ATTENTION!

• The ECUMASTER PMU is designed for motorsport applications only and cannot be used on public roads!

• The installation of this device should be performed only by trained specialists. Installation by untrained individuals may cause damage to both the device and the vehicle!

• Incorrect configuration of the ECUMASTER PMU can cause serious damage to vehicle components!

• Never modify the device’s settings while the vehicle is moving as it may cause an accident!

• ECUMASTER assumes no responsibility for damage caused by incorrect installation and/or configuration of the device!

• To ensure proper use of ECUMASTER PMU and to prevent risk of damage to your vehicle, you must read these instructions and understand them thoroughly before attempting to install this unit.

• Never short-circuit the wires of the vehicle’s wiring loom or the outputs of the ECUMASTER PMU!

• All modifications to the vehicle's wiring loom must be performed with the negative terminal of the battery disconnected.

• It is critical that all connections in the wiring loom are properly insulated!

• The device must be disconnected before performing any welding on the vehicle!
Table Of Contents

ECUMASTER PMU .................................................................................................................. 5
  Technical ................................................................................................................................. 5
  Specification ............................................................................................................................ 6

SOFTWARE INSTALLATION .................................................................................................... 8
  Compatibility ............................................................................................................................. 8
  Downloading the PMU software ............................................................................................... 8
  Installing PMU software ........................................................................................................... 8

CONNECTING PMU TO PC ....................................................................................................... 11
  CAN – USB interface .................................................................................................................. 11
  ECUMASTER USBtoCAN interface ......................................................................................... 11
  ECUMASTER USBtoCAN pinout .............................................................................................. 12
  Connecting USB interface, wiring schematics ......................................................................... 13
  PMU status ................................................................................................................................ 13
  CAN protocol, CAN topology .................................................................................................... 14

USING PMU SOFTWARE .......................................................................................................... 15
  Launching PMU software .......................................................................................................... 15
  Using PMU software .................................................................................................................. 15
  PMU Client workflow ................................................................................................................ 17

ELEMENT TYPES ..................................................................................................................... 19
  Analog Input .............................................................................................................................. 19
  Power Output ............................................................................................................................. 19
  CANbus Message Object (Mob) ................................................................................................. 19
  CANbus Input ............................................................................................................................. 19
  CANbus Keyboard ..................................................................................................................... 19
  Switch ........................................................................................................................................ 19
  Number ...................................................................................................................................... 20
  Function ...................................................................................................................................... 20
  Wipers Module ........................................................................................................................... 20
  Blinkers Module .......................................................................................................................... 20
  CANbus Export ............................................................................................................................ 20

MANAGING ELEMENTS .......................................................................................................... 20
  Saving elements ......................................................................................................................... 20
  Loading saved elements ............................................................................................................. 20

FUNCTIONS ............................................................................................................................... 21
  Main Principle ............................................................................................................................. 21
  Operations ................................................................................................................................ 21
  Function examples ...................................................................................................................... 23

CHANNEL LOGGING .................................................................................................................. 25
  Graph Log ................................................................................................................................ 25
  Logging Frequency ..................................................................................................................... 26
  Custom Log ................................................................................................................................. 26

PMU PINOUT ............................................................................................................................... 27
  Output Pins ................................................................................................................................. 28
  Input Pins ................................................................................................................................... 28

PMU TEMPERATURE, PLACEMENT AND LOAD BALANCING .................................................. 28
  PMU Placement .......................................................................................................................... 28
  PMU Temperature ....................................................................................................................... 28
  Output Load Balancing ................................................................................................................ 29
ECUMASTER PMU

ECUMASTER PMU is an intelligent power management unit designed to replace the old, traditional and often unreliable fuses and relays. PMU is not only an electronic switch, but a device that can perform all kinds of advanced operations, validate logical conditions, manage abnormal situations, as well as log its parameters for you to have an oversight. PMU allows you to power up external devices such as fan, blinkers, wipers, oil pump etc. and create advanced strategies for those devices using logic with failsafes, condition checking and many more functions. It can communicate and work in tandem with other ECUMASTER CAN devices. It is equipped with over and under current protection, surge protection, 3D gyroscope, accelerometer, LED Status lights, Soft Start, Pulse Width Modulation with Duty Cycle control and more.

Technical
# Specification

## GENERAL

<table>
<thead>
<tr>
<th>Specification</th>
<th>Details</th>
</tr>
</thead>
<tbody>
<tr>
<td>Temperature Range</td>
<td>ACEQ100 GRADE1 (-40 to 120°C)</td>
</tr>
<tr>
<td>CPU</td>
<td>32 bits automotive, 90MIPS</td>
</tr>
<tr>
<td>Reverse polarity protection</td>
<td>Yes, internal up to 16V</td>
</tr>
<tr>
<td>Operating voltage</td>
<td>6-22V immunity to transients according to ISO 7637</td>
</tr>
<tr>
<td>Enclosure</td>
<td>IP65, bespoke CNC machined aluminium</td>
</tr>
<tr>
<td>Size and weight</td>
<td>131x112x32.5mm, 345g</td>
</tr>
<tr>
<td>Connectors</td>
<td>1 x 39 Automotive connector</td>
</tr>
<tr>
<td></td>
<td>1 x M6 stud for battery connection</td>
</tr>
<tr>
<td>PC communication</td>
<td>CAN (Peak CAN, ECUMASTER USBCAN, Kvaser)</td>
</tr>
<tr>
<td>Multiple PMUs</td>
<td>Up to 5 PMUs can work in tandem</td>
</tr>
</tbody>
</table>

## OUTPUTS

<table>
<thead>
<tr>
<th>Specification</th>
<th>Details</th>
</tr>
</thead>
<tbody>
<tr>
<td>10 x high side 25A output</td>
<td>Overcurrent and overheating protection.</td>
</tr>
<tr>
<td></td>
<td>Outputs may be paired to increase continuous current capability.</td>
</tr>
<tr>
<td></td>
<td>Current and voltages measured for each output.</td>
</tr>
<tr>
<td></td>
<td>Inductive load clamp: Vbat - 36V</td>
</tr>
<tr>
<td></td>
<td>Inductive energy dissipation Ear: 460mJ</td>
</tr>
<tr>
<td></td>
<td>Peak current: 120A</td>
</tr>
<tr>
<td>6 x high side 15A output</td>
<td>Overcurrent and overheating protection.</td>
</tr>
<tr>
<td></td>
<td>Outputs may be paired to increase continuous current capability.</td>
</tr>
<tr>
<td></td>
<td>Current and voltages measured for each output</td>
</tr>
<tr>
<td></td>
<td>Inductive load clamp with diode to: -0.5V</td>
</tr>
<tr>
<td></td>
<td>Peak current: 120A</td>
</tr>
<tr>
<td>Total current output</td>
<td>150A continuous</td>
</tr>
<tr>
<td>Output current control step</td>
<td>100mA</td>
</tr>
<tr>
<td>PWM</td>
<td>Yes, available for each 25A output</td>
</tr>
<tr>
<td></td>
<td>Programmable variable Duty Cycle control for each output</td>
</tr>
<tr>
<td></td>
<td>Separate frequency setting ranging from 4Hz to 400Hz for each output</td>
</tr>
<tr>
<td>Soft Start</td>
<td>Yes, available for each 25A output</td>
</tr>
<tr>
<td>Low side wipers braking output</td>
<td>Dedicated output with wiper braking feature.</td>
</tr>
<tr>
<td></td>
<td>Pin shared with output #8</td>
</tr>
<tr>
<td></td>
<td>Current limit: 6A</td>
</tr>
<tr>
<td>+5V</td>
<td>monitored 5V, 500mA output for powering external sensors</td>
</tr>
</tbody>
</table>

