

# やさしい CAN でござーる

## カーエレクトロニクスの進歩と多重通信

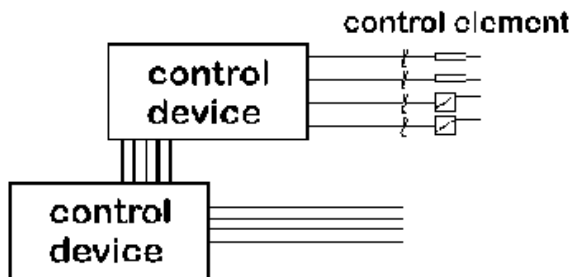
1970年代後半から1980年代にかけて登場した自動車に対する排出ガス規制と燃費規制は自動車のエンジンをそれまでの機械的メカニズム主体による制御から電子式制御へと一変させました。すなわち、最適な点火時期制御、空気と燃料の混合比の精密な制御を実現するためにマイクロコンピュータを使ったデジタル式コントローラ (ECU)によるエンジン制御を導入したのです。

ECUによる高機能かつ高精度な制御はそのための多くの情報を必要としたため、既存の機器情報や新たに設けられたセンサーからの情報を電気信号としてECUに伝送したり逆にECUからの制御指令を電気信号として各機器に伝送したりするための電線(ワイヤーハーネス)が多数必要になりました。

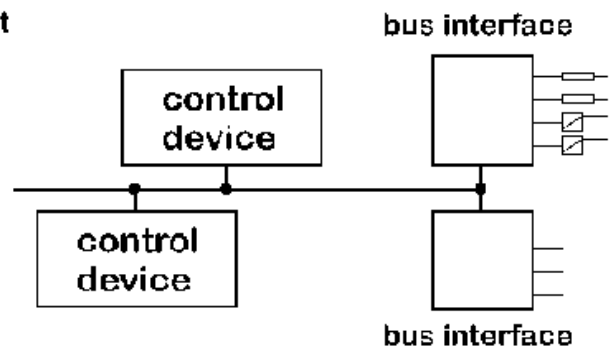
マイクロコンピュータの発達と普及はエンジンのみならず自動車のあらゆる部分での制御を高度化、高機能化へと加速させ、安全性、快適性の向上を実現するためのABSやエアバック、オートエアコン、クルーズコントロールなど、様々な装備が採用されるようになりました。そしてこれらの装備はそれぞれがECUを持ち、前述と同じく多数のワイヤーハーネスが必要となり、他の既存電装品と合わせて電線本数の増加、重量増、コスト増大へとつながっています。

そこで自動車メーカー各社はこれらのECUをバス結合し情報を共有したり、集約したりすることや、他の電装品にも積極的にECUを採用しバス結合することで省配線を実現すると同時に協調動作や故障診断といったさらなる機能アップおよび信頼性向上を計るため、多重通信による車内ネットワークの採用を始めました。

## 従来の結線



## バス結線



最近では、カーナビゲーションに代表されるメディア情報機器、道路交通情報通信システム VICS (Vehicle Information and Communication System) やノンストップ自動料金収受システム ETC (Electronic Toll Collection) といった車外情報機器の発達がめざましく、これらの情報を連携し、さらなる安全性、快適性向上に利用する動きが活発化しており、もはや通信ネットワークなしには対応不可能な状況になりつつあります。

さらに、カーエレクトロニクスは自動車のより重要な制御への関わりが深くなっています。将来、自動操縦などが実現されてくると、スロットル制御やブレーキ制御はもはやペダルと直接メカニカルにリンクするのではなく、DCモータやステップモータを使ったアクチュエータが代行するようになり、その制御システムには航空機なみの信頼性が要求されるようになりますが、その際に必要な Fault Tolerant (故障診断) にも CAN の技術が応用されつつあります。

