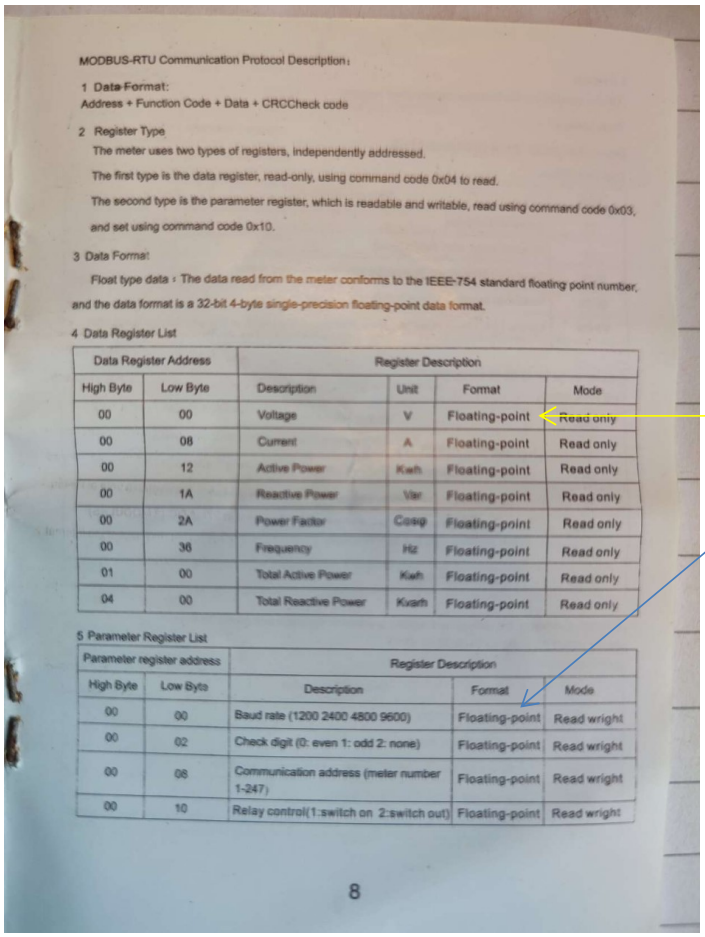
Programming a DDM18SD Power Meter to Read & Write

USE THIS DOCUMENT AT YOUR OWN RISK – I learn by trial & error

Requirements:

- 'Simple Modbus Master 8.1.2' Software on a PC – free trial use
- a USB to RS485 CH340 Modbus Adapter (cheap generics on eBay)
- Window's 'Device Manager' to identify the relevant USB port being used by the above adapter to setup the software / communications to your Modbus device
- Microsoft Store has a free float to hex calculator called 'Float to Hex' ! It also does hex to float … very helpful to understand this system.

From the DDM18SD manual:



MODBUS-RTU Communication Protocol Description:

1 Data Format:
Address + Function Code + Data + CRCCheck code

2 Register Type
The meter uses two types of registers, independently addressed.
The first type is the data register, read-only, using command code 0x04 to read.
The second type is the parameter register, which is readable and writable, read using command code 0x03, and set using command code 0x10.

3 Data Format
Float type data : The data read from the meter conforms to the IEEE-754 standard floating point number, and the data format is a 32-bit 4-byte single-precision floating-point data format.

4 Data Register List

Data Register Address		Register Description			
High Byte	Low Byte	Description	Unit	Format	Mode
00	00	Voltage	V	Floating-point	Read only
00	08	Current	A	Floating-point	Read only
00	12	Active Power	Kwh	Floating-point	Read only
00	1A	Reactive Power	Var	Floating-point	Read only
00	2A	Power Factor	Cosφ	Floating-point	Read only
00	36	Frequency	Hz	Floating-point	Read only
01	00	Total Active Power	Kwh	Floating-point	Read only
04	00	Total Reactive Power	Kvarh	Floating-point	Read only

5 Parameter Register List

Parameter register address		Register Description		
High Byte	Low Byte	Description	Format	Mode
00	00	Baud rate (1200 2400 4800 9600)	Floating-point	Read wright
00	02	Check digit (0: even 1: odd 2: none)	Floating-point	Read wright
00	08	Communication address (meter number 1-247)	Floating-point	Read wright
00	10	Relay control(1:switch on 2:switch out)	Floating-point	Read wright

