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 2025 Nagoya 25/05/31

Sunny Giken

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「とぱめ」

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)
ビジネス開発部　テクノベート課　副課長

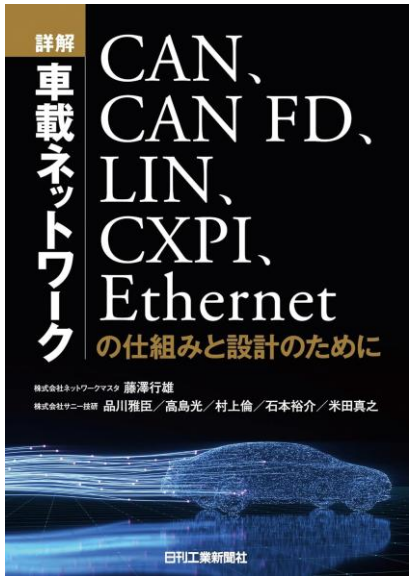
神戸生まれ大阪育ちのトラキチ。小学校1年でMSXのBASICに触れ、父のお下がりのポケコンでプログラミングの楽しさを知る。

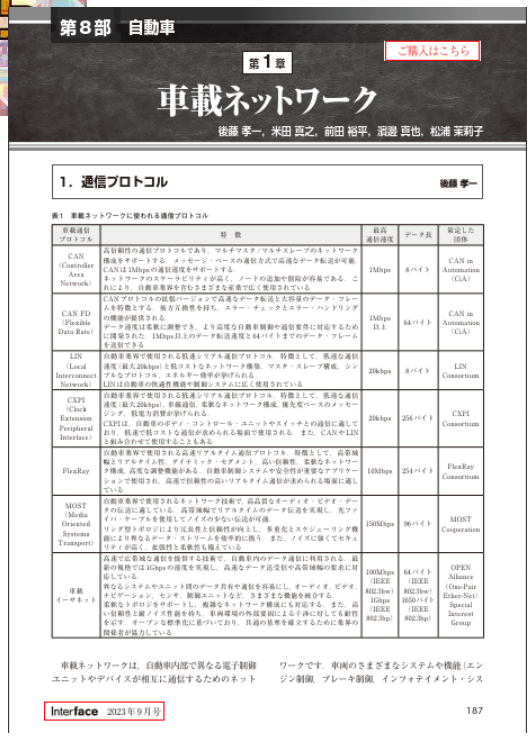
初めての自分用PCはPC-9821(当時Windowsはなく、MS-DOS v5.0)。CONFIG.SYSをいじくりまわし、メインメモリ640KBを如何に空けるのかに没頭
高校からDelphi(Object Pascal)に触れ、卒論のプログラムもDelphiで製作。**今でも言語仕様はDelphiが一番好き。**

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





Sunny Giken

自己紹介

- 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

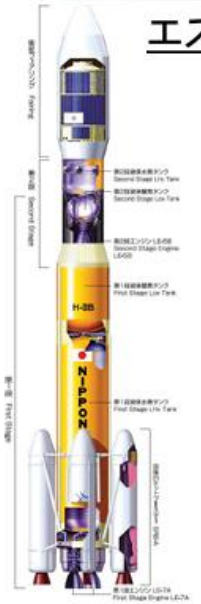
開発成果物の主な利用事例

エスクード（スズキ）

スカイラインハイブリッド（日産）

IPSiO GX e3300（リコー）

UA-101（Roland）



H-IIB（JAXA）

Cell³iMager duos（SCREENホールディングス）

OSP-P300（オークマ）

SoftBank 945SH（シャープ）

PM-A970(エプソン)

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]
  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
> エアコン・電子制御AT・ABS・サスペンション・パワーウィンドウ・電動シート・キーレスエントリー・電動ミラー・エアバッグ・パワーステアリング、エアコン、故障診断、最近ではADASも
> →現在では**100を超える**ECUが載ったものも・・・

