# *Type ML-x17*
# Low power remote data loggers

| | | | | | |
|---|---|---|---|---|---|
| **Title** | : | User Manual ML-x17 | **Date** | : | June-2025 |
| **Version** | : | FW 5.1B1 | | | |
| **Manufacturer** | : | Your Data Our Care B.V. | | | |
| **Address** | : | Da Vincilaan 13c<br>6716WC, EDE<br>The Netherlands | | | |

# WARNING

THE FOLLOWING OPERATING INSTRUCTIONS ARE FOR USE BY QUALIFIED PERSONNEL ONLY. TO AVOID DAMAGE OR MALFUNCTION, DO NOT PERFORM ANY OPERATING OTHER THAN THAT CONTAINED IN THIS MANUAL. ANY OPERATOR SHOULD BE SKILLED WITH A TECHNCAL BACKGROUND BEFORE OPERATING THE DEVICE.

# PREFACE

Congratulations!

With your purchase of an YDOC Low Power data logger with mobile network capabilities.

This manual describes the operation and (hardware) installation of the ML-x17 data logger.

The chapter *Getting Started* briefly describes the data logger, prepares you to install it, and tells you how to put it into operation.

The Chapter *Operating Basics* covers basic principles of operation of the data logger. The operating interface (menu) and the tutorial examples, rapidly help you to understand how your data logger operates.

The Chapter *Reference* teaches you how to perform specific tasks and provides a complete list of operating tasks and useful background information.

The Appendices provide a list with all available options, and other useful information.

We recommend reading this manual carefully before installation of the data logger.

**Warranty**

All YDOC instruments are warranted for 3 years against defective materials and workmanship. Any questions with respect to the warranty mentioned above should be taken up with your YDOC Distributor.

***Table of Contents***

1. Product description ..... 12
1.1 Models and Editions ..... 13
2 Getting started ..... 15
2.1 Vibration ..... 15
2.2 Do's & Don'ts ..... 15
2.3 Inserting the SIM-card ..... 15
2.4 Power on for the First time ..... 16
2.5 Connect to a PC, tablet or smart phone ..... 16
3 Operating Basics ..... 17
3.1 Configuration menu ..... 17
4 Reference ..... 23
4.1 Principle of Operation ..... 23
4.1.1 Intervals ..... 23
4.1.1.1 Sample interval ..... 23
4.1.1.2 Data logging interval ..... 23
4.1.1.3 Independent data logging interval ..... 23
4.1.1.4 Send Interval ..... 23
4.1.1.5 Example ..... 24
4.1.2 Common data output settings ..... 24
4.1.2.1 Send delay ..... 24
4.1.2.2 Output type ..... 24
4.1.2.3 Data format ..... 25
4.1.2.4 Data filter ..... 25
4.1.2.5 Max payload ..... 25
4.2 Parameters ..... 26
4.2.1 Name ..... 26
4.2.2 Code ..... 26
4.2.3 Unit ..... 26
4.2.4 Initial value ..... 26
4.2.5 Value factor ..... 26
4.2.6 Value offset ..... 26
4.2.7 Decimals ..... 27
4.2.8 Data log ..... 27
4.2.9 Data output ..... 27
4.2.10 Boundary control ..... 27
4.2.11 Alarm message ..... 27
4.2.12 Alarm log ..... 27
4.2.13 Alarm output ..... 27
4.2.14 Alarm limits and delays ..... 27
4.3 Configuration Menu Settings ..... 28
4.3.1 Date & Time ..... 28
4.3.2 System Information ..... 28
4.4 Configuration Setup ..... 29
4.5 General Settings ..... 30
4.5.1 Exit ..... 30
4.5.2 System Name ..... 30
4.5.3 Data log interval ..... 30
4.5.3.1 Normal mode ..... 30
4.5.3.2 Alarm mode ..... 30
4.5.4 Timestamp round down ..... 30
4.5.5 Avoid equal timestamps ..... 30
4.5.6 Power supply board ..... 30
4.5.7 Battery (protection & lifespan) ..... 30
4.5.7.1 Modem protection ..... 31
4.5.8 Deployment Date and Time ..... 31

4.5.9 Daily operating time slot ..... 31
4.5.9.1 Daily schedule time shift ..... 31
4.5.10 Time zone ..... 31
4.5.11 Summer time ..... 32
4.6 Modem Settings ..... 32
4.6.1 Network ..... 32
4.6.2 Technology ..... 32
4.6.3 PLMN code ..... 32
4.6.4 Provider Selection ..... 32
4.6.5 APN settings ..... 33
4.6.6 Internet protocol ..... 33
4.6.7 Customization ..... 33
4.6.8 Standby threshold time ..... 33
4.6.9 Network signal test ..... 34
4.6.10 APN login test ..... 34
4.7 NTP time update ..... 34
4.7.1 Time Update ..... 34
4.7.2 Update interval ..... 34
4.7.3 Server ..... 34
4.7.4 NTP Port ..... 34
4.7.5 Time update test ..... 34
4.8 Alarm settings ..... 34
4.8.1 Continuous Alarm Sampling ..... 34
4.8.2 Instant measurement on data alarm ..... 35
4.8.3 Exclude independent drivers ..... 35
4.8.4 Instant data output on data alarm ..... 35
4.8.5 Alarm messages ..... 35
4.8.5.1 System alarm messages ..... 35
4.8.5.2 Data alarm messages ..... 36
4.8.6 Alarm output port ..... 36
4.8.7 Alarm output port active ..... 36
4.8.8 Alarm parameters ..... 36
4.8.8.1 Register value & reset ..... 37
4.8.8.2 Alarm count ..... 37
4.8.8.3 Alarm state ..... 37
4.8.8.4 Total on-time ..... 37
4.8.8.5 Total Interval on-time ..... 37
4.9 Option boards ..... 38
4.9.1 ML-OI-AD-10V/20MA/80MV/2000MV (analog inputs) ..... 38
4.9.1.1 Calibration ..... 38
4.9.1.2 Zero-calibration ..... 38
4.9.2 ML-OI-AD-PT1000 ..... 39
4.9.2.1 Calibration ..... 39
4.9.3 ML-OI-AX (Auxiliary inputs) ..... 39
4.9.4 ML-OO (Auxiliary outputs) ..... 39
4.9.4.1 Output interval ..... 39
4.10 Internal Sensors ..... 40
4.10.1 Name ..... 40
4.10.2 Sample interval ..... 40
4.10.3 Battery Capacity ..... 40
4.10.4 Battery Replaced ..... 40
4.10.5 Rest Capacity ..... 40
4.10.6 Rest power ..... 40
4.10.7 Processor Temperature ..... 40
4.10.8 Average Voltage ..... 40
4.10.9 Max Voltage ..... 40
4.10.10 Min Voltage ..... 40

4.10.11 Average, Max, Min current ..... 41
4.10.12 Operating cycle ..... 41
4.10.13 Free disk space ..... 41
4.11 Analog inputs ..... 42
4.11.1 Sensor power switch ..... 42
4.11.2 Sample interval ..... 42
4.11.3 Port mode ..... 42
4.11.4 Parameter settings ..... 42
4.11.5 Parameter value at (min & max range) ..... 42
4.11.6 Determine linear conversion function (2 calibration points) ..... 42
4.11.7 Determine linear offset only (1 calibration point) ..... 42
4.11.8 Burst-mode ..... 43
4.12 Digital inputs – Pulse counter ..... 44
4.12.1 Sample interval ..... 44
4.12.2 Port mode ..... 44
4.12.2.1 Pull up ..... 45
4.12.2.2 Pull down ..... 45
4.12.3 Register mode ..... 45
4.12.4 Anti-bounce filter ..... 45
4.12.5 Units per pulse, or pulses per unit ..... 45
4.12.6 Register value ..... 45
4.12.7 Register Reset ..... 45
4.12.8 Log each counter change ..... 45
4.12.9 Counter ..... 45
4.12.10 Quantity ..... 46
4.12.11 Mean-, Max- and Min Rate ..... 46
4.12.12 Pulse event ..... 46
4.12.13 Activity period ..... 46
4.13 Digital inputs - Alarm trigger ..... 47
4.13.1 Trigger delay ..... 47
4.13.2 Register value ..... 47
4.14 Digital inputs – Action trigger ..... 48
4.14.1 Trigger on ..... 48
4.14.2 Trigger action ..... 48
4.15 Digital inputs - State input / On-time meter ..... 49
4.15.1 Total On-time ..... 49
4.15.2 Register value ..... 49
4.15.3 Register reset ..... 49
4.15.4 Interval On-time ..... 49
4.15.5 Input state ..... 49
4.16 Network Signal Sensor ..... 50
4.16.1 Independent data log ..... 50
4.16.2 Signal bars ..... 50
4.16.3 Signal dB/%/indication/bit error rate ..... 50
4.17 Aggregation Channels ..... 51
4.17.1.1 Average, minimum, maximum, gust, standard deviation & change ..... 52
4.17.1.2 Percentiles ..... 52
4.18 Calculation Channels ..... 52
4.18.1 Syntax ..... 53
4.18.2 Attributes ..... 53
4.18.2.1 Alarm limits ..... 53
4.18.2.2 Initial value ..... 53
4.18.2.3 Age ..... 53
4.18.2.4 Range ..... 53
4.18.3 Functions ..... 53
4.18.4 Example 1 ..... 53
4.18.5 Example 2 ..... 54

4.18.6 Supported mathematical functions with one argument: 54
4.18.6.1 abs(x) 54
4.18.6.2 sqrt(x) 54
4.18.6.3 ln(x) 54
4.18.6.4 exp(x) 54
4.18.6.5 log(x) 54
4.18.6.6 sin(radians) 55
4.18.6.7 cos(radians) 55
4.18.6.8 tan(radians) 55
4.18.6.9 asin(x) 55
4.18.6.10 acos(x) 55
4.18.6.11 atan(x) 55
4.18.6.12 torad(degrees) 55
4.18.6.13 todeg(radians) 55
4.18.6.14 floor(x) 55
4.18.6.15 ceil(x) 55
4.18.6.16 round(x) 55
4.18.7 Supported mathematical functions with multiple arguments: 55
4.18.7.1 atan2(x;y) 55
4.18.7.2 mod(x;y) 55
4.18.7.3 pow(x;y) 55
4.18.7.4 clip(x;min;max) 55
4.18.8 Supported comparisons with 2 arguments: 56
4.18.8.1 lt(x;y) 56
4.18.8.2 le(x;y) 56
4.18.8.3 gt(x;y) 56
4.18.8.4 ge(x;y) 56
4.18.9 Supported comparisons with 4 arguments: 56
4.18.9.1 eq(x;y;q;p) 56
4.18.9.2 lt(x;y;q;p) 56
4.18.9.3 le(x;y;q;p) 56
4.18.9.4 gt(x;y;q;p) 56
4.18.9.5 ge(x;y;q;p) 56
4.18.10 Supported bitwise functions with 2 arguments 56
4.18.10.1 and(x;y) 56
4.18.10.2 or(x;y) 57
4.18.10.3 xor(x;y) 57
4.18.11 Supported empirical formulas 57
4.18.11.1 ntc(Vin[;R0[;Rd[;B[;D]]]]) 57
4.18.11.2 wc(t;v) 58
4.18.11.3 dp(t;rh) 58
4.18.12 Various supportive functions: 58
4.18.12.1 pi 58
4.18.12.2 time 58
4.19 Serial Port 59
4.19.1 RS232 59
4.19.2 Generic Modbus 59
4.19.2.1 Port Settings 59
4.19.2.2 Protocol type 59
4.19.2.3 Register type 60
4.19.2.4 Register Start address 60
4.19.2.5 Data Type 60
4.19.2.6 Parameters 60
4.19.2.7 Timing 60
4.19.2.8 Measurement commands 60
4.19.2.9 Start maintenance command 60
4.19.2.10 Maintenance interval 61

4.19.2.11 Maintenance ready status ..... 61
4.19.2.12 Maintenance ready after ..... 61
4.19.2.13 Start measurement command ..... 61
4.19.2.14 Measurement ready status ..... 61
4.19.2.15 Measurement ready after ..... 61
4.19.3 Generic ASCII ..... 62
4.19.3.1 Log raw data String ..... 62
4.19.3.2 Decimal symbol ..... 62
4.19.3.3 Separator Character ..... 62
4.19.3.4 Start/Stop character ..... 62
4.19.3.5 Output request ..... 62
4.19.3.6 Parameters ..... 63
4.19.3.7 Sentence filter ..... 63
4.19.3.8 Field position ..... 63
4.19.3.9 Field type ..... 63
4.19.3.10 Example: ..... 63
4.19.4 Generic NMEA ..... 63
4.19.4.1 GPS ..... 64
4.19.5 RS485 ..... 64
4.19.6 SDI-12 ..... 64
4.19.6.1 Measurement command ..... 64
4.19.6.2 Parameters ..... 64
4.19.7 Serial port tunnel ..... 65
4.19.7.1 Port settings ..... 65
4.19.7.2 Sensor power ..... 65
4.20 Accessory port ..... 66
4.20.1 ASCII output ..... 66
4.20.1.1 Send interval ..... 66
4.20.2 Radio output ..... 67
4.20.2.1 Accessory power ..... 67
4.20.2.2 Send interval ..... 67
4.20.2.3 Send delay ..... 67
4.20.2.4 Power down delay ..... 67
4.20.2.5 Repeat previous messages ..... 67
4.20.2.6 Output format ..... 67
4.20.2.7 NMEA-0183 output ..... 68
4.20.3 Iridium Satellite ..... 69
4.20.3.1 Iridium SBD Service ..... 69
4.20.3.2 Settings menu ..... 70
4.20.3.3 Send interval ..... 70
4.20.3.4 Data transpond ..... 70
4.20.3.5 Backup mode ..... 70
4.20.3.6 Message format ..... 70
4.20.4 GPS ..... 71
4.20.4.1 Minimum satellites to use ..... 71
4.20.4.2 Time to fix ..... 71
4.20.4.3 Position drift alarming ..... 71
4.20.4.4 GPS time update ..... 72
4.20.4.4.1 Calculate alarm limits on deployment ..... 72
4.20.4.4.2 Latitude/Longitude hi/lo alarm drift ..... 72
4.20.4.5 Satellites ..... 72
4.20.4.6 Latitude / Longitude / Altitude ..... 72
4.20.4.7 GPS Quality ..... 72
4.20.5 Camera ..... 73
4.20.5.1 Picture taking ..... 73
4.20.5.2 Interval shift ..... 73
4.20.5.3 Daily operating time slot ..... 73

4.20.6 Display ..... 74
4.20.6.1 Power down delay ..... 74
4.20.1 Accessory port tunnel ..... 75
4.20.1.1 Port settings ..... 75
4.20.1.2 Sensor power ..... 75
4.21 Modem output - Email ..... 76
4.21.1 Server ..... 76
4.21.2 SMTP port ..... 76
4.21.3 Security ..... 76
4.21.4 Originator address ..... 76
4.21.5 Destination address ..... 76
4.21.1 Output type ..... 77
4.21.1 Data container ..... 77
4.21.1 File name ..... 77
4.21.1 Data format ..... 77
4.21.2 Remote configuration ..... 77
4.22 Modem Output - FTP ..... 78
4.22.1 Server ..... 78
4.22.2 Port ..... 78
4.22.3 FTP mode ..... 78
4.22.4 Security ..... 78
4.22.5 Directory ..... 78
4.22.1 Output type ..... 78
4.22.2 File name ..... 78
4.22.3 Verification ..... 79
4.22.4 Input parameters ..... 79
4.22.5 Remote configuration and firmware upgrade ..... 79
4.22.6 Remote access ..... 80
4.23 Modem Output - HTTP ..... 81
4.23.1 Server ..... 81
4.23.2 Extended path ..... 81
4.23.3 TCP port ..... 82
4.23.4 Security ..... 82
4.23.4.1 Basic ..... 82
4.23.4.2 HTTPS ..... 82
4.23.4.3 Azure-IoT SAS ..... 82
4.23.4.4 Token ..... 82
4.23.4.5 Header ..... 82
4.23.5 Data format ..... 82
4.23.6 Remote access ..... 83
4.24 Modem Output – HTTP Azure-IoT ..... 84
4.24.1 Server ..... 84
4.24.2 Device ID ..... 84
4.24.3 Device Key ..... 84
4.25 Data Output - MQTT ..... 85
4.25.1 Server ..... 85
4.25.2 Port ..... 85
4.25.3 Security ..... 85
4.25.4 Root topic ..... 85
4.25.5 Client ID ..... 86
4.25.6 Clean Session ..... 86
4.25.7 Data format ..... 86
4.25.8 Max Payload ..... 86
4.25.9 Input Parameters ..... 86
4.25.10 Remote configuration and firmware upgrade ..... 87
4.25.11 Remote access ..... 88
4.25.12 COM-Tunnel ..... 89

4.25.12.1 Topics ..... 89
4.25.12.2 Limits ..... 89
4.26 Data Output - TCP ..... 90
4.26.1 Server ..... 90
4.26.2 Port ..... 90
4.26.3 Security ..... 90
4.27 SMS output ..... 91
4.27.1 Phone number ..... 91
4.27.2 Data send ..... 91
4.27.3 Data format ..... 91
4.27.4 Remote configuration ..... 91
4.28 Data Output - Satellite ..... 91
4.29 Parameter overview ..... 92
4.29.1 Parameter list ..... 92
4.29.2 Parameter order ..... 92
4.29.3 Measurement list ..... 93
4.29.4 Output list ..... 93
4.29.5 Alarm list ..... 94
4.29.6 Alarm limits ..... 94
4.30 Maintenance Menu ..... 95
4.30.1 Field Testing ..... 95
4.30.1.1 Verify Analog inputs ..... 95
4.30.1.2 Digital input test ..... 96
4.30.1.3 SD card test ..... 96
4.30.1.4 Battery test ..... 96
4.30.1.5 COM port terminal ..... 97
4.30.1.6 Data Download ..... 98
4.30.1.7 Format SD Card ..... 99
4.30.1.8 Configuration Down/upload ..... 99
4.30.1.9 Firmware Upgrade ..... 99
4.30.1.10 Modem Maintenance ..... 99
4.30.1.11 Bootloader Menu ..... 99
4.31 Users & rights ..... 99
4.31.1 Administrator ..... 100
4.31.2 Operator ..... 100
4.32 SDI-12 ..... 100
4.32.1 SDI-12 Hardware ..... 100
4.32.2 SDI-12 Wiring ..... 100
4.32.3 SDI-12 Baud Rate and Frame Format ..... 101
4.33 RS232 ..... 101
4.34 RS485 ..... 102
4.35 Analog Inputs (0/4..20mA) ..... 103
4.35.1 Loop Powered Devices ..... 103
4.35.2 Analog Inputs (0 .. 10 V) ..... 103
4.36 Analog Differential Inputs ..... 103
4.36.1 Differential input ports theory of operation ..... 103
4.36.2 Common mode noise rejection ..... 104
4.36.3 Offset calibration ..... 104
4.36.4 Using Load Cells ..... 104
4.36.5 Bridge of Wheatstone ..... 104
4.37 Potentiometer input ..... 104
4.38 Digital inputs ..... 106
4.38.1 Pull up type ..... 106
4.38.2 Pull down type ..... 106
4.38.3 Electrical specifications Digital inputs ..... 106
4.39 Alarming ..... 107
4.39.1 Alarming - principal of operation ..... 107

4.39.2 Advanced Alarming ... 109
4.40 Firmware Upgrade ... 110
4.40.1 When to use Firmware upgrades ... 110
4.40.2 Firmware upgrade procedure ... 110
4.40.3 Firmware upgrade over the air ... 110
4.40.4 Firmware Driver limitations ... 110
4.40.5 Power Switch Limitations ... 111
4.40.6 Modem Firmware Upgrade ... 111
4.41 SD-card ... 111
4.41.1 Inserting an SD-card ... 111
4.42 Native TXT Data Format ... 112
4.42.1.1 Header: ... 112
4.42.1.2 Parameter Code: ... 112
4.42.1.3 Parameter Name: ... 112
4.42.1.4 Parameter Unit: ... 112
4.42.2 D-Records ... 113
4.42.2.1 Parameter Code: ... 113
4.42.2.2 Parameter value: ... 113
4.42.3 System records ... 113
4.42.4 Data Modifiers ... 114
4.43 JSON Data Format ... 115
4.44 Compacted Data Format ... 116
4.44.1 Records ... 117
4.44.1.1 First byte of record ... 117
4.44.1.2 Number-bytes ... 117
4.44.1.3 Time-record ... 118
4.44.1.4 Value-record ... 118
4.44.1.5 JPG-header-record ... 119
4.44.1.6 JPG-data-record ... 119
4.45 Sparkplug-B Data format (V1.0) ... 120
4.45.1 Configuring the MQTT-driver ... 120
4.45.1.1 Root topic ... 120
4.45.1.2 Name of Group ... 120
4.45.2 Data output ... 121
4.45.3 Metrics ... 121
4.45.3.1 Name of Metric ... 121
4.45.4 Datatype ... 122
4.45.5 Message examples ... 122
4.45.5.1 NBIRTH ... 122
4.45.5.2 NDEATH ... 123
4.45.5.3 NDATA ... 123
4.46 Configuration snippets ... 124
4.46.1 Payload format ... 124
4.46.1.1 Global system Settings ... 124
4.46.1.2 Alarm settings ... 125
4.46.1.3 Driver Settings ... 125
4.46.1.4 Parameter Settings ... 125
4.46.2 Payload example ... 126
4.47 Input-drivers ... 127
4.47.1 Analog sensors ... 127
4.47.2 Digital Pulse Sensor ... 127
4.47.2.1 Example configuration Rain Measurement ... 127
4.47.2.2 Counter ... 128
4.47.2.3 Quantity ... 128
4.47.2.4 Rate ... 129
4.48 Power supply ... 130
4.48.1 Internal RTC backup battery ... 130

4.48.2 Power consumption & Battery Life ..... 130
5 Pin configuration ..... 131
5.1 Pin descriptions ..... 132
5.1.1 Analog Inputs ..... 132
5.1.1.1 0/4..20mA inputs ..... 132
5.1.1.2 0…10V inputs ..... 132
5.1.1.3 Potentiometer input ..... 132
5.1.2 RS485 A & B ..... 132
5.1.3 Power Switch ..... 132
5.1.4 VBAT + ..... 132
5.1.5 RS232 RX & TX ..... 132
5.1.6 SDI-12 Hi ..... 132
5.1.7 Digital inputs ..... 132
5.1.8 +3V6 ..... 133
5.1.9 Antenna placement and field strength ..... 133
6 Maintenance and Repair ..... 134
6.1 RTC Lithium Battery replacement ..... 134
6.2 Recalibration ..... 134
6.3 XRAY ..... 134
7 Safety ..... 135
7.1 Power supply ..... 135
7.2 ESD ..... 135
8 Environment and disposal ..... 135
9 Transport and Storage ..... 135
10 Appendix ..... 136
10.1 Specifications ..... 136
10.2 CE Declarations of Conformity ..... 137
10.3 Open Collector output ..... 138
10.3.1.1.1 Using a relay to drive an external load ..... 138
10.4 Trouble shooting ..... 139
10.5 System messages ..... 140

# 1. Product description

The data logger is designed to retrieve, and store data from various sensors. This data is logged onto the embedded / removable SD-card. Also, the stored data can be send from the data logger to any remote computer you like. To use this feature, you need a valid SIM-card. Contact your local telecommunications provider for more information on the SIM card you will need. The unit accepts various power sources, selectable by the different version types. The user should connect his sensor(s) of preference to the connector board of the device. Captured data can be stored, send, visualized and manipulated in many ways.

The data logger is a small, ultra-low power, high-end data logger with built-in 2G/3G/4G cellular-modem. This small data logger, is further provided with an internal temperature sensor, 8 or 16 GB micro SD-card and an SIM card slot. The logger can be powered by an internal 3.6 Volt Lithium battery that will last for years when the logger is configured in a low-power mode.

Depending on model and edition a data logger can acquire physical signals by current loop inputs, voltage inputs, potentiometer input and digital inputs. They can also be provided with RS232/RS485/SDI-12 port to capture serial data (ASCII, MODBUS-RTU/ASCII, NMEA-0183, SDI-12) or to connect an accessory (Camera, Display, GPS, Satellite modem). More or special inputs/outputs can be added by means of internal stackable option boards/converters.

External sensors can be powered by the data logger itself, to prevent them to consume power while the data logger is a sleep. The excitation voltage is switched off during sleep as well.

Key features:

- A maximum sample rate of 4 Hz,
- Recording buffer up to 4 GB,
- Analog inputs (Except DS editions),
- Serial port(s) (Except AD editions),
- Digital input(s)
- Digital output
- Internal micro SD-card with standard FAT-32 File system for easy use with a PC,
- Easy configuration by menu items of embedded terminal application,
- Can be operated from Windows or Android app.
- Low power / long Battery life (see specification sheet),
- Embedded 2G/3G/4G modem for remote operation,
- Alarming by SMS, e-Mail or MQTT
- Log data transfer by HTTP(S), e-Mail, FTP(S), secure TCP, MQTT or SMS
- Camera picture transfer by HTTP(S), e-Mail, FTP(S), secure TCP or MQTT
- Remote FW-upgrade & Configuration update by HTTP(S), secure TCP or MQTT
- Internal voltage convertor for supplying 12 VDC power to the connected sensor(s),
- Firmware upgrade over the air for adding new features to your device.
- Remote configuration over the air, for adjusting your configuration from remote.

## 1.1 Models and Editions

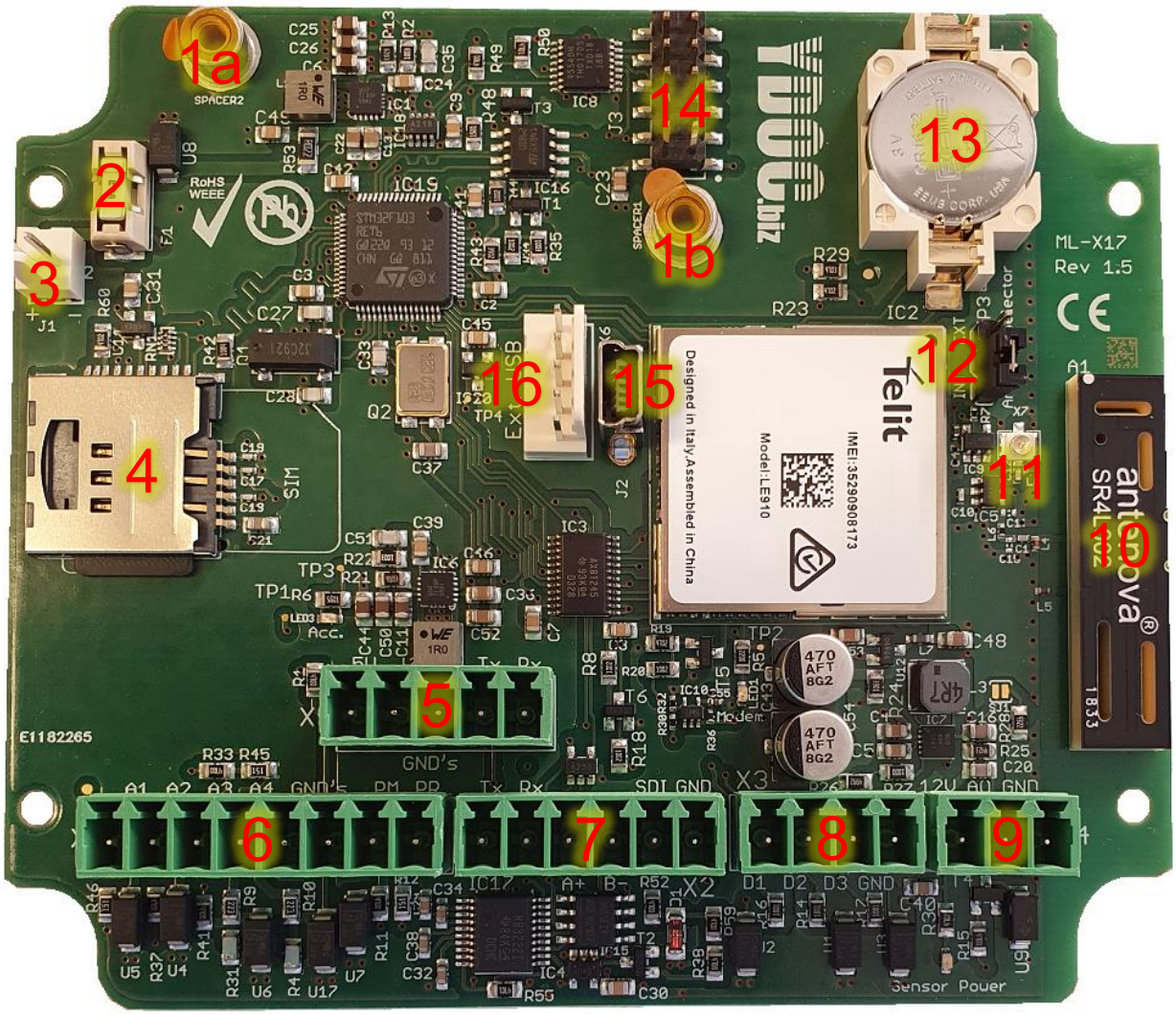
The ML-x17 data logger is a small, ultra-low power, cost effective data logger with built-in 2G/3G/4G cellular modem available for the EMEA, APAC and NA regions. This small data logger, is the same as mlx-15, with the addition of an option port for expansion of possibilities.

The data logger can acquire physical signals by 2 current loop inputs, 3 voltage inputs (1 potentiometer input) and 3 digital inputs. The data logger is provided with one serial port to capture measurements from ASCII, MODBUS, NMEA or SDI-12 compatible sensors. External sensors can be powered by the data logger itself, to prevent them to consume power while the data logger is a sleep. Up to 8 mathematical channels are available to calculate meaningful engineering values derived from sensor input values (e.g. a polynomial to calculate a flow from a stream level).

| SKU Table ML-x17 | | |
|---|---|---|
| **SKU format:** | **ML-x17y-z** (x=Modem, y=Edition, z=Power supply) | |
| **Modem** | **Description** | |
| **ML-217y-z** | With built in 2G cellular modem (Quad band GPRS) | |
| **ML-317y-z** | With built in global 3G modem (Penta band & Quad band GPRS fallback) | |
| **ML-417y-z** | With built in global 4G/5G modem (LTE-M / NB-IoT) | |
| **Edition** | **Description** | **Remarks** |
| **ML-x17TFT-z** | With 5 analog & 3 digital inputs, serial port & TFT-display cover on accessory port. | |
| **ML-x17ADS-z** | With 5 analog & 3 digital inputs, serial & accessory port. | |
| **ML-x17AD-z** | With 5 analog & 3 digital inputs (no serial port, no accessory port). | |
| **ML-x17DS-z** | With 3 digital inputs & serial port (no analog inputs, no accessory port). | |
| **Power supply** | **Description** | |
| **ML-x17y-LI** | Powered from a single 3.6V DC SAFT LSH20 or equivalent D-size lithium battery. | |
| **ML-x17y-3LI** | Powered from 3x 3.6V DC SAFT LSH20 or equivalent D-size lithium batteries. | |
| **ML-x17y-DC** | Powered from external 8..28V DC source. | |
| **ML-x17y-DC-LI** | Powered from external 8..28V DC source or 3.6V DC SAFT LSH20 lithium battery. | |
| **ML-x17y-DC-NM** | With integrated 3 x AA NiMH holder & charger for external 8..28V DC source. | |
| **ML-x17y-PV** | With integrated 1Wp solar panel and 1x 26650 3.2V LiFePO4 holder &charger. | |
| **ML-x17y-LFP** | With 4x 18650 3.2V LiFePO4 holder & charger for 12V (21VOC) ext. solar panels. | |
| **ML-x17y-SLA** | With integrated 12V battery charger for 12V (21VOC) external solar panels. | |

Example: **ML-417ADS-PV** is a data logger with built in global LTE-m/NB-IoT modem with PV-cover, digital & analog inputs, a serial and accessory port.

The data logger will be supplied without pre-mounted cable glands, giving the user the freedom to choose the number and size of the glands them self to avoid unnecessary points of risk for moisture penetration. We recommend removing the PCB before drilling.




1) Mounting bus for option board
2) Main Fuse (2A Fast Acting)
3) 3V6 Power Supply Connector
4) SIM(2FF) & Micro SD-card Holder
5) Accessory port (TFT, CAM, etc.)
6) Analog input port
7) RS232/RS485/SDI-12 Port
8) Digital input port
9) 12 V Sensor Power Output & Alarm Output connector
10) Hexaband antenna for 2G/3G/4G communications
11) U.FL Connector for external antenna
12) Internal/External antenna selection jumper
13) RTC clock Battery
14) Option port
15) USB port
16) Connector for optional external IP67 USB connector

# 2 Getting started

## 2.1 Vibration

At all times the data logger must be protected against vibrations. These vibrations can harm the performance of the data logger. Especially the real-time clock can be harmed by long-lasting vibrations

## 2.2 Do's & Don'ts

**Do's**

- Always provide a dry and clean environment when you open the case of a data logger,
- When you open the case, use a Philips screwdriver of 2mm for loosening the screws,
- Protect the data logger against mechanical stress and vibrations

**Don'ts**

- Don't try to use a plain screwdriver for loosening the screws, you will damage the screws,
- Avoid touching the PCB directly.

## 2.3 Inserting the SIM-card

The wireless data functions will only work when an activated SIM, with a valid subscription is placed in the data logger. In the menu the configuration and network settings must also match to those supplied by the SIM card provider. The pin code of the SIM card must be removed prior to insertion in the data logger. To prevent problems with the SIM card it can be inserted in a mobile phone or 2G/3G data modem. In a Mobile phone or 2G/3G modem the 2G/3G, SMS and GSM data functionality can be tested.

When you have obtained a 2FF SIM-card, you can insert it into the SIM-card-holder.

**Beware:**

- Remove the PIN code (this can be done with the use of a mobile phone),
- Check the settings of your mobile provider,
- Check the settings for communicating via HTTP, FTP, E-mail or TCP,
- Check the capability of data communication for your service-provider,
- Make sure the SIM-card is installed correctly, and not upside down. The Oblique side of the card should be visible. (see picture).
- Installation of the SIM card needs to be done in a clean and dry environment.
- Avoid contact with the electronic parts around the SIM card.
- Remove power before changing or inserting a SIM card.
- The same applies to the SD-card.

**ESD Attention:** Although the data logger is designed to withstand certain amounts of electrostatic discharge, it is advised to avoid discharged risks. Especially when the housing is open and the electronic parts are exposed. Please do not touch the PCB if you don't have to. It is strongly recommended to use an earthed wrist-band when touching the PCB.

The data logger must be handled with care and never exposed to ESD discharges. When installing a sensor or other wiring, make sure there is no power on both devices. ESD discharges could cause invisible damage. This endangers long term stability and proper operation.

## 2.4 Power on for the First time

In the factory the data logger is programmed with the necessary system information. This information is viewable in the menu. The data logger is ready to use out of the box if the preparations are checked.

- SIM card is inserted,
- SD-card is inserted,
- Internal battery, or mains power is connected
- Antenna is connected.

Next, connect your sensors, please consult your sensor's manual for wiring.

## 2.5 Connect to a PC, tablet or smart phone

OS compatibility: The data logger can be connected by USB2.0 to a Windows PC or Android tablet or smart phone. ydocTerminal for Windows can be downloaded from **www.ydoc.biz**, ydocTerminal for Android can be downloaded from the Android Playstore.

# 3 Operating Basics

## 3.1 Configuration menu

The data logger can be configured by means of terminal emulation software like 'hyper terminal' or our own terminal emulation software 'YDOC-terminal', which can be downloaded from **www.ydoc.biz** or ydocTerminal for Android from the 'Google PlayStore'.

The menu is comprehensive and easy to use. For each different sensor the same approach is used. Below, one example is given to fully understand the operation of the menu. The example takes you through a configuration from start to final stage. Only a few menu-items are used in this example, for a complete overview of all menu-items, see chapter Reference. All menu items use the same approach which is explained in this example. This example explains only the configuration of the firmware, NOT the wiring. For wiring information see the reference.

**Example:**

Let's Configure an ML-317ADS for operation with the following:

- Analog pressure transmitter 4 .. 20mA
- CT2X conductivity / temperature sensor (INW)
- FTP data output
- EMAIL data output

Connect the logger to a free USB port on your computer and open with a terminal emulator the virtual COM-port (e.g. COM7) as assigned by Windows to the logger. The first time Windows will ask (if not already installed) for an USB-driver, which can be installed by installing 'YDOC-Terminal'.

