論文 9-1-8

CAN 네트워크를 이용한 단일 프로세서에 의한 복수 인버터 구현에 관한 연구

鄭義憲†, 李賢寧*, 李弘熙**, 全泰園**

A Study on Driving Dual Inverters with Single Processor Using Controller Area Network

Ui-Hurn Jeong, Hyun-Young Lee, Hong-Hee Lee, and Tae-Won Chun

요 약

일반적으로 두 대의 전동기를 독립적으로 동시에 구동하려면 2개의 독립적인 제어용 프로세서를 사용해 각각의 전력회로를 구동해야 한다. 본 논문에서는 CAN 네트워크를 이용해 한 개의 제어기만으로 두 대의 인버터 전력회로를 동시에 제어할 수 있는 단일 프로세서에 의한 복수 인버터 제어기법을 제안했다. 이 시스템은 두 대의 전동기를 한 개의 프로세서를 사용해 제어하기 때문에 두 대의 전동기를 연동하여 운전할 경우에도 별도의 인터페이스 장치나 새로운 프로세서의 추가 없이 소프트웨어적으로 간단히 구현할 수 있다. 제안된 시스템을 구현하고 실험을 통해 그 타당성을 입증해 보였다.

ABSTRACT

Two processors are generally used to drive the power circuits for controlling the dual motors independently. In this paper, we propose the new control scheme to drive dual inverters using only one controller with the aid of CAN network. The proposed system is very useful compared to conventional techniques especially in case of controlling the combined dual motors because the control algorithm can be implemented by the software program only without any additional processor or hardware interfacing. The proposed system is implemented and verified experimentally.

Key Words : Dual motors, Dual inverters, CAN Network, CAN protocol, Bit time

1. 서 론

두 대의 유도전동기를 독립적으로 제어하려면 두 대의 인버터가 필요하고 이들 인버터에는 적어도 한 개 이상의 제어를 위한 프로세서가 있어야 한다는 것이 일반적인 관념이다. 특히 자동화 공장에서 사용되는 전동기는 서로 연관 관계를 가지고 운전되는 경우가 많은데, 두 대의 전동기가 협조하여 하나의 작업을 수행하려면 두 대의 전동기 사이에 실시간 정보 교환이 필요하다. 이 경우 두 대의 전동기가 정보를 교환하려면 듀얼 포트 램(dual port RAM)과 같은 소자가 필요하고 두 대의 전동기의 협조제어를 위한 별도의 프로세서가 필요하게 되어 대단히 비효율적이다. 특히, 제어하고자 하는 전동기의 제어장치가 서로 멀리 떨어져 있을 경우 제어명령이나 속도 또는 위치정보의 교환을

†교신저자 : 학생회원, 울산대 전자정보시스템공학부 박사과정
E-mail : uihern@hotmail.com
* 학생회원, 울산대 전기전자정보시스템공학부 석사과정
** 정회원, 울산대 전기전자정보시스템공학부 교수
접수일자 : 2003. 5. 20 1차 심사 : 2003. 6. 3
2차 심사 : 2003. 8. 26 심사완료 : 2003. 11. 14

위해 많은 신호선들이 필요하게 되어 배선구조가 복잡해지고 노이즈 및 유지보수의 어려움으로 인해 시스템의 신뢰성과 유연성이 떨어지고 경제적 측면에서도 부담이 된다.

이러한 단점을 개선하기 위해 본 논문에서는 CAN (Controller Area Network)을 이용한 실시간 네트워크 구축을 바탕으로 한 개의 제어기만으로도 두 대의 전동기를 동시에 독립적으로 제어할 수 있는 기법을 개발하여 시스템의 경제성과 효율을 증대시키고자 한다. 즉 각각의 전동기를 제어하는데 필요한 전류 및 속도 정보를 CAN 네트워크를 통해 제어용 프로세서로 전달하면 이 프로세서는 두 대의 전동기를 제어하는데 필요한 스위칭 정보를 각각의 전력회로로 전달하여 궁극적으로 하나의 프로세서로 두 대의 인버터 전력회로를 제어할 수 있도록 했다.

CAN은 차량용 네트워크 프로토콜로 1980년 초 독일의 R. Bosch GmbH에 의해 연구, 개발되었으며 1992년에 와서 Mercedes S-class 차량 내에서 고속 통신이 요구되는 엔진제어기, 기어박스제어기, dash board 사이의 네트워크와 저속 데이터 통신으로도 가능한 에어컨 시스템의 분산제어에 처음 적용되었다. 나아가 1995년 이후에는 차량 분야에서 가장 인정받는 프로토콜로 자리 잡았으며 지난 수년간 차량 내의 열악한 환경에서도 안정적으로 동작하여 그 신뢰도를 인정받았고 가격 대비 성능비가 우수하여 공장자동화용 네트워크에도 적용되고 있는 추세이다[1]-[3].