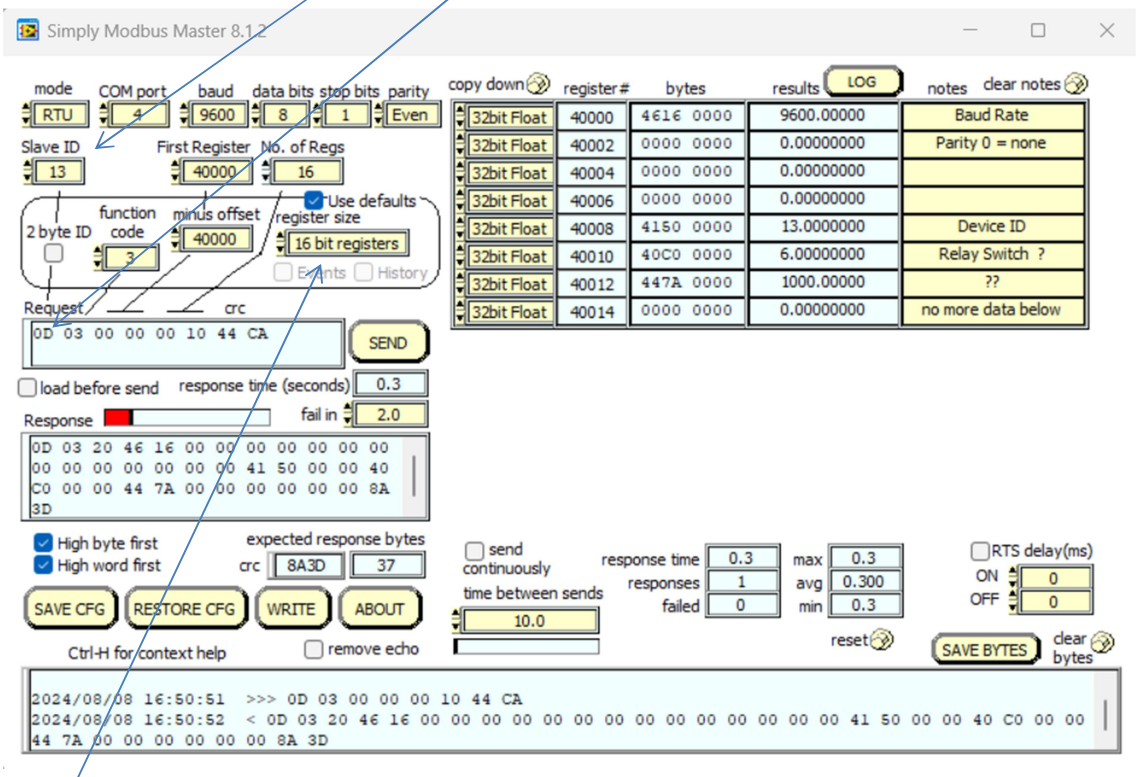
8

Note: 32 Bit Floating Point Data/Parameters are converted to Hex in the Software for transmission: examples below

NB The 'Simple Modbus Master 8.1.2' software tends to have to be given time to respond to inputs eg the 'Send Command" and sometimes the result windows (light blue background) need to be clicked on to cause an update ☺

Simple Data Reading The Code ‘3’ command

This is a data reading of the DDM18SD with its default ‘Even” parity. (This can also be changed see below). The Device has ID: ‘13’ (ie ID: ‘0D’ in hex as converted by the software)



Notes:

Here only 16 registers have been requested, up to 50 can be selected with this device/software.

The yellow boxes are the user input boxes. Some will need changing to achieve the results above. Many Modbus devices **do not use 32 bit Floating (decimal) Point** values so other combinations will be required for 16 bit data formats types. The advantage of this software is that you can see what codes are being sent (in hex) before sending. This command data is often given by device manufactures for their various relays so you can be confident knowing that you have got it correct before sending if you have this data and it matches the Request box above before sending.

