CAN 시간동기를 이용한 복수 전동기 동기제어

Synchronization Control of Multiple Motors using CAN Clock Synchronization

Le Minh Khoa Do, 서 영 수*
(Le Minh Khoa Do and Young Soo Suh)

Abstract : This paper is concerned with multiple motor control using a distributed network control method. Speed and position of multiple motors are synchronized using clock synchronized distributed controllers. CAN (controller area network) is used and a new clock synchronization algorithm is proposed and implemented. To verify the proposed control algorithm, two disks which are attached on two motor shafts are controlled to rotate at the same speed and phase angle with the same time base using network clocks.

Keywords : clock synchronization, distributed system, CAN network

I. 서론

제어시스템의 구성요소들이 네트워크를 통해서 연결된 네트워크 기반 제어시스템이 최근에 많이 사용되고 있다 [1,2]. 네트워크 기반 제어시스템은 유연한 구성이 가능하고 원격 감시 등이 용이한 장점을 가지고 있다.

복수의 전동기를 네트워크 기반으로 제어할 때 모든 전동기를 하나의 제어기에서 제어하는 중앙집중형 제어기(centralized controller)와 각각의 전동기에 제어기를 사용하고 이를 네트워크를 통해서 통합관리하는 분산형 제어기(distributed controller)의 두 가지 방식이 있다. 최근의 마이크로프로세서의 속도 향상 및 저가격화로 분산형 제어기가 많이 사용되고 있다.

분산형 제어시스템에서 복수의 전동기를 동기제어하기 위해서는 각 제어기의 시간이 정확하게 동기되어 있어야 한다. 본 논문에서는 CAN(Controller Area Network)을 사용하였고 시간동기 알고리즘을 사용하여 각 제어기의 시간을 동기하였다.

기존의 CAN에서의 시간동기 알고리즘으로는 패킷 전송이 완료된 시점의 타임스탬프(time stamp)를 시간동기의 기준으로 하는 알고리즘이 가장 보편적이다[3,4]. 패킷 전송 시점을 기준으로 하는 것은 패킷 시작시점은 네트워크의 노드에서 알기 어렵지만 패킷 전송 완료시점은 각 노드에서 비교적 정확하게 측정할 수 있기 때문이다. 시간동기의 정밀도는 [3]의 알고리즘은 $\pm 20\mu s$, [4]의 알고리즘은 $\pm 50\mu s$ 이다.

더 많은 대역폭을 시간동기에 사용하여 시간동기의 정밀도를 높이는 알고리즘도 제시되고 있다. [5]에서는 $\pm 10\mu s$의 정밀도를 달성하였고, 시간동기에 전용 하드웨어를 사용하는 [6]에서도 $\pm 10\mu s$의 정밀도를 달성하고 있다.

본 논문에서는 패킷 전송의 완료 타임스탬프를 사용하는 [3] 방식에 바탕을 둔 간단한 방법으로 $\pm 5\mu s$의 정밀도를 가지는 알고리즘을 제안하고 있다. 또한 제안하는 알고리즘은 매스터 노드가 고장일 때에도 동작할 수 있는 고장 허용 기능을 포함하고 있다.

시간동기를 바탕으로 두 대의 전동기의 동기제어기를 설계하고 검증하였다. 두 대의 전동기에 각각 회전판을 설치하여 두 회전판의 속도와 위치가 정확하게 동기되는 제어기를 설계하였다. 이 제어시스템은 [7]의 논문에서 제안된 것으로 [7]에서는 시간동기로 IEEE 1588을 사용하였다. 본 논문에서는 전용의 시간동기 하드웨어 없이 CAN 기반에서 간단한 시간동기 알고리즘을 사용하여 두 대의 전동기를 제어하는 알고리즘을 제시하고 있다.

II. CAN 시간동기 알고리즘

CAN기반 시간동기 알고리즘 구현을 위한 각 노드들의 구조가 그림 1에 주어져 있다.

각 노드들은 CPU와 Timer의 클럭펄스 제공을 위한 수정발진자를 가지고 있다. 저가격의 수정발진자는 시간이 지남에 따라 오차가 누적되어 각 노드들의 시간이 어긋나게 된

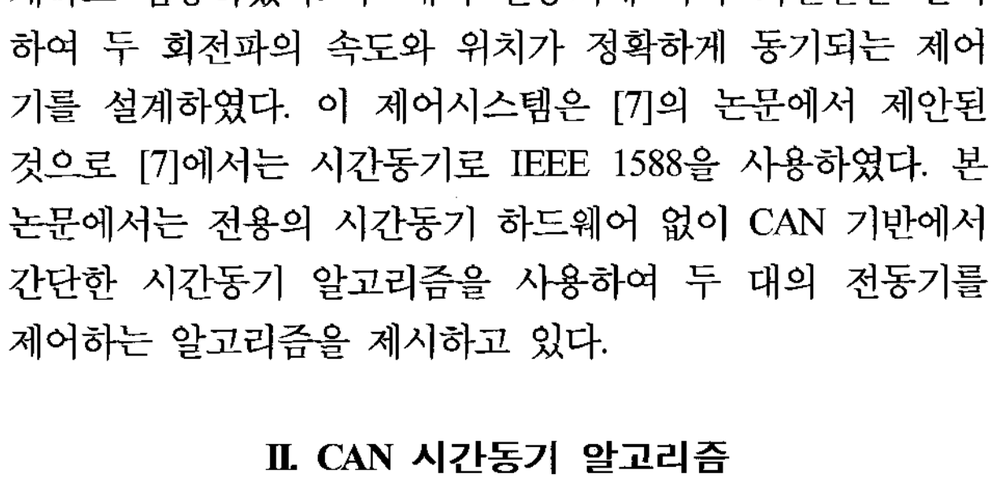


그림 1. CAN 노드의 시간동기 모듈.

Fig. 1. Clock synchronization module of CAN node.

* 책임저자(Corresponding Author)
논문접수 : 2007. 2. 5., 채택확정 : 2008. 4. 28.
Le Minh Khoa Do : 울산대학교 자동차선박기술대학원
(doleminhkhoa@yahoo.com)
서영수 : 울산대학교 전기전자정보시스템공학부(yssuh@ulsan.ac.kr)
※ 본 연구는 2005년도 울산대학교 교내연구비에 의해서 지원되었음.

