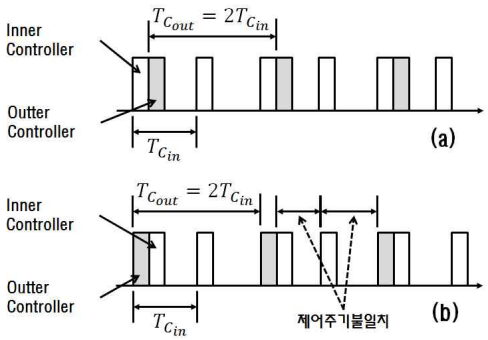

그림 6. 데이터 충돌로 인한 제어 주기 오류
Fig. 6. A control timing error occurred because of data collision

그림 6 (a)의 경우 내부 제어기가 전송 선로를 먼저 점유한 경우로 내부제어기의 데이터 송수신이 이루어진 다음에 순차적으로 외부 제어기의 제어 데이터의 송수신이 이루어진다. 이 경우에는 외부 주제어기의 제어 시간이 약간씩 뒤로 밀리는 현상이 있을 뿐 내부 및 외부 제어기 모두 제어 타이밍에는 문제가 없다. 그림 6 (b)의 경우에는 주 제어기가 전송선로를 우선 점유한 경우이다. 이 경우는 내부 제어기의 제어 타이밍이 주제어기에 밀려 일정 시간 지연되면서 내부 제어기의 제어 타이밍에 문제가 발생한다. 즉 외부 제어기의 선로 점유 시간만큼 지연이 발생하고 이 때문에 이 시점에서 내부 제어 타이밍이 어긋나게 되는 것이다. 따라서 주 제어기와 내부 제어기가 전송 선로의 우선권 다툼이 발생할 경우 반드시 내부 제어기가 우선권을 가질 수 있도록 할 필요가 있다. CAN 버스에서 선로의 우선권은 기본적으로 빈 BUS를 누가 먼저 점유하느냐에 달려있으므로 그림 7에서 보는 바와 같이 주 제어기의 제어 시작점을 내부 제어기와 α 시간만큼 지연을 주면 자연스럽게 이 문제를 해결할 수가 있다.

물론 이러한 문제가 발생하였을 경우 내부 제어기 제어 주기 사이에 주 제어기가 동작하도록 타이밍을 조작하는 것도 가능하지만, 이 경우는 각 제어기의 제어 간격이 넓고 제어 동작시간도 짧은 경우에 한정된다. 따라서 일반적인 경우로 보편화 할 수 없으므로 본 논문에서는 배제한다.

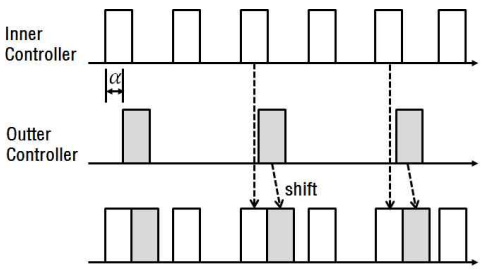

그림 7. 시작 시간 조정을 통한 오류 해결 방법
Fig. 7. A solution of control timing error by adjusting start time

3.3 패킷의 크기에 따른 데이터의 충돌

BUS 형태의 단일 선로를 사용하는 경우 또한 고려해야 할 사항이 각 서브시스템의 패킷 크기이다. 그림 8 (a), (b)를 보면, 통신 선로가 별도로 구성되어 있는 일반적인 제어 구조라면 주제어기의 전송 데이터 크기에 관계없이 각각의 제어기의 제어 주기에 문제가 발생하지 않는다. 그런데 이 구조를 BUS형 선로를 이용하는 네트워크형에 적용한 그림 8 (c)를 보면 외부 제어기의 전송 패킷이 선로를 점유하는 시간이 길기 때문에 내부 제어기의 제어 주기가 T_1, T_2로 어긋나는 것을 볼 수 있다.

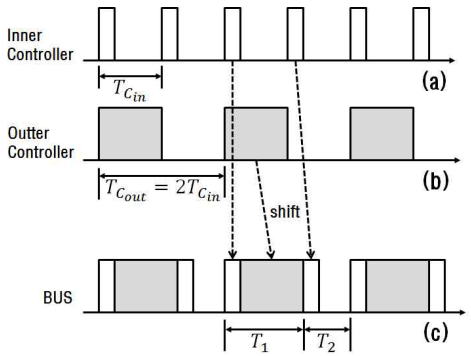

그림 8. 데이터 밀림 현상으로 인한 제어 주기 오류
Fig. 8. A control timing error occurred because of data shifting