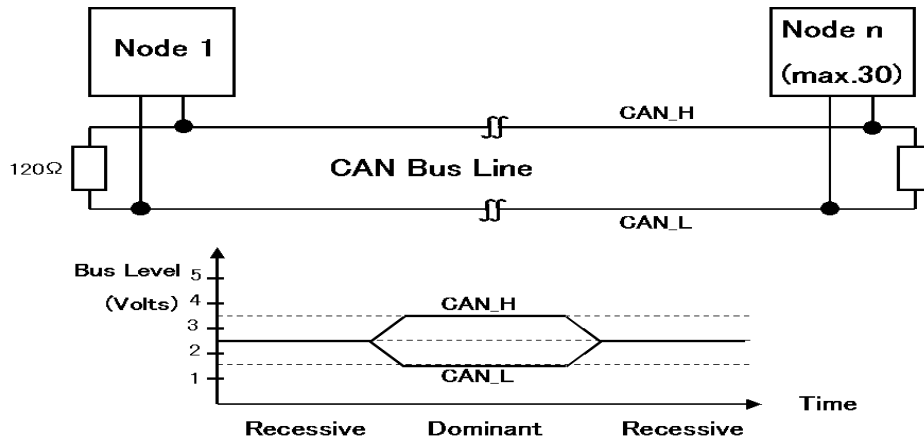
## 車内ネットワークとしての CAN

コンピュータ同士のネットワークとしては最も身近な例ではインターネットがありますし、オフィスにおけるイーサネットや産業分野での各種フィールドバスなどが存在します。簡単に言えば **CAN (Controller Area Network)** は後者のフィールドバスの一つで自動車特有の環境や要求に適合したものとと言えます。

最初に述べたように、車のデジタル制御化はエンジン周辺の制御(点火時期、燃料噴射)から始まりました。一般的なエンジンで最高回転数が 6000rpm とすると、各気筒あたりの燃焼は 1 秒間に 50 回、6 気筒車なら 300

回、つまりわずか数 msec 間隔で燃料噴射、点火という動作を繰り返していますから、これを制御している ECU が必要とする情報はさらに早いスピードで正しくかつ確実に送られてくる必要があります。このため、自動車内ネットワークでは、その通信に非常に高い伝送速度と信頼性、確実性が求められ、さらには徹底したコストメリットまで求められるのです。

CAN はこのことを踏まえ BOSCH 社が提案し、開発した高速の分散型リアルタイム車内情報ネットワークシステムです。それぞれの ECU(ノードと呼ぶ)と実際の CAN バスとの接続例を以下に示します。



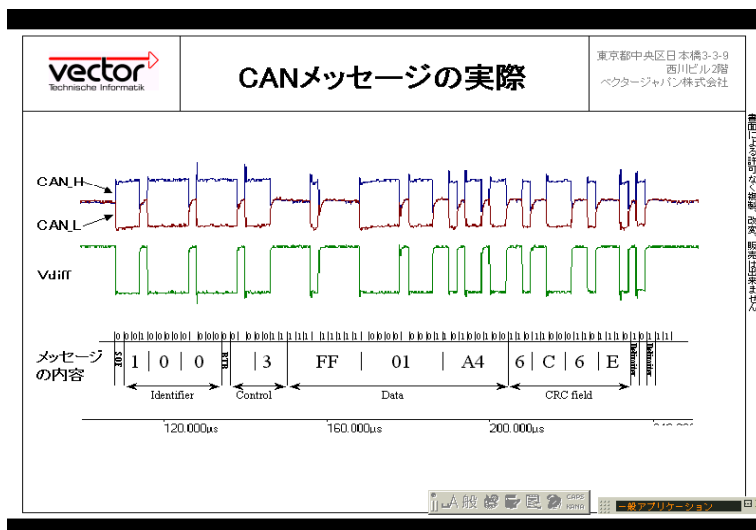
CAN は通信スピード(ボーレート)125Kbps を境に Hi-Speed CAN と Low-Speed CAN の二つにわかれます。上図は Hi-Speed CAN の例で、最大ノード数が 30、必ず終端抵抗が必要です。

(Low-Speed CAN についてはここでは割愛します)

CAN では 2 本のワイヤーの間に発生する差動電圧を信号レベルとして検出する方式を取っており外部からのノイズに影響されにくい特長があります。つまり、ノイズが信号線に侵入しても、H, L 両ライン共電位が上昇し、2 本のワイヤー間の電位差は変化しないのです。

