

# How To Use AGL CAN Signal

---

**05/31/2017**  
**Yuichi Kusakabe**  
**SS Engineering Group**  
**Fujitsu TEN LIMITED**

- Yuichi Kusakabe (Fujitsu TEN LIMITED)
- Software Engineer of IVI about 10 years  
(for 16-bit and 32-bit architecture)
- Linux Software Engineer(2011–2013)
- Linux Software Lead Engineer(2013–Now)
- BSP Porting/Customizing
- Supporting for in-house software developers
- AGL(Automotive Grade Linux) Advisory Board member

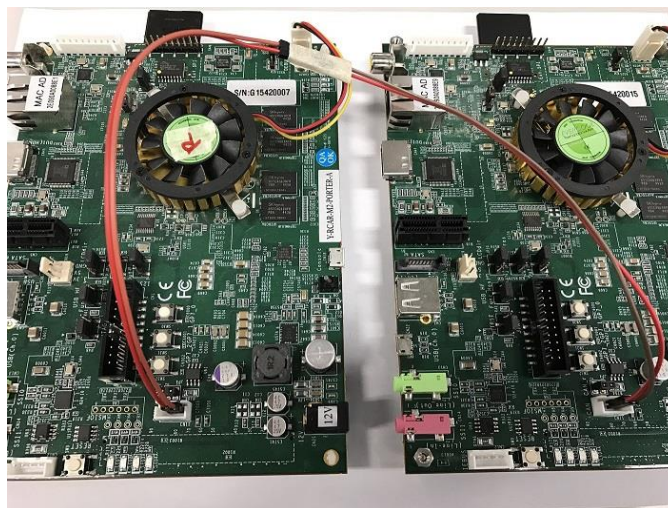


- **What's CAR CAN signal**
- **Standard Linux CAN IF & OSS CAN Tool**
- **How to use CAN signal to AGL**
- **Demonstration & Results**
- **Conclusion**

# What's CAR CAN Signal

**Standard CAN Signal is Low Speed (500kbps) , But High frequency (\*\*us) .**

- Standard CAN Signal format(11bit).
  - Data line: D+/D-/GND(want)
  - Baud rate: 500kbps
  - CAN ID: 11bit(0x000~0x7FF)
  - Data size: 0~8byte
  - CAN Bus load: 20~75%



# Standard Linux CAN IF & OSS CAN Tool

## Linux kernel all ready CAN IF with Socket CAN

### SocketCAN

From Wikipedia, the free encyclopedia

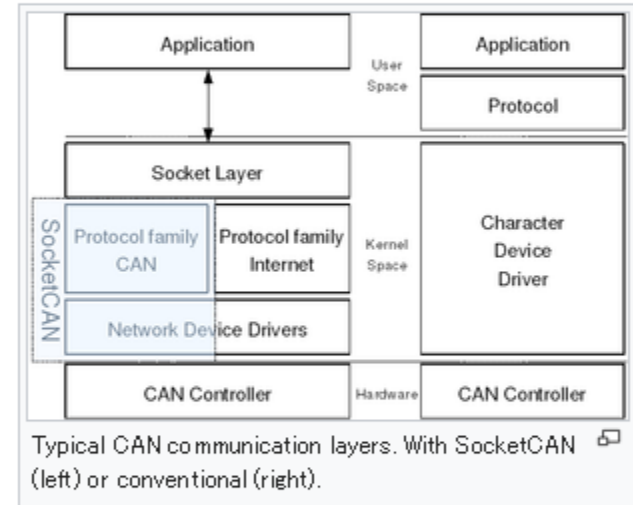
**SocketCAN** is a set of [open source CAN](#) drivers and a networking stack contributed by [Volkswagen Research](#) to the [Linux kernel](#). Formerly known as *Low Level CAN Framework* (LLCF).



Traditional CAN drivers for Linux are based on the model of character devices. Typically they only allow sending to and receiving from the CAN controller. Conventional implementations of this class of device driver only allow a single process to access the device, which means that all other processes are blocked in the meantime. In addition, these drivers typically all differ slightly in the interface presented to the application, stifling portability. The SocketCAN concept on the other hand uses the model of network devices, which allows multiple applications to access one CAN device simultaneously. Also, a single application is able to access multiple CAN networks in parallel.

The SocketCAN concept extends the [Berkeley sockets](#) API in Linux by introducing a new protocol family, PF\_CAN, that coexists with other protocol families like PF\_INET for the [Internet Protocol](#). The communication with the CAN bus is therefore done analogously to the use of the Internet Protocol via sockets. Fundamental components of SocketCAN are the network device drivers for different CAN controllers and the implementation of the CAN protocol family. The protocol family, PF\_CAN, provides the structures to enable different protocols on the bus: Raw sockets for direct CAN communication and transport protocols for point-to-point connections. Moreover the broadcast manager which is part of the CAN protocol family provides functions e.g. for sending CAN messages periodically or realize complex message filters.

Patches about CAN were added in the 2.6.25 [Linux kernel](#). Meanwhile some controller drivers were added and work is going on to add drivers for a variety of controllers.



