**BOSCH CAN Specification Version 2.0**

**Recital** 상설

점점 더 많은 응용 프로그램에 대한 직렬 통신의 수용 및 도입은 통신 기능에 대한 메시지 식별자의 할당이 특정 응용 프로그램에 대해 표준화되어야한다는 요구 사항을 야기했습니다. 원래 11 개의 식별자 비트로 정의 된 주소 범위가 확장되면 이러한 응용 프로그램을 CAN을 통해 더 쉽게 구현할 수 있습니다. 따라서 29 비트로 정의 된 더 큰 주소 범위를 제공하는 두 번째 메시지 형식 ( '확장 형식') 이 도입되었습니다. 이렇게 하면 구조화가 잘된 명명 체계를 정의하는 것과 관련하여 시스템 설계자를 해치지 않아도됩니다. 확장 된 형식으로 제공되는 식별자 범위가 필요없는 CAN 사용자는 기존의 11 비트 식별자 범위 ( '표준 형식')를 계속 사용할 수 있습니다. 이 경우 시장에서 이미 사용 가능한 CAN 구현 또는 두 가지 형식을 구현하는 새로운 컨트롤러를 사용할 수 있습니다.

표준 및 확장 형식을 구별하기 위해 CAN 규격 1.2에 정의 된대로 CAN 메시지 형식의 첫 번째 예약 비트가 사용됩니다. 이는 CAN Specification 1.2의 메시지 형식이 표준 형식과 동일하므로 여전히 유효합니다. 또한 표준 형식 및 확장 형식의 메시지가 동일한 네트워크 내에서 공존 할 수 있도록 확장 형식이 정의되었습니다.

이 CAN 사양은 다음 두 부분으로 구성됩니다.

- CAN 규격 1.2에 정의 된 CAN 메시지 형식을 설명하는 Part A

- 표준 및 확장 메시지 형식을 모두 설명하는 파트 B

이 CAN 사양 2.0과 호환되기 위해서는 CAN 구현이 Part A 또는 Part B와 호환되어야합니다.

**Note**

CAN의 파트 A 또는 이전 CAN 스펙에 따라 설계된 CAN 구현과 본 스펙의 파트 B에 따라 설계된 CAN 구현은 확장 된 형식을 사용하지 않는 한 서로 통신 할 수 있습니다.

**PART A**

**1 INTRODUCTION................................................................................4**

소개

**2 BASIC CONCEPTS............................................................................5**

기본 개념

**3 MESSAGE TRANSFER .....................................................................10**

메시지 전송

**3.1 Frame Types ......................................................................................10**

프레임 유형

**3.1.1 DATA FRAME ....................................................................................10**

데이터 프레임

**3.1.2 REMOTE FRAME ..............................................................................15**

원격 프레임

**3.1.3 ERROR FRAME.................................................................................16**

오류 프레임

**3.1.4 OVERLOAD FRAME..........................................................................17**

오버로드 프레임

**3.1.5 INTERFRAME SPACING...................................................................18**

프레임 간 간격

**3.2 Definition of TRANSMITTER/RECEIVER ..........................................20**

트랜스미터 / 리시버의 정의

**4 MESSAGE VALIDATION ...................................................................21**

메시지 확인

**5 CODING.............................................................................................22**

코딩

**6 ERROR HANDLING...........................................................................23**

오류처리

**6.1 Error Detection ...................................................................................23**

오류감지

**6.2 Error Signalling...................................................................................23**

오류 신호

**7 FAULT CONFINEMENT.....................................................................24**

잘못된 구성

**8 BIT TIMING REQUIREMENTS ..........................................................27**

비트 타이밍 요구사항

**9 INCREASING CAN OSCILLATOR TOLERANCE..............................31**

증가하는 오실레이터 공차

**9.1 Protocol Modifications ........................................................................31**

프로토콜 수정

**1 INTRODUCTION 소개**

CAN (Controller Area Network)은 매우 높은 수준의 보안으로 분산 된 실시간 제어를 효율적으로 지원하는 직렬 통신 프로토콜입니다. 이 제품의 적용 범위는 고속 네트워크에서 저가의 멀티 플렉스 배선까지 다양합니다. 자동차 전자 장치에서 엔진 제어 장치, 센서, 스키드 방지 시스템 등은 CAN을 사용하여 최대 1Mbit / s의 비트율로 연결됩니다. 동시에 차체 전자 장치를 제작하는 것이 비용면에서 효율적입니다. 램프 클러스터, 전기 창 등 필요한 경우 와이어 링 하네스를 교체하십시오. (뭔소린지;; lamp clusters, electric windows etc. to replace the wiring harness otherwise required)

이 사양의 의도는 두 CAN 구현 간의 호환성을 달성하는 것입니다. 그러나, 호환성은 예를 들어, 전기 특성 및 전송할 데이터의 해석 설계 투명성과 구현 유연성을 달성하기 위해 CAN은 다른 계층으로 세분되었습니다.

- (CAN-) 객체 계층

- (CAN-) 전달 층

- 물리 계층

객체 계층과 전송 계층은 ISO / OSI 모델에 의해 정의 된 데이터 링크 계층의 모든 서비스와 기능을 포함합니다. 객체 계층의 범위에는 다음이 포함됩니다.

- 어떤 메시지가 전송 될지 찾기

- 전송 계층에 의해 수신 된 메시지가 실제로 사용될 것인지를 결정하고,

- 어플리케이션 계층 관련 하드웨어에 대한 인터페이스를 제공합니다.

객체 처리를 정의하는 데는 많은 자유가 있습니다. 전송 계층의 범위는 주로 전송 프로토콜, 즉 프레이밍(framing) 제어, 중재, 오류 검사, 오류 신호 및 오류 감금을 수행하는 것 입니다. 전송 계층 내에서 버스가 새로운 전송을 시작하기 위해 자유로운지 또는 수신이 방금 시작 중인지 여부가 결정됩니다. 또한 비트 타이밍의 일부 일반적인 기능은 전송 계층의 일부로 간주됩니다. 전송 계층의 특징 상, 변경을 위한 자유가 없습니다.

물리 계층의 범위는 모든 전기적 특성에 대해 서로 다른 노드 사이의 비트 전송을 의미합니다. 물론 하나의 네트워크 내에서 물리적 계층은 모든 노드에 대해 동일해야합니다. 그러나 물리 계층을 선택할 때 많은 자유가 있을 수 있습니다.

이 규격의 범위는 전송 레이어와 그 주변 레이어에 대한 CAN 프로토콜의 결과를 정의하는 것 입니다.

**2 BASIC CONCEPTS 기본 개념**

CAN은 다음과 같은 속성을 가집니다.

