CAN 통신간의 메시지 전송 프로토콜에 관한 연구

김영근*, 류광렬*, 허창우*
*목원대학교 전자공학과

A Study On Transmission Protocol for Controller Area Network

Young-Keun Kim*, Kwang-Ryol Ryu*, Chang-Wu Hur*
*Mokwon University

요 약

차량의 전자제어장치 및 마이크로컨트롤러의 향상으로 전자제어장비가 증가하였고, 그 장비들 간의 통신 방식이 필요하게 되었다. 통신 및 제어를 위해 중앙 제어 방식에서 분산 제어 방식으로 기술 발전을 하고 있었다. 이런 제어 방식으로 CAN(Controller Area Network)통신이 개발되었다. CAN의 통신 모델 중 마스터-슬레이브 모델은 하나의 마스터 노드가 전체 시스템을 동기화하고 스케줄링하는 형태로 본 연구에 사용하였다.

본 논문에서는 CAN통신 기반으로 기울기 측정 시스템을 구현하여 하나의 마스터와 여러개의 슬레이브로 구성하였다. 슬레이브 모듈에서 측정한 데이터가 버스라인을 통해 마스터모듈로 전송 될 때 충돌로 손실이 발생할 수 있다. 본 연구에서는 데이터 손실을 최소화와 여러개의 슬레이브 모듈에서 전송되어진 데이터가 에러가 나지 않도록 하기 위한 전송 프로토콜을 제안한다.

Ⅰ. 서 론

오늘날 자동차 산업은 새로운 기능 구현을 위하여 전자제어 장비 및 마이크로컨트롤러와 같은 전자제어 장치들의 증가를 가져왔다. 이 장치들 간의 고속으로 정확한 데이터를 처리 할 수 있는 통신 방식이 필요하게 되었다.

중앙 집중형 제어 방식은 시스템의 구조가 복잡해지고 다양해지면서 많은 양의 데이터 처리 요구량을 부합하지 못하고 시스템의 확장이 어려운 단점이 있다. 이러한 문제를 해결하기 위해 버스 형태인 네트워크 통신 방식이 필요로 하게 되었다. 이러한 필요로 인해 CAN(Controller Area Network)통신이 독일의 BOSCH 사에서 처음으로 개발되었다. 현재 차량내의 각종 제어 장치들 간의 네트워크를 제공하는데 사용될 뿐만 아니라 산업 전반에 걸쳐 사용되어진다.

본 연구에서는 하나의 마스터 노드가 전체의 시스템을 동기화하고 스케줄링하는 형태의 모델을 사용하였다. CAN 통신 기반으로 기울기 센서를 사용한 시스템을 구현하였으며 하나의 네트워크 모듈과 여러대의 센서 모듈로 구성하였다. 실시간으로 여러 센서 모듈에서 측정한 데이터들이 버스라인을 통하여 네트워크 모듈로 전송될 때 각 데이터들 간의 충돌과 간섭으로 인한 데이터 손실, 외곡될 수 있다.

본 논문에서는 충돌로 인한 데이터 손실 및 외곡을 최소화한 센서 모듈의 전송 프로토콜과 알고리즘을 제안한다. 네트워크 모듈과 센서 모듈의 CAN controller는 PHILLIPS 사의 SJA1000과 CAN transceiver는 PHILLIPS 사의 PCA82C250을 사용하여 구현하였다.

Ⅱ. CAN(Controller Area Network) 통신

1. CAN의 개념

CAN(Controller Area Network)은 원래 자동차 내의 각종 계측제어 장비들간에 디지털 직렬 통신을 제공하기 위하여 Bosch와 Intel에서 개발된 차량용 네트워크 시스템이다. 근래에는 자동차 분야뿐만 아니라 산업 전 분야에 폭 넓게 적용되고 있다. 마스터/슬레이브(master/slave), 다중 마스터(multiple master), 피어 투 피어(peer to peeer)등을 지원하는 매우 유연성 있는 네트워크이다. 또한 공장의 열악한 환경이나 고온, 충격이나 진동, 노이즈가 많은 환경에서도 잘 견딜 수 있다. 이러한 장점들로 인하여 최근에 와서 공장자동화와 공정의 분산제어 각종 산업 설비에서 제어 및 자동화 관련 장비들간 데이터 교환을 위한 통신망으로 널리 사용되고 있다.

