配布用

車載通信入門
はじめてのLIN

2021年06月23日

株式会社 東海理化
稲垣 修

目次

- はじめに
- LINとは
- LINプロトコルについて
- LINプロトコルの詳細
- LINの実現方法
- 実際に使用する際の注意点

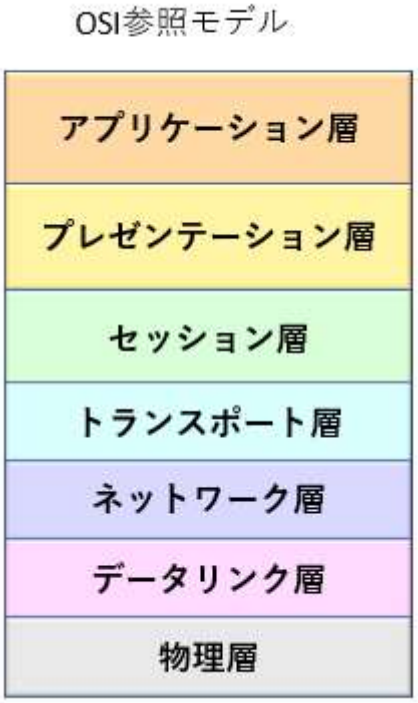
はじめに

- これから自動車業界で業務を行う皆さんに、通信、ソフトウェア品質に関する情報の提供です

  是非、覚えて帰ってください

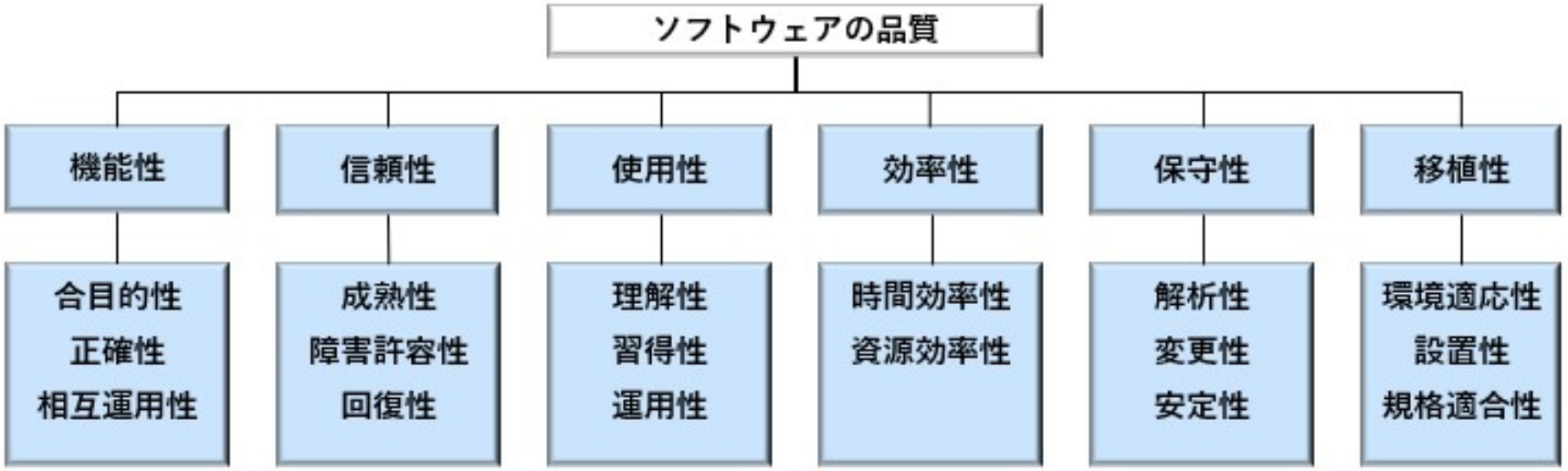
はじめに

・ISO7498 OSI参照モデル
(通信機能を階層化したモデル)



はじめに

・ソフトウェア品質特性　ISO9126



LINとは

LINとは

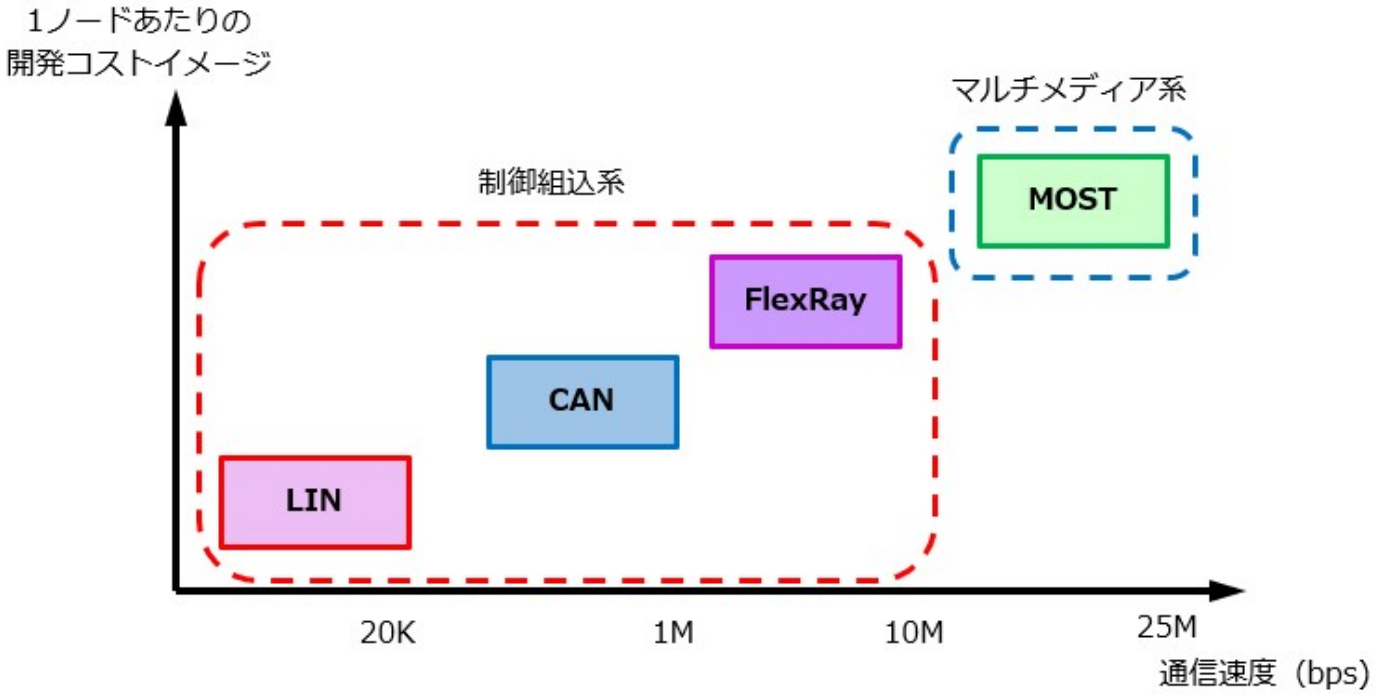
■ LIN (Local Interconnect Network)

- **LINコンソーシアムで発案された通信の規格**
  欧州の自動車メーカ、半導体メーカが主導で規格を策定
  運営委員会：Audi, BMW, Daimler, NXP, Menter, Volkswagen, Volvo など
- **低コストネットワークの実現を目的**
  <CANプロトコルと比較した場合>
  - 部品コスト削減 （部品点数の削減、廉価部品へ変更）
    ・ワイヤ：2本 → 1本
    ・MCU用発振子：水晶／セラミック発振子 → RC発振 または マイコン内蔵発振回路
    ・トランシーバ：差動式AMP → コンパレータ方式
  - 部品点数削減によるアセンブリコストの削減
  - 通信ソフト開発コストの削減
    ・開発負荷が小さい（ネットワークマネージメントが容易）
    ・インプリ用コンフィグレータ、適合試験仕様あり
- **マンマシンインタフェースがアプリターゲット**
  ドアスイッチ、ドアミラー、ダッシュボード回りのスイッチやアクチュエータ制御など
- **CANのサブネットワークとして普及**
  普及状況
  海外自動車メーカ Audi, BMW, Daimler, Volvo, VW, Fiat, Opel, Renault, PSA
  国内自動車メーカ トヨタ、ホンダ、日産、スバル　など

LINとは

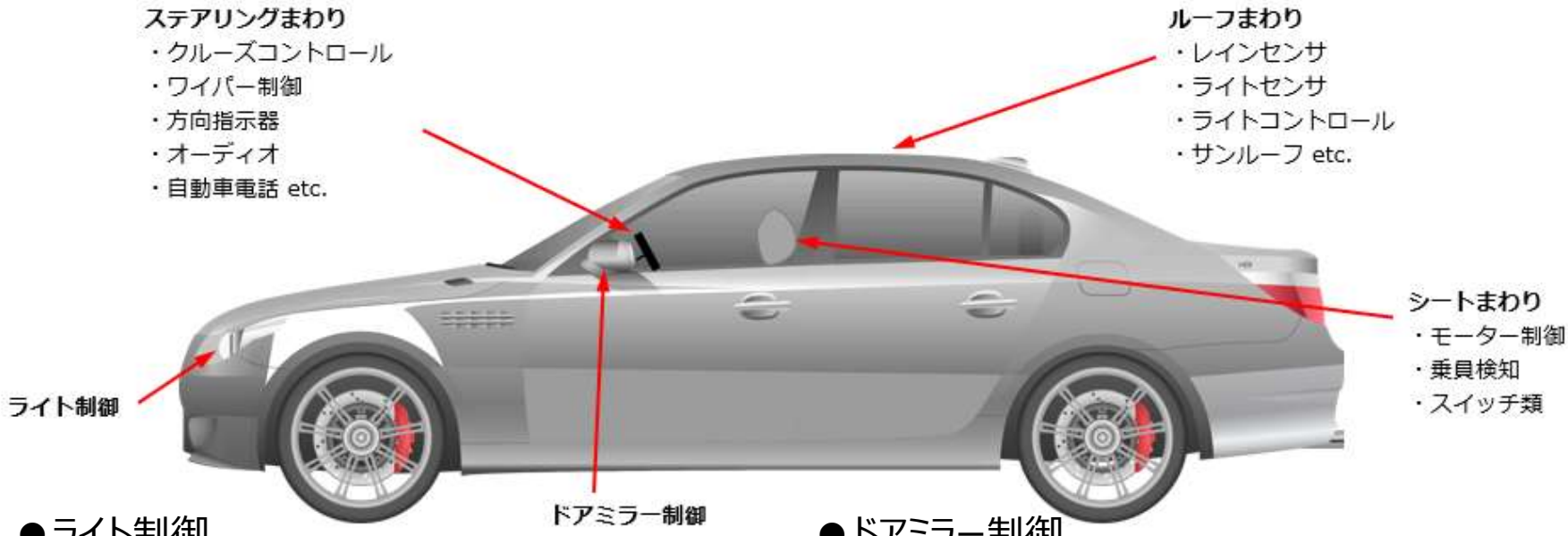
他の車載通信プロトコルと LINの位置付け

- LINはCANやFlexRayに比べて通信速度が遅く**開発コストが抑えられる**
- CANやFlexRayほどの通信速度や情報量を必要としないネットワークで使用されている