## INPUTS
<table>
<thead>
<tr>
<th><strong>Analog Inputs</strong></th>
<th>16 inputs, 10 bit resolution, 0-5V (protected), with software selectable 10K Ohm pullup and pulldowns</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>CAN Keypads</strong></td>
<td>2 x Ecumaster keypads (4, 6, 8, 12 keys), LifeRacing PDU Keypad</td>
</tr>
<tr>
<td><strong>OTHER</strong></td>
<td>16 bicolor LEDs</td>
</tr>
<tr>
<td><strong>Accelerometer/Gyroscope</strong></td>
<td>3D accelerometer with 3D gyroscope for logging and crash detection</td>
</tr>
<tr>
<td><strong>Real Time Clock</strong></td>
<td>Yes, super capacitor for backup power (up to 3 days)</td>
</tr>
<tr>
<td><strong>CAN BUS</strong></td>
<td>2 x CAN2.0 A/B</td>
</tr>
<tr>
<td><strong>CAN standard</strong></td>
<td>2.0A/B 125, 250, 500, 1000 Kbps</td>
</tr>
<tr>
<td><strong>Input/Output Stream</strong></td>
<td>User defined with bit masking</td>
</tr>
<tr>
<td></td>
<td>Up to 48 input messages</td>
</tr>
<tr>
<td><strong>LOGGING (PMU16DL only)</strong></td>
<td>256 Mbytes</td>
</tr>
<tr>
<td><strong>Logging Memory</strong></td>
<td>Variable, defined per channel, up to 500Hz</td>
</tr>
<tr>
<td><strong>PC LOGGING</strong></td>
<td>250 Mbytes</td>
</tr>
<tr>
<td><strong>Logging Speed</strong></td>
<td>Variable, defined per channel, up to 500Hz</td>
</tr>
<tr>
<td><strong>FUNCTIONS</strong></td>
<td>100</td>
</tr>
<tr>
<td><strong>Logical Operations</strong></td>
<td>isTrue, isFalse, =, !=, &lt;, &lt;=, &gt;, &gt;=, AND, OR, XOR, Flash, Pulse, Toggle, Set/Reset Latch</td>
</tr>
<tr>
<td><strong>Number of functions</strong></td>
<td>100</td>
</tr>
<tr>
<td><strong>Number of operations</strong></td>
<td>250</td>
</tr>
<tr>
<td><strong>Update frequency</strong></td>
<td>500Hz</td>
</tr>
<tr>
<td><strong>Special functions</strong></td>
<td>Wipers, Blinkers</td>
</tr>
</tbody>
</table>
SOFTWARE INSTALLATION

Compatibility

PMU Software is compatible with Windows XP/Vista/7/8/8.1/10.

Downloading the PMU software

To configure PMU device a PMU Client must be used (both PMU16 and PMU16DL use the same client). To download PMU software please head to www.ecumaster.com/pmu page.

Installing PMU software

To install PMU software, double click the downloaded “PMUSetup_X_XXX_X.exe”.

Click Next to proceed.
Choose which folder to install the PMU Client to.

Choose whether you want a desktop icon or not.
This is the summary of your installation, if both the folder and icon choice are correct, press Install to proceed. If not, you can go back to make a quick correction.

After the installation is finished, you can choose to launch PMU Client right away.
CONNECTING PMU TO PC

CAN – USB interface

To properly connect PMU to PC using USB 2.0 a special interface must be used. PMU Client supports three interfaces:
- ECUMASTER USBtoCAN interface (can be bought directly from www.ecumaster.com)
- PEAK Systems PCAN-USB
- Kvaser USBcan
Each interface must have its drivers installed. Interface must be bought separately.

ECUMASTER USBtoCAN interface

To use ECUMASTER USBtoCAN interface, user must be equipped in following items:
- USB A to USB B adapter to connect the interface to PC
- DB-9 Cable to connect interface to PMU

To install drivers run EUSBtoCAN_Driver_v1.0.exe and follow installation instructions. Basic cable connection should look like this.

<table>
<thead>
<tr>
<th>Color</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Green Continuous</td>
<td>Device turned on</td>
</tr>
<tr>
<td>Green Flashing</td>
<td>Device turned on and connected to PC</td>
</tr>
<tr>
<td>Green and Orange Flashing</td>
<td>Data transfer in progress</td>
</tr>
<tr>
<td>Orange Continuous</td>
<td>Device turned on, currently in bootloader</td>
</tr>
<tr>
<td>Orange Flashing</td>
<td>Device turned on, firmware update in progress</td>
</tr>
<tr>
<td>Red Continuous</td>
<td>Temporary CAN communication error</td>
</tr>
<tr>
<td>Red Flashing</td>
<td>Permanent CAN communication error</td>
</tr>
</tbody>
</table>
ECUMASTER USBtoCAN is equipped with 120Ohm CAN terminator which can be switched on or off by user. Picture below shows terminator switch location:

![Terminator Switch Location](image)

**ECUMASTER USBtoCAN pinout**

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>+5V</td>
</tr>
<tr>
<td>2</td>
<td>CAN LOW</td>
</tr>
<tr>
<td>3</td>
<td>GND</td>
</tr>
<tr>
<td>4</td>
<td>-</td>
</tr>
<tr>
<td>5</td>
<td>-</td>
</tr>
<tr>
<td>6</td>
<td>GND</td>
</tr>
<tr>
<td>7</td>
<td>CAN HIGH</td>
</tr>
<tr>
<td>8</td>
<td>-</td>
</tr>
<tr>
<td>9</td>
<td>+5V</td>
</tr>
</tbody>
</table>
Connecting USB interface, wiring schematics

To connect PMU to PC, CAN1 CANbus must be used. This is the CAN provided for us for PC communication. To wire the USB - CAN interface, CAN1H and CAN1L pins (See PMU Pinout section) must be used. Twisted pairing is also recommended. CANbus must also be terminated at both ends of the bus. ECUMASTER USBtoCAN is equipped with one terminator that can be switched on or off (See ECUMASTER USBtoCAN interface section). If the Interface is connected, PMU must be supplied with power, both on the +12SW Pin (See PMU Pinout section) and the M6 stud.

ATTENTION!

Ground wire connection between USBtoCAN and PMU-16 can be DANGEROUS! Ground connection is only allowed if devices before connecting have the same ground potential. Potential difference can be checked by using voltmeter between corresponding grounds.