## Linux kernel all ready CAN IF with Socket CAN

Readme file for the Controller Area Network Protocol Family (aka SocketCAN)

This file contains

- 1 Overview / What is SocketCAN
- 2 Motivation / Why using the socket API
- 3 SocketCAN concept
  - 3.1 receive lists
  - 3.2 local loopback of sent frames
  - 3.3 network problem notifications
- 4 How to use SocketCAN
  - 4.1 RAW protocol sockets with can\_filters (SOCK\_RAW)
    - 4.1.1 RAW socket option CAN\_RAW\_FILTER ←
    - 4.1.2 RAW socket option CAN\_RAW\_ERR\_FILTER
    - 4.1.3 RAW socket option CAN\_RAW\_LOOPBACK
    - 4.1.4 RAW socket option CAN\_RAW\_RECV\_OWN\_MSGS
    - 4.1.5 RAW socket option CAN\_RAW\_FD\_FRAMES
    - 4.1.6 RAW socket option CAN\_RAW\_JOIN\_FILTERS
    - 4.1.7 RAW socket returned message flags
  - 4.2 Broadcast Manager protocol sockets (SOCK\_DGRAM)
    - 4.2.1 Broadcast Manager operations
    - 4.2.2 Broadcast Manager message flags
    - 4.2.3 Broadcast Manager transmission timers
    - 4.2.4 Broadcast Manager message sequence transmission
    - 4.2.5 Broadcast Manager receive filter timers
    - 4.2.6 Broadcast Manager multiplex message receive filter
    - 4.2.7 Broadcast Manager CAN FD support
  - 4.3 connected transport protocols (SOCK\_SEQPACKET)
  - 4.4 unconnected transport protocols (SOCK\_DGRAM)
- 5 SocketCAN core module
  - 5.1 can.ko module params
  - 5.2 procs content
  - 5.3 writing own CAN protocol modules

```
CONFIG_CAN=y
CONFIG_CAN_RAW=y
CONFIG_CAN_BCM=y
CONFIG_CAN_GW=y
CONFIG_CAN_RCAR=y
```



## can-utils easy to debug CAN Signal (read/write/play)

linux-can / can-utils Unwatch 60 Star 229 Fork 110

Code Issues 1 Pull requests 2 Projects 0 Wiki Pulse Graphs

Linux-CAN / SocketCAN user space applications

300 commits 2 branches 0 releases 18 contributors

Branch: master New pull request Create new file Upload files Find file Clone or download

jihochu committed with hartkopp bcmserver: allow CAN netdevice names greater than 6 characters Latest commit 99f1664 19 days ago

config/m4	add autotools infrastructure	7 years ago
include/linux	bcm: add support for CAN FD frames	4 months ago
.gitignore	gitignore: add isotpperf	2 years ago
Android.mk	can-utils: added isotpperf tool for performance measurements	2 years ago
GNUmakefile.am	configure: switch to new libtool-2.0 macro	2 years ago
Makefile	can-utils: added isotpperf tool for performance measurements	2 years ago
README.md	Create README.md	a year ago
asc2log.c	janitorial: asc2log: properly close infile	2 years ago
autogen.sh	do not use --symlink for autoreconf	3 years ago

## CAN data send (cansend)

ID=333(11bit), DATA=33 send=can0

# **cansend can0 333#33**

ID=00004444(24bit), DATA=44 send=can0

# **cansend can0 00004444#44**

## CAN data recv (candump)

recv=can

# candump can0 -ta

root@porter:~# **candump can0 -ta**

(1478869757.430017) can0 344 [8] FF EE 00 00 00 00 EE AA

(1478869757.431290) can0 226 [8] E4 00 00 EE 00 EE EE 00

recv=all

# **candump any -ta**

# How to use CAN signal to AGL

This time AGL provide AMB, but AGL remake new CAN Signal handing FW.

otcshare / automotive-message-broker

Watch 25 Star 33 Fork 34

Code Issues 3 Pull requests 0 Projects 0 Wiki Pulse Graphs

No description, website, or topics provided.

1,277 commits 22 branches 352 releases 12 contributors

**This is OLD software**

ambd	fixed the warning message when enabling -Wall option	a month ago
docs	[amb.fidl] fix Ignition type and some whitespace issues	6 months ago
examples	Added chrony sink plugin	2 years ago
lib	fixed the warning message when enabling -Wall option	a month ago
packaging.in	[PACKAGING] add unpackaged files into rpm package	6 months ago
plugins	fixed the warning message when enabling -Wall option	a month ago
tests	Merge pull request #49 from tripzero/master	2 years ago
tools	fixed the warning message when enabling -Wall option	a month ago
xwalk	[bluemonkey] websocket server implemented	2 years ago
CMakeLists.txt	fixed the warning message when enabling -Wall option	a month ago
COPYING	added COPYING file with license	5 years ago

<https://github.com/otcshare/automotive-message-broker>

## Low level CAN service made to decode and write on CAN bus

LINUX FOUNDATION COLLABORATIVE PROJECTS



[Account signup](#) | [Wiki](#) | [Jira](#) | [Doors](#) | [Gerrit](#) | [Jenkins](#) | [Mailing lists](#) |

[Code Review](#) / [apps](#) / [low-level-can-service.git](#) / [summary](#)

[summary](#) | [shortlog](#) | [log](#) | [commit](#) | [commitdiff](#) | [review](#) | [tree](#)

? search:

**description** Low level CAN service made to decode and write on CAN bus.

**owner** Gerrit Service User

**last change** Thu, 25 May 2017 01:38:54 +0900 (18:38 +0200)

**URL** <https://gerrit.automotivelinux.org/gerrit/p/apps/low-level-can-service.git>

<ssh://kusakabe@gerrit.automotivelinux.org:29418/apps/low-level-can-service.git>

### shortlog

2 days ago Romain Forlot **Fix: file already exists on build demo app** [master](#) [3.99.1](#) [dab/3.99.1](#) [dab\\_3.99.1](#) [commit](#) | [commitdiff](#) | [tree](#) | [snapshot](#)

2 days ago Romain Forlot **Update config.cmake path** [commit](#) | [commitdiff](#) | [tree](#) | [snapshot](#)

2 days ago Romain Forlot **Cmake WIP** [commit](#) | [commitdiff](#) | [tree](#) | [snapshot](#)

2 days ago Romain Forlot **Move and update app-templates submodule** [commit](#) | [commitdiff](#) | [tree](#) | [snapshot](#)

3 days ago Romain Forlot **Close diagnostic manager socket if there isn't any...** [sandbox/claneys/import](#) [commit](#) | [commitdiff](#) | [tree](#) | [snapshot](#)

3 days ago Romain Forlot **Initialize the new socket member.** [commit](#) | [commitdiff](#) | [tree](#) | [snapshot](#)