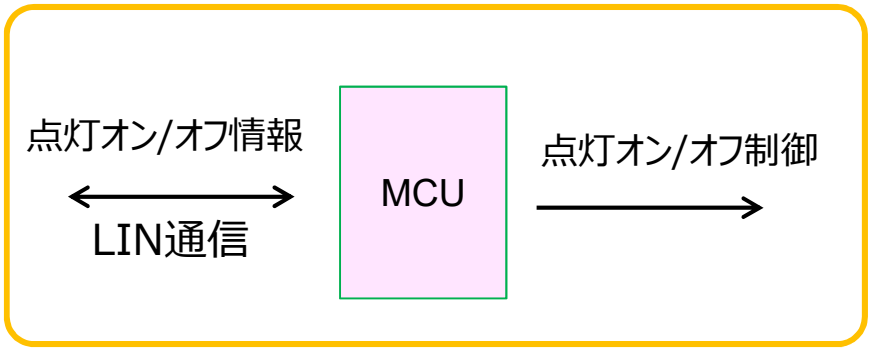


LINとは

LIN応用例（車両イメージ）

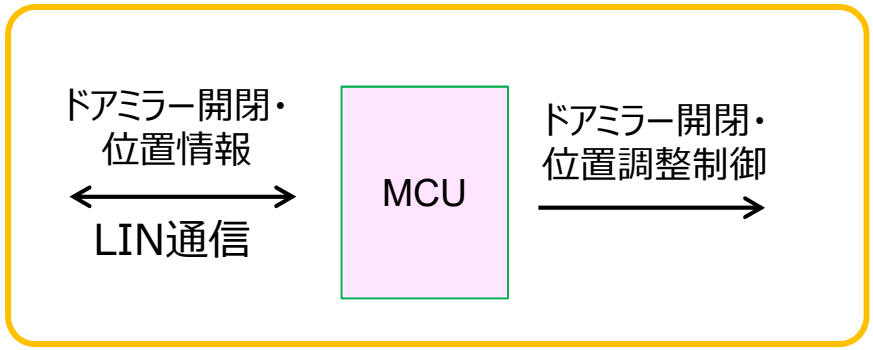


●ライト制御



ヘッドライトの点灯、ルームランプ点灯、
ハザード点滅等

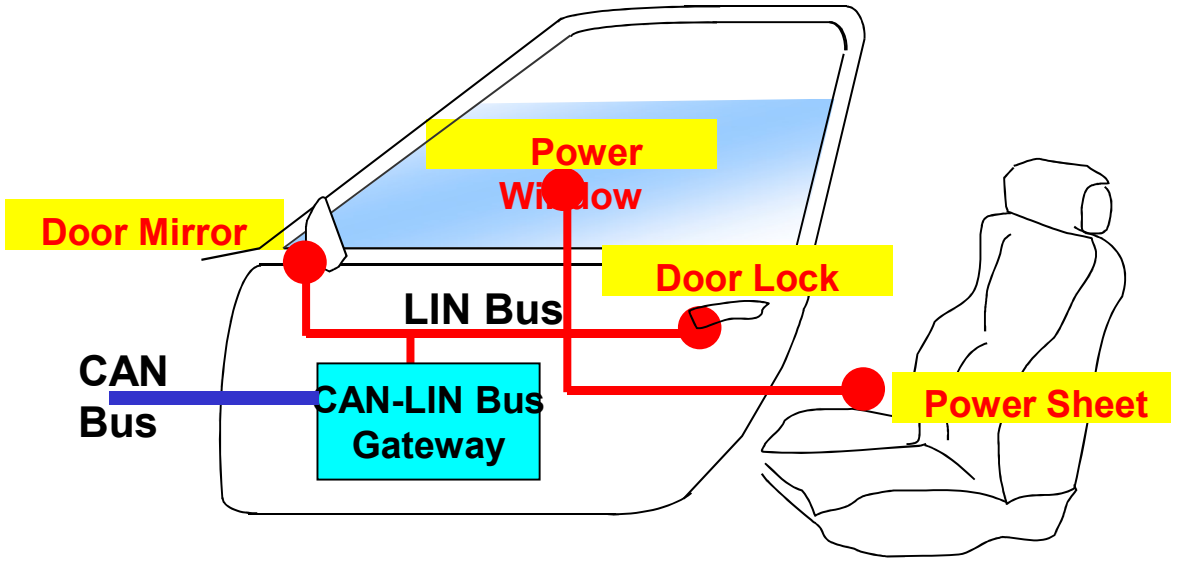
●ドアミラー制御



ドアミラーの開閉、運転者の登録位置に合わせて
ドアミラー位置を調整

LINとは

LIN応用例（ドアまわり）



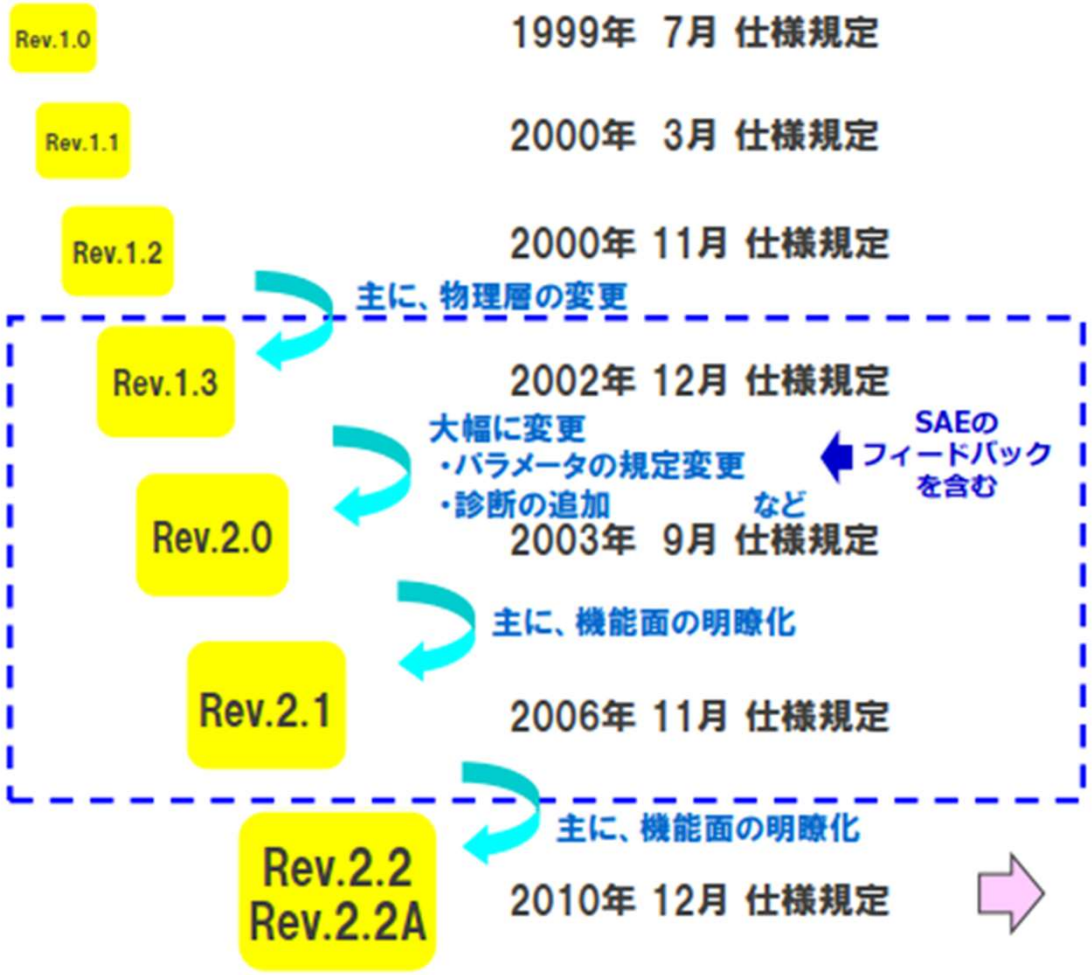
車載LAN構成イメージ図

スライドのみ

LINプロトコルについて

LINプロトコルについて

LINプロトコル(コンソーシアム)の歴史



仕様の改訂が繰り返され、
現在では 主にこの3種類の
Revisionが使用されている

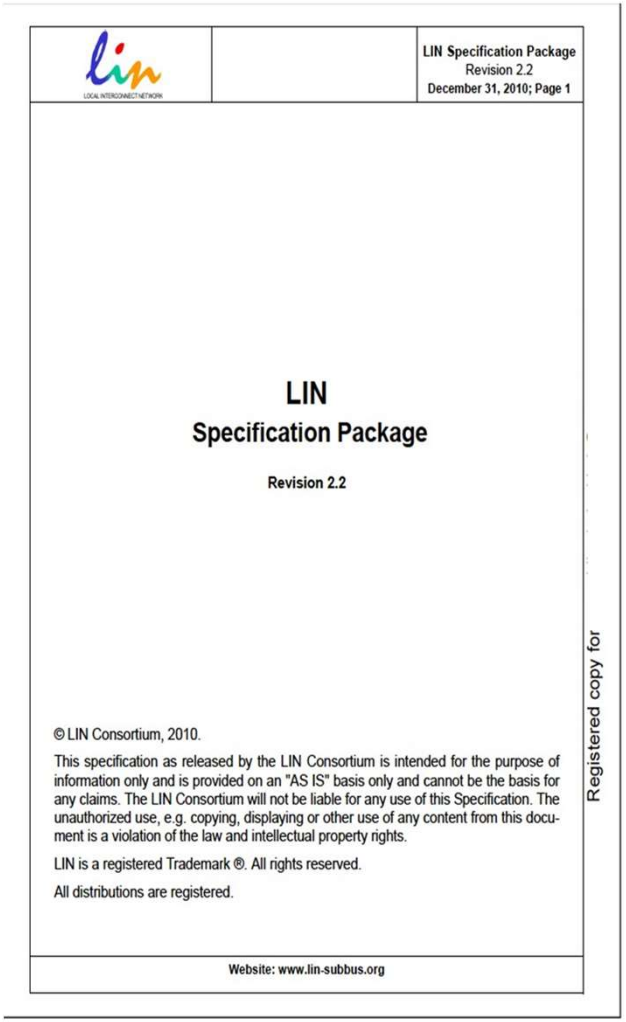
Rev.1.3
Rev.2.0
Rev.2.1

LINコンソーシアムとしての改訂は行わず、ISOに移管。
2016年8月に『ISO17987』発行

LINプロトコルについて

No.ZA21FT99-002

仕様書



LIN Specification Package
Revision 2.2
December 31, 2010; Page 1

LIN
Specification Package
Revision 2.2

Website: www.lin-subbus.org

INTERNATIONAL STANDARD
ISO 17987-1

First edition
2016-08-15

Road vehicles — Local Interconnect Network (LIN) —
Part 1:
General information and use case definition

Véhicules routiers — Réseau Internet local (LIN) —
Partie 1: Information générale et définition des cas d'usage

Reference number
ISO 17987-1:2016(E)

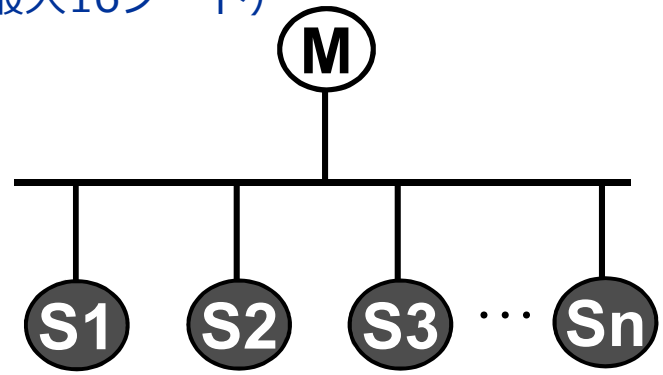
ISO17987では、以下のパートが発行済み。

Part 1 :
General information and use case definition
Part 2 :
Transport protocol and network layer services
Part 3 :
Protocol specification
Part 4 :
Electrical physical layer (EPL) specification
12 V/24 V
Part 5 :
Application programmers interface (API)
<TECHNICAL REPORT>
Part 6 :
Protocol conformance test specification

今回は通信の基本となる
Protocol Specificationの
内容をメインに説明します

LINプロトコルの主な特徴

- **ネットワーク構成**
  シングルマスタ方式 （アービトレーション動作は禁止）
  ※ 1つのマスタと複数のスレーブで構成される（推奨最大16ノード）
- **伝送路 （ISO9141 準拠）**
  シングルワイヤ方式を採用
- **通信コントローラ**
  UART（Universal Asynchronous Receiver/Transmitter）
  (半2重方式、8bit char、1stop bit、Parityなし)
- **発振子精度(最大許容誤差)**
  マスタノード： ±0.5%
  スレーブノード：
  [同期補正あり] ±15%(Rev.1.3)
  ± 14%(Rev.2.0、2.1、2.2)
  [同期補正なし] ± 1.5%
- **スリープ/ウェイクアップ機能サポート**
- **通信速度 最大 20Kbps まで**
- **同期方式 フレーム毎に補正**