The crc correction code (last 2 digits in the command instruction) is provided by this software and can be ignored or used to compare with externally supplied commands when available

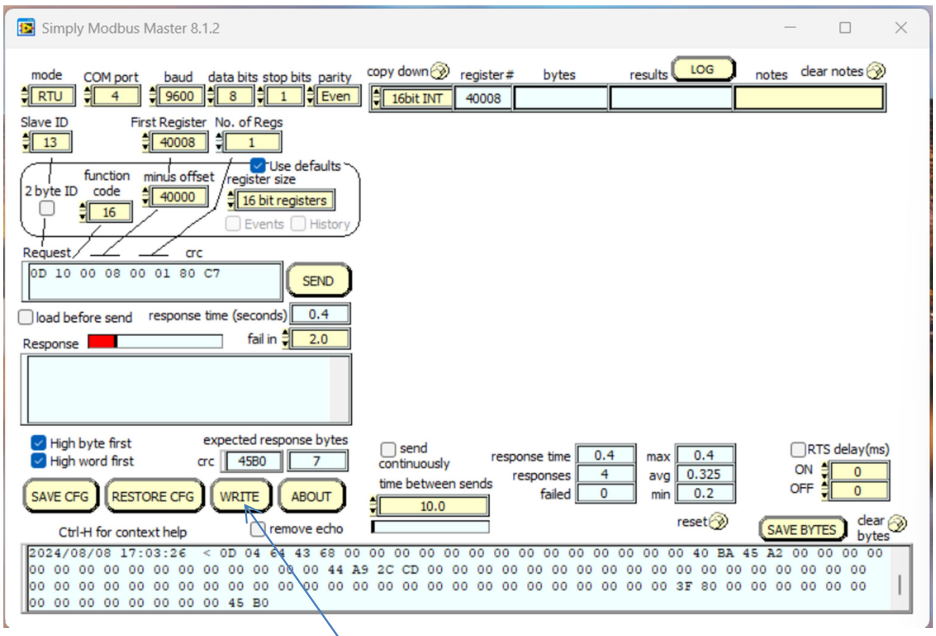
It can be somewhat confusing (to newbies) that decimal integers up of the value of 9 are identical to their hex equivalent, but as far as I know every time you see these command instructions, they will always be in hex.

In this example the Parity “Even” box needed to be tapped to update the display to “none” … I only noticed this later and this reading was taken just after I changed the Parity to ‘none’ as seen later in the documentation. I wanted a more complete data set than I initially had for this Reading example.

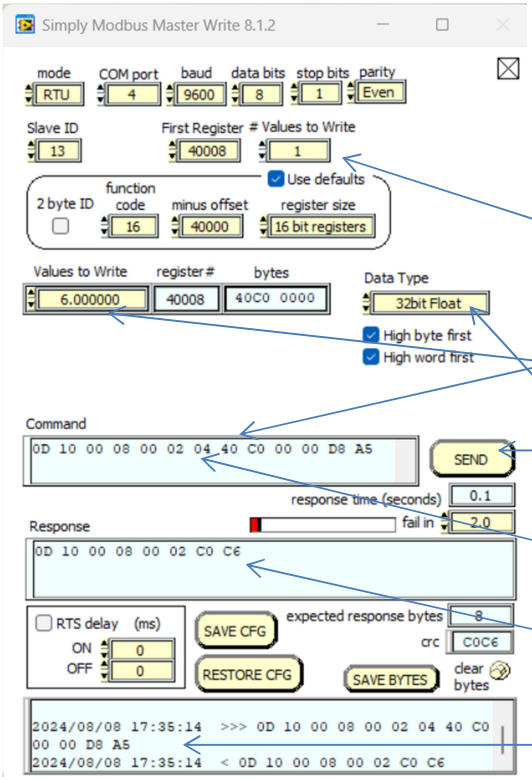
Now the next setup will change our new Device 13 to ID 6..

Set Device ID 13 to ID 06 on the even parity default of the DDM18SD device

Step 1



Step 2 Select ‘Write’ to get next Popup Screen to finish the setup a ‘Write’ Parameter!



