CAN (Controller Area Network)

CAN은 무엇인가?

CAN 버스는 마이크로컨트롤러들 간의 통신을 위해 설계되었습니다 - 자동차 분야에서 이것은 엔진 관리 시스템, 변속장치 제어, 계기판, 그리고 차체 전자 기술 같은 온-보드 전자 제어 장치(ECU)들 간의 정보 교환에 사용되곤 합니다.
이론적으로 하나의 단일 네트워크에는 최대 2032개의 디바이스들이 한 개의 CAN 버스(한 개의 ID 번호를 가진 한 개의 노드를 가정) 에 연결될 수 있습니다. 그러나 하드웨어 (즉, 송수신기)의 현실적인 제한으로 인해, 이것은 실제적으로는 한 개의 단일 네트워크에 최대 110개 노드들을 (필립스 82C250 CAN 컨트롤러를 사용) 허용합니다. CAN은 최대 1 Mbit/sec 의 데이터 통신을 제공하여, 실시간 제어를 촉진합니다. 덧붙여, 오류 제한(Error Confinement)과 오류 탐색(Error Detection) 기능들은 noise-critical 환경에서의 신뢰성을 더욱 높여줍니다.

누가 CAN을 개발했는가?

Controller Area Network는 원래 1980년대 후반에 자동차 산업을 위해 독일 회사 Robert Bosch GmbH에 의해 개발되었습니다. 그들이 CAN을 개발하게 된 동기는, 차량에 더욱 많은 기능들이 필요로 되는데, 이는 대부분 전자식으로 작동되며 더욱 많은 배선을 필요로합니다. 또한 다른 온-보드 시스템과의 어떤 통신 형태를 필요로 하는 것이기 때문에, 현대 자동차에서의 내부-ECU 통신에 필요한 꾸준히 증가하고 있는 거대한 배선 작업의 문제에 대한 해결책을 제공하기 위해서였습니다. 그들의 결론은 모든 온-보드 주변장치들이 부착될 수 있는 하나의 단일 네트워크 버스를 설계하는 것이였습니다. 1993년 CAN은 표준 ISO 11898(고속 애플리케이션용) 과 ISO 11519(저속 애플리케이션용) 가 되었습니다. 이것은 다중-마스터 직렬 통신 버스로 이것의 기본 설계 사양은 고속, 높은 잡음 면역성, 그리고 오류-검출 기능들에 적합하게 되어있습니다. 이러한 기능들의 결과로, CAN 버스는 기타 제조 산업과 항공 우주 산업들에서도 폭넓게 사용되게 되었습니다.

CAN은 어떻게 동작하는가?

CAN 통신 프로토콜은 CAN 버스에서 디바이스들 통신 사이로 데이터가 전달되는 방법을 명시합니다. 이것은 ISO의 개방형 시스템 상호연결 모델(Open System Interconnection model)을 따르며, 이 모델은 통신 네트워크 표준인 일곱 계층으로 되어 있습니다. 이 OSI 모델은 두 개 네트워크 노드들 간의 계층화된 통신 시스템을 기술하며, 이론상 각 계층은 로컬 모드에서는 오직 자신의 직접적인 위, 아래의 계층들과 통신할 수 있으며, 원격 모드에서는 동등한 계층과 통신할 수 있습니다. OSI 모델의 계층들은 아래의 표에 나와 있습니다. 사실 CAN 프로토콜은 OSI 모델의 가장 낮은 두 개 층들로 설명될 수 있습니다 - 데이터 링크 계층과 물리적 계층. 애플리케이션 계층 프로토콜들은 개별적인 CAN 사용자들에 의해 개발된 독자 구조, 또는 특정

산업 내에서 사용되는 새로운 표준들 중의 하나입니다. 프로세스-제어/제조 분야에서 사용되는 공통적인 애플리케이션 표준 층은 DeviceNet으로, 이것은 PLC의 네트워킹과 지능 센서들 그리고 액츄에이터에 특히 적합합니다. 자동차 산업에서 많은 제조업체들은 그들 자신의 독자적인 표준을 사용하고 있습니다.