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

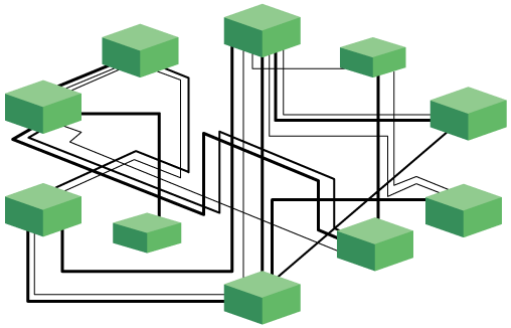
ネットワークならEthernetでええやん？ → オーバースペック
→車載ECUは少しでも小さく、1円でもコストカットしていくことが重要
→ECUはバッテリー容量の取り合いでもある
小サイズ・省コスト・省電力・高信頼性のわがままネットワークが必要！

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。

CANって何だ！？

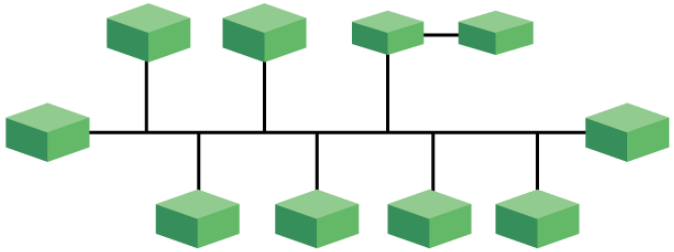
- **Controller Area Networkの略**
  1983年ドイツの電装メーカーのBOSCH社が自動車の電子制御システム向けの通信プロトコルとして提唱。**ワイヤーハーネスを削減**、複数のLANを介して通信する。

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、車内空間が広くなった

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！？

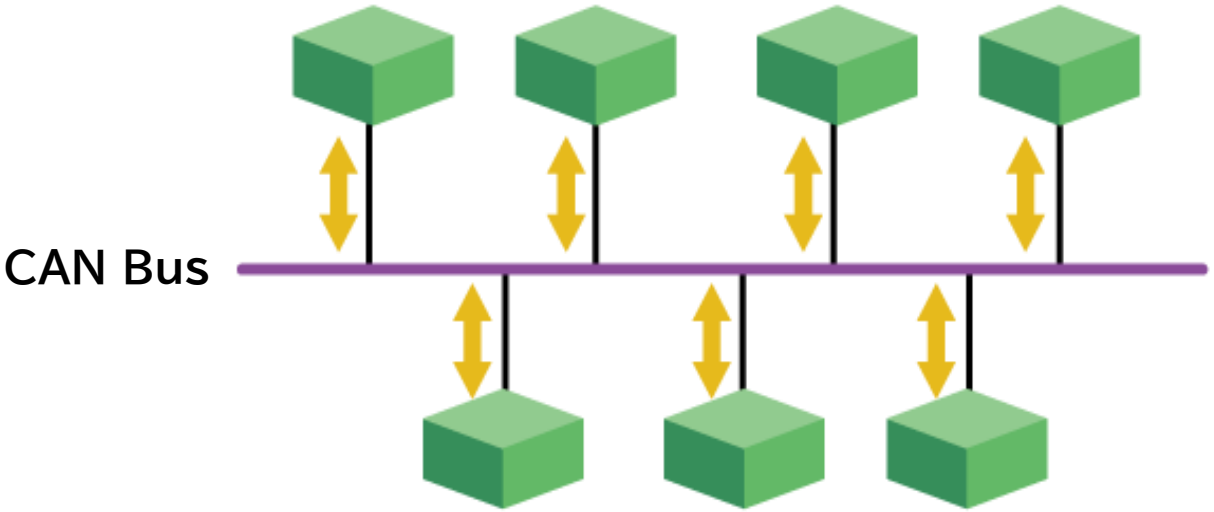
CANの主な特徴

- 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

これらのうちCANが自動車で使われてい理由となる特徴や仕様について厳選して紹介

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ので**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

