ETH2CAN – FMS firmware

Obsah:

Ing. David Španěl

Mgr. Vítězslav Rejda

CANLAB s.r.o.

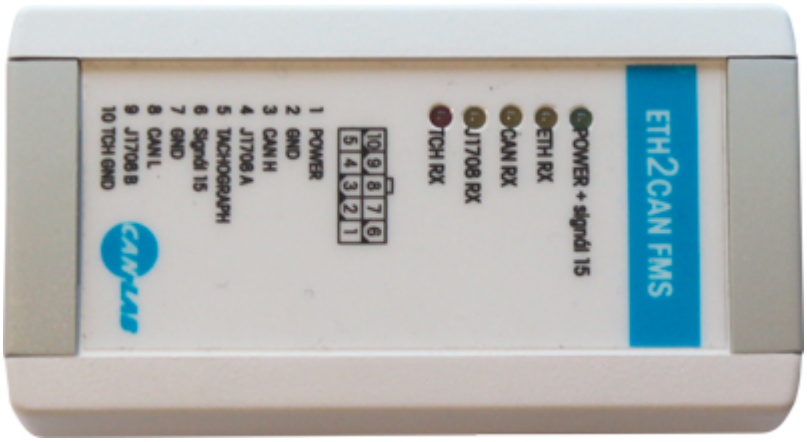
Základní popis

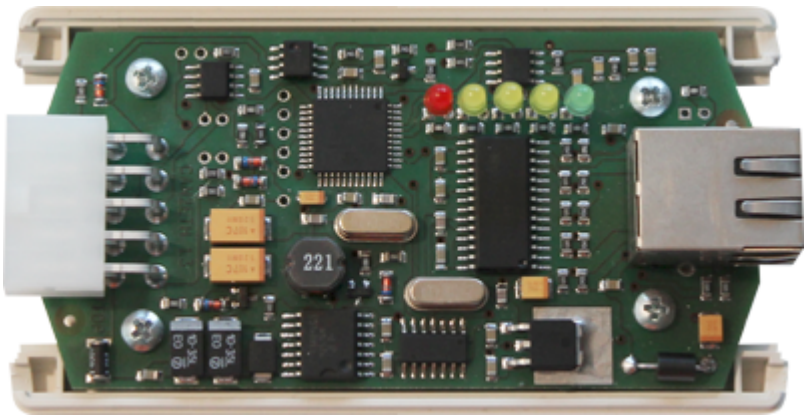
FMS firmware pro interface ETH2CAN je určeno pro poskytování vozidlových provozních dat z nákladních a osobních vozidel. Pro připojení do vozidla je interface vybaven těmito rozhraními:

- CAN bus (high speed)
- Interface digitálního tachografu DTCO1381
- Rozhraní pro sběrnici J1708 (starší vozidla)

Pro připojení do nadřazeného systému je pak určeno rozhraní ETHERNET o rychlosti 10Mbitu.

Zařízení neposkytuje veškerá data uvedená v datových paketech ale jen data dostupná. Dostupnost pak závisí na typu, výrobci, modifikaci a roku výroby vozidla.





Komunikace po rozhraní ETHERNET

Zařízení má svoji IP adresu a TCP port prostřednictvím kterého probíhá veškerá komunikace. Chová se jako server, tedy klient se připojuje k tomuto zařízení.

Pro komunikaci se používá několika paketů, každý paket obsahuje položku magic, ta je určena k ověření endianu zařízení. Má hodnotu 0xAA123456. Dále pak položku packet_type, ta rozlišuje typ paketu. Položka packet_size pak určuje velikost paketu. Ve vývojovém stádiu se používá hodnota položky packet_size nastavená na hodnotu 0xFFFF. Toto nastavení udává, že velikost není signalizována a aplikace klienta musí zjistit velikost paketu pomocí sizeof struktury. Položka id je určena k identifikaci paketu. Je li například zaslán do zařízení dotaz, je možné nastavit položku id na libovolnou hodnotu. Paket s odpovědí pak má nastavenou stejnou hodnotu. Lze tak rozlišit dvě odpovědi od sebe při zaslání dvou požadavků.

Typy paketů:

```
#define   PACKET_UNKNOWN_PACKET_ID          0
#define   PACKET_RUN                        1
#define   PACKET_SHUTDOWN                   2
#define   PACKET_MODE                       3
#define   PACKET_RESET                      4
#define   PACKET_SET_TIMESTAMP              8
#define   PACKET_GET_TIMESTAMP              9
#define   PACKET_FMS                        32
#define   PACKET_DTI                        33
#define   PACKET_FMS_EXT                    34
#define   PACKET_J1708                      35
#define   PACKET_J1708_EXT                  36
#define   PACKET_SPEC_SOR                   37
#define   PACKET_DM1                        40
#define   PACKET_STATUS                     128
#define   PACKET_REBOOT_DATA                129
#define   PACKET_VEHICLE_TYPE               245
#define   PACKET_SAE_REQUEST                246
#define   PACKET_GET_TACHOGRAPH_CONNECTION  249
#define   PACKET_RESP_TACHOGRAPH_CONNECTION 249
#define   PACKET_SET_TACHOGRAPH_CONNECTION  250
#define   PACKET_SERIAL_NUMBER              253
#define   PACKET_FIRMWARE_VERSION           254
#define   PACKET_CONFIGURATION              255
```

Podpora paketů:

Paket	Bootloader	Aplikace
PACKET_UNKNOWN_PACKET_ID	Y	Y
PACKET_RUN	Y	Y

PACKET_SHUTDOWN	Y	Y
PACKET_MODE	Y	Y
PACKET_RESET	N	Y
PACKET_SET_TIMESTAMP	N	Y
PACKET_GET_TIMESTAMP	N	Y
PACKET_FMS	N	Y
PACKET_DTI	N	Y
PACKET_FMS_EXT	N	Y
PACKET_REBOOT_DATA	Y	N
PACKET_SERIAL_NUMBER	Y (jen čtení)	Y
PACKET_FIRMWARE_VERSION	Y	Y
PACKET_CONFIGURATION	Y (jen čtení)	Y
PACKET_GET_TACHOGRAPH_CONNECTION	N	Y
PACKET_RESP_TACHOGRAPH_CONNECTION	N	Y
PACKET_SET_TACHOGRAPH_CONNECTION	N	Y

Paket UNKNOWN_PACKET_ID

Paket ve směru ETH2CAN -> Klient (nadřazený systém).

```
typedef struct _ETH_UNKNOWN_PACKET_ID {
     uint32_t  magic;
     uint8_t   packet_type;
     uint16_t  packet_size;
     uint8_t   id;
     uint8_t   unknown_packet_type;
}    ETH_ UNKNOWN_PACKET_ID;
```

Paket je vrácen pokud interface obdrží paket s neznámou hodnotou packet_type. Například v režimu bootloaderu je vrácen po zaslání paketu FMS nebo DTI, v režimu aplikace při zaslání REBOOT_DATA.