Here is a simple diagram of USB connection, please note that it assumes that the second terminator is applied by the USBtoCAN interface:

![PMU - PC communication diagram]

PMU status

PMU device is fitted with LED that signals status of the PMU device.

Picture showing the LED location:

There are 6 possible states of PMU LED:

<table>
<thead>
<tr>
<th>Color</th>
<th>Status</th>
</tr>
</thead>
<tbody>
<tr>
<td>Green Continuous</td>
<td>Device is active</td>
</tr>
<tr>
<td>Orange Continuous</td>
<td>Device is active and connected to PC</td>
</tr>
<tr>
<td>Green Flashing Slowly</td>
<td>Device is waiting for Firmware Upgrade</td>
</tr>
<tr>
<td>Orange Flashing Slowly</td>
<td>Device is performing Firmware Upgrade and is connected to PMU</td>
</tr>
</tbody>
</table>
CAN protocol, CAN topology

CAN topology looks like this:

For 1Mbit/s connection (CAN 1), following rules must be abided:
- Maximum unterminated cable length from device to CANbus is 30 cm
- Maximum bus length is 40m.
- Maximum of 30 nodes
- 120 Ohm terminators must be applied at both ends of CAN bus.
- Twisted pairing is required.

For 500kbit/s connection (CAN 2) following rules must be abided:
- Maximum unterminated cable length from device to CANbus is 30 cm
- Maximum bus length is 100m.
- Maximum of 30 nodes
- 120 Ohm terminators must be applied at both ends of CAN bus. PMU is equipped with CAN2 terminator which can be turned on or off via PMU Client (Tree View → CAN Setup)
- Twisted pairing is required.
USING PMU SOFTWARE

Launching PMU software

To run PMU software either doubleclick the icon on desktop, or use start menu to find it.

Using PMU software

When PMU Client is launched for the first time, user will be asked to enter the name of his device. All projects will then be saved to the directory corresponding to devices name.

Using the Client is pretty straightforward. The Project Tree window is the most significant one. You can use it to set up analog inputs, configure power outputs, create functions etc. To create elements, either use the toolbar with icons located on Project Tree window, use Alt + A Keyboard shortcut or click Add button, then select the type of element you want to create.
To upload current project to on-device flash memory, either use the Make Permanent button, or use F2 keyboard button. The PMU status LED will flash with orange color (See PMU Status section).

To save a copy of your current project on hard drive either use Ctrl+S keyboard shortcut, or use the Clients toolbar.

Right on the main screen there are also windows with monitoring functionality. They are updated in real time and show various parameters reported from particular pins as well as values of elements created by user.
On the left, there is a *Tree View* double clicking any item on it, will bring up it’s window to current desktop.

---

**PMU Client workflow**

PMU Client workflow is really simple. You can monitor parameters of your channels, make adjustments, follow the graph log to understand channels behavior, you can create elements, then use those to control other elements or channels.

Elements and channels in this case can be anything, Power Output, Analog Input, Function, Switch etc. Let’s take a look at two more in-depth use cases.
These are just two examples of how PMU Client can be used, but the possibilities are endless.
ELEMENT TYPES

--- Analog Input

Analog inputs are input devices connected directly to the PMU. Below is the list of analog inputs supported by PMU.

- Switch - Active High
- Switch - Active Low
- Rotary Switch
- Analog Sensor

Head to Wiring section to see wiring diagrams for each type of analog input.

--- Power Output

Power Outputs are devices powered by PMU which are connected to particular output pins. Power Output examples:

- Fuel Pump
- Fan
- Oil Pump

Every Power Output has over current, under current (both user configurable) and overheat protection. They can be either turned on by default, controlled by function or triggered by another element such as Analog Input.

--- CANbus Message Object (Mob)

CANbus Message Objects are interfaces needed for data gathering from CAN Stream. They read a particular number of frames (user defined, up to 8) starting from particular ID (also user defined) from chosen CANbus. Maximum of 8 CANbus Message Objects can be created for each CANbus. They also have special timeout flag which can be used to set up a different strategy if a connection from another CAN device times out.

--- CANbus Input

CANbus Inputs use CANbus Message Objects to extract data from chosen frame. They are able to read a user defined number of bits starting from user defined position, then apply offsets, division or multiplication to the extracted data.

--- CANbus Keyboard

CANbus Keyboard is an element that provides communication with Keyboard. It is used to define and handle Keyboard buttons, set their color, type etc.

--- Switch

You can use two types of switches:

- Latching Switch – switches between user defined number of states.
- Press Hold Switch – short press switches between high and low state, holding it triggers third state.
**Number**

Number is simply an integer. It can be either a typed in constant or value of chosen channel. Number is mostly used for comparison or to control Duty Cycle in Pulse Width Modulation (See *PWM* section).

**Function**

Function is one of the most important elements. It can be used to create a set of rules, conditions to Power on an output device. Function always evaluates to either 0 or 1. If the function is used as control channel for a Power Output, the Power Output will be turned on if function returns 1 and turned off if function returns 0. Functions will be explained in depth later on in *Functions* section.

**Wipers Module**

Wipers module is a separate module to control the Wipers. It consists of two Power Outputs for slow and fast wiper speed and an Analog Input for Park Switch. To use park functionality, Output Pin O8 is provided to use for slow wiper speed. To see the wiring instructions, head to *Wiring* part of the manual.

**Blinkers Module**

Blinkers module is a bit like *Wipers Module*, it is a separate element which controls the blinkers. It consists of two Power Outputs for left and right blinkers and needs three inputs (ie. CANbus Keyboard) to control the Left Blinker, Right Blinker and Hazard Lights.

**CANbus Export**

CANbus Export allows you to broadcast (send) data to the CANbus, which then can be used by other CAN device. For example, you can transmit the status of a Fan to other PMU or EMU Black.

**MANAGING ELEMENTS**

**Saving elements**

Most elements can be saved to hard drive. To save your element, click icon on elements toolbar and choose where you want to save it.

**Loading saved elements**

Elements that can be saved, can also be loaded. To load your element, click icon on elements toolbar and choose file to load.
FUNCTIONS

Main Principle

The idea behind functions is to create a set of rules by combining different operations for various channels or elements. This set of rules is evaluated to a logical true (1) or false (0) result. Function then can be used to turn, for example, Power Output on or off. Quick example could be turning the secondary fuel pump on if a fault is detected on the primary one.