差動電圧なしをデジタル信号の“1”(これをレセッシブという)、差動電圧ありをデジタル信号の“0”(同ドミナント)として規格しています。“0”(ドミナント)は“1”(レセッシブ)を上書きすると覚えてください。

下図は実際の CAN メッセージとその電圧波形です。少し小さくて解り辛いかもしれませんが、差動電圧値  $V_{diff}$  と信号の“0”、“1”とを対応させて見る事ができると思います。



## CAN プロトコルの実際

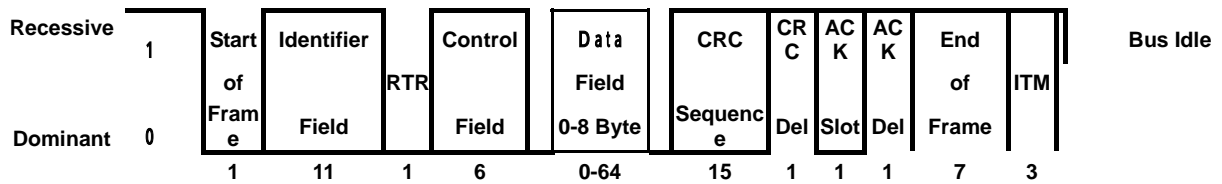
通信されるデジタル信号のまとまりを**フレーム**といいます。

CAN には以下の 4 つのフレームタイプがあります。

1. データ・フレーム : 送信ノードから制御パラメータ等のデータを送信する
2. リモート・フレーム : 他のノードに対してデータの送信を要求する
3. エラー・フレーム : 他のノードに対するエラーの通知をする
4. オーバードロード・フレーム : 受信ノードが受信処理未了時に出力する

## データ・フレーム/リモート・フレーム

代表例として、データ・フレームの構造を以下に示します。各枠の下段にある数字はビット数を表します。  
リモート・フレームは RTR がリセッシブになり、Data Field が無い形になります。



### a) Start of Frame

データ・フレーム又はリモート・フレームの始まりを示すもので、必ず 1 ビットのドミナントです。このバスアイドル状態(リセッシブ) ドミナントへのエッジは他のすべてのノードに対する同期信号となります。

### b) Identifier Field

識別子フィールドともいわれ、通常(標準フォーマット) 11 ビットで構成され、2048 種類(実際は 2032)のメッセージ設定が可能です。メッセージの内容、種別を表す重要な役割を持っており、通信調停(バスアクセス優先順位決定)に使われます。又、この識別種別数を増やすために、この部分を 29 ビット(500 万種以上)に拡大したものが、拡張フォーマットとして規格化されています。

### c) RTR

リモート送信要求ビット。このビットがドミナントの場合このメッセージがデータ・フレームであることを、又、リセッシブの場合はリモート・フレームであることを表します。

### d) Control Field

次のデータフィールド内で何バイトが送信されるかを示します。リザーブビットと呼ばれる 2 ビットのドミナントと 4 ビットのデータ長情報から構成されます。

### e) Data Field

メッセージ内におけるデータバイトの部分です。最大 8 データ、64 ビットまで設定できます。

### f) CRC Sequence

巡回冗長検査といい、データ送信時のデータ破壊をチェックさせています。Start of Frame から Data Field までのビットスタックされていない(後述)すべてのビットについてある種の計算処理をして得られる符号を送信ノードと受信ノードで検出し、比較します。

### g) CRC Del

通常は 1 ビットのリセッシブとなりますが、上記の CRC チェックでエラーの場合はドミナントに変わり、End of Frame を待ってエラー・フレームに移行します。

### h) ACK Slot

正常受信確認のためのフィールドです。送信ノードはリセッシブにて送信し、CRC シーケンスが一致した受信ノードによりドミナントに上書きされます。

### i) ACK Del

通常は 1 ビットのリセッシブとなります。

### j) End of Frame

送信 / 受信の終了を示す部分で、7 ビットのリセッシブで構成されます。

### k) ITM

データ・フレーム、リモート・フレーム、エラー・フレーム、オーバーロード・フレームから次のフレームの間に挿入されるフィールドで、各フレームの区切りとして、通常 3 ビットのリセッシブで構成されます。インターミッション中はいかなるノードもメッセージの送信を開始できませんが、オーバーロード・フレームを送信し、次のメッセージの送信開始を遅らせることができます。

