VE.Direct Protocol FAQ

More information on the VE.Direct protocol is available here:

- Data communication whitepaper
- VE.Direct protocol document: look for VE.Direct on our whitepaper page
- Open source

FAQ

Can I use RS232 instead of USB?

Yes, use the *VE.Direct to RS232 interface*, part number ASS030520500.

When using that RS232 interface, what pins do I need?

For the communication use the GND, RX and TX pins: pin 5, 2 and 3 on the DB9 connector.

Also the DTR signal (pin 4 on the DB9 connector) and/or the RTS signal (pin 7 on the DB9 connector) must be driven high to power the isolated side of the the interface. How to program the DTR and RTS differs between used operating systems and hardware. Please note that most RS232 drivers are inverting so the logic level of the DTR must be programmed to zero in most cases. When one of those pins is not driven high, you will not be able to receive data!

Background: the VE.Direct to RS232 interface provides galvanic isolation between the VE.Direct product and the host (your computer/PLC/etc). The VE.Direct side of the PCB is powered from VE.Direct. And the RS232 side takes its power from the DTR and RTS pins.

How do I calculate the HEX checksum?

I have been able to get the data I need from the sensor (BMV-700H), as show in your examples (im quoting your last email below): Consider the following example:

```
Get Battery Capacity
:70010003E<LF>     -> Command; checksum 0x55 – 0x7 – 0x0 – 0x10 – 0x0 = 0x3E
:7001000C80076<LF> -> Response; checksum 0x55 – 0x7 – 0x0 – 0x10 – 0x0 –
0xC8 – 0x0 = 0x76
```

So, this works fine when the code of the operation is less than 55 (for instance, 0x10 and 0x00 = 0x1000), and I get the same checksums as you do. However, to ask for the SOC of the batteries, Im not being able to understand how to construct the request, since the code is 0x0FFF, which is greater than 0x55. Can you provide me with an example of a request, for the SOC of the batteries? I would be much appreciated.

Anwser: VE.Direct/VE.Hex data is encoded as little endian. The checksum needs to be a byte, therefore you need to wrap it while calculating the checksum.

So in code you can loop through the message as in the following pseudo code: byte checksum = 0x55; byte message[] = { 0x7, 0xff, 0x0f, 0x00 }; for (int i =0; i < sizeof(message); i++)

```
            checksum -= message[i];
```

Get command: :7<register id><flags><checksum>\n

Get Soc, register 0x0FFF :7FF0F0040\n → Checksum = 0x55 – 7 – 0xFF – 0xF – 0 = 0x40

DISQUS

~~DISQUS~~