Operations

Following operations are available to use in functions:

<table>
<thead>
<tr>
<th>Test Operations</th>
<th>Returns true (1) if channel is true (its value is other than 0)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Is True</td>
<td></td>
</tr>
<tr>
<td>Is False</td>
<td>returns false (0) if channel is false (its value is equal to 0)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Comparison Operations</th>
<th>Returns 1 if Channel is equal to Constant</th>
</tr>
</thead>
<tbody>
<tr>
<td>Equal</td>
<td></td>
</tr>
<tr>
<td>Not Equal</td>
<td>returns 1 if Channel is not equal to Constant</td>
</tr>
<tr>
<td>Less</td>
<td>returns 1 if Channel is less than Constant</td>
</tr>
<tr>
<td>Less or Equal</td>
<td>returns 1 if Channel is less or equal to Constant</td>
</tr>
<tr>
<td>Greater</td>
<td>returns 1 if Channel is greater than Constant</td>
</tr>
<tr>
<td>Greater or Equal</td>
<td>returns 1 if Channel is greater than or equal to Constant</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Logical Operations</th>
<th>Returns 1 if both Channel #1 and Channel #2 are true</th>
</tr>
</thead>
<tbody>
<tr>
<td>And</td>
<td></td>
</tr>
<tr>
<td>Or</td>
<td>returns 1 if either Channel #1 or Channel #2 or both are true</td>
</tr>
<tr>
<td>Xor (Exclusive Or)</td>
<td>returns 1 if either Channel #1 or Channel #2 are true, but not both at the same time</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Pulse Generation Operations</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Flash</td>
<td>Flash is triggered by Channel and returns 1 for the time specified in Time On and 0 for the time specified in Time Off as long as Channel is in High State. If Low State is detected, Flash will immediately turn off.</td>
</tr>
<tr>
<td>Pulse</td>
<td>Pulse switches between 1 for the time specified in Time On and 0 for the time specified in Time Off for the amount of times specified in Count. It can be set to trigger by Rising Edge or Falling Edge. Pulse will still continue for Count number of times even if Channels signal is lost. Pulse will also ignore any Channel input until Count is reached. If Time On is set to 0, Pulse will generate a short impulse.</td>
</tr>
</tbody>
</table>
## Operations with state

<table>
<thead>
<tr>
<th>Feature</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Toggle</strong></td>
<td>Toggles between 0 and 1 whenever <strong>Channel</strong> triggers a signal edge. It can be set to trigger by Rising Edge or Falling Edge. If Default State is checked, default state of the Function after device is turned on will be 1, otherwise 0. Toggle remembers its last state, therefore if <strong>Channel</strong> signal is lost, Toggle will remain in the last steady state.</td>
</tr>
<tr>
<td><strong>Set-Reset Latch</strong></td>
<td><strong>Set Channel</strong> sets the Latch to 1 when a high state is detected on the selected <strong>Channel</strong>, <strong>Reset Channel</strong> resets the Latch to 0 when a high state is detected on the <strong>Channel</strong> selected here. Reset has a priority, therefore if Channels both for Set and Reset are at high level, the Latch will be reset. If Default State is checked, default state of the Function after device is turned on will be 1, otherwise 0. Set-Reset Latch remembers its last state, therefore if both Channels are at low state, Set-Reset Latch will remain in the last steady state.</td>
</tr>
</tbody>
</table>
Function examples

We want the secondary fuel pump to turn on if a fault is detected on the first one or when fuel pressure drops below or is equal to 400kPa. To do this, we need to create a new function, let's call it `f_switchPumps`. Now to test for the first condition we will use *Is True* operation and to test for the second, we will use *Less Or Equal* operation. Their configuration will look like this:
And the final function will look like this:

```
function f_switchPumps
  value =
    o_fuelPump.fault
    and ... 
    or
    c_fuelPressure <= 400
    and ...
    or
    ...
```

The flow diagram for that function:

```
Fuel Pressure from CAN
  Lower than 400kPa

OR

Primary Fuel Pump Detected

Secondary Fuel Pump Output
```

Switch Pumps function diagram
Another diagram example of Fan Output:

CHANNEL LOGGING

Graph Log

PMU Client is equipped with a tool to show channels graph in real time. There are two ways to add a channel to the graph. First way is to right click on the graph log window, then select Add and type the channels name. The second way is to select a channel in Output Monitor, Analog Monitor Variable Selector, or any window from Tree View → Log group, then press Insert keyboard key. By default two channels are displayed on graph log – the pmu.diag.cpuLoad which displays the CPU load and the pmu.totalCurrent which displays total current drawn from PMU.

Here is how the Graph Log looks like with few more channels added:
Logging Frequency

You can change the logging frequency of a channel by right clicking the channel on graph log, then selecting Set Log Frequency and choosing the desired frequency. Another way is to use Logged Channels window (Tree View → Logged Channels). In that window you can see how parameters are grouped and their logging frequency. It is possible to change logging frequency of a whole group in the same way as a single channel.

Bolded parameters are the ones being logged (not all parameters are logged by default!). Double clicking an item here will toggle between logging and not logging it.

If a new element is created, every subchannel related to it is logged as well.

If you are familiar with EMU logging these are the key differences:

- Not all parameters are logged by default
- Parameters are grouped and their logging frequencies can be changed from 1 Hz to 500 Hz
- Every time an element is created, its subchannels are logged

Custom Log

PMU Client allows user to have 3 separate custom logging groups. Any channel can be added to custom log (Tree View → Custom1, Custom2, Custom3). To add channel to custom group, press right on the channel you want to add to custom log group and select Add to Custom then choose the group.
Output Pins

There are 16 output pins for you to use, 10 25A pins and 6 15A pins. They can be also used in parallel to increase current capacity (See Using output pins in parallel section). All Pins are equipped with over and under current protection, short circuit protection, as well as thermal protection. In case of any of this scenario happening, the output pin will be shut down and appropriate message will be displayed in the PMU Client and the PMU itself. For 25A Output Pins, Soft Start is available and PWM with Duty Cycle control (See PWM section).

Input Pins

There are 16 input pins for you to use as well as a separate +5V Pin to provide power for Analog Inputs such as rotary switches, or analog sensors.

PMU TEMPERATURE, PLACEMENT AND LOAD BALANCING

PMU Placement

PMU should be located in a place that protects it from weather conditions, road debris and road hazard. Even though PMU case works as a radiator, it is also recommended to place PMU somewhere where heat can be dissipated easily, preferably with good airflow.

PMU Temperature

PMU has three thermometers located on its board to independently measure temperature in three separate places.

There are two places where temperature can be checked in PMU Client, first is the status bar and the values TL, TR, TF (Temperature Left, Temperature Right, Temperature Flash):

Second is the PMU window which can be accessed by double clicking PMU on Tree View:
If you are experiencing high temperatures, it is advised to move the PMU to a colder place or provide better airflow to current location.

**Output Load Balancing**

Another way to handle high temperatures is to connect the devices that draw the most current in a way that the transistors for their Output Pins are not grouped up together. In other words, it is a good rule of thumb not to connect devices that draw the most current to output pins which transistors are located next to each other. This way they can dissipate heat better, resulting in lower temperatures. Picture below shows the transistor placement for every output pin:
Output Pin signaling and status

PMU has an ability to signal condition of each Power Output. It is displayed on the device itself and in PMU software. When a new Power Output is created, a special variable which represents the status of Power Output, called `output_name.status` is created as well. This variable can be used to create failsafes or conditions, to set up redundancy etc.