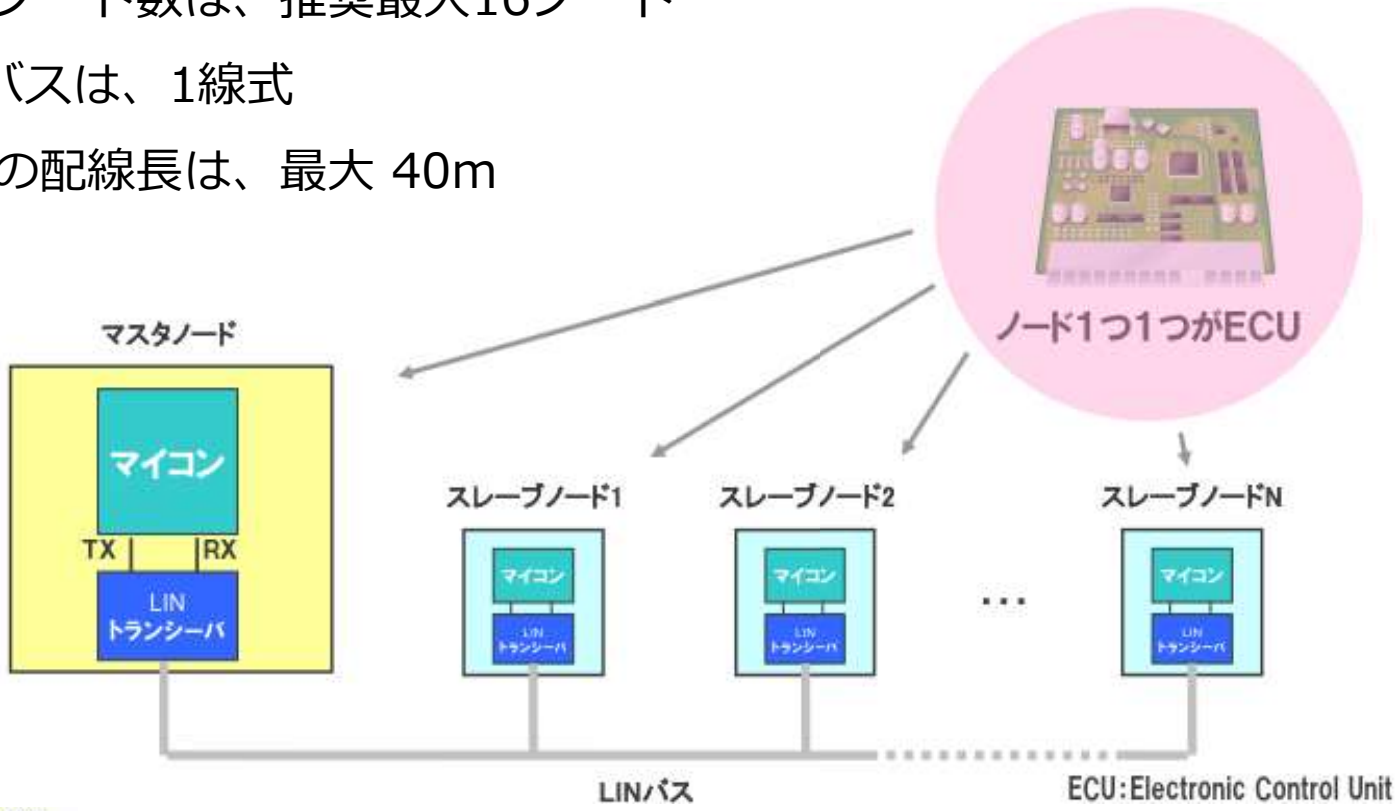


LINプロトコルの詳細

LINプロトコルの詳細

ネットワーク構成

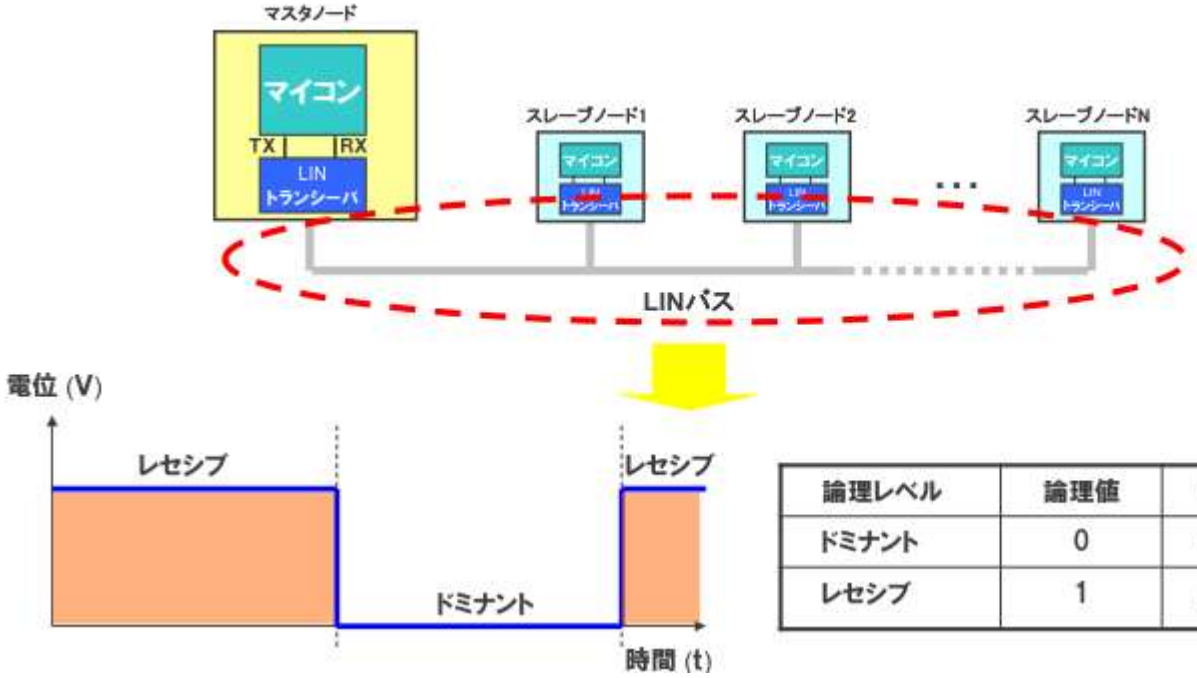
- 1つのマスタ と 複数のスレーブで構成(シングルマスタ方式)される
- バスを介してネットワークを構成する個々を**ノード**と呼ぶ
  - 接続ノード数は、推奨最大16ノード
  - LINバスは、1線式
  - バスの配線長は、最大 40m



LINプロトコルの詳細

LINバス

- LINバスには、「ドミナント」、「レセシブ」の2つのレベルがある
  - ・バスのアイドル状態は「レセシブ」



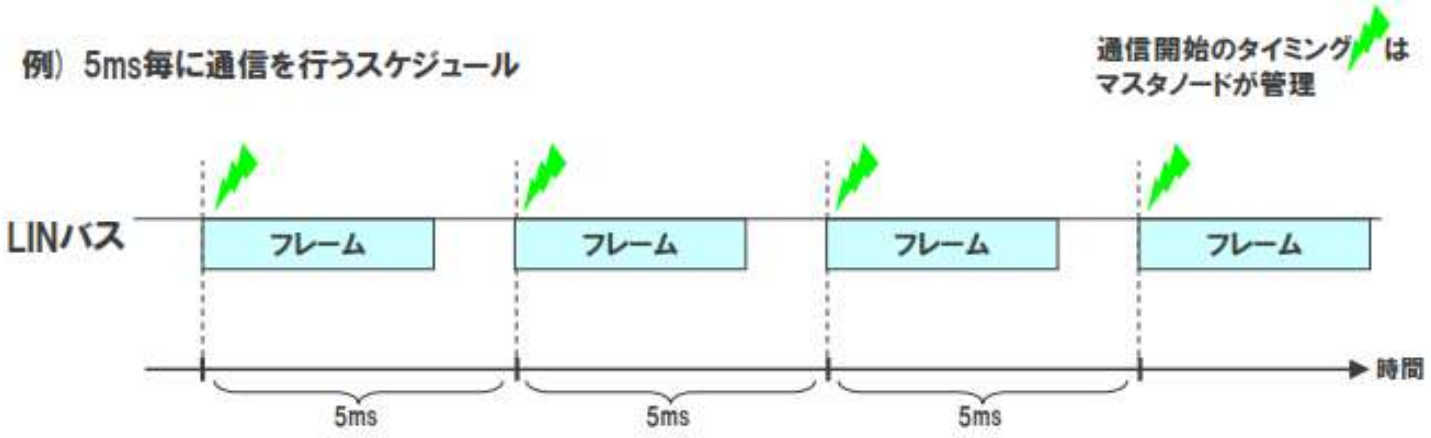
論理レベル	論理値	電圧レベル
ドミナント	0	グランド
レセシブ	1	バッテリ（8～18V）