- 메시지의 우선 순위 지정

- 대기 시간 보장

- 구성 유연성

- 시간 동기화 된 멀티 캐스트 수신

- 시스템 전반의 데이터 일관성

- 멀티 마스터

- 오류 감지 및 신호 처리

- 버스가 다시 유휴 상태가 되는 즉시 손상된 메시지의 자동 재전송

- 일시적인 오류와 노드의 영구적인 실패와 결함 노드의 자율적인 전환 사이의 구별

CAN 노드의 계층화 된 구조

|  |
| --- |
| 응용 계층 |
|  |
| 객체 계층  - 메시지 필터링  - 메시지 및 상태 처리 |
| 전송 계층  - 결함 제한  - 오류 감지 및 신호  - 메시지 유효성 검사  - 승인(Acknowledgment)  - 중재  - 메시지 프레이밍(Framing)  - 전송 속도 및 타이밍 |
| 물리 계층  - 신호 레벨 및 비트 표현  - 전송 매체 |

- 물리 계층은 신호가 실제로 전송되는 방법을 정의합니다. 이 규격 내에서 물리 계층은 전송 매체 및 신호 레벨 구현이 자신의 애플리케이션에 맞게 최적화 될 수 있도록 정의되지 않습니다.

- 전송계층은 CAN 프로토콜의 커널을 나타냅니다. 그것은 수신 된 메시지를 객체 계층에 제시하고 객체 계층에서 전송할 메시지를 받아들입니다. 전송 계층은 비트 타이밍 및 동기화, 메시지 프레이밍, 중재, 수신 확인, 오류 감지 및 신호 처리, 오류 제한을 담당합니다.

- 객체 계층은 상태 및 메시지 처리뿐만 아니라 메시지 필터링과 관련이 있습니다.

이 규격의 범위는 전송 레이어와 그 주변 레이어에 대한 CAN 프로토콜의 결과를 정의하는 것이다.

Messages 메시지

버스에 대한 정보는 길이가 다르지만 제한적인 고정된 형식의 메시지로 전송됩니다. ( 3절 : 메시지 전송 참조). 버스가 해제되면 연결된 장치가 새 메시지를 전송하기 시작할 수 있습니다.

Information Routing 정보 라우팅

CAN 시스템에서 CAN 노드는 시스템 구성에 관한 정보를 사용하지 않습니다. (e.g. station addresses). 여기에는 몇 가지 중요한 결과가 있습니다.

System Flexibility 시스템 유연성

노드 및 응용 프로그램 계층의 소프트웨어 또는 하드웨어를 변경하지 않고도 노드를 CAN 네트워크에 추가 할 수 있습니다.

Message Routing 메시지 라우팅

메시지의 내용은 IDENTIFIER에 의해 명명됩니다. IDENTIFIER는 메시지의 목적지를 나타내지 않으며, 데이터의 의미를 기술하여 네트워크의 모든 노드가 메시지 필터링을 통해 데이터가 해당 데이터에 따라 작동하는지 여부를 결정할 수 있습니다.

Multicast 멀티캐스트

MESSAGE FILTERING의 개념의 결과로서, 임의의 수의 노드가 동일한 메시지를 수신하고 동시에 행동 할 수 있습니다.

Data Consistency 데이터 일관성

CAN 네트워크 내에서 모든 노드 또는 노드가 동시에 메시지를 받아 들일 수 있습니다. 따라서 시스템의 데이터 일관성은 멀티 캐스트 및 오류 처리의 개념에 의해 달성됩니다.

Bit rate 비트 전송률

CAN의 속도는 시스템마다 다를 수 있습니다. 그러나 주어진 시스템에서 비트율은 일정하고 고정되어 있습니다.

Priorities 우선 순위

IDENTIFIER는 버스 액세스 중에 정적 메시지 우선 순위를 정의합니다.

Remote Data Request 원격 데이터 요청

REMOTE FRAME을 송신함으로써, 데이터를 필요로 하는 노드는 다른 노드에 대응하는 데이터 프레임을 송신하도록 요구할 수 있습니다. DATA FRAME과 해당 REMOTE FRAME은 동일한 IDENTIFIER에 의해 명명됩니다.

Multimaster 멀티 마스터

버스가 비어있는 경우 모든 유닛이 메시지를 전송하기 시작할 수 있습니다. 전송 될 더 높은 우선 순위의 메시지를 갖는 유닛은 버스 액세스를 얻습니다.

Arbitration 중재

버스가 사용 가능할 때마다 모든 장치가 메시지 전송을 시작할 수 있습니다. 둘 이상의 장치가 동시에 메시지 전송을 시작하면 IDENTIFIER를 사용하는 비트 단위 중재에 의해 버스 액세스 충돌이 해결됩니다. 중재의 메커니즘은 정보나 시간이 손실되지 않는다는 것을 보장합니다. 동일한 IDENTIFIER가 있는 DATA FRAME과 REMOTE FRAME이 동시에 시작되면 DATA FRAME이 REMOTE FRAME보다 우선합니다. 중재가 진행되는 동안 모든 송신기는 전송된 비트의 레벨을 버스에서 모니터되는 레벨과 비교합니다. 이 레벨이 같으면 유닛은 계속해서 보낼 수 있습니다. '열성' 레벨이 전송되고 '우성' 레벨이 모니터링되면 (버스 값 참조) 유닛은 중재를 잃고 한 번 더 보내지 않고 철회해야합니다.

Safety 안전

데이터 전송의 최대한의 안전성을 확보하기 위해 모든 CAN 노드에서 오류 감지, 신호 및 자가 점검을 위한 강력한 조치가 구현됩니다.

Error Detection 오류 감지

오류를 탐지하기 위해 다음과 같은 조치가 취해졌습니다.

- 모니터링 (송신기는 전송할 비트 레벨을 버스에서 감지 된 비트 레벨과 비교)

- 순환 중복 검사 (Cyclic Redundancy Check)

- 비트 채우기 (Bit Stuffing)

- 메시지 프레임 확인

Performance of Error Detection 오류 감지 성능

오류 감지 메커니즘에는 다음과 같은 특성이 있습니다.

- 모든 전역 오류가 감지됩니다.

- 송신기의 모든 로컬 오류가 감지됩니다.

- 메시지에 최대 5개의 임의로 분산된 오류가 감지됩니다.

- 메시지에서 길이가 15 미만인 버스트 오류가 감지됩니다.

- 메시지에서 홀수 번호의 오류가 감지됩니다.

발견되지 않은 손상된 메시지에 대한 총 잔여 오류 확률 : 보다 작음

**message error rate \* 4.7 \* 10 ^ -11.**

Error Signalling and Recovery Time 오류 신호 및 복구 시간

손상된 메시지는 오류를 감지하는 모든 노드에 의해 플래그가 지정됩니다. 이러한 메시지는 중단되고 자동으로 재전송됩니다. 더 이상의 오류가 없는 경우, 오류 감지에서 다음 메시지 시작까지의 복구 시간은 최대 29 비트입니다.

Fault Confinement 결함 제한

CAN 노드는 짧은 장애를 영구 장애와 구별 할 수 있습니다. 결함 노드는 스위치 오프 됩니다.

Connections 연결

CAN 직렬 통신 링크는 여러 장치가 연결될 수 있는 버스입니다. 이 숫자에는 이론적인 제한이 없습니다. 실질적으로 총 유닛 수는 버스 라인의 지연 시간 및 / 또는 전기 부하에 의해 제한됩니다.