Paket RUN

Paket ve směru Klient (nadřazený systém) -> ETH2CAN.

```
typedef struct _ETH_RUN {
     uint32_t  magic;
     uint8_t   packet_type;
     uint16_t  packet_size;
     uint8_t   id;
}    ETH_RUN;
```

Paket je určen k aktivaci zařízení. Po připojení zařízení k napájení a připojení signálu 15 se zařízení nachází v režimu bootloaderu. Tento režim je určen pro snadnou změnu firmware v zařízení. Zasláním tohoto paketu dojde k aktivaci firmware. Bootloader je automaticky přepnut do režimu firmware po uplynutí intervalu 30 sekund, pokud bootloader nedetekuje příchod paketu PACKET_REBOOT_DATA.

Paket ve směru ETH2CAN -> Klient (nadřazený systém).

```
typedef struct _ETH_RUN2 {
     uint32_t  magic;
     uint8_t   packet_type;
     uint16_t  packet_size;
     uint8_t   id;
     uint8_t   mode;
}    ETH_RUN2;
```

Paket je generován jako odpověď na příchozí paket ETH_RUN. Potvrzuje přijetí tohoto paketu a zároveň v položce mode signalizuje aktuální režim firmware (bootloader: mode = 1, application-firmware: mode = 2)

Paket MODE

Paket ve směru Klient (nadřazený systém) -> ETH2CAN.

```
typedef struct _ETH_MODE {
     uint32_t  magic;
     uint8_t   packet_type;
     uint16_t  packet_size;
     uint8_t   id;
}    ETH_MODE;
```

Paket je určen k vyžádání aktuálního režimu firmware.

Paket ve směru ETH2CAN -> Klient (nadřazený systém).

```
typedef struct _ETH_MODE2 {
     uint32_t  magic;
     uint8_t   packet_type;
     uint16_t  packet_size;
     uint8_t   id;
     uint8_t   mode;
}    ETH_MODE2;
```

Paket je generován jako odpověď na příchozí paket ETH_MODE. Potvrzuje přijetí tohoto paketu a zároveň v položce mode signalizuje aktuální režim firmware (bootloader: mode = 1, application: mode = 2).

Paket SET_TIMESTAMP

Paket ve směru Klient (nadřazený systém) -> ETH2CAN.

```
typedef struct _SET_TIMESTAMP {
     uint32_t  magic;
     uint8_t   packet_type;
     uint16_t  packet_size;
     uint8_t   id;
     uint16_t  timestamp;
}    SET_TIMESTAMP;
```

Paket je určen k přenastavení čítače timestampu. Timestamp je položka nastavená pro většinu měřených veličin z CAN sběrnice a udává stáří veličiny. Hodnota je inkrementována každých 100ms. Po startu je hodnota timestampu nastavena na hodnotu 0.

Paket ve směru ETH2CAN -> Klient (nadřazený systém).

```
typedef struct _SET_TIMESTAMP2 {
     uint32_t  magic;
     uint8_t   packet_type;
     uint16_t  packet_size;
     uint8_t   id;
}    SET_TIMESTAMP 2;
```

Paket GET_TIMESTAMP

Paket ve směru Klient (nadřazený systém) -> ETH2CAN.

```
typedef struct _GET_TIMESTAMP {
     uint32_t  magic;
     uint8_t   packet_type;
     uint16_t  packet_size;
     uint8_t   id;
}    GET_TIMESTAMP;
```

Paket je určen k přečtení aktuální hodnoty čítače timestampu. Timestamp je položka nastavená pro většinu měřených veličin z CAN sběrnice a udává stáří veličiny. Hodnota je inkrementována každých 100ms.

Paket ve směru ETH2CAN -> Klient (nadřazený systém).

```
typedef struct _GET_TIMESTAMP2 {
     uint32_t  magic;
     uint8_t   packet_type;
     uint16_t  packet_size;
     uint8_t   id;
     uint16_t  timestamp;
}    GET_TIMESTAMP 2;
```

Paket je generován jako odpověď na příchozí paket SET_TIMESTAMP. Potvrzuje přijetí tohoto paketu a vrací zpět hodnotu čítače timestampu.

Paket PACKET_REBOOT_DATA

Paket ve směru Klient (nadřazený systém) -> ETH2CAN.
Paket přenáší v režimu bootloaderu 1 řádek HEX souboru.

```
typedef struct _ETH_REBOOT_DATA {
     uint32_t  magic;
     uint8_t   packet_type;
     uint16_t  packet_size;
     uint8_t   id;
     uint8_t   data[64];
}    ETH_REBOOT_DATA;
```

Paket ve směru ETH2CAN -> Klient (nadřazený systém).
Paket potvrzuje přijetí a naprogramování zaslaného řádku HEX souboru, tím signalizuje připravenost k přijetí dalšího řádku.

```
typedef struct _ETH_REBOOT_ACK {
     uint32_t  magic;
     uint8_t   packet_type;
     uint16_t  packet_size;
     uint8_t   id;
     uint8_t   error_code;
     uint8_t   dummy[68];
}    ETH_REBOOT_ACK;
```

Po zaslání celého souboru se nový FW aktivuje paketem RUN.

Položka error_code:
0 – OK
1 – chybná adresa
2 – chybná délka dat
3 – chyba zápisu do flash
4 - chyba verifikace zápisu

Paket SHUTDOWN , RESET

```
typedef struct _ETH_SHUTDOWN {
     uint32_t  magic;
     uint8_t   packet_type;
     uint16_t  packet_size;
     uint8_t   id;
     uint8_t   why;
}    ETH_SHUTDOWN;
```

Paket ve směru ETH2CAN -> Klient (nadřazený systém).

Položka why:
0 – na požadavek klienta
1 – ethernet watchdog
2 – vypnutí signálu 15

Tímto paketem zařízení indikuje ukončení činnosti po odpojení signálu 15 nebo restart zařízeni.

```
typedef struct _ETH_SHUTDOWN {
     uint32_t  magic;
     uint8_t   packet_type;
     uint16_t  packet_size;
     uint8_t   id;
}    ETH_SHUTDOWN2;
```

Paket ve směru Klient (nadřazený systém) -> ETH2CAN. Po přijetí tohoto paketu dojde k restartu firmware řídicího procesoru. Aplikace (ne bootloader) podporuje variantu příkazu RESET která resetuje pouze aplikaci.

Paket TACHOGRAPH_CONNECTION

```
typedef struct _ETH_TCH_CONNECTION_SET {
     uint32_t  magic;
     uint8_t   packet_type;
     uint16_t  packet_size;
     uint8_t   id;
     uint8_t   tachograph_connection;
}    ETH_TCH_CONNECTION_SET;
```

Paket ve směru Klient (nadřazený systém) -> ETH2CAN.

```
tachograph_connection =  0    - OFF
                         1    - AUTO (FW sam určí připojení)
                         2    - MAX3100
                         3    - Direct

typedef struct _ETH_TCH_CONNECTION_REQ {
     uint32_t  magic;
     uint8_t   packet_type;
     uint16_t  packet_size;
     uint8_t   id;
}    ETH_TCH_CONNECTION_REQ;
```

Paket ve směru ETH2CAN -> Klient (nadřazený systém).

```
typedef struct _ETH_TCH_CONNECTION {
     uint32_t  magic;
     uint8_t   packet_type;
```

```
     uint16_t  packet_size;
     uint8_t   id;
     uint8_t   tachograph_connection_actual;
     uint8_t   tachograph_connection_EEPROM;
}    ETH_TCH_CONNECTION;
```

tachograph_connection_actual – aktuálně použité nastavení
tachograph_connection_EEPROM – nastavení v EEPROM (nastavené pomocí ETH_TCH_CONNECTION_SET) které bude použito po restartu.