When the terminal emulator has opened the COM-port, press:
`<Ctrl>A<Shift>M<Ctrl>D` to enter the configuration menu.

You'll see a screen similar to this:

Running
ML-A417ADS Logger Version 4.0 Build 3 (108173612)

```
Configuration Menu ML-317ADS Logger Version 4.0

[0] Run
[1] Date & time          >> 2020/03/13 13:12:14
[2] System information   >> 108173612
[3] Configuration setup  >>
[4] Parameter overview   >> 6
[5] Maintenance
[6] Users & rights
>
```

First we like to give this data logger appropriate identification codes.

So, press 3 <Configuration Setup>

(To avoid unauthorized local access we recommend to specify a local password).

You'll see this screen:

First Rename your Device by
Selecting option 1
(General Settings)

```
Configuration setup

[0] Exit
[1] General settings      >> YD0C
[2] Modem settings
[3] NTP time update       >> Not used
[4] Alarm settings        >> Not used
[5] Option boards         >> None
[6] Internal sensors      >> Not used
[7] Analog inputs         >> Not used
[8] Digital inputs        >> Not used
[9] Network signal sensor >> Not used
[A] Serial port           >> Not used
[B] Accessory port        >> Not used
[C] Derived channels      >> Not used
[D] Modem output          >> Not used
[-] Wifi output           >> N/A
>
```

After selecting "General
Settings" the screen will look like this:

```
General settings

[0] Exit
[1] System name                    >> YD0C
[2] Data log interval              >> 00:10:00
[3] Timestamp round down           >> 00:01:00
[-] Avoid equal timestamps         >> Off
[5] Alarm output port              >> Disabled
[6] Battery (protection & lifespan) >> Rechargeable NiMH
[7] Permanent SD storage           >> Data & Diagnostics
[8] Deployment start date and time >> 2020/03/13 12:00:00
[9] Deployment end date time       >> 2020/06/01 00:00:00
[A] Daily operating time slot      >> 24 hour
[B] Daily schedule time shift      >> Not used
[C] Time zone                      >> 0
[D] Summer time (daylight saving)  >> Not used
>
```

1. Press 1 and enter the name of your preference
2. Choose your data logging interval. We used 10 minutes, and NO Alarming.
3. Enter the deployment date & time. This can be a time in the future when the logger should start working. We will use the actual date & time to start right away.
4. Exit and save changes.
5. Now, the overall configuration is setup and we proceed with the configuration of the sensors and data output.

Next type 7 for Analog inputs

```
Analog inputs

[0] Exit
[1] Port 1 (mA)        >> Analog1
[2] Port 2 (mA)        >> Not used
[3] Port 3 (V)         >> Not used
[4] Port 4 (V)         >> Not used
[5] Potentiometer      >> Potentiometer
[T] Analog input test  >> Not done
>
```

Choose 1 for Port 1 (mA) and the next screen will appear:

```
Analog input

[0] Exit
[1] Name                      >> Analog1
[2] Sensor power              >> Disabled
[3] Sample interval           >> Data log interval
[4] Port mode                 >> Port 1: 4-20 mA
[5] Parameter settings        >> Analog
[6] Parameter value at 4 mA   >> 0 units
[7] Parameter value at 20 mA >> 100 units
[8] Determine linear conversion function (2 calibration points)
[9] Determine linear offset only (1 calibration point)
[R] Remove
[T] Test measurement
>
```

Assign a name to the sensor (option 1: Name)

1. Set the power switch to enabled and enter the warm-up time. (the power switch will supply the sensor with 12 Volts, and will be activated the time you specify, before a measurement is taken)
2. At default Set the sample interval to the 'Data log interval' so only one sample per data log interval will be taken. You can set the sample interval to a specific value in case you want to record periodic averages (e.g. a 10-minute wind direction, see chapter: Aggregation channels) or continues alarm sampling.
3. Set the parameter name.
4. Set both minimum (option 6) and maximum (option 7) values of your sensor at 4mA and at 20mA. If you don't know those values, you could determine the scaling by measuring two calibration points (option 8). Those calibration points don't have to be at the sensors absolute minimum and maximum, but just two different points within the range of the sensor. (e.g. a measurement at 1m water level and a measurement at 2m water level, while the sensors range is 0...10m)
5. If necessary option 9 can be used to perform an offset correction by measuring a single calibration point.
6. Save and Exit

Your screen will look similar to the one underneath:

```
Analog sensor

[0] Exit
[1] Name                      >> Water height
[2] Sensor power              >> Enabled; Warm up 00:00:02
[3] Sample interval           >> Data log interval
[4] Port mode                 >> Port 1: 4-20 mA
[5] Parameter settings        >> Analog
[6] Parameter value at 4 mA   >> 0 meter
[7] Parameter value at 20 mA >> 10 meter
[8] Determine linear conversion function (2 calibration points)
[9] Determine linear offset only (1 calibration point)
[R] Remove
[T] Test measurement
>
```

Now we add the CT2X to the system:

Before we can do this, we need to know the MODBUS address and register address of this sensor, always consult the manual of your sensor first. In this case the address of the sensor is #1. This sensor measures two parameters: Temperature and conductivity. So, we need to look up two register addresses. (Temperature: 62592 and Conductivity 62594, baud rate 38400)

Go to menu-option A <serial port 1> => RS 485 sensors => MODBUS.
After configuring your screen should look like this:

```
MODBUS sensor

[0] Exit
[1] Name                  >> MODBUS
[2] Port settings         >> RS485 8N1; 19200 Baud; Address 1
[3] Sensor power          >> Disabled
[4] Sample interval       >> Data log interval
[5] Protocol type         >> RTU
[6] Measurement commands  >> Not used
[7] Register type         >> Holding
[8] Data type             >> WORD (unsigned 2 bytes)
[9] Data start register   >> 62592 (462593)
[A] Parameter 1           >> Water temperature
[B] Parameter 2           >> Conductivity
[C] Parameter 3           >> Not used
[D] Parameter 4           >> Not used
[E] Parameter 5           >> Not used
[F] Parameter 6           >> Not used
[G] Parameter 7           >> Not used
[H] Parameter 8           >> Not used
[I] Parameter 9           >> Not used
[M] More parameters       >> Not used
[R] Remove
[T] Test measurement
>
```

The parameters are always read from Register start address onwards, so the specific address (62594) of the conductivity-measurement is not used in the configuration. It automatically increases the register-address in order to read it. All you have to do is select how many parameters are used, and the data type. More on this topic is included in the reference chapter of this manual.

Multiple CT2X sensors can be connected to an RS485 bus so you need to specify its address (assuming 1 in this example). This sensor is a multiple parameter sensor, so please specify which you want to log.

**Attention:**

Because this is a digital sensor, it takes a little bit more time to measure than an analog sensor. So don't set the sample interval of digital sensors too short. 1 second is possible, but probably not ¼ sec. Just to be sure we choose to sample at the data log interval. Consult the manual of your digital sensor and check the response time. The sample-interval must be larger than the response time of the sensor.

Next thing is the setup of the data outputs.

Before that, we have to configure the general modem settings for mobile operation.

Go to Menu-option 2 <Modem settings>

```
Modem settings

[0] Exit
[1] Provider selection
[-] Network              >> 4G
[3] Technology           >> LTE-M
[...
[S] Network signal test >> Not done
[T] APN login test       >> Passed
>
```

First, perform [T] an APN login test. If successful you can leave [0] this menu without specifying any APN-settings manually.

***Note**: In case of several successive failures it might be because your SIM requires manual APN settings, please ask your provider for the correct settings first.*

```
APN test

Modem full power
Please wait...
ATE0;+CMEE=2;+CCLK="15/11/26,14:26:21+00";+CFUN=5;+CPIN?
+CPIN: READY
OK
AT+CREG=2;+CREG?
+CREG: 2,1,"17F2","0A5C825",2
OK
Registered on 3G network
ATS12=20;#DST0=1;+CGDCONT=1,"IP","internet";#SGACTCFG=1,5,180,1
OK
AT#SGACT?
#SGACT: 1,0
OK
AT#SGACT=1,1,"xs4all","****"
OK
#SGACT: 10.137.12.103
Modem idle

APN test OK

Press any key! >
```

Now we can setup the FTP, TCP and Email output.

Therefore, go to menu-option D:  <Modem output> and choose the wished output protocol e.g. Email:

```
Email settings

[0] Exit
[1] Name                >> Email
[2] Send interval       >> Normal 01:00:00; Alarm Not Used
[3] Send delay          >> Not Used
[4] SMTP Server         >> smtp.provider.com
[5] SMTP Port           >> 25
[6] User name           >> rehba01
[7] Password            >> ********
[8] Originator address  >> r.kleine@provider.com
[9] Destination address >> your@provider.com
[A] Subject             >> YDOC upper stream
[R] Remove
[T] EMAIL test          >> Not Done
>
```

Beware:
Some providers strictly check the Originator address. So make sure this address is valid.

Now perform an Email test and check if it is working right.

It is strongly recommended to include some internal sensors in the configuration, because of the monitoring of the performance of the data logger itself. Most users like to keep track of the battery-life for example.

Go To menu-option 6 <internal sensors>

```
Internal sensors

[0] Exit
[1] Name                   >> Internal
[2] Sample interval        >> Data log interval
[3] Battery capacity       >> 14000 mAh
[4] Battery replaced       >> Yes
[5] Rest capacity          >> Not used
[6] Rest power             >> Not used
[7] Processor temperature  >> Processor temperature
[8] Average voltage        >> Not used
[9] Max voltage            >> Not used
[A] Min voltage            >> Min voltage
[B] Average current        >> Not used
[C] Max current            >> Not used
[D] Min current            >> Not used
[E] Operating cycle        >> Operating cycle
[F] Free disk space        >> Not used
[R] Remove
[T] Test measurement
>
```

Set the “Battery Replaced” to “Yes”, only when you installed a new lithium battery.
Select the items you like to be informed about.
See the reference for a complete description of the items.

Now your data logger is configured and ready to use.

# 4 Reference

## 4.1 Principle of Operation

### 4.1.1 Intervals

Your YDOC data logger is capable of collecting and storing data of multiple sensors. To accomplish this, many tasks are performed. These “tasks” are scheduled and executed on their specified time. The timing of this process is very important and is determined by the internal scheduler. This scheduler keeps track of all the internal states of the various tasks and assigns processor time to the different tasks. Each task is executed on its own interval. To understand more about this, Fist we explain the different intervals.
There are three different intervals:

1) Sample Interval
2) Data Log Interval
3) Send Interval

#### 4.1.1.1 Sample interval

The sample interval is the interval at which a sample from the sensor is taken (expected). So, measurements from sensors are done at the sample interval. The sample interval is valid ONLY when the device is in the active state. When the data logger is in sleep-mode, the tasks, triggered by the sample interval will NOT execute, unless they are used in aggregations (e.g. a 10-minute average, see chapter Aggregation channels), scheduled for continuous alarm sampling, or if a pending alarm is detected and you have set “alarm set delay” to number of samples”. So, in most cases you could follow the default data log interval to take a sample.

#### 4.1.1.2 Data logging interval

This interval determines when a data value, obtained by the sample interval-task, is stored onto the SD-card. This type of interval is ALWAYS valid. So, even when the data logger is in sleep-mode, it will wake up when the Data logging Interval has reached its count.

#### 4.1.1.3 Independent data logging interval

By default, there is only one data logging rhythm, either following the normal data log interval or the alarm interval. However, for power- or time-consuming serial sensors or accessory port inputs, you can specify a separate, independent data logging interval.

***Attention***: Each sensor or input with an independent data logging interval will create its own data log record in the data storage. As a result, log files may contain multiple records or lines with the same timestamp if they happen to be recorded at the same moment. Additionally, in CSV files, such records or lines may contain empty fields.

#### 4.1.1.4 Send Interval

The send interval determines the interval on which data is send, via the internal modem. This interval is ALWAYS valid, even if the data logger is in sleep-mode.

#### 4.1.1.5 Example

Let's evaluate the following settings of the data logger:

- Sample Interval: 5 seconds
- Data Log Interval: 10 minutes
- Send Interval: 3 hours

When the configuration is ready, and the user disconnects the USB-Cable;

1. Data logger is switched into sleep-mode, and current draw is reduced to a minimum level.
2. The Sample interval of 5 seconds is discarded (except for special alarming modes), because this interval is only active when the data logger is NOT in sleep-mode. So, nothing happens until the Data Log interval has reached his count. (So, this happens on 0, 10, 20, 30, 40, 50 minutes every hour)
3. When the Data Log Interval count has reached his count, the data logger will awake from the sleep mode, and will take a sample and stores the data on the SD-card. When the sample is taken, the data logger goes into sleep-mode again. This is repeated, until the time has matched the Send interval. So, in this example, this is 3 hours.
4. When Data Send interval is reached, the data logger will wake-up, and starts to send the previously collected data (stored on the SD-card) to the server. So, in this example, every 10 minutes a sample is taken, and every 3 hours, 18 samples are send.

**Note:** So, the data logger does not perform any averaging. Even if the sample interval is much faster than the data log interval, only one sample is stored.
The use of the sample interval is for evaluating proper behaviour of the system, while the USB is connected, and for special alarming modes. Because in that case, the sample interval is valid, and the user can observe the value's obtained from the sensor in real time. In this case, the user can evaluate these values every 5 seconds.

### 4.1.2 Common data output settings

Data can be transferred by various internet protocols a/o FTP & MQTT of which the specific settings are described in individual chapters. They however share some protocol independent features that are describe below.

#### 4.1.2.1 Send delay

Data output transmissions will be performed at scheduled discrete intervals. Setting-up a cellular connection takes some time, which is normally more than enough to acquire measurements scheduled at the same moment. If acquiring a sample from a slow sensor or when taking a hires camera picture, it might not be ready in time to be included in the current transmission. Specifying a few seconds of send delay might avoid this to happen. A send delay could also be applied to avoid overload on a constraint server due to simultaneous transmissions of a lot of units.

#### 4.1.2.2 Output type

Per output driver you can specify which type of data you want to output: 1) All logged data since the previous successful transmission, 2) Just the last know sampled actual values or 3) a heartbeat message.

The primary output is normally chosen for transmission of the logged data.
A heartbeat can be used to indicate thru an alternate output that a system is still a live or to have an entry point in time to do a remote configuration if not possible thru the primary output (see: chapters remote configuration).

#### 4.1.2.3 Data format

This defines the data format of the transmitted content/file. This can be our native TXT format, JSON or a CSV (comma separated value) format which can easily be picked up by third party applications like Microsoft Excel. The column order of the CSV file, follows the order of the parameter list (see: chapter “Parameter overview”) . We prefer native or JSON format as it can contain data as well is diagnostic info.

#### 4.1.2.4 Data filter

The logger records Log Data (measurements) as well as system Diagnostics. With a data filter you can specify which you want (or both) to include in the transmission.

#### 4.1.2.5 Max payload

In circumstances like a network outage a data log file will grow bigger and bigger and depending on the application to several megabytes or more. If the payload becomes bigger than 1MB the logger will chop the transmission up in several parts. In areas with unstable coverage you can trim down the maximum payload setting to an appropriate value.

## 4.2 Parameters

All through the configuration-settings of the data logger, you will encounter parameters to setup your logger. They are used on all ports (analog, digital, serial) and are generic in use. This section describes the use and properties of parameters. Below a screenshot of some parameter-settings are shown.

```
[0] Exit
[1] Name                    >> Analog
[2] Code                    >> AIN
[3] Unit                    >> meter
[4] Initial value           >> 0
[5] Value factor            >> 1
[6] Value offset            >> 0
[7] Decimals                >> 3
[8] Data log                >> On
[9] Data output             >> Modem
[A] Boundary control        >> None
[-] Low boundary limit      >> N/A
[-] High boundary limit     >> N/A
[-] Alarm message           >> Disabled
[E] Alarm log               >> On
[-] Alarm output            >> Disabled
[-] Low-low alarm limit     >> N/A
[H] Low alarm limit         >> N/A
[I] High alarm limit        >> N/A
[-] High-high alarm limit   >> N/A
[K] Alarm set delay         >> 0 samples
[L] Alarm reset hysteresis  >> 0.5 meter
[M] Alarm reset delay       >> 0 samples
[R] Remove
>
```

### 4.2.1 Name

A appropriate name for the parameter (real life measurement).

### 4.2.2 Code

A short alphanumeric code to designate the parameter. This code will be used in the data files.

### 4.2.3 Unit

The unit of the real-life parameter (i.e. meters)

### 4.2.4 Initial value

This is the value assigned to a parameter at re-boot. This value has is importance for parameters that are not immediately sampled after re-boot. The initial value can also be used as a literal/reference value in calculation channels. The initial value can also be altered remotely by sending a configuration snippet to the data logger (see: See chapter configuration snippets).

### 4.2.5 Value factor

Multiplier to transform the electrical signal into a real-life value. Default is 1. So, when the value factor is 2, all measurements are multiplied by 2. (in math: F(x) = ax+b, where a equals the value factor)

### 4.2.6 Value offset

Also to transform the electrical signal into a real life value, but now an addition (+) default = 0.
(in math: f(x) = ax+b, where b equals value offset) This is very convenient while using 4..20 mA sensor, because they introduce an offset. (4 mA = 0 centimetre)

### 4.2.7 Decimals

The numeric precision of the value, displayed and stored on the SD card. (of course, not the real-life precision, only the numeric presentation)

### 4.2.8 Data log

Setting which enables the storage of the parameter to the SD card. When set to off it will be measured and displayed, but NOT stored on the SD card. Default set to “ON”.

### 4.2.9 Data output

Specify the data output mode for this parameter, the default is output by cellular Modem (TCP, FTP, MQTT etc) and depended on the available features you can also output the parameter to WiFi, Accessory port or an Auxiliary (e.g. ML-OO-MODBUS slave or ML-OO-SW switches) output board.

### 4.2.10 Boundary control

If enabled you can specify boundary limits for the parameter and indicate if values should be clipped to the specified boundaries or skipped and invalidated when outside the specified boundaries.

***Note***: Boundaries are mutual exclusive with LoLo & HiHi alarm limits.

### 4.2.11 Alarm message

Here, the user can select whether he wants to receive an alarm message on this parameter (in alarming conditions) or not. Alarm messages are send by SMS, e-Mail or MQTT.

### 4.2.12 Alarm log

Setting which enables the storage of the alarm condition of the parameter to the SD card. When set to on it will store an unscheduled data log in a single data record, and the passing of the alarm limits in a system record.

### 4.2.13 Alarm output

Setting which sets or clears the digital alarm output port when the selected alarm limit is passed (Low, High or both).

### 4.2.14 Alarm limits and delays

Within every parameter settings menu, the user can define all limits and delays to customize his alarming mode. More info on these settings and the alarming mode is found in section: Alarming

## 4.3 Configuration Menu Settings

This Chapter describes the details of the configuration-settings of your data logger. The configuration menu is entered by using YDOC-terminal and typing <Ctrl>A<Shift>M<Ctrl>D.

### 4.3.1 Date & Time

This section allows you to set date and time manually. If pressed enter, the existing system data/time is displayed.

### 4.3.2 System Information

A read only menu-section which displays all specific details about the data logger, an example is given below:

```
System Information ML-317ADS Logger Version 4.0 Build 1

[0] Exit
[-] Serial number            >> 5024367
[-] Hardware model           >> ML-X17
[-] Modem type               >> 3G UE910-GL
[-] IMEI number              >> 355856050243670
[-] Hardware edition         >> ADS (Analog+Digital+Serial)
[-] Production date          >> 2015/06/22
[-] Test date                >> 2015/06/23
[-] RTC adjustment frequency >> 512.0115 Hz
[-] Low power sleep current  >> 0.08 uA
>
```

## 4.4 Configuration Setup

This menu-section allows you to configure your data logger for your specific task. This section is divided into logical items, which contains more settings will be discussed next: Below a screenshot of the configuration setup is displayed:

```
Configuration setup

[0] Exit
[1] General settings      >>
[2] Modem settings
[3] NTP time update       >> Not used
[4] Alarm settings        >> Not used
[5] Option boards         >> None
[6] Internal sensors      >> Internal
[7] Analog sensors        >> Water height; Potentiometer
[8] Digital sensors       >> Display
[9] Network signal sensor >> Not used
[A] Serial port           >> Not used
[B] Accessory port        >> Not used
[C] Derived channels      >> Not used
[D] Modem output          >> Not used
[-] Wifi output           >> N/A
>
```

Each item is explained below, including sub items

```
General settings

[0] Exit
[1] System name                      >>
[2] Data log interval                >> 00:01:00
[3] Timestamp round down             >> Not used
[4] Avoid equal timestamps           >> Off
[5] Continuous alarm sampling        >> Off
[6] Direct data output on data alarm >> Off
[7] Alarm output port                >> Disabled
[8] Power supply board               >> ML-PB-LFP
[9] Battery (protection & lifespan)  >> Rechargeable LFP
[A] Permanent SD storage             >> Data & Diagnostics
[B] Deployment start date and time   >> 2018/04/26 12:19:47
[C] Deployment end date and time     >> N/A
[D] Daily operating time slot        >> 24 hour
[E] Daily schedule time shift        >> Not used
[F] Time zone                        >> 0
[G] Summer time (daylight saving)    >> Not used
>
```

## 4.5 General Settings

Here you enter your global settings, like data log interval etc. These are the settings:

### 4.5.1 Exit

Exits the menu

### 4.5.2 System Name

Allows the user to give a name to the data logger (up to 32 characters). This name will be used in the data files, produced by the logger.

### 4.5.3 Data log interval

This is the interval on which the logger will write a measurement to its SD-card. These records will be present in the data file and this setting defines the size of the data file per interval. The data log interval knows 2 different modes:

#### 4.5.3.1 Normal mode

Normal data logging mode, the regular data will be logged

The normal interval must be greater than the alarm interval
and can't be greater than the smallest normal send interval
and can't be smaller than the highest normal sample interval

#### 4.5.3.2 Alarm mode

Alarm mode data log interval defines the (often faster) rate of data logging, during alarming conditions

The alarm interval must be less than the normal interval
and can't be greater than the smallest alarm send interval
and can't be smaller than the highest alarm sample interval

### 4.5.4 Timestamp round down

This feature allows you to round down the timestamps in the data log to fixed time brackets. Some sensors will take a few seconds to finish data reading, and as a result you get timestamps like 12:50:07. If you want to round it down to 12:50:00, you can set the round down to 10 seconds.

### 4.5.5 Avoid equal timestamps

A log file consist of time stamped records, but as some sensors are logged with an independent interval, it could occur to have multiple records with the same timestamp. If your collection software can't handle duplicated timestamps check ON this option to add 1 second to consecutive equal timestamped records.

### 4.5.6 Power supply board

An ML-x17 can be provisioned with various power supply boards working with different type of battery chemistries. Please select a board type matching with your hardware. If you don't know specify the Battery type instead in the next option.

### 4.5.7 Battery (protection & lifespan)

This setting is automatic applied if you have previously chosen a matching power supply board.

Choose 'LFP' when using LiFePO4 3.2V batteries, in this mode the data logger will not start if the voltage is below 2.8V, giving a DC-source (e.g. solar panel) the chance to sufficiently replenish a depleted battery

and it will also if necessary prevent the batteries from over charging by applying a load (by staying awake) till voltage drops below a safe level.

Choose 'NiMH' when using NiMH AA batteries, in this mode the data logger will not start if the voltage is below 3.4V, giving a DC-source (e.g. solar panel) the chance to sufficiently replenish a depleted battery.

Choose 'Lithium batteries' when primary batteries are used, this will prevent the batteries to be drained while connected to USB by going in to low-power sleep after 3 minutes of non-activity. In this mode the data logger can also detect when the Lithium battery is used in a DC-LI edition, to be able to calculate the remaining Lithium battery capacity.

Choose 'None' when using an external 12V battery or a DC-source without batteries.

#### 4.5.7.1 Modem protection

When a cellular modem switches on, it consumes a relatively high amount of power. If it encounters an unstable power source, such as depleted, deteriorated or wrong type of batteries, there's a risk of a modem crash at an inconvenient time, potentially resulting in permanent malfunction. While selecting the appropriate battery protection described previously can mitigate this risk to some extent, in practice, it's not always chosen correctly, or unstable third-party power supplies are utilized.

Starting from FW4.8B16, we've implemented an additional layer of protection to minimize the risk of damage to modems. When the supply low-load voltage drops below a certain determined dectection limit, upcoming modem operations will be suspended for till a sufficient voltage level is restored.

### 4.5.8 Deployment Date and Time

Here you enter the specific Date & Time On which you like the measurements to start for the first time. This feature allows you to configure your data logger (in a nice and warm environment) before you install it in the field. It prevents the device to measure and send fake data, due to the dislocation of the unit. If this feature is used, the unit will sleep until the deployment start date & time. You can also specify a date time when the data logger should stop with measuring and transmitting.

### 4.5.9 Daily operating time slot

This setting allows you to disable the data logger from operation, outside this specified time slot. This makes sense when measurements are correlating with working hours or in other specific situations. E.g. a data logger won't alarm, or log data, when the temperature of the boiler is too low, while no-one makes use of that. The logger will alarm you, when this happens during a working day.

*Keep in mind that this feature will also move the starting point of the logging-interval to the beginning of the daily operating slot. i.e. if you have a 24 hr. log interval, which is normally active a midnight, it will now log @ the start of the Daily operating slot.*

#### 4.5.9.1 Daily schedule time shift

All timing is standard relative to midnight, e.g. if you do one transmission per day it will occur at midnight unless you specify a time shift. If you want one daily transmission at 7:00 AM you should specify a 7h time shift.

### 4.5.10 Time zone

Time offset to meet local time zones (half's and quarters are allowed).

### 4.5.11 Summer time

This setting allows the data logger to adjust its internal clock to summer time conventions. Worldwide, there are multiple conventions, and therefore the user can adjust it to manually. Also a couple of preprogramed summertime conventions are selectable (Europe / USA). OR the user can disable the feature (default)

When the user enables the summertime feature, as a consequence, there will be an anomaly in the data. At the beginning and at the end of the summertime-period.

## 4.6 Modem Settings

This section allows the user to setup the communication with the 2G/3G/LTE network, according to the settings, received from the service provider. Please consult your service provider for the right settings, before consulting your YDOC dealer.

```
Modem settings

[0] Exit
[1] Network                  >> 4G Only
[2] Technology               >> LTE-M
[3] Provider PLMN code       >> Automatic
[4] Provider selection       >> Automatic
[5] APN settings             >> Managed by SIM card
[6] Internet protocol        >> IPV4
[7] Customization (FWSWITCH) >> Global (ROW)
[8] Standby threshold time   >> 00:01:00
[S] Network signal test      >> Not done
[T] APN login test           >> Passed
>
```

### 4.6.1 Network

Choose which network generation is preferred (4G, 3G or 2G) and whether or not to use fall-back.

### 4.6.2 Technology

In case of an ML-417 you can choose the cellular network technology to use, being LTE-m or NB-IoT. If both technologies are available, its best to choose superior LTE-m. When using NB-IoT you have to enter a PLMN code matching with your provider (e.g. 20404 for Vodafone the Netherlands). The internet protocols used by the ML-x17, like MQTT, are TCP-based. TCP-based communication requires a certain network performance, which is not everywhere guaranteed by NB-IoT, it's a matter of trying…

### 4.6.3 PLMN code

A PLMN (Public Land Mobile Network) code is a unique identifier for a mobile network, consisting of an MCC (Mobile Country Code) and an MNC (Mobile Network Code). Keep this setting on automatic unless roaming restrictions apply, such as when using NB-IoT in some cases.

### 4.6.4 Provider Selection

This is an automated procedure which gives the user the overview of networks in the vicinity of the logger. Information about the provider and it's 2G or 3G capabilities are shown. The user can select his network of interest, or, he can set the device to automatic.

```
Provider selection

AT+COPS=3,0;+COPS?
+COPS: 0,0,"NL KPN",2
AT+COPS=?
+COPS: (2,"NL KPN",,"20408",2),(2,"NL KPN",,"20408",0),(1,"T-Mobile
NL",,"20416",2),(1,"vodafone NL",,"20404",2),(1,"T-Mobile
NL",,"20416",0),(1,"vodafone NL",,
"20404",0),,(0-4),(0,2)

[0] Exit
[A] Automatic selection >> On
[1] NL KPN (3G)         >> Automatic selected
[2] NL KPN (2G)         >> Automatic selected
[3] T-Mobile NL (3G)
[4] Vodafone NL (3G)
[5] T-Mobile NL (2G)
[6] Vodafone NL (2G)
[D] Deregister network
>
```

The use of this feature is, to let the user decide which network he will use, or automate that for him. In cases where the logger is situated near a foreign border, it is very important to make sure that the logger won't use the foreign network. Very often there are high costs concerned with the use of a foreign network. In these cases it is best to manually select your internet network provider (ISP). It will save you a lot of costs. All providers that are within reception area are shown, and the user can select his network of preference. The Fallback option 3G->2G is very good in areas where both are available but not that strong.

### 4.6.5 APN settings

Please perform an APN login test first. If successful, you don't need to specify any APN-settings manually.

***Note**: In case of several successive failures it might be because your SIM requires manual APN settings, please ask your provider for the correct settings first (being an Access Point Name with optionally a user name/ password and authentication method).*

### 4.6.6 Internet protocol

"If supported by the modem, you can choose to use IPv4, IPv6 with an extended addressing range, or IPv6 with IPv4 fallback.

### 4.6.7 Customization

Some US and Australian providers have specific network administration rules that require the modem to run a customized firmware version, which you can select using this setting.

### 4.6.8 Standby threshold time

To save power, the data logger turns off the modem after data transfers. However, since network registration consumes power and time, keeping the modem on can be beneficial if the next session occurs soon. This setting allows you to specify how long the modem stays powered for reuse within the set time.

#### 4.6.9 Network signal test

This is no setting, but a test to check network reception. It is very convenient to check your network reception at the installation site. A logger, which is configured at the office may NOT be working fine at the installation site. So, test this first, and make sure that the reception is sufficient. (1 bar absolute minimum, 3 bars is very nice)

#### 4.6.10 APN login test

You should test the APN login after changing APN settings, in order to verify the network connection.

## 4.7 NTP time update

This feature enables the automatic synchronization of the internal clock, by means of a Network Time Protocol (NTP) server on the internet.

#### 4.7.1 Time Update

Sets the feature on or off.

#### 4.7.2 Update interval

Interval on which the time is checked by the NTP mechanism, its fixed to 24 hours.

#### 4.7.3 Server

IP-address or URL of the NTP-server

#### 4.7.4 NTP Port

Port of the protocol, default is well known port 123

#### 4.7.5 Time update test

Manually forces device to update time via NTP, in order to test the feature.

## 4.8 Alarm settings

```
Alarm settings

[0] Exit
[1] Continuous alarm sampling         >> On
[2] Instant measurement on data alarm >> Off
[-] Exclude independent drivers       >> N/A
[4] Instant data output on data alarm >> Off
[5] Alarm messages                    >> Not used
[6] Alarm output port                 >> Repeating pulse 00:00:04
[7] Alarm output port active          >> Closed
[8] Alarm output parameters           >> Not used
>
```

#### 4.8.1 Continuous Alarm Sampling

Alarms situations are evaluated only when the data logger is awake to perform some task, typically a data log cycle. In case of long data log intervals you might want to alarm faster than per data log interval.

When you switch-on “Continuous alarm sampling” the data logger will wake-up at sample intervals as well (You can define sample intervals, which should be shorter than the regular data log intervals).

During sampling the logger performs a measurement cycle on parameters with configured alarm limits as well on those used in aggregation channels. Alarm conditions will be evaluated after the measurement cycle, but there will be no measurements stored to SD-card.

**Please take caution** when using this feature as it can cause extra ordinary power consumption.

*Example:*

*A user has chosen a data log (normal) interval of 10 minutes. The temperature-sensor must NOT get above 25 degrees. When the temperature reaches the limit, it takes the logger up to 10 minutes, before it gets into the alarming mode (with a faster alarming interval, of course). Now, the continuous alarming mode, changes this. When the sample interval of concerned sensor is set to i.e. 1 minute, the data logger will wake every minute, and take a measurement, but NOT store this on the SD Card. So, the user ends up with a normal data file, with every 10 minutes a new data record, and the system is responding within 1-minute max. on an alarm situation.*

### 4.8.2 Instant measurement on data alarm

In case of a data alarm raised outside the normal data log interval e.g. during continues alarm sampling of by a digital input the data logger can be instructed to perform a data log cycle to reflect the situation at alarm the moment of alarm raise and fall.

### 4.8.3 Exclude independent drivers

In case of switching on the above mentioned “instant measurement on data alarm” you might want to exclude measurements take by drivers deviating from the standard data log interval for the same reasons why you configured them on an independent data log interval.

### 4.8.4 Instant data output on data alarm

When using a host with his own advanced alarm handling and dispatching, you might want to feed the host system with recent measurement data before the regular reporting interval occurs, which can be a multitude of the regular data log interval. When switching on this feature, the data logger will perform a data output to all configured outputs as soon as the data logger gets in alarm mode or falls back to normal again.

### 4.8.5 Alarm messages

When the data logger enters or leaves alarm mode you can dispatch a message to a recipient by SMS, MQTT or e-Mail and indicate if they want to receive ‘System alarm messages’, ‘Data alarm messages’ or both. In case of e-mail or MQTT you need to enable or configure the corresponding output driver for the specific protocol. If you don't want to transmit log data as well, it's recommended to configure the driver to send just a “Heartbeat” once a day to avoid unnecessary payload and reduce power consumption.

#### 4.8.5.1 System alarm messages

Check this option if you want to send system alarms messages (like Deployment start, sensor timeouts (sensor could be broken), or TCP/FTP/HTTP server not available, etc.)

#### 4.8.5.2 Data alarm messages

Check this option if you want to send data alarms messages (e.g. when a measurement is outside its defined limits).

### 4.8.6 Alarm output port

```
Alarm output port

[0] Exit
[1] Disabled
[2] Continuous state
[3] One-time pulse
[4] Repeating pulse
>
```

The “Alarm output port” can be used to “switch on” or “trigger” an external device in case of an alarm situation by means of an “open collector output”. When in “Continuous state” the output will be activated as long as an alarm situation exists. If you only want to trigger an external device when a new alarm situation occurs, you can specify a “One-time pulse” with a certain pulse width in seconds. If you want to re-trigger the external device at every sample interval, while in alarm, choose “Repeating pulse”.

***Note**: When a logger is awake (e.g. during USB connection) and frequently sampling its sensors and inputs then “repeating pulse” may look similar to “continuous state”, due to overlapping pulses.*

### 4.8.7 Alarm output port active

The alarm output port is an “open collector output”, that can be seen as an electronic actuated switch. With this option you can indicated when the switch should be closed (make) or opened (break) on alarm.

### 4.8.8 Alarm parameters

```
Alarm output settings

[0] Exit
[1] Name                  >> Alarm output
[2] Register value        >> 0 Alarms; 0 Hour
[3] Daily register reset  >> Not used
[4] Alarm count           >> Not used
[5] Alarm state           >> Not used
[6] Total on-time         >> Not used
[7] Interval on-time      >> Not used
[R] Remove
[T] Test measurement
>
```

When using the alarm output port to trigger some device you might want to monitor metrics for predictive maintenance purpose (e.g. pump running hours).

#### 4.8.8.1 Register value & reset

This is the internal registry keeping count of the total number of raised alarm conditions an the total cumulated alarm durations. This registry can be cleared manually or on a daily basis at a given time.

#### 4.8.8.2 Alarm count

This parameter monitors the total number of raised alarm conditions, since the previous register reset.

#### 4.8.8.3 Alarm state

This parameter reflects the current alarm state (0=No alarm, 1=Alarm active).

#### 4.8.8.4 Total on-time

This parameter monitors the total cumulated hours of all alarm durations, since the previous register reset.

#### 4.8.8.5 Total Interval on-time

This parameter monitors the total cumulated hours of all alarm durations, since the previous data log interval.