3 days ago Romain Forlot **Fix memory leaks** [commit](#) | [commitdiff](#) | [tree](#) | [snapshot](#)

3 days ago Romain Forlot **Be able to copy active diagnostic request objects with...** [commit](#) | [commitdiff](#) | [tree](#) | [snapshot](#)

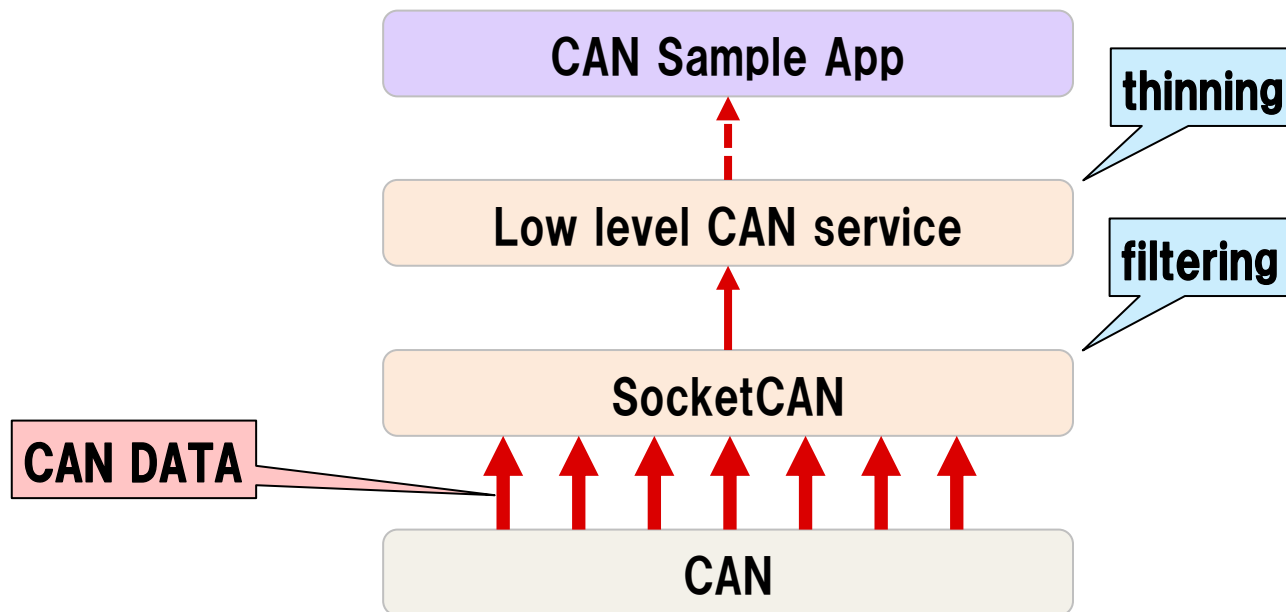
3 days ago Romain Forlot **Static code review fixes.** [commit](#) | [commitdiff](#) | [tree](#) | [snapshot](#)

3 days ago Romain Forlot **Fix a dependency problem with populate htdocs targets.** [commit](#) | [commitdiff](#) | [tree](#) | [snapshot](#)

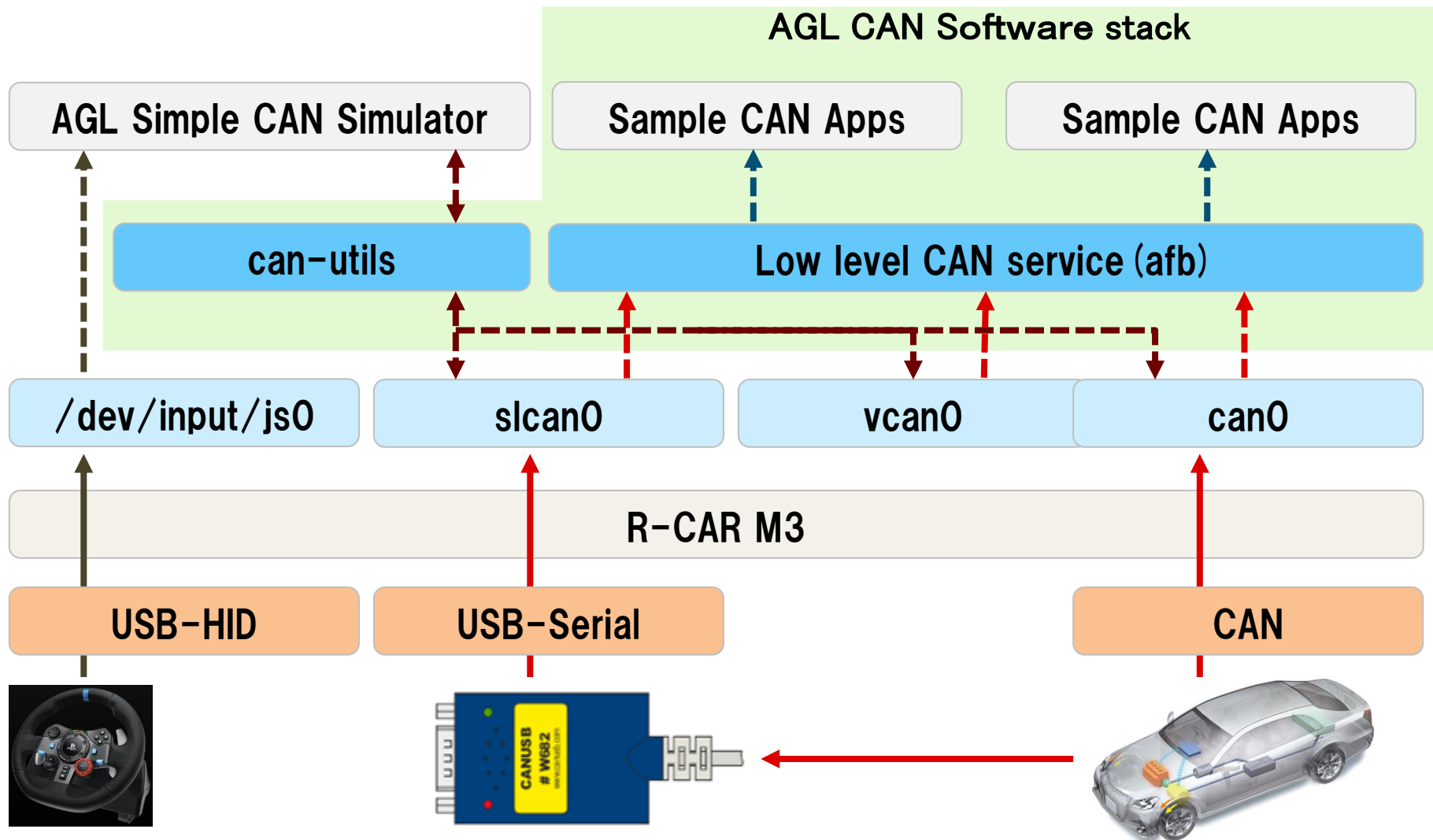
<https://gerrit.automotivelinux.org/gerrit/gitweb?p=apps/low-level-can-service.git;a=summary>