Paket CONFIGURATION

Paket ve směru Klient (nadřazený systém) ->ETH2CAN. Nastavuje novou konfiguraci zařízení. Zařízení odpovídá zasláním tohoto paketu zpět.

```
typedef struct {
     uint32_t  magic;
     uint8_t   packet_type;
     uint16_t  packet_size;
     uint8_t   id;
     uint8_t   can_speed;
     uint8_t   listen_only;
     uint8_t   st_ext;
     uint8_t   ip[4];
     uint16_t  port;
     uint16_t  startup_timeout;
     uint16_t  shutdown_timeout;
     uint16_t  eth_watchdog;
     uint8_t   mac[6];
     uint8_t   ipmask[4];
     uint16_t  app_start_timeout;
     uint8_t   mask[4];
     uint8_t   tachograph_mode;
     } ETH2CAN_SETTINGS;
```

Paket ve směru ETH2CAN -> Klient (nadřazený systém).
Slouží k zjištění aktuální konfigurace zařízení:

```
typedef struct _ETH2CAN_SETTINGS_REQ {
     uint32_t  magic;
     uint8_t   packet_type;
     uint16_t  packet_size;
     uint8_t   id;
}    ETH2CAN_SETTINGS_REQ;
```

can_speed – rychlost CAN sběrnice, hodnoty

0	10k
1	20k
2	33.3k
3	50k
4	62.5k
5	83.3k

6	100k
7	125k
8	250k
9	500k
10	1M

listen_only
0 normální mód (připojení na FMS bránu)
1 listen only mód (připojení na CAN bus vozidla, motorový CAN)

st_ext
0 standardní identifikátory
1 rozšířené identifikátory

ip
IP adresa zařízení. Defaultně přednastavena na 192.168.12.150. Je možné však vyžádat z výroby jinou hodnotu.

port
TCP port na kterém probíhá komunikace. Defaultně 3000.

startup_timeout
Opoždění aktivace zařízení po připojení signálu 15. Zpoždění eliminuje aktivaci zařízení při krátké aktivaci signálu 15. Čas nastavován v sekundách. Rozsah 1..200s. Defaultně 5 s.

shutdown_timeout
Opoždění deaktivace zařízení po odpojení signálu 15. Zpoždění eliminuje deaktivaci zařízení při krátkém vypnutí signálu 15. Čas nastavován v sekundách. Rozsah 1..200s. Defaultně 5 s.

eth_watchdog
Timeout v sekundach. Pokud po uvedený čas není detekována aktivita klienta dojde k resetu zařízení. Je li nastavena hodnota 0, funkce není aktivní. Rozsah 20..300s.

mac
MAC adresa zařízení. Defaultně 00-04-A3-00-00-00.

app_start_timeout
Čas po kterém je bootloader automaticky přepnut do aplikace pokud není přijat paket, který provádí změnu firmware .

Po přijetí tohoto paketu je nové nastavení uloženo do interní EEPROM interface. Aby se nové nastaveni uplatnilo, je nutné restartovat firmware příkazem shutdown nebo reset.

mask
Maska sítě. Defaultně přednastavena na 255.255.255.0.

tachograph_mode
Nastavení typu připojeného digitálního tachografu, 0-VDO Siemens, 1-Stoneridge, 2-Actia.

Paket FIRMWARE VERSION

Tímto paketem jsou je vyžadována verze firmware v zařízení interface ETH2CAN.

Požadavek klienta má tvar:

```
typedef struct _ETH_FIRMWARE_REQ {
     uint32_t  magic;
     uint8_t   packet_type;
     uint16_t  packet_size;
     uint8_t   id;
}    ETH_FIRMWARE_REQ;
```

Odpověď interface ETH2CAN má tvar:

```
typedef struct _ETH_FIRMWARE {
     uint32_t  magic;
     uint8_t   packet_type;
     uint16_t  packet_size;
     uint8_t   id;
     uint8_t   fw_version_string[20];
}    ETH_FIRMWARE;
```

Položka obsahuje string s verzí firmware. Neobsahuje ukončovací 0 řetězce. Řetězec má tvar například CANLABsro-01.10. V režimu bootloaderu pak CANLABsro-01.10boot. Bootloader využívá jiné číslování než aplikace!

Paket SERIAL NUMBER

Tento paket je určen k přečtení sériového čísla interface ETH2CAN.

Požadavek klienta má tvar:

```
typedef struct _ETH_SERNUM _REQ {
     uint32_t  magic;
     uint8_t   packet_type;
     uint16_t  packet_size;
     uint8_t   id;
}    ETH_SERNUM _REQ;
```

Odpověď interface ETH2CAN má tvar:

```
typedef struct _ETH_SERNUM {
     uint32_t  magic;
     uint8_t   packet_type;
     uint16_t  packet_size;
     uint8_t   id;
     uint8_t   serial_number[14];
}    ETH_SERNUM;
```

Položka obsahuje string se sériovým číslem interface. Sériové číslo má tvar E2Cxxxxxxxxxx. První 3 znaky jsou vždy E2C. Dalších 10 znak; jsou číslice, tedy sériové číslo může obsahovat hodnotu 0000000000-9999999999. Poslední, čtrnáctý znak má hodnotu 0, tedy konec řetězce.

Paket FMS

Tímto paketem jsou vyžadována data čtená ze sběrnice CAN.

Požadavek klienta má tvar:

```
typedef struct _ETH_FMS_REQ {
     uint32_t  magic;
     uint8_t   packet_type;
     uint16_t  packet_size;
     uint8_t   id;
}    ETH_FMS_REQ;
```

Odpověď interface ETH2CAN má tvar:

```
typedef struct _ETH_FMS {
     uint32_t  magic;
     uint8_t   packet_type;
     uint16_t  packet_size;
     uint8_t   id;
     uint16_t  rpm;
     uint16_t  speed;
     uint8_t   acc_pedal;
     uint8_t   brake_pedal;
     uint32_t  total_fuel_used;
     uint32_t  total_engine_hours;
     uint8_t   fuel_level;
     uint16_t  fuel_consumption;
     uint8_t   axle_weight_captured[12];
     uint8_t   axle_weight_location[12];
     uint16_t  axle_weight[12];
     uint32_t  total_vehicle_distance;
     uint16_t  daily_vehicle_distance;
     uint16_t  service_distance;
     uint8_t   engine_coolant_temperature;
     uint16_t  tachograph_speed;
     uint8_t   tachograph[4];
```

```
    uint8_t   tire_pressure_captured[12];
    uint8_t   tire_pressure_location[12];
    uint8_t   tire_pressure[12];
    uint16_t  door;
    uint16_t  fuel_instantaneous;
    uint16_t  fuel_rate;

    //ROZSIRENI - DETEKOVAT NA ZAKLADE VELIKOSTI PAKETU
    uint8_t   secondary_fuel_level;
    uint32_t  hires_total_fuel_used;

    uint8_t   percent_torque;
    uint8_t   service_brake_air_pressure[2];
    uint8_t   diesel_exhaust_fluid_level;
    uint8_t   tell_tale_status[8*4];
    uint8_t   clutch_brake_cruise_control;
    uint8_t   engine_load_at_speed;
    uint16_t  gross_combination_vehicle_weight;
    uint8_t   retarder_torque_mode;
    uint8_t   actual_retarder_percent_torque;
    uint8_t   retarder_selection_non_engine;
    uint8_t   air_suspension_control[8];
    uint8_t   selected_gear;
    uint8_t   current_gear;
    uint8_t   door2[8];

} ETH_FMS;

    * ZMĚNA ve verzi FW od 2.00, revize HW 1.30
    * ZMĚNA ve verzi FW od 1.60, revize HW 1.20
    * ZMĚNA ve verzi FW od 2.12, revize HW 1.30
```