## 4.9 Option boards

Option boards can be used to add additional inputs, outputs, ports to an ML-x17 data logger or to perform signal conditioning. You can mechanically stack up to 3 option boards, but only one board per category ID (CID, internal bus address). Some boards are having a switch or solder jumper to choose a different CID, so you can stack multiple similar boards. Converter boards (OC) don't have CID's and you can stack as many OC-boards as mechanical feasible. OC-boards don't make use of the internal control bus, but are wired to existing inputs/outputs/ports.

```
Option boards

[0] Exit
[1] Analog (0B1)           >> ML-0I-AD-80MV
[2] Analog (0B2)           >> ML-0I-AD-PT1000
[3] Barometric             >> None
[4] Humidity               >> None
[5] Serial port            >> ML-0I-C0M-SDI12
[5] Accessory port         >> ML-0A-RS232; 0B Camera
[6] Auxiliary inputs (0B1) >> ML-0I-AX-RH
[7] Auxiliary inputs (0B2) >> None
[8] Auxiliary output (0B1) >> ML-00-M0DBUS
[9] Auxiliary output (0B2) >> None
>
```

This menu is to indicate which option boards are mounted.

- ML-OI-AD (Input) boards can be used to add additional analog inputs.
- ML-OI-COM (Input) boards can be used to add an additional serial input port.
- ML-OA (Accessory) boards can be used to add a (secondary) accessory, e.g. to have a TFT-display as well as a camera on the same logger, or a dual camera to capture up/down stream/track pictures.
- ML-OI-AX (Auxiliary Input) boards can be used to add additional pre-processed sensors/inputs.
- ML-OO (Auxiliary Output) boards can be used to add additional outputs (e.g. open collectors).
- ML-OU-BLE to add a Bluetooth LE user interface for Android ydocTerminal.

### 4.9.1 ML-OI-AD-10V/20MA/80MV/2000MV (analog inputs)

#### 4.9.1.1 Calibration

To get the best accuracy from your analog inputs we recommend to perform a two point user calibration (See chapter: Analog inputs – Determine linear conversion). When a two point calibration is not feasible due to the nature of the input source, we recommend to do a 1 point calibration (See chapter: Analog inputs – Determine linear offset) . If a one point calibration is not feasible either, please perform a zero-calibration.

#### 4.9.1.2 Zero-calibration

1. Specify the correct theoretic range of your sensor (see chapter: Analog inputs - Parameter value at)
2. Disconnect your sensor
3. For single ended inputs connect the input to ground and for differential inputs short-cut the two differential inputs.
4. Perform a 1 point calibration (See chapter: Analog inputs – Determine linear offset). For the calibration point enter the expected physical value when the sensor output signal is zero. (e.g. 0W/m2 for a pyranometer).

### 4.9.2 ML-OI-AD-PT1000

This board provides two PT1000 temperature sensor inputs using a “bridge of Wheatstone” to accurately measure the resistance. Because sensing is resistance based any deviations in resistance in the loop will affect the reading. Cable resistance can be ruled out by using 3 wires instead of 2 and setting the board jumpers accordingly.

#### 4.9.2.1 Calibration

As temperature vs resistance is rather linear you can, after determining the difference between the real temperature and measured temperature, specify a temperature offset in the parameter settings.

### 4.9.3 ML-OI-AX (Auxiliary inputs)

These boards are equipped with their own processor running their own firmware to do specific signal processing or data acquisition on their inputs and providing the results as additional input channels to the data logger. Most boards are having a ‘Terminal’-interface for testing/configuration that can be accessed thru the ‘Terminal’-interface of the data logger. There are boards available to add additional counters, digital thermometers, humidity sensors etc.

### 4.9.4 ML-OO (Auxiliary outputs)

These boards are equipped with their own processor running their own firmware to do specific signal processing or data publishing on their outputs. A board can have up to 32 channels that can be regularly updated with parameters from the data logger, parameters with their data output flags" set to the concerned auxiliary output board. The order in which parameters are provided to the board follows the order of parameters as organized through the "Parameter overview"-menu of the data logger. There are boards available to add additional output switches, publishing parameters thru MODBUS/RTU, etc.

#### 4.9.4.1 Output interval

The output interval determines how often data is sent to the auxiliary output. You can set this interval freely. At each interval, the system will output the most recent sampled values.

***Note***: If a sensor's sample interval is different from the data log interval, the sensor will be powered and sampled whenever its timing aligns with the output interval.

## 4.10 Internal Sensors

The data logger is equipped with “internal sensors” in order to provide an insight in the status of the system. We strongly recommend using this feature, because it tells you whether a system is running OK or that a problem may arise in the near future (e.g. battery status)

### 4.10.1 Name

Name of this driver. Default is “internal” may be edited.

### 4.10.2 Sample interval

The sample interval of the internal driver. Default is same as data log interval. When a user like to have fresh data while using USB, he can adjust it to a higher rate. More on intervals see chapter: Principle_of_Operation

### 4.10.3 Battery Capacity

Used for keeping track of the remaining battery capacity. The users here define the capacity of the new battery placed.

### 4.10.4 Battery Replaced

The user must select this when he installs a new lithium battery. The calculations of the rest capacity will be reset (battery rest capacity 100%)

### 4.10.5 Rest Capacity

Remaining capacity of the lithium battery. In percentage of a full battery.

### 4.10.6 Rest power

Same parameter, but in mAh

*Attention: the battery-full parameters are only representative when used in lithium battery-only systems. When using a solar panel or external power they won’t work. Their readings are not true in this case. (this is because the battery is charged, whithout updating the calculations). This is no problem, because in these systems, the battery status is of minor importance. With solar systems, keep track of the minimum battery voltage instead of rest capacity or rest power.*

### 4.10.7 Processor Temperature

Temperature of internal processor, which is related to the ambient temperature. Because the data logger is very low power, and is only awake for a short time, the processor won’t get warm, and has practically the same temperature as the environment. You can use this sensor for approximation measurements. And also for diagnostic measurements.

### 4.10.8 Average Voltage

The voltage of the connected battery, averaged to get a real practical value

### 4.10.9 Max Voltage

Battery Voltage, maximum of interval

### 4.10.10 Min Voltage

Battery voltage, minimum value of interval. This parameter is very practical to use for diagnostic measurements. Particularly when the modem sends data, the battery voltage will drop, and must not get too low (2 Volts) This parameter is a nice quality figure for the status of the battery. When this value drops a lot, it is wise to exchange the battery, or charge it. With solar systems, in the dark months, this can be an issue, and this can be solved with the aid of good weather.

### 4.10.11 Average, Max, Min current

Same as above, but for current. The data-send sessions are clearly recognized by the use of these parameters.

### 4.10.12 Operating cycle

This virtual sensor measures the time that the data logger is awake, performing tasks. This gives also good diagnostic information. When i.e. the GSM signal is very weak, retries will occur, and the session will consume more time. This parameter provides information on the proper operation of a system. The Operating cycle is measured in seconds.

### 4.10.13 Free disk space

Shows the remaining disk space on the SD Card. In most cases there is sufficient space left.

## 4.11 Analog inputs

```
Analog input

[0] Exit
[1] Name                    >> Water height
[2] Sensor power switch     >> Disabled
[3] Sample interval         >> Data log interval
[4] Port mode               >> Port 1; 4-20 mA
[5] Parameter settings      >> Analog
[6] Parameter value at 4 mA >> 0 meter
[7] Parameter value at 20 mA >> 10 meter
[8] Determine linear conversion function (2 calibration points)
[9] Determine linear offset only (1 calibration point)
[A] Burst mode              >> Enabled
[B] Burst length            >> 2.5 sec (10 ms interval)
[C] Burst start             >> Value trigger; Deadband (+/-) 0.2 meter
[R] Remove
[T] Test measurement
>
```

### 4.11.1 Sensor power switch

Here, you can specify if you want to power the sensor from the 12V sensor power switch. You also need to define a 'Warm up'-time, which is the time needed by the sensor to produce a valid and stable value after applying power. More on the power switch see chapter: Power Switch

### 4.11.2 Sample interval

Interval on which the sensor is measured. Normally set to data log interval.

### 4.11.3 Port mode

The user can select the mode of the analog input (0..20 mA or 4..20 mA or Volts (volt port))

### 4.11.4 Parameter settings

See chapter: Parameters_Parameters

### 4.11.5 Parameter value at (min & max range)

Here, the user can specify the mapping between the physical values to be measured and the sensor output signal. Please specify the theoretical expected physical value at the minimum output signal of the sensor and the expected physical value at the maximum output signal (e.g. 4mA equals a level of 0m of water and 20mA equals to 10m. A linear interpolation will be performed on all values in between.

### 4.11.6 Determine linear conversion function (2 calibration points)

This is an automated function to help the user to set the right conversion function into the data logger. This feature will determine the same function as described above, but will use a physical readings instead of a manually entered figures. The user will be prompted to measure and enter physical values at 2 calibration points (e.g. 2 different sensor heights in a water column). This feature is convenient while performing a field calibration and no factory calibration data of the sensor is available.

### 4.11.7 Determine linear offset only (1 calibration point)

This features is an automated calibration function for the offset only, make sure you have first specified the theoretic correct range (see above: Parameter value at). The user will be prompted to measure and

enter the physical value at 1 calibration point. This procedure is a/o convenient when you have installed a level sensor, but you don't know it's exact well depth.

### 4.11.8 Burst-mode

One of the analog inputs can be used for digital and/or dead-band triggered 100Hz burst recording to capture up to 5 seconds of a signal transition in detail.

Burst recorded details can be used to predict the state of the monitored system, allowing for timely repairs or replacements before any catastrophic malfunction occurs. For example, this feature could be used to monitor the current passing through an electric motor that drives a railroad track switch, where the current profile could tell something about the state of the switching mechanism. Or monitor the shock waves send thru a water distribution network on opening or closing of valves and starting and stopping of pumps (water hammer).

With this feature enabled a low-power subsystem continuously samples an analog port with 100Hz (even when the data logger is a sleep). A burst recording can be triggered by one of the digital inputs or when the sampled signal surpasses a configured dead-band value. E.g. when you want to record a sudden change in water level you could specify a dead-band value of lets say 0.2m and as soon as the level, since the previous regular interval recoding, changes by 0.2m or more a burst recording of up to 5 seconds is started.

During burst recording all other data logger tasks are postponed. When burst-mode is enabled the data loggers quiescent current is increased by about 1mA and therefore not recommend for premium battery powered applications.

## 4.12 Digital inputs – Pulse counter

This section describes the operation of digital inputs, NOT to be confused with serial sensors, which also is a digital technique. A digital input is an input which measures the digital value of the input signal. This is a discrete value “0” or “1”. Examples of digital sensors are: switches, rain gauges and digital pulse devices. Digital inputs have the advantage that they are not that depended on their sample rate as analog sensors are. This is because they are interrupt-driven. There are several modes of operation on a digital input: counting pulses up to 10kHz, generating alarms, counting on-time, recording states or trigger actions.

Below, a menu of the settings of a digital pulse input is shown.

```
Digital pulse input

[0] Exit
[1] Name                    >> Digital pulse
[2] Sample interval         >> Data log interval
[3] Port mode               >> Port 1; Internal pull up
[4] Trigger on              >> Falling edge
[5] Register mode           >> Pulse (low frequency)
[6] Anti bounce filter      >> Off
[7] Units per pulse         >> 1
[8] Register value          >> 2103 pulses
[9] Daily register reset    >> Not used
[A] Log each counter change >> Off
[B] Counter (unit)          >> Counter
[C] Quantity (unit)         >> Not used
[D] Mean rate (unit/h)      >> Not used
[E] Max rate (unit/h)       >> Not used
[F] Min rate (unit/h)       >> Not used
[G] Pulse event (unit)      >> Not used
[H] Activity period         >> Water intake period
[I] Activity end detection  >> 00:00:04
[R] Remove
[T] Test measurement
>
```

### 4.12.1 Sample interval

This is exactly the same feature as in other menu’s, although there is something interesting to notice about the sample interval of a digital input (pulse sensor). With analog inputs, the sample rate defines how many times an input is sampled. So, if your sample interval is very low, you may miss some events. But with digital inputs, this is NOT the case. The digital inputs are interrupt driven, and respond immediately on an event. So, even when your sample rate for the digital input is very low, no events are missing. This is an important difference! So, what is the use of the sample interval in respect to digital inputs? It defines the update rate of the counter value connected to that port.

*Example: Suppose a user connects a wind speed sensor (anemometer) to the digital input, and this sensor generates a pulse with a frequency of 13 Hz. So, every second, the sensor has generated 13 pulses. Now the sample interval was set to 5 seconds. What happens is this: The data logger will count all pulse, and update the counter value every 5 seconds. So, the user has every 5 seconds a new data value.*

### 4.12.2 Port mode

This defines the electrical behaviour of the input, a **pull up** resistor for inputs switching to ground or a **pull down** resistor for inputs switching to max 5V)

#### 4.12.2.1 Pull up

In this mode, a resistor is connected (from the input) to the positive rail of the internal power supply. So, in order to create an event, the external device must pull the input to the ground. So, use this mode when you have a “negative switching” sensor. (a device that switches to ground and has no voltage to drive the port, e.g. a tipping bucket rain gauge)

#### 4.12.2.2 Pull down

In this mode, a resistor is connected (from the input) to the ground of the data logger. So, in order to create an event, the external device must drive the input with a positive voltage. So, use this mode when you have a “positive switching” device. (a device that drives the data logger’s input with a voltage)

### 4.12.3 Register mode

Here you choose if the logger should register into the flash memory each single pulse, or units, which can consist out of hundreds of pulses. Mostly you use Pulse, because most digital sensors like rain gauge are generating at max speed 1 or a few pulses per second. If you use for instance a flow meter, or anemometer that generates multiple pulses per second, and a completed unit in a few seconds, its high frequency and you should use unit as a register mode, or else the logger could not keep up the speed to write all the individual pulses to the flash memory.

### 4.12.4 Anti-bounce filter

If your pulse giver is not bounce free, you may suppress unwished counts by enabling a bounce-filter. Appling the bounce-filter will as a consequence limit the max count frequency to about 250Hz.

### 4.12.5 Units per pulse, or pulses per unit

Factor for transforming input pulses to real-life unit values. It depends on register mode setting.

### 4.12.6 Register value

Sometimes you like the counter to start from an existing value (perhaps the counter value of an electricity meter you just connect?) Here you enter the initial value the counter must start with. This value is updated when a pulse or unit (depends on register mode) is measured on the input, so each time you consult this menu, this setting will be accurate automatically.

### 4.12.7 Register Reset

User can select if he wants the counter value to be reset at midnight, or that the counter is always incrementing (32 bits).

### 4.12.8 Log each counter change

This parameter is logged independently of the data log interval and is recorded every time a counter change occurs. For an occasional change, the logged counter value will increase by one unit, or by more if multiple changes occur in a short time. At regular data logging intervals, the value of this parameter will reflect the last counter value.

***Attention***: Each recorded change creates its own data log record in the data storage. As a result, log files may contain multiple records or lines with the same timestamp if they happen to be recorded at the same moment. Additionally, in CSV files, such records or lines may contain empty fields.

### 4.12.9 Counter

This is the parameter, connected to the digital input. It behaves like a normal parameter (see Parameters)
*Note that the counter is not directly related to the amount of pulses at the input, the value is first scaled by the “pulses per unit” or “units per pulse” factor* Parameters

### 4.12.10 Quantity

The Quantity is the number of units occurred during the log interval. So, if you add up all Quantities, the value is the same as the counter. This is convenient with i.e. rain measurements. The counter gives you a total of all precipitation since installation (or last automatic reset), and the quantity gives you the value per log interval.

### 4.12.11 Mean-, Max- and Min Rate

This is the speed of increase of the counter value. When e.g. a water meter is connected. The counter contains the total pumped volume, and the rate contains the flow of the water. The unit is: units/h (in case of the water meter l/h).

### 4.12.12 Pulse event

This parameter is logged independently of the data log interval and is recorded every time a pulse event occurs. For an occasional pulse, the logged value will be 1 unit, but it may be greater than 1 if a pulse burst occurs. At the regular data logging intervals, the value of this parameter will be zero.

***Attention***: Each recorded event creates its own data log record in the data storage. As a result, log files may contain multiple records or lines with the same timestamp if they happen to be recorded at the same moment. Additionally, in CSV files, such records or lines may contain empty fields.

### 4.12.13 Activity period

This can be used to detect the length of e.g. a period of water flow, the time is measured from the first pulse till pulsing stops for a certain amount of time (end detection time). The start time will be logged with a 0 value and the end time (time of last pulse in the period) will be logged with the time past since the start time. Regular data logs in between will be logged with a value representing the time past between the data log timestamp and the start time.

## 4.13 Digital inputs - Alarm trigger

This feature uses the same electrical input as the previous (digital input: pulse counter), but it handles the signal differently. When the input is configured as an alarm input, instead of a pulse input, it has another menu and the purpose of the measurements is different. The difference between the pulse measurement and the alarming mode, is that with pulse measurement the change of the input signal is expected, and very often frequently, while in the alarming mode, a change in input signal is an exception. That's why you don't find functions like min, max, average. The Digital Alarm itself is not an Data Alarm, but puts the logger into alarming mode (using alarm intervals if applicable), and starts a direct measurement of all sensors trying for instance to catch data alarms, and the driver also directly logs it's changing state. If you also have put on direct data output on alarm (general settings), the logger will also start the output drivers (at both rising and falling levels). Below the menu-settings of an alarm-input is shown.

```
Digital alarm input

[0] Exit
[1] Name                    >> Digital alarm
[2] Port mode               >> Port 2: Internal pull up
[3] Digital input           >> High active
[4] Trigger delay           >> 00:00:01
[5] Register value          >> 13 Alarms
[6] Alarm                   >> Alarm
[7] Counter                 >> Counter
[R] Remove
[T] Test measurement
>
```

### 4.13.1 Trigger delay

The amount of time the data logger waits before changing to the alarm state (when the input condition is true). When the logger measures an alarming input-condition it enters the pre-alarming mode. This mode will change into alarming mode when the trigger delay has finished. If the alarming condition disappears during this time, the logger will NOT enter alarming mode, but will continue in normal mode.

### 4.13.2 Register value

Holds the amount of alarms that occurred. The user can edit this value.

## 4.14 Digital inputs – Action trigger

This feature uses the same electrical input as the previous digital sensors, but it handles the signal differently. This driver can be used to initiate an instantaneous action.

```
Digital trigger input

[0] Exit
[1] Name           >> Digital trigger
[2] Port mode      >> Port 3: Internal pull up
[3] Trigger on     >> Rising edge
[4] Trigger action >> Data output
[5] Register value >> 170 triggers
[6] Triggers       >> Triggers
[R] Remove
[T] Test measurement
>
```

### 4.14.1 Trigger on

The edge of the input signal where triggering must take place: Rising Edge ( from low to high) or Falling Edge ( from high to low).

### 4.14.2 Trigger action

The following actions can be triggered:

1. Perform all the tasks below (default)
2. Take measurements
3. Take a measurements and create a data log record
4. Perform a data transfer
5. Take a picture (if a camera is connected and camera triggering is enabled).

## 4.15 Digital inputs - State input / On-time meter

This feature uses the same electrical input as the previous digital sensors, but it handles the signal differently. This driver can be used to log state transitions (on-off and off-on) and to count the on-time (active time) of the input.

```
Digital state input

[0] Exit
[1] Name             >> Digital state
[2] Port mode        >> Port 2: Internal pull up
[3] Digital input    >> Low active
[4] Register value   >> 0 hour
[5] Register reset   >> Not used
[6] Total on-time    >> Total on-time
[7] Interval on-time >> Not used
[8] Input state      >> Not used
[R] Remove
[T] Test measurement
>
```

### 4.15.1 Total On-time

Is a meter counting the total on-time (active time) of the digital input in hours. This counter could be useful in maintenance application (e.g. monitoring pump operating hours). The on-time is recorded at the general log interval.

### 4.15.2 Register value

The latest know total on-time, you can set to another value if required (e.g. to zero after replacing a pump).

### 4.15.3 Register reset

The total on-time meter can run for ever or automatically be set to 0 at particular time at a daily bases.

### 4.15.4 Interval On-time

Like the Total On-time, but counts the On-time over a log interval only. By applying a factor you can record this interval in minutes or seconds if you like.

### 4.15.5 Input state

Use to record the state of the input at every log interval and at every state transition.

## 4.16 Network Signal Sensor

This is an internal sensor that measures the quality of the 2G/3G network signal. This is a very important status measurement, because it is directly related to the connectivity of the data logger. We advise to include this sensor in your configuration, and to monitor this parameter regularly.

```
Network signal sensor

[0] Exit
[1] Name                  >> Network signal
[2] Independent data log  >> On
[3] Sample log interval   >> 06:00:00
[4] Signal bars (0-5)     >> Signal
[5] Signal quality (0-7)  >> Not used
[6] Signal (dB)           >> Not used
[7] Signal (%)            >> Not used
[8] Signal indication     >> Not used
[9] Bit error ratio       >> Not used
[R] Remove
[T] Signal test           >> Passed
>
```

### 4.16.1 Independent data log

The interval on which this sensor is sampled. This sensor is a virtual sensor, in fact the information is obtained from the internal modem. Because the modem draws a lot of energy, it is not a good idea to use these sensors along with the other sensors at the same interval. It is much more economical to use this sensor only at the send interval. Because the modem has to work on that interval anyway, so you can get this info without losing any extra energy. The name “independent data log” is derived from this situation. Set it according to your needs, but preferably use the send interval as setting.

### 4.16.2 Signal bars

A comprehensive presentation of the quality of the sensor (the network coverage at your installation site) 0 bars = no reception, it will not work. The system will work from 1 bar and above. Try to setup your (external?) antenna for optimum reception. 3, or more is a very good reception.

### 4.16.3 Signal dB/%/indication/bit error rate

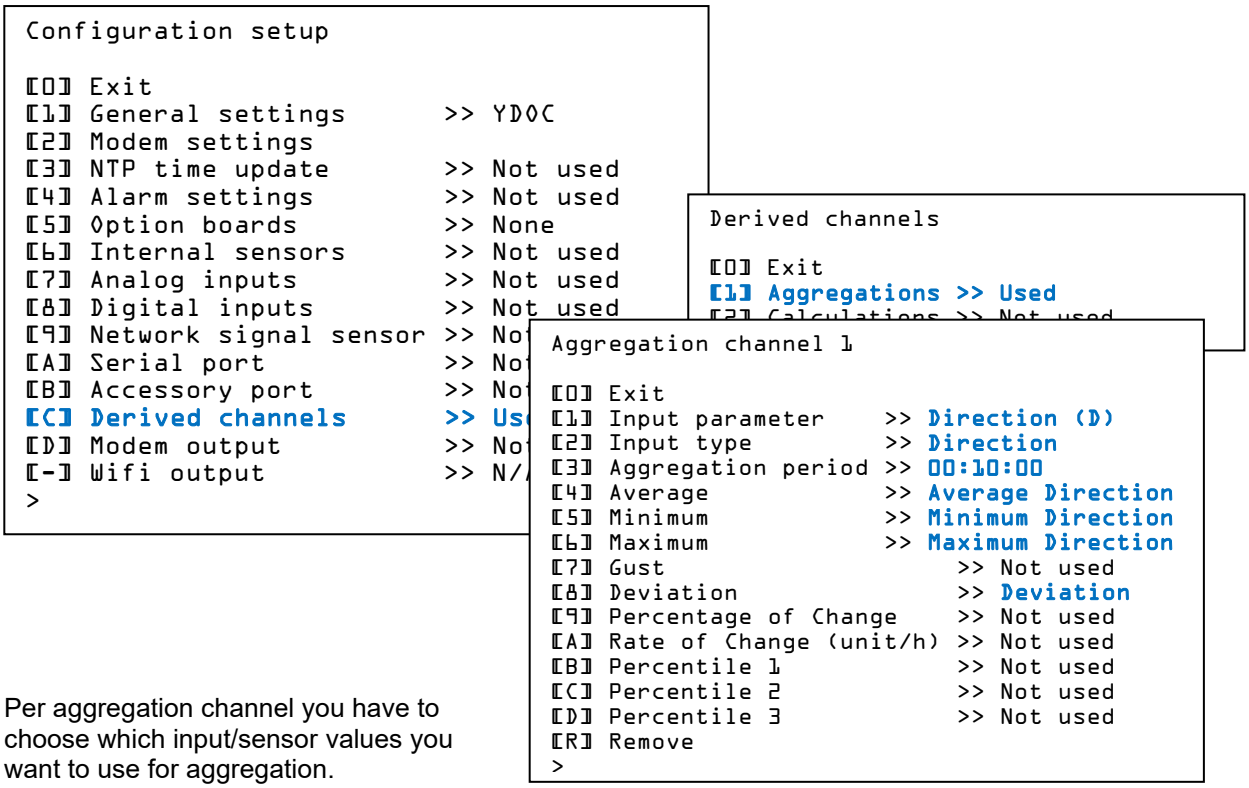
A technical presentation of the quality of the signal. These figures are technical and harder to comprehend. (example: a signal of -34 dB is twice as weak as a signal of -31 dB, dB is a logarithmic presentation. Also, the % presentation is NOT linear.) We advise to use the signal bars.

## 4.17 Aggregation Channels

The logger is equipped with 8 aggregation channels which can be used to record aggregated values (average, minimum, maximum, gust and standard deviation) calculated over a period of your choosing. The number of samples taken into account during the aggregation period is depending on the period duration and the 'Sample interval' as configured for the used input parameter/sensor.

Each channel has an aggregation buffer of 600 values. So, a 10-minute aggregation period (600 seconds) can be sampled with 1Hz, longer aggregation periods e.g. 20 minutes can be sampled with 1Hz but will cause the use of intermediate averages. As 1Hz is the highest possible aggregation sample rate, shorter aggregation periods will contain less samples (e.g. 120 samples for a 2-minute aggregation interval). An aggregation interval can be shorter or longer than a data log interval, so it's possible to record a 10 minute (rolling) average every 2 minutes or record the last 2-minute average every 10 minutes.

With a 1Hz sample rate the logger needs to wake-up and take a sample every second, to avoid draining the batteries the used sensors should require a short warm-up (a fraction of the sample interval) consume as little power as possible preferable none (like a reed switch based anemometer), self-powered (like a pyranometer) or negligible (like a 20KOhm wind vane potentiometer). The power draw is about reversed proportional to the sample interval, a twice longer interval will draw about 50% less power.




Per aggregation channel you have to choose which input/sensor values you want to use for aggregation.

For input type you can choose 'Scalar', 'Direction' or 'dB'.

A normal 'Scalar' value like a windspeed can be averaged by accumulation and division. A direction (0..360 deg) like wind direction needs to be dissolved in two x and y components first and their average combined back into a direction again.

E.g. a northern wind flapping between 340 and 10 degrees will result in an average northern wind of 355 instead of a faulty 175 degrees southern wind.

Some values (like Noise) are expressed in dB and as dB is a logarithmic value the individual values have to be linearized first before averaging (e.g. the average of 10dB +20dB is not 2 but 17.4dB).

Choose an aggregation period, a commonly used wind speed aggregation period is 10 minutes. Each aggregation buffer can hold max 600 values, so values for a 10 minute aggregation can be sampled with 1Hz. If you want a 20 min aggregation sampled with 1Hz, then 2 second averages will be used.

#### 4.17.1.1 Average, minimum, maximum, gust, standard deviation & change

Choose which aggregation results you want to record, being average, minimum, maximum gust, standard deviation, gust and change. A gust is the highest 3 sample average occurred in an aggregation period. Percentage of change is the change in percentage relative the oldest value in the aggregation period. Rate of Change is the difference between the youngest and oldest value over the aggregation period and scaled to a rate in unit per hour.

#### 4.17.1.2 Percentiles

It's also possible to define up to 3 different percentiles in ascending or descending order. E.g. to determine the background noise in a certain area you can sample the noise in dBA and take the 95% percentile of all sample noise values in descending order. The value returned (LA95) will be the noise level not exceeded by 95% of the values. The noise peak load (LA10) can be determined by taking the 10% percentile in descending order. A 50% percentile will give the median.

## 4.18 Calculation Channels

Besides the different Physical channels, the data logger contains, it also has a number of "calculation channels". These are very powerful, virtual channels. i.e. they are not real physical input channels, but virtual channels to hold i.e. an in between result, for a complex calculation. It allows the user to perform complex calculations in order to transform the electrical signal from the input to a real-life value. It enables the user the connection of a very wide range of sensors, including non-linear ones. All kind of mathematical functions can be used inside a calculated channel. The section covers how to use calculated channels and gives you a real-life example. Calculated channels are found in the configuration setup (below)

```
Configuration setup

[0] Exit
[1] General settings       >> YDOC
[2] Modem settings
[3] NTP time update        >> Not used
[4] Alarm settings         >> Not used
[5] Option boards          >> None
[6] Internal sensors       >> Not used
[7] Analog inputs          >> Not used
[8] Digital inputs         >> Not used
[9] Network signal sensor  >> Not used
[A] Serial port            >> Not used
[B] Accessory port         >> Not used
[C] Derived channels       >> Used
[D] Modem output           >> Not used
[-] Wifi output            >> N/A
>
```

```
Derived channels

[0] Exit
[1] Aggregations >> Used
[2] Calculations >> Used
>
```

A calculated channel can be used to derive a meaningful engineering value from sensed input values using mathematical operators, parentheses and functions (a/o cos, sin, atan2, ln, sqrt).

### 4.18.1 Syntax

Within the data logger each sensed/calculated parameter/channel has a parameter code chosen by the user (e.g. TEMPC for a temperature in Celsius, **Attention**: codes should not contain any spaces or signs). These parameter codes can be used in equations by preceding them with a colon (:). When calculating, the parameter codes will be substituted with the last recent measured values. It's possible to use multiple parameters in one equation. The order of arithmetic operations is the same as when using a calculator, but parentheses can be used freely to group operations to make it more clear.

### 4.18.2 Attributes

A parameter code can be trailed with an attribute code to use the value of the concerned attribute instead of the last measured value of a parameter (e.g. configured alarm limits).

#### 4.18.2.1 Alarm limits

The attribute codes for the various alarm limits are .LOLO, .LO, .HI, and .HIHI. The alarm hysteresis is provided by the .HYS attribute. The .ALR attribute can be used to check the alarm status of a parameter, so you don't have to evaluate the hysteresis and various alarm levels to determine if a particular parameter is in alarm yourself. The attribute value is assembled from 4 bits where bit0=LOLO, bit1=LO, bit2=HI and bit3=HIHI indicates the various alarm limit violations.

#### 4.18.2.2 Initial value

This is the value assigned to a parameter at re-boot. The initial value can be used with .IV attribute as a literal/reference value in your calculations. You can manipulate the outcome of your calculations remotely by sending a configuration snippet altering the initial value (see: See chapter configuration snippets).

#### 4.18.2.3 Age

When an input value is acquired successfully it will be timestamped, you can use the .AGE attribute to check the age of a value in seconds since the last valid acquisition (this attribute can be handy when occasionally receiving input parameters from remote sources through MQTT or FTP).

#### 4.18.2.4 Range

Analog inputs are having a user defined/calibrated slope to convert analog input values to engineering values. You can use the minimum and maximum engineering values of the slope (e.g. to calculate a percentage from a water level) in your calculations by trailing a parameter code with the attribute codes .MIN or .MAX (e.g. :AIN1.MAX).

### 4.18.3 Functions

Besides using simple operators, mathematical functions can be used as well. The syntax of a function is: <function_name>(<arguments>). All supported functions are listed after the next examples.

Example: sin (x) (calculates the sinus of x)

### 4.18.4 Example 1

A measured tank liquid level could be compensated with temperature and passed thru an equation to calculate the contents of the tank in kg.

Within the data logger each sensed parameter/channel has a parameter code chosen by the user (e.g. TEMPC for a temperature in Celsius). These parameter codes can be used in equations by preceding

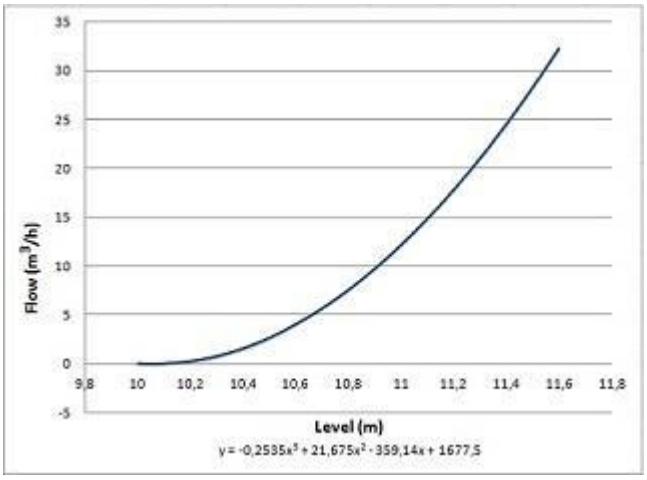
them with a colon (:). When calculating the parameter codes will be substituted with the last recent measured values. You could enter the equation **round((:TEMPC*1.8+32)*10)/10** to convert a temperature in Celsius to Fahrenheit and round the result to one decimal.

```
Channel 1

[0] Exit
[1] Parameter >> Temp F
[2] Formula >> round((:TEMPC*1.8+32)*10)/10
[-] Result >> 90.1
[R] Remove
>
```

### 4.18.5 Example 2

Assume we measure a river height in m (parameter code LEVEL) and want to convert it to a flow by using the pre-determined polynomial: (see graph below) $y=-0.253x^3 + 21.675x^2 – 359.14x + 1677.5$
Then the formula should be entered as follows: -0.2535*pow(:LEVEL;3) + 21.675*pow(:LEVEL;2) - 359.14*:LEVEL + 1677.5




### 4.18.6 Supported mathematical functions with one argument:

#### 4.18.6.1 abs(x)

returns the absolute value of x
*Example: abs(-10.7) returns 10.7*

#### 4.18.6.2 sqrt(x)

returns the square root of x

#### 4.18.6.3 ln(x)

returns the natural logarithm of x

#### 4.18.6.4 exp(x)

calculates the exponent of e to x

#### 4.18.6.5 log(x)

returns the base 10 logarithm of x

#### 4.18.6.6 sin(radians)
returns the sinus of radians

#### 4.18.6.7 cos(radians)
returns the cosine of radians

#### 4.18.6.8 tan(radians)
returns the tangent of radians

#### 4.18.6.9 asin(x)
returns the arc sinus of x in radians

#### 4.18.6.10 acos(x)
returns the arc cosine of x in radians

#### 4.18.6.11 atan(x)
returns the arc tangent of x in radians

#### 4.18.6.12 torad(degrees)
converts degrees to radians

#### 4.18.6.13 todeg(radians)
converts radians to degrees

#### 4.18.6.14 floor(x)
returns the largest integer not greater than x, e*xample: floor(10.7 returns 10*

#### 4.18.6.15 ceil(x)
returns the smallest integer not less than x, example: ceil(10.3) returns 11

#### 4.18.6.16 round(x)
rounds to the nearest integer

### 4.18.7 Supported mathematical functions with multiple arguments:

#### 4.18.7.1 atan2(x;y)
return the arc tangent of x/y

#### 4.18.7.2 mod(x;y)
return the reminder of x/y

#### 4.18.7.3 pow(x;y)
returns x to the power of y

#### 4.18.7.4 clip(x;min;max)
returns x clipped between min & max

*Example: clip(5;0;100) returns 5*
*Example: clip(-10;0;100) returns 0*
*Synonym: If(x<min) return min; Else if(x>max) return max; Else return x*

### 4.18.8 Supported comparisons with 2 arguments:

#### 4.18.8.1 lt(x;y)

returns x if x smaller than y else return y
*Example: lt(10;11) returns 10; Synonym: If (x<y) return x; else return y*

#### 4.18.8.2 le(x;y)

returns x if x smaller or equal than y else return y
*Example: le(10;11) returns 10; Synonym: If (x<=y) return x; else return y*

#### 4.18.8.3 gt(x;y)

returns x if x greater than y else return y
*Example: gt(10;11) returns 11; Synonym: If (x>y) return x; else return y*

#### 4.18.8.4 ge(x;y)

returns x if x greater or equal than y else return y
*Example: ge(10;11) returns 11; Synonym: If (x>=y) return x; else return y*

### 4.18.9 Supported comparisons with 4 arguments:

#### 4.18.9.1 eq(x;y;q;p)

returns q if x is equal to y else return p
*Example: eq(10;11;0;3) returns 0; Synonym: If (x==y) return y; else return p*

#### 4.18.9.2 lt(x;y;q;p)

returns q if x smaller than y else return p
*Example: lt(1;2;3;4) returns 3; synonym: if(x<y) return q; else return p*

#### 4.18.9.3 le(x;y;q;p)

returns q if x smaller or equal than y else return p
*Example: le(1;2;3;4) returns 3; synonym: if(x<=y) return q; else return p*

#### 4.18.9.4 gt(x;y;q;p)

returns q if x greater than y else return p
*Example: gt(1;2;3;4) returns 4; synonym: if(x>y) return q; else return p*

#### 4.18.9.5 ge(x;y;q;p)

returns q if x greater or equal to y else return p
*Example: ge(1;2;3;4) returns 4; synonym: if(x>=y) return q; else return p*

### 4.18.10 Supported bitwise functions with 2 arguments

These functions accept two arguments which are both converted to discrete unsigned values on which bitwise operations are performed and the result returned as a number. These functions can be handy when controlling ML-OO-SW output switches (all bits 0 will cause a break, one bit set a make).

