OVMS

Open Vehicle Monitoring System

www.openvehicles.com

Renault Twizy v3.6.1
OVMS Hardware Module v2
User Guide v1.6 (2015/10/02)

History

v1.0	2013/01/05	Initial version (RT2.5)
v1.0.1	2013/01/11	Minor fixes & optimization
v1.1	2013/01/14	Firmware release RT2.6
v1.1.1	2013/01/19	MSG commands, OVMS power usage
v1.1.2	2013/04/07	Minor documentation fixes
v1.2.0	2013/12/29	CFG command
v1.2.1	2014/01/02	CFG power map calculations
v1.2.2	2014/01/04	Fixes & clarifications
v1.2.3	2014/01/09	CFG POWER changes
v1.2.4	2014/01/18	SEVCON faults alert & log access, READS & WRITEO
v1.2.5	2014/01/26	Twizy45 support, RT-PWR-Log
v1.2.6	2014/02/16	CFG profile management, SimpleConsole support
v1.2.7	2014/05/30	Trip stats, CFG RAMPL, CFG TSMAP speed support
v1.2.8	2014/06/20	Battery max power levels, SEVCON power shaping info
v1.3	2014/06/21	Lock mode, Valet mode, Homelink
v1.4	2014/08/05	Auto recup, GPS log min/max power
v1.5	2014/09/28	Kickdown function
v1.5.1	2014/11/08	Details on motor and gear box, OVMS error codes
v1.5.2	2015/04/12	GPS & battery monitor log format changes
v1.6	2015/10/02	Firmware release RT3.6

Recent changes
- Firmware release RT3.6:
  - Battery power & energy based on battery current & voltage
  - Separate coulomb (Ah) & energy (Wh) measurement
  - Auto BMS power adjustment extended (ATT: parameter changes!)
  - Battery capacity estimation
  - CFG CLEAR
  - Ideal range calculation with temperature influence
- Firmware release RT3.5:
  - Kickdown function
- Firmware release RT3.4:
  - Auto recuperation adjustment, min/max power statistics, CFG POWER allows up to 139% for power_low, better accel/decel stats
- Firmware release RT3.3:
  - New trip stats&report, battery power levels, Lock mode, Valet mode, Homelink
- Firmware release RT3.2:
  - CFG LOAD/SAVE/INFO, profile selection & display via SimpleConsole
- Firmware release RT3.1:
  - CFG power command names changed (!), power map generation extended

- Firmware release RT3.0:
  - CFG command for SEVCON configuration
- Firmware release RT2.6:
  - Battery monitor extended by standard deviation alert mode
    (includes extension of `RT-PWR-BattPack` data)
  - Power statistics extended by distance / efficiency data
    (includes change of `RT-PWR-UsageStats` data)

Table of contents

Welcome!

The OVMS (Open Vehicle Monitoring System) team is a group of enthusiasts who are developing a means to remotely communicate with our cars, and are having fun while doing it.

The OVMS module is a low-cost hardware device that you install in your car simply by installing a SIM card, connecting the module to your car's Diagnostic port connector, and positioning a cellular antenna. Once connected, the OVMS module enables remote control and monitoring of your car.

There are two ways for you to communicate with the OVMS module:

1. Send text messages from a cell phone to the OVMS module's phone number. The module will respond back via text messaging. If you want, the OVMS module can also send text messages to you when the car reaches certain states, such as if charging is interrupted.

2. Use a smartphone App. Both the OVMS module and the App communicate with an OVMS server via UDP/IP or TCP/IP over the Internet. The smartphone Apps provide a richer experience and more functionality, but they do require a data plan on the SIM car you purchase and install in the OVMS module.

This Guide will help you setup and configure your OVMS module. Initial configuration of the OVMS module is done via SMS. Once configured, you can use either SMS and/or the cellphone Apps to communicate with the OVMS module.

Warning!
OVMS is a hobbyist project, not a commercial product. It was designed by enthusiasts for enthusiasts. Installation and use of this module requires some technical knowledge, and if you don't have that we recommend you contact other users in your area to ask for assistance.

Legal disclaimer: by using the OVMS you agree to do so <u>completely on your own risk</u>. Being a hobbyist project, the OVMS has neither CE approval nor undergone any official EMC tests. It has no ECE approval, so depending on your country <u>may not be legal on public roads</u>.

Warning!
*The OVMS module currently draws about **2.7 Ah per day** from the **12V battery**, which unfortunately has only 12-14 Ah capacity on the Twizy. If you do **not charge or drive for <u>more than 3 days</u>, you **must charge the Twizy or unplug the OVMS** to avoid deep discharge of the 12V battery! Failure to do this can result in unrecoverable failure of the 12V battery. If the car does not power up, that can be due to low voltage on the 12V battery: try charging! The OVMS will send an alert if it detects the voltage dropping below 11.5V reminding you to charge in time.*

Parts needed

You can buy all parts at the OVMS hardware partner Fasttech:

> http://www.fasttech.com/link/ovms

Note: Fasttech is shipping from China, import customs and tax will apply for EU.

For each Twizy you'll need:

- Universal GPS Antenna (SMA Connector)
  GPS Antenna: SMA: $4.89
- OpenVehicles OVMS GSM Antenna
  GSM Antenna: $2.50
- ODB-II to DB9 Data Cable for OVMS (left)
  ODB-II OVMS Cable: $9.50
- OVMS Car Module v2
  OVMS Module: $99.00

If you want to avoid having two antennas: There are combi antennas integrating GSM & GPS available (e.g. "shark fin antenna"). If you test one of these, please report your results :-). Both antenna connectors are SMA, an active GPS antenna needs to run on 3 V.

To update the OVMS firmware, you'll also need one of these:
- PICKIT 2 Compatible Programmer/Debugger
  PICKIT 2 Compatible Programmer: $16.99
- PICKIT 3 Compatible Programmer/Debugger
  PICKIT 3 Compatible Programmer: $26.28

If you want to do OVMS development and/or debugging, a serial interface or serial to USB adapter will be needed.

SIM card

You'll need a **standard size** SIM card with a data plan. The Twizy module will normally need about **1-10 MB of data per month**, depending on your driving and GPS logging. Data though will be sent in small and infrequent packets, so be aware of providers rounding up prepaid data transfers (like Congstar.de) or book some minimal flat rate. Also, mobile **GPRS coverage and stability** depends on the network; for Germany, D1 (T-Mobile) offers best coverage and stability, followed by D2 (Vodafone). E-Plus is usable in urban areas, O2 is not recommended.

Some pointers for Germany (conditions may have changed!):

- T-Mobile "Xtra Call" + "Xtra Flat Daten": D1, best coverage, flat rate required, limit 50 MB (should be sufficient in most cases)

- Congstar "Prepaid Starter" + "Surf Flat Option 200": D1, best coverage, flat rate required, limit 200 MB

- Allmobil: D2, good coverage, fair rates, no flat rate required

- Blau "9 cent Tarif": E-Plus, usable urban coverage, fair rates, no flat rate required

Follow your provider's guidelines on activating the SIM card (full activation can take up to 24 hours). The card needs to be unlocked to start without PIN entry. This can easily be done using your mobile phones SIM card management App. Using your mobile phone you can also test the card activation status and GPRS access.

Firmware update / SIM installation

See "Firmware Easy-Install" on the OVMS home page.

Before closing the OVMS, insert the SIM card and make sure the switch besides the LEDs is set inwards (outwards deactivates GPRS).

Dry run

For advanced users: if you want to "dry run" the module on your desk before installing into your car, you just need to provide 12 V =. The module needs about 110 mA. Connect GND to the OBD-II plug at pin 4 and +12V at pin 16:
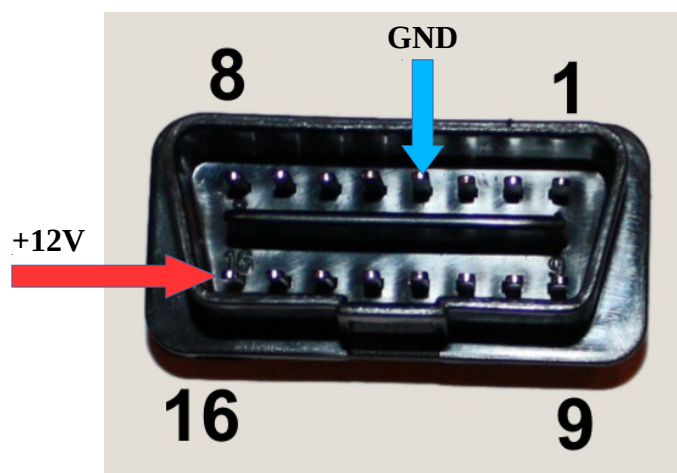
If you connect the antennas, you can fully interact with the module on your desk using SMS commands and the OVMS Apps.

You can also connect to the DIAG port serial interface (9600 baud, 8N1, no handshake) to see the

module working. If you want to preconfigure modules for other users, you can also use the DIAG mode to enter configuration commands without need of using the GSM network. See document "Firmware-Development" for details.

Car installation

The OBD-II port (diagnostic port) is located at the bottom of the left front glove box. Remove the connector cover and plug in the data cable (only possible in one orientation). Attach the antennas and connect the module to the data cable. The module will power on as soon as the data cable is connected.

For a first installation, it's sufficient to simply put the complete OVMS module and both antennas into the left front case. This will normally provide enough signal quality for some first tests, may even be sufficient for productive use – try out.

There are many options for a better placement of both the module and the antennas. For example you may want to put the module in the right (lockable) glove box – for a first simple installation, the data cable will fit under the glove box cover, you may later decide to hide it under the dashboard.

Antenna position should be best at the center top back of the Twizy (there's no steel frame). Cables can be put under the plastic body covers, see forums for instructions on how to unmount those. You may want to leave some space between the GSM and GPS antenna, as the GSM antenna can emit short high power bursts.

Configuration

Basic registration and configuration of the Twizy OVMS is basically the same as for the Tesla Roadster, except the **vehicle type** needs to be "RT" (for "Renault Twizy").

Follow the Twizy FAQ or Tesla Roadster configuration guide. After "REGISTER" and "PASS", you should first send the "MODULE" command with the fourth argument set to "RT", for example:

```
MODULE <your-car-id> K SMSIP RT
```

This will activate the Twizy specific vehicle data processing and command extensions.

The module should now respond to the "VERSION" command like this:

```
OVMS Firmware version: 2.8.1/RT3.6.1/V2
```

This is the framework version, the vehicle type + version and the hardware version.

If you haven't done already, turn the Twizy on, so the module can read the diagnostic data from the CAN bus. **The Twizy CAN bus is available only while the car is on or charging.**

Now, send a "STAT?" SMS to the module. It should reply with the Twizy specific status message looking like this:

```
Not charging
Full charge: 18 min.
Range: 59 - 78 km
SOC: 97.38% (50.00%..97.38%)
ODO: 4437 km
```

Next, send a "GPS?" SMS to the module to query the current coordinates. Please note the module may need about a minute after setup as "RT" to get the first GPS fix.

Twizy specific commands & capabilities

The OVMS framework defines a common set of commands and capabilities originating at the Tesla Roadster implementation (see Tesla guide). As the framework currently evolves to cover any kind of vehicle, the commands and capabilities now become dynamic properties of the vehicle type.

Some common commands will have no effect on the Twizy, for example locking and unlocking the car. Some common diagnostic capabilities are "virtual" on the Twizy (derived), and some capabilities are unavailable.

The Twizy also changes some standard command behaviour and adds some new commands.

The Apps currently only provide a user interface for the standard capabilities but will support the vehicle specific configuration in the future. At the time writing, most Twizy specific functions need to be adressed by SMS, nevertheless the Apps still provide a nice UI for the standard functions.

(Todo: Feature support matrix)

Overview of Twizy SMS commands

Commands	Function
STAT?	Status output (extended)
RANGE ...	Ideal range configuration
CA ...	Charge alerts (SOC/range)
BATT ...	Battery monitoring
POWER ...	Power usage statistics
CFG …	SEVCON configuration

Extended Status

The Twizy includes the following information in the SMS command "STAT?" response as well as battery status text messages sent to the App:

- Charge state: one of
  - "Not charging"
  - "Charging"
  - "Charging Stopped" – if charge was interrupted
  - "Charging, Topping off" – if SOC is above charge alert or 94%
  - "Charging Done" – if fully charged
- Charge power sum (Wh)
  - ...at the battery plus grid energy estimation in Wh (**V3.6**)
- Time estimation for full charge (minutes)
- Range (unit as configured)
- SOC (State of charge in %)
- Odometer (unit as configured)
- Battery capacity estimation in % and Ah (**V3.6**)

Being able to rely on a charge working as planned is crucial. If charging is interrupted, the OVMS will automatically send the extended status message by SMS and/or IP (as configured). So you'll be informed immediately if some fuse blows or someone pulls the plug.

The status message will also be sent if charging starts after an interrupt, and when fully charged. You may also configure a minimum SOC alert (using standard FEATURE #9), and a charge alert for a sufficient range and/or SOC (see "Charge alerts").

Since **V3.6** there's also an option to request the status message for every charge start. This is useful to monitor remote or time controlled charging, for example in combination with the EDImax or Homematic home automation scripts. To let the module send the charge start status message, add 64 to FEATURE #14 (car bits).

Charge time estimation

The SOC value of the Twizy (any Lithium SOC value to be precise) is a battery model based estimation and will be quite wrong sometimes (too low normally, especially on low temperatures). The Twizy recalibrates it's SOC to 100% when full charge is reached.

As the charge time estimation is designed to be on the safe side, times normally will be better by 5-30 Minutes depending on the actual cell state and the amount of error in the Twizy estimation.

