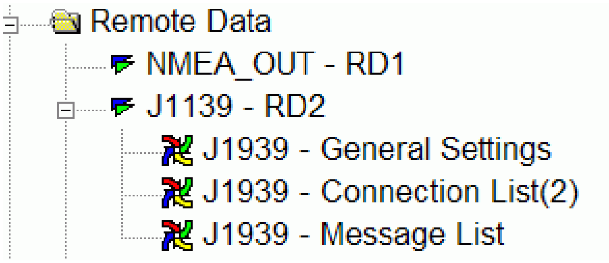
Remote Data

The following Remote Datas are available:

- J1939
- ATA
- CANOpen
- Caterpillar CCM
- EAC-300
- IHCS
- Line In
- Modbus Master
- Modbus Slave
- Modbus Master TCP/IP
- Modbus Slave TCP/IP
- MG
- MRU
- MTU Slave
- NMEA2K
- NMEA Listener
- NMEA Talker
- OPC Client
- SF_Control
- Simulation
- System IO
- Tank Calculation
- TCP/IP
- TTP
- TXT

J1939 – Plugin

After selecting 'Remote Data' and J1939 – RD xx



Remote Data
NMEA_OUT - RD1
J1139 - RD2
J1939 - General Settings
J1939 - Connection List(2)
J1939 - Message List

There are several items of this remote data:

1. J1939 - General Settings
2. J1939 - Connection List
3. J1939 - Message List
4. J193 - Diagnostic List

Introduction

J1939 is a set of standards defined by SAE. They are used in heavy-duty vehicles (trucks and buses, mobile hydraulics, etc.)

In many ways, J1939 is similar to the older J1708 and J1587 standards, but J1939 is built on CAN.

The physical layer (J1939/11) describes the electrical interface to the bus. The data link layer (J1939/21) describes the rules for constructing a message, accessing the bus, and detecting transmission errors. The application layer (J1939/71 and J1939/73) defines the specific data contained within each message sent across the network.

Quick facts

- Higher-layer protocol built on CAN
- Used in heavy-duty vehicles
- The speed is nearly always 250 kbit/s

Message Format and Usage (J1939/21)

Most messages defined by the J1939 standard are intended to be broadcast. This means that the data is transmitted on the network without a specific destination. This permits any device to use the data without requiring additional request messages. This also allows future software revisions to easily accommodate new devices (address assignments). When a message must be directed to a particular device, a specific destination address can be included within the message identifier. For example, a request for a specific torque value from the engine instead of a specific torque value from the brake controller.

J1939 uses the 29-bit identifier defined within the CAN 2.0B protocol shown in Figure 1. The identifier is used slightly different in a message with a destination address ("PDU 1") compared to a message intended for broadcast ("PDU 2").

PDU stands for Protocol Data Unit (i.e. Message Format).

The SOF, SRR, and IDE bits are defined by the CAN standard and will be ignored here. The RTR bit (remote request bit) is always set to zero in J1939.

The 29-bit identifier used in J1939 is structured in the following way.

Priority	Reserved	Data page	PDU format	PDU specific	Source Address
3 bits	1 bit	1 bit	8 bits	8 bits	8 bits

Table 1: Structure of a 29-bit identifier.

The first three bits of the identifier are used for controlling a message’s priority during the arbitration process. A value of 0 has the highest priority. Higher priority values are typically given to high-speed control messages, for example, the torque control message from the transmission to the engine. Messages containing data that is not time critical, like the vehicle road speed, are given lower priority values.

The next bit of the identifier is reserved for future use and should be set to 0 for transmitted messages.

The next bit in the identifier is the data page selector. This bit expands the number of possible Parameter Groups that can be represented by the identifier.

The PDU format (PF) determines whether the message can be transmitted with a destination address or if the message is always transmitted as a broadcast message.

The interpretation of the PDU specific (PS) field changes based on the PF value:

- If the PF is between 0 and 239, the message is addressable (PDU1) and the PS field contains the destination address.
- If the PF is between 240 and 255, the message can only be broadcast (PDU2) and the PS field contains a Group Extension.

The Group extension expands the number of possible broadcast Parameter Groups that can be represented by the identifier.

The term Parameter Group Number (PGN) is used to refer to the value of the Reserve bit, DP, PF, and PS fields combined into a single 18 bit value.

Example: The ID 0xCF004EE can be divided into the following fields in table 2.

0x0C				**0xF0**	**0x04**	**0xEE**
000	011	0	0	11110000	00000100	11101110
---	Prio	R	DP	PF	PS	SA

Table 2. PGN example.

PGN = the R, DP, PF and PS fields - in this case 0x0F004.
PF = 0xF0 = 240, i.e. this is a PDU2 (broadcast) message
PS = 0x04, i.e. the Group Extension = 4
The last 8 bits of the identifier contain the address of the device transmitting the message. The address is the label or "handle" which is assigned to provide a way to uniquely access a given device on the network. For a given network, every address must be unique (254 available). This means that two different devices (ECUs) cannot use the same address.

Addresses and Names (J1939/81)

The Name is a 64 bit (8 bytes) long label which gives every ECU a unique identity.
The Name is composed of 10 fields and has the following structure shown in table 3.

Table 3. Structure of the Name.
1. Arbitrary address bit
2. Industry group, length 3 bits
3. Vehicle system instance, length 4 bits
4. Vehicle system, length 7 bits
5. Reserved bit
6. Function, length 8 bits
7. Function instance, length 5 bits
8. ECU instance, length 3 bits
9. Manufacturer code, length 11 bits
10.Identity number, length 21 bits

Byte number in CAN message	Contents/Meaning	
0	Identity number, LSB	
1	Identity number	
2	Bits 0-4: Identity number, MSB Bits 5-7: Manufacturer code, LSB	
3	Manufacturer code, MSB	
4	Bits 0-2: ECU instance Bits 3-7: Function instance	
5	Function	
6	Bit 0: Reserved bit Bits 1-7: Vehicle system	
7	Bits 0-3: Vehicle system instance Bits 4-6: Industry group Bit 7: Arbitrary address bit	

The main purpose of the Name is to describe an ECU. The lower Function field values, 0 to 127, are pre-assigned to "standard" functions or devices. The values 128 to 254 are dependent on the Industry Group and the Vehicle System values. This dependence makes it possible to have the same arrangement of functions in different vehicles. This system also allows devices such as trailers and agricultural equipment to limit their search for an available address and thus minimize the time and difficulty of dynamically claiming an address. When claiming an address, the Name is used to determine which ECU has higher priority and therefore will get the address that was claimed.

Each device on the network will be associated with at least one Name and one address. However, multiple device Names and multiple addresses may coexist within a single ECU. For example, an engine and engine brake (retarder) residing in a common device with a single physical bus connection. The device address defines a specific communications source or destination for messages. The Name identifies the functionality and adds a unique instance number of that functionality when multiple devices of the same type coexist on the network. Only 254 different devices of the same type can coexist on the network due the address limit. Address 255 is reserved as a global address for broadcast and address 254 is reserved as the "null address" used by devices that have not yet claimed an address or failed to claim an address.

Address Claim

In general, most addresses are pre-assigned and used immediately upon power up. In order to permit J1939 to accommodate future devices and functions which have not yet been defined, a procedure has been specified for dynamically assigning addresses. Each device must announce which address it is associated with. This is the identification (address claim) feature. Two options are available:

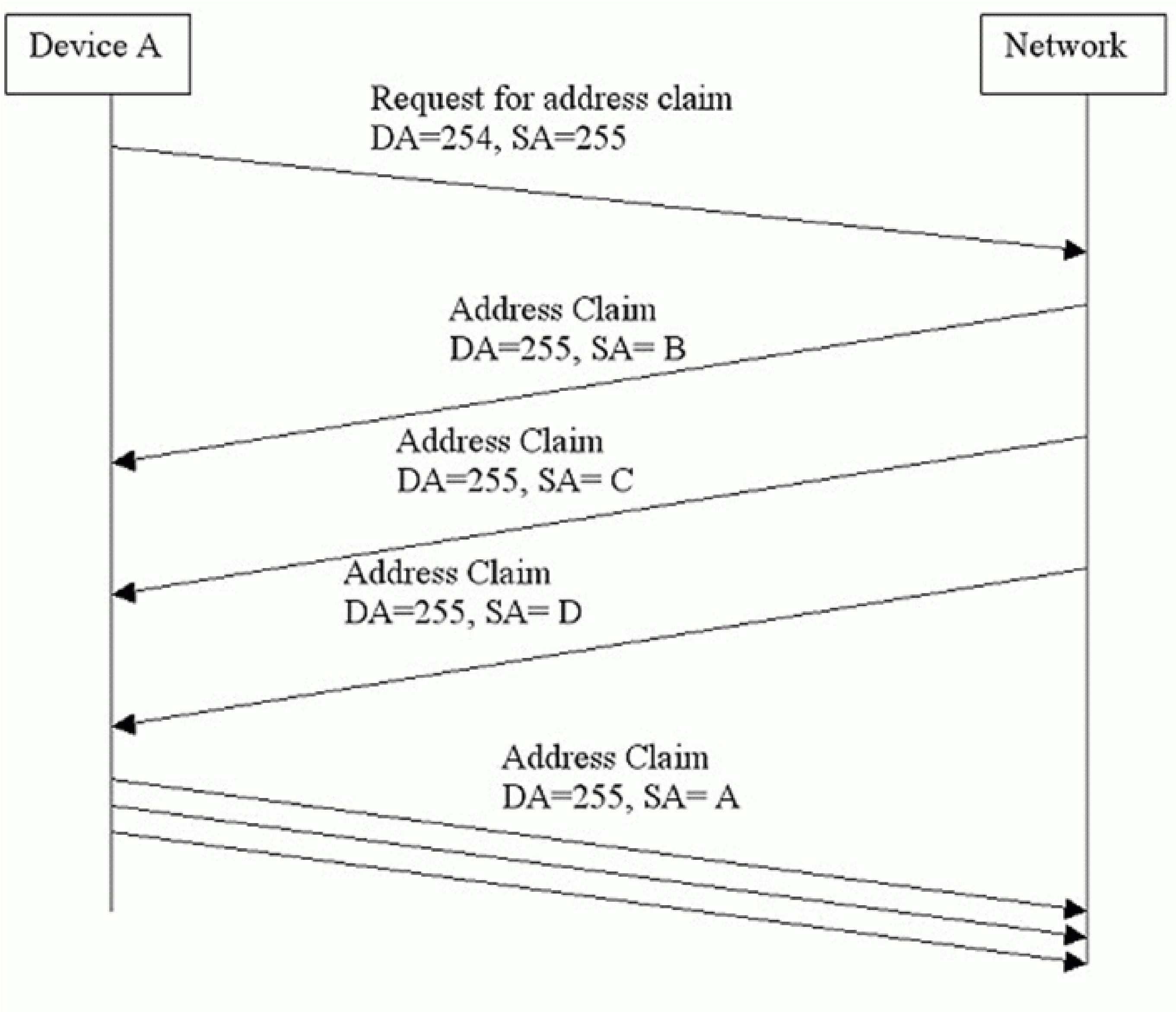
1. Send an Address Claim message to claim an address.

When a device sends an Address Claim message to claim an address, all devices compare this newly claimed address to their own table of devices on the network. If the address is already claimed by a device with a higher priority, that device transmits an Address Claim message indicating that the address is already in use. The Name, which is sent as data in the Address Claim message, determines which device has higher priority.

2. Send a Request for Address Claim.

When a device sends a Request for Address Claim, all devices respond by transmitting their Address Claimed messages. This permits transitional devices (tools, trailers, etc.) or devices powering up late to obtain the current address table so that an available address can be selected and claimed. See figure 2.

Dynamic address assignment support is optional and only those devices which might be expected to encounter address conflicts must support this capability. To eliminate the need to support dynamic address assignment and speed up this “identity process”, most ECUs are associated with a preferred address. These preferred addresses are described in the document J1939/71. If the preferred address is already in use by another ECU, the device can attempt to claim another address if self-configuration is supported by the device.



Address Claim with no complications.

Transmitting Messages (J1939/21 and J1939/7x)

To send a particular data item, a message must be constructed with overhead that describes the data to be sent. Related data items are typically packed together within a message to reduce overhead. J1939/71 defines some standard PGNs which describe the parameters to be sent in a message. The J1939/71 document also includes information about message priority and transmission rate. Note that when a device does not have data available for a given parameter, the byte that should have contained this parameter is set to "not available" (0xFF) so that a receiver knows that the data is missing. Messages which need more than eight bytes of data can be sent as multi-packet messages. Multi-packet messages are transmitted by means of the Transport Protocol Functions defined in J1939/21. However there are two ways of transmitting multi-packet messages:

1.Broadcast Announce Message (TP_BAM) 2.Connection Management (TP_CM)

TP_BAM messages

TP_BAM messages use a global destination address which means that all devices on the network will receive these messages. The transmission is started with a Connection Management (CM) message, PGN = 0x00EC00, with a Control byte indicating TP_BAM. The message data follows in Data Transfer (DT) messages, PGN = 0x00EB00.

TP_CM messages

TP_CM messages are sent point to point between two devices. The transmission starts with a CM message with a Control byte indicating Request To Send (RTS). The receiving device responds with a CM message with the Control byte indicating Clear To Send (CTS). The transmitting device then sends the portion of the data indicated in the CTS using DT messages. This handshake of CTS then DT messages continues until the entire message is transmitted. The connection is terminated at the completion of the message by the receiver transmitting a CM message with a Control byte indicating End Of Message Acknowledgement (EOM). Note that for this process to work, the CM message contains additional data based on what the control byte is. The RTS includes: number of bytes, number of packets and the PGN whose data will be transported. The CTS includes the number of packets the receiver expects next and the packet number to start with.

Receiving Messages (J1939/21 and J1939/7x)

There are various techniques (and chips) available for capturing selected messages off the network. Several general observations can be made, however regarding received messages:

1) If a message is a destination specific request or command, the device must determine if the destination address matches an address claimed by the device. If there is a match, the receiving device must process the message and provide some type of acknowledgment.

2) If a message is a global request, every device, even the originator, must process the request and respond if the data is available.

3) If a message is broadcast, each device must determine if the content is of relevance or not.

ECU Design (J1939/1x, J1939/21, and J1939/7x)

Although every manufacturer will have different performance requirements for the electronic control unit (ECU) contained within their product,
several observations should be made regarding the resources needed to support J1939.
The current data rate of J1939 is 250 Kbps.
A typical message containing 8 data bytes is 128 bits long (excluding bits used for bit stuffing) which in time is approximately 500 microseconds.
The shortest message is 64 bits long. This means that a new message could be sent every 250 microsecond.
Although not every message is relevant, nor is the bus loading likely to be above 50%,
the receiving processor must be able to handle (or buffer) back to back messages for short bursts of time.
This will require some RAM space as well as processor time for memory transfers.

Wiring Topology - Physical Layer (J1939/1x)

The J1939 network is intended to be a single, linear, shielded twisted pair of wires running around the vehicle to each ECU. A short stub is permitted between the ECU and the "bus". This simplifies routing the main bus wiring by not requiring the main bus to connect directly to each ECU. The linear bus is necessary at a data rate of 250 Kbps in order to minimize electrical signal reflections. The termination resistor at each end of the bus also reduces reflections.

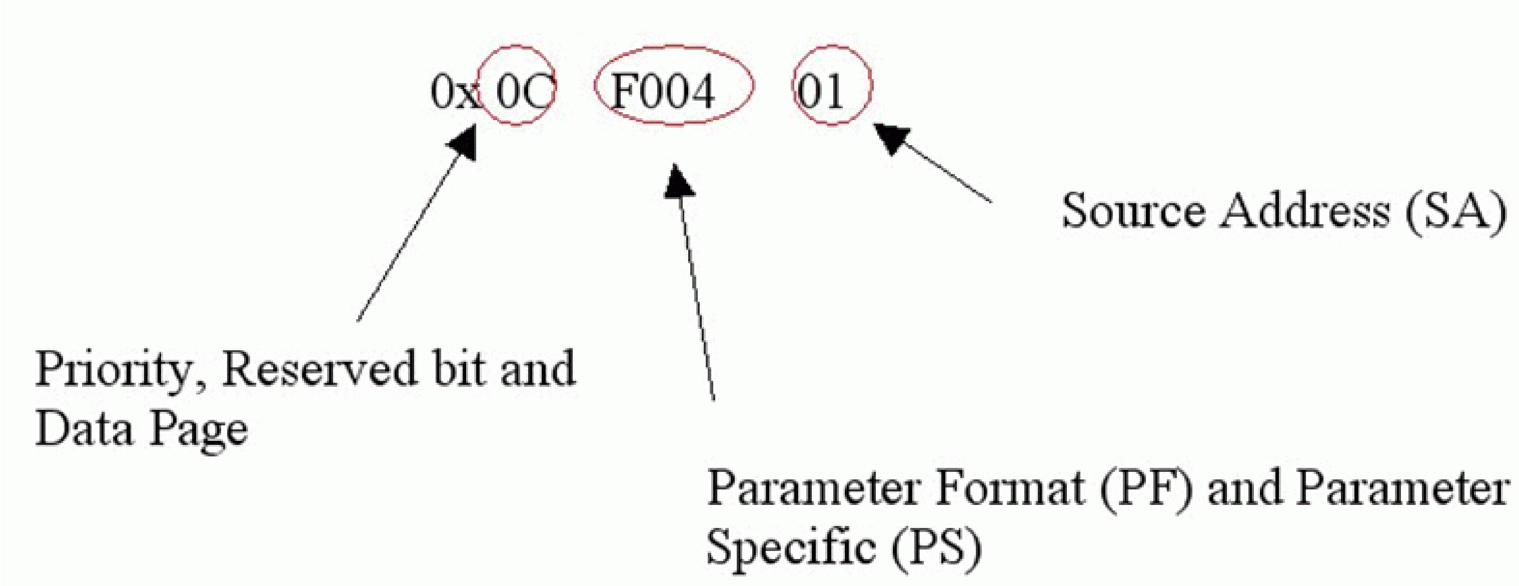
The J1939 network may actually be composed of multiple segments, with an in-line device known as a bridge present between them. These segments do not need to be directly compatible with each other. For instance, the segments may run at different data rates or use a different physical medium. The main function of the bridge is to provide electrical isolation between segments. In the event of a break on the wire between the tractor and trailer, the main J1939 segment on the tractor will continue to function. The bridge can also selectively filter which messages need to be stored and forwarded from one segment to another.

Example of how to interpret a J1939 message

This example is intended to provide the principles of how to interpret a J1939 message. Let’s look at a J1939 message with the following content:

CAN identifier: 0xCF00401
Data Bytes: 0xFF FF 82 DF 1A FF FF FF

What information does the CAN-ID provide?


0x 0C F004 01
Source Address (SA)
Priority, Reserved bit and Data Page
Parameter Format (PF) and Parameter Specific (PS)

First two bytes = 0x0C = 00001100 in binary format. The first 3 bits aren't used since the identifier only consists of 29 bits. The following 3 bits represents the message priority which in this case is 3. Thereafter follows a reserved bit and then the data page which are used to determine the complete PGN.

The last byte of the CAN-ID is the Source Address (address of the sending device) which in this case is 1.

The PGN = 0x0F004 which corresponds to the Electric Engine Controller #1 (EEC1) according to the J1939/71 document. This document also describes the parameters and their position within the data bytes. The data field consists of the following bytes in this example:

Data bytes	FF	FF	82	DF	1A	FF	FF	FF
Position	1	2	3	4	5	6	7	8

Data bytes 1, 2, 6, 7 and 8 in this example are not available and are therefore set to 0xFF. No raw parameter value (single byte) could have the value 0xFF.

Data byte 3 is the parameter Actual engine percent torque. The raw value 0x82 is 130 decimal. According to the J1939/71 document the scaling 1% per bit and the offset is -125. Therefore, the actual value for this parameter is 5%.

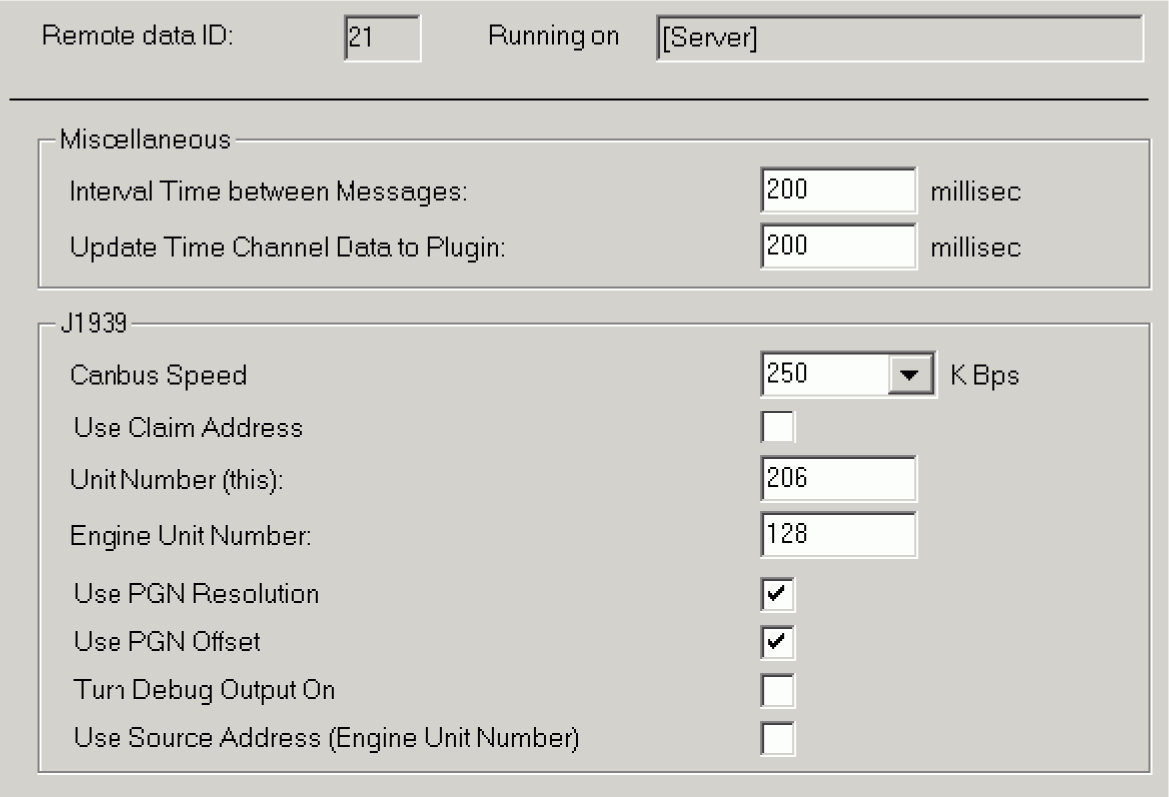
Data bytes 4 and 5 form the parameter Engine speed. The first byte (4) is the least significant (Intel byte order). The raw value 0x1ADF = 6879 decimal. The scaling is 0.125 rpm per bit and the offset is 0. The actual value for this parameter is therefore just under 859.875 rpm.

Shortcuts

Icon:

Place: Remote Data\

J1939 – General Settings



Remote data ID: 21

Running on [Server]

Miscellaneous

Interval Time between Messages: 200 millisec

Update Time Channel Data to Plugin: 200 millisec

J1939

Canbus Speed 250 K Bps

Use Claim Address □

Unit Number (this): 206

Engine Unit Number: 128

Use PGN Resolution ☑

Use PGN Offset ☑

Turn Debug Output On □

Use Source Address (Engine Unit Number) □

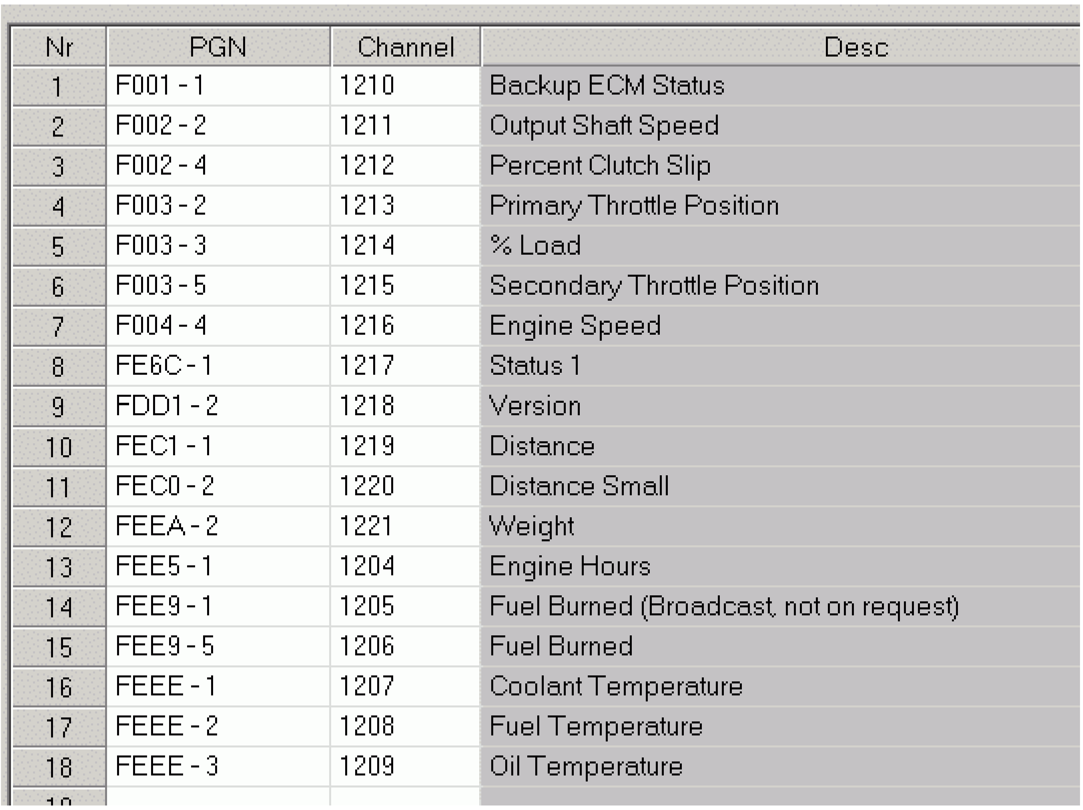
Setting	Description	Default
Interval Time between Messages	Minimal Time between two messages before reporting an error (50-100000 in milli-seconds	1000
Update Time channel Data to Plugin	When this protocol running on Server, IOServer uses this time to update channel values (50-100000 in milli-seconds	1000
Can Speed	Speed Setting for Canbus 125K, 250k, 500k 1M etc.	250k
Use Address Claim	Login Procedure to Canbus - yes/no	no
Unit Number (this)	Unit Address Number of Plugin	208
Engine Unit Number	Unit Address Number of Engine	128
Use PGN Resolution*	when receiving channel value scale to a certain resolution - yes/no	yes
Use PGN Offset*	when receiving channel value recalc with a certain offset - yes/no	yes
Turn Debug Output On	connect on comport1, text output what messages are received on Canbus - yes/no	no
Use Source Address (Engine Unit Number)	when having mutiple engines connected to same canbus - yes/no	no

* remark : resolution and offset are configured inside J1939 Messages List.

To Use Debug Setting to Monitor Received Canbus Messages:
J1939 How To Debug with UDP Port and Virtual Comport.

See also: J1939 Plugin

J1939 – Connection List



Nr	PGN	Channel	Desc
1	F001 - 1	1210	Backup ECM Status
2	F002 - 2	1211	Output Shaft Speed
3	F002 - 4	1212	Percent Clutch Slip
4	F003 - 2	1213	Primary Throttle Position
5	F003 - 3	1214	% Load
6	F003 - 5	1215	Secondary Throttle Position
7	F004 - 4	1216	Engine Speed
8	FE6C - 1	1217	Status 1
9	FDD1 - 2	1218	Version
10	FEC1 - 1	1219	Distance
11	FEC0 - 2	1220	Distance Small
12	FEEA - 2	1221	Weight
13	FEE5 - 1	1204	Engine Hours
14	FEE9 - 1	1205	Fuel Burned (Broadcast, not on request)
15	FEE9 - 5	1206	Fuel Burned
16	FEEE - 1	1207	Coolant Temperature
17	FEEE - 2	1208	Fuel Temperature
18	FEEE - 3	1209	Oil Temperature
19			

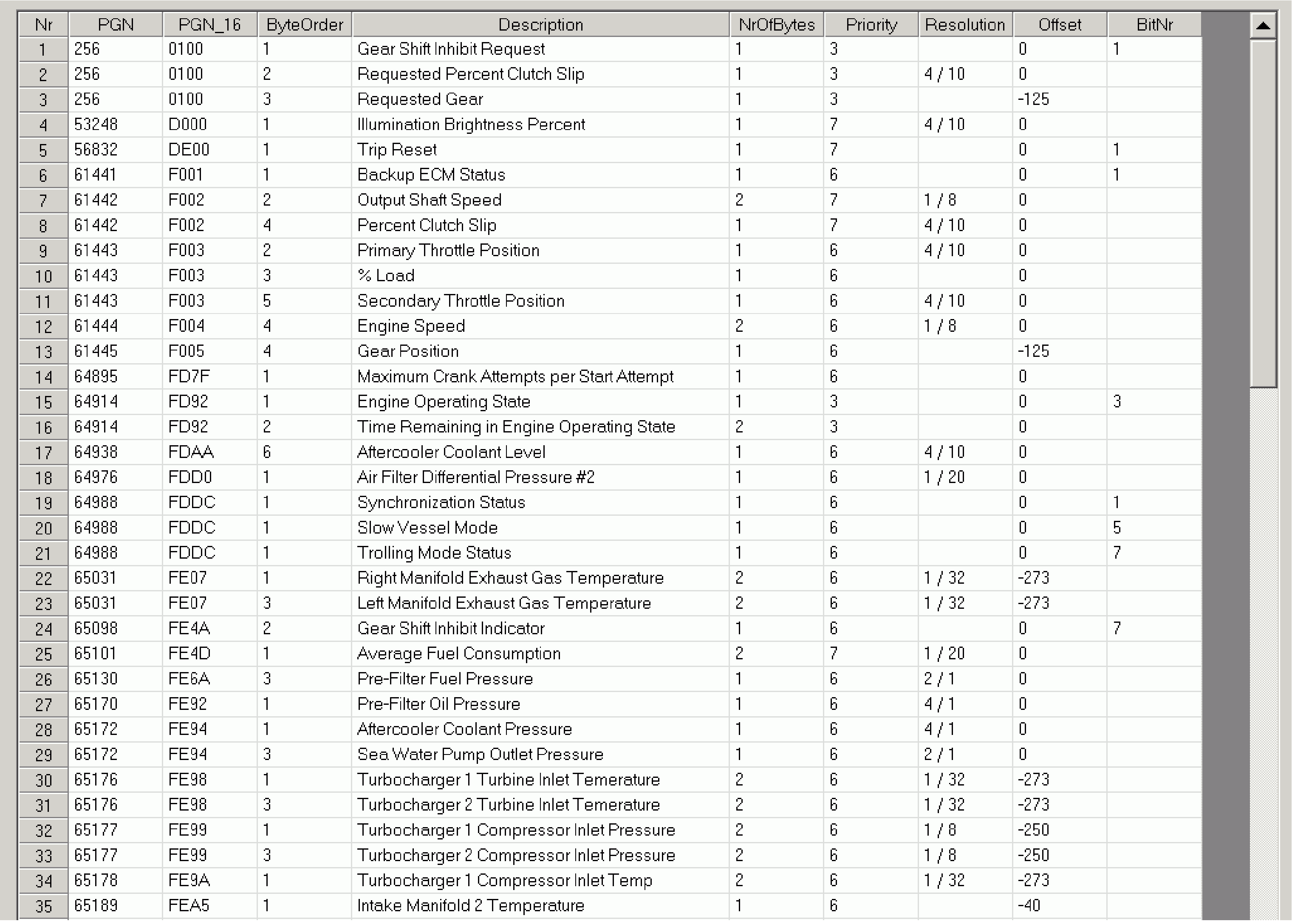
From the current selected row message definition is shown.

Field	Description
Nr	Number, row number
PGN	PGN hex Number, see Message List
Channel	Channel to retrieve Value from (Analog in) eq. Leaving field empty, this will delete entry
Desc	Description of PGN

See also:

J1939 Plugin

J1939 – Messages List



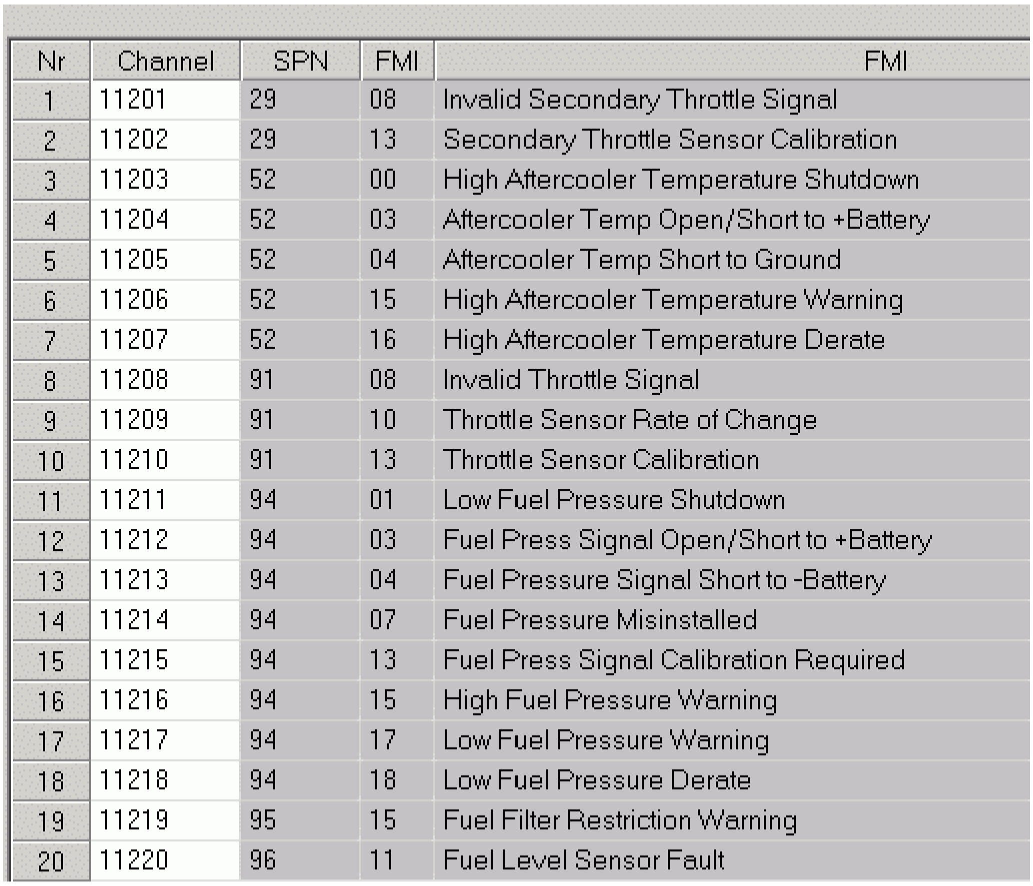
Nr	PGN	PGN_16	ByteOrder	Description	NrOfBytes	Priority	Resolution	Offset	BitNr
1	256	0100	1	Gear Shift Inhibit Request	1	3		0	1
2	256	0100	2	Requested Percent Clutch Slip	1	3	4 / 10	0	
3	256	0100	3	Requested Gear	1	3		-125	
4	53248	D000	1	Illumination Brightness Percent	1	7	4 / 10	0	
5	56832	DE00	1	Trip Reset	1	7		0	1
6	61441	F001	1	Backup ECM Status	1	6		0	1
7	61442	F002	2	Output Shaft Speed	2	7	1 / 8	0	
8	61442	F002	4	Percent Clutch Slip	1	7	4 / 10	0	
9	61443	F003	2	Primary Throttle Position	1	6	4 / 10	0	
10	61443	F003	3	% Load	1	6		0	
11	61443	F003	5	Secondary Throttle Position	1	6	4 / 10	0	
12	61444	F004	4	Engine Speed	2	6	1 / 8	0	
13	61445	F005	4	Gear Position	1	6		-125	
14	64895	FD7F	1	Maximum Crank Attempts per Start Attempt	1	6		0	
15	64914	FD92	1	Engine Operating State	1	3		0	3
16	64914	FD92	2	Time Remaining in Engine Operating State	2	3		0	
17	64938	FDAA	6	Aftercooler Coolant Level	1	6	4 / 10	0	
18	64976	FDD0	1	Air Filter Differential Pressure #2	1	6	1 / 20	0	
19	64988	FDDC	1	Synchronization Status	1	6		0	1
20	64988	FDDC	1	Slow Vessel Mode	1	6		0	5
21	64988	FDDC	1	Trolling Mode Status	1	6		0	7
22	65031	FE07	1	Right Manifold Exhaust Gas Temperature	2	6	1 / 32	-273	
23	65031	FE07	3	Left Manifold Exhaust Gas Temperature	2	6	1 / 32	-273	
24	65098	FE4A	2	Gear Shift Inhibit Indicator	1	6		0	7
25	65101	FE4D	1	Average Fuel Consumption	2	7	1 / 20	0	
26	65130	FE6A	3	Pre-Filter Fuel Pressure	1	6	2 / 1	0	
27	65170	FE92	1	Pre-Filter Oil Pressure	1	6	4 / 1	0	
28	65172	FE94	1	Aftercooler Coolant Pressure	1	6	4 / 1	0	
29	65172	FE94	3	Sea Water Pump Outlet Pressure	1	6	2 / 1	0	
30	65176	FE98	1	Turbocharger 1 Turbine Inlet Temerature	2	6	1 / 32	-273	
31	65176	FE98	3	Turbocharger 2 Turbine Inlet Temerature	2	6	1 / 32	-273	
32	65177	FE99	1	Turbocharger 1 Compressor Inlet Pressure	2	6	1 / 8	-250	
33	65177	FE99	3	Turbocharger 2 Compressor Inlet Pressure	2	6	1 / 8	-250	
34	65178	FE9A	1	Turbocharger 1 Compressor Inlet Temp	2	6	1 / 32	-273	
35	65189	FEA5	1	Intake Manifold 2 Temperature	1	6		-40	

Field	Description
Nr	row index of this grid, this number will be used in connection list
PGN	indentifier of message which will be received and handle by this plugin
PGN_16	same as PGN now displayed in hex code
Byte Order	Index number where data starts
Description	commentary on message for documentation only
Nr Of Bytes	data size of value
Priority	Priority level of Message
Resolution	value scale to engineering unit
Offset	value scale with offset
BitNr	start of bit number for digital status only

See also:

J1939 Plugin

J1939 – Diagnostic List



Nr	Channel	SPN	FMI	FMI
1	11201	29	08	Invalid Secondary Throttle Signal
2	11202	29	13	Secondary Throttle Sensor Calibration
3	11203	52	00	High Aftercooler Temperature Shutdown
4	11204	52	03	Aftercooler Temp Open/Short to +Battery
5	11205	52	04	Aftercooler Temp Short to Ground
6	11206	52	15	High Aftercooler Temperature Warning
7	11207	52	16	High Aftercooler Temperature Derate
8	11208	91	08	Invalid Throttle Signal
9	11209	91	10	Throttle Sensor Rate of Change
10	11210	91	13	Throttle Sensor Calibration
11	11211	94	01	Low Fuel Pressure Shutdown
12	11212	94	03	Fuel Press Signal Open/Short to +Battery
13	11213	94	04	Fuel Pressure Signal Short to -Battery
14	11214	94	07	Fuel Pressure Misinstalled
15	11215	94	13	Fuel Press Signal Calibration Required
16	11216	94	15	High Fuel Pressure Warning
17	11217	94	17	Low Fuel Pressure Warning
18	11218	94	18	Low Fuel Pressure Derate
19	11219	95	15	Fuel Filter Restriction Warning
20	11220	96	11	Fuel Level Sensor Fault

Field	Description
Nr	row index of this grid
Channel	Digital Channel which will be triggered into alarm/status high when fault occurs
SPN	Suspect Parameter Number
FMI	Failure Mode Identifier
Description	FMI Description

J1939 DTC

In SAE J1939, the acronym DTC stands for Diagnostic Trouble Code, also known as a fault code, and serves to identify the failed parameter. A DTC contains the Suspect Parameter Number (SPN) for the failed parameter, how many times failure has occured (OC), and how it has failed (FMI).

Active DTCs are transmitted by the DM1* message while non-active DTCs (i.e. historic) are transmitted by the DM2 message. The DM1 and DM2 messages may contain multiple DTCs which mean the message may be transmitted using the Transport Protocol (TP). Both the DM1 and DM2 messages are defined by the J1939-73 specification.

DTC Definition:
Suspect Parameter Number (SPN) 19 bits
Failure Mode Identifier (FMI) 5 bits
Occurrence Count (OC) 7 bits
SPN Conversion Method (CM) 1 bit

Note :DM1 message has PGN_16 on Canbus of "0xFECA"

What is an SPN?

Each parameter used in the J1939 network is described by the standard.
A Suspect Parameter Number (SPN) is a number that has been assigned by the SAE committee to a specific parameter.
Each SPN has the following detailed information associated with it:

-data length (in bytes)
-data type
-resolution
-offset
-range
-and a tag (label) for reference.

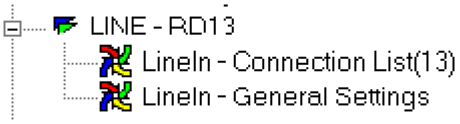
SPNs that share common characteristics will be grouped into a Parameter Group (PG) and will be transmitted to the network using the same PGN.

See also:

J1939 Plugin

LI – Line In Plugin

After selecting ‘Remote Data’ and LineIn



LINE - RD13

LineIn - Connection List(13)

LineIn - General Settings

There are several items of this remote data:

1. Line In - General Settings

2. Line In - Connection List

Line In Protocol only receives ASCII Messages

Message Format

<String>

In detail:

<Sub String 1>…< Sub String N ><End Of Line>

Where a sub string is a text to identify a unique line in the received data

Examples:

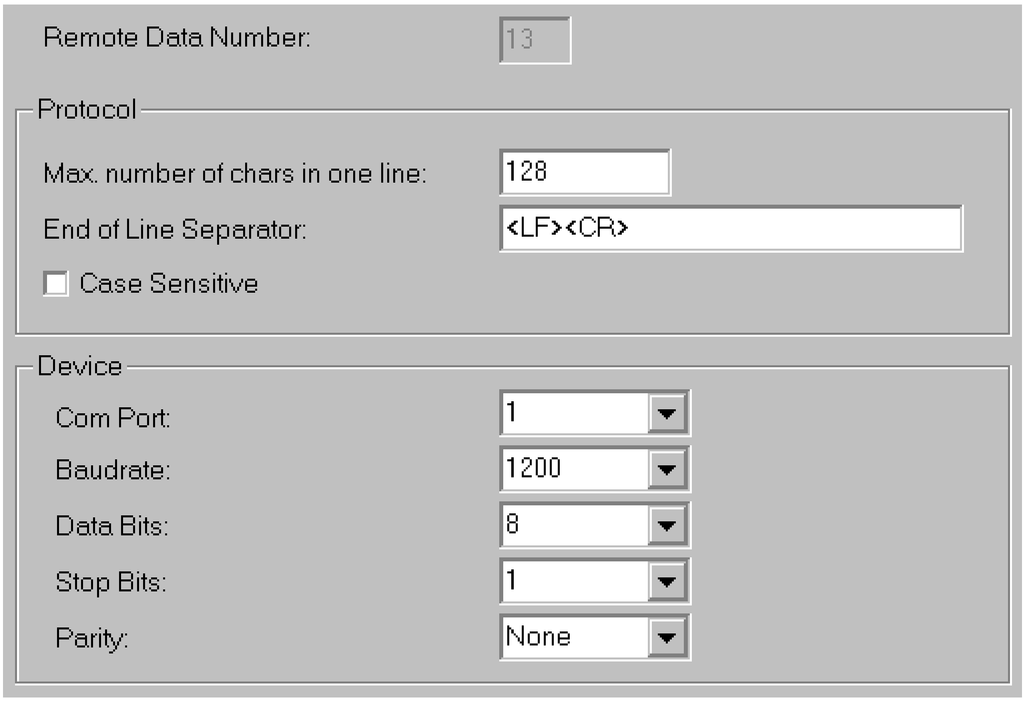
```
4:59p07aprAlarm! 66 PIR BD FBD P In Area #2 BRIDGE DECK
5:02p07aprRestore! 70 PIR BD AFT S In Area #2 BRIDGE DECK
```

Shortcuts

Icon:

Place: Remote Data\

LineIn – General Settings



Remote Data Number: 13

Protocol

Max. number of chars in one line: 128

End of Line Separator: <LF><CR>

□ Case Sensitive

Device

Com Port: 1

Baudrate: 1200

Data Bits: 8

Stop Bits: 1

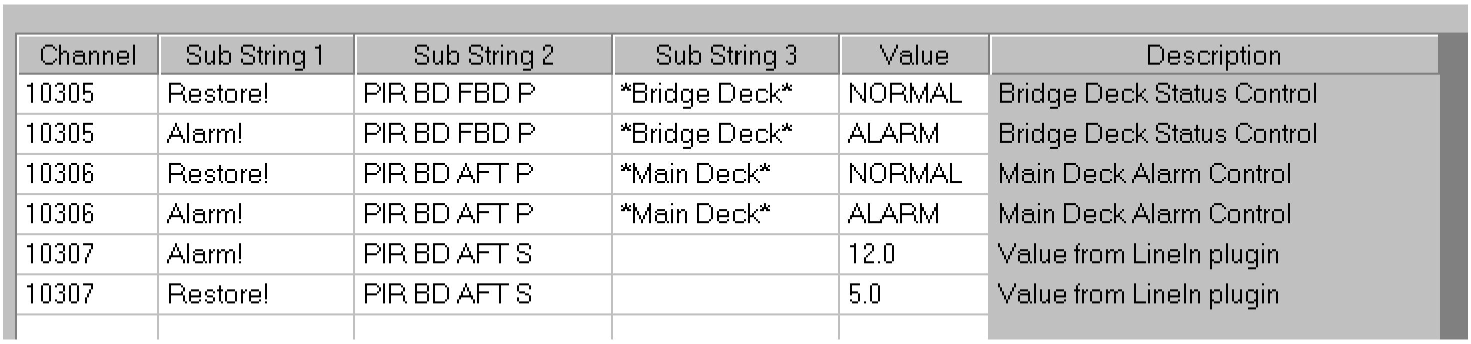
Parity: None

Setting	Description	Default
Com Port	Comport to use (1-4)	2
Baudrate	Communication Speed (1200, 2400, 4800, 9600, 19200)	1200
Data Bits	Number of Data bits (7, 8)	8
Parity	Communication Parity (None, Odd, Even, Space, Mark)	None
Stop Bits	Number of Stop Bits (0, 1, 2)	1
End Of Line Separator	Separator between two received lines	<LF><CR>
Case Sensitive	if Checked, use lower case and upper case chars from configuration	
Maximum number of chars in one line	Max. length of received line	128

See also:

Line In Plugin

LineIn – Connection List



Channel	Sub String 1	Sub String 2	Sub String 3	Value	Description
10305	Restore!	PIR BD FBD P	*Bridge Deck*	NORMAL	Bridge Deck Status Control
10305	Alarm!	PIR BD FBD P	*Bridge Deck*	ALARM	Bridge Deck Status Control
10306	Restore!	PIR BD AFT P	*Main Deck*	NORMAL	Main Deck Alarm Control
10306	Alarm!	PIR BD AFT P	*Main Deck*	ALARM	Main Deck Alarm Control
10307	Alarm!	PIR BD AFT S		12.0	Value from LineIn plugin
10307	Restore!	PIR BD AFT S		5.0	Value from LineIn plugin

Field	Description
Channel	Channel number to update (Analog in; Digital in; Analog out; Digital out)
Sub String 1	first find string
Sub String 2	second find string
Sub String 3	third find string
Value	Update value, (if setuped line is received, channel is updated with this value) (Alarm/Normal for Digital in/out or numeric value for Analog in/out)
Description	Channel Description, information

Remarks:

For faster configuration it is possible to use CTRL+D:

- configure one complete row
- select one column and multiple empty rows
- press CTRL+D, automaticially a default will filled in

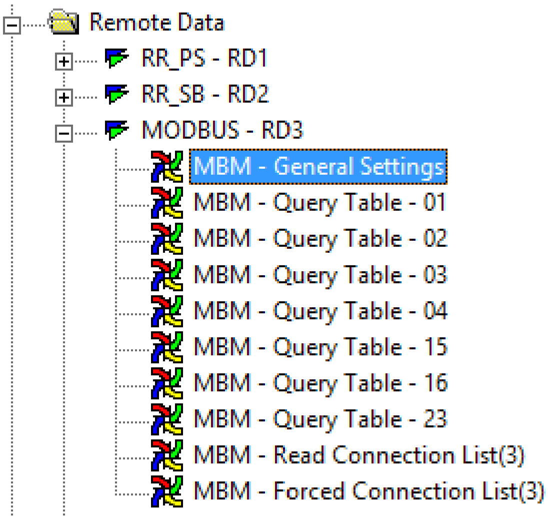
By double-clicking on channel column header sort action is performed.

