J1939Display

(Visualisierung der Betriebsdaten eines Blockheizkraftwerks
lokal und über das Internet)

Dokumentation

MHS Elektronik GmbH & Co. KG
Fuchsöd 4 ~ D-94149 Kößlarn
Tel: +49 (0) 8536/919 740 ~ Fax: +49 (0) 8536/919 738
Email: info@mhs-elektronik.de ~ Internet: www.mhs-elektronik.de

Version: 1.1 vom 02.12.2017

Inhaltsverzeichnis

1. Vorwort

Hard- und Software zur Überwachung von Blockheizkraftwerken.

Key Features:

- Hardware: Raspberry PI oder ein beliebiger Linux PC, Tiny-CAN Interface
- 7“ Touch Display: Lokale Anzeige der wichtigsten Motordaten
- Web Visualisierung: HTML5, Java, XML-Datenbank, RRDtool, SQLite
- TCP/IP: Modbus Interface
- Automatische Baudraten Erkennung
- Plug & Play: Automatische Erkennung und Initialisierung des USB-CAN Adapters
- CAN Trace Daten Aufzeichnung

J1939Display soll als Leitfaden für eigene Projekte unter Linux, speziell auf dem Raspberry PI dienen, bei dem die Visualisierung über das Internet nicht zu kurz kommt.

2. Hardware

Folgende Hardware Komponenten werden benötigt:

- Rasberry PI 3 Model B oder Rasberry PI 2 Model B Ver:.1.2
- microSDHC Speicherkarte min. 16GB mit „RASBIAN DESKTOP“ Betriebssystem
- Raspberry Pi Touch-LCD 7“
- Gehäuse z.B:
  ModMyPi Rasberry Pi 7“ Touch Display Gehäuse Schwarz, Art.-Nr. EB5684, Bezugsquelle: http://www.raspppishop.de
  Gehäuse für das RPi 7“ Touch-Display (transparent) Art.-Nr. 17556, Bezugsquelle: http://www.elektor.de
  CBRPP-TS-BLK/WHT-Geäuse für Rasberry Pi 7“ Touchscreen, Art.-Nr. 2494691, Bezugsquelle: http://www.farnell.de
- Tiny-CAN I-XL (Passendes USB-Kabel ist dabei) Art.-Nr. 700012, Bezugsquelle: http://www.mhs-elektronik.de
- AC/DC-Netzteil USB 5V/2A, Art.-Nr. 2456304, Bezugsquelle: http://www.farnell.de

Abbildung 1: „Rasberry PI 2 B“ mit “Pi Touch-LCD 7“ im „CBRPP-TS-BLK/WHT-Geäuse“ vorne die kleine Tiny-CAN I-XL Platine

3. Installation

Zuerst muss das Tiny-CAN Softwarepaket installiert werden, für den Betrieb der Software wird nur die „libmhstcan.so“ benötigt. „J1939Display“ sucht diese Datei im Verzeichnis „/opt/tiny_can/can_api“.

„tiny_can_raspberry_XXX.tar.gz“ von „http://www.mhs-elektronik.de“ downloaden. XXX durch die aktuellste Version ersetzen. Die Datei dann im „/opt“ Verzeichnis entpacken, nach dem entpacken kann das Archiv gelöscht werden. Zuvor müssen noch die Zugriffsrechte für das „/opt“ Verzeichnis gesetzt werden.

```
> sudo chmod -R 777 /opt
> cd /opt
> mv /home/pi/tiny_can_raspberry_XXX.tar.gz .
> tar -xzvf  tiny_can_raspberry_XXX.tar.gz
> rm tiny_can_rasberry_XXX.tar.gz
```

Die Tiny-CAN API kompilieren:

```
> cd /opt/tiny_can/can_api/src/mhstcan/linux
> make
> mv libmhstcan.so ../../..
```

Die Lib ist dem Paket bereits enthalten, in der Regel ist das kompilieren nicht notwendig.

„Git“ installieren:

```
> sudo apt-get install git
```

„J1939Display“ von „Git-Server“ holen:

```
> cd /opt
> git clone http://github.com/MHS-Elektronik/J1939Display.git
```

Benötigte Lib´s installieren:

```
> sudo apt-get install libmodbus5 libsqlite3-0 rrdtool apache2
```

Development Pakete installieren:

```
> sudo apt-get install gtk2.0-dev libmodbus-dev libxml2-dev libsqlite3-dev
```

Nur erforderlich wenn die Applikation kompiliert werden soll.

„J1939Display“ kompilieren:

```
> cd /opt/J1939Display/linux
> make
```

Dem Paket ist „J1939Display“ als Binary enthalten, ein Kompilieren ist für den Rapberry PI nicht notwendig.