Single Channel 단일 채널

버스는 비트를 운반하는 단일 채널로 구성됩니다. 이 데이터로부터 재동기화 정보가 도출 될 수있습니다. 이 채널이 구현되는 방식은 고정되어 있지 않습니다. 이 채널이 구현되는 방식은 이 사양에서 수정되지 않았습니다. 예 : 단일 와이어 (플러스 접지), 2 개의 차동 와이어, 광섬유 등

Bus values 버스 값

버스는 두 개의 상보적인 논리 값 중 하나를 가질 수 있습니다 : '우성' 또는 '열성'. '우성' 및 '열성' 비트를 동시에 전송하는 동안 결과 버스 값은 '우성' 이됩니다. 예를 들어, 버스의 wired-AND 구현의 경우 '우성' 레벨은 논리 '0'으로 표시되고 '열성' 레벨은 논리 '1'로 표시됩니다. 논리 레벨을 나타내는 물리적 상태 (예를 들어, 전압, 빛)는 본 논문에서 제시되지 않습니다.

Acknowledgment 승인

모든 수신자는 수신중인 메시지의 일관성을 검사하고 일관된 메시지를 확인하고 일치하지 않는 메시지에 플래그를 지정합니다.

Sleep Mode / Wake-up 절전 모드 / 웨이크 업

시스템의 전력 소비를 줄이기 위해 CAN 장치는 내부 활동 및 연결이 끊어진 버스 드라이버 없이 절전 모드로 설정 될 수 있습니다. 절전 모드는 모든 버스 작동 또는 시스템의 내부 조건에 의해 웨이크 업 (wake-up)으로 완료됩니다. 웨이크 업에서는 전송 계층이 시스템의 오실레이터가 안정화 될 때까지 대기하지만 내부 액티비티가 다시 시작되고 버스 드라이버가 "온-버스" 로 다시 설정되기 전에 그 자체가 버스 활동(11개의 연속적인 '열성' 비트 확인)과 동기화 될 때까지 대기합니다. 절전 모드에 있는 시스템의 다른 노드를 깨우기 위해 가장 낮은 가능 식별자 (rrr rrrd rrrr; r = '열성 recessive' d = '우성 dominant')가 있는 특수 wake-up 메시지가 사용될 수 있습니다 .

**3 MESSAGE TRANSFER 메시지 전송**

**3.1 Frame Types 프레임 유형**

메시지 전송은 네 가지 프레임 유형으로 명시되고 제어됩니다 :

데이터 프레임은 송신기에서 수신기로 데이터를 전달합니다. REMOTE FRAME은 버스 장치에 의해 전송되어 동일한 IDENTIFIER로 DATA FRAME의 전송을 요청합니다.

버스 오류를 감지하면 모든 장치에서 오류 프레임을 전송합니다. OVERLOAD FRAME은 이전 및 이후의 DATA 또는 REMOTE FRAME 사이에 추가 지연을 제공하는 데 사용됩니다.

DATA FRAME과 REMOTE FRAME은 **INTERFRAME SPACE**에 의해 앞선 프레임과 분리되어 있습니다.

**3.1.1 DATA FRAME 데이터 프레임**

데이터 프레임은 7 개의 서로 다른 비트 필드로 구성됩니다.

START OF FRAME,

ARBITRATION FIELD,

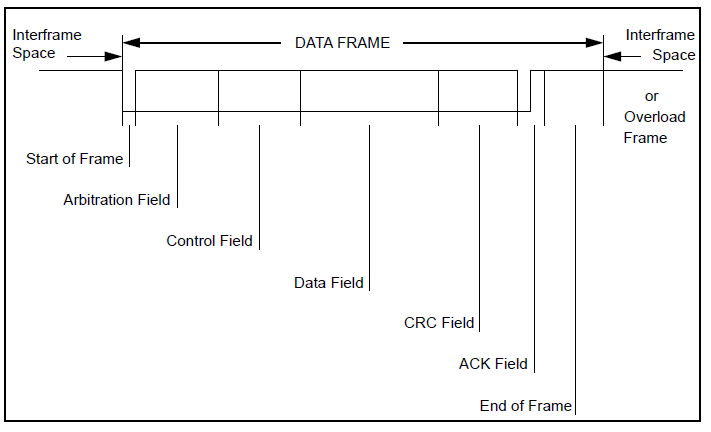
CONTROL FIELD,

DATA FIELD,

CRC FIELD,

ACK FIELD,

END OF FRAME. 데이터 필드의 길이는 0 일 수 있습니다.



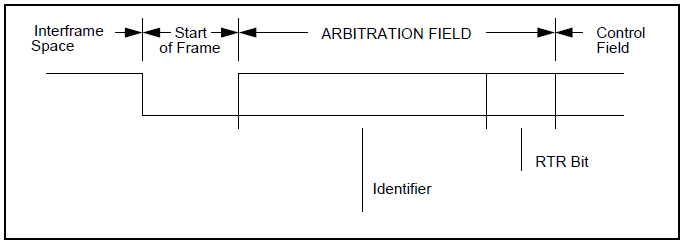
**START OF FRAME 프레임 시작**

DATA FRAMES 및 REMOTE FRAME의 시작을 표시합니다. 그것은 '우성' 단일 비트로 구성됩니다.

스테이션은 버스가 유휴 상태 일 때만 전송을 시작할 수 있습니다 (BUS IDLE 참조). 모든 스테이션은 전송을 먼저 시작하는 스테이션의 시작 프레임 ( '하드 동기화 참조' )에 의해 발생하는 리딩 에지(leading edge)에 동기화해야합니다.

**ARBITRATION FIELD 중재 영역**

중재 필드는 식별 기호와 RTR-BIT로 구성됩니다.



IDENTIFIER 식별자

IDENTIFIER의 길이는 11 비트입니다. 이 비트는 ID-10에서 ID-0 순서로 전송됩니다. 최하위 비트는 ID-0입니다. 7 개의 최상위 비트 (ID-10-ID-4)는 모두 '열성' 이어서는 안됩니다.

RTR BIT

원격 전송 요청 비트

데이터 프레임에서 RTR BIT는 '우성' 이어야합니다.

REMOTE FRAME 내에서 RTR BIT는 '열성'이어야합니다.

**CONTROL FIELD 제어 영역**

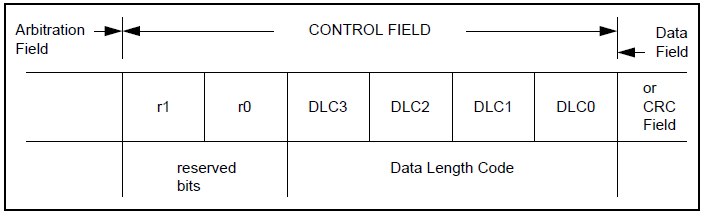
CONTROL FIELD는 6 비트로 구성됩니다. 여기에는 DATA LENGTH CODE와 향후 확장을 위해 예약된 2 비트가 포함됩니다. 예약된 비트는 '우성' 으로 보내야합니다.

수신기는 모든 조합에서 '우성' 비트와 '열성' 비트를 허용합니다.