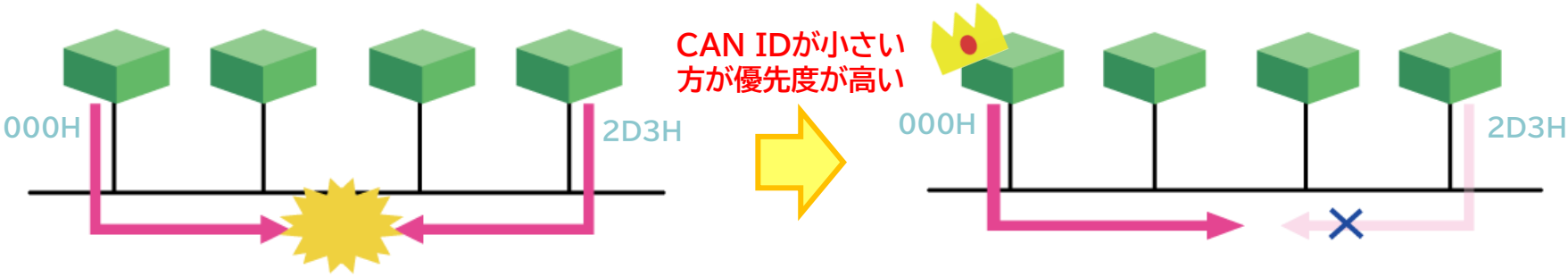


500kbpsならば
最大バス長は100m
※スタブ長は0.3m

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

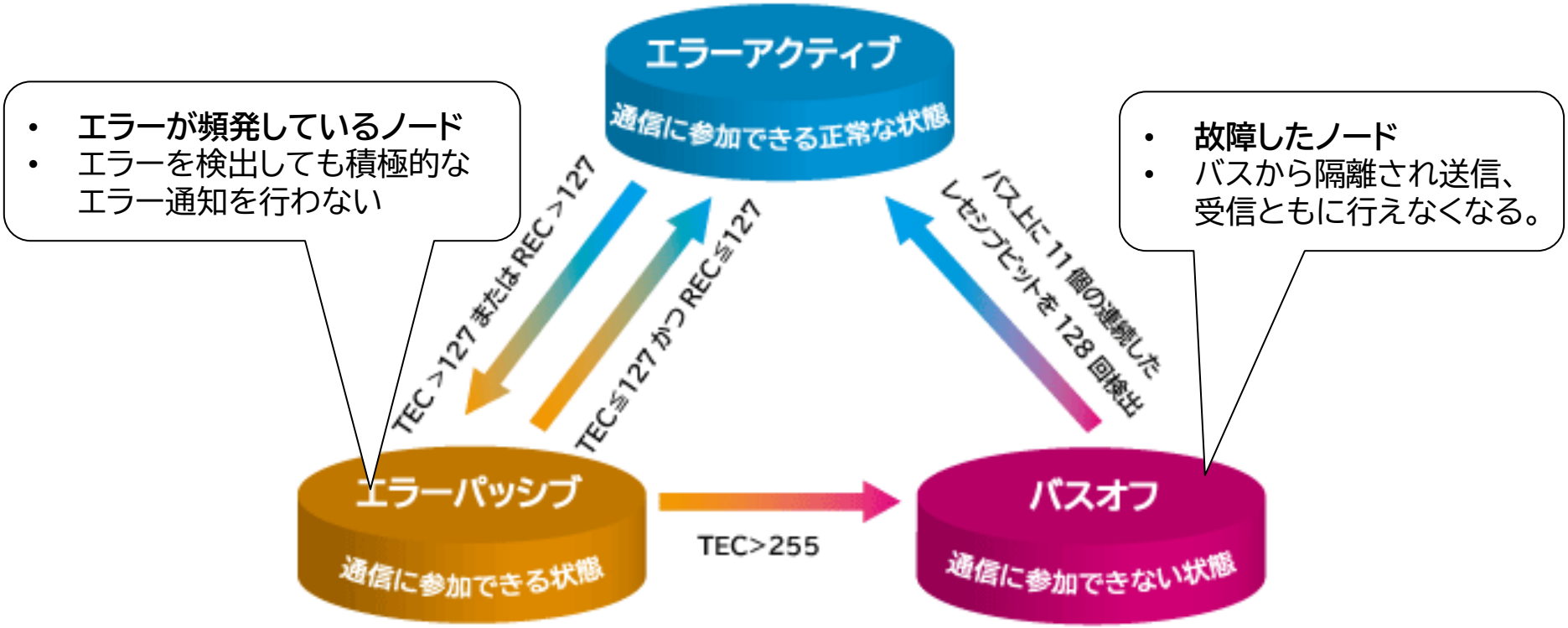


メッセージが衝突しそうになった時には、優先度に従って通信調停が行われる
→車体制御など、よりリアルタイム性を求める情報のCAN IDを小さいものにすることで、**低遅延でのデータ到達を保証**する