LINプロトコルの詳細

通信方式

- LINは **スケジュール**に基づいて通信する
  - ・通信スケジュールはマスタノードが管理する

- LINプロトコルで規定する**フレーム**を使用する
  - ・フレームは 必ずマスタノードの送信から始まる



LINプロトコルの詳細

通信速度

- LINプロトコルが規定する通信速度は、1Kbps～20Kbps
  - ・Rev.1.3で推奨する通信速度は 3種類

低速	中速	高速
2.4Kbps	9.6Kbps	19.2Kbps

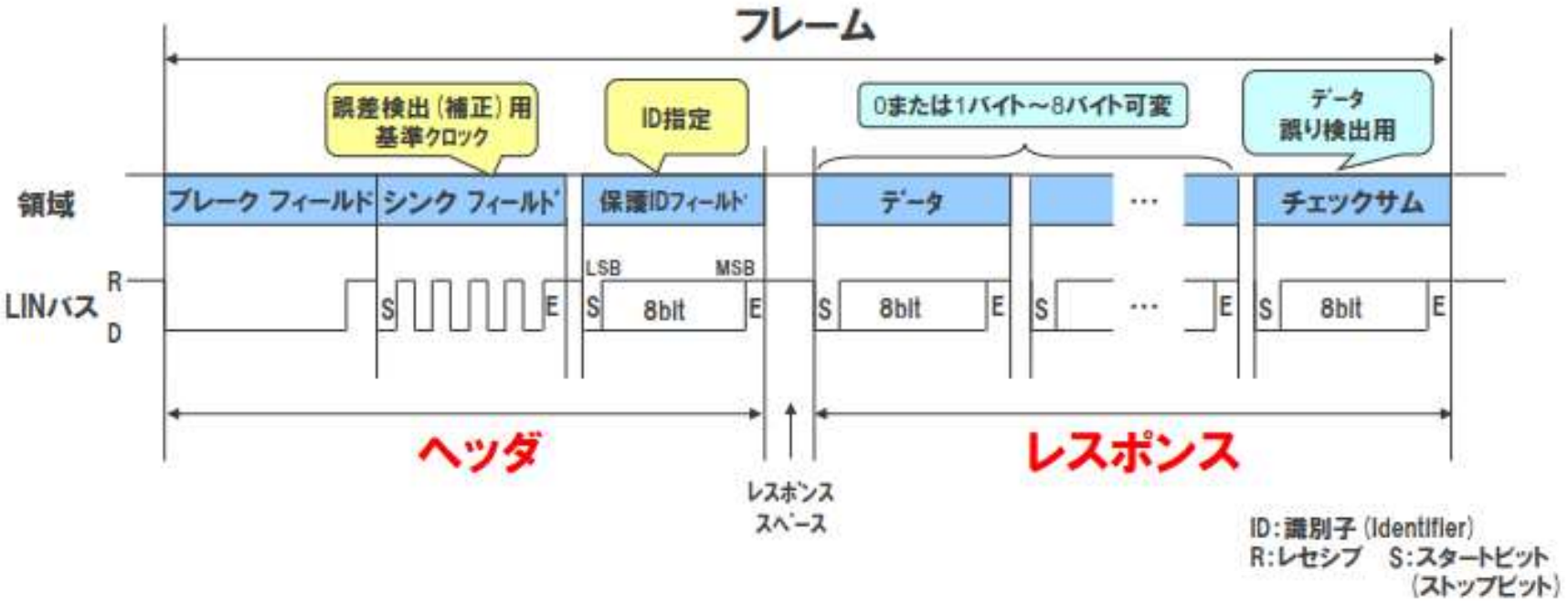
LINプロトコルの詳細

LINフレーム

- LINフレームは、ヘッダとレスポンスで構成される

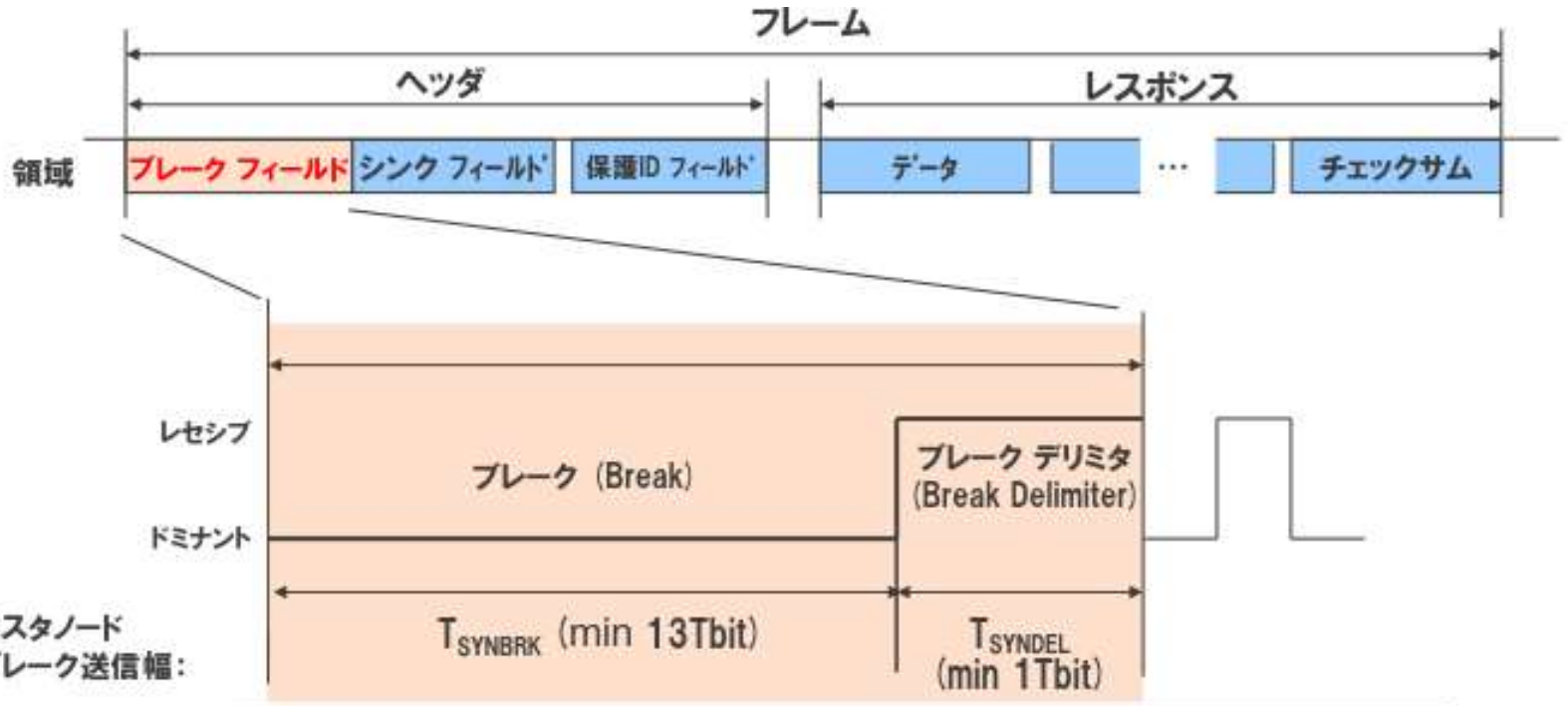
  ヘッダ：マスタノードが送信

  レスポンス：マスタノード または スレーブノードが送信



ブレーク フィールド(Break Field)

- フレームの開始を明示する領域（マスタノードが送信）



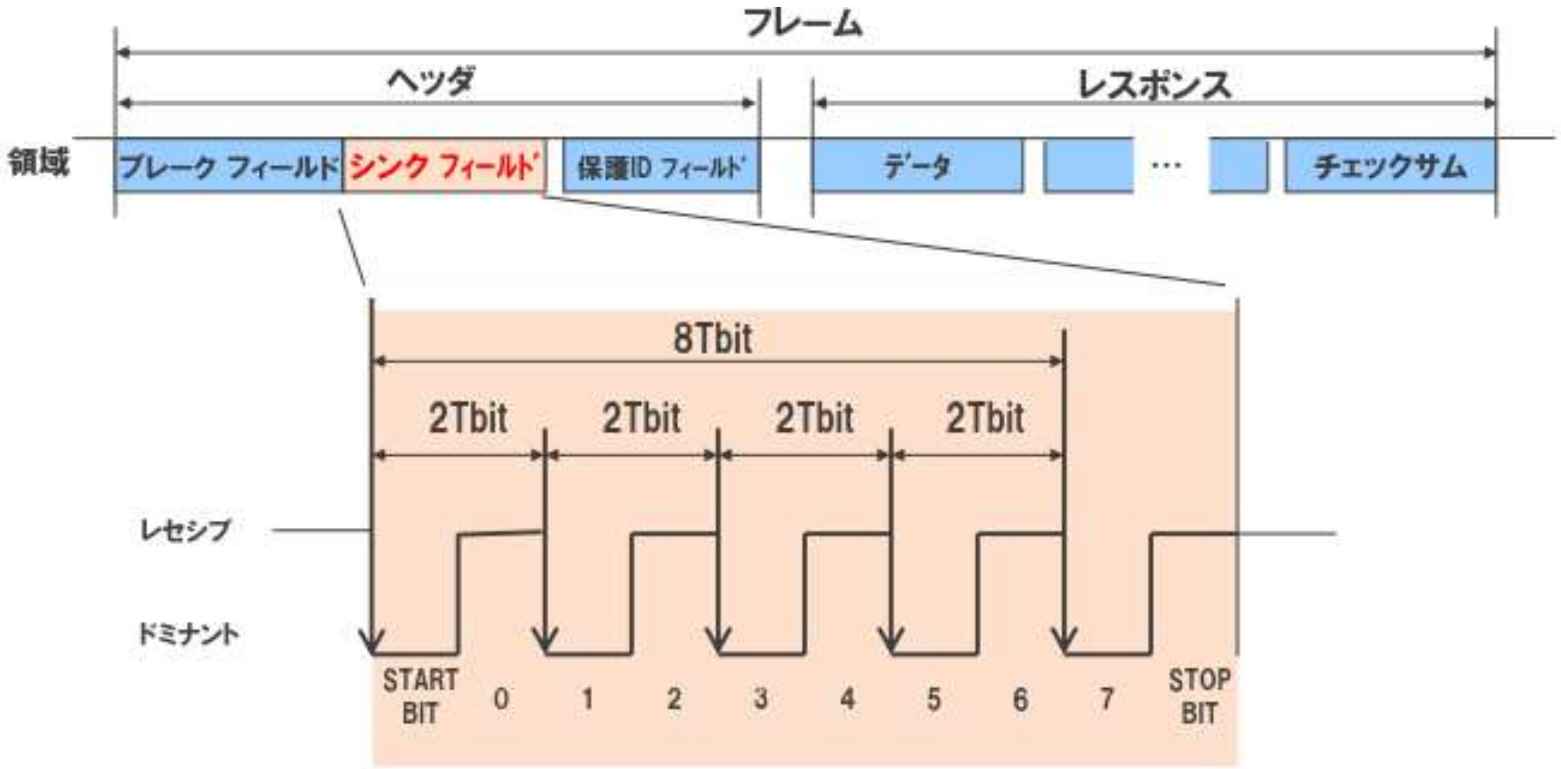
スレーブノード
ブレーク検出幅:

	Revision 1.3	Revision 2.0, 2.1, 2.2, 2.2A, ISO
RC発振 または マイコン内蔵発振回路	11Tbit	
セラミック または 水晶発振	9Tbit	9.5Tbit

*Tbit=Bit time

シンク フィールド(Synch Field)

- スレーブノードが同期合わせをするための基準クロック領域
  （マスタノードが送信）



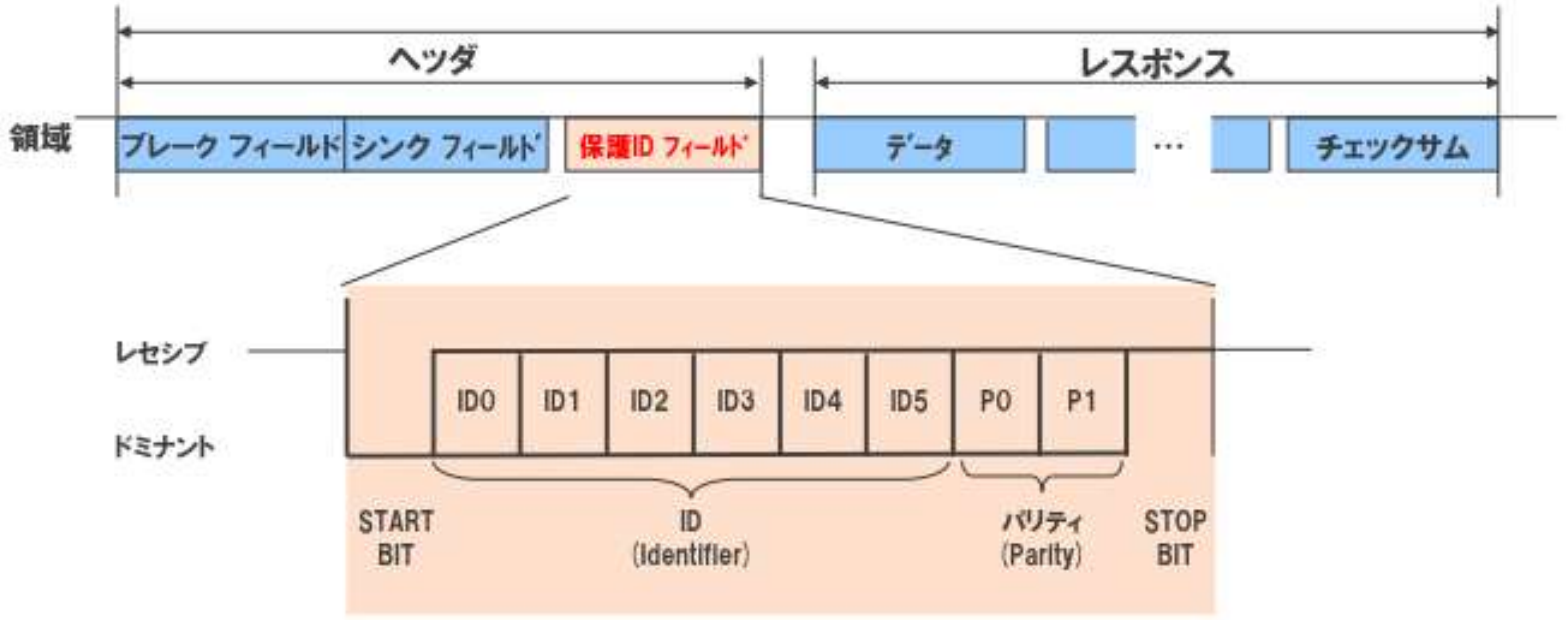
$$1\ \mathrm{Tbit} = \frac{\text{START BIT の立下りエッジから bit7 の立下りエッジまでの時間}}{8}$$

*Tbit=Bit time

LINプロトコルの詳細

保護ID フィールド(Protected Identifier Field)

- 誰がレスポンスを返すのかを指定する領域（マスタノードが送信）



・6ビットの”ID”と2ビットの”パリティ”で構成される

・スレーブノードは、パリティによってIDの正しさを判定し、IDによりレスポンス領域に対して “送信”、“受信” または“何もしない” といった自ノードの動作を判断する

保護 ID (Protected Identifier)について

- **ID(Identifier)**

  ID0～ID5の6ビットにより64(00h～3Fh) 設定可能 (以下IDは条件あり)

  3Ch：コマンドフレームとして使用 (Master Request Frame)
  3Dh：コマンドフレームとして使用 (Slave Response Frame)
  3Eh：ユーザ定義自由の拡張フレーム用 (Rev.2.1ではLIN仕様拡張用)
  3Fh：今後のLIN仕様拡張用として予約

  ※ Rev.1.3では オプション機能として ID4、ID5の
  2ビットでデータ長の指定も可能
  ⇒ Rev.1.1との互換のための仕様

ID5	ID4	データバイト数
0	0	2
0	1	2
1	0	4
1	1	8

- **パリティ(Parity)**

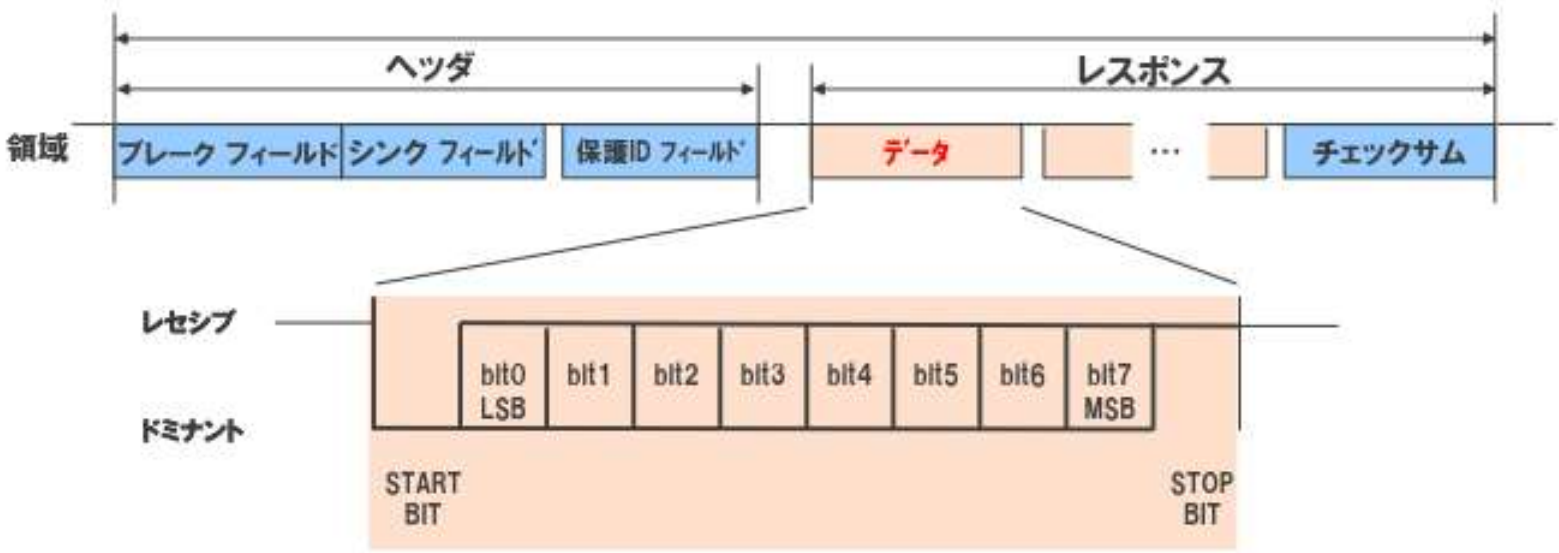
  $P0 = ID0 \oplus ID1 \oplus ID2 \oplus ID4$
  $P1 = \neg(ID1 \oplus ID3 \oplus ID4 \oplus ID5)$