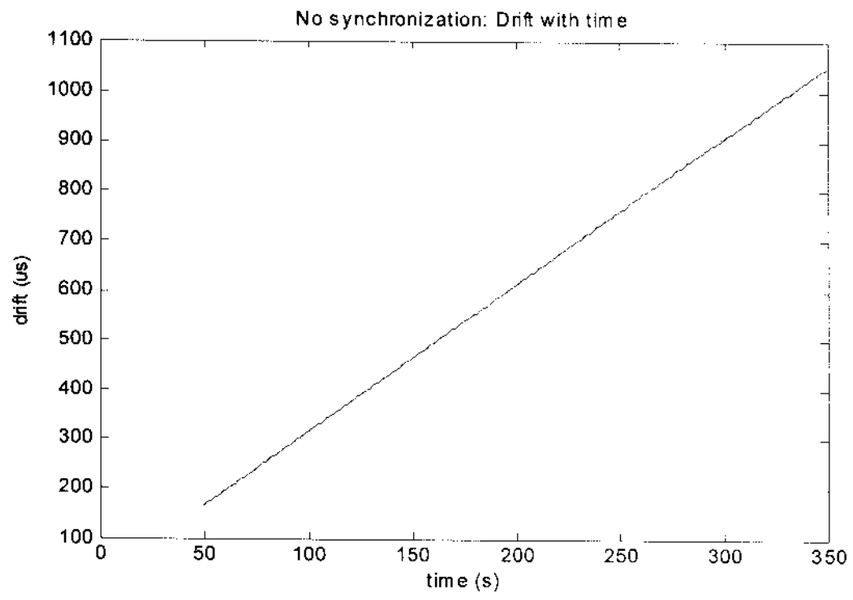


그림 2. 동기화 되지 않은 경우의 시간차.

Fig. 2. Clock drift without synchronization.

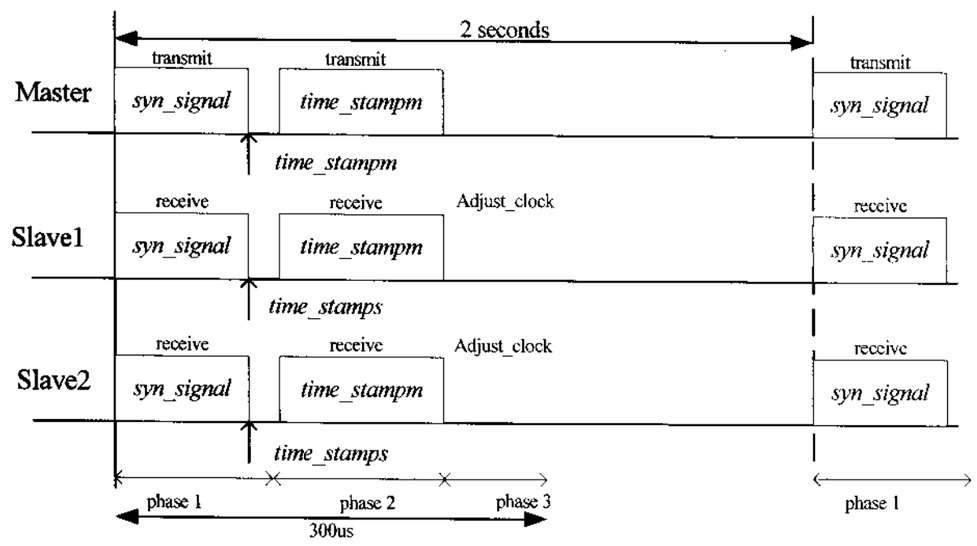


그림 3. CAN에서의 시간 동기의 단계.

Fig. 3. Phases of clock synchronization in CAN.

다. 그림 2는 동기화 되지 않은 두 노드의 시간의 차이를 나타낸 것이다. 본 실험에서 사용한 시스템의 경우 1초당 약 294μs의 시간차가 발생하는 것을 알 수 있다.

이러한 시간차를 없애기 위해 시간동기 알고리즘을 사용한다. IEEE 1588의 알고리즘과 비슷하게 매스터 노드에서 시간 동기화 패킷 (phase 1의 syn_signal 패킷)을 보내고 연이어서 시간 스탬프 패킷 (phase 2의 time_stampm 패킷)을 보낸다. 이 정보를 바탕으로 phase 3에서 시간을 보정한다 (그림 3).

각 단계별로 좀 더 자세히 살펴보면 다음과 같다. 1단계 (phase 1)에서는 지정된 매스터 노드에서 CAN 상의 각 노드들에 패킷 (syn_signal 패킷)을 방송(broadcasting)한다. 패킷 전송이 완료된 순간에 CAN 상의 모든 노드들 (매스터 및 슬레이브)은 패킷 전송완료시간을 저장한다. CAN의 특성상 보내는 노드 (즉, 매스터 노드)에서의 전송완료시간과 받는 노드 (즉, 슬레이브 노드들)에서의 수신완료시간이 일치함으로 이 때의 시간을 비교함으로서 시간동기화를 이룰 수 있다. 본 논문에서는 각 노드들에 TMS320F280x DSP를 사용하였는데 이 DSP의 CAN 모듈은 CAN 패킷에 대한 하드웨어 타임스탬프 기능을 가지고 있어서 패킷전송 완료의 정확한 시간을 기록할 수 있다.

2단계에서는 매스터 보드가 자신의 전송완료시간을 저장한 패키 (time_stampm 패킷)을 방송한다. 3단계에서는 각 슬레이브 노드에서 전송받은 매스터의 syn_signal 패킷의 전송완료 시간과 기록하였던 전송완료시간을 비교하여 시간보정을 한다. 매스터의 전송완료 시간을 time_stampm이라고 하고 슬레이브의 수신완료시간을 time_stamps라고 할 때 시간보정 알고리즘은 다음과 같다.