Due to the Lithium chemistry and battery pack structure (cell balancing), the charge speed is nearly linear up to ~94% SOC and much slower on the last ~6%. A typical charge curve looks like this:
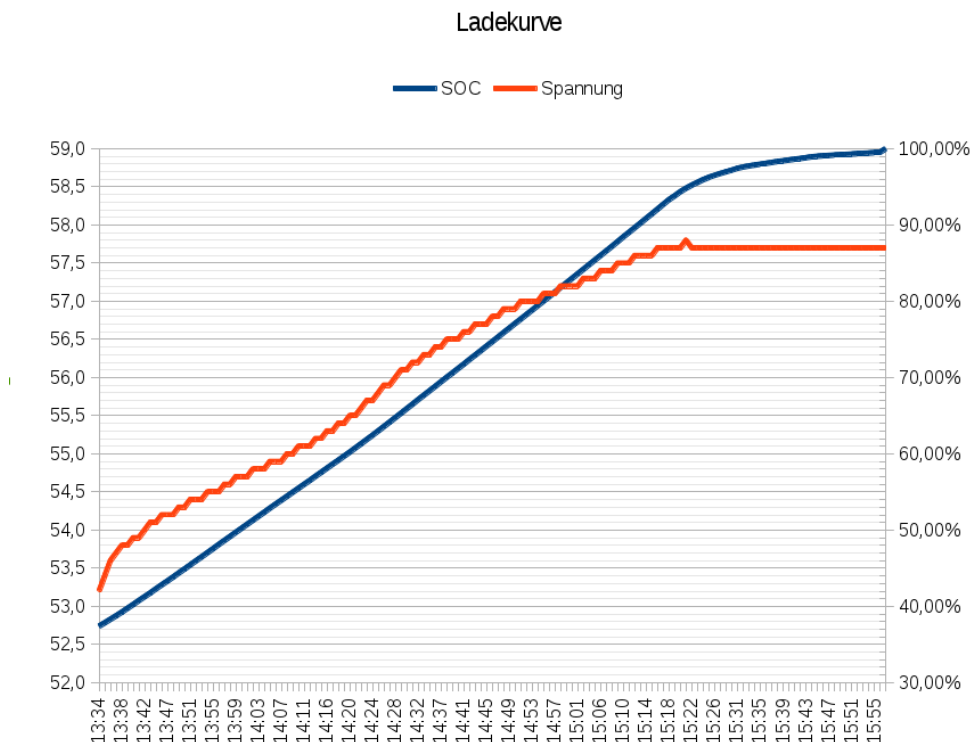
(BTW: you can draw graphs like that by using the "RT-BAT-P" records, see below!)

Time estimation takes this non-linearity into account by dividing the charge process in two sections (0-94% = 180 minutes / 94-100% = 40 minutes).

Keep in mind: if you need the quickest possible intermediate charge, start with a warm battery and stop at or below 95%.

Range

Range will include two values, an estimated and an ideal range. The ideal range is based on your max range configuration (see "Ideal range configuration"). Estimated range is based on the Twizy's own range estimation, which in turn is based on the last 3 km driving. This will be scaled up to the current SOC during charge, so will differ from the value, the Twizy will tell you on the next key-on (as that is a different Twizy estimation based on the last 150 km). As soon as you turn on the Twizy, the module will update it's estimation as well.

Since **V3.6**, the ideal range takes the temperature influence into account. A cold battery loses range in the typical magnitude of 0.6 km / °C. The max range is assumed to be at 20 °C, so for example at 10 °C the ideal range of a full battery will be 6 km lower than the max range configured.

Please note: The OVMS firmware internally stores odometer and range values in miles. Until that can be changed, you will occasionally see a 1 km difference between the range displayed in the dashboard and the OVMS outputs. This is due to the fact integer mile values do not provide enough precision to cover all possible km values. Conversion math has been tuned to minimize this error.

SOC

The SOC output includes the current SOC and the current SOC usage range (lowest & highest SOC). The SOC range reflects your last battery usage or charging cycle.

As mentioned above, the SOC is a model based estimation of the Twizy. So it's quite normal it will sometimes be wrong. To protect the battery from over-discharging, it will normally tend to show you a SOC lower than real. There's currently nothing the OVMS can do about that.

Grid energy estimation (V3.6)

Prior to V3.6, the energy use was derived from the battery current. That also applied to the Wh sum for charging. Beginning with V3.6, the real battery power is calculated from the battery current and voltage.

This now allows to measure how much energy really is stored in the battery and later available for traction. The data observed from this measurement and power meter readings shows the typical grid to battery efficiency for the Twizy is ~88.1% for the main charge and ~86.4% including a full 12V charge cycle.

The grid energy estimation shown by the STAT message calculates from the battery energy with the mean efficiency of 87.2%. Real grid usage is normally quite close to this for typical charges, but keep in mind this is an estimation, don't expect this to be as exact as a physical power meter.

Battery capacity estimation (V3.6)

The STAT output displays the battery capacity in percent and absolute Ah.

100% capacity is assumed to be equivalent to 108 Ah. The nominal Ah capacity of the battery is unknown up to now, the best assumption matching the available data is 120 Ah, with 90% usable capacity at 108 Ah.

The capacity estimation is updated every time charging from below 40% to 100%.

Estimation is the more precise the lower the SOC at charge start. To smooth out SOC and measurement errors, the estimation builds the mean value over the last 10 charges with sufficient discharge depth.

To reset the mean value and get a direct single capacity estimation, set feature #13 to 0 before the charge process reaches 100%. Feature #13 stores the running average capacity value (percentage) persistently, so it's not lost over power losses or reboots. Note: feature #13 will be shown as an integer value by the app and feature queries, but the internal precision is at 1/100 %.

Background

Lithium-ion batteries lose capacity by aging (calendar time) and usage (charge cycles). There are lots of parameters for the capacity degradation. Generally, storing with high SOC at very high temperatures, charging with high currents at very low temperatures and deep discharging are the primary causes for raised degradation.

The Twizy battery is normally rented, with Renault providing a new battery if the capacity drops

below 75%. We need to pay for a capacity check by Renault if the result is above 75%, so we should be sure it's worth the check when we reach that point.

The capacity of a lithium based battery cannot be measured directly, instead it needs to be derived from other measurements – just like the SOC. There exist some advanced instruments probing for the voltage drop under load (elasticity), but the basic method available to us is to check the capacity by discharging & charging and measuring the energy amount (coulomb count) that fits into the battery, then relate that to the nominal capacity.

Ideally we should discharge to 0%, then charge to 100%. That's because these are the most reliable SOC points the BMS can provide. Keep in mind, the SOC is also just an estimation derived by complex algorithms from coulomb counting and many other parameters of the BMS. Even 0% is not necessarily really "empty", it's just the BMS saying "no more discharging, please".

The charging process of a lithium battery consists of basically two phases, the "CC" (constant current) followed by the "CV" (constant voltage) phase usually beginning around 94% SOC. The CC phase current is not necessarily constant though, for example if it needs to be cut back due to high temperature.

Capacity can be given in Wh or Ah. Capacity measurement based on the Wh sums induces an error due to the energy discharge/charge efficiency and inner battery resistance related side effects. The coulombic discharge/charge efficiency of a lithium-ion battery on the other hand is always near 100%, so is the best choice for capacity measurements.

The OVMS capacity algorithm detects the end of the CC phase by monitoring the battery voltage during charge. It scales the coulomb count measured during the CC phase from the CC phase SOC window to a theoretical full range CC phase beginning at 0%. It then adds the coulomb count from the CV phase, resulting in the overall Ah capacity. The percentage is then calculated based on the assumption of 108 Ah default usable capacity.

Ideal range configuration

The Twizy's range estimation is based on the energy use of the last 3 km driving, and it will drop faster on high usage than it will raise on low. So if you happen to have a hill climb at the beginning of your tour, the estimation will be completely wrong for a long time.

Most people don't change their driving style and routes frequently and will get about the same range from a full charge most of the time. So, using the OVMS, you can configure your personal maximum range to let the OVMS do a second "ideal" range estimation just on base of the current SOC.

Since **V3.6**, the ideal range takes the temperature influence into account. A cold battery loses range in the typical magnitude of 0.6 km / °C. The max range is assumed to be at 20 °C.

Enter ranges in the unit configured (assumed "K" = km here).

Command	Function
`RANGE 80`	Set ideal maximum range at 20°C to 80 km
`RANGE?`	Query current ideal maximum range setting
`RANGE`	Disable ideal range estimation

The range setting is stored in feature slot #12, so can also be queried by the OVMS "`FEATURES?`" command.

Charge alerts

If you need to wait for a charge to get somewhere (home for example), you can let the OVMS notify you as soon as a sufficient charge level has been reached.

Sufficient charge alerts can be triggered by a SOC value and/or by a range. If both triggers are set, it will use the first one reached.

Command	Function
`CA 30`	Enable charge alert on 30 km range
`CA 60%`	Enable charge alert on 60% SOC
`CA 30 60%`	Enable charge alert on either range or SOC
`CA?`	Query current charge alerts and charge time estimations
`CA`	Disable charge alerts

Range alerts will be triggered on ideal range reached. If you want to use the Twizy range estimation, disable the ideal range estimation for that charge.

All "`CA`" command responses includes a charge time estimation for the nearest alert threshold set as well as for a full charge.

Charge alert thresholds are stored in feature slots #10 (SOC) and #11 (range), so can also be queried by the OVMS "`FEATURES?`" command.

Battery monitoring

Although the battery is leased, it's health status still is of vital importance for us users. If you want to or need to rely on your car, you must be able to detect all signs of potential problems, before they become hazardous.

Fortunately, the Twizy BMS (battery management system) outputs some values on the CAN bus, so the OVMS can provide a little more insight about the battery status than just the SOC. Values currently identified on the CAN bus include cell voltages, pack voltages and cell module temperatures.

Some tech background

The battery pack of the Twizy consists of 42 cells, arranged in 7 modules of 6 cells each. Within a module, cells are layed out 2S3P, that means there's a total series of 14 packs of 3 parallel cells each. Cell chemistry is supposed to be LiMn based, so each cell has a theoretical voltage range of about 2,5 - 4,3 V, nominal voltage around 3,6-3,7 V. Renault restricts the usable SOC window to maximize battery cycle life. Charging top voltage is 57,8 V for the pack = ~4,13 V per cell.

Important for good performance of a battery pack is that all cells in a serial configuration are at about the same voltage level and temperature. If a single cell drifts out of line in either direction, that's a sign of some possible defect.

Now there are 14 cell voltages and 7 module temperatures on the CAN bus, plus two pack voltages. So, what the OVMS can do to detect potential problems, is to monitor all cell voltages and temperatures, and to produce an alert if it detects a critical deviation. It also collects the maximum deviations from the mean values over a usage cycle and sends these statistics on SMS request as well as to the OVMS server for further analysis and evaluation.

The limits are, the 14 voltages are collective for 3 parallel cells each, and the 7 temperatures each cover a module of 6 cells. So we're not quite on the single cell level yet, but it's sufficient to detect critical situations.

Battery max power levels

The Twizy BMS shapes the available maximum power levels for charging and discharging the battery pack depending on parameters like SOC and temperature. This info is sent on the CAN bus every 100 ms and read by the SEVCON (while driving), which in turn uses the limits to control maximum motor drive and recuperation power levels.

Typical Twizy power limit curves look like this:
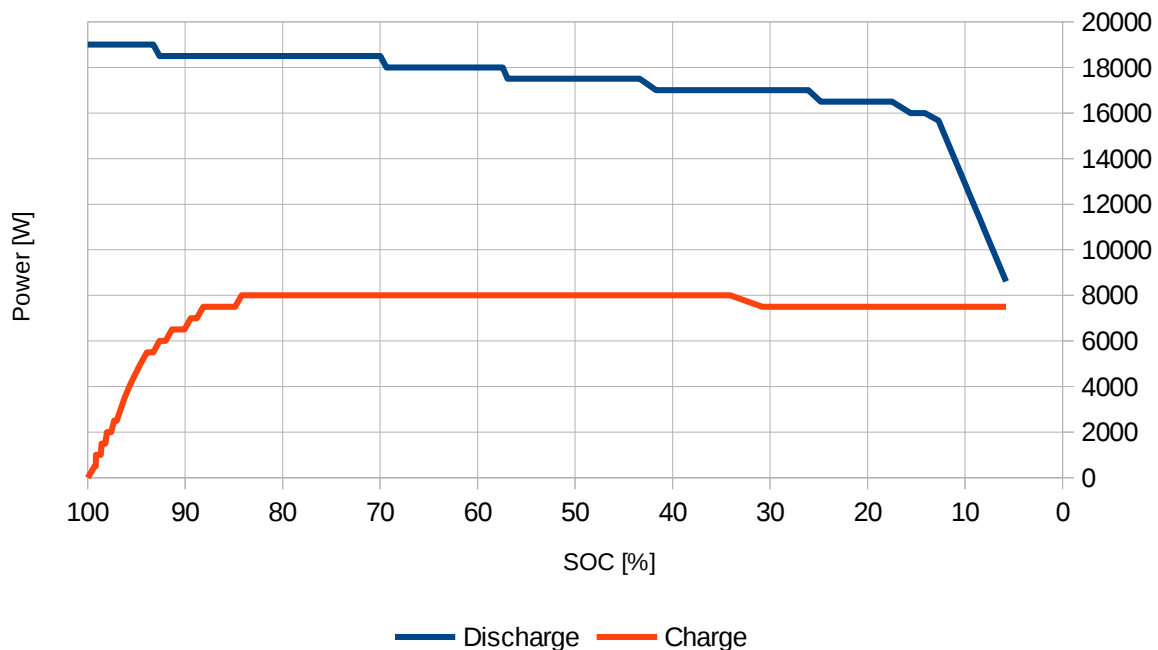
This graph additionally shows the default SEVCON linear drive power cutback below 15% SOC.

The **SEVCON** will limit **motor power** levels to the "motor peak electrical power" as configured in SDO 0x3813.23 which defaults to 18000 W. As you can see above, the battery will do 19000 W if fully charged and 18500 W above 70% SOC, so this may be a tuning option.