Sunny Giken

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

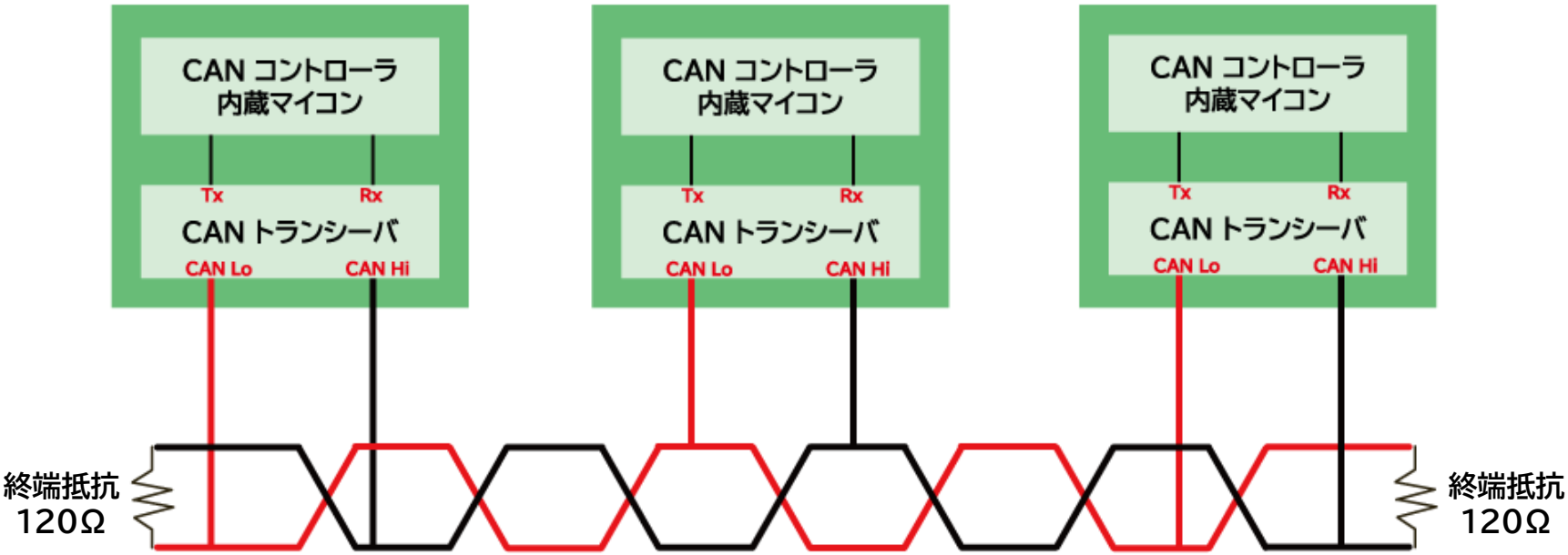


エラーを多発するノードは通信から隔離することで、他のノードの通信を妨げない

Sunny Giken

CANって何だ！？

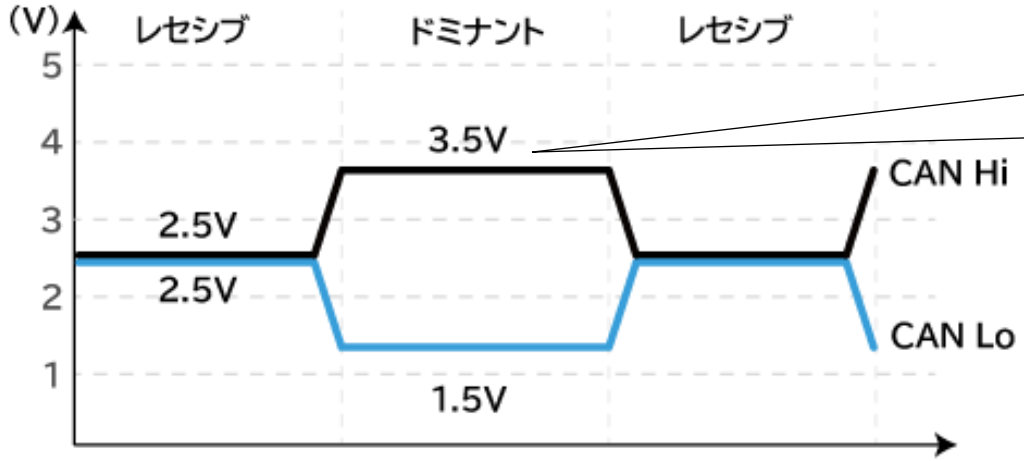
- **CANの特徴：伝送路**