<table>
<thead>
<tr>
<th>Status value in Client</th>
<th>LED color</th>
<th>Status</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>None</td>
<td>Off</td>
</tr>
<tr>
<td>1</td>
<td>Green</td>
<td>On</td>
</tr>
<tr>
<td>2</td>
<td>Orange</td>
<td>Under Current</td>
</tr>
<tr>
<td>3</td>
<td>Red</td>
<td>Over Current</td>
</tr>
</tbody>
</table>
This is the basic PMU communication and connection diagram which contains few key elements:

- PC Communication takes place on CAN 1 as this is the CAN specified to use for PC connection.
- CAN 1 has two 120 Ohm terminators on CAN bus. They are necessary, as PMU does not provide termination on CAN 1.
- CANbus Keyboard is connected to CAN 2.
- Power to PMU is supplied in two ways: First using the ignition which connects to +12SW Pin (See PMU Pinout section), second using positive battery terminal which connects to M6 Bolt located on PMU case.
Wire Size

Suggested wire size for continuous current (chassis wiring, FLRY)

<table>
<thead>
<tr>
<th></th>
<th>0.75mm²</th>
<th>0.75mm²</th>
<th>1.5mm²</th>
<th>2.0mm²</th>
<th>&gt;=2.5mm²</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt; 5A</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>10A</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>15A</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>20A</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>25A</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>AWG 20</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>AWG18</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

For the power connector at least 25mm² (AWG3) is required!

Load examples for popular devices

<table>
<thead>
<tr>
<th>Device</th>
<th>Inrush Current [A]</th>
<th>Continuous Current [A]</th>
</tr>
</thead>
<tbody>
<tr>
<td>Fuel Pump</td>
<td>15</td>
<td>7</td>
</tr>
<tr>
<td>Fan</td>
<td>50</td>
<td>20</td>
</tr>
<tr>
<td>Front Lights</td>
<td>16</td>
<td>10</td>
</tr>
<tr>
<td>EMU Black</td>
<td>30</td>
<td>25</td>
</tr>
</tbody>
</table>
ANALOG INPUT WIRING

Wiring schematics

Most important thing to remember is that PMU has dedicated +5V Pin (See PMU Pinout section) can be used for things like rotary switch, active low switches and analog sensors. Below are basic wiring diagrams.

- Switch connected to GND with Pull Up
• PMU Client configuration:

[Image of PMU Client configuration GUI]

• Switch connected to +5V Pin with pull down:

[Image of analog input pull down configuration]
PMU Client configuration:

- Rotary switch
PMU Client configuration:

Analog Sensor configuration:

Analog Sensor diagram:

To connect analog Sensor, +5V and GND Pins must be used (See PMU Pinout section). Output from Analog Sensor should be connected as a regular Analog Input to PMU.

Analog Sensor diagram:
PMU Client Configuration:

OUTPUT DEVICES

Power Output

Power Outputs are elements that control external devices. You can set up the Minimum Current, Maximum Current, Inrush Current, Inrush Time, PWM, and mechanisms to switch the Power Output on or off.

PMU Client Configuration:
Simple Power Output Setup

This is a quick example of two popular setups, their wiring diagrams and PMU Client configuration. Please note that output pins shown here are just an example.

- Power Output that is turned on as soon as PMU is turned on:

![Power Output Diagram example 1](image)

PMU Client Configuration:

[Power Output configuration example 1](image)
• Power Output that gets switched on by an Analog Input:

![Power Output diagram example 2]

**PMU Client Configuration:**

![Analog Input configuration example 2](Image)

![Power Output configuration example 2](Image)
Wipers

To connect wipers O8 Output Pin must be used for Slow Wiper output. This is the Pin provided to use and it's the only way to use the park switch ability. For fast wiper output any output pin can be used except for O8. Park switch must be connected as an Analog Input.

Configuring wipers in PMU Client is pretty easy. There is a module made specifically for wiper configuration and should be used to set up wipers trouble-free (Project Tree → Wipers Module).

PMU Client Configuration:
Blinkers

Blinkers should be simply connected to PMU Output Pins. Three input channels should be used in this configuration - two channels to control blinkers, one channel to control hazard lights.

PMU Client also provides special module designed to configure blinkers in easy way (Project Tree → Blinkers Module).

PMU Client blinkers configuration
CANbus Keyboard can be connected to any CANbus. If CAN2 was chosen, user has the ability to change CAN2 speed for CANbus Keyboards such as Life Racing (Tree View → CANbus Setup). CAN connection must be terminated at both ends. It is possible to use CAN2 terminator from PMU (Tree View → CANbus Setup). To provide power to CANbus Keyboard one of Output Pins can be used. Up to two CANbus Keyboards can be connected.

PMU Client provides special module for CANbus Keyboard configuration and setup (Project Tree → CANbus Keyboard). Refer to your CANbus Keyboard manufacturers documentation to find out the ID under which CANbus Keyboard works at.
Power Output configuration for CANbus Keyboard:

Exporting Keyboard state

While configuring CANbus Keyboard, you can choose to broadcast the Keyboard state and buttons on chosen CANbus Interface, ID and frequency so that other device such as secondary PMU or EMU Black can acquire and use them.

- PMU controls the CANbus Keyboard, handles key switches, backlighting
- PMU exports keyboard state and buttons to CAN under ID 0x0662
- EMU Black reads data exported by PMU from CANbus on ID 0x0662

Using CANbus Keyboard to signal status

CANbus Keyboard can also be used as a signaling device. You can change the color of any button to represent either the PMU status, or any Power Output status (see Output Pin signaling and status section). To achieve that, a CANbus Keyboard must be configured and connected (See CANbus Keyboard section) and chosen button must be set up as an indicator button in PMU Client.

Here is an example configuration:
This example configures button \texttt{k\_button1} to display the value of \texttt{pmu.status} channel in the following way:

<table>
<thead>
<tr>
<th>Status</th>
<th>Color</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>None</td>
<td>All Power Outputs are Off (\texttt{pmu.status} equal to 0)</td>
</tr>
<tr>
<td>1</td>
<td>Green</td>
<td>At least one Power Output is on and there is no under or over current condition (\texttt{pmu.status} equal to 1)</td>
</tr>
<tr>
<td>2</td>
<td>Orange</td>
<td>At least one Power Output is under current and there is no over current condition (\texttt{pmu.status} equal to 2)</td>
</tr>
<tr>
<td>3</td>
<td>Red</td>
<td>At least one Power Output is over current (\texttt{pmu.status} equal or above 3)</td>
</tr>
</tbody>
</table>

Keep in mind that the button still works as a normal non-latching switch and can be set up to, for instance, trigger Master Retry (See \texttt{Master Retry Channel} section) to reset all outputs.
Using output pins in parallel

PMU allows user to connect output pins in parallel to increase current capacity. Up to three output pins of the same amperage can be used, granting up to 75A continuous current.

PMU Client configuration example:

![Power Output configuration with parallel pins](image1)

**INERTIA SWITCH**

PMU is equipped with a failsafe switch which gets triggered in case of a crash. PMU constantly monitors its gyroscopic values and acceleration in each axis to react quickly in the worst case scenario.