**CAN ID Filtering and CAN DATA Thinning out is very important**

- CAN signal filtering(setting SocketCAN)
  - CAN ID xxx → xx
- CAN signal thinning out
  - CAN cycle xx ms → xxx ms
- CAN data convert AGL Standard



**AGL support easy to use CAN data**

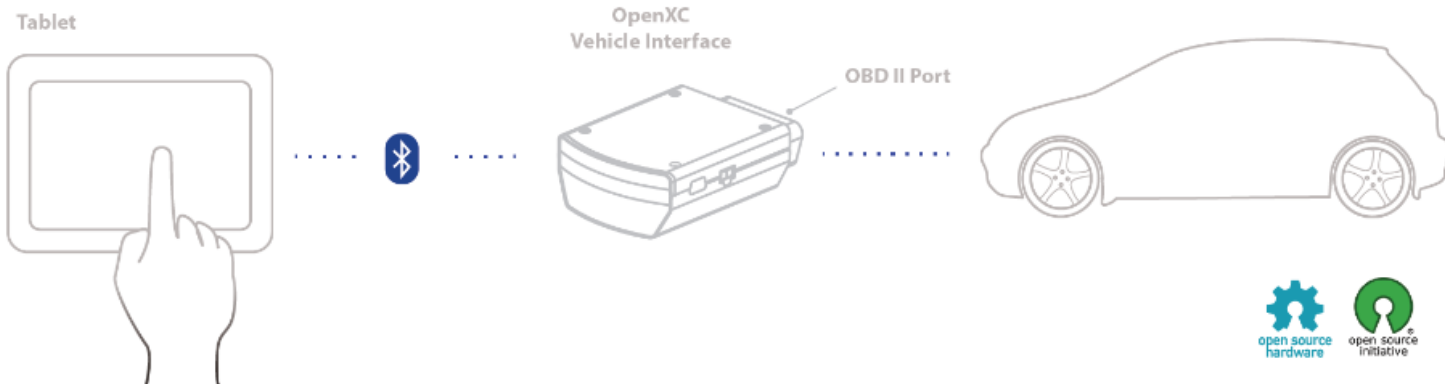


## The OpenXC Platform

OpenXC™ is a combination of open source hardware and software that lets you extend your vehicle with custom applications and pluggable modules. It uses standard, well-known tools to open up a wealth of data from the vehicle to developers, even beyond OBD-II.

### OpenXC

OpenXC is an open source hardware and software platform that lets you extend your vehicle with custom applications and pluggable modules.



### What is OpenXC™?

OpenXC™ is an open source, data-focused API for your car. By installing a small hardware module, the vehicle data becomes accessible to Android or other desktop applications using the OpenXC library.

[OpenXC Overview](#)

### Unlocking Rich Vehicle Data

OpenXC™ allows consumer devices, such as smart phones, to access data from any vehicle. Using OpenXC™, you can monitor many of the sensors on a vehicle, enabling new and innovative vehicle-centric applications. Some data is required by law and more can be unlocked with support from an automaker - or a little reverse engineering effort!

[See the Data](#)



## Official Signals

These signal names are a part of the OpenXC specification, although some manufacturers may support custom message names.

- steering\_wheel\_angle
  - numerical, -600 to +600 degrees
  - 10Hz
- torque\_at\_transmission
  - numerical, -500 to 1500 Nm
  - 10Hz
- engine\_speed
  - numerical, 0 to 16382 RPM
  - 10Hz
- vehicle\_speed
  - numerical, 0 to 655 km/h (this will be positive even if going in reverse as it's not a velocity, although you can use the gear status to figure out direction)
  - 10Hz
- accelerator\_pedal\_position
  - percentage
  - 10Hz
- parking\_brake\_status
  - boolean, (true == brake engaged)
  - 1Hz, but sent immediately on change