#### 4.18.10.1 and(x;y)

returns the bitwise result of x AND y
*Example: and(1;3) returns 1;*

#### 4.18.10.2 or(x;y)

returns the bitwise result of x OR y
*Example: or(1;3) returns 3;*

#### 4.18.10.3 xor(x;y)

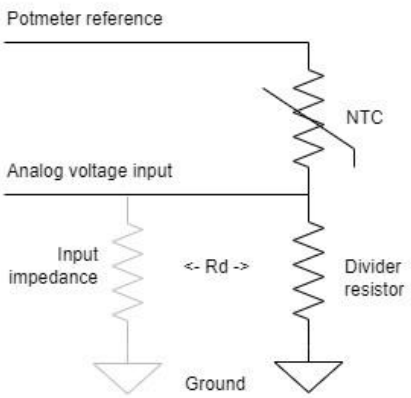
returns the bitwise result of x XOR y
*Example: xor(1;3) returns 2;*

### 4.18.11 Supported empirical formulas

#### 4.18.11.1 ntc(Vin[;R0[;Rd[;B[;D]]]])

returns the temperature calculated from the voltage drop measured across a divider resistance in a voltage divider configuration with an NTC thermistor.

The thermistor should be connected between the 3.3V potentiometer reference output and an analog voltage input, preferably the potentiometer input. Set the potmeter parmeter to 0 .. 3.3V instead of 0..100%




A divider resistor should be connected between the analog voltage input and ground. It is recommended that this resistor has approximately the same resistance as the NTC thermistor.

The formula accepts 1 to 5 arguments.

Vin, the measured voltage drop across the divider resistance.
R0, the nominal resistance of the NTC (default 10kohm).
Rd, the value of the divider resistance (default 10kohm).
B, the so called B-Value of the used thermistor (default 3950).
D, the self heating dissipation constant of the used thermistor in mW/°C (default 2).

**Tips to increase the accuracy of temperature measurements:**

1. Choose a high-accuracy NTC thermistor, ideally with 0.1% tolerance.
2. Enter the exact divider resistance: Calculate the exact divider resistance, taking into account the input impedance of the system. If the input impedance of the measurement circuit is high (e.g., the potentiometer input), it can often be neglected. However, this is not true for lower impedance analog inputs, which need to be considered. To measure the divider resistance accurately, temporarily replace the NTC with a precise resistor (0.1% tolerance, labeled R1) and measure the voltage across the divider resistor. The formula to calculate divider resistance Rd is: *(VRd * (R1 + 150Ω)) / (3.3V – VRd)* where 150 Ω is the internal protection resistor in series with the reference voltage. The voltage can be measured with the data logger [T] test function.
3. Optionally optimize the divider resistance: Select a divider resistor value so that the divider resistance corresponds to the value of the NTC thermistor at the midpoint of the temperature range of interest.
4. Enter the correct B-Value as provided by the manufacturer.
5. Enter the correct dissipation factor as provided by the manufacturer.

#### 4.18.11.2 wc(t;v)

returns the wind chill factor calculated from ambient temperature in °C and wind speed in m/s

#### 4.18.11.3 dp(t;rh)

returns the dewpoint calculated from ambient temperature in °C and relative humidity in %

### 4.18.12 Various supportive functions:

#### 4.18.12.1 pi

returns 3.1415926

#### 4.18.12.2 time

returns seconds since midnight local time. *Example: 43200 is 12:00:00 and 86399 is 23:59:59*

## 4.19 Serial Port

The serial port of your data logger provides different forms of serial communications. This chapter describes the details of this communication.

### 4.19.1 RS232

This mode allows to connect to RS232 sensors, see RS232 for details of the electrical properties. RS232
Below a menu-settings screen is shown, for an RS232 sensor.

```
RS232 sensors

[0] Exit
[1] Generic MODBUS/RTU
[2] Generic Serial
[3] Generic NMEA
>
```

### 4.19.2 Generic Modbus

This is a versatile protocol that can be used with RS232, but you'll encounter it more frequent with RS485. Below the settings menu for Modbus is shown:

```
MODBUS sensor

[0] Exit
[1] Name                  >> MODBUS
[2] Port settings         >> RS485 8N1; 19200 Baud; Address 1
[3] Sensor power          >> Disabled
[4] Sample interval       >> Data log interval
[5] Protocol type         >> RTU
[6] Measurement commands  >> Not used
[7] Register type         >> Holding
[8] Data type             >> WORD (unsigned 2 bytes)
[9] Data start register   >> 62592 (462593)
[A] Parameter 1           >> Water temperature
[B] Parameter 2           >> Conductivity
[C] Parameter 3           >> Not used
[D] Parameter 4           >> Not used
[E] Parameter 5           >> Not used
[F] Parameter 6           >> Not used
[G] Parameter 7           >> Not used
[H] Parameter 8           >> Not used
[I] Parameter 9           >> Not used
[M] More parameters       >> Not used
[R] Remove
[T] Test measurement
>
```

#### 4.19.2.1 Port Settings

The user can select the right baud rate and Modbus Address of his sensor here. The address of the sensor is very often adjustable, with the aid of a configuration tool of the sensor's manufacturer. So, the user can change this according to its need. You can build a multi-sensor network, by using unique addresses.

#### 4.19.2.2 Protocol type

Choose the MODBUS protocol type: RTU or ASCII

#### 4.19.2.3 Register type

Consult the manual of your sensor for this. 3 types are supported:

- Bit register
- Holding register (default register)
- Input register

#### 4.19.2.4 Register Start address

MODBUS is a protocol to read from and write data to a memory map of a device, which is constructed of a list of 16 bit registers. How this memory map is formatted is up to the manufacturer of the device. Please specify the register in the memory map where the first parameter of your interest starts.

#### 4.19.2.5 Data Type

The type of the data that is stored in the sensor @ the starting address. See table below.

| Datatype | Size | Signed | Remarks |
|---|---|---|---|
| **WORD** | 1 register (2 bytes) | unsigned | |
| **Short** | 1 register (2 bytes) | signed | |
| **DWORD** | 2 register (4 bytes) | unsigned | |
| **Integer** | 2 register (4 bytes) | signed | |
| **Float** | 2 register (4 bytes) | | IEEE754 |
| **Double** | 4 register (8 bytes) | | IEEE754 |

All multi register data types are also available for reversed logic (word swapping).
Consult the manual of your device to check with data type applies.

#### 4.19.2.6 Parameters

Parameter 1 starts at the chosen 'Start register' and the next 1, 2 or 4 registers further, depending on the chosen 'Data type'. You can read-out up to 20 parameters. However if you are not interested in some positions or if do not appear in consecutive register order, you can skip these positions. If you want to read-out more parameters, you can define another instance of the MODBUS-driver. You can like any other parameter change its name, unit and more see: Chapter 4.2 Parameters_Parameters

#### 4.19.2.7 Timing

The logger expects the sensor to respond within 1 second after issuing a request an will try the request up to 3 times if not.

#### 4.19.2.8 Measurement commands

Some sensors do not automatically start sampling after power on or might require a maintenance cycle now and then (e.g. wiping a lens of a turbidity sensor). If necessary measurement and maintenance commands can be configured with the menu shown below:

```
MODBUS Measurement commands

[0] Exit
[1] Start maintenance command >> Register 2 (40003); Value 1
[2] Maintenance interval      >> 1:24 Samples
[3] Maintenance ready status  >> Register 3 (30004); Value 1
[4] Maintenance ready within  >> Not used
[5] Start measurement command >> Register 0 (40001); Value 1
[6] Measurement ready status  >> Register 1 (30002); Value 1
[7] Measurement ready within  >> Not used
>
```

#### 4.19.2.9 Start maintenance command

Specify which value should be written to which register to instruct the device to perform a maintenance cycle. Leaving both blank or 0 will switch-off the command.

***Note***: Modbus provides two function codes for writing holding registers: Function code 6 (0x06) to write a single 16-bit register, and Function code 16 (0x10) to write one or more 16-bit registers. Please select the function code supported by your device.

#### 4.19.2.10 Maintenance interval

If it's not necessary to perform a maintenance cycle at every measurement sample, then please specify the maintenance interval in number of samples, this will possible reduced power consumption and wear of the maintenance device.

#### 4.19.2.11 Maintenance ready status

If available specify the register and its expected value indicating when a maintenance cycle is concluded.

#### 4.19.2.12 Maintenance ready after

If the device does not support a maintenance ready status register, please indicate how long a maintenance cycle takes to complete.

#### 4.19.2.13 Start measurement command

Specify which value should be written to which register to instruct the device to take a sample. Leaving both blank or 0 will switch-off the command.

***Note***: Modbus provides two function codes for writing holding registers: Function code 6 (0x06) to write a single 16-bit register, and Function code 16 (0x10) to write one or more 16-bit registers. Please select the function code supported by your device.

#### 4.19.2.14 Measurement ready status

If available specify the register and its expected value indicating when a sample is concluded.

#### 4.19.2.15 Measurement ready after

If the device does not support a measurement ready status register, please indicate how long a measurement sample takes to complete.

### 4.19.3 Generic ASCII

This is based upon a serial device that autonomous outputs sentences of data, like an GPS device. The user can capture this data, split it into various parameters, by specifying a separator character and by specifying the start and stop characters of a data sentence. If a serial sensor outputs multiple different sentences, like an NMEA-0183 device, a parameter can be linked to a matching sentence by supplying a sentence filter text.

```
[0] Exit
[1] Name                        >> Serial
[2] Port settings               >> RS232 8N1; 9600 Baud
[3] Sensor power switch         >> Enabled; Warm up 00:00:01
[4] Sample interval             >> Data log interval
[5] Maximum wait time           >> 00:00:10
[6] Log raw data string         >> Off
[7] Decimal symbol              >> '.'
[8] Seperator character         >> ','
[9] Start character             >> '$'
[A] Start character 2           >> (None)
[B] Stop character              >> (CR)
[C] Output request string       >>
[D] Output request terminator   >> (CR)
[P] Parameters                  >> 3
[R] Remove
[T] Test measurement
>
```

#### 4.19.3.1 Log raw data String

The user can select this to log the whole data string, including separator char's, checksums etc. This is convenient when debugging / testing a new sensor.

#### 4.19.3.2 Decimal symbol

The user can select which decimal symbol is used by the sensor, being a dot or a comma.

#### 4.19.3.3 Separator Character

Character which separates the various data fields (numbers). Very often a space (0x20) is used. This is the default-setting. A comma, semi-colon or tab-character are commonly in use as well.

#### 4.19.3.4 Start/Stop character

These define with which character a data sentence starts and terminates. E.g. an NMEA sentence starts with a '$' and terminates with a carriage return (CR).

#### 4.19.3.5 Output request

Most serial probes start transmitting their data sentences at regular intervals as soon as they got powered. Some serial probes output their data sentences on request only and for such cases you can specify an optional "Output request string" and/or "Output request terminator" character. The request string will send first to the probe directly followed by the request terminator (e.g. a carriage return or line feed).

#### 4.19.3.6 Parameters

These are the values you want to get recorded from the data sentence(s).

Each parameter has 3 properties.

```
Serial parameter input settings

[0] Exit
[1] Sentence filter    >> MWV
[2] Field position     >> 1
[3] Field type         >> Numeric
[4] Parameter settings >> Wind direction
>
```

#### 4.19.3.7 Sentence filter

If a serial device can output multiple different sentences you can distinguish them from each other by specifying a sentence filter text to capture your parameter from the correct matching sentence. You can leave the filter 'blank', if the device outputs one type of sentence only.

#### 4.19.3.8 Field position

A captured sentence is split up in multiple fields based on the chosen separator character. The first field starts has position number 0.

#### 4.19.3.9 Field type

The format of the field vale, this can be a 'Numeric' ASCII presentation (e.g. -123.45), a 'Hexadecimal' ASCII presentation (e.g. 00FF) or a single character. Note: a single character will be recorded as a its ASCII numeric value (e.g. 'A' will be recorded as 65).

#### 4.19.3.10 Example:

An NMEA-0183 compliant weather station could output the wind direction and speed with the following sentence. This sentence contains 6 comma separated fields.

$WIMWV,190.0,R,11.0,N,A*1B

```
Serial parameter input settings

[0] Exit
[1] Sentence filter    >> MWV
[2] Field position     >> 1
[3] Field type         >> Numeric
[4] Parameter settings >> Wind direction
>
```

The 'Wind direction' value is found at position 1 in the sentence and the sentence can be distinguished from other NMEA sentence by specifying 'MWV' as the 'Sentence filter'.

### 4.19.4 Generic NMEA

Similar to 'Generic ASCII'-driver, but you don't have to specify start/stop and separation characters as they are defined by the NMEA-0183 protocol. Additionally, the 'Generic NMEA' driver supports the NMEA-0183 checksum verification.

#### 4.19.4.1 GPS

$GPGGA,172814.0,3723.46587704,N,12202.26957864,W,2,6,1.2,18.893,M,-25.669,M,2.0,0031*4F

When using the 'Generic NMEA' driver instead of the 'GPS' driver to input GPS coordinates you might need to do a conversion from dddmm.mmmm N/S/E/W format to +/- ddd.dddddd format. This can be accomplished by using a calculated channel for both the latitude and the longitude.

Latitude formula:

```
(floor(:GGA2/100)+((:GGA2/100)-floor(:GGA2/100))/0.6)*eq(:GGA3;83;-1;1)
```

Longitude formula:

```
(floor(:GGA4/100)+((:GGA4/100)-floor(:GGA4/100))/0.6)*eq(:GGA5;87;-1;1)
```

Assuming: GGA2 is the latitude (2nd field position), GGA3 is the direction (3th field, 83='S')
GGA4 is the longitude (4th field position), GGA5 is the direction (5th field, 87='E')

### 4.19.5 RS485

Similar as the RS232 port, but with RS-485 levels, see: Generic_Modbus/RTU

### 4.19.6 SDI-12

Used for SDI-12 communications, see below. For generic information on SDI-12 see: SDI-12_SDI-12

```
SDI12 sensor

[0] Exit
[1] Name                >> SDI12
[2] Port settings       >> SDI12; 1200 Baud; Address 0
[3] Sensor power switch >> Enabled; Warm up 00:00:01
[4] Sample interval     >> Data log interval
[5] Measurement command >> 0C!
[6] Parameter 1         >> Par1
[7] Parameter 2         >> Not used
[8] Parameter 3         >> Not used
[9] Parameter 4         >> Not used
[A] Parameter 5         >> Not used
[B] Parameter 6         >> Not used
[C] Parameter 7         >> Not used
[D] Parameter 8         >> Not used
[E] Parameter 9         >> Not used
[M] More parameters     >> Not used
[R] Remove
[T] Test measurement
>
```

#### 4.19.6.1 Measurement command

The user can choose between concurrent measurements (c), or non-concurrent (m). When the sensor supports concurrent measurements, we strongly advise to use it. Because it saves a lot of energy. (all sensors are measuring simultaneously). The command number is asked by the firmware, for retrieving the right data value. For more info consult your sensors manual. After selecting the right measurement command, the command is displayed in the menu, so the user can check this against the sensors manual

#### 4.19.6.2 Parameters

The various data values that were retrieved from the SDI-12 sensor, can be stored into different parameters. They behave like normal generic parameters. See more info: Parameters

### 4.19.7 Serial port tunnel

If you have connected a serial device, but it's not used to record measurements, then you can access this device thru a serial port tunnel (e.g. thru the COM-port terminal in the Maintenance-menu). You can configure the parameter of the serial device thru the screens below:

```
COM port tunnel settings

[0] Exit
[1] Name              >> Serial port tunnel
[2] Port settings     >> RS232 8N1; 9600 Baud
[3] Sensor power      >> Enabled; Warm up 00:00:01
[R] Remove
>
```

#### 4.19.7.1 Port settings

Specify it the device is connected by RS485 or RS232, along with its serial communication characteristics (baud rate & parity).

```
Port settings

[0] Exit
[1] Mode      >> RS232 8N1
[2] Baud rate >> 9600
[-] Address   >> Not used
>
```

#### 4.19.7.2 Sensor power

When the device should be powered from the 12V sensor power switch, please specify how long it takes before the device is capable to communicate meaningful.

## 4.20 Accessory port

This is a serial port which is used for supporting special serial devices, called accessory port modules. The accessory port is only equipped on the MLx17-ADS models. Below a list of supported accessories is shown.

```
Accessory port

[0] Exit
[1] Display
[2] Camera
[3] GPS
[4] ASCII output
[5] Radio
[6] Iridium satellite
[7] Swarm satellite
[8] Accessory port tunnel
>
```

These are the accessory modules that can be connected to the accessory-port. Each option is described below. For detailed information consult the manual of the accessory.

### 4.20.1 ASCII output

This option is used to output a line with ASCII values (a so called D-record, see chapter D-Records) on the accessory port. Name and port settings is the same as used with input ports, so, not explained here. This driver can be used to send serial data to a custom system by cable.

```
ASCII outputs settings

[0] Exit
[1] Name          >> ASCII output
[2] Port settings >> RS232 8N1; 115200 Baud
[3] Send interval >> 00:00:05
[R] Remove
>
```

#### 4.20.1.1 Send interval

The send interval determines how often data is sent to the accessory port. You can set this interval freely. At each interval, the system will output the most recent sampled values.

***Note***: If a sensor's sample interval is different from the data log interval, the sensor will be powered and sampled whenever its timing aligns with the output (send) interval.

**Note:** only parameters marked for ASCII output will be included.

### 4.20.2 Radio output

This feature is designed to transmit measurement values over a radio connection, where the radio modems used should facilitate a transparent transfer of data to their serial ports, simulating a 'virtual' serial cable.

```
Radio settings

[0] Exit
[1] Name                                >> Radio
[2] Port settings                       >> RS232 8N1; 9600 Baud
[3] Accessory power                     >> Enabled; Warm up 00:00:10
[4] Send interval                       >> 00:01:00
[5] Send delay                          >> Not used
[6] Power down delay                    >> 00:00:05
[7] Repeat previous messages (0..2)     >> 0
[8] Output format                       >> NMEA-0183
[9] Sentence identifier                 >> YDC
[A] Parameter indicators                >> Disabled
[B] Quality indicators                  >> Disabled
[R] Remove
[T] Radio test
>
```

#### 4.20.2.1 Accessory power

If the radio modem should be powered directly or switched-on by the 5V accessory power you can specify the time needed by the radio modem before its fully operational after a power-on.

#### 4.20.2.2 Send interval

The send interval determines how often the data is outputted to the accessory port. It must be equal to or greater than the data log interval. More on send interval see: Send Interval

**Note:** only parameters marked for Radio output will be included in a radio message.

#### 4.20.2.3 Send delay

When using multiple simple radio transmitters sending transmissions simultaneously, they can jam each other. To avoid this, you could specify a different send delay for each transmitter.

#### 4.20.2.4 Power down delay

Once the logger has sent its data to the radio modem, the radio modem might require some time to transmit the data over the air. This time is defined here (default 5 seconds).

#### 4.20.2.5 Repeat previous messages

When using a simple radio transmitter, its transmissions can be jammed by others. To minimize the chance of missing data, you can specify that you want to include up to two previous messages in each transmission.

#### 4.20.2.6 Output format

The data can be transmitted in YDOC format or as custom NMEA-0183 sentences. We encourage the use of the NMEA-0183 format as it is less proprietary.

#### 4.20.2.7 NMEA-0183 output

Our custom NMEA sentences have the following format:

```
$<custom sentence identifier>,<s/n>,<ddmmyy>,<hhmmss>
{[,parameter indicator],<parameter value>[,<quality indicator>]}
*<checksum><CR><LF>
```

Date/time fields are in UTC. Parameters marked for Radio output are included in a sentence in their defined parameter order. Each parameter value can optionally be preceded by a parameter indicator (the parameter code) and optionally trailed by a parameter indicator. The possible indicators are: good (G), alarm (A), timeout (T), and invalid (I).

Please find below various examples of a sentence containing temperature, humidity, and battery values.

Without parameter and quality indicators:
```
$YDC,125718987,170424,102724,21.9,41.9,4.22*7A
```

With parameter indicators:
```
$YDC,125718987,170424,102912,T,21.9,RH,41.9,BATT,4.22*10
```

With quality indicators:
```
$YDC,125718987,170424,103204,21.9,G,0.0,I,3.1,A*17
```

With parameter and quality indicators:
```
$YDC,125718987,170424,103332,T,21.9,G,RH,41.9,G,BATT,3.1,A*72
```

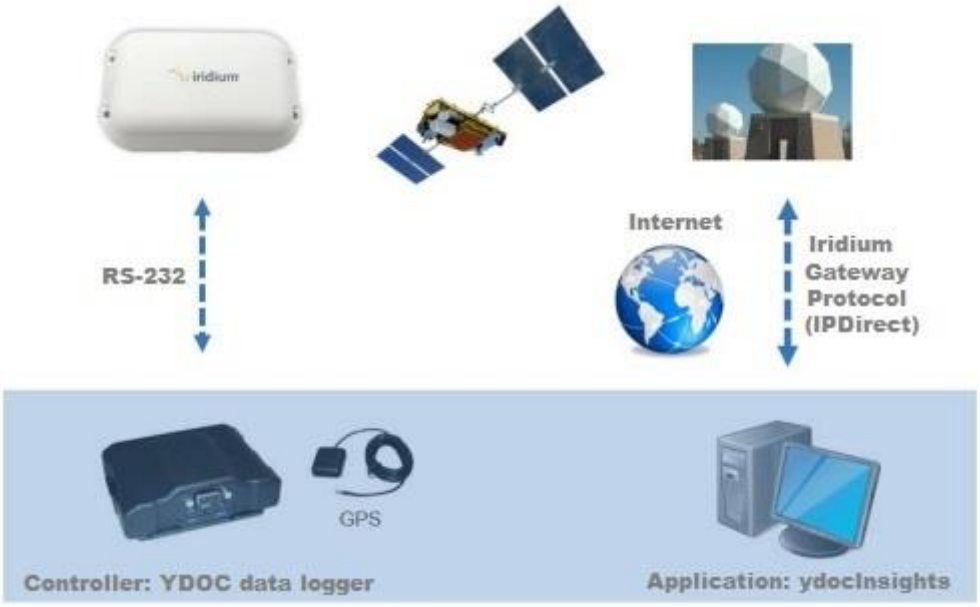
### 4.20.3 Iridium Satellite

This option is for using satellite communication for data transfer. Not all satellite modems can be used on the data logger. It is designed to work with modems based on Iridium SBD 960x transceivers. We recommend the: https://ydoc.biz/data-logger-accessoires/data-logger-satellite-transceiver/ as it supports RS-232 and has a weatherproof enclosure with integrated antenna.

The satellite transceiver can be deployed as the data logger's main communication device or as backup in case the 2G/3G/4G network is temporarily unavailable or out of reach. This backup function could be very interesting for traveling data logging applications or remote location at the edge of cellular coverage.

Satellite communication requires the transceiver to have a **clear view at the sky** e.g. it won't work well inside buildings or under the canopy of trees. By using "comport redirect" (see: Maintenance Menu - Field testing) you can check the signal strength by issuing the AT+CSQ command or even try to send a text message with the AT+SBDWT command. Please download www.ydoc.biz/download/ISU_AT_Command_Ref.pdf for a full description of these and other AT commands.

#### 4.20.3.1 Iridium SBD Service




Download : www.ydoc.biz/download/IRDM_IridiumSBDService.pdf for more info.

SBD messages are transmitted to an Iridium server (GSS: Gateway Short Burst Data Subsystem) and you can ask your Iridium provider to pass SBD messages to an e-mail address of your choosing or via IPdirect to an IP-address/domain and TCP-port of your choosing. When using your own software, you have to either pick-up and import the e-mails or implement an TCP-server to receive SBD messages.

We have implemented two flavours of SBD messages: "Text" and "Compacted".

The body of an "Text" message contains one timestamped log record at max.
The body of an "Text" message is formatted as follows:

*<logger s/n>;<yyddmmhhmmss;<par code1>;<par value1>;…;<par codex>;<par valuex>

Example: *5304783;160513140000;TEMP;23.6;LEVEL;3.32;BATT;4.2

See chapter: “Compacted Data Format” for a description of the “Compacted” SBD message format. When using ydocInsights you should ask your Iridium provider to use IPDirect and pass SBD messages to the same IP-address/domain and TCP-port as used for 2G/3G/4G TCP-output from common YDOC data loggers.

#### 4.20.3.2 Settings menu

```
Iridium Satellite settings

[0] Exit
[1] Name                >> Iridium Satellite
[2] Port settings       >> RS232 8N1; 19200 Baud
[3] Module power switch >> Enabled; Warm up 00:00:20
[4] Send interval       >> 01:00:00
[5] Send delay          >> Not used
[6] Data transpond      >> Actual values
[7] Backup mode         >> Not used
[8] Message format      >> Compacted
[R] Remove
>
```

#### 4.20.3.3 Send interval

The send interval on which the data is send by satellite communication. Beware that satellite communication is rather expensive, so please choose a moderate Send interval and consider the use of the aggregation features of the data logger (e.g. sample evert 5 minutes, but only record an hourly average). In case satellite communication is used as backup, the send interval is determined by the primary communication channel, but only when primary communication fails.

**Note:** only parameters which are configured to output data to Satellite will be included in an SBD message.

#### 4.20.3.4 Data transpond

As satellite communication is rather expensive you can choose to send only the last sampled values instead of logged values. When choosing to transpond logged values be aware to select a moderate log interval to avoid high communication costs.

#### 4.20.3.5 Backup mode

When the ‘Backup’ mode is selected satellite communication will be performed only if data transfer by TCP, FTP, e-Mail or HTTP thru the cellular network is failing. You can also specify the number of cellular transmission failures before satellite transfer takes place. When you set failures to 3 data transfer by satellite will be performed every third cellular data transmission failure.

#### 4.20.3.6 Message format

The SBD messages can be transmitted in readable ASCII or in a compacted binary format. The compacted binary format is about 20 to 80% more efficient saving a lot of communication costs, but needs specific processing to be able to interpret the message (The ydocInsights data collector is provisioned with the algorithms to interpret the message).

See chapter “Compacted Data Format” for a description of this format.

The compacted format is limited to contain up to 15 logger parameters/channels. As textual parameter codes would use to much data, the identification of the individual channels is based on a number between 1 and 15. Assigning the numbers to individual parameters is a matter of configuration by starting their code names with a capital P followed by a number between 1 and 15. The P will be stripped of, as well as any trailing characters, so you could specify meaning full codes like: P1TMP or P02HUM or P15LEV.

### 4.20.4 GPS

This is the driver for a standard NMEA-GPS. Some settings are discussed below. Although every standard NMEA-GPS can be used, it is convenient to use an YDOC–GPS, because it can be powered from the accessory port also. For info about the YDOC GPS receiver read: https://ydoc.biz/datalogger-GPS-E3329-receiver.html

***Note:*** *If GPS sentences are combined with other NMEA sentences with measurement data you want to log, its recommended to use the 'Generic NMEA' driver instead of this 'GPS' driver.*

```
GPS settings

[0] Exit
[1] Name                                >> GPS
[2] Port settings                       >> RS232 8N1; 9600 Baud
[3] Accessory power                     >> Enabled; Warm up 00:00:10
[4] Independent log interval            >> 06:00:00
[5] Minimum satellites to use           >> 3
[6] Minimum wait time to Fix            >> 00:00:00
[7] Maximum wait time to Fix            >> 00:01:00
[8] Log raw data string                 >> Off
[9] GPS time update                     >> Off
[A] Determine location on deployment    >> Off
[B] Geofence settings                   >> Disabled
[C] Satellites                          >> Satellites
[D] Latitude                            >> Latitude
[E] Longitude                           >> Longitude
[F] Drift                               >> Not used
[G] Altitude                            >> Not used
[H] Direction                           >> Not used
[I] Speed                               >> Not used
[J] GPS quality                         >> Not used
[R] Remove
[T] Test measurement
>
```

#### 4.20.4.1 Minimum satellites to use

The GPS can calculate a position based upon the data of multiple satellites, the minimum amount of satellites for this calculation is 3. But the calculation gets better, when more satellites are used. This parameter defines the minimum number of satellites that is used during the calculation. The data logger tracks the SVS (space vehicles) parameter from the GPS output string to determine this. So, the data logger will wait for the SVS in the data string to be equal or more than this figure.

#### 4.20.4.2 Time to fix

A GPS receiver has a certain time to Fix (TTF). This is the time from power-on to the calculation of a valid position. The data logger knows 2 limits to define the timeframe in which it waits for a fix. These are minimum wait time to fix and maximum wait time to fix. Please use these according the specs of your GPS. Of course, when a position is found, within the range of this timeframe, the position will be stored, and the logger will proceed to his next task. If the fix is NOT found, a sensor timeout will occur.

#### 4.20.4.3 Position drift alarming

This is a feature that simplifies the process of setting up an alarm on the position of the device. Sometimes, it is important to track the position of the system, and to couple an alarm to it. I.e. with buoys this might be the case. The buoy must measure on a specific location and may drift a few meters, but must not drift too far. To setup an alarm, with the corresponding limits is very unpractical, because the user has to work with the coordinates, coming from the GPS, and they are not comprehensive at all. Also, very often, the user has a good idea about the maximum allowable drift of the buoy, but NOT yet the starting location of it.

#### 4.20.4.4 GPS time update

If the data logger cannot use NTP to stay in sync with UTC time, you can use the timestamp transmitted by the satellites to sync with.

### *4.20.4.4.1 Calculate alarm limits on deployment*

This feature helps the user to set up a region which will be guarded by the data logger. This region can be setup at the office, far away from the installation site. The region is setup by defining the size of a rectangular, in which the buoy is supposed to operate. The buoy can drift in this box, without causing an alarm. The parameters of this box are:

### *4.20.4.4.2 Latitude/Longitude hi/lo alarm drift*

Both limits represent (geographical) seconds, and likewise, a box is created with the size of 2 times the latitude hi/lo alarm drift by 2 times longitude hi/lo alarm drift. So, this box has a user defined size, but is unrelated to the position of deployment. At deployment, these values are automatically related to the (starting) position of the system. It automatically calculates the alarm limits for the parameters latitude and longitude. Thanks to this, the user can define a “box of allowed movement” at the office, and during deployment, the absolute alarm limits for latitude and longitude are calculated.

Also, the user can define a second, bigger, box, which corresponds with the hi/hi limits of the data logger. So, a second level of alarming can be defined.

#### 4.20.4.5 Satellites

The amount of satellites, used in the calculation of the position. This parameter is retrieved from the standard NMEA string.

#### 4.20.4.6 Latitude / Longitude / Altitude

Notation of the position, in the WSG84 system,

#### 4.20.4.7 GPS Quality

Quality of the measurement. More info on this is found in the manual of your GPS.

### 4.20.5 Camera

Only YDOC cameras can be connected. It can take still pictures only, which can be stored to SD-card or transferred to a host. More info on this camera's can be found here: www.ydoc.biz/datalogger-accessories.html

```
Camera settings

[0] Exit
[1] Name                  >> Camera
[-] Port settings         >> RS232 8N1; 115200 Baud
[3] Accessory power       >> Enabled; Warm up 00:00:05
[4] Picture taking        >> Log interval; Any alarm; Digital trigger
[5] Independent data log  >> On
[6] Sample log interval   >> 01:00:00
[7] Interval shift        >> Not used
[8] Daily operating time  >> 09:00:00 - 15:00:00
[9] Picture size          >> Normal
[A] Picture sending       >> FTP
[B] Picture storage       >> Delete after send
[C] Picture number        >> Picture
[R] Remove
[T] Camera test           >> Passed
>
```

Select a picture size matching your needs. Please consider that a high-resolution picture results in more data payload, more transfer time and power consumption. So don't exaggerate with the picture-size.

Pictures can be transferred via HTTP, E-mail, FTP, TCP, or MQTT. To do this, you need to enable or configure the corresponding output driver for the specific protocol. If you don't want to transmit log data as well, it's recommended to configure the driver to send just a "Heartbeat" once a day to avoid unnecessary payload and reduce power consumption.

For an estimation of power consumption and payload, please visit: www.ydoc.biz/datalogger-power-consumption.html

#### 4.20.5.1 Picture taking

Pictures can be taken independent from the normal data logging interval, e.g. just once a day. Picture are also taken on any alarm, unless you switch it off. You can also use a digital input to take an instantaneous picture (See chapter: "Digital inputs – Action trigger").

#### 4.20.5.2 Interval shift

Intervals are standard aligned with midnight, so an interval of 12h occurs at 00:00 and 12:00. You can apply an interval shift to deviate from the standard alignment, e.g. when applying an 8h shift the above mentioned intervals will occur at 08:00 and 16:00.

#### 4.20.5.3 Daily operating time slot

If you take several pictures per day, but don't want to take pictures at night or in opposite during the day, then you can specify a daily operating time slot in which pictures will be taken (e.g. between 06:00 and 18:00 or between 18:00 and 06:00.

### 4.20.6 Display

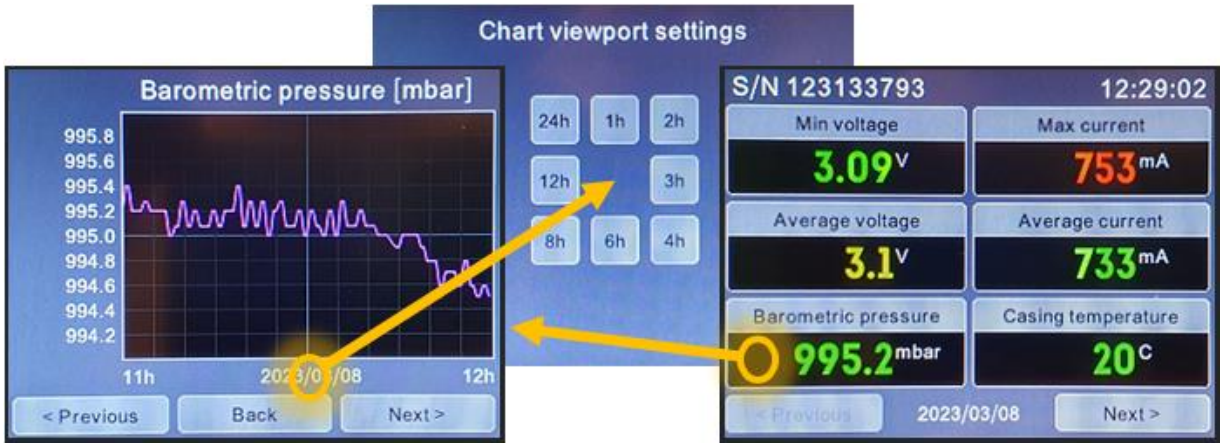
This driver is intended for the operation of an YDOC 3.5" resistive touch colour TFT display (320x240 pixels). The display is used for displaying actual values and historical data as graphs, it's not meant for configuration purposes.

To preserve power the display can be switched on/off with a mechanical push button. After showing the start-up picture the display will show the last known values of up to 6 parameters simultaneously in a matrix of 3 rows and 2 columns. The values of additional parameters can be shown by stepping through available columns. The columns are ordered from left to right and their row items from top to bottom.

See chapter "Parameter overview" to alter the order of parameters.

Values will be displayed in various colours, where: green is okay, yellow is low/high limit exceeded, red is parameter invalid or low-low/high-high exceeded and grey means value not recently acquired.

By touching a parameter area a graph with historical data over a viewport of the last 24h will be shown. You can step through historic data per viewport page. The viewport cab be changed to 24h, 12h, 8h, 6h, 4h, 3h, 2h or 1h by touching the date in the y-axis of the graph.




```
Display settings

[0] Exit
[1] Name              >> Display
[-] Port settings     >> RS232 8N1; 115200 Baud
[3] Power down delay >> 00:01:00
[R] Remove
>
```

#### 4.20.6.1 Power down delay

Auto power down delay when display is not touched for a while (default: 60s).