## エラー・フレーム/オーバーロード・フレーム

エラー・フレームは受信、送信ノードがそれぞれの状態でエラーを検出すると、直ちに送信が開始されます。バスのレベルを連続的にドミナントに上書きしてしまい(後述のエラーアクティブの場合)、誤った送信のあったことを通知します。

オーバーロード・フレームは CAN コントローラが前回のメッセージを処理完了できていない時に受信ノードが送信し、次のメッセージが送信されるのを阻止します。但し、最近の CAN コントローラは処理速度が速くなり、このフレームが現出することは殆どありません。

## CAN バスの特徴

CAN バスの代表的な特徴として次の 5 つがあります。

1. 通信速度が速い
2. 耐ノイズ性が高い
3. 優れた通信調停
4. ビットスタッフィング等による精密な同期
5. 優れたエラー検出とハンドリング

### 通信調停

バス接続によるデジタル通信の方式として様々な種類が規定されていますが、バスへの通信アクセスの方法で分類すると代表的に次の 3 つがあります。

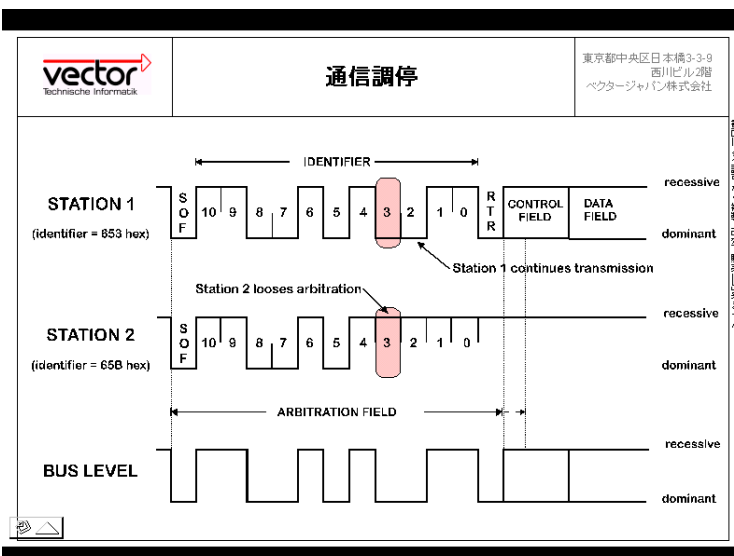
1. マルチ・マスター方式
2. マスター・スレーブ方式
3. トークンパッシング方式

それぞれの方式ともバスに接続されている各ノード (ECU) は自分のデータを別のノードに伝送したい、又は別のノードのデータが欲しいといったメッセージをバス上に流すために通信をスタートさせます。ひとつのバス上で複数の通信が同時に存在することはできないのでそのスタートの仕方に各種ルールを取り決めてあります。これをカラオケ BOX の中でのマイク争奪に例えるならば**マルチ・マスター方式**はとにかく早い者勝ち、**マスター・スレーブ方式**は司会者が次に歌う人を指名するやり方、**トークンパッシング方式**は仲良く順番にマイクを回して歌うという様な方式だと言えます。CAN はマルチ・マスター方式のバスです。一般にマルチ・マスター方式では 2 つ以上のノードが同時に通信を開始しようとしたときは両方をストップさせる方式がとられており、これを CSMA/CD 方式と呼んでいます。CAN はこれをさらに改良したものを採用しています。

CAN では、バスアクセス方式として、**CSMA/CA (Carrier Sense Multiple Access with Collision Avoidance)**方式が使われます。これは、通信を開始するタイミングが衝突しない場合は CSMA/CD 方式と同じように早く通信を開始したノードがその通信を継続する権利を得ます。しかし、同時に通信を開始しようとして 2 つ以上のメッセージがバス上で衝突してしまった場合にどちらかに優先順位をもたせるようにした方式です。前述のカラオケ BOX で言うと、運悪く上司と同時にマイクを握ってしまったら上司に譲れとか、レディファーストで女性に優先権を与えるとか、睨み合いに負けた時は潔く引き下がるとかの喧嘩になることを防ぐルールを作っておくようなものと言えます。