본 논문에서는 두 대의 인버터를 사용하여 복수 유도전동기를 구동함에 있어 각각의 인버터 전력회로를 제어하기 위해 TMS320C32 프로세서 하나만을 사용하고 벡터제어기법을 바탕으로 각각의 유도전동기에 대한 독립적인 속도제어 시스템을 구현했다. 제안된 시스템을 구현하고 실험적으로 그 타당성을 입증해 보였다.

2. CAN

2.1 CAN 프로토콜

CAN은 CSMA/CD+AMP (Carrier Sense Multiple Access / Collision Detection + Arbitration on Message Priority) 프로토콜을 가지는데 이는 IEEE 802.3 CSMA/CD 프로토콜과 유사하며 ISO/OSI 7계층 중 1, 2계층인 물리계층과 데이터 링크 계층만을 표준화하여 제시하고 있다[4]-[7]. 이를 통해 서로 다른 제조사에서 양산되는 디바이스들과 완벽한 호환성을 유지하게 된다. 하지만 응용계층은 표준화 작업에 앞서 시장성이 있는 응용분야에 따라 독자적인 그룹들이 형성되어 있으며 CANOpen, DeviceNet, CANKingdom 등이 있다. CAN 사양서에 제시된 물리계층의 전송속도는 거리에 따라 차이가 있으나 최대 1Mbps의 전송속도를 가진다.

CAN은 식별자(Identifier)의 길이에 따라 기본형인 CAN 2.0A와 확장형인 2.0B로 구분된다. 2.0A의 경우 11bits, 2.0B의 경우 29bits 길이의 식별자를 가지고 있으며 2.0B를 지원하는 제품에서는 보다 확장된 기능들이 지원되고 있다. CAN 프로토콜은 기본적으로 데이터 프레임, 원격 프레임, 에러 프레임, 오버로드 프레임을 제공하고 있으며 본 논문에서 사용한 CAN 2.0A의 데이터 프레임은 그림 1과 같다[5],[7].

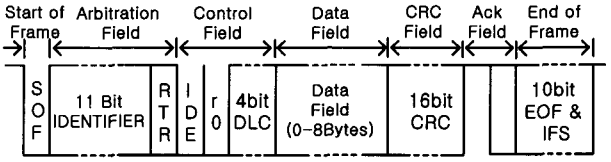

그림 1 CAN 데이터 프레임
Fig. 1 CAN data frame

CAN 프로토콜은 표준 통신 프로토콜이므로 서로 다른 제작사에서 만든 소자일지라도 네트워크에 간단히 연결시킬 수 있으며 고장 검출과 응답도 CAN 소자 내에서 매우 효과적으로 처리된다. 아울러 CAN 통신 전용 소자가 통신을 전적으로 감당하고 있기 때문에 CPU가 통신에 따른 부담 없이 다른 작업을 수행할 수 있다. 또한, 표준 프로토콜이므로 시장성이 뛰어나고 이로 인해 많은 업체들이 CAN 소자를 제작하고 있으며 가격 또한 저렴하다.

그림 1에서 나타낸 식별자(Identifier)는 버스 사용권에 대한 우선순위를 결정하게 되는데 지정된 값이 작을수록 높은 우선순위를 갖는다. 가령 2개 이상의 노드가 동시에 메시지를 전송하려고 할 경우, 각각의 식별자는 서로 대응하는 비트값을 비교하여 로직 '0'이 우선권을 갖게 되고, 우선권을 확보한 노드는 전송상의 지연없이 남은 데이터를 계속해서 전송하는 NBA (Non-deductive Bitwise Arbitration) 방식을 사용하고 있다[4]. 이때 우선 순위가 낮은 노드는 데이터 전송을 즉각 중단한 후 버스 감지상태로 들어가게 되고 버스상에 캐리어가 감지되지 않으면 버스 사용권을 획득하기 위해 데이터 프레임을 재전송하기 시작한다. 따라서 CAN의 경우 식별자의 지정은 전체 네트워크 상의 노드들이 정해진 시간 이내에 데이터 전송을 보장받기

위해 대단히 중요하며 이에 대한 연구가 계속되고 있다.[8],[9] 또한, 식별자를 오프라인 상에서 미리 결정할 경우 개별 노드에 대한 주소로도 사용될 수 있으며 노드가 필요로 하지 않는 메시지를 필터하는 기능 등이 있다.