DATA LENGTH CODE 데이터 길이 코드

DATA FIELD의 바이트 수는 DATA LENGTH CODE로 표시됩니다. 이 DATA LENGTH CODE는

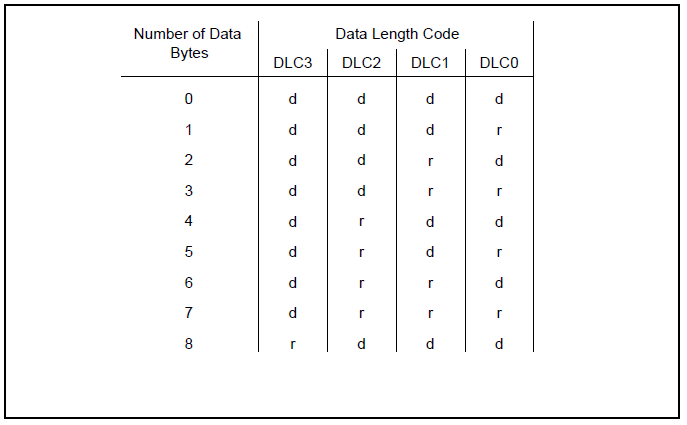
4비트 폭이며 CONTROL FIELD 내에서 전송됩니다.



DATA LENGTH CODE에 의한 데이터 바이트 수의 부호화

Abbreviations : **d ’dominant’ 우성**

**r ’recessive’ 열성**



데이터 프레임 : 허용되는 데이터 바이트 수 : {0,1,....,7,8}

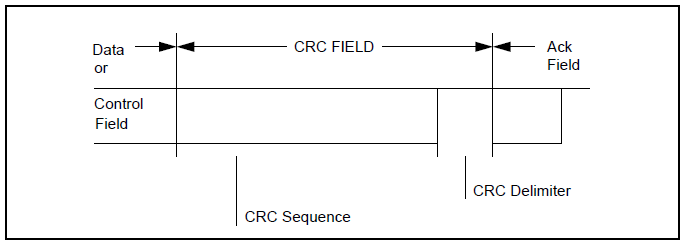
다른 값은 사용할 수 없습니다.

**DATA FIELD**

DATA FIELD는 DATA FRAME 내에서 전송 될 데이터로 구성됩니다. 0 ~ 8 바이트를 포함 할 수 있습니다. 각 바이트는 MSB를 먼저 전송한 8비트를 포함합니다.

**CRC FIELD**

CRC SEQUENCE 뒤에 CRC DELIMITER가 옵니다.



CRC SEQUENCE

프레임 체크 시퀀스는 127 비트 (BCH 코드) 미만의 비트 카운트를 갖는 프레임에 가장 적합한 순환 중복 코드로부터 유도된다.

CRC 계산을 수행하기 위해 분할될 다항식은 다항식으로 정의되며, 그 계수는 시작 프레임, 중재 필드, 제어 필드, 데이터 필드 (존재하는 경우)로 구성되는 destuffed 비트 스트림에 의해 주어지며, 가장 낮은 15 계수의 경우 0입니다. 이 다항식은 제너레이터-다항식에 의해 나누어집니다

(계수는 modulo-2로 계산됩니다)



이 다항식 부분의 나머지는 버스를 통해 전송되는 CRC SEQUENCE입니다. 이 기능을 구현하기 위해 15 비트 시프트 레지스터 CRC\_RG (14 : 0)를 사용할 수 있습니다. NXTBIT가 START OF FRAME에서 DATA FIELD의 끝까지의 destuffed 비트 스트림에 의해 주어진 비트 스트림의 다음 비트를 나타내는 경우, CRC SEQUENCE는 다음과 같이 계산됩니다.

CRC\_RG = 0; // initialize shift register

**REPEAT**

CRCNXT = NXTBIT EXOR CRC\_RG(14);

CRC\_RG(14:1) = CRC\_RG(13:0); // shift left by

CRC\_RG(0) = 0; // 1 position

**IF CRCNXT THEN**

CRC\_RG(14:0) = CRC\_RG(14:0) EXOR (4599hex);

**ENDIF**

UNTIL (CRC SEQUENCE starts or there is an ERROR condition)

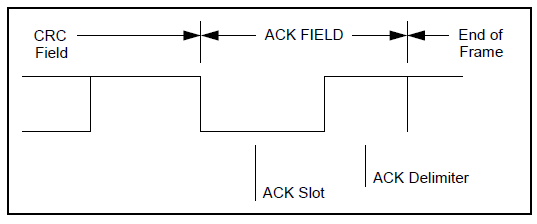
데이터 필드의 마지막 비트의 송/수신 후에, CRC\_RG는 CRC 시퀀스를 포함합니다.

CRC DELIMITER

CRC SEQUENCE 뒤에는 단일 열성 비트로 구성된 CRC ELIMITER가옵니다.

**ACK FIELD**

ACK FIELD는 2 비트 길이이고 ACK SLOT과 ACK DELIMITER를 포함합니다. ACK FIELD에서 전송 스테이션은 두 개의 '열성 비트'를 보냅니다. 유효한 메시지를 올바르게 수신한 수신자는 ACK SLOT 동안 '우성' 비트를 전송하여 송신자에게 이를 보고합니다 ( 'ACK' 를 보냄).



ACK SLOT

일치하는 CRC SEQUENCE를 수신한 모든 스테이션은 TRANSMITTER의 '열성 비트

(recessive bit)'를 '우성 비트 (dominant bit)'로 씁니다.

ACK DELIMITER

ACK DELIMITER는 ACK FIELD의 두 번째 비트이며 '열성 비트' 여야합니다. 결과적으로 ACK 슬롯은 두 개의 '열성' 비트 (CRC DELIMITER, ACK DELIMITER)로 둘러싸여 있습니다.

**END OF FRAME**

각 DATA FRAME 및 REMOTE FRAME은 7 개의 '열성'비트로 구성된 플래그 시퀀스에 의해 구분됩니다.

**3.1.2 REMOTE FRAME 원격 프레임**

특정 데이터에 대한 수신자 역할을 하는 스테이션은 REMOTE FRAME을 전송함으로써 소스 노드에 의한 각 데이터의 전송을 시작할 수 있습니다.

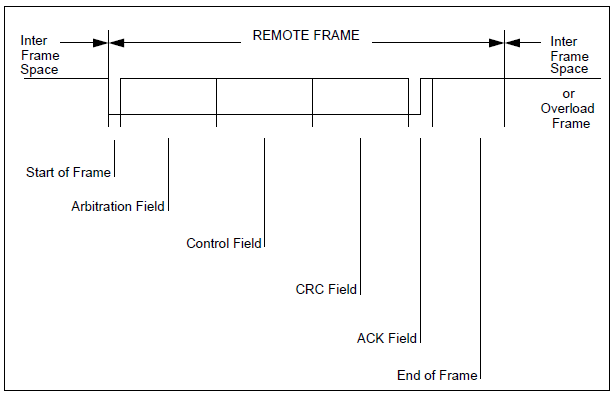
REMOTE FRAME은 6 개의 서로 다른 비트 필드로 구성됩니다.

**START OF FRAME, ARBITRATION FIELD, CONTROL FIELD, CRC FIELD, ACK FIELD,**

**END OF FRAME.**

DATA FRAME과는 달리 REMOTE FRAME의 RTR 비트는 '열성' 입니다. DATA LENGTH CODE의 값에 관계없이 DATA FIELD는 없으며, 허용 범위 0 ... 8 내의 어떤 값에도 사용 할 수 있습니다.

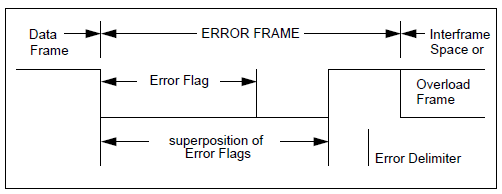
값은 해당 DATA FRAME의 DATA LENGTH CODE입니다.