The SEVCON also cuts back recuperation at the upper and drive power at the lower SOC end. The **recuperation cutback** point is configured in SDO 0x2872.02 and defaults to -35 A (CFG WRITE needs this 16 bit encoded, so this is 65536 - 35 = 65501). 35 A is normally available below ~97% SOC (around 2000 W charge pwr / ~57.2 V battery voltage). If you set this to e.g. -20 A, you will get recuperation earlier. Take care not to overcharge your battery (i.e. if you live up a high hill...).

The **drive power cutback** is configurable as well. The default map begins cutback at 15% SOC with 100% power and ends at 0% SOC with 15% power left. That's more a psychological trick than a technical necessity, as the BMS itself will also lower the discharge power level to protect the battery. If you don't need the cutback you may change the map using CFG WRITE:

Cutback start SOC:	SDO 0x3813.13 = <SOC%> * 50	Default=15%
Final power level:	SDO 0x3813.12 = <Power%> * 50	Default=15%

Using the battery monitor

The battery monitoring needs no configuration. It's enabled by default and works automatically in the background and normally will only alert you about critical conditions.

Critical (=alert) conditions are:

- Single cell voltage offset from mean value >= 30 mV

- – Standard deviation of cell voltages >= 25 mV

- – Single module temperature offset from mean value >= 3 °C

- – Standard deviation of module temperatures >= 3 °C

In addition to critical conditions, the monitoring will also set a "watch" flag if the voltage or temperate has an offset from the mean value that is higher than the current standard deviation of all cells / modules, or if the standard deviations higher than normal. You may want to keep an eye on these cells, if the deviations tend to raise over time, these may be the source of performance loss or even become the part that breaks.

Standard deviations are a measurement of the overall cell drift = overall pack health. A healthy pack will have a maximum voltage standard deviation of 5-10 mV and a maximum temperature standard deviation of 1-2 °C. Standard deviation "watch" status is triggered at voltage deviations from 15 mV and temperature deviations from 2 °C.

All battery statistics and alert flags are **automatically reset each time you start the Twizy**. So if an alert occurs, you can easily see if it was a temporary stress issue or if it persists with the next drives.

Remember, the CAN bus is offline while the car is switched off, so temperatures and voltages are not updated while the car is off. The last values read from the bus will be kept by the OVMS.

Command	Function
BATT -or- BATT S	Output current alert and watch status. This will be sent automatically if any alert condition changes, but not on watch conditions. Example: `Volts: SD:5mV ?C1:+10mV Temps: SD:1C OK` Read: Max standard deviation has been 5mV / 1°C. Cell 1 had a max deviation of 10 mV above average, thus has been tagged "suspicious". An alert would be tagged by a "!". Temperatures have been nominal.

BATT V	Output current voltage levels.
	
	Example:
	`P:57.50V ?1:4.120V 2:4.110V 3:4.110V 4:4.110V`
	`5:4.110V 6:4.110V 7:4.110V 8:4.110V 9:4.110V`
	`10:4.110V 11:4.110V 12:4.105V 13:4.105V 14:4.120V`
	
	Read: Pack voltage is 57.5 V, followed by the 14 cell voltages. Note the "?" tagging the suspicious cell. Precision of cell voltages is 0.005 V.

BATT VD	Output collected max voltage deviations.

Command	Function
	Example: `SD:5mV ?1:+10mV 2:+0mV 3:+0mV 4:+0mV 5:+5mV 6:+0mV 7:+0mV 8:+0mV 9:+0mV 10:+0mV 11:+0mV 12:-5mV 13:-5mV 14:+10mV` Read: Standard deviation was at most 5 mV. Cell 1 (suspicious) was at most 10 mV above average, cell 12+13 had a max deviation of 5 mV below average. Note cell 14 also once had +10 mV deviation, but that did not trigger the "watch" flag, so was inside standard deviation at that time.
`BATT T`	Output current cell module temperatures. Example: `P:10C (8C..10C) 1:10C 2:10C 3:10C 4:10C 5:10C 6:10C 7:10C` Read: Current pack temperature is 10 °C, module temperature range of last usage was from 8..10 °C. Following are the 7 module temperatures, no watches or alerts.
`BATT TD`	Output collected max temperature deviations. Example: `SD:1C 1:+0C 2:+0C 3:+1C 4:+0C 5:+0C 6:+0C 7:-1C` Read: standard deviation max 1 °C, followed by the 7 max deviations during the last use, no watch or alert flags.
`BATT R`	Reset battery monitor. Normally automatically done on each switch-on, the command enables resets during charge or without turning off the car.

More details and log records may be queried from the OVMS server, using historical message records of types "`RT-BAT-P`" and "`RT-BAT-C`". See "Perl client" on how to retrieve these.

Battery pack record format:

Note: the format of the battery pack and cell records changed on Version 3.5.2.

```
RT-BAT-P,<timestamp>,0,
        ,<volt_alertstatus>,<temp_alertstatus>
        ,<soc>,<soc_min>,<soc_max>
        ,<volt_act>,<volt_min>,<volt_max>
        ,<temp_act>,<temp_min>,<temp_max>
        ,<cell_volt_stddev_max>,<cmod_temp_stddev_max>
        ,<max_drive_pwr>,<max_recup_pwr>
```

- "alertstatus" can be one of
  - 0 = unknown
  - 1 = nominal
  - 2 = watch
  - 3 = alert
  - 4 = failure
- SOC values are in 1/100 %
  - SOC min+max reflect the current discharge/charge cycle
- Voltages are in 1/10 V
  - ...except standard deviation maximum, which is in mV
- Temperatures are in °C
- Voltage and temperature min & max values are from the current use cycle (since last switch-on)
- Max powers for drive (discharge) & recuperation (charge) are in 100 W, i.e. "185" = 18500 W. Note: these are the Limits delivered by the BMS, the SEVCON may overlay some limits, see above.

Example:

```
RT-BAT-P,2013-01-01 19:06:45,0
     ,2,1
     ,8612,8612,9698
     ,561,539,574,410
     ,10,8,10
     ,5,1
     ,185,25
```

(line breaks added for readability)

Battery cell record format:

```
RT-BAT-C,<timestamp>,<cellnr>
        ,<volt_alertstatus>,<temp_alertstatus>,
        ,<volt_act>,<volt_min>,<volt_max>,<volt_maxdev>
        ,<temp_act>,<temp_min>,<temp_max>,<temp_maxdev>
```

- "alertstatus" is encoded as above (battery pack)
- Voltages are in mV (but Twizy sensor resolution is 5 mV)
- Temperatures are in °C
- "maxdev" = maximum deviations, these are signed to denote the direction

Example:

```
RT-BAT-C,2013-01-01 19:06:45,1
     ,2,1
     ,4015,3865,4115,15
     ,10,8,10,1
```

(line breaks added for readability)

Power usage statistics

One of the things the Twizy lacks is a usable ecometer, to give you feedback on your driving style. The builtin ecometer (the four circles) is not very helpful in optimizing power usages.

The power statistics of the OVMS is a first attempt (read: work in progress) to provide some more detailed information about your power usage profile. It currently is limited to text message and server records, a graphical user interface needs to be implemented in the App.

Power statistics currently are collected in two categories, speed and level. The speed category is divided into three sections, constant speed, acceleration and deceleration. The level category is divided into two sections, up and down. In each section, used and gained (by recuperation) power will be collected separately.

Be aware, that level change detection depends on the GPS altitude. So, if GPS altitude is unavailable or inaccurate, so are level power usages.

Power sums are currently totals of Wh for the sections, collecting section track lengths is on the todo list. Until then you might consider correlating the GPS log with the power usage log.

Important note: the interpretation of the CAN value as **power** ("W"/"Wh") prior to **V3.6** has been **false**. The correct interpretation is **current / coulombs** ("A"/"Ah"), to get these divide the old "W" values by 64.

Since **V3.6**, both values are delivered separately by the OVMS. The power and energy values are calculated from the actual battery current and voltage (power = current * voltage). The current and coulomb counts can be seen as very precise, while power and energy may contain measurement errors: currently the only identified battery voltage info on the CAN bus has a time resolution of 1 second (vs. 10 ms for the battery current), so the power calculated from that will contain an error during a second with a sharp voltage jump (i.e. from sudden high currents).

New for V3.2.1 are speed and acceleration averages. These are only taken from driving phases, i.e. moving with at least 1 kph. A speed change of more than 1 kph per second is interpreted as an acceleration or deceleration phase.

Note: the acceleration and deceleration averages were too high before **version 3.4.0**. Version 3.4.0 improves the calculation but is still just an estimation.

Power statistics are currently reset when switching the Twizy on, and the report is sent after switching the Twizy off, it at least 25 Wh have been consumed. No configuration is currently needed, statistics are enabled by default and will run in the background.

To suppress the power statistics notifications, add 16 to feature #14 (car bits). This will suppress all informational notifications.

Command	Function

`POWER`
`-or-`
`POWER E`
	Output current power efficiency report. This is also sent automatically when turning the Twizy off (and at least 1 Wh was used).
	
	Example:
	`Trip 23.4km 32.7kph 132Wpk/6% SOC-12.5%=57.1%`
	`=== 33% 93Wpk/7%`
	`+++ 34% 4.7kps 204Wpk/1%`
	`--- 33% 3.2kps 97Wpk/13%`
	`^^^ 65m 108Wpk/14%`
	`vvv 33m 140Wpk/7%`
	
	Read: trip length was 23.4 km with an average speed of 32.7 kph. Total efficiency on that trip was 132 Wh/km, and 6% of the energy used has been regained by recuperation. 12.5% SOC has been used for the trip, the new SOC is 57.1%.
	
	The following three lines give the speed sections with their percentage of the whole tour. Accel (+++) and decel (---) phases also include the average acceleration in kps = kph per second.
	
	Up (^^^) & down (vvv) finally show the level sections, the meters are the totals of height differences collected.

`POWER T`	Output current power usage totals report.
	
	Example:
	`Power -299 +16 Wh 2.1 km`
	`Const 33% -69 +5 Wh`
	`Accel 34% -150 +1 Wh`
	`Decel 33% -80 +11 Wh`
	`Up 65m -76 +11 Wh`
	`Down 33m -76 +5 Wh`
	
	Read: trip length was 2.1 km, total power used was 299 Wh, gained 16 Wh. Following three lines give the speed sections with their percentage of the whole tour. Up & down show the level sections, the meters are the totals of height differences collected.

Besides text notification, the function will send power usage records to the OVMS server once per minute. The history message type is "`RT-PWR-Stats`", see "Perl client" on how to retrieve these from the server. Hold time is 24 hours.

ATT: in **V3.6**, the previous record name "RT-PWR-UsageStats" has been shortened to "RT-PWR-Stats", the fields have been reordered and units are now converted. Also coulomb count

discharge/charge sums (Ah) have been added, so you will need to change processing based on this data.

Since Twizy release 3.1.3, another history message type "RT-PWR-Log" will be sent each time the car is switched off or a charge alert event occurs (i.e. charge done or stopped). You can build nice long term power usage statistics from these. The server is told to store these log entries for 365 days, but don't rely on that. The shell script "rt_fetchdata.sh" will fetch these records as well.

Power usage record format:

```
RT-PWR-Stats,<timestamp>,0
 ,<speed_CONST_dist>,<speed_CONST_use>,<speed_CONST_rec>
 ,<speed_CONST_cnt>,<speed_CONST_sum>
 ,<speed_ACCEL_dist>,<speed_ACCEL_use>,<speed_ACCEL_rec>
 ,<speed_ACCEL_cnt>,<speed_ACCEL_sum>
 ,<speed_DECEL_dist>,<speed_DECEL_use>,<speed_DECEL_rec>
 ,<speed_DECEL_cnt>,<speed_DECEL_sum>
 ,<level_UP_dist>,<level_UP_hsum>,<level_UP_use>,<level_UP_rec>
 ,<level_DOWN_dist>,<level_DOWN_hsum>,<level_DOWN_use>,<level_DOWN_rec>
 ,<charge_used>,<charge_recd>
```

- All speed & level "use" and "rec" values are in Wh

- "dist" values represent distances driven, unit is 1/10 m on speed sections and 1 m on level sections

- "hsum" values are totals of height changes in meters

- Speed "cnt" are in 1/10 seconds, "sum" are 1/100 kph

- Charge sums (coulomb counts) are in Ah

Drive & charge power log record format:

```
RT-PWR-Log,<timestamp>,0
 ,<OdometerKm>,<Latitude>,<Longitude>,<Altitude>
 ,<ChargeState>,<ParkTimeMin>
 ,<SOC>,<SOCMin>,<SOCMax>
 ,<PowerUseWh>,<PowerRecWh>,<DistanceDriven>
 ,<RangeEstimKm>,<RangeIdealKm>
 ,<BatteryVolt>,<BatteryVoltMin>,<BatteryVoltMax>
 ,<BatteryTemp>,<BatteryTempMin>,<BatteryTempMax>
 ,<MotorTemp>,<PEMTemp>,
 ,<TripLengthKm>,<TripSOCUsage>
 ,<TripAvgSpeedKph>,<TripAvgAccelKps>,<TripAvgDecelKps>
V3.6:
 ,<charge_used_ah>,<charge_recd_ah>,<batt_capacity_prc>
```

- ChargeState: 0=not charging, 1=charging, 2=top off, 4=done, 21=stopped charging

- Charging power will be logged in PowerRecWh

- **V3.6**: Charge used/recd (Ah) and battery capacity (%) added

- **V3.6**: all Wh values now derived from real battery current & voltage (see above, to get Ah from previous Wh data, divide by 64)

Example:

```
RT-PWR-Log,2014-01-26 18:29:28,0,
      13365.22,51.257704,7.160899,201,
      4,113,
      100.00,32.08,100.00,
      0,4536,0,
      64,80,
      57.8,49.9,57.8,
      21,6,22,
      27,44 ...
```

(line breaks added for readability)

Note: the parking timer is not accurate – if you need more accuracy, use the timestamps.

