VAN MIERLO INGENIEURSBUREAU
Intelligente elektronica voor meet- en regeltechniek

Postbus 6466
5600HL Eindhoven

+31-40-2969999
www.vanmierlo.com

AEMON bridge data interface

author: Bert van Kraaij
project code: VMI-AEMON
document number: BK19.343-A
date: 2020-03-27

1. Scope

The bridge used in the Aemon project logs data into a database on a portal hosted by Van Mierlo. This is requirement because it must be possible to access the gathered data externally for research purposes. To be able to view and process the data locally, a second bridge type is built. This document defines the interface for this device.

2. Hardware interface

The bridge has a UART connection to a PC. For this interface, a smart USB cable from FTDI can be used. When using the cable type TTL-232RG-VREG3V3-WE it can also supply power to the bridge. Note that this cable incorporates a 3V3 regulator to supply power to the bridge device. Any other cable can be used, but in that case the dridge power must come from a different source (e.g. a battery pack). The USB interface shows up as a serial port and can be accessed on 115200 baud, 8 databits, no parity and 1 stopbit.

The cable connection to the MyriaModenm is as follows:

Black	GND	connect to battery gnd or MyriaModem pin 1
Red	VCC	connect to battery vcc or MyriaModem pin 2
Yellow	RXD	connect to MyriaModem TXD, pin 6
Orange	TXD	connect to MyriaModem RXD, pin 7
Brown	CTS	do not connect
Green	RTS	do not connect

3. Software interface

The interface is text-based. Each line holds data from all valid sensors of a wireless node. Sample data lines are shown below.

```
D 15RR 0000 295a 26 3c5e327e 25 40ff740c 24 3f2621d8 1a 443afe9b cb
D 15RM 2a08 2a06 26 3f72bd4e 25 3f438c79 24 41f38b80 1a 444cec8d 67
D 15RR 2a4c 2a4b 26 3c5f30e9 25 40ff5450 24 3f246158 1a 44394752 45
D 15RM 2af9 2af7 26 3f80a33a 25 3f3a5ac5 24 41f1e36a 1a 4453590c e4
D 15RR 2b3d 2b3c 26 3c5e8746 25 40fe37f0 24 3f233789 1a 443d755f 88
D 15RM 2be9 2be8 26 3f8171da 25 3f41441f 24 41eba726 1a 445331a0 05
D 15RR 2c2e 2c2d 26 3c63d6f5 25 40fc064e 24 3f27e477 1a 443b9108 d0
D 15RM 2cdb 2cda 26 3f88cf36 25 3f4270bb 24 41e50240 1a 44571c18 fd
```

Each line has the following structure::

- the first character (always the character 'D', others can be ignored);
- the sensor ID. In this case there are two sensors generating data: 15RM and 15RR. The value is always alphanumerical, in capitals only;
- two time stamps. This enables a receiver to determine the exact time of measurement, independent from the flight time of the message in the network. These figures are two 16-bit hex numbers. See 3.1 for details;

- the sensor data. The sensor data starts with an 8-bit hex number defining the sensor type. See 3.2 for a list of all sensor types in use within the Aemon project. The following number is a 32 bit float value with the actual sensor data.
- more sensor data. The sample lines shown above hold data for four sensors, with sensor type 0x26, 0x25 0x24 and 0x1a.
- a crc8 calculated over the data (see 3.3)
- an end-of-line sequence \r\n

Note that the number of sensors on a node can vary which makes the line longer or shorter.

3.1. Calculating the time of measurement

To calculate the "exact" time of measurement, each message is sent with two time stamps (16-bit hex numbers). The first one is the actual network time. The second one is the network time at the moment of the measurement. The increments are 500ms tics. At the end a rollover will occur from 0xffff to 0x0001. The value 0x0000 is invalid and indicates that the network time is not (yet) defined.

3.2. Aemon sensor types

The following sensor types are defined for the Aemon project:

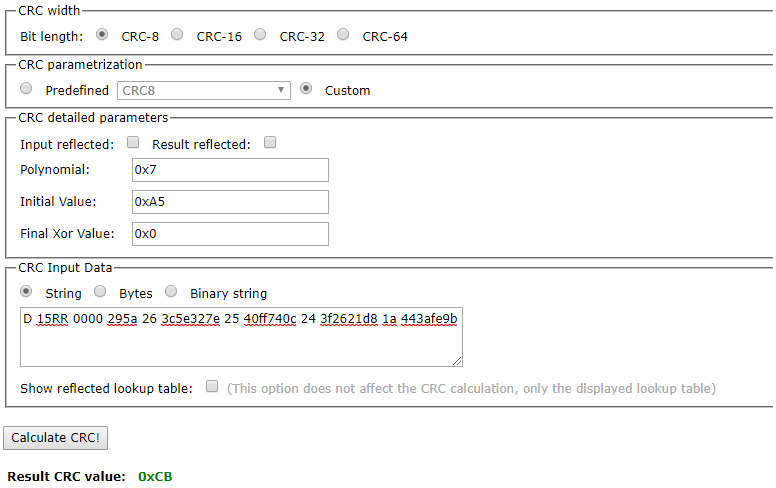
- TEMPERATURE = 0x01, C
- HUMIDITY = 0x02, 0…100%
- LIGHT_INTENSITY = 0x1a, lux
- NH3_PPM = 0x24, ppm
- NO2_PPM = 0x25, ppm
- CO_PPM = 0x26, ppm
- CO2_PPM = 0x27, ppm
- BAROMETRIC_PRESSURE = 0x28, mbar
- NH3_PPM_REF = 0x29 ppm

Note that there are two possible NH3 sensor types: a reference sensor `NH3_PPM_REF` (Draeger) generating an absolute NH3 concentration (0…100 ppm as 0…10V) and a sensor `NH3_PPM` (MiCS6814), that shows a more relative NH3 change.

3.3. Data verification (crc8)

To be able to check the transmission a CRC8 is calculated over the complete data line. The calculation starts with the character 'D' and ends with the whitespace before the two CRC characters.

The crc8 calculation uses 0xA5 as an initial value and 0x07 as polynomial. There is no additional data operation on the output. An example of a CRC crc calculator can be found on http://www.sunshine2k.de/coding/javascript/crc/crc_js.html. A screenshot with the calculation on the first dataline is shown below.


CRC width
Bit length: CRC-8 CRC-16 CRC-32 CRC-64
CRC parametrization
Predefined CRC8 Custom
CRC detailed parameters
Input reflected: Result reflected:
Polynomial: 0x7
Initial Value: 0xA5
Final Xor Value: 0x0
CRC Input Data
String Bytes Binary string
D 15RR 0000 295a 26 3c5e327e 25 40ff740c 24 3f2621d8 1a 443afe9b
Show reflected lookup table: (This option does not affect the CRC calculation, only the displayed lookup table)
Calculate CRC!
Result CRC value: 0xCB