See also:

Line In Plugin

MBM – Modbus Master Plugin

After selecting ‘Plugins’ and MBM – RD xx



Remote Data
RR_PS - RD1
RR_SB - RD2
MODBUS - RD3
MBM - General Settings
MBM - Query Table - 01
MBM - Query Table - 02
MBM - Query Table - 03
MBM - Query Table - 04
MBM - Query Table - 15
MBM - Query Table - 16
MBM - Query Table - 23
MBM - Read Connection List(3)
MBM - Forced Connection List(3)

There are several items of this plugin:

1. MBM - General Settings
2. MBM – Force Connection List
3. MBM - Query Table 01..04 or 15..16 or 23
4. MBM – Read Connection List

This plugin supports the following modbus functions:

Inputs

01 – Coils (Digital, error code 0x81)

02 – Status (Digital, error code 0x82))

03 – Hold Registers (Analog, error code 0x83)

04 – Input Registers (Analog, error code 0x84)

Outputs

05 – Write Single Coil (Digital, error code 0x85)

06 – Write Single Register (Analog, error code 0x86)

15 - Write Multiple Coils (Digital, error code 0x8F)

16 - Write Multiple registers (Analog, error code 0x90)

Inputs and Outputs

23 - Read/Write Multiple registers (Analog, error code 0x97)

Remark: There is no use for a treeview items like "MBM - Query Table 05" or 06.
Because the number of registers is 1 at function 05 and 06.
In this case Modbus Master will send a complete message with only one status or value.

Structures are sent according Motorola (MSB LSB or Big Endian) processor set.

Register Values are handled depending of configuration:

Register Range	Signed	Unsigned	8000H Offset
Negative	-32768..-1 (8000.. FFFF)		-32768..-1 (0..7FFF)
Positive	0..32767 (0000..7FFF)	0..65535 (0000..FFFFF)	0..32767 (8000..FFFF)
Error Value (Out of range)	-32768 (8000)	65535 (FFFF)	-32768 (0000)

(Values) use hex notation

For value extension and value precision another types of register handling is available.

The base types of these are:

- long (signed 32 bits)
- ulong (unsigned 32 bits)
- float (32 bits, contains one sign bit and exponent (two complement) and 23 bit mantissa)

Furthermore there could be a distinction between low byte and high byte sending/receiving order. (Abbreviation L/H or H/L) Normally H/L is most frequently used.

Therefore a Modbus register is always 16 bits, two registers addresses are used for a 32 bit value presentation.

Example Function 02 digital status request

RTU

SlaveNr (one byte)	Function Nr (one byte)	Address (two bytes)	Nr of statusses (two bytes)	CRC (two bytes)
0x01	0x02	0x01 0x00	0x00 0x09	*CRC*

ASCII

Header	SlaveNr (two byte)	Function Nr (two byte)	Address (four bytes)	Nr of statusses (four bytes)	LRC (two bytes)	Footer CR+LF
:	01	02	0100	0009	*LRC*	*CR+LF*

Example Function 02 digital status reply

SlaveNr (one byte)	Function Nr (one byte)	Byte Count (one byte)	Data (Byte Count bytes)	CRC (two bytes)
0x01	0x02	0x02	0xFF 0xFF	*CRC*

:010202FFFF
LRC+ **Example Function 02 digital status exception reply**

SlaveNr	Function Nr	Exception code	CRC

(one byte)	(one byte)	(one byte)	(two bytes)
0x01	0x**82**	**0x02**	*CRC*

:01**8202***LRC+CR+LF*

To get the status of the points with MODBUS address 100 to 113, when e.g. the points on address 103 and 108 are 'on' and the other points are 'off':

Slave number : 1

Data start field : 100 (0064 hex)

Data number : 14 (000E hex)

MODBUS request : 01 02 00 64 00 0E B8 11 (hex)

MODBUS reply : 01 02 02 08 01 7F B8 (hex)

Note: that the second data byte in the reply contains 2 dummy coils, their value is supposed random (here taken to be 'off') but do contribute to the CRC.

Function 23 (0x17) Read/Write Multiple registers

This function code performs a combination of one read operation and one write operation in a single MODBUS transaction.

The write operation is performed before the read. Holding registers are addressed starting at zero.

Therefore holding registers 1-16 are addressed in the PDU as 0-15.

The request specifies the starting address and number of holding registers to be read as well as the starting address, number of holding registers, and the data to be written.

The byte count specifies the number of bytes to follow in the write data field.

The normal response contains the data from the group of registers that were read.

The byte count field specifies the quantity of bytes to follow in the read data field.

Request:

Description	Size	Range
Slave Number	1 Byte	0x01 .. 0xFF
Function code	1 Byte	0x17
Read Starting Address	2 Bytes	0x0000 .. 0xFFFF
Quantity to Read	2 Bytes	0x0001 .. 0x007D
Write Starting Address	2 Bytes	0x0000 .. 0xFFFF
Quantity to Write	2 Bytes	0x0001 .. 0X0079
Write Byte Count	1 Byte	2 x N*
Write Registers Value	N*x 2 Bytes	-
Error Check	2 Bytes	CRC

*N = Quantity to Write

Response:

Description	Size	Range
Slave Number	1 Byte	0x01 .. 0xFF
Function code	1 Byte	0x17

Byte Count	1 Byte	2 x N*
Read Registers Value	N*x 2 Bytes	-
Error Check	2 Bytes	CRC

*N' = Quantity to Read

Expection Codes:

01 ILLEGAL FUNCTION
The function code received in the query is not an allowable action for the server (or slave).
This may be because the function code is only applicable to newer devices, and was not implemented in the unit selected.
It could also indicate that the server (or slave) is in the wrong state to process a request of this type,
for example because it is unconfigured and is being asked to return register values.

02 ILLEGAL DATA ADDRESS
The data address received in the query is not an allowable address for the server (or slave).
More specifically, the combination of reference number and transfer length is invalid.
For a controller with 100 registers, the PDU addresses the first register as 0, and the last one as 99.
If a request is submitted with a starting register address of 96 and a quantity of registers of 4,
then this request will successfully operate (address-wise at least) on registers 96, 97, 98, 99.
If a request is submitted with a starting register address of 96 and a quantity of registers of 5,
then this request will fail with Exception Code 0x02 “Illegal Data Address” since it attempts to operate on registers 96, 97, 98, 99
and 100, and there is no register with address 100.

03 ILLEGAL DATA VALUE
A value contained in the query data field is not an allowable value for server (or slave).
This indicates a fault in the structure of the remainder of a complex request, such as that the implied length is incorrect.
It specifically does NOT mean that a data item submitted for storage in a register has a value outside the expectation
of the application program, since the MODBUS protocol is unaware of the significance of any particular value of any particular register.

04 SLAVE DEVICE FAILURE
An unrecoverable error occurred while the server (or slave) was attempting to perform the requested action.

05 ACKNOWLEDGE
Specialized use in conjunction with programming commands.
The server (or slave) has accepted the request and is processing it, but a long duration of time will be require
dto do so. This response is returned to prevent a timeout error from occurring in the client (or master).
The client (or master) can next issue a Poll Program Complete message to determine if processing is completed.

06 SLAVE DEVICE BUSY
Specialized use in conjunction with programming commands.
The server (or slave) is engaged in processing a long–duration program command.
The client (or master) should retransmit the message later when the server (or slave) is free.

08 MEMORY PARITY ERROR
Specialized use in conjunction with function codes 20 and 21
and reference type 6, to indicate that the extended file area failed to pass a consistency check.
The server (or slave) attempted to read record file, but detected a parity error in the memory.
The client (or master) can retry the request, but service may be required on the server (or slave) device.

0A GATEWAY PATH UNAVAILABLE

Specialized use in conjunction with gateways, indicates that the gateway was unable to allocate an internal communication path
from the input port to the output port for processing the request.
Usually means that the gateway is misconfigured or overloaded.

0B GATEWAY TARGET DEVICE FAILED TO RESPOND

Specialized use in conjunction with gateways, indicates that no response was obtained from the target device.
Usually means that the device is not present on the network.

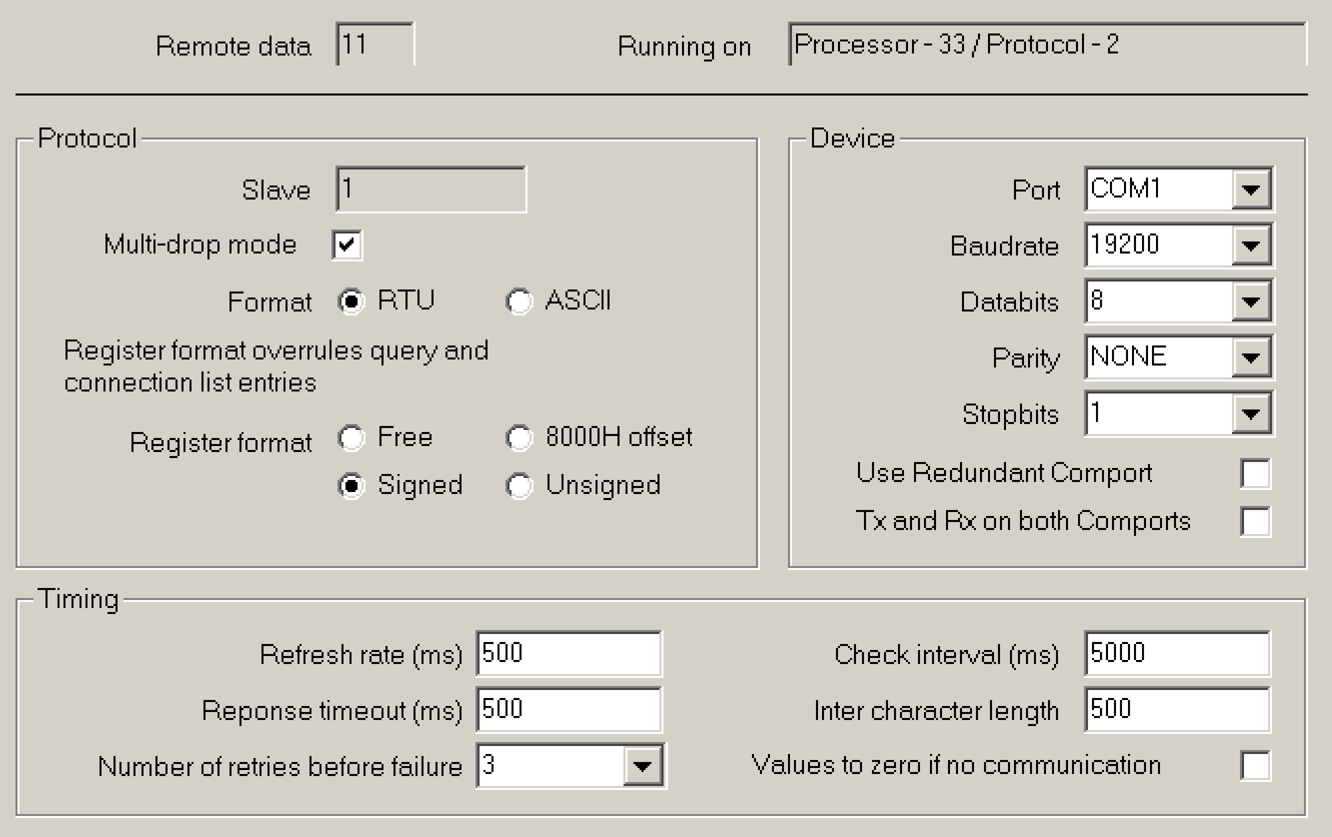
For more information reference the Modbus Specification PI_MBUS_300.pdf Revision A, See http://www.modbus.org/.

Shortcuts

Icon:

Place: Plugins\

MBM – General Settings



Remote data 11

Running on Processor - 33 / Protocol - 2

Protocol

Slave 1

Multi-drop mode ☑

Format ● RTU ○ ASCII

Register format overrules query and connection list entries

Register format ○ Free ○ 8000H offset ● Signed ○ Unsigned

Device

Port COM1

Baudrate 19200

Databits 8

Parity NONE

Stopbits 1

Use Redundant Comport □

Tx and Rx on both Comports □

Timing

Refresh rate (ms) 500

Reponse timeout (ms) 500

Number of retries before failure 3

Check interval (ms) 5000

Inter character length 500

Values to zero if no communication □

Header settings

Field	Options/Values and Function
Remote data	Remote data number which is being configured
Running on	Server: Protocol runs on the Server as a Plug-in or on LCD / XP / TFT : Processor Number Protocol Order: 1-4 for XP, 1-2 for TFT, 1 for LCD

Protocol settings

<table>
<tr><th>Field</th><th>Options/Values and Function</th></tr>
<tr><td><b>Slave</b></td><td>Single slave number, is used for entire plug-in.
(default 1)
<i>The “SLAVE” field of the conection list and of all valid query blocks in the database using this plug-in must be updated to hold this number</i></td></tr>
<tr><td><b>Multi-drop mode</b></td><td>Slave number is ignored and multiple slave numbers can be entered in the other Configuration screens.
(default unchecked/off)</td></tr>
<tr><td><b>Format</b></td><td>RTU: binary data exchange (Device 8 databits)
ASCII: text data exchange (Device 7 databits)
(default RTU)</td></tr>
<tr><td><b>Register Handling</b></td><td>How are Registers Handled (Inhibits configuration of this Field in Queries and connectionlist)
<table>
<tr><th>Value</th><th>Function</th></tr>
<tr><td><b>Free</b></td><td>Query or Connection list Register handling is used</td></tr>
<tr><td></td><td></td></tr>
</table></td></tr>
</table>

	Signed	Negative/Positive values (Sign bit)
	Unsigned	Positive values
	8000H Offset	zero point becomes 0x08000 (no Sign bit)
	(default Signed)	

Device settings

Field	**Options/Values and Function**
Port	Communication port which is used for communication. 1 till 20 (Default 1) *Server: No other plug-in/applications must use the same COM port* *XP: COM1 - COM4*
Baudrate	Baudrate of the Com Port, 1200, 2400, 4800, 9600 or 19200 (default: 19200)
Databits	Databits per byte used to communicate, depending on protocol format 5, 6, 7, 8 (default: 7 (Ascii),8 (RTU))
Parity	Parity used to communicate NONE, ODD, EVEN, MARK, SPACE (default: NONE)
Stopbits	Stopbits per byte used to communicate 1, 2 (default: 1)
Use Redundant Comport	special feature to use a second comport ('Port'+1) some engines, PLC's supports this feature (default unchecked/off)
Tx and Rx on both Comports	this feature depends of 'Use Redundant Comport' all queries will be send/receive on both Comports (default unchecked/off)

Timing settings

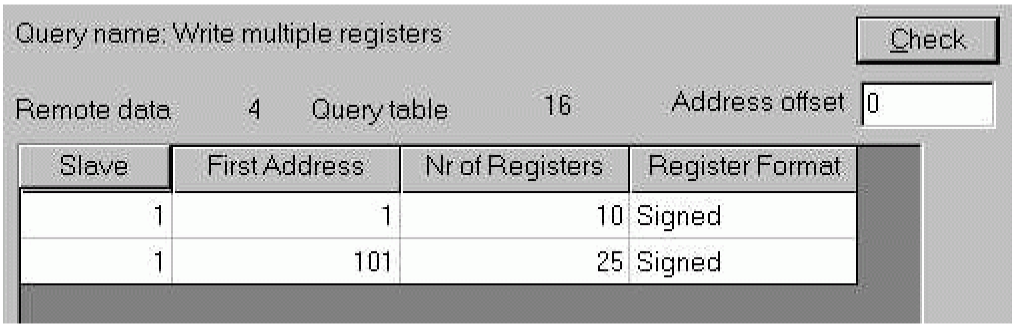
Field	**Options/Values and Function**
Refresh Rate	Time between same Queries, when refresh rate is too small the Communication runs as fast as possible, Time is used divided by configured Queries (Also function 5 and 6) to spread Queries instead of burst sending. 200-300000 millisec (default: 1000)
Response Time Out	Maximum Time between Query and Answer, when time expires a retry is done. Modbus Slave Device must supply this information 50-5000 millisec (default: 200)
Number Of Retries Before Failure	Number of re-queries before a slave not present is generated. 0 - 10 (default: 3)

Check Interval	Time between queries to not present Slaves to see if they are present now. 200-300000 millisec (default: 5000)
Inter Character Length	Time allowed between received bytes, is not trustworthy on Server. 0-10000 bits (default: 3000) *value may NOT be smaller when protocol running on Server* Processor XP eq. 125, when long Queries are retried increase value until responses are received.
Values to zero if no communication	after a certain time ('Refresh time' * 'Number Of Retries Before Failure') and nothing is received (no response on any Query) the to be received values and statusses will be defaulted to zero (default unchecked/off)

See also:

MBM - General Information Register Handling etc.
MBM - Query Table
MBM – Read Connection List
MBM – Force Connection List

MBM –Query Table



Query name: Write multiple registers

Remote data 4 Query table 16 Address offset 0

Slave	First Address	Nr of Registers	Register Format
1	1	10	Signed
1	101	25	Signed

Grid Column	Options/Values and Function
Slave Nr	Slave Number, can be forced by General Settings, 1..255
First Address	Address value, 0..65535
Nr Of Registers	Number of registers to be received or be sent, when too many registers are queried multiple queries are automatically generated, 0..65535
Register Handling	How is register handled, can be forced by General Settings - 8000H Offset means zero point becomes 0x08000 Signed, Unsigned, 8000H Offset or Free (default Signed) Long (H/L), Long (L/H), ULong (H/L), ULong (LH), Float (H/L), Float (L/H)

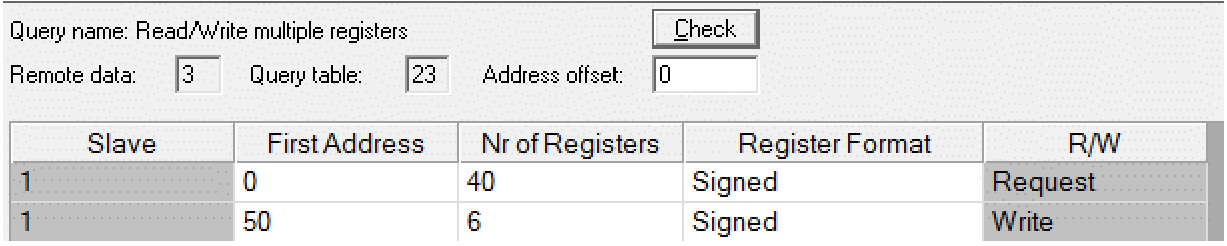
By pressing on column "Slave Nr" a sort action is performed, first time ascending second descending.

Field/Control Function

Query Table	Modbus function number of shown queries
Address Offset	Offset so Configured Modbus addresses are in accordance with the Modbus list and queries with hardware Modbus addresses used, certain plc's send 40001 as 0 and 10001 as 0., 0..65535
Check Button	Check if all connection list items are inside the configured modbus queries

Remark: There is no use for a treeview items like "MBM - Query Table 05" or 06.
Because the number of registers is 1 at function 05 and 06.
In this case only configuration at force connection list is sufficient.

Example: Function 23, one query configured by two lines.



Query name: Read/Write multiple registers

Remote data: 3 Query table: 23 Address offset: 0

Slave	First Address	Nr of Registers	Register Format	R/W
1	0	40	Signed	Request
1	50	6	Signed	Write

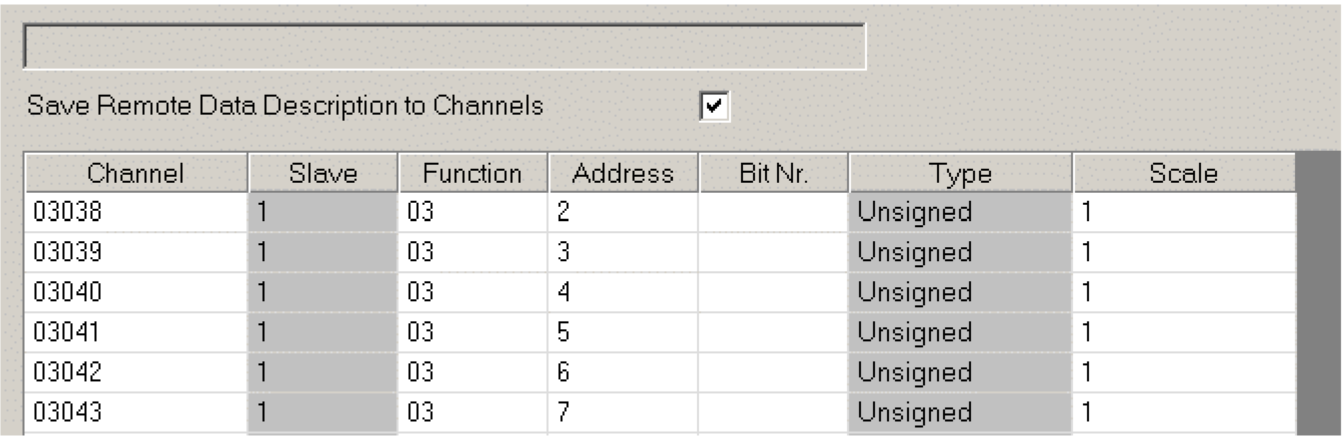
See also:

MBM - General Settings

MBM – Force Connection List

MBM – Read Connection List

MBM – Read Connection List



Save Remote Data Description to Channels ☑

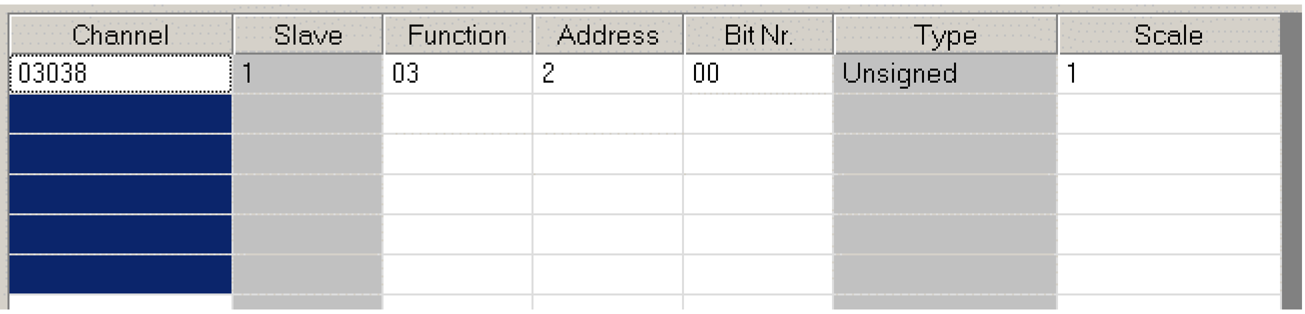
Channel	Slave	Function	Address	Bit Nr.	Type	Scale
03038	1	03	2		Unsigned	1
03039	1	03	3		Unsigned	1
03040	1	03	4		Unsigned	1
03041	1	03	5		Unsigned	1
03042	1	03	6		Unsigned	1
03043	1	03	7		Unsigned	1

Grid Column	Options/Values and Function
Channel	Channel to receive the data; type in an existing digital/analog channel number, or get here from the channel's configuration form (**status/value link**)
Slave Nr	Slave Number, can be forced by General Settings, 1..255
Function	Modbus function number, 01,02,03 or 04
Address	Modbus address of modbus function where value is retrieved from. 0...65535
Bit Nr	Bit to extract from register 0..15 or (nothing) where (nothing) means 'no bit' or entire register
Type	How is register handled - 8000H Offset means zero point becomes 0x08000 Signed, Unsigned, 8000H Offset or Free (default Signed) Long (H/L), Long (L/H), ULong (H/L), ULong (LH), Float (H/L), Float (L/H)
Scale	Scale factor of send register values *1000, *100, *10, 1, /10, /100 or /1000 (Default 1)
Channel Description	for showing channel description this field is read only

By pressing on column "Channel" or "Slave" or "Function" or "Address" a sort action is performed. First time ascending second time descending.

Return Button: Return from gateway, only available when getting there from a channel configuration (status/value link)

Tip: For fast configuration, configure a field, select one column and multiple rows by mouse, (click on first column, keep mouse left button down, drag mouse pointer to next row) and press CTRL+D

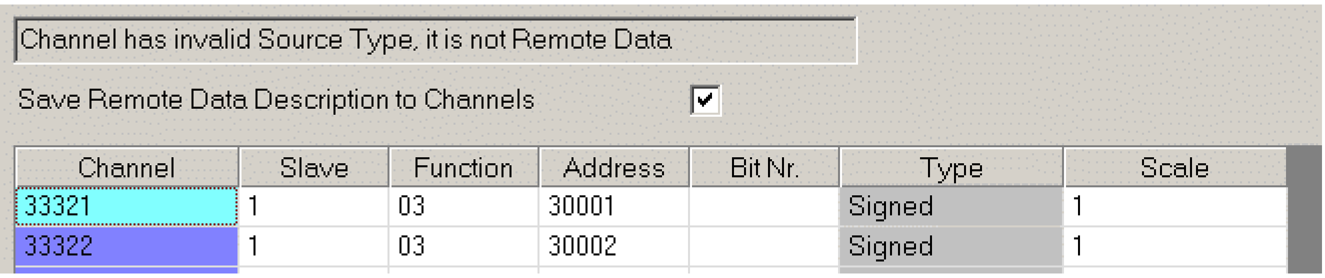


Channel	Slave	Function	Address	Bit Nr.	Type	Scale
03038	1	03	2	00	Unsigned	1

Slave field in single slave mode will not be editable and show general slave number.

Register-handling field is non-editable when overruling Register Handling is active this is when:
- General Register Handling isn't Free
- Queries Register Handling for the Modbus Slave/Function where Modbus Address is part of isn't free
- Digital status/coil or binary packed register

Scale 1 means rounded whole channel values are send.
Scale *10 means rounded whole 'channel values*10' are send.
Scale /10 means rounded whole 'Channel values/10' are send.



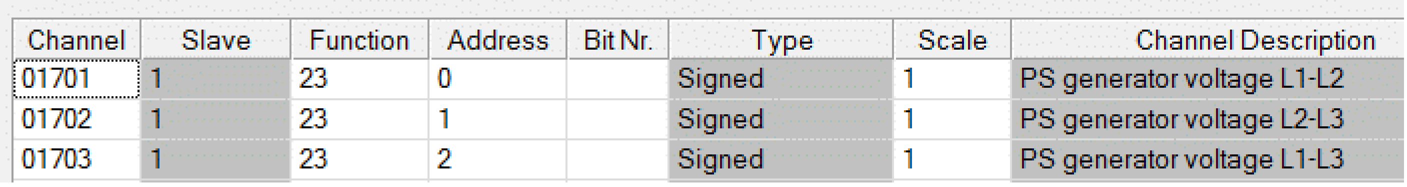
Channel has invalid Source Type, it is not Remote Data

Save Remote Data Description to Channels ☑

Channel	Slave	Function	Address	Bit Nr.	Type	Scale
33321	1	03	30001		Signed	1
33322	1	03	30002		Signed	1

It is possible to have different colors at 'Channel' field
It marks a configuration error.
In this example it says that a channel is not configured as remote data.
But this is required because MBM has to write it's received value to a channel!

Example: Function 23



Channel	Slave	Function	Address	Bit Nr.	Type	Scale	Channel Description
01701	1	23	0		Signed	1	PS generator voltage L1-L2
01702	1	23	1		Signed	1	PS generator voltage L2-L3
01703	1	23	2		Signed	1	PS generator voltage L1-L3

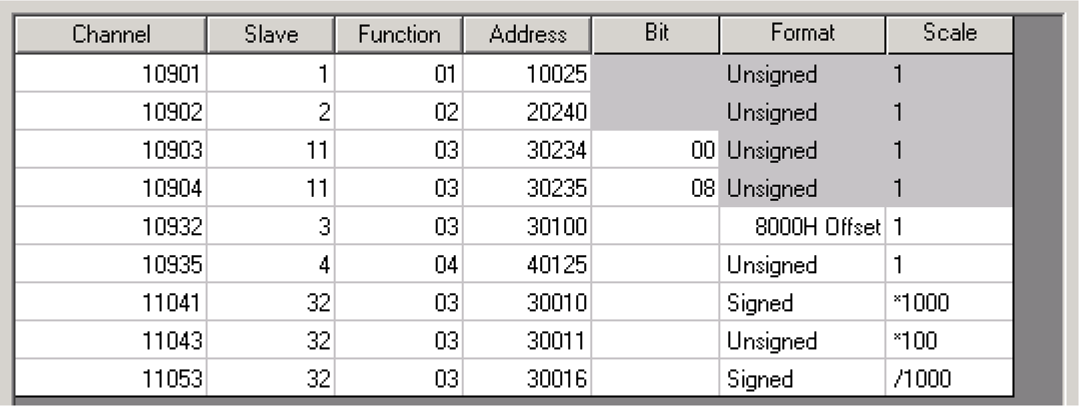
See also:

MBM - General Settings

MBM – Force Connection List

MBM - Query Table

MBM – Forced Connection List



Channel	Slave	Function	Address	Bit	Format	Scale
10901	1	01	10025		Unsigned	1
10902	2	02	20240		Unsigned	1
10903	11	03	30234	00	Unsigned	1
10904	11	03	30235	08	Unsigned	1
10932	3	03	30100		8000H Offset	1
10935	4	04	40125		Unsigned	1
11041	32	03	30010		Signed	*1000
11043	32	03	30011		Unsigned	*100
11053	32	03	30016		Signed	/1000

Grid Column	**Options/Values and Function**
Channel	Channel to send the data; type in an existing digital/analog channel number, or get here from the channel’s configuration form (**status/value link**)
Slave Nr	Slave Number, can be forced by General Settings, 1..255
Function	Modbus function number, 05,06,15 or 16
Address	Modbus address of modbus function where value is send to. 0...65535
Register Bit	Bit in register 0..15 or (nothing) where (nothing) means ‘no bit’ or entire register
Register Handling	How is register handled, can be forced by General Settings or Query Settings - 8000H Offset means zero point becomes 0x08000 Signed, Unsigned, 8000H Offset or Free (default Signed) Long (H/L), Long (L/H), ULong (H/L), ULong (LH), Float (H/L), Float (L/H)
Scale	Scale factor of send register values *1000, *100, *10, 1, /10, /100 or /1000 (Default 1)
Channel Description	for showing channel description this field is read only

By pressing on column “Channel” or “Slave” or “Function” or “Address” a sort action is performed. First time ascending second time descending.

Return Button: Return from gateway, only available when getting there from a channel configuration (status/value link)

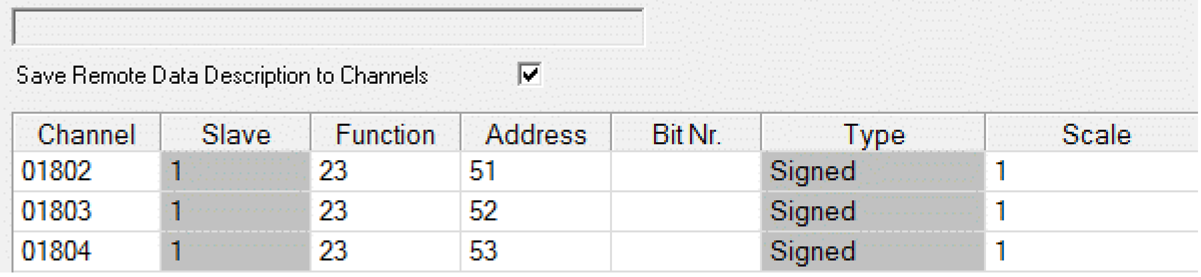
Tip: For fast channel configuration, configure last line complete, press not enter at the last column, go back to first column, give a channel range in (like 10101-10125) and press enter, now multiple rows are created!

Slave field in single slave mode will not be editable and show general slave number.

Register-handling field is non-editable when overruling Register Handling is active this is when:
- General Register Handling isn't Free
- Queries Register Handling for the Modbus Slave/Function where Modbus Address is part of isn't free
- Digital status/coil or binary packed register

Scale 1 means rounded whole channel values are send.
Scale *10 means rounded whole 'channel values*10' are send.
Scale /10 means rounded whole 'Channel values/10' are send.

Example: Function 23



Save Remote Data Description to Channels ☑

Channel	Slave	Function	Address	Bit Nr.	Type	Scale
01802	1	23	51		Signed	1
01803	1	23	52		Signed	1
01804	1	23	53		Signed	1

See also:

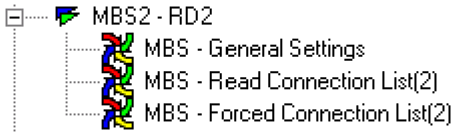
MBM - General Settings

MBM - Query Table

MBM – Read Connection List

MBS – Modbus Slave Plugin

After selecting ‘Remote Data’ and MBS – RD xx



MBS2 - RD2

MBS - General Settings

MBS - Read Connection List(2)

MBS - Forced Connection List(2)

There are several items of this remote data:

1. MBS - General Settings
2. MBS - Read Connection List
3. MBS – Forced Connection List

This plugin supports the following modbus functions:

Outputs (Read Connection List)

01 – Coils (Digital, error code 0x80)

02 – Status (Digital, error code 0x82))

03 – Hold Registers (Analog, error code 0x83)

04 – Input Registers (Analog, error code 0x84)

Inputs (Forced Connection List)

05 – Single Coil (Digital, error code 0x85)

06 – Single Register (Analog, error code 0x86)

15 - Multiple Coils (Digital, error code 0x8F)

16 - Multiple registers (Analog, error code 0x90)

Structures are sent according Motorola (MSB LSB or Big Endian) processor set.

For Read Connection list it’s possible to send with function 03 or 04 16 digital values mapped to one modbus address. For each value specify the bitnumber within this address. (not for modicon addressing)

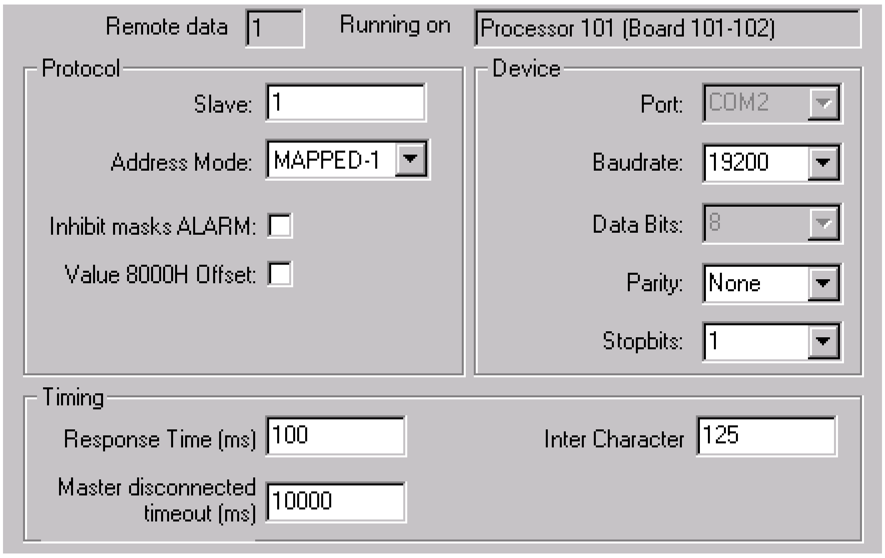
For more information reference the Modbus Specification PI_MBUS_300.pdf Revision A, See http://www.modbus.org/.

Shortcuts

Icon:

Place: Remote Data\

MBS – General Settings



Remote data 1 Running on Processor 101 (Board 101-102)

Protocol

Slave: 1

Address Mode: MAPPED-1

Inhibit masks ALARM: ☐

Value 8000H Offset: ☐

Device

Port: COM2

Baudrate: 19200

Data Bits: 8

Parity: None

Stopbits: 1

Timing

Response Time (ms) 100

Inter Character 125

Master disconnected timeout (ms) 10000

Header settings

Field	Options/Values and Function
Remote data	Remote data being configured
Running on	Server: Protocol runns on the Server as a Plug-in Processor LXX (LBB-LBB): Protocol on Processor board (L = Link, XX = Processor number, BB = Board number) eq. Processor 203 (215-221)

Protocol settings

<table>
<tr><th>Field</th><th>Options/Values and Function</th></tr>
<tr><td><b>Slave</b></td><td>Slave number
1-255
(default 1)</td></tr>
<tr><td><b>Address Mode</b></td><td>One of three options can be selected:
<table>
<tr><th>Value</th><th>Function</th></tr>
<tr><td><b>Mapping</b></td><td>Addresses are used on the serial line as configured</td></tr>
<tr><td><b>Mapping-1</b></td><td>Configured Addresses are used with a offset of - 1 to the serial line
eq. PAL 40001 -> Line 40000</td></tr>
<tr><td><b>Modicon Style</b></td><td>MODICON Offsets are used for each modbus function
See Forced or Read Connection-list for offsets</td></tr>
</table>
(default Mapping)</td></tr>
<tr><td><b>Value 8000H Offset</b></td><td>Selected zero point becomes 0x8000 (replaces sign bit) (default: deselected)</td></tr>
<tr><td><b>Inhibits Masks Alarm</b></td><td><i>true</i>: if channel is inhibited all alarm bits return zero irrespective their actual value
<i>false</i>: all alarm bits return actual value whether inhibited or not</td></tr>
</table>

Device settings

Field	Options/Values and Function

Com Port	Communication port which is used for communication. 1 till 20 (Default 2)
Baudrate	Baudrate of the Com Port, 1200, 2400, 4800, 9600 or 19200 (default: 19200)
Databits	Databits per byte used to communicate 8 (default: 8 (RTU))
Parity	Parity used to communicate NONE, ODD, EVEN, MARK, SPACE (default: NONE)
Stopbits	Stopbits per byte used to communicate 1, 2 (default: 1)

Timing settings

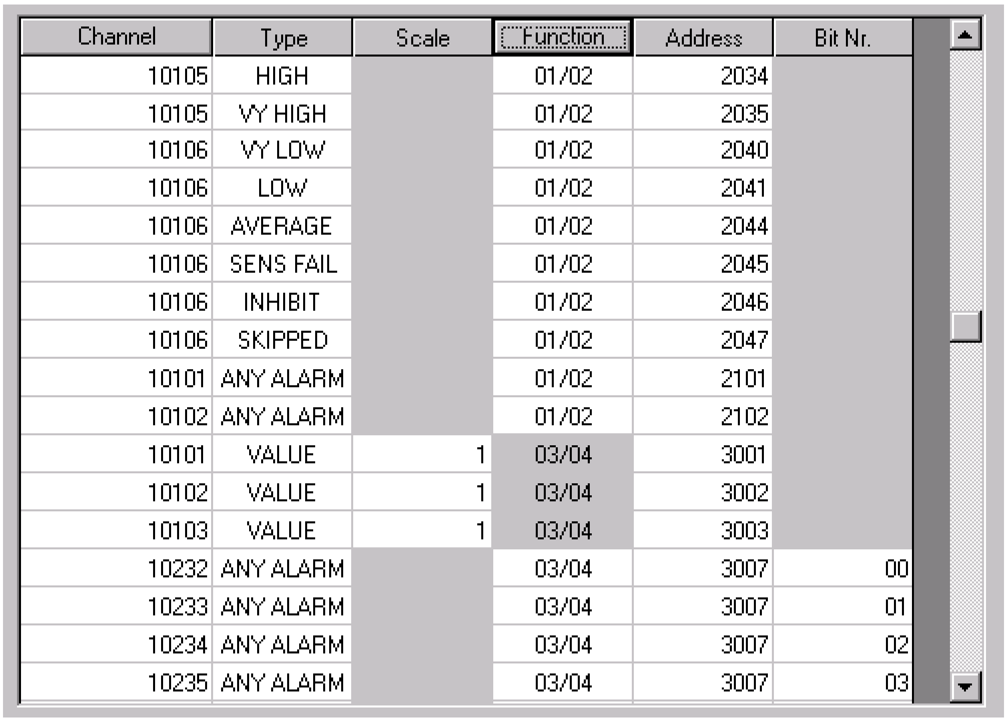
Field	Options/Values and Function
Response time (ms)	Time to wait between Receiving complete query and processing and sending Answers. Needed if Modbus master needs time to switch handshaking. (default: 100)
Master disconnected timeout (ms)	Time that has to expire before diagnostics are triggered, see Diagnostics. - No communication, on serial line no bytes are received - No master query, no valid requests received since last answer send time of 0 disables these diagnostics. (default: 10000)
Inter Character Length	Time allowed between received/send bytes, is not trustworthy on Server. 0-10000 bits (default: 3000) Server is not trustworthy use value of 3000 Processor a lower value can be used eq 35 (3,5 charachterlength)

Remarks:
When protocol running on Processor only the com2 port can be used and is overruled in the board software.
When protocol running on Server a com port can only be used by one Application/Plug-in.

See also:

Modbus Slave Plugin
Modbus Slave Diagnostics
Modbus slave read connection list
Modbus slave forced connection list

MBS – Read Connection List



Channel	Type	Scale	Function	Address	Bit Nr.
10105	HIGH		01/02	2034	
10105	VY HIGH		01/02	2035	
10106	VY LOW		01/02	2040	
10106	LOW		01/02	2041	
10106	AVERAGE		01/02	2044	
10106	SENS FAIL		01/02	2045	
10106	INHIBIT		01/02	2046	
10106	SKIPPED		01/02	2047	
10101	ANY ALARM		01/02	2101	
10102	ANY ALARM		01/02	2102	
10101	VALUE	1	03/04	3001	
10102	VALUE	1	03/04	3002	
10103	VALUE	1	03/04	3003	
10232	ANY ALARM		03/04	3007	00
10233	ANY ALARM		03/04	3007	01
10234	ANY ALARM		03/04	3007	02
10235	ANY ALARM		03/04	3007	03

Column	Options or Values
Channel	Channel to send the data; type in an existing digital/analog channel number, or get here from the channel's configuration form (**status/value link**)
Type	Value or one of the status bits Value, Very Low, Low, High, Very High, Sensfail, Average, Any Alarm, Inhibited, Skipped (default depends on channel)
Scale	Scale value before send *1000, *100, *10, 1, /10, /100, /1000 (default 1)
Function	Function number, like 01/02 or 03/04 for MODICON addressing like 01, 02, 03 or 04
Address	Address value
Bit Nr.	bit number, applies to function 03 or 04 with status connection only 00, 01 till 15 (default 00)

A modbus master receives channel information from the slave. (=Output List)

Sorting
By pressing on column "Channel" or "Function" a sort action is performed, first time ascending second descending.

Addres field of actual message depends on Addres Mode, MBS - General Settings

<table>
<tr><th>Option</th><th>Conversion</th></tr>
<tr><td>"Mapping / No Offset"</td><td>straight copy of address</td></tr>
<tr><td>"Mapping / Offset - 1"</td><td>address - 1</td></tr>
<tr><td>"Modicon Style"</td><td><table><tr><th>Function</th><th>Conversion</th></tr><tr><td></td><td></td></tr></table></td></tr>
</table>

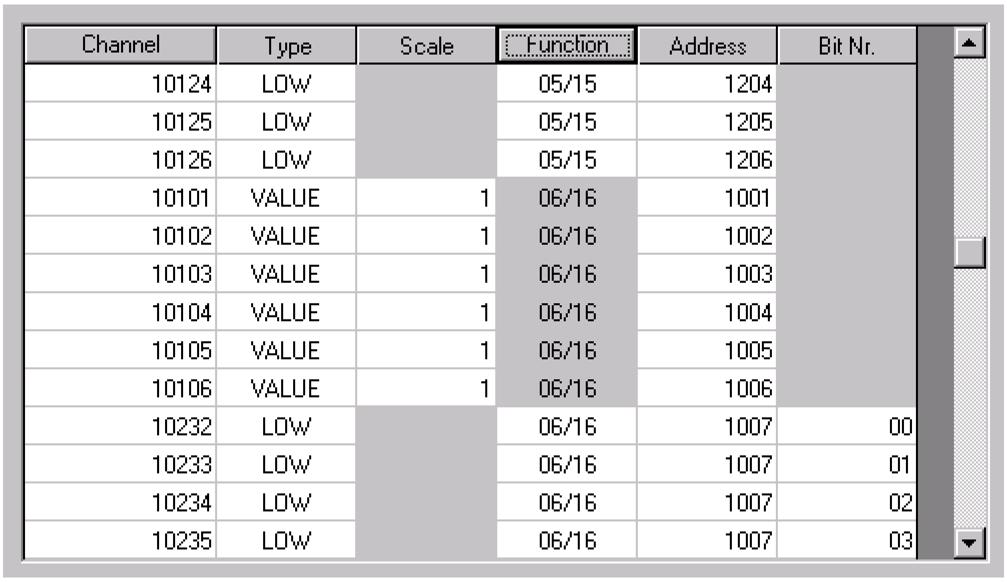
	01	address - 1
	02	address - 10001
	03	address - 40001
	04	address - 30001

Tips: For fast channel configuration, configure the last line complete, press not enter at the last column, go back to first column, give a channel range in (like 10101-10125) and press enter, now multiple rows are created!

See also:

Modbus Slave Plugin
Modbus slave forced connection list

MBS – Forced Connection List



Channel	Type	Scale	Function	Address	Bit Nr.
10124	LOW		05/15	1204	
10125	LOW		05/15	1205	
10126	LOW		05/15	1206	
10101	VALUE	1	06/16	1001	
10102	VALUE	1	06/16	1002	
10103	VALUE	1	06/16	1003	
10104	VALUE	1	06/16	1004	
10105	VALUE	1	06/16	1005	
10106	VALUE	1	06/16	1006	
10232	LOW		06/16	1007	00
10233	LOW		06/16	1007	01
10234	LOW		06/16	1007	02
10235	LOW		06/16	1007	03

Column	Options or Values
Channel	Channel to send the data; type in an existing digital/analog channel number, or get here from the channel's configuration form (**status/value link**)
Type	Value or one of the status bits (only "Low" is supported yet)
Scale	Scale value after received
Function	Function number, like 05/15 or 06/16
Address	Address value
Bit Nr.	bit number, status only

A modbus master send channel information to the slave. (=Input List)

By pressing on column "Channel" a sort action is performed, first time ascending second descending.

Return Button: Return from gateway, only available when getting there from a channel configuration (status/value link)

Addres field of actual message depends on Addres Mode, MBS - General Settings

<table>
<tr><th>Option</th><th>Conversion</th></tr>
<tr><td>"Mapping / No Offset"</td><td>straight copy of address</td></tr>
<tr><td>"Mapping / Offset - 1"</td><td>address - 1</td></tr>
<tr><td>"Modicon Style"</td><td><table><tr><th>Function</th><th>Conversion</th></tr><tr><td>05/15</td><td>address - 1</td></tr><tr><td>06/16</td><td>address - 40001</td></tr></table></td></tr>
</table>

Tips: For fast channel configuration, configure the last line complete, press not enter at the last column, go back to first column, give a channel range in (like 10101-10125) and press enter, now multiple rows are created!

See also:

Modbus Slave Plugin
Modbus slave read connection list

MBS - Diagnostics

Diagnostics are used to report Failures or important statusses
the modbus slave plugin has the following diagnostics

- **MBS: RD(%RD32) ComPort Error**

  Diagnostic is generated when the comport can not be used, while active the plugin will try to open the comport.

  Solution: Configure an unused comport; Close application using comport or attach the USB device

(i) Only triggered on server
(i) Make sure you put the usb cable in the right usb connector, each connector has it's own Comport

- **MBS: RD(%RD32) Error of Receive of Master**

  Diagnostic is generated when runtime receives no reply from Modbus slave device

  Solution: Check Modbus master if it generates messages correctly

- **MBS: RD(%RD32) Answer Not Transmitted**

  Diagnostic is generated when modbus slave receives data before an Answer can be made/send

  Problem timing issue, *Solution*: change timing on Master/Slave side

- **MBS: RD(%RD32) No Communication**

  Diagnostic is generated when Modbus slave receives no data for a "Master disconnected timeout" time

  Solution: check if all wires are correct from Modbus Master to Modbus slave

- **MBS: RD(%RD32) No Master Query**

  Diagnostic is generated when Modbus slave receives no requests for it's Slave Number for a "Master disconnected timeout" time

  Solution: check in Modbus master if data is retrieved from configured Slave number, or change Modbus slave Slave number

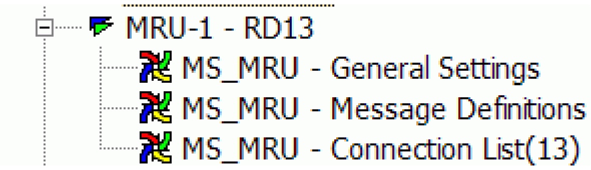
Legend Tag	Replaced by
(RD%32)	Remote data number eq. 31

  See also:

  Modbus Slave Plugin
  Modbus slave general settings

MRU – Plugin

After selecting 'Remote Data' and MRU – RD xx



MRU-1 - RD13

MS_MRU - General Settings

MS_MRU - Message Definitions

MS_MRU - Connection List(13)

There are several items of this remote data:

1. MRU - General Settings
2. MRU - Connection List
3. MRU - Message List

MRU Protocol only receives binair messages

This communication protocol is associated with the 3DM-GX1™ Gyro Enhanced Orientation Sensor delivered by MicroStrain.

Example of message:

Send Gyro-Stabilized Euler Angles & Accel & Rate Vector

Function: The 3DM-GX1™ will transmit the gyro-stabilized Euler Angles and the Instantaneous Acceleration Vector and the drift compensated Angular Rate vector.

Number	**Description**
Command Byte:	0x31
Command Data:	None
Response:	23 bytes defined as follows
Byte 1	Header byte = 0x31
Byte 2	Roll MSB
Byte 3	Roll LSB
Byte 4	Pitch MSB
Byte 5	Pitch LSB
Byte 6	Yaw MSB
Byte 7	Yaw LSB
Byte 8	Accel_X MSB
Byte 9	Accel_X LSB
Byte 10	Accel_Y MSB
Byte 11	Accel_Y LSB
Byte 12	Accel_Z MSB
Byte 13	Accel_Z LSB

Byte 14	CompAngRate_X MSB
Byte 15	CompAngRate_X LSB
Byte 16	CompAngRate_Y MSB
Byte 17	CompAngRate_Y LSB
Byte 18	CompAngRate_Z MSB
Byte 19	CompAngRate_Z LSB
Byte 20	TimerTicks MSB
Byte 21	TimerTicks LSB
Byte 22	Checksum MSB
Byte 23	Checksum LSB

Euler

This is the set of three Euler angles (Pitch, Roll, and Yaw) which describe the orientation of the 3DM-GX1™ with respect to the fixed earth. These angles are calculated according to the “ZYX” or “Aircraft” coordinate system. Users should be aware that there are other valid formulations of Euler Angles that will yield different results. The earth fixed coordinate system has X pointing North, Y pointing East, and Z pointing down. The Euler quantities are derived from the Accel and MagField vectors, and therefore do not incorporate any gyroscopic stabilization. If the 3DM-GX1™ is exposed to linear accelerations, or magnetic interference, artifacts will be present. The Roll and Yaw angles have a range of –32768 to +32767 representing –180 to +180 degrees. The Pitch angle has a range of –16384 to +16383 representing –90 to +90 degrees. To obtain angles in units of degrees, the integer outputs should multiplied by the scaled factor (360/65536).

