



株式会社サニー技研

お手軽ネットワーク「CAN」の世界

Open Source Conference 2026 Nagoya 26/05/23



TOPPERS

お品書き

- 自己紹介
- CANって何だ？
- 使ってみよう！
- まとめ



TOPPERS公式マスコット「とばめ」

自己紹介

米田 真之(よねだ まさゆき)

株式会社サニー技研(TOPPERSプロジェクト 正会員)
イノベーション企画部 副部長

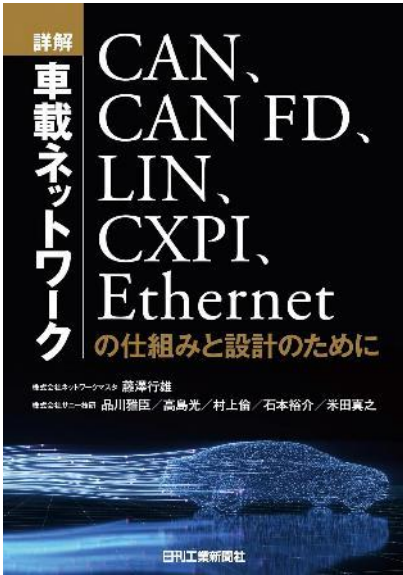
神戸生まれ大阪育ちのトラキチ。小学校1年でMSXのBASICに触れ、父のお下がりのポケコンでプログラミングの楽しさを知る。

初めての自分用PCはPC-9821(当時Windowsはなく、MS-DOS v5.0)。CONFIG.SYSをいじくりまわし、メインメモリ640KBを如何に空けるのかに没頭
高校からDelphi(Object Pascal)に触れ、卒論のプログラムもDelphiで製作。今でも言語仕様はDelphiが一番好き。

2004年就職後は車載ネットワーク(CAN/LIN/FlexRay等)関連でC言語メインとなるも、途中から車載開発向けツールやGUI開発に異動。C++、C#で開発(一部Delphiも)。社内向けシステムでJava。AIやSBC関連でPython、Arduino言語 etc…。

自己紹介

CANを含む車載ネットワークは企業向けに講師、書籍・雑誌投稿等の経験



R4時代 Arduino × トランシーバ MCP2551 でいろいろ使える！
CAN通信&回路入門…
クルマ診断OBD-IIモニター
 米田 真之 Masayuki Yoneda

Arduino Uno R4ではマイコンとしてRA6M1(ルネサス・エレクトロニクス)が採用されています。従来のArduinoなどで、自動車向け通信ネットワークCANを利用する際に使っていたCANバス・シールド(CANコントローラICとトランシーバなどがセットになった基板)などを使わず、簡易ドライバ回路を用意するだけでCAN通信を行うことができます。そこで、CANの送受信ボックスをArduinoを用いて製作し、自動車に接続してその通信内容をのぞいてみます(図1)。

CANはなぜクルマや産業に使えるか

● 2線だけで1対1だけでなくネットワーク化できる
 CAN(Controller Area Network)は、1983年にドイツの電気メーカーであるボッシュ社が、自動車の電子制御システム向けの通信プロトコルとして発明したもので、1994年にISO11898で規格化されています。昔の自動車はワイヤ・ハーネスにより電子機器がそれぞれ1対1で接続されており、その重量だけで数kgもあり、燃費を考えると軽量化が望まれていました。その後、CANをはじめとする車載ネットワークの登場により、ワイヤ・ハーネスが減ったことによる軽量化だけでなく、車内空間が広くとれるなど多くのメリットが生まれました。

● リアルタイムで高信頼性
 シリアル通信方式にはUARTやSPIなどがありますが、CANはUARTとは異なり多対多の通信が可能です。またSPIと比較して配線数が少なく、配線を長くともできる上にノイズに強く、リアルタイム性も高いです。自動車に搭載されている電子制御ユニット(ECU: Electronic Control Unit)に最も適した通信手段となっています。CANとUART、SPIとの比較を表1に示します。

● 高速度も選んでいる
 また、CANにはその派生としてペーロードを増やした、高速化したCAN FD(CAN with Flexible Data rate)や、CAN FDを用いて照明やセンサー制御に特化したCAN FD Light、またCAN FDよりもさらに大容量化、高速化を図ったCAN XLもあります。その他CANをベースにした通信プロトコルとしては表2に示したものがあります。どれもCANのリアルタイム性や高信頼性があるからこそこのプロトコルとな



図1 実際のクルマのCAN通信モニター実験
 CAN通信が完了した状態でArduino Uno R4を使用し、実際の自動車の高電圧ネットワークをモニタリングしている。

表1 シリアル通信方式の比較
 CAN通信の強みとして、送受信を専用のチップで実現し、配線数を減らすことができる。リアルタイム性も高い。自動車の制御に欠かせないプロトコルである。

	CAN	UART	SPI
通信距離	マルチメータ	10m	メートルメートル
配線数	2 (CAN, J1939)	2 (TX, RX)	最低4 (MOSI, MISOなど)
通信速度	10Mbps	115200bps	最高10Mbps
遅延	40ns	中程度	約100ns
ケーブル	1Mbpsの場合	バリエーションあり	なし
価格	高電圧機能あり	さまざまあり	(チップ間で実装)

第8部 自動車
 第1章
車載ネットワーク
 後藤 幸一、米田 真之、前田 裕平、浪遊 晋也、松浦 実莉子

1. 通信プロトコル

表1 車載ネットワークの通信規格と通信プロトコル

車載通信プロトコル	用途	最高通信速度	ケーブル	規格化団体
CAN (Controller Area Network)	高信頼性の通信プロトコルであり、マルチメータ・マルチメータネットワークで最も多く使われる。メカニカル・ハードウェアの通信方式で伝送エラー発生率を低く抑える。CANは1Mbpsの通信速度で動作するが、CAN FDは10Mbpsの通信速度で動作する。また、CAN FDはCANよりも高速で動作し、ノイズに強い。通信距離を伸ばすことも可能で、通信距離が長い。	10Mbps	2線(12V, GND)	CAN in Automotive (CSA)
CAN FD (Flexible Data Rate)	CANプロトコルをベースにした高速通信プロトコルであり、CANよりも高速で動作し、ノイズに強い。通信距離を伸ばすことも可能で、通信距離が長い。また、CAN FDはCANよりも高速で動作し、ノイズに強い。通信距離を伸ばすことも可能で、通信距離が長い。	10Mbps以上	4線(12V, GND)	CAN in Automotive (CSA)
LIN (Local Interconnect Network)	自動車内で最も多く使われる低速シリアル通信プロトコル。目標として、通信速度が20kbit/s、エラー率10^-7以下を実現。コストが低く、配線が簡単。リアルタイム性が求められる。エラー率10^-7以下を実現。コストが低く、配線が簡単。リアルタイム性が求められる。	20kbit/s	2線(12V, GND)	LIN Consortium
CXPI (Clock Extension Protocol Interface)	自動車内で最も多く使われる高速シリアル通信プロトコル。目標として、通信速度が100kbit/s、エラー率10^-7以下を実現。コストが低く、配線が簡単。リアルタイム性が求められる。エラー率10^-7以下を実現。コストが低く、配線が簡単。リアルタイム性が求められる。	20kbit/s	2線(12V, GND)	CXPI Consortium
FlexRay	自動車内で最も多く使われる高速シリアル通信プロトコル。目標として、通信速度が10Mbps、エラー率10^-17以下を実現。コストが低く、配線が簡単。リアルタイム性が求められる。エラー率10^-17以下を実現。コストが低く、配線が簡単。リアルタイム性が求められる。	10Mbps	2線(12V, GND)	FlexRay Consortium
MOST (Media Oriented System Transport)	自動車内で最も多く使われる高速シリアル通信プロトコル。目標として、通信速度が150Mbps、エラー率10^-17以下を実現。コストが低く、配線が簡単。リアルタイム性が求められる。エラー率10^-17以下を実現。コストが低く、配線が簡単。リアルタイム性が求められる。	150Mbps	3線(12V, GND)	MOST Consortium
OPES (Open Platform for Ethernet-based Systems)	自動車内で最も多く使われる高速シリアル通信プロトコル。目標として、通信速度が100Mbps、エラー率10^-17以下を実現。コストが低く、配線が簡単。リアルタイム性が求められる。エラー率10^-17以下を実現。コストが低く、配線が簡単。リアルタイム性が求められる。	100Mbps	4線(12V, GND)	OPES Alliance (Auto-Fair Ether-Net) Special Interest Group