7	Application Layer	최상위층. 이것은 사용자가 네트워크와 상호 작용하는데 사용되는 소프트웨어를 설명하는 계층입니다. 예를 들어 DeviceNet이 해당됩니다.
6	Presentation Layer	변환될 데이터의 구문을 서술합니다. - 예를 들어 서로 다른 수학 포맷을 사용하는 두 개 시스템들 간의 부동 소수점 수 변환
5	Session Layer	하위 계층들에 의해 처리된 패킷들보다 큰 데이터 순차들 처리에 관하여 서술합니다.
4	Transport Layer	두 개 통신 노드들 간의 데이터 전송 품질과 속성을 설명합니다. 재전송과 오류 복구같은 문제들을 다룹니다.
3	Network Layer	다양한 데이터 링크를 거쳐 일련의 교환들이 어떻게 한 네트워크에서 임의의 두 개 노드들 간의 데이터를 전송할 수 있는지 설명합니다. 라우팅과 어드레싱 같은 문제들을 다룹니다.
2	Data Link Layer	특정 매체를 통해 전송된 데이터 비트의 배열과 조직을 서술합니다. 예를 들어, 이 계층은 checksum과 framing을 다룹니다.
1	Physical Layer	교환된 신호들의 해석, 전기적 특성들과 함께 통신 매체의 물리적 특징들도 서술합니다.

물리 계층에서, CAN은 광섬유 또는 꼬인 2선(가장 보편적) 같은 다양한 종류의 매체를 사용하여 통신할 수 있습니다. 꼬인 2선 시그널링은 각각의 전선에서 서로 다른 전압들을 사용하여 실행되므로 (balanced-line signalling 으로도 알려져 있습니다), 한 전선에서의 신호 전압은 또한 다른 전선에서 전송되지만, 반전됩니다. 수신기에서, 이 신호는 한 신호를 반전하고 두 개의 신호를 합해서 복원됩니다 - 이것은 두 개 전선들에 대해 공통적인 것이므로, 이 방법은 버스상에서 발견된 어떤 노이즈도 줄일 수 있습니다. 바로 이 과정에서 CAN은 자체의 잡음 면역(noise immunity)과 결함 허용(fault tolerance) 기능들을 유도합니다. 두 개의 전선들은 CAN_H (또는 CAN High)와 CAN_L (또는 CAN Low)로 불리웁니다. 정지 상태(또는 "recessive") 에서 CAN_H 와 CAN_L 은 2.5V에 놓입니다. 이것은 디지털 "1"로 표시되며, 또한 "recessive" 비트로도 알려져 있습니다. 디지털 '0' 은 또한 "dominant' 비트로도 알려져 있으며, CAN_L보다 큰 CAN_H에 의해 지시됩니다. 일반적으로 디지털 '0' 의 경우, 관련된 전압은 CAN_H = 3.5V 그리고 CAN_L = 1.5V입니다.

CAN의 특징

CAN 의 가장 매력적인 특징들은 다음과 같습니다:

- 낮은 비용.
- 극대화된 견고성
- 빠른 데이터 전송 속도 (최대 1MBit/sec)
- 신뢰성. 탁월한 오류 처리와 오류 제한 기능들
- 결함 메시지들의 자동적인 재-전송
- 물리적으로 결함이 추정되는 노드들의 자동적인 버스 연결절단
- 기능위주의 어드레싱 - 데이터 메시지들은 소스 혹은 목적지 주소들을 포함하지 않으며, 그들의 함수 그리고(또는) 우선순위와 연관된 식별자들만을 포함합니다.

CSMA/CD-NDBA

CSMA/CD 는 충돌 탐지 기능을 가진 Carrier Sense Multiple Access 를 나타냅니다. CSMA/CD를 이용하면 버스 주소지정이 버스 상(carrier sense)에서 반송파를 감지(listening)하여 이루어지며, 단지 버스가 유휴상태일 때 전송됩니다. 이 방식에서, 다중 노드들이 동일한 네트워크에 연결되는 것이 가능합니다. 충돌이 탐지되면, 재전송 하기 전에 임의 시간이 지날 때까지 전송을 시작했던 모든 노드들이 다시 "listen"으로 되돌아가게 됩니다. 그러나 이 기법은 과중하게-로드된 버스에서는 여전히 약간의 지연을 유발합니다.

이러한 지연을 피하기 위해, CAN 버스에서 두 개 노드들이 동시에 전송할 때, 한 개 메시지가 우선 순위를 갖게하는 방법이 필요합니다 - 이것은 'Non-Destructive Bit-wise Arbitration' 을 사용하여 달성됩니다. CAN 버스의 각 노드는 유일한 식별자(ID)를 가지며, 그것은 11비트 또는 29 비트 숫자입니다. CAN은 이진 0 이 이진 1 에 우선하도록 결정합니다. 따라서 더 낮은 ID 번호 - 더 높은 우선 순위, 그러므로 식별자 0 (즉, 11 이진 0 들) 이 버스에서 최고의 우선 순위를 갖습니다. 이것을 알아보는 다른 방법으로 메시지 충돌 상황이 있는데, 다른 노드가 1 을 보낼 때 0 을 보내는 최초의 노드가 버스 제어를 획득하게 되고 성공적으로 자신의 메시지를 전송하는 것입니다.