SEVCON controller access & configuration

Warning!
USING THIS CAN VOID YOUR RENAULT WARRANTY!
USING THIS CAN BREAK YOUR TWIZY!
USING THIS CAN MAKE YOUR TWIZY ILLEGAL FOR PUBLIC ROADS!
HIGHLY EXPERIMENTAL FEATURE!

Legal disclaimer: by using the OVMS you agree to do so <u>completely on your own risk</u>. Being a hobbyist project, the OVMS has no means to test configuration changes on multiple vehicles or performance situations. A change that seems usable on one Twizy <u>can completely break</u> another one. All configuration changes can lead to <u>losing your public roads allowance</u>.

Ok, you've been warned.

<u>Please note</u>: all values mentioned in this document refer to the Twizy 80 if not explicitly named. The Twizy 45 has other limits, see spreadsheet "Twizy-Limits".

Motor

The Twizy motor is a **Letrika Iskra AMV7118** AC induction motor:

Renault specs the Twizy 80 with continous power of 8 kW, peak 13 kW and torque of 57 Nm. According to the motor label it's specified at 12 kW for 1 minute (IEC S2 = short-time duty cycle).

The most similar generally available motor from Letrika seems to be the AMV7122 contained in their L7e conversion kit:

> http://www.letrika.com/katalog/basic.html?lang=E&tab=0&koda=IM7043

The PDF specsheet for the conversion kit says it's also rated at 12 kW nominal power for 1 minute and can do a maximum power output of 15 kW for 30 seconds, and a maximum torque of 70 Nm at 400 A and a maximum speed of 8.000 rpm.

Controller

The Twizy motor controller is a **SEVCON Gen4™**, a masterpiece of current AC controller technology:

> http://www.sevcon.com/ac-controllers/gen4%E2%84%A2.aspx

The controller can be configured using the standard SEVCON DVT (or older DriveWizard) software package or any CANopen configuration/monitoring software and a standard CAN bus controller... oh, or using the OVMS of course ;-)

To be able to make a complete **backup of your controller configuration**, you currently need to use the SEVCON DVT software. The OVMS cannot store a backup due to lack of memory.

There are two command classes, low level and macro.

If you want to dig deep, please read the Gen4 Product Manual first to understand the basic principles of operation and features of the controller. Especially chapters 5 (System design) and 6 (Configuration) will provide valuable information. You should also read through the DVT software manuals and help system.

If you're in a hurry, the OVMS provides some useful macro commands for the most common basic configuration tweaks. These *should* need no backup as they provide "reset" commands – **but you never know**.

Gear box

The Twizy uses a generally available differential gear box with two-stage of reduction and parallel axes created by **COMEX S.p.A.** Italy. Specsheet can be downloaded from here:

> http://www.comexspa.com/index.php/en/products/item/38-differential-gearboxes

There are two oddities about this.

The gearbox is specified with max output torque of 1000 Nm. Considering that the Twizy 80 has a ratio of 9.23:1 (Twizy 45: 13.4:1), this means the maximum input torque can be as high as 1000 / 9.23 = 108.34 Nm (Twizy 45: 74.63 Nm), so there's lots of headroom for tuning on the torque side.

But the gearbox is also specified at a maximum input speed of just 7000 rpm. That's the first oddity: the default config of the Twizy 80 already exceeds this, as the default max speed is at 7250 rpm. So if you're driving max speed with an unmodified Twizy, you're already outside of the manufacturer specs.

The second oddity is that the manufacturer recommends to use an SAE gear box oil 80W90, while Renault uses ELF Tranself TRJ 75W80 API GL5 EP.

A leaked info on this claims the Renault oil is medium quality and caused overheating during testing, and that the oil retention ring in the gearbox is made of polyamide and will start to wear out quickly if gearbox oil temperatures exceeds 74° Celsius.

The **recommendation** from the same source is to **exchange the gearbox oil** with a high quality 80W90 API GL5 gearbox oil with EP (extreme pressures) additives and change it every year,

especially for increased motor torque, because a non-fresh oil caused "fretting" on the gears during testing.

While the source is unofficial, this explains most of the problems Twizy owners experienced with the gear box, especially oil loss and noise.

Exchanging the oil is easy and has been done by many Twizy drivers with good, sometimes great improvements on the gearbox noises. There's more info on this on the forums.

Another modification to think about is to replace the polyamide oil gasket by some better quality, especially if you're already losing oil or if you're going to travel long distances at high speed.

Some more technical background

Current

Technical limitations apply especially on the battery and motor current levels.

Battery current needs to be limited to ensure a long life. Lithium batteries are especially endangered by high charging currents under low temperature conditions, this can lead to irreversable downgrading due to plating of metallic lithium on the anode. Healthy limits for Lithium type batteries normally are 2C continuous for discharging, 0.5C continuous for charging. 2C means 2 times the capacity of the battery. Continuous means over a time period of several minutes or hours. Short peaks can be higher without damage, but we don't know much about the Twizy battery capabilities here.

An AC motor will produce constant torque from constant current up to the rated speed (RPM), at which is also will output the rated maximum power. From this RPM point upwards, the current needs to be scaled down following the speed/torque curve of the motor to not exceed the rated maximum power.

Rated stator / phase current for the motor is 175 A (rms), this can be found in the motor nameplate SDO 0x4640.02 and the motor configuration SDO 0x4641.07. The maximum stator current is configured to 450 A (rms) (0x4641.02), this is assumed to be the sum for all three phases and fits the SEVCON controller type (G4845).

The Twizy motor is rated for a maximum of 13 kW mechanical output, the maximum electrical power to feed into it or get from it can be **18 kW**. This value can be found in the SEVCON configuration at SDO 0x3813.23 "Motor peak power" and at SDO 0x2870.06 "Maximum battery discharge power".

The battery pack is sized at 7 x 966 Wh = 6762 Wh capacity (6100 Wh usable), so 18 kW is ~2.66C and translates to ~313 A at the maximum pack voltage of 57,5 V. Not surprising, SDO 0x4623.02 "Maximum battery discharge current" is configured to 314 A on a full battery.

The battery charging limit default is set to 2.5 kW (SDO 0x2870.07) and 43 A (SDO 0x4623.01) on a nearly full battery, and raises up to a maximum of **8 kW** between 80% and 30% SOC (sometimes 8.5 kW). This limit is basically under BMS control, but it seems the SEVCON will allow a bit more for a short period of time.

So, these are the current limits of the motor, controller and battery, and you normally should not try to raise these configuration items or exceed these currents, otherwise you'll risk your components. As there should normally be no need to change these, there is no macro command to do so. Just keep in mind there are safety limits at work even if you raise every macro parameter to max.

Speed

The speed limit serves two purposes: to protect the gear box and to protect the driver (you!).

The gear box *seems* to be bad, it is very loud and sounds out of the box broken. Word of mouth is it's been used on other vehicles (i.e. the german Hotzenblitz) and not as poor quality as it sounds.

For the driver protection, it should be obvious that the Twizy gets the harder to control the faster it goes. After all this is a very light, very small vehicle with small wheels and poor aerodynamics. Strong side wind is an issue at 80 kph already, especially if you're using addon windows. Bad roads can make the Twizy jump. Now imagine a jumping Twizy at 100 kph... not funny.

Of course the motor power limits the maximum speed as well, even if you also raise the torque level, an aerodynamically unmodified Twizy will run into it's power limitations finally at ~110 kph. The SEVCON's severe overspeed fault level is set to 11000 rpm = ~121 kph, but that does not seem to be reachable without a long steep downhill passage. Nevertheless, the macro commands will use an upper limit of 10.900 rpm for all overspeed limits and an upper limit of 10.000 rpm for the forward speed.

See below for necessary changes to the power map.

Torque

Higher torque means stronger acceleration and is especially useful when starting on a steep incline.

Using a higher torque results in higher battery and controller currents and higher load on the gear box. Currents should (?) be limited to the safety frame defined by the basic controller configuration and capabilities, the gear box is on it's own.

Of course, higher currents will also mean higher temperatures for all components, this may be an issue for hot climate conditions. All component temperatures are monitored, critical temperatures will result in power cutback, but you should keep an eye on the OVMS temperature readings when experimenting.

The Twizy motor has a rated torque of 57 Nm at 2100 rpm. Interestingly, only 55 Nm are configured usable by default in the SEVCON (0x6076 and points 1+2 in the torque/speed map at 0x4611), while the flux map (0x4610) is defined up to 70.125 Nm. The flux map also defines maximum magnetising currents to be 149 A at 55 Nm and just +7 A (156 A) at 70 Nm.

So it seems the motor should safely be capable of producing 70 Nm (not continuous of course!) but has been limited to 55 Nm, possibly because the controller is rated for 450 A, so 3 x 149 A keeps just below that point while 3 x 156 A will exceed this by 18 A or 4%. The controller will do short boosts up to 540 A, and the motor is specified for 175 A max phase current. So none of these limits should be exceeded by 70 Nm, but temperatures will rise fast at this current level.

70 Nm at 2100 rpm equals a mechanical output of ~15.4 kW. This would be ~18% above the motor's rated output, so the power map will be adjusted (see below). Electrical input depends on the

actual load situation, but we can assume the controller will limit the power according to the controller and battery current limits, so the full 70 Nm will probably not be available in every load (and temperature) situation.
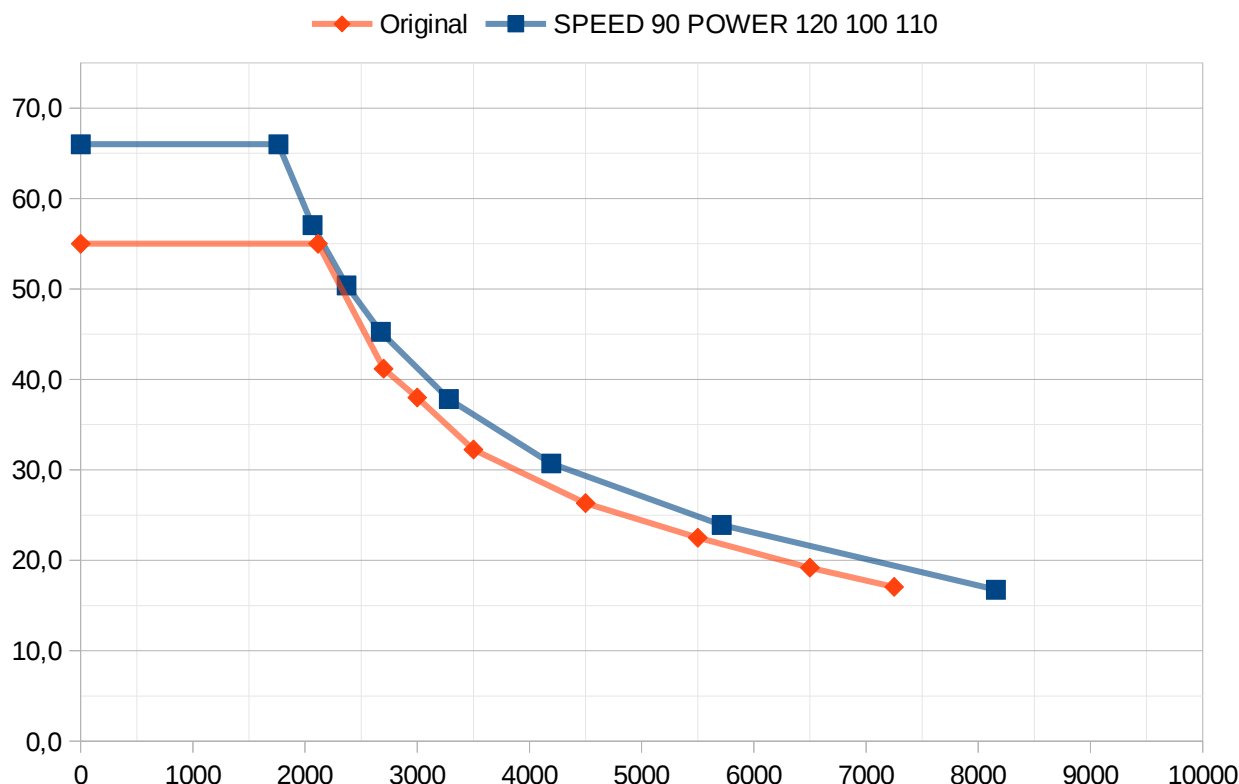
Power map / torque curve

The power map relates the available torque level to the motor speed. An AC motor will produce constant torque up to the point of maximum power, from which on the torque needs to be scaled down to maintain the power limit.

Physically there are basically three parameters for the map: maximum speed, maximum torque and maximum mechanical power. There may also be an influence of nonlinearity of the motor efficiency, i.e. if the motor needs nonlinear power input to produce mechanical output depending on the speed.

To be able to produce a new power map closely resembling the original one and to enable maximum flexibility for configuration, the POWER command allows to specify separate maximum power levels for the beginning (low speed) and end (high speed) of the constant power range.

I have included a spreadsheet document called "Twizy Powermap Calculator" which produces a power map from the four main parameters, which may be entered as needed by the CFG command. The sheet includes the original Twizy power map and allows to compare both.

You may use the sheet to test the effects of speed, torque and power tweaks before applying them. Here's an example comparing the original map to one for 90 kph, 120% torque and power ranging from 100% to 110%:

The original power map of the Twizy cuts back torque in the rpm range up to ~61 kph. It contains a nonlinearity, but one that does not look natural: if you take a look at 2700 and 3000 rpm, these do not fit into the overall curve, even if considering efficiency nonlinearity. Also, the mechanical power at the original turning point of 2115 rpm is 12.2 kW instead of the 13 kW. 13 kW will only be available from 5500 rpm upwards.