The user should be aware that the Euler angle formulation in general contains a mathematical singularity at Pitch = +90 or –90 degrees. In practice, poor numerical results will be present if the Pitch angle exceeds +/-70 degrees. In applications where the Pitch angle cannot be guaranteed to exceed these values, it is recommended that the orientation matrix output be utilized instead.

SMC2

The Ship Data is received at a message frequency of 100Hz +/- 1Hz, and at a timing interval (first byte of successive messages) of 10ms +/-1 ms. This ensures a regular and consistent data input to the system.

The serial messages are sent at 38400 bps, 8 data bits, 1 start bit, 1 stop bit, even Parity. (total 11 bits per character).
The interface will be receive-only. No software or hardware handshaking will be employed. Messages will be checked on receipt for parity and credible data content.

Message Segment	Byte Description	Hex	Remark
Header (32 Bits)	SOH	0x01	Start of Header Byte
-	Message Length	0x0D	Remaining No. of Bytes to Follow
-	Message Type	0x00	Message Code
-	EOH	0x1E	End of Header
1 - Data	Pitch Byte 1	LSB	32 Bits representing Ship’s (absolute) Pitch as a signed integer +ive Pitch = Bow up; -ive Pitch = Bow down. See Note 1 for precision.

-	Pitch Byte 2	-	-
-	Pitch Byte 3	-	-
-	Pitch Byte 4	MSB	-
2 - Data	Roll Byte 1	LSB	32 Bits representing Ship’s (absolute) Roll as a signed integer. +ive Roll = Port up; -ive Roll = Port down. See Note 1 for precision.
-	Roll Byte 2	-	-
-	Roll Byte 3	-	-
-	Roll Byte 4	MSB	-

3 - Data	Pitch Invalid	-	Invalidity byte flag:
			0x00 = Valid;
			0x01-0xFF = Invalid;

4 - Data	Roll Invalid	-	-
Footer	EOM	0x04	End of Message

NOTES
Note 1 : Pitch/Roll Value = 360/(232).
0 (0000000000 Hex) = 0 degrees
1 (0000000001 Hex) = +0.00000008382 Degrees
-1 (0FFFFFFFF Hex)= -0.00000008382 Degrees

HPR 400

Header Information

Index	Content	Size
000	Start character	BYTE
001	Block length N	WORD_16
003	Message type	BYTE
004	Destination	BYTE
005	Data Block	with N bytes
N+5	Sumcheck	WORD_16
N+7	Stop character	BYTE

Start character The start character is 55 hex.

Block length The block length defines the length of the data block.

Message type The message type defines the message transmitted. It is a number between 1 and 255.

Destination The destination defines the device to which this telegram is transferred. It is not in use, and it is always set to 0.

Data block The data block contains the message itself.
The length N depends on the Message type. The data block for the different message types are explained in the next chapters.

Sumcheck The sumcheck is the 16 bit sum of all bytes in the telegram, except the sumcheck itself and the stop character. The sum is calculated by byte+byte addition.

Stop character The stop character is equal to 0AAH.

Transponder position data
The position message telegram contains SSBL transponder position data and sensor data related to the position measurement. It is transmitted each time a new position is calculated.

Data Block of Message 1

Nr	Block content	Size
1	Tp_index	WORD_16
2	Operation_mode	BYTE
3	Sync_mode	BYTE
4	Tp_type	BYTE
5	Tp_operation	BYTE
6	Pos_data_form	BYTE
7	Reply_status	BYTE
8	Filt_X_pos	REAL
9	Filt_Y_pos	REAL
10	Filt_Z_pos	REAL
11	X_pos	REAL
12	Y_pos	REAL
13	Z_pos	REAL
14	Slant_range	REAL
15	P_course	REAL
16	P_roll	REAL
17	P_pitch	REAL
18	Td_beam	BYTE
19	Td_type	BYTE
20	Td_num	WORD_16
21	Diagnostic	WORD_16
22	Stand_dev	REAL
23	Instr_data (*)	REAL

LBL position

The LBL position telegram contains a position relative to the origin of the Tp array.
The position is of the vessel or of another object.
The telegram is transmitted each time a new position is calculated.
If the Transponder array is north oriented, the coordinates are relative to true North, else they are relative to local north.

Data Block of Message 2

Nr	Block content	Size
1	Sequence_number	WORD_16
2+3	Time_header (7)*	BYTE
4	Interrogation_age	WORD_16
5	Tp_array	BYTE
6	Td_num	BYTE
7	Pos_east	REAL_64
8	Pos_north	REAL_64
9	Depth	REAL
10	Hor_err_ellipse_direction	REAL
11	Hor_err_ellipse_major	REAL
12	Hor_err_ellipse_minor	REAL
13	Z_standard_deviation	REAL
14	Pos_type	BYTE
15	Pos_status	BYTE
16	P_course	REAL
17	P_roll	REAL
18	P_pitch	REAL
19	Diagnostic	WORD_16

*splitted into date and time

LBL Ranges

The LBL_ranges message contains raw measured ranges to the transponders, and VRU and compass data.
This Message is transmitted just after the Message 2 (LBL position).
The two messages have the same sequence number.

Data Block of Message 4

Nr	Block content	Size
1	Sequence_number	WORD_16
2..9	Range_age (8)	BYTE
10	Tp_array	BYTE
11	Td_num	BYTE
12	Operation_mode	BYTE
13	Sync_mode	BYTE
14	Pos_type	BYTE
15..22	Reply_status (8)	BYTE

23..30	Range (8)	BYTE
31	P_course	REAL
32	P_roll	REAL
33	P_pitch	REAL
34	Diagnostic	WORD_16

EM3000

The EM3000 format consists of a fixed length message using single byte unsigned, 2-byte unsigned and 2-byte two complement integer data elements. For the 2-byte elements, the least significant byte is transmitted first.
Data sent:
Roll
Pitch
Heave
Heading (SMC output 0)

Nr - Element	Scaling	Format	Bytes	Value
1 - Status byte	-	Unsigned	1	-
Header	-	Unsigned	1	90 Hex
2 - Roll	0.01 degrees	Integer	2	-17999 to 17999
3 - Pitch	0.01 degrees	Integer	2	-17999 to 17999
4 - Heave	1 cm	Integer	2	-999 to 999
5 - Heading	0.01 degrees	Unsigned	2	SMC output 0

Roll is positive with port side up. Pitch is positive with bow up. Heave is positive up.
The status byte information:

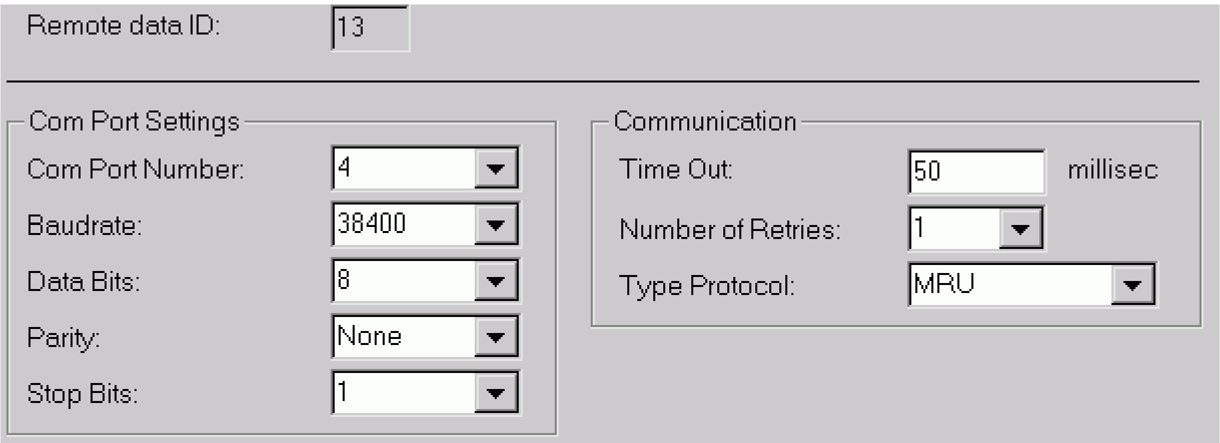
Value	Interpretation
90 Hex	Normal mode
91 Hex	Unsettled mode

Shortcuts

Icon:

Place: Remote Data\

MRU – General Settings



Remote data ID: 13

Com Port Settings

Com Port Number: 4

Baudrate: 38400

Data Bits: 8

Parity: None

Stop Bits: 1

Communication

Time Out: 50 millisec

Number of Retries: 1

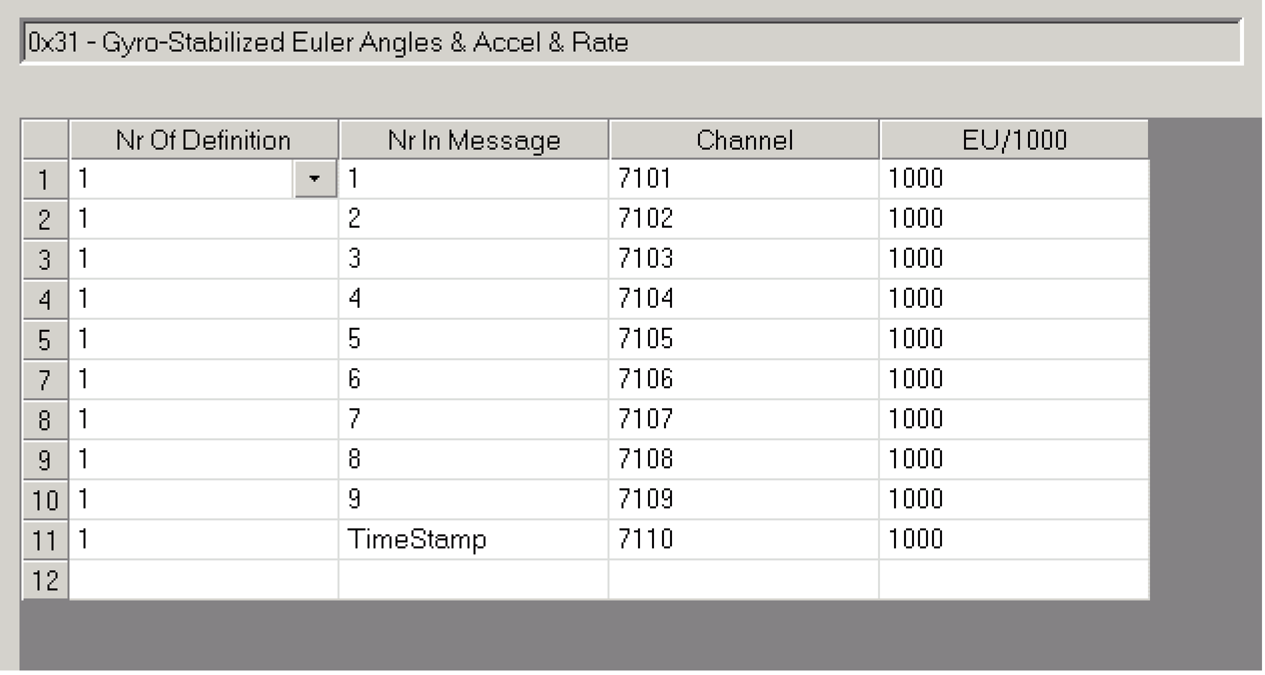
Type Protocol: MRU

Setting	Description	Default
Com Port	Comport to use (1-16)	1
Baudrate	Communication Speed (1200, 2400, 4800, 9600, 19200, 38400, 57600, 115200)	38400
Data Bits	Number of Data bits (7, 8)	8
Parity	Communication Parity None, Odd, Even	None
Stop Bits	Number of Stop Bits (0, 1, 2)	1
Time Out	Minimal Time between two messages before reporting an error (50-100000 in milli-seconds	1000
Number Of Retries	amount of retries before reporting an error (0..7)	1
Type Protocol	MRU, SMC2, HPR3000 or EM3000	MRU

See also:

MRU Plugin

MRU – Connection List



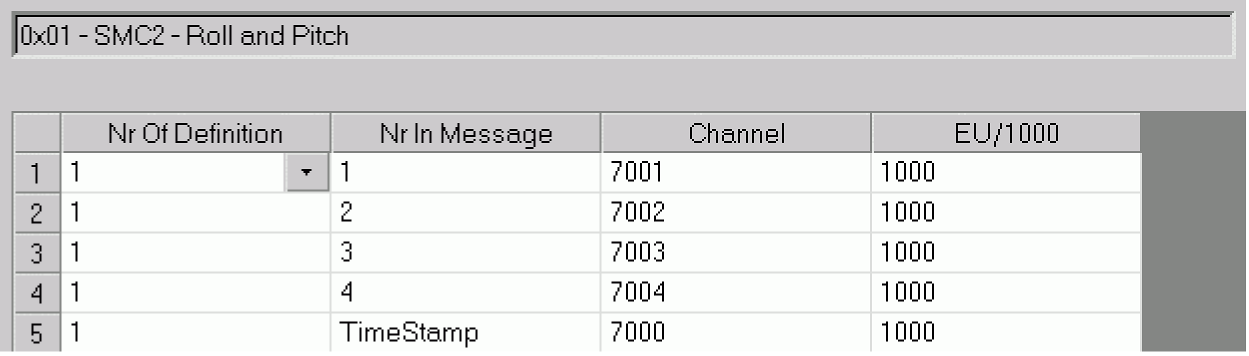
0x31 - Gyro-Stabilized Euler Angles & Accel & Rate

	Nr Of Definition	Nr In Message	Channel	EU/1000
1	1	1	7101	1000
2	1	2	7102	1000
3	1	3	7103	1000
4	1	4	7104	1000
5	1	5	7105	1000
7	1	6	7106	1000
8	1	7	7107	1000
9	1	8	7108	1000
10	1	9	7109	1000
11	1	TimeStamp	7110	1000
12				

From the current selected row message definition is shown.

Field	Description
	Number, row number
Nr of definiton	Number, see Message List
Nr In Message	values are separated by commas, this number gives position in the message
Channel	Channel to retrieve Value from (Analog in) eq. Leaving field empty, this will delete entry
EU/1000	Channel Value is stored as value*1000

Example of SMC2



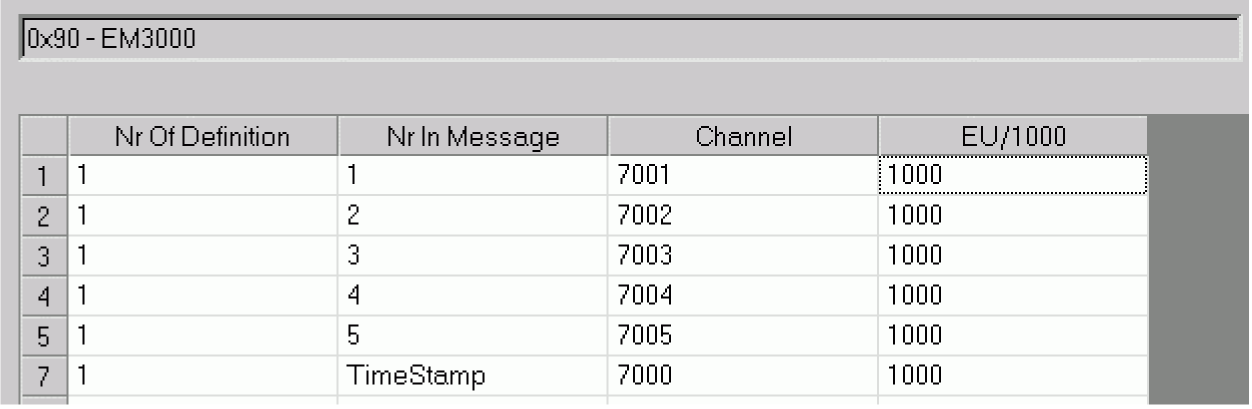
0x01 - SMC2 - Roll and Pitch

	Nr Of Definition	Nr In Message	Channel	EU/1000
1	1	1	7001	1000
2	1	2	7002	1000
3	1	3	7003	1000
4	1	4	7004	1000
5	1	TimeStamp	7000	1000

For HPR-300

Nr Of Definition	Nr in Message
1	1 - 23 + Time Stamp
2	1 - 19 + Time Stamp
4	1 - 34 + Time Stamp

Example of EM3000



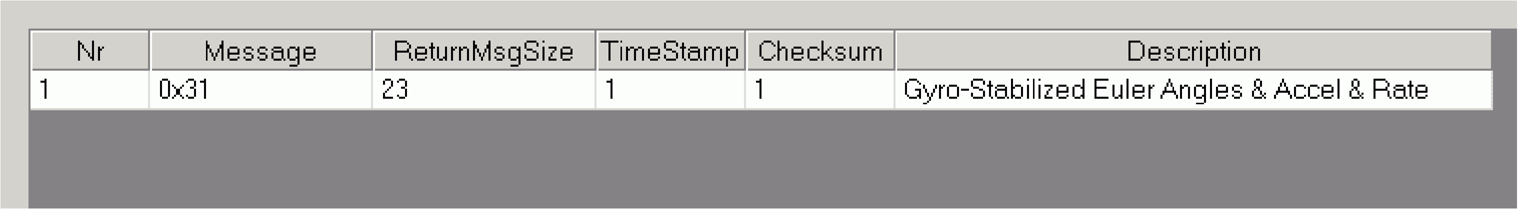
0x90 - EM3000

	Nr Of Definition	Nr In Message	Channel	EU/1000
1	1	1	7001	1000
2	1	2	7002	1000
3	1	3	7003	1000
4	1	4	7004	1000
5	1	5	7005	1000
7	1	TimeStamp	7000	1000

See also:

MRU Plugin

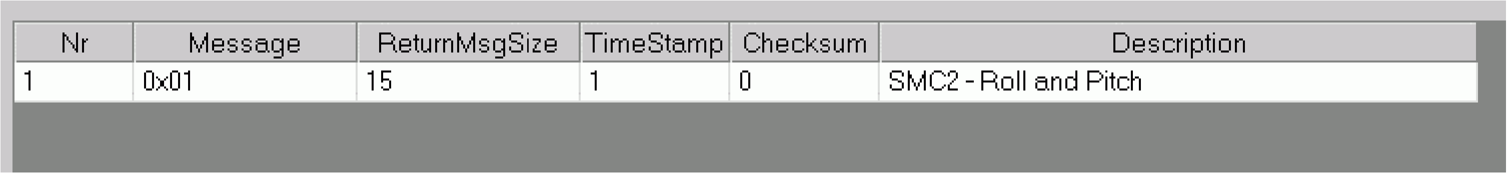
MRU – Messages List



Nr	Message	ReturnMsgSize	TimeStamp	Checksum	Description
1	0x31	23	1	1	Gyro-Stabilized Euler Angles & Accel & Rate

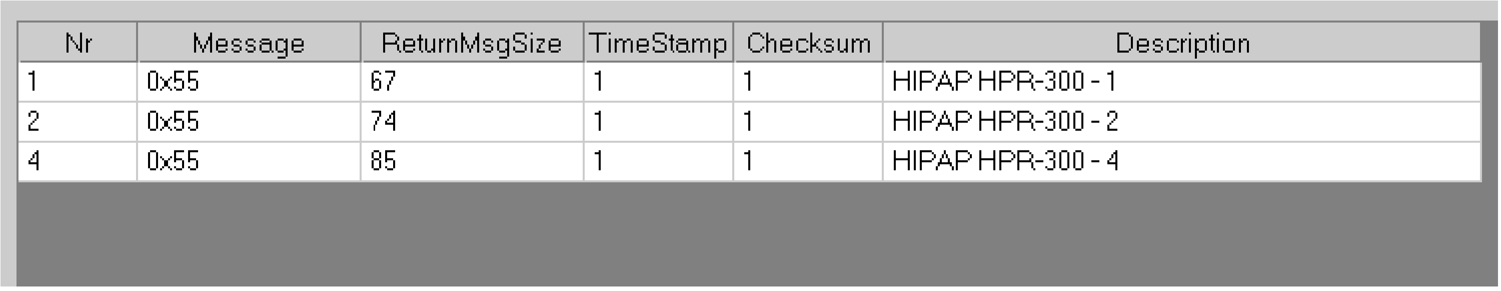
Field	Description
Nr	row index of this grid, this number will be used in connection list
Message	indentifier of message which will be received and handle by this plugin
Return Msg Size	number of bytes of the message
TimeStamp	message contains timestamp or not (1 or 0)
Checksum	message contains checksum or not (1 or 0)
Description	commentary on message for documentation only

Example of SMC2:



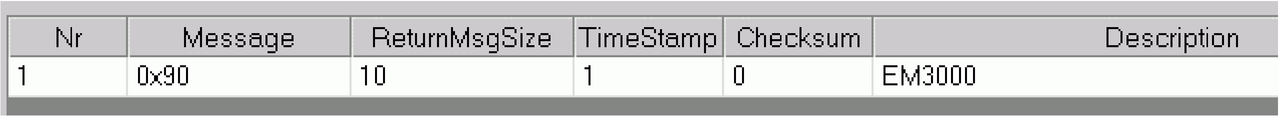
Nr	Message	ReturnMsgSize	TimeStamp	Checksum	Description
1	0x01	15	1	0	SMC2 - Roll and Pitch

Example of HPR 300:



Nr	Message	ReturnMsgSize	TimeStamp	Checksum	Description
1	0x55	67	1	1	HIPAP HPR-300 - 1
2	0x55	74	1	1	HIPAP HPR-300 - 2
4	0x55	85	1	1	HIPAP HPR-300 - 4

Example of EM3000:



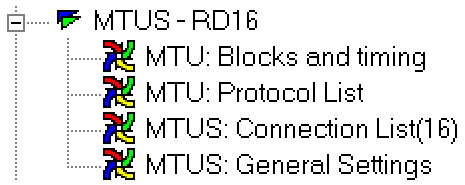
Nr	Message	ReturnMsgSize	TimeStamp	Checksum	Description
1	0x90	10	1	0	EM3000

See also:

MRU Plugin

MTUS – Plugin

After selecting ‘Remote Data’ and MTUS – RD xx



MTUS - RD16
- MTU: Blocks and timing
- MTU: Protocol List
- MTUS: Connection List(16)
- MTUS: General Settings

There are several items of this remote data:

1. MTU – Blocks and Timing
2. MTU – Protocol List
3. MTUS - General Settings
4. MTUS - Connection List

MTU stands for Motors- and Turbines-Union.

S for Slave.

Communication is Full Duplex over a serial line both systems are sending and receiving data in cycles.

Send cycle is independent of receive cycle.

Cycles consist of a number of blocks being transmitted/received.

Block is a Message send over the serial line in the Motorola storage format (MSB … LSB)

Block Type:

- 03 Analog measurement value block
- 02 Binary measurement value block
- 01 Request block

Request Block

Start_1	Start_2	Block_No	Block Type	Frame Size	Checksum	End_1	End_2
(1 byte)	(1 byte)	(2 bytes)	(2 bytes)	(4 bytes)	(4 bytes)	(1 byte)	(1 byte)

Example 1: Request Block

FA Start 1

F5 Start 2

00 00 Block_No: Request block

00 01 Block Type: Request block

00 00 00 10 Frame size 16 (=size of entire message, no data)

00 00 02 00 Checksum

AF End 1

5F End 2

Analog Measurement Value Block

Start_1	Start_2	Block_No	Block Type	Frame Size	Meas.Value(s)	Checksum	End_1	End_2
(1 byte)	(1 byte)	(2 bytes)	(2 bytes)	(4 bytes)	(m*4 bytes)	(4 bytes)	(1 byte)	(1 byte)

m = number of measurement values

Example 2: Analog Measurement Value Block

FA Start 1

F5 Start 2

00 01 Block_No: 1

00 03 Block Type: Analog values

00 00 00 20 Frame size 32 (=4 Analog Values)

00 06 B1 27 Value 438567

00 0E D6 7A Value 972410

00 00 C4 A4 Value 50340

00 00 3A B8 Value 15032

00 00 06 A9 Checksum

AF End 1

5F End 2

Digital Measurement Value Block

Start_1	Start_2	Block_No	Block Type	Frame Size	Meas.Value	Checksum	End_1	End_2
(1 byte)	(1 byte)	(2 bytes)	(2 bytes)	(4 bytes)	(m*1 byte)	(4 bytes)	(1 byte)	(1 byte)

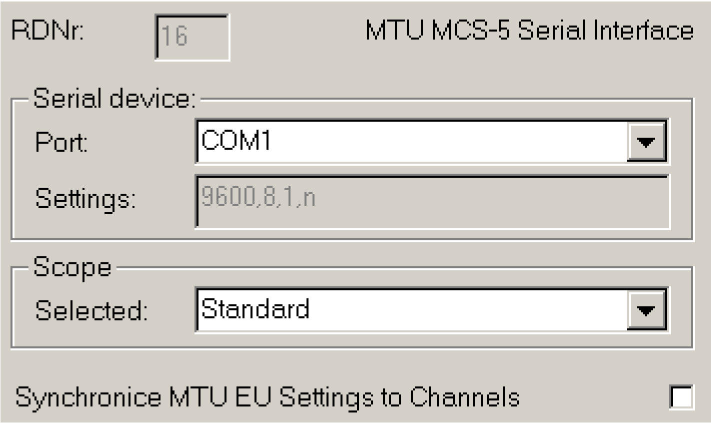
m = number of measurement values

Shortcuts

Icon:

Place: Remote Data\

MTUS – General Settings



RDNr: 16

MTU MCS-5 Serial Interface

Serial device:

Port: COM1

Settings: 9600,8,1,n

Scope

Selected: Standard

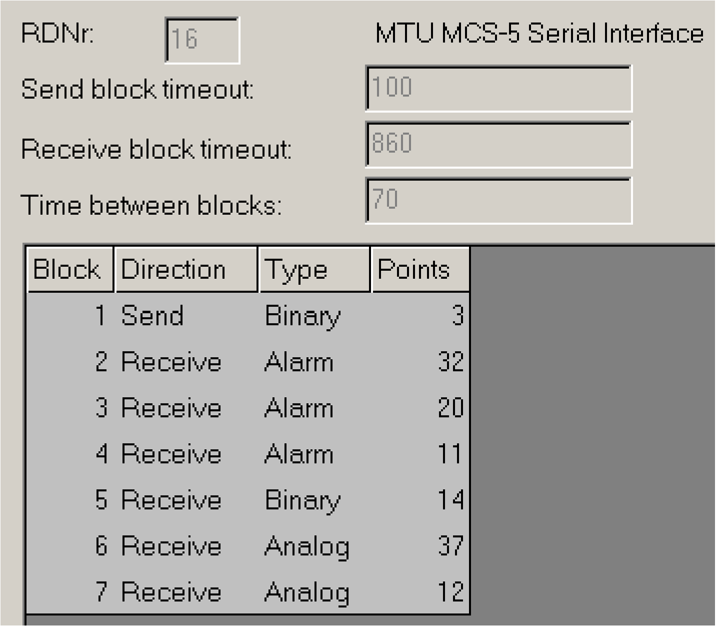
Synchronice MTU EU Settings to Channels □

Setting	Description
Com Port	Communication port which is used for sending the output *No other plug-in must use the same COM port* (only if plugin is on running on server)
Settings	Fix settings, Baudrate = 9600, Databits =8, Stopbits = 1,Parity = None
Selected	Classifiable or Standard (default: Classifiable)
Synchronize MTU EU Settings to channels	Synchronize channels with settings from Protocol list (Synchronize button in connection list has to be used to update channels): • Engineering unit range low • Engineering unit range high • Engineering unit • Description

See also:

MTUS Plugin

MTU – Blocks and Timing



RDNr: 16

MTU MCS-5 Serial Interface

Send block timeout: 100

Receive block timeout: 860

Time between blocks: 70

Block	Direction	Type	Points
1	Send	Binary	3
2	Receive	Alarm	32
3	Receive	Alarm	20
4	Receive	Alarm	11
5	Receive	Binary	14
6	Receive	Analog	37
7	Receive	Analog	12

- Block	Block Number, 1..8
- Direction	Direction of block Send / Receive
- Type	Type of Block

Send block timeout: Time between start of send cycle

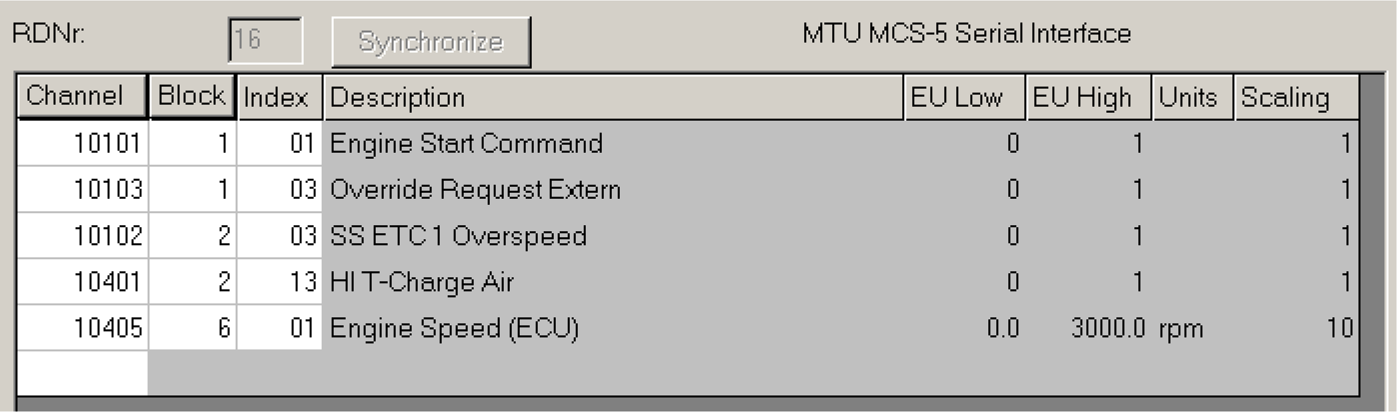
Receive block timeout: Time between start of receive cycle

Time between blocks: Time between blocks (Send and/or receiving)

See also:

MTUS Plugin

MTUS – Connection List



Channel	Block	Index	Description	EU Low	EU High	Units	Scaling
10101	1	01	Engine Start Command	0	1		1
10103	1	03	Override Request Extern	0	1		1
10102	2	03	SS ETC 1 Overspeed	0	1		1
10401	2	13	HI T-Charge Air	0	1		1
10405	6	01	Engine Speed (ECU)	0.0	3000.0	rpm	10

Setting	Description
Channel	Channel to receive/send the data Type in an existing digital/analog channel number, or get here from the channel's configuration form (**status/value link**)
Block	Block where value is stored in
Index	Index in block where value is stored
Description	Description of value
EU Low	Engineering unit range low
EU High	Engineering unit range high
Units	Engineering units
Scaling	Scaling factor between send/received value and EU Value

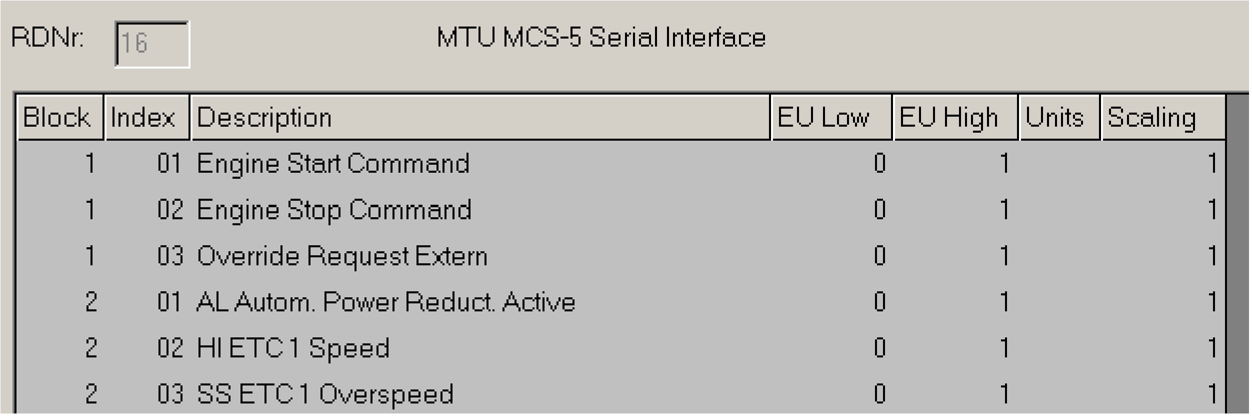
By pressing on column “Channel” or “Block” a sort action is performed, first time ascending second descending.

Return Button: Return from gateway, only available when getting there from a channel configuration (status/value link)

See also:

MTUS Plugin

MTU – Protocol List



RDNr: 16

MTU MCS-5 Serial Interface

Block	Index	Description	EU Low	EU High	Units	Scaling
1	01	Engine Start Command	0	1		1
1	02	Engine Stop Command	0	1		1
1	03	Override Request Extern	0	1		1
2	01	AL Autom. Power Reduct. Active	0	1		1
2	02	HI ETC 1 Speed	0	1		1
2	03	SS ETC 1 Overspeed	0	1		1

Setting	Description
Block	Block where value is stored in
Index	Index in block where value is stored
Description	Description of value
EU Low	Engineering unit range low
EU High	Engineering unit range high
Units	Engineering units
Scaling	Scaling factor between send/received value and EU Value

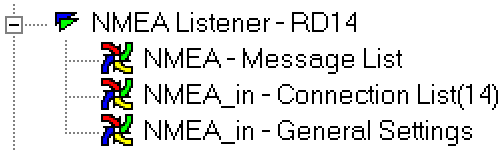
It is not possible to change standard protocol list.

See also:

MTUS Plugin

NMEA – Listener Plugin

After selecting ‘Plugins’ and NMEA Listener – RD xx



NMEA Listener - RD14
NMEA - Message List
NMEA_in - Connection List(14)
NMEA_in - General Settings

There are several items of this plugin:

1. NMEA in - General Settings

2. NMEA in - Connection List

3. NMEA - Message List

Protocol Description

Based on

- Standard IEC 1162-1 (DIS 80/105/DIS) based on

- NMEA 0183 version 2.01

Sentence going over the Serial line:

$aaccc,c---c*hh<CR><LF>

Sentence description

$ Start of sentence, Start delimiter
HEX: 24

Aa Address field, Talker ID
P means custom message Mnemonic can take up to 7 characters

Ccc fields Address field, Sentence formatter mnemonic:
Identifying the data type and string format of the successive fields.
Mnemonics will be used as far as possible to facilitate read-outs by users.

, Field Delimiter
Starts each field except Address and checksum fields.
HEX: 2C

c---c Data sentence block:
Follows address filed and is a series of data fields containing all of the data to be transmitted. Data field sequence is fixed and identified by third and subsequent characters of the address field (Sentence formatter) Data fields may be of variable length and are preceded by delimiters “,”.

• Checksum Delimiter (Optional)
Follows last data field of the sentence it indicates that the following two alphanumeric characters show the HEX value of the checksum.
HEX: 2A

Hh Checksum field (Optional)
The absolute value calculated by exclusive-OR’ing the eight data bits of each character in the sentence, between, but excluding “$” and “*”. The hexadecimal value of the most significant and least significant four bits of the result is converted to two ASCII characters (0-9, A-F) for transmission. The most

significant character is transmitted first. The checksum field is optional, except when indicated as mandatory.

<CR><LF> End of Sentence:
Sentence terminating delimiter
HEX: 0D 0A

Field types (Supported)

llll.ll Latitude
Degrees, Minutes and Decimal – two fixed digits of degrees, two fixed digits of minutes and a variable number of digits for decimal fraction of minutes. Leading zeros always included for degrees a minutes to maintain fixed length. The decimal point and associated decimal fraction are optional if full resolution is not required.

yyyy.yy Longitude
See Latitude

hhmmss.ss Time
Hours/Minutes/Seconds and decimal - two fixed digits of hours; two fixed digits of minutes; two fixed digits of seconds and a variable number of digits for decimal fraction of seconds. Leading zeros always included for hours; minutes and seconds to maintain fixed length. The decimal point and associated decimal fraction are optional if full resolution is not required.

b Binair Value 0 or 1

bbbb Binair Packed Value, do not use more than 32 b's (32 bits value)

h Hexadecimal Value 0..F

hhhh Hex Packed Value, do not use more than 8 h's (32 bits value)

x Value with whole numbers

x.x Value with fraction

LLLL.LLLLL Latitude high resolution

YYYYY.YYYYY Longitude high resolution

cx extended value splitted into two values,
example: T1, value1 = T, value2 = 1

xc extended value splitted into two values,
example: 1T, value1 = 1, value2 = T

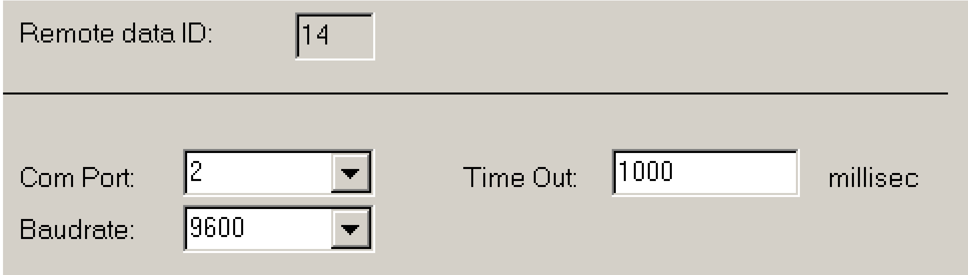
cxi extended multiple values with input-buffer splitted into two values
example: B1, B2, B5, value1 = B / value2 = 1 or 2 or 5

xci extended multiple values with input-buffer splitted into two values
example: 1B, 2B, 5B, value1 = 1 or 2 or 5 / value2 = B

xi extended multiple value with input-buffer splitted into one value
example 1, 2, 5, value1 = 1 or 2 or 5

Icon:
Place: Plugins\

NMEA in – General Settings



Remote data ID: 14

Com Port: 2

Baudrate: 9600

Time Out: 1000 millisec

Com Port Communication port which is used for sending the output. No other plug-in must have the same COM port (only if plugin is on running on server)

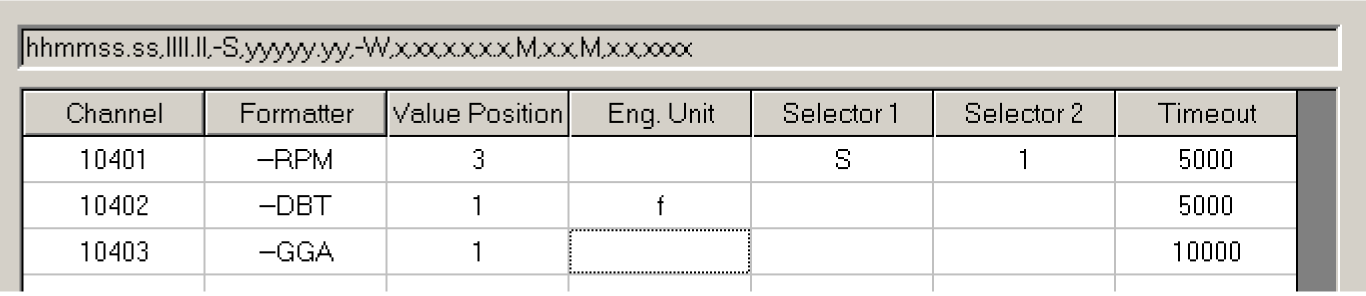
Baudrate Baudrate of Com Port, 1200, 2400, 4800, 9600 or 19200 (default: 19200).

Time Out 200-300000 millisec Timeout between NMEA Messages (default: 2000)

See also:

NMEA Listener Plugin

NMEA in – Connection List



hhmmss.ss,llll.ll,-S,yyyyy.yy,-W,x,xx,x.x,x.x,M,x.x,M,x.x,xxxx

Channel	Formatter	Value Position	Eng. Unit	Selector 1	Selector 2	Timeout
10401	–RPM	3		S	1	5000
10402	–DBT	1	f			5000
10403	–GGA	1				10000

Field	Description
Channel	Channel to receive/send the data; type in an existing digital/analog channel number, or get here from the channel's configuration form (**status/value link**)
Formatter	Formatter from configured NMEA Message List
Value Position	Position number in message, like 01,02,03 or 04
Eng. Unit	Channel where to send/receive its value [10101-49668] (Configured Analog in; Digital in; Analog out; Digital out)
Selector 1	Multiple Message identifier 1, See: Special Codes
Selector 2	Multiple Message identifier 2, See: Special Codes
Time Out	Update Time between two Updates of one message, 1000-25500 msec, 0=No Timeout, (if this time is passed a sensor failure on input channel will be triggered when configured)

Return Button: Return from gateway, only available when getting there from a channel configuration (status/value link)

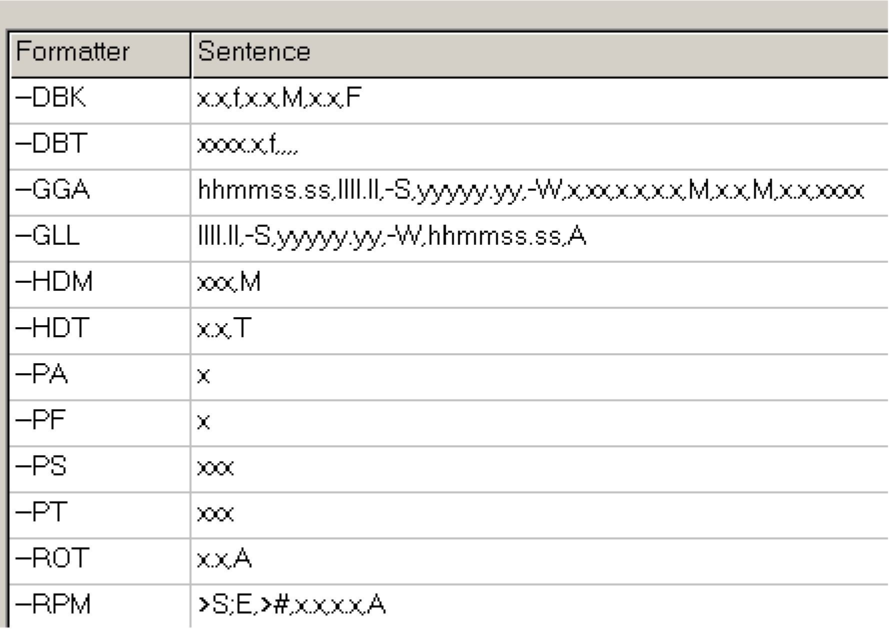
For Special Codes see:

NMEA in Message List

For Character Engineering Unit Code see:

NMEA Listener Plugin

NMEA in – Messages List



Formatter	Sentence
–DBK	x.x,f,x.x,M,x.x,F
–DBT	xxxx.x,f,,,,
–GGA	hhmmss.ss,llll.ll,-S,yyyyy.yy,-W,x,xx,x.x,x.x,M,x.x,M,x.x,xxxx
–GLL	llll.ll,-S,yyyyy.yy,-W,hhmmss.ss,A
–HDM	xxx,M
–HDT	x.x,T
–PA	x
–PF	x
–PS	xxx
–PT	xxx
–ROT	x.x,A
–RPM	>S;E,>#,x.x,x.x,A

Field	Description
Formatter	Formatter NMEA Message List
Sentence	Sentence Description See: Character Engineering Unit Code See: Special Codes

Special codes

>x;y Select from list (x or y) (eq E or W)

># Select Number (0-9) (eq 0 or 8)

-x x means negative value else positive for previous value

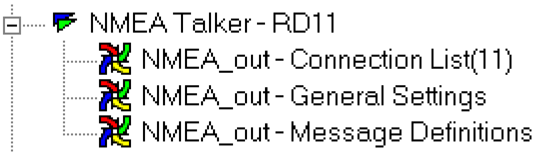
(eq S)

For Character Engineering Unit Code see:

NMEA Listener Plugin

NMEA – Talker Plugin

After selecting ‘Remote Data’ and NMEA Talker – RD xx



NMEA Talker - RD11
NMEA_out - Connection List(11)
NMEA_out - General Settings
NMEA_out - Message Definitions

There are several items of this remote data:

1. NMEA out - General Settings
2. NMEA out - Connection List
3. NMEA out - Message List

Protocol Description

Based on

- Standard IEC 1162-1 (DIS 80/105/DIS) based on

- NMEA 0183 version 2.01

Sentence going over the Serial line:

$aaccc,c---c*hh<CR><LF>

Sentence description

$	Start of sentence, Start delimiter HEX: 24
Aa	Address field, Talker ID P means custom message Mnemonic can take up to 7 characters
Ccc fields	Address field, Sentence formatter mnemonic: Identifying the data type and string format of the successive
,	Field Delimiter Starts each field except Address and checksum fields. HEX: 2C
c---c	Data sentence block: Follows address filed and is a series of data fields containing all of the data to be transmitted. Data field sequence is fixed and identified by third and subsequent characters of the address field (Sentence formatter) Data fields may be of variable length and are preceded by delimiters “,”.
•	Checksum Delimiter (Optional) Follows last data field of the sentence it indicates that the following two alphanumeric characters show the HEX value of the checksum. HEX: 2A
Hh	Checksum field (Optional) The absolute value calculated by exclusive-OR’ing the eight data bits of each character in the sentence, between, but excluding “$” and “*”. The hexadecimal value of the most significant and least significant four bits of the result is converted to two ASCII characters (0-9, A-F) for transmission. The most significant character is transmitted first. The checksum field is optional, except when indicated as mandatory.
<CR><LF>	End of Sentence: Sentence terminating delimiter HEX: 0D 0A

Field types (Supported)

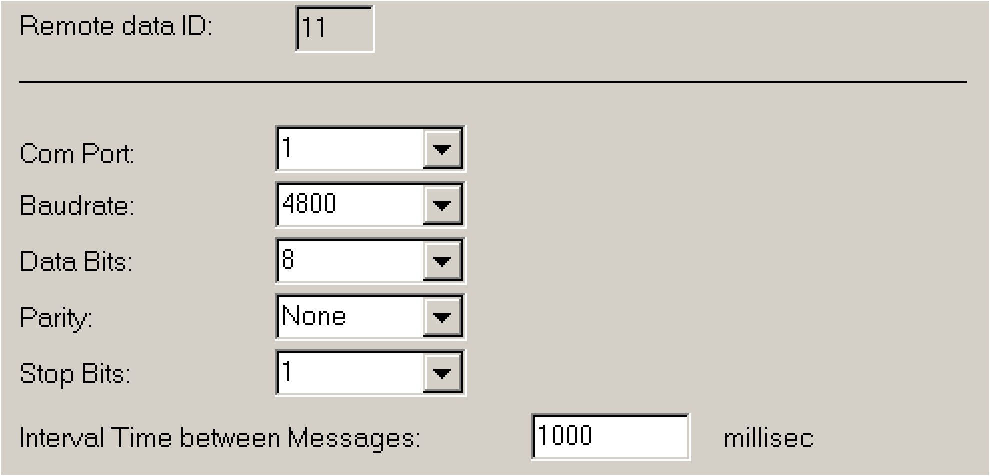
x	Value with whole numbers
x.x	Value with fraction

Shortcuts

Icon:

Place: Remote Data\

NMEA out – General Settings



Remote data ID: 11

Com Port: 1

Baudrate: 4800

Data Bits: 8

Parity: None

Stop Bits: 1

Interval Time between Messages: 1000 millisec

Com Port Communication port which is used for sending the output. No other plug-in must have the same COM port (only if plugin is on running on server)

Baudrate Baudrate of Com Port, 1200, 2400, 4800, 9600 or 19200 (default: 19200).