**Inertia Switch**

Inertia Switch immediately shuts down all Power Outputs to prevent any mishaps happening in the event of a crash. Inertia Switch gets triggered on after exceeding acceleration threshold specified by user. Inertia Switch can be accessed by Tree View → Inertia Switch :

![Inertia Switch configuration](image2)
MASTER RETRY CHANNEL

Master Retry Channel allows user to perform reset of Power Outputs. If a Power Output has signaled a fault, due to either over or under current condition, triggering a Master Retry will reset it back to default state. Master Retry can be triggered by any channel or element. Master Retry configuration can be found in Tree View → Global Output Settings → Master Retry Channel.

PWM (PULSE WIDTH MODULATION)

PMU has the ability to apply PWM to 25A Output Pins. The main principle of Pulse Width Modulation is to limit the amount of Power supplied to Power Output by alternating power Output on and off.

Keep in mind that PWM introduces energy loss to heat due to transistor switching on and off. Higher frequencies generate more heat, therefore If you are experiencing overheated status on Power Outputs or high heat in general, either lower the Frequency of PWM, or use a flyback diode to eliminate flyback and lower the thermal load (See Using flyback diode section).
**Soft Start**

PMU has the ability to enable a Soft Start on 25A output Pins. Soft start should be used to prevent switch-on surges of greater altitude. By varying the time on and time off with chosen frequency and duration, a Power Output reaches its state gradually.

![Graph Log](image)

**Duty Cycle**

Duty Cycle represents the percentage of time the Power Output is turned On. If the Duty cycle is set to 50%, the Power Output will be turned on for $\frac{1}{2}$ of time period.

![Graph Log](image)
Duty Cycle can be either set to constant value, or controlled by a Channel.

PMU Client Configuration Example:

![PWM Configuration example](image)

**Using flyback diode**

PWM does introduce a bit of heat depending on the Device powered, and PWM settings. If you are experiencing *Overheated* status on any Pulse Width Modulated Power Output, a flyback diode can be used to eliminate flyback therefore lowering the output transistors thermal load.

![Flyback diode diagram](image)
USING MULTIPLE PMUs

Basic Diagram

To use PMUs in tandem, it is recommended to use CANbus 1. Both devices must be connected to the same bus, terminated at both ends with 120 Ohm resistors. Up to five PMUs can be used at once. For more informations regarding CAN bus see CAN Topology section.

Communication

There is no need for a special configuration per se, but for this setup to work optimally it is advised to make use of CANbus Export and CANbus Input functionality to transmit important data between PMUs. It is vital for those two PMUs to communicate with each other, so that proper strategies can be set up. See CAN Stream section to find out more about CANbus communication.
Using PMU Client with multiple PMUs

PMU Client allows up to 5 PMUs to be connected simultaneously. You can easily switch between them either using Menu Bar → Devices → Set Device #X or using Ctrl + Shift + X keyboard shortcut, where X is the device Number.

All currently connected PMUs are also displayed as a list.

<table>
<thead>
<tr>
<th>#1: PMU_Front</th>
<th>#2: -</th>
<th>#3: -</th>
<th>#4: -</th>
<th>#5: -</th>
</tr>
</thead>
</table>

List of connected PMUs
CANbus Export allows user to broadcast various informations on chosen CANbus. It is essential tool to communicate with other PMUs or ECUMASTER EMU Black device. CANbus Export can broadcast channel states, values, voltage readings, which then can be read using CANbus Input on second PMU or EMU Black. PMU Client also allows you to save your CANbus Export as a .CANX file. This file can be imported by any other PMU.

CANbus Export Example:

CANbus Input allows user to read data from CANbus stream using CANbus Mob (See CANbus Message Object section). You can choose the starting frame, how many bytes to read, how many bits to read and the endianess (byte order) of the data read. If a CANbus Export was created on secondary PMU, CANbus Input (with correct CANbus Message Object) can be used to read exported parameters.
Importing .CANX File

When creating a CANbus Export, you have the option to save this Export as a .CANX file. This file can be imported by other PMU to automatically create CANbus Mob with correct ID, Mob name and CANbus Inputs that correspond to exported Channel.

Example:

Create a CANbus Export, choose broadcasted channels and save it as a .CANX file.

Use Import .CANX option from Project Tree to open the .CANX file.
As we can see, a CANbus Mob and CANbus Inputs were automatically created with correct ID.
Standard CAN Stream

Standard CAN Stream (*Tree View → Standard CAN Stream*) gives user the ability to broadcast key PMU parameters over CAN bus to have an oversight. Parameters themselves are predefined, but user has the ability to broadcast only some part of them, on chosen CAN bus with chosen ID.

Here is how CAN Stream frames are constructed:

### ID BaseID + 0 Frequency: 20Hz

<table>
<thead>
<tr>
<th>ByteID</th>
<th>Channel</th>
<th>Data Width</th>
<th>Data Type</th>
<th>Range</th>
<th>Resolution</th>
<th>Offset</th>
<th>Unit</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>PMU Status</td>
<td>4 bits</td>
<td>Unsigned</td>
<td>0-15</td>
<td>-</td>
<td>0</td>
<td>-</td>
</tr>
<tr>
<td>1</td>
<td>Total Current</td>
<td>8 bits</td>
<td>Unsigned</td>
<td>0-255</td>
<td>1A/bit</td>
<td>0</td>
<td>A</td>
</tr>
<tr>
<td>2</td>
<td>Battery Voltage</td>
<td>8 bits</td>
<td>Unsigned</td>
<td>0-27.75</td>
<td>0.1088V/bit</td>
<td>0</td>
<td>V</td>
</tr>
<tr>
<td>3</td>
<td>Board Temperature Left</td>
<td>8 bits</td>
<td>Unsigned</td>
<td>0-255</td>
<td>1C/bit</td>
<td>0</td>
<td>C</td>
</tr>
<tr>
<td>4</td>
<td>Board Temperature Right</td>
<td>8 bits</td>
<td>Unsigned</td>
<td>0-255</td>
<td>1C/bit</td>
<td>0</td>
<td>C</td>
</tr>
<tr>
<td>5</td>
<td>Flash Temperature</td>
<td>8 bits</td>
<td>Unsigned</td>
<td>0-255</td>
<td>1C/bit</td>
<td>0</td>
<td>C</td>
</tr>
</tbody>
</table>