CAN のデータ長は最大で 8 バイトと規定されていますから、一回の通信時間は短時間で終了するため、通信の衝突は起こり難いのですが、最悪、衝突が発生しても両方ともストップするのではなく、優先度の低い通信は、優先度の高い通信が終了してから再度送信を開始できるようになっています。これを通信調停と呼んでおり、効率の良い通信を実現させています。

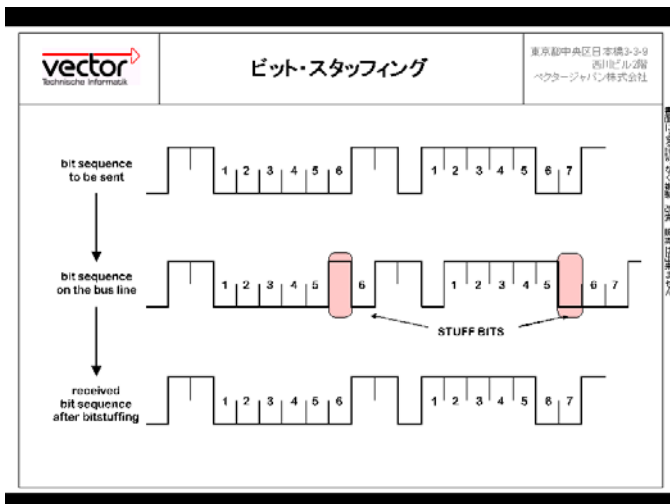
この優先度を決定する仕組みはフレーム内の先頭にある Arbitration Field に ID として書き込まれます。



実際に優先度はどのようにして決定されるかというと、まず各ノードは通信時に常に自分がバス上に送ったデータと、実際にバス上に現れたデータをチェックします。通信を開始したい場合、各ノードはフレームの初めにその通信の優先度を示す 11 ビットの Identifier をバス上に送ります。もし、たった一つのノードしか通信をしないのならそのノードが送出した Identifier はそのままバス上に現れます。しかし複数のノードが同時に通信を始めた場合、それぞれのノードが送ったデジタルの 0 / 1 信号は上書きされ、前述のように、“0”が優位なので“1”の信号は消えてしまいます。すると、ノードは自分が“1”の信号を送っているのに、バス上に“0”の信号が現れたことにより、自分より優先度の高いノードが通信をはじめていることを知るわけです。このとき、優先度の低いノードは即座に通信をストップさせるので、結果として高い優先度を持ったノードが通信を継続できることになります。