In order to adapt to the user speed & power configuration, the original power map will thus not just be scaled but replaced based on the physical calculations you can see in the spreadsheet. This will result in a slightly smoother curve and slightly more power in the speed range from ~23 to ~61 kph, even if you change just one parameter. On setting all three parameters back to default, the original map will be restored.

The three power command parameters are scaled to 100% = default value.

The command will fill missing parameters from those given for convenience, so a "CFG POWER 110" will raise **all three** levels to 110%. The torque maximum is effectively limited to 128% = 70.125 Nm (but can be given as 130 for convenience).

By raising just speed and/or torque you can get a map that still stays within the 13000 W mechanical power limit, so the motor will stay within the standard limits.

If you raise the max power output levels, this will allow to operate the motor outside its specification, be sure to understand the implications before doing so.

The power command max values translate to a mechanical power output of 16900 W, which would also need an electrical input power of ~19200 W – this is above the configured maximum motor input of 18000 W (SDO 0x3813.23) and mostly above the dynamic BMS limit, so will trigger the SEVCON power limiter without further tweaks.

The power command will not change or disable the standard power/current limiters, if you really want to go this far, you need to do so manually.

Keep in mind, even with disabled limiters the SEVCON will protect itself by limiting continous high currents to short periods of time (see "peak" & "boost" specification).

Drive and recuperation levels & torque/speed maps

Default drive level is 100% (of the available power), the main tweak here is lowering the level to increase economy / range or adapt to bad road conditions.

Default recuperation levels are rather low at 18.2% for both neutral and foot braking modes.

For each of the three situations "driving", "neutral braking" and "foot braking", the SEVON uses both a speed lookup map and the overall level to determine how much of the actually technically available maximum torque will be used at a given speed.

"Available maximum torque" differs for drive and neutral/brake mode: in drive mode, the available torque is defined by the power map. In neutral and brake mode, the power map does not apply, maximum torque equals the configured peak torque (first power parameter) at any speed.

The controller will first consult the drive mode specific torque map. Each of the three maps contains

four speed reference points at (by default) ~33, ~39, ~50 and ~66 kph. The torque level will be interpolated for the current speed and multiplied with the overall level.

The default drive mode map does no limiting by speed, full power is available at every speed.

The default neutral and footbraking mode maps will limit recuperation power for higher speeds:
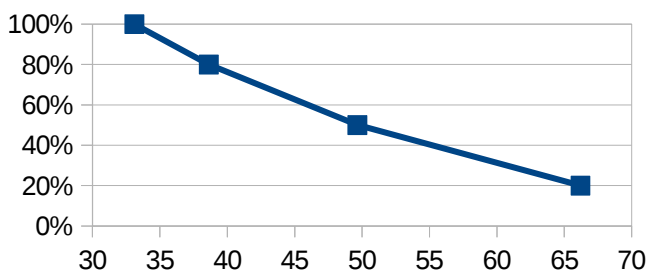
The default power levels are
         100 / 80 / 50 / 20 for
         33 / 39 / 50 / 66 kph respectively.

So the maximum recuperation of 18.2% (of the peak torque) will apply up to 33 kph.

If you raise the overall recuperation level, the maps will provide the means to shape recuperation strength by speed and seemlessly integrate with braking. For example:

```
CFG RECUP 18 40
CFG TSMAP B 80 100 60 40
```

This will raise braking recuperation to 40% but scale the strength down for low speeds. The effect is an electrical brake booster that integrates smoothly at every speed. If you want to fine tune the curve to your torque profile, you can extend any or all of the four percentages by speed coordinates in kph like this:

```
CFG TSMAP B 80@20 100 60 40@80
```

This would move the first 80% point from the default 33 to 20 kph and the last 40% point from 66 to 80 kph.

If you want recuperation free coasting on the highway, just set the 4th level in map "N" to 0.

Important: the maps "N" and "B" shape the torque used for recuperation, and therefore define the "reverse" power maps (for using the motor as a generator). As the power generated by recuperation will follow the same laws as the power needed to generate drive torque, these maps also shape the power delivered to the battery. The battery (BMS) will only allow limited power levels, which also get adjusted dynamically as a function of the SOC.

If recuperation power exceeds the allowed level, the SEVCON limiter will be triggered – you know this just happened when you feel a sudden cutback of braking torque. The limiter protects the battery, so is a good thing. But if you want to get the most out of recuperation, you need to shape the recuperation maps to not trigger the limiter in your normal driving range.

The included "**Twizy Powermap Calculator**" also contains a **recuperation map development tool** that will graph both torque and power levels over speed of a given configuration, and will warn about exceeding the standard recuperation power of 8000 W. It includes the original configuration as well as two optimized examples.

The drive mode map can also lead to interesting changes in the driving character of the Twizy, especially if combined with higher peak torque. You can for example reshape the torque curve to cut

down power in low speed and simulate a "boost" for higher speeds. Of course, this can also be used to optimize range.

I have included a spreadsheet called "Torque Speed Profiles" containing some first examples. It can be used to design new profiles as well, please share your creations with the community.

Please note: be aware that a change of your power configuration will also affect the TSMAP effects, as the map effects are relative to the available power & torque.

Be aware of these caveats on raising the recuperation levels:

1. High recuperation means high charging currents, which can lead to battery degradation in very cold environments. The battery charging power limits apply, but if you own your Twizy battery, consider setting the recuperation even below 18% in winter. This can also lead to a more stable drive (recuperation can be bad on slippery roads).

2. High recuperation in neutral braking will irritate the driver behind you. The Twizy does not let the SEVCON control the brake lights, so you'll be slowing down rapidly without any signal to the car following you. Consider only raising the braking recuperation level.

3. Footbraking detection may need firm pedal pressure, depending on how sensible your Twizy brake pedal reacts. Check the sensibility by checking how strong you need to push the pedal in order to unfasten the parking brake (same switch).

Auto power level adjustment (*V3.6*)

The OVMS can be configured to dynamically adjust both the DRIVE and RECUP levels according to the actual BMS power limits.

Of course the SEVCON will limit power levels by itself as necessary. But if it does, the effect is very harsh and abrupt, and it cuts back the power further than necessary. So by adjusting the DRIVE and RECUP levels you can avoid triggering the SEVCON limiter by keeping your power levels below the allowed limit at all SOC levels.

Another positive effect is that the battery stress is automatically reduced with the SOC, which can lead to a better discharge efficiency due to less losses at the battery inner cell resistances.

There's one caveat to mention: only the general power level (DRIVE/RECUP) can be adjusted while driving, that applies to the power and to the torque. The torque level could be kept high for launches, as the power level is low in that area. Unfortunately, there is only the general power level.

To accomodate to this, an automatically lowered DRIVE level also enables the kickdown feature, just like a manually lowered DRIVE level does. The kickdown sets the level back to 100%, so during a kickdown, the SEVCON limiter may come back.

Auto power adjustment is now based on a reference point defining the maximum electrical battery power to be expected, and an optional minimum level. You can use the power map calculator to estimate the electrical power levels needed for your mechanical power output. This is an estimation because power efficiency is not a constant. You should experiment with slightly higher and lower reference values to find your personal optimum.

DRIVE and RECUP calculations differ: *100%* DRIVE level is assumed to translate to the reference drive power, while the RECUP levels *configured* translate to the reference recuperation power. That's because the RECUP levels normally are tailored to the TSMAP N/B configuration to deliver maximum recuperation, while the DRIVE level is normally only set below 100% to reduce the energy use.

You can monitor the auto power adjustment by reading the "RT-GPS-Log" records.

How to determine the reference power level

1. Enter your configuration parameters into the power map calculator.

2. Start with the maximum "auto power ref" values calculated by the sheet.

3. If the limiter still kicks in, raise the reference values.

4. If you think the power adjustment is too early, lower the values.

The reference levels are configured in 100 W resolution. For example, to set normal drive power level to 100% and activate auto power with reference power level at **17.7 kW**, issue:

> CFG DRIVE 100 **177**

To set recuperation levels to neutral 36% and braking 65% with **8.8 kW** reference, issue:

> CFG RECUP 36 65 **88**

See command reference for complete parameter explanation.

Background

Battery power levels are *electrical*, while motor power configurations are *mechanical*. To translate between these, we need the combined power efficiency of the motor + controller + BMS. The power efficiency is nonlinear dependant on speed, load, voltage, temperature and more parameters, so we need to select a single value that fits most situations.

The observed minimum power efficiency for **driving** is ~88.4%. So if your power profile has a mechanical maximum of e.g. 16 kW, the maximum electrical power needed will be...

> 16 kW / 88.4% = **18.1 kW**

Taking that +/- 0.1 kW as your reference drive power usually works quite well. Now if the BMS limits power to e.g. 17.5 kW, that translates to an automatic DRIVE level of...

> 17.5 kW / 18.1 kW = 96.7 %

Recuperation is more complex. Efficiency is ~88% for low power as well (recup level < 40%), but can be close to 100% for higher recuperation levels. An asynchronous AC generator has a low speed turning point below which it needs more power input to build the magnetical field than can be generated from recuperation. The SEVCON does brake feathering below ~5 kph to accomodate this

and to provide a smooth handling in low speeds.

The following chart is based on a test drive using a CAN logger to capture all drive data. It shows the theoretical mechanical power available (yellow, zero at 1.1 kph) and the actual electrical recuperation power feeded into the battery (blue points, y offset -300W).

Recuperation profile was the default torque map with level raised to 65% to produce the limiter effect. BMS limit was 7500 W, but the chart shows the limitation will actually start at ~6000 W and allow a maximum battery input of ~6500 W:
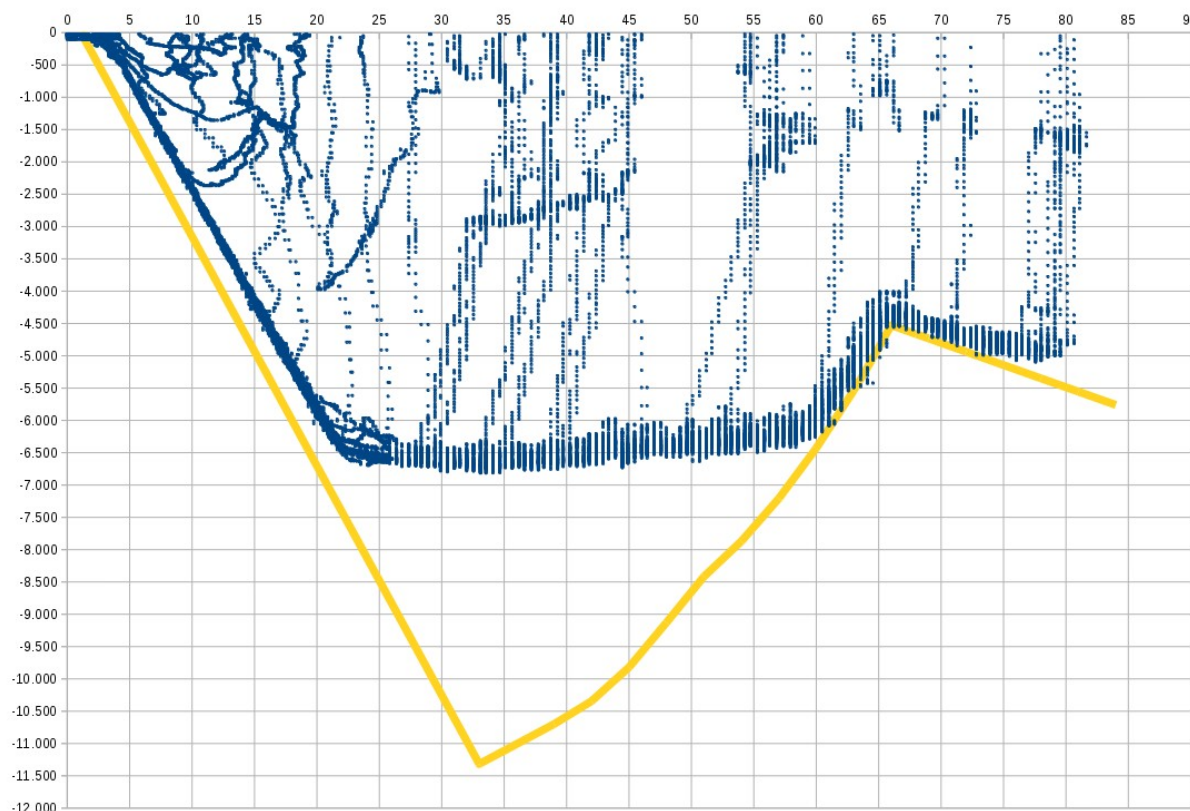
Recuperation power and limiter effect
(default torque map, 65% recup level, 7500 W limit)

Possible explanations are: a) the SEVCON calculates power from a higher voltage (i.e. charging voltage ~58 V), and b) the cutback range of (by default) 15 A is responsible for the early limiting.

Rule of thumb is to assume a power efficiency of 100% and add 800 W to get to the limiter level.

So if your recuperation profile has a mechanical maximum at 5.0 kW, the limiter will regard that as an electrical power of **5.8 kW**. Begin with that as the reference point and try raising if the limiter still kicks in.

Let me know if you find a better rule to determine the reference point.

Kickdown functionality (added V3.5.0)

Following an idea from the german Twizy community: if the DRIVE level is set lower than 100% (**V3.6**: either manually or automatically), the OVMS checks for accelerator pedal kickdown events. On kickdown, the DRIVE level will temporarily be set to 100%. After releasing the pedal for 5 seconds, the DRIVE level of the profile will be restored.

Any fast pushdown of the pedal is considered to be a kickdown. This means you don't need to kick the pedal all the way down, you can trigger kickdown at any position of the pedal by giving it a fast kick.