RTR 비트의 극성은 전송 된 프레임이 데이터 프레임 (RTR 비트 '우성')인지 원격 프레임 (RTR 비트 '열성')인지를 나타냅니다.

**3.1.3 ERROR FRAME**

ERROR FRAME은 두 개의 다른 필드로 구성됩니다. 첫 번째 필드는 다른 스테이션에서 제공된 ERROR FLAGs의 중첩에 의해 제공됩니다. 다음 두 번째 필드는 ERROR DELIMITER입니다.



오류 프레임을 정확하게 종료시키기 위해, '오류 패시브' 노드는 버스가 적어도 3 비트 시간 동안 '버스 유휴' 상태 여야합니다 ( '오류 수동' 수신기에 로컬 오류가 있는 경우). 따라서 버스를 100%로 로드해서는 안됩니다.

ERROR FLAG

ERROR FLAG에는 **ACTIVE ERROR FLAG**와 **PASSIVE ERROR FLAG**의 두 가지 형식이 있습니다.

1. ACTIVE ERROR FLAG는 여섯 개의 연속 '우성' 비트로 구성됩니다.

2. PASSIVE ERROR FLAG는 다른 노드의 '우성' 비트로 덮어 쓰지 않는 한 6개의 연속적인 '열성'비트로 구성됩니다.

'에러 활성 (error active)' 스테이션은 ACTIVE ERROR FLAG의 전송에 의해 에러 상태를 검출한다.

ERROR FLAG의 형식은 START OF FRAME에서 CRC DELIMITER까지 모든 필드에 적용되는 비트 채우기 (CODING 참조)의 법칙을 위반하거나 ACK FIELD 또는 END OF FRAME 필드의 고정 양식을 파괴합니다. 결과적으로, 다른 모든 스테이션은 오류 조건을 감지하고 그 부분에서 오류 플래그 전송을 시작합니다. 따라서 버스에서 실제로 모니터링 할 수 있는 '우성' 비트 시퀀스는 개별 스테이션에서 전송한 다양한 ERROR FLAG를 중첩하여 발생합니다. 이 시퀀스의 전체 길이는 최소 6에서 최대 12 비트 사이에서 달라집니다. 에러 상태를 검출하는 '에러 수동' 스테이션은 패시브 에러 플래그 (PASSIVE ERROR FLAG)의 전송에 의해 이것을 알려줍니다. '오류 수동'스테이션은 PASSIVE ERROR FLAG (패시브 오류 플래그)의 시작에서 시작하여 같은 극성의 6 개의 연속 비트를 기다립니다. 이 6 개의 등가 비트가 검출되면 패시브 오류 플래그가 완료됩니다.

ERROR DELIMITER

ERROR DELIMITER는 8 개의 '열성' 비트로 구성됩니다.

ERROR FLAG가 전송 된 후 각 스테이션은 열성 비트를 전송하고 '열성'비트를 감지 할 때까지 버스를 모니터링합니다. 이후 7 개의 더 열성적인 비트를 전송하기 시작합니다.

**3.1.4 OVERLOAD FRAME**

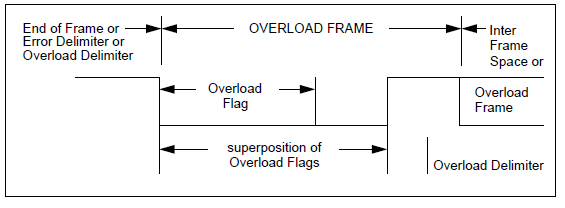
OVERLOAD FRAME에는 OVERLOAD FLAG 및 OVERLOAD DELIMITER의 두 비트 필드가 있습니다.

OVERLOAD 조건에는 두 가지 종류가 있는데, 두 가지 모두 OVERLOAD FLAG의 전송을 유도합니다.

1. 수신기의 내부 조건으로 다음 DATA FRAME 또는 REMOTE FRAME의 지연이 필요합니다.

2. 휴식 시간(INTERMISSION) 중에 '우성' 비트 감지

OVERLOAD 조건 1에 의한 OVERLOAD FRAME의 시작은 예상된 INTERMISSION의 첫 번째 비트 시간에서만 시작될 수 있지만 OVERLOAD 조건 2로 인한 OVERLOAD FRAME은 '우성' 비트 감지 후 한 비트 시작합니다.



다음 DATA 또는 REMOTE FRAME을 지연시키기 위해 최대 2 개의 OVERLOAD FRAME이 생성 될 수 있습니다.

OVERLOAD FLAG

6 개의 '우성' 비트로 구성됩니다. 전체 양식은 ACTIVE ERROR FLAG의 양식과 일치합니다.

OVERLOAD FLAG의 형태는 INTERMISSION 필드의 고정된 형태를 파괴합니다. 결과적으로 다른 모든 스테이션에서도 OVERLOAD 조건을 감지하고 OVERLOAD FLAG 전송을 시작합니다.

(일부 노드에서 INTERMISSION의 세 번째 비트 중에 '우성' 비트가 로컬로 탐지 된 경우 다른 노드는 OVERLOAD FLAG를 올바르게 해석하지 않으며, 이 6개의 '우성' 비트 중 첫 번째 비트를 START OF FRAME으로 해석하세요. 여섯 번째 '우성' 비트가 오류 상태를 일으키는 비트 스터핑의 규칙을 위반합니다. )

OVERLOAD DELIMITER

8 개의 '열성' 비트로 구성됩니다.

OVERLOAD DELIMITER는 ERROR DELIMITER와 동일한 형식입니다. OVERLOAD FLAG를 전송 한 후 스테이션은 '우성' 비트에서 '열성' 비트로의 전환을 감지 할 때까지 버스를 모니터링합니다.

이 시점에서 모든 버스 스테이션은 OVERLOAD FLAG 전송을 끝내고 모든 스테이션은 동시에 7 개의 '열성' 비트 전송을 시작합니다.

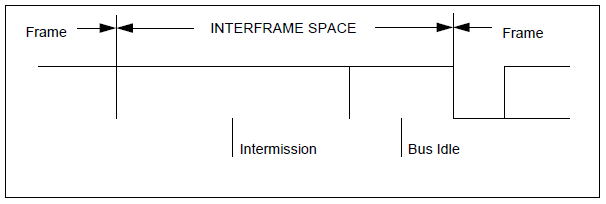
**3.1.5 INTERFRAME SPACING**

DATA FRAME과 REMOTE FRAME은 INTERFRAME SPACE라는 (DATA FRAME, REMOTE FRAME, ERROR FRAME, OVERLOAD FRAME) 비트 필드에 의해 이전 프레임과 구분됩니다. 반대로, 오버로드 프레임과 오류 프레임 앞에는 INTERFRAME SPACE가 없고 여러 오버로드 프레임은 INTERFRAME SPACE로 구분되지 않습니다.