Data Bits Number of Data bits of Com Port, 7or 8

Parity Kind of Parity of Com Port, like None, Odd or Even

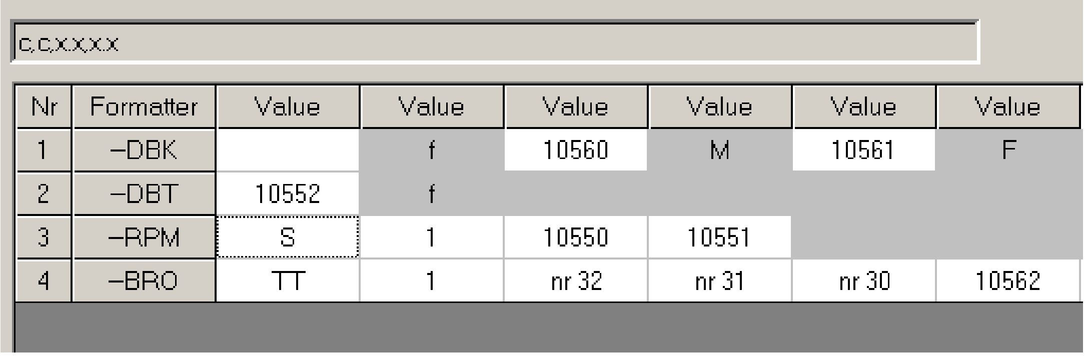
Stop Bits Number of Stop Bits of Com Port, 1 or 2

Interval Time between Messages Minimal Time to wait between two NMEA messages (in milli-seconds)

See also:

NMEA Talker Plugin

NMEA out – Connection List



c,c,x,x,x,x

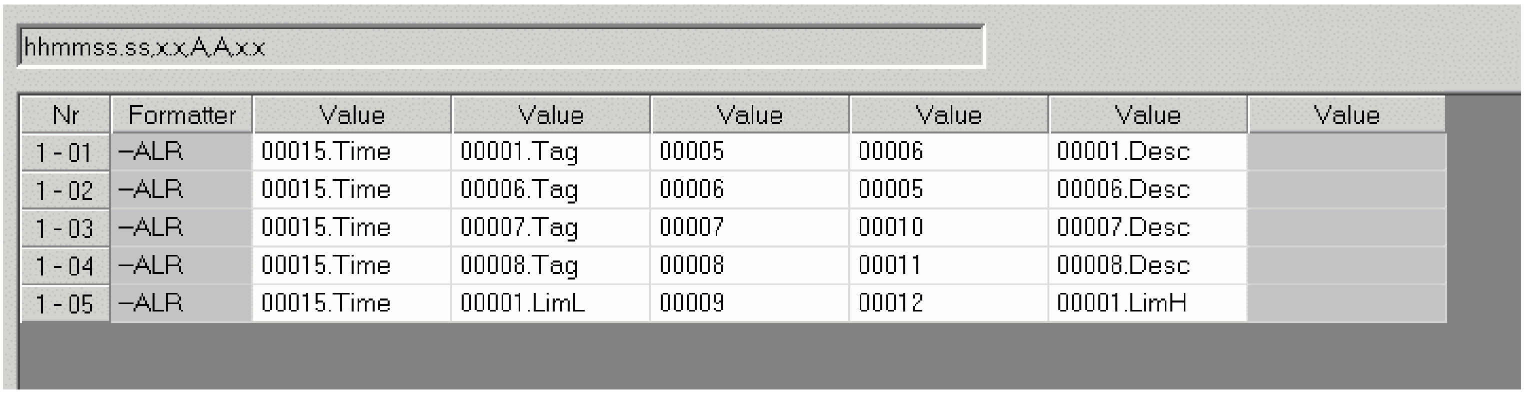
Nr	Formatter	Value	Value	Value	Value	Value	Value
1	–DBK		f	10560	M	10561	F
2	–DBT	10552	f				
3	–RPM	S	1	10550	10551		
4	–BRO	TT	1	nr 32	nr 31	nr 30	10562

From the current selected row message definition is shown.

-Type in Channel Number where value is coming from, like 10560

-Type characters in variable text fields

Field	Description
Formatter	Formatter from configured NMEA Message List
Value (Channel or Text)	Channel to receive/send the data; type in an existing digital/analog channel number, or get here from the channel's configuration form (**status/value link**)



hhmmss.ss,x,x,A,A,x,x

Nr	Formatter	Value	Value	Value	Value	Value	Value
1 - 01	–ALR	00015.Time	00001.Tag	00005	00006	00001.Desc	
1 - 02	–ALR	00015.Time	00006.Tag	00006	00005	00006.Desc	
1 - 03	–ALR	00015.Time	00007.Tag	00007	00010	00007.Desc	
1 - 04	–ALR	00015.Time	00008.Tag	00008	00011	00008.Desc	
1 - 05	–ALR	00015.Time	00001.LimL	00009	00012	00001.LimH	

Example with alarm list

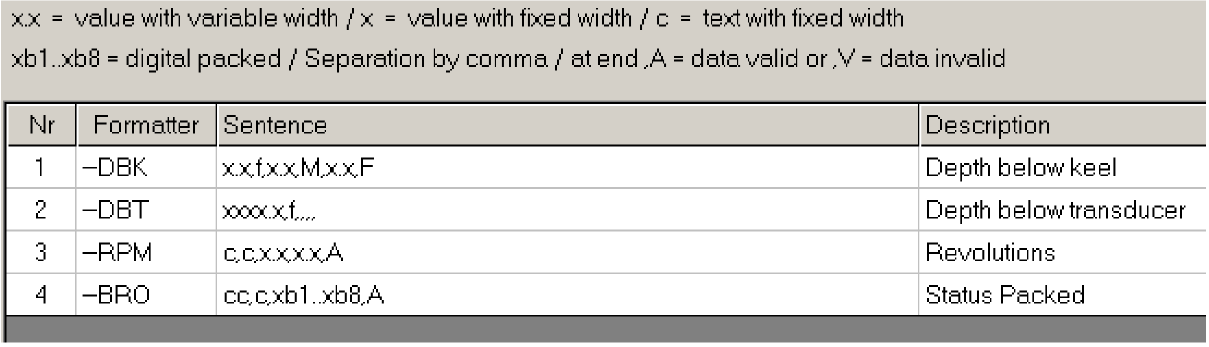
next options are available:
.Desc
.Tag
.LimL
.LimH
.Time
remark: these options are case sensitive

See also:

NMEA Talker Plugin

NMEA out – Messages List

For learning purposes four NMEA messages are default created.



x.x = value with variable width / x = value with fixed width / c = text with fixed width

xb1..xb8 = digital packed / Separation by comma / at end ,A = data valid or ,V = data invalid

Nr	Formatter	Sentence	Description
1	–DBK	x.x,f,x.x,M,x.x,F	Depth below keel
2	–DBT	xxxx.x,f,,,,	Depth below transducer
3	–RPM	c,c,x.x,x.x,A	Revolutions
4	–BRO	cc,c,xb1..xb8,A	Status Packed

Field	Description
Nr	Line number of all messages or message number
Formatter	Formatter NMEA Message List
Sentence	Sentence Description
Description	Information about this message

Special codes

>x**;**y Select from list (x or y) (eq E or W)

># Select Number (0-9) (eq 0 or 8)

-x x means negative value else positive for previous value

(eq S)

Remove the messages if you don't want to use then. After that insert de message(s) you like to use.

A NMEA message exits of formatter (header of message) and sentence which contains channel value(s) and/or characters which are separated by a comma.

Formatter start usually with two - which are known as "don't cares" (Like --DBK).

More about Sentence:

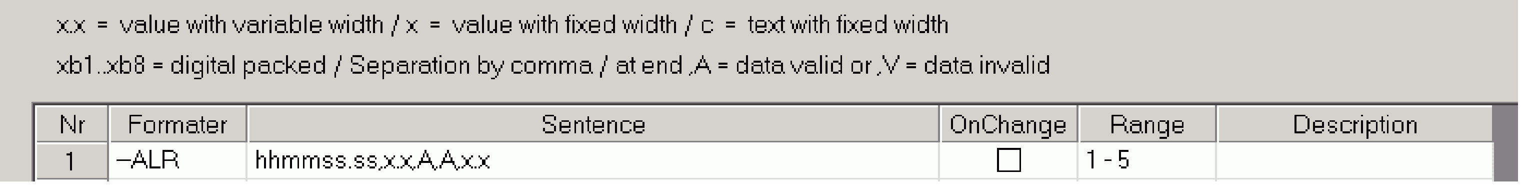
-x.x one channel value, like 10101, variable width

-xxxx.x one channel value, like 10101, fixed width

-cc two variable characters like mA

-xb1..xb8 status packed (VDR) with a maximum of (8*4 =) 32 channel statuses

-M, f, F a fixed character



x.x = value with variable width / x = value with fixed width / c = text with fixed width

xb1..xb8 = digital packed / Separation by comma / at end ,A = data valid or ,V = data invalid

Nr	Formater	Sentence	OnChange	Range	Description
1	–ALR	hhmmss.ss,x.x,A,A,x.x	□	1 - 5	

Example of AlarmList;
Be aware 'A' will be channel status: A = exceeded (alarm/on) and V = not exceeded (normal/off)

See also:

NMEA Talker Plugin

General information OPC

Terms used:

OPC Item	Process value.
OPC group	Collection of OPC Items that are accessed at the same time.
OPC Client	Application that uses process values
OPC Server	Application that supplies access to process values

Technology

OPC is a technology to access Process data. The client Server model is used, so a distributed control network can be established. The communication between the client and server is standardized so interoperability is possible.

OPC clients connect to the OPC server to access the process values, process values can be read and written.

OPC Servers provide Process data retrieved from computer memory; PLCs or other devices, on Process Value changes the clients who registered the OPC Item will get an update.

General information OPC OPC Client

OPC Client

Requirements:

1. OPC Server must be OPC DA 2.05 Compatible.
2. Runs on the Marine PC
3. OPC Core Components run on Local Machine
4. OPC Enumerator runs on Target Machine

Read the Installation Instructions when OPC Client is used for the first time.

For more information reference "OLE for Process Control Data Access Custom Interface Standard Version 2.0 (Release Candidate 5)" April 10, 1998 opcda205_cust.pdf, See http://www.opcfoundation.org/.

Plugin buildup:

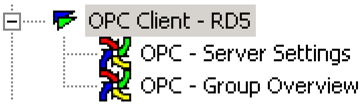
The plugin consists of:

- PAL Configuration part
- IOServer Plugin Runtime

PAL Plugin Configuration:

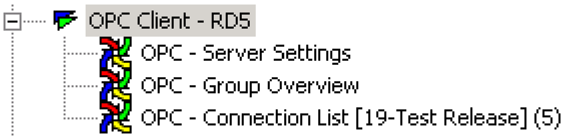
Inserting Plugin:

After selecting 'Remote Data' and OPC - RD xx In field filename enter OPCClient.dll The plugin shows the following Tree items:



OPC Client - RD5
OPC - Server Settings
OPC - Group Overview

After configuring groups additional Tree items are shown:



OPC Client - RD5
OPC - Server Settings
OPC - Group Overview
OPC - Connection List [19-Test Release] (5)

There are several tree items of this plugin:

1. OPC - Server Settings to configure server settings and functionality
2. OPC - Group Overview to assign Goups to OPC Groups that are created on the server
3. OPC - Connection List [xx-yy] to assign channels to the OPC Items

IOServer Plugin Runtime:

Plugin startup:

- retrieves the configuration from the config database

The runtime:

States:

- Connect
  - Connect to OPC Server
  - Creates OPC Groups
  - Creates OPC Group Items
  - Register OPC Item updates (on change)
- Exchange
  - Write values
  - Process changed values
- disconnect
  - Unregister OPC Item updates (on change)
  - Delete OPC Group Items
  - Delete OPC Groups
  - Disconnect to OPC Server
- Reconnect
- Reconnect after opc server shutdown

Common functionality:

- Update diagnostics
- Watchdog to see if Runtime is running at the expected rate

Special functionality:

- Resend All

Plugin shutdown:

Remove configuration.

Diagnostics:

OPC: RD(%RD32) Server disconnected
OPC Server is down; unreachable or being connected to.

OPC: RD(%RD32) Server shutdown
OPC Server is shutdown for maintainance

OPC: RD(%RD32) Server Busy
OPC Server is Busy, not responded in Busy Timeout.

See also:

General information OPC OPC Client
OPC Client Forms Server Settings Group Overview Connection List
Appendix A:Value Conversion
Appendix B:RSLinx OPC Server
Appendix C:Installation

OPC Functionality

OPC Functionality does OPC Related Special Functionality:

- Resend All

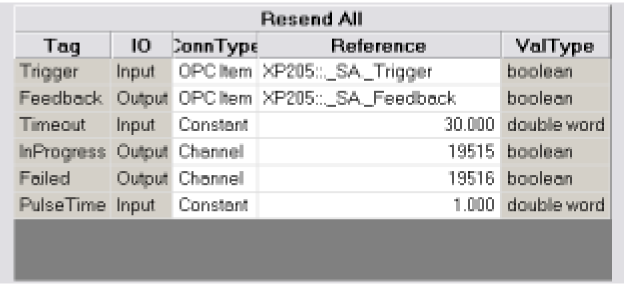
Functionality has Inputs or Outputs that can be:

- (none)
- Channel
- Constant
- OPC Item

Depending on the configuration of the functionality certain IO options are disabled.

Column	Description
Tag	Functionality Tag describing IO
IO	Input or Output
ConnType	(none) Channel Constant OPC Item
Reference	Channel number Constant Value OPC Item
ValType	boolean double word double

Resend All Functionality



Resend All

Tag	IO	ConnType	Reference	ValType
Trigger	Input	OPC Item	XP205::_SA_Trigger	boolean
Feedback	Output	OPC Item	XP205::_SA_Feedback	boolean
Timeout	Input	Constant	30.000	double word
InProgress	Output	Channel	19515	boolean
Failed	Output	Channel	19516	boolean
PulseTime	Input	Constant	1.000	double word

Brief:

After a trigger send all Written data anew to OPC Server and give feedback when completed.

Requirements before functionality works:

- Configure Trigger
- Configure Timeout
- Configure PulseTime

Description:

- After a UP Edge of input 'Trigger' of the IO, input 'Timeout' and 'PulseTime' are latched
- Output 'In Progress' goes HIGH
- All OPC Items in all configured OPC Groups, that are configured as writeable, are written
- When Writing is finished in the Timeout value, Output 'Feedback' goes HIGH, Output are 'In Progress' goes LOW
- When Writing isn't finished in the Timeout value, Output 'Failed' goes HIGH, Output are 'In Progress' goes LOW
- After the 'PulseTime' is expired the 'Feedback' or 'Failed' output goes LOW
- Before a new Trigger is detected the 'Trigger' input needs to be seen LOW, continue on top of the list.

Name	**Input Output**	**Description**	**ConnType**
Trigger	Input	Triggers functionality, UP Edge Triggered	Channel OPC Item
Feedback	Output	High when All Data is written successfully	(none) Channel OPC Item
Timeout	Input	Time out value (ms) between trigger and Failure output	Channel Constant OPC Item
InProgress	Output	High when Functionality is in progress	(none) Channel OPC Item
Failed	Output	High when Failed to send all Write opc items OR Timeout	(none) Channel OPC Item
PulseTime	Input	Timespan (ms) that Outputs are High	Channel Constant OPC Item

See also:

General information OPC OPC Client
OPC Client Forms Server Settings Group Overview Connection List

Installation Instructions

Installing on a Clean Marine PC

The Following steps need to be done:

A. **Install Ship Automation System on Local Machine**
- Follow instructions in Installation Guide

Upgrading on a Marine PC

The Following steps need to be done:

1. **Upgrade Ship Automation System on Local Machine**
   - Follow instructions in Installation/Upgrade Guide

2. **OPC Core Components run on Local Machine**
   - Install by clicking on "OPC Core Components 2.00 Redistributable 1.04.msi"
   - Follow instructions on screen

3. **OPC Enumerator runs on Target Machine**
   - Install by clicking on "OPCEnumInst.exe" - Follow instructions on screen

   - Make sure OPC Enum Service runs in same account as the Ship Automation Software.

4. **Install OPC Server on Target Machine**
   - Follow instructions delivered by OPC Server

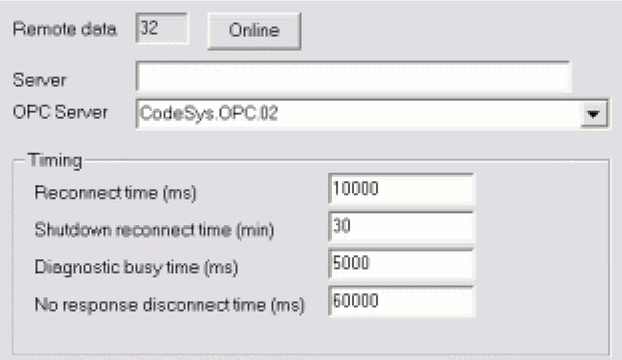
   - Make sure OPC Server runs in same account as the Ship Automation Software.
   - Make sure OPC Server supports OPC DA 2.05

See also:

General information OPC OPC Client
OPC Client Forms Server Settings Group Overview Connection List

OPC client form server settings

The form is used to define the server and client to server connection behaviour.
Also OPC Functionality is defined here.



Remote data 32 Online

Server

OPC Server CodeSys.OPC.02

Timing

Reconnect time (ms) 10000

Shutdown reconnect time (min) 30

Diagnostic busy time (ms) 5000

No response disconnect time (ms) 60000

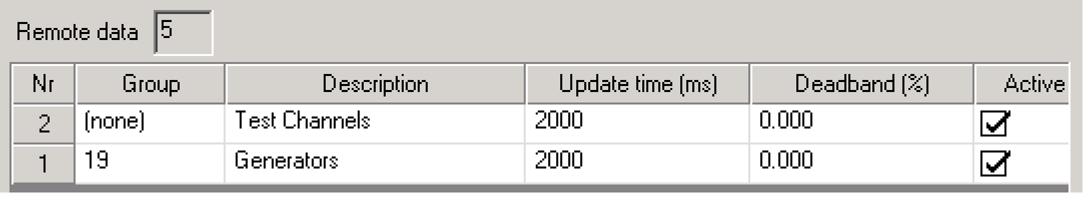
Setting	Description
Server	Machine where OPC Server(s) resides. Empty means local machine. (Needed for OPC Servers that don't accept remote connections)
OPC Server	Name of OPC Server (Dropdown shows some examples)
Reconnect time	When an OPC Server communication failure occurs the Client disconnects, after the reconnect time the OPCServer is connected again.
Shutdown reconnect time	When an OPC Server shuts down for maintainance the Client disconnects, after the shutdown reconnect time the OPCServer is connected again.
Busy time	OPC Client get server state 10 times per second, when OPC Server is busy it will not respond. Here a time can be given so that a Busy diagnostic is triggered.
Busy disconnect time	When an OPC Server is busy the cause can be network failure, here a timeout can be given after which the OPC Client disconnects.

See also:

General information OPC OPC Client
OPC Client Forms Server Settings OPC Functionality Group Overview Connection List

OPC client form group overview

The form is used to add/remove/modify Group assignments to OPC Groups.



Remote data 5

Nr	Group	Description	Update time (ms)	Deadband (%)	Active
2	(none)	Test Channels	2000	0.000	☑
1	19	Generators	2000	0.000	☑

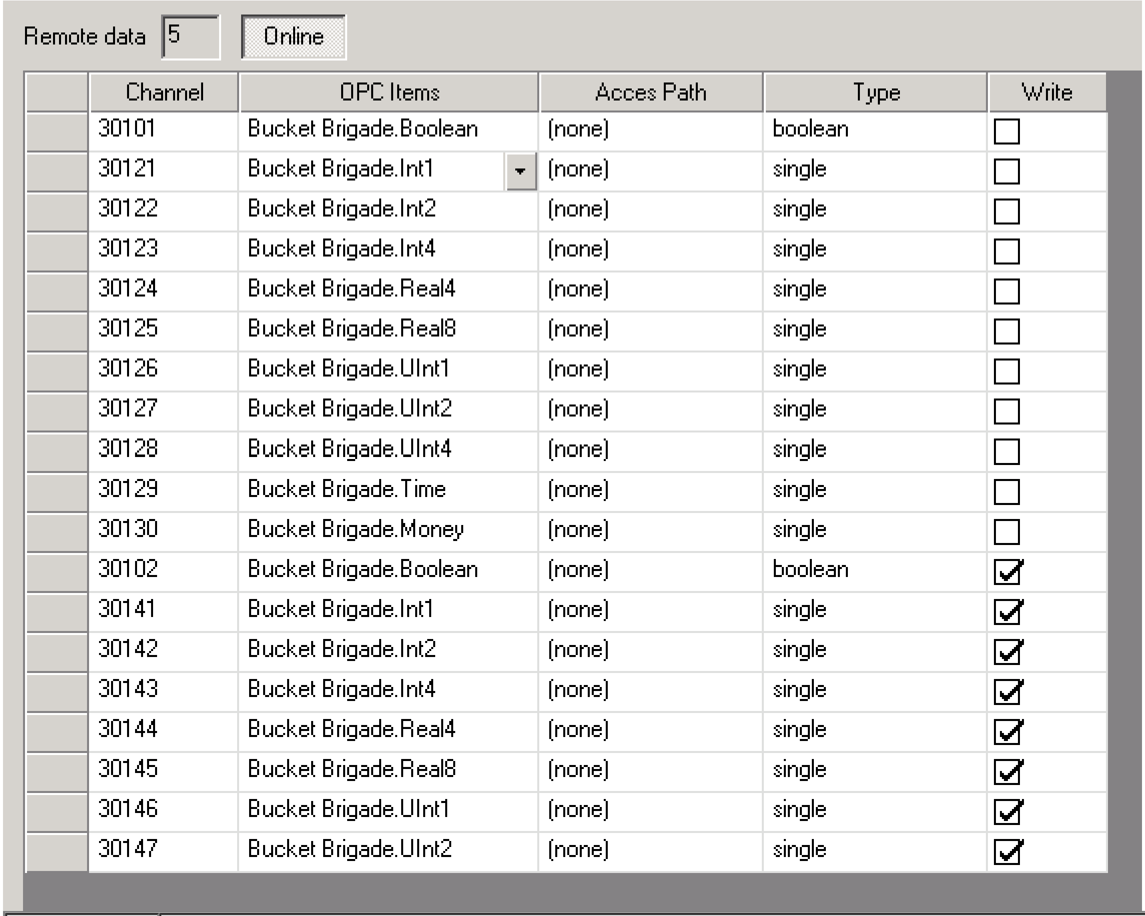
Column	Description
Nr	Group Number
Group	Group used to supply channels from. '(none)' means the channels can be retrieved from the entire system.
Description	Brief description of group (used in group connection-list tree items)
Update Time	Time between updates
Deadband	Percentage Deadband of full scale EU Range, before values are send.
Active	Group is active in Client Group is not used in runtime

See also:

General information OPC OPC Client
OPC Client Forms Server Settings Group Overview Connection List

OPC client form Connection list

The form is used to assign/remove or modify channel connections to OPC Items.



Remote data 5 Online

	Channel	OPC Items	Acces Path	Type	Write
	30101	Bucket Brigade.Boolean	(none)	boolean	□
	30121	Bucket Brigade.Int1	(none)	single	□
	30122	Bucket Brigade.Int2	(none)	single	□
	30123	Bucket Brigade.Int4	(none)	single	□
	30124	Bucket Brigade.Real4	(none)	single	□
	30125	Bucket Brigade.Real8	(none)	single	□
	30126	Bucket Brigade.UInt1	(none)	single	□
	30127	Bucket Brigade.UInt2	(none)	single	□
	30128	Bucket Brigade.UInt4	(none)	single	□
	30129	Bucket Brigade.Time	(none)	single	□
	30130	Bucket Brigade.Money	(none)	single	□
	30102	Bucket Brigade.Boolean	(none)	boolean	☑
	30141	Bucket Brigade.Int1	(none)	single	☑
	30142	Bucket Brigade.Int2	(none)	single	☑
	30143	Bucket Brigade.Int4	(none)	single	☑
	30144	Bucket Brigade.Real4	(none)	single	☑
	30145	Bucket Brigade.Real8	(none)	single	☑
	30146	Bucket Brigade.UInt1	(none)	single	☑
	30147	Bucket Brigade.UInt2	(none)	single	☑

<table>
<tr><th>Setting</th><th>Description</th></tr>
<tr><td>ChannelO</td><td>Channel to receive/send the data; type in an existing digital/analog channel number, or get here from the channel’s configuration form (status/value link)</td></tr>
<tr><td>OPC Item</td><td>ID of OPC Item to retrieve values from</td></tr>
<tr><td>Access Path</td><td>OPC Servers can supply multiple ways to retrieve data, here a fixed route can be selected. ‘(none)’ means the OPC Server may decide how to retrieve the data.</td></tr>
<tr><td>Type</td><td>Data type to retrieve value in.
<table>
<tr><th>Subtype</th><th>Description</th></tr>
<tr><td><b>Boolean</b></td><td>Contains either <b>True</b> or <b>False</b>.</td></tr>
<tr><td><b>Byte</b></td><td>Contains integer in the range 0 to 255.</td></tr>
<tr><td><b>Integer</b></td><td>Contains integer in the range -32,768 to 32,767.</td></tr>
<tr><td><b>Long</b></td><td>Contains integer in the range -2,147,483,648 to 2,147,483,647.</td></tr>
<tr><td><b>Single</b></td><td>Contains a single-precision, floating-point number in the range -3.402823E38 to -1.401298E-45 for negative values; 1.401298E-45 to 3.402823E38 for positive values.</td></tr>
<tr><td><b>Double</b></td><td>Contains a double-precision, floating-point number in the range -1.79769313486232E308 to -4.94065645841247E-324 for negative</td></tr>
</table>
</td></tr>
</table>

	values; 4.94065645841247E-324 to 1.79769313486232E308 for positive values.
Write	Data is written from Client to Server Data is read from Server to Client

The Online button is used to retrieve the OPC Items 'live' from the OPC Server, afterwards the OPC Item can be selected from a drop down list.

See also:

General information OPC OPC Client
OPC Client Forms Server Settings Group Overview Connection List
Appendix A:Value Conversion

OPC client process value conversions

The OPC Client has to convert the Variant (Process Value from OPC Server) to/from a channel value.

Channel value have a limited range, so certain values are not allowed.
Out of range values are not updated to the Channels.

Read conversions (1 decimal)

Type	Range low	Range High	Channel EU Low	Channel EU High	Connection
Boolean	0	1	Normal	Low Alarm	STATUS
Single	$-3.402823e^{38}$ to $-1.401298e^{-45}$	$1.401298e^{-45}$ to $3.402823e^{38}$	-214,748,364.8	214,748,364.7	VALUE
Double	$-1.79769313486232e^{308}$ to $-4.94065645841247e^{-324}$	$4.94065645841247e^{-324}$ to $1.79769313486232e^{308}$	-214,748,364.8	214,748,364.7	VALUE

Read conversions (3 decimals)

Type	Range low	Range High	Channel EU Low	Channel EU High	Connection
Boolean	0	1	Normal	Low Alarm	STATUS
Single	$-3.402823e^{38}$ to $-1.401298e^{-45}$	$1.401298e^{-45}$ to $3.402823e^{38}$	-2,147,483.648	2,147,483.647	VALUE
Double	$-1.79769313486232e^{308}$ to $-4.94065645841247e^{-324}$	$4.94065645841247e^{-324}$ to $1.79769313486232e^{308}$	-2,147,483.648	2,147,483.647	VALUE

Write conversions (1 decimal)

Channel EU Low	Channel EU High	Connection	Type	Range low	Range High
Normal	Low Alarm	STATUS	Boolean	0	1
-214,748,364.8	214,748,364.7	VALUE	Single	$-3.402823e^{38}$ to $-1.401298e^{-45}$	$1.401298e^{-45}$ to $3.402823e^{38}$
-214,748,364.8	214,748,364.7	VALUE	Double	$-1.79769313486232e^{308}$ to $-4.94065645841247e^{-324}$	$4.94065645841247e^{-324}$ to $1.79769313486232e^{308}$

Write conversions (3 decimals)

Channel EU Low	Channel EU High	Connection	Type	Range low	Range High
Normal	Low Alarm	STATUS	Boolean	0	1
-2,147,483.648	2,147,483.647	VALUE	Single	$-3.402823e^{38}$ to $-1.401298e^{-45}$	$1.401298e^{-45}$ to $3.402823e^{38}$
-2,147,483.648	2,147,483.647	VALUE	Double	$-1.79769313486232e^{308}$ to $-4.94065645841247e^{-324}$	$4.94065645841247e^{-324}$ to $1.79769313486232e^{308}$

Notes:

Single values use:

- 23 bits for the fraction value
- 8 bits for signed exponent
- 1 bit for the sign

Unscaled whole values: -16,777,216 to 16,777,215

Double values use:

- 52 bits for the fraction value
- 11 bits for signed exponent
- 1 bit for the sign

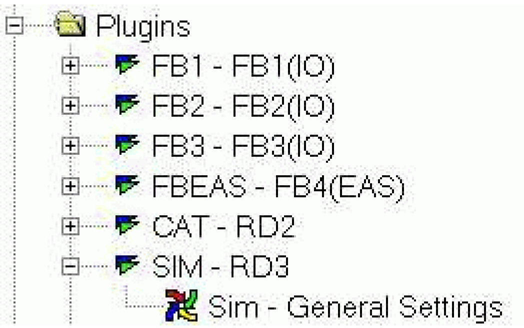
Unscaled whole values: -9,007,199,254,740,991 to 9,007,199,254,740,990

See also:

General information OPC Client
OPC Client Forms Connection List
Appendix A:Value Conversion

SIM – Simulation Plugin

After selecting ‘Remote Data’ and SIM – RD xx



Plugins
- FB1 - FB1(IO)
- FB2 - FB2(IO)
- FB3 - FB3(IO)
- FBEAS - FB4(EAS)
- CAT - RD2
- SIM - RD3
  - Sim - General Settings

There are several items of this remote data:

1. SIM - General Settings
2. SIM - Connection List

The Simulation DLL can be used for demo purpose to show Graphic and Monitoring Windows software with dynamic values. The channels have to be configured on board 80 of link 0, in range 0 to 19.
Each channel has to be configured as a remote data channel.

See next table for exact channel information:

UpDownRamps:

Channel	Scanrate	RangeMin	RangeMax	Step	Init	InitUpOrDown
8000	360 ms	650,00	850,00	0,8000	790,000	UP
8001	334 ms	325,00	425,00	0,4000	395,000	DOWN
8002	242 ms	26,00	34,00	0,0320	32,160	UP
8003	1200 ms	2,60	3,40	0,0032	3,224	DOWN
8004	180 ms	1,30	1,70	0,0080	1,616	UP
8005	190 ms	0,65	0,85	0,0040	0,792	DOWN
8006	200 ms	475,45	634,05	3,1720	589,642	UP
8007	210 ms	51,00	79,00	0,5600	71,440	DOWN

Ramps:

Channel	Scanrate	RangeMin	RangeMax	Step	Init
8008	150 ms	450,00	550,00	0,5000	500,000
8009	200 ms	225,00	275,00	0,2500	250,500
8010	250 ms	18,00	22,00	0,0200	20,080
8011	1500 ms	1,80	2,20	0,0010	2,012
8012	111 ms	0,90	1,10	0,0022	1,008
8013	200 ms	0,45	0,55	0,0010	0,505

8014	227 ms	321,35	401,65	0,7300	366,318
8015	229 ms	23,00	37,00	0,1167	30,980

Digital:

Channel	Init	TimebeforeOn (Sec)	TimebeforeOff (Sec)
8016	ON	100	1
8017	OFF	200	10
8018	ON	20	120
8019	OFF	10	10

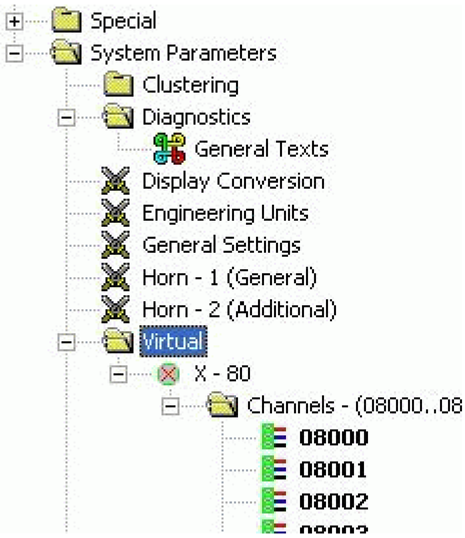
Legend:

Item	Description
Channel	Channel number in the Virtual board
Scanrate	Time before a new value is generated
RangeMin	Lowest Engineer uint value generated
RangeMax	Highest Engineer uint value generated
Step	Value increments per scanrate
Init	Initial value
InitUpOrDown	Initial direction of value
TimeBeforeOn	Time the value is low before going high
TimeBeforeOff	Time the value is high before going low

Steps to take for configuration:

1. In PAL, setup virtual board 80 (see "Virtual" branch in the "System Parameters" branch)
2. Setup channel 8000 to 8015 as Analog Input, Remote Data, take ranges from above table,
3. Setup channel 8016 to 8019 as Digital Input, Remote Data.
4. Use the channels in mimics and alarms as you whish.
5. Add Remote Data Simulation.DLL

See next picture for Virtual board 80:



```
Special
System Parameters
    Clustering
    Diagnostics
        General Texts
    Display Conversion
    Engineering Units
    General Settings
    Horn - 1 (General)
    Horn - 2 (Additional)
    Virtual
        X - 80
            Channels - (08000..08
                08000
                08001
                08002
                08003
```

Shortcuts

Icon:

Place: Remote Data\

SIM – General Settings

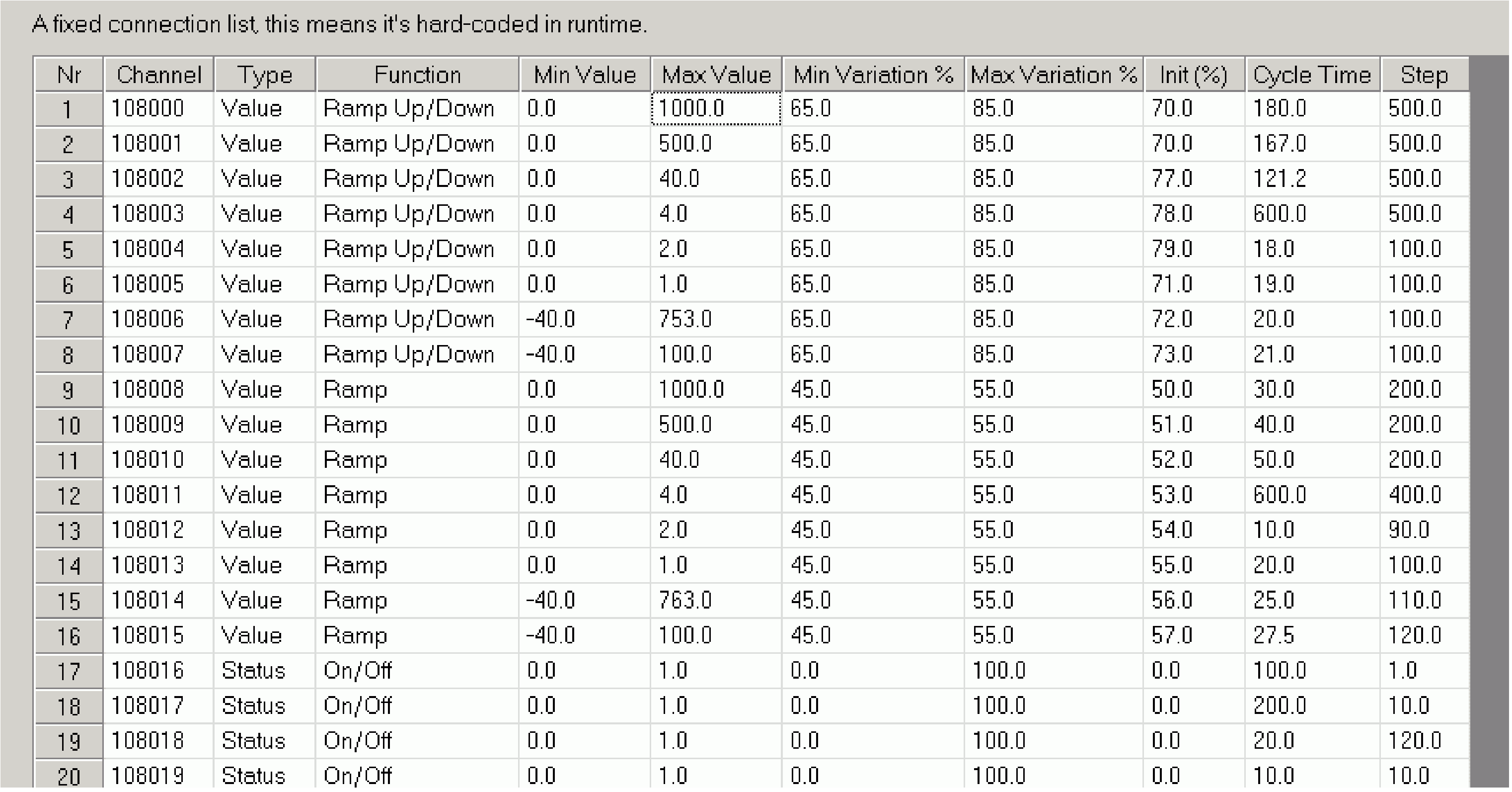
Not implemented yet!

See also:

Simulation Plugin

SIM – Connection List

A channel list which can't be changed.



A fixed connection list, this means it's hard-coded in runtime.

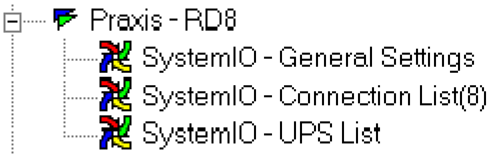
Nr	Channel	Type	Function	Min Value	Max Value	Min Variation %	Max Variation %	Init (%)	Cycle Time	Step
1	108000	Value	Ramp Up/Down	0.0	1000.0	65.0	85.0	70.0	180.0	500.0
2	108001	Value	Ramp Up/Down	0.0	500.0	65.0	85.0	70.0	167.0	500.0
3	108002	Value	Ramp Up/Down	0.0	40.0	65.0	85.0	77.0	121.2	500.0
4	108003	Value	Ramp Up/Down	0.0	4.0	65.0	85.0	78.0	600.0	500.0
5	108004	Value	Ramp Up/Down	0.0	2.0	65.0	85.0	79.0	18.0	100.0
6	108005	Value	Ramp Up/Down	0.0	1.0	65.0	85.0	71.0	19.0	100.0
7	108006	Value	Ramp Up/Down	-40.0	753.0	65.0	85.0	72.0	20.0	100.0
8	108007	Value	Ramp Up/Down	-40.0	100.0	65.0	85.0	73.0	21.0	100.0
9	108008	Value	Ramp	0.0	1000.0	45.0	55.0	50.0	30.0	200.0
10	108009	Value	Ramp	0.0	500.0	45.0	55.0	51.0	40.0	200.0
11	108010	Value	Ramp	0.0	40.0	45.0	55.0	52.0	50.0	200.0
12	108011	Value	Ramp	0.0	4.0	45.0	55.0	53.0	600.0	400.0
13	108012	Value	Ramp	0.0	2.0	45.0	55.0	54.0	10.0	90.0
14	108013	Value	Ramp	0.0	1.0	45.0	55.0	55.0	20.0	100.0
15	108014	Value	Ramp	-40.0	763.0	45.0	55.0	56.0	25.0	110.0
16	108015	Value	Ramp	-40.0	100.0	45.0	55.0	57.0	27.5	120.0
17	108016	Status	On/Off	0.0	1.0	0.0	100.0	0.0	100.0	1.0
18	108017	Status	On/Off	0.0	1.0	0.0	100.0	0.0	200.0	10.0
19	108018	Status	On/Off	0.0	1.0	0.0	100.0	0.0	20.0	120.0
20	108019	Status	On/Off	0.0	1.0	0.0	100.0	0.0	10.0	10.0

See also:

Simulation Plugin

SystemIO – System Input/Output Plugin

After selecting ‘Remote Data’ and SystemIO – RD xx



Praxis - RD8
- SystemIO - General Settings
- SystemIO - Connection List(8)
- SystemIO - UPS List

There are several items of this remote data:

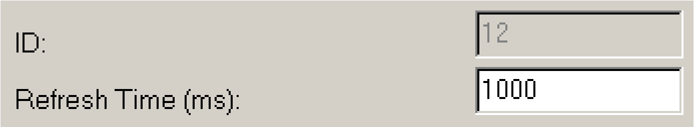
1. SystemIO - General Settings
2. SystemIO - Connection List
3. SystemIO – UPS List

Shortcuts

Icon: ◤

Place: Remote Data\

SystemIO – General Settings



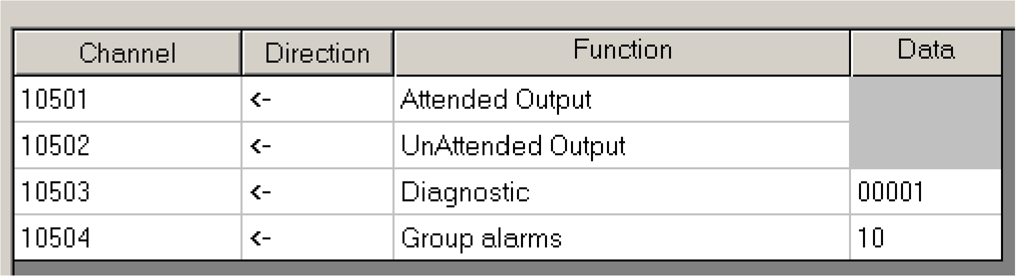
ID:	12
Refresh Time (ms):	1000

Refresh Rate 200-300000 millisec (default: 1000)

See also:

SystemIO Plugin

SystemIO – Connection List



Channel	Direction	Function	Data
10501	<-	Attended Output	
10502	<-	UnAttended Output	
10503	<-	Diagnostic	00001
10504	<-	Group alarms	10

Field	Description
Channel	Channel to receive/send the data Type in an existing digital/analog channel number, or get here from the channel’s configuration form (**status/value link**)
Direction	points out, which side data is flows
Function	Function, like Attended Output, Group, etc.
Data	Function data, like group number or diagnostic number

By pressing on column “Channel” or “Direction” a sort action is performed, first time ascending second descending.

Return Button: Return from gateway, only available when getting there from a channel configuration (status/value link)

For Direction <-

Attended input
UnAttended input
Reset Timer
GEA Input
Key Switch / Timer Off
Reset GEA Input
Acknowlegde Input
Stop Horn Input
Stop Horn1 Input
Stop Horn2 Input
Select Engineer On Duty
Start Stop Patrol Timer
Bridge Calls Cabin:
ECR Calls Cabin:
Bridge Calls All
ECR Calls All

For Direction ->

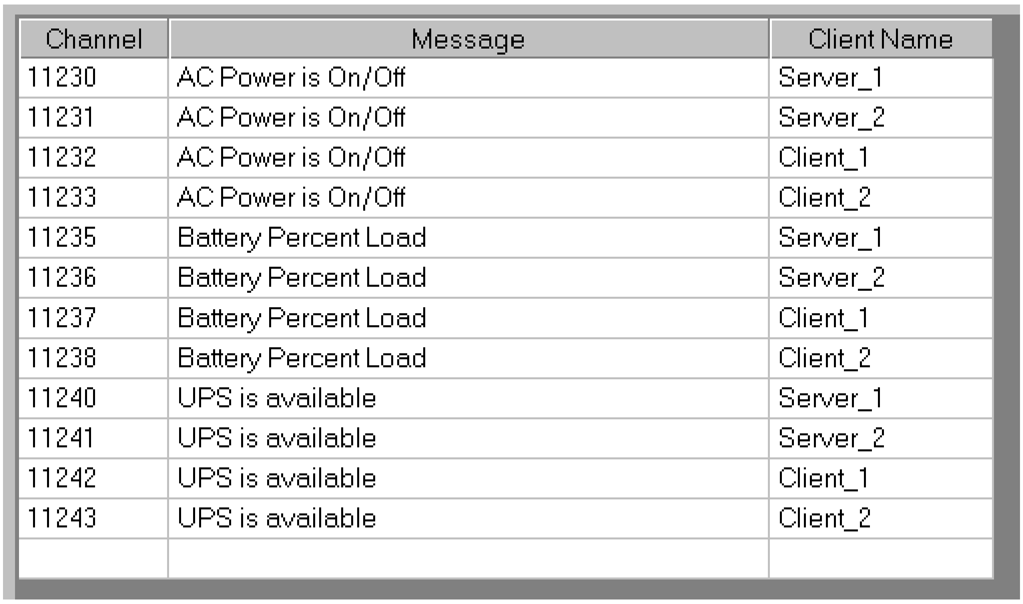
Attended Output
UnAttended Output
Timer Expired Output
GEA or DeadMan Output
Horn 1 Output
Horn 2 Output
Group alarms
Diagnostic
Group acknowledged alarms

Group not acknowledged alarms
Server Ack
Server Stop Horn
Server Ack + Stop Horn
Hourcounter
Current GMT Time
DeadMan Output
GEA Output
Timer On Output
Hourcounter (seconds)
Hourcounter (minutes)
Hourcounter (hours)
On Duty Selection
Along Side Output
Sailing Output
Output Bridge Calls Cabin:
Output ECR Calls Cabin:
Output Bridge Calls All
Output ECR Calls All

See also:

SystemIO Plugin

SystemIO – UPS List



Channel	Message	Client Name
11230	AC Power is On/Off	Server_1
11231	AC Power is On/Off	Server_2
11232	AC Power is On/Off	Client_1
11233	AC Power is On/Off	Client_2
11235	Battery Percent Load	Server_1
11236	Battery Percent Load	Server_2
11237	Battery Percent Load	Client_1
11238	Battery Percent Load	Client_2
11240	UPS is available	Server_1
11241	UPS is available	Server_2
11242	UPS is available	Client_1
11243	UPS is available	Client_2

Field	Description
Channel	Channel to receive/send the data; type in an existing digital/analog channel number, or get here from the channel's configuration form (**status/value link**)
Message	message description of UPS status
Client Name	Computer name

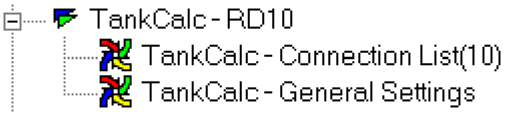
By pressing on column "Channel" a sort action is performed, first time ascending second descending.

See also:

SystemIO Plugin

TankCalc – Tank Calculation Plugin

After selecting ‘Remote Data’ and TankCalc – RD xx



TankCalc - RD10
TankCalc - Connection List(10)
TankCalc - General Settings

There are several items of this remote data:

1. TankCalc - General Settings

2. TankCalc - Connection List

This Remote Data is a plugin that does the following functions:

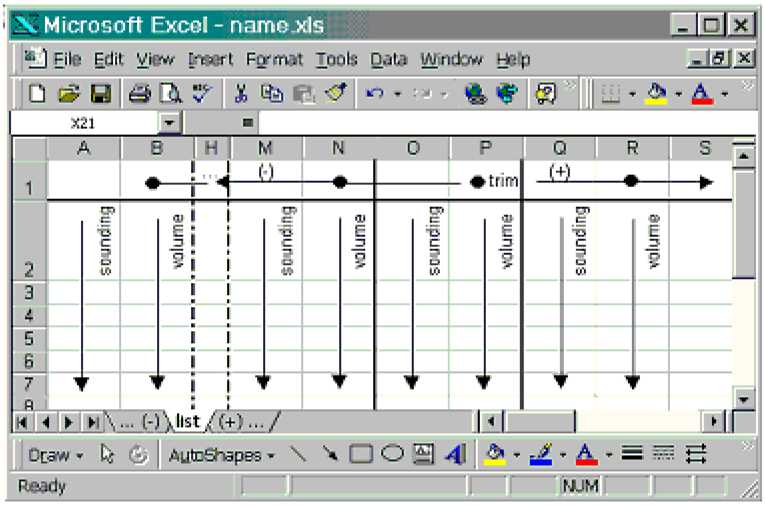
- Retrieve sounding tables from a predefined location on the system
- Read trim, list and sounding data from channels
- Convert sounding data to tank contents according the tables
- Place tank contents into channels

Tank Table = An Excel .xls file containing the sounding data for one tank.

TANK TABLES

- For each tank a separate xls file must be present with extension “xls” in the native Excel file format, located in an appropriate folder.
- For each list a separate sheet must exist with as name the corresponding list value in ‘floating point’ notation; e.g. “-1.200” (*not the quotes!*).

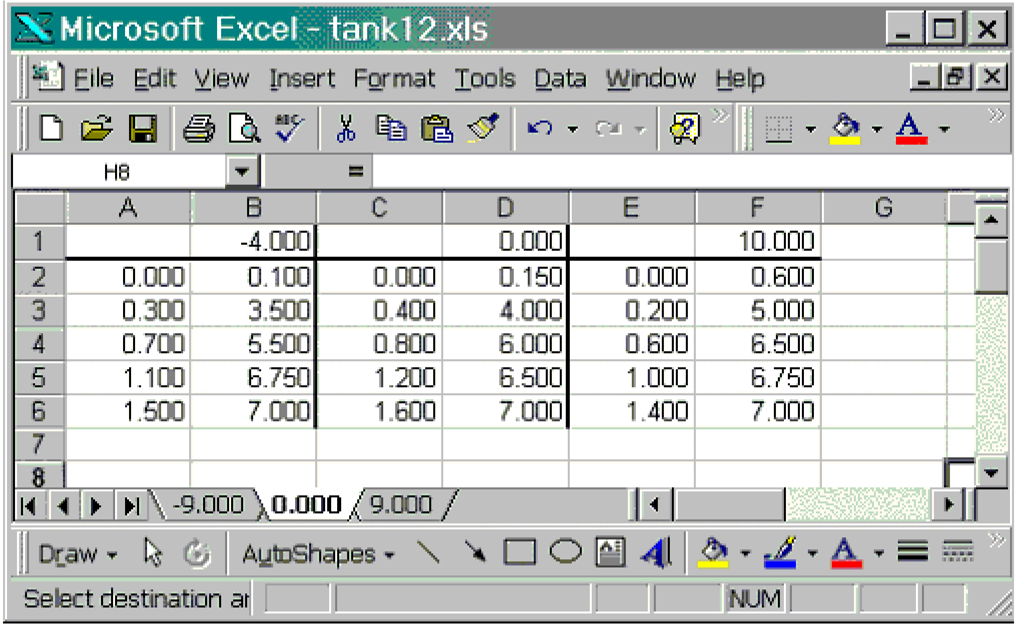
Any number of sheets is allowed.



- Sheet order is not important; keep it in ascending list value order for clarity.
- If list correction is not used, only one sheet should be present; its name is irrelevant.
- Per sheet:
  - Don’t use any headers; all formatting/styling is allowed.
  - All cells holding a value should be of type ‘number’ and hold a floating point value.
  - The first row holds the trim values in ascending order *on each alternate column*; start from ‘B1’.
  - The first column holds the measurement values (sounding/pressure), related to the first trim value, in ascending order; start from ‘A2’.
  - The second column (below first trim value) holds the corresponding contents values (volume); start from ‘B2’.
  - The third column holds the measurement values, related to the second trim value; start from ‘C2’.