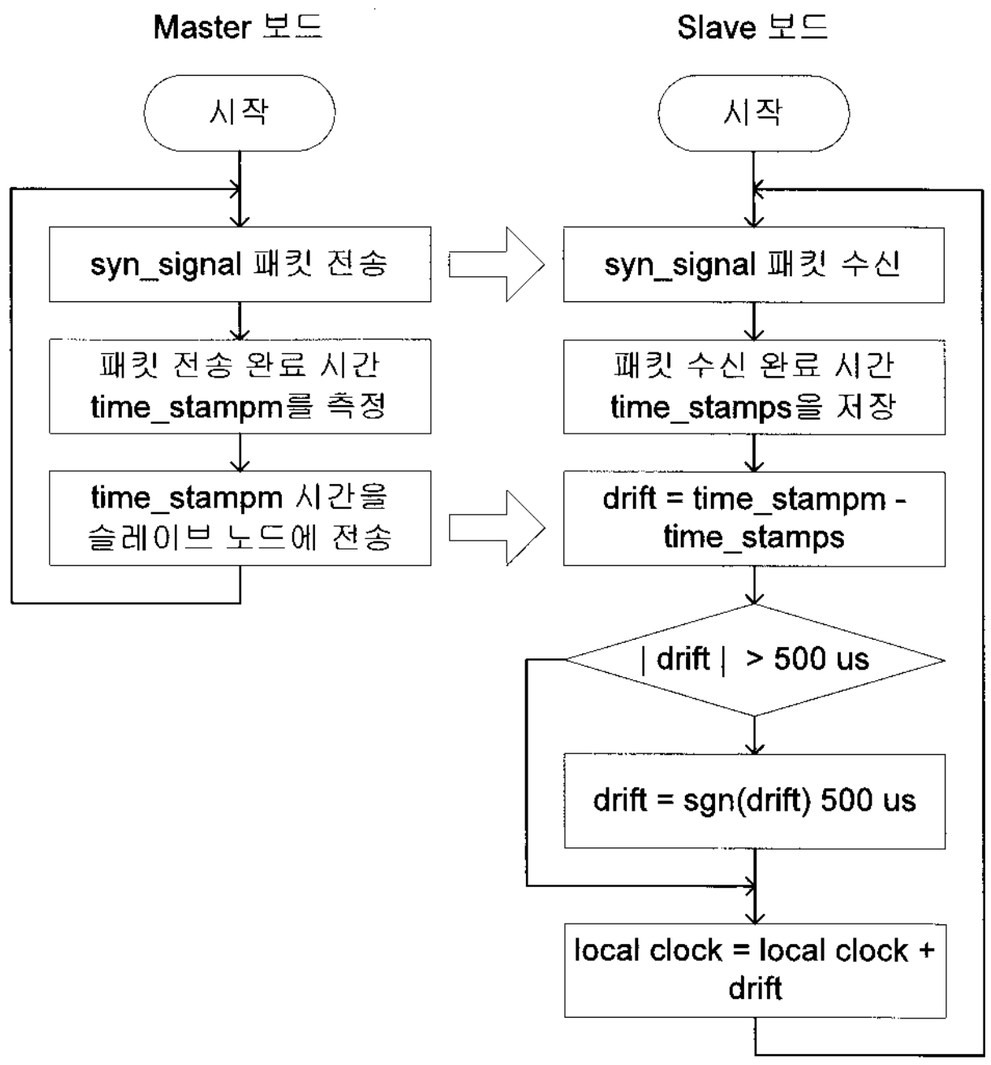


그림 4. 시간 동기 알고리즘.

Fig. 4. Clock synchronization algorithm.

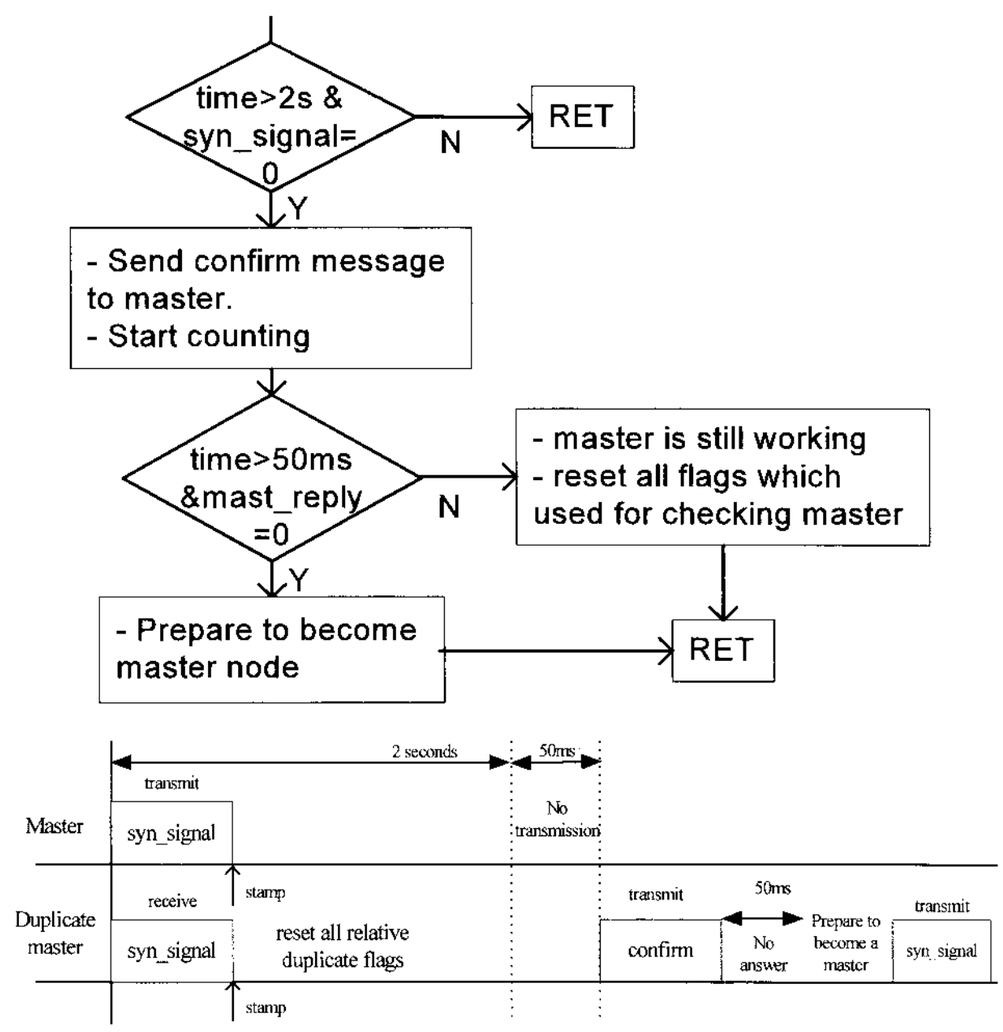
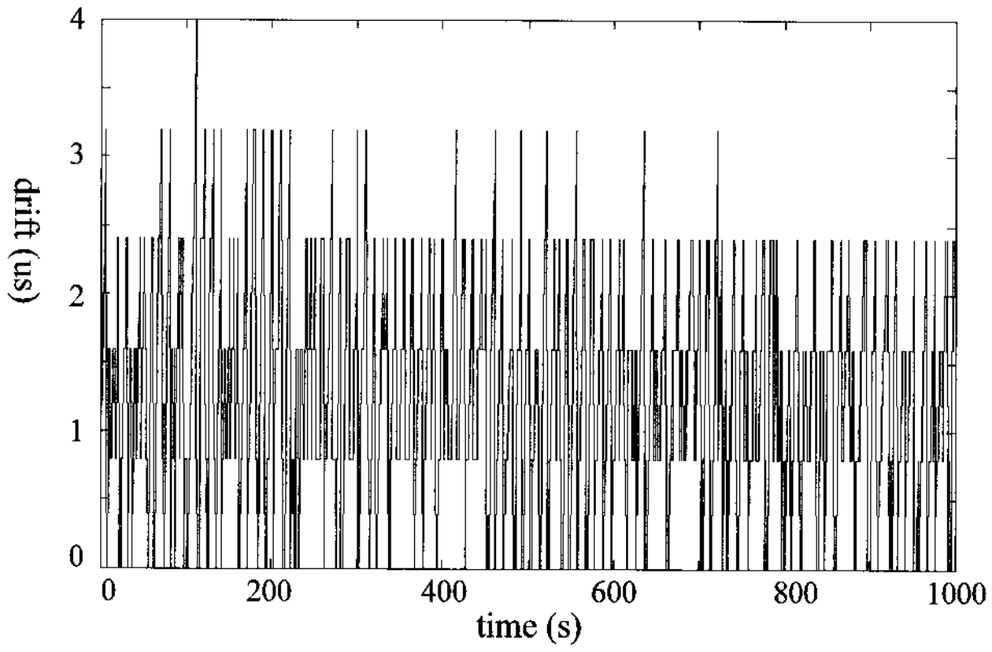