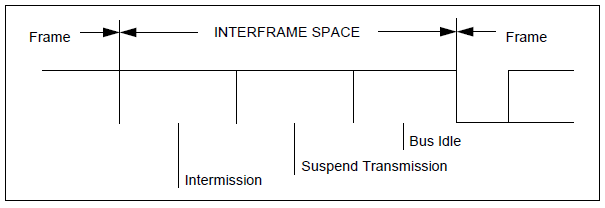
INTERFRAME SPACE

INTERMISSION 및 BUS IDLE 비트 필드를 포함하고, 이전 메시지의 송신기인 '오류 수동' 스테이션의 경우에는 일시 중지 된 전송을 포함합니다.

'오류 수동'이 아니거나 이전 메시지의 수신자인 스테이션의 경우 :



이전 메시지의 TRANSMITTER 인 '오류 수동' 스테이션의 경우 :



Suspend Transmission 전송 중단

INTERMISSION

세 개의 '열성' 비트로 구성됩니다.

INTERMISSION 동안 어떤 스테이션도 DATA FRAME 또는 REMOTE FRAME의 전송을 시작할 수 없습니다. 취할 수 있는 유일한 조치는 OVERLOAD 조건을 신호하는 것입니다.

BUS IDLE

버스 유휴 기간은 임의의 길이 일 수 있습니다. 버스는 free로 인식되며 전송할 대상이 있는 모든 스테이션이 버스에 액세스 할 수 있습니다. 다른 메시지의 전송 중에 전송을 위해 보류 중인 메시지는 INTERMISSION 다음의 첫 번째 비트에서 시작됩니다.

버스의 '우성' 비트 감지는 시작 프레임으로 해석됩니다.

SUSPEND TRANSMISSION

'오류 수동' 스테이션이 메시지를 전송 한 후에 INTERMISSION 다음에 8 개의 '열성' 비트를 전송하여 추가 메시지를 전송하거나 유휴 상태임을 인식합니다. 한편 (다른 스테이션에 의한) 전송이 시작되면 스테이션은 이 메시지의 수신자가 됩니다.

**3.2 Definition of TRANSMITTER / RECEIVER**

TRANSMITTER

메시지를 생성하는 단위를 해당 메시지의 "송신자 (TRANSMITTER)" 라고합니다. 버스가 유휴 상태가 되거나 장치가 중재를 잃을 때까지 장치는 TRANSMITTER를 유지합니다.

RECEIVER

메시지가 송신자가 아니고 버스가 유휴 상태가 아닌 경우, 장치는 메시지의 "수신자" 라고합니다.

**4 MESSAGE VALIDATION 메시지 검증**

메시지가 유효한 것으로 간주되는 시점은 메시지의 송신자와 수신자에 따라 다릅니다.

Transmitter:

END OF FRAME의 끝까지 오류가 없으면 송신기에 유효합니다. 메시지가 손상된 경우 재전송은 우선 순위에 따라 자동으로 수행됩니다. 다른 메시지와의 버스 액세스를 경쟁 할 수 있으려면 버스가 유휴 상태가 되는 즉시 재전송이 시작되어야합니다.

Receivers:

END OF FRAME의 마지막 1 비트까지 오류가 없는 경우 메시지는 수신자에게 유효합니다.

**5 CODING**

BIT STREAM CODING

프레임 세그먼트 시작, 중재 필드, 제어 필드, 데이터 필드 및 CRC 시퀀스는 비트 스터핑 (bit stuffing) 방법으로 코딩됩니다. 송신기가 전송될 비트 스트림에서 동일한 값의 5개의 연속적인 비트를 검출 할 때마다 자동으로 실제 송신된 비트 스트림에 상보 비트를 삽입합니다.

DATA FRAME 또는 REMOTE FRAME의 나머지 비트 필드 (CRC DELIMITER,

ACK FIELD 및 END OF OF FRAME)은 고정 된 형식이며 채워지지 않습니다. ERROR FRAME과 OVERLOAD FRAME은 고정 된 형식이기 때문에 비트 스터핑 방법으로 코딩되지 않습니다.

메시지 내의 비트 스트림은 NRZ (Non-Return-to-Zero) 방식에 따라 부호화 됩니다. 이것은 총 비트 시간 동안 생성 된 비트 레벨이 '우성' 또는 '열성' 임을 의미합니다.

메시지 내의 비트 스트림은 NRZ (Non-Return-to-Zero) 방식에 따라 부호화된다. 이것은 총 비트 시간 동안 생성 된 비트 레벨이 '지배적'또는 '열성'임을 의미합니다.

**6 ERROR HANDLING**

**6.1 Error Detection**

상호 배타적이지 않은 5가지 오류 유형이 있습니다.

- BIT ERROR

버스에 비트를 전송하는 장치도 버스를 모니터링합니다. BIT ERROR는 모니터링되는 비트 값이 전송 된 비트 값과 다른 비트 시간에 감지되어야합니다. 중재 필드의 채워진 비트 스트림 동안 또는 ACK 슬롯 동안 '열성' 비트의 전송은 예외입니다. 그러면 '우성' 비트가 모니터 될 때 BIT ERROR가 발생하지 않습니다. PASSIVE ERROR FLAG를 보내고 '우성' 비트를 검출하는 송신기는 이것을 BIT ERROR로 해석하지 않습니다.

- STUFF ERROR

STUFF ERROR는 비트 스터핑 (bit stuffing) 방법으로 코딩되어야하는 메시지 필드에서 6연속으로 동일한 비트 레벨의 비트 시간에서 감지되어야 합니다.

- CRC ERROR

CRC 시퀀스는 송신기에 의한 CRC 계산 결과로 구성됩니다. 수신기는 송신기와 동일한 방식으로 CRC를 계산합니다. 계산된 결과가 CRC 시퀀스에서 수신 된 것과 같지 않으면 CRC 오류를 감지해야합니다.

- FORM ERROR

고정 양식 비트 필드에 하나 이상의 잘못된 비트가 포함되어 있으면 양식 오류를 감지해야합니다.

- ACKNOWLEDGMENT ERROR 승인 오류

ACK SLOT 동안 '우성' 비트를 모니터하지 않을 때마다 ACKNOWLEDGMENT ERROR가 송신기에 의해 감지되어야합니다.

**6.2 Error Signalling**

에러 상태를 검출하는 스테이션은 ERROR FLAG를 전송하여 이것을 알려줍니다. '오류 활성' 노드의 경우 이는 활성 오류 플래그이고 '오류 수동' 노드의 경우 수동 오류 플래그 입니다. 어떤 스테이션에서 BIT ERROR, STUFF ERROR, FORM ERROR 또는 ACKNOWLEDGMENT ERROR가 발견 될 때마다 다음 비트에서 해당 스테이션에서 ERROR FLAG의 전송이 시작됩니다.

CRC 오류가 검출 될 때마다, 다른 조건을 위한 ERROR FLAG가 이미 시작되지 않은 한, ACK DELIMITER 다음의 비트에서 ERROR FLAG의 전송이 시작됩니다.

**7 FAULT CONFINEMENT 오류 정정**

결함 제한과 관련하여 유닛은 다음 세 가지 상태 중 하나에 있을 수 있습니다.

- ’error active’

- ’error passive’

- ’bus off’

'오류 활성' 장치는 일반적으로 버스 통신에 참여할 수 있으며 오류가 감지되면 ACTIVE ERROR FLAG (ACTIVE 오류 플래그)를 전송합니다.