Only one register write needed for Device ID

New ID 32 Bit Floating No. ‘6.000000’ then changed to hexadecimal in Command Box ie now shown as ‘40 C0’ hex

Note: Only press when convinced all settings are correct! (may need a ‘long’ press)

32 Bits requires 04 bytes (software inserts this value) when 32Bit Float data type selected

Expected Response (all in hex)

Response after a successful write ID change

NB New ID takes effect after rebooting the device

Another example of Device ID Change 01 -> 02

(3)Write of the second type register (parameter register)

Modify meter address:

Deliver Data (HEX): 01 10 00 08 00 02 04 40 00 00 00 E7 C9 （Change the meter address to 02）

Data Description:

Data	Detailed Description
01	Instrument address
10	Function code, write meter internal register data
00 08	Write data starting from the 00 08 register address inside the meter
00 02	Number of registers, 2 (4 bytes)
04	Bytes, 4 bytes
40 00 00 00	The meter address of the written meter, 4 bytes of data, floating-point data
E7 C9	CRC check

Return: 01 10 00 08 00 02 C0 0A

Indicates that the setting is successful.

NOTE 40 00 00 00 hex = 02 Floating decimal point ie decimal number with floating decimal point

i.e. 02 in Floating Point format is 40 00 in hex (0x40 00 00 00) but using only one register and written in the simplified form by the software

NOTE: This process can also be used to fix a ‘damaged’ meter with an address of ‘00’ to give it a legitimate ID’.

WARNING all Modbus devices on the network will respond to commands addressed to ‘00’ … make sure you do not do this process with more than one device on the network or they will all get the new ID address ☺!

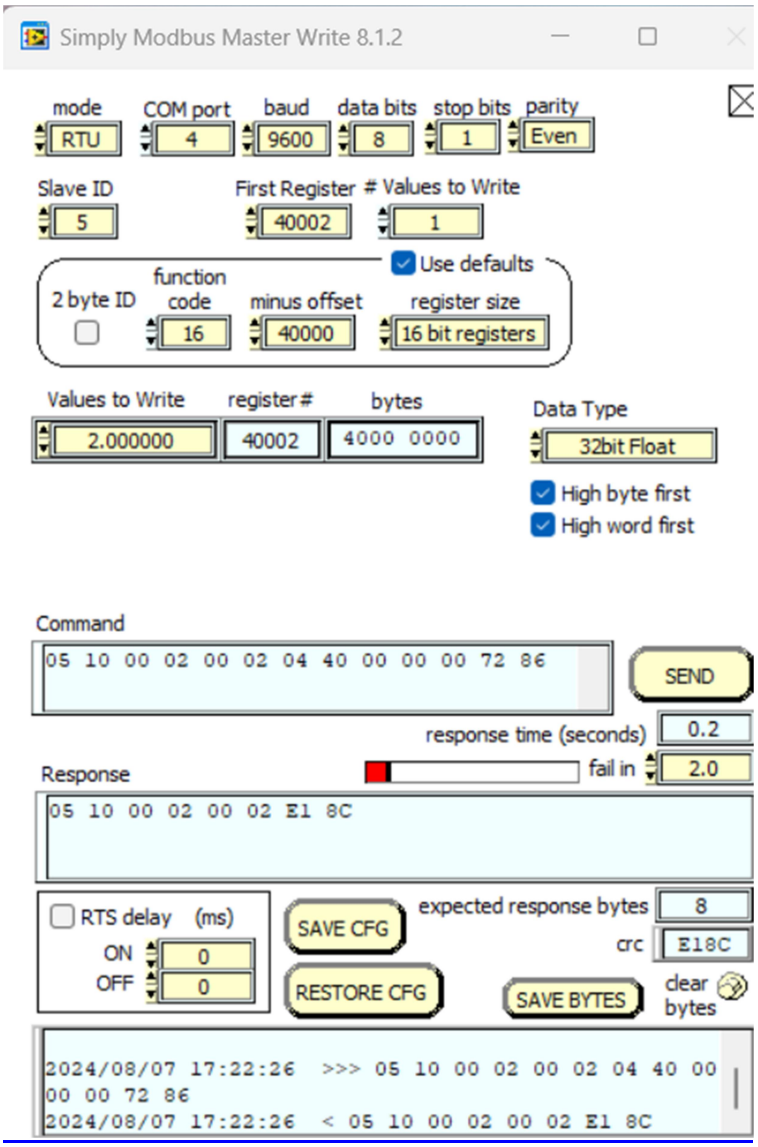
Reference: This example taken from:

https://www.re-innovation.co.uk/docs/talking-to-a-ddm18sd-energy-meter/#comment-2428

More wiring and useful background data is found here.

Up next, changing the device’s Parity setting….