그림 5. 매스터 노드 고장시 슬레이브 노드가 매스터가 됨.

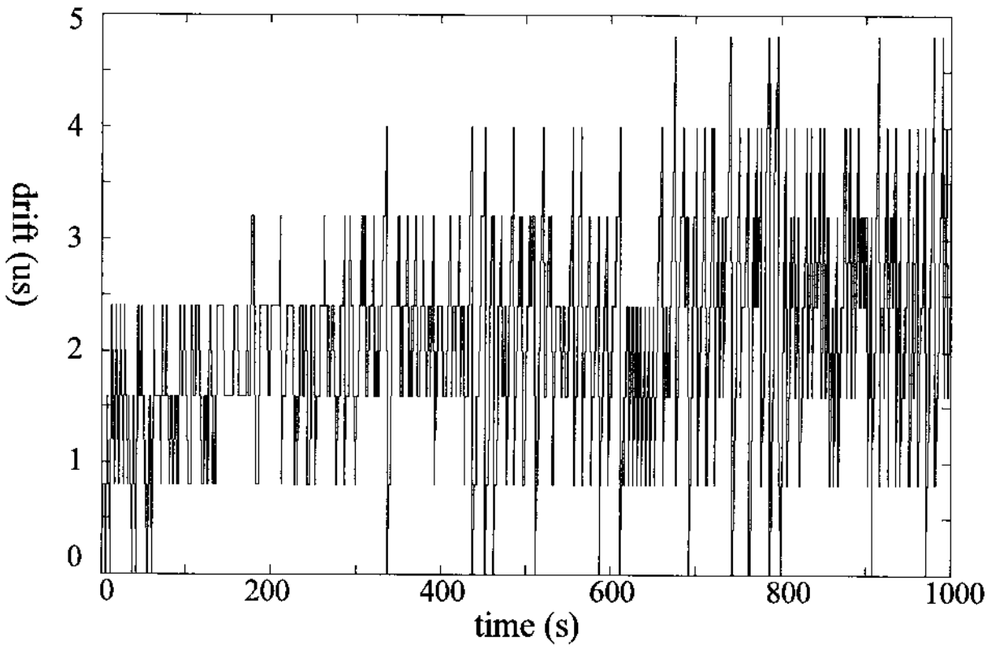
Fig. 5. If the master node fails, a slave node becomes the master.

경우에 따라서는 매스터 노드에 문제가 발생할 수도 있다. 매스터 노드에서 예정된 시각에서 50㎳가 지나도 syn_signal 패킷을 보내지 않으면 주어진 순서에 따라 슬레이브 노드가 매스터 노드가 되도록 하였다(그림 5).

그림 6은 제안한 알고리즘을 사용한 시간동기의 결과이다. 한 개의 매스터 노드와 두 개의 슬레이브 노드로 구성된 시스템에서의 실험이었는데 매스터 노드와 슬레이브 노드의 시간차가 5μs를 넘지 않는 것을 확인할 수 있다. 그림 7은 매스터 노드와 슬레이브 노드 2와의 시간차를 확대한 그림이다.



(a) drift between master and slave 1



(b) drift between master and slave 2

그림 6. 매스터 노드와 슬레이브 노드와의 시간차 (1000초).

Fig. 6. Clock drift between a master node and slave nodes in 1000 seconds.