Web-Page einrichten:

```
> cd /var/www
> sudo chown -R pi html
> sudo chgrp -R pi html
> cd html
> cp -R /opt/J1939Display/www/* .
```

J1939Display starten:

```
> cd /opt/J1939Display/linux/bin
> ./J1939Display
```

Fahren Sie nun mit Kapitel 4. Tipps und Tricks fort um ihrer Installation noch den letzten Schliff zu geben.

4. Tipps und Tricks

Das Programm Automatisch starten

Sollte keine Maus oder Tastatur an dem PI angeschlossen sein ist es sinnvoll das Programm automatisch zu starten. Kopieren Sie dazu die Datei „J1939Display.desktop“ aus dem Verzeichnis „tools“ ins Verzeichnis „/etc/xdg/autostart“.

```
> sudo cp /opt/J1939Display/tools/J1939Display.desktop /etc/xdg/autostart
```

Bildschirmschoner abschalten

Die Datei "/etc/lightdm/lightdm.conf" im Editor als "admin" öffnen.

```
> sudo leafpad /etc/lightdm/lightdm.conf
```

In der Sektion "SeatDefaults" die Zeile "xserver-command" wie folgt abändern oder wenn nicht vorhanden ergänzen.

```
[SeatDefaults]
....
xserver-command=X -s 0 -dpms
....
```

Bildschirm-Anzeige drehen

Die Datei "/boot/config.txt" im Editor als "admin" öffnen.

```
> sudo leafpad /boot/config.txt
```

In die Datei folgende Zeile eintragen

```
lcd_rotate=2
```

Nach dem Neustart sollte das Bild um 180° gedreht sein und Du kannst das Display umdrehen, womit sich der microUSB Stecker oben befindet.

Maus-Zeiger verschwinden lassen

Für unsere spezielle Anwendung ist der Mauszeiger störend. Um den Cursor ganz einfach zu entfernen, können wir ein Paket installieren, welches ihn ausblendet:

```
sudo apt-get install unclutter
```

Nach einem Neustart ist der Cursor nicht mehr sichtbar.

5. J1939 Protokoll

Das Protokoll wurde von SAE (Society of Automotive Engineers) definiert und arbeitet mit 29-Bit-Identifier (CAN 2.0B) auf dem Physical Layer mit CAN-Highspeed nach ISO 11898. Bei J1939 handelt es sich um ein Multimaster-System mit dezentralisiertem Netzwerk-Management ohne kanalbasierte Kommunikation. Der überwiegende Teil der Kommunikation erfolgt meist zyklisch und kann von allen Steuergeräten ohne explizite

Anforderung von Daten empfangen werden (Broadcast). Unsere Software macht sich das zunutze und horcht die bnötigeten Daten rein passiv am Bus ab.

Die PGN ist eine in der SAE J1939-Norm definierte Nummer, die mehrere Signale zu einer PDU (Protokoll Data Unit) zusammenfügt. Die PGN ist Teil des CAN-Identifiers.

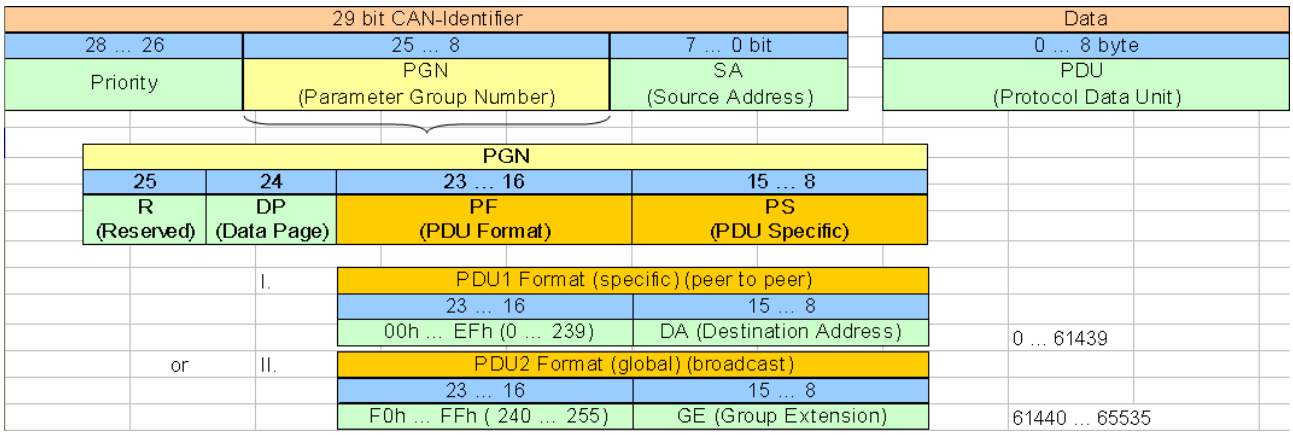

Abbildung 2: Aufbau J1939 CAN-Frame *Quelle: Wikipedia*

Beispiel: PGN 65262 (0xFEEE) – ET1 Engine Temperatur 1
Interpretation der 8 Byte CAN Daten, PDU:

Byte	Signal Name
1	Engine Coolant Temperature
2	Engine Fuel Temperature 1
3, 4	Engine Oil Temperature 1
5, 6	Engine Turbocharger Oil Temperature
7	Engine Intercooler Temperature
8	Engine Intercooler Thermostat Opening

Die Beschreibung des J1939 Protokolls wurde speziell für unsere Anwendung gekürzt, natürlich ist das Protokoll noch viel umfangreicher, weitere Informationen zu J1939 finden Sie z.B. auf: http://de.wikipedia.org/wiki/SAE_J1939 oder http://www.can-cia.org

6. Die Software

Die Applikation ist in „C“ unter Verwendung der GTK+ Bibliothek entwickelt, das Programm lässt sich auf jedem Linux PC übersetzen nicht nur auf dem Raspberry PI.
Warum GTK+? Der LXDE Desktop des „Raspian Immages“ basiert auf GTK+, das Programm ist also „nativ“ entwickelt, gerade auf kleinen Embedded Systemen ist es von großem Vorteil wenn keine zusätzlichen Bibliotheken geladen werden müssen die Speicher und Systemressourcen benötigen. Dank „C“ ist das Programm sehr klein und schnell.

Das Modul „can.c“ beinhaltet die wichtigsten Funktionen, die nachfolgend kurz erläutert werden.
Die Funktion „J1939CanInit“:

- Tiny-CAN API Treiber laden

- API Treiber im „EX-Modus“ initialisieren
- CAN Device erzeugen
- 2 FIFOs erzeugen, für den CAN-Trace und die Automatische Baudratenerkennung
- Die CAN-Nachrichten Filter konfigurieren
- API Treiber Informationen auslesen (Version, ...)

Die Funktion „J1939CanDown“:
- CAN Schnittstelle schließen
- Plug & Play Thread beenden („J1939PnPStop“)

Die Funktion „J1939CanOpen“:
- Tiny-CAN Device öffnen und konfigurieren
- Informationen der Tiny-CAN Hardware auslesen (Firmware-Version, …)
- CAN-Controller im (LOM) „Listen-Only-Mode“ starten

Die Funktion „J1939CanClose“:
- Tiny-CAN Device schließen

Die Funktion „J1939CanReadMessages“:
- Liest die Filter-Empfangspuffer des aus
- Bereitet die Empfangenen CAN Nachrichten in Signale auf (WaterTemp, FuelTempm, OilTemp)
- Wird zyklisch von „EventTimerProc“ (min.c) aus aufgerufen

Die Funktion „J1939PnPStart“:
- Startet den Plug & Play Thread der das An- und Abstecken der Tiny-CAN Hardware überwacht.

Die Funktion „J1939PnPScan“:
- Triggert den Plug & Play Thread und aktualisiert die Liste der verbundenen CAN Hardware

Die Funktion „J1939PnPStop“:
- Beendet den Plug & Play Thread

6.1 Baudratenerkennung

Der CAN Controller wird im „Listen-Only-Mode“ betrieben und probiert in einem zyklischen Durchlauf alle CAN Standard Baudraten durch, werden Datenframes empfangen ist die richtige Geschwindigkeit erkannt. Der „Liste-Only-Mode“ vermeidet die aktive generierung von Error-Frames, wodurch der CAN-Bus nicht beeinflusst wird. Werden Fehler am Bus erkannt, kann zur nächsten Baudrate gesprungen werden, nur das Tiny-CAN IV-XL hat diese Funktion.
Im Modul „can_autobaud.c“ kümmert sich ein eigener Thread um die Baudratenerkennung, der mit „StartCanAutobaud“ gestartet und mit „StopCanAutobaud“ beendet wird. Der Thread sendet Fortschritt/Erfolgsmeldungen an den GTK-Main Thread.

7. Schnittstellen zum World Wide Web

Java und HTML5 erwecken das Webfrontend zum Leben, die Funktion „init“ aus „index.htm“ lädt über „loadPage“ aus „control.js“die einzelnen „Tab-Seiten“. Angezeigt wird eine „Tab-Seite“ mit der Funktion „showTab“ („control.js“), die Funktion führt folgende 3 Schritte aus:
1. Selektierte „Tab-Seite“ anzeigen
2. „Tab-Header“ markieren
3. Setzt abhängig von der selektierten Seite einen Intervall-Timer, der alle 500ms die Funktion „loadData“ aufruft.