Data ze struktury je možné převést na skutečné hodnoty pomocí této tabulky:

Data	Počet bitů	Váha 1 bitu	Offset
Rychlost	16	1/256 km/h	0
Poloha pedálu akcelerace	8	0.4 %	0
Poloha brzdového pedálu	8	0.4 %	0
Celkově spotřebované palivo	32	0.5 litru	0
Celkově spotřebované palivo - HIRES	32	0.001 litru	0
Stav palivové nádrže	8	Truck:0.4 VW:1litr	0
Otáčky motoru	16	0.125 otáčky	0
Zatížení nápravy	16	0,5 kg	0
Celkový počet motohodin	32	0,05 hod.	0
Celkově najeté kilometry	32	5 m	0
Vzdálenost v kilometrech do servisní prohlídky	16	5 km	-160 635
Teplota chladící kapaliny.	8	1°C	-40
Průměrná spotřeba.	16	1/512 km/L	0

Axle weight

Položka axle_weight_location[x] udává lokaci hodnoty zatížení nápravy v položce axle_weight[x]. Hodnota axle_weight_captured[x] = 0 udává, že položka neobsahuje žádnou (platnou) hodnotu, axle_weight_captured[x] = 1 udává, že položka obsahuje platnou hodnotu.

V položce axle_weight_location[x] je zakódována informace o čísle měřené nápravy a kole této nápravy. Dolní 4 bity udávají index kola, horní 4 bity udávají index nápravy. Jsou-li všechny 4 bity nastaveny na 1, lokace není známa.

Položka tachograph[4]

Tato položka obsahuje informace, které je možno dekódovat dle následujícího popisu:

tachograph[0]
Bit 2..0 :Driver 1 working state
000 = Rest
001 = Driver available
010 = Work
011 = Drive
110 = Error
111 =
Bit 5..3 :Driver 2 working state
000 = Rest
001 = Driver available
010 = Work
011 = Drive
110 = Error
111 = not available
Bit 7..6 :Drive recognize
00 = Vehicle motion not detected
01 = vehicle motion

tachograph[1]
Bit 3..0 : Driver 1 time rel states
0000 = normal
0001 = 15 min bef. 4 ½ h
0010 = 4 ½ h reached
0011 = 15 min bef. 9 h
0100 = 9 h reached
0101 = 15 min bef. 16 h
0110 = 16h reached
1110 = Error
1111 = not available
Bit 5..4 :Driver 1 card
00 = Card not present
01= Card present
Bit 7..6 :Overspeed
00 = No overspeed

01 = Overspeed

tachograph[2]

Bit 3..0 : Driver 2 time rel states

0000 = normal
0001 = 15 min bef. 4 ½ h
0010 = 4 ½ h reached
0011 = 15 min bef. 9 h
0100 = 9 h reached
0101 = 15 min bef. 16 h
0110 = 16h reached
1110 = Error
1111 = not available

Bit 5..4 :Driver 2 card

00 = Card not present
01= Card present

Bit 7..6 :Not used

tachograph[3]

Bit 0..1 :System event

00 = no tachogr. Event
01 = tachogr. Event

Bit 2..3 :Handling information

00 = no handling information
01 = handling information

Bit 5..4 :Tachgraph performance

00 = Normal performance
01 = Performance

Bit 7..6 :Direction indicator

00 = Forward
01 = Reverse

Paket FMS_EXT

Tímto paketem jsou vyžadována data čtená ze sběrnice CAN.

Požadavek klienta má tvar:

```
typedef struct _ETH_FMS_REQ {
     uint32_t  magic;
     uint8_t   packet_type;
     uint16_t  packet_size;
     uint8_t   id;
}    ETH_FMS_REQ;
```

Odpověď interface ETH2CAN má tvar:

```
typedef struct _ETH_FMS {
     uint32_t  magic;
     uint8_t   packet_type;
     uint16_t  packet_size;
     uint8_t   id;
     uint16_t  rpm;
     uint16_t  speed;
     uint8_t   acc_pedal;
     uint8_t   brake_pedal;
     uint32_t  total_fuel_used;
     uint32_t  total_engine_hours;
     uint8_t   fuel_level;
     uint16_t  fuel_consumption;
     uint8_t   axle_weight_captured[12];
     uint8_t   axle_weight_location[12];
     uint16_t  axle_weight[12];
     uint32_t  total_vehicle_distance;
     uint16_t  daily_vehicle_distance;
     uint16_t  service_distance;
     uint8_t   engine_coolant_temperature;
     uint16_t  tachograph_speed;
     uint8_t   tachograph[4];
     uint8_t   tire_pressure_captured[12];
     uint8_t   tire_pressure_location[12];
     uint8_t   tire_pressure[12];
     uint16_t  door;
     uint16_t  fuel_instantaneous;
     uint16_t  fuel_rate;
     //TIMESTAMP
     uint16_t  rpm_captured;
     uint16_t  speed_captured;
     uint16_t  acc_pedal_captured;
     uint16_t  brake_pedal_captured;
     uint16_t  total_fuel_used_captured;
     uint16_t  total_engine_hours_captured;
     uint16_t  fuel_level_captured;
     uint16_t  fuel_consumption_captured;
     uint16_t  total_vehicle_distance_captured;
     uint16_t  daily_vehicle_distance_captured;
     uint16_t  service_distance_captured;
     uint16_t  engine_coolant_temperature_captured;
     uint16_t  tachograph_speed_captured;
     uint16_t  tachograph_captured;
     uint16_t  fuel_instantaneous_captured;
     uint16_t  fuel_rate_captured;

     //ROZSIRENI - DETEKOVAT NA ZAKLADE VELIKOSTI PAKETU
     uint8_t         secondary_fuel_level;
     uint32_t  hires_total_fuel_used;
     uint8_t         percent_torque;
```

```
    uint8_t   service_brake_air_pressure[2];
    uint8_t   diesel_exhaust_fluid_level;
    uint8_t   tell_tale_status[8*4];
    uint8_t   clutch_brake_cruise_control;
    uint8_t   engine_load_at_speed;
    uint16_t  gross_combination_vehicle_weight;
    uint8_t   retarder_torque_mode;
    uint8_t   actual_retarder_percent_torque;
    uint8_t   retarder_selection_non_engine;
    uint8_t   air_suspension_control[8];
    uint8_t   selected_gear;
    uint8_t   current_gear;
    uint8_t   door2[8];
    //TIMESTAMP
    uint16_t  hires_total_fuel_used_captured;
    uint16_t  service_brake_air_pressure_captured;
    uint16_t  diesel_exhaust_fluid_level_captured;
    uint16_t  tell_tale_status_captured;
    uint16_t  gross_combination_vehicle_weight_captured;
    uint16_t  retarder_captured;
    uint16_t  air_suspension_control_captured;
    uint16_t  gear_captured;
    uint16_t  door2_captured;

} ETH_FMS;
* ZMĚNA ve verzi FW od 2.00, revize HW 1.30
* ZMĚNA ve verzi FW od 1.60, revize HW 1.20
* ZMĚNA ve verzi FW od 2.05, revize HW 1.30

* ZMĚNA ve verzi FW od 2.12, revize HW 1.30
```