### 4.20.1 Accessory port tunnel

If you have connected a serial device, but it's not used during normal operation, then you can access this device thru the accessory port tunnel (e.g. thru the COM-port terminal in the Maintenance-menu). You can configure the parameter of the serial device thru the screens below:

```
COM port tunnel settings

[0] Exit
[1] Name             >> Accessory port tunnel
[2] Port settings    >> RS232 8N1; 9600 Baud
[3] Accessory power  >> Enabled; Warm up 00:00:10
[R] Remove
>
```

#### 4.20.1.1 Port settings

Specify the serial communication characteristics (baud rate & parity) of the serial device.

```
Port settings

[0] Exit
[1] Mode      >> RS232 8N1
[2] Baud rate >> 9600
[-] Address   >> Not used
>
```

#### 4.20.1.2 Sensor power

When the device should be powered from the 5V accessory power switch, please specify how long it takes before the device is capable to communicate meaningful.

## 4.21 Modem output - Email

This output driver is used to send the collected data from the SD Card to a mailbox of the user. The data is included in an attachment of the email. Below the settings are shown: Some of its settings are commented below, other settings are generic and covered before in this manual.

**Note:** The E-mail output is mutual exclusive with the MQTT output.

```
Email settings

[0] Exit
[1] Name                >> Email
[2] Send interval       >> 01:00:00
[3] Send delay          >> Not used
[4] SMTP server         >>
[5] SMTP port           >> 465
[6] Security            >> TLS(implicit)
[7] Username            >>
[8] Password            >>
[9] Originator address  >>
[A] Destination address >>
[B] Subject             >>
[C] Output type          >> Log data
[D] Data container      >> File attachment
[E] File name           >> YDOC_STATION01_140006330_241024_133400.txt
[F] Data format         >> YDOC
[G] Max payload         >> 1000 kB
[H] Data filter         >> Data & Diagnostics
[R] Remove
[T] Email test          >> Not done
>
```

### 4.21.1 Server

This is the IP-address or domain name of the SMTP-server you want to use for e-mailing. You can use you ISP's server, or any other server you like.

### 4.21.2 SMTP port

The port on which the SMTP server is listening. Port 25 or 587 is typically used for plain SMTP. Port 465 is used for secure SMTP with implicit TLS. Explicit TLS (STARTTLS) over port 587 is not supported by the data logger.

### 4.21.3 Security

The data logger supports basic SMTP (credentials and data unencrypted) or encrypted SMTP over TLS (implicit).

### 4.21.4 Originator address

This is the "From"-address, the email address of the used e-mail account.

### 4.21.5 Destination address

This is the "To"-address, the email address(es) to where the data should be send to. You can enter multiple addresses by using a comma as separator between them. To minimize entry space, you can omit the @domain part of addresses in which case the domain of the previous address is used or that of the originator address.

#### 4.21.1 Output type

This defines what kind of data you want to send. This can be “Log data”, “Actual values” or a “Heartbeat” (a minimal status message).

#### 4.21.1 Data container

This defines if you want the data to be included in the e-mail body or as attached file.

#### 4.21.1 File name

The default file name is formatted as “YDOC_<sysname>_<sn>_<yy><mm><dd>_<hh><nn><ss>” and appended with the file extension (.txt. .csv. or .json).

You can customize the file name to your liking and use substitutions like <sysname> for your “system name”, <sn> for the serial number, <name> for the FTP “driver name” as well as various timestamp tokens like <hh> for a 2-digit hour of the day or <h> for hour of the day without leading 0’s. For a 4 digit year presentation use <yyyy>. **Note that** <m> or <mm> is used for months and <n> or <nn> for minutes.

When all possible substitution tokens (9) are used you will have just 7 characters left to customize the file name, but you could manipulate the file by specifying the “system/driver” names wisely. i.e. use the “system name” for unique system identification and the “driver name” as a common file name part.

#### 4.21.1 Data format

This defines the data format of the file to post. This can be our native TXT format, CSV (comma separated value) format or our JSON format

#### 4.21.2 Remote configuration

Not supported, if you want remote configuration possibilities use MQTT, FTP, HTTP or TCP output instead.

## 4.22 Modem Output - FTP

This is for sending data to an FTP- or FTPS-server, see details below

```
FTP settings

[0] Exit
[1] Name           >> FTP
[2] Send interval  >> 01:00:00
[3] Send delay     >> Not used
[4] Server         >> collector.ydoc.biz
[5] Port           >> 21
[6] FTP mode       >> Active
[7] Security       >> TLS(explicit)
[8] Username       >> YDOC
[9] Password       >> ******
[A] Directory      >> logfiles
[B] Output type    >> Log data
[C] File name      >> YDOC_STATION01_123128883_220503_103700.json
[D] Verification   >> Yes
[E] Data format    >> JSON
[F] Max payload    >> 1000 kB
[G] Data filter    >> Data & Diagnostics
[P] Input parameters >> 1
[R] Remove
[T] FTP test       >> Done
>
```

### 4.22.1 Server

This is the IP-address or domain name of the FTP-server you want to transfer files to.

### 4.22.2 Port

Port on which the FTP server is listening, which is 21 by default.

### 4.22.3 FTP mode

This is to choose between Active or passive FTP. More on this topic see:
https://en.wikipedia.org/wiki/File_Transfer_Protocol

### 4.22.4 Security

The data logger supports basic FTP (credentials and data unencrypted) or secure FTPS (using explicit TLS to encrypt credentials and data).

### 4.22.5 Directory

The relative path to the destination on your server.

### 4.22.1 Output type

This defines what kind of data you want to send. This can be “Log data”, “Actual values” or a “Heartbeat” (a minimal status message).

### 4.22.2 File name

The default file name is formatted as “YDOC_<sysname>_<sn>_<yy><mm><dd>_<hh><nn><ss>” and appended with the file extension (.txt. .csv. or .json).

You can customize the file name to your liking and use substitutions like <sysname> for your “system name”, <sn> for the serial number, <name> for the FTP “driver name” as well as various timestamp

tokens like <hh> for a 2-digit hour of the day or <h> for hour of the day without leading 0's. For a 4 digit year presentation use <yyyy>. **Note that** <m> or <mm> is used for months and <n> or <nn> for minutes.

When all possible substitution tokens (9) are used you will have just 7 characters left to customize the file name, but you could manipulate the file by specifying the "system/driver" names wisely. i.e. use the "system name" for unique system identification and the "driver name" as a common file name part.

### 4.22.3 Verification

To check if the file is fully transferred the data logger will verify if the transferred file has the expected file size. If the transferred file is removed to quickly by some FTP-client or if the FTP-server does not allow the file size command the verification will fail and you have to switch this option to "Off". When switching off verification there is a limited chance on data loss, it's at your own risk (Note: All log data is still available on the SD-card).

### 4.22.4 Input parameters

An FTP-client is able to transmit up to 8 different parameters to the data logger. Such a parameter can, like any other sensor parameter, be logged, alarmed up on or used in a derived channel (e.g. a calculated channel that evaluates to an alarm condition, in example when a level exceeds x and flow exceeds y).

The FTP-client should place the file with input parameters on the FTP-server in the same directory where data log files are transferred to.

The name of the file should be formatted as **YDOC_<SN>.dat** and contain one or multiple lines separated from each other with a carriage return or carriage return line feed, with one **<name>=<value>** pair per line. The name should match with a parameter code of one of the defined input parameters.

e.g.

**MAXLEV=3.2**
**MAXFLOW=4.5**

The decimals should be separated with a dot.
The data logger will remove the file from the server once processed
.

### 4.22.5 Remote configuration and firmware upgrade

As of FW V4.5B3 its possible to update the data logger configuration or upgrade the FW by placing a file on the FTP-server in the same directory where data log files are transferred to.

The names of the files should be formatted as follows:

- **YDOC_<SN>.bin** for a FW file.
- **YDOC_<SN>.cfg** for a config file, this can be a complete binary configuration file or a file containing just some changes by means of tiny configuration snippets (See chapter configuration snippets).

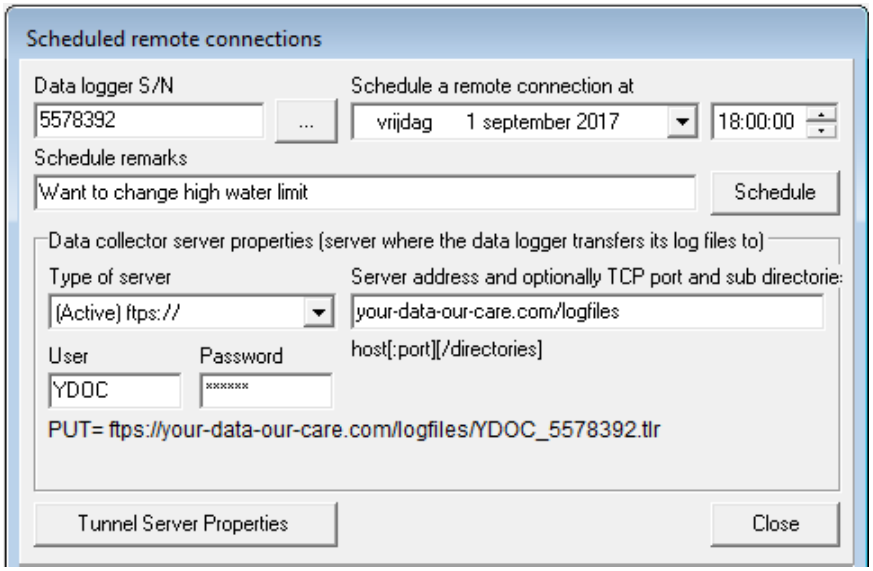
The data logger will remove the file from the server once processed.

### 4.22.6 Remote access

If the logger has FW V3.0B1 or newer, remote access to the data logger can be scheduled by ydocTerminal V3.0. using our secure tunnel server (tunnel.ydoc.biz).

***Note:*** The data logger should be configured to use NTP time synchronization.

Within ydocTerminal click: File->New->Schedule a remote connection.




Specify the S/N of the data logger you want to access and when you think it's a convenient moment to access the logger.

**Obviously**: A tunnel cannot be established before the next scheduled data transfer interval of the data logger.

A tunnel can only be established if ydocTerminal is running at the scheduled moment. When running, ydocTerminal will pop-up a terminal window a few minutes before the scheduled moment.

**Type of Server**

ydocTerminal supports 4 different types of FTP-server (Passive FTP, Active FTP, Passive FTP with Explicit TLS or Active FTP with Explicit TLS.

**Server address**: Specify the same server and directory as used by the data logger (e.g. ydoc.biz/logfiles)

**User/Password:** Specify the exact same credentials as used by the data logger as they are a/o used in the encryption keys. The user should have read, write and delete rights in the specified directory.

**Tunnel Server Properties**

This is the Server servicing the secure and encrypted channel between data logger and ydocTerminal. Standard our server is used (tunnel.ydoc.biz), don't worry we can't eavesdrop your data.

## 4.23 Modem Output - HTTP

This is for posting data to an HTTP- or HTTPS-server, see details below. The HTTP output is of interest when you are hosting your own monitoring web or cloud server. When using ydocInsights we recommend using TCP output, as TCP-output has far less overhead with less data payload (costs) and less transfer time (power consumption).

**Note:** The HTTP output is mutual exclusive with the TCP output.

```
HTTP settings

[0] Exit
[1] Name             >> HTTP
[2] Send interval    >> 01:00:00
[3] Send delay       >> Not used
[4] Server           >> your-data-our-server.com/datacollector/
[5] Extended path    >> ?sn=12345&name=station1
[6] Port             >> 80
[7] Security         >> Basic
[8] Username         >> YD0C
[9] Password         >> ******
[A] Output type      >> Log data
[B] Data format      >> JSON
[C] Max payload      >> 1000 kB
[D] Data filter      >> Data & Diagnostics
[R] Remove
[T] HTTP test        >> Passed
>
```

### 4.23.1 Server

This is the URI of a location on an HTTP-server where log file and camera pictures should be posted to. In case of a long URI you can type the remainder, not fitting in the 'Server'-part, as 'Extended path'.

When at server-side multiple loggers should share the same URI, the loggers can be distinguished from each other by examining the custom HTTP-header 'X-DeviceSN' which will contain a logger's unique serial number.

### 4.23.2 Extended path

The extended path can be used of the URL is to long to be specified under server or if you need to specify server application required parameters.

You can also use substitutions like <sysname> for your "system name", <sn> for the serial number, <name> for the HTTP "driver name" as well as various timestamp tokens like <hh> for a 2-digit hour of the day or <h> for hour of the day without leading 0's. For a 4 digit year presentation use <yyyy>. **Note that** <m> or <mm> is used for months and <n> or <nn> for minutes.

Assume your server application requires you to pass the name and serial number of the device as URL parameters sn & name, then you could in example enter: `?sn=<sn>&name=<sysname>` as extended path.

### 4.23.3 TCP port

Port on which the HTTP-server is listening, which is 80 by default for HTTP or 443 for HTTPS.

### 4.23.4 Security

The data logger supports basic HTTP (credentials and data unencrypted) or HTTPS (using SSL/TLS to encrypt credentials and data).

The following security options are available:
0 = Basic, 1 = HTTPS, 2 = Azure-IoT SAS, 3 = Token, 4 = Header.

#### 4.23.4.1 Basic

Plain unencrypted HTTP using basic authorization (Authorization: basic base64(username:Password)

#### 4.23.4.2 HTTPS

TLS encrypted HTTPS using basic authorization (Authorization: basic base64(username:Password)

#### 4.23.4.3 Azure-IoT SAS

TLS encrypted HTTPS using Azure Shared Access Token authorization (see paragraph 4.24 for details).

#### 4.23.4.4 Token

TLS encrypted HTTPS using shared token authorization (Authorization: <token name> <token value>)

#### 4.23.4.5 Header

TLS encrypted HTTPS using custom HTTP-header authorization (<header name>: <header value>, e.g. X-ApiKey: 715b6935-e3d8-48b1-86a3-a6e2a268e1f)

### 4.23.5 Data format

This defines the data format of the file to post. This can be our native TXT format, CSV (comma separated value) format or our JSON format which can easily be picked up by server-side scripting of an HTTP-server.

Depending on the data format the following 'Content-Type' HTTP header will be added to the post.

| | | | |
|---|---|---|---|
| **text/plain** | Our native TXT format | **application/json** | Our JSON format |
| **text/csv** | CSV format | **image/jpeg** | A camera picture |

If used, camera pictures are posted to the same URI as the log file so you have to examine the 'Content-Type' header to distinguish a picture post from a log file post.

### 4.23.6 Remote access

If the logger has FW V3.0B1 or newer, remote access to the data logger can be scheduled by ydocTerminal V3.0. using our secure tunnel server (tunnel.ydoc.biz).

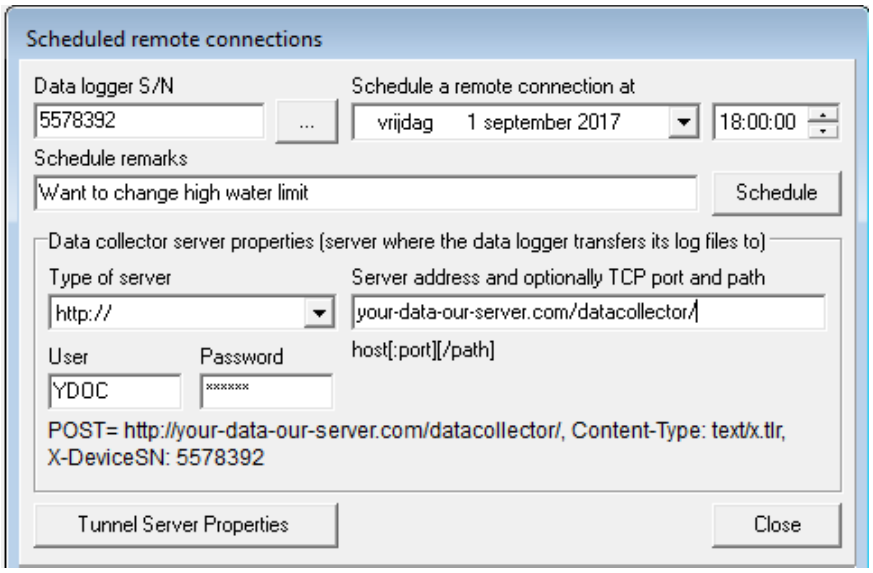
***Note:*** The data logger should be configured to use NTP time synchronization.

Within ydocTerminal click: File->New->Schedule a remote connection.

Specify the S/N of the data logger you want to access and when you think it's a convenient moment to access the logger.

**Obviously**: A tunnel cannot be established before the next scheduled data transfer interval of the data logger.

A tunnel can only be established if ydocTerminal is running at the scheduled moment. When running, ydocTerminal will pop-up a terminal window a few minutes before the scheduled moment.




**Type of Server:** HTTP or HTTPS with basic authorization.

**Server address**: The HTTP-server must honor POST requests from ydocTerminal at the specified URL.

ydocTerminal will POST the request with 'content-type: **text/x.tlr'**. The S/N of the concerned data logger is given in the custom HTTP-header 'X-DeviceSN' The tunnel link request string itself is contained in the body of the POST. This request string should be stored until the concerned data logger performs a POST itself (its log data). The HTTP-server should copy the request string as the value of the custom HTTP-header 'X-Command' in the response to the data logger.

**User/Password:** Specify the exact same credentials as used by the data logger as they are a/o used in the encryption keys.

**Tunnel Server Properties**

This is the Server servicing the secure and encrypted channel between data logger and ydocTerminal. Standard our server is used (tunnel.ydoc.biz), don't worry we can't eavesdrop your data as the communication is 'Peer-to-Peer' encrypted by security keys not know to the tunnel server. If having doubts, please feel free to run your own tunnel server, please ask our staff how to do it.

## 4.24 Modem Output – HTTP Azure-IoT

This is the same driver as the HTTP(S)-driver as described above, but used in a special Azure-IoT security mode generating the “Shared Access Token”. To enable this mode, please select security mode “Azure-IoT SAS”.

```
HTTP settings

[0] Exit
[1] Name          >> HTTP
[2] Send interval >> 06:00:00
[3] Send delay    >> Not used
[4] Server        >> your-IoT-hub-name.azure-devices.net
[-] Extended path >>
[6] Port          >> 443
[7] Security      >> Azure-IoT SAS
[8] Device ID     >> your-device-ID
[9] Device key    >> your-base64-device-key
[A] Output type   >> Log data
[B] Data format   >> JSON
[C] Max payload   >> 1000 kB
[D] Data filter   >> Data & Diagnostics
[R] Remove
[T] HTTP test     >> Not done
>Security (0=Basic, 1=HTTPS, 2=Azure-IoT SAS, 3=Token, 4=Header):
```

### 4.24.1 Server

This is your IoT-Hub name concatenated with a dot and with the domain name of the Azure-IoT hub server (azure-device.net)

### 4.24.2 Device ID

This is the device ID given by you in your Azure-portal

### 4.24.3 Device Key

This is the primary or secondary key in base64 as generated by you in your Azure-portal. If the generated key is to long to fit, please use a manual chosen base64 key in your Azure-portal.

## 4.25 Data Output - MQTT

This is for posting data to an MQTT-broker, see details below. The MQTT output is of interest when you want to use a third party IoT monitoring system or smart phone apps supporting generic MQTT. When using ydoc-Insights we recommend using TCP output, as TCP-output has far less overhead with less data payload (costs) and less transfer time (power consumption).

This MQTT driver can be used for sending alarm messages, remote configuration update and firmware upgrade as well, but if you don't want to use the driver to transmit the "Log data", its recommended to schedule the driver to transmit just the "Actual Values" or just a "Heartbeat" once a day to avoid unnecessary communication payload and according power consumption.

**Note:** The MQTT output is mutual exclusive with the E-mail output.

```
MQTT settings

[0] Exit
[1] Name              >> MQTT
[2] Send interval     >> 01:00:00
[3] Send delay        >> Not used
[4] Server            >> m21.cloudmqtt.com
[5] Port              >> 12656
[6] Security          >> Basic
[7] Username          >> YD0C
[8] Password          >> ******
[9] Root topic        >> YD0C/5025064
[A] Client ID         >> 359180082361087
[B] Clean session     >> No
[C] Output type       >> Log data & Actual values
[D] Data format       >> JSON
[E] Max payload       >> 1000 kB
[F] Data filter       >> Data & Diagnostics
[P] Input parameters >> 0
[R] Remove
[T] MQTT test         >> Passed
>
```

### 4.25.1 Server

This is the IP-address or domain name of the broker to publish to. Most brokers offer free accounts with a connection or payload limit. For mass transfer you might need to use a paid account or run your own MQTT-broker.

### 4.25.2 Port

The TCP port on which the MQTT-broker is listening, which is 1883 by default or 8883 for MQTT over TLS.

### 4.25.3 Security

The data logger supports basic MQTT (credentials and data unencrypted) or MQTT over TLS.

### 4.25.4 Root topic

The MQTT protocol works with so called "Topics" and MQTT clients can subscribe to the "Topics" of their interest (of course in the domain they have access to, you don't want unauthorized entities to subscribe to publications from your equipment).

Within the data logger you can define a “Root topic”, which is used as prefix for all "Sub topics". The default “Root topic” is "YDOC/<device serial number>", but it can be changed according to your wishes. E.g. you could define a "Root topic" starting with your own company identification code, followed by some device classification code and your own chosen station identification code.

In example: (ACME/METEOSTATIONS/STATION_001). Your central monitoring system could subscribe to all topics starting with ACME and when a logger publishes for the first time, then your system could create a profile for "STATION_001" based on the specified "METEOSTATIONS" classification.

Clearing the ‘Root topic’ will set it back to the default YDOC/<SN>

The logger uses the following publishing topics:

- **<root>/sensors/<parameter code>** to publish an "Actual value" of a specific parameter.
- **<root>/data/jsn** to publish the "Log data" in JSON format.
- **<root>/data/csv** to publish the "Log data" in CSV file format.
- **<root>/data/txt** to publish the "Log data" in YDOC native text format.
- **<root>/alarm/sys** to publish "System alarms" in plain text, e.g. about a failing sensor.
- **<root>/alarm/data** to publish "Data alarms" in plain text, e.g. about reaching a low tank level.
- **<root>/jpg/<yymmdd_hhmmss>** to publish "Camera pictures with timestamp" in JPEG format.
- **<root>/status/time** the time of the logger at start of the MQTT session (yyyy/mm/dd hh:mm:ss).

### 4.25.5 Client ID

Every MQTT client should connect with a unique client identification for which we use the unique IMEI-number of the built in modem. If necessary you can specify your own client ID.

### 4.25.6 Clean Session

**Only check this option** if your MQTT-broker requires it, else you won’t be able to do remote configuration or “over the air” configuration updates or firmware upgrades anymore.

### 4.25.7 Data format

The MQTT-driver can be configured to transfer its historical data in various payload formats (csv, txt and json) and is provided with a Sparkplug B option as well. Sparkplug-B is becoming a defacto IIoT standard to easily connect to industrial systems like SCADA systems. See Chapter ‘Sparkplug-B Data Format’ for more info and deployment options.

### 4.25.8 Max Payload

In circumstances like a network outage a data log file can grow bigger than the maximum topic payload allowed by your MQTT-broker. Please specify the limit if any and the logger will chop the payload in parts.

### 4.25.9 Input Parameters

Another MQTT-client is able to transmit up to 8 different parameters to the data logger. Such a parameter can, like any other sensor parameter, be logged, alarmed up on or used in a derived channel (e.g. a calculated channel that evaluates to an alarm condition, in example when a level exceeds x and flow exceeds y).

An MQTT-client can publish a numeric value to one of the defined input parameters with the following topic format: “YDOC/<device serial number>/inputs/<parameter code>” (e.g. **YDOC/1234567/inputs/MAXLEV**). The payload should contain the numeric value in plain text presentation, using dot as decimal separator.

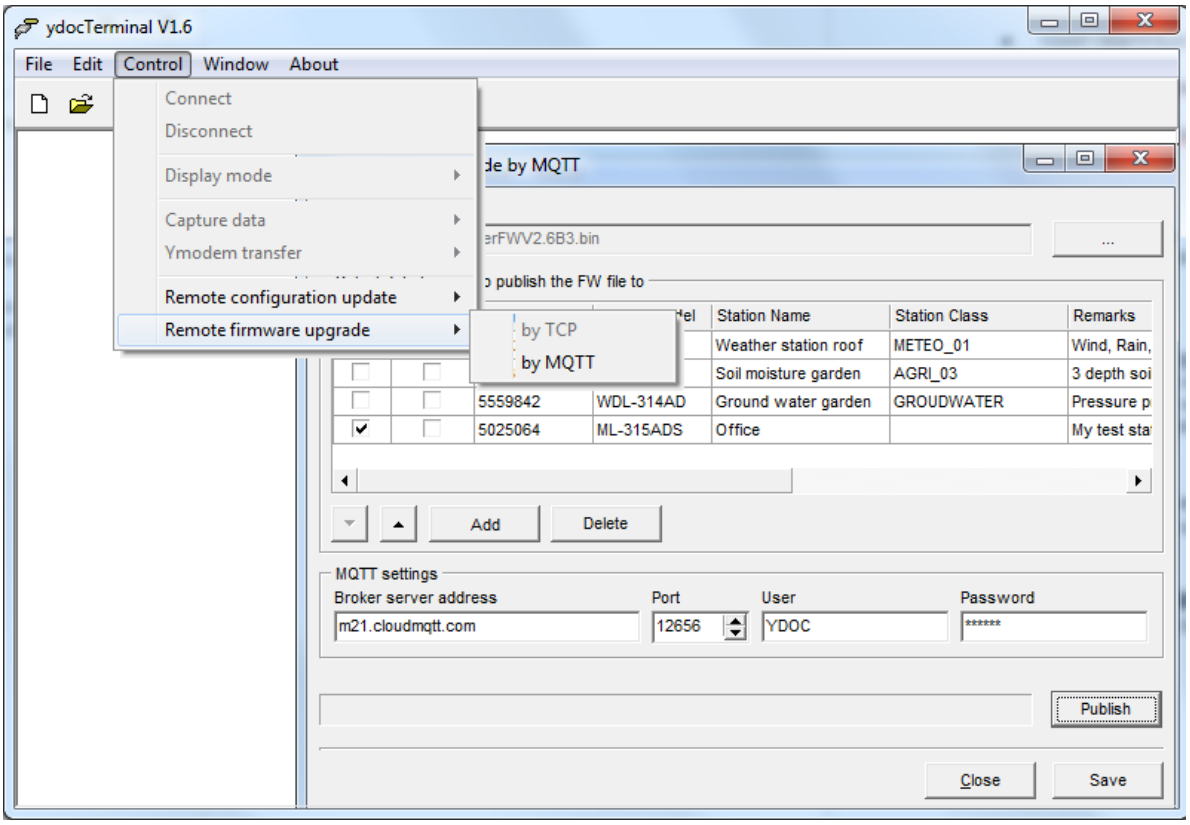
***Note***: The topic must be published with QOS=1, else the broker will not retain it till the next logger connection.

### 4.25.10 Remote configuration and firmware upgrade

It's possible to update the data logger configuration or upgrade the FW by publishing to the following topics:

- **YDOC/<SN>/fw/set** where the payload should contain the contents of a FW file.
- **YDOC/<SN>/cfg/set** where the payload should contain the contents of a complete binary config file or just some changes by means of tiny configuration snippets (See chapter configuration snippets).

You can also use ydocTerminal to publish config or FW files to a data logger. If you want to publish a configuration snippets by ydocTerminal, please save the snippets to a file first.

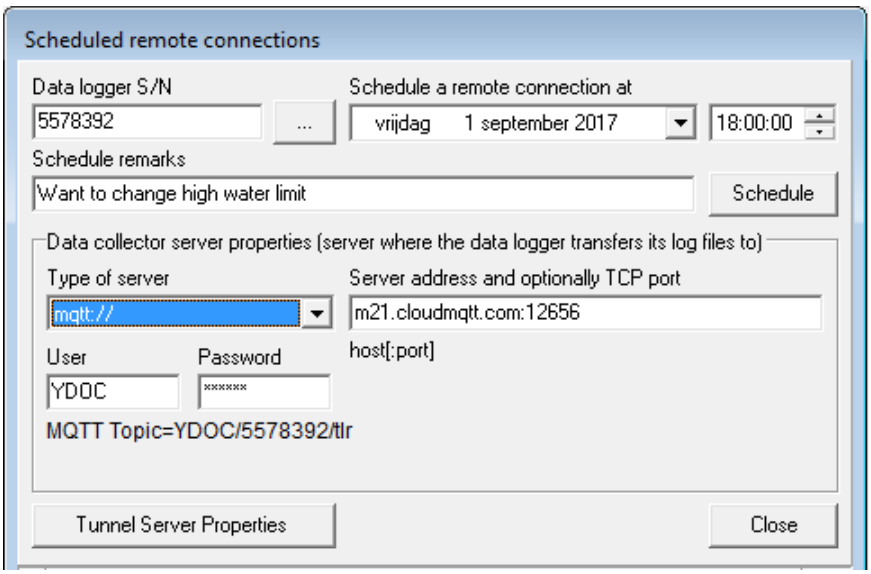


### 4.25.11 Remote access

If the logger has FW V3.0B1 or newer, remote access to the data logger can be scheduled by ydocTerminal V3.0. using our secure tunnel server (tunnel.ydoc.biz).

***Note:*** The data logger should be configured to use NTP time synchronization.

Within ydocTerminal click: File->New->Schedule a remote connection.




Specify the S/N of the data logger you want to access and when you think it's a convenient moment to acces the logger.

**Obviously**: A tunnel cannot be established before the next scheduled data transfer interval of the data logger.

A tunnel can only be established if ydocTerminal is running at the scheduled moment. When running, ydocTerminal will pop-up a terminal window a few minutes before the scheduled moment.

**Type of Server:** MQTT

**Server address**: Specify the same server and port as used by the data logger (e.g. m21.cloudmqtt.com:12656)

**User/Password:** Specify the exact same credentials as used by the data logger as they are a/o used in the encryption keys.

**Tunnel Server Properties**

This is the Server servicing the secure and encrypted channel between data logger and ydocTerminal. Standard our server is used (tunnel.ydoc.biz), don't worry we can't eavesdrop your data as the communication is 'Peer-to-Peer' encrypted by security keys not know to the tunnel server. If having doubts, please feel free to run your own tunnel server, please ask our staff how to do it.

### 4.25.12 COM-Tunnel

With MQTT its possible to send/receive data to/from serial devices connected to the data logger. In example you could send commands to change some device specific settings. Commands can be send to the serial sensor or accessory port on the main PCB as well as on option boards. The (binary) data contained in the MQTT payload will be passed transparently 1 to 1 to the concerned serial device and the response will be replied as (binary) payload 1 to 1 as well, it's up to the MQTT-client to format and interpret the data to/from the serial device (which could be MODBUS/RTU, SDI-12, NMEA-0183 or any other protocol).

When a COM-tunnel message is received for a device connected at a serial sensor port and the device requires power from the 12V sensor power switch, the data logger will switch on the 12V power and wait a certain warm-up time before sending the command to the device (See chapter: Serial port).The warm-up time will not be applied in case of sequential commands in the same MQTT-session.

When a COM-tunnel message is received for a device connected at an accessory port and the device requires power from the 5V accessory power switch, the data logger will switch on the 5V power and wait a certain warm-up time before sending the command to the device (See chapter: Accessory port).The warm-up time will not be applied in case of sequential commands in the same MQTT-session.

After sending data to the serial device, the data logger will wait max 3 seconds for reply from the connected device. In case of no response, the data logger will publish an MQTT message with empty payload. If the serial device replies, the reply is considered complete after a character silence time of 0.25s. The data logger will publish an MQTT message with the full 1 to 1 (binary) reply as the payload.

#### 4.25.12.1 Topics

An MQTT-client should format a command publishing topic as follows:

"**YDOC**/<device serial number>/**comtx**/<port>[/<client defined sup-topics]" (QoS=1)

Where <port> can be (if installed and configured in the data logger):
- "sp1", the serial sensor port on the main PCB
- "ap1", the accessory port on the main PCB
- "sp2", the serial sensor port on an option board
- "ap2", the accessory port on an option board.

The MQTT-client can specify optional sub-topics for its own internal management, the used sub-topics are also included in the MQTT reply messages. Such sub-topics could be used for grouping or matching commands with replies (e.g. by using some sequence# as sub-topic).

Example: "YDOC/99091660/**comtx**/sp1" or "YDOC/99091660/**comtx**/sp1/S123/METEO/"

The data logger will publish the reply with the same topic as the command, but will replace the text "com**t**x" with "com**r**x" (Example: "YDOC/99091660/**comrx**/sp1/S123/METEO/")

An MQTT-client can use wildcards to subscribe to any COM-tunnel message from any data logger, by subscribing to: "**YDOC**/+/**comrx**/#" (QoS=1)

#### 4.25.12.2 Limits

Topic size: 128 characters.
Payload size: 256 bytes for commands, 512 bytes for replies.
Device reply timeout: 3s (The device should answer within 3 sec).
Silence time: 0.25s (A reply is considered complete after a communication silence time of 0.25s).

## 4.26 Data Output - TCP

This is for sending data to an TCP server, see details below

**Note:** The TCP output is mutual exclusive with the HTTP output.

```
TCP settings

[0] Exit
[1] Name            >> TCP
[2] Send interval   >> 01:00:00
[3] Send delay      >> Not used
[4] Server          >>
[5] Port            >> 37
[6] Security        >> Basic
[7] Username        >>
[8] Password        >>
[9] Output type     >> Log data
[A] Data format     >> Native(txt)
[B] Max payload     >> 1000 kB [R] Remove
[C] Data filter     >> Data & Diagnostics
[T] TCP test        >> Not done
>
```

### 4.26.1 Server

This is the IP-address or domain name of the TCP-server you want to send data to.

### 4.26.2 Port

Port on which the TCP-server is listening, which is 37 by default. When running your own TCP-server for data collection (e.g. the one that comes with ydoc-Insights) make sure that the chosen port is forwarded by your internet router to the local IP-address of the system/computer running your TCP-server.

See the tutorials at: www.ydoc.biz/datalogger-manuals.html

### 4.26.3 Security

The method used to logon to the TCP-server, this can be ‘Basic’, ‘Secure Authentication’ or ‘AES-128 encrypted’. When using ‘Basic’ the credentials are transferred by TCP in plain text. This is quick and therefore consuming less power. When using ‘Secure Authentication’ the password is uni-directional encrypted with a unique challenge token received from the server. ‘Secure Authentication’ is more secure but will consume about 1 second more precious power time per transfer. ‘AES-128 encryption’ includes ‘Secure Authentication’ and encrypts all communication, this is the most secure and recommend method.

## 4.27 SMS output

This output driver is used to send the collected data from the SD Card to a cell phone number by SMS. As an SMS is limited in size transferring data by SMS has some limitations too. We recommend to use this driver only if other alternatives are not feasible. Below the settings are shown: Some of its settings are commented below, other settings are generic and covered before in this manual.

```
SMS data settings

[0] Exit
[1] Name          >> SMS
[2] Send interval >> 00:01:00
[3] Send delay    >> Not used
[4] Phone number  >>
[5] Output type   >> Log data
[R] Remove
[T] SMS data test >> Not done
>
```

### 4.27.1 Phone number

The cell phone number you want to send data messages to.

### 4.27.2 Data send

An SMS message is limited to 160 characters, so it's not feasible (like with FTP for example) to send a complete log file in one SMS message. You can choose to configure this driver to only send the last logged record (Actual Values) or all logged records since the last successful data transmission (Log data). Each log record will be sent as an individual SMS message**. Note**: Independ data logs (not part of the regular data log intervals) are not include in SMS data output.

### 4.27.3 Data format

Each data SMS start with an asterisk, followed by a logger serial number, timestamp and followed by one or more parameter code/value pairs in the same format as with D-record (see: Chapter D-records).

*<logger s/n>;<yyddmmhhmmss;<par code1>;<par value1>;…;<par codex>;<par valuex>

Example: *5304783;160513140000;TEMP;23.6;LEVEL;3.32*A;BATT;4.2

### 4.27.4 Remote configuration

Not supported, if you want remote configuration possibilities use MQTT, FTP, HTTP or TCP output instead.

## 4.28 Data Output - Satellite

A satellite transceiver can be deployed as the data logger's main communication device or as backup in case the cellular network is temporarily unavailable or out of reach. Please read more at chapter "Accessory port – Iridium Satellite".