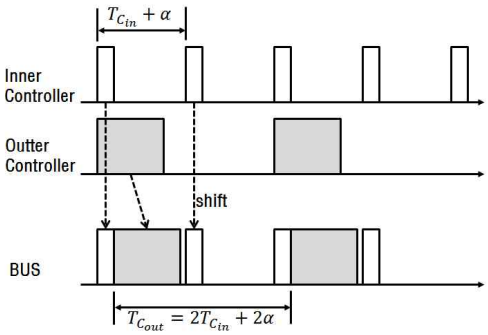

그림 9. 제어 주기 연장을 통한 오류 해결 방법
Fig. 9. A solution of control timing error by extending control time period

따라서 이 경우에는 내부 제어기의 제어 주기를 기존보다 더 길게 하여야 한다. 이것으로 내부 제어기의 제어 주기는 내부 제어기의 패킷 길이와 외부 제어기의 패킷 길이의 합 보다 크게 설정해야만 함을 알 수 있다. 그림 9에서 보면 이런 근거에 따라 내부 제어기

의 제어 주기를 α만큼 늘려 하면 주 제어기의 제어 주기도 2α만큼 늘어나야만 하는 것을 볼 수 있다(이 경우 3.2절의 방식에 따라 내부 제어기가 선로를 우선 점유하는 경우를 적용하였다).

4. 실험 구성, 비교 및 결과

네트워크형 분산 제어 구조의 유용성을 검증하기 위하여 실제 시스템 제작을 통한 실험을 진행한다. 다만 컴퓨터 시뮬레이션의 경우 [4,11-12] 등에서 구한 모델링과 실험 조건 등이 동일하므로 본 논문에서는 따로 컴퓨터 시뮬레이션 결과를 제시하지 않는다.

4.1 실험 시스템

그림 10은 제작된 아라고 원판 시스템이다. 직류 전동기는 Tamakawa사의 TS3881이며 여기에는 직결로 7V/1000rpm의 출력을 갖는 Tacho Generator가 연결되어 있다. 원판은 지름 150mm의 알루미늄이며, 자석 막대에는 3000G의 영구자석이 설치되어 있다. 막대와 직결로 포텐셔미터(Copal사의 J40S)를 연결하여 막대의 회전 각도를 측정한다.

그림 10. 제작된 아라고 원판 시스템
Fig. 10. Designed Arago's Disk system

그림 11은 아라고 원판 시스템의 실험 시스템의 회로 구성 부분이다. 모두 5개의 CAN 노드로 구성되어 있다. 각각의 노드는 AT90CAN128 기반의 ATMEL 사의 DVK90CAN 개발 보드를 이용하여 하나의 보드에서 CAN 노드와 내부 제어 프로그램을 통합하여 구성할 수 있도록 하였으며, 모든 노드는 하나의 CANBUS와 연결된 단일 통신 선로를 사용하도록 구성하였다. 각 노드 컨트롤러의 역할은 내부 Firmware에 의하여 지정되어 있으며, 특히 CN1(주 제어기)과 CN2(내부 제어기)의 제어 타이밍을 변경할 수 있도록 구성하여 다양한 상황에 대한 실험이 가능하도록 하였다.

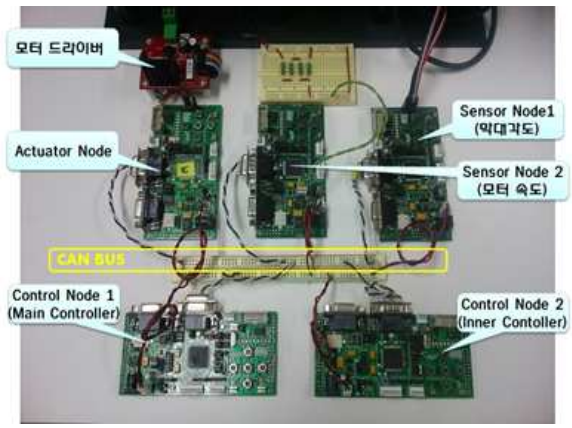

그림 11. 전체 실험 시스템
Fig. 11. Circuits for a experimental

4.2 내부 및 외부 제어기의 제어 주기 결정을 위한 패킷 분석

제어 주기를 결정하기 위하여 각 노드 제어기의 패킷의 크기를 분석하였다. 각 노드 제어기가 가장 길게 동작하는 상황을 고려하여 일정한 시간 간격으로 CAN 버스 선로를 점유하도록 하였고, 이때 CAN 데이터를 분석하여 각 노드 제어기에서 패킷의 최대 길이를 측정하였다. 측정 도구로 CAN 분석 모듈을 장착한 MSO-3014(Tektronix) 오실로스코프를 이용하였다. CAN 분석 모듈은 오실로스코프 상에서 패킷 데이터의 분석을 도와 주는 일종의 측정 서브 모듈이다.