  ※ パリティにより保護IDフィールドの8ビット全てが
  ドミナント または レセシブになることはない

LINプロトコルの詳細

データ

- レスポンスデータ領域（マスタノード または スレーブノードが送信）

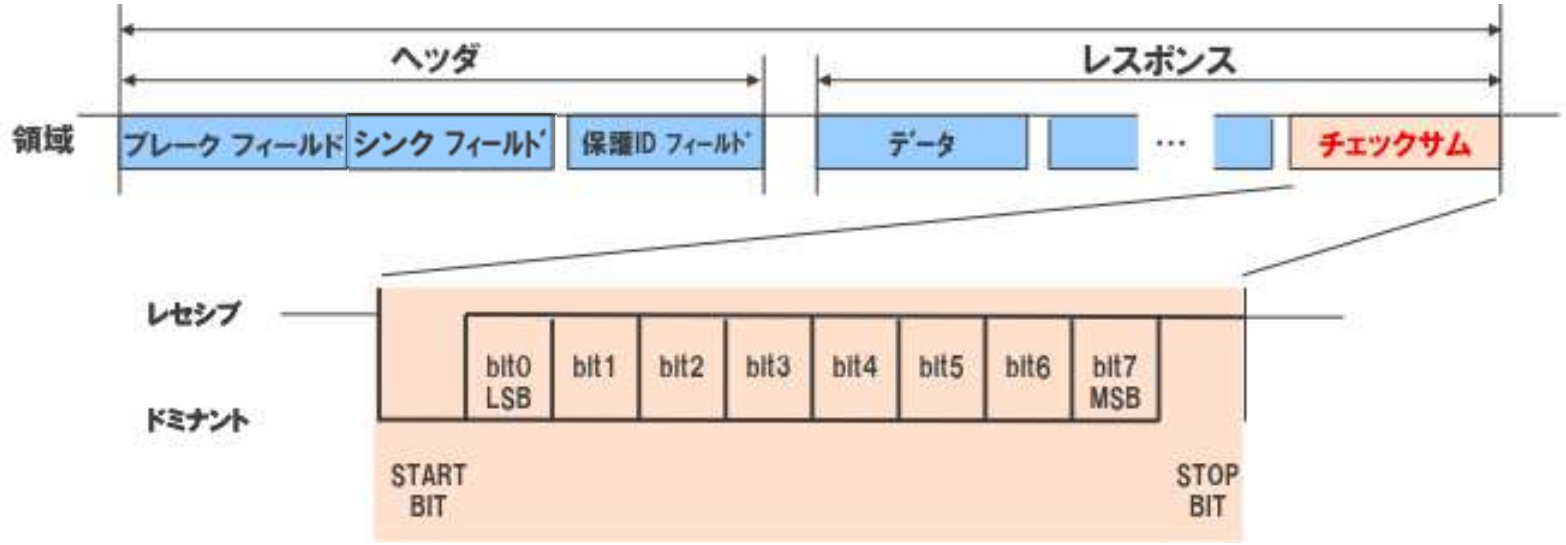


※ データサイズは、フレーム毎に可変(最大8バイト)

LINプロトコルの詳細

チェックサム

- レスポンスデータの誤り検出用の領域
  （マスタノード または スレーブノードが送信）



計算方式 ： Modulo-256方式 (キャリー付き加算)

チェックサム方式 ：

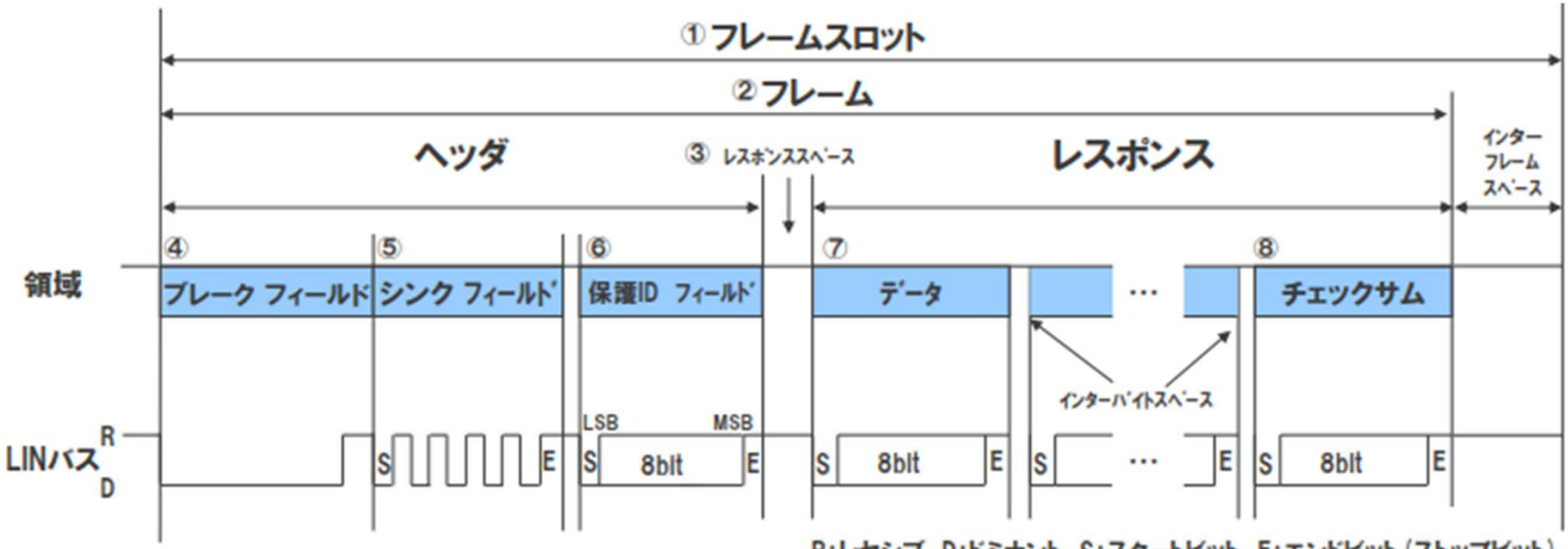
クラシック(classic) – チェックサム対象が”全データ”。Rev.1.3では本方式のみ

エンハンス(enhanced) – チェックサム対象が “保護ID＋全データ” 。Rev.2.0で追加

チェック方法：「全データ」または 「保護拡張子＋全データ」 + チェックサム領域の値 = FFh

LINプロトコルの詳細

Revisionによる名称の違い



R:レセシブ、D:ドミナント、S:スタートビット、E:エンドビット(ストップビット)

Revision	①	②	③	④	⑤	⑥	⑦	⑧
1.3	ー	メッセージフレーム	インフレーム-レスポンススペース	シンク ブレークフィールド	シンクフィールド	ID フィールド	データフィールド	チェックサムフィールド
2.0、2.1、2.2、2.2A、ISO	フレームスロット	フレーム	レスポンススペース	ブレークフィールド	シンク(バイト)フィールド	保護IDフィールド	データ	チェックサム

LINプロトコルの詳細

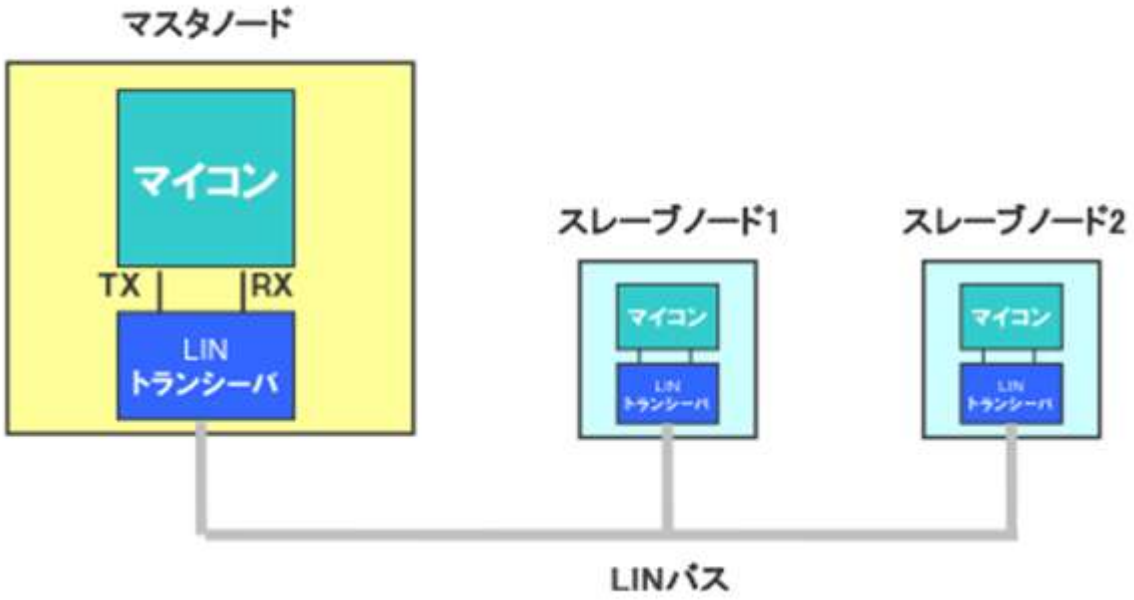
LIN通信のイメージ（例）

・ID=01h：スレーブノード1がレスポンスを送信する

・ID=02h：スレーブノード2がレスポンスを送信する

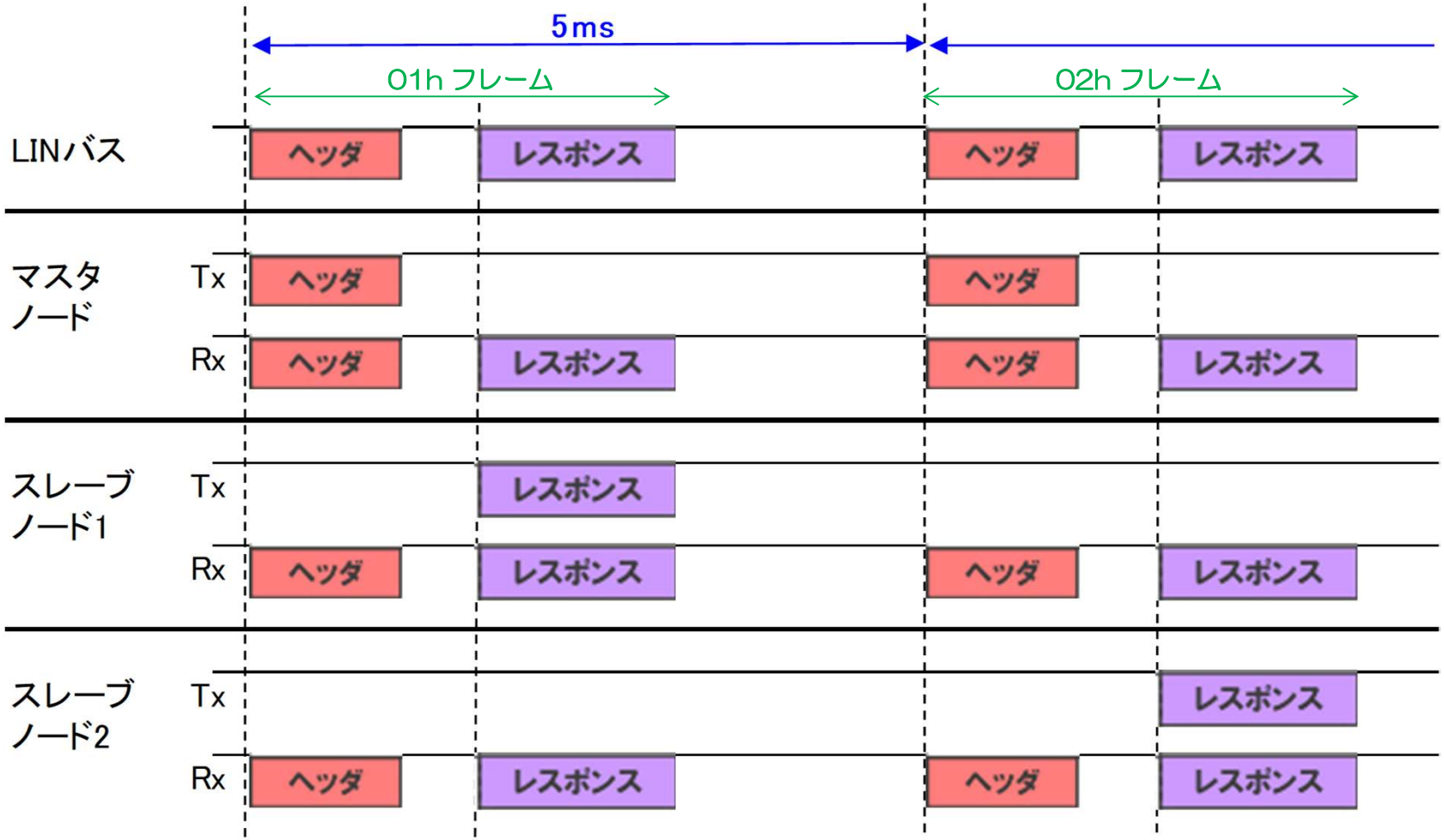
・マスタは 5msごと に以下のフレームヘッダを送信する

スケジュール：ID=01h → ID=02h → ID=01h → ID=02h → …



LINプロトコルの詳細

先程の動作をLINバス、各ノードのTx(送信端子)、Rx(受信端子)で確認すると…



スリープ/ウェイクアップ

- システムの消費電力を減らすために、LINではスリープをサポートしている

スリープ遷移条件

マスタノード　：自ノードによる判断

スレーブノード：下記いずれか
- マスタからのスリープ要求(go-to-sleep-command)あり
- LINバスに一定時間(※)以上 通信がなかった[Rev.2.0で追加]