시스템에서 일반적으로 사용되는 CAN 버스는 마이크로 컨트롤러 사이에서 통신망을 형성하며, 2가닥의 꼬임선(Twist Pair Wire) 으로 연결되어 반이중 통신(Half Duplex) 방식으로 짧은 메시지를 사용하는 고속 응용 시스템에 적당하다. 또한 외부의 요인(노이즈 등) 등에 강인한 특성을 가져 통신 에러율을 최소화 하여 높은 신뢰성을 가지고 있다.

2. CAN의 규격

CAN 메시지에 있는 식별자의 길이에 의해 11비트 식별자인 표준 CAN(버전 2.0A)과 29비트 식별자인 확장 CAN(버전 2.0B)으로 나누어진다. 또한 ISO 규격에 의해 1Mbps의 고속 통신과 125Kbps의 통신으로 구별된다. 그림 2-2는 식별자 길이에 따른 메시지 처리를 나타낸 것이다. 대부분의 CAN 2.0A Controller는 오직 표준 CAN 포맷 방식의 메시지만 전송 및 수신이 가능하며 확장 CAN 포맷 방식 CAN 2.0B 의 메시지를 수신 하더라도 그 데이터를 무시해 버린다. 즉, CAN 2.0A Controller에서 보내온 메시지 데이터만 유효하다. 그러나 CAN 2.0B Controller는 양쪽 모두의 메시지 포맷을 송수신 가능하다.

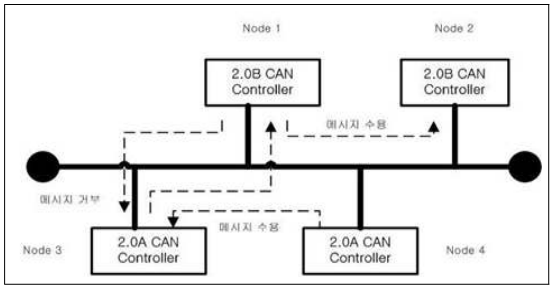

그림 2-2. 식별자 길이에 따른 메시지 처리

3. CAN의 동작 원리

CAN 노드에 메시지를 보내기 전에 CAN 버스 라인을 사용하는지를 파악한다. 어떠한 Node (시스템) 로부터 보내어진 데이터 메시지는 송신측이나 수신측의 주소를 포함하지 않는다. 대신에 각 노드의 데이터 메시지 항목에 CAN 네트워크상에서 각각의 노드를 식별할 수 있도록 각 노드마다 유일한 식별자(ID) 11bit 또는 29bit를 가지고 있다.

네트워크상에 연결된 모든 노드는 네트워크상에 있는 메시지를 수신한 후 자신에게 필요한 메시지인지를 식별자를 통하여 평가한 후 자신이 필요로 하는 식별자의 메시지인 경우만 취하고 그렇지 않은 경우의 메시지는 무시한다.

CAN 통신 라인에 흘러 다니는 여러 노드의 데이터들이 동시에 사용자가 필요로 하는 노드로 유입되는 경우에 식별자의 숫자를 비교하여 먼저 취할 메시지의 우선순위를 정한다. 식별자의 숫자가 낮은 경우가 우선순위가 가장 높다. 우선순위가 높은 메시지가 CAN 버스의 사용 권한을 보장 받으며 이때 낮은 순위의 메시지는 자동적으로 다음 버스 사이클에 재전송을 수행한다. 이때 까지도 높은 우선순위를 가진 메시지가 완료되지 않은 상태이면 전송을 완료할 때까지 대기한다. 각 CAN 메시지는 11비트의 식별자 CAN 2.0A, 또는 29비트의 식별자 CAN 2.0B를 가지면 CAN 메시지의 맨 처음 시작부분에 위치한다. 또한 식별자는 메시지의 형태를 식별시켜 주는 역할과 메시지의 우선순위를 부여하는 역할을 한다.

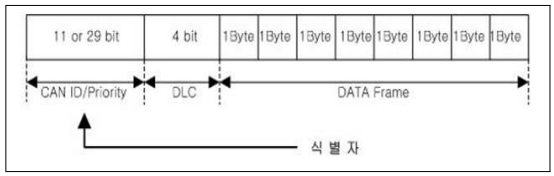

그림 2-3. CAN 통신의 데이터 Frame

III. 시스템 구성

1. 네트워크 모듈

그림 3-1은 네트워크모듈 시스템 블록도를 나타낸 것이다. 네트워크 모듈은 각 센서 모듈에 명령을 지시하거나 각 센서 데이터를 일시적으로 저장하였다가 사용자 요청에 의해 PC로 송신하는 역할을 한다.

