OVMS

Open Vehicle Monitoring System

www.openvehicles.com

Renault Twizy v3.0.1
OVMS Hardware Module v2
User Guide v1.2.2 (2014/01/04)

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

Recent changes

- 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
  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 **1-5 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.

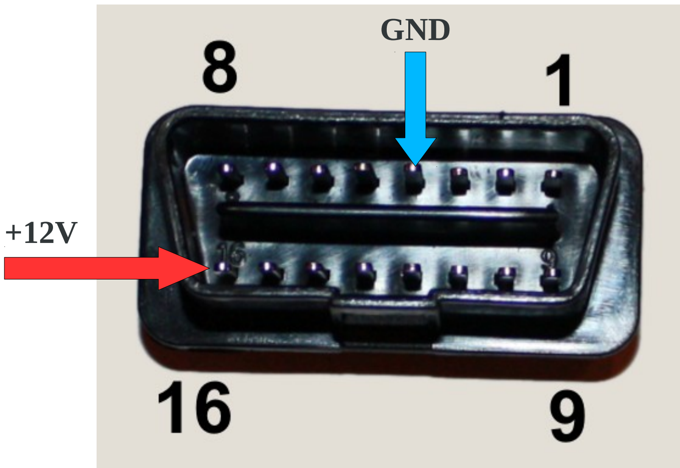
Firmware update / SIM installation

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 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 TWIZY42 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: 2.2.2/RT2.6/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

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)
- Time estimation for full charge (minutes)
- Range (unit as configured)
- SOC (State of charge in %)
- Odometer (unit as configured)

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.

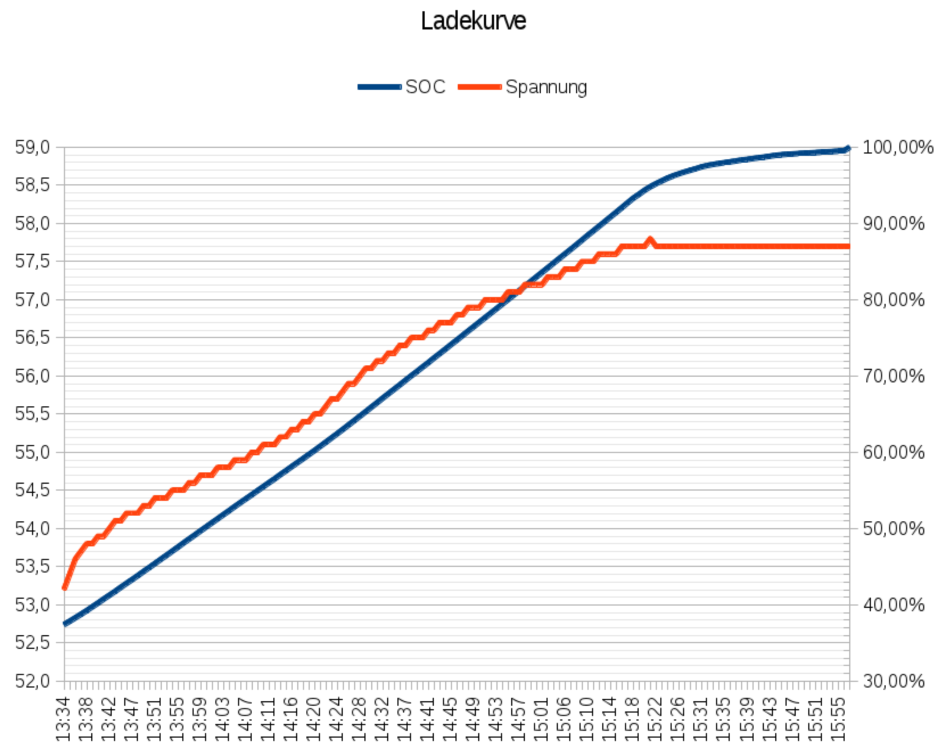
The status message will also be sent if charging starts after an interrupt, and when fully charged. You may also configure a minimum SOC alert (using standard FEATURE #9), and a charge alert for a sufficient range and/or SOC (see "Charge alerts").

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-PWR-BattPack`" 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.

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.