車載ネットワークは、自動車内部で異なる電子制御ユニットやデバイスが相互に通信するためのネットワークです。車内のさまざまなシステムや機能(エンジン制御、ブレーキ制御、インフォテインメント、シス...

Interface 2023年9月号

- TOPPERSプロジェクトについて
TOPPERS = Toyohashi Open Platform for
Embedded and Real-Time Systems
- プロジェクトの活動内容
ITRON仕様の技術開発成果を出発点として、組み込みシステム構築の基盤となる各種の高品質なオープンソースソフトウェアを開発するとともに、その利用技術を提供
組み込みシステム分野において、Linuxのように広く使われるオープンソースOSの構築を目指す!
- プロジェクトの推進主体
 - 産学官の団体と個人が参加する産学官民連携プロジェクト
 - 2003年9月にNPO法人として組織化
 - それ以前は、名古屋大学(2002年度までは豊橋技術科学大学)高田研究室を中心とする任意団体として活動

自己紹介

■ TOPPERSプロジェクトについて 開発成果物の利用事例

ETロボコン2023走行体
HackEV

ETロボコン実行委員会
<https://www.etrobo.jp>



デジタルピアノ AP/PXシリーズ



カシオ計算機(株)

<https://www.casio.com/jp/>

デジタル一眼カメラ α6700



ソニー(株)

<https://www.sony.co.jp/>

GPSタイムサーバ TSG-M10



(株)タカコム

<https://www.takacom.co.jp/>

HD AVミキサー VR-4HD



ローランド(株)

<https://www.roland.com/jp/>

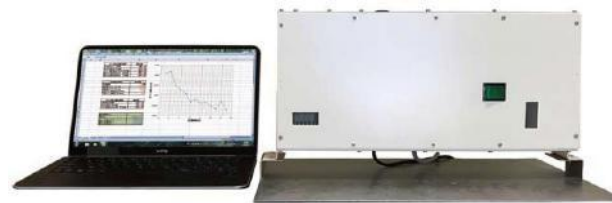
HII-B



宇宙航空研究開発機構

<https://www.jaxa.jp>

LEDオゾン濃度測定装置



光テック(株)(旧(有)光電鍍工業所)

<http://www.hikari-tech.net/index.html>

細胞形態解析イメージングシステム
Cell3 iMager duos



(株)SCREENホールディングス

<https://screen-cell3imager.com/>

CANって何だ！？

■ CANの前に・・・車載ネットワークについて

[自動車は走るコンピュータ]

昔の自動車はメカの固まり

→ 現在はコンピュータの固まり

車にはECU(Electronic Control Unit)と呼ばれる電子制御装置が搭載されている

車の「走る・曲がる・止まる」や安全性、快適性向上の為

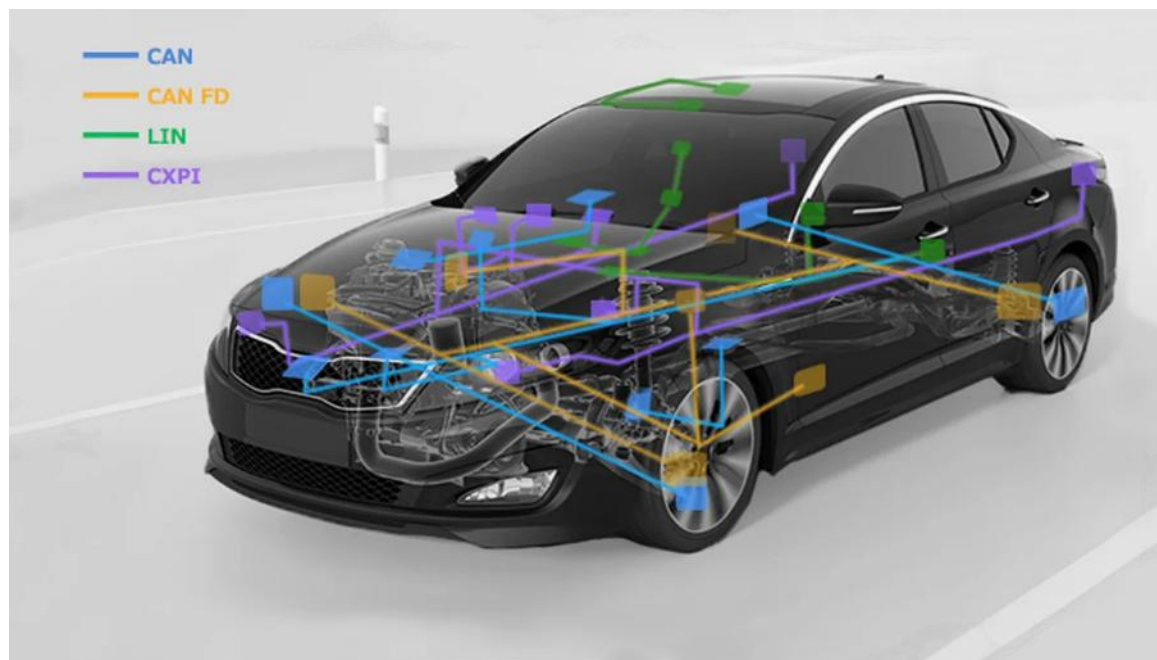
エンジン/トランスミッション/インジェクター/
ブレーキ/サスペンション/パワーステアリング/

エアコン/パワーウィンドウ/電動ミラー/ドアロック/
電動シート/キーレスエントリー/

エアバッグに加え最近ではADAS(先進運転支援システム)も

CANって何だ! ?