### ID BaseID + 1 Frequency: 62.5Hz

<table>
<thead>
<tr>
<th>ByteID</th>
<th>Channel</th>
<th>Data Width</th>
<th>Data Type</th>
<th>Range</th>
<th>Resolution</th>
<th>Offset</th>
<th>Unit</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>o1.status, o1.active</td>
<td>3 bits</td>
<td>Unsigned</td>
<td>0-7</td>
<td>0-1</td>
<td>0</td>
<td>-</td>
</tr>
<tr>
<td></td>
<td>o2.status, o1.active</td>
<td>3 bits</td>
<td>Unsigned</td>
<td>0-7</td>
<td>0-1</td>
<td>0</td>
<td>-</td>
</tr>
<tr>
<td>1</td>
<td>o3.status, o3.active</td>
<td>3 bits</td>
<td>Unsigned</td>
<td>0-7</td>
<td>0-1</td>
<td>0</td>
<td>-</td>
</tr>
<tr>
<td></td>
<td>o4.status, o4.active</td>
<td>3 bits</td>
<td>Unsigned</td>
<td>0-7</td>
<td>0-1</td>
<td>0</td>
<td>-</td>
</tr>
<tr>
<td>2</td>
<td>o5.status, o5.active</td>
<td>3 bits</td>
<td>Unsigned</td>
<td>0-7</td>
<td>0-1</td>
<td>0</td>
<td>-</td>
</tr>
<tr>
<td></td>
<td>o6.status, o6.active</td>
<td>3 bits</td>
<td>Unsigned</td>
<td>0-7</td>
<td>0-1</td>
<td>0</td>
<td>-</td>
</tr>
<tr>
<td>3</td>
<td>o7.status, o7.active</td>
<td>3 bits</td>
<td>Unsigned</td>
<td>0-7</td>
<td>0-1</td>
<td>0</td>
<td>-</td>
</tr>
<tr>
<td></td>
<td>o8.status, o8.active</td>
<td>3 bits</td>
<td>Unsigned</td>
<td>0-7</td>
<td>0-1</td>
<td>0</td>
<td>-</td>
</tr>
<tr>
<td>ID</td>
<td>ByteID + 2</td>
<td>Frequency: 62.5Hz</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>----</td>
<td>------------</td>
<td>-------------------</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>BaseID + 2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Channel</td>
<td>Data Width</td>
<td>Data Type</td>
<td>Range</td>
<td>Resolution</td>
<td>Offset</td>
<td>Unit</td>
</tr>
<tr>
<td>0</td>
<td>a1.voltage</td>
<td>8 bits</td>
<td>Unsigned</td>
<td>0-5</td>
<td>0.0196V/bit</td>
<td>0</td>
<td>V</td>
</tr>
<tr>
<td>1</td>
<td>a2.voltage</td>
<td>8 bits</td>
<td>Unsigned</td>
<td>0-5</td>
<td>0.0196V/bit</td>
<td>0</td>
<td>V</td>
</tr>
<tr>
<td>2</td>
<td>a3.voltage</td>
<td>8 bits</td>
<td>Unsigned</td>
<td>0-5</td>
<td>0.0196V/bit</td>
<td>0</td>
<td>V</td>
</tr>
<tr>
<td>3</td>
<td>a4.voltage</td>
<td>8 bits</td>
<td>Unsigned</td>
<td>0-5</td>
<td>0.0196V/bit</td>
<td>0</td>
<td>V</td>
</tr>
<tr>
<td>4</td>
<td>a5.voltage</td>
<td>8 bits</td>
<td>Unsigned</td>
<td>0-5</td>
<td>0.0196V/bit</td>
<td>0</td>
<td>V</td>
</tr>
<tr>
<td>5</td>
<td>a6.voltage</td>
<td>8 bits</td>
<td>Unsigned</td>
<td>0-5</td>
<td>0.0196V/bit</td>
<td>0</td>
<td>V</td>
</tr>
<tr>
<td>6</td>
<td>a7.voltage</td>
<td>8 bits</td>
<td>Unsigned</td>
<td>0-5</td>
<td>0.0196V/bit</td>
<td>0</td>
<td>V</td>
</tr>
<tr>
<td>7</td>
<td>a8.voltage</td>
<td>8 bits</td>
<td>Unsigned</td>
<td>0-5</td>
<td>0.0196V/bit</td>
<td>0</td>
<td>V</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>ID</th>
<th>ByteID + 3</th>
<th>Frequency: 62.5Hz</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>BaseID + 3</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Channel</td>
<td>Data Width</td>
</tr>
<tr>
<td>0</td>
<td>a9.voltage</td>
<td>8 bits</td>
</tr>
<tr>
<td>1</td>
<td>a10.voltage</td>
<td>8 bits</td>
</tr>
<tr>
<td>2</td>
<td>a11.voltage</td>
<td>8 bits</td>
</tr>
<tr>
<td>3</td>
<td>a12.voltage</td>
<td>8 bits</td>
</tr>
<tr>
<td>4</td>
<td>a13.voltage</td>
<td>8 bits</td>
</tr>
<tr>
<td>5</td>
<td>a14.voltage</td>
<td>8 bits</td>
</tr>
<tr>
<td>6</td>
<td>a15.voltage</td>
<td>8 bits</td>
</tr>
<tr>
<td>7</td>
<td>a16.voltage</td>
<td>8 bits</td>
</tr>
</tbody>
</table>
### Frequency: 62.5Hz