## 4.29 Parameter overview

The parameter overview is used to generate an overview of the configuration. It enables the user to evaluate the settings of the logger in an easy way as well as order the order of parameters, it consists of multiple lists:

```
Parameter overview

[0] Exit
[1] Parameter list
[2] Parameter order
[3] Measurement list
[4] Output list
[5] Alarm list
[6] Alarm limits
>
```

### 4.29.1 Parameter list

This gives an overview of the configured parameters and their units in a comprehensive ordered list.

```
Parameter list

Parameter       Name                    Unit            Min / Max

[1] AVGVi       Average voltage         V
[2] AVGCi       Average current         mA
[3] OCi         Operating cycle         sec

[0] Exit
>
```

By clicking the character between brackets in front of the concerned parameter brings you directly to the parameter configuration screen (e.g. to change its name or unit).

### 4.29.2 Parameter order

This gives an overview of the configured parameters and their units in an ordered list.

```
Parameter order

Order   Code    Name

[1] 1   AVGVi   Average voltage
[2] 2   AVGCi   Average current
[3] 3   OCi     Operating cycle

[0] Exit
```

You can change the order of a parameter in the list by specifying a new position. To change the position, please click the character between brackets in front of the concerned parameter, this will pop-up a prompt to alter its current position.

### 4.29.3 Measurement list

This gives an overview of used measurements and its sample rates etc.

```
Measurement list

Parameter       Sample interval Alarm interval  Factor  Offset
Decimals

[1] AVGVi       00:00:05        Not used        1       0       2
[2] AVGCi       00:00:05        Not used        1       0       0
[3] 0Ci         00:00:05        Not used        1       0       2

[0] Exit
>
```

By clicking the character between brackets in front of the concerned parameter brings you directly to the parameter configuration screen (e.g. to change a parameters sample interval).

### 4.29.4 Output list

This gives an overview of the configured parameters and their outputs in an ordered list.

```
Output list

Parameter       Log interval    Alarm interval  Log     Output

[1] AVGVi       00:00:05        Not used        On      Modem; Satellite
[2] AVGCi       00:00:05        Not used        On      Modem; Aux (0B1)
[3] 0Ci         00:00:05        Not used        On      Modem

 [0] Exit
 >
```

By clicking the character between brackets in front of the concerned parameter brings you directly to the parameter configuration screen (e.g. to change a parameters data output mode).

### 4.29.5 Alarm list

This gives an overview of all alarms used. See example below:

```
Alarm list

Code    Alarm SMS       Alarm log       Alarm output    Alarm samples

PM      Disabled        Off             Disabled
RPT     Disabled        Off             Disabled
TPT     Disabled        Off             Disabled
AIN     Disabled        On              On              0
MINVi   Disabled        Off             Disabled
OCi     Disabled        Off             Disabled
PTi     Disabled        Off             Disabled
CNT     Disabled        Off             Disabled

[0] Exit
>
```

### 4.29.6 Alarm limits

This gives an overview of alarm limits, used in the configuration. See example below:

```
Alarm limits

Code    Low-low Low     High    High-high       Hysteresis

PM
RPT
TPT
AIN     N/A     20      50      N/A             0.5
MINVi
OCi
PTi
CNT

[0] Exit
>
```

## 4.30 Maintenance Menu

This menu is used to perform several diagnostic tests and includes tools for the maintenance of the data logger. Below the maintenance menu is shown:

```
Maintenance

[0] Exit
[1] Field testing
[2] Comport redirect
[3] Data download
[4] Format SD card
[5] Configuration download
[6] Configuration upload
[7] Firmware upgrade
[8] Modem maintenance
[9] Bootloader menu
>
```

It is recommended that the user first consult this menu and run some tests, before deployment. Or, in the case of failure after proper operation, this menu has to be consulted first. The user can enter this by using either USB or remotely with TCP.

### 4.30.1 Field Testing

This is a very important menu which holds several test-tools. They are discussed below:

#### 4.30.1.1 Verify Analog inputs

This feature allows the user to check the operation of the analog inputs. It shows the exact voltage, measured at the input. It shows the plain voltage / current without the addition of multipliers/offset values, that are present in the configuration. The use of this feature is to let the user check the calibration of the data logger against a well-known voltage/current source. Below a screenshot of this test is shown:

```
Verify analog inputs

Observe values refreshed every second

When finished press any key

Internal Vref  = 1.2061 V  (ADC = 1497)
Analog input 1 = 0.0046 mA (ADC = 0)
Analog input 2 = 0.0044 mA (ADC = 0)
Analog input 3 = 0.0043 V  (ADC = 1)
Analog input 4 = 0.0070 V  (ADC = 2)
Analog input 5 = 1.6498 V  (ADC = 2047)

Are the analog input values OK?

Choose yes/no >
```

The internal reference must always be 1.2 Volts +- 0.1 %. For using this feature in the field, just disconnect your sensor and attach your calibrator and check the value.

#### 4.30.1.2 Digital input test

This test allows the user to test the proper operation of the digital inputs.
Below a screenshot is shown:

```
Digital input test

Apply interrupts on the Digital input pin

When finished press any key

Port 1 Counter = 0
Port 2 Counter = 0
Port 3 Counter = 0
```

The pulses generated during this test will not interfere with your real measurements, defined in the configuration.

#### 4.30.1.3 SD card test

Use this feature to test the SD card. There will be data written on the card, and afterwards erased.
Below a screenshot is given:

```
SD card test

Data log file size = 4578 KB
Email file size    = 0 Bytes
FTP file size      = 0 Bytes
TCP file size      = 0 Bytes
Watchdog file size = 162 KB
Free disk space    = 1896 MB
File write/read OK

Press any key! >
```

*The information from this test is very important. Not only the health of the SD card is tested, but also if there are some exception logged. Exceptions, where the watchdog was called to end a certain situation must NOT exist frequently. So, the size of the watchdog-file tells a lot about the operation of the system.*

#### 4.30.1.4 Battery test

This test is for evaluating the battery condition of Lithium Batteries only.
The internal resistance is measured, by measuring the voltage of the cell, while increasing current consumption. A battery with an internal resistance of >1500 m ohms must be replaced.

```
Battery test

Battery voltage = 3.585 V
Battery current = 57 mA

Increasing power consumption...

Battery voltage = 3.517 V
Battery current = 157 mA
Battery resistance = 680 mOhm

Was the Battery test OK?
Choose yes/no >
```

*Since the condition of rechargeable batteries changes with the state of charge, it is no use to test the batteries with this tool. Also, the internal resistance is in another region, so use this feature with lithium (LSH-20 form Saft) only!*

#### 4.30.1.5 COM port terminal

This feature allows the user to communicate with the sensor directly. It offers a transparent connection, from the USB terminal screen, to the sensor. Baud rate settings can be selected, to match the connected sensor. During the session, the sensor power switch of the data logger is enabled, to supply the sensor. So, a very convenient way of testing, troubleshooting, configuring your sensor is possible.
Also comport redirect can be used for upgrading firmware of your sensor, through the data logger. The software package of the sensor can be used, after comport redirect is enabled.

The following ports are possible (if equipped with):

```
COM port terminal

[0] Exit
[1] Serial port       >> Not
used
[2] Accessory port    >> Camera
[-] 0B Serial port    >> N/A
[-] 0B Accessory port >> N/A
[-] 0B BLE port       >> N/A
[M] Modem
```

The following baud rate/modes are supported:

```
Port mode

[0] Exit
[1] RS232 8N1
[2] RS485 8N1
[3] SDI12
[4] RS232 7E1
[5] RS232 8E1
[6] RS232 801
[7] RS485 7E1
[8] RS485 8E1
[9] RS485 801
[A] RS485 8N2
>
```

#### 4.30.1.6 Data Download

This is used to extract the logged data from the data logger, using the USB interface. (you can do this remotely, but that makes no sense, since the data logger is able to send the logged data also.
The user can select 4 different file-formats: TXT, JSON and CSV (2 different versions available).
The user must enter the starting- and ending date/time for the data he likes to extract. Y-modem transfer is used for data transfer data transportation.

Below an example of data download is given.

```
Data download

Start date or <CR> for this day (YY/MM/DD):
Start time or <CR> for midnight (hh:mm:ss):

End date or <CR> for this day (YY/MM/DD):
End time or <CR> for midnight (hh:mm:ss):

Download data from 2016/02/02 00:00:00 till 2016/02/03 00:00:00

Data format (0 = Native(txt), 1 = CSV(.,), 2 = CSV(,;), 3 = JSON): 3

<15:37:42>

Searching...

Start record found: S;160202104917;POWER_ON;ML-315 V2.2B1

Copying...

Press <Esc> to abort copying data to output file

<15:37:46>

Select Y-Modem transfer from your terminal program menu to receive the
data file "ML-417_.json"

Press <Esc> to abort transfer

Sending...

Data file sending OK

Press any key! >
```

Afterwards, the file is downloaded on your computer.

#### 4.30.1.7 Format SD Card

Use this tool if you like to delete all the files on the SD card, for instance when you move the data logger to a new location. It uses FAT32 to format the card, and you don't need to extract the card from the data logger. After format the card will be blank, ready for new data storage.

#### 4.30.1.8 Configuration Down/upload

When using multiple data loggers, the configuration of them can get labour-intensive. Especially when multiple data loggers are performing the same tasks. To assist the user with configuring the loggers, configuration transfer is possible. The configuration of the data logger can be retrieved via the USB or remotely through TCP Terminal. Use YDOC-Terminal for retrieving the config from your data logger. When you like to retrieve a configuration-file from the data logger, select the option "configuration download" and use Y-modem -> receive file option. Navigate to your path of preference, and the configuration file will be downloaded to your computer. This file is a non-readable-binary file which contains all settings of your data logger. You can store this for backup-reasons or as a template to configure new data loggers.

When you like to configure a data logger from a configuration-file, select configuration upload from the menu and use y-modem-> send file for sending the configuration-file to the data logger. A configuration file can be a binary file containing a complete configuration (except the system name, MQTT root topic or the serial number, which is a static property) or a file containing just some changes by means of tiny configuration snippets (See chapter configuration snippets).

A very practical method of configuring YDOC data loggers is to use one configuration-file to configure the data logger and afterwards personalize the logger with a unique system name manually.

#### 4.30.1.9 Firmware Upgrade

Select this option for upgrading firmware. More on FW upgrade see: Firmware Upgrade_Firmware_Upgrade

#### 4.30.1.10 Modem Maintenance

This feature allows the testing of the modem and also supports modem firmware upgrade. This is a seldom used feature, but it is important in some rare occasions. For more info see: Modem Firmware Upgrade_Modem_Firmware_Upgrade

#### 4.30.1.11 Bootloader Menu

This menu is for expert-use only. It allows the user to jump to the bootloader, where the firmware is NOT loaded yet. Normally, the user does NOT use this menu or its options. Only when the logger has severe problems, this menu can help. (it is this like the old rescue disk you received with your windows computer.)

## 4.31 Users & rights

When the logger is accessible from a public location or by BLE, its recommend to specify a local password.

```
Users & rights

[0] Exit
[1] Administrator password >> ******
[2] Operator password      >> ******
[3] Operator rights        >> Parameter view; Data download
>
```

### 4.31.1 Administrator

An Administrator has all rights, including the right to upgrade the firmware and make changes to the configuration and specify an Operator password.

**Note:** When the local Administrator password is forgotten, it can be cleared/set remotely or by erasing the configuration by placing a file on the SD-Card named “erase.cfg”.

### 4.31.2 Operator

An Operator has limited rights as specified by the Administrator. The following rights can be specified:

- Configuration edit
- Parameter edit
- Parameter view
- Data download
- Firmware upgrade

## 4.32 SDI-12

The ML-x17DS or ML-x17ADS is provided with and SDI-12 port, but it is shared with the RS-232 and RS-485 port. You can add an (additional) SDI-12 port to any ML-x17 by means of an ML-OI-COM-SDI12 option board.

When SDI-12 is selected, it acts like a SDI-12 recorder and its specific SDI-12 commands are embedded in the driver of the input-sensor. So, the user can easily select his sensor and specify its SDI-12-adress. For more information see the description of your SDI-12 sensor.

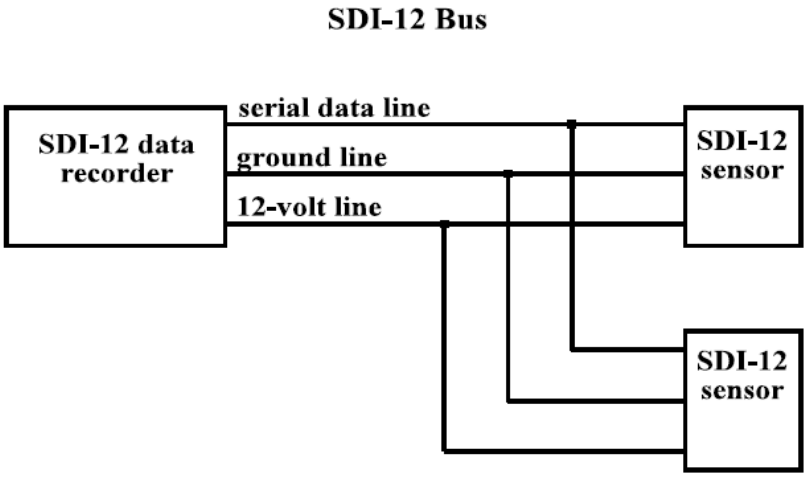
### 4.32.1 SDI-12 Hardware

The SDI-12 standard is a very commonly used interface-standard. The signal levels are quite different from those of RS232 and RS485. So, you cannot connect a SDI-12 sensor to a RS232 or RS485 port, it won’t work. The use of convertors between RS232/485 and SDI-12 is discouraged, because of the high pricing of the convertors and the bad performance (see note).

### 4.32.2 SDI-12 Wiring

The SDI-12 electrical interface uses the SDI-12 bus to transmit serial data between SDI-12 data recorders and sensors. The SDI-12 bus is the cable that connects multiple SDI-12 devices. This is a cable with three conductors:

1) a serial data line, 2) a ground line and 3) a 12VDC line




The wiring length between a sensor and the data-recorder must not exceed 60 meters. The maximum number of sensor connected to a SDI-12 bus is limited to 10. The data logger is protected against transients on the SDI-12 bus.

### 4.32.3 SDI-12 Baud Rate and Frame Format

The baud rate for SDI-12 is 1200. Frame format is as follows:

1 start bit
7 data bits, least significant bit transmitted first
1 parity bit, even parity
1 stop bit

**Note:**

SDI-12 is a half-duplex protocol, so the data-recorder has to switch between transmitting and receiving. A convertor from RS232 ⇔ SDI-12, must perform this task. However, it is not aware of the exact timing of the protocol. Therefore, it uses fixed (or configurable) delays to switch between TX and RX. After each byte send by the convertor, it waits, during the fixed delay, for another character, and if it doesn't arrive, it switches to RX. The intelligence needed to perform these tasks is mostly done by a microcontroller inside the convertor, that's the main reason for its high pricing. This method is doing the job for most cases, but it is not as good as a real SDI-12 port. The real SDI-12 port is aware of the exact protocol-timings and after the last character it switches to RX-mode immediately, without the delay. Therefore, no replies are missing. Your YDOC data logger has a true SDI-12 port.

For more information on the SDI-12 protocol see: www.sdi-12.org

## 4.33 RS232

The ML-x17DS or ML-x17ADS is provided with and RS-232 port, but it is shared with the SDI-12 and RS-485 port. You can add an (additional) RS232 port to any ML-x17 by means of an ML-OI-COM-RS232 option board.

RS232 is a widely spread interface standard, which uses 3 wires (minimum) for data communication. It is a so called asymmetric interface, that uses one wire for TX, one for RX and one for ground. It is called asymmetrical, because it uses only one wire per signal. Therefore, it's susceptible for interference, and hence, the maximum cable length is limited to 15meters.

Please keep in mind these limitations when you design your system.

RS232 is not a bus system, and therefore it is only allowed for one device to be connected to a RS232 port. So, the maximum number of serial devices to connect to your ML-x17 is 1. RS232 sensors should be connected to the data logger with their signals crossed. That is RX ⇔ TX.

We strongly recommend using RS485 instead of RS232, when cable length exceeds the 15m.

## 4.34 RS485

The ML-x17DS or ML-x17ADS is provided with and RS-485 port, but it is shared with the SDI-12 and RS-232 port. You can add an (additional) RS485 port to any ML-x17 by means of an ML-OI-COM-RS485 option board.

RS485 is a serial bus-system, which uses 3 wires for its communication. It uses a “differential balanced line”, which can span relatively large distances (up to 4000 feet (1200 m)). A rule of thumb is that the speed in bit/s multiplied by the length in meters should not exceed $10^8$. Thus a 50-meter cable should not signal faster than 2 Mbit/s.

Instead of RS232, RS485 is capable of communicating with more than one device. After all, it is a bus-system.

RS485 sensors are called “slaves” and must have their unique address. The data logger acts as a master and retrieves the information from the slaves. Only one slave can respond to the requests of the master at a time.

To set up your RS485 sensor for use with the data logger, make sure that the address is programmed correctly, and that the sensor address is unique.

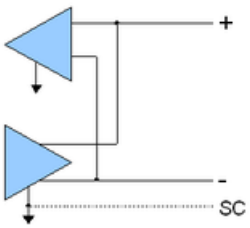



**Figure 1: RS485 Wiring**

RS485 is often used with MODBUS/RTU-sensors, and is less susceptible for Electrical interference than RS232.

Your data logger has one RS485 port which is capable of driving multiple sensors (maximum number of sensors depends on specs from the manufacturer of the sensor, a practical figure is 10. The maximum number of slaves, defined by EIA/RS485 is 32). For the exact number of sensors, you can connect to your ML-x17 see chapter Firmware Driver limitations_Firmware_Driver_limitations

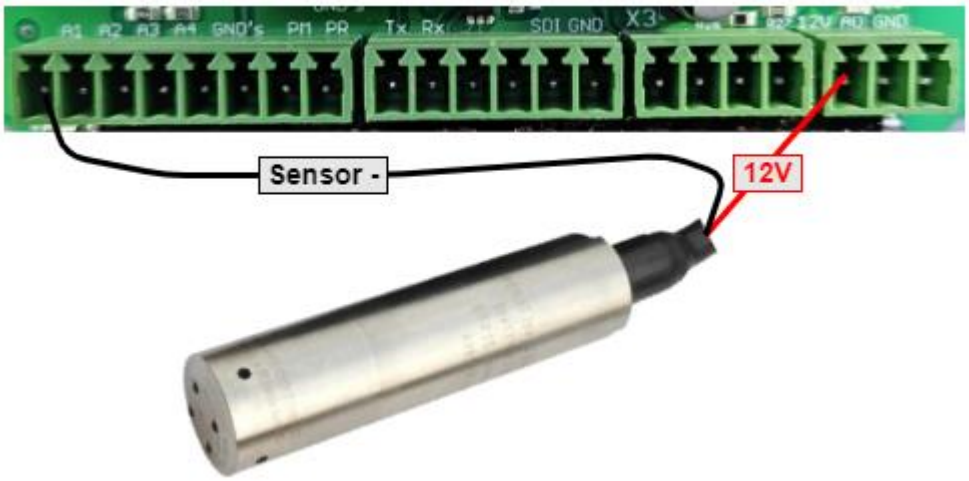
We recommend using twisted pair cable to connect to the sensors.

## 4.35 Analog Inputs (0/4..20mA)

The ML-x17AD or ML-x17ADS is provided two analog 12 bits AD-Conversion current loop inputs. The input signal must be a 0/4..20 mA current loop. The impedance of the system is 150 ohms. You can add additional 0/4..20mA inputs to any ML-x17 by means of an ML-OI-AD-20MA option board.

### 4.35.1 Loop Powered Devices

Some devices don't need a power supply, but take their power from the current loop. But the primary circuit of the data logger does NOT provide power for this. In this case, you need the power switch to provide the energy for the current-loop. Consult the manual of the loop-powered device you like to connect, and use the positive side of the power switch for the power supply. In most cases the data logger is connected to a device with an active output signal, so there will be no problems connecting it. If you have questions about interfacing your device with our data logger, contact your YDOC-dealer.




### 4.35.2 Analog Inputs (0 .. 10 V)

The ML-x17AD or ML-x17ADS is provided two analog 12 bits AD-Conversion voltage inputs. The input signal must be a dc signal which must not exceed 10 Volts. The user can adopt a higher voltage level if he uses external resistors. This is done by a simple voltage divider. However this possibility offers a flexible way to expand the range of the instrument, this is NOT covered in this manual, and the user may not seek for support from YDOC on this topic. You can add additional 0..10V inputs to any ML-x17 by means of an ML-OI-AD-10V option board.

## 4.36 Analog Differential Inputs

An ML-x17 is not provided with differential voltage inputs, but can be added by means of an ML-OI-AD-80MV or ML-OI-AD-2000MV option board.

These inputs are very sensitive and particularly suitable for measuring signals from load cells or pyranometers.

### 4.36.1 Differential input ports theory of operation

Differential inputs are very convenient for measuring differential or floating signals. The performance of a differential input is much better than a normal, single ended one, especially with small mV signals. Therefore, the differential inputs are very suitable for measuring load cells, pyranometers and other low-level mV output sensors. A differential input consists of a negative (-) and a positive (+) input. The voltage difference between these two inputs is the signal to be measured.

### 4.36.2 Common mode noise rejection

One of the major advantages over a single ended input is the common mode noise rejection. It “removes” practically all noise that is present on the input signal. Especially with long cables, noise is always present on the leads. Since the input acts like a differential amplifier, the noise on the negative input is subtracted from the noise on the positive input. What is left is the sensor-signal.

### 4.36.3 Offset calibration

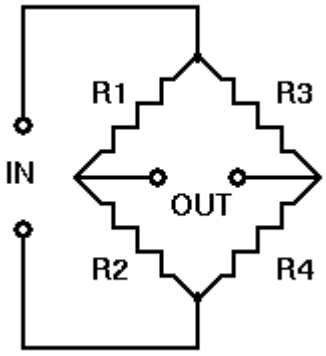
The ML-OI-AD-80MV has an offset of approximately 150 µV, which can be negated by performing a null-point calibration using a short circuit between the differential inputs.

### 4.36.4 Using Load Cells

The ML-x17 when equipped with differential inputs is very suitable for connecting load cells and other resistive elements. A load cell acts as a bridge of Wheatstone and is a very sensitive and precise passive component.

### 4.36.5 Bridge of Wheatstone

The bridge of Wheatstone (the principle of operation of a load cell) is a circuit, consisting of 4 resistors. Below a circuit is given:




## 4.37 Potentiometer input

The ML-x17AD or ML-x17ADS is equipped with a potentiometer input. This input is a 0..3300mV voltage input. It is designed especially for potentiometers, like the ones in winddirection meters and angle meters. This input is connected to the internal ADC directely and is translated into a 0 % - 100% value. The user has to connect the potentiometer between the 3.30V Vref and ground. The ML-x17 has a Vref output for that. (Terminal X1.8). Because of the fact of the potentiometer is connected between Vref an ground, the ADC value will always be the full range (0..4095) INDEPENDENT of the value of the connected potentiometer. So, all types and all values of potentiometers are supported.

Although all values are supported,we strongly recommend to use high values only. This is because the lower values are draining more power from the data logger. Values between 100K and 4M7 are recommended.

A potentiometer gives a value what is translated to 0.. 100%, and can be re-scaled to engineering values by the user. A potentiometer is a variable resistor with three terminals. (see picture below)

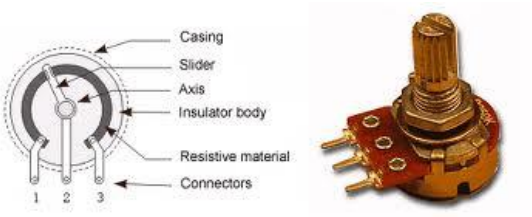
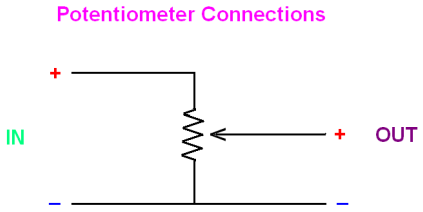



**Figure 2: Potentiometer**

The two outer terminals are connected to the fixed resistor and the middle terminal is connected to a slider. So, when a potentiometer is turned, the resistance between the middle terminal and one of the outer terminals is changing with the angle of the potentiometer. Very often potentiometers are used to regulate a voltage from 0% to 100 % of the input voltage. In the next figure this well know circuit is drawn.




The output voltage is adjustable from 0% to 100% of Vin, in this case. The ML-x17 data logger has 3 terminals (on connector X1) for connecting a potentiometer to the logger:

X1.6 = GND
X1.7 = 0..100% Resistance
X1.8 = Resistance Reference terminal

So, the circuit above is connected as follows:

IN+ = X1.8
GND = - = X1.6
OUT = X1.7

The 3.3V potentiometer reference voltage can also be used as a stable excitation source for connecting thermistors to the potentiometer input or other analog voltage inputs. For more information on using NTC thermistors, refer to the 'Supported Empirical Functions' section.

The resistance reference terminal is derived from the internal ADC reference and has a 150-ohm series resistor (to prevent damage in case of a wiring fault). The internal reference has a level of 3.30 Volts. When relative low impedance potentiometers are used, there will be a voltage drop across the internal series resistor, and the max ADC value won't be reached. Another reason to use only high impedance potentiometers. Of course, when a sensor has got a low impedance value, you can use it, but you have to compensate for the voltage-drop.

## 4.38 Digital inputs

The ML-x17 is equipped with 3 digital input(s). These inputs are interrupt-driven what means that they activate the data logger to wake up when sleeping, and that a signal-change on the input is never missed. So, these inputs are ideal to use for counting events (like the pulses from an energy-meter or a rain gage), or to set an Alarm state (e.g. level or float switch). The signal level needs to be zero volts and 3.6 volts ("0" level and "1" level). The inputs are "5 Volts tolerant" So standard 5 Volts signals are also OK. Any other voltage needs to be adapted to the right range, before connecting. The user can select whether the input has to be "pull-up" or "pull down"

### 4.38.1 Pull up type

The pull up type of input means that there is an internal resistor (40 K) mounted between the input and the Vcc Power supply. So, when NO signal is connected, the input will be logical high. This type of inputs is very convenient for use with "open collector" systems or "NPN outputs"

### 4.38.2 Pull down type

The pull up type of input means that there is an internal resistor (40K) mounted between the input and the ground. So, when NO signal is connected, the input will be logical low. This type of inputs is very convenient for use with "active output" systems or "PNP outputs"

### 4.38.3 Electrical specifications Digital inputs

Below table give an overview of the Electrical specifications of the Digital inputs
Vil is logical low level
Vih is logical high level
Pull up/ down resistors are 40 k typical.

| Parameter | Min | Max |
|---|---|---|
| **Vil** | 0V | 1.25V |
| **Vih** | 1.67V | 5 V |
| **Impedance** | 30K | 50K |

## 4.39 Alarming

In some circumstances, normal data-logging is not sufficient for managing your process. For keeping track of certain, often critical, conditions, the data logger is equipped with direct alarming options. Alarming-limits and hysteresis are used to manage these special events. The table below shows the different types of alarming-limits.

| Alarm Limit | Description | Remarks |
|---|---|---|
| **Low-Low** | Alarm level for lowest value | This alarm level is reached when the data logger encounters a value which is lower than the low-low Limit, this is the 2nd and most urgent state of alarming. This type of alarming is used for very rare and critical conditions. (often called STOP level) |
| **Low** | Alarm level for low value | This alarm level is reached when the data logger encounters a value which is Lower than the Low Limit, but Higher than the Low-Low Limit. This is the first stage of alarming. (often called WARNING level) |
| **High** | Alarm level for high value | This alarm level is reached when the data logger encounters a value which is higher than the high Limit, but lower than the High-High Limit. This is the first stage of alarming. (often called WARNING level) |
| **High-High** | Alarm level for highest value | This alarm level is reached when the data logger encounters a value which is higher than the high-high Limit, this is the 2nd and most urgent state of alarming. This type of alarming is used for very rare and critical conditions (often called STOP level). |

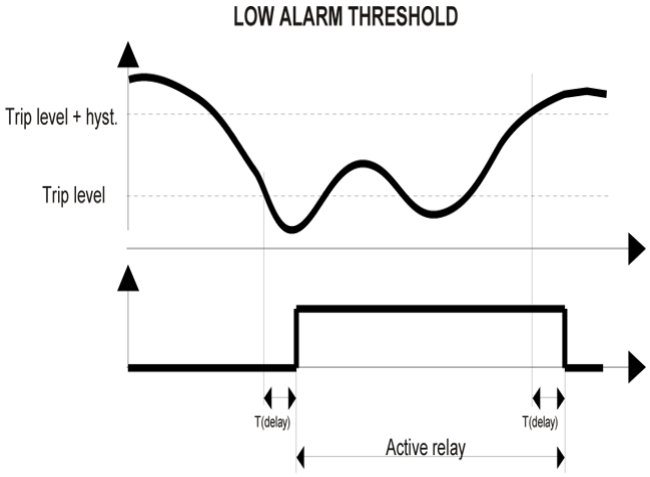
### 4.39.1 Alarming - principal of operation

When a data logger is running, and a measurement is out of boundary, the data logger will immediately switch over to the alarm interval. So, the first-time a –out of boundary-value will be detected is at the normal interval and from this moment on, the data logger will increase its interval to the alarm interval. The “alarm sample delay” determines what happens next. If this value is equal to zero, action is taken immediately. If the interval delay is 1, the logger will wait for one more sample before taking action. If the alarm interval delay is 2, it will take 2 samples, etc.
When this alarm-state is entered, the data logger will add the ‘*A’ data modifier to the data-records. The user has the following options for the requested action:

1) Alarm log (log an alarm level)
2) Alarm message (SMS, E-mail or MQTT)
3) Direct log data output on alarm (HTTP, E-mail, FTP, TCP or MQTT)
4) Activate alarm output.

So, when the conditions of alarming are met (data value out of bounds and the alarm sample delay is counted down) one or more of these actions can take place.

**Note:** The alarm log is a log of synoptically data into an S-record. The data modifier *A is automatically placed into the D-records.




So, the entry of an alarm state is NOT affected by a hysteresis. Hysteresis is used exclusively to switch back to normal mode. The level of hysteresis influences the sensitivity of the alarm system, reducing the likelihood of rapid, repeated alarms. Introducing a certain level of hysteresis is strongly recommended to stabilize alarming by preventing premature 'resolved' messages from quickly being followed by new alarm triggers. The optimal amount of hysteresis is typically determined based on experience with a given application. Additionally, the return to normal mode can be further delayed by waiting for a few additional samples.

### 4.39.2 Advanced Alarming

From firmware-version V2.1B1 and above, we implemented an extended alarming feature that enables the use of calculated channels. This feature was implemented to calculate meaningful engineering values derived from raw sensor input values, however they could also be used for advanced alarming using multiple input conditions. It was already possible to raise an alarm (SMS or trigger a digital output) on exceeding of configured limits for a single input value (e.g. alarm to warn for reservoir overflow). By using calculated channels, it is now possible to alarm on a combination of input values.

Example:

Assume an open water reservoir for irrigation purpose and you want to be warned when the water in the pond falls below a certain level or when the water level reaches the edge of the reservoir. But why ring any bells if the pond level is just falling below the warning level (due to vaporization) while there is no demand for water or why alarm when water is just over a high limit (due to precipitation) while there is no forced supply.

To avoid unnecessary level alarms, you could define a calculated channel with the following equation:

ALARMLEVEL = gt(:FLOWIN; 0; :LEVEL; gt(:FLOWOUT; 0; :LEVEL; LEVELOK))

Where:
gt(a; b; c; d) is a function that returns the value c when a>b otherwise it returns d.
:FLOWIN is the monitored supply flow or a digital switch indicating an open valve/floodgate
:FLOWOUT is the monitored demand flow or a digital switch indicating an open valve/floodgate
:LEVEL is the monitored level in the pond
LEVELOK should be substituted by a level value that does not ring any bells, a pond level somewhere between low and high limit.
ALARMLEVEL is the defined calculated channel, which will be set to the same value as :LEVEL when there is a supply (:FLOWIN > 0) or demand (:FLOWOUT > 0) flow, else it will return a level that does not ring any bells (LEVELOK value).
Instead of setting low/high limits to :LEVEL you should set the warning limits to the ALARMLEVEL channel.

For safety reasons you could still set low-low/high-high limits to :LEVEL.

## 4.40 Firmware Upgrade

The data logger is equipped with a boot loader, which enables the firmware upgrade feature.
Firmware upgrading allows a user to overwrite the internal firmware of the data logger with a (newer) updated version of the firmware. Firmware upgrading can be done by using:

- USB connection
- Wireless via 2G/3G/4G (TCP, HTTP or MQTT) or optionally BLE
- Via Comport

### 4.40.1 When to use Firmware upgrades

Normally, a user never uses this feature, as long as he is satisfied with the performance of the system.
In time however, the need for additional features may arise. For example, a new serial sensor is introduced on the market and a customer wants to connect this sensor to the data logger. When YDOC has extended the firmware to support that sensor, a new version of the firmware is released. After the user has performed the firmware upgrade, his “old” data logger, now supports the new sensor. Normally, when the system is running fine, and no additional requests exists, we recommend NOT to perform a firmware-upgrade.

### 4.40.2 Firmware upgrade procedure

How to perform a firmware upgrade:

- First download the latest version of the firmware from www.ydoc.biz/datalogger-firmware.html
- Download the YDOC Terminal setup from www.ydoc.biz/datalogger-utilities.html
- Use the menu and select the option “Maintenance”.
- Follow instructions from the menu.
- Use “Y-Modem protocol” to send the new firmware to the data logger.
- Don’t remove the USB nor power from the data logger
- When these steps are completed, the new firmware is active.

We recommend the use of the USB-connection over the wireless function. This is because of possible drop outs in communication. When a firmware upgrade procedure is interrupted, the upgrade will fail, but the unit continues to operate, with the previous version. The USB- connection is faster and more reliable.

### 4.40.3 Firmware upgrade over the air

It is also possible to perform a firmware upgrade over the air. This can be done by ydocTerminal in combination with YDOC Java TCP-server or ydocInsights data collector. It can also be perfomed by ydocTerminal in combination with an MQTT-broker or by HTTP from your webserver.

### 4.40.4 Firmware Driver limitations

The data logger is equipped with various drivers, for several tasks. The number of total drivers is limited to 16. This means you can choose maximum 16 drivers from all the drivers available. Each driver can handle (collect or send) a number of parameters. The total number of parameters is limited to 64. So, theoretically, the data logger can handle up to 16 sensors with each 4 parameters. But in this case, there is no output possible. In most cases two or three drivers are needed for minimum operation (internal driver, tcp output-driver, or email-driver). When a user has an additional analog sensor, it will take a driver as well.

*Example:*
A user wants to connect a large number of INW CT2X sensors to an ML-x17. This sensor measures temperature and conductivity. So, each sensor “uses” two parameters. Also, he likes to send the data via

FTP, and he enables the following internal sensors: Rest Capacity, Battery Voltage, and Current. So, besides the Sensor drivers there are 2 drivers needed:

- Internal Driver
- Output Driver

So, the maximum number of Sensor-drivers is: 16 – 2 = 14
So, there are 14 Sensor-drivers left for use with the CT2x sensors.
The total number of parameters is: 14 * 2 + 3= 31
So, the number of parameters is no problem (31 < 64)
He can connect 14 CT2X sensors max.

### 4.40.5 Power Switch Limitations

The Power Switch is capable of powering sensors up to 200 mA. The output voltage is 12 Volts DC factory default. You can alter the output voltage to 24V at max 100mA by bridging solder jumper SJ1 on your own risk.

### 4.40.6 Modem Firmware Upgrade

The modem inside the data logger is equipped with the latest firmware-version available. There is no need for upgrading the software, even in the near future we don't expect this to be needed. But just in the unlikely event of the need of a modem firmware, it is possible to do so. This can be done with the modem mounted on the board (in circuit programming). The procedure of upgrading modem firmware is not covered in this manual. For more information, consult your local YDOC dealer.

## 4.41 SD-card

The SD-card used with the data logger is an 8GB type micro SD-Card, but can be replaced by smaller or bigger SD-cards up to 32GB. It is formatted in FAT 32, and is compatible for use with a PC. Don't use other cards than this type, because the performance of the data logger may be harmed. (This is because of the speed of the SD-card, and even the low power performance can be affected by using a slower card). The contents of the card may be read on a PC by using a card adaptor, or can be downloaded from the card, by using the menu-option "Data-download". We recommend not removing the SD-card from the logger.

### 4.41.1 Inserting an SD-card

At the factory, the SD-card is already installed, but when the cards need to be re-inserted, pay attention to the orientation of it. It should be inserted with the (gold-plated) terminals up.

## 4.42 Native TXT Data Format

The log file exits out of plain text record lines terminated by a carriage return line feed. Record fields are separated with a semicolon and the dot is used as decimal separator.