2.2 CAN 네트워크 구성

그림 2는 CAN을 사용해 통신 노드를 구성할 경우 이용되는 3가지 방식을 도식적으로 나타내고 있다. 그림 2의 (a), (b), (c)는 각각 Stand alone, Integrated (micro-controller based), Serial Link I/O Device CAN 제어기를 나타내고 있는데 Stand alone 형태로 구성하는 것이 가장 일반적이며 본 논문에도 제어용 프로세서의 CAN 인터페이스를 위해 이 방식을 사용하였다.

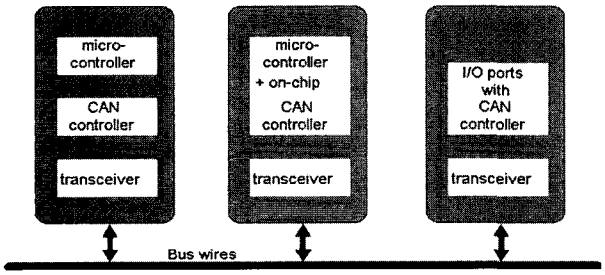

그림 2 CAN 노드 구성 예
(a) 독립형 (b) 집적형 (c) 직렬 연결 입출력형
Fig. 2 An example of CAN node configuration
(a) Stand alone (b) Integrated
(c) Serial Link I/O Architecture

CAN 프로토콜은 기본적으로 개방형 구조를 지향하고 있어 구성상에는 큰 문제가 없지만 안정적인 동작을 위해서는 비트 타이밍 계산에 각별한 주의가 요구된다. 즉, 같은 조건으로 동작하는 노드일지라도 거리에 따른 신호의 전달 지연과 찌그러짐(jitter)으로 인해 동기상의 문제점이 발생할 수 있으므로 정해진 규격에 따라 버스타이밍 레지스터 값을 정확히 설정하여야 한다. 그림 3은 본 논문에서 사용한 CAN 제어기 SJA1000의 버스타이밍 레지스터를 보여주고 있다.[10]

그림 3의 BTR0에서 SJW (Synchronization Jump Width)는 비트타임의 찌그러짐이 발생할 경우 이를 보정하는데 사용되고 BRP(Baud Rate Prescaler)는 각 통신 노드에서 사용되는 비트타임을 조정하므로 클럭 주파수가 다를 경우 보드레이트를 일치시키기 위해 사용될 수 있다.

BTR0

BIT 7	BIT 6	BIT 5	BIT 4	BIT 3	BIT 2	BIT 1	BIT 0
SJW.1	SJW.0	BRP.5	BRP.4	BRP.3	BRP.2	BRP.1	BRP.0

BTR1

BIT 7	BIT 6	BIT 5	BIT 4	BIT 3	BIT 2	BIT 1	BIT 0
SAM	TSEG2.2	TSEG2.1	TSEG2.0	TSEG1.3	TSEG1.2	TSEG1.1	TSEG1.0

그림 3 버스타이밍 레지스터
Fig. 3 Bus timing registers

BTR1에서 SAM(Sample Point)은 송·수신기가 비트값을 감지하는 시점을 결정하는데 사용되고 TSEG2 (Time segment1) 및 TSEG1(Time segment2)는 하나의 비트타임에 들어가는 퀀텀의 수를 결정하는데 사용된다.

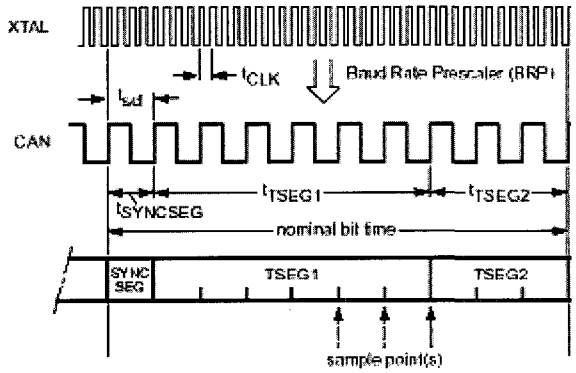

그림 4 비트타임의 구성
Fig. 4 General structure of a bit time

그림 4는 그림 3에서 결정한 버스타이밍 레지스터에 의한 비트타임을 보여주고 있다. 식(1)~(6)은 그림 4와 같은 비트타임 상에서의 각종 시간값들에 대한 계산식을 보여주고 있다.[10]