  CAN Hi、CAN Loとよばれる2本の通信線を使って、送信受信の両方を行っており、配線の両端に120Ωの終端抵抗を2つ設置する必要がある。

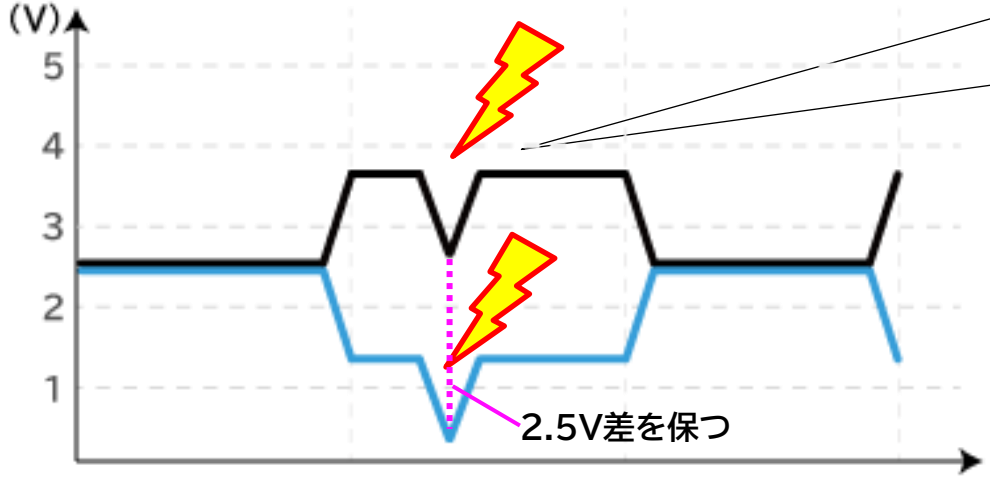


ISO11898-2（Highspeed CAN）の接続イメージ

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
  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