The fourth column (below second trim value) holds the corresponding contents values; start from ‘D2’.

- Etc.
- Any dimension is allowed; but all columns, must have the same length *per sheet* (repeating the last sounding-volume pair if needed).
- example:



Microsoft Excel - tank12.xls
File Edit View Insert Format Tools Data Window Help
H8
=
Draw
AutoShapes
-9.000
0.000
9.000
Select destination ar
NUM

	A	B	C	D	E	F	G
1		-4.000		0.000		10.000	
2	0.000	0.100	0.000	0.150	0.000	0.600	
3	0.300	3.500	0.400	4.000	0.200	5.000	
4	0.700	5.500	0.800	6.000	0.600	6.500	
5	1.100	6.750	1.200	6.500	1.000	6.750	
6	1.500	7.000	1.600	7.000	1.400	7.000	
7							
8							

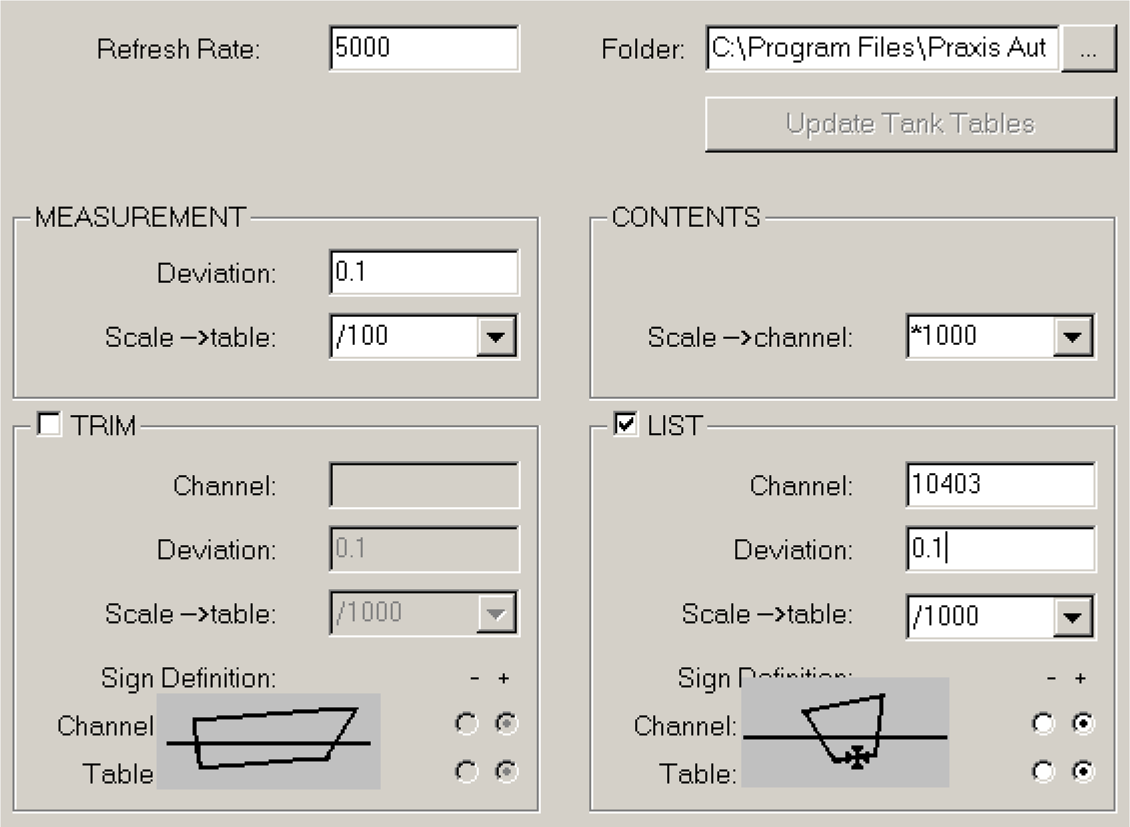
- If trim correction is not used, only one measurement column and one corresponding contents column should be present; its trim value is irrelevant, *but still start from row 2 (A2-B2)*!

Shortcuts

Icon:

Place: Remote Data\

TankCalc – General Settings



Refresh Rate: 5000

Folder: C:\Program Files\Praxis Aut ...

MEASUREMENT

Deviation: 0.1

Scale –>table: /100

CONTENTS

Scale –>channel: *1000

☐ TRIM

Channel:

Deviation: 0.1

Scale –>table: /1000

Sign Definition: - +

Channel ○ ●

Table ○ ●

☑ LIST

Channel: 10403

Deviation: 0.1

Scale –>table: /1000

Sign Definition: - +

Channel: ○ ●

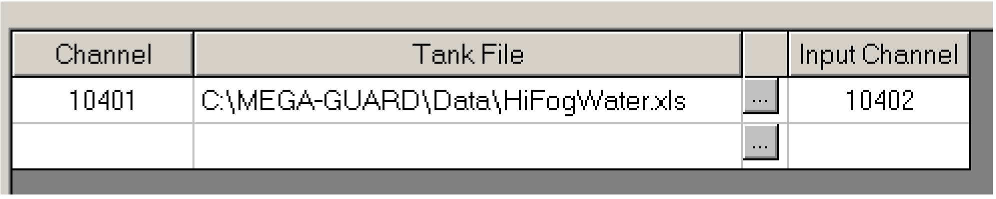
Table: ○ ●

Refresh Rate	update time for all tanks; 2000-300000 millisec (default: 5000)
Folder	folder path, path were all tanks tables are stored; if left empty each table can have its own specific
(*browse*)	path (default: *empty*)
	when only tables (Excel xls spreadsheet files) are changed, press this button to ensure they are
Update Tank	reloaded by the run-time; the button is dimmed if reloading will already take place
Tables	minimum difference in value before a new calculation is performed; 0.1-20000.0 (default: 0.1)
	scale factor from channel to tank table; “/1000” – “*1000” (default: “/1000”)
	scale factor from tank table to channel; “*1000” – “/1000” (default: “*1000”)
Deviation	select if trim correction is to be performed; if *off* but table still holds trim data, trim = 0 is
	presumed; if *on* but table holds no trim data, trim settings are ignored (default: *off*)
Scale ->table	select if list correction is to be performed; if *off* but table still holds list data, list = 0 is presumed; if
	on but table holds no list data, list settings are ignored (default: *off*)
Scale -	input channel; 10101-49668, if left empty the corresponding selection is forced to *off* (default:
>channel	*empty*)
	for input channel and tank tables the sign can be defined
TRIM	forepeak / starboard up = negative value
	forepeak / starboard up = positive value (default)
LIST	
Channel	
Sign Definition	
Channel/Table –	
Channel/Table +	

See also:

Tank Calculation Plugin

TankCalc – Connection List



Channel	Tank File		Input Channel
10401	C:\MEGA-GUARD\Data\HiFogWater.xls	...	10402
		...	

Field	Description
Channel	Channel to receive the calculated tank contents, type in an existing analog channel number or get here from the channel's configuration form **value link**
Tank File *(browse)*	Name of the Excel .xls file with the tank sounding data with or without a preceding path, ".xls" must be present; it is appended to the folder path entered on the General Settings form, if selected with the browse button, the complete path is set when the folder on the General Settings form was left empty; just the filename only is set if a folder path was entered
Input Channel	Channel which holds the sounding value, type in an existing analog channel number

By pressing on column "Channel" or "Tank File" or "Input Channel" a sort action is performed, first time ascending second descending.

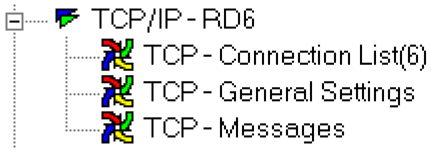
Return Button: Return from gateway, only available when getting there from a channel configuration (status/value link)

See also:

Tank Calculation Plugin

TCP/IP – Plugin

After selecting ‘Remote Data’ and TCP/IP– RD xx



TCP/IP - RD6
TCP - Connection List(6)
TCP - General Settings
TCP - Messages

There are several items of this remote data:

1. TCP - General Settings
2. TCP - Connection List
3. TCP - Message List

Protocol Description

Three types are support yet:
TCP/IP – Type Protocol VVVF
TCP/IP – Type Protocol BS
TCP/IP – Type Protocol PATDX

See Also:

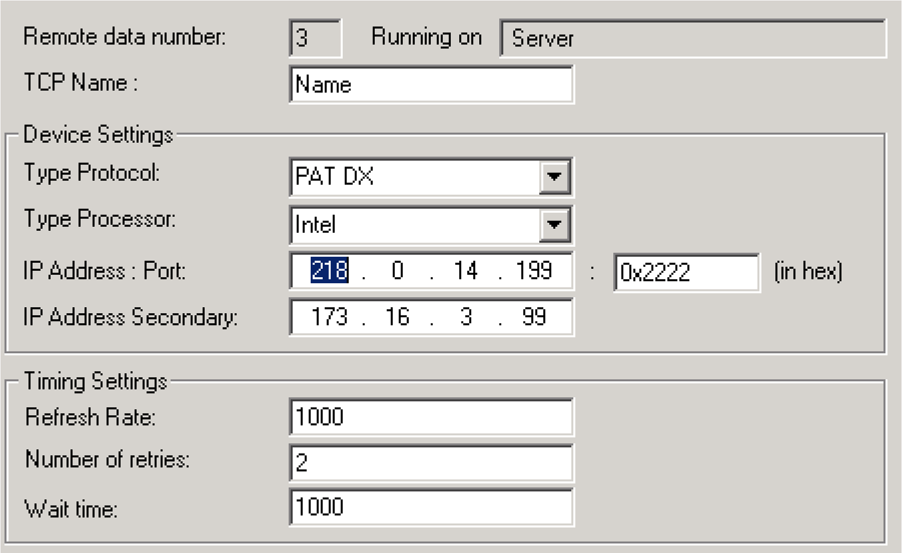
TCP/IP Registry Settings

Shortcuts

Icon:

Place: Remote Data

TCP/IP – General Settings



Remote data number:	3	Running on	Server
TCP Name :	Name		

Device Settings

Type Protocol:	PAT DX		
Type Processor:	Intel		
IP Address : Port:	218 . 0 . 14 . 199	: 0x2222	(in hex)
IP Address Secondary:	173 . 16 . 3 . 99		

Timing Settings

Refresh Rate:	1000
Number of retries:	2
Wait time:	1000

Remote data Remote data being edited, 1-32

Running on Server or Protocol Plug-in running on Marine PC
Processor LXX (LBB-LBB) Protocol running on Processor (IO, Terminal, control Processor) not possible so a message will be given.
L = Link, XX = Processor number, BB = Board number

TCP Name Name, documentation only

Type Protocol VVVF / BS / PATDX

Type Processor Intel or Motorola (used for sending/receiving data) (BS only)

IP Address : Port TCP/IP Address + Port

IP Address Secondary TCP/IP Address

Refresh Rate Time to do a complete cycle, 1000

Number Of Retries Number of retries before failure, 2

Wait Time Wait time after a complete cycle, 100

Remarks:
This protocol is always running on server.
An IP port can only be used by one application or plug-in.
Wait time is used in PAT DX as send or receive timeout.

See also:

TCP/IP - Plugin

TCP/IP – Type Protocol VVVF

TCP/IP – Type Protocol BS

TCP/IP – Type Protocol PATDX

TCP/IP – Type Protocol VVVF

- TCP/IP Plugin will open the communication
- TCP/IP Plugin will start sending data every 1 second
- VVVF will reply with data every time.

TCP/IP Plugin will send the following message to VVVF:

Variable	*Type (size)*	*Initial value*
Length	WORD (2 bytes)	104
Message Type	BYTE	2. x02
Data Type	BYTE	0x0
Communication Type	BYTE	0x0
Message Number	BYTE	1.
Counter	WORD (2 bytes)	0
DW [99]	WORD (99 * 2 bytes)	0 – 10000

In reply to this message VVVF will send to TCP/IP Plugin:

Variable	*Type (size)*	*Initial value*
Length	WORD (2 bytes)	104
Message Type	BYTE	2. x02
Data Type	BYTE	0x0
Communication Type	BYTE	0x0
Message Number	BYTE	101
Counter	WORD (2 bytes)	0
DW [99]	WORD (99 * 2 bytes)	0 – 10000

- *Header: Length* contains the number of bytes of message (excluding this one)
- *Header: MessageType* always contains 0x02. It is used in our system, and identifies these structures.
- *Header: DataType* contains 0x0. Not defined
- *Header: Communication Type* identifies message delivery service over the network, contains 0x0 = virtual circuit service (in which the connection between the connected nodes is periodically checked automatically)
- *Header Message Number:* identifies the message block (1 in the provided information)
- *Header Counter:* identifies current counter, change every time when a correct send and receive is done
- *Data:* DW variable contains a list of 99 values in the message block

On this manner several blocks of information can be send between the systems.

See also:

TCP/IP Plugin

TCP/IP – Type Protocol BS

BS stands for Bakker Sliedrecht. A company PLC interface

- TCP/IP Plugin will open the communication
- BS will start sending data every 2 seconds
- TCP/IP Plugin will reply with data every time.

BS will send the following message to TCP/IP Plugin:

Variable	*Type (size)*	*Initial value*
MessageType	WORD (2 bytes)	0x02
Length	WORD (2 bytes)	5
MessageRequest	WORD (2 bytes)	53
MessageNumber	WORD (2 bytes)	1.
DW [5]	WORD (6 * 2 bytes)	0 - 10000

In reply to this message TCP/IP Plugin will send to BS:

Variable	*Type (size)*	*Initial value*
MessageType	WORD (2 bytes)	2. x02
Length	WORD (2 bytes)	16
MessageRequest	WORD (2 bytes)	1.
MessageNumber	WORD (2 bytes)	53
DW [16]	WORD (16 * 2 bytes)	0 - 10000

- *Header: MessageType* always contains 0x02. It is used in our system, and identifies these structures.
- *Header: Length* contains the number of values in the DW list.
- *Header: MessageRequest* identifies which message must be returned, zero (0) means none.
- *Header:* MessageNumber identifies the message block (53 in the provided information)
- *Data:* DW variable contains a list of 16 values in the message block

On this manner several blocks of information can be send between the systems. If there is nothing to send MessageNumber is set to 0. If nothing should be returned, MessageRequest is set to 0. The BS sends messages (with information and request for information) with intervals of 10 seconds. (If a longer interval is sufficient, that is preferred)

The system is redundant. If the BS is not responding at the primary tcp-ip address, TCP/IP Plugin will try to connect to/communicate with the secondary tcp-ip address.

See also:

TCP/IP Plugin

TCP/IP – Type Protocol PATDX

PAT DX data exchange possibilities:
- Value word (16 bits, signed)
- Binary packed word (16 separate bits)
- Value double float (IEEE)

TCP is used for remote equipment to receive data from, or send data to MEGA-GUARD via any communication method supporting TCP/IP. Such equipment will be referred to as "remote system".

-MEGA-GUARD TCP Plugin opens communication.
-MEGA-GUARD TCP Plugin sends data 2 times per second to both remote systems,
in a data-send from MEGA-GUARD TCP a requests for data is given, remote system replies with data every time.

`SYSTEMTIME Structure`

`SYSTEMTIME`	*Type (size)*	*Description*
wYear	WORD (2 bytes)	Specifies the current year. The year must be greater than 1601.
Wmonth	WORD (2 bytes)	Specifies the current month; January = 1, February = 2, and so on.
WdayOfWeek	WORD (2 bytes)	Specifies the current day of the week; Sunday = 0, Monday = 1, and so on.
Wday	WORD (2 bytes)	Specifies the current day of the month.
Whour	WORD (2 bytes)	Specifies the current hour.
WMinute	WORD (2 bytes)	Specifies the current minute.
WSecond	WORD (2 bytes)	Specifies the current second.
WMilliseconds	WORD (2 bytes)	Specifies the current millisecond.

System time is in Coordinated Universal Time (UTC), words are aligned and in Intel format (LSB,MSB)

Message layout

Remote system sends following message to MEGA-GUARD TCP plugin:

Variable	*Type (size)*	*Initial value*
MessageType	WORD (2 bytes)	0x03
Length	WORD (2 bytes)	n
MessageRequest	WORD (2 bytes)	101
MessageNumber	WORD (2 bytes)	1
Counter	WORD (2 bytes)	C
Timestamp	`SYSTEMTIME` (16 bytes)	Current UTC Time
Data [n]	WORD (n * 2 bytes)	-10000 – 10000

In this message, Counter c starts at 1. In reply to this message MEGA-GUARD TCP plugin sends to remote system:

Variable	*Type (size)*	*Initial value*
MessageType	WORD (2 bytes)	0x03
Length	WORD (2 bytes)	N
MessageRequest	WORD (2 bytes)	2
MessageNumber	WORD (2 bytes)	101
Counter	WORD (2 bytes)	c=c+1
Timestamp	`SYSTEMTIME` (16 bytes)	Current UTC Time
Data [n]	WORD (n * 2 bytes)	-10000 - 10000

In reply to this message remote system sends following message to MEGA-GUARD TCP plugin:

Variable	*Type (size)*	*Initial value*
MessageType	WORD (2 bytes)	0x04
Length	WORD (2 bytes)	n
MessageRequest	WORD (2 bytes)	102
MessageNumber	WORD (2 bytes)	2
Counter	WORD (2 bytes)	c=c+1
Timestamp	`SYSTEMTIME` (16 bytes)	Current UTC Time
Data [n]	DOUBLE (n * 8 bytes)	IEEE standard

In reply to this message MEGA-GUARD TCP plugin sends to remote system:

Variable	*Type (size)*	*Initial value*
MessageType	WORD (2 bytes)	0x04
Length	WORD (2 bytes)	n
MessageRequest	WORD (2 bytes)	0
MessageNumber	WORD (2 bytes)	102
Counter	WORD (2 bytes)	c=c+1
Timestamp	`SYSTEMTIME` (16 bytes)	Current UTC Time
Data [n]	DOUBLE (n * 8 bytes)	IEEE standard

If no request, communication loop has finished here. A new request can be started:

- Header: MessageType always contains 0x03. It is used in our system, and identifies these structures.

- Header: Length contains the number of values in the DW list.
- Header: MessageRequest identifies which message must be returned, zero (0) means none.
- Header: MessageNumber identifies a message block (1 to 99 for Remote System and 101 to 199 for MEGA-GUARD)
- Counter: Range 0-65535. This number starts at zero and increases with each message that is send. When it reaches 65535 it jumps back to 0 and continue increasing from there.
- Timestamp is UTC creation time of the message.
- Data: Variable contains a list of n values in the message block. In message type 0x03 these are 16 bit words that can contain a Boolean value per bit, or just integer values in the range -32767 to 32766. In message type 0x04 Data are doubles in 8 bytes according IEEE with a sign bit, 11-bit exponent, and 52-bit mantissa.

If values must be send that are larger then the limits allow, the maximum value will be send.

Note that

- MessageType 0x02 is used with "VVVF",

- MessageType 0x02 is also used "TCP 02",

- MessageType 0x03 and 0x04 are used with "PAT DX" (this protocol).

For example

First an example with system time 2005-07-05 01:02:03.004 noted as below in a message

Timestamp	`SYSTEMTIME` (16 bytes)	2005; 07; 2; 05; 01; 02;03; 004

In system time structure it then looks as below

wYear	WORD (2 bytes)	2005
wMonth	WORD (2 bytes)	07 (July)
wDayOfWeek	WORD (2 bytes)	2 (Tuesday)
wDay	WORD (2 bytes)	05
wHour	WORD (2 bytes)	01
wMinute	WORD (2 bytes)	02
wSecond	WORD (2 bytes)	03
wMilliseconds	WORD (2 bytes)	004

MEGA-GUARD TCP plugin sends to remote system:

Variable	*Type (size)*	*Example values*
MessageType	WORD (2 bytes)	**0x03**
Length	WORD (2 bytes)	**2**
MessageRequest	WORD (2 bytes)	**101**
MessageNumber	WORD (2 bytes)	**1**

Counter	WORD (2 bytes)	**1**
Timestamp	SYSTEMTIME (16 bytes)	2005; 07; 2; 05; 01; 02;03; 004
Data [0]	WORD (2 bytes)	**1234**
Data [1]	WORD (2 bytes)	**5678**

Reply to this message remote system sends to MEGA-GUARD TCP plugin:

Variable	*Type (size)*	*Initial value*
MessageType	WORD (2 bytes)	**0x03**
Length	WORD (2 bytes)	**3**
MessageRequest	WORD (2 bytes)	**0**
MessageNumber	WORD (2 bytes)	**101**
Counter	WORD (2 bytes)	**2**
Timestamp	SYSTEMTIME (16 bytes)	2005; 07; 2; 05; 01; 02;03; 112
Data [0]	WORD (2 bytes)	**9999**
Data [1]	WORD (2 bytes)	**8888**
Data [2]	WORD (2 bytes)	**7777**

After that MEGA-GUARD TCP plugin sends to remote system:

Variable	*Type (size)*	*Initial value*
MessageType	WORD (2 bytes)	**0x04**
Length	WORD (2 bytes)	**5**
MessageRequest	WORD (2 bytes)	**102**
MessageNumber	WORD (2 bytes)	**2**
Counter	WORD (2 bytes)	**3**
Timestamp	SYSTEMTIME (16 bytes)	2005; 07; 2; 05; 01; 02;03; 225
Data [0]	DOUBLE (8 bytes)	**1.234**
Data [1]	DOUBLE (8 bytes)	**2.345**
Data [2]	DOUBLE (8 bytes)	**3.456**
Data [3]	DOUBLE (8 bytes)	**4.567**
Data [4]	DOUBLE (8 bytes)	**5.678**

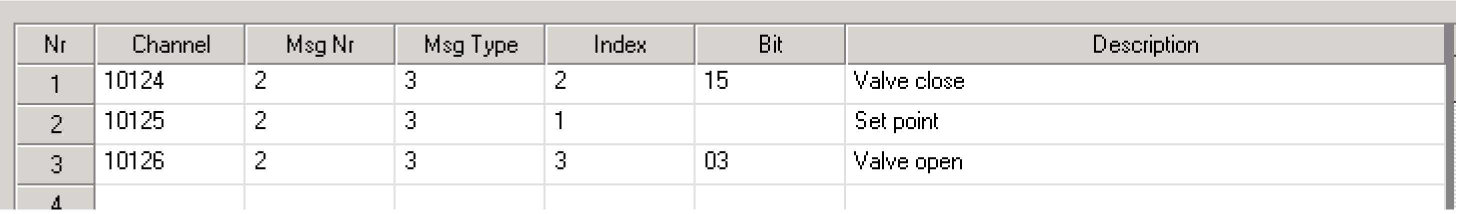
In reply to this message remote system sends:

Variable	*Type (size)*	*Initial value*
MessageType	WORD (2 bytes)	**0x04**
Length	WORD (2 bytes)	**2**
MessageRequest	WORD (2 bytes)	**0**
MessageNumber	WORD (2 bytes)	**102**
Counter	WORD (2 bytes)	**4**
Timestamp	`SYSTEMTIME` (16 bytes)	2005; 07; 2; 05; 01; 02;03; 338
Data [0]	DOUBLE (8 bytes)	**1.234**
Data [1]	DOUBLE (8 bytes)	**2.345**

See also:

TCP/IP Plugin

TCP/IP – Connection List



Nr	Channel	Msg Nr	Msg Type	Index	Bit	Description
1	10124	2	3	2	15	Valve close
2	10125	2	3	1		Set point
3	10126	2	3	3	03	Valve open
4						

Grid Column	**Options/Values and Function**
Channel	Channel to receive/send the data; type in an existing digital/analog channel number, or get here from the channel's configuration form (**status/value link**)
Msg Nr	1..255, be sure configure first the message (see: TCP – Message List)
Msg Type	Msg Type related to Msg Number, 1 till 255
Index	which word in data array, 1 - Size of Message
Bit	empty for analog value, 0..15 one bit for digital value
Description	documentation, what kind of value

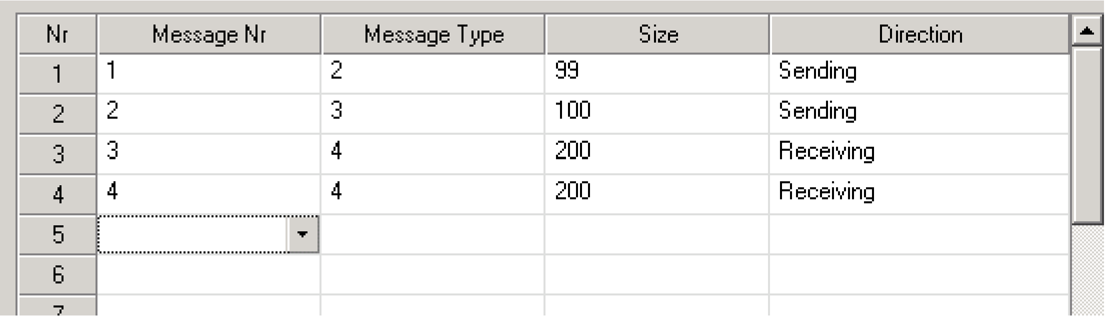
Remarks:

For faster configuration it is possible to use CTRL+D:
-configure one complete row
-select one column and multiple empty rows
-press CTRL+D, automaticially smart default will filled in

See also:

TCP/IP - Plugin

TCP/IP – Messages List



Nr	Message Nr	Message Type	Size	Direction
1	1	2	99	Sending
2	2	3	100	Sending
3	3	4	200	Receiving
4	4	4	200	Receiving
5				
6				
7				

Field	Description
Message Nr	Message Number, 1..255
Message Type	Message Type, Sub Protocol dependant, 1 till 255
Size	Count of Values in message, 0 till 255
Direction	Kind of message Sending or Receiving

Remarks:
Message need to be configured as sub protocol defines
BS; TCP 02; VVVF Message Type always 2
PAT DX Message Type 3 or 4 are supported

See also:

TCP/IP - Plugin

TCP/IP - Registry Settings

Windows:

Be sure the following key values are existent:

HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\Tcpip\Parameters

Value: **KeepAliveInterval** (DWORD, 500 decimal)

Value: **KeepAliveTime** (DWORD, 1000 decimal)

Value: **TcpMaxConnectRetransmissions** (DWORD, 0 decimal)

Remark: After adding/editing these keys windows needs to be restarted.

IOServer:

HKEY_CURRENT_USER\Software\Praxis\IOServer\Settings

Value: **RemoteDataShutDownDelay** (DWORD, 30000 decimal)* (VVVF Fast, 12000 decimal)

Remark: After editing this key ioserver needs to be restarted.

(*30 sec = plugin1(=25 sec) + 5 sec, if 2 plugins, 55 sec = plugin1 + plugin2 + 5 sec)

Plugin Itself:

HKEY_CURRENT_USER\Software\Praxis\tcp\Settings

Value: **ShutDownDelay** (DWORD, 25000 decimal) (VVVF Fast, 5000 decimal)

Value: **ReceiveTimeOut** (DWORD, 8000 decimal) (VVVF Fast, 1500 decimal)

Remark: After editing this key plugin (could be done by restart of ioserver) needs to be restarted.

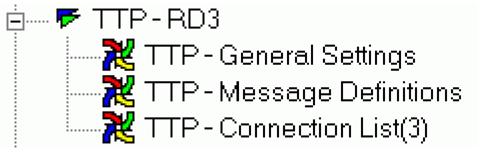
VVVF Fast, could only be used for VVVF protocol type and fast shutdown is required. If not required please use defaults. This feature cannot be used for protocol type BS.

See also:

TCP/IP Plugin

TTP - Plugin

After selecting ‘Remote Data’ and TTP – RD xx



TTP - RD3
TTP - General Settings
TTP - Message Definitions
TTP - Connection List(3)

There are several items of this remote data:

1. TTP - General Settings
2. TTP - Connection List
3. TTP - Message List

Protocol Description

Based on

- NMEA 0183 version 2.01

Sentence going over the Serial line:

$aaccc,c---c*hh<CR><LF>

Sentence description

$	Start of sentence, Start delimiter HEX: 24
Aa	Address field, Talker ID P means custom message Mnemonic can take up to 7 characters
Ccc fields	Address field, Sentence formatter mnemonic: Identifying the data type and string format of the successive fields. Mnemonics will be used as far as possible to facilitate read-outs by users.
,	Field Delimiter Starts each field except Address and checksum fields. HEX: 2C
c---c	Data sentence block: Follows address filed and is a series of data fields containing all of the data to be transmitted. Data field sequence is fixed and identified by third and subsequent characters of the address field (Sentence formatter) Data fields may be of variable length and are preceded by delimiters “,”.
•	Checksum Delimiter (Optional) Follows last data field of the sentence it indicates that the following two alphanumeric characters show the HEX value of the checksum. HEX: 2A
Hh	Checksum field (Optional) The absolute value calculated by exclusive-OR’ing the eight data bits of each character in the sentence, between, but excluding “$” and “*”. The hexadecimal value of the most significant and least significant four bits of the result is converted to two ASCII characters (0-9, A-F) for transmission. The most significant character is transmitted first. The checksum field is optional, except when indicated as mandatory.
<CR><LF>	End of Sentence: Sentence terminating delimiter HEX: 0D 0A

Character Engineering Unit codes

A	Ampere
F	Fathoms
f	Feet
I	Inches
K	Kilometers
k	Kilograms
l	Liter per second l/s
M	Meters
N	Nautical Miles
P	Pascal (pressure)
R	RPM
S	Statute miles
V	Volt

Field types (Supported)

llll.ll Latitude
Degrees, Minutes and Decimal – two fixed digits of degrees, two fixed digits of minutes and a variable number of digits for decimal fraction of minutes. Leading zeros always included for degrees a minutes to maintain fixed length. The decimal point and associated decimal fraction are optional if full resolution is not required.

yyyy.yy Longitude
See Latitude

hhmmss.ss Time
Hours/Minutes/Seconds and decimal - two fixed digits of hours; two fixed digits of minutes; two fixed digits of seconds and a variable number of digits for decimal fraction of seconds. Leading zeros always included for hours; minutes and seconds to maintain fixed length. The decimal point and associated decimal fraction are optional if full resolution is not required.

b Binair Value 0 or 1

bbbb Binair Packed Value, do not use more than 32 b's (32 bits value)

h Hexadecimal Value 0..F

hhhh Hex Packed Value, do not use more than 8 h's (32 bits value)

x Value with whole numbers

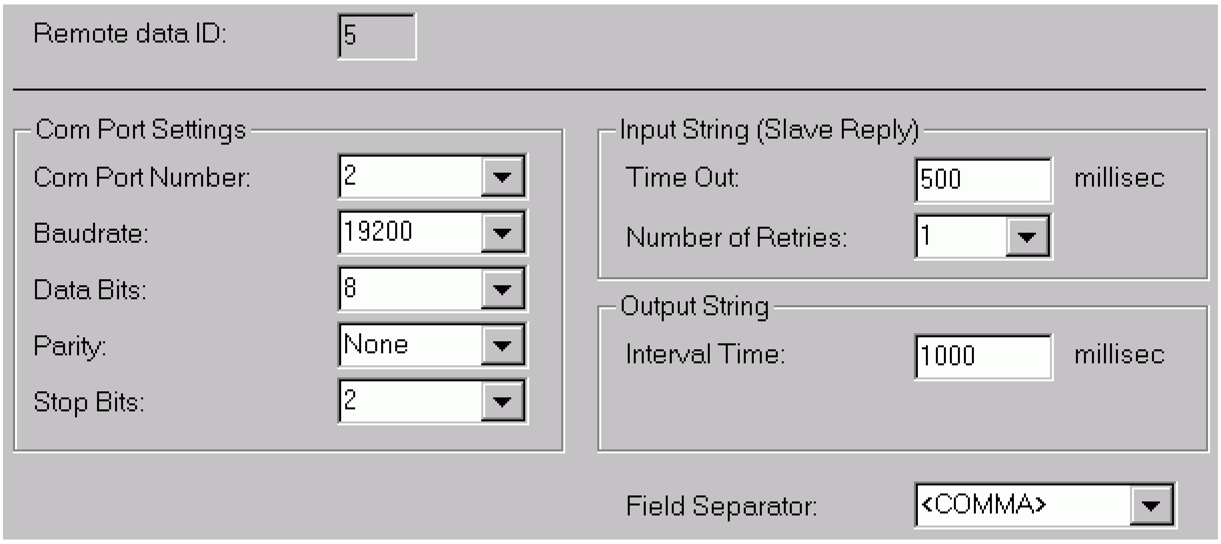
x.x Value with fraction

Shortcuts

Icon: ◤

Place: Remote Data\

TTP – General Settings



Remote data ID: 5

Com Port Number: 2

Baudrate: 19200

Data Bits: 8

Parity: None

Stop Bits: 2

Time Out: 500 millisec

Number of Retries: 1

Interval Time: 1000 millisec

Field Separator: <COMMA>

Com Port Communication port which is used for sending the output. No other plug-in must have the same COM port (only if plugin is on running on server)

Baudrate Baudrate of Com Port, 1200, 2400, 4800, 9600 or 19200 (default: 19200).

Data Bits Number of Data bits of Com Port, 7 or 8

Parity Kind of Parity of Com Port, like None, Odd or Even

Stop Bits Number of Stop Bits of Com Port, 1 or 2

Time Out Maximum time before next message will be processed (in milli-seconds), diagnostic could be displayed

Number Of Retries Number of times to try when message is not received before error is reported

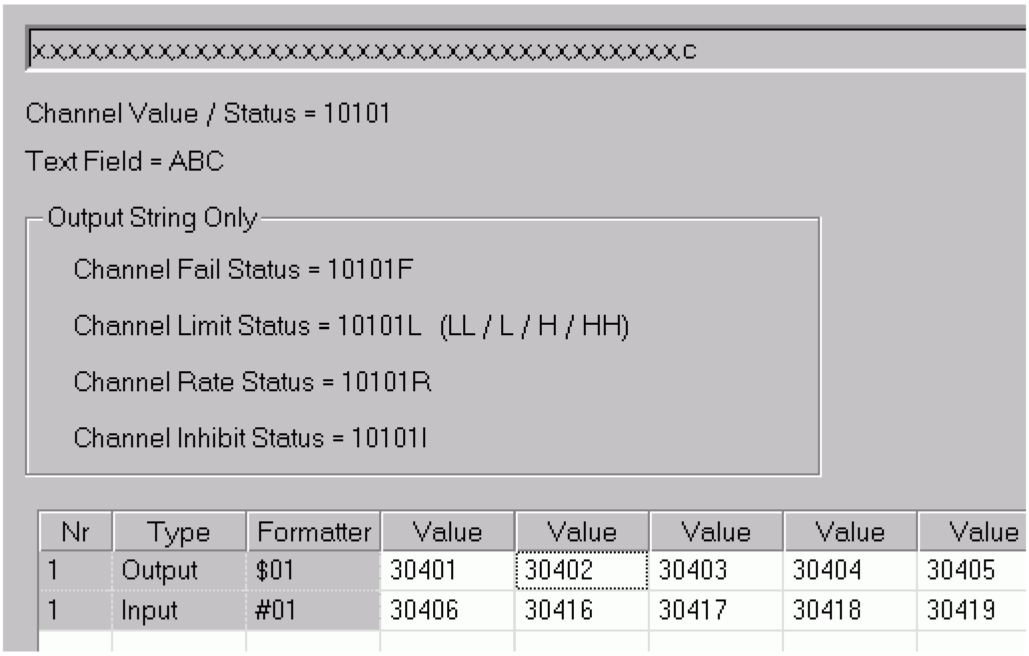
Interval Time Minimal Time to wait between two messages (in milli-seconds)

Field separator separator for data fields, default comma

See also:

TTP Plugin

TTP – Connection List



xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxc

Channel Value / Status = 10101

Text Field = ABC

Output String Only

Channel Fail Status = 10101F

Channel Limit Status = 10101L (LL / L / H / HH)

Channel Rate Status = 10101R

Channel Inhibit Status = 10101I

Nr	Type	Formatter	Value	Value	Value	Value	Value
1	Output	$01	30401	30402	30403	30404	30405
1	Input	#01	30406	30416	30417	30418	30419

From the current selected row message definition is shown.

-Type in Channel Number where value is coming from, like 10560

-Type characters in variable text fields

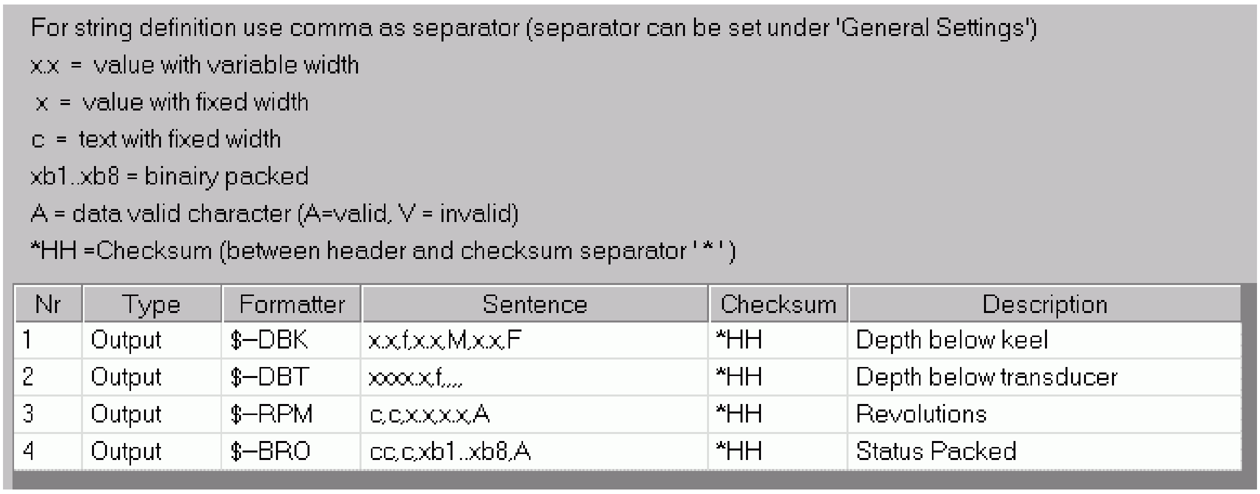
Field	Description
Nr	Number from configured TTP Message List
Type	Output or Input Type from configured TTP Message List
Formatter	Formatter from configured TTP Message List
Value (Channel or Text)	Channel to receive/send the data; type in an existing digital/analog channel number, or get here from the channel’s configuration form (**status/value link**)

See also:

TTP Plugin

TTP – Messages List

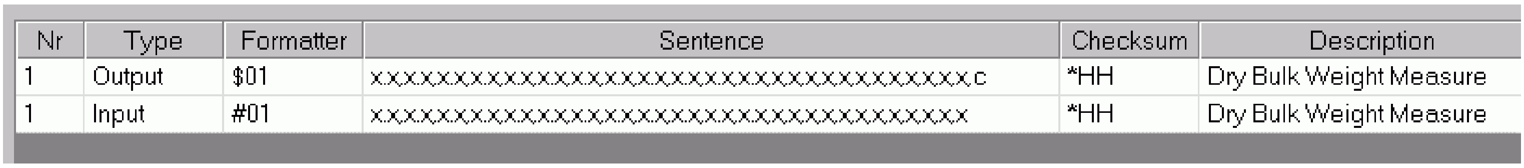
For learning purposes four messages are default created.



For string definition use comma as separator (separator can be set under 'General Settings')
x.x = value with variable width
x = value with fixed width
c = text with fixed width
xb1..xb8 = binairy packed
A = data valid character (A=valid, V = invalid)
*HH =Checksum (between header and checksum separator ' * ')

Nr	Type	Formatter	Sentence	Checksum	Description
1	Output	$--DBK	x.x,f,x.x,M,x.x,F	*HH	Depth below keel
2	Output	$--DBT	xxxx.x,f,,,	*HH	Depth below transducer
3	Output	$--RPM	c,c,x.x,x.x,A	*HH	Revolutions
4	Output	$--BRO	cc,c,xb1..xb8,A	*HH	Status Packed

Next picture shows a configuration of a combination of one output and one input message.



Nr	Type	Formatter	Sentence	Checksum	Description
1	Output	$01	x.x,x.x,x.x,x.x,x.x,x.x,x.x,x.x,x.x,x.x,x.x,x.x,x.x,x.x,x,x,x,x,x,x,x,x,x,x,x,x,c	*HH	Dry Bulk Weight Measure
1	Input	#01	x.x,x.x,x.x,x.x,x.x,x.x,x.x,x.x,x.x,x.x,x.x,x.x,x.x,x.x,x,x,x,x,x,x,x,x,x,x,x,x	*HH	Dry Bulk Weight Measure

Field	Description
Nr	Line number of all messages or message number
Type	Output or Input, Message to Send = Output, Message to Receive = Input
Formatter	Formatter TTP Message List
Sentence	Sentence Description
Checksum	Add Checksum to message, *HH or *hh or empty
Description	Information about this message

Special codes

>x**;**y Select from list (x or y) (eq E or W)

># Select Number (0-9) (eq 0 or 8)

-x x means negative value else positive for previous value

(eq S)

Remove the messages if you don’t want to use then. After that insert de message(s) you like to use.

A TTP message exits of formatter (header of message) and sentence which contains channel value(s) and/or characters which are separated by a comma.

Formatter has usually with two - which are known as “don’t cares” (Like $--DBK).

More about Sentence:

-x.x one channel value, like 10101, variable width

-xxxx.x one channel value, like 10101, fixed width

-x or xx one channel status, like 10101, value is equal to 0 => status normal, value is not equal to 0 => status high

-cc two variable characters like mA

-xb1..xb8 status packed (VDR) with a maximum of (8*4 =) 32 channel statuses

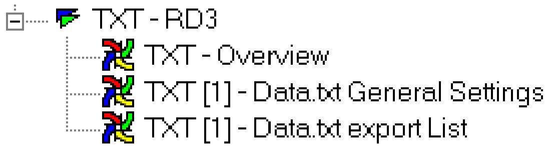
-M, f, F a fixed character

See also:

TTP Plugin

TXT – Text Plugin

After selecting ‘Remote Data’ and TXT – RD xx


TXT - RD3
TXT - Overview
TXT [1] - Data.txt General Settings
TXT [1] - Data.txt export List

There are several items of this remote data:

1. TXT – Data.txt General Settings
2. TXT – Data.txt import/export List
3. TXT - Overview

The Remote Data Txt Plugin is a remote data that does the following functions:

- Place channel data into a text file

- Retrieve channel data from a text file

The Format the data is written to file is:

<Data><Seperator>…<Data><End of line Separator>

….

<Data><Seperator>…<Data><End of line Separator>

Example File, Comma Separated

21-02-2002,15:02:02

345.3,1.3

1800.0,0.0,1000.0,0.0,0.0,0.0,0.0

0.0,0.0,15.0

0.0,0.0,0.0

0.0,0.0,980.0

0.0,0.0,0.0

There is a Special Separator and that is ‘Fixed Width’, this means that instead of a Separator the width of a column and orientation have to be specified.

<Width x, Left Aligned, Data>…< Width y, Right Aligned, Data ><End of line Separator>

…

<Width x, Left Aligned, Data>…< Width y, Right Aligned, Data ><End of line Separator>

Example File, Fixed width separated:

10101 0.1

10102 0.5

10105 25.0

10110 9999.9

Shortcuts

Icon:

Place: Remote Data\

TXT – Overview

Before you can configure Tables you have to add them, there is a Maximum of 32 Tables per Remote Data.

-Double click on the last table entry, or press a numeral key.

-Select a Table number to Add and confirm with Enter

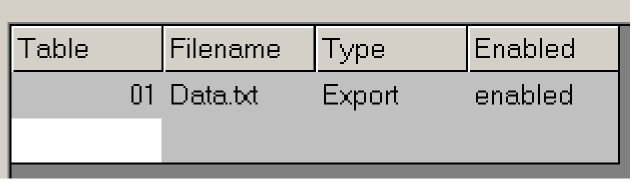


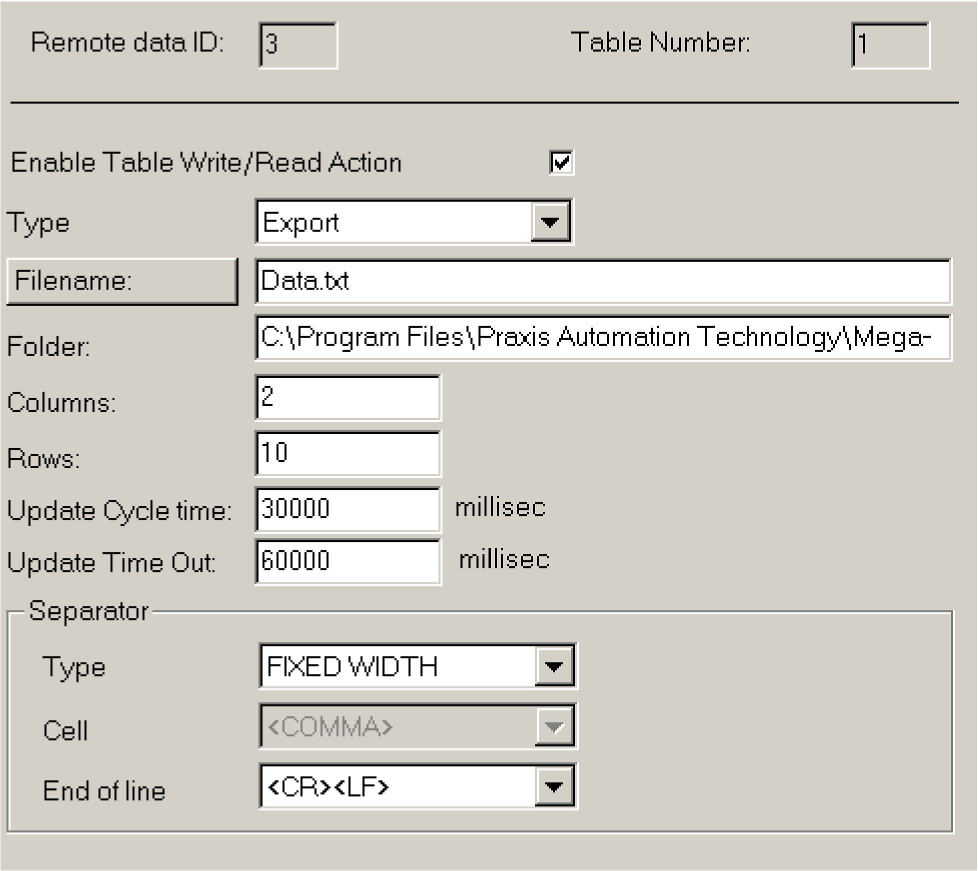
Table	Filename	Type	Enabled
01	Data.txt	Export	enabled

Field	Description
Table	Table number (1-32)
Filename	name of .txt file, which is used
Type	How is the file used, as Export or Import
Enable	Enable or Disable, (Active or Not) Status of configured parts.

See also:

TXT Plugin

TXT – (Data.txt) General Settings



Remote data ID: 3

Table Number: 1

Enable Table Write/Read Action ☑

Type: Export

Filename: Data.txt

Folder: C:\Program Files\Praxis Automation Technology\Mega-

Columns: 2

Rows: 10

Update Cycle time: 30000 millisec

Update Time Out: 60000 millisec

Separator

Type: FIXED WIDTH

Cell: <COMMA>

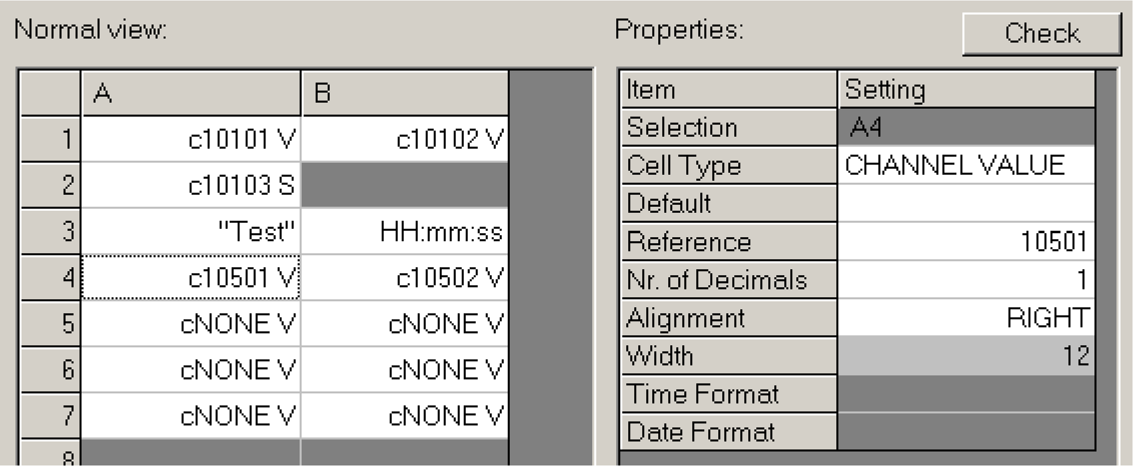
End of line: <CR><LF>

Setting	**Description**
Enabled	Change table creation/processing on/off
Type	Select table type import/export
Filename	Name of the File where data is stored to/ retrieved from.
Folder	Local directory or network map where filename resides.
Columns	Number of Cells Maximal in one row, max 64
Rows	Number of Rows in the table, max 256
Update Cycle Time	Time between read/write action
Update Time out	Time before diagnostic is generated
Separator Type	Select Separator Type Fixed Width or Separator
Separator Cell	Separator to use when type is Separator
Separator end of line	Separator to use between rows

See also:

TXT Plugin

TXT – (Data.txt) Export/Import List



Normal view:

	A	B
1	c10101 V	c10102 V
2	c10103 S	
3	"Test"	HH:mm:ss
4	c10501 V	c10502 V
5	cNONE V	cNONE V
6	cNONE V	cNONE V
7	cNONE V	cNONE V
8		

Properties:

Item	Setting
Selection	A4
Cell Type	CHANNEL VALUE
Default	
Reference	10501
Nr. of Decimals	1
Alignment	RIGHT
Width	12
Time Format	
Date Format	

Import and export lists are the same in buildup, but channel criteria is different.