$$t_{scl} = 2 \times t_{clk} \times (32 \times BRP.5 + 16 \times BRP.4 + 8 \times BRP.3 + 4 \times BRP.2 + 2 \times BRP.1 + BRP.0 + 1) \quad (1)$$

$$t_{clk} = \frac{1}{f_{XTAL}} \quad (2)$$

$$t_{SJW} = t_{clk} \times (2 \times SJW.1 + SJW.0 + 1) \quad (3)$$

$$t_{SYNCSEG} = 1 \times t_{scl} \quad (4)$$

$$t_{TSEG1} = t_{scl} \times (8 \times TSEG1.3 + 4 \times TSEG1.2 + 2 \times TSEG1.1 + TSEG1.0 + 1) \quad (5)$$

$$t_{TSEG2} = t_{scl} \times (4 \times TSEG2.2 + 2 \times TSEG2.1 + TSEG2.0 + 1) \quad (6)$$

CAN의 경우 최대 전송속도는 1Mbps이나 본 논문에서는 복수전동기의 제어 성능을 높이기 위해 2Mbps의 속도로 통신하고 있다. 비교적 짧은 거리에서 실험하였으나 이에 따른 오동작은 없었다. 그런데 SJA1000의 사양에 의하면 16MHz 입력 주파수를 제시하고 있는데 이 주파수로는 2Mbps의 전송속도를 얻을 수 없어 20MHz의 입력 클럭을 사용했다. 입력주파수가 20MHz일 경우 전송속도를 2Mbps로 하기 위해 식 (1)~(6)을 사용하여 버스타이밍 레지스터 값을 찾으면 표 1과 같이 된다.

표 1 SJA1000의 버스타이밍 레지스터
Table 1 Bus timing register of SJA1000

f_{XTAL}	20MHz	SAM	0
SJW	0	TSEG1	1
BRP	1	TSEG2	1

표 1에서 설정한 값에 따라 식 (7)과 같이 비트타임을 계산해 보면 0.5μS가 되므로 전송속도는 2Mbps로 됨을 알 수 있다.

$$\text{비트타임} = t_{SYNCSEG} + t_{TSEG1} + t_{TSEG2} = 0.5\mu S \quad (7)$$

그림 5와 6은 CAN을 이용해 통신을 할 경우 송·수신 노드에서의 데이터 송·수신 알고리즘을 나타내고 있다.

3. 시스템 구성

그림 7은 본 논문에서 제안하고 있는 단일 프로세서에 의한 복수전동기 제어시스템을 나타내고 있다. 본 시스템에서 사용된 유도전동기 제어기법은 정밀, 속응 제어용으로 많이 사용되는 벡터제어 방식이다.[11],[12] 그림 7에서 알 수 있는 바와 같이 TMS320C32 DSP 1개로 2대의 유도전동기 벡터제어 알고리즘을 구현하고 있다. 즉, 단일 프로세서 TMS320C32 DSP에서 각각의 인버터를 구동하기 위한 게이트 신호를 만들어내고 이를 CAN 네트워크를 통해 게이트 구동회로로 전달하고 있다.