Položky xxx_captured udávají stáří veličiny od jejího přečtení z CAN bus sběrnice ve stovkách milisekund. Veličina která nebyla z CAN sběrnice přečtena má hodnotu 65535.

Položka secondary_fuel_level; má timestamp stejný jako fuel_level.

Paket DTI

Tímto paketem jsou vyžadována data čtená z digitálního tachografu.

Požadavek klienta má tvar:

```
typedef struct _ETH_DTI_REQ {
     uint32_t  magic;
     uint8_t   packet_type;
     uint16_t  packet_size;
}    ETH_DTI_REQ;
```

Odpověď interface ETH2CAN má tvar:

```
typedef struct _ETH_DTI {
     uint32_t  magic;
     uint8_t   packet_type;
     uint16_t  packet_size;
     uint8_t   id;
     uint8_t   seconds;
     uint8_t   minutes;
     uint8_t   hours;
     uint8_t   month;
     uint8_t   day;
     uint8_t   year;
     uint8_t   local_minute_offset;
     uint8_t   local_hour_offset;
     uint8_t   work_states;
     uint8_t   driver_1_states;
     uint8_t   driver_2_states;
     uint8_t   tachograph_status;
     uint16_t  tachograph_vehicle_speed;
     uint32_t  total_vehicle_distance;
     uint32_t  trip_distance;
     uint16_t  k_factor;
     uint16_t  engine_speed;
     uint16_t  additional_information;
     uint8_t   vehicle_id_len;
     uint8_t   vehicle_id[20];
     uint8_t   vehicle_reg_len;
     uint8_t   vehicle_reg[20];
     uint8_t   driver_1_len;
     uint8_t   driver_1[20];
     uint8_t   driver_2_len;
     uint8_t   driver_2[20];
} ETH_DTI;
```

Paket J1708

Paket přidán ve verzi FW od 2.00, revize HW 1.30.

```
typedef struct _ETH_J1708{
     uint32_t  magic;
     uint8_t   packet_type;
     uint16_t  packet_size;
     uint8_t   id;
     uint8_t   road_speed;
     uint8_t   fuel_level;
     uint8_t   engine_temperature;
     uint16_t  fuel_rate;
     uint16_t  fuel_economy;
     uint16_t  aver_fuel_economy;
     uint16_t  engine_speed;
     uint32_t  total_fuel;
     uint32_t  total_km;
```

```
    uint32_t  total_hours;
    uint32_t  manuf_total_fuel;
}ETH_J1708;
```

Pozor, data ze sběrnice J1708 jsou normou udávána v anglosaských jednotkých.

Data	Počet bitů	Váha 1 bitu	Offset
Rychlost	8	0.805 km/h	0
Stav palivové nádrže	8	0.5 %	0
Teplota motoru	8	1°C	0
Průtok paliva	16	16,428 x 10-6 l/s	0
Okamžitá spotřeba	16	1,66072 x10-3 km/l	0
Průměrná spotřeba	16	1,66072 x10-3 km/l	0
Otáčky motoru	16	0,25 rpm	0
Celkově spotřebované palivo	32	0,473 l (0,125 gal)	0
Celkové kilometry	32	0,05 h	0
Celkove motohodiny	32	0,161 km (0,1 mi)	0
Celkově spotřebované palivo-manufactured specific	32	0,01 l	0

Paket J1708_EXT

```
Paket přidán ve verzi FW od 2.00, revize HW 1.30.
typedef struct _ETH_J1708_EXT{
    uint32_t  magic;
    uint8_t   packet_type;
    uint16_t  packet_size;
    uint8_t   id;
    uint8_t   road_speed;
    uint8_t   fuel_level;
    uint8_t   engine_temperature;
    uint16_t  fuel_rate;
    uint16_t  fuel_economy;
    uint16_t  aver_fuel_economy;
    uint16_t  engine_speed;
    uint32_t  total_fuel;
    uint32_t  total_km;
    uint32_t  total_hours;
    uint32_t  manuf_total_fuel;

    //TIMESTAMP
    uint16_t  road_speed_captured;
    uint16_t  fuel_level_captured;
    uint16_t  engine_temperature_captured;
    uint16_t  fuel_rate_captured;
    uint16_t  fuel_economy_captured;
    uint16_t  aver_fuel_economy_captured;
    uint16_t  engine_speed_captured;
    uint16_t  total_fuel_captured;
    uint16_t  total_km_captured;
    uint16_t  total_hours_captured;
    uint16_t  manuf_total_fuel_captured;
```

```
} ETH_J1708_EXT;

* ZMĚNA ve verzi FW od 2.10, revize HW 1.30
```

Paket SPEC_SOR

Přidán ve verzi FW 3.00.

```
typedef struct  {
     uint32_t  magic;
     uint8_t   packet_type;
     uint16_t  packet_size;
     uint8_t   id;

     uint8_t   interior_temperature;
     uint8_t   outdoor_air_temperature;
     uint8_t   air_conditioning;
     uint8_t   heating;

     uint16_t  interior_temperature_captured;
     uint16_t  outdoor_air_temperature_captured;
     uint16_t  air_conditioning_captured;
     uint16_t  heating_captured;

     uint32_t  air_conditiontotal_time;
     uint32_t  heating_total_time;

     uint16_t  air_conditiontotal_time_captured;
     uint16_t  heating_total_time_captured;


} ETH_FMS_SPEC_SOR;

* ZMĚNA ve verzi FW od 3.08, revize HW 1.40
```

interior_temperature - rozlišení 0.5 °C, offset -40 stupňů
outdoor_air_temperature - rozlišení 0.5 °C, offset -40 stupňů
air_conditioning – bitová informace, b0=1, aktivace klimatizace
heating_captured – bitová informace, b0=1, aktivace topení v prostoru cestujících
heating_captured – bitová informace, b1=1, aktivace ventilů u nezávislého topení
air_conditiontotal_time – rozlišení 1 sekunda
heating_total_time – rozlišení 1 sekunda

Paket DM1

Od verze FW 3.08 je doplněna podpora funkce čtení chybových kódů DM1 dle SAE J1939. Firmware podporuje uložení až 16 aktivních chybových kódů. Čtení chyb se provádí po jedné pomocí tohoto dotazu:

```
typedef struct _ETH_DM1_REQ {
     uint32_t  magic;
     uint8_t   packet_type;
     uint16_t  packet_size;
     uint8_t   id;

     uint8_t   index;
     } ETH_DM1;
```