The pedal change threshold can be configured to adapt to personal likes or pedal characteristics, but due to EEPROM space limitations this is currently a RAM only configuration, meaning you need to reconfigure after any power down of the module.

Pedal value range is 0..255 in theory, normally will end ~247. The pedal has a slight nonlinearity, resulting in higher changes on the beginning and lower towards the end position. To compensate, the pedal threshold will be lowered linearly by a factor of 1/8 beginning from a configurable pedal position ("compzero").

Default threshold is 35, compensation zero point 120, resulting in:
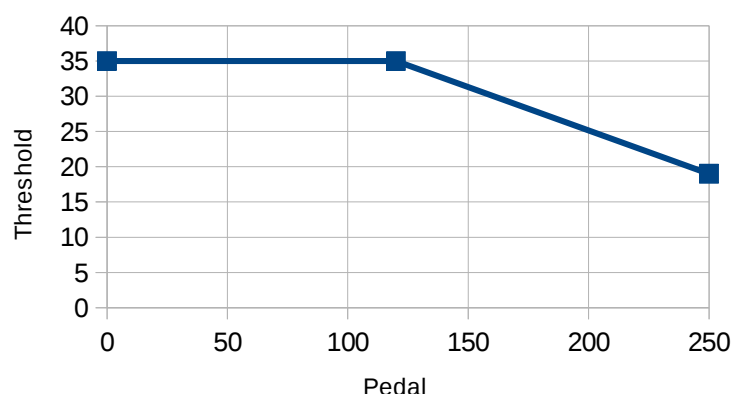
Setting the threshold too low results in kickdown triggers from road bumps.

Setting the theshold to 0 deactivates the kickdown function.

Taking back the pedal for ½ threshold from the kickdown maximum pedal level is considered to be a kickdown release. On kickdown release a 5 second timer will start and count down unless the pedal is pushed down into the kickdown area again. This allows to take back acceleration for a moment if necessary without immediately losing the drive level boost. Drive level will only be restored to normal if the release state continues for full 5 seconds.

The kickdown function uses the SimpleConsole to display its state: the first LED (corresponding to the builtin profile) will be lit (additionally to the current profile LED) as soon as the drive level has successfully been set to 100%. During the last 2 release timer seconds, the LED will flash to signal "going back to normal soon", after successful restoration of the normal drive level the LED will be switched off.

The kickdown threshold and compensation zero are global configurations stored in OVMS Feature slots #1 and #2. You can change the kickdown configuration directly by altering the feature values (using the App or SMS command "FEATURE"). Be aware that feature slots #1 and #2 are temporary, meaning they will be reset to defaults on power loss.

For convenience the kickdown configuration can also be changed using the CFG DRIVE command, for example:

> CFG DRIVE 70 181 10 30 100

...will set drive level to 70%, activate auto power with ref=18.1 kW and minimum=10%, set kickdown threshold to 30 and compensation zero to 100.

Currently the compensation factor (1/8) is fixed, as well as the kickdown release delay (5 seconds).

Be aware that the OVMS will not always be able to react immediately on a kickdown event. For example an ongoing network data transmission needs to finish or time out before the next kickdown check can be done. The kickdown event is not "lost", just the drive level adjustment will be delayed. This is normally in the order of some tenths of a second, but recognizable. To get best kickdown reaction times, cut down data transmissions – for example don't use GPS streaming (or at least restrict to streaming mode 1).

Ramps and smoothing

These control how the SEVCON shapes the actual torque curve depending on the driver input (throttle and brake pedals).

Higher ramp levels mean faster changes. Lower smoothing means less leveling out.

The first ramp parameter ("start") is mapped to the gear box oscillation damping ramp. It depends on your gear box quality how high you can set this.

Important note: the "start" ramp actually applies to every load change situation, so for starts as well as for transitions from driving to braking and vice versa. There's a side effect from this: if you set this ramp too fast combined with high recuperation, an abrupt change to braking/neutral can trigger the recuperation power limitation function, as a sudden gear hit can produce a current peak. This can also lead to a feedback loop between torque ramping and power limiting, if you encounter that you really need to lower the ramp. Also take into account that raising maximum torque and/or the overall ramp speeds (RAMPL, see below) can make lowering the start ramp necessary as well.

Since version 3.4.0, the start ramp parameter is interpreted in **per mille** and allows a range of 1..250. This is to allow fine control over this ramp, most Twizy gear boxes will not allow to make the ramp much faster than the default (40 per mille on T80, 30 on T45).

The overall rpm change limits for all ramps can be set using the RAMPL command. Default ramp limits are 6000 rpm/s for both acceleration and deceleration, which translates to 30% of the valid range. 100% will primarily affect intermediate accelerations and decelerations.

Note: even if you set the start and accel level to 100% there still is a short start delay. The source of this delay needs to be identified yet, current candidates are the maximum rpm change per second (0x2920.15) or some throttle pedal signal smoothing (0x3814). There's also a battery current ramp

that could apply.

Twizy 45

The OVMS firmware reads the controller type on every login and automatically uses the appropriate parameter defaults. This means you can plug a T80 OVMS module into a T45 any time, and vice versa. Caveat: if you upgrade your T45 to a G4845, the OVMS will think this is a Twizy 80 – this will not work without modifications.

All values mentioned above refer to the Twizy 80. The Twizy 45 limits are lower, see spreadsheet "Twizy-Limits" and "Twizy-SDO-List" for details. In short:

- maximum battery discharge power is 10.5 kW (T80: 18 kW)

- maximum configurable speed is 78 kph

- maximum torque will be at 111% = 36 Nm (default: 32.5 Nm)

- power levels of 130% can be configured, but watch temperatures! Default max power is ~7650 W = ~191% of continuous specs. 130% will raise this to ~9950 W = ~249% – the battery limit will probably cut this down.

- default recuperation levels are 21% (instead of 18%, maybe due to the lower torque)

- default ramps are 3% for start and 21% for acceleration (T80: 4% & 25%)

- TSMAP points are at the same speeds (except drive map, which may be a config error)

Note: the "Twizy-Powermap-Calculator" also contains a Twizy 45 sheet.

Macro commands

Macro commands are **shortcuts** that simplify tuning by automatically modifying the **set of registers** working together for a certain tuning function. All macro commands **check** their parameters for valid ranges, to avoid faulty configurations. Macro commands can easily **reset** their parameters to the default configuration. They also **automatically** switch into PRE mode and back to OP mode as necessary, so you just need to issue the macro command itself.

All macro commands work identically for the 80 kph and the 45 kph versions of the Twizy, automatically using the respective limits (see previous section on limit differences). The limits will be applied on setting the registers, so you can even load a T80 macro profile into a T45 (but don't expect similar driving results).

Generally, you should have your "Plan B" ready before tweaking. **If you break your OVMS in the middle of a config session, you will need some other way to login to the SEVCON and switch it back to operational!**

Reminder: you need to **enable** CAN write access (feature #15) first, and you may need to **clear** bit 2 (value=4) in the car bits (feature #14) if you want to get the CFG results.

Example session:

1. Set speed limit to 90 kph, keep default warning @89 kph:
    `CFG SPEED 90`

2. Set max torque level to 110% = 60.5 Nm, keep max power levels at default:
    `CFG POWER 110 100`

3. Set max recuperation to 25% on neutral and 40% on braking:
    `CFG RECUP 25 40`

4. Set torque @ 66 kph to 25% in neutral and braking mode:
    `CFG TSMAP NB 100 80 50 25`

5. Set start ramp to 45 per mille and acceleration ramp to max speed, keep defaults for rest:
    `CFG RAMPS 45 100`

6. Set ramp smoothing to minimum:
    `CFG SMOOTH 0`

Example #2: Twizy 45 emulator

1. Set speed limit to 45 kph:
    `CFG SPEED 45`

2. Set all power levels to 59% = 32.5 Nm / 7.65 kW:
    `CFG POWER 59`

Command	Function
`CFG SPEED [max_kph] [warn_kph]`	Set speed limit and warn level. Allowed value ranges: max_kph: 6..111, default 80 (T45: 45) warn_kph: 6..111, default 89 (T45: 56) Both parameters can be omitted, the command will then use their defaults. Warn level may be lower than max speed, usable for example to avoid speeding tickets. V3.3: The reverse speed will also be limited, but only up to the default reverse speed (~10 kph). *NOTE: this command will need to go pre-operational, so the vehicle needs to be stopped. Also a power cycle should be done after using this command.*
`CFG SPEED`	Reset speed configuration to default. *NOTE: this command will need to go pre-operational, so the vehicle needs to be stopped. Also a power cycle should be done after using this command.*
`CFG POWER [torque%] [power_low%] [power_high%]`	Set maximum motor torque for starting and maximum motor power output levels for low and high speed. torque%: 10..130 (128), default 100 power_low%: 10..139, default 100 power_high%: 10..130, default 100 100% torque = 55 Nm = default 128% torque = 70.125 Nm = max 104% torque = ~57 Nm = motor peak rating 100% power_low = 12182 W = default turning point 100% power_high = 13000 W = motor peak rating Submitting only one parameter will set all three levels to the same value. If submitting two parameters, the first will be used for the torque level, and the second one will be used for both power levels. **ATT: Raising can overload the motor and controller! Monitor temperature levels!** *NOTE: this command will need to go pre-operational, so the vehicle needs to be stopped. Also a power cycle should be done after using this command.*

Command	Function
`CFG POWER`	Reset motor torque and power configuration to default. *NOTE: this command will need to go pre-operational, so the vehicle needs to be stopped. Also a power cycle should be done after using this command.*

`CFG DRIVE [power%]`
`[autopower_ref]`
`[autopower_min%]`
`[kickdown_threshold]`
`[kickdown_compzero]`
	Set default power usage level and optional auto power adjustment and kickdown parameters for driving.
	
	power%: 10..100, default 100
	autopower_ref: 0..250, default 0/-1=off
	autopower_min%: 0..100, default 0/-1=none
	kickdown_threshold: 0..250, default 35 (= Feature #1)
	kickdown_compzero: 0..250, default 120 (= Feature #2)
	
	If power% is less than 100, the kickdown function is automatically enabled unless the kickdown threshold is set to 0.
	
	Auto power reference level sets the maximum electrical power to be expected from your torque configuration (in 100 W resolution, i.e. 177 = 17.7 kW). Auto power min% sets the allowed minimum level for auto adjustments. See section "Auto power level adjustment" for more details.
	
	Kickdown threshold sets the accelerator pedal change that triggers kickdown, the lower the more sensitive. "Compzero" sets the pedal point where pedal nonlinearity compensation begins, 120 is at about 40% of the mechanical pedal range.
	
	Both values may also be set directly by altering the feature slots.

`CFG DRIVE`	Reset maximum power usage level to default (100%) and deactivate auto power adjustment.

Command	Function
`CFG RECUP [neutral%] [brake%] [autopower_ref] [autopower_min%]`	Set maximum recuperation power levels for neutral braking and footbraking + optionally automatic level adjustment by BMS limit. neutral%: 0..100, default 18 (T45: 21) brake%: 0..100, default = neutral% autopower_ref: 0..250, default 0/-1=off autopower_min%: 0..100, default 0/-1=none Submitting only one parameter will set neutral and brake to the same value and disable auto adjustment. Auto power reference level sets the maximum electrical power to be expected from your torque configuration (in 100 W resolution, i.e. 87 = 8.7 kW). Auto power min% sets the allowed minimum level for auto adjustments. See section "Auto power level adjustment" for more details.
`CFG RECUP`	Reset maximum recuperation power levels to default (18%) and disable automatic adjustment.

Command	Function
`CFG TSMAP [DNB] [t1% [@s1]] [t2%[@s2]] [t3% [@s3]] [t4%[@s4]]`	Change one or multiple torque scaling map(s) (power level per speed lookup tables). DNB: select the maps to change (may be any combination): D = Drive, N = Neutral braking, B = Footbraking t1%: level at 33 kph, 0..100, default D=100 N/B=100 t2%: level at 39 kph, 0..100, default D=100 N/B=80 t3%: level at 50 kph, 0..100, default D=100 N/B=50 t4%: level at 66 kph, 0..100, default D=100 N/B=20 Each of the four percentage level points can be extended by an "@" char (no spaces!) followed by the speed coordinate of that point in kph: s1: speed for t1, 0..120, default 33 kph s2: speed for t2, 0..120, default 39 kph s3: speed for t3, 0..120, default 50 kph s4: speed for t4, 0..120, default 66 kph Points will not be sorted by speed, so take care to give them in order. If multiple maps shall be changed, the command will stop if it encounters an error, the remaining maps will not be changed. *NOTE: this command will need to go pre-operational, so the vehicle needs to be stopped.*
`CFG TSMAP [DNB]`	Reset specified torque scaling maps to default. *NOTE: this command will need to go pre-operational, so the vehicle needs to be stopped.*
`CFG TSMAP`	Reset all three torque scaling maps to default. *NOTE: this command will need to go pre-operational, so the vehicle needs to be stopped.*

Command	Function
`CFG RAMPS [start‰] [accel%] [decel%] [neutral%] [brake%]`	Set rates of torque demand changes (higher means faster). start‰: 1..250, default 40 (T45: 30) accel%: 1..100, default 25 (T45: 21) decel%: 0..100, default 20 neutral%: 0..100, default 40 brake%: 0..100, default 40 Omitted parameters will use their default values. ***New for version 3.4.0:** the interpretation of the "start" parameter changed from percent to per mille, you need to adjust your configurations!*
`CFG RAMPS`	Reset ramp rate configurations to default.
`CFG RAMPL [accel%] [decel%]`	Set rpm change limits for ramps. accel%: 1..100, default 30 = 6000 rpm/s decel%: 0..100, default 30 = 6000 rpm/s Omitted parameters will use their default values.
`CFG RAMPL`	Reset ramp change limits to default.
`CFG SMOOTH [level%]`	Set smoothing curve (S-curve) level. level%: 0..100, default 70
`CFG SMOOTH`	Reset smoothing configuration to default.
`CFG BRAKELIGHT [on_lev] [off_lev]`	**V3.6**: command has been excluded from default firmware build due to ROM space issues. Set brake light controls for neutral braking. levels: 0..100 default 100 = control disabled These are deceleration levels with 100 = ~14 m/s². Submitting only one parameter will set both to the same value. *NOTE: sadly, this will not work without a hardware modification, as the Twizy brake light does not seem to be controllable by the SEVCON. If you know how to do this modification, please share your knowledge. Additional info: brake light signal should be available at pins 11+12 of the SEVCON connector.* *NOTE: this command will need to go pre-operational, so the vehicle needs to be stopped.*

Command	Function
CFG BRAKELIGHT	**V3.6**: command has been excluded from default firmware build due to ROM space issues. Reset brake light controls to default (disabled).
CFG RESET	Reset all currently active macro settings to their default values. (See also "CFG RESET without network" below) Please note: this only changes the current working set, not the stored profile, unless you also do a "CFG SAVE" afterwards. It's meant to provide a clean start for a new profile. If you want to reset the Twizy to the factory default profile without changing anything else, do "CFG LOAD 0". If you issued a CFG RESET by mistake, just do a "CFG LOAD" to reload your last saved profile.
CFG CLEAR	Clear the SEVCON fault logs. This is the same as issueing MSG command `ResetLogs` with "99" = all logs. See section "SEVCON logs" for details.