Every log file starts since FW 4.7B3 with two header records instead of one, the I- and the L-record. The remainder of the file exists out of multiple D- and/or S-records.

#### 4.42.1.1 Header:

The I-record provides information about the data logger and has the following syntax:

<'I'> <';'> <Model><space><FW version><';'><Name><space><S/N>[<';'> <IMEI#> <';'><SIM#>]

Example I-record:

```
I;ML-E417ADS V4.7B2;Station-01 1234567;IMEI 358310512720047;SIM
89310815210985988837
```

This means that the line starts with an 'I' character, followed by hardware model with firmware version, Station name with S/N and optional in case of modem transfer IMEI- and SIM-number.

The L-record provides information about the logged parameters and has the following syntax:

<'L'> <';'> [ <Parameter Code> <';'> < Parameter Name > <';'>  < Parameter Unit> <';'>]

This means that the line starts with an 'L' character, followed by the code, name and unit of a logged parameter. These last 3 elements are repeated for each logged parameter.

Example L-record:

```
L;RCi;Rest Capacity;%;PTi;Processor
Temperature;C;Vi;Voltage;V;AVGCi;Average Current;mA;0Ci;0perating
Cycle;sec;S%;GSM Signal;%;MAXCi;Max Current;mA
```

#### 4.42.1.2 Parameter Code:

The abbreviation of the full parameter name, it may be up to 7 characters long.

#### 4.42.1.3 Parameter Name:

The name of the logged parameter, it may be up to 31 characters long.

#### 4.42.1.4 Parameter Unit:

The unit representing the physical dimensions of the measurement, it may be up to 15 characters long.

All characters are allowed except ';' this is reserved for the field separator.

### 4.42.2 D-Records

Most of the data is logged into D-records. D-records stand for Data records. The syntax of this record is;

<'D'> <';'>,<Timestamp> <';'> [ <Parameter Code> <';'> < Parameter Value >[<Data Modifier>] <';'>]

So, the D-record consists of a 'D' character followed by a timestamp, and after that, one or more series of parameter code and parameter name. So, the D-record consists of these elements:

#### 4.42.2.1 Parameter Code:

The abbreviation of the full parameter name, it may be up to 7 characters long.

#### 4.42.2.2 Parameter value:

The numeric value of the measurement, together with a optional data modifier, forms the measurement.

Example Data-record(s):

```
D;1109280302005;RCi;95.8;PTi;50.1;Vi;3.6;AVGCi;71;0Ci;0.25;MAXCi;71
D;1109280303005;RCi;95.8;PTi;49.3;Vi;3.6;AVGCi;71;0Ci;0.25;MAXCi;72
D;1109280304005;RCi;95.8;PTi;49.5;Vi;3.6;AVGCi;72;0Ci;0.25;MAXCi;72
D;1109280305005;RCi;95.8;PTi;49.1;Vi;3.6;AVGCi;72;0Ci;0.25;MAXCi;54
D;1109280306005;RCi;95.8;PTi;49.1;Vi;3.6;AVGCi;54;0Ci;0.25;MAXCi;72
```

### 4.42.3 System records

The System-records are used to log system related information. System-records do not contain normal measurements. A system-record is made, when a deviating situation has occurred, for instance, when a sensor is not replying to a request from the logger. System-records are used for monitoring system-performance.

The syntax of an S-record is:

S;<Timestamp>;<Message code>[;<Supplemental code or text;["<Explanatory text>"]]
So, the S-record consists of an 'S' character followed by a timestamp, and after that, one or more series of parameter code and parameter name.

So the S-record consists of these elements

1. S (to distinguish a system record from a data record)
2. Timestamp (yymmddhhss)
3. Message code
4. Optional supplemental code or text
5. Optional explanatory text

See Appendix 'System messages' for a list of possible messages and their meanings.

Example S-records:

```
S;1109222020545;CFG_RESET
S;1109222020545;CFG_CHANGED;Brasil_3002389
S;1109222021245;MODEM_WDT;STATE 10
```

### 4.42.4 Data Modifiers

Normally a data value, presented in D-records is recorded without a Data Modifiers, but in case of a malfunction, or rare circumstances, a Data Modifier is added to the data value. See table below:

| Data Modifier | Description | Remarks |
|---|---|---|
| ***T** | Timeout | The sensor did not provide the data logger with a data value, and the timeout has expired. The previous data value is recorded, with the addition of this exception. |
| ***I** | Data Invalid | The data logger did receive a data value from the sensor, but it was out of boundary. This exception is very rare. |
| ***A** | Alarm Value | The data logger has received a value, which is outside the limits of the particular parameter. |

## 4.43 JSON Data Format

The JSON Data Format contains about the same info as our native TXT data format, but formatted in JavaScript Object Notation (JSON) format. This format is preferred by web developers as the data can be easily accessed by JavaScript or other scripting languages like PHP. When using ydocInsights we recommend using our native TXT format as it has lesser overhead resulting in lower data payload (costs) and data transfer time (power consumption).

The JSON object contained in the file exists out of three main objects:

1) The **"device"-object** is having three variables:
   1. "sn" a string giving the unique serial number of the logger.
   2. "name" a string giving a user chosen name for the data logger.
   3. "v" a string giving the version of the Firmware

```
"device":{"sn":5152860,"name":"Demo 1","v":"2.2B2"},
```

2) The **"channels"-object** is an array of objects listing all configured data log parameters, each channel/parameter object is having 3 variables:
   1. "code" a string specifying the Parameter's user defined code, e.g. PTi
   2. "name" a string specifying the Parameter's user defined name, e.g. Processor Temperature
   3. "unit" a string specifying the Parameter's user defined unit, e.g. °C

```
"channels":[
{"code":"PTi","name":"Processor Temperature","unit":"°C"},
{"code":"CHx","name":"Channel x","unit":"Unit x"},
{"code":"SB","name":"Signal quality","unit":"bars"},
{}],
```

The "channels" array is, for convenience, terminated by an empty object {}

3) The **"data"-object** is an array of objects listing timestamped events, a timestamped object always starts with a timestamp variable "$ts", followed by one or more variables, where a variable can be a logged channel value or a system (error) message. A system message variable is always identified by $msg and a channel variable is always identified by its Parameter code (see: "channels"). Variables not being $msg or not occurring in the "channels" array can be ignored.

The "data" array is, for convenience, terminated by an empty object {} as well.

```
"data":[
{"$ts":160225105846,"$msg":"POWER_ON;ML-215;V2.2B2"},
{"$ts":160225105900,"CHx":"0*T", "PTi":23.5},
{"$ts":160225110000,"SB":4},
{}]
```

$ts is a timestamp formatted as a number yymmddhhmmss
$msg is a string giving some system message (See Appendix 'System messages' for a list of possible messages and their meanings). Other variables starting with $ can be added in the future and can be ignored.

Variables NOT starting with $ are considered to be logged channel variables of which the variable name should occur as "code" in the "channels" array.

A channel variable can be a number (the logged value) or a string being the logged value concatenated with a data modifier. (See: 4.35.3 Data Modifiers)

## 4.44 Compacted Data Format

The compacted data format is meant for use in case of message size constrained or payload expensive communication, like satellite communication.

Compacting is required to get sufficient measurements in a size constrained message or to reduce expensive payload costs.

A common method to compact data is by using algorithms like “zip” and “rar”, but those algorithms are not suitable to compress size constrained messages and would make the payload rather bigger than smaller.

A first stage in compacting, is eliminating data that is not strictly necessary and to include the real interesting data only. The compacted data contains timestamped measurement values only, it does not contain diagnostic data or supportive info like system identifications, parameter codes, names or units.

The compacted format is limited to contain up to 15 logger parameters/channels. As textual parameter codes would use to much data, the identification of the individual channels is based on a number between 1 and 15. Assigning the numbers to individual parameters is a matter of configuration by starting their code names with a capital P followed by a number between 1 and 15. The P will be stripped of, as well as any trailing characters, so you could specify meaning full codes like: P1TMP or P02HUM or P15LEV. Make sure that the number parts are unique.

The compacted data is a binary and unreadable format as representing times and values in ASCII format would take too much space. Storing values as IEEE floats is out of the question either, because every value would take at least 4 bytes, while less bytes would be sufficient in most cases.

The compacted data exists out of time and value records. Each record specifies the delta to the previous corresponding record. So, in case of big numbers (like with time or totalizers) only the first occurrence will take the max required bytes, while consecutive occurrence might need just one byte.

When a time record is specifying a delta of 5 minutes, all value records following will be 5 minutes younger than the ones before. When a value record is specifying a delta of 0 it means that the measured parameter value did not change since the previous measurement.

The starting reference time for each message is 1-jan-2017 00:00:00. The first record in the message will be a time record specifying the difference in time since 1-jan-2017 00:00:00 for the first value records.

The starting reference value is 0.0 for each individual parameter.

**Note***: The “Source” of the “Compacted Data” can be identified by the info contained in the encapsulating “Delivery Message” (e.g. the IMEI number of an Iridium Satellite Transceiver). The format and transfer method of the “Delivery Message” is dependent of the used “communication provider” (e.g. an Iridium TCP-direct message).*

### 4.44.1 Records

A record has a variable length.

#### 4.44.1.1 First byte of record

| Bit 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| Type of record | | | | | | | |

The most significant nibble of the first byte describes the type of record, the least significant nibble is record type dependent. See paragraphs “Time/Value-record” for more details.

The first byte can be followed by zero or more bytes depending on the record type.

#### 4.44.1.2 Number-bytes

The “number”-bytes form an unsigned integer (mantissa), which can be converted to a real number by applying the scaling and sign.

| Bit 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| Following byte bit | 7-bits value * $2^0$ (0 to 127) | | | | | | |

| Bit 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| Following byte bit | 7-bits value * $2^7$ (128 to 16383) | | | | | | |

| Bit 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| Following byte bit | 7-bits value * $2^{14}$ | | | | | | |

| Bit 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| Following byte bit | 7-bits value * $2^{21}$ | | | | | | |

| Bit 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| 0 | 7-bits value * $2^{28}$ | | | | | | |

Each “number” byte contains a 7-bits value (0 to 127) and a bit to indicate that another “Number” byte is following or not. If “1” then another 7-bits value is following, which should be multiplied by 128 ($2^7$) and added up to the first byte, if there is a next byte it should be multiplied by 16384 ($2^{14}$) etc, etc., until the “following byte” bit is set to “0”. When the “following byte” bit is “0”, then a next record starts unless the end of message is reached.

**Number examples:**

- 00000001 = 1
- 10000001 00000010 = 1 + (2*128) = 257
- 10000001 10000010 00000011 = 1+(2*128)+(3*16384) = 49409
- 10000001 10000010 10000011 00000100 = 1+(2*128)+(3*16438)+(4*2097152) = 8438017

#### 4.44.1.3 Time-record

| Bit 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| Type of record=0 | | | | Sign | Single byte | Scaling value | |

When the most significant nibble of the first record byte equals 0, then the record specifies a change in timestamp for all following value records.

Bit 0 & 1 indicates the scaling of the “Time”-record: 00= Day, 01=Hour, 10=Minute and 11=Second.

Bit 2, When this bit is set to “1” it indicates that there are no “Number”-bytes following this record, the record exists out of just one byte where the value equals to the scaling value (e.g. 1 day). When this bit is set to “0” the delta time is given by the value constructed from the following “Number”-bytes.

Bit 3 is the sign bit and when set to “1” it indicates that the time difference given in this record should be subtracted from the current timestamp and when “0” added to it.

Important: When the sign-bit of the first record in a compacted message is set, the total message should be ignored or processed as a picture (jpg) transfer record. See paragraphs “JPG-header/data-record” for more details.

#### 4.44.1.4 Value-record

| Bit 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| Parameter number (1 to 15) | | | | Sign | Scaling value | | |

When the most significant nibble of the first record byte is unequal to 0, then the record is about a value measured at the current timestamp of a parameter with the specified number.

**Note**: a “Value”-record has at least one following “Number”-byte.

Bit 0,1 and 2 is a 3 bits scaling value (a value between 0 to 7). To get the real number, the integer number, constructed from the following “Number”-bytes, should be divided by 10 to the power of the scaling value. (e.g. 1= divide by 10, 2=divide by 1000).

Bit 3 is the sign bit and when set to “1” it indicates that the given delta value in this record should be subtracted from the current parameter value and when “0” added to it.

#### 4.44.1.5 JPG-header-record

| Bit 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| Type of record=0 | | | | 1 | 0 | Scaling value | |

The most significant nibble of this record is 0 and can be distinguished from a Time-record, because it can only be a first record in a message with bit 3=1 and bit 2=0.

Bit 0 & 1 indicates the scaling of the “Timestamp”-value of the picture.: 00= Day, 01=Hour, 10=Minute and 11=Second.

The first byte is followed by 5 values (Number-bytes series): Timestamp, Reserved value, Sequence#, File size and CRC16

**Timestamp** is an integer value representing the time in number of days, hours, minutes or seconds since 1-jan-2017 00:00:00

**Reserved value,** this value is reserved for future use and should be 0. If not 0 the assembled file should be discarded as it might not be a valid JPG picture.

**Sequence#** this number is used to be able to correlate JPG-header and JPG-data records belonging to the same JPG-picture. To minimize payload the data logger will probably wrap this number at value 100.

**File size**, the size of the file in bytes. This number can be used in your administration together with the sequence# to determine if all messages to assemble a complete picture are received.

**CRC16**, This values is a CRC16 (same as used for MODBUS/RTU) calculated over all bytes of the file contents.

The JPG-header record will immediately be followed by a JPG-data-record consuming the remaining space of the compacted message.

#### 4.44.1.6 JPG-data-record

| Bit 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| Type of record=0 | | | | 1 | 1 | Reserved | |

The most significant nibble of this record is 0 and can be distinguished from a Time-record, because its directly following a JPG-header-record or it’s the first record in a message. It has bit 3  2 set to 1.

The first byte is followed by 2 values (Number-bytes series) : Sequence# and File position

**Sequence#** this number is used to be able to correlate JPG-header and JPG-data records belonging to the same JPG-picture.

**File position**, this value indicates the position in the file where the data bytes should be written to.

**Data bytes**, the File position value is followed by picture data bytes up to the end of the compacted message.

## 4.45 Sparkplug-B Data format (V1.0)

Sparkplug™ provides an open and freely available specification for how Edge of Network (EoN) gateways or native MQTT enabled end devices and MQTT Applications communicate bi-directionally within an MQTT Infrastructure.

This chapter details how Sparkplug™ can be deployed with YDOC data loggers as EoN Nodes.

Because Sparkplug defines a dedicated topic namespace, all other topics used by the data logger can coexist, not jeopardising the native features like remote configuration update, firmware upgrade, etc.

As well, this chapter details how you can specify 'group_id', 'edge_node_id', 'Metrics' and 'Properties', where its assumed that the reader is already familiar with how to configure an YDOC data logger.

### 4.45.1 Configuring the MQTT-driver

How to configure the MQTT-driver is already described in a previous chapter, so only the items concerning Sparkplug-B will be detailed.

```
MQTT settings

[0] Exit
[1] Name              >> sp-grp-x
[2] Send interval     >> 01:00:00
[3] Send delay        >> Not used
[4] Server            >> m21.cloudmqtt.com
[5] Port              >> 22656
[6] Security          >> SSL/TLS
[7] Username          >> sp-user-y
[8] Password          >> ******
[9] Root topic        >> YDOC/5025064
[A] Client ID         >> 359180082361087
[B] Clean session     >> No
[C] Data output       >> Log data
[D] Data format       >> Sparkplug B
[E] Max payload       >> 1000 kB
[F] Data filter       >> Data & Diagnostics
[P] Input parameters  >> 0
[R] Remove
[T] MQTT test         >> Passed
>
```

#### 4.45.1.1 Root topic

As Sparkplug works with a dedicated topic namespace, the "root topic" of this driver is of no concern to the Sparkplug feature and can be ignored.

#### 4.45.1.2 Name of Group

The name (max 31 tokens) of the driver is used as a place holder for the Sparkplug group_id. In example, if you specified "sp-grp-x" as name, the topic namespace for the NBIRTH and NDATA messages will be generated as follows: "spBv1.0/sp-grp-x/message_type/edge_node_id"

The "edge_node_id" is substituted by the name you specified for the data logger under "General settings". This name (max 31 tokens) can only be entered manually and will not be overwritten by a future configuration update.

When no name for the data logger is specified the unique serial number of the data logger will be taken as “edge_node_name” and the generated topic namespace would look similar to:

“spBv1.0/sp-grp-x/message_type/5025064”

### 4.45.2 Data output

At regular intervals the data logger connects with the server to transfer its readings and depending on the applications and/or power constrains it can be any minute, once a day or something in between. The data logger can be instructed to send just the last known values (actual values) or the total history recorded between now and the previous succeeded transfer session (log data).

In case of “Actual Values” only, the data logger will publish an NBIRTH message with the last know values and gracefully disconnect from the server.

In case of “Log Data”, the data logger will first publish an NBIRTH message with the last know values along with their “aliases” and followed by one NDATA message containing the total history recorded between now and the previous succeeded transfer session. “Aliases” are used to keep the payload limited in size.

***Note**:* The “alias” assignments will not change between sessions, unless something has changed to the configuration of the “Metrics”.

### 4.45.3 Metrics

An YDOC data logger, depending on the edition and installed options, can acquire physical signals by current loop, voltage and digital inputs as well as capture readings from serial sensors by MODBUS, NMEA, SDI-12 or ASCII protocols. It’s also equipped with a calculation and aggregation engine.

A total of max 64 channels/tags, which we refer to as parameters, can be defined. These parameters will be inserted as Metrics at the Node level (NBIRTH and NDATA).

The name and other attributes of each parameter can be customized as needed.

```
Parameter settings

[0] Exit
[1] Name          >> Inputs/A
[2] Code          >> INA
[3] Unit          >> mA
[4] Value factor  >> 1
[5] Value offset  >> 0
[6] Decimals      >> 3
[7] Data log      >> On
[8] Data output   >> Modem
[-] Alarm message >> Disabled
[A] Alarm log     >> Off
[-] Alarm output  >> Disabled
>
```

#### 4.45.3.1 Name of Metric

The name (max 23 tokens) of a parameter is used as a place holder for the name of the concerned Metric. The name of Metric may contain one or more slashes to create a hierarchical structure, however keep in mind the max size of 23 tokens including slashes.

### 4.45.4 Datatype

All parameters will be exposed as datatype 10 (double) even totalizers and digital inputs.

### 4.45.5 Message examples

Please find below examples of YDOC NBIRTH, NDEATH and NDATA messages. Sparkplug-B payload is formatted in Google Protocol Buffer format and therefor human unreadable. We have represented the examples below in JSON format for readability purpose only.

#### 4.45.5.1 NBIRTH

Topic: spBv1.0/sp-grp-x/NBIRTH/5025064

Up to 64 parameters/channels can be included and referred to by aliases 1 till 64. Their names, as described before, are user definable.

```
{
  "timestamp": 1486144902122,
  "metrics":
  [
    { "name":"bdSeq", "timestamp":1486144902122, "dataType":"Uint64",
      "value":0 },

    { "name":"Properties/Hardware Make", "timestamp":1486144902122,
      "dataType":"String", "value":"YDOC"},
    { "name": "Properties/Hardware Model", "timestamp": 1486144902122,
      "dataType": "String", "value": "ML-417ADS" },
    { "name": "Properties/FW", "timestamp": 1486144902122,
      "dataType": "String", "value": "4.4B8" },
    { "name": "Properties/SN", "timestamp": 1486144902122,
      "dataType": "String", "value": "1234567890"},

    { "name": "System/Battery", "alias":1, "timestamp": 1486144902122,
      "dataType": "Double", "value": 3.3}
    { "name": "Inputs/AI1", "alias":2, "timestamp": 1486144902122,
      "dataType": "Double", "value": 14.2}
    { "name": "Inputs/AI2", "alias":3, "timestamp": 1486144902122,
      "dataType": "Double", "value": 9.5}
    { "name": "Counters/CNT1", "alias":4, "timestamp": 1486144902122,
      "dataType": "Double", "value": 1234567}
    { "name":"Calculations/FLW1", "alias":5, "timestamp":1486144902122,
      "dataType":"Double","value":84.7}
  ],
  "seq": 0
}
```

#### 4.45.5.2 NDEATH

Topic: spBv1.0/sp-grp-x/NDEATH/5025064

Please find below an example of the NDEATH message in JSON.

```
{
  "timestamp": 1486144902122,
  "metrics":
  [
    { "name":"bdSeq", "timestamp":1486144902122, "dataType":"Uint64",
      "value":0 }
  ]
}
```

#### 4.45.5.3 NDATA

**Topic:** spBv1.0/sp-grp-x/NDATA/5025064

This payload will contain all values recorded between now and a previously succeeded transfer session. To keep the payload limited the recorded values will be referred to by their aliases as given in the preceding NBIRTH payload.

```
{
  "timestamp": 1486144902222,
  "metrics":
  [
    { "alias":1, "ts": 1486144300000, "dataType": "Double", "value": 3.32}
    { "alias":2, "ts": 1486144300000, "dataType": "Double", "value": 14.0}
    { "alias":3, "ts": 1486144300000, "dataType": "Double", "value": 8.7}
    { "alias":4, "ts": 1486144300000, "dataType": "Double", "value": 1234507}
    { "alias":5, "ts": 1486144300000, "dataType": "Double", "value":79.6}

    { "alias":1, "ts": 1486144600000, "dataType": "Double", "value": 3.31}
    { "alias":2, "ts": 1486144600000, "dataType": "Double", "value": 14.4}
    { "alias":3, "ts": 1486144600000, "dataType": "Double", "value": 9.1}
    { "alias":4, "ts": 1486144600000, "dataType": "Double", "value": 1234537}
    { "alias":5, "ts": 1486144600000, "dataType": "Double", "value":81.3}

    { "alias":1, "ts": 1486144900000, "dataType": "Double", "value": 3.3}
    { "alias":2, "ts": 1486144900000, "dataType": "Double", "value": 14.2}
    { "alias":3, "ts": 1486144900000, "dataType": "Double", "value": 9.5}
    { "alias":4, "ts": 1486144900000, "dataType": "Double", "value": 1234567}
    { "alias":5, "ts": 1486144900000, "dataType": "Double", "value":84.7}
  ],
  "seq": 1
}
```

## 4.46 Configuration snippets

A data logger can be configured manually through the terminal interface or by loading a file containing a complete configuration in binary format. For fast deployment you could build up a list of configuration files for your different classes of monitoring stations (e.g. soil moisture, weather, hydrology, etc.) where each different class uses the same type of sensors/devices. However the need could arise to make little changes afterwards, changes like setting a level offset or an alarm limit and preferable automated from your own application server. Updating these changes could be done by uploading a complete binary config file, but it's a bit cumbersome to maintain many varieties of various classes of configuration on your server or to patch changes in a complex binary configuration file.

As of FW 4.6B1 it becomes possible to make small changes to the configuration of a data logger by uploading "configuration snippets". Snippets are expressed in JSON and can be transferred to a data logger in the same way as a regular configuration, but its content/payload being a tiny readable JSON object rather than a vast unreadable binary structure.

E.g. a snippet to change the data log & alarm interval to one hour resp. one quarter is just a few bytes.

```
{"gs":{"ni":3600,"ai":900}}
```

### 4.46.1 Payload format

The contents/payload should contain valid formatted JSON and the very first byte should be an opening brace **{** else the data logger will not interpret the payload as JSON and report an error.

Objects or members unknow to the data logger will be ignored. Supplying invalid values for known members will be ignored as well, but an error will be reported and stored in the data log.

To keep the payload as small as possible, to support payload constrained media, names of objects, arrays and members are expressed in short abbreviated notation.

The following root objects and arrays can be included in the payload:

- **"gs"** an object { } containing Global system Settings, like system name and data log interval.
- **"as"** an object { } containing Alarm Settings, like SMS alarm phone numbers (since FW4.8B16).
- **"ds"** an array [ ] with Driver Setting objects, like send intervals and warm-up times.
- **"ps"** an array [ ] with Parameter Setting objects, like parameter offset and alarm limits.

#### 4.46.1.1 Global system Settings

The following members can be included in the global settings object:

- **"dn"** a "string" of max 31 tokens providing the device/station name.
- **"ni"** a number providing the normal data log interval in seconds. **ni** can't be any arbitrary value, it should be in discrete divisors of 60 seconds, 60 minutes or 24 hours (e.g. "ni":11 is invalid, while "ni":12 is not.
- **"ai"** a number providing the alarm data log interval in seconds. **ai** should be equal or lower than **ni** and should be in discrete divisors of **ni** (e.g. "ai":17 is invalid when "ni":60 is provided). Specifying 0 will make **ai** equal to **ni**.
- **"tz"** a number in the range of -12 to +13 in steps of 0.25 to provide the time zone.
- **"dt"** a "string" formatted as "(yy)yymmdd hh:mm:ss" to set the deployment time, the data logger will go in deep sleep until the deployment time is reached. This feature could come in handy when the end of a monitoring season and you want to start it at the beginning of the next season.

#### 4.46.1.2 Alarm settings

The following members can be included in the alarm settings object:

- **"r1"** a "string" of max 15 tokens providing the phone# of SMS recipient 1
- **"r2"** a "string" of max 15 tokens providing the phone# of SMS recipient 2

#### 4.46.1.3 Driver Settings

The following members can be included in a driver settings object:

- **"id"** a "string" to identify the concerned driver by its name in the current active configuration.
- **"dn"** a "string" of max 31 tokens providing the new driver name.
- **"ni"** a number providing the driver initiation interval in seconds. **ni** can't be any arbitrary value, it should be in discrete divisors of 60 seconds, 60 minutes or 24 hours (e.g. "ni":11 is invalid, while "ni":12 is not. If the driver is concerning input(sensors) the interval should be equal or in discrete divisors of the data log interval. If the driver is concerning outputs (e.g. modem) the interval should be equal or in multiples of the data log interval. Specifying 0 will make the interval equal to the normal data log interval.
- **"ai"** a number providing the driver initiation alarm interval in seconds. **ai** can't be longer than **ni** and should be in discrete divisors of **ni** (e.g. "ai":1700 is invalid when "ni":3600 is provided). Specifying 0 will make **ai** equal to **ni**.
- **"wu"** a number providing the "warm-up" time in seconds of the connected device, this is the time needed by the external device to provide stable service after applying power.
- **"ct"** to preset a cumulative counter (if the driver has one) to a certain start value, e.g. 0 to reset the cumulative counter of a digital input driver (since FW4.8B16).

#### 4.46.1.4 Parameter Settings

The following members can be included in a parameter settings object:

- **"id"** a "string" to identify the concerned parameter by its code in the current active configuration.
- **"pc"** a "string" of max 7 tokens providing the new parameter code.
- **"pn"** a "string" of max 23 tokens providing the new parameter name.
- **"pu"** a "string" of max 15 tokens providing the new parameter unit.
- **"nd"** an integer up to 6 to specify the number of decimals to display.
- **"vf"** a number providing a multiplication factor to the raw input value.
- **"vo"** a number providing an offset to add to the input value.
- **"iv"** initial value of the parameter if not sampled yet after a re-start.
- **"ll"** a number providing the low-low alarm limit.
- **"lb"** a number providing the low boundary limit.
- **"lo"** a number providing the low alarm limit.
- **"hi"** a number providing the high alarm limit.
- **"hh"** a number providing the high-high alarm limit.
- **"hb"** a number providing the high boundary limit.
- **"ad"** an integer up to 9 specifying how many consecutive samples should be violating limits before an alarm situation is set.
- **"ah"** a number providing the hysteresis before an alarm situation will be reset.
- **"rd"** an integer up to 9 specifying how many consecutive samples should be within limits before an alarm situation will be reset.

### 4.46.2 Payload example

Please find below an example containing multiple snippets.

Note that the comment // is include for informative purpose only and should not be included in real JSON.

```
{
  "gs":                         // Global Settings
  {
    "dn":"Station 1",           //New device name
    "ni":60,                    //New normal data log interval in seconds
    "ai":30,                    //New alarm data log interval in seconds
    "tz":3,                     //New time zone   in hours
    "abc":"xyz"                 //Unknown members will be ignored
  },

  "ds":                         // Driver Settings
  [
    {
        "id":"MODBUS",          //Current name of the addressed driver
        "ni":0,                 //Set sample interval equal to data log interval
        "wu":10,                //New warm-up time of connected sensor
        "dn":"Probe 1"          //New driver name
      },
      {
        "id":"MQTT",            //name of the addressed driver
        "ni":1800               //New send interval
      }

  ],

  "ps":                         // Parameter/channel Settings
  [
    {
        "id":"P3",              //Current code of addressed parameter
        "pu":"F",               //New unit text
        "vf":1.85,              //New value factor
        "vo":-31,               //New value offset
        "pc":"TMP",             //New parameter code
        "pn":"Temperature"      //New parameter name
      },
      {
        "id":"P4",              //Current code of addressed parameter
        "pu":"NTU",             //New unit text
        "vf":1.85,              //New value factor
        "vo":-31,               //New value offset
        "pc":"TURB",            //New parameter code
        "pn":"Turbidity"        //New parameter name
      }
  ],

  "abc":                        //Unknown objects will be ignored
  {
      "xyz":123
  }
}
```

## 4.47 Input-drivers

Input-drivers obtain data from sensors. Various sorts of sensors can be connected to the data logger. When a sensor needs a “warm-up time” the power output switch can be used to power the sensor before the measurement is taken. The maximum time of a power delay is 5 minutes

**Note:**
The Power Switch output is consuming a lot of power, so try to minimize this. Consult the manual of the sensor for warm up times. A warm up time of 5 minutes is possible and can be used in rare situations, but the battery-life will be shortened enormously. In such rare cases, consult your local YDOC-supplier for a calculation of battery-life, before exploiting your data logger.

### 4.47.1 Analog sensors

The **ML-x17** (except the DS edition) is provided with 5 factory-calibrated analog inputs with 12 bit ADC resolution.

| Input | Range | Accuracy | Impedance |
|---|---|---|---|
| **1** | 4 .. 20mA | <0.1% FS | 150 Ohm |
| **2** | 4 .. 20mA | <0.1% FS | 150 Ohm |
| **3** | 0 .. 10V | <0.1% FS | 32 k Ohm |
| **4** | 0 .. 10V | <0.1% FS | 32 k Ohm |
| **5** | Potentiometer (0..100%) or as 0..3300mV input | <0.1% FS | 500 k Ohm |

### 4.47.2 Digital Pulse Sensor

Digital Pulse sensors like for instance rain gauges are based upon the “reed contact” principle. The rain gauge has an internal bucket with a very precise volume. It is constructed to tip over when it reaches a specified amount of water. The water is drained and while the bucked was turning, a magnet triggered a magnetic switch, a so called “reed contact” So, the rain gauge itself works like a passive switch. The data logger has a special input to trigger on these events. Even when the data logger is sleeping, the event of a tipping bucked is never missed. The data logger uses a so called “interrupt-input” to make this possible. To connect a rain-gauge, use this interrupt input and connect the other site of the rain gauge to the 3V6 output.

**Note:**
The digital input offers the most energy-friendly measurements available. This is because the data logger is allowed to sleep most of the time, and only capture the events of the digital interrupt (e.g. the tipping bucket of a rain gauge). In the situation where only one digital sensor is used, the battery is probably going to last much longer than any other measurement. For safety-reasons a user can include some internal measurements in the configuration, to allow monitoring of the performance of the system.

#### 4.47.2.1 Example configuration Rain Measurement

Here an example is shown for a tipping Bucket Rain Gauge, and how to set it up in the data logger.
The Rain-gauge should be connected to the “Digital input”

Underneath the menu for this input is shown.

```
Digital pulse sensor

[0] Exit
[1] Name                    >> Rain
[2] Sample interval         >> Data log interval
[3] Port mode               >> Port 1; Internal pull up
[4] Register mode           >> Pulse (low frequency)
[5] Units per pulse         >> 0.2
[6] Register value          >> 16979 pulses
[7] Register reset          >> Off
[8] Log each counter change >> On
[9] Counter (unit)          >> Counter
[A] Quantity (unit)         >> Quantity
[B] Mean rate (unit/h)      >> Mean Rate
[C] Max rate (unit/h)       >> Max Rate
[D] Min rate (unit/h)       >> Min Rate
[R] Remove
>
```

First, change the name “Digital Pulse” into a more comprehensive one. We use “Rain” here.
The menu-item “pulses per unit” is very important and converts the input pulses into a physical value.
It is advised to test the hardware first, before proceeding to selecting the right settings for bucket-size etc.
Therefore, leave this value (1) and test your sensor first.

To test, just connect it, and apply a known amount of pulses tot the data logger. You can verify this count with the command <Ctrl>A<Shift>V<Ctrl>D When this is correct you can proceed to set up your rain-gauge.

Now you have to enter the physical details of your rain-gauge.
i.e. when your rain-gauge has a tipping bucket with a size of 0.2 mm rain, enter 0.2 Units per Pulse.
You can reset the counter-value, caused by the previous test, if you like. You can do this by entering zero into the register via option 6.
If you like to automatically reset the counter-value at midnight, use option 7.

**PARAMETERS**
There are three parameters for using the digital input:

#### 4.47.2.2 Counter

This is the most important parameter. It’s a plain counter that counts every single pulse, and keeps on counting forever (unless you use midnight reset). The maximum count value is: 4294967295 (2^32) It will reset to 0 when it reaches this count.
The counter is working at all times, even at sleep mode. When the battery is replaced, this value is NOT lost, and is resuming after replacement of the battery.

#### 4.47.2.3 Quantity

Quantity is the difference between the actual counter-value and the previous counter-value. So, when your data-log interval is set to 10 minutes, this parameter shows you the amount of pulses per 10 minutes. Every log-interval, this count is reset to zero.

***Important!***
*So, when you use “Actual values” keep in mind that the parameter “quantity” is a running value. It will increase during the interval. And what you see at that particular moment is NOT the value that will be stored on the SD-card. This value could lead you to incorrect assumptions!!!*

#### 4.47.2.4 Rate

The parameter Rate is defined as the time between the last two pulses applied to the data logger, scaled to one hour. For rain-measurement, the parameter rate can be used for calculating “rain intensity”. It allows you to differentiate a rain-shower from drizzling rain.

**Example:**

So, when two pulses, with a delay of 5 seconds between them, are send to the data logger, and every pulse represents 0.2 mm rain, the rate is: 1 mm per 25 seconds = 144 mm / hour

***Important!***

So, also this parameter is a “running value”, it extrapolates the rainfall in the next hour, based upon an actual situation. So, keep this in mind.

## 4.48 Power supply

The data logger PCB is designed to work with a power supply of minimum 0.8 to maximum 5V DC

The ML-x17 can be supplied with different power provision PCB's mounted in the enclosure cover.

| Power supply | Description |
|---|---|
| **ML-x17y-LI** | Powered from a single 3.6V DC SAFT LSH20 or equivalent D-size lithium battery. |
| **ML-x17y-3LI** | Powered from 3x 3.6V DC SAFT LSH20 or equivalent D-size lithium batteries. |
| **ML-x17y-DC** | Powered from external 8..28V DC source. |
| **ML-x17y-DC-LI** | Powered from external 8..28V DC source or 3.6V DC SAFT LSH20 lithium battery. |
| **ML-x17y-DC-NM** | With integrated 3 x AA NiMH holder & charger for external 8..28V DC source. |
| **ML-x17y-PV** | With integrated 1Wp solar panel and 1x 26650 3.2V LiFePO4 holder &charger. |
| **ML-x17y-LFP** | With 4x 18650 3.2V LiFePO4 holder & charger for 12V (21VOC) ext. solar panels. |
| **ML-x17y-SLA** | With integrated 12V battery charger for 12V (21VOC) external solar panels. |

### 4.48.1 Internal RTC backup battery

The data logger contains an internal coin cell battery to keep the internal real-time-clock running. The lifetime of the battery is at least 10-20 years, so this battery requires no exchange during the lifetime of the data logger.