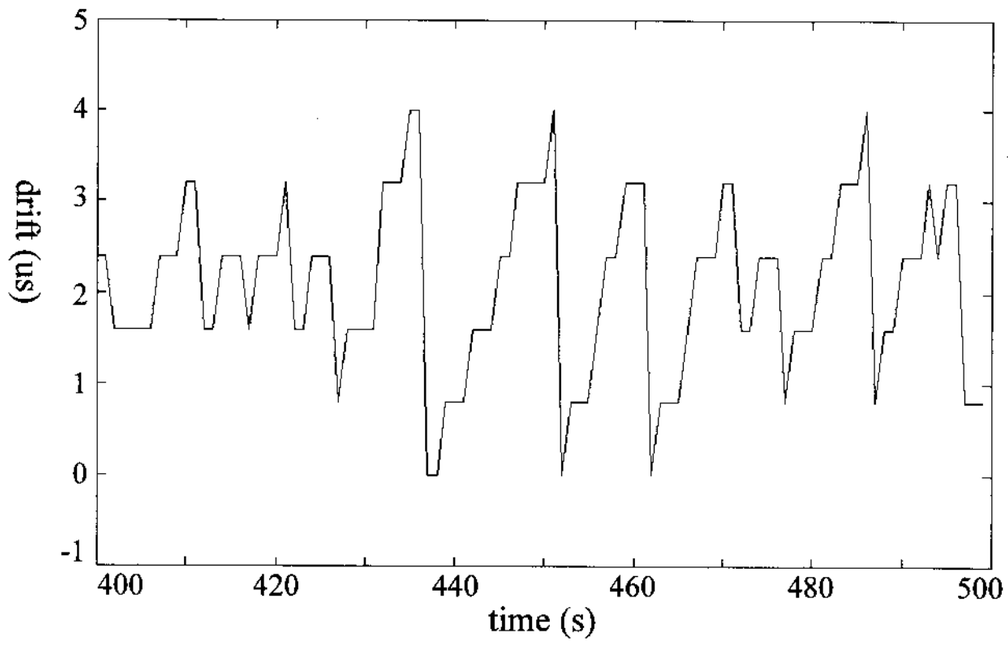


그림 7. 확대한 매스터노드와 슬레이브 노드2와의 시간차.

Fig. 7. Enlarged clock drift between a master node and slave node 2.

III. 두 개의 전동기의 동기제어

그림 8은 두 개의 전동기의 네트워크 기반 제어시스템을 나타내고 있다. 각각의 전동기는 DSP 보드에 의해서 제어되고 이 DSP 보드는 CAN을 통해서 연결되어 있다.

각각의 전동기에는 그림 9와 같은 디스크가 달려 있다. 이 시스템의 제어목적은 두 전동기의 속도와 위상각(두 디스크의 상대적인 위치. 정확한 정의는 나중에 나옴)을 일치시키는 것이다.

두 개의 전동기가 동기제어 되고 있을 때에는 두 디스크의 슬롯의 상대위치가 일치해 그림 8의 광센서에서는 디스크 한 회전당 정확하게 12개의 펄스가 발생한다.

전동기의 기준속도가 V_{ref} (rpm)일 때 디스크가 1회전하는데 걸리는 시간 T_{rot}는 다음과 같다.

$$T_{rot} = \frac{60 \times 1000}{V_{ref}} (ms). \tag{1}$$

전동기에 부착되어 있는 인코더는 2500 pulse/revolution의 분해능을 가지고 있고 한 회전마다 한 번씩 펄스가 나오는 인덱스 펄스도 제공하고 있다. 전동기의 속도가 정확하게 V_{ref}이고 인덱스 펄스가 $T_{ref} = I_k$시각에 발생했으면, 한 회전 후에 인덱스 펄스가 발생하는 시점은 $T_{ref} + T_{rot}$가 된다. 일반적으로는 전동기의 속도가 정확하게 V_{ref}가 아니기 때문에 한 회전후의 인덱스 펄스 발생시점은 정확하게

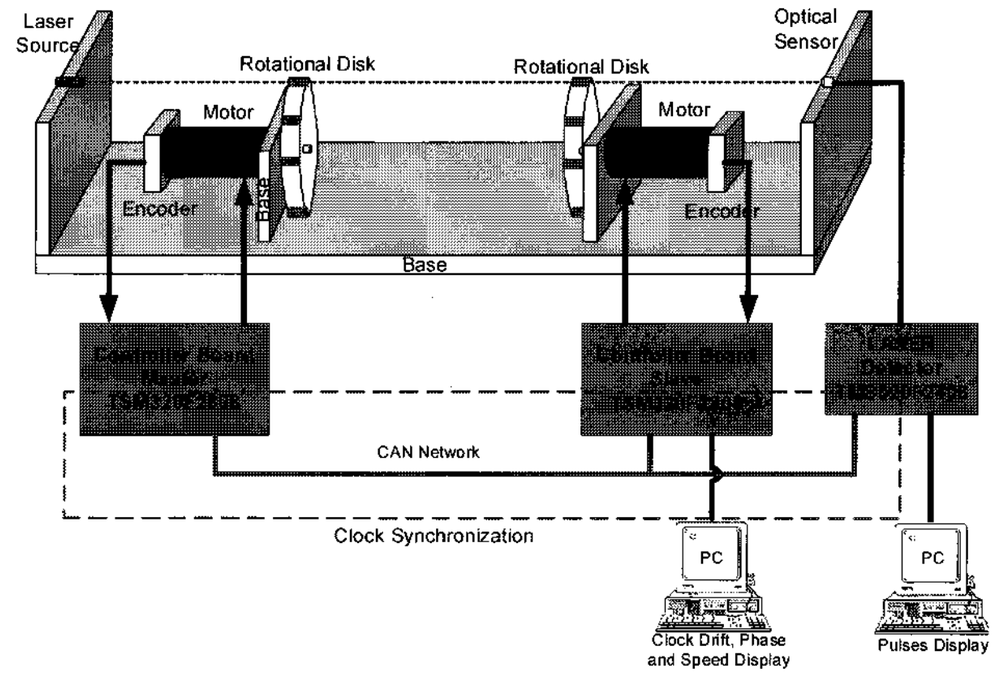


그림 8. 두 개의 전동기의 네트워크 기반 제어시스템.

Fig. 8. Two motor networked control system.