'오류 패시브' 장치는 ACTIVE ERROR FLAG를 보내면 안됩니다. 버스 통신에 참여하지만 오류가 감지 된 경우 패시브 오류 플래그만 전송됩니다. 또한 전송 후 '오류 수동' 장치는 추가 전송을 시작하기 전에 대기합니다. (SUSPEND TRANSMISSION 참고)

'bus off' 장치는 버스에 영향을 미치지 않습니다. ( 예 : 출력 드라이버가 꺼졌습니다. )

오류 감금의 경우 모든 버스 장치에 두 개의 카운트가 구현됩니다.

1) TRANSMIT 오류 개수

2) RECEIVE 오류 개수

이 수는 다음 규칙에 따라 수정됩니다. :

(주어진 메시지 전송 중에 둘 이상의 규칙이 적용될 수 있음에 유의하십시오)

1. RECEIVER가 오류를 감지하면 RECEIVE ERROR COUNT는 감지 된 오류가 ACTIVE ERROR FLAG 또는 OVERLOAD FLAG 전송 중 BIT ERROR 일 때를 제외하고는 1씩 증가합니다.

2. RECEIVER가 ERROR FLAG를 전송 한 후 첫 번째 비트로 '우세' 비트를 감지하면 RECEIVE ERROR COUNT가 8 씩 증가합니다.

3. 송신기가 ERROR FLAG를 보내면 TRANSMIT ERROR COUNT가 8만큼 증가합니다.

예외 1 :

TRANSMITTER가 'error passive'이고 '우성' ACK를 탐지하지 못하여 ACKNOWLEDGMENT ERROR를 탐지하고 PASSIVE ERROR FLAG를 보내는 동안 '우성' 비트를 탐지하지 못하는 경우

예외 2 :

STUFFBIT가 RTR 비트 이전에 위치하며 STUFF ERROR가 발생하여 STUFFBIT가 RTR 비트 앞에 위치하고 '열성' 이었고 '열성'으로 전송되었지만 '우성' 으로 모니터링 되었기 때문에 TRANSMITTER가 ERROR FLAG를 보내는 경우

예외 1과 2에서 TRANSMIT ERROR COUNT는 변경되지 않습니다.

4. 송신기가 ACTIVE ERROR FLAG 또는 OVERLOAD FLAG를 보내는 동안 BIT ERROR를 감지하면 TRANSMIT ERROR COUNT가 8만큼 증가합니다.

5. 수신기가 ACTIVE ERROR FLAG 또는 OVERLOAD FLAG를 보내는 동안 BIT ERROR를 감지하면 RECEIVE ERROR COUNT가 8만큼 증가합니다.

6. 모든 노드는 ACTIVE ERROR FLAG, PASSIVE ERROR FLAG 또는 OVERLOAD FLAG를 전송 한 후 최대 7 개의 '우성' 비트를 허용합니다. 14 번째 연속 '우성' 비트 (ACTIVE ERROR FLAG 또는 OVERLOAD FLAG의 경우)를 검출 한 후 또는 PASSIVE ERROR FLAG 다음에 8 번째 연속 '우성' 비트를 검출 한 후, 그리고 연속적인 8개의 '우성' 비트 모든 송신기는 송신 오류 수를 8만큼 증가시키고 모든 수신자는 수신 오류 수를 8만큼 증가시킵니다.

7. 메시지가 성공적으로 전송 된 후 (ACK 수신 및 END OF FRAME이 완료 될 때까지 오류 없음) TRANSMIT ERROR COUNT는 이미 0이 아닌 한 1 씩 감소합니다.

8. 메시지의 성공적인 수신 (ACK SLOT까지 오류없는 수신 및 ACK 비트의 성공적인 전송) 후에 수신 오류 수는 1에서 127 사이의 경우 1 씩 감소합니다. RECEIVE ERROR COUNT가 0이면 0을 유지하고 127보다 크면 119와 127 사이의 값으로 설정됩니다.

9. TRANSMIT ERROR COUNT가 128 이상이거나 RECEIVE ERROR COUNT가 128 이상일 때 노드는 '오류 패시브' 입니다. 노드를 '오류 패시브' 로 만드는 오류 조건은 노드가 ACTIVE ERROR FLAG를 전송하게합니다.

10. TRANSMIT ERROR COUNT가 256보다 크거나 같으면 노드는 'bus off' 상태입니다.

11. TRANSMIT ERROR COUNT와 RECEIVE ERROR COUNT가 127보다 작거나 같으면 '오류 수동' 노드가 '오류 활성' 상태가 됩니다.

12. 'bus off' 상태 인 노드는 버스에서 11개의 연속적인 '열성' 비트가 128번 모니터 된 후에 오류 카운터가 모두 0으로 설정된 '오류 활성' (더 이상 'bus off'가 아님) 이 될 수 있습니다.

**Note:**

약 96보다 큰 오류 계수 값은 심하게 방해받는 버스를 나타냅니다. 이 상태를 테스트 할 수 있는 수단을 제공하는 것이 효과적 일 수 있습니다.

**Note:**

Start-up / Wake-up:

시작하는 동안 하나의 노드만 온라인 상태이고 이 노드가 일부 메시지를 전송하면 승인을 받지 않고 오류를 감지하고 메시지를 반복합니다. 이 오류로 인해 '오류 수동' 이 될 수 있지만 'bus off'인 것 은 아닙니다.

**8 BIT TIMING REQUIREMENTS**

**8 비트 타이밍 요구 사항**

NOMINAL BIT RATE

Nominal Bit Rate는 이상적인 송신기에 의한 재 동기화가 없을 때 전송 된 초당 비트 수입니다.

NOMINAL BIT TIME

NOMINAL BIT TIME = 1 / NOMINAL BIT RATE

Nominal Bit Time은 서로 겹치지 않는 별도의 시간 세그먼트로 나누어져 있다고 생각할 수 있습니다. 이 세그먼트들

- SYNCHRONIZATION SEGMENT (SYNC\_SEG)

- PROPAGATION TIME SEGMENT (PROP\_SEG)

- PHASE BUFFER SEGMENT1 (PHASE\_SEG1)

- PHASE BUFFER SEGMENT2 (PHASE\_SEG2)

그림 1과 같이 비트 시간을 형성하십시오.

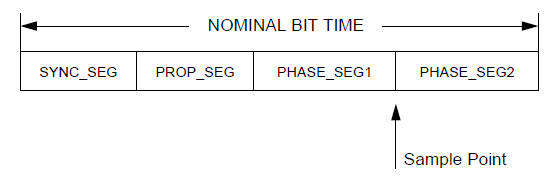


Fig. 1 Partition of the Bit Time 1비트 시간의 분할

SYNC SEG

비트 시간의 이 부분은 버스의 여러 노드를 동기화하는 데 사용됩니다. 가장자리가 이 세그먼트 내에 있어야합니다.

PROP SEG

비트 시간의 이 부분은 네트워크 내의 물리적 지연 시간을 보완하는 데 사용됩니다. 이것은 버스 라인에서의 신호 전파 시간, 입력 비교기 지연 및 출력 드라이버 지연의 두 배입니다.

PHASE SEG1, PHASE SEG2

이 위상 버퍼 세그먼트는 에지 위상 오차를 보상하는 데 사용됩니다. 이러한 세그먼트는 재 동기화를 통해 길어지거나 짧아 질 수 있습니다.