Jako index se zadává položka v bufferu chyb, hodnota indexu je 0 až 15. Je li zadán index mimo rozsah, převodník dotaz ignoruje. Odpověď na dotaz má tento tvar:

```
typedef struct _ETH_DM1_REQ {
     uint32_t  magic;
     uint8_t   packet_type;
     uint16_t  packet_size;
     uint8_t   id;

     uint8_t   index;
     uint8_t   size;
     uint8_t   max_size;
     uint8_t   flags;
     uint8_t   SPN[3];
     uint8_t   FMI;
     uint8_t   OC;
     uint8_t   LAMP;
     uint8_t   SECU;
     } ETH_DM1;
```

index – index bufferu chyb DM1 jaký byl vyžádán.
size – celkový počet uložených chyb
max_size – maximální velikost bufferu pro DM1 zprávy, FW 3.08 má buffer nastavený na délku 6 (index 0..15).
SPN – kód chyby (Suspect Parameter Numbers), SPN[2] horní bajt, SPN[1] prostřední bajt, SPN[0] dolní bajt, část kódů specifikována dle SAE J1939, část kódů je továrních
FMI - Failure Mode Identifier, význam dle tabulky:

- 0 = Data Valid but Above Normal Operational Range, Most Severe Level
- 1 = Data Valid but Below Normal Operational Range, Most Severe Level
- 2 = Data Erratic, Intermittent or Incorrect (rationality)
- 3 = Voltage Above Normal, or Shorted to High Source
- 4 = Voltage Below Normal, or Shorted to High Source
- 5 = Current Below Normal, or Open Circuit
- 6 = Current Above Normal, or Grounded Circuit
- 7 = Mechanical System not Responding or Out of Adjustment
- 8 = Abnormal Frequency or Pulse Width or Period
- 9 = Abnormal Update Rate
- 10 = Abnormal Rate of Change
- 11 = Failure Code not Identifiable
- 12 = Bad Intelligent Device or Component

- 13 = Out of Calibration
- 14 = Special Instructions
- 15 = Data Valid but Above Normal Range: Least Severe Level
- 16 = Data Valid but Above Normal Range: Moderately Severe Level
- 17 = Data Valid but Below Normal Range: Least Severe Level
- 18 = Data Valid but Below Normal Range: Moderately Severe Level
- 19 = Received Network Data in Error: (Multiplexed Data)
- 20 = Data Drifted High (rationality high)
- 21 = Data Drifted Low (rationality low)

OC - occurrence count
ECU – adresa ECU

- 0 = Engine #1
- 1 = Engine #2
- 3 = Transmission #1
- 4 = Transmission #2
- 5 = Shift Console – Primary
- 6 = Shift Console – Secondary
- 7 = PTO
- 8 = Axle – Steering
- 9 = Axle - Drive #1
- 10 = Axle - Drive #2
- 11 = Brakes - System Controller
- 12 = Brakes - Steer Axle"
- 13 = Brakes - Drive axle #1
- 14 = Brakes - Drive Axle #2
- 15 = Retarder – Engine
- 16 = Retarder – Driveline
- 17 = Cruise Control
- 18 = Fuel Systém
- 19 = Steering Controller (AST)
- 20 = Suspension - Steer Axle
- 21 = Suspension - Drive Axle #1
- 22 = Suspension - Drive Axle #2
- 23 = Instrument Cluster #1
- 33 = Body Controller
- a další dle SAE J1939
-

LAMP – varovné kontrolky – vliv indikované chyby

bits 8-7	Malfunction Indicator Lamp Status, Lamp to indicate when there is an emission related trouble code aktive.
bits 6-5	Red Stop Lamp Status, Lamp to indicate a problem that is severe enough to warrant stopping the vehicle.
bits 4-3	Amber Warning Lamp Status, Lamp to indicate a problem with the vehicle system but the vehicle does not need to be stopped immediately.
bits 2-1	Protect Lamp Status, Lamp to indicate a problem with a

	vehicle systém that is most likely not electronic subsystem related. e.g. Coolant Temperature has exceeded it defined range

Paket STATUS

Paket je určen pro ladění firmware. Obsahuje například délky front dat,hodnoty registrů a podobně. Význam položek a jeho délka se může měnit podle verze firmware. Je odesílán pouze na dotaz.

Paket VEHICLE_TYPE

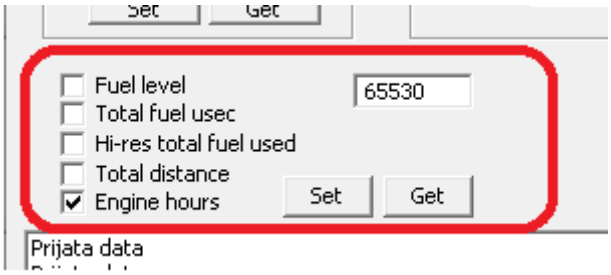
Paket obsahuje za hlavičkou dvoubajtovou hodnotu, která blíže specifikuje vozidlo a způsob dekódování dat.

- 0 = Generic SAE J1939
- 1 = SOR_V1
- 2 = Solaris_V1
- 3 = SOR_V2

Paket PACKET_SAE_REQUEST

Pomocí tohoto paketu lze nastavit vyžadování některých dat dle SAE J1939. Přednastavení provádějte pomocí servisního programu.

Příklad konfigurace v servisním a testovacím SW pro SOR V2, pokud je třeba provádět dotazy na stav motohodin:



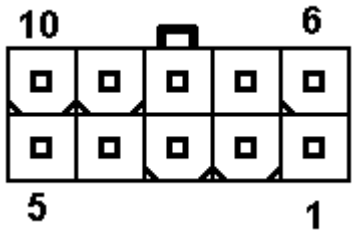
Připojení zařízení

Zařízení je uloženo v krabičce TOPTEC 102 firmy OKW. Zařízení má 2 konektory. Prvním je konektor RJ45, tedy klasický ethernetový konektor. Druhým konektorem je konektor MOLEX, který je určen při připojení napájení a vozidlových sběrnic.

Zařízení pracuje s rozsahem napájecího napětí 8-36V. Spotřeba zařízení v provozním stavu je 1.7W. V deaktivovaném stavu po odpojení signálu 15 je spotřeba rovna téměř nule. Signál 15 je aktivován cca od úrovně 1V.

..