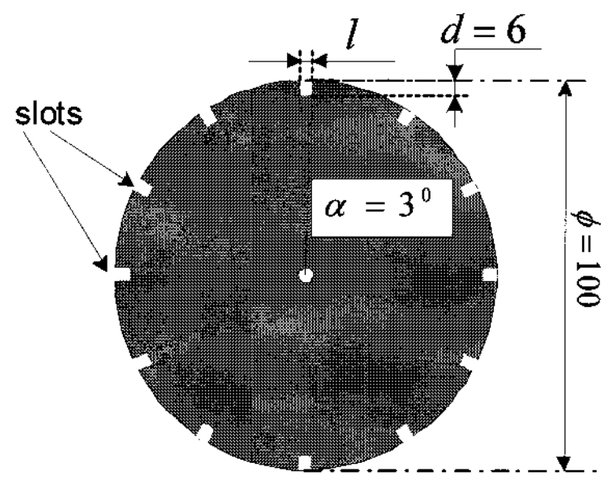


그림 9. 전동기축에 부착된 디스크.

Fig. 9. Disk attached at the motor shaft.

$T_{ref}+T_{rot}$가 아니고 다른 값 I_{k+1}을 가지게 된다. 인덱스 펄스의 발생시점 I_{k+1}와 기준시점 $(T_{ref}+T_{rot})$의 차이를 각도로 환산한 것을 위상각(P_{k+1})으로 정의한다.

$$P_{k+1}=360^{\circ}\frac{\delta_{k+1}}{T_{rot}} \qquad (2)$$
$$\delta_{k+1}=I_{k+1}-(k+1)T_{rot}$$

위상각을 계산할 때 kT_{rot}가 기준시각으로 사용되는데 이는 매스터 노드에서 계산하고 슬레이브 노드에는 CAN을 통해서 전달한다. 매스터 노드에서는 자체적으로 계산한 기준시각과의 차에서 계산된 위상각을 0으로 만들고, 슬레이브 노드에서는 매스터 노드의 기준시각과 차에서 계산된 위상각을 0으로 만들어 위상동기를 구현한다.

전동기의 속도와 위상각 제어를 위해서 그림 11과 같이 각각 PI 제어기를 사용하였다.

$$u(t)=K_p(e(t)+K_i\int_{-\infty}^{t}e(r)dr) \qquad (3)$$

안쪽의 속도제어 루프는 일정한 주기로 제어되고 바깥쪽의 위상제어 루프는 인덱스 펄스가 발생할 때마다 제어가 된다. 전동기의 속도는 일정시간동안의 인코더 펄스의 개수에서 계산하는 M방식 [9]을 사용하였다.

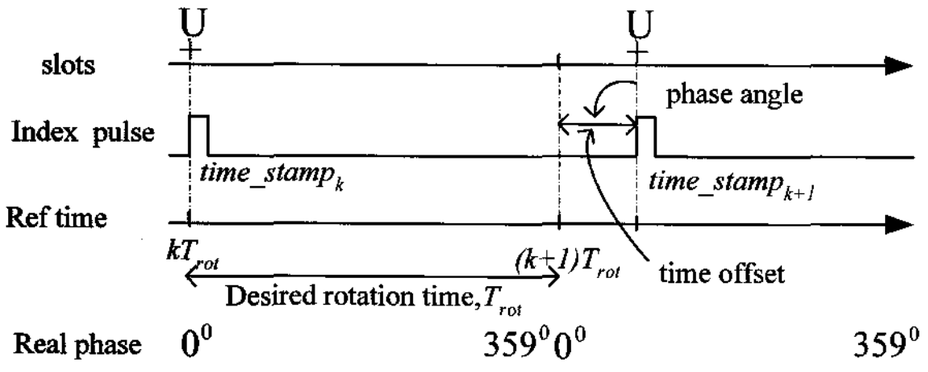


그림 10. 위상각.

Fig. 10. Phase angle.

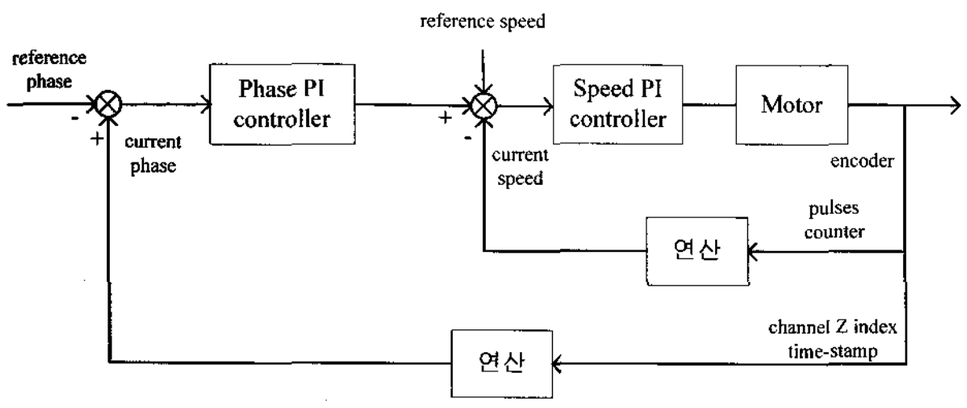


그림 11. 위상각 및 속도 제어 루프.

Fig. 11. Phase angle and speed control loops.

표 1. PI 제어기의 게인.

Table 1. PI controller gain.

	30 rpm		200 rpm	
	K_p	K_i	K_p	K_i
속도	13.7	60	13.73	60
위상각	0.081	0.3	0.517	2

PI 제어기의 게인은 Ziegler-Nichols [10]의 폐루프 튜닝 방법을 사용하였다. 전동기의 기준속도가 30 rpm 및 200 rpm인 경우에 대해서 먼저 속도 PI 제어기의 게인을 구하고 다음에 위상각 PI 제어기의 게인을 구하였다(표 1).

표 1의 PI 게인을 사용하여 기준속도 30 rpm의 경우와 200 rpm의 경우에 대해서 동기제어를 해 보았다 (그림 12 및 13 참조). 정상상태에서 기준속도가 200 rpm인 경우에는 속도오차가 ±0.8rpm, 위상각오차가 ±0.3이고, 기준속도가 30rpm인 경우에는 속도오차가 ±1.2rpm, 위상각오차가 ±0.3로 동기제어가 잘 이루어지고 있음을 알 수 있다. 그림 12 및 13에서 속도 및 위상각의 값이 튀는 구간이 있는데 이는 인위적으로 매스터 노드측의 디스크를 손으로 만져서 외란을 준 경우이다. 외란에 대한 반응을 보기 위해 기준속도 200 rpm인 경우를 그림 13에서 확대하였다. 외란에 의해서 동기가 깨어졌을 때 약 2초 후에 다시 동기가 되는 것을 알 수 있다. 참고로 기준속도가 200 rpm인 경우에는 (1)에서 $T_{rot}=0.3\sec$가 되어 위상각 제어주기는 0.3초가 된다.