SAMPLE POINT

샘플 포인트는 버스 레벨이 읽혀지고 해당 비트의 값으로 해석되는 시점입니다. 위치는 PHASE\_SEG1의 끝 부분입니다.

INFORMATION PROCESSING TIME

정보 처리 시간은 후속 비트 레벨을 계산하기 위해 예약 된 샘플 포인트로 시작하는 시간 세그먼트입니다.

TIME QUANTUM

TIME QUANTUM은 오실레이터주기에서 파생 된 고정 된 시간 단위입니다. 최소한 1에서 32까지의 정수 값을 갖는 프로그래밍 가능한 프리스케일러가 존재합니다. MINIMUM TIME QUANTUM에서 시작하여 TIME QUANTUM은

TIME QUANTUM = m \* MINIMUM TIME QUANTUM

m은 프리스케일러 값

Length of Time Segments 시간 세그먼트의 길이

- SYNC\_SEG는 1 TIME QUANTUM입니다.

- PROP\_SEG는 1,2, ..., 8 TIME QUANTA로 프로그래밍 할 수 있습니다.

- PHASE\_SEG1은 1, 2, ..., 8 TIME QUANTA로 프로그래밍 할 수 있습니다.

- PHASE\_SEG2는 최대 PHASE\_SEG1이고 정보 처리 시간 입니다.

- 정보 처리 시간은 2 TIME QUANTA보다 길거나 같습니다.

비트 타임의 TIME QUANTA의 총 수는 최소 8에서 25까지 프로그래밍 할 수 있어야 합니다.

SYNCHRONIZATION RULES 동기화 규칙

하드 동기화 및 재동기화는 SYNCHRONIZATION의 두 가지 형태입니다. 그들은 다음 규칙을 준수합니다.

1. 한 비트 시간 내에 하나의 SYNCHRONIZATION 만 허용됩니다.
2. 이전 SAMPLE POINT (이전 읽기 버스 값)에서 감지 된 값이 가장자리 직후의 버스 값과 다른 경우에만 에지가 SYNCHRONIZATION에 사용됩니다.
3. 하드 동기화는 BUS IDLE 중에 '열성' 에지에 '열성' 이 있을 때마다 수행됩니다.
4. 규칙 1과 2를 수행하는 다른 모든 '열성' 에지에서 '우성' 에지 (선택적으로 '우세' 에서 '낮은 비트율의 경우 '우성 에지' ) 는 우위 비트를 전송하는 노드가 우위 비트를 전송한다는 것을 제외하고는 RESYNCHRONIZATION에 사용됩니다. 재결합을 위해 '열성' 에지에서 '열성' 에지만 사용되는 경우, 긍정적인 위상 오류로 '열성' 에지에서 '열성' 에지의 결과로 RESYNCHRONIZATION을 수행하지 않습니다.

**9 INCREASING CAN OSCILLATOR TOLERANCE**

**증가하는 오실레이터 공차**

이 절은 CAN 프로토콜의 상위 호환성 수정에 대해 Section 1-8 항에 명시된대로 설명합니다.

**9.1 Protocol Modifications 프로토콜 수정**

최대 오실레이터 허용 오차를 현재 가능한 0.5 %에서 1.5 %로 높이려면 기존 CAN 사양과 상향 호환되는 다음과 같은 수정이 필요합니다.

[1] CAN 노드가 INTERMISSION의 세 번째 비트에서 우성 비트를 샘플링하면 이 비트가 FROM 비트 시작으로 해석됩니다.

[2] CAN 노드가 전송을 기다리는 메시지를 가지고 INTERMISSION의 세 번째 비트에서 우성 비트를 샘플링하면, 이 비트는 START OF FRAME 비트로 해석되고, 다음 비트와 함께 처음 프레임 시작 비트를 전송하지 않고, 수신기가 되지 않으며 IDENTIFIER의 첫 번째 비트를 사용하여 메시지를 전송하기 시작합니다.

[3] CAN 노드가 ERROR DELIMITER 또는 OVERLOAD DELIMITER의 8 번째 비트 (마지막 비트)에서 우성 비트를 샘플링하면 다음 비트에서 OVERLOAD FRAME (ERROR FRAME이 아님) 전송을 시작합니다. 오류 카운터는 증가하지 않습니다.

[4] 우성 에지에서 열성 에지까지만 동기화에 사용됩니다.

기존 사양과 일치하여 다음 규칙이 여전히 유효합니다.

[5] 모든 CAN 컨트롤러는 START OF FRAME 비트에서 하드 동기화로 동기화됩니다.

[6] CAN 컨트롤러는 INTERMISSION의 세 비트를 카운트 할 때까지 START OF FRAME 비트를 전송하지 않습니다.

이 수정으로 인해 1.58 %의 최대 오실레이터 허용 오차와 최대 125Kbps의 버스 속도에서 세라믹 공진기를 사용할 수 있습니다. CAN 프로토콜의 전체 버스 속도 범위를 위해서는 여전히 수정 발진기가 필요합니다. 향상된 프로토콜과 기존 프로토콜의 호환성은 다음과 같이 유지됩니다.

[7] 하나의 동일한 네트워크에서 사용되는 향상된 프로토콜과 기존 프로토콜을 갖춘 CAN 컨트롤러에는 모두 수정 발진기가 제공됩니다.

발진기 정확도가 가장 높은 칩은 모든 다른 노드에서 요구되는 발진기 정확도를 결정합니다. 세라믹 공진기는 네트워크의 모든 노드가 확장 프로토콜을 사용할 때만 사용할 수 있습니다.

**PART B**

**1 INTRODUCTION................................................................................34**

소개

**2 BASIC CONCEPTS............................................................................36**

기본 개념

**3 MESSAGE TRANSFER .....................................................................42**

메시지 전송

**3.1 Frame Formats...................................................................................42**

프레임 형식

**3.2 Frame Types ......................................................................................42**

프레임 타입

**3.2.1 DATA FRAME ....................................................................................42**

데이터 프레임

**3.2.2 REMOTE FRAME ..............................................................................49**

원격 프레임

**3.2.3 ERROR FRAME.................................................................................50**

오류 프레임

**3.2.4 OVERLOAD FRAME..........................................................................51**

오버로드 프레임

**3.2.5 INTERFRAME SPACING...................................................................53**

프레임 간 간격

**3.3 Conformance with regard to Frame Formats .....................................55**

프레임 포맷에 대한 적합성

**3.4 Definition of TRANSMITTER/RECEIVER ..........................................55**

TRANSMITTER/RECEIVER의 정의

**4 MESSAGE FILTERING......................................................................56**

메시지 필터링

**5 MESSAGE VALIDATION ...................................................................57**

메시지 확인

**6 CODING.............................................................................................58**

코딩

**7 ERROR HANDLING...........................................................................59**

오류 처리

**7.1 Error Detection ...................................................................................59**

오류 감지

**7.2 Error Signalling...................................................................................60**

오류 신호

**8 FAULT CONFINEMENT.....................................................................61**

잘못된 구성

**9 OSCILLATOR TOLERANCE..............................................................64**

오실레이터 공차

**10 BIT TIMING REQUIREMENTS ..........................................................27**

비트 타이밍 요구사항