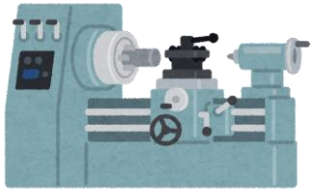
電位差をとるので
ノイズが打ち消され、
誤動作しにくい
高耐ノイズ性

ツイストペア線

Sunny Giken

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



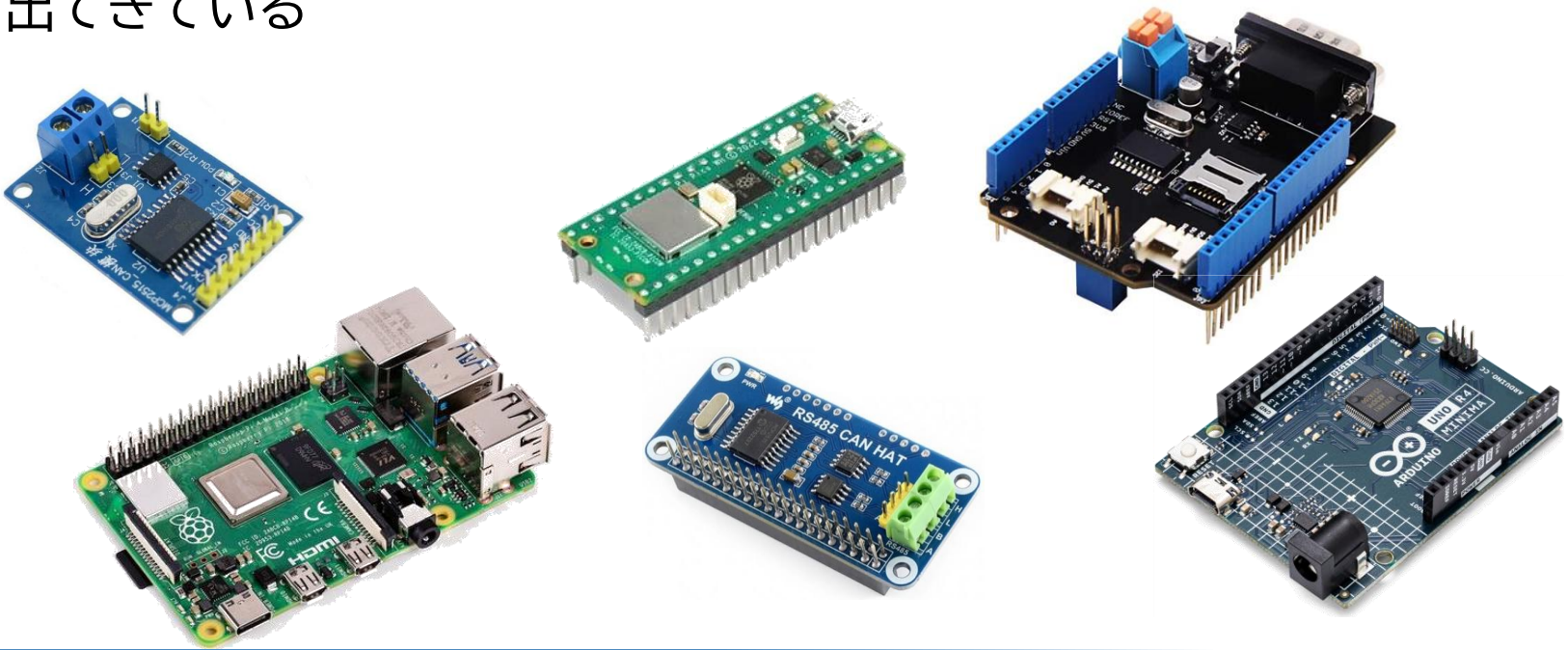

  - CANを応用したプロトコル(一部)
    - J1939
      トラック、バス、建機車両
    - ISOBUS
      農業機械(トラクタと作業機の通信共通化)
    - CANopen / DeviceNet
      モータ、バルブ、センサ、エンコーダ、表示機、
      シーケンサ、PLC etc…


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！

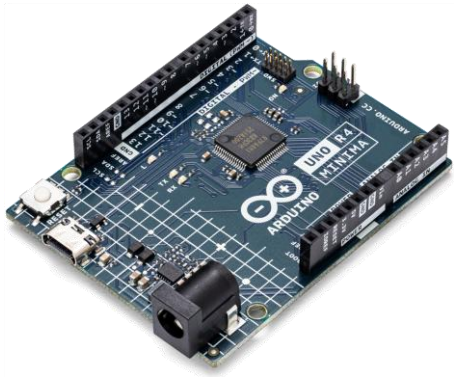
  自動車向けに生まれたCAN
  →車載用マイコンに搭載、個人では入手しづらい
  　専門のH/W知識、実装のノウハウが必要

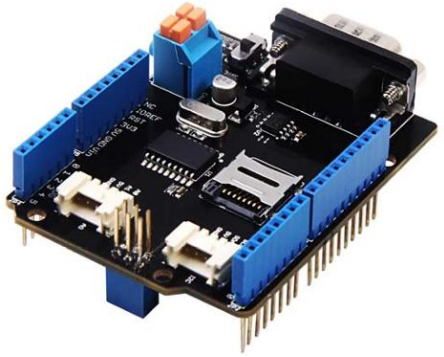
  最近では個人で購入可能なラズパイ用CAN拡張ボード（HAT）やそもそもCANが機能として載っているマイコンが搭載されたSBCなど、出てきている