<table>
<thead>
<tr>
<th>ID</th>
<th>BaseID + 4</th>
<th>BaseID + 5</th>
<th>BaseID + 6</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>o1.current</td>
<td>o9.current</td>
<td>o1.current</td>
</tr>
<tr>
<td>1</td>
<td>o2.current</td>
<td>o10.current</td>
<td>o2.current</td>
</tr>
<tr>
<td>2</td>
<td>o3.current</td>
<td>o11.current</td>
<td>o3.current</td>
</tr>
<tr>
<td>3</td>
<td>o4.current</td>
<td>o12.current</td>
<td>o4.current</td>
</tr>
<tr>
<td>4</td>
<td>o5.current</td>
<td>o13.current</td>
<td>o5.current</td>
</tr>
<tr>
<td>5</td>
<td>o6.current</td>
<td>o14.current</td>
<td>o6.current</td>
</tr>
<tr>
<td>6</td>
<td>o7.current</td>
<td>o15.current</td>
<td>o7.current</td>
</tr>
<tr>
<td>7</td>
<td>o8.current</td>
<td>o16.current</td>
<td>o8.current</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>ByteID</th>
<th>Channel</th>
<th>Data Width</th>
<th>Data Type</th>
<th>Range</th>
<th>Resolution</th>
<th>Offset</th>
<th>Unit</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>o1.current</td>
<td>8 bits</td>
<td>Unsigned</td>
<td>0-63.75</td>
<td>0.25A/bit</td>
<td>0</td>
<td>A</td>
</tr>
<tr>
<td>1</td>
<td>o2.current</td>
<td>8 bits</td>
<td>Unsigned</td>
<td>0-63.75</td>
<td>0.25A/bit</td>
<td>0</td>
<td>A</td>
</tr>
<tr>
<td>2</td>
<td>o3.current</td>
<td>8 bits</td>
<td>Unsigned</td>
<td>0-63.75</td>
<td>0.25A/bit</td>
<td>0</td>
<td>A</td>
</tr>
<tr>
<td>3</td>
<td>o4.current</td>
<td>8 bits</td>
<td>Unsigned</td>
<td>0-63.75</td>
<td>0.25A/bit</td>
<td>0</td>
<td>A</td>
</tr>
<tr>
<td>4</td>
<td>o5.current</td>
<td>8 bits</td>
<td>Unsigned</td>
<td>0-63.75</td>
<td>0.25A/bit</td>
<td>0</td>
<td>A</td>
</tr>
<tr>
<td>5</td>
<td>o6.current</td>
<td>8 bits</td>
<td>Unsigned</td>
<td>0-63.75</td>
<td>0.25A/bit</td>
<td>0</td>
<td>A</td>
</tr>
<tr>
<td>6</td>
<td>o7.current</td>
<td>8 bits</td>
<td>Unsigned</td>
<td>0-63.75</td>
<td>0.25A/bit</td>
<td>0</td>
<td>A</td>
</tr>
<tr>
<td>7</td>
<td>o8.current</td>
<td>8 bits</td>
<td>Unsigned</td>
<td>0-63.75</td>
<td>0.25A/bit</td>
<td>0</td>
<td>A</td>
</tr>
<tr>
<td>ID</td>
<td>BaseID + 7</td>
<td>Frequency: 62.5Hz</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>----</td>
<td>------------</td>
<td>------------------</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>o9.voltage</td>
<td>8 bits Unsigned</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>o10.voltage</td>
<td>8 bits Unsigned</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>o11.voltage</td>
<td>8 bits Unsigned</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>o12.voltage</td>
<td>8 bits Unsigned</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>o13.voltage</td>
<td>8 bits Unsigned</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>o14.voltage</td>
<td>8 bits Unsigned</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>o15.voltage</td>
<td>8 bits Unsigned</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>o16.voltage</td>
<td>8 bits Unsigned</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>ByteID</th>
<th>Channel</th>
<th>Data Width</th>
<th>Data Type</th>
<th>Range</th>
<th>Resolution</th>
<th>Offset</th>
<th>Unit</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>o9.voltage</td>
<td>8 bits</td>
<td>Unsigned</td>
<td>0-16.19</td>
<td>0.063V/bit</td>
<td>0</td>
<td>V</td>
</tr>
<tr>
<td>1</td>
<td>o10.voltage</td>
<td>8 bits</td>
<td>Unsigned</td>
<td>0-16.19</td>
<td>0.063V/bit</td>
<td>0</td>
<td>V</td>
</tr>
<tr>
<td>2</td>
<td>o11.voltage</td>
<td>8 bits</td>
<td>Unsigned</td>
<td>0-16.19</td>
<td>0.063V/bit</td>
<td>0</td>
<td>V</td>
</tr>
<tr>
<td>3</td>
<td>o12.voltage</td>
<td>8 bits</td>
<td>Unsigned</td>
<td>0-16.19</td>
<td>0.063V/bit</td>
<td>0</td>
<td>V</td>
</tr>
<tr>
<td>4</td>
<td>o13.voltage</td>
<td>8 bits</td>
<td>Unsigned</td>
<td>0-16.19</td>
<td>0.063V/bit</td>
<td>0</td>
<td>V</td>
</tr>
<tr>
<td>5</td>
<td>o14.voltage</td>
<td>8 bits</td>
<td>Unsigned</td>
<td>0-16.19</td>
<td>0.063V/bit</td>
<td>0</td>
<td>V</td>
</tr>
<tr>
<td>6</td>
<td>o15.voltage</td>
<td>8 bits</td>
<td>Unsigned</td>
<td>0-16.19</td>
<td>0.063V/bit</td>
<td>0</td>
<td>V</td>
</tr>
<tr>
<td>7</td>
<td>o16.voltage</td>
<td>8 bits</td>
<td>Unsigned</td>
<td>0-16.19</td>
<td>0.063V/bit</td>
<td>0</td>
<td>V</td>
</tr>
</tbody>
</table>
## DOCUMENT REVISION HISTORY

<table>
<thead>
<tr>
<th>Revision</th>
<th>Date</th>
<th>Changes</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.00</td>
<td>1.05.2017</td>
<td>- initial release</td>
</tr>
<tr>
<td>1.01</td>
<td>10.07.2017</td>
<td>- added information about ground connection with USBtoCAN</td>
</tr>
<tr>
<td>1.02</td>
<td>4.06.2018</td>
<td>- clarified outputs specification</td>
</tr>
<tr>
<td>Output Number</td>
<td>Terminal</td>
<td>Max Current</td>
</tr>
<tr>
<td>---------------</td>
<td>----------</td>
<td>-------------</td>
</tr>
<tr>
<td>O1</td>
<td>38</td>
<td>25A</td>
</tr>
<tr>
<td>O2</td>
<td>39</td>
<td>25A</td>
</tr>
<tr>
<td>O3</td>
<td>26</td>
<td>25A</td>
</tr>
<tr>
<td>O4</td>
<td>13</td>
<td>25A</td>
</tr>
<tr>
<td>O5</td>
<td>12</td>
<td>25A</td>
</tr>
<tr>
<td>O6</td>
<td>11</td>
<td>15A</td>
</tr>
<tr>
<td>O7</td>
<td>10</td>
<td>15A</td>
</tr>
<tr>
<td>O8</td>
<td>9</td>
<td>15A</td>
</tr>
<tr>
<td>O9</td>
<td>5</td>
<td>15A</td>
</tr>
<tr>
<td>O10</td>
<td>4</td>
<td>15A</td>
</tr>
<tr>
<td>O11</td>
<td>3</td>
<td>15A</td>
</tr>
<tr>
<td>O12</td>
<td>2</td>
<td>25A</td>
</tr>
<tr>
<td>O13</td>
<td>1</td>
<td>25A</td>
</tr>
<tr>
<td>O14</td>
<td>14</td>
<td>25A</td>
</tr>
<tr>
<td>O15</td>
<td>27</td>
<td>25A</td>
</tr>
<tr>
<td>O16</td>
<td>28</td>
<td>25A</td>
</tr>
<tr>
<td>Analog Number</td>
<td>Terminal</td>
<td>Type</td>
</tr>
<tr>
<td>---------------</td>
<td>---------</td>
<td>------</td>
</tr>
<tr>
<td>A1</td>
<td>29</td>
<td></td>
</tr>
<tr>
<td>A2</td>
<td>16</td>
<td></td>
</tr>
<tr>
<td>A3</td>
<td>30</td>
<td></td>
</tr>
<tr>
<td>A4</td>
<td>17</td>
<td></td>
</tr>
<tr>
<td>A5</td>
<td>31</td>
<td></td>
</tr>
<tr>
<td>A6</td>
<td>18</td>
<td></td>
</tr>
<tr>
<td>A7</td>
<td>32</td>
<td></td>
</tr>
<tr>
<td>A8</td>
<td>19</td>
<td></td>
</tr>
<tr>
<td>A9</td>
<td>6</td>
<td></td>
</tr>
<tr>
<td>A10</td>
<td>33</td>
<td></td>
</tr>
<tr>
<td>A11</td>
<td>20</td>
<td></td>
</tr>
<tr>
<td>A12</td>
<td>34</td>
<td></td>
</tr>
<tr>
<td>A13</td>
<td>21</td>
<td></td>
</tr>
<tr>
<td>A14</td>
<td>8</td>
<td></td>
</tr>
<tr>
<td>A15</td>
<td>35</td>
<td></td>
</tr>
<tr>
<td>A16</td>
<td>22</td>
<td></td>
</tr>
</tbody>
</table>