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.

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

Command	Function
`RANGE 80`	Set ideal maximum range 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

Command	Function
`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 57,7 V for the pack = 4,12 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.

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`

Command	**Function**
	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. 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-PWR-BattPack`" and "`RT-PWR-BattCell`". See "Perl client" on how to retrieve these.

Battery pack record format:

```
RT-PWR-BattPack,<timestamp>,<packnr>,
```

```
            ,<nr_of_cells>,<cell_startnr>
            ,<volt_alertstatus>,<temp_alertstatus>
            ,<soc>,<soc_min>,<soc_max>
            ,<volt_act>,<volt_act_cellnom>
            ,<volt_min>,<volt_min_cellnom>
            ,<volt_max>,<volt_max_cellnom>
            ,<temp_act>,<temp_min>,<temp_max>
            ,<cell_volt_stddev_max>,<cmod_temp_stddev_max>
```

- "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 1/100 V; "cellnom" = nominal single cell voltage
  - ...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)

Example:

```
RT-PWR-BattPack,2013-01-01 19:06:45,1,14,1
      ,2,1
      ,8612,8612,9698
      ,5610,400,5390,380,5740,410
      ,10,8,10
      ,5,1
```

(line breaks added for readability)

Battery cell record format:

```
RT-PWR-BattCell,<timestamp>,<cellnr>,<packnr>
        ,<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-PWR-BattCell,2013-01-01 19:06:45,1,1
      ,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.

Power statistics are currently reset when switching the Twizy on, and the report is sent after switching the Twizy off. No configuration is currently needed, statistics are enabled by default and will run in the background.

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:
	`Efficiency 132 Wh/km R=6%`
	`Const 33% 93 Wh/km R=7%`
	`Accel 34% 204 Wh/km R=1%`
	`Decel 33% 97 Wh/km R=13%`
	`Up 65m 108 Wh/km R=14%`
	`Down 33m 140 Wh/km R=7%`
	
	Read: total efficiency on that trip was 132 Wh/km, and 6% of the energy used has been regained by recuperation. Following three lines give the speed sections with their percentage of the

Command	Function
	whole tour. Up & down 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-UsageStats`", see "Perl client" on how to retrieve these from the server. Hold time is 24 hours.

Power usage record format:

```
RT-PWR-UsageStats,<timestamp>,0
 ,<speed_CONST_dist>,<speed_CONST_use>,<speed_CONST_rec>
 ,<speed_ACCEL_dist>,<speed_ACCEL_use>,<speed_ACCEL_rec>
 ,<speed_DECEL_dist>,<speed_DECEL_use>,<speed_DECEL_rec>
 ,<level_UP_dist>,<level_UP_hsum>,<level_UP_use>,<level_UP_rec>
 ,<level_DOWN_dist>,<level_DOWN_hsum>,<level_DOWN_use>,<level_DOWN_rec>
```

- All "use" and "rec" values are in internal integer representation, to convert to Wh divide by 22500.
- "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

Example:

```
RT-PWR-UsageStats,2013-01-14 11:41:42,0
      ,6947,1555358,101562
      ,7283,3371721,31364
      ,7108,1793290,236781
      ,600,65,1702762,246228
      ,510,33,1719996,116393
```

(line breaks added for readability)

SEVCON controller 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!

NO SUPPORT FOR TWIZY 45, ONLY 80 KPH VERSION!
USING THIS ON TWIZY 45 CAN BREAK YOUR CONFIGURATION IRREVERSABLY!

Legal disclaimer: by using the OVMS you agree to do so completely on your own risk. 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 can completely break another one. All configuration changes can lead to losing your public roads allowance.

Ok, you've been warned.

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 Twizy motor is a **Letrica** (former Iskra) AC induction motor with max output 13 kW, rated torque 57 Nm and rated speed at 2.100 rpm.

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**.

Some 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 7.3 kWh capacity (6.1 kWh usable), so 18 kW is ~2.5C 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.

The battery charging limit default is set to **2.5 kW** (SDO 0x2870.07) and 43 A (SDO 0x4623.01). This is supposed to be a safety limit for recuperation, 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.

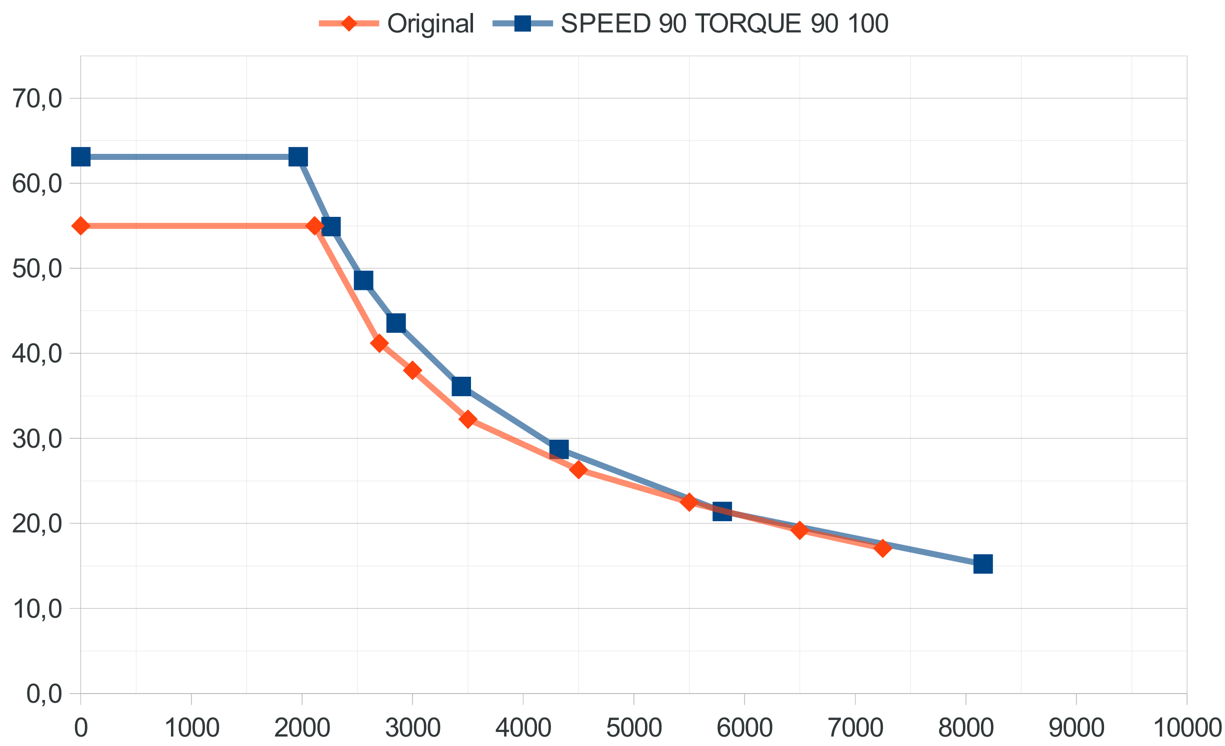
Power map / torque curve

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.

I have included a spreadsheet document called "Twizy Powermap Calculator" which produces a power map from the three 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 and torque tweaks before applying them. Here's an example comparing the original map to one for 90 kph, 90% torque and 100% power:



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 & torque 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.

Raising just speed or torque will produce a map that still stays within the 13000 W mechanical power limit, so the motor will stay within the standard limits. **If you also raise the max power level (param 2 on torque command), this will allow overloading the motor (and possibly the controller), be sure to understand the implications before doing so.**

That's the reason why the maximum power parameter is scaled to 100% = 13000 W. It's maximum allowed value of 130% is more a theoretical one, nothing I would be willing to test on my Twizy without a spare motor and controller in my backyard.

Power and recuperation levels & torque/speed maps

Default power level is 100%, the main tweak here is lowering the level to increase economy / range.

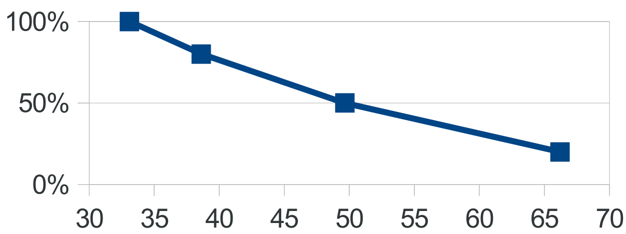
Default recuperation levels are rather low at 18% for both neutral and foot braking modes.

For each of the three situations "driving", "neutral braking" and "foot braking", the SEVON uses both a speed lookup map and the maximum level limit to determine how much of the the actually technically available power will be used at a given 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 power level will be interpolated for the current speed, then multiplied with the maximum 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 default recuperation of ~18% will only be available below 33 kph, at 50 kph this will be scaled down to ~9%, and above 66 kph the recuperation gets down to ~3.6%.

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

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.

Please note: currently, the TSMAP macro command will not change the speeds, just the levels.

<u>Be aware of these caveats on raising the recuperation levels:</u>

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.

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.

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.

Macro commands

ATT: all macro commands are currently designed for the 80 kph version of the Twizy only! You may try using the low level commands (see below) on the Twizy 45, if you know what you're doing.

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.

<u>Example session:</u>

1. Set speed limit to 90 kph, keep default warning @89 kph:
   `CFG SPEED 90`
2. Set max torque level to 90% = ~63.1 Nm, keep max power level at 13 kW:
   `CFG TORQUE 90`
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 and acceleration ramps to max speed, keep defaults for rest:
`CFG RAMPS 100 100`
6. Set ramp smoothing to minimum:
`CFG SMOOTH 0`

Command	**Function**
`CFG SPEED [max_kph] [warn_kph]`	Set speed limit and warn level. Allowed value ranges: max_kph: 6..111, default 80 warn_kph: 6..111, default 89 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. *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 TORQUE [torque%] [power%]`	Set maximum motor torque and (optionally) maximum motor power level. torque%: 10..100, default 78 power%: 10..130, default 100 100% torque = 70.125 Nm 78% torque = default = 55 Nm 81% torque = ~57 Nm = motor peak rating 100% power = 13000 W = motor peak rating **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 TORQUE`	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 POWER [drive%]`	Set maximum power usage level for driving. drive%: 10..100, default 100
`CFG POWER`	Reset maximum power usage level to default (100%).
`CFG RECUP [neutral%] [brake%]`	Set maximum recuperation power levels for neutral braking and footbraking. neutral%: 0..100, default 18 brake%: 0..100, default = neutral% Submitting only one parameter will set both to the same value.
`CFG RECUP`	Reset maximum recuperation power levels to default (18%).
`CFG TSMAP [DNB] [t1%] [t2%] [t3%] [t4%]`	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, 1..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 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..100, default 4 accel%: 1..100, default 25 decel%: 0..100, default 20 neutral%: 0..100, default 40 brake%: 0..100, default 40 Omitted parameters will use their default values.
`CFG RAMPS`	Reset ramp rate configurations 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]`	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 <u>should</u> 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.*
`CFG BRAKELIGHT`	Reset brake light controls to default (disabled).
`CFG RESET`	Reset all macro commands to default.

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.

SERV lamp (pre-operational state)

Some macro commands (i.e. SPEED and TORQUE) 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 the SERV lamp, 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.

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 SERV lamp will be on if successful.

Command	Function
`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 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!**

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.

Error sources may be:

- SEVCON login fails
- operational state cannot be changed
- Parameters out of range
- SDO cannot be read
- SDO cannot be written

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 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.

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>
```

- 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

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 client

The perl client is a simple text client for the OVMS server. It communicates with the server just the same way an App does, but it's not limited by a graphical user interface. Think of it as an OVMS shell.

To use the 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.

The command client is "cmd.pl" located in the "server" directory. It needs a vehicle and password configuration, edit "ovms_client.conf" accordingly.

Cd to the server 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)

Hint: to import 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.

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` • 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.

Function	Code	Parameters	Results & Remarks
`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.

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
- Renault for the Twizy
- Tesla Motors for kicking fossilized butts just in time

Contact / Feature requests

If you need help, want to give some feedback, find bugs, have an idea on improving or miss some feature, please don't hesitate to post on the OVMS forum:

http://www.openvehicles.com/forum

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!