Another Parameter change may be needed to change the Parity of the DDM18SD meter from its default ‘Even” parity to the more common ‘None’ here we are changing the parity on a device with ID ‘05’



This was done successfully. It may need the device to be rebooted to implement the change.

The Value for the Parity options is shown on page 1 of this document. Value ‘2’ sets the device to ‘No parity’ as shown here.

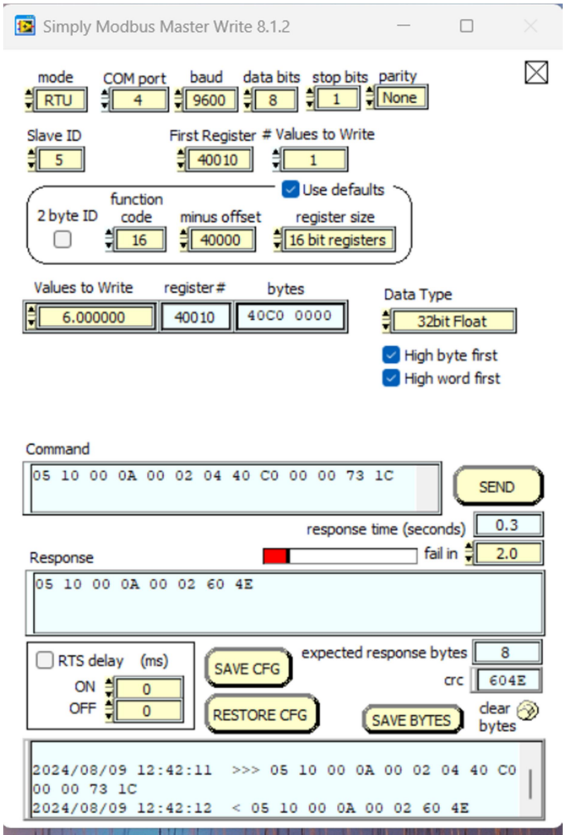
Similarly the Baud rate could be changed using this software and the data on page 1.

Next… Repairing a damaged register 10 in a DDM18SD power meter

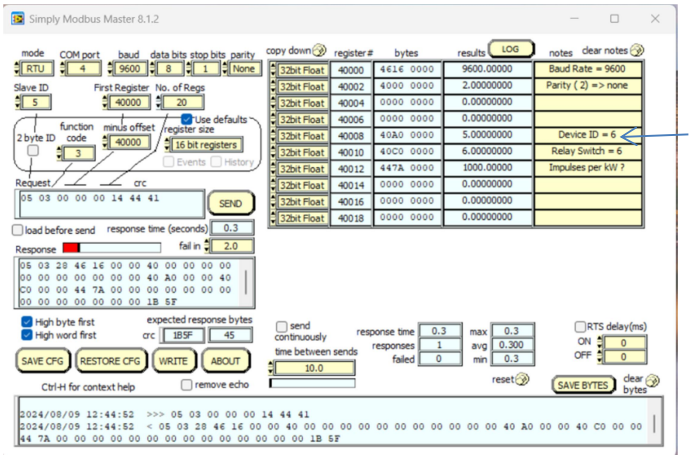
Repair damaged register No. 10 in Device (05) which is reading 0. The LCD display has stopped cycling due to this (programming induced error).. I had to learn somehow ☺

NB The ‘Relay Switch’ register reads as 6 in a healthy relay thus…

Write 6 (Float 32) to Register 10 of Device 05… see below



Success in writing, the 05 meter now has identical setup parameters to a healthy DDM18SD meter



Note the ID is shown incorrectly here

I forgot to up-date the Notes which are written in manually.. the result column to the left is correct.

See over …

The meter 05's LCD display is now cycling its readings again! And so I fully restored the DDM18SD kWhr meter which I 'stuffed' when playing around with different software packages along with my 'poke and see' attempts at conquering this relay. Yes, I am self-taught ☺

Note: Some results are well down in the register (see the first page above). eg the 'Total Reactive Power' is read at the register addressed at '40' high byte with '00' low byte ie 100 registers down with zero offset. This free software cannot read all of the data registers in one call. That said, my RaspberryPi Codesys PLC software can read all the data off the DDM18SD device in one hit, but I don't particularly need it all!

Bernie_BBQ