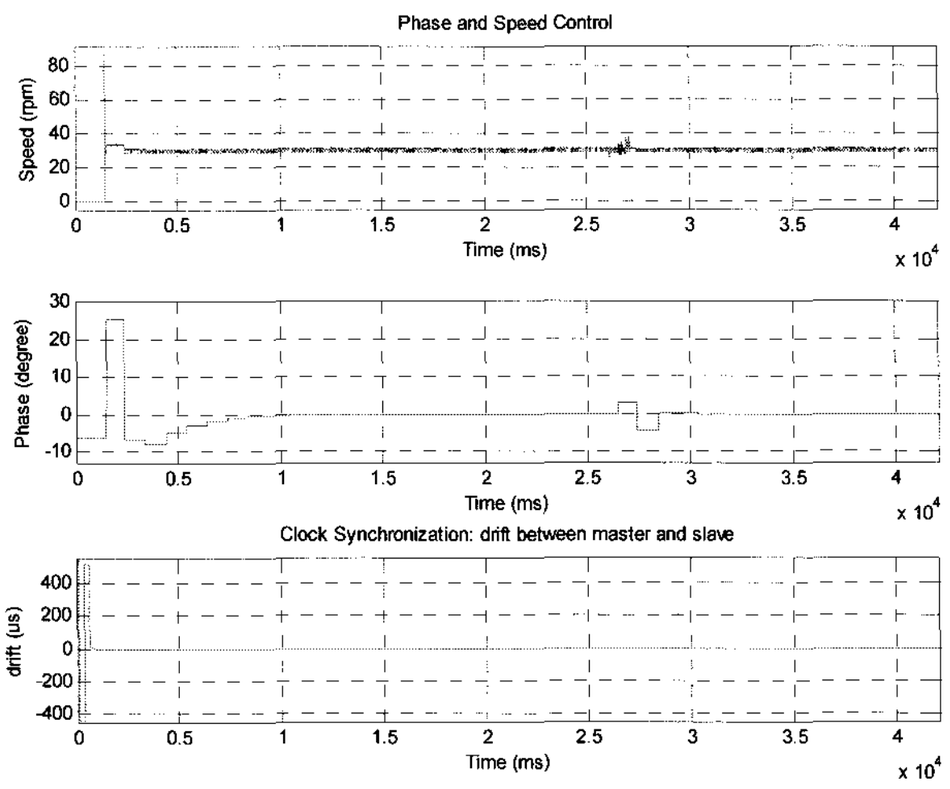


그림 12. 동기제어 결과 (기준속도 30 rpm).

Fig. 12. Synchronization result (reference speed : 30 rpm).

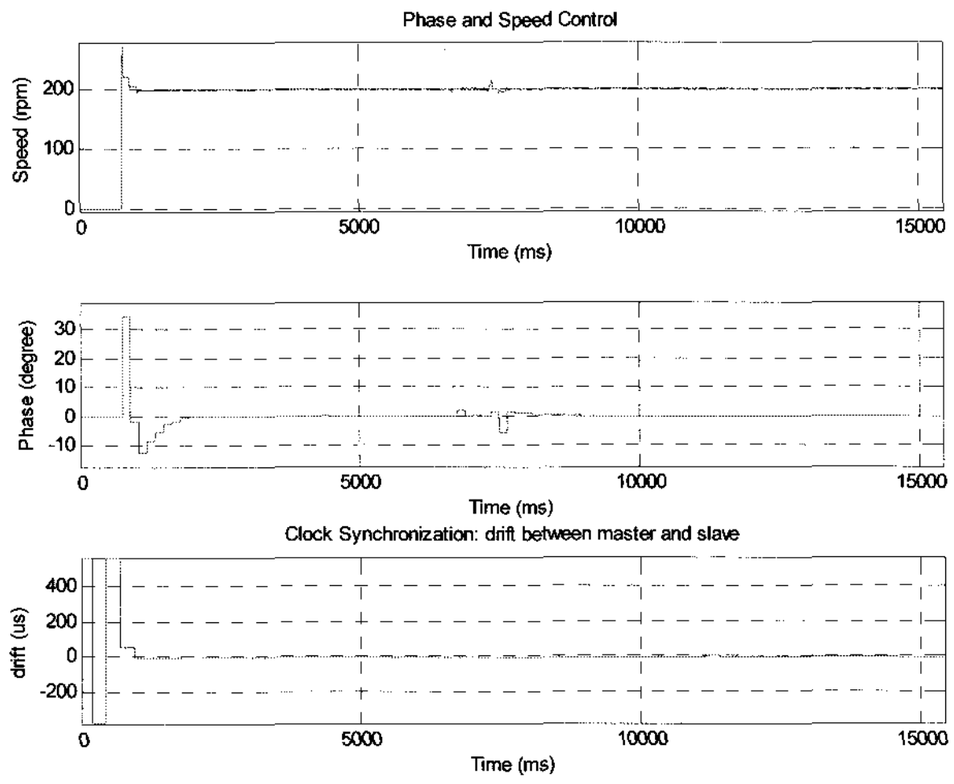


그림 13. 동기제어 결과 (기준속도 200 rpm).

Fig. 13. Synchronization result (reference speed : 200 rpm).