# 19 CAN Data

**Daimler AG**  
**Daimler Buses - EvoBus GmbH**  
**MAN Truck & Bus AG**  
**Scania AB**

**Scania CV**  
**Volvo Truck Corporation**  
**Volvo Bus Corporation**  
**Renault Trucks**

**Iveco SpA**  
**DAF Trucks N.V.**  
**VDL Bus & Coach B.V.**

## **FMS-Standard description**

**Version 03**

**14.09.2012**

**© HDEI / BCEI Working Group**

page	PGN	SPN	(signal) name e.g. mileage, fuel consumption	Mandatory Truck only	rep. rate in ms	remarks / comments	
7	65257	250	Engine total fuel used		1000	4 bytes, 0 to +2 105 540 607,5 L	
8	65276	96	fuel level 1	X (worldwide)	1000	1 Byte	
9	61444	513	Actual Engine - Percent Torque	X (worldwide)	20	1 % / Bit, +125 % offset	
9	61444	190	engine speed	X (worldwide)	20	2 Byte, 0-8031,875 rpm	
10	65253	247	Engine total hours of Operation	X (worldwide)	1000	4 bytes, 0 to 210 554 060,75 h	
11	65260	237	vehicle identification number	X (worldwide)	10000	variable, max 200 char.	
12	64977	2806	SW-version supported	X (worldwide)	10000	indicator for SW version supported	
12	64977	2804	Diagnostics supported	X (worldwide)	10000	indicator for diagnosis session support	
12	64977	2805	Requests supported	X (worldwide)	10000	indicator for request supported	
14	65217	917	High resolution total vehicle distance	X (worldwide)	1000	4 bytes, 0 - 21 055 406 km; without TCO	
15	65132	1611	Vehicle motion	X (EU)	20/50	With digital tachograph	
15	65132	1613	driver 2 working state	X (EU)	20/50	With digital tachograph	
15	65132	1612	driver 1 working state	X (EU)	20/50	With digital tachograph	
15	65132	1614	Vehicle overspeed		20/50	With digital tachograph	
15	65132	1617	Driver 1 time rel. states		20/50	With digital tachograph	
15	65132	1618	Driver 2 time rel. states		20/50	With digital tachograph	
15	65132	1615	Driver 1 card	X (EU)	20/50	With digital tachograph	
15	65132	1616	Driver 2 card	X (EU)	20/50	With digital tachograph	
15	65132	1619	Direction indicator		20/50	With digital tachograph	
15	65132	1620	Tachograph performance	X (EU)	20/50	With digital tachograph	
15	65132	1621	Handling information	X (EU)	20/50	With digital tachograph	
15	65132	1622	System event	X (EU)	20/50	With digital tachograph	
15	65132	1624	Tachograph vehicle speed	X (EU)	20/50	With digital tachograph - 2 bytes	
17	65262	110	engine coolant temperature	X (worldwide)	1000	-40° to 210°	
18	65269	171	Ambient Air Temperature	X (worldwide)	1000	0.03125 °C / Bit gain	
19	65131	1625/1626	Driver 1 / Driver 2 Identification	X (EU)	10000	If a driver ID is available the message is sent with a Broadcast Announce Message (BAM)	
20	65266	183	Fuel rate	X (worldwide)	100	0.05 L/h per bit, 0 to 3,212,75 L/h	
20	65266	184	Instantaneous Fuel Economy	X (worldwide)	100	1/512 km/L per bit, 0 to 125,5 km/L	
21	65198	1087	Service Brake Air Pressure Circuit #1	X (worldwide)	1000	8 kPa/Bit, 0 offset	
21	65198	1088	Service Brake Air Pressure Circuit #2	X (worldwide)	1000	8 kPa/Bit, 0 offset	
22	64777	5054	High resolution engine total fuel used		1000	0.001 L/bit, 0 to 4,211,081,215 L	
23	65110	1761	Aftertreatment 1 Diesel Exhaust Fluid Tank 1 Level		1000	0.4 %/bit, 0 % offset	
24	64893		FMS Tell Tale Status	X (worldwide ,EU) Not all tell tales	1000		
Truck only Section		Truck only Section		Truck only Section		Truck only Section	
27	65265	84	wheel based speed		100	may differ from TCO1	
27	65265	598	clutch switch		100	two bit status	
27	65265	597	Brake switch		100	two bit status	
27	65265	595	cruise control active		100	two bit status	
27	65265	976	PTO state		100	Either SPN 3948 (PTODE) or SPN 976 is sent	
29	61443	91	accelerator pedal position 1	X (worldwide)	50	1 Byte	
29	61443	92	Engine Percent Load At Current Speed	X (worldwide)	50	1 % / bit, 0 to 125 % operational range	
30	65258	928	Axle location		1000	-	
30	65258	928	Tire location		1000	-	
30	65258	582	Axle weight		1000	-	
32	65216	914	Service distance		1000	-	
33	64932	3948	At least one PTO engaged		100	Either SPN 3948 or SPN 976 (CCVS) is sent	
34	65136	1760	Gross Combination Vehicle Weight		10000	0 to 642,550 kg	
35	61440	900	Retarder Torque Mode		100	16 states/4 bit, 0 offset	
35	61440	520	Actual Retarder - Percent Torque		100	1 %/bit, -125 % offset	
35	61440	1716	Retarder Selection, non-engine		100	0.4 %/bit, 0 % offset	

# 56 CAN Data (include Bus)

[http://www.fms-standard.com/Truck/down\\_load/fms\\_document\\_ver03\\_vers\\_14\\_09\\_2012.pdf](http://www.fms-standard.com/Truck/down_load/fms_document_ver03_vers_14_09_2012.pdf)

# Demonstration & Results

## AGL reference Hardware to Renesas R-CAR M3



**CPU** R-Car M3搭載

- ARM@Cortex-A57 (ARMv8) 1.7 GHz dual core、with NEON/VFPv4、L1 cache I/D 48K/32K、L2 cache 2MB
- ARM Cortex-A53 (ARMv8) 1.3 GHz quad core、with NEON/VFPv4、L1 cache I/D 32K/32K、L2 cache 512K
- memory controller for LPDDR4 2GB in 2 channels、each 32-bit wide

- three-dimensional graphics engines
- video processing units
- 3 channels display output
- 6 channels video input
- SD card host interface
- USB3.0 and USB2.0 interfaces
- CAN interfaces
- Ethernet AVB
- PCI Express Interfaces

**メモリ**

- internal 384KBytes system RAM
- 64 MBytes HyperFlash™ (512 Mbits)
- 16MBytes QSPI flash(128 Mbits)1 header QSPI module
- 8 GBytes eMMC
- microSD card slot

