Changing settings & calibration on-line

At first startup, the default settings for configuration (whether to use the radio, pulse counting, temperature sensors etc) and calibration (adjustments to improve accuracy) are used.

As soon as the unit has started up, the user has the opportunity to change those settings. To enter the settings mode, using the serial monitor part of the Arduino IDE (or almost any terminal emulator application), enter “**+++**” followed by the [Enter] key.

You will then see a menu:

```
Available commands:
  l        - list the settings
  r        - restore sketch defaults
  s        - save settings to EEPROM
  v        - show firmware version
  x        - exit to lock and continue
  ?        - show this text again

  w<x>     - turn RFM Wireless data off: x = 0 or on: x = 1
  b<n>     - set r.f. band n = a single numeral: 4 = 433MHz, 8 = 868MHz, 9 = 915MHz (may
                 require hardware change)
  p<nn>    - set the r.f. power. nn - an integer 0 - 31 representing -18 dBm to +13 dBm.
                 Default: 25 (+7 dBm)
  g<nnn>   - set Network Group  nnn - an integer (OEM default = 210)
  n<nn>    - set node ID  n= an integer (standard node ids are 1..60)

  d<xx.x>  - xx.x = a floating point number for the datalogging period
  c<n>     - n = 0 for OFF, n = 1 for ON, enable voltage, current & power factor values
                 to serial output for calibration.
  f<xx>    - xx = the line frequency in Hz: normally either 50 or 60
  k<x> <yy.y> <zz.z>
           - Calibrate an analogue input channel:
           - x = a single numeral: 0 = voltage calibration, 1 = ct1 calibration, 2 = ct2
                 calibration, etc
           - yy.y = a floating point number for the voltage/current calibration constant
           - zz.z = a floating point number for the phase calibration for this c.t. (z is
                 not needed, or ignored if supplied, when x = 0)
           -  e.g. k0 256.8
           -       k1 90.9 2.00
  a<xx.x>  - xx.x = a floating point number for the assumed voltage if no a.c. is
                 detected
  m<x> <yy> - meter pulse counting:
             x = 0 for OFF, x = 1 for ON, <yy> = an integer for the pulse minimum period
                 in ms. (y is not needed, or ignored when x = 0)
  t0 <y>   - turn temperature measurement on or off:
           -   y = 0 for OFF, y = 1 for ON
  t<x> <yy> <yy> <yy> <yy> <yy> <yy> <yy> <yy>
           - change a temperature sensor's address or position:
           - x = a single numeral: the position of the sensor in the list (1-based)
           - yy = 8 hexadecimal bytes representing the sensor's address
               e.g.  28 81 43 31 07 00 00 D9
               N.B. Sensors CANNOT be added.
```

Note: The sketch might not make use of all of the options listed here. The on-screen menu will list those available.

On-line settings are disabled by default. They must be turned on by entering the access code described above, and you are strongly advised to turn off as soon as practical, so that the risk of accidentally changing a setting is removed. You turn off settings mode and lock access when you exit and continue (option ‘x’).

If the serial output is in use, the settings mode is of course not available unless the commands come through that serial device (e.g. a Raspberry Pi).

If you change one or more of the settings, the command will be acknowledged and the change will take effect immediately, but you might not see the full effect until the set of readings after the next appears.

Take care that the correct RF frequency is selected to match your hardware. Operating the transmitter at high power on the wrong frequency, or without an effective antenna, can destroy the RFM module.

Option (‘s’) will save all the changes. If you do not do this, the settings will revert to the previous values at the next restart. After you save (‘s’) the changes, the new settings will be used forever, or until changed again.

If you restore the sketch default values (‘r’), the sketch restarts immediately using the values set in the sketch, and all the EEPROM data is ignored. There is then no means of recovering the EEPROM data.

If the EEPROM has been used previously and has had non-compliant values written, the EEPROM content will be ignored and the sketch will start using its own default values. Saving the configuration will format the EEPROM and the sketch’s set values will be stored.

Note that Data Whitening is now done entirely within the RFM69CW, and must not be enabled in emonHub.