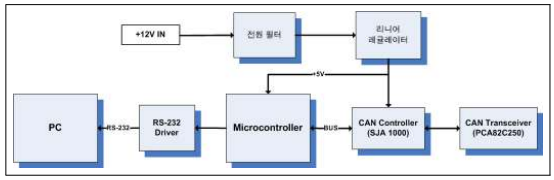

그림 3-1. 네트워크 모듈 시스템 블록도

CAN 버스라인으로 수신된 센서 데이터들은 RS-232 Driver를 통해 PC로 출력된다. CAN Controller는 필립스사의 SJA1000을 사용하여 버스 제어한다. CAN Transceiver는 컨트롤러와 같

은 회사인 필립스사의 PCA82C250을 사용하였다. CAN Driver를 통하여 CAN_H, CAN_L로 변환 되어 두 가닥 꼬임선으로 통신이 가능하게 만든다.

2. 센서 모듈

그림 3-2는 센서 모듈 시스템의 블록도를 나타낸 것이다. CAN Controller와 Transceiver는 네트워크 모듈에 사용한 모델과 같은 것을 사용하였다. 일정 주기마다 SPI 통신으로 단일 축 가속도계의 데이터를 CAN 버스라인을 통하여 네트워크 모듈로 전송한다.

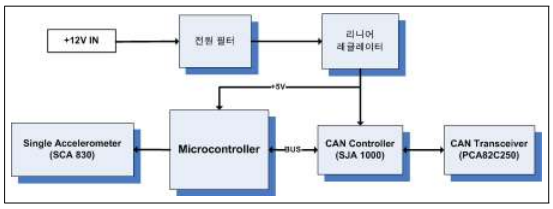

그림 3-2. 센서 모듈 시스템 블록도

Ⅳ. 실험 및 고찰

그림 4-1은 본 연구에 사용한 GUI로 각 센서 모듈의 데이터 출력을 나타낸다. 각 센서 모듈의 ID와 Slot 번호를 부여하여 센서 모듈의 데이터를 모두 수신하는지 확인한다. 또한 센서 모듈의 ID와 Slot 번호가 CPU 내의 EEPROM에 저장되어 있어 전원을 꺼도 이 데이터는 그대로 남아 있다. 전송 시키고자 하는 센서 모듈의 ID와 Slot 번호가 맞아야 데이터를 CAN 버스를 통하여 네트워크 모듈로 송신한다.

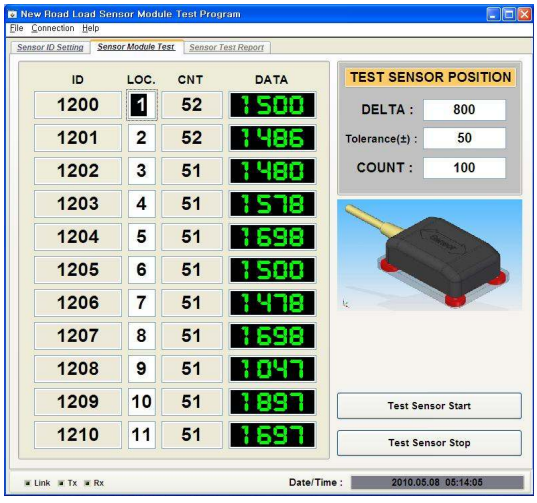

그림 4-1. 센서 모듈 데이터 출력 GUI

GUI의 각 센서 모듈에 데이터를 받으면 카운트 값이 증가하여 센서 별로 데이터의 손실 여부를 판단할 수 있다. 각 데이터 값은 가속도계의 디지털 값을 표시하였다.

Ⅴ. 결론

본 연구에서는 각 센서 모듈에 ID와 Slot 번호를 부여하여 모든 데이터들이 정확하게 출력이 되는지 확인하였다. 각 Slot 번호 마다 데이터 출력의 시간을 각각 다르게 하여 CAN 버스 라인에 데이터들의 충돌을 방지하였다.

기존의 똑같은 시간으로 센서 모듈의 데이터를 보낸 결과 데이터 끼리 충돌로 잃어버린 Slot 번호가 자주 발생하는 것을 알 수 있었다. 본 논문에 사용한 프로토콜과 알고리즘으로 구현한 결과 100%의 전송률을 나타내었다.

참고 문헌

[1] Bosch CAN specification Ver 2.0 Published by Robert Bosch GmbH, September 1991

[2] Philips data sheet, SJA1000, PCA82C250 ; CAN Controller Interface

[3] Prof.Dr-Ing. K. Etschberger "Controller Area Network Basics, Protocols, Chips and Applications