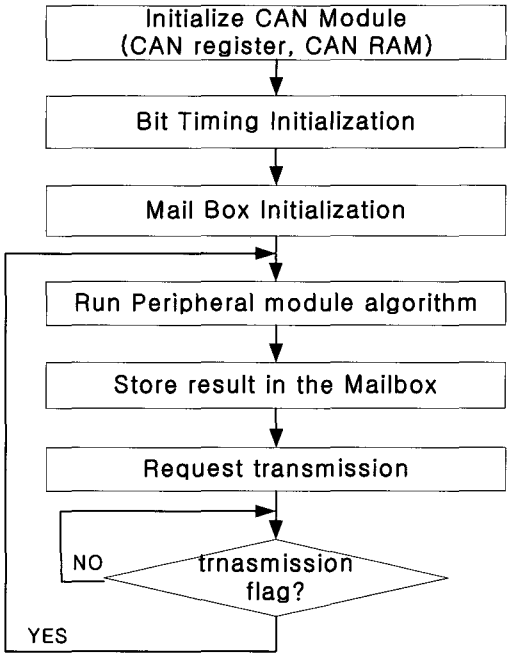

그림 5 CAN 데이터 송신 알고리즘
Fig. 5 CAN data transmission algorithm

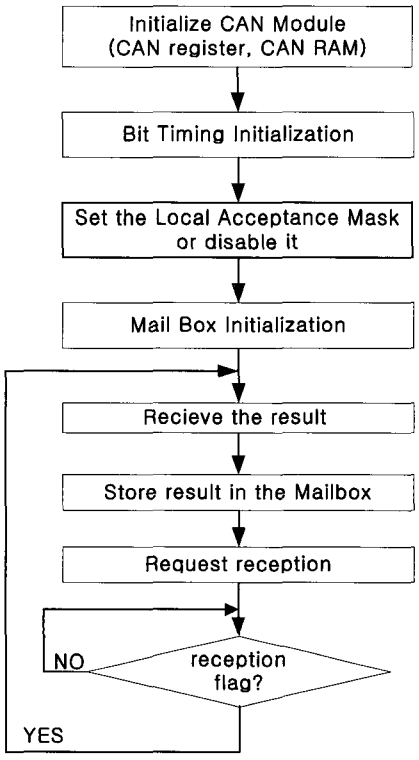

그림 6 CAN 데이터 수신 알고리즘
Fig. 6 CAN data receiving algorithm

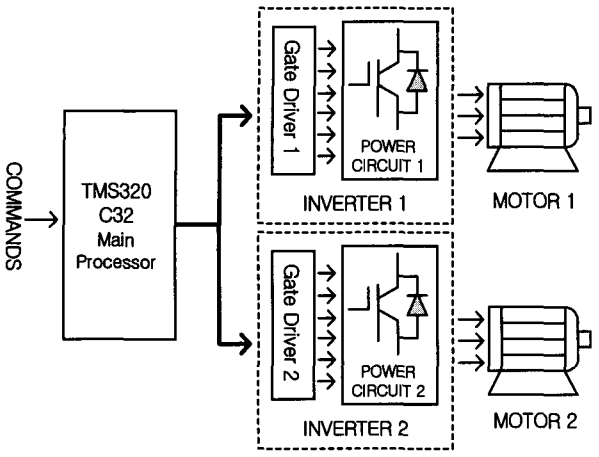

그림 7 시스템 블록도
Fig. 7 System block diagram

또한, 벡터제어를 위해서는 각각의 유도전동기의 속도 및 전류 정보를 알아야 하는데 이러한 정보들도 CAN 네트워크를 통해 읽어와야 한다. 이 경우 각종 데이터의 흐름은 그림 8과 같다.

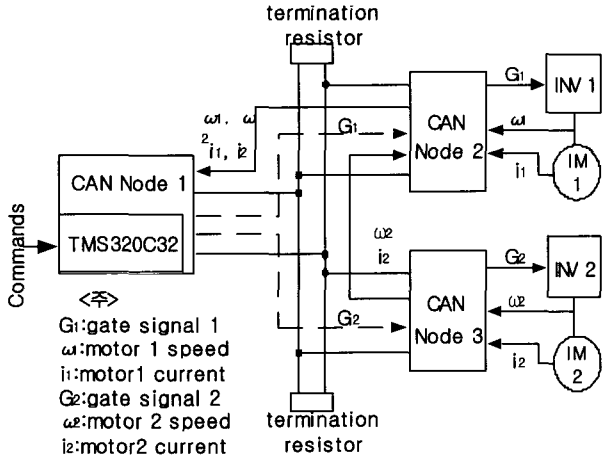

그림 8 CAN 노드 사이의 데이터 흐름
Fig. 8 Data flow between CAN nodes

CAN 프로토콜은 데이터를 전송할 경우 그림 1의 데이터 프레임에서 알 수 있는 바와 같이 CAN 2.0A의 경우 비트 스터핑(bit stuffing)을 고려하지 않더라도 실제 데이터 이외에 부가되는 비트 수가 47비트나 된다.[4],[6] 그림 8에서 전동기 2의 속도 및 전류정보를 DSP 노드(CAN 노드 1)에서 노드 2, 3으로부터 따로 따로 읽어올 경우 데이터를 전송하기 위해 부가되는 비트수가 너무 많고 데이터 전달시간이 너무 길어 실시간 제어가 어렵게 될 수 있다. 본 논문에서 사용하고 있는 전동기 속도 및 전류 메시지의 길이는 4바이트인데 이를 각각 따로 전송할 경우 최소한 158비트 타임을 필요로 한다. 이때 코딩 효율(coding efficient)을 살펴보면 비트 스터핑(bit stuffing)이 없을 경우라도 코딩 효율(coding efficient) η=32/(32+47) = 0.405로 대단히 낮다. 이러한 단점을 없애기 위해 본 연구에서는 DSP가 유도전동기 벡터제어 알고리즘을 수행하고 있는 동안에 CAN 노드2가 CAN 노드3을 통해 전동기 2의 정보를 미리 읽어와 전동기 1의 정보와 함께 데이터화하여 정해진 시점에 DSP 소자가 모든 전동기의 정보를 CAN 노드 2에서 읽어가도록 했다. 이렇게 할 경우 데이터 전달시간은 데이터량만 늘어나게 되므로 총 111비트타임에 불과하고 코딩 효율도 η=64/(64+47) = 0.577로 약 42%정도 코딩 효율이 향상된다. TMS320C32에서 수행된 벡터제어 알고리즘에 의한 전력회로의 스위칭 패턴에 대한 정보는 CAN 네트워크에 의해 각각의 게이트 구동회로로 인가된다.