Konektor MOLEX



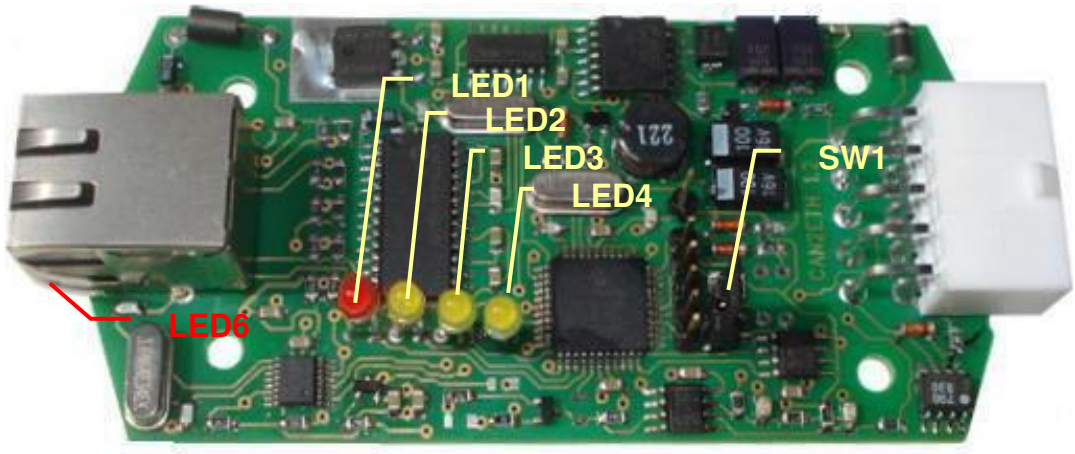
Konektor na PCB.

Pin	Popis
1	Napájecí napětí 8-36V
2	GND
3	CAN H
4	J1708 A
5	Tachograf A – signál
6	Signál 15 (startup-shutdown)
7	GND
8	CAN L
9	J1708 B
10	Tachograf B – GND

Funkce indikačních LED

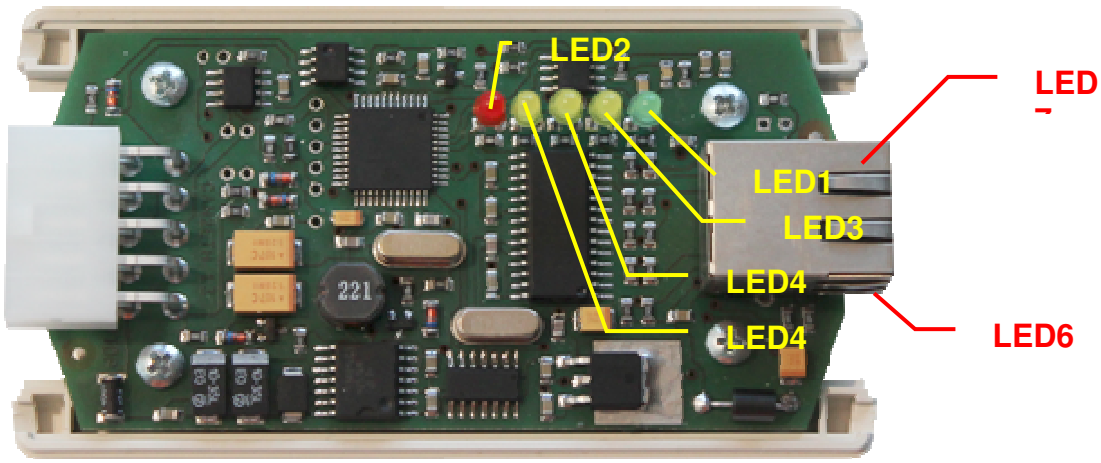
HW <= rev.1.2

LED#	Barva	Popis
1	**RED**	Chyba CAN sběrnice – bus off (například chybně nastavena rychlost sběrnice, není funkční v režimu Listen only). V režimu bootloaderu tato LED bliká v intervalu 1s.
2	**YELLOW**	Indikuje činnost CAN sběrnice, LED mění stav.
3	**YELLOW**	Indikuje činnost interface digitálního tachografu, LED mění stav.
4	**YELLOW**	Indikuje činnost sběrnice J1708, LED mění stav.
5	**GREEN**	Indikuje příchod paketu (TCPIP paket, ping apod.)
6	**YELLOW**	Indikuje připojení ethernetového kabelu.



HW >= rev.1.3

LED#	Barva	Popis
1	GREEN	Power LED, aktivní je li připojen signál „15“.
2	RED	Indikuje činnost ETHERNETU, při příjmu paketu LED mění stav.
3	YELLOW	Indikuje činnost interface CANu, LED mění stav.
4	YELLOW	Indikuje činnost sběrnice J1708, LED mění stav.
5	YELLOW	Indikuje činnost interface digitálního tachografu, LED mění stav.
6	GREEN	Indikuje příchod paketu (TCPIP paket, ping apod.)
7	YELLOW	Indikuje připojení ethernetového kabelu.



Zkratovací propojka **SW1** je určena pro aktivaci zakončovacího odporu 120 ohmu na CAN sběrnici. CAN sběrnice je vždy zakončena na obou stranách zakončovacími odpory 120 ohmu. Při připojení na motorový CAN není třeba zakončovaní odpor aktivovat, v případě připojení na FMS bránu je to obvykle nutné. Existenci správného

počtu zakončovacích odporů je možné ověřit ve vypnutém vozidle pomocí ohmmetru. Mezi vodiči CAN H a CAN L je ve správném stavu odpor cca 60 ohmu.
Propojka **SW2** je určena pro reset převodníku do výrobního nastavení.

Příkaz SETTINGS u nejčastěji monitorovaných vozidel.

Nákladní vozidla – páteřní CAN bus

- listen only
- rozšířený CAN ID
- rychlost 250k

Nákladní vozidla – FMS gateway (FMS brána)

- normální mód
- rozšířený CAN ID
- rychlost 250k

Vozidla Škoda/VW, motorový CAN bus

- listen only
- standardní CAN ID
- rychlost 500k

Změna FW uživatelským programem

Interface podporuje možnost update firmware. Po připojení interface k napájení dojde vždy automaticky ke spuštění bootloaderu. Bootloader je ukončen příkazem z nadřazeného systému (zasláním paketu **ETH_RUN**), nebo po uplynutí doby nakonfigurované v parametru **`app_start_timeout.`**

Firmware je uložen v souborech s příponou HEX v textovém formátu jako řada hexadecimálních číslic.

Řádek tohoto souboru má následující tvar:

```
:1034B00029F070C30CF371C30DF3000E0DBFFF0EA6
```

Jednotlivé řádky jsou do zařízení odesílány celé s výjimkou počáteční dvojtečky. K zaslání je použit paket **PACKET_REBOOT_DATA**. Data řádku HEX souboru bez počáteční dvojtečky jsou uložena v sekci data.

Po provedení zápisu zaslaného řádku vygeneruje interface zpět odpověď pomocí paketu **ETH_REBOOT_ACK**.Je li zápis proveden správně, obsahuje položka **error_code** hodnotu 0. Protože HEX soubor obsahuje i některá automaticky vygenerovaná data na adresách mimo povolený rozsah, vrací v některých případech interface v položce error_code hodnotu 1 (chybná adresa). Tento chybový kód ignorujte a pokračujte v bootování následujícím řádkem stejně jako v případě návratové hodnoty 0. Je li vrácen návratový kód 2 (chybná délka dat), je chyba ve formátu HEX souboru. U návratových kódů 3 (chyba zápisu do flash) a 4 (chyba verifikace zápisu) je možné se pokusit o opakovaný zápis zaslaného řádku HEX souboru.

Další řádek HEX souboru není možné zaslat před obdržením odpovědi **ETH_REBOOT_ACK** se stavem zápisu předchozího řádku.