## R-CarスタータキットがAutomotive Grade Linuxの標準リファレンスプラットフォームに採用、次世代コネクテッドカーのIVI開発を加速

～最新のUnified Code Base (UCB) 3.0の64ビットソフトウェア環境に対応～

2017年05月24日

ルネサス エレクトロニクス株式会社



R-CarスタータキットがAGLの標準リファレンスに採用

Infotainment) 用アプリケーションソフト  
R-Carスタータキットは、すでにAGLプロジェクト  
Unified Code Base (UCB) 3.0に対応して  
来の32ビット環境に比べて、コンテナ技術  
シームレスに車載向けへ応用可能となる最

ルネサス エレクトロニクス株式会社  
(代表取締役社長兼CEO：呉 文精、以下ルネサス) は、このたびR-Carスタータキットが、Linuxベースの車載情報機器のオープンソースプロジェクトAutomotive Grade Linux (AGL) のソフトウェア開発用標準リファレンスプラットフォームの1つに採用されたことを発表しました。これにより、同プロジェクトが開発したソフトウェアを動かすハードウェア環境を容易に入手可能となり、コネクテッドカー時代に向けて、IVI (In-Vehicle


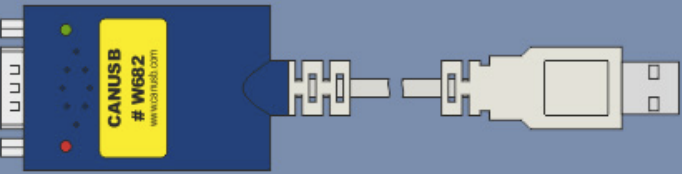
また、このたびR-Carスタータキットと組み合わせて使用するIVI開発用拡張ボード2種類 (シマフジ電機社製) が7月より発売されます。拡張ボードのスタンダードモデルは、マルチディスプレイや各種ネットワーク用インタフェースを搭載、さらにアドバンスドモデルは、最大8チャンネルのカメラ入力や高速/大容量のストレージを拡張できるインタフェースを装備しています。

## CANUSB easy connect CAN IF

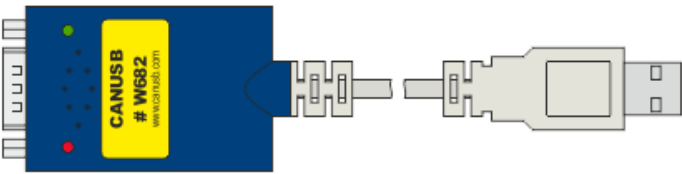
### CAN Tools

[Home](#)
[News](#)
[Products](#)
[Projects](#)
[Downloads](#)
[Where to Buy?](#)
[About](#)

---





### CANUSB




#### General Information:

CANUSB is a very small dongle that plugs into any PC USB Port and gives an instant CAN connectivity. This means it can be treated by software as a standard COM Port (virtual serial RS232 port) with the FTDI USB drivers which eliminates the need for any extra drivers (DLL) or by installing a direct driver DLL (D2XX) together with our CANUSB DLL for faster communications and higher CAN bus loads. Sending and receiving can be done in standard ASCII format.



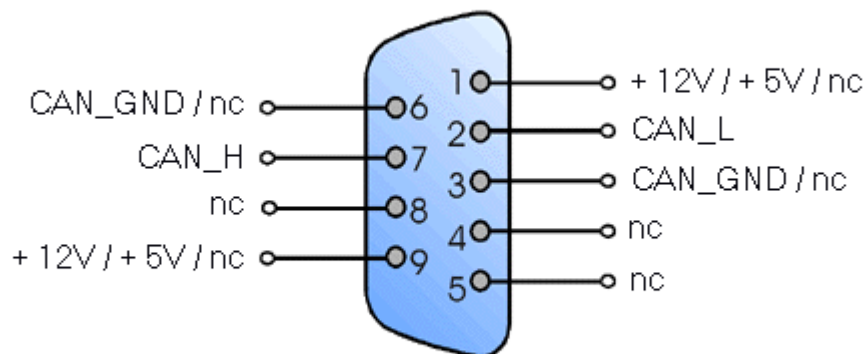
#### Menu

- > Home
- > News
- > Products
  - > CAN232
  - > CANUSB
- > Projects
- > Downloads
  - > CAN232 Download
  - > CANUSB Download
- > Where to Buy?
- > About



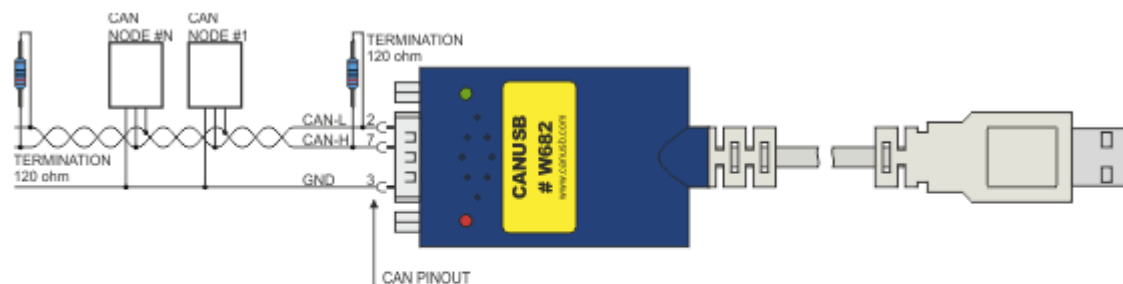
## CANUSB connected CAN IF simple Hardware

CAN Pin assignement:



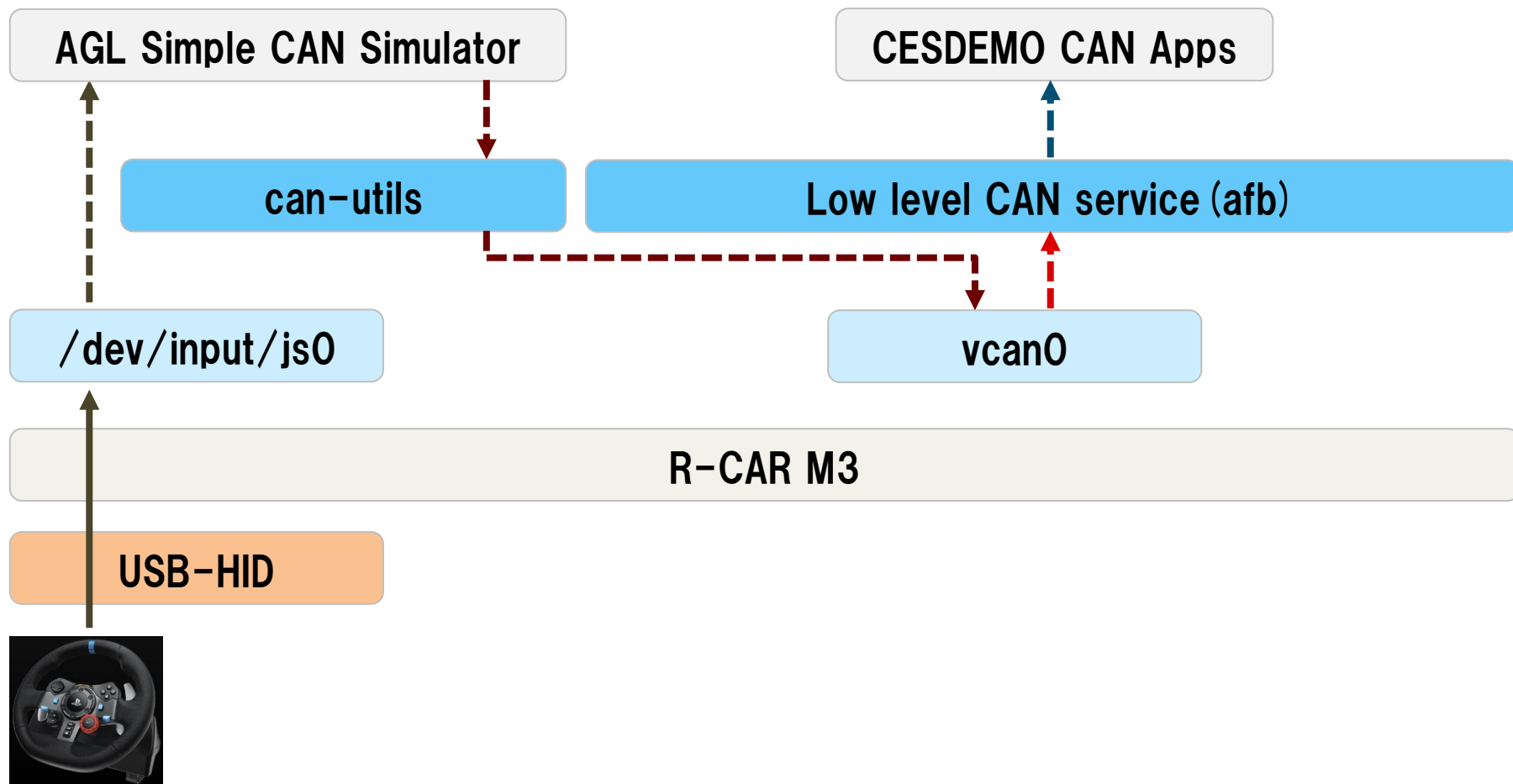
Pin assignement according to CiA recommendations DS102-1.

The CANUSB is powered from USB port, so no need to connect external power on pin 9. Use only CAN\_L (pin2), CAN\_H (Pin7) and CAN\_GND (pin3).



The picture above shows how to connect the CANUSB ([click here](http://www.can232.com/?page_id=16) for a larger view). No external power is needed, the CANUSB uses 5VDC/100mA from USB.





## Add Kernel defconfig CAN driver and CANUSB

```
CONFIG_CAN=y  
CONFIG_CAN_VCAN=y  
CONFIG_CAN_RCAR=y ← Renesas board only  
CONFIG_CAN_SLCAN=y  
CONFIG_USB_SERIAL=y  
CONFIG_USB_SERIAL_FTDI_SIO=y
```

## Add rootfs “can-utils” and “iproute2”

```
yocto local.conf  
IMAGE_INSTALL_append = “ can-utils iproute2”
```

## Setup CAN and CANUSB

```
CAN0  
ip link set can0 type can bitrate 500000  
ip link set can0 up
```

```
CANUSB  
slcand -o -s 6 -t hw /dev/ttyUSB*  
ip link set slcan0 up
```

## Start Low level CAN service (afb-daemon)

```
./afb-daemon --token=${AFB_CANIVI_TOKEN} --ldpaths=. |
--port=${AFB_CANIVI_PORT}- |
--rootdir=. ${SERVICE_VERBOSE}
```

## Setting CAN data receive

```
./afb-client-demo ws://localhost:5555/api?token=3210
```

```
canivi subscribe { "event" : "*" } <- receive all data
```

```
canivi subscribe { "event" : "VehicleSpeed" } <- receive Vehicle Speed only
```