Errors

Error codes 0x003n indicate command parameter #n is outside it's valid range, please check the command description before retrying. Pay attention to wrong characters or misplaced spaces.

For other errors, especially cryptic hexadecimal things: **Don't panic**.

1. **Issue the last command again**.

2. If that doesn't help do a power cycle without entering "GO", wait for 10-15 seconds, then try again.

Most errors occur from the SEVCON being too busy at the moment to react to some SDO read/write request (timeout). The command then stops execution. Timeouts occur more often if the Twizy is in "GO" state.

STOP lamp (pre-operational state)

Some macro commands (i.e. SPEED and POWER) need to enter the so-called pre-operational state of the controller. The macro commands will try to switch states automatically but sometimes this can fail.

Pre-operational state is signaled by three beeps, the STOP lamp and the LCD battery warning symbol, the Twizy will not react to key presses. You can go back to operational state

- either by sending the `CFG OP` command

- or by just switching the Twizy off and on again and waiting for ~10 seconds.

On switch-on, the OVMS will check if the Twizy is still in pre-operational state and will automatically switch back to operational if needed.

CFG RESET without network connectivity

To manually issue a `CFG RESET` command, do these steps:

1. Switch off the Twizy

2. Switch on to first state (before "GO")

3. Press "D" or "R" three times ...and don't forget to say "there's no place like home" ;-)

4. Wait 5 seconds, then switch off the Twizy and start as usual.

If the OVMS still has connectivity, it will send the RESET result by SMS.

If the RESET command fails (i.e. timeout or CAN access problems), you can try again without switching off the Twizy.

Profile management

While there are three driving profiles in the SEVCON, the OVMS does not use them. This is due to two limitations: a) the SEVCON profiles only cover a subset of the parameters controllable by the OVMS (i.e. no torque/power modifications), and b) you need to modify your Twizy to connect a profile switch to the SEVCON.

The OVMS can store up to three custom profile configurations (slots #1 - #3) as persistent parameters (EEPROM). Slot #0 is virtual and represents the Twizy factory default configuration.

The profile storage covers all macro command parameters. The last profile selection is also stored persistently, so the OVMS will retain your complete profile setup over power loss.

Macro commands modify the working set configuration in RAM. To save the working set, issue a "CFG SAVE", to switch to another profile, issue a "CFG LOAD" or push the corresponding button on your console (see below).

Profile switching is divided into two parts:

1. Standard "user mode" changes: DRIVE, RECUP, RAMPS, RAMPL, SMOOTH

2. Pre-operational "config mode" changes: SPEED, POWER, TSMAP, BRAKELIGHT

User mode changes can be applied any time, also during driving (as with the SEVCON builtin profiles).

The changes requiring pre-op mode will only be done if the Twizy is stopped (not moving) and not in "GO" mode, so you need to do a power cycle before these can take effect.

Command	Function
`CFG SAVE [slot#]`	Save the current working set into an EEPROM slot. slot#: 1..3 profile slot number to save to (optional) If you omit the slot number, the working set will be saved to the currently selected slot (unless it's #0). Saving will clear the modification flag (see console) and switch the selected profile accordingly.
`CFG LOAD [slot#]`	Load a profile slot into the current working set. slot#: 0..3, 0=factory default config, 1..3=custom profile If you omit the slot number, the command will reload the current profile from the last save (or defaults), thus discarding any changes made to the working set. The profile loaded from EEPROM will be checked for consistency using a checksum. If the check fails all parameters will be reset to defaults. Loading will clear the modification flag (see console) and switch the selected profile accordingly. Important: (see above) a full profile switch can only be done if the Twizy is not moving and not in "GO" mode. I.e. after switch-on, before "ignition". The command response will inform you if the switch was partial. A partial switch can be done any time, it only covers DRIVE, RECUP, RAMPS and SMOOTH configurations.
`CFG INFO`	Output currently active profile slot# and working set parameters for these commands: SPEED, POWER, DRIVE, RECUP, RAMPS, SMOOTH
`CFG RESET`	Additional note: this macro command will not switch to profile slot #0, only the current working set will be reset. An offline reset using the D/R buttons has the same effect. Issue a RESET to begin a new profile configuration from scratch.

Using an OVMS SimpleConsole to access profiles

Of course it's a bit awkward having to send an SMS to switch profile on the road. In most countries using a mobile phone while driving is illegal (for good reason). But in certain situations you want to be able to switch some parameters immediately: i.e. if you encounter black ice, you want to be able to deactivate recuperation as fast as possible.

That's where the OVMS SimpleConsole comes in. It can look like this:

OVMS SimpleConsole for immediate profile access

Unfortunately this is not yet available at Fasttech. See "/docs/OVMS-SimpleConsole-HowTo.pdf" on how to build one yourself – it's easy and needs very little electronics experience.

The console has four push buttons with (in this example) four integrated LEDs.

- The LED state shows the currently active profile while the Twizy is on (and not charging).
- The buttons select profile slots #0-3 (same as "CFG LOAD" but without SMS notification).
- If an error occurs on profile switching, all four LEDs flash for about 1/10 second.
- If a profile switch was partial, the LED corresponding to the former profile (still partially active) flashes once per second.
- If your working set has unsaved changes, the profile LED flashes off once per second.

If a profile is partially active or modified, the button may be pushed again to retry switching or to reload the profile from EEPROM.

Please note: the information about partial and unsaved profiles will retain over switching off the Twizy, but not be kept over power loss (i.e. if you disconnect the module from the OBD2 port).

Please also note: the key presses and LEDs are not interrupt driven, but polled about every 1/10 second. The polling run may be irregular if the module is busy, so you may need to push for longer than 1/10 second, and LED flash times may become irregular. That's not an error, just a little bit annoying – it's on my list ;-)

Using the OVMS App to access profiles

The OVMS App currently lacks driving profile functionality. On the other hand it has a "Homelink" control for sending buttons labeled 1-3, while the Twizy has no Homelink device. So we hijack the Homelink function for switching profiles.

The Homelink buttons 1-3 are mapped onto the CFG profiles #1-3. So there is no access to profile #0 (default config) from the App.

Be aware the Homelink MSG command (24) expects button codes 0-2 instead of 1-3. Using the command client you may select profile #0 by either sending "-1" as the parameter or by omitting the parameter.

The module will send a text notification containing the switch result info.

Example profiles

Here are some profiles as examples and/or starting points for your own experiments:

	#1 Power	#2 Eco	#3 Ice
POWER	130 110 120	130 115 107	-
SPEED	110 111	110 111	-
TSMAP D	-	-	90@0 90@23 100@40 100@90
TSMAP N	100@45 82@57 68@68 52@90	100@45 82@57 68@68 52@90	-
TSMAP B	100@20 52@42 34@68 26@90	100@20 52@42 34@68 26@90	-
DRIVE	-	70	-
RECUP	25 50	25 50	0 0
RAMPL	100 100	-	20 20
RAMPS	35 100	30	30
SMOOTH	0	100	-

These profiles are pre/op combinable, so if you load #1 before "GO", do the necessary power cycle and enter "GO" mode, then load #2, you get an "Eco light" variant. Combining #3+#2 will result in an "Eco max" variant (probably only usable on flat terrain). If you're in "Eco" and need more power for some hill climb, just push #1 for that hill and push #2 again afterwards (a good example why you'll like having a SimpleConsole).

Braking in these profiles has been optimized to deliver maximum recuperation power and to minimize need for mechanical braking. So neutral braking will be a bit higher than ICE motor brake, and active braking torque will be stronger than many car brakes below 20 kph. This needs time to get used to, but rewards with high recuperation gains and (hopefully) longer brake life.

The "Power" profile will get about the same range as the default profile. The "Eco" profile will achieve about 6-8% more range than the "Power" profile on average, for some tracks up to 20%.

Low level commands

These commands provide direct access to the SEVCON configuration objects (SDO = service data object, see CANopen standard).

Reminder: you need to enable CAN write access (feature #15) first, and you may need to clear bit 2 (value=4) in the car bits (feature #14) if you want to get the CFG results.

Command	Function
CFG PRE	Enter pre-operational state (needed for some config objects). Twizy will beep and STOP lamp will be on if successful.
CFG OP	Enter operational state.
CFG READ index_hex subindex_hex	Read CANopen SDO object. index_hex: SDO address (hexadecimal) subindex_hex: SDO field (hexadecimal)
CFG READS index_hex subindex_hex	Read CANopen SDO object string contents. (Used for fetching fault code descriptions) index_hex: SDO address (hexadecimal) subindex_hex: SDO field (hexadecimal)
CFG WRITE index_hex subindex_hex value_dec	Write CANopen SDO object. index_hex: SDO address (hexadecimal) subindex_hex: SDO field (hexadecimal) value_dec: new value (decimal) The command will first read and output the old value before trying to write the new value. **ATT: the OVMS cannot store any backup values! Save/copy the old value to be able to restore it later on!**
CFG WRITEO index_hex subindex_hex value_dec	Write-only CANopen SDO object (i.e. no READ before write). index_hex: SDO address (hexadecimal) subindex_hex: SDO field (hexadecimal) value_dec: new value (decimal) The command will **not** output the old value. This needs to be used for write-only SDOs, as the read attempt would result in an error. **ATT: the OVMS cannot store any backup values! Save/copy the old value to be able to restore it later on!**

Error codes

...will be shown as hexadecimal values. If an SDO cannot be read or written, the SDO index and subindex will be indicated as well. Example:

```
ERROR 0020 SDO 0x5110.00: 0x0005
```

This means:

- – Error code "0020" containing...

    - – ...error class "002x" = switching to pre-operational mode

    - – …error detail (x) "0" = no low level error

  - – last accessed SDO was 0x5110.00 = current mode register

  - – value of SDO was 5 = "operational mode"

As there was no low level error, the pre-op mode switch was denied by the OVMS due to security constraints (i.e. no pre-op mode while Twizy is in normal drive mode).

Error classes are:

  - – 001x = SEVCON login

  - – 002x = switching to pre-operational mode

  - – 003x = command parameter out of range (x = failed parameter number)

  - – 004x = unknown hardware

CAN I/O detail error codes (x) may be:

  - – 1 = No CAN bus write access (feature #15)

  - – 2..5 = SDO read errors

  - – 8..9 = SDO write errors

  - – A = Timeout

  - – B = Component offline

So for example a code "004B" means the OVMS cannot check the SEVCON type as it's offline.

If on a CAN error the SDO value is "0x08...", it's the CANopen abort code. See SEVCON manual for a list of these.

CAN level errors are temporary, repeating the last command will normally succeed. If a state change cannot be performed, try to switch the Twizy off and on again before repeating the command. When switching on, do not continue to "GO". See above.

If an error persists or can be reproduced in a certain way, please post a complete description to the OVMS forum or mailing list.

MSG API

This has not been implemented yet as an alternative for the CFG commands but is not much of a problem. It will be most useful only for an App extension for the Twizy though. If you'd like to develop that, contact the OVMS team.

Two MSG commands exist for querying and clearing the SEVCON logs. See below.

SEVCON fault alerts

The SEVCON continuously monitors itself for critical conditions and tries to alert you about them by flashing an error LED... which unfortunately is not visible on the Twizy.

The OVMS automatically checks for active SEVCON faults every 10 seconds. Multiple errors can be active at once, if a new highest level error is detected, the OVMS sends a text alert message containing the error code and a short textual description as delivered by the SEVCON. The alert will be sent by MSG and/or SMS protocol depending on your alert configuration (parameter #3 "Notifications").

The alert will only contain the highest priority error. As the SEVCON will clear all active faults on the next power cycle, the OVMS will also send all active faults to the server using historical records, so you can retrieve them later on.

Note: if you switch to pre-operational mode for longer than about a second, the SEVCON will consider this an alert as well – the OVMS does not suppress this error code, so you can use this to test the alert function.

SEVCON logs

The SEVCON logs and/or counts faults, critical conditions and system events like logins and firmware upgrades. It also monitors some important values and logs their minimum and maximum values. The logs can be used to detect errors and to check if a new configuration works well. The min/max logs can be used to measure engine, controller and battery load.

The OVMS can access these logs:

1. Alerts (currently active faults)

2. Fault events (max 40 entries)

3. System events (max 20 entries)

4. Event counters (10)

5. Min/max log (2)

On categories 2-4, time stamps of the occurences are available. These refer to the "key hours" = time the SEVCON has been switched on. To be able to relate the log entries to the current time, the OVMS also reads the current key hours time stamp.