Select via mouse button click a cell in normal view. Set it's cell properties via the porperties table view.

Normal view:

Field	Description
A	Column 1, show contents of this cell
B	Column 2, show contents of this cell

Properties:

Field	Description
Selection	Current selection in other view
Cell type	Select Cell type: • NONE (when selected in a cell, for this row is 'End of Line') • Channel Status • Channel Value • Text • Date • Time
Default	Channel Related, Text to be written when channel value cannot be retrieved or Cell Width isn’t big enough to show Value.
Reference	Channel Number Table type export: installed channel Table type import: channel with Remote Data Source.
Nr of Decimals	Digits behind Value 0,1,2 or 3
Alignment	Fixed Width: place cell data on left or Right side and fill up with spaces
Width	Fixed Width: Size of Cell data that can be written.

Time Format	Select Time format to use.
Date Format	Select Date format to use.

Criteria for channel cells:
At table type **Import** - Make sure that channel type that supports remote data. If not during configuration channel type will be changed
At table type **Export** - Installed Channel.

Criteria for Text:
At table type **Import** - Do not check content
At table type **Export** - Write Text into file.

Criteria for Date Time:
At table type **Import** - Check date/time falls between (current GMT – cycle time). and current GMT.
At table type **Export** - Write current GMT date/time into file.

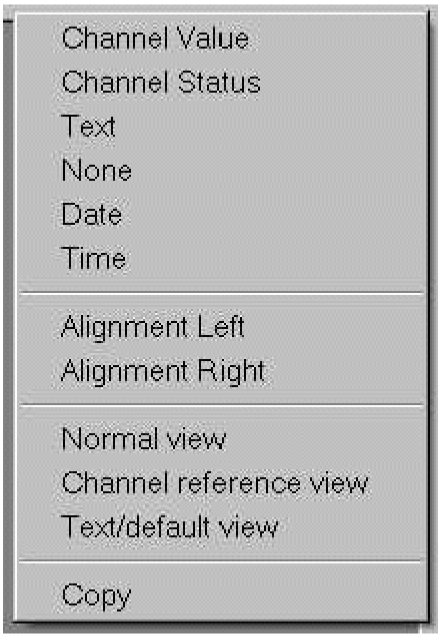
Configuration Tips:

- Select Multiple cells to change properties of multiple cells. In this case channel reference can not be changed.
- Select entire column to change the Width attribute of cells.

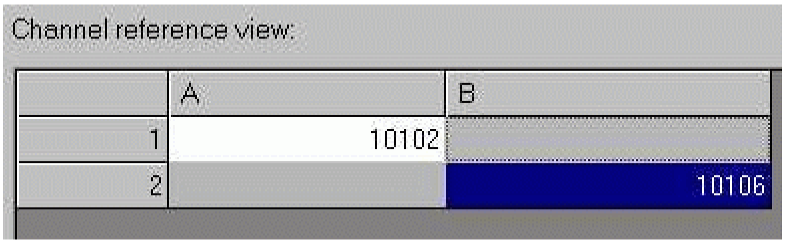
Check Button:
Use Check Button to do a consistency check of the Database.

Return Button:
Return from gateway, only available when getting there from a channel configuration (status/value link)

By pressing right mouse, a menu with many functions is appearing:



Besides 'normal view' there is also so-called 'Channel reference view'.
When activated users would just type in normal channel numbers.



Channel reference view:

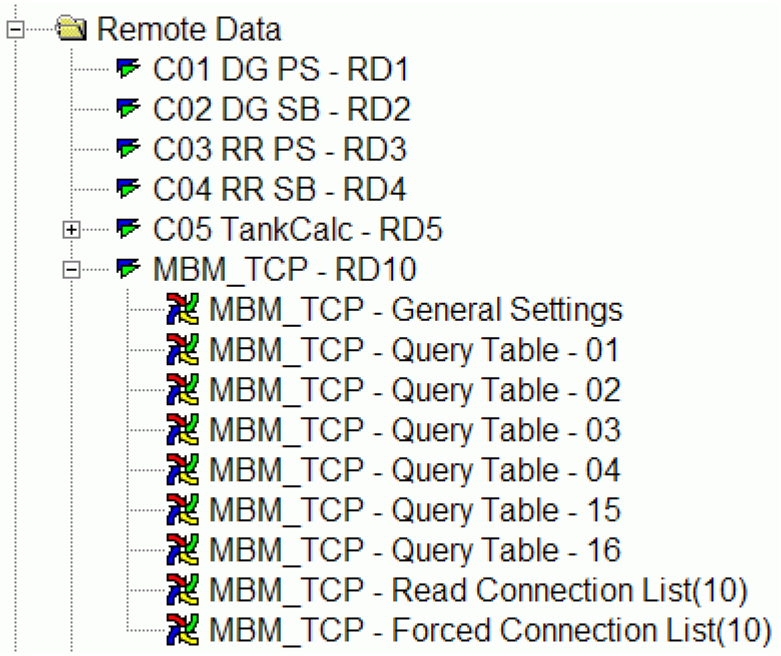
	A	B
1	10102	
2		10106

See also:

TXT Plugin

MBM_TCP – Modbus Master TCP/IP Plugin

After selecting ‘Remote Data’ and MBM_TCP – RD xx



Remote Data
- C01 DG PS - RD1
- C02 DG SB - RD2
- C03 RR PS - RD3
- C04 RR SB - RD4
- C05 TankCalc - RD5
- MBM_TCP - RD10
  - MBM_TCP - General Settings
  - MBM_TCP - Query Table - 01
  - MBM_TCP - Query Table - 02
  - MBM_TCP - Query Table - 03
  - MBM_TCP - Query Table - 04
  - MBM_TCP - Query Table - 15
  - MBM_TCP - Query Table - 16
  - MBM_TCP - Read Connection List(10)
  - MBM_TCP - Forced Connection List(10)

There are several items of this remote data:

1. MBM_TCP - General Settings
2. MBM_TCP – Force Connection List
3. MBM_TCP - Query Table 01..04 or 15..16
4. MBM_TCP – Read Connection List

This remote data supports the following modbus functions:

Inputs:
01 – Coils (Digital, error code 0x81)
02 – Status (Digital, error code 0x82))
03 – Hold Registers (Analog, error code 0x83)
04 – Input Registers (Analog, error code 0x84)

Outputs:
05 – Write Single Coil (Digital, error code 0x85)
06 – Write Single Register (Analog, error code 0x86)
15 - Write Multiple Coils (Digital, error code 0x8F)
16 - Write Multiple registers (Analog, error code 0x90)

Remark: There is no use for a treeview items like "MBM_TCP - Query Table 05" or 06.
Because the number of registers is 1 at function 05 and 06.
In this case Modbus Master will send a complete message with only one status or value.

Structures are sent according Motorola (MSB LSB or Big Endian) processor set.

Register Values are handled depending of configuration:

Register Range	Signed	Unsigned	8000H Offset
Negative	-32768..-1		-32768..-1

	(8000.. FFFF)		(0..7FFF)
Positive	0..32767 (0000..7FFF)	0..65535 (0000..FFFFF)	0..32767 (8000..FFFF)
Error Value (Out of range)	-32768 (8000)	65535 (FFFF)	-32768 (0000)

(Values) use hex notation

For value extension and value precision another types of register handling is available.
The base types of these are:

- long (signed 32 bits)
- ulong (unsigned 32 bits)
- float (32 bits, contains one sign bit and exponent (two complement) and 23 bit mantissa)

Furthermore there could be a distinction between low byte and high byte sending/receiving order. (Abbreviation L/H or H/L) Normally H/L is most frequently used.

Therefore a Modbus register is always 16 bits, two registers addresses are used for a 32 bit value presentation.

MBM TCP/IP message exists of:

MBAP Header	Function Number	Data

This message provides some differences compared to the MODBUS RTU application data unit used on serial line:

- The MODBUS ‘slave address’ field usually used on MODBUS Serial Line is replaced by a single byte ‘Unit Identifier’ within the MBAP Header. The ‘Unit Identifier’ is used to communicate via devices such as bridges, routers and gateways that use a single IP address to support multiple independent MODBUS end units.

- All MODBUS requests and responses are designed in such a way that the recipient can verify that a message is finished. For function codes where the MODBUS PDU has a fixed length, the function code alone is sufficient. For function codes carrying a variable amount of data in the request or response, the data field includes a byte count.

- When MODBUS is carried over TCP, additional length information is carried in the MBAP header to allow the recipient to recognize message boundaries even if the message has been split into multiple packets for transmission. The existence of explicit and implicit length rules, and use of a CRC-32 error check code (on Ethernet) results in an infinitesimal chance of undetected corruption to a request or response message.

MBAP Header

Fields	Length	Description	Client	Server
Transaction Identifier	2 Bytes	Identification of a MODBUS Request / Response transaction.	Initialized by the client	Recopied by the server from the received request
Protocol Identifier	2 Bytes	0 = MODBUS protocol	Initialized by the client	Recopied by the server from the received request
Length	2 Bytes	Number of following bytes	Initialized by the client (request)	Initialized by the server (Response)

Unit Identifier	1 Byte	Identification of a remote slave connected on a serial line or on other buses.	Initialized by the client	Recopied by the server from the received request

The header is 7 bytes long:

• Transaction Identifier - It is used for transaction pairing, the MODBUS server copies in the response the transaction identifier of the request.

• Protocol Identifier – It is used for intra-system multiplexing. The MODBUS protocol is identified by the value 0.

• Length - The length field is a byte count of the following fields, including the Unit Identifier and data fields.

• Unit Identifier – This field is used for intra-system routing purpose. It is typically used to communicate to a MODBUS+ or a MODBUS serial line slave through a gateway between an Ethernet TCP-IP network and a MODBUS serial line. This field is set by the MODBUS Client in the request and must be returned with the same value in the response by the server.

Example Function 02 digital status request

MBAP Header (seven bytes)	Function Nr (one byte)	Address (two bytes)	Nr of statusses (two bytes)
header	0x02	0x01 0x00	0x00 0x09

Example Function 02 digital status reply

MBAP Header (seven bytes)	Function Nr (one byte)	Byte Count (one byte)	Data (Byte Count bytes)
header	0x02	0x02	0xFF 0xFF

Example Function 02 digital status exception reply

MBAP Header (seven bytes)	Function Nr (one byte)	Exception code (one byte)
header	0x**82**	**0x02**

To get the status of the points with MODBUS address 100 to 113, when e.g. the points on address 103 and 108 are 'on' and the other points are 'off':

For more information reference the Modbus Specification:
Modbus_Messaging_Implementation_Guide_V1_0b.pdf

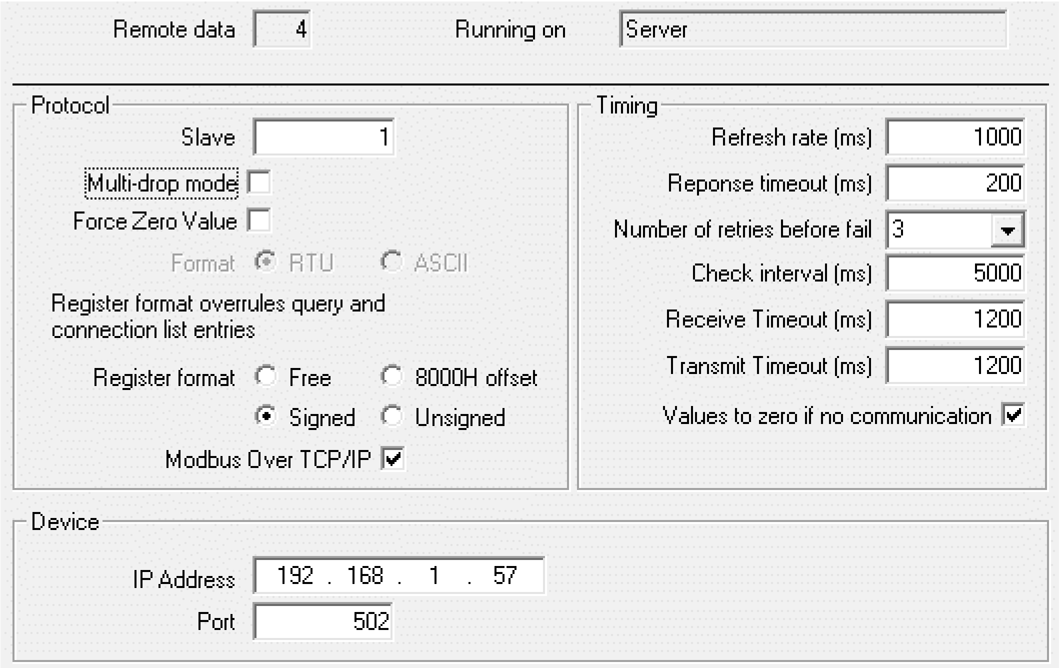
See http://www.modbus.org/.

Shortcuts

Icon:

Place: Remote Data\

MBM_TCP – General Settings



Remote data 4
Running on Server

Protocol

Slave 1
Multi-drop mode □
Force Zero Value □
Format ● RTU ○ ASCII
Register format overrules query and connection list entries
Register format ○ Free ○ 8000H offset
● Signed ○ Unsigned
Modbus Over TCP/IP ☑

Timing

Refresh rate (ms) 1000
Reponse timeout (ms) 200
Number of retries before fail 3
Check interval (ms) 5000
Receive Timeout (ms) 1200
Transmit Timeout (ms) 1200
Values to zero if no communication ☑

Device

IP Address 192 . 168 . 1 . 57
Port 502

Header settings

Field	Options/Values and Function
Remote data	Unique Number of selected Remote Data Plugin
Running on	Server: Protocol runs on the Server as a Plug-in Board (LBB-LBB): Protocol runs on Processor board (L = Link, BB = Board number) eq. Board (101-108)

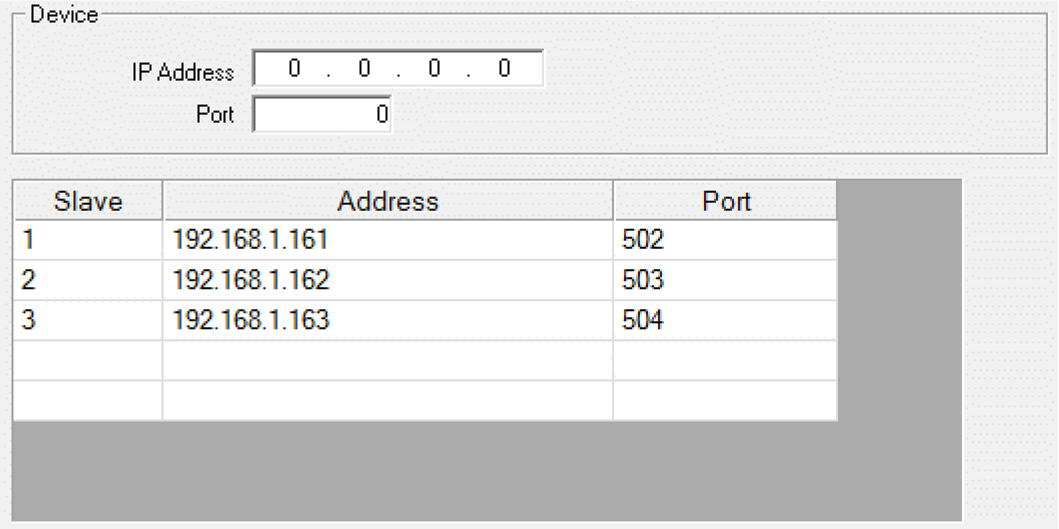
Protocol settings

<table>
<tr><th>Field</th><th>Options/Values and Function</th></tr>
<tr><td><b>Slave</b></td><td>Single slave number used for all queries.
(default 1)
<i>The “SLAVE” field in the connection list will be updated to this number for all query blocks.</i></td></tr>
<tr><td><b>Multi-drop mode</b></td><td>When Multi-drop is enable the Slave number will be ignored and
multiple slave numbers can be entered via the Slave / Address / Port Configuration list.
(default unchecked/off)</td></tr>
<tr><td><b>Force Zero Value</b></td><td>Force registers values to zero for functions (05/06 or 15/16) when channels are not available or the registers are not configured
(default unchecked/off)</td></tr>
<tr><td><b>Format</b></td><td>RTU: binary data exchange (Device 8 databits)
ASCII: text data exchange (Device 7 databits)
(default RTU)</td></tr>
<tr><td><b>Modbus over TCP/IP</b></td><td>Modbus RTU message transmitted with a TCP/IP wrapper and sent over a network instead of serial comport.
The Server uses an IP Address therefore a SlaveID is not needed.
(default unchecked/off)</td></tr>
<tr><td><b>Register Handling</b></td><td>Options for Registers Handling:
<table>
<tr><th>Value</th><th>Function</th></tr>
<tr><td><b>Free</b></td><td>Query or Connection list Register handling is used</td></tr>
</table></td></tr>
</table>

Signed	Negative/Positive values (Sign bit)
Unsigned	Positive values
8000H Offset	zero point becomes 0x08000 (no Sign bit)

This overrules configuration of this Field in Queries and connection list.
(default Signed)

Example: Multi-drop Mode



IP Address 0 . 0 . 0 . 0

Port 0

Slave	Address	Port
1	192.168.1.161	502
2	192.168.1.162	503
3	192.168.1.163	504

Device settings

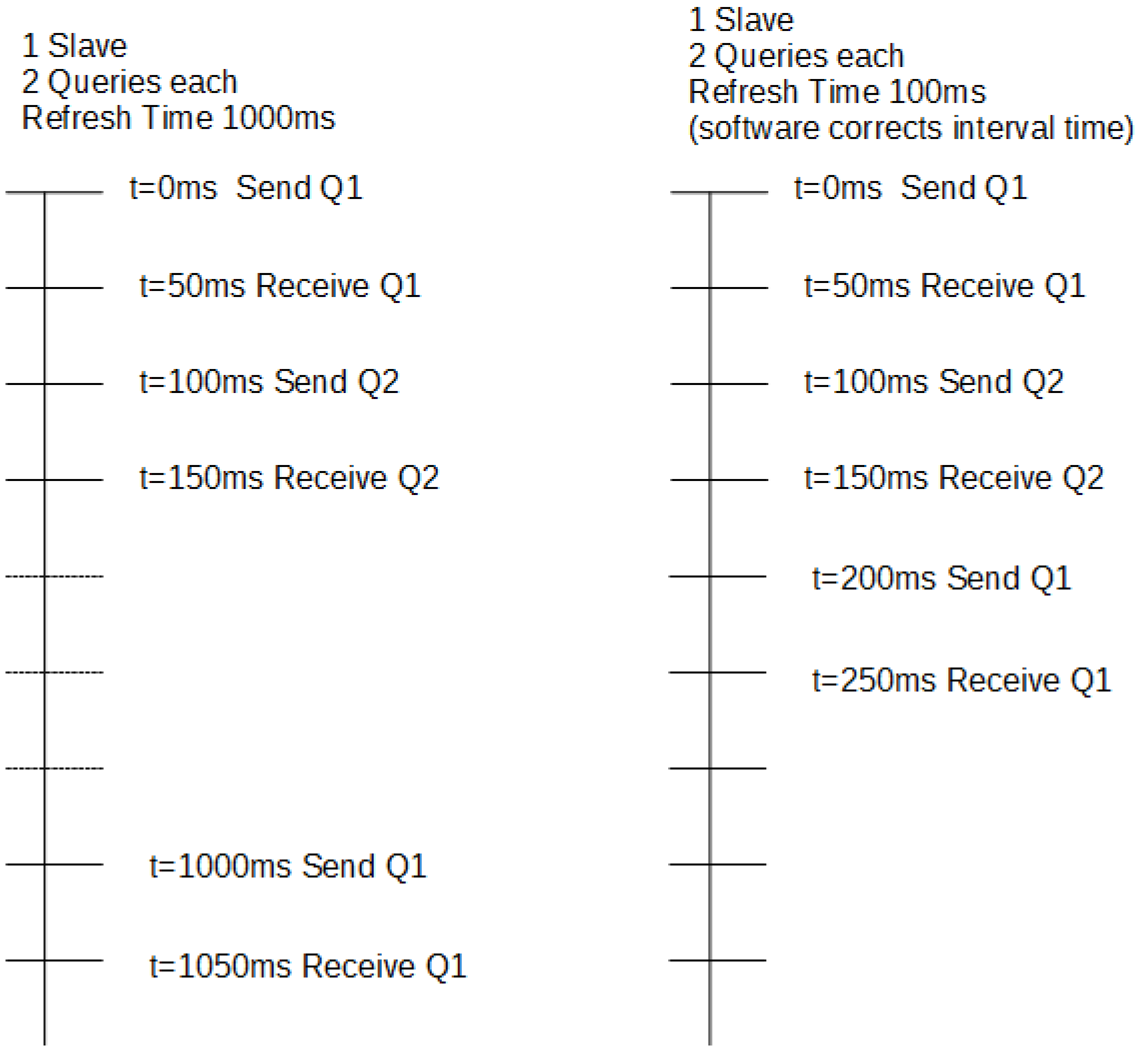
Field	Options/Values and Function
IP Address	Network Address of Modbus Machine
Port	communication port for Ethernet usage (default: 502)

Timing settings

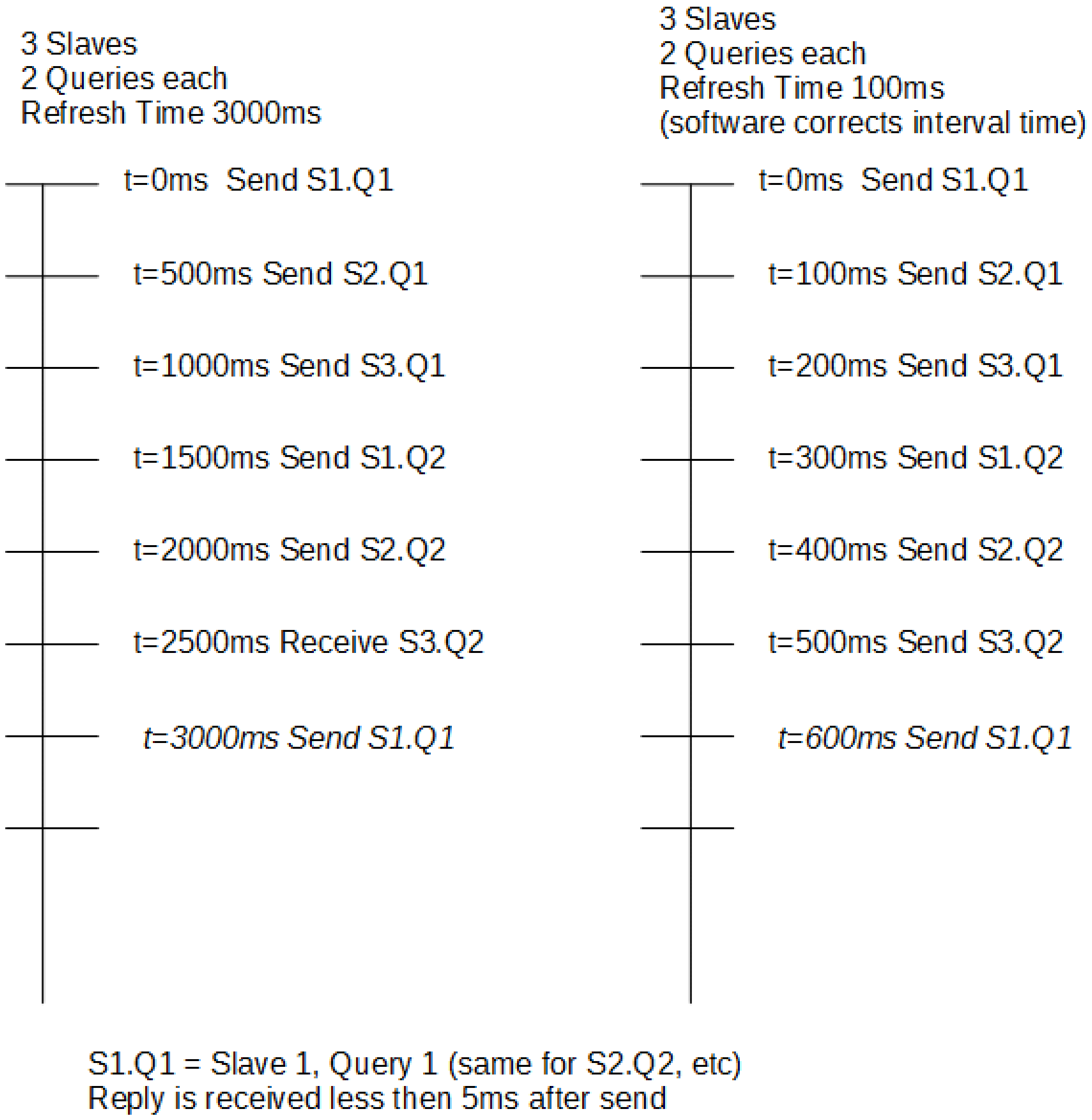
Field	Options/Values and Function
Refresh Rate	Update cycle time for one time execution of all configured queries. Each query takes 100ms. Low refresh rate will increase load on network and CPU. When refresh rate is minimal time required to transmit and receive all queries once. If a lower number is entered it will not be used. The refresh time must be equal or larger than required time for the queries. For example: 10 queries take 10 x 100ms = 1000ms => Refresh time must be 1000ms or larger. Other CPU Tasks will also consume time which needs to be taken into account. The modbus update cycle can be delayed because of this. For example an 1131 program with execution time of more 50 ms will slowdown MBM_TCP. XP will run 20 cycles per second (if it is not overloaded). So one cycle per 50ms. A request is send out every cycle, a Reply is received and handled next cycle. Because of this one query takes at least 100 ms. **When using multi-drop mode on XP some software optimizations were implemented** **which speed up MBM_TCP process.**

	200-300000 ms (default: 1000)
Response Time Out	Maximum time between a message request and reply. When this time expires the retry sequence starts. Check Modbus Slave Manual from the Slave device manufacturer for response time as an minimal of this value. Set it to larger time as indicated to prevent faults and false retries. For example: if Slave response time is 100ms then set Response Time Out to 200ms or higher. 50-5000 ms (default: 200)
Number Of Retries Before Failure	The number of times after each other a timeout has been occured. This will generate a diagnostic: 'Slave Not Present'. 0 - 10 (default: 3) Remark: only when the sending goes well, in that case there is a connection. 1) TCP Master(=Client) sends a request 2) TCP Slave(=Server) receives request 3) TCP Slave(=Server) sends an ACK 4) ACK is received by TCP Master(=Client) 5) but further on there is no answer is coming on the request. In General: no communication diagnostic comes after 5 sec (hard-coded) when is no connection at all. In that case the sending part is failed, no ACK is received
Check Interval	Time between queries for checking Slaves to see if they are present. If a Slave is not present, this time will be waited before a Slave will be retried. 200-300000 millisec (default: 5000)
Receive Timeout (ms)	Defines maximum time the process will wait for answer before triggering a communication error. Used via Windows TCP Socket Setting. Not used on XP. 200-300000 millisec (default: 5000)
Transmit Timeout (ms)	Defines maximum time the process will wait for confirmation (Acknowledgement) before triggering a communication error. Used on Windows as TCP Socket Setting. Not used on XP. 200-300000 millisec (default: 5000)
Values to zero if no communication	When no communication error occurs; all channel values are set to zero. Involved all channels which are set with function 01/02/03/04. On/Off (default unchecked/off)

An Internet Protocol address (IP address) is a numerical label assigned to each device (e.g., computer, printer) participating in a computer network that uses the Internet Protocol for communication



Single Slave Mode: Time Diagrams

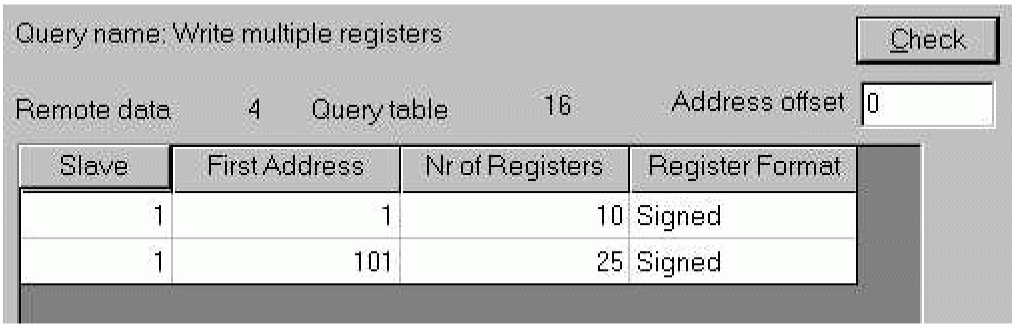


Multi-drop Mode: Time Diagrams

See also:

MBM_TCP - General Information Register Handling etc.
MBM_TCP - Query Table
MBM_TCP – Read Connection List
MBM_TCP – Force Connection List

MBM_TCP –Query Table



Query name: Write multiple registers

Remote data 4 Query table 16 Address offset 0

Slave	First Address	Nr of Registers	Register Format
1	1	10	Signed
1	101	25	Signed

Grid Column	Options/Values and Function
Slave Nr	Slave Number, can be forced by General Settings, 1..255
First Address	Address value, 0..65535
Nr Of Registers	Number of registers to be received or be sent, when too many registers are queried multiple queries are automatically generated, 0..65535
Register Handling	How is register handled, can be forced by General Settings - 8000H Offset means zero point becomes 0x08000 Signed, Unsigned, 8000H Offset or Free (default Signed) Long (H/L), Long (L/H), ULong (H/L), ULong (LH), Float (H/L), Float (L/H)

By pressing on column “Slave Nr” a sort action is performed, first time ascending second descending.

Field/Control	Function
Query Table	Modbus function number of shown queries
Address Offset	Offset so Configured Modbus addresses are in accordance with the Modbus list and queries with hardware Modbus addresses used, certain plc’s send 40001 as 0 and 10001 as 0., 0..65535
Check Button	Check if all connection list items are inside the configured modbus queries

Remark: There is no use for a treeview items like "MBM_TCP - Query Table 05" or 06.
Because the number of registers is 1 at function 05 and 06.
In this case only configuration at force connection list is sufficient.

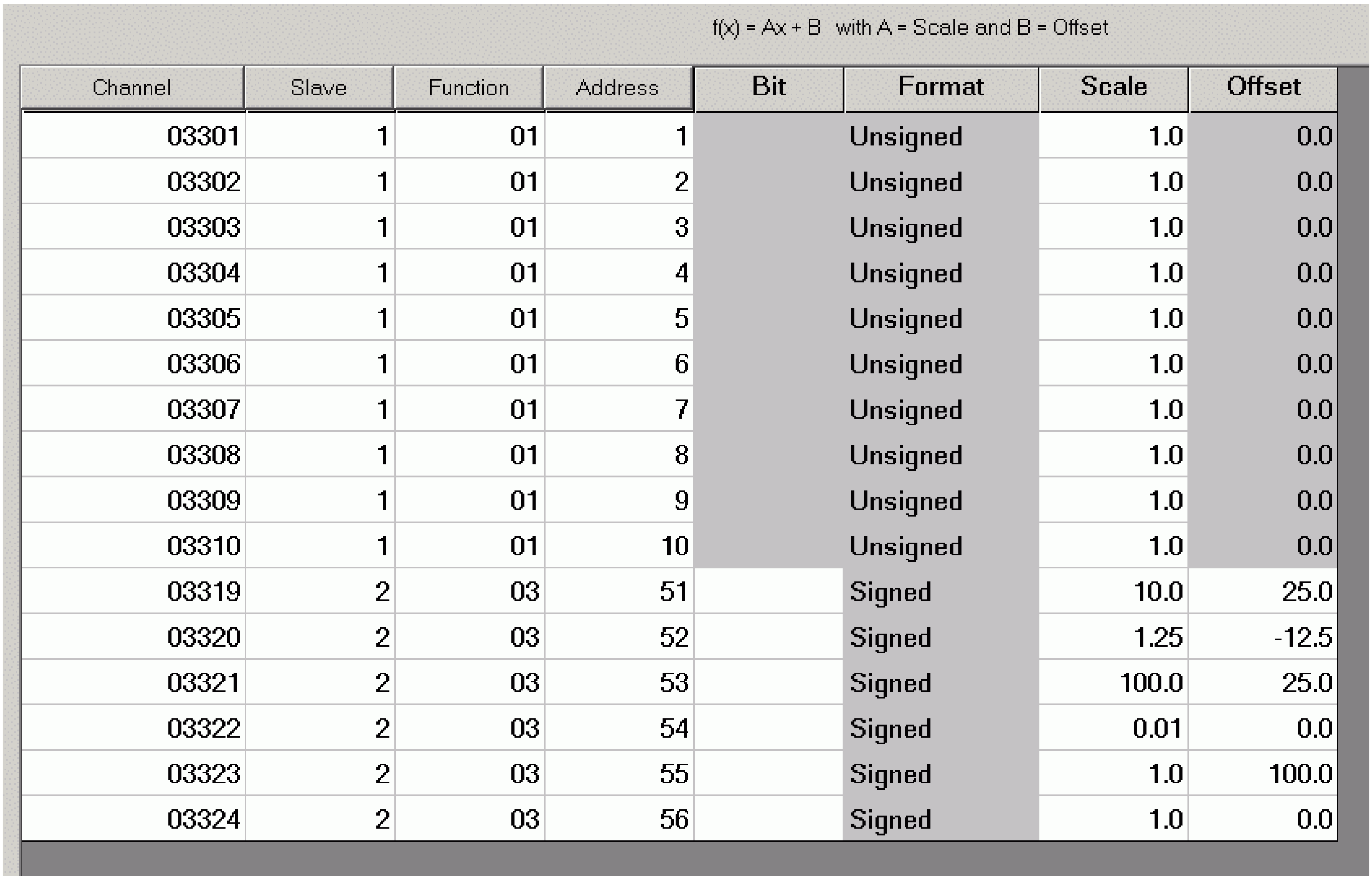
See also:

MBM_TCP - General Settings

MBM_TCP – Force Connection List

MBM_TCP – Read Connection List

MBM_TCP – Read Connection List



f(x) = Ax + B   with A = Scale and B = Offset

Channel	Slave	Function	Address	Bit	Format	Scale	Offset
03301	1	01	1		Unsigned	1.0	0.0
03302	1	01	2		Unsigned	1.0	0.0
03303	1	01	3		Unsigned	1.0	0.0
03304	1	01	4		Unsigned	1.0	0.0
03305	1	01	5		Unsigned	1.0	0.0
03306	1	01	6		Unsigned	1.0	0.0
03307	1	01	7		Unsigned	1.0	0.0
03308	1	01	8		Unsigned	1.0	0.0
03309	1	01	9		Unsigned	1.0	0.0
03310	1	01	10		Unsigned	1.0	0.0
03319	2	03	51		Signed	10.0	25.0
03320	2	03	52		Signed	1.25	-12.5
03321	2	03	53		Signed	100.0	25.0
03322	2	03	54		Signed	0.01	0.0
03323	2	03	55		Signed	1.0	100.0
03324	2	03	56		Signed	1.0	0.0

Grid Column	**Options/Values and Function**
Channel	Channel to receive the data; type in an existing digital/analog channel number, or get here from the channel's configuration form (**status/value link**)
Slave Nr	Slave Number, can be forced by General Settings, 1..255
Function	Modbus function number, 01,02,03 or 04
Address	Modbus address of modbus function where value is retrieved from. 0...65535
Register Bit	Bit to extract from register 0..15 or (nothing) where (nothing) means 'no bit' or entire register
Register Handling	How is register handled - 8000H Offset means zero point becomes 0x08000 Signed, Unsigned, 8000H Offset or Free (default Signed) Long (H/L), Long (L/H), ULong (H/L), ULong (LH), Float (H/L), Float (L/H)
Scale	Scale factor of send register values examples: 1000, 100, 10, 1, 0.1, 0.01, 0.001 (Default 1.0) for digital channel only "-1.0" for inverse value
Offset	offset factor of send register values examples: 10.0 or 1000 (Default 0.0)

By pressing on column "Channel" or "Slave" or "Function" or "Address" a sort action is performed. First time ascending second time descending.

Return Button: Return from gateway, only available when getting there from a channel configuration (status/value link)

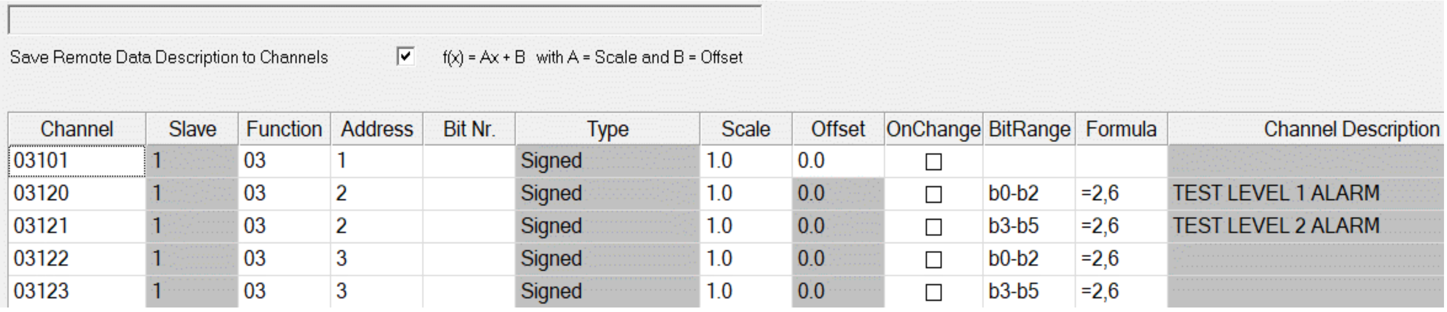
Tip: For fast channel configuration, configure last line complete, press not enter at the last column, go back to first column, give a channel range in (like 10101-10125) and press enter, now multiple rows are created!

Slave field in single slave mode will not be editable and show general slave number.

Register-handling field is non-editable when overruling Register Handling is active this is when:
- General Register Handling isn't Free
- Queries Register Handling for the Modbus Slave/Function where Modbus Address is part of isn't free
- Digital status/coil or binary packed register

Scale 1.0 and Offset 0.0 means rounded whole channel values are send.
Scale 10.0 and Offset 0.0 means rounded whole 'channel values*10' are send.
Scale 0.1 and Offset 0.0 means rounded whole 'Channel values/10' are send.
Scale 10.0 and Offset 1000.0 means rounded whole 'channel values*10 + 1000' are send.

New fields are added.



Save Remote Data Description to Channels ☑ f(x) = Ax + B with A = Scale and B = Offset

Channel	Slave	Function	Address	Bit Nr.	Type	Scale	Offset	OnChange	BitRange	Formula	Channel Description
03101	1	03	1		Signed	1.0	0.0	□			
03120	1	03	2		Signed	1.0	0.0	□	b0-b2	=2,6	TEST LEVEL 1 ALARM
03121	1	03	2		Signed	1.0	0.0	□	b3-b5	=2,6	TEST LEVEL 2 ALARM
03122	1	03	3		Signed	1.0	0.0	□	b0-b2	=2,6	
03123	1	03	3		Signed	1.0	0.0	□	b3-b5	=2,6	

Grid Column	Options/Values and Function
On Change	When same Channel is added to a forced and read connection list, this field should be checked if other side is changing the value too.
Bit Range	to select specific bit range of a value (b0-b15), when selecting b3-b5 the result value is right shifted example: value 48(%00110000) right shift 3 gives 6(%0110)

Formula	in combination with bitrange, options are =, <, >, <=, >=, < >
	equation gives digital result (true of false)
	example =2,6 can be seen as:
	IF value = 2 OR value = 6 THEN
	  Channel is true
	ELSE
	  Channel is false

Channel Description	description from channel layout to know where channel is put/read from

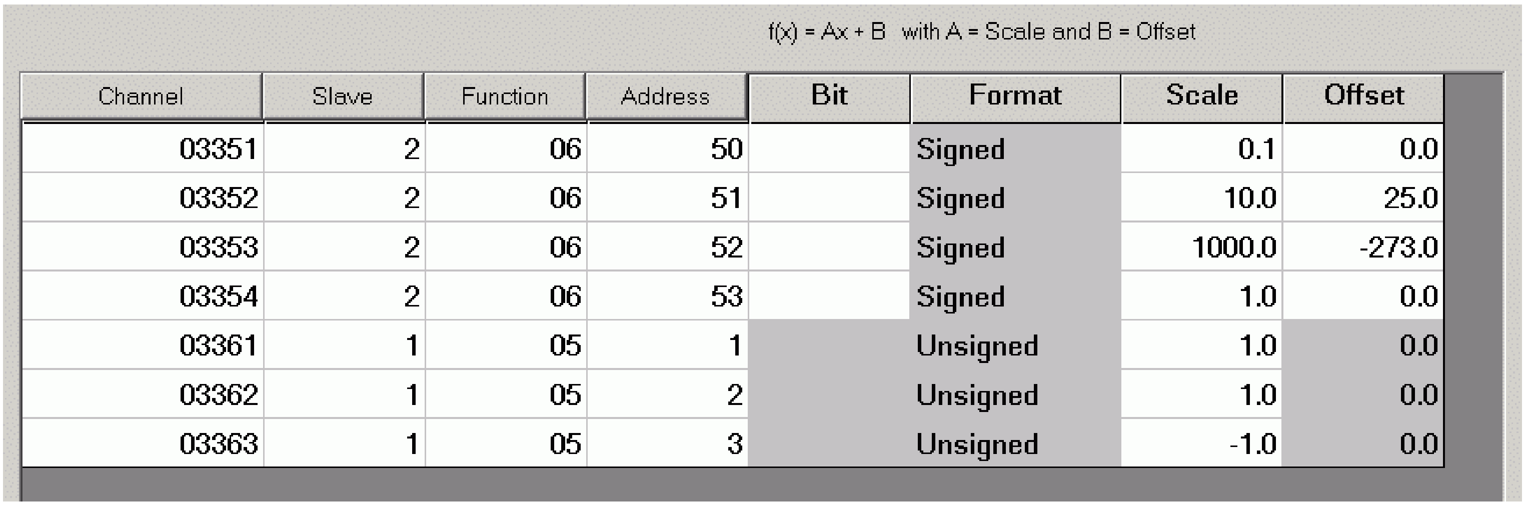
See also:

MBM_TCP - General Settings

MBM_TCP – Force Connection List

MBM_TCP - Query Table

MBM_TCP – Forced Connection List



f(x) = Ax + B with A = Scale and B = Offset

Channel	Slave	Function	Address	Bit	Format	Scale	Offset
03351	2	06	50		Signed	0.1	0.0
03352	2	06	51		Signed	10.0	25.0
03353	2	06	52		Signed	1000.0	-273.0
03354	2	06	53		Signed	1.0	0.0
03361	1	05	1		Unsigned	1.0	0.0
03362	1	05	2		Unsigned	1.0	0.0
03363	1	05	3		Unsigned	-1.0	0.0

Grid Column	Options/Values and Function
Channel	Channel to send the data; type in an existing digital/analog channel number, or get here from the channel's configuration form (**status/value link**)
Slave Nr	Slave Number, can be forced by General Settings, 1..255
Function	Modbus function number, 05,06,15 or 16
Address	Modbus address of modbus function where value is send to. 0...65535
Register Bit	Bit in register 0..15 or (nothing) where (nothing) means 'no bit' or entire register
Register Handling	How is register handled, can be forced by General Settings or Query Settings - 8000H Offset means zero point becomes 0x08000 Signed, Unsigned, 8000H Offset or Free (default Signed) Long (H/L), Long (L/H), ULong (H/L), ULong (LH), Float (H/L), Float (L/H)
Scale	Scale factor of send register values examples: 1000, 100, 10, 1, 0.1, 0.01, 0.001 (Default 1.0) for digital channel only "-1.0" for inverse value
Offset	offset factor of send register values examples: 10.0 or 1000 (Default 0.0)

By pressing on column "Channel" or "Slave" or "Function" or "Address" a sort action is performed. First time ascending second time descending.

Return Button: Return from gateway, only available when getting there from a channel configuration (status/value link)

Tip: For fast channel configuration, configure last line complete, press not enter at the last column, go back to first column, give a channel range in (like 10101-10125) and press enter, now multiple rows are created!

Slave field in single slave mode will not be editable and show general slave number.

Register-handling field is non-editable when overruling Register Handling is active this is when:
- General Register Handling isn't Free
- Queries Register Handling for the Modbus Slave/Function where Modbus Address is part of isn't free
- Digital status/coil or binary packed register

Scale 1.0 and Offset 0.0 means rounded whole channel values are send.
Scale 10.0 and Offset 0.0 means rounded whole 'channel values*10' are send.

Scale 0.1 and Offset 0.0 means rounded whole 'Channel values/10' are send.
Scale 10.0 and Offset 1000.0 means rounded whole 'channel values*10 + 1000' are send.

See also:

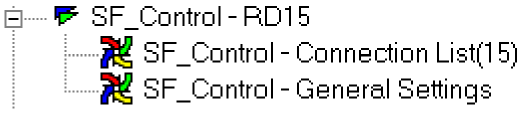
MBM_TCP - General Settings

MBM_TCP - Query Table

MBM_TCP – Read Connection List

SF Control Plugin

After selecting ‘Plugins’ and SF Control – RD xx



SF_Control - RD15
SF_Control - Connection List(15)
SF_Control - General Settings

There are several items of this plugin:

1. SF Control - General Settings

2. SF Control - Connection List

This protocol is based on LevelDatic 100s. Electropneumatic level measurement system from the company with the name SF-Control.

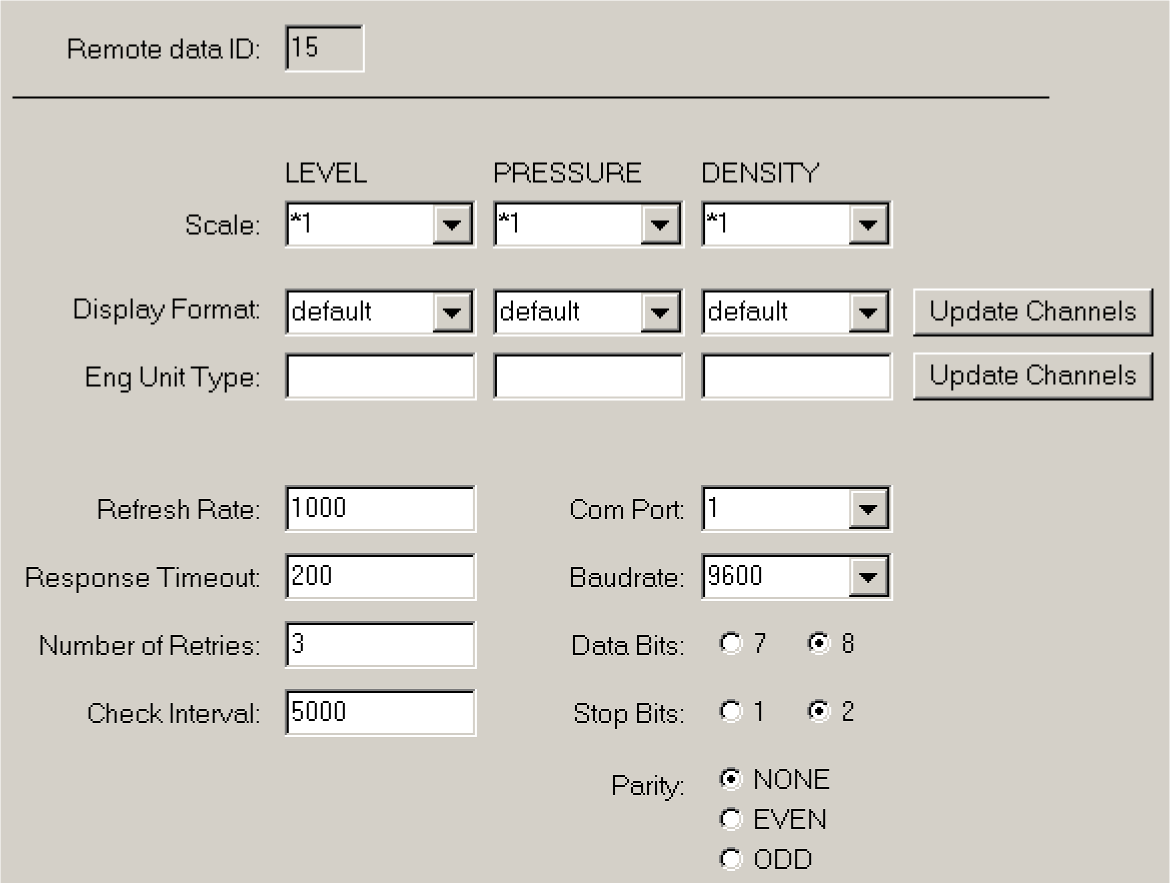
This is using for gauging systems

Shortcuts

Icon:

Place: Plugins\

SF Control – General Settings



Remote data ID: 15

	LEVEL	PRESSURE	DENSITY
Scale:	*1	*1	*1
Display Format:	default	default	default
Eng Unit Type:			

Refresh Rate: 1000

Response Timeout: 200

Number of Retries: 3

Check Interval: 5000

Com Port: 1

Baudrate: 9600

Data Bits: ○ 7 ● 8

Stop Bits: ○ 1 ● 2

Parity: ● NONE ○ EVEN ○ ODD

Com Port Communication port which is used for sending the output. No other plug-in must have the same COM port (only if plugin is on running on server)

Baudrate Baudrate of Com Port, 1200, 2400, 4800, 9600 or 19200 (default: 19200).