Abbildung 3: Screenshot des Webinterface

7.1. XML-Datenbank

Alle 5 Sekunden schreibt das Programm die Dateien „status.xml“ und „dashboard.xml“, welche den aktuellen Ist-Zustand aller Messwerte widerspiegelen. Ein „Java Script“ der Webpage liest diese Dateien alle 10 Sekunden ein und visualisiert die Messwerte in HTML und mit HTML5 Widgets.
Die XML-Datenbanken werden mit dem Modul „xml_database.c“ erstellt, die XML-Dateien sind so simpel aufgebaut, dass zum schreiben dieser keine explizite XML-Library verwendet wurde.

7.2. SQLite Datenbank

Alle 10 Minuten wird ein Datensatz in die Datenbank geschrieben. Existiert die Datenbank „engine.db“ nicht wird sie automatisch erzeugt. Zur Zeit existiert für die SQLite Datenbank noch kein Webfrontend. Die Datenbank kann aber mit dem „DB Browser for SQLite“ inspiziert werden. Zu Installieren mit: „sudo apt-get install sqlitebrowser“. Mehr Informationen und Versionen für andere Betriebsysteme auf: http://www.sqlitebrowser.org

7.3. RRDtool

Das RRDtool uralt und immer noch ein Industriestandart, mit dem RRDtool lassen sich Daten aufzeichnen und als Grafik visualisieren. Jede Minute wird ein Datensatz

gespeichert. Das RRDtool erzeugt dann die entsprechenden Verlaufsgrafiken für eine Stunde und einen Tag die auf der Webpage dann einfach als „png“ Datei angezeigt werden. Das Modul „rrd_tool_database.c“ ruft das Kommandozeilen RRDtool mit den entsprechenden Parametern auf um die Datenbank und Grafiken zu erzeugen. Weitere Informationen zum RRDtool finden Sie auf: http://de.wikipedia.org/wiki/RRDtool

7.4. Modbus TCP

Ein sehr einfaches Protokoll zum Austausch von Daten. Die aktuellen Messwerte werden in 16 Bit Registern bereitgestellt und können von einen Client ausgelesen werden. Für das Modbus Protokoll gibt es ein frei erhältliches Analyse Tool was auf http://qmodbus.sourecforge.net zu finden ist. Weitere Informationen zum Modbus finden Sie unter: http://de.wikipedia.org/wiki/Modbus Für das Handling des Modbus Protokolls sind die Module „modbus_io.c“ und „mhs_file_event.c“ zuständig. Eine Client APP für das Modbus Protokoll existiert im Augenblick noch nicht. Für Linux gibt es für das Modbus Protokoll eine Bibliothek, die Implementierung gestaltet sich dadurch recht einfach, das und das freie Monitoring Tool waren die Entscheidung Modbus zu verwenden.

8. Simulation

Um die J1939 CAN Nachrichten auf dem Bus zu generieren verwenden wir ein „CAN-LCD 2“ Display. Das Projekt und der Verdrahtungsplan sind im Verzeichnis „…/J1939Display/tools“ zu finden. Zu beachten ist das „CAN-LCD 2“ Display noch einen aktiven Empfänger benötigt, da der CAN Controller unseres J1939Displays ja im „Listen only Mode“ betrieben wird und empfangene CAN Nachrichten nicht mit einen Acknowledge bestätigt werden. Bitte auch die Terminierung nicht vergessen.

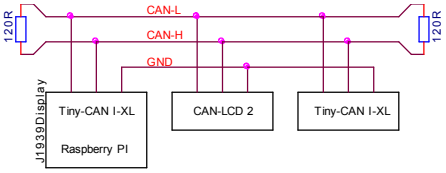

Abbildung 4: Verdrahtung der J1939 Restbussimulation (Versuchsaufbau)

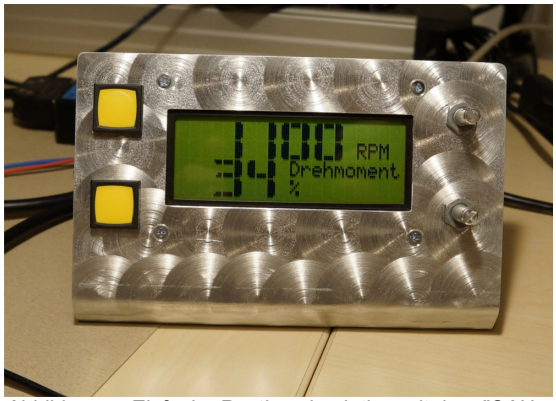

Abbildung 5: Einfache Restbussimulation mit dem "CAN-LCD 2" realisiert

9. Links

Hintergrundgrafik:
Achtzylinder Dieselmotor von MAN (Typ D2868LF)
Quelle: https://de.wikipedia.org/wiki/MAN-Motoren

Webfrontend:
Einiges an Quellcode für das Webfrontend stammt von diesen Projekt
https://github.com/collin80/GEVCU

Die Homepage der GTK Organisation:
http://www.gtk.org