# Demonstration

## ➤ Data used for measurement

➤ logtime = 1122 sec,

➤ CAN ID cnt = 129, datacnt = 1042673

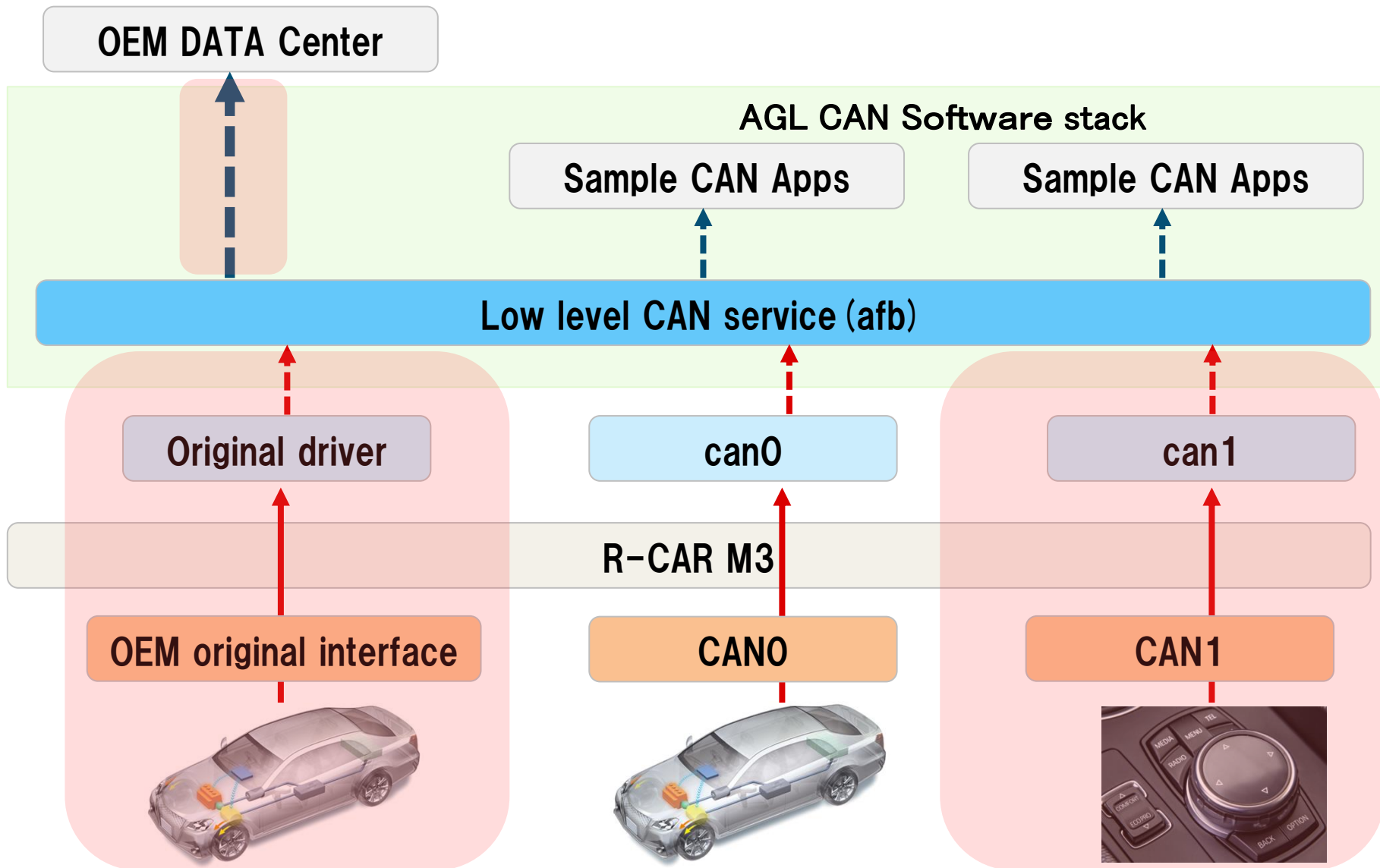
➤ can load ave= 19.04%, min = 18.82%, max = 37.89%

➤ cyc ave[us] = 929, min = 8, max = 8565

## ➤ Support CAN ID 42, Thinning out time \*\* → 100ms

	AMB(d-bus)		AFB (websocket)		AMB/AFB (vs)
	Process name	CPU load(%)	Process name	CPU load(%)	
Service	ambd	16.97	afb-daemon	1.34	17.4
	dbus-daemon	6.33			
Client(App)	AMBDBus_tp	4.26	afb-client-demo	0.82	5.2
CAN Sim	canplayer	2.36	canplayer	1.51	-
Total	*	30.65	*	3.67	8.3

**Support all in-vehicle communication (many CAN DATA and many Vehicle IF)**



- Linux Kernel all ready use to CAN
- OSS CAN Tool “can utils” is good software
- AGL support easy to use CAN data
- Next step
  - Define AGL public CAR CAN data format
    - With the cockpit architecture team
  - AGL simple CAN simulator provide
  - Support all in-vehicle communication

# Extra information



Qiita

Hot Markdownによる情報共有サ...
投稿する
ストック一覧 8


Fujitsu extended Advent Calendar 2016 | 1日目

22  
いいね

0  
コメント

## LinuxでCAN(車載通信)を動作させてみる

Linux 5310
can 14


yuichi-kusakabe
2016年12月22日に更新
2
投稿を編集
989views

ストック
⚙

この記事は、Fujitsu Advent Calender 2枚目の1日目の記事です。

## はじめに

組み込みLinux評価ボードでCAN(車載通信)を使用する方法について調べてみました。

CANには色々な規格がありますが、今回は一般的な500kビット/秒の標準フォーマット(11bit)を使用してみました(CANの詳細はwikipedia等を参照して下さい)。

## 1.ターゲットボードの用意

手頃な価格でCANを使用出来る組み込み評価ボードが無いかが調べてみましたが、SoCがCANに対応していても評価ボードにCAN IFが搭載されていないボードが大半です。今回はAutomotiveGradeLinuxの標準リファレンスボードに指定されているRenesas社のPorterボードを使用してみます。

ボードのCAN IFについては下記に詳しく説明が記載されていました。

Tweet
BI
1
G+
0

Like 12
Pocket
5



**yuichi-kusakabe**  
 22 Contribution

人気の投稿

- [LinuxでCAN\(車載通信\)を動作させてみる](#)

はじめに

- 1.ターゲットボードの用意
- 2.CANドライバについて
- 3.CAN送受信の確認ツールについて
- 4.CAN送受信テスト

おまけ

まとめ

<http://qiita.com/yuichi-kusakabe/items/e5b50aa3edb712bb6916>

# Thank you!!!

[yuichi.kusakabe@jp.fujitsu.com](mailto:yuichi.kusakabe@jp.fujitsu.com)