※ 一定時間：Rev.2.0では4s～、Rev.2.1以降では4～10s

LINプロトコルの詳細

スリープ復帰(ウェイクアップ)条件

マスタノード　：下記いずれか
- ・ウェイクアップシグナルの受信
- ・自ノード内のLIN以外の外部要因

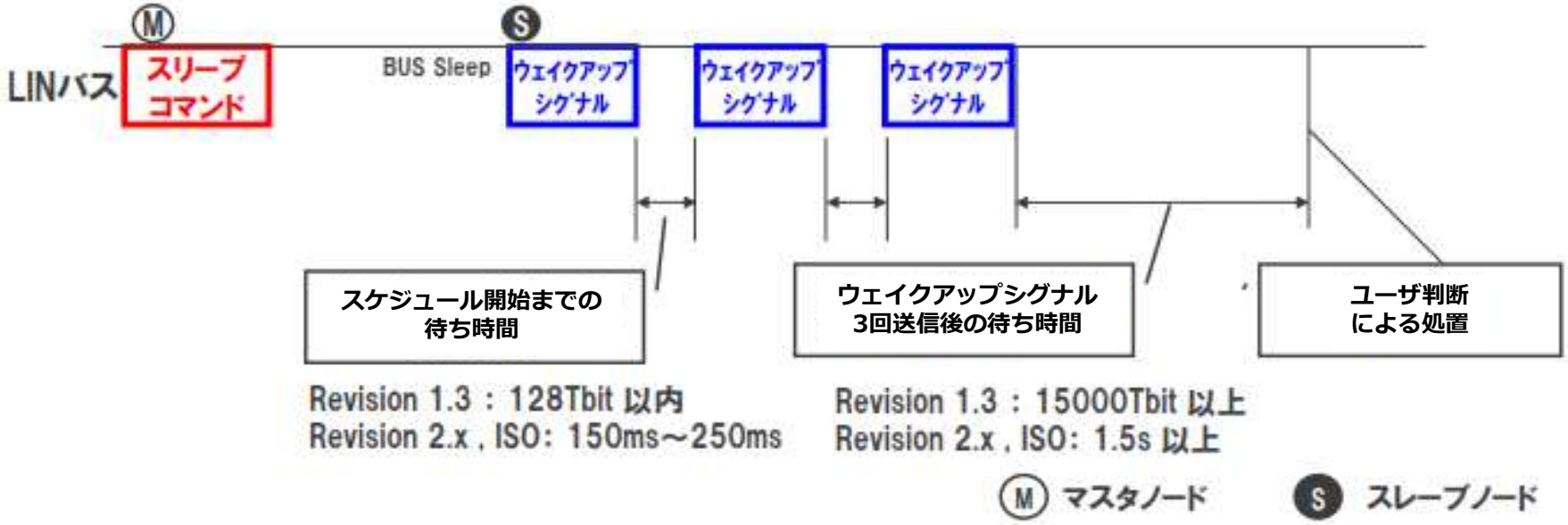
スレーブノード：下記いずれか
- ・ウェイクアップシグナルの受信
- ・自ノード内のLIN以外の外部要因

※ 受信側は 150μs 以上の連続するドミナントをウェイクアップシグナルとして判定
(Rev.2.0以降で規定あり)

スリープ/ウェイクアップの動作

- **通常動作時**

- **異常動作時(マスタノードがウェイクアップしない)**



LINプロトコルの詳細

エラー

- LINでは バスの異常やノードの故障を検出するために6種類のエラーを規定している。

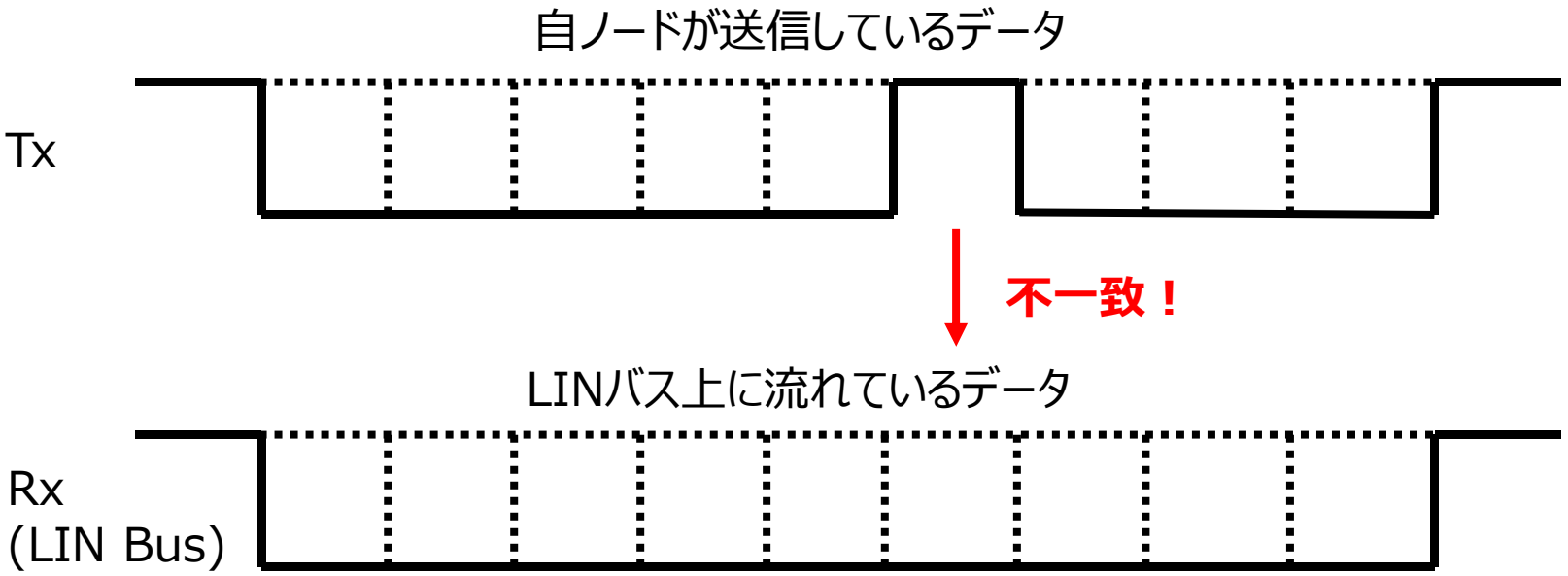
  - **ビットエラー (Bit-Error)**
  - **チェックサムエラー (Checksum-Error)**
  - **シンクフィールドエラー (Inconsistent-Synch-Field-Error)**
  - **IDパリティエラー (Identifier-Parity-Error)**
  - **ノーレスポンスエラー (Slave-Not-Responding-Error)**
  - **フィジカルバスエラー (Physical-Bus-Error)**

具体的にエラーの規定をしているのは、Rev.1.3のみ

LINプロトコルの詳細

ビットエラー(Bit-Error)

検出ノード：マスタノード／スレーブノード
検出条件：「自分が送信したデータ」と「LINバス上のデータ」が異なる

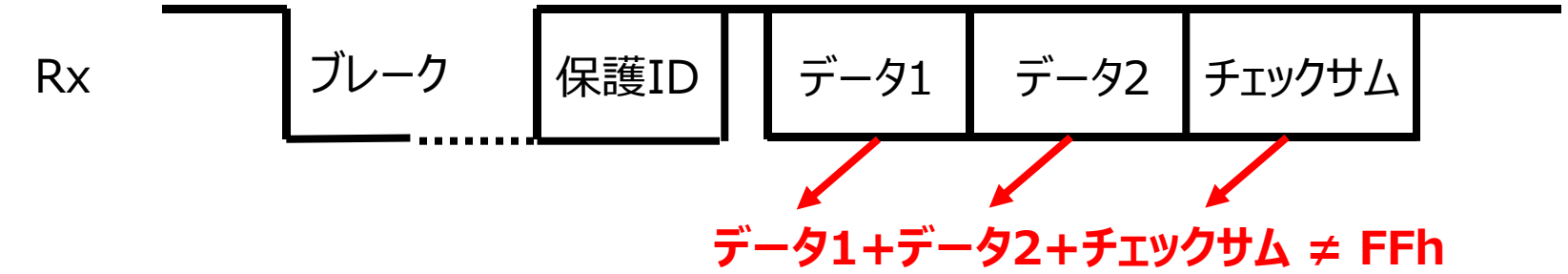


※ ビットエラーを検出すると、検出ノードは以降の送信を中断する

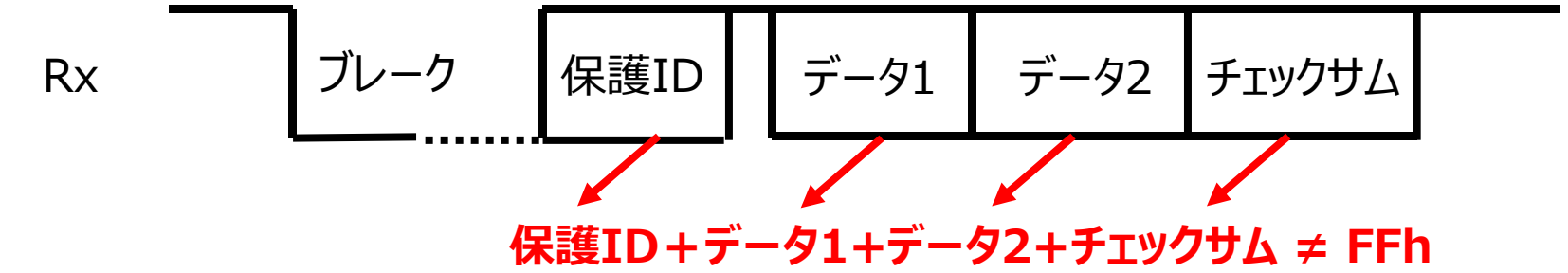
チェックサムエラー(Checksum-Error)

検出ノード：マスタノード／スレーブノード
検出条件：受信したデータとチェックサムを足した結果がFFhでない。

[クラシック(classic)の場合]



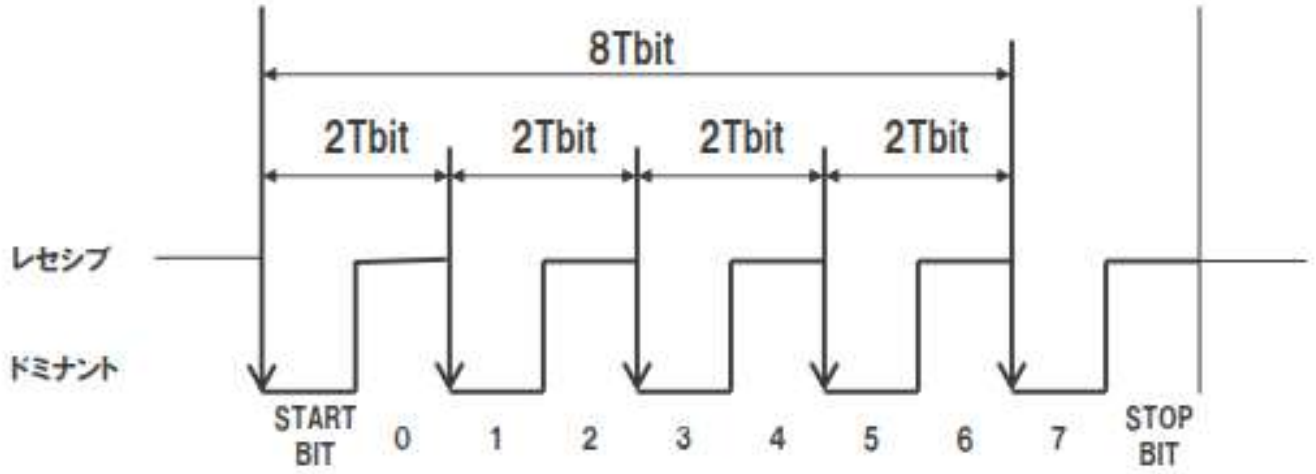
[エンハンス(enhanced)の場合]