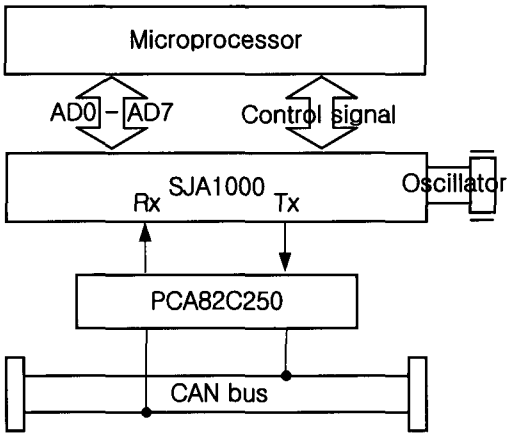

그림 9 CAN 노드의 구성
Fig. 9 Configuration of CAN Nodes

네트워크 제어를 위한 CAN 제어기는 Phillips사의 SJA1000를 사용했으며 본 논문에서 사용한 단일 제어기 형태의 CAN 노드의 구성도는 그림 9와 같다.[7],[10]

표 2는 본 논문에서 사용된 CAN 및 데이터 전송 사양을 나타내고 있다.

단일 프로세서 TMS320C32에 의한 유도전동기 제어 순서도는 그림 10과 같다. 즉, 소프트웨어 타이머를 이용해 일정 시간 주기를 가지면서 2대의 전동기를 동시에 제어하고 있으며 제어 주기는 450μs이다. 이 시간은 디지털 방식으로 고성능 전류제어를 구현하기에

는 다소 길지만 어셈블리어로 제어프로그램을 짠다면 제어 주기를 현저히 단축시킬 수 있을 것이다.

표 2 CAN 데이터 전송 사양
Table 2 Specification of CAN data transmission

CAN 프로토콜	2.0 A
데이터 전송 속도	2 Mbps
게이트 정보 메시지 길이	8 Bytes
속도 및 전류메시지 길이	8 Bytes
게이트 정보 전송 주기	450us
속도 정보전송 주기	1.8ms
전류 정보전송 주기	450us

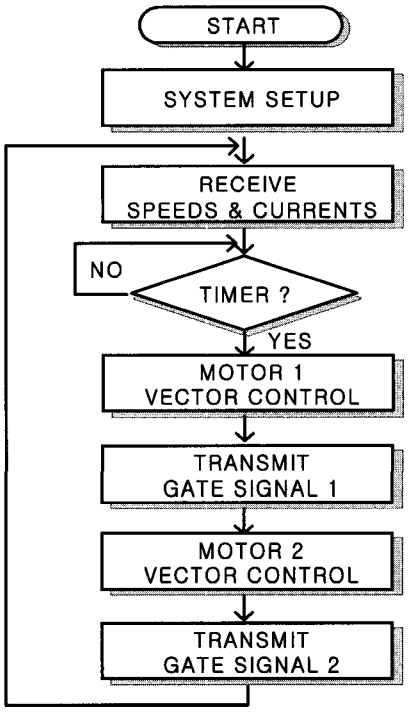

그림 10 전동기 제어 순서도
Fig. 10 Motor control sequence

4. 실험 결과

그림 7과 같은 단일 프로세서에 의한 복수 인버터 제어 시스템을 구성하고 그 타당성을 실험을 통해 살펴보고자 한다. 표 3은 본 실험에서 사용된 유도전동기 관련 정수들이다.

표 3 유도전동기 정수
Table 3 Induction motor parameters

	MOTOR 1	MOTOR 2
용량/전압	1.5kW / 220V	2.2kW / 220V
고정자 저항	4.5 Ω	0.6865 Ω
회전자 저항	5.32 Ω	0.6460 Ω
고정자 리액턴스	0.0145 H	0.08397 H
회전자 리액턴스	0.0171 H	0.08528 H
상호 리액턴스	0.0148 H	0.08136 H
극 수	4 poles	4 poles

그림 11은 2바이트의 데이터를 전송할 경우 최대 전송속도 2Mbps에서의 비트타이밍의 한 예를 보여주고 있는데 스터핑 비트가 없을 경우 63 비트타임이 소요됨을 고려하면 정확한 전송이 이루어짐을 알 수 있다.