- CANの前に・・・車載ネットワークについて
[自動車は走るコンピュータ]
一台の自動車にどれくらいのECUが載っているか？



数十～100個程度

- 小型車: 30～50
- 高級車: 70～100

これらECU同士の情報のやり取りの為に
使用されているのが**車載ネットワーク**

CANって何だ！？

- CANの前に・・・車載ネットワークについて
ネットワークならEthernetでええやん？
→ オーバースペック

車載ECUは少しでも小さく、1円でもコストカットしていくことが重要
ECUはバッテリー容量の取り合いでもある
小サイズ・省コスト・省電力・高信頼性のわがままネットワークが必要！

再送時間がランダムとなるEthernetはECU同士の同調や、車体安定制御などのリアルタイム制御を行いたい自動車には不向き。

車体安定制御

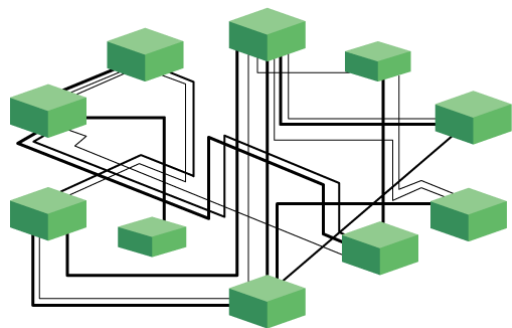
- ABS(Anti-lock Brake System)
- TCS(Traction Control System:タイヤ空転防止)
- ESC(Electronic Stability Control:横滑り防止)
これら制御はエンジン・ブレーキ・ステアリングに加え各種センサ
(車速・加速度・ヨーレート等)情報の同時性が重要

CANって何だ！？

■ Controller Area Networkの略

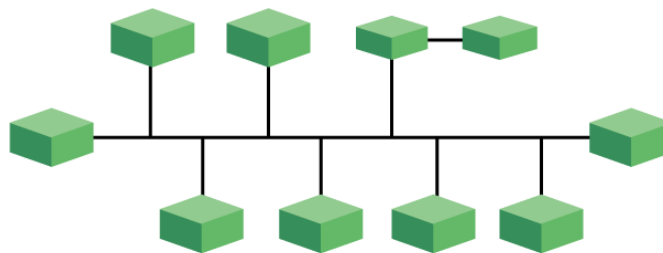
1983年ドイツの電装メーカーのBOSCH社が自動車の電子制御システム向けの通信プロトコルとして提唱。ワイヤーハーネスを削減、複数のLANを介して通信する。ISO-11989で規格化されている。

従来のシステムの場合




昔の自動車はワイヤーハーネスだけで数10kg以上
→重量は燃費に直結。

CANを導入



軽量化だけでなく、配線が少なくなり、車内空間が広がった

車載ネットワークの導入で、それまで1対1ずつで結線するしかなかった制御システムは多対多での結線が可能に。

 : ノード (通信の主体となる個々の機器のこと、ECU等)

CANって何だ！？

■ CANの主な特徴

- 通信方式 : マルチマスタ、CSMA/CR方式、半二重通信
- 伝送路 : 2線式
- 接続形態 : バス型
- 通信速度/データ長 : Max 1Mbps、0Byte~8Byte
- データ誤り検出 : CRC方式

**CANが自動車に使われてい理由となる
特徴や仕様について紹介**

CANって何だ！？

■ CANの特徴:接続形態

● CANはバス型

1本のメイン通信路=バスに各ノードがぶら下がる形

✓ メリット

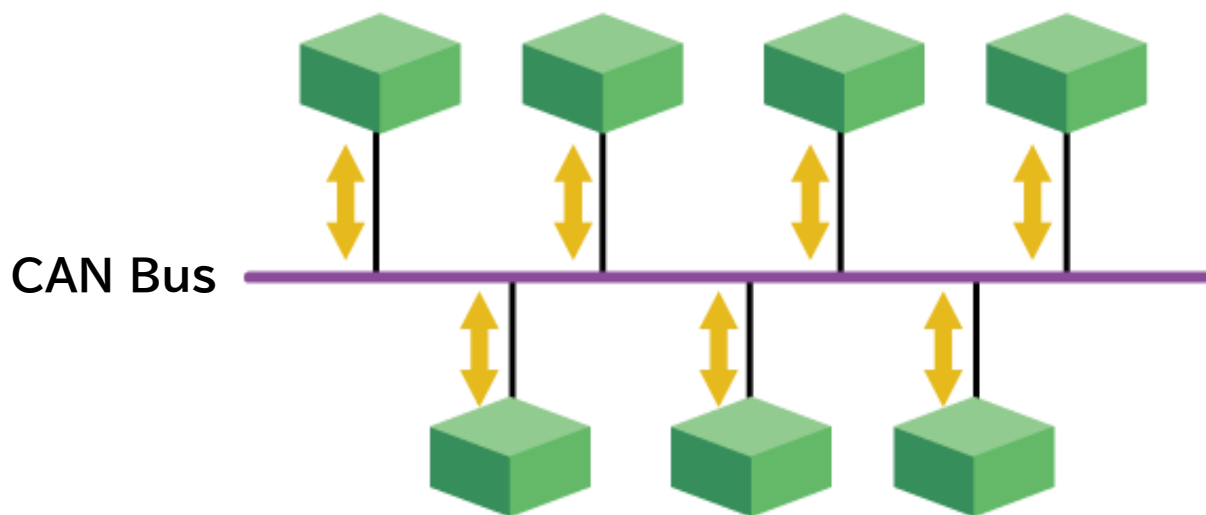
1つのノードが**故障しても通信を維持**できる

コストパフォーマンス性が高い(ローコスト)

✓ デメリット

1つの通信路を使用しているため**衝突回避の仕組み**が必要

バスがショートするとシステム全体がダウンする

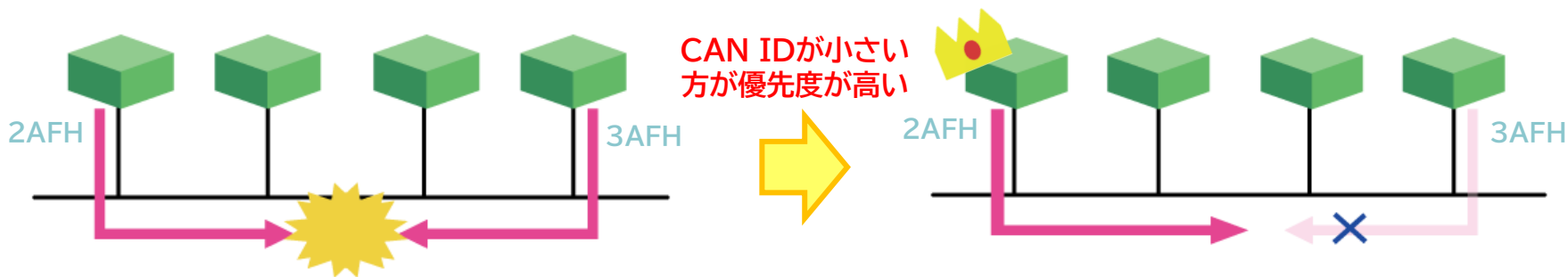


500kbpsならば
最大バス長は100m
※スタブ長は0.3m

CANって何だ！？

■ CANの特徴:通信方式

- マルチマスタ
全てのノードが自分で送信する権利を持っている
- CSMA/CR方式(古い資料ではCAと記載)
バスが空いているタイミングならば、どのノードでも送信を開始できるが、複数ノードが同時に送信開始するとメッセージが衝突。そのため送信するメッセージに優先度(CAN ID)が割り振られている