Refresh Rate 200-300000 millisec (default: 1000)

Response Time Out 50-5000 millisec (default: 200)

Data Bits Com. Port setting: 7 or 8 (default: 8)

Stop Bits Com. Port setting: 1 or 2 (default: 1)

Parity Com. Port setting: None, Odd or Even (default: None)

Check Interval 200-300000 millisec (default: 5000)

Number Of Retries Before Failure 0-10 (default: 3)

Level- / Pressure- / Density Scaling "/1000" – "*100" (default: "*1")

Three additional special input fields are available to set the channel display format for each type:

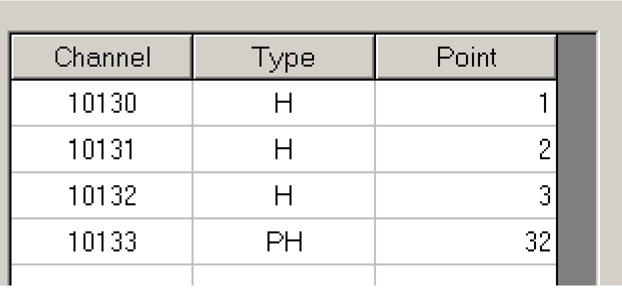
Set Level / Set Pressure / Set Density Display Format:

"X.XXXXX" – "xxxxxxX" (default: "xxxxX.X")

See also:

SF Control Plugin

SF Control – Connection List



Channel	Type	Point
10130	H	1
10131	H	2
10132	H	3
10133	PH	32

Field	Description
Channel	Channel to receive the data; type in an existing analog channel number, or get here from the channel's configuration form (**value link**)
Type	Type, H, PH, D, SD
Point	Point, range 1..999

By pressing on column "Channel" or "Type" or "Point" a sort action is performed, first time ascending second descending.

Command for asking output level message:

<ESC>SEND xx X<CR>

X is taken from the "TYPE" field of the current point

The cabinet number xx is calculated from the current point:

1+ENTIER((Point–1)/10)

Example: <ESC>SEND_03_H<CR>
Send level data message from cabinet 3. After SEND command cabinet 3 will answer the following:
Output level message (unit = meters or feets)

```
<CR><LF>
C03_Hm_1013.3_--PPPPPPPPPPP*<CR><LF>
H021/_4.01_H022/15.27_H023/10.09_H024/_5.98_H025/22.37_<CR><LF>
H026/11.34_H027/12.78_H028/_9.54_H029/28.28_H030/_0.07_<CR><LF>
#<CR><LF>
```

Explanations :

<CR><LF> = Line clear (2 marks, HEX 0D (carriage return), HEX 0A (line feed)).

C03_Hm_1013.3_--PPPPPPPPPPP*<CR><LF> = LD 100S status line (30 marks).

H021/_4.01_H022/15.27_H023/10.09_H024/_5.98_H025/22.37_<CR><LF> =
First LD 100S measurement data line (57 marks).

H026/11.34_H027/12.78_H028/_9.54_H029/28.28_H030/_0.07_<CR><LF> =
Second LD 100S measurement data line (57 marks).

#<CR><LF> = End of message (3 marks HEX 23, HEX 0D, HEX 0A)

Following types are supported:

Level = H
Pressure= PH
Density = D
Setting Density = SD

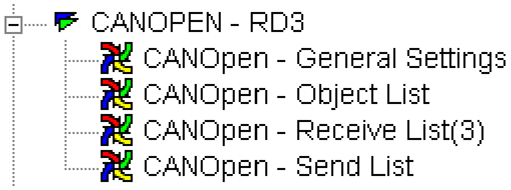
H,PH, D are values from received from Level Datic 100S system.
SD are values send to Level Datic 100S system.

See also:

SF Control Plugin

CANOpen – Plugin

After selecting 'Plugins' and CANOpen – RD xx



CANOPEN - RD3
CANOpen - General Settings
CANOpen - Object List
CANOpen - Receive List(3)
CANOpen - Send List

There are several items of this plugin:

1. CANOpen - General Settings
2. CANOpen - Object List
3. CANOpen - Receive List
4. CANOpen - Send List

Introduction (from Wikipedia)

CANopen is a communication protocol and device profile specification for embedded systems used in automation.
In terms of the OSI model, CANopen implements the layers above and including the network layer.
The CANopen standard consists of an addressing scheme, several small communication protocols and an application layer defined by a device profile.
The communication protocols have support for network management, device monitoring and communication between nodes,
including a simple transport layer for message segmentation/desegmentation.
The lower level protocol implementing the data link and physical layers is usually Controller Area Network (CAN), although devices using some other means of communication (such as Ethernet Powerlink, EtherCAT) can also implement the CANopen device profile.

The basic CANopen device and communication profiles are given in the CiA 301 specification released by CAN in Automation.
Profiles for more specialized devices are built on top of this basic profile,
and are specified in numerous other standards released by CAN in Automation,
such as CiA 401 for I/O-modules and CiA 402 for motion control.

Device model

Every CANopen device has to implement certain standard features in its controlling software.

- A **communication** unit implements the protocols for messaging with the other nodes in the network.

- Starting and resetting the device is controlled via a **state machine**. It must contain the states Initialization, Pre-operational, Operational and Stopped. The transitions between states are made by issuing a network management (NMT) communication object to the device.

- The **object dictionary** is an array of variables with a 16-bit index. Additionally, each variable can have an 8-bit subindex.
The variables can be used to configure the device and reflect its environment, i.e. contain measurement data.

- The application part of the device actually performs the desired function of the device, after the state machine is set

to the operational state.
The application is configured by variables in the object dictionary and the data are sent and received through the communication layer.

Object dictionary

CANopen devices must have an object dictionary, which is used for configuration and communication with the device. An entry in the object dictionary is defined by:

Index, the 16-bit address of the object in the dictionary

Object name, a symbolic type of the object in the entry, such as an array, record, or simple variable

Name, a string describing the entry

Type, gives the datatype of the variable (or the datatype of all variables of an array)

Attribute, which gives information on the access rights for this entry, this can be read/write, read-only or write-only

The **Mandatory/Optional** field (M/O) defines whether a device conforming to the device specification has to implement this object or not

The basic datatypes for object dictionary values such as booleans, integers and floats are defined in the standard (their size in bits is optionally stored in the related type definition, index range 0x0001-0x001F),
as well as composite datatypes such as strings, arrays and records (defined in index range 0x0040-0x025F).
The composite datatypes can be subindexed with an 8-bit index;
the value in subindex 0 of an array or record indicates the number of elements in the data structure, and is of type UNSIGNED8.

For example, the device communication parameters, standardized in the basic device profile CiA 301[4] are mapped in the index range 0x1000-0x1FFF ("communication profile area").

The first few entries in this area are as follows:

Index	Object name	Name	Type	Attribute	M/O
0x1000	VAR	device type	UNSIGNED32	ro	M
0x1001	VAR	error register	UNSIGNED8	ro	M
..	-	-	-	-	-
0x1008	VAR	manufacturer device name	Vis-String	const	O

Given suitable tools, the content of the object dictionary of a device, based on an electronic data sheet (EDS),
can be customized to a device configuration file (DCF) to integrate the device into a specific CANopen network.
According to CiA 306, the format of the EDS-file is the INI file format.
There is an upcoming XML-style format, that is described in CiA 311.

Communication objects

CAN bus, the data link layer of CANopen, can only transmit short packages consisting of an 11-bit id, a remote transmission request (RTR) bit and 0 to 8 bytes of data.

The CANopen standard divides the 11-bit CAN frame id into a 4-bit function code and 7-bit CANopen node ID.

This limits the number of devices in a CANopen network to 127 (0 being reserved for broadcast).

An extension to the CAN bus standard (CAN 2.0 B) allows extended frame ids of 29 bits, but in practice CANopen networks big enough to need the extended id range are rarely seen.

In CANopen the 11-bit id of a CAN-frame is known as communication object identifier, or **COB-ID.**

In case of a transmission collision, the bus arbitration used in the CAN bus allows the frame with the smallest id to be transmitted first and without a delay.
Using a low code number for time critical functions ensures the lowest possible delay.

Contents of a standard CANopen frame:

-	COB-ID	RTR	Data length	Data
Length	11 bits	1 bit	4 bits	0-8 bytes

The default COB-ID mapping sorts frames by attributing a function code (NMT, SYNC, EMCY, PDO, SDO...) to the first 4 bits, so that critical functions are given priority.
This mapping can however be customized for special purposes (except for NMT and SDO, required for basic communication).

-	Function code	Node ID
Length	4 bits	7 bits

The standard reserves certain COB-IDs to network management and SDO transfers.
Some function codes and COB-IDs have to be mapped to standard functionality after device initialization, but can be configured for other uses later.

Communication models

Different kinds of communication models are used in the messaging between CANopen nodes.

- In a **master/slave** relationship, one CANopen node is designated as the master, which sends or requests data from the slaves.
The NMT protocol is an example of a master/slave communication model.

- A **client/server** relationship is implemented in the SDO protocol, where the SDO client sends data (the object dictionary index and subindex) to an SDO server,
which replies with one or more SDO packages containing the requested data (the contents of the object dictionary at the given index).

- A **producer/consumer** model is used in the Heartbeat and Node Guarding protocols.
In the push-model of producer/consumer, the producer sends data to the consumer without a specific request, whereas in the pull model, the consumer has to request the data from the producer.

Electronic Data Sheet

Electronic Data Sheet (EDS) is a file format, defined in CiA306, that describes the communication behaviour and the object dictionary entries of a device.
This allows tools such as service tools, configuration tools, development tools, and others to handle the devices properly.

Those EDS files are mandatory for passing the CiA CANopen conformance test. A free EDS checker is CANchkEDS.

Glossary of CANopen terms

PDO: Process Data Object - Inputs and outputs. Values of type rotational speed, voltage, frequency, electric current, etc.

SDO: Service Data Object - Configuration settings, possibly node ID, baud rate, offset, gain, etc.

COB-ID: CAN Object Identifiers.

CAN ID: CAN Identifier. This is the 11-bit CAN message identifier which is at the beginning of every CAN message on the bus.

EDS: Electronic Data Sheet. This is an INI style or XML style formatted file.

DCF: Device Configuration File. This is a modified EDS file with settings for node ID and baud rate.

Remark:

At this time, MEGA-GUARD has only very little of the complete CANOpen protocol implemented.

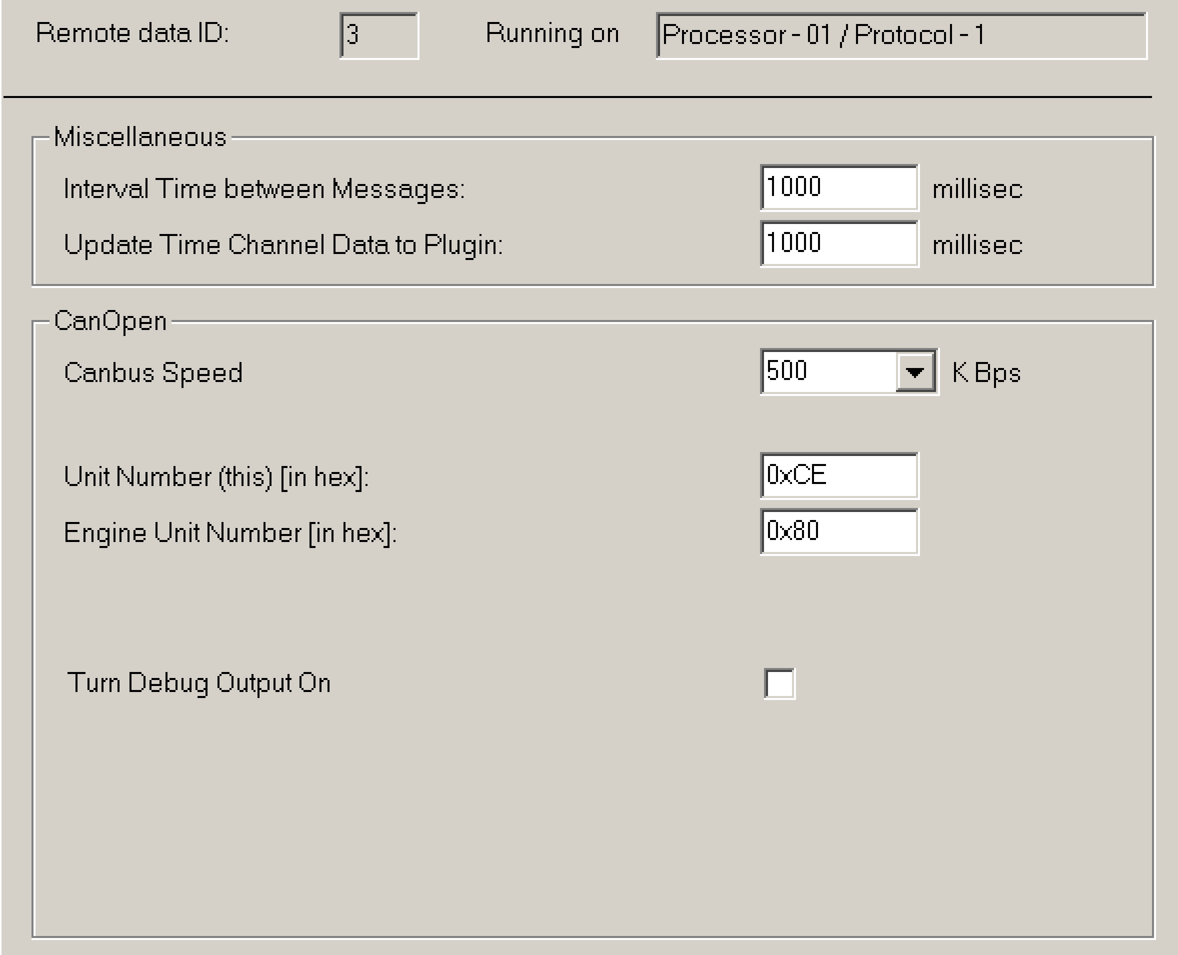
An object list with CAN Messages which works with communiction model **producer/consumer** can be handled.

Shortcuts

Icon:

Place: Plugins\

CANOpen – General Settings



Remote data ID: 3

Running on: Processor - 01 / Protocol - 1

Miscellaneous

Interval Time between Messages: 1000 millisec

Update Time Channel Data to Plugin: 1000 millisec

CanOpen

Canbus Speed: 500 K Bps

Unit Number (this) [in hex]: 0xCE

Engine Unit Number [in hex]: 0x80

Turn Debug Output On □

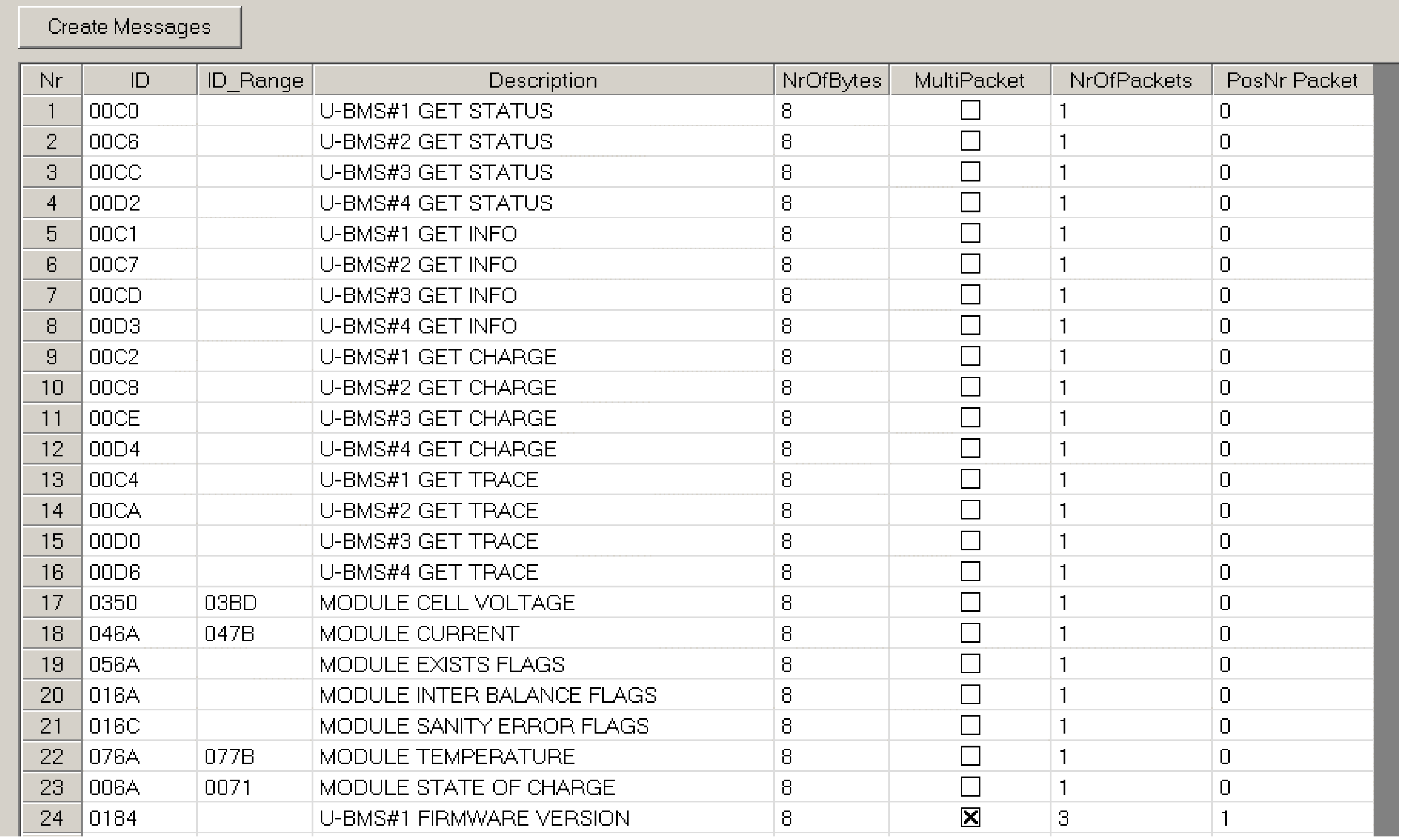
Setting	**Description**	**Default**
Interval Time between Messages	Minimal Time between two messages before reporting an error (50-100000 in milli-seconds	1000
Update Time channel Data to Plugin	When this protocol running on Server, IOServer uses this time to update channel values (50-100000 in milli-seconds	1000
Can Speed	Speed Setting for Canbus 125K, 250k, 500k 1M etc.	500k
Unit Number (this) [in hex]	Unit Address Number of Plugin	0xCE
Engine Unit Number [in hex]	Unit Address Number of Engine	0x80
Turn Debug Output On	connect on comport1, text output what messages are received on Canbus - yes/no	no

See also:

CANOpen Plugin

How to use Debug Output

CanOpen – Object List



Nr	ID	ID_Range	Description	NrOfBytes	MultiPacket	NrOfPackets	PosNr Packet
1	00C0		U-BMS#1 GET STATUS	8	□	1	0
2	00C6		U-BMS#2 GET STATUS	8	□	1	0
3	00CC		U-BMS#3 GET STATUS	8	□	1	0
4	00D2		U-BMS#4 GET STATUS	8	□	1	0
5	00C1		U-BMS#1 GET INFO	8	□	1	0
6	00C7		U-BMS#2 GET INFO	8	□	1	0
7	00CD		U-BMS#3 GET INFO	8	□	1	0
8	00D3		U-BMS#4 GET INFO	8	□	1	0
9	00C2		U-BMS#1 GET CHARGE	8	□	1	0
10	00C8		U-BMS#2 GET CHARGE	8	□	1	0
11	00CE		U-BMS#3 GET CHARGE	8	□	1	0
12	00D4		U-BMS#4 GET CHARGE	8	□	1	0
13	00C4		U-BMS#1 GET TRACE	8	□	1	0
14	00CA		U-BMS#2 GET TRACE	8	□	1	0
15	00D0		U-BMS#3 GET TRACE	8	□	1	0
16	00D6		U-BMS#4 GET TRACE	8	□	1	0
17	0350	03BD	MODULE CELL VOLTAGE	8	□	1	0
18	046A	047B	MODULE CURRENT	8	□	1	0
19	056A		MODULE EXISTS FLAGS	8	□	1	0
20	016A		MODULE INTER BALANCE FLAGS	8	□	1	0
21	016C		MODULE SANITY ERROR FLAGS	8	□	1	0
22	076A	077B	MODULE TEMPERATURE	8	□	1	0
23	006A	0071	MODULE STATE OF CHARGE	8	□	1	0
24	0184		U-BMS#1 FIRMWARE VERSION	8	☒	3	1

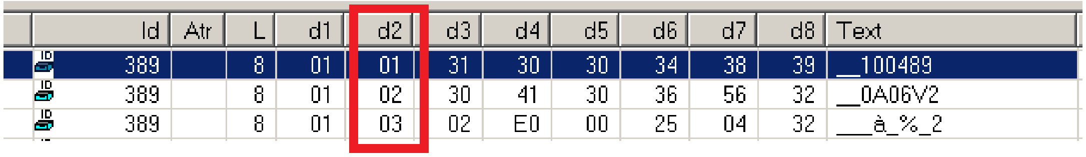
Create Messages Button to create some messages for communication with Valence U-BMS System.

Field	Description
ID	indentifier of message which will be received and handle by this plugin
IDRange	for range of indentifiers
Description	description of indentifier
Nr Of Bytes	data size of value
MultiPacket	is this message a multi-packet
NrOfPackets	multi-packets, number of packets which forms the complete message
Pos Nr Packet	multi-packets, position in message of packet counter, range 1-32

ID Enter your ID in hex with 4 digits.
Example: C0 have to be entered as 00C0.

IDRange Insert in a range from a value through value (always positive range)
example: 0350 - 03BD (03BD is last valid ID in this range)

Multi-Packet example
Number of packets is 3
Position number packet is 1 (0-7, second data byte)

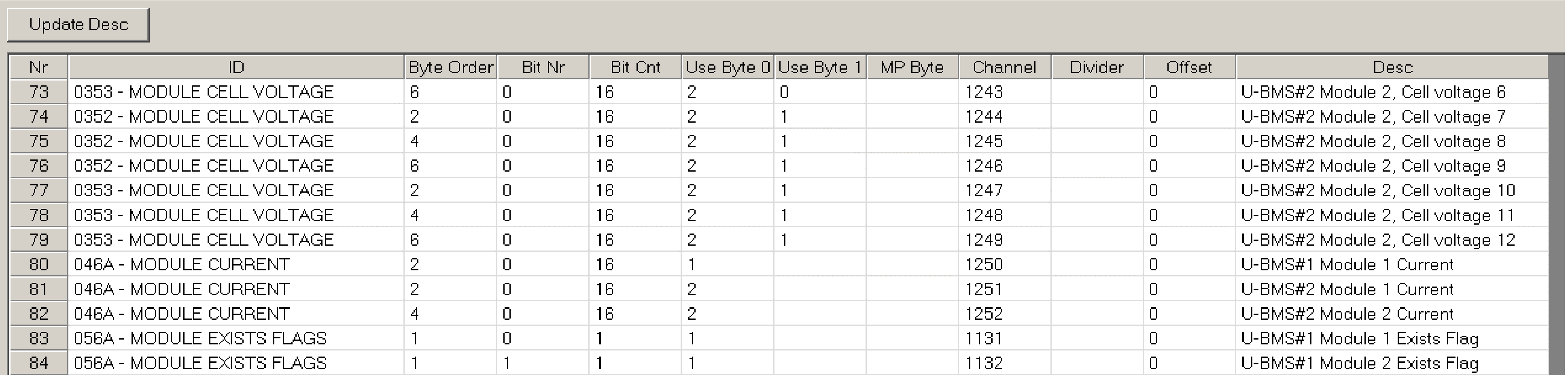


	Id	Atr	L	d1	d2	d3	d4	d5	d6	d7	d8	Text
ID	389		8	01	01	31	30	30	34	38	39	__100489
ID	389		8	01	02	30	41	30	36	56	32	__0A06V2
ID	389		8	01	03	02	E0	00	25	04	32	___à_%_2

See also:

CANOpen Plugin

CANOpen – Receive List



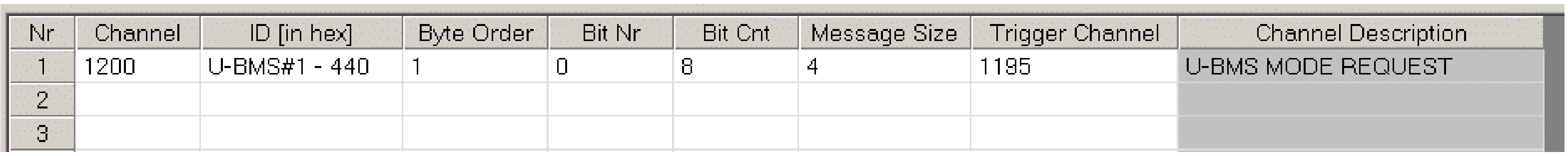
Nr	ID	Byte Order	Bit Nr	Bit Cnt	Use Byte 0	Use Byte 1	MP Byte	Channel	Divider	Offset	Desc
73	0353 - MODULE CELL VOLTAGE	6	0	16	2	0		1243		0	U-BMS#2 Module 2, Cell voltage 6
74	0352 - MODULE CELL VOLTAGE	2	0	16	2	1		1244		0	U-BMS#2 Module 2, Cell voltage 7
75	0352 - MODULE CELL VOLTAGE	4	0	16	2	1		1245		0	U-BMS#2 Module 2, Cell voltage 8
76	0352 - MODULE CELL VOLTAGE	6	0	16	2	1		1246		0	U-BMS#2 Module 2, Cell voltage 9
77	0353 - MODULE CELL VOLTAGE	2	0	16	2	1		1247		0	U-BMS#2 Module 2, Cell voltage 10
78	0353 - MODULE CELL VOLTAGE	4	0	16	2	1		1248		0	U-BMS#2 Module 2, Cell voltage 11
79	0353 - MODULE CELL VOLTAGE	6	0	16	2	1		1249		0	U-BMS#2 Module 2, Cell voltage 12
80	046A - MODULE CURRENT	2	0	16	1			1250		0	U-BMS#1 Module 1 Current
81	046A - MODULE CURRENT	2	0	16	2			1251		0	U-BMS#2 Module 1 Current
82	046A - MODULE CURRENT	4	0	16	2			1252		0	U-BMS#2 Module 2 Current
83	056A - MODULE EXISTS FLAGS	1	0	1	1			1131		0	U-BMS#1 Module 1 Exists Flag
84	056A - MODULE EXISTS FLAGS	1	1	1	1			1132		0	U-BMS#1 Module 2 Exists Flag

Update Desc button, updates desc field when it is known to software (hard-coded)

Field	Description
Nr	Number, row number
ID	ID hex Number + description, see Object List
Byte Order	start byte in message, 0..7
Bit Nr	start bit in message, 0..7
Bit Cnt	number of bits, 1 = digital, 8 = byte, 16 = word, 32 = dword
Use Byte 0	first byte in message should equal to .. Leaving this field empty, it will not be used
Use Byte 1	second byte in message should equal to .. Leaving this field empty, it will not be used
MP Byte	when having multi-packet message, this number equals packet number Leaving this field empty, it will not be used
Channel	Channel to retrieve value/status from eq. Leaving this field empty, this will delete entry
Divider	received value with be calculated with this divider before placing it into a channel value 1/100 or *50 Remark: *50 will be shown as 50 / 1
Offset	received value will be added to this offset before placing it into a channel value
Desc	Details description depends on contents on message

See also: CANOpen Plugin

CANOpen – Send List



Nr	Channel	ID [in hex]	Byte Order	Bit Nr	Bit Cnt	Message Size	Trigger Channel	Channel Description
1	1200	U-BMS#1 - 440	1	0	8	4	1195	U-BMS MODE REQUEST
2								
3								

Field	Description
Nr	row index of this grid
Channel	Channel contains value to be sent
ID [in hex]	Canbus ID for Message
Byte Order	byte number where to store value in message, 0-7
Bit Nr	bit number where value to start, 0-7
Bit Cnt	number of bits, 1 = digital, 8 = byte, 16 = word, 32 = dword
Message Size	number of data bytes to be sent in this message, 0-8
Trigger Channel	when digital channel is status on, message will be sent, leaving blank: message is always sent
Channel Description	Description of Channel

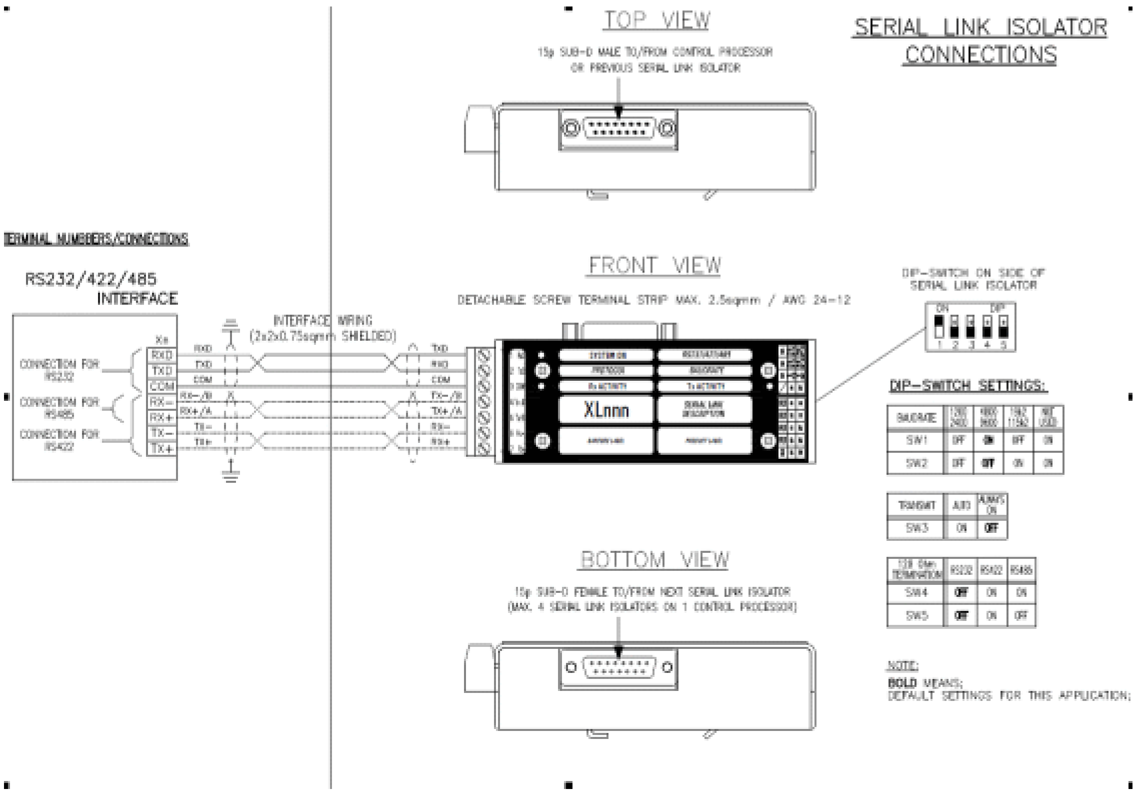
See also:

CANOpen Plugin

CANOpen - How to use Debug Output

Set option **'Turn Debug Output on'** to on by checking this box (see: CANOpen - General Settings)

Create a serial connection between XP processor and PC.
The processor could have ComPort 1 - 4. The initial version (product version 6008) is the debug option always on ComPort 2
For a serial link (each comport) on a control processor a serial link isolator product ID 98.6.040.800 is required.



TOP VIEW
15p SUB-D MALE TO/FROM CONTROL PROCESSOR OR PREVIOUS SERIAL LINK ISOLATOR
SERIAL LINK ISOLATOR CONNECTIONS
TERMINAL NUMBERS/CONNECTIONS
RS232/422/485 INTERFACE
Xn
RXD
TXD
COM
RX–
RX+
TX–
TX+
CONNECTION FOR RS232
CONNECTION FOR RS485
CONNECTION FOR RS422
INTERFACE WIRING (2x2x0.75sqmm SHIELDED)
RXD
TXD
COM
RX–/B
RX+/A
TX–
TX+
TXD
RXD
COM
TX–/B
TX+/A
RX–
RX+
FRONT VIEW
DETACHABLE SCREW TERMINAL STRIP MAX. 2.5sqmm / AWG 24–12
SYSTEM OK
RS232/422/485
PROTOCOL
BAUDRATE
Rx ACTIVITY
Tx ACTIVITY
XLnnn
SERIAL LINK DESCRIPTION
DIP–SWITCH ON SIDE OF SERIAL LINK ISOLATOR
ON
OFF
1 2 3 4 5
BOTTOM VIEW
15p SUB–D FEMALE TO/FROM NEXT SERIAL LINK ISOLATOR (MAX. 4 SERIAL LINK ISOLATORS ON 1 CONTROL PROCESSOR)

DIP–SWITCH SETTINGS:

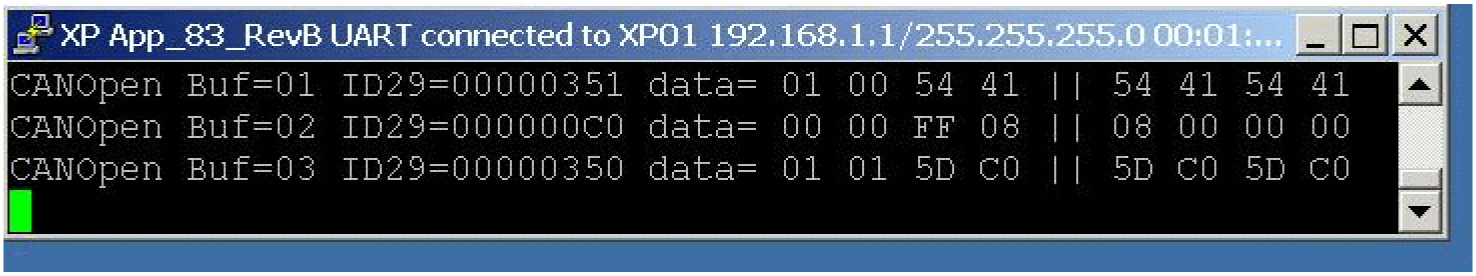
BAUDRATE	1200 2400	4800 9600	19k2 115k2	NOT USED
SW1	OFF	**ON**	OFF	ON
SW2	OFF	**OFF**	ON	ON

TRANSMIT	AUTO	ALWAYS ON
SW3	ON	**OFF**

120 Ohm TERMINATION	RS232	RS422	RS485
SW4	**OFF**	ON	ON
SW5	**OFF**	ON	OFF

NOTE:
BOLD MEANS;
DEFAULT SETTINGS FOR THIS APPLICATION;

By opening a putty client (serial port, 115200, 8, N, 1, flow control off) on PC
debug information can be retrieved for displaying it on a screen.
It will show text output which contains the canbus messages are received on Control Processor.



XP App_83_RevB UART connected to XP01 192.168.1.1/255.255.255.0 00:01:...

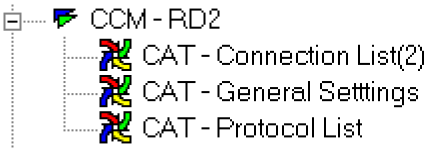
```
CANOpen Buf=01 ID29=00000351 data= 01 00 54 41 || 54 41 54 41
CANOpen Buf=02 ID29=000000C0 data= 00 00 FF 08 || 08 00 00 00
CANOpen Buf=03 ID29=00000350 data= 01 01 5D C0 || 5D C0 5D C0
```

See also:

[CANOpen - General Settings]

Caterpillar CCM – Plugin

After selecting ‘Plugins’ and CCM – RD xx



There are several items of this plugin:

1. CAT - General Settings

2. CAT - Connection List

3. CAT - Protocol List

Caterpillar is protocol that receives/sends data from/to diesel engines.

CCM = Customer Communication Module

At first login procedure is done, after that Single Parameter Read Request is send and a Single Parameter Read Response is returned.

Examples:

IID 24 - Single Parameter Read Request

$50 00 24 zz 00 24 F515 cs 0D

IID 25 - Single Parameter Read Response

$50 01 25 zz 00 24 F515 dddd cs 0D

Bytes:

1-4 standard preamble

5 Reply Format ($00 =ASCII or $01 = Binary)

6 Unit Number ($24 Elec.Engine Controller, $61 = CCM)

7-8 PID Parameter Identifier

dddd data value of parameter

cs – checksum following by carriage return

Standard preamble:

$50 = M5X protocol

$00 = User host device or $01 = CCM

$24 Instruction Identifier (=IID) like 24 = Single Parameter Read Request

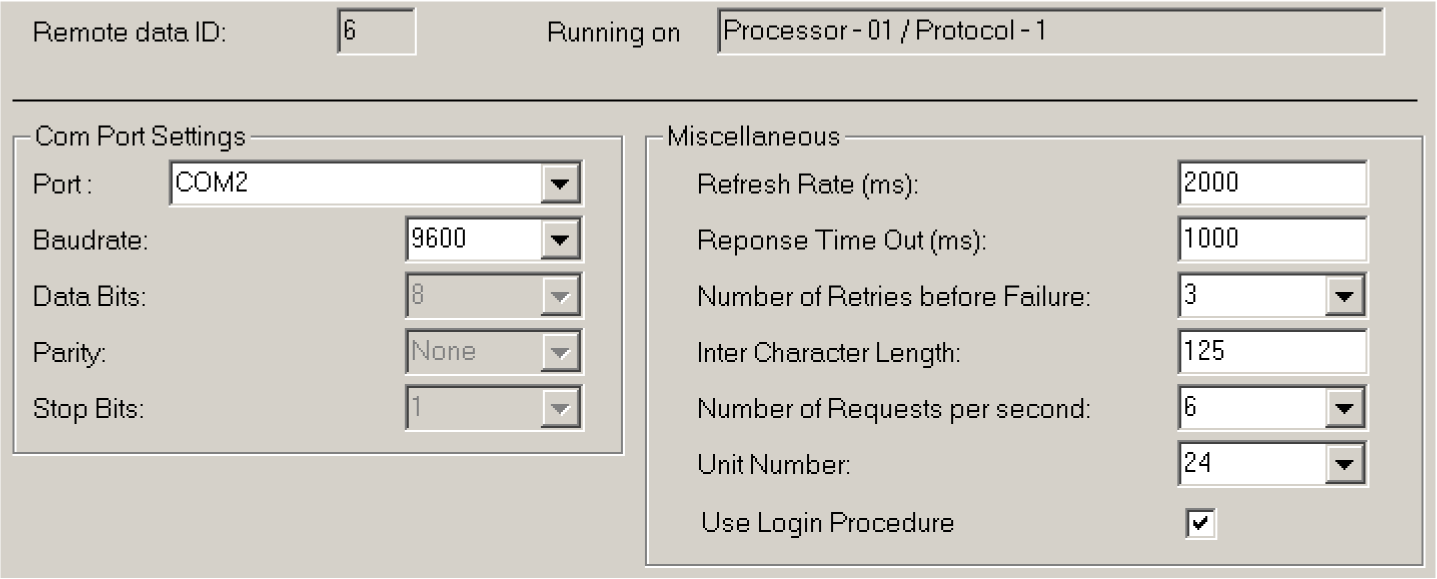
zz = is number of bytes in message after this byte

Shortcuts

Icon:

Place: Plugins\

CAT – General Settings



Remote data ID: 6

Running on Processor - 01 / Protocol - 1

Com Port Settings

Port : COM2

Baudrate: 9600

Data Bits: 8

Parity: None

Stop Bits: 1

Miscellaneous

Refresh Rate (ms): 2000

Reponse Time Out (ms): 1000

Number of Retries before Failure: 3

Inter Character Length: 125

Number of Requests per second: 6

Unit Number: 24

Use Login Procedure ☑

Com Port Communication port, No other plug-in should have the same COM port

Baudrate Baudrate of Com Port, 1200, 2400, 4800, 9600 or 19200 (default: 19200).

Refresh Rate 200-300000 millisec (default: 1000)

Response Time Out 50-5000 millisec (default: 200)

Unit Number 21,22,24,61, (default 61)

Nr of Retries before Failure 0-10 (default: 3)

Inter Character Length 0-10000 bits (default: 100)

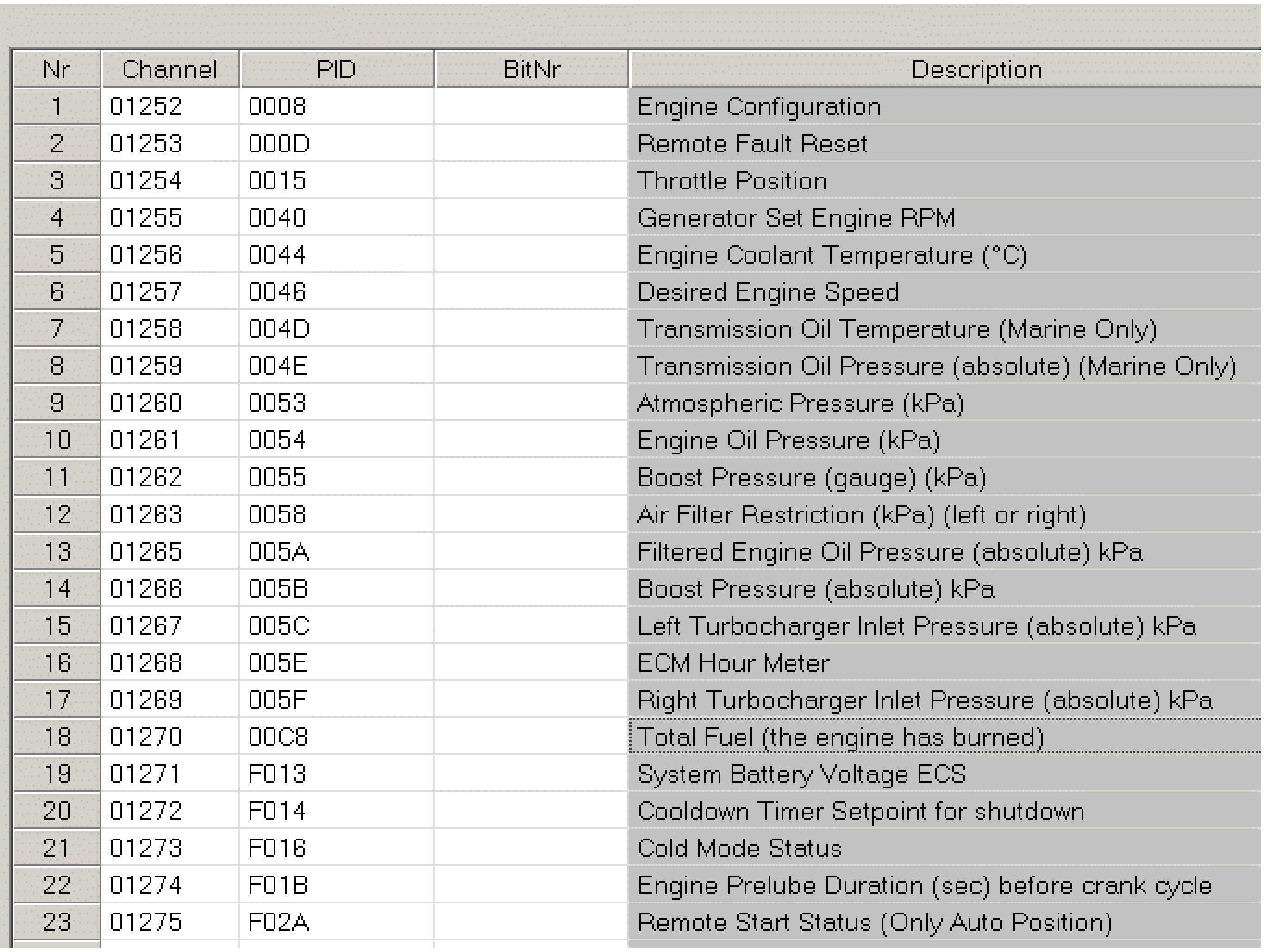
Number of Request per Second 0-10 (default: 3)

Use Login Procedure true or false via checkbox

See also:

Caterpillar CCM Plugin

CAT – Connection List



Nr	Channel	PID	BitNr	Description
1	01252	0008		Engine Configuration
2	01253	000D		Remote Fault Reset
3	01254	0015		Throttle Position
4	01255	0040		Generator Set Engine RPM
5	01256	0044		Engine Coolant Temperature (°C)
6	01257	0046		Desired Engine Speed
7	01258	004D		Transmission Oil Temperature (Marine Only)
8	01259	004E		Transmission Oil Pressure (absolute) (Marine Only)
9	01260	0053		Atmospheric Pressure (kPa)
10	01261	0054		Engine Oil Pressure (kPa)
11	01262	0055		Boost Pressure (gauge) (kPa)
12	01263	0058		Air Filter Restriction (kPa) (left or right)
13	01265	005A		Filtered Engine Oil Pressure (absolute) kPa
14	01266	005B		Boost Pressure (absolute) kPa
15	01267	005C		Left Turbocharger Inlet Pressure (absolute) kPa
16	01268	005E		ECM Hour Meter
17	01269	005F		Right Turbocharger Inlet Pressure (absolute) kPa
18	01270	00C8		Total Fuel (the engine has burned)
19	01271	F013		System Battery Voltage ECS
20	01272	F014		Cooldown Timer Setpoint for shutdown
21	01273	F016		Cold Mode Status
22	01274	F01B		Engine Prelube Duration (sec) before crank cycle
23	01275	F02A		Remote Start Status (Only Auto Position)

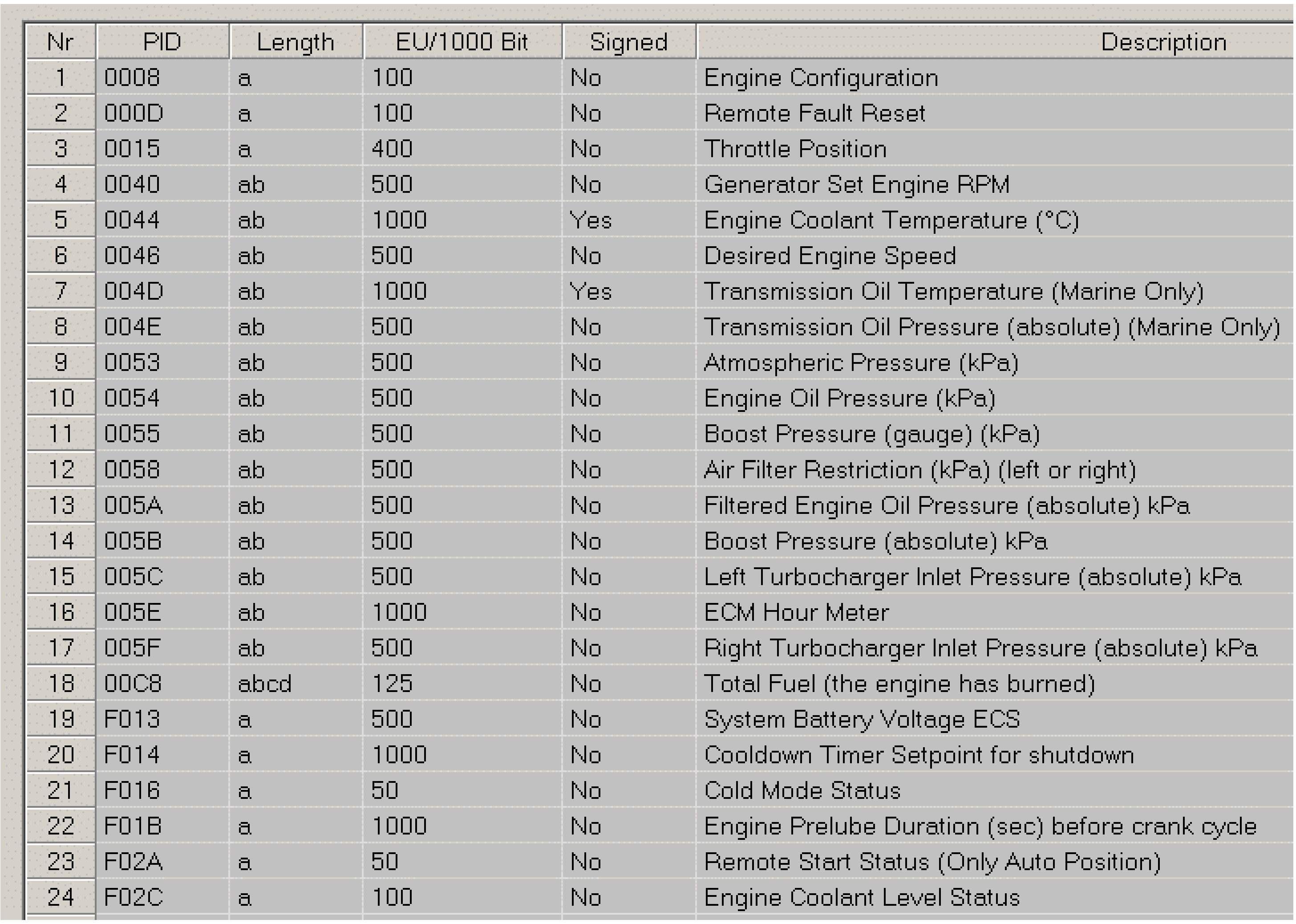
Field	Description
Channel	Channel to receive the data; type in an existing digital channel number, or get here from the channel's configuration form (**status/value link**)
PID	Parameter identifier 0-FFFF, hex value, via droplist see protocol list for the ones supported
Bit Nr	0 or empty / (abcd) a1..a8 / b1..b8 / c1..c8 / d1..d8

By pressing on column “Channel” or “PID” a sort action is performed, first time ascending second descending.