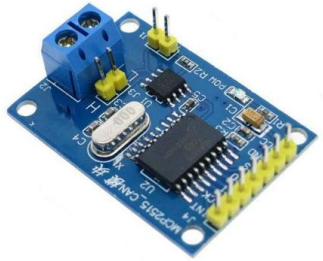
使ってみよう！

- 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



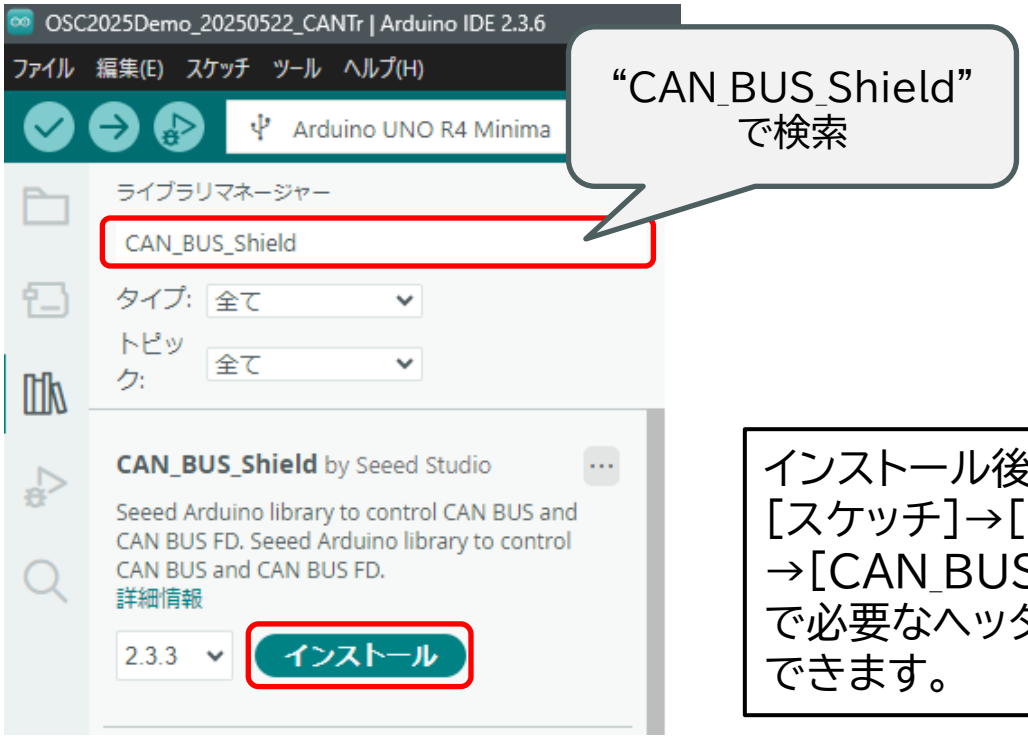
で通信を行い、CAN解析ツールでモニタリング
それぞれの実装について簡単に解説

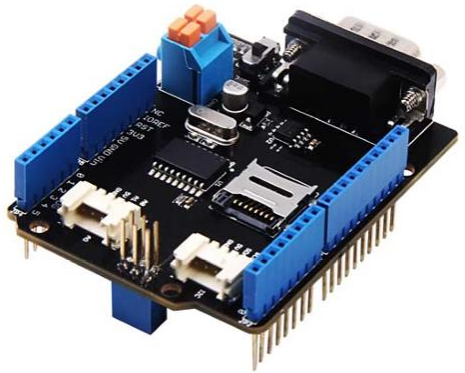
小型車のCANと同等の約20mの配線長で通信させています。
高級車なら50m
バスなどは100mも

Sunny Giken

使ってみよう！

- Auduino 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」をと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！

- Auduino 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.sendMsgBuf(0x100, 0, sizeof(tx_Data_a), tx_Data_a);

// 受信
byte rx_Data[8];
byte rxLen;
can.readMsgBuf(&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	int socket_fd; //ディスクリプタ
	struct sockaddr_in addr; //Ethernet構造体
		int socket_fd; //ディスクリプタ
		struct sockaddr_can addr; //CANデータ構造体