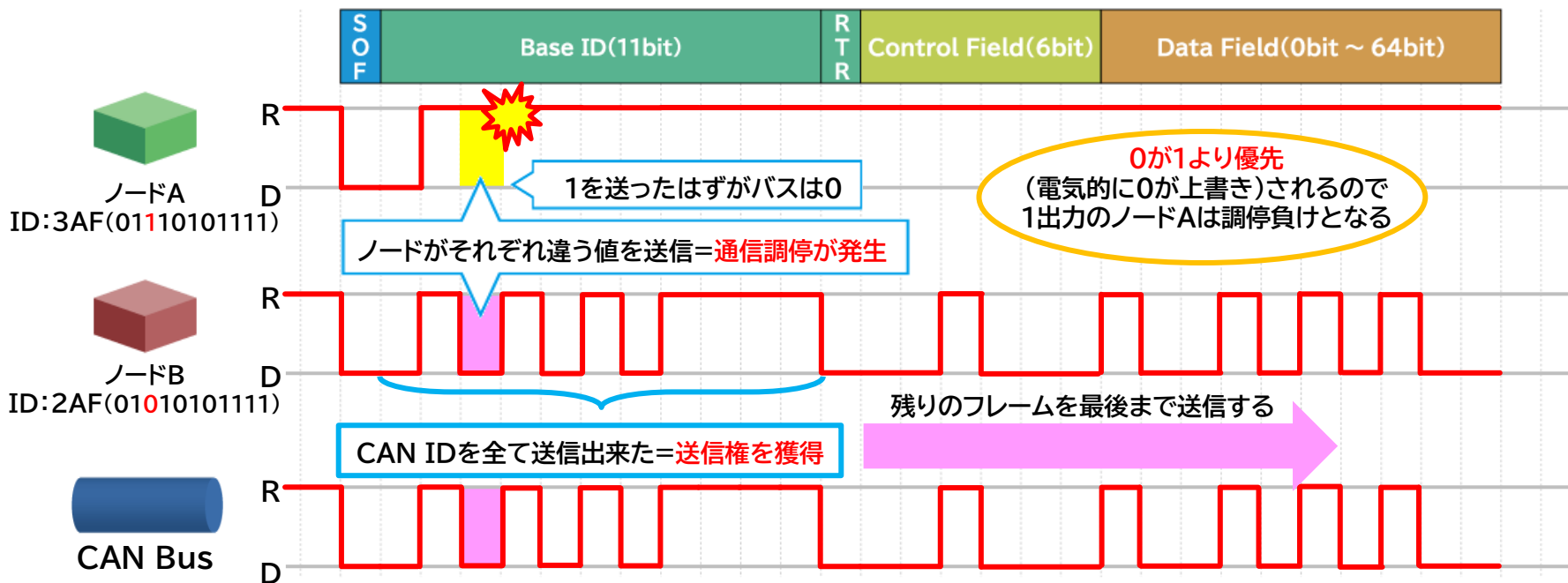
メッセージが衝突しそうになった時には、優先度に従って通信調停が行われる
→エンジン制御や車体制御など、より安全性やアルタイム性を求める情報のCAN IDを小さいものにする事で、**低遅延でのデータ到達を保証する**

CANって何だ! ?

■ CANの特徴:通信方式

- 通信調停(アービトレーション)

送信するノードは通信バスの電圧を監視しており、自身が送信したデータが本当にバスに流れているかを1bitずつ確認。



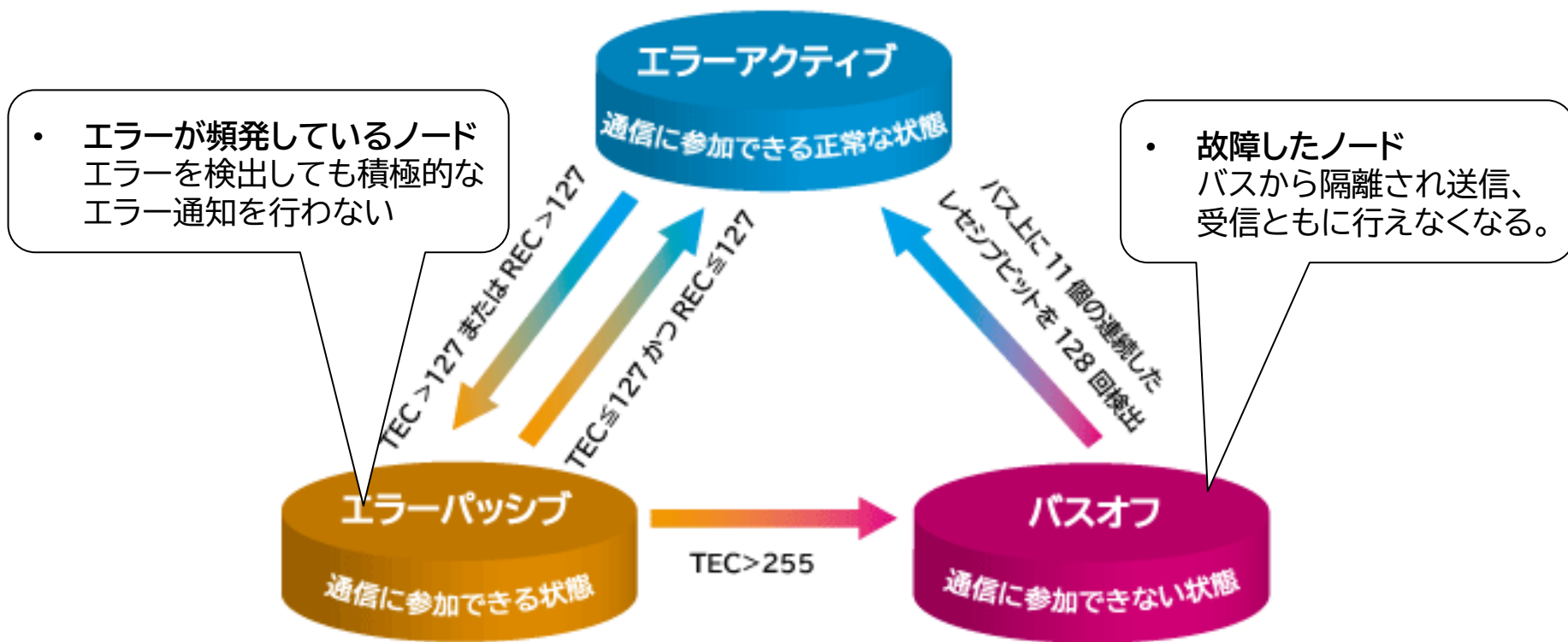
通信調停に負けたノードは次のbitから送信動作を停止し
受信動作に切り替わる。

CANって何だ!?