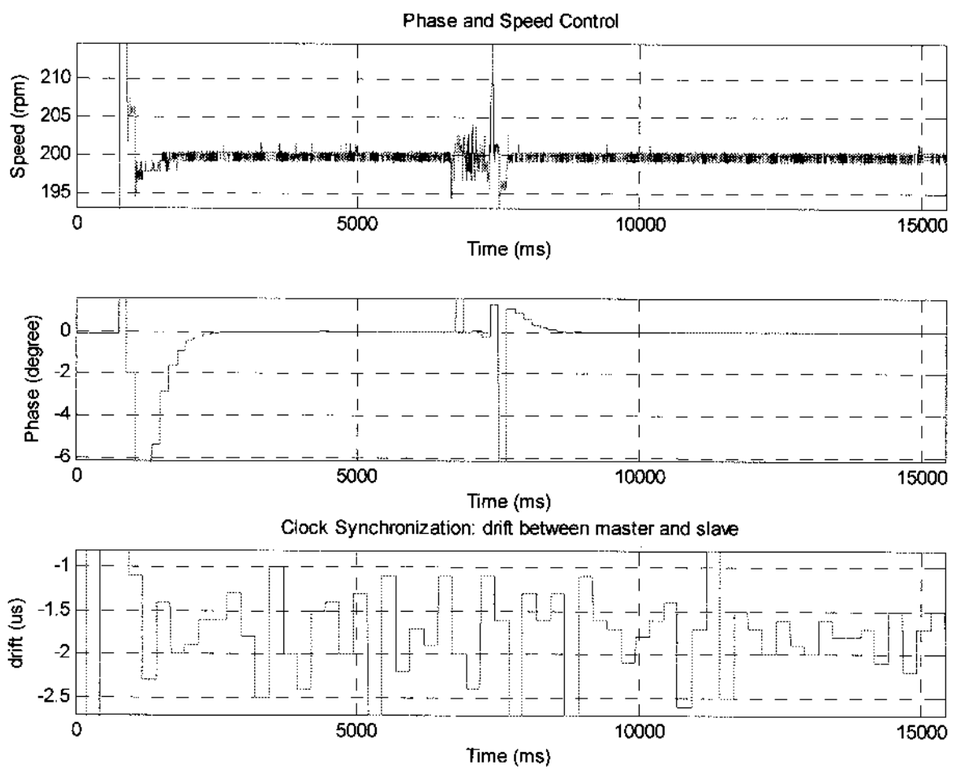


그림 14. 기준속도 200 rpm 경우를 확대.

Fig. 14. Enlarged result of 200 rpm case.

IV. 결론

본 논문에서는 CAN 기반 분산제어시스템을 위한 시간동기 알고리즘을 제시하였다. 이 알고리즘은 구현이 간단하고 5μs의 정밀도를 가지고 있다. 시간이 동기된 두 개의 제어기 노드에서 전동기의 속도 및 위상각을 제어하였다. IEEE 1588을 사용한 [7]과는 달리 전용 하드웨어 없이 CAN 상에서 비교적 작은 시간 동기 패킷 오버헤드를 통해 2개의 모터를 제어한 것이 본 논문의 의의라고 할 수 있다.

실험에서는 두 대의 전동기의 경우였으나 3대 이상의 경우에도 네트워크의 트래픽 증가 없이 확장가능하다. 시간동기정보를 제어 패킷에 함께 보냄으로서 네트워크의 트래픽을 더욱 줄이는 알고리즘을 개발하는 것이 앞으로의 과제이다.

참고문헌

[1] G. C. Walsh and H. Ye, "Scheduling of networked control systems," *IEEE Control System Magazine*, pp. 57-65, 2001.

[2] 김동성, 최동혁, "네트워크 기반 이산 시간 제어 시스템의 최대 허용 지연 한계 및 실시간 스케줄링 기법에 관한 연구," 제어 자동화 시스템공학 논문지 제 12 권 제 7 호, pp. 719-727, 2006.

[3] M. Gergeleit and H. Streich, "Implementing a distributed high-resolution, real-time clock using the CAN-bus," *Proceedings of the 1st International CAN Conference*, Mainz, Germany, 1994.

[4] L. Rodrigues, M. Guimaraes, and J. Rufino, "Fault-tolerant clock synchronization in CAN," *Proceedings of the 19th IEEE Real-Time Systems Symposium*, pp. 420-429, Madrid, Spain, 1998.

[5] D. Lee and G. Allan, "Fault-tolerant clock synchronization with microsecond-precision for CAN network," *Proceedings of the 9th International CAN Conference*, Munich, Germany, 2003.

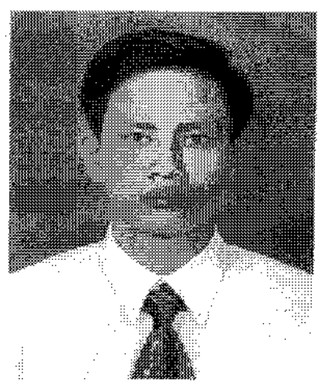
[6] T. Fuhrer, B. Muller, W. Dieterle, F. Hartwich, R. Hugel, M. Walther, and R. B. GmbH, "Time triggered communication on CAN," *Proceedings of the 7th International CAN Conference*, pp. 53-58, Amsterdam, The Netherlands, 2000.

[7] K. H. Leong, "Rotational synchronization via IEEE 1588," *2005 Conference on IEEE 1588*, Switzerland.

[8] Standard for a Precision Clock Synchronization Protocol for Networked Measurement and Control Systems, *IEEE Instrumentation and Measurement* Society, 2002.

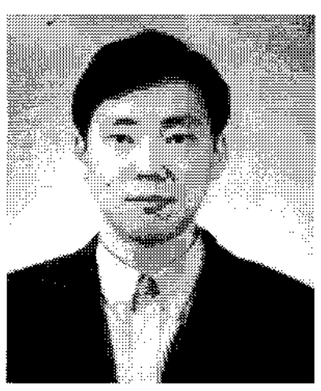
[9] T. Ohmae et al., "A microprocessor-controlled high-accuracy wide-range speed regulator for motor drives," *IEEE Tr. Ind. Electron.* vol. IE-29, no. 3, pp. 207-211, 1982.

[10] K. J. Astrom and T. Hagglund, *PID controllers : Theory Design and Tuning*, 2nd Ed, Instrument Society of America, 1995.



Le Minh Khoa Do

2004년 베트남 호치민 대학교 전기전자공학부 졸업. 2007년 울산대학교 자동차선박기술대학원 석사. 관심분야는 네트워크 기반 제어시스템.



서 영 수

1990년 서울대 제어계측과 졸업. 1992년 동 대학원 석사. 1997년 동경대학교 공학박사. 2000년~현재 울산대학교 전기전자정보시스템공학부 부교수. 관심분야는 네트워크 기반 제어시스템.