ソケット生成	if((socket_fd = socket(PF_INET, SOCK_DGRAM, 0)) < 0) { /* エラー処理 */ }	if((socket_fd = socket(PF_CAN, SOCK_RAW, CAN_RAW)) < 0) {/* エラー処理 */ }

NIC指定	strcpy(ifr.ifr_name,"eth0");
	if(ioctl(socket_fd,SIOCGIFINDEX, &ifr) < 0)
	{/* エラー処理 */ }
		strcpy(ifr.ifr_name, "can0");
		if(ioctl(socket_fd, SIOCGIFINDEX, &ifr)<0)
		{/* エラー処理 */ }

設定	memset(&addr, 0, sizeof(addr));
	addr.sin_family = AF_INET;
	addr.sin_addr.s_addr = htonl(INADDR_ANY);
	addr.sin_port = htons(servPort);
		memset(&addr, 0, sizeof(addr));
		addr.can_family = AF_CAN;
		addr.can_ifindex = ifr.ifr_ifindex;

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"
```

応答

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

can0がCANインターフェースとして認識されている

  - 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  socketfd; /* 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 */
socketfd = socket(PF_CAN, SOCK_RAW, CAN_RAW);

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

/* ソケットと設定をバインド */
bind(socketfd, (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

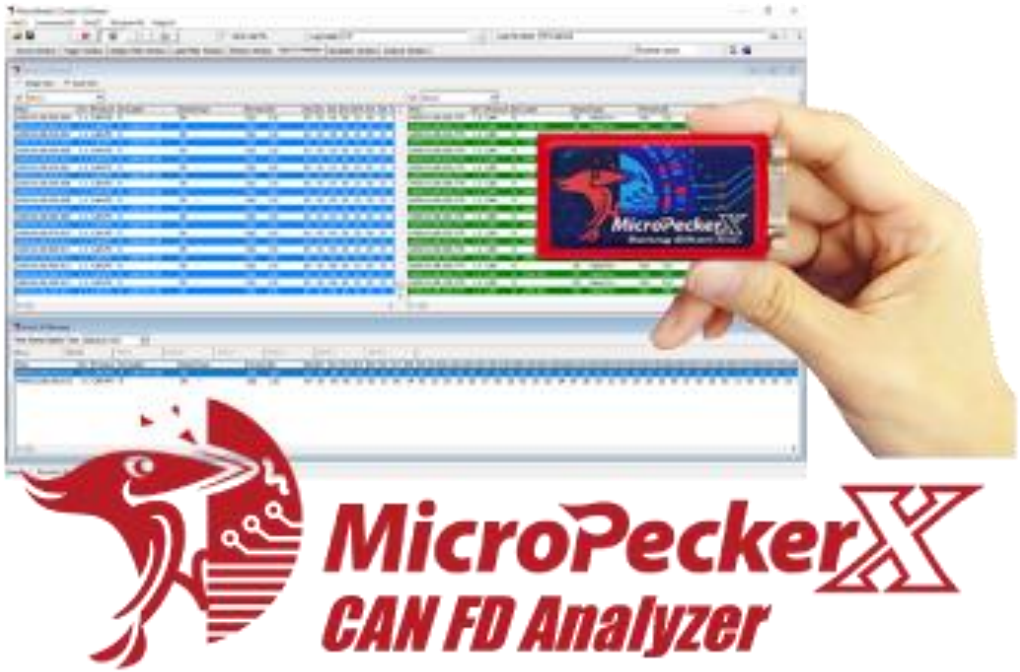
```
/* CAN ID */
frame.can_id = 0x100;
/* データ長 */
frame.can_dlc = 8;

/* 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

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、高信頼性が重要
    CANはそれに適う通信として生まれたもの
    最近はセキュリティも重要となってきている
- CANは自動車に限らずあらゆる分野で活用・応用されている
  - 自動車で熟れた技術が他分野に
- 最近では個人でもCANが使える道具がたくさんそろってきている
  - ハードウェア、ソフトウェア共に簡単に扱える
  - オープンソースも

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？