■ CANの特徴:故障の封じ込み

- エラーステータス

各ノードはエラー状態というステータスを持ち、送信エラーカウンタ、受信エラーカウンタの値に応じて遷移



エラーを多発するノードは通信から隔離することで、他のノードの通信を妨げない

CANって何だ！？

■ CANの特徴:故障の封じ込み

● エラーの種類

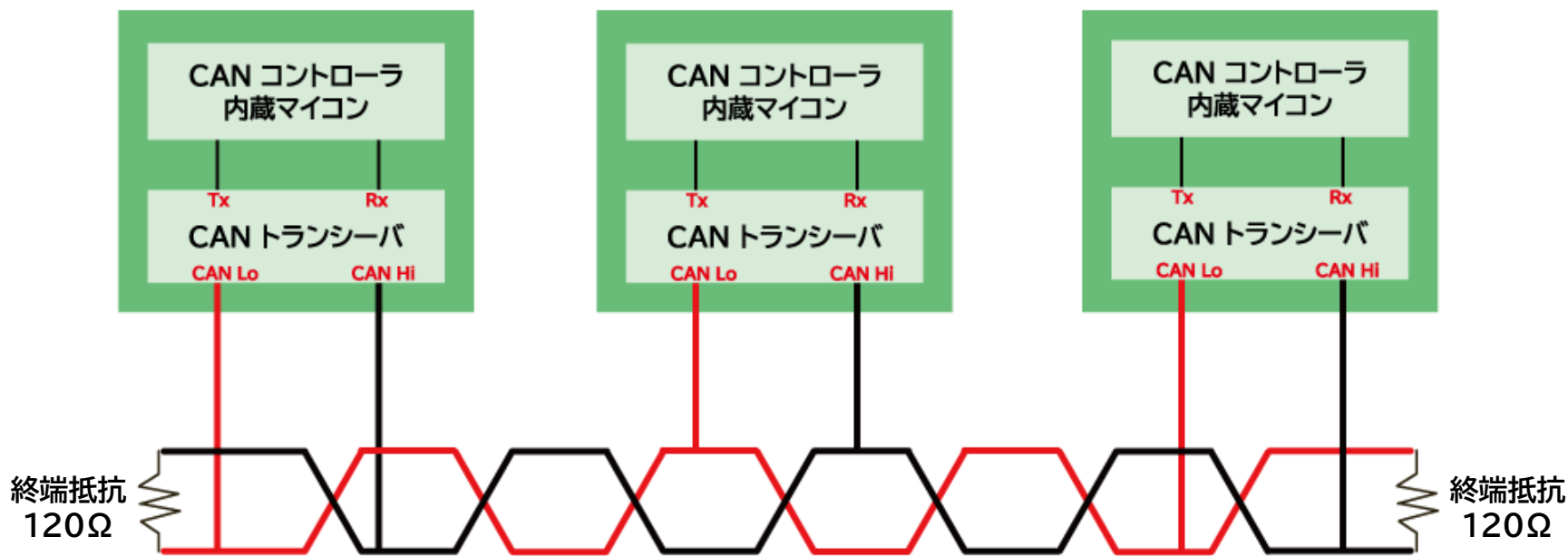
- ビットエラー
アービトレーション以外で出力レベルとバスレベルが不一致
- スタッフエラー
同期ずれを防ぐため5連続した0または5連続した1を送信したら、その反転ビットを挿入しなければならない仕様になっているが、同一レベルが6bit連続の場合
- CRCエラー
受信されたメッセージから算出したCRCの値と受信されたCRCの値が異なる
- フォームエラー
固定のフォーマットのビットフィールドに違反した時
- ACKエラー
送信ノードのACKスロットがレセシブレベルの場合
→送信したものの他のノードが受信を検知しなかった