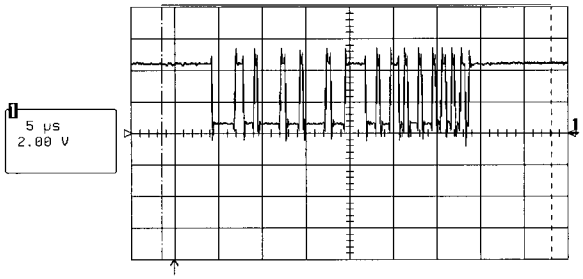

그림 11 비트타이밍 예
Fig. 11 Example of the bit timing

그림 12는 CAN 네트워크 상에서의 데이터 전송 파형을 나타낸 것인데 그림의 아래쪽 파형은 노드 2로부터 전동기 1, 전동기 2의 속도 및 전류 정보를 CAN 노드 1이 수신했을 때 제어기(TMS320C32)에서 발생하는 수신 인터럽트를 보여주고 있다. 그림 12의 위쪽 파형에서 첫 번째 펄스상의 파형은 노드 1에서 노드 2와 노드 3으로 전송되는 인버터 게이트 신호 정보, 두 번째 파형은 노드 3에서 노드 2로 전송되는 전동기 2의 속도 및 전류 정보, 그리고 세 번째 파형은 노드 2에서 노드 1로 가는 전동기 1, 전동기 2의 속도 및 전류 정보를 차례로 보여주고 있다.

그림 13과 14는 2대의 전동기에 속도지령치가 독립적으로 입력될 경우의 응답을 보인 것으로 그림의 하단이 속도지령치를 나타내고 있다. 즉, 속도지령치를 500rpm에서 1000rpm으로 동시에 증가시켰다가 500rpm으로 동시에 감속할 경우 각각의 전동기에 대한 응답특성을 살펴본 것이다. 실험을 통해 알 수 있듯이 1

개의 프로세서로 2대의 전동기를 제어하는 것이 가능하기 때문에 2대의 전동기를 연동시켜 제어할 경우에도 별도의 하드웨어의 추가 없이 소프트웨어적으로 간단히 처리할 수 있게 되어 대단히 효과적이다.