CAN higher level protocol

CAN higher level protocol (또한 Application layer protocol로도 알려져 있습니다)은 현재의 하위 CAN 계층(물리 계층과 데이터 링크 계층)들의 "위"에서 실행되는 프로토콜입니다. 상위 단계 프로토콜은 CAN의 물리 계층과 데이터 링크 계층들을 개발된 응용 계층의 밑바탕으로 사용합니다. 많은 시스템들, (예. 자동차 산업)은 독자적 응용 계층을 사용하지만, 많은 다른 산업들에서, 이 방법은 비용-효과적이지 않습니다. 여러 단체들이 시스템 통합을 쉽게 하기 위해서 표준화된 개방형 애플리케이션 계층들을 개발하고 있습니다.
몇가지 이용할 수 있는 CAN 상위 단계 프로토콜들입니다:

- Open DeviceNet Vendors Association 의 DeviceNet
- Honeywell 의 Smart Distributed System (SDS)
- CiA의 CAN Application Layer (CAL)

- CiA의 CANOpen (subset of CAL)
- Kvaser사의 CANKingdom

Bitwise arbitration

CAN 버스에서 두 개 노드들이 동시에 전송할 때, 한 메시지가 우선 순위를 갖는 방법이 필요합니다 -이것은 'non-destructive bit-wise arbitration' 을 사용하여 달성됩니다.; CAN 버스의 각 노드는 유일한 식별자(ID)를 가지며, 그것은 11비트 또는 29 비트 숫자입니다. CAN은 이진 0 이 이진 1 에 우선하도록 결정합니다. 따라서 더 낮은 ID 번호 - 더 높은 우선 순위, 그러므로 식별자 0 (즉, 11 이진 0 들)이 버스에서 최고의 우선 순위를 갖습니다. 이것을 알아보는 다른 방법으로 메시지 충돌 상황이 있는데, 다른 노드가 1 을 보낼 때 0 을 보내는 최초의 노드가 버스 제어를 획득하게 되고 성공적으로 자신의 메시지를 전송하는 것입니다.

BasicCAN 과 FullCAN

BasicCAN 과 FullCAN 이라 이름 붙여진, 두 개의 일반적인 CAN 접근법이 있습니다. 이것들은 들어오고 나가는 데이터가 처리되는 방식에서 서로 다릅니다. 간단히 말하면, BasicCAN은 CAN 메시지 전송과 수신을 처리하기 위해 호스트 CPU를 필요로 하며, 프레임 저장소를 다룹니다. FullCAN은 메시지 전송, 메시지 수신 그리고 최대 16개 메시지들의 저장을 CAN 컨트롤러에 맡깁니다. CPU는 정보가 필요할 때 CAN 컨트롤러에 문의합니다. 이 방식에서, FullCAN은 CAN 처리의 책임을 CPU에서 배제시킵니다 - 다른 작업들을 처리할 수 있는 자유를 부여. FullCAN에서 CAN 컨트롤러는 인터럽트가 설정되어 있다면 CPU를 인터럽트 할 수 있습니다 - 그리고 어떤 지정된 ID를 가진 메시지가 수신된다면 컨트롤러가 CPU를 인터럽트 하면서 "필터링 허용' 같은 작업들을 처리할 수 있습니다.

Standard CAN 과 Extended CAN

표준 CAN에서는 식별자들이 11비트 길이를 가지며 확장 CAN에서는 식별자들이 29비트 길이를 갖습니다. CAN 프로토콜 버전 2.0 에서 명시된 것에 의하면, V2.0A 로 컴파일하는 CAN 컨트롤러는 반드시 11-비트 식별자를 가져야만 합니다. 반면 V2.0B 에서는, 11비트 또는 29비트 아무 것이나 될 수 있습니다. V2.0B active를 이용하면, CAN 컨트롤러는 표준 포맷과 확장 포맷 모두를 전송하고 수신할 수 있습니다. V2.0B passive를 이용하면, CAN 컨트롤러는 표준 프레임을 전송하고 수신하게 되며 확장 프레임은 오류 없이 무시하게 됩니다.

CAN2.0A 와 CAN2.0B

CAN 2.0A 는 표준 CAN을 위한 규격서입니다. CAN 2.0B 는 확장 CAN을 위한 규격서입니다