エラーを検出したノードはエラーフレームを出力
送信ノードの場合はその後再送を試みる

CANって何だ！？

■ CANの特徴:伝送路

CAN Hi、CAN Loとよばれる2本の通信線を使って、送信受信の両方を行っており、配線の両端に120Ωの終端抵抗を2つ設置する必要がある。



ISO11898-2(Highspeed CAN)の接続イメージ

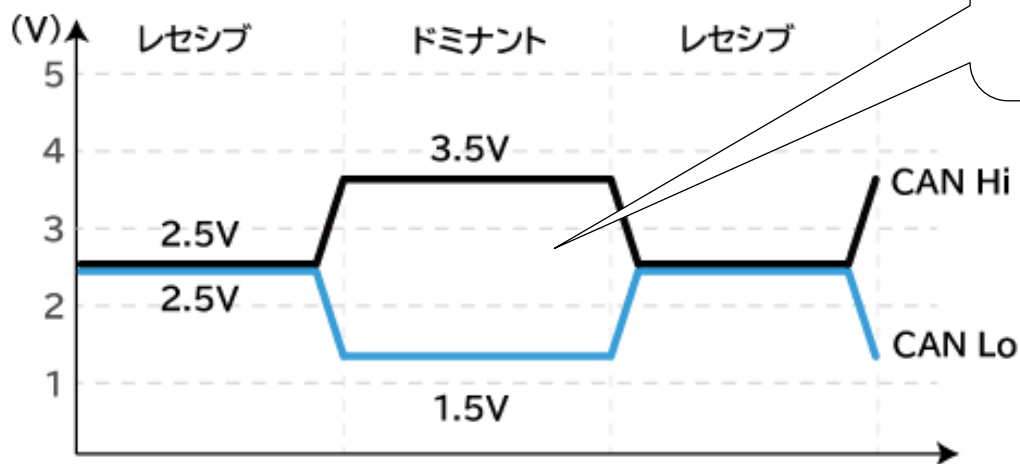
CANって何だ！？

■ CANの特徴:伝送路

CAN HiとCAN Loの2線の差動信号による通信

- バスの電氣的構造

ドミナント:論理値 0、レセシブ:論理値 1



電位差が規定以上あるかないかで決まる

CANバスはオープンドレイン構成

レセシブ：ハイインピーダンス

ドミナント：電流を流す

バス上はドミナントが優勢(Dominant)となる

CANって何だ！？

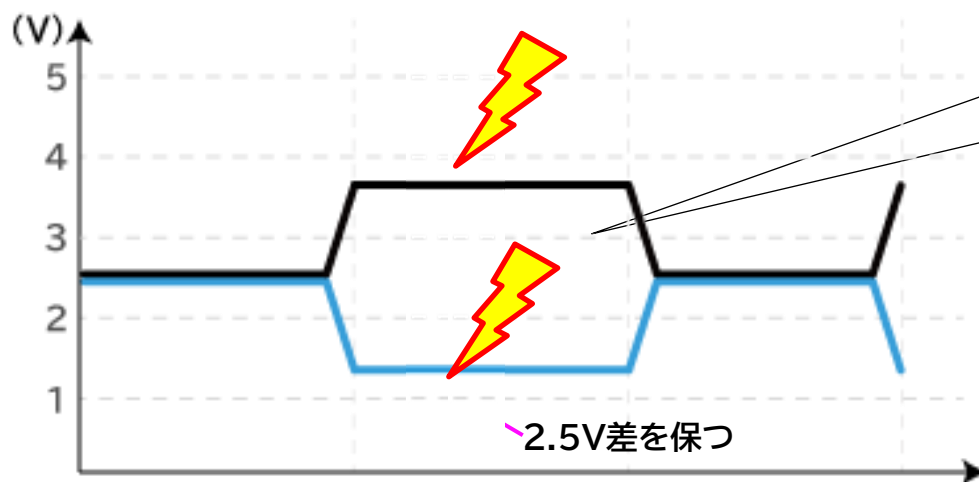
■ CANの特徴:伝送路

CAN HiとCAN Loの2線の差動信号による通信

- 耐ノイズ性

自動車はノイズだらけ

点火プラグやオルタネータ、パワーウィンドウなどのモータ、インバータ、コンプレッサに加え、外部からのノイズ(高圧送電線/携帯電話/ラジオ塔/雷など)からも影響を受ける。



電位差をとるので
ノイズが打ち消される



ツイストペア線

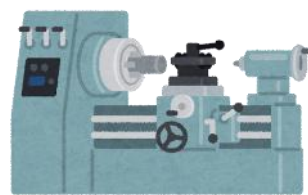
CANって何だ！？

■ 車以外でも？至る所にCAN

- FA(Factory Automation)
- 産業機械、工作機械
- 鉄道、船舶、航空宇宙
- 医療

● CANを応用したプロトコル(一部)

- J1939
トラック、バス、建機車両
- ISOBUS
農業機械(トラクタと作業機の通信共通化)
- CANopen / DeviceNet
モータ、バルブ、センサ、エンコーダ、表示機、
シーケンサ、PLC etc…



自動車で培われた技術が広く利用されている

CANって何だ！？

■ CAN以外の車載ネットワーク

自動車にはCANのほかにも様々な車載ネットワークが採用されている

- CAN FD(CAN Flexible Data-rate)
は「CAN」の通信速度を高速化(最大5Mbps)し、データ容量を最大64バイト(従来は8バイト)に拡張したもの
- LIN(Local Interconnect Network)は
単線で通信、UARTベースでCANより更に低コスト
Master/Slave通信
- CXPI(Clock eXtension Peripheral Interface)
LINの低コストを維持し、より即時性を高めたCANとLINのイイ
とこ取り

とはいえ近年ではADASや自動運転への対応のため、高速・大容量なEthernetの採用も。ただし領域が限定的であったり、独自の物理層(10BASE-T1S、ツイストペア)となっている

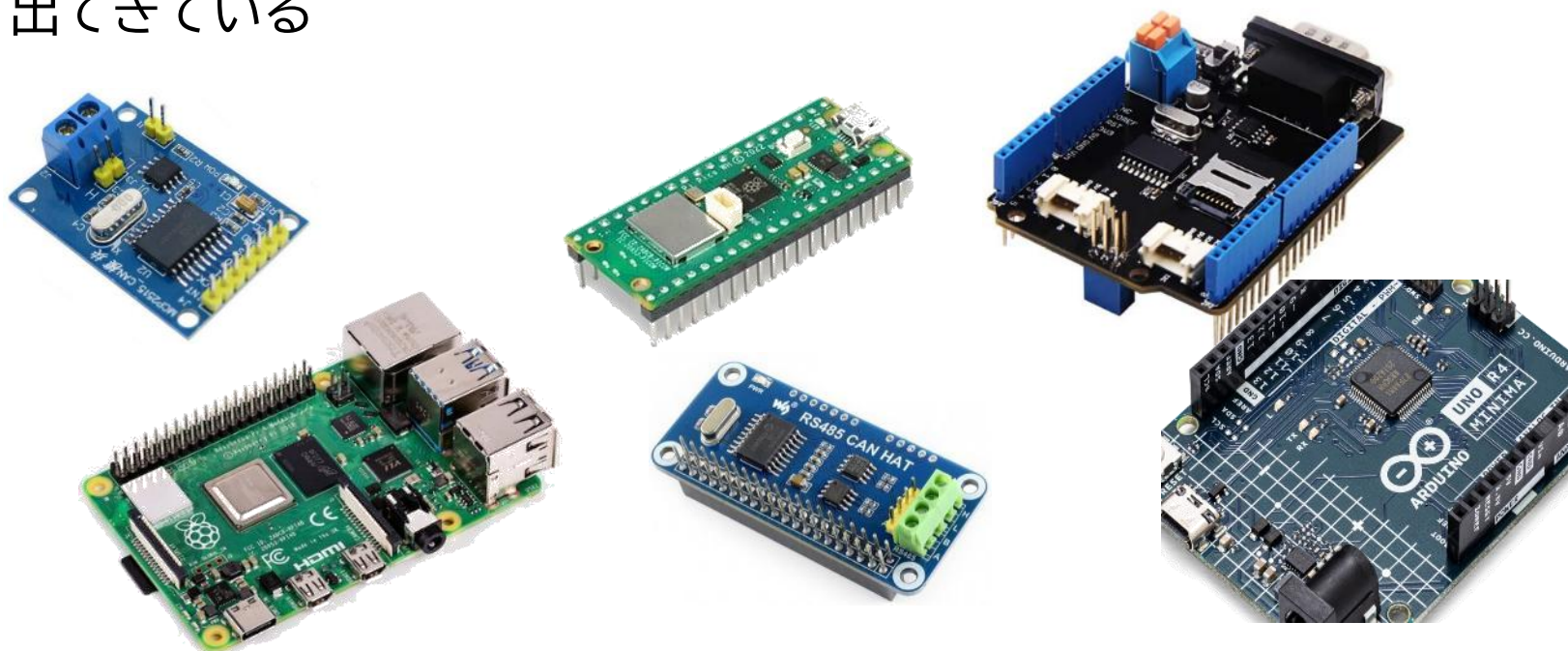
使ってみよう！

- 最近では簡単にCANを使える環境が揃ってきた！

自動車向けに生まれたCAN

→車載用マイコンに搭載、個人では入手しづらい
専門のH/W知識、実装のノウハウが必要

最近では個人で購入可能なラズパイ用CAN拡張ボード(HAT)やそもそもCANが機能として載っているマイコンが搭載されたSBCなど、出てきている



使ってみよう！

- 今回のTOPPERSプロジェクトブース展示では
Auduino UNO R4 + CANシールドを使ったCAN



ラズパイ + SocketCAN対応デバイスを使ったCAN



で通信を行い、CAN解析ツールでモニタリング
それぞれの実装について簡単に解説

小型車のCANと同等
の約20mの配線長で
通信させています。
高級車なら50m
バスなどは100mも

使ってみよう！

- Arduino UNO R4 + CANシールドを使ったCAN SPIを用い、CAN通信が実現可能なMicrochip製「MCP2515」とCANトランシーバなどを搭載したシールド基板「CAN-BUS Shield V2」を使用。使う為のライブラリはArduino IDEのライブラリマネージャーから検索してインストールすることができます。