그림 12는 내부 및 외부 제어기의 패킷 분석 결과로 각각 1.2154msec, 1.2674ms의 시간이 소요되는 것으로 측정되었고, 이 값을 3.3절의 내용에 적용하면 내부 제어기의 제어 주기는 최소 약 2.5msec 이상이어야 한

다는 것을 알 수 있다. 본 논문에서는 여유분을 고려하여 내부 제어기의 제어 주기를 5msec로 결정하였다. 각 노드 제어기의 제어 주기는 노드 제어기 내부 클록에 동기하여 동작하도록 함으로써 외부 잡음 등에 의하여 발생할 수 있는 일시적인 제어 주기 놓침 등의 현상을 보완하도록 한다.

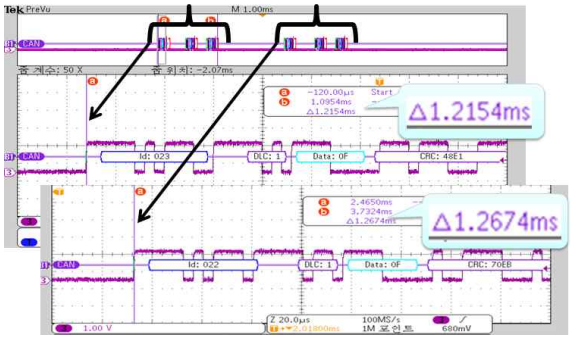

그림 12. 내부 및 외부 제어기의 패킷 분석 결과
Fig. 12. A packet analysis result of a internal and a external controller

4.3 내부 및 외부 제어기의 제어 주기비에 따른 실험 결과

실험은 내부 및 외부 제어기의 제어 주기를 각각 5msec와 20msec(1 : 4)로 결정하고 이에 맞추어 내부 및 외부 제어기의 값을 조정한 것을 기준으로 하였다. 또한 3.2절에서 제시한 바와 같이 내부 제어기가 외부 제어기 보다 먼저 선로를 점유하도록 외부 제어기는 처음 전원이 인가된 후 1msec의 시간 지연 후에 동작하도록 내부 프로그래밍 하였다. 그림 13은 이 실험 결과 파형으로 약 1초 정도의 시간에 목표에 도달하는 것을 알 수 있다.

그림 14는 제어 주기를 다르게 하여 비교 실험한 경우이다. 여기에서 제어 주기를 짧게 하거나(a) 길게 (c), (d)할 경우에 기준(b)과 비교하여 모두 제어 성능이 떨어지는 것을 볼 수 있다. 특히 (d)의 경우 불안정한 진동이 계속되었고 그 이상의 제어 주기에서는 제어를 할 수 없는 상황이 발생하였다. 이는 디지털 제어기의 경우 같은 물리적 플랜트라도 샘플링 시간이 달라지면 다른 플랜트인 것처럼 계산되는 것과 유사한 특성을 보이는 것으로 판단된다.

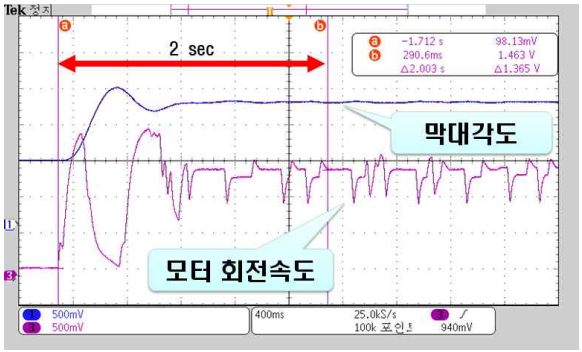

그림 13. 실험 시스템을 이용한 막대 각도 제어 실험 결과
Fig. 13. A angle control result of the Arago Disk System using the experimental system

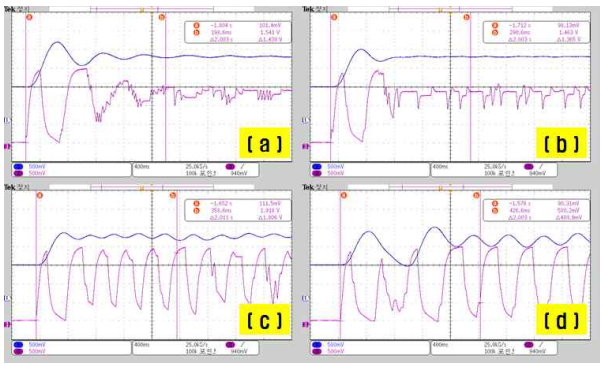

그림 14. 외부 제어기 주기에 따른 실험 결과 : (a) 10msec, (a) 20msec, (c) 30msec, (d) 50msec
Fig. 14. Experimental results at each periodic time, (a) 10msec, (a) 20msec, (c) 30msec, (d) 50msec