Log categories 2-5 can be reset. A reset may not clear all entries, very important entries can remain. Also note there are internal logs that cannot be reset by the user, only by SEVCON (for example the second entry of the min/max log).

To tell the OVMS to query a log and send the contents to the server, issue the `QueryLog` command, to let the OVMS reset one or all logs, send the `ResetLog` command (see MSG/IP command section and shell scripts for details).

Key time record format:

```
RT-ENG-EvtLogKeyTime,<timestamp>
      ,0,<KeyHour>,<KeyMinSec>
```

- Key hour = total count of hours controller has been switched on

- Minutes and seconds: resolution 1 = 15 seconds

Alerts record format:

```
MP-0 HRT-ENG-EvtLogAlerts,<timestamp>
      ,<n>,<Code>,<Description>
```

- n = entry number

- Code = hexadecimal error code

- Description = short text description

Faults and system record format:

```
MP-0 HRT-ENG-EvtLogFaults,<timestamp>
MP-0 HRT-ENG-EvtLogSystem,<timestamp>
      ,<n>,<Code>,<Description>
      ,<TimeHour>,<TimeMinSec>
      ,<Data1>,<Data2>,<Data3>
```

- Time = key time of occurence

- Data1..3 = additional internal data

Counter record format:

```
MP-0 HRT-ENG-EvtLogCounts,<timestamp>
      ,<n>,<Code>,<Description>
      ,<LastTimeHour>,<LastTimeMinSec>
      ,<FirstTimeHour>,<FirstTimeMinSec>
      ,<Count>
```

- First and last time of occurence is logged

- Count = total event occurence count

Min/max record format:

```
MP-0 HRT-ENG-EvtLogMinMax,<timestamp>
      ,<n>
```

```
,<BatteryVoltageMin>,<BatteryVoltageMax>
,<CapacitorVoltageMin>,<CapacitorVoltageMax>
,<MotorCurrentMin>,<MotorCurrentMax>
,<MotorSpeedMin>,<MotorSpeedMax>
,<DeviceTempMin>,<DeviceTempMax>
```

- n: 0 = Service/customer log, 1 = SEVCON log (read only)

- Voltages: in 1/16 V, i.e. 917 = 57,313 V

- Currents: in A

- Speeds: in RPM (7250 = 80 kph)

- Temperatures: in °C

Lock mode and Valet mode support

Att: these functions need firmware CFG mode support and CAN write access.

The Twizy has no native locks and valet mode, so we just implement our own ;-)

There may be limited parking service use situations, but often you want to let someone do a short tour or test drive in your Twizy.

So the idea is to lock the Twizy to a limited speed, either permanently or automatically when exceeding a defined trip length.

Command	Function

`LOCK [kph]`
`/ App Lock`
`/ MSG cmd 20`
	Immediately and persistently limit the maximum speed. The limit will survive profile switching, even CFG RESETs, and will still apply after unplugging the OVMS.
	
	From the App, the PIN input will be used as the kph parameter. Default is 6 kph if omitting the parameter.
	
	If kph is lower than the default reverse speed (~10 kph), the reverse speed will also be limited.
	
	If the LOCK is issued while driving faster than the new limit, the Twizy will not brake down, it will just roll out until the new limit has been reached.

`UNLOCK`
`/ App Unlock`
`/ MSG cmd 22`
	Clear the speed lock and reset maximum speed to the current profile configuration (or default if not configured).
	
	If the Twizy was locked by the valet mode (see below), the valet trip length will be extended by 1 km but stay active.
	
	The App's PIN input is ignored.

Command	Function
VALET [km] / App Valet On / MSG cmd 21	Activate valet mode and set the allowed trip length in km. The result notification will contain the trigger odometer value. *Note: valet mode will not work if the OVMS is unplugged before reaching the trip length trigger!* From the App, the PIN input will be used as the km parameter. Default trip length = 1 km. As soon as the allowed trip length is exceeded, the OVMS will issue a LOCK to 6 kph. If you UNLOCK with valet mode active, you allow for 1 more km. You may also use the LOCK command to set another speed limit, after (!) the valet lock has been triggered.
UNVALET / App Valet Off / MSG cmd 23	Clear valet mode trip length restriction. If a speed lock is currently active, also clears the lock. The App's PIN input is ignored.

GPS track logging

The Twizy OVMS will automatically send GPS records to the OVMS server. This is an extension to the standard location message of the framework: the records are history entries that will be kept for 24 hours, so you can retrieve a complete GPS track after your trip.

You'll currently need to use the perl client to retrieve these records. Record type is "RT-GPS-Log". Remember, records will be deleted after 24 hours.

Normal checkpoint frequency is one per minute. If you enable location streaming mode via App or by setting "FEATURE 8 1", log entries will be sent every five seconds.

GPS log record format:

```
RT-GPS-Log,<timestamp>,<odometer_mi_10th>
      ,<latitude>,<longitude>,<altitude>
      ,<direction>,<speed>
      ,<gps_fix>,<gps_stale_cnt>,<gsm_signal>
      ,<current_power_w>,<power_used_wh>,<power_recd_wh>
      ,<power_distance>,<min_power_w>,<max_power_w>,<car_status>
V3.6:
      ,<max_drive_pwr_w>,<max_recup_pwr_w>
      ,<autodrive_level>,<autorecup_level>
      ,<min_current_a>,<max_current_a>
```

- Odometer is in 1/10th miles

- Speed will be output in the unit configured (mph/kph)

- GPS fix is 0/1

- Stale counter begins at 120 on reception of valid coordinates, counts down about once per second to 0

- GSM signal quality is in the range 0 .. 31, higher means better

- Car status is hexadecimal encoded bit field:

    bit 0 = 0x01: 1 = Footbrake

    bit 1 = 0x02: 1 = Forward mode "D"

    bit 2 = 0x04: 1 = Reverse mode "R"

    bit 3 = 0x08: 1 = "GO" = Motor ON (Ignition)

    bit 4 = 0x10: 1 = Car awake (key turned)

    bit 5 = 0x20: 1 = Charging

    bit 6 = 0x40: 1 = Switch-ON/-OFF phase / 0 = normal operation

    bit 7 = 0x80: 1 = CAN-Bus online (test flag to detect offline)

- **V3.6**: max drive/recup power levels (BMS limits) in W

- **V3.6**: auto power adjustment levels (drive/recup) as a factor 0..1

- **V3.6**: battery current min/max in A measured on the GPS section

Example:
```
RT-GPS-Log,2013-01-01 19:03:40,27461,51.257704,7.160899,139,242,48,1,119,20,...
```

Antenna optimization

The GPS log entries include the current vehicle speed, a GPS stale counter (counting down from 120 to 0, the lower it is the staler the coordinates) and the GSM signal quality (value range 0..31, the higher the better).

Analyzing these values + connection drops, you can use the GPS log to optimize your antenna positioning:

– Define a fixed route for a test drive.

– Take the tour, retrieve the GPS log.

– Change the antenna position.

– Take another tour, retrieve the new GPS log.

– Compare GPS and GSM signal qualities along the track, keep the antenna position that delivers better values.

Perl clients & shell scripts

The perl clients are simple text clients for the OVMS server. They communicate with the server just the same way an App does, but are not limited by a graphical user interface. Think of it as an OVMS shell.

To use a perl client, you need a working perl installation and some additional perl packages (all available by CPAN). Follow the instructions in the "HOWTO-Server" document.

All client scripts are located in the "client" directory.

"**status.pl**" retrieves the current status (last known to the server) of a standard protocol message type, for example:

Type	Meaning
S	State (SOC, charge state etc.) (Default type)
D	Environment (temperatures etc.)

L	Location
T	Update time
F	Firmware version
V	Firmware capabilities

See document "OVMS Protocol" for details on the record structures for these.

The **command client** is "**cmd.pl**". It needs a vehicle and password configuration, edit "ovms_client.conf" accordingly.

Cd to the "client" directory. Some usage examples:

    ./cmd.pl 31

=> lists the directory of currently stored history records

    ./cmd.pl 32 "RT-GPS-Log" >gpslog.csv

=> save all GPS log records to "gpslog.csv"

    ./cmd.pl 41 "*100#"

=> query SIM card account balance (change USSD code if needed)

"**querybalance.sh**" is a convenience shell script for the last example.

"**rt_fetchdata.sh**" retrieves the Twizy specific historical records (data logs) from the server, adds header lines and writes into CSV files using a common file name prefix. So for example to fetch the records from your latest trip to berlin, do:

    ./rt_fetchdata.sh berlin-140805

Hint: to import the CSV files into [OpenOffice](#) Calc, just set the field separator to "," and adjust the timestamp column type to "date ymd". You may also need the "US" type for columns containig floating point values.

The "**rt_*logs.sh**" scripts can be used to retrieve and/or reset the SEVCON event counters and logs. As these logs are not sent to the server by default, you first need to tell the OVMS module to do so by issueing:

    ./rt_querylogs.sh

Note: the Twizy needs to be switched on.

After the command succeeded, retrieve the logs from the server by:

    ./rt_fetchlogs.sh [fileprefix]

To reset all resettable logs, do:

```
    ./rt_resetlogs.sh
```

Note: you can also reset specific logs or counters, see command description for more.

Twizy MSG/IP Commands

OVMS functions can generally be called via SMS or the OVMS network protocol ("MSG"). SMS are meant for humans, MSG primarily for software (Apps), but as MSG commands can be sent using the Perl client, they can also be useful for automated remote access and configuration.

For info on the common OVMS MSG functions, see the general OVMS user guide.

Any command request will generally be replied to (if processed) by one or more text lines. Each line will start with the protocol header "MP-0" followed by the command code "c..." and a general result code. Result codes are:

- 0 = ok / normal command execution

- 1 = failed (normally followed by error description)

- 2 = unsupported

- 3 = unimplemented

If the command call returns data, it will follow the result code. Arguments and return values of commands typically are separated by comma ",".

Example call to command code 204 with two arguments:

```
    cmd.pl 204 55,80
=>
    MP-0 c204,0,55,80,0,0,48
```

That's an OK (0) followed by five numerical return values (55,80,0,0,48).

The following table describes the Twizy specific commands. Only command specific return values (following the command result code) are described.

Please note: this API definition currently is in **preliminary** state, not fixed. It may well change if/as needed by the App development. Please contact me if you need to fix one or more commands of the API.

Function	Code	Parameters	Results & Remarks
QueryRange	201	–	`maxrange` (in user unit)
SetRange	202	`maxrange`	same as `QueryRange`
QueryChargeAlerts	203	–	`range,soc,etr_range,etr_soc,etr_full`

Function	Code	Parameters	Results & Remarks
			• range in user unit • soc in percent • ETR... = estimated times remaining to reach charge state in minutes
SetChargeAlerts	204	range, soc	same as `QueryChargeAlerts`
BatteryAlert	205	–	– Internal command: send text alert on new battery alert status, normally not called by a user/client.
BatteryStatus	206	–	– Request battery status update: OVMS sends current (last known) battery status by historical records `RT-PWR-BattPack` and `RT-PWR-BattCell`. These are not returned by the command call but must be retrieved separately using command code 32.
PowerUsageNotify	207	mode	– Request power usage text notification. Mode can be "`E`" for efficiency report or "`T`" for power totals.
PowerUsageStats	208	–	– Request power usage stats update: OVMS sends current (last known) power usage statistics by historical record `RT-PWR-UsageStats`. These are not returned by the command call but must be retrieved separately using command code 32.
ResetLogs	209	which	The command call returns the new log entry count or an error description if anything fails (i.e. CAN access). which: 1 = Alerts (not resettable, do power cycle) 2 = Fault events 3 = System events 4 = Counters 5 = Min/max log or 99 = all
QueryLogs	210	which, start	The command call returns the log entry count or an error description if anything fails (i.e. CAN access).

Function	Code	Parameters	Results & Remarks

			which:
			 1 = Alerts (not resettable, do power cycle)
			 2 = Fault events
			 3 = System events
			 4 = Counters
			 5 = Min/max log
			
			start = first entry # to fetch, default 0
			At most 10 entries can be sent per call, so if a log contains more, you need to query the next chunk with a new call.
			
			History server records returned:
			`RT-ENG-LogKeyTime`
			`RT-ENG-LogAlerts`
			`RT-ENG-LogFaults`
			`RT-ENG-LogSystem`
			`RT-ENG-LogCounts`
			`RT-ENG-LogMinMax`

Thanks to...

– the whole OVMS team for developing the OVMS platform

– Mark Webb-Johnson for excellent support and constructive feedback

– Bruce McMillan for the Twizy App artwork

– everyone at the german Twizy forum http://www.twizy-online.de/ who participated in decoding the CAN messages and shared feature ideas

– Eric and Julien for Twizy 45 readouts and tests

– Francesco Cariati for OVMS support in Italy and italian translations

– Renault for the Twizy

– Tesla Motors for kicking fossilized butts just in time

Contact / Feature requests / Firmware downloads

If you need help, want to give some feedback, find bugs, have an idea on improving or miss some feature, you can contact me (Michael Balzer) on the OVMS developer list, by email (dexter@dexters-web.de) or as "dexter" on the german Twizy forum.

If you're looking for the latest full featured ready to flash Twizy firmware build, also have a look at the german Twizy forum:

TWIZY-Forum → Tips und Tricks → Projekte → OVMS

If you want to support our work or express your gratitude, feel free to donate via PayPal. The OVMS is a non-profit project, any donation is welcome. You can support the general OVMS project on the http://www.openvehicles.com/helpme page, and you can support the OVMS Twizy development by sending a donation to dexter@dexters-web.de – thank you!

If you want to take part in the development in any way, please subscribe to the OVMS developers mailing list:

http://lists.teslaclub.hk/mailman/listinfo/ovmsdev

Remember, this is a community project, any help is appreciated :-)

Thank you!