Paket **ETH_REBOOT_ACK** obsahuje pole dummy ve kterém je uložen obraz paměťového segmentu, do kterého byl zápis prováděn. Tato položka je určena pouze pro verifikaci chování firmware bootloaderu. Během bootování ji ignorujte.

Jakmile jsou zaslány všechny řádky HEX souboru, pokračujte v činnosti spuštěním aplikace paketem **ETH_RUN**.

Testovací aplikace

Tlačítko pro navázání spojení se zařízením

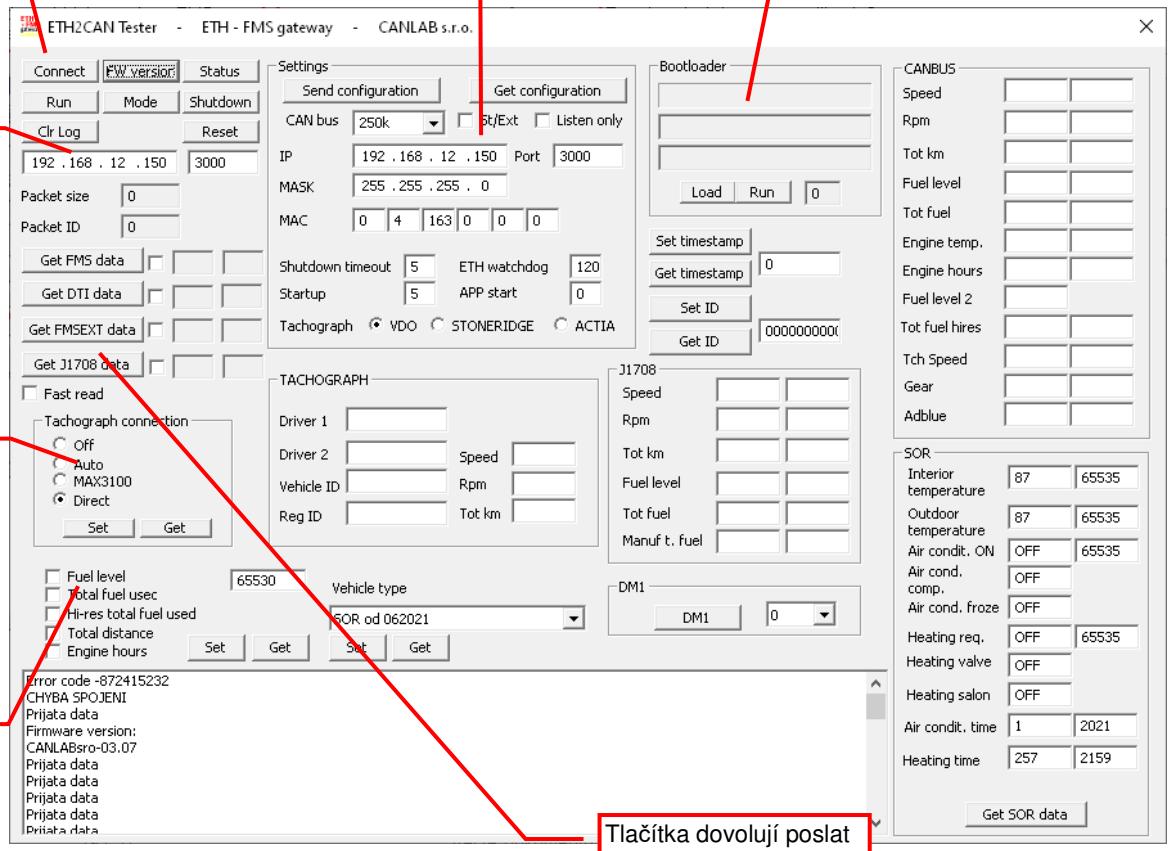
Nastavení zařízení. Doporučený postup je použít Get, provést změny a nastavit pomocí Send. Pro nákladní vozidla nastavení 250k, zatrhnout St/Ext a při přímém připojení na CAN aktivovat Listen only, u FMS brány Listen only vypnout.

Změna FW. Nelze kombinovat FW pro HW verzi 1.2 a 1.3 !!! Režim Bootloaderu je nastaven několik sekund po připojení jednotky k napájení. Čas je nastaven dle parametru APP start timeout v Settings. Manuálně lze aplikaci resetovat do Bootloaderu příkazem Shutdown (vlevo). Tlačítko Reset restartuje pouze aplikaci. Tlačítko Mode zjistí, zda je zařízení v režimu bootloader nebo aplikace.

IP adresa a port zařízení. Standardně 192.168.12.150, port 3000. Zkratovací propojkou v zařízení lze tuto adresu aktivovat.

Tento režim je určen pouze pro HW 1.2 a dovoluje nastavit typ připojení tachografu v závislosti na variantě osazení HW. Nepoužívat bez znalosti varianty osazení!!!

Konfigurace dotazů na data dle SAE J1939. Dotazy jsou možné pouze s deaktivovaným režimem listen only.



Tlačítka dovolují poslat požadavek na jednorázové čtení dat. Zatržítka aktivují periodické čtení dat.

Změny ve verzích firmware

1.13 boot
- doplněna podpora paketu PACKET_UNKNOWN_PACEKT_ID

1.13
- doplněna podpora paketu PACKET_UNKNOWN_PACEKT_ID
- doplněna podpora paketu PACKET_RESET
- doplněna podpora paketu PACKET_SET_TIMESTAMP
- doplněna podpora paketu PACKET_GET_TIMESTAMP
- doplněna podpora paketu PACKET_FMS_EXT
- paket PACKET_CONFIGURATION již neprovádí automaticky restart firmware. Nutný je restart paketem SHUTDOWN (do bootloaderu) nebo RESET (restart aplikace firmware).

1.20 boot
- přechod na Microchip TCPIP Stack verze 4.55

1.20
- přechod na Microchip TCPIP Stack verze 4.55

1.21
- doplněna možnost čtení sériového čísla
- konfigurační paket doplněn o možnost nastavení masky sítě
- konfigurační paket doplněn o možnost volby připojeného tachografu (vlastní čtení z tachografu Stoneridge a Acta není ještě implementováno, funkční pouze VDO).

1.23 boot
- přidána podpora čtení konfigurace zařízení
- znovu aktivován funkce automatického spuštění aplikace po uplynutí zadaného intervalu
- zrychlena rychlost načítání firmware-aplikace

1.23
- sériové číslo rozšířeno na 10 míst

1.30 boot
- podpora PIC 18F4680 Rev7.

1.30
- podpora PIC 18F4680 Rev7.

1.45
- Podpora připojení tachografu přímo na PIC bez použití extenderu MAX3100.

1.60
- Podpora secondary fuel level a hires total fuel used ve FW pro HW revizi 1.2.

2.00
- Redesign HW na verzi 1.3. Podpora připojení J1708, Změna struktury paketu FMS.
- FW nelze použít pro HW revize 1.2 a starší.

2.12

- Rozšíření množství čtených dat u paketu FMS a FMS EXT

3.00

- Doplněn paket SPEC_ SOR

3.06

- Dotazy na data dle SAE J1939

3.08

- Doplněna podpora DM1 a rozšířen paket SPEC_SOR