### 4.48.2 Power consumption & Battery Life

Average current consumption @3.6V

| Subject | Value | Remarks |
|---|---|---|
| **Data logger in low power sleep** | <100uA | Preferred mode of operation. |
| **Data logger in MODEM sleep** | 2mA | MODEM in stand-by during SMS ACVA reception. |
| **Data logger is awake** | 65mA | The logger is awake to be able to take and log a measurement. |
| **Data logger is transferring GPRS data** | 220mA | Requires a good GPRS signal. |

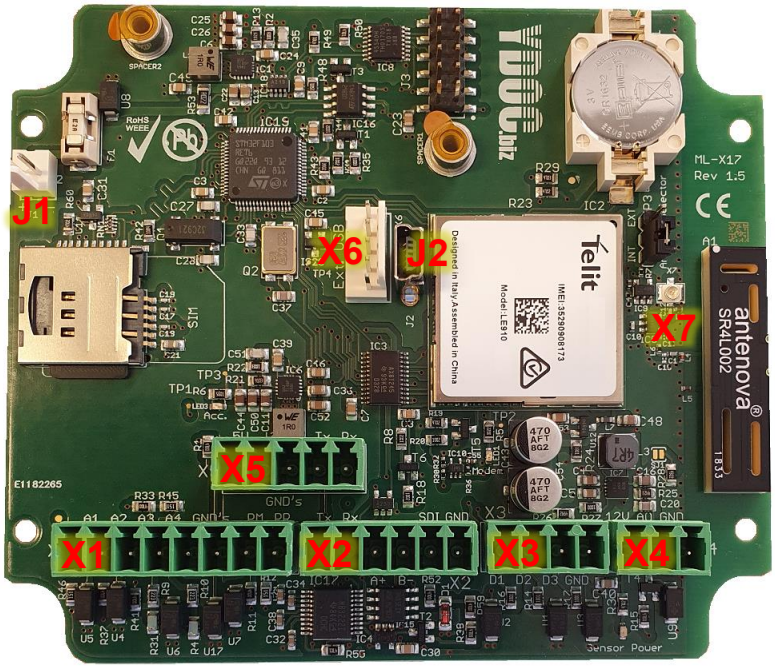
The data logger is equipped with an internal power monitor. During the active mode of the data logger, this power monitor keeps track of the power consumption of the device. When the device is going into sleep-mode, a fixed value is used to calculate the power consumption. Both are calculated and offer a fairly accurate measurement of the power consumption. Unfortunately, the behaviours of batteries are, in practice, much more complicated than the calculation made inside the data logger. So, the capacity, written on the back of the battery is only a typical value. Things like: Shelf life, ambient temperature, current draw, and peak current draw, affect the performance of the battery.

Therefore, we strongly advise to use the measurements regarding battery life as an indication only. We also recommend replacing the battery, fairly above 0%. If you want the best performance and the most optimized settings for your particular measurement location, contact the supplier of the Battery Saft Batteries. They can provide you more specific details and advise on your application. You can provide them information by sending them a bit of previously measured data, for analysis. When you have received the advice, you probably decrease the value of "Battery Capacity" in the configuration setup, to a bit lower value than the default (14 Ah)

When using the cover with integrated NiMH AA solar charger, we recommend to use LSD (long self-discharge) NiMH AA rechargeable batteries with at least a capacity of 2000mAh (e.g. GP Recyko, Sony Eneloop or Vapex Instant).

Please consult the ydoc.biz website for an online power consumption calculator.

# 5 Pin configuration




| Connector | Pin | | Name | Description |
|---|---|---|---|---|
| X1 | 1 | A1 | 0..20mA, input 1 | Positive terminal analog input 1 |
| | 2 | A2 | 0..20mA, input 2 | Positive terminal analog input 2 |
| | 3 | A3 | 0..10V, input 3 | Positive terminal analog input 3 |
| | 4 | A4 | 0..10V, input 4 | Positive terminal analog input 4 |
| | 5,6 | GNDs | GND | Ground |
| | 7 | PM | 0..100% (0..3300mV), input 5 | Positive terminal potentiometer/analog input 5 |
| | 8 | PR | Reference voltage terminal | 3.30V potentiometer reference terminal |
| X2 | 1 | Tx | RS232 TX | Transmit line RS232 |
| | 2 | Rx | RS232 RX | Receive line RS232 |
| | 3 | A+ | RS485 A | +RS485 positive terminal |
| | 4 | B- | RS485 B | -RS485 negative terminal |
| | 5 | SDI | SDI-12 (0..5V) | Terminal to connect SDI-12 sensors |
| | 6 | GND | GND | Ground |
| X3 | 1 | D1 | Digital input 1 (0..5V) | Positive terminal digital input 1 or wake-up line |
| | 2 | D2 | Digital input 2 (0..5V) | Positive terminal digital input 2 |
| | 3 | D3 | Digital input 3 (0..5V) | Positive terminal digital input 3 |
| | 4 | GND | GND | Ground |
| X4 | 1 | 12V | 100 or 200mA (ML-x17) sensor power | Terminal to supply power to sensors |
| | 2 | AO | Alarm output | Open collector (max 100mA sink current) |
| | 3 | GND | GND | Ground |
| X5 | 1 | 5V | 600mA or 800mA (ML-x17) acc. power | To power an accessory (e.g. TFT, CAM, GPS) |
| | 2,3 | GNDs | GND | Ground |
| | 4 | Tx | RS232 TX | Transmit line RS232 to accessory RX |
| | 5 | Rx | RS232 RX | Receive line RS232 from accessory TX |
| X6 | 1 | | GND | USB shield GND |
| | 2 | | GND | USB GND |
| | 3 | | USB_data_P | Positive USB data signal |
| | 4 | | USB_data_N | Negative USB data signal |
| | 5 | | USB_VBUS | USB bus 5 Volts input |
| X7 | | | External antenna connector | U.FL connector for external antenna |
| J1 | 1 | | +VBAT (3.6V DC) | Positive terminal for power source |
| | 2 | | -VBAT (3.6V DC) | Negative terminal for power source |
| J2 | | | Internal USB connector | USB connector for local configuration |
| | | | | |

## 5.1 Pin descriptions

### 5.1.1 Analog Inputs

#### 5.1.1.1 0/4..20mA inputs

Analog Input 1 to 2 (ML-x17AD/ADS/TFT)
These are Current-inputs, with an input impedance of 15 ohms. The range is 4 .. 20 mA. The circuits are equipped with over current-protection. To use this input connect the + of the sensor to the + of the power switch and the – of the sensor to the analog input pin.

#### 5.1.1.2 0…10V inputs

Analog Input 3 to 4 (ML-x17AD/ADS/TFT)
0..10V single ended

#### 5.1.1.3 Potentiometer input

Analog Input 5 (ML-x17AD/ADS/TFT)
Resistance 100K to 4M7 potentiometer type recommended / 0..3300mV single ended

There are some terminals which hold ground level, This provides both sensor-ground and battery ground. For your convenience, these terminals are connected to multiple pads on the connector PCB, because every single sensor will need his own ground. You can connect multiple sensors. When more connections are needed, just connect a wire from there and put the additional connections in parallel.

### 5.1.2 RS485 A & B

These are the pins for RS485 communication. Use these pins together with a ground signal. These signals are ESD-protected by the driver-circuit. The signal levels are according to the *TIA/EIA-485* Standard.

### 5.1.3 Power Switch

This is an output for powering sensors up to 200 mA. The output voltage is 12 Volts DC factory default. You can alter the output voltage to 24V at max 100mA by bridging solder jumper SJ1 on your own risk.

### 5.1.4 VBAT +

This is the main power supply input for the board. The level is 3.6 Volts.
Note: This signal is NOT the same as the internal 3.6 volts level. The power-supply circuit converts this level to the fixed, internal, 3.6 Volts level. This voltage level is allowed to be between 0.8 volts and 5 Volts. We strongly recommend using a 3.6 Volts Power source only. The actual voltage on this pin is monitored by the firmware. It is called “Primary input Voltage”. Also the current, flowing through the 2 wires, is monitored, and is called “Primary input Current”.

### 5.1.5 RS232 RX & TX

These are the pins for RS232 communications. Use these pins together with ground. Al pins are protected against ESD. Voltage levels are according RS232 standard.

### 5.1.6 SDI-12 Hi

This is the in/out terminal for SDI-12 communication. It is protected against overvoltage. Use this terminal together with ground. See www.sdi-12.org for more information.

### 5.1.7 Digital inputs

These are interrupt-driven inputs, with an internal pull-down or pull-up resistor. To use it, connect a switch between the 3V6 or the GND respectively and this terminal. It is suitable for pulse counting up to 10kHz for energy meters, water meters and rain-gauges.

### 5.1.8 +3V6

This is a power output. It is used to power external sensors or a potentiometer. It has a voltage of 3.6 Volts and is capable of driving up to 100 mA.

### 5.1.9 Antenna placement and field strength

An antenna is required for 2G/3G/4G and GSM operation. Normally you will require a dual-band antenna suitable for 900&1800 MHz or 850&1900MHz.

Depending on local field strength the integrated antenna or a simple whip antenna direct connected to the data logger will work, or a better antenna and/or better antenna placement might be required.

You can monitor the actual field strength through the configuration software(menu). The field strength may vary on atmospheric conditions, so we recommend you to make sure that the indication is maximized at installation
The field strength may also vary on the growth of vegetation (trees tend to block the signal). We also recommend configuring the data logger in such a way that the 2G/3G/4G field strength is recorded during data transfer. In this way you can get an early warning when the field strength gets low.

What to do to get a better field strength signal;

- Make sure the antenna is mounted in accordance with the manufacturer's instructions. Note there are antennas (whip antennas) that require a metal surface below the antenna; others (dipole antennas) do not.
- Make sure that all connectors on the antenna and antenna cable are tightened and free of moisture.
- Make sure the antenna is in vertical position; as the GSM and 2G/3G/4G radio signals are vertically polarize, the antenna should be vertical positioned for maximum performance.
- Do not place the antenna near metal surfaces or structures. Be aware that various building structures contain metal (e.g. steel mesh as reinforcement for concrete).
- Place the antenna outdoors.
- Identify the nearest GSM tower of your provider. Place the antenna in a location that provides a free line-of-sight to the tower.
- If you cannot identify the nearest GSM tower of your provider, place the antenna on a higher position; generally, higher is better.
- Use good quality (low-loss) antenna cables. Generally, the thicker the cable, the better.
- Avoid unnecessary adaptors and connectors in the antenna cable, as every "joint" cause a significant signal loss (0.5 to 1 dB).
- Use an antenna with a higher antenna gain. (simple stubby antennas can have a gain of -9db, a rod antenna can have an antenna gain of 0 or 4 dB or higher; Note that the allowed radio power is limited to 1W/2W. An antenna with a higher gain is only allowed when this only compensates for the cable and connector losses).
- Seal your antenna-connector with vulcanizing tape, to prevent from oxidation.
- When the integrated antenna is used, make sure that the data logger is NOT mounted on a metal plate.
- When the data logger is encapsulated in a metal enclosure or cage of faraday, you may NOT use the integrated antenna, but you must use an external antenna to avoid damaging the MODEM.

Make sure the SIM you intend to use is compatible with your network and the pin code protection is disabled.

# 6 Maintenance and Repair

## 6.1 RTC Lithium Battery replacement

The battery of the data logger is designed to last for the lifetime of the instrument. It should not be necessary to replace this battery.

## 6.2 Recalibration

Calibration of the data logger has been performed while manufacturing. YDOC guarantees the calibration to last for 2 years. However in most cases the calibration will last for the lifetime of the instrument.
Calibration is important for high accuracy measurements and in situations where time stamping is very important. The logger has a NTP-time-synchronize option, which is selectable by the user. The parts of the data logger that could need re-calibration are:

- Analog inputs
- Real time clock

For most applications, the analog inputs are sufficiently accurate, and need no re-calibration for the lifetime of the instrument. But in special cases, where the user demands a high precision measurement, the analog interface may be re-calibrated after that period. High temperature deviations and harsh environment are factors that needed to be considered. Please contact your local supplier for more information on recalibration needs and –support. The real time clock is also calibrated during the manufacturing process, and has very good long life stability (see spec. sheet). Also, when operating in a harsh environment, the need for a recalibration can be applicable. YDOC can perform overall calibrations any time you like.

## 6.3 XRAY

In the uncommon event of exposure to XRAY, extra precautions are needed. When the device is shipped many times, and is scanned for a security check, the analog input calibration will be harmed. Although the level of radiation is very low, the data logger can be harmed if the number of times that it is exposed to radiation exceeds 10. What will happen is that the analog interface will drift outside its spec's. As a precaution the user can shield his device, with a metal can, to prevent from damage. Normally, the impacts of these security-scans are very low and cause no problems.

# 7 Safety

Don't work on the wiring of the data logger when powered from an external supply.

## 7.1 Power supply

The data logger is protected against reversed polarity of the battery power. The mains power supply is protected by a 4AT fuse type TR5.

## 7.2 ESD

The data logger is equipped with an ESD (Electronic Static Discharge) protection on all "outside world" leads. i.e. comports and analog inputs etc. Though it is designed to withstand a certain amount of electrical discharge (human body model) it is strongly advised to take precautions while operating or servicing the data logger.

# 8 Environment and disposal

The data logger is manufactured in compliance with the RoHS directive (Reduction of Hazardous Substances) EU directive 2002/95/EC, which means in popular terms that the product is "lead-free".

When the data logger is taken out of service, dispose the data logger in accordance to the local regulations at the time the product is disposed.

Regulations for disposal of batteries may be different. Remove the batteries and dispose them in accordance to the local regulations for batteries.

# 9 Transport and Storage

The following requirements are applicable for transport and storage of the data logger.

Storage:

| | |
|---|---|
| Humidity | < 95% (Non condensing) |
| Temp | 10 .. 30 °C |

Transport:

| | |
|---|---|
| Humidity | < 95% (Non condensing) |
| Temp | 10 .. 30 °C |

If the data logger is delivered in its standard protecting enclosure, it is strongly recommended to use this case for all transportation, until the final location of operation. This enclosure is especially designed to protect the data logger from being damaged.

# 10 Appendix

## 10.1 Specifications

| | | | | | |
|---|---|---|---|---|---|
| **Power Supply** | | | | | |
| Protection | Overvoltage and reverse polarity protection ( by transorp and Fuse) | | | | |
| Input Range | 3.3 ~ 5 Vdc | | | | |
| Type of Power | Battery | | | | |
| Power Consumption * | Sleep mode | | Operating mode | | Send mode |
| | 360 uW | 100 uA @ 3.6 Volts | 180 mW | 50 mA @3.6 Volts | ~1 Watt | ~ 300 mA @ 3.6 Volts |
| **General Enviroment** | | | | | |
| Temperature | Operating: -30 ~ + 75 °C; Storage -40 ~ +85 °C | | | | |
| Humidity | 5 ~ 100 % RH | | | | |
| IP Protection | IP 54, IP67, IP68, depending on type of enclosure | | | | |
| **Operation** | | | | | |
| Batterylife | up to 4 years; consult user manual for more information | | | | |
| Configuration Programming | Via USB port or remote ; no special software needed; uses Ydocterminal or other terminal program | | | | |
| Data Retrieval | Also configuration via Android is offible: via USB or via optional BLE module<br>Manually exchange micro SD-card; Automatic via Web ( HTTP(S), FTP(S), e-Mail , TCP or MQTT) Via USB-Connection by means of a Computer (Windows), or Android | | | | |
| Alarming | On pre-defined thresholds of measurements; | | | | |
| Sensor Excitation | Internal boost convertor for supplying remote sensors ; 200 mA @ 12 Volts | | | | |
| **System** | | | | | |
| CPU | ARM Cortex M3 | | | | |
| Clock Frequency | 72 Mhz | | | | |
| Watchdog | Yes | | | | |
| RTC(Real Time Clock) | Yes, internally calibrated; accuracy < 100 ppm; Battery Backuped | | | | |
| FLASH Memory | 512 KB | | | | |
| SRAM | 64 KB | | | | |
| NVRAM | 84 bytes , battery backup, data valid up to 20 years | | | | |
| Analog inputs | 12 bits | | | | |
| Temperature sensor | Yes | | | | |
| Power Sensor | Yes, Monitors power consumption, rest-capacity of battery | | | | |
| Option Port | One, for optional modules. ( Extra I/O, special boards see website) | | | | |
| USB port | USB 2.0 full speed interface | | | | |
| Sample Frequency | max 4 Hz | | | | |
| Datalog Frequency | max 4 Hz | | | | |
| Rohs Complient | Yes | | | | |
| **Analog Inputs** | | | | | |
| Number of Channels | 4 | | | | |
| Resolution | 12 bits | | | | |
| Input type | 0 ~ 20 mA (Channel 1 & 2); 0 ~ 10 V (Channel 3 & 4) | | | | |
| | | | | | |
| **Memory Card** | | | | | |
| Type | micro-SD | | | | |
| Capacity | 8 GB | | | | |
| Filesystem | FAT 32 | | | | |
| **Communication Ports** | | | | | |
| SER1 | RS232; TxD, RxD; Non-isolated;Enhanced ESD Specification: ±15kV Human Body Model; | | | | |
| | Speed: 115200 bps max. | | | | |
| | RS485 | | | | |
| | SDI12 | | | | |
| SER2 | RS232; TxD, RxD; Non-isolated;Enhanced ESD Specification: ±15kV Human Body Model; | | | | |
| | Speed: 115200 bps max. | | | | |
| **Counter input** | | | | | |
| Type | One Digital input 0 ~ 3.6 Volts; Internal pull-up/down; 10 kHz max. storage of value in Non Volatile Ram, even after battery replacement | | | | |
| **Cellular Modem** | | | | | |
| Frequency Range | 2G / 3G / 4G | | | | |
| Capabilities | HTTP(S), FTP(S), e-Mail , TCP or MQTT | | | | |
| | | | | | |
| **Dimensions** | | | | | |
| D x H | 120 X 130 x 90 mm | | | | |
| **Weight** | | | | | |
| Netto Weight | 320 Grams (excluding Batteries) | | | | |

* The Power consumption in sleep mode is when Datalogger is idle, and no tasks performing. Only the RTC is running

## 10.2 CE Declarations of Conformity

CE

**EU Declaration of Conformity**

(in accordance with ISO/IEC 17050)

Swammerdamlaan 14
6721 BK Bennekom
www.ydoc.biz
info@ydoc.biz

We,

***Your Data Our Care B.V.***

declare under our sole responsibility that the product:

***Type ML-x17yyy-zz Low Power Cellular Datalogger***

***SKU explanation:***

***X= Cellular technology***
*0= no cellular modem; 2=2G;3=3G;4=4G*
***Y=Port Configuration***
*A= Analog Port; D=Digital Port; S=Serial Port*
***Z=Power Option***
*Li=Lithium Non Rechargeable; PV= Photovoltaic; 3Li=triple Lithium Non Rechargeable; LFP=LiFEPO4; NM= Nickel Metal Hydride; SLA=Sealed Lead-acid.*

*Example: ML-317AD-PV : 3G technology; with analog and digital ports; without serial ports; with Solar Panel integrated enclosure*

This Declaration of Conformity is in accordance with the European Standard EN ISO/IEC 17050:2010 Conformity Assessment *"Supplier's conformation of conformity"*

The basis for the criteria has been found in International documentation, particularly in ISO/IEC.

This declaration is an EC Type Declaration of Conformity as referenced in EC directive 2014/30/EU *The EMC-directive* and as in Appendix III.B

**The object of the declaration described above is in conformity with the relevant EU harmonization legislation: at date of this declaration:**

**2014/30/EU** The Electromagnetic Compatibility Directive (EMCD)
**2014/53/EU The Radio Equipment Directive (RED) (formerly R&TTE 99/5/EC)**
**2014/35/EU The Low Voltage Directive (LVD)**
**2011/65/EU** The Restriction of Hazardous Substances Directive (RoHs)

**Standards used during compliance assessment:**

EN 61326-1:2013 Electrical equipment for measurement, control and laboratory use- EMC requirements - for use in industrial locations
EN 61000-6-1:2001
EN61000-6-3:2001

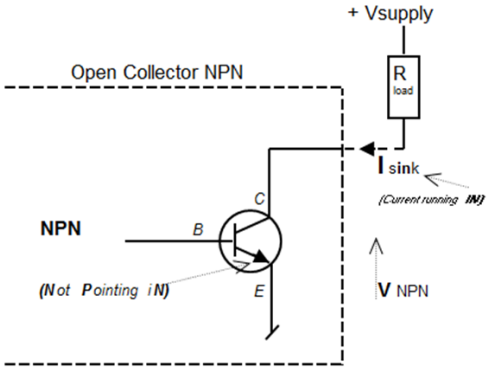
The product uses a Telit GPRS/UMTS/2G/3G/4G Module type xE-910-XXX,which is tested by the manufacturer and also notified bodies, registered under (nobo) number: 1909

| |
|---|
| *R. Kleine* |
| *Technical Director* |
| *Signature:* |

*Zeewolde,12-11-2017*

## 10.3 Open Collector output

In electronics a common way of driving external loads is the use of an “open collector” output. This is a transistor, with its collector left unconnected, for the benefit of user to connect to its load of interest. The open collector (o.c.) is very versatile, and provides the possibility to use various voltages.




If the NPN transistor is **OFF,** $I_{sink}$ equals 0 mA.
This means that $V_{NPN}$ equals approx. + $V_{Supply}$.

If the NPN transistor is **ON,** $I_{sink}$ equals approx. (+$V_{supply}$ / $R_{load}$ ).
This means that $V_{NPN}$ equals 0 V.

In the picture above, an open collector output is shown. The o.c. is only capable of sinking current and NOT sourcing current. That’s why the “load” in the picture above is located in the “high side” . The transistor will sink the bottom side of the “load”, enabling current to flow through.

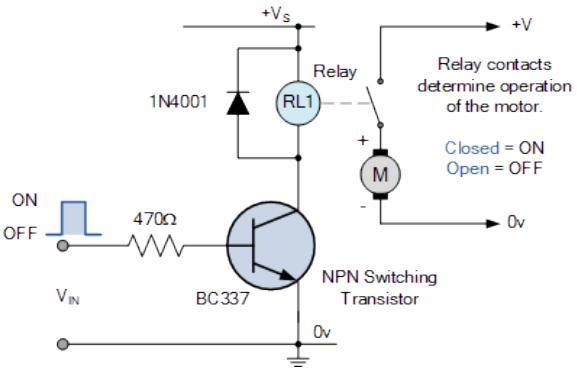
The user can choose his own voltage level, because the transistor only acts as a switch and NOT supplying any voltage. In your YDOC data logger the maximum “supply voltage” for the o.c. switch is limited to 40 volts DC.

### *10.3.1.1.1 Using a relay to drive an external load*

If you like a relay, in order to switch an external “load”, please bear in mind that a relay can produce very high these spikes to occur, always use a diode, connected in parallel with the “load” .

Picture below shows how to do this.

The transistor and resistor shown, are part of the data logger PCB.



## 10.4 Trouble shooting

If you encounter problems with the data logger, you can start checking the following.

- First try to set up a connection, via USB, and use the program, YDOC-terminal, to communicate with it.
- If that doesn't work, you have to check the battery-power, so you have to open the case. Do this in a dry and clean environment, NOT in the field. Normally a flat battery is preceded by an alarm-message.
- Check the fuse

Most parts of the data logger are tested at start-up. To monitor the messages which are issued at start-up, you have to connect a PC/terminal to the debug port.

*Attention:*

*The default DEBUG port is serial port 1, but when a sensor is connected to this port, the DEBUG port becomes port 2. If both ports are in use, you can use the USB port as debug port, but you won't be able to see start-up-messages. In this case it is advised to remove a sensor from the configuration, temporarily.*

```
YDOC Logger Version 1.6 Build 1
<12:20:05>
2012/10/07 12:20:05
Init User Interface
Init Modem Interface
Init SD-card
File system OK
Init System Monitor
Start up from Power on
POWER_ON;System log...done
SYS_START;System log...done
Init NTP Time update task
Init Sensors Internal
Init Sensor GSM signal
Init Email
Init TCP
Running
```

These are typical start up messages:

**Explanation:**
The data logger starts and initializes its peripherals. First, an overview of the firmware version is given with a timestamp. This timestamp should be accurate, carefully check the timestamp. If it is slightly wrong, it must be adjusted. If it is totally out of date, it designates a RTC-problem. There should be NO errors on this start-up. If there are errors contact your local YDOC-dealer. A screen dump of the start-up messages will help to solve the problem.

```
YDOC Logger Version 1.6 Build 1
<12:29:21>
2012/10/07 12:29:21
Init User Interface
Init Modem Interface
Init SD-card
File system error
Init System Monitor
Start up from Power on
POWER_ON;System log...File system
error
SYS_START;System log...File system
error
Init NTP Time update task
Init Sensors Internal
Init Sensor GSM signal
Init Email
Init TCP
Running
```

An example of a defective data logger is given below. This data logger has its micro SD-card not installed. You can clearly notify the problem, by looking at the start-up messages.

When there is no debug output visible at all, contact your local YDOC-supplier.

Attention:

Always connect the USB-cable to the PC, even when you are connected to the serial port for debug output. When the data logger is NOT connected by USB, it will switch into low power mode (Auto Sleep), and you won't get any debug data.

## 10.5 System messages

Meaning of the System messages (S-records in TXT and CSV or $Msg-lines in JSON data log files)

A System message contains a message code, an optional supplemental code/text and an optional explanatory text.

| CODE | CODE or text | Meaning | Possible reasons |
|---|---|---|---|
| **SYS_START** | "System name & S/N" | System is started and deployed | At power on, or reset |
| **SYS_DEPLOYED** | "System name & S/N" | System is deployed | After a resume command or a delayed deployment |
| **SYS_SUSPENDED** | "System name & S/N" | System is suspended until deployment is started | When not yet deployment start time is reached |
| **SYS_UPGRADED** | "System name & S/N" | Firmware upgrade detected | After firmware upgrade |
| **CFG_CHANGED** | "System name & S/N" | Configuration has changed | After configuration editing |
| **CFG_ERR** | | Configuration error | Firmware downgrading, or configuration upload with different version |
| **NO_CFG** | | No configuration detected | Data logger is not yet configured |
| **BLE_INIT** | "BLE module init failure" | BLE option board was detected, but BLE module does not respond as expected | Lack of power or module defect |
| **CFG_UPL** | "cause of failure" | Configuration upload failed | Disruption in the data transfer, or SD card failure |
| **SNP_UPL** | "cause of failure" | Configuration snippet upload failed | Wrong settings or unknown drivers or parameters |
| **FW_UPG** | "cause of failure" | Firmware upgrade failed | Disruption in the data transfer, or SD card failure |
| **POWER_ON** | “logger type and version” | System is powered on | The external power was cut off and put on again |
| **CFG_RESET** | “logger type and version” | Configuration has changed | After a configuration upload or configuration menu change |
| **UPG_RESET** | “logger type and version” | Firmware is upgraded | After a successful firmware upgrade |
| **SW_RESET** | “logger type and version” | Software reset by the system | After a flash failure or CRC error (which usually results in a non-functioning logger) |
| **USR_RESET** | “logger type and version” | The user has issued the reboot command in the maintenance menu | Up to the user |
| **HW_RESET** | “logger type and version” | Unscheduled reboot has occurred | Unknown to us, else it would not occur |
| **PVD_RESET** | “logger type and version” | After a internal power voltage detection lower than 2.6V which takes longer then 8 seconds | Batteries are depleted, or the external power adapter can’t deliver enough power |

| | | | |
|---|---|---|---|
| **WDT_RESET** | “logger type and version” | Firmware reset after a possible software deadlock | Unknown |
| **BL_RESET** | “logger type and version” | The user has issued the bootloader command in the maintenance menu | E.g. to erase the configuration or to perform a firmware upgrade via the bootloader |
| **HARD_EXCP** | “logger type and version” | Auto system reset | Unknown hardware failure |
| **MEM_EXCP** | “logger type and version” | Auto system reset | Unknown software failure |
| **BUS_EXCP** | “logger type and version” | Auto system reset | Unknown hardware failure |
| **USAGE_EXCP** | “logger type and version” | Auto system reset | Unknown software failure |
| **PVD_DIP** | "Internal power too low, X dip's at 2.6 V" | Power voltage detection dips | When the internal power (temporarily) drops unexpectedly below 2.6V (power has returned before switch off occurred) |
| **SD_FAIL** | "number" | Number of temporarily SD card failures | SD card failure |
| **SYS_PAUZED** | "System name & S/N" | Operation is suspended until resume command is given | When the user gives a suspend command |
| **SYS_RESUMED** | "System name & S/N" | Operation is resumed again | When the user gives a resume command |
| **CFG_MENU** | "Operation suspended" | Edit mode | When entering the configuration menu |
| **CFG_MENU** | "Operation resumed" | Run mode | When leaving the configuration menu |
| **LP_SLEEP** | "Start voltage too low" | LFP battery power has dropped to 2.8 Volt | System goes to sleep until batteries are recharged again |
| **WDT** | "Sensor name" | Shows the sensor driver name were the watchdog timeout occurred | Sensor defect, or wrongly connected, or warmup time too short |
| **WDT** | WIFI_INIT | Watchdog timeout while initializing Wi-Fi module | No response from Wi-Fi Module |
| **WDT** | APN_SET | Watchdog timeout while setting Wi-Fi APN | No response from Wi-Fi Module |
| **WDT** | IMEI_CHECK | Watchdog timeout while getting IMEI number | Modem not responding due to lack of power or Modem has died because of flash corruption due to power failures |
| **WDT** | SIM_DETECT | Watchdog timeout while getting SIM status | Modem not responding due to lack of power or Modem has died because of flash corruption due to power failures |
| **WDT** | MODEM_INIT | Watchdog timeout while initializing Modem | Modem not responding |
| **WDT** | NETWORK_REG | Watchdog timeout while trying to register on a network | No response, or (temporarily) no network coverage, or SIM subscription failure (inactive/invalid/blocked or black listed) |
| **WDT** | SECURITY_SET | Watchdog timeout during security setting | No response |
| **WDT** | APN_CHECK | Watchdog timeout while checking the internet access point | No response, or APN access point name wrong |
| **WDT** | APN_LOGIN | Watchdog timeout while trying to login to the internet access point | No response, or APN user or password wrong, or authentication failure |
| **WDT** | SERVER_LOGIN | Watchdog timeout while trying to login to the Email/FTP/TCP/HTTP/MQTT server | No response, or server not available, or credentials wrong |

| | | | |
|---|---|---|---|
| **WDT** | FILE_OPEN | Watchdog timeout while trying to open a file on the FTP server | No response, or unexpected network disruption, or server error |
| **WDT** | SERVER_CLOSE | Watchdog timeout while closing the sever | No response, or unexpected network disruption, or server error |
| **WDT** | CHANGE_DIR | Watchdog timeout while changing directory on the FTP server | No response, or unexpected network disruption, or server error |
| **WDT** | FILE_OPEN | Watchdog timeout while opening a file on the FTP server | No response, or unexpected network disruption, or server error |
| **WDT** | FILE_GET | Watchdog timeout while reading a file on the FTP server | No response, or unexpected network disruption, or server error |
| **WDT** | FILE_SIZE | Watchdog timeout while getting the file size on the FTP server | No response, or unexpected network disruption, or server error |
| **WDT** | DATA_SENDING | Watchdog timeout while sending data to the server | No response, or unexpected network disruption, or server error |
| **WDT** | TCP_RESPONCE<br>HTTP_RESPONCE | Watchdog timeout while waiting for acknowledge from the TCP/HTTP server | No response, or unexpected network disruption, or server error |
| **WDT** | DATA_END | Watchdog timeout while trying to disconnect from the server connection | No response, or unexpected network disruption, or server error |
| **WDT** | TCP_TERMINAL | Watchdog timeout during remote TCP session | Unexpected network disruption, or server error |
| **WDT** | SMS_SEND | Watchdog timeout during SMS sending | No response, or unexpected network disruption, or server error |
| **WDT** | "STATE number" | State machine step where the watchdog timeout occurred | No response |
| **ERR** | "Driver name" | Shows the driver name were an error occurred, with if applicable some additional information | Sensor defect, or unknown failure |
| **ERR** | WIFI_INIT | Error while initializing Wi-Fi module | Error in Wi-Fi settings |
| **ERR** | APN_SET | Error while setting Wi-Fi APN | Error in APN settings |
| **ERR** | IMEI_CHECK | Error while getting IMEI number | Wrong IMEI number reported |
| **ERR** | SIM_DETECT | Error while getting SIM status | No SIM, or SIM not detected, or PIN on SIM or Modem not responding due to power failure due to lack of power |
| **ERR** | MODEM_INIT | Error while initializing Modem | Modem not responding |
| **ERR** | NETWORK_REG | Error while trying to register on a network | (Temporarily) no network coverage, or SIM subscription failure (inactive/invalid/blocked or black listed) |
| **ERR** | SECURITY_SET | Error during security setting | Error in security settings |
| **ERR** | APN_CHECK | Error while checking the internet access point | Error in APN and authentication settings |

| | | | |
|---|---|---|---|
| **ERR** | APN_LOGIN | Error while trying to login to the internet access point | APN access point, user name or password wrong, or authentication failure |
| **ERR** | SERVER_LOGIN | Error while trying to login to the Email/FTP/TCP/HTTP/MQTT server | Server not available or credentials wrong |
| **ERR** | SERVER_CLOSE | Error while closing the sever | Unexpected network disruption, or server error |
| **ERR** | CHANGE_DIR | Error while changing directory on the FTP server | Wrong or none existing directory on FTP server |
| **ERR** | FILE_OPEN | Error while trying to open a file on the FTP server | File handling error on the server (read/write rights) |
| **ERR** | FILE_GET | Error while reading a file on the FTP server | File handling error on the server (read/write rights) |
| **ERR** | FILE_SIZE | Error while getting the file size on the FTP server | File is removed to quickly on FTP server |
| **ERR** | DATA_SENDING | Error while sending data to the server | Unexpected network disruption, or server error |
| **ERR** | TCP_RESPONCE HTTP_RESPONCE | Error while waiting for acknowledge from the TCP/HTTP server | Unexpected network disruption, or server error |
| **ERR** | DATA_END | Error trying to disconnect from server | Unexpected network disruption, or server error |
| **ERR** | TCP_TERMINAL | Error during remote TCP session | Unexpected network disruption, or server error |
| **ERR** | SMS_SEND | Error during SMS sending | Unexpected network disruption, or server error |
| **ERR** | TLR | Error in Terminal Link Request | When security verification failed or when scheduled time is in the past |
| **ERR** | PAR | Error in processing FTP input parameter | Wrong or none existing parameter |
| **ERR** | "STATE number" | Modem state machine step were the error occurred | Unknown |
| **REG_SET** | "digital sensor name" | Shows the setting of the counter register | When the register is set during a configuration change |
| **REG_RESET** | "digital sensor name" | Shows the reset of the counter register | When daily reset is on, or after manual change |
| **STR** | "Driver name; raw data" | Shows the raw data log | When this option is enabled (generic serial string and GPS) |
| **TIME_FIX** | "+/- number of sec" | Shows the number seconds the internal clock was adjusted | After a NTP time synchronization |
| **TIME_FIX** | "Summer time started" | Shows that the clock is adjusted to the start of summer time | When summer time (daylight saving) option is enabled |
| **TIME_FIX** | "Summer time ended" | Shows that the clock is adjusted to the end of summer time | When summer time (daylight saving) option is enabled |
| **ALARM_SET** | "Code; limit; value" | Shows the parameter code, alarm limit and actual value | When the alarm is raised |
| **ALARM_RESET** | "Code; limit; value" | Shows the parameter code, alarm limit and actual value | When the alarm is cleared |
| **ALARM_SEND_OK** | | The Alarm message has been send | When the message has send |
| **ALARM_SEND_ERR** | | The Alarm message has not been send | There will also be a Modem or Wi-Fi failure shown |

| | | | |
|---|---|---|---|
| **INV** | "Driver name" | "Has invalid data" | For aggregations it means there are no samples to aggregate. For sensors the data is invalid |
| **INV** | "Driver name" | "Maximum value reached" | If pulse rate is higher then 1 per second, the Max rate parameter value is invalid |
| **STATUS** | "Power too low" | When starting the logger and detecting (depending on power board and battery type setting), that the power is not sufficient to properly operate the logger | Batteries are flat, or the external power adapter can't deliver enough power |
| **STATUS** | "Power too low; Modem temporarily disabled" | Modem is disabled due to lack of sufficient power | Batteries are flat, or the external power adapter can't deliver enough power |
| **STATUS** | "Analog burst start" or "Analog burst end" | To indicate the start and end of the Analog burst log | |
| **DEBUG** | "Deregistering from the not responding network" | In order to try to reregister on to the Network | When it takes to long to register on to the Network (3 minutes) |
| **DEBUG** | "BLE interface terminated due to inactivity timeout" | The BLE connection is terminated in order to save power | When there is 3 minutes no user interaction on the BLE connection |
| **DEBUG** | "option board not detected" | Specified option board is not detected | Option board is not present, removed or broken |