インストール後、メニューから
[スケッチ]→[ライブラリをインクルード]
→[CAN_BUS_Shield]を選択することで
必要なヘッダファイルをソース上に挿入
できます。

wiki: https://wiki.seeedstudio.com/CAN-BUS_Shield_V2.0/

使ってみよう！

- Arduino UNO R4 + CANシールドを使ったCAN簡単なコードで、CANの送信や受信が可能。

```
#include <SPI.h>
#include <mcp2515_can.h>

// CAN Shieldとの通信用オブジェクトmcp2515_can can(CS_PIN);

// CAN Shieldの初期化
if (can.begin(CAN_500KBPS) == CAN_OK) {
  Serial.println("CAN Shield init ok!");
}

// CAN送信データの設定
byte tx_Data_a[] = {0x01, 0x02, 0x03, 0x04};

// 送信
can.sendMessage(0x100, 0, sizeof(tx_Data_a), tx_Data_a);

// 受信
byte rx_Data[8];
byte rxLen;
can.readMessage(&rxLen, rx_Data);
```

使ってみよう！

■ ラズパイ+SocketCAN対応デバイスを使ったCAN

● SocketCANって？

Socket CAN は、Linuxカーネルが持つコンピュータに接続されたCANインタフェースを利用してCAN通信ができる機能

- ✓ カーネル機能のためディストリビューションに関わらず利用が可能
- ✓ ディストリビューションによっては標準で有効化
- ✓ Socket CANに対応のインタフェースを接続するだけ

名前の通りSocket APIはTCP/UDPなどのSocket通信に似たインターフェースとなっている

	TCP/UDP Socket API	Socket CAN API
変数	<pre>int socket_fd; //ディスクリプタ struct sockaddr_in addr; //Ethernet構造体</pre>	<pre>int socket_fd; //ディスクリプタ struct sockaddr_can addr; //CANデータ構造体</pre>
ソケット生成	<pre>if((socket_fd = socket(PF_INET, SOCK_DGRAM, 0)) < 0) { /* エラー処理 */ }</pre>	<pre>if((socket_fd = socket(PF_CAN, SOCK_RAW, CAN_RAW)) < 0) { /* エラー処理 */ }</pre>
NIC指定	<pre>strcpy(ifr.ifr_name, "eth0"); if(ioctl(socket_fd, SIOCGIFINDEX, &ifr) < 0) { /* エラー処理 */ }</pre>	<pre>strcpy(ifr.ifr_name, "can0"); if(ioctl(socket_fd, SIOCGIFINDEX, &ifr) < 0) { /* エラー処理 */ }</pre>
設定	<pre>memset(&addr, 0, sizeof(addr)); addr.sin_family = AF_INET; addr.sin_addr.s_addr = htonl(INADDR_ANY); addr.sin_port = htons(servPort);</pre>	<pre>memset(&addr, 0, sizeof(addr)); addr.can_family = AF_CAN; addr.can_ifindex = ifr.ifr_ifindex;</pre>

使ってみよう！

- ラズパイ+SocketCAN対応デバイスを使ったCAN
 - 対応デバイス
 - USB-CAN対応デバイス

PEAK-System PCAN-USB
Kvaser Leaf Light
サニー技研 MicroPeckerX
etc...



- CAN HAT(今回はこちらを使用)

CANコントローラ+トランシーバが
搭載されている基板
→ラズパイとはSPIで接続



Amazonなどでも購入できます。

使用するデバイスごとに、LinuxカーネルがCANインタフェースを認識できる設定は必要

使ってみよう！

■ ラズパイ+SocketCAN対応デバイスを使ったCAN

● SocketCANの有効化

```
# 各モジュールの有効化
sudo modprobe can
sudo modprobe can_raw
sudo modprobe can_dev
```

● 確認

```
ip addr | grep "can"
```

can0がCANインターフェースとして認識されている

応答

```
3: can0: <NOARP,ECHO> mtu 16 qdisc noop state DOWN
group default qlen 10 link/can
```

● 開始

```
# ボーレートの設定
sudo ip link set can0 type can bitrate 500000

# link up状態に
sudo ip link set up can0
```

使ってみよう！

- ラズパイ+SocketCAN対応デバイスを使ったCAN
こちらも簡単なコードで通信が可能

初期化処理(一部省略)

```
int  sockfd; /* Socket用ディスクリプタ */
struct sockaddr_can addr;
struct ifreq ifr;
struct canfd_frame frame; /* CAN-FD送信用フレームデータ */

/* CANプロトコルを指定してソケットを開く */
sockfd = socket(PF_CAN, SOCK_RAW, CAN_RAW);

addr.can_family = AF_CAN;
strcpy(ifr.ifr_name, "can0");
ioctl(sockfd, SIOCGIFINDEX, &ifr);
addr.can_ifindex = ifr.ifr_ifindex;

/* ソケットと設定をバインド */
bind(sockfd, (struct sockaddr *)&addr, sizeof(addr));
```

使ってみよう！

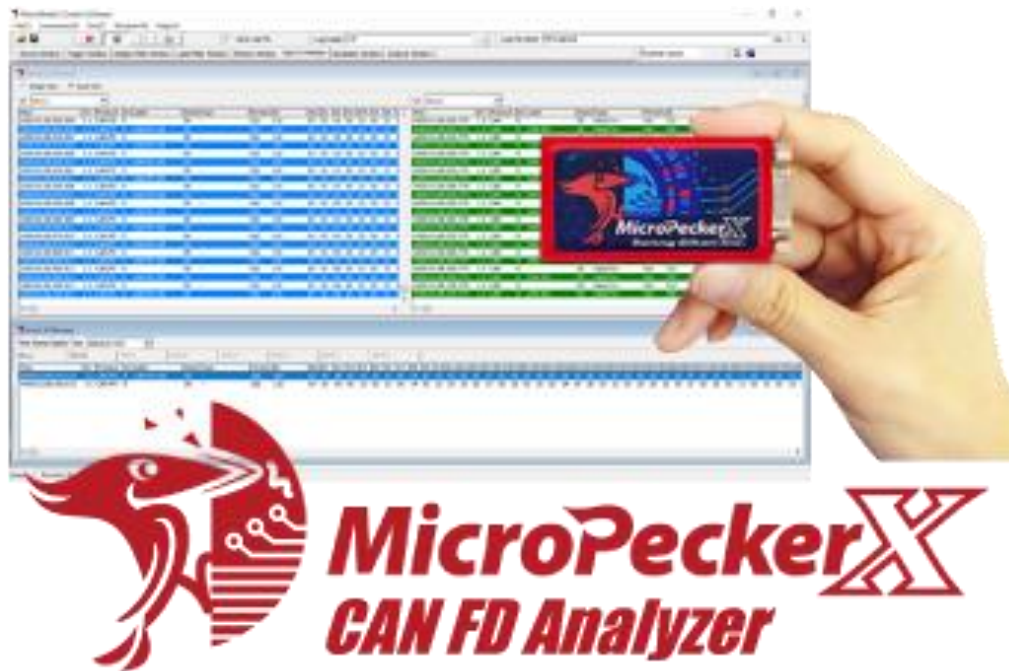
- ラズパイ+SocketCAN対応デバイスを使ったCAN
こちらも簡単なコードで通信が可能