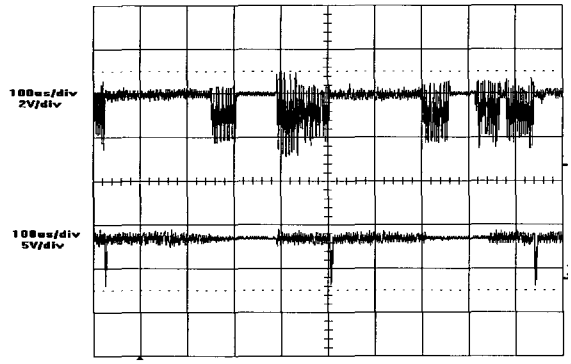

그림 12 CAN 네트워크 데이터 흐름
Fig. 12 Data flow in CAN network

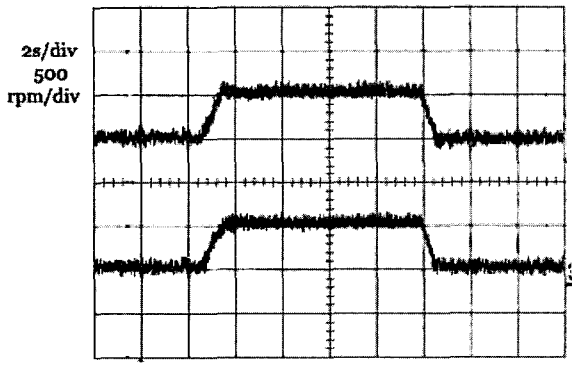

그림 13 유도전동기 1의 속도응답
Fig. 13 Speed response of Induction motor 1

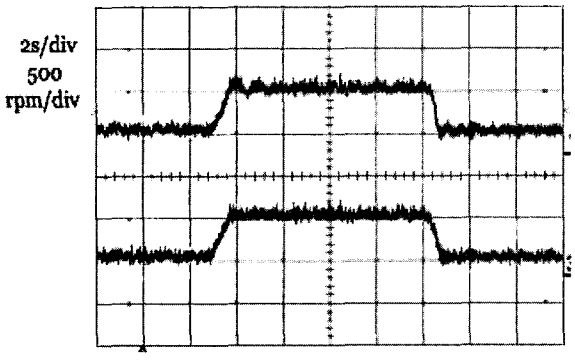

그림 14 유도전동기 2의 속도응답
Fig. 14 Speed response of Induction motor 2

5. 결 론

본 논문에서는 하나의 제어기로 2대의 전동기를 구동할 수 있음을 보였다. 즉, CAN을 이용한 실시간 네트워크를 사용해 단일 제어기로 2대의 유도전동기를 구동할 수 있는 시스템을 구축하고 이를 바탕으로 벡터제어 알고리즘에 의해 2대의 유도전동기를 제어하여 단일 프로세서에 의한 복수전동기의 구동 가능성을 실험적으로 확인했다.

본 논문에서 제안된 시스템은 2대의 전동기를 연동해서 운전할 경우에도 듀얼 포트 램과 같은 별도의 인터페이스 장치나 별도의 프로세서가 없어도 소프트웨어적으로 간단히 구현할 수 있다. 따라서 제안한 방식은 대단히 경제적일 뿐 아니라 원거리에 위치하고 있는 전동기들도 네트워크를 이용한 효과적인 제어 및 감시가 가능하므로 공장자동화와 관련된 종합적인 프로세서 제어를 용이하게 수행할 수 있다.

향후 보다 향상된 실시간 네트워크 기술과 고성능 프로세서의 개발을 통해 통신시간의 단축, 통신의 신뢰성 확보 및 제어알고리즘의 연산 시간을 단축시킬 수 있다면 2대 이상의 전동기도 단일 프로세서에 의한 제어가 가능해질 수 있을 것이다.

본 연구는 울산대학교와 한국과학재단 지정 울산대학교 네트워크 기반 자동화연구센터의 지원에 의한 것입니다.

참 고 문 헌

[1] G. Cena and A. ValenzaNo., "An Improved CAN Fieldbus for Industrial Application" *IEEE* Trans. on Industrial Electronics, Vol. 44, No. 4, Aug. 1997.
[2] Jan Bosteels, "Coordinated multi-axis motion control via CANo.pen", *CAN Newsletter*, Feb. 2002.
[3] 김동식 외, "CAN 네트워크를 사용한 레미콘 온라인 생산 시스템 설계" *전력전자학회 논문지* 제8권 제4호, 2003.
[4] Wolfhard Lawrenz,"CAN System Engineering From Theory to Practical Application", *Springer*, 1997.
[5] CiA, CAN Specification 2.0 PartA, B, *www.can-cia.de*, 1997.
[6] CiA, CAN Physical Layer, *www.can-cia.de*, 1997.
[7] CiA, CAN Datalink Layer, *www.can-cia.de*, 1997.
[8] G. Cena and A. ValenzaNo., "Achieving Round-Robin Access in Controller Area Networks", *IEEE* Trans. on

Industrial Electronics, Vol 49, No. 6, Dec. 2002.
[9 G. Leen and D. Heffernan, "Time-triggered controller area network", *Computing and Control Eng.* Jr., Dec. 2001.
[10 Phillips, "SJA1000 Stand-alone CAN Controller", Ph illips Semiconductor, 1997.
[11 Peter Vas, "Vector Control of AC Machine", Oxford University Press, 1998.
[12] 나석환, 정영국, 임영철, "인버터 구동 시스템을 위한 새로운 공간벡터 Random PWM 기법", *전력전자학회 논문지* 제6권 제6호, 2001.

저 자 소 개

정의헌(鄭義憲)
1973년 1월16일생. 1999년 울산대 제어계측공학과 졸업. 2001년 동 대학원 제어계측공학과 졸업(석사). 2003 동 대학원 제어계측공학과 박사과정.

이현영(李賢寧)
1976년 11월 19일생. 2002년 울산대 제어계측공학과 졸업. 2003년 동 대학원 전기전자 정보시스템공학부 석사과정.

이홍희(李弘熙)
1957년 10월 15일생. 1980년 서울대 전기공학과 졸업. 1982년 동 대학원 전기공학과 졸업(석사). 1990년 동 대학원 전기공학과 졸업(공박). 1994~1995년 Texas A&M 방문교수. 현재 울산대 전기전자 정보시스템공학부 교수.

전태원(全泰園)
1959년 1월 30일생. 1981년 부산대 전기공학과 졸업. 1983년 서울대 대학원 전기공학과 졸업(석사). 1987년 동 대학원 전기공학과 졸업(공박). 1997~1997 Tennes see 대학 방문교수. 현재 울산대 전기전자 정보시스템공학부 교수. 당 학회 편집이사.