5. 결 론

현재까지 네트워크형 분산 제어 구조를 사용하는 경우 제어 주기의 설정이나 그 설정 논리가 다분히 실험적이고 임의적인 경우가 많았다. 또한 제어 주기 또는 센싱 주기를 일정하게 가져야만 하는 시스템을 네트워크형 분산 제어 구조로 만들고자 할 때, 버스에서 데이터 충돌로 발생하는 타이밍 오류 등의 문제가 발생하는 것에 대한 논리적 해결 방안도 미비하였다.

본 논문에서는 서로 다른 제어 주기를 가질 수밖에

없는 내부 궤환 제어구조의 시스템을 CAN 프로토콜을 이용하여 네트워크형 분산 제어할 경우 발생할 수 있는 문제점을 살펴보고 이를 해결하기 위한 몇 가지 방안을 제시함으로써 조금 더 이론적인 접근이 가능하도록 하고자 하였다.

네트워크형 분산 제어 시스템의 경우 각 서브 시스템의 제어 안정도 보장 등의 문제, 제어 노드가 고장이 발생하였을 때 해결 방안 등에 대한 문제 등을 보완한다면, 이를 통해 다른 산업 전반의 더욱 다양한 분야로의 확장 및 응용이 가능하리라 예상한다.

감사의 글

이 논문은 2014학년도 경성대학교 신임교수정착연구비에 의하여 연구되었음.

References

[1] Feng-Li Lian, James Moyne, and Dawn Tilbury, "Network Design Consideration for Distributed Control Systems," IEEE Trans. Control sys. Tech., vol.10, No.2, pp.297-307, Mar. 2002.
[2] J. H. Jung, J. H. Lee, T. D. Park, and K. H. Park, "Development of a Network-Based Distributed Control System," CASS 2008 , pp.777-782, 2008.
[3] Robin Qiu and Phillip Laplante, "Design and Development of Component-based Equipment Connectivity for Semiconductor Manufacturing," The 4th .Int. Confr. On Control and Automation (ICCA'03), Montreal, Canada, June, 2003.
[4] G. H. Choi, "Network Type Distributed Control of a System with Inner Loop Control Structure," J. of the KIIEE, Vol.28, No.2, 2014.
[5] Ken Tindell, Alan Burns, "Guaranteed Message Latencies for Distributed Safety-Critical Hard Real Time Control Network," report YCS229, Department of Computer Science, University of York, May 1994.
[6] J. Yun, S. Nam, K. W. Kim, and S. Lee, "Evaluation of Network Protocols for Automotive Data Communication", J. of Control Automation and Systems Engineering, vol.3, no.6, pp.632-638, Dec. 1997.
[7] H. J. Kim, J. W. Kim, and K. W. Lee, "Flight control of a small unmanned aerial vehicle using a dynamic compensator," The journal of Korea Navigation Institute, Vol.16, No.4, 2012.
[8] B. K. Kim, and W. K. Chung, "Design of Robust Motion Controllers with Internal-Loop Compensator," The J. of KSME, Vol.25, No.10, 2001.
[9] H. J. Yeo, "Design of a Robust Controller Using Disturbance Rejection Controller," The J. of KAIS, Vol.7, No.2, 2006.
[10] J. H. Jung, S. Y. Choi, and K. H. Park, "A Study on the Power System Control and Monitoring Technique Using CAN," The Transactions of the Korean Institute of Electrical Engineers, Vol.52D, No.5. pp.268-276, May 2003.
[11] G. H. Choi, A Study on the Position Control of Arago's Disk Systems, Master Thesis, SungKyunKwan Univ., 1994.
[12] W. M. Lee, Control of arago's disk system using CAN(Controller Area Network), Master Thesis, SungKyunKwan Univ., 2002.

◇ 저자소개 ◇

최군호(崔君鎬)
1969년 11월 20일생. 1993년 성균대학교 졸업. 1995년, 1999년 동 대학원 전기공학과 제어공학 전공 석사, 박사학위 수여. 1999~2005년 (주)한미반도체, (주)한울로보틱스, (주)동부로봇 재직. 2006~2011년 한국기술교육대 대우교수. 2012년~현재 경성대학교 전기공학과 교수.