送信処理例

```
/* CAN ID */  
frame.can_id = 0x100;  
/* データ長 */  
frame.can_dlc = 8;  
  
/* 送信データの設定 */  
for (int i = 0; i < 8 ;i++) {  
    frame.data[i] = i;  
}  
  
/* 送信 */  
write(socketfd, &frame, sizeof(frame));
```

使ってみよう！

- モニタや解析は専用のソフトウェアが便利
Windows用 CAN FD対応アナライザ「MicroPeckerX」
<https://sunnygiken.jp/product/micropeckerx/s810-mx-fd1/>
→モニタだけでなく送信も可能、通信の解析も



デモでは、Arduino⇔ラズパイ間のCAN通信モニタを実施

使ってみよう！

- ネットワークアナライザ？
各ECUベンダはCAN通信を実装後、評価を行う
→CANは単体では通信が成立せず、相手が必要
 - 通信モニタ
正しい通信データが送信されているか
 - 送信シミュレータ
受信によって正しい動作をするか

だけではなく・・・色々ベンダで実装されたECUをつなぐので
自動車全体・自動車ならではの視点での評価も必要

- 通信品質の確認(1対多の環境)
自動車においては通信の同時性が重要(前述)
 - ・ 送信周期の確認(ジッタなど)
 - ・ ゲートウェイ経路による遅延

なのでネットワークモニタではなくアナライザ(解析機)

誰でも簡単に実装できるということは・・・

- セキュリティリスクにつながっている



自動車の運転席の下には
画像のようなCANに接続
可能なコネクタが必ず搭載
されている

→OBD-IIコネクタ
本来は自動車の診断用

CANはそのままではスヌーピングやなりすましに対し非常に脆弱で、ECUが送信するデータに対し、同じIDで異なるデータを送ることで簡単に受信先を騙すことができる。

CANインバーダーと呼ばれる、ハッキング装置を使用した盗難も

誰でも簡単に実装できるということは・・・

- CANインバーダーとは
従来の盗難とは異なり、鍵を壊したり窓ガラスを割らない



フロントバンパーを外し、
内部の配線と機器をケーブル
で接続し制御システムに不正
な送信をして開錠やエンジン
始動を行う

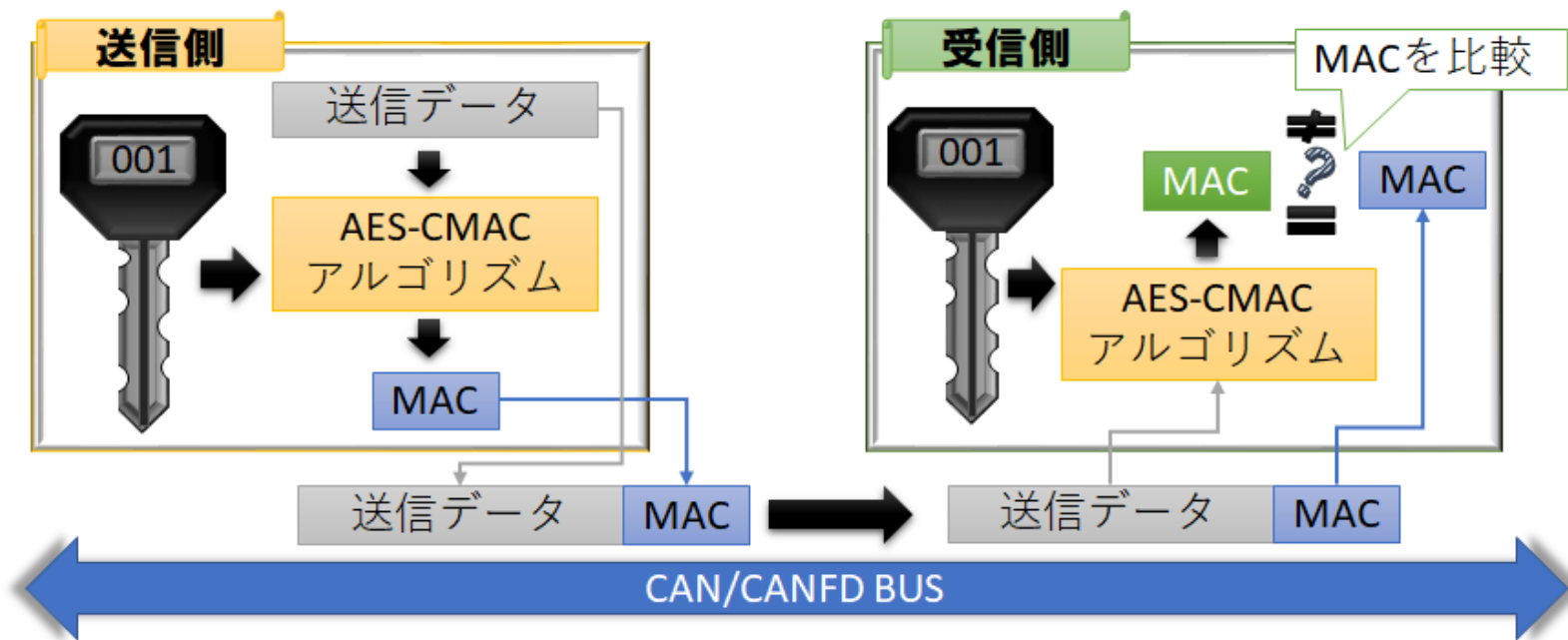
さらに最近では「つながるクルマ」として、インターネット回線等に接続し、ドライバへの情報提供や自動車内のソフトウェアのアップデート(OTA)などもおこなえるようになっており、外部経由での**CANへの攻撃**なども

CAN通信そのものだけではなく、セキュリティ対策含め、CAN通信に付随する技術への対応が必要

誰でも簡単に実装できるということは・・・

■ セキュリティ対策：メッセージ認証

送信側と受信側で「同じ鍵」と「同じアルゴリズム」を利用して生成するメッセージ認証コード(Message Authentication Code : MAC)を比較。MACは送信データと鍵の一方でも異なると異なる値となるため、データの改ざんが検出できる。



まとめ

- 自動車に車載ネットワークは必須
命を預かるため、高信頼性・リアルタイム性が重要
 - CANはそれに適う通信として生まれた
自動車もEthernet化が進んでいるが、低コスト・省電力の観点ではCANはまだまだ優位
 - 医療、宇宙など多分野で応用される「枯れた技術」の強み
 - IoT化、コネクテッド化に伴い、**セキュリティ対策も重要**に
- 個人でも「お手軽」に使える時代に！
ハードウェアもソフトウェアも、誰でも安価に環境構築が可能

ボード同士のちょっとした情報共有にSPIやUARTなどよりも耐ノイズ性や送信線をより長くできるCANを使ってみてはいかがでしょうか？

「枯れた技術」だからこそ奥が深い！まずはLチカの次に『CAN通信』を始めてみませんか？