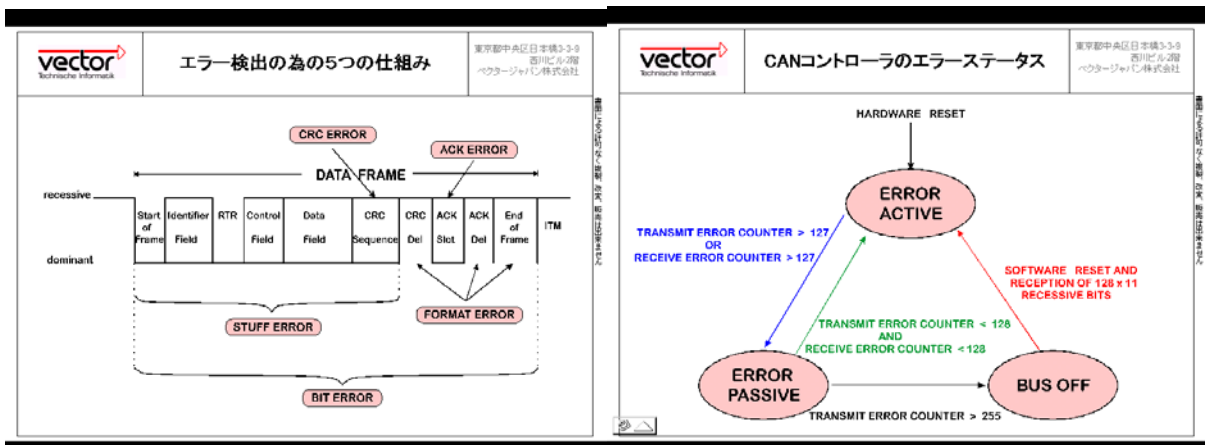
## ビットスタッフィングによる同期

ビットスタッフィングとは、同一信号レベルが5ビット以上連続した場合に反転データを付加して再同期を取りやすくする機構をいい、Start of Frame から CRC Sequence までの間でのみ行われます。CAN では常に送信レベルとバスレベルとを比較監視しており、ビットの同期ずれはすべてエラーとなってしまいます。ビットスタッフィングによる再同期は通信の確実性を向上させます。



## エラー検出とハンドリング

CAN はすぐれた 5 つのエラー検出機構を持っています。そして、その検出によって得られたスコアで、各ノードの状態を管理し、故障したノードによるエラー・フレーム連発によりバスが占有されるような状況を回避しています。



#### a) BIT ERROR

すべての領域で常に送信した信号レベルとバスに現れる信号レベルとを比較監視しており、不一致があった場合はエラー・フレームが送信される。但し、Arbitration Field と ACK Field では判定は行わない。

#### b) ACK ERROR

アキュレシスロットにおいて、どの受信ノードもドミナント(正常受信した)を返さなかった、つまりバスレベルがリセツシブのままの時は、全てのノードが誤ったメッセージを受信したか、ネットワークに他のノードが存在しないことになり、エラー・フレームが送信されます。

#### c) CRC ERROR

CRC Sequence の項でも説明したとおり、送信データと受信データをお互いにある種の計算処理をした結果を比較することにより、不一致の場合に発するエラーです。

#### d) FORMAT ERROR

CRC Del, ACK Del, End of Frame のように、本来フォーマットが固定されている(必ずリセツシブ)ところで、ドミナントを受信した時に発するエラーです。

#### e) STUFF ERROR

ビットスタッフィングルールが守られずに 6 ビット以上の連続した同一レベルの信号を受信した時に発するエラーです。

上図右に示すように、エラー状態の種類として、次の 3 種があります。

1. エラー・アクティブ
2. エラー・パッシブ
3. バス・オフ

各ノードはエラーカウンタを持っており、エラー・フラッグが出されるごとに「8」カウントアップされてゆき、「127」をこえるとエラー・パッシブに移行します。送信ノードがエラーなしで送信を完了したり、受信ノードがエラー無しで受信したりできた場合は「1」減算されます。最悪「256」まで達すると通信ができなくなり、復帰は不可能となります。

サッカーに例えると、エラー・アクティブは通常のプレイヤーで、エラー・パッシブはイエローカードをもらったプレイヤー、バス・オフはレッドカードにて退場処分と言えれば解り易いでしょう。

## おわりに

CAN は自動車メーカー各社が独自のバスシステムを開発している中で、唯一オープン化され、標準採用されつつあるシステムです。オープン化は他のバスシステムでも見られるようなメリットを生みます。つまり、機能やインターフェースが公開されているために、誰でもその仕様にしたがった製品を開発でき、異なるベンダー間での相互運用が可能のため、ベンダー、ユーザーの双方に多大のメリットをもたらしています。

ベンダーは通信部分の標準化により、アプリケーション開発に注力できますし、ユーザー側から見ても通信部分を気にしないで求める機能で幅広く調達が可能になるわけです。

このように優れた特徴を持つ CAN は、自動車だけではなく、産業、船舶、農業機器、医療機器など様々な分野への適用拡大も進んでいます。又、他の通信プロトコルとの関係も進んでいます。

CAN の普及が進みつつある理由の中に、優れた開発支援ツールが存在することを忘れてはなりません。一般的なイメージとしてデジタル信号の解析は難しいという概念がありますが、信号を視覚的に表示したり、シミュレーションしたりできるツールが非常に充実していることは開発者にとって、開発の効率化と完成度の高さを同時に提供します。