LINプロトコルの詳細

シンクフィールドエラー(Inconsistent-Synch-Field-Error)

検出ノード：スレーブノード
検出条件：シンクフィールドから算出した通信速度が許容以上ずれている



$$1\ \text{Tbit} = \frac{\text{START BIT の立下りエッジから bit7 の立下りエッジまでの時間}}{8}$$

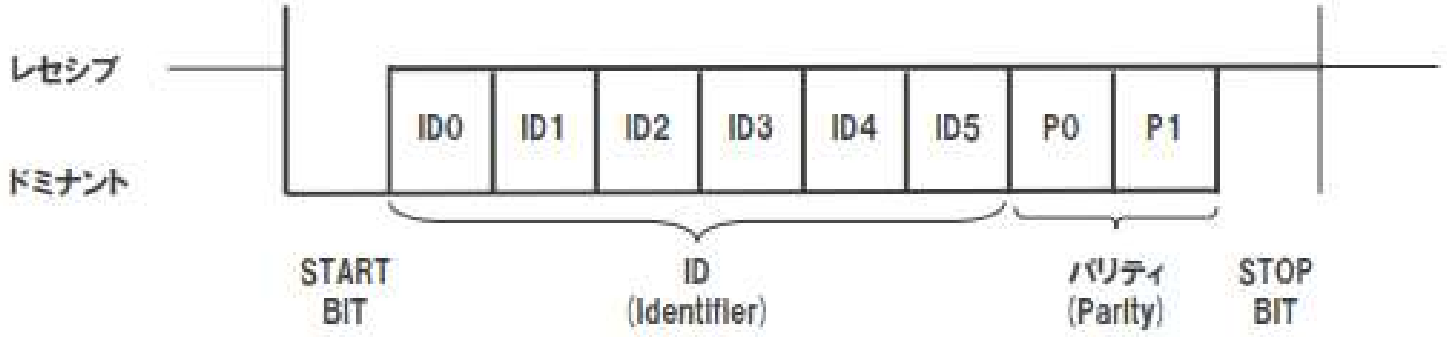
通信速度の 1Tbit と計測結果との差が
Rev.1.3で±15%、Rev.2.x、ISOで ±14% を超えた場合にエラーと判定

*Tbit=Bit time

LINプロトコルの詳細

IDパリティエラー(Identifier-Parity-Error)

検出ノード：スレーブノード
検出条件：受信したパリティ値と算出したパリティ値が異なる



受信した P0 ≠ ID0 ⊕ ID1 ⊕ ID2 ⊕ ID4
受信した P1 ≠ ¬(ID1 ⊕ ID3 ⊕ ID4 ⊕ ID5)

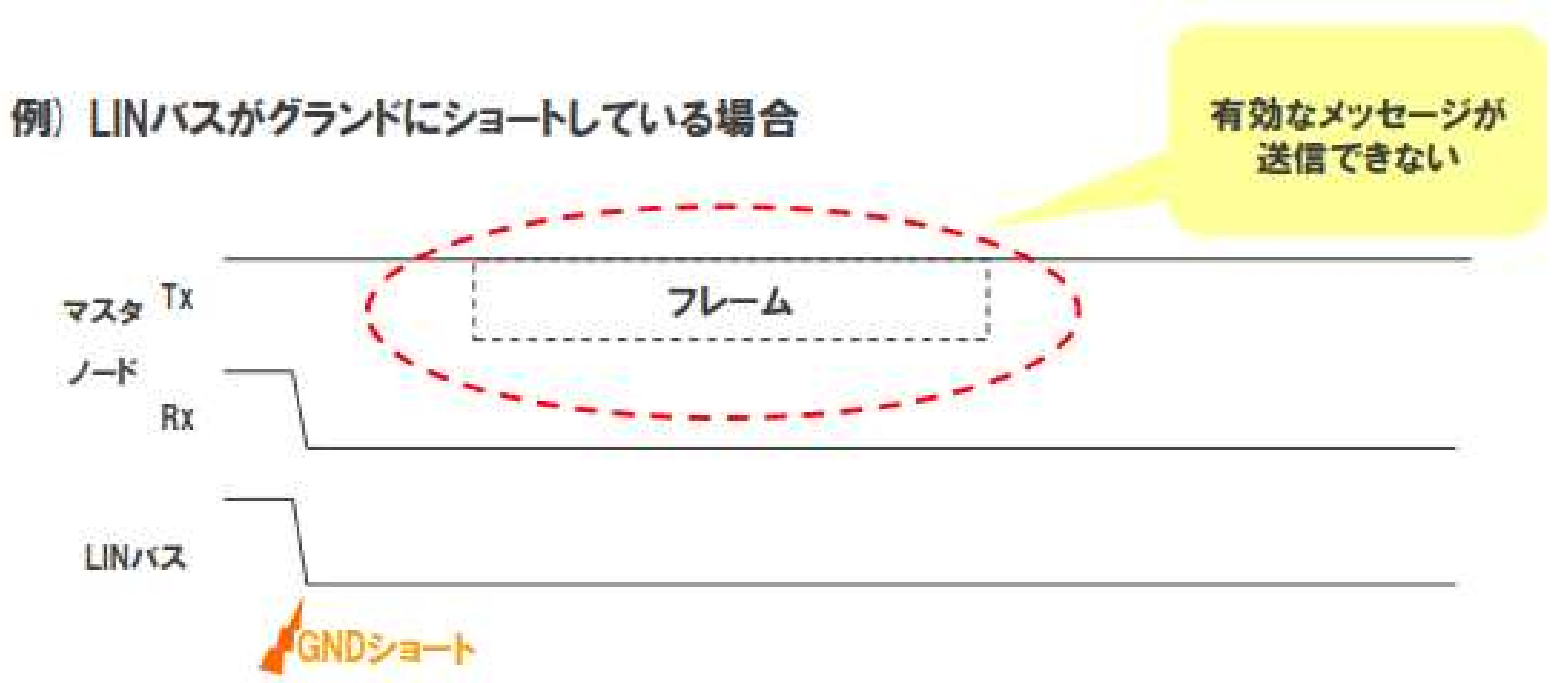
LINプロトコルの詳細

ノーレスポンスエラー(Slave-Not-Responding-Error)

検出ノード：マスタノード／スレーブノード
検出条件：フレーム開始から規定時間内にレスポンスが完了しない

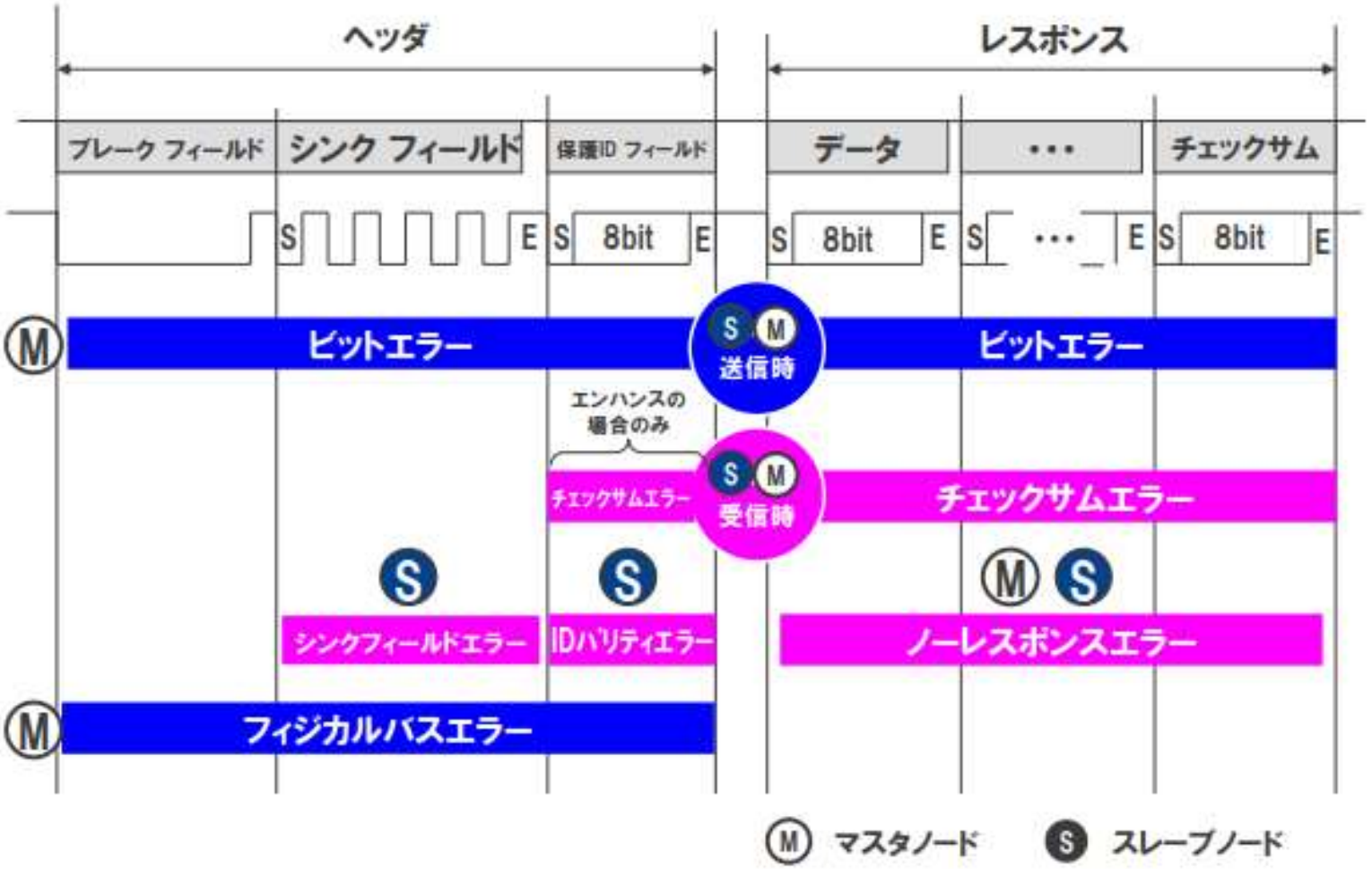
フィジカルバスエラー(Physical-Bus-Error)

検出ノード：マスタノード
検出条件：LINバス上に有効なメッセージを送信できない
（例：LINバスがグランド 又は バッテリー(V_{BAT})にショートしている場合）



LINプロトコルの詳細

エラー検出領域

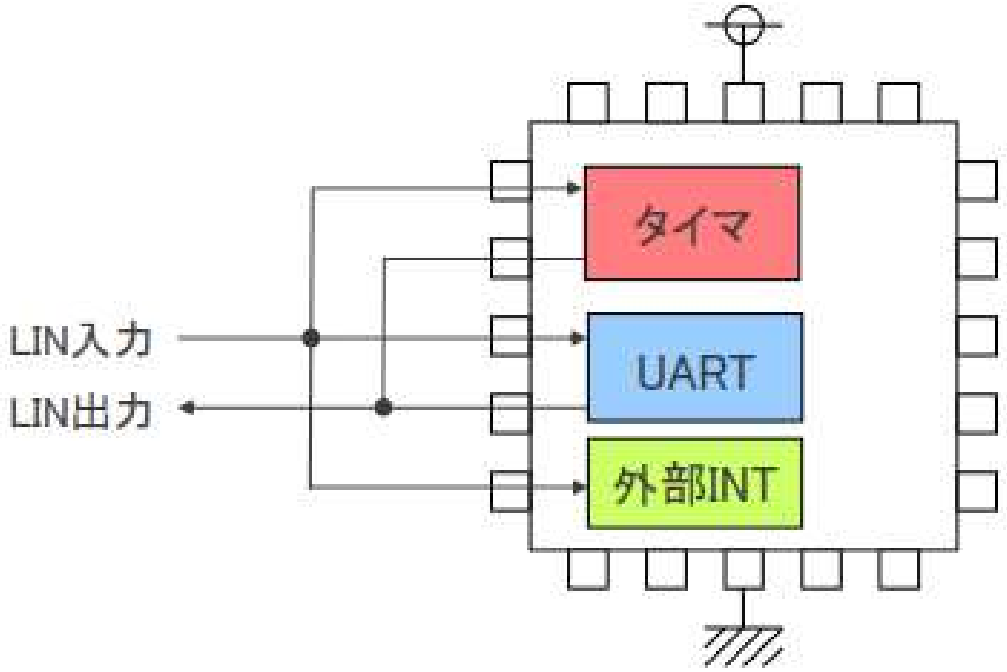


LINの実現方法

LINの実現方法

従来のソフトウェアによるLIN構築例

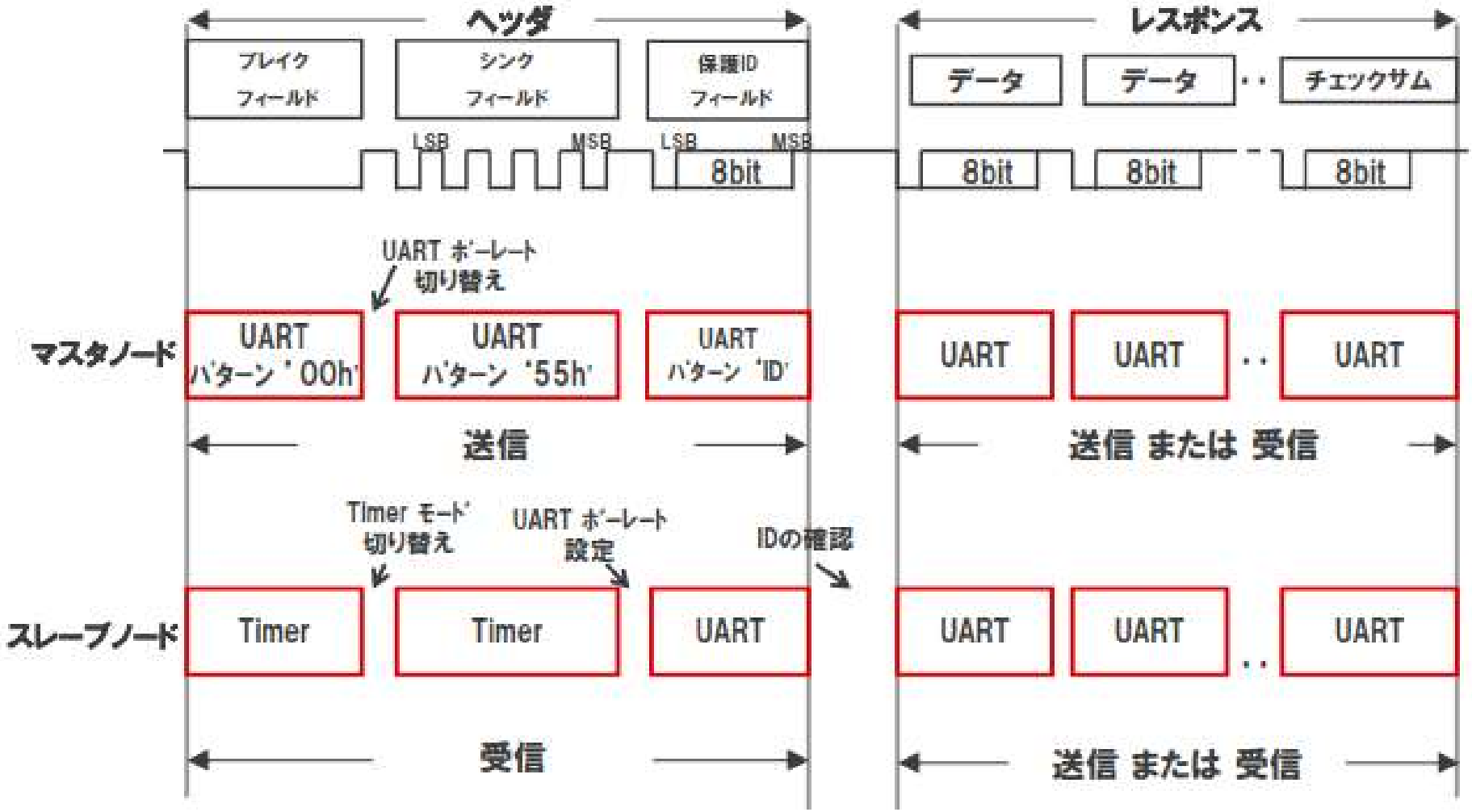
- マイコンの既存リソースであるUART・タイマ・外部INTを使用して、ソフトウェアによりLIN通信を実現



※ 品種によりタイマ・UART・外部INT構成が違うため、構成は上記の限りではない
外付け結線が不要なマイコンもある

LINの実現方法

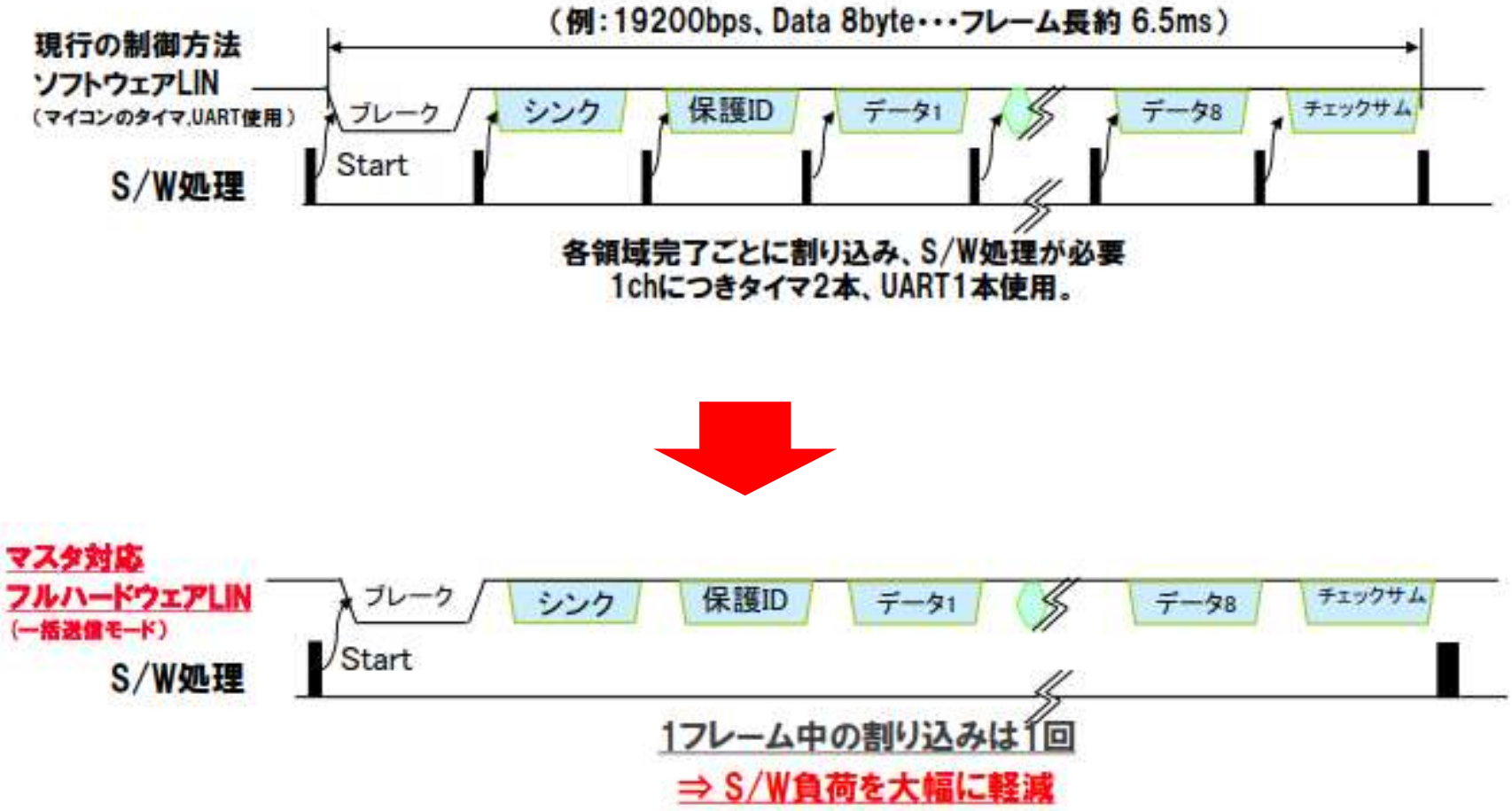
制御例



LINの実現方法

マスタ対応フルハードウェア

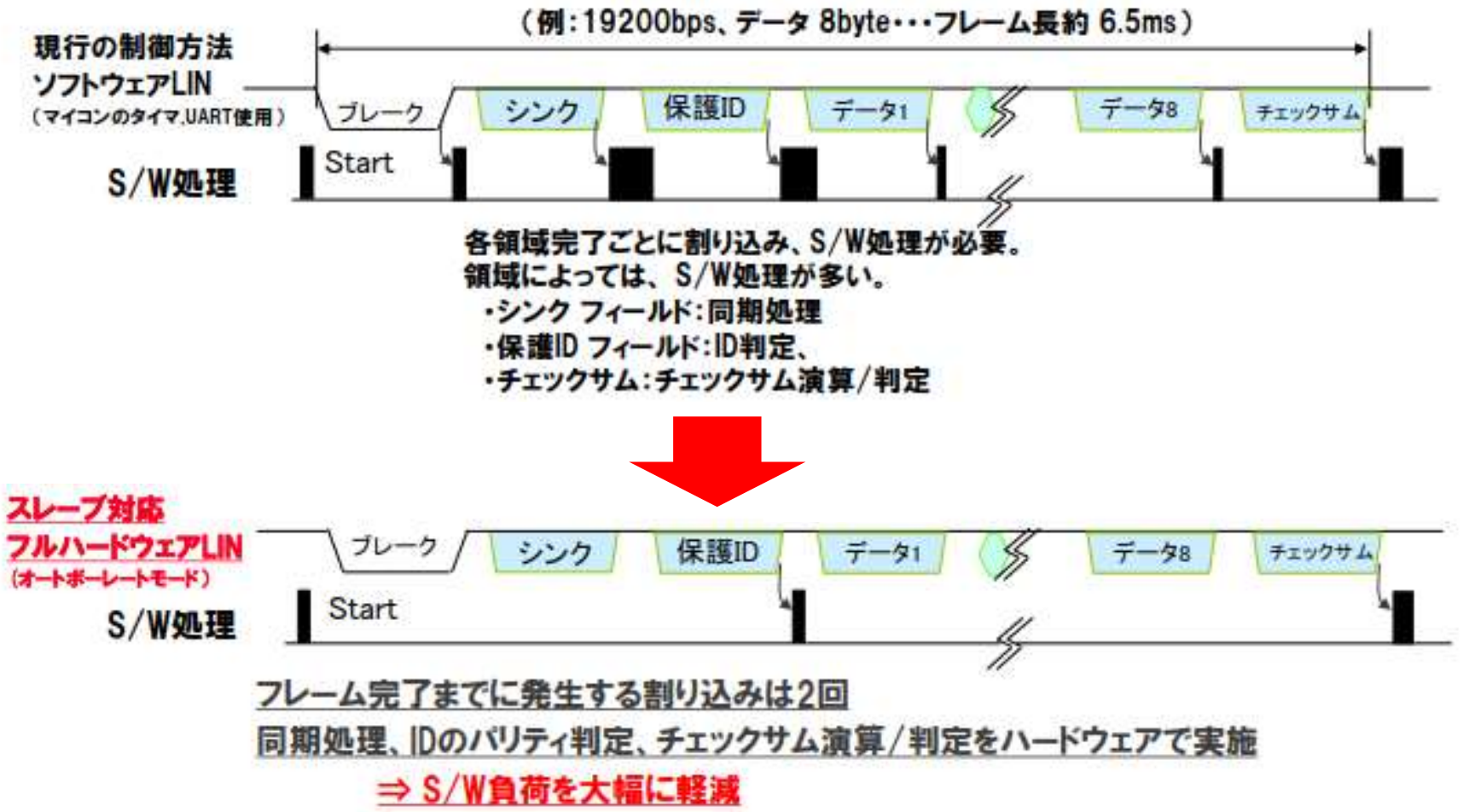
- マスタの1フレーム通信を完全自動化



LINの実現方法

スレーブ対応フルハードウェア

- ヘッダ受信、レスポンスの一括送信/受信を自動化



実際に使用する時の注意点

実際に使用する時の注意点

No.ZA21FT99-002

- Rev.の違い
- ダイアグ通信
- 電源投入時のレスポンス

【ネットワーク設計時】

- 推奨最大ノード数は16だが実用的には３または４ノード

ご清聴ありがとうございました