See also:

Caterpillar CCM Plugin

CAT – Protocol List



Nr	PID	Length	EU/1000 Bit	Signed	Description
1	0008	a	100	No	Engine Configuration
2	000D	a	100	No	Remote Fault Reset
3	0015	a	400	No	Throttle Position
4	0040	ab	500	No	Generator Set Engine RPM
5	0044	ab	1000	Yes	Engine Coolant Temperature (°C)
6	0046	ab	500	No	Desired Engine Speed
7	004D	ab	1000	Yes	Transmission Oil Temperature (Marine Only)
8	004E	ab	500	No	Transmission Oil Pressure (absolute) (Marine Only)
9	0053	ab	500	No	Atmospheric Pressure (kPa)
10	0054	ab	500	No	Engine Oil Pressure (kPa)
11	0055	ab	500	No	Boost Pressure (gauge) (kPa)
12	0058	ab	500	No	Air Filter Restriction (kPa) (left or right)
13	005A	ab	500	No	Filtered Engine Oil Pressure (absolute) kPa
14	005B	ab	500	No	Boost Pressure (absolute) kPa
15	005C	ab	500	No	Left Turbocharger Inlet Pressure (absolute) kPa
16	005E	ab	1000	No	ECM Hour Meter
17	005F	ab	500	No	Right Turbocharger Inlet Pressure (absolute) kPa
18	00C8	abcd	125	No	Total Fuel (the engine has burned)
19	F013	a	500	No	System Battery Voltage ECS
20	F014	a	1000	No	Cooldown Timer Setpoint for shutdown
21	F016	a	50	No	Cold Mode Status
22	F01B	a	1000	No	Engine Prelube Duration (sec) before crank cycle
23	F02A	a	50	No	Remote Start Status (Only Auto Position)
24	F02C	a	100	No	Engine Coolant Level Status

List of Parameter Identifiers 0000-FFFF supported for this protocol

Examples:

$0015 Throttle Position

$0040 Engine RPM

$0044 Engine Coolant Temp

$AA8A Login Password (not needed to setup, automatic done if “Use Login Procedure” is checked,see also CAT - General Settings)

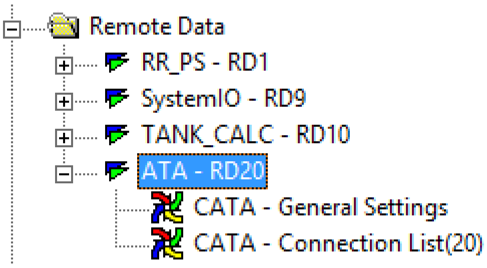
$FC07 Warning Status (abcd)

$FC08 ShutdownStatus (abcd)

$FC09 Engine Derate Status (abcd)

ATA – Plugin

After selecting ‘Plugins’ and ATA – RD xx



There are several items of this plugin:

1. ATA - General Settings

2. ATA - Connection List

ATA is protocol that receives data from heavy machines.

Joint SAE/TMC Electronic Data Interchange between Microcomputer systems in heavy duty vehicle applications. (SA J1587)

Transmission Data (ASCII string without spaces ending):

MID PID Data Checksum

Message Identifier start of each message

-example: 0x80 (128)

Parameter Identification character

-example: 0xBE (190) – Engine Speed

Normally:

PID 000 127 has 1 data char

PID 128-191 has 2 data chars

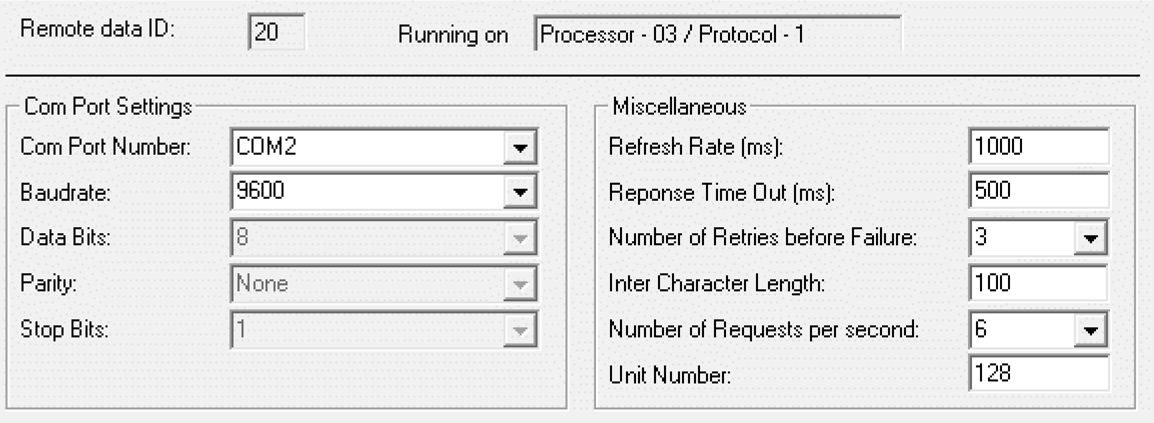
PID 192-253 has variabel data chars

Shortcuts

Icon: ⚑

Place: Plugins\

ATA – General Settings



Remote data ID: 20 Running on Processor - 03 / Protocol - 1

Com Port Settings

Com Port Number: COM2

Baudrate: 9600

Data Bits: 8

Parity: None

Stop Bits: 1

Miscellaneous

Refresh Rate (ms): 1000

Reponse Time Out (ms): 500

Number of Retries before Failure: 3

Inter Character Length: 100

Number of Requests per second: 6

Unit Number: 128

Com Port Communication port which is used for receiving the input.
No other plug-in is allowed to have the same COM port number.
This applies only if plugin is on running on server.

Baudrate Baudrate of Com Port, 1200, 2400, 4800, 9600 or 19200 (default: 19200).

Refresh Rate 200-300000 millisec (default: 1000)

Response Time Out 50-5000 millisec (default: 200)

Nr of Retries before Failure 0-10 (default: 3)

Inter Character Length 0-10000 bits (default: 3000)

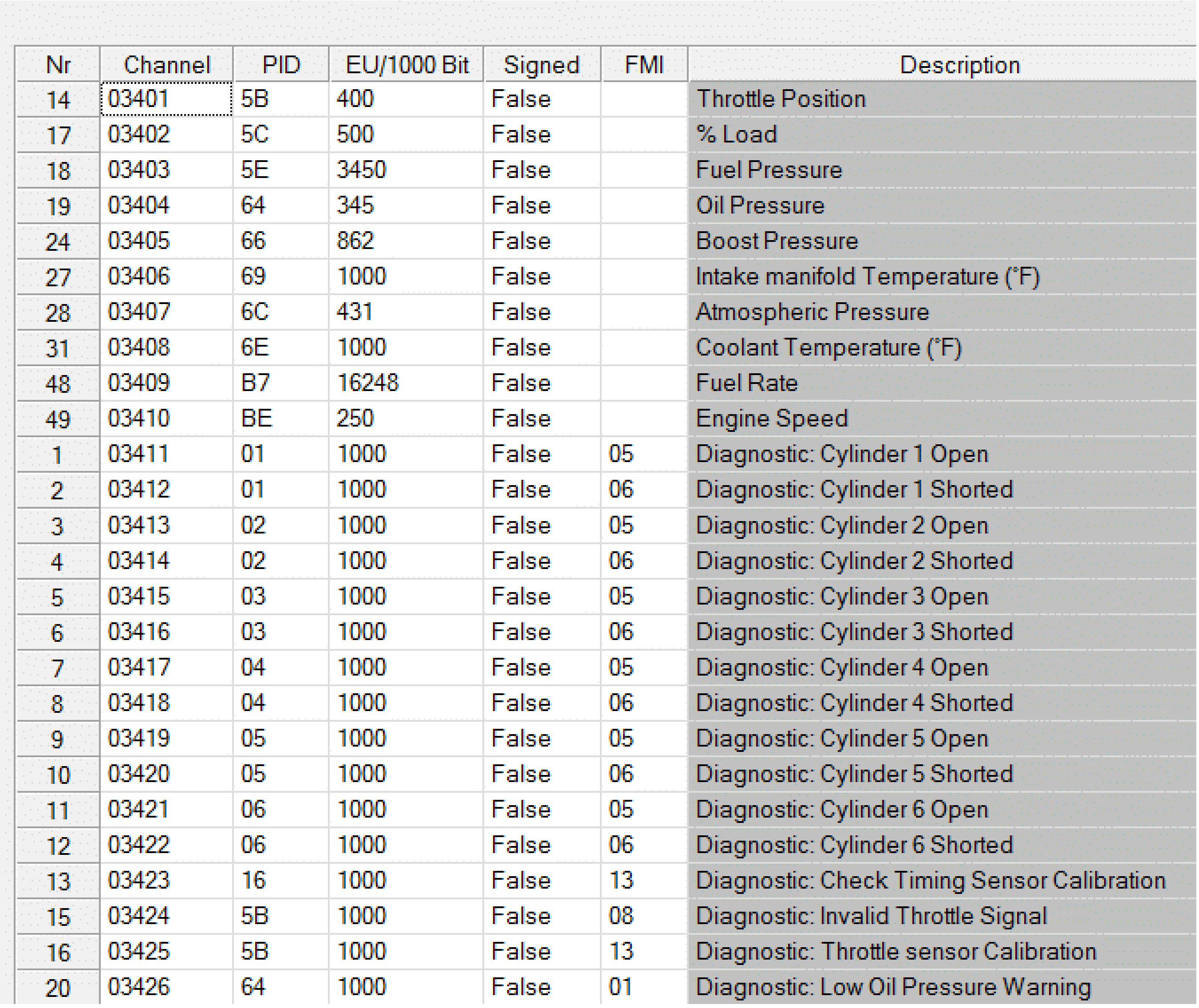
Number of Request per Second 0-10 (default: 3)

Unit Number 0-255, (default 80) Message Identifier

See also:

ATA Plugin

ATA – Connection List



Nr	Channel	PID	EU/1000 Bit	Signed	FMI	Description
14	03401	5B	400	False		Throttle Position
17	03402	5C	500	False		% Load
18	03403	5E	3450	False		Fuel Pressure
19	03404	64	345	False		Oil Pressure
24	03405	66	862	False		Boost Pressure
27	03406	69	1000	False		Intake manifold Temperature (°F)
28	03407	6C	431	False		Atmospheric Pressure
31	03408	6E	1000	False		Coolant Temperature (°F)
48	03409	B7	16248	False		Fuel Rate
49	03410	BE	250	False		Engine Speed
1	03411	01	1000	False	05	Diagnostic: Cylinder 1 Open
2	03412	01	1000	False	06	Diagnostic: Cylinder 1 Shorted
3	03413	02	1000	False	05	Diagnostic: Cylinder 2 Open
4	03414	02	1000	False	06	Diagnostic: Cylinder 2 Shorted
5	03415	03	1000	False	05	Diagnostic: Cylinder 3 Open
6	03416	03	1000	False	06	Diagnostic: Cylinder 3 Shorted
7	03417	04	1000	False	05	Diagnostic: Cylinder 4 Open
8	03418	04	1000	False	06	Diagnostic: Cylinder 4 Shorted
9	03419	05	1000	False	05	Diagnostic: Cylinder 5 Open
10	03420	05	1000	False	06	Diagnostic: Cylinder 5 Shorted
11	03421	06	1000	False	05	Diagnostic: Cylinder 6 Open
12	03422	06	1000	False	06	Diagnostic: Cylinder 6 Shorted
13	03423	16	1000	False	13	Diagnostic: Check Timing Sensor Calibration
15	03424	5B	1000	False	08	Diagnostic: Invalid Throttle Signal
16	03425	5B	1000	False	13	Diagnostic: Throttle sensor Calibration
20	03426	64	1000	False	01	Diagnostic: Low Oil Pressure Warning

Field	Description
Channel	Channel to receive the data; type in an existing channel number
PID	Parameter identifier 0-FF, hex value
EU/1000 Bit	0 – 10000, bit resolution, example Engine Speed has bit resolution 0.25 rpm -> 250 have to be filled in
Signed	True or False
FMI	Failure Mode Identifier, bit nr 0-15 Only support with PID like 100-01 “Low oil pressure warning” with Diagnostic Message (194, fixed so 194 don’t be filled in)

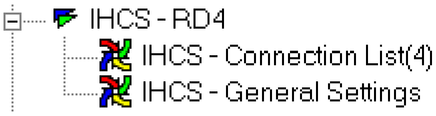
By pressing on column “Channel” or “PID” a sort action is performed, first time ascending second descending.

See also:

ATA Plugin

IHCS – System Plugin

After selecting ‘Plugins’ and IHCS – RD xx



IHCS - RD4

IHCS - Connection List(4)

IHCS - General Settings

There are several items of this plugin:

1. IHCS - General Settings
2. IHCS - Connection List

IHC Systems stands for Industrial Handels* Combination (*=Business, trans. dutch)

A band of companies who are dealing in automation systems for dregding industry.

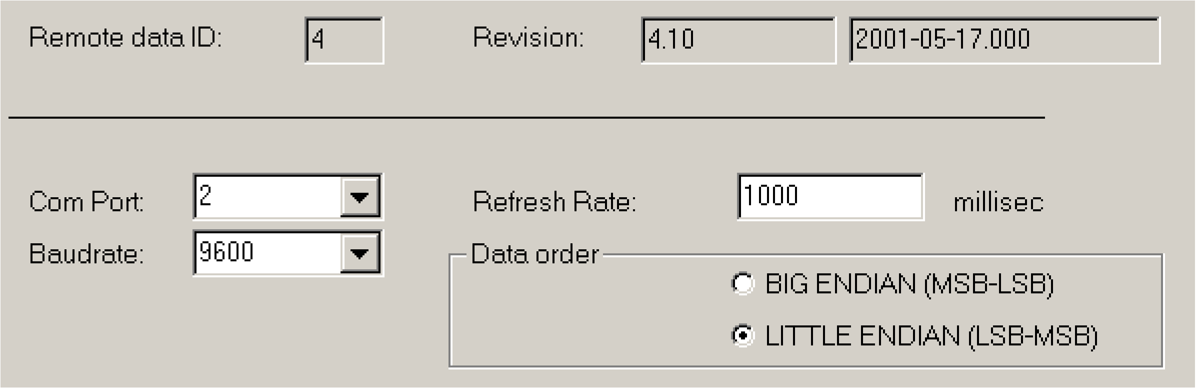
This protocol is sending protocol only.

Shortcuts

Icon: ◤

Place: Plugins\

IHCS – General Settings



Remote data ID: 4

Revision: 4.10 2001-05-17.000

Com Port: 2

Refresh Rate: 1000 millisec

Baudrate: 9600

Data order

○ BIG ENDIAN (MSB-LSB)

● LITTLE ENDIAN (LSB-MSB)

Com Port Communication port which is used for sending the output. No other plug-in must have the same COM port (only if plugin is on running on server)

Baudrate Baudrate of Com Port, 1200, 2400, 4800, 9600 or 19200 (default: 19200).

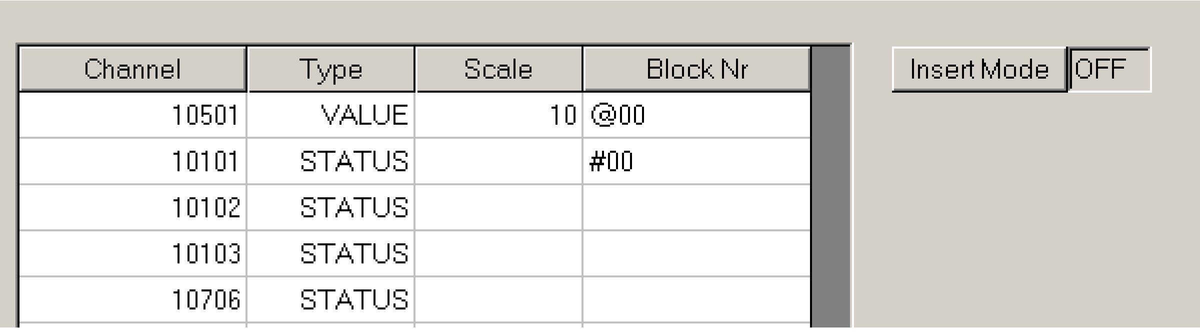
Refresh Rate 200-300000 millisec (default: 1000)

Data Order Big Endian(Intel based), Little Endian(Motorola based processors)

See also:

IHC Systems Plugin

IHCS – Connection List



Channel	Type	Scale	Block Nr
10501	VALUE	10	@00
10101	STATUS		#00
10102	STATUS		
10103	STATUS		
10706	STATUS		

Insert Mode: OFF

Field	**Description**
Channel	Channel to receive the data; type in an existing channel number, or get here from the channel's configuration form (**status/value link**)
Type	Type of Channel Status # or Value @
Scale	Scale of channel value
Block	Sending Block Number, not editable, shows message layout

By pressing on column "Channel" a sort action is performed, first time ascending second descending. REMARK: NO EDITING IS ALLOWED AFTER SORTING ON CHANNEL, BE SURE SORTED FIRST ON BLOCK!

By pressing on column "Block" sending block information is shown more detailed. After pressing again sending block information is hidden except start identifier.

More than one occurrence of the same channel is allowed.

'0' can be entered to create a spare location within the message (you do not have to enter spares at the end of the last analog and last digital message).

When "INSERT MODE" is on, a blank row is created after the channel just entered.

When the "CHANNEL" field is (made) empty, the row is deleted.

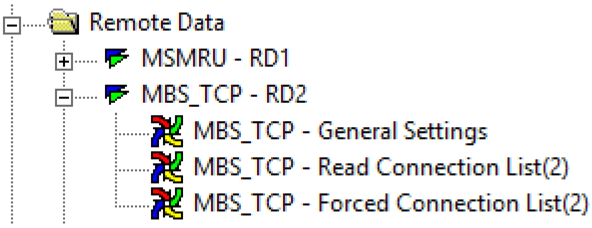
Tips: For fast channel configuration, give a channel range in (like 10101-10125) and press enter, when analog channels are in that given range a popup-dialog is shown. After selecting a scale and value multiple rows are created!

See also:

IHC Systems Plugin

MBS_TCP – Modbus Slave TCP Plugin

After selecting 'Plugins' and MBS_TCP – RD xx



Remote Data
MSMRU - RD1
MBS_TCP - RD2
MBS_TCP - General Settings
MBS_TCP - Read Connection List(2)
MBS_TCP - Forced Connection List(2)

There are several items of this plugin:

1. MBS_TCP - General Settings
2. MBS_TCP - Read Connection List
3. MBS_TCP – Forced Connection List

This plugin supports the following modbus functions:

Outputs (Read Connection List)

01 – Coils (Digital, error code 0x80)

02 – Status (Digital, error code 0x82))

03 – Hold Registers (Analog, error code 0x83)

04 – Input Registers (Analog, error code 0x84)

Inputs (Forced Connection List)

05 – Single Coil (Digital, error code 0x85)

06 – Single Register (Analog, error code 0x86)

15 - Multiple Coils (Digital, error code 0x8F)

16 - Multiple registers (Analog, error code 0x90)

For Read Connection list it's possible to send with function 03/04:
16 digital statuses mapped to one modbus address.
For each status specify the bitnumber within this address. (not for modicon addressing)

Structures are sent according Motorola (MSB LSB or Big Endian) processor set.

Register Values are handled depending of configuration:

Register Range	Signed	Unsigned	8000H Offset
Negative	-32768..-1 (8000.. FFFF)		-32768..-1 (0..7FFF)
Positive	0..32767 (0000..7FFF)	0..65535 (0000..FFFFF)	0..32767 (8000..FFFF)
Error Value (Out of range)	-32768 (8000)	65535 (FFFF)	-32768 (0000)

(Values) use hex notation

For value extension and value precision another types of register handling is available.
The base types of these are:

- long (signed 32 bits)
- ulong (unsigned 32 bits)
- float (32 bits, contains one sign bit and exponent (two complement) and 23 bit mantissa)

Furthermore there could be a distinction between low byte and high byte sending/receiving order. (Abbreviation L/H or H/L) Normally H/L is most frequently used.

Therefore a Modbus register is always 16 bits, two registers addresses are used for a 32 bit value presentation.

MBM TCP/IP message exists of:

MBAP Header	Function Number	Data

This message provides some differences compared to the MODBUS RTU application data unit used on serial line:

- The MODBUS ‘slave address’ field usually used on MODBUS Serial Line is replaced by a single byte ‘Unit Identifier’ within the MBAP Header. The ‘Unit Identifier’ is used to communicate via devices such as bridges, routers and gateways that use a single IP address to support multiple independent MODBUS end units.

- All MODBUS requests and responses are designed in such a way that the recipient can verify that a message is finished. For function codes where the MODBUS PDU has a fixed length, the function code alone is sufficient. For function codes carrying a variable amount of data in the request or response, the data field includes a byte count.

- When MODBUS is carried over TCP, additional length information is carried in the MBAP header to allow the recipient to recognize message boundaries even if the message has been split into multiple packets for transmission. The existence of explicit and implicit length rules, and use of a CRC-32 error check code (on Ethernet) results in an infinitesimal chance of undetected corruption to a request or response message.

MBAP Header

Fields	Length	Description	Client	Server
Transaction Identifier	2 Bytes	Identification of a MODBUS Request / Response transaction.	Initialized by the client	Recopied by the server from the received request
Protocol Identifier	2 Bytes	0 = MODBUS protocol	Initialized by the client	Recopied by the server from the received request
Length	2 Bytes	Number of following bytes	Initialized by the client (request)	Initialized by the server (Response)
Unit Identifier	1 Byte	Identification of a remote slave connected on a serial line or on other buses.	Initialized by the client	Recopied by the server from the received request

The header is 7 bytes long:

• Transaction Identifier - It is used for transaction pairing, the MODBUS server copies in the response the transaction identifier of the request.

- Protocol Identifier – It is used for intra-system multiplexing. The MODBUS protocol is identified by the value 0.

- Length - The length field is a byte count of the following fields, including the Unit Identifier and data fields.

- Unit Identifier – This field is used for intra-system routing purpose. It is typically used to communicate to a MODBUS+ or a MODBUS serial line slave through a gateway between an Ethernet TCP-IP network and a MODBUS serial line. This field is set by the MODBUS Client in the request and must be returned with the same value in the response by the server.

Example Function 02 digital status request

MBAP Header (seven bytes)	Function Nr (one byte)	Address (two bytes)	Nr of statusses (two bytes)
header	0x02	0x01 0x00	0x00 0x09

Example Function 02 digital status reply

MBAP Header (seven bytes)	Function Nr (one byte)	Byte Count (one byte)	Data (Byte Count bytes)
header	0x02	0x02	0xFF 0xFF

Example Function 02 digital status exception reply

MBAP Header (seven bytes)	Function Nr (one byte)	Exception code (one byte)
header	0x**82**	**0x02**

To get the status of the points with MODBUS address 100 to 113, when e.g. the points on address 103 and 108 are 'on' and the other points are 'off':

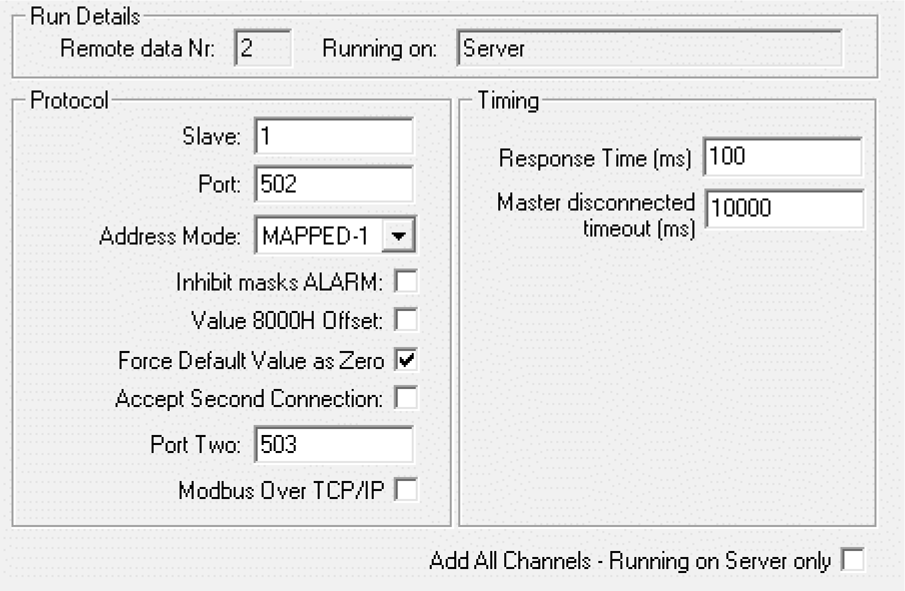
For more information reference the Modbus Specification: Modbus_Messaging_Implementation_Guide_V1_0b.pdf See http://www.modbus.org/.

Shortcuts

Icon:

Place: Plugins\

MBS_TCP – General Settings



Run Details

Remote data Nr: 2 Running on: Server

Protocol

Slave: 1

Port: 502

Address Mode: MAPPED-1

Inhibit masks ALARM: □

Value 8000H Offset: □

Force Default Value as Zero ☑

Accept Second Connection: □

Port Two: 503

Modbus Over TCP/IP □

Timing

Response Time (ms) 100

Master disconnected timeout (ms) 10000

Add All Channels - Running on Server only □

Run Details

Field	Options/Values and Function
Remote data	Remote data being configured
Running on	Server: Protocol runs on the Server as a Plug-in Processor XX : Protocol on Processor board (XX = Processor number / Protocol Index) eq. Processor - 04 / Protocol - 1

Protocol settings

Field	Options/Values and Function
Slave	Slave number 1-255 (default 1)
Port	communication port for Ethernet usage (default: 502)
Address Mode	One of three options can be selected: <table><tr><th>Value</th><th>Function</th></tr><tr><td>**Mapping**</td><td>Addresses are used on the serial line as configured</td></tr><tr><td>**Mapping-1**</td><td>Configured Addresses are used with a offset of - 1 to the serial line eq. PAL 40001 -> Line 40000</td></tr><tr><td>**Modicon Style**</td><td>MODICON Offsets are used for each modbus function See Forced or Read Connection-list for offsets</td></tr></table> (default Mapping)
Inhibits Masks Alarm	*true*: if channel is inhibited all alarm bits return zero irrespective their actual value *false*: all alarm bits return actual value whether inhibited or not
Value 8000H Offset	Selected zero point becomes 0x8000 (replaces sign bit) (default: deselected)
Force Zero Value	Force registers values to zero for functions (05/06 or 15/16) when channels are not

	available or the registers are not configured (default unchecked/off)
Accept Second Connection	for setup an extra Ethernet connection, only when protocol is running on Server (default: unchecked/off)
Port Two	communication port for Ethernet usage, when 'Accept Second Connection' is checked (default: 503)
Modbus over TCP/IP	Modbus RTU message transmitted with a TCP/IP wrapper and sent over a network instead of serial comport. The Server uses an IP Address therefore a SlaveID is not needed. (default unchecked/off)

Timing settings

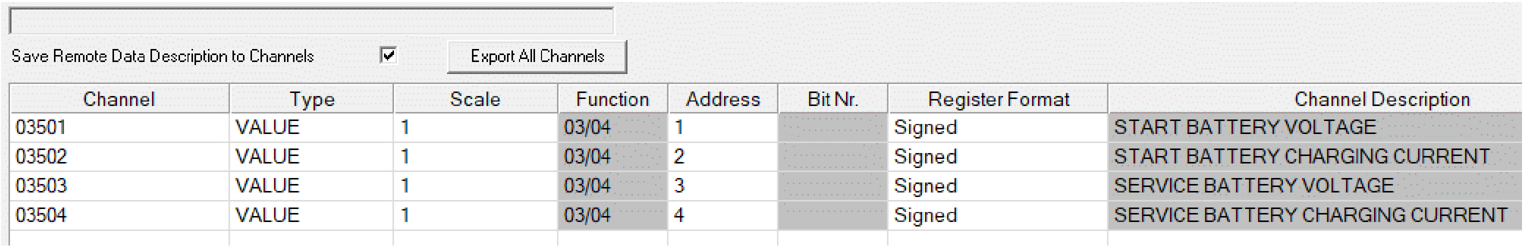
Field	Options/Values and Function
Response time (ms)	Time to wait between Receiving complete query and processing and sending Answers. Needed if Modbus master needs time to switch handshaking. (default: 100)
Master disconnected timeout (ms)	Time that has to expire before diagnostics are triggered, see Diagnostics. - No communication, on line no bytes are received time of 0 disables these diagnostics. (default: 10000)

Add All Channels - Running on Server Only
To put all channels in system into this plugin, when checked: no read connection list is needed to be configured.

See also:

Modbus Slave TCP Plugin
Modbus Slave TCP Diagnostics
Modbus Slave TCP Read Connection List
Modbus Slave TCP Forced Connection List

MBS_TCP – Read Connection List



Save Remote Data Description to Channels ☑

Channel	Type	Scale	Function	Address	Bit Nr.	Register Format	Channel Description
03501	VALUE	1	03/04	1		Signed	START BATTERY VOLTAGE
03502	VALUE	1	03/04	2		Signed	START BATTERY CHARGING CURRENT
03503	VALUE	1	03/04	3		Signed	SERVICE BATTERY VOLTAGE
03504	VALUE	1	03/04	4		Signed	SERVICE BATTERY CHARGING CURRENT

Column	Options or Values
Channel	Channel to send the data; type in an existing digital/analog channel number, or get here from the channel's configuration form (**status/value link**)
Type	Value or one of the status bits Sensfail, Any Alarm, Inhibited, Skipped, All Status (default depends on channel)
Scale	Scale value before send *1000, *100, *10, 1, /10, /100, /1000 (default 1)
Function	Function number, like 01/02 or 03/04 for MODICON addressing like 01, 02, 03 or 04
Address	Address value
Bit Nr.	bit number, applies to function 03 or 04 with status connection only 00, 01 till 15 (default 00)
Register Format	How is register handled - 8000H Offset means zero point becomes 0x08000 Signed, Unsigned, 8000H Offset or Free (default Signed) Long (H/L), Long (L/H), ULong (H/L), ULong (LH), Float (H/L), Float (L/H)
Channel Description	Description of channel for documentation only

A modbus master receives channel information from the slave. (=Output List)

Sorting
By pressing on column "Channel" or "Function" a sort action is performed, first time ascending second descending.

Addres field of actual message depends on Addres Mode, MBS_TCP - General Settings

<table>
<tr><th>Option</th><th>Conversion</th></tr>
<tr><td>"Mapping / No Offset"</td><td>straight copy of address</td></tr>
<tr><td>"Mapping / Offset - 1"</td><td>address - 1</td></tr>
<tr><td>"Modicon Style"</td><td><table>
<tr><th>Function</th><th>Conversion</th></tr>
<tr><td>01</td><td>address - 1</td></tr>
<tr><td>02</td><td>address - 10001</td></tr>
<tr><td>03</td><td>address - 40001</td></tr>
<tr><td>04</td><td>address - 30001</td></tr>
</table></td></tr>
</table>

Export All Channels
See: MBS_TCP - General Settings to make this option active.
The Export option to creates a .csv file with all channel specifications from the AMCS system.

CSV file Example with column header and example data
Channel , TagName, Type , Scale , Function , Address , Signedness , Description
1001 , 01001 , DS , 1 , 3 , 0001 , Unsigned , TIMER KEYSWITCH ON/OFF TEST
1017 , 01017 , DS , 1 , 3 , 0017 , Unsigned , DEADMAN ALARM
1020 , 01020 , AV , 1 , 3 , 0018 , Signed , BAROMETRIC SENSOR
1020 , 01020 , AS , 1 , 3 , 0019 , Unsigned , BAROMETRIC SENSOR
1021 , 01021 , DS , 1 , 3 , 0020 , Unsigned , BAROMETRIC SENSOR FAIL

TagName Column:
Tag name is a 10 character unique name for channel. Each analog channel has value register row and a status register row.

Type Column:

DS : digital status
AV : analog value
AS : analog status

Scale Column:
Multiplier to allow usage of decimal numbers in integer format.

Function Column:
Modbus function for transmitting value to master.

Signed / Unsigned Column:
Analog values use signed, digital values (bits) are using unsigned.

The size of a single signed value is 2 bytes. For large values such as latitude/longitude and hour counters the register holds the lower 16 bits.

Description Column:
Description is textual information with maximum length of 40 characters.
This data is displayed on screen and logged on file in alarm lists and group information.

Status register format (with Type column value AS, DS):

Description	Value (hex)
Normal	0x00
Very Low	0x01
Low & Boolean value*	0x02
High	0x04
Very High	0x08
Average	0x20
Sensor Fail	0x40
Not Available	0x80
Skip	0x100
Inhibit	0x200
Ack All	0x400

* Boolean values for digital status channels are stored in Bit 2.
When using normally open:
- for open relay "Register Value or 0x02 = False"
- for closed relay "Register Value or 0x02 = True"

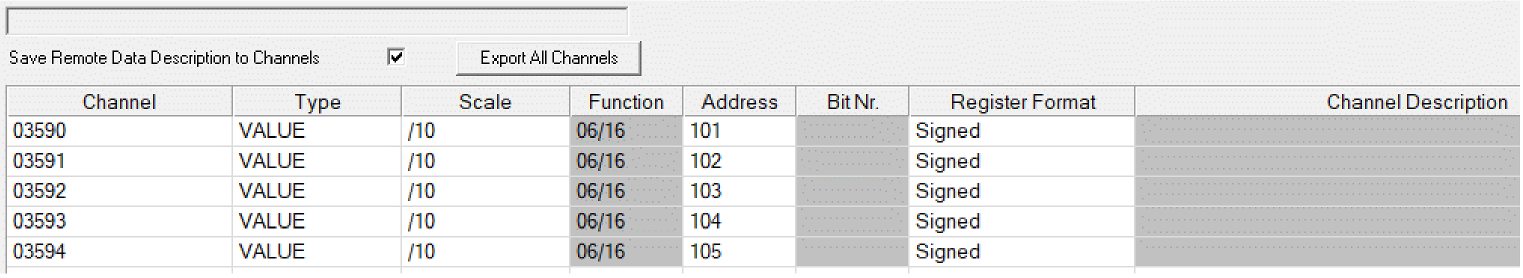
When using normally closed above values are inverted.

Tip: For fast channel configuration, configure the last line complete,
press not enter at the last column, go back to first column, give a channel range in
(like 10101-10125) and press enter, now multiple rows are created!

See also:

Modbus Slave TCP Plugin
Modbus Slave TCP Forced Connection List

MBS_TCP – Forced Connection List



Save Remote Data Description to Channels ☑ Export All Channels

Channel	Type	Scale	Function	Address	Bit Nr.	Register Format	Channel Description
03590	VALUE	/10	06/16	101		Signed	
03591	VALUE	/10	06/16	102		Signed	
03592	VALUE	/10	06/16	103		Signed	
03593	VALUE	/10	06/16	104		Signed	
03594	VALUE	/10	06/16	105		Signed	

Column	Options or Values
Channel	Channel to send the data; type in an existing digital/analog channel number, or get here from the channel's configuration form (**status/value link**)
Type	Value or one of the status bits (only "Low" is supported yet)
Scale	Scale value after received
Function	Function number, like 05/15 or 06/16
Address	Address value
Bit Nr.	bit number, status only
Register Format	How is register handled - 8000H Offset means zero point becomes 0x08000 Signed, Unsigned, 8000H Offset or Free (default Signed) Long (H/L), Long (L/H), ULong (H/L), ULong (LH), Float (H/L), Float (L/H)
Channel Description	Description of channel for documentation only

A modbus master send channel information to the slave. (=Input List)

By pressing on column "Channel" a sort action is performed, first time ascending second descending.

Return Button: Return from gateway, only available when getting there from a channel configuration (status/value link)

Addres field of actual message depends on Addres Mode, MBS_TCP - General Settings

Option	Conversion
"Mapping / No Offset"	straight copy of address
"Mapping / Offset - 1"	address - 1
"Modicon Style"	<table><tr><th>Function</th><th>Conversion</th></tr><tr><td>05/15</td><td>address - 1</td></tr><tr><td>06/16</td><td>address - 40001</td></tr></table>

Tips: For fast channel configuration, configure the last line complete, press not enter at the last column, go back to first column, give a channel range in (like 10101-10125) and press enter, now multiple rows are created!

See also:

Modbus Slave TCP Plugin
Modbus Slave TCP Read Connection List

MBS_TCP - Diagnostics

Diagnostics are used to report Failures or important statusses
the modbus slave plugin has the following diagnostics

- **MBS_TCP: RD(%RD64) TCP/IP Socket Error**

  Diagnostic is generated when the socket can not be used, while active the plugin will try to open this socket/port.

  (i) Only triggered on server

- **MBS_TCP: RD(%RD64) No Communication**

  Diagnostic is generated when Modbus slave receives no data for a "Master disconnected timeout" time

  Solution: check if all wires are correct from Modbus Master to Modbus Slave

- **MBS_TCP: RD(%RD64) TCP Socket Error 2nd Port**

  Diagnostic is generated when the socket can not be used, while active the plugin will try to open this socket/port.

- **MBS_TCP: RD(%RD64) No Communication 2nd Port**

  Diagnostic is generated when Modbus slave receives no data for a "Master disconnected timeout" time

  Solution: check if all wires are correct from Modbus Master to Modbus Slave

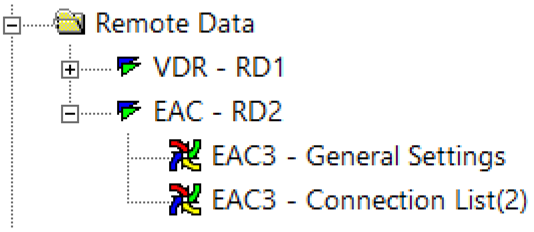
Legend Tag	Replaced by
(RD%64)	Remote data number

See also:

Modbus Slave TCP Plugin
Modbus Slave TCP general settings

EAC3 Plugin

After selecting ‘Plugins’ and EAC



- Remote Data
  - VDR - RD1
  - EAC - RD2
    - EAC3 - General Settings
    - EAC3 - Connection List(2)

There are several items of this plugin:

1. EAC3 - General Settings
2. EAC3 - Connection List

AutroCARGO 2000 Product contains a EAC-300 unit the IO-Processor talk to.

ASAP (AUTRONICA STANDARD ASCII PROTOCOL) is a protocol for asynchronous serial communication between AUTRONICA products and systems.

The describes the protocol frame contents for data exchange to/from the EAC-300 microcomputer is based on the ASAP protocol.
We will use the ASAP Definitions in this document.

The protocol only applies to point to point half duplex transmission mode where the EAC-300 microcomputer is regarded as a slave to a IO-Processor.
The transmission is done by RS 232C, by 4 20 mA current loop or by RS-422.

The following ASAP directive is used to request data for a number of sensors:

#EA0CxxCSAnnn

E means a data read from the EAC-300 microcomputer
xx is function code, see table below
nnn is number of data/sensor values to read
is frame parity character, see calculation example chapter 5.
is carriage return character

This is the answer from EAC-300:

#EA0CxxCSAnnn<data 1><data 2>...<data nnn><Block parity><fp><CR>

<data 1> to <data nnn> is 7 characters each as shown in column Format in the table below.
<Block parity> is 7 characters and is the sum of all values after removing the comma.
<fp> is frame parity character, see calculation example chapter 5.
<CR> is carriage return character>

Example:

Reading 4 ullage values from EAC-300 with function code 40:

ASAP request: #EA0C40CSA004<CR>
ASAP answer: #EA0C40CSA004019.060004.360013.897013.8340051151l<CR>

That is:
ullage1 = 019.060, ullage2 = 004.360, ullage3 = 013.897, ullage4 = 013.834
Block parity = 0051151

Function code summary

* = normally for Autronica use only

xx = function number	Format, 7 characters	<unit>
40 = Cargo ullage	nnn.nnn	<m >
41 = Cargo level	nnn.nnn	<m >
42 = Cargo volume	nnnn.nn	<m3 >
..	nnnnn.n	(large tanks)
43*= Cargo line press. alarms	nnnnnnn	<integer>
44 = Cargo weight	nnnn.nn	<Mt >
..	nnnnn.n	(large tanks)
45*= Cargo avg. temp. in liquid alarms	nnnnnnn	<integer>
46 = Cargo avg. temperature in liquid	nnnnn.n	<°C >
..	nnnn.nn	(LNG ships)
47 = Cargo calc. Density	nn.nnnn	<Mt/m3 >
48 = Inert gas pressure	nnnnnnn	<mmH2O >
50 = Cargo weight rate	nnnnn.n	<Mt/h >
51*= Cargo ullage alarms	nnnnnnn	<integer>
52*= HB, Cargo tank dimension	nnnnnnn	<mm >
53*= HR, Cargo tank dimension	nnnnnnn	<mm >
54*= HT, Cargo tank dimension	nnnnnnn	<mm >
55*= Cargo temp. sensors error	nnnnnnn	<integer>
56 = Cargo line pressure	nnnn.nn	<Bar >
57*= Inert gas alarms	nnnnnnn	<integer>
58*= Vapour alarms	nnnnnnn	<integer>
59 = Vapour pressure	nnnnnnn	<mmH2O >
60*= LON alarms	nnnnnnn	<integer>
70 = Water ballast level	nnnnnnn	<mm >
71 = Water ballast volume	nnnnn.n	<m3 >
72*= Water ballast alarm	nnnnnnn	<integer>
73 = Service level	nnnnnn	<mm >
74 = Service tanks volume	nnnnn.n	<m3 >
75 = Service temperature	nnnnn.n	<°C >
76*= Service level alarms	nnnnnnn	<integer>

77 = Misc. pressure	nnnn.nn	<Bar >
78 = Misc. temperature	nnnnn.n	<°C >
79*= Misc. pressure alarms	nnnnnnn	<integer>
80 = Atm/Draft/Trim/List	Sensor 1..17	next page
81 = Cargo lower temperature	nnnnn.n	<°C >
..	nnnn.nn	(LNG ships)
82 = Cargo mid temperature	nnnnn.n	<°C >
..	nnnn.nn	(LNG ships)
83 = Cargo upper temperature	nnnnn.n	<°C >
..	nnnn.nn	(LNG ships)
86*= Class reason error codes	...	...
87*= Service temperature alarms	nnnnnnn	<integer>
88*= Misc. temperature alarms	nnnnnnn	<integer>
89*= LR, Cargo tank dimension	nnnnnnn	<mm >
90*= Reserved	-	-
91 = Cargo 4th temperature	nnnnn.n	<°C >
..	nnnn.nn	(LNG ships)
92 = Cargo 5th temperature	nnnnn.n	<°C >
..	nnnn.nn	(LNG ships)
93 = Resistance 1 value	nnnnnnn	<0.01 %>
94 = Resistance 2 value	nnnnnnn	<0.01 %>
95 = Cargo avg. temperature in gas alarms	nnnnnnn	<integer>
96 = Cargo avg. temperature in gas	nnnnn.n	<°C >
..	nnnn.nn	(LNG ships)
97 = Cargo temp. sensor in liquid indication	nnnnnnn	<integer>
98 = Cargo 6th temperature	nnnnn.n	<°C >
..	nnnn.nn	(LNG ships)

Sensor definition for function number 80:

xx = function number	**Format, 7 characters**	**<unit>**
1 = Atm. Pressure	nnnnnnn	<mmH2O >
2 = Draft Fore calculated	nnnnnnn	<mm >
3 = Draft Mid calculated	nnnnnnn	<mm >
4 = Draft Aft calculated	nnnnnnn	<mm >

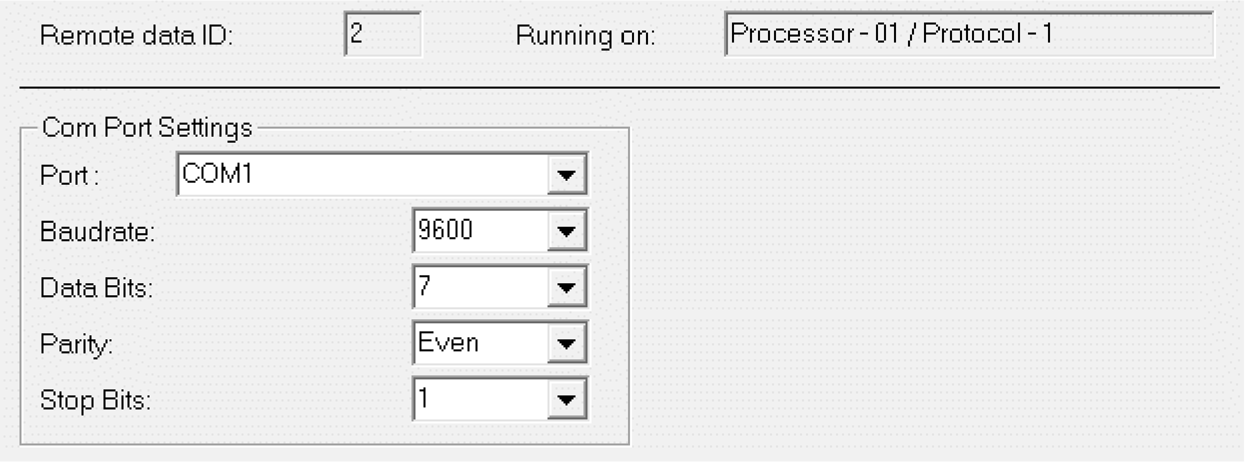
5 = Trim	nnnnnnn	<mm >
6 = List	nnnnnnn	<0.01 degr.>
7* = Manual trim	nnnnnnn	<mm > 0099999=auto trim
8* = Manual list	nnnnnnn	<0.01 degr.> 0099999=auto list
9* = Manual atm. Pressure	nnnnnnn	<mmH2O > 0099999=auto atm.
10*= Seagoing mode	nnnnnnn	0099999= not seagoing
11*= Draft sensor errors	nnnnnnn	<integer>
12 = Draft Mid Port calculated	nnnnnnn	<mm >
13 = Draft Mid Stb calculated	nnnnnnn	<mm >
14*= Draft Fore sensor	nnnnnnn	<mm >
15*= Draft Mid Port sensor	nnnnnnn	<mm >
16*= Draft Mid Stb sensor	nnnnnnn	<mm >
17*= Draft Aft sensor	nnnnnnn	<mm >

Shortcuts

Icon:

Place: Plugins\

EAC3 – General Settings



Remote data ID: 2

Running on: Processor - 01 / Protocol - 1

Com Port Settings

Port : COM1

Baudrate: 9600

Data Bits: 7

Parity: Even

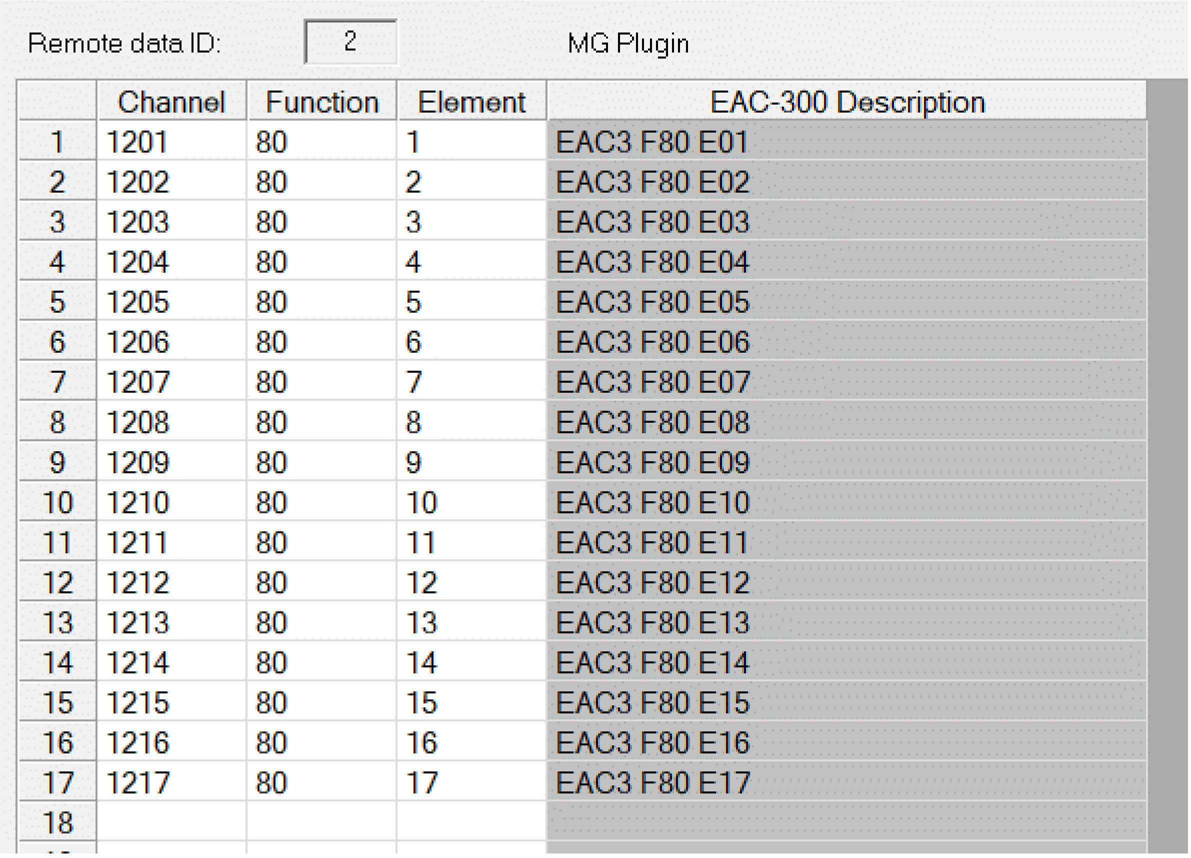
Stop Bits: 1

Setting	Description	Default
Com Port	Comport to use (1-4)	2
Baudrate	Communication Speed (1200, 2400, 4800, 9600, 19200)	1200
Data Bits	Number of Data bits (7, 8)	8
Parity	Communication Parity (None, Odd, Even, Space, Mark)	None
Stop Bits	Number of Stop Bits (0, 1, 2)	1

See also:

EAC3 Plugin

EAC3 – Connection List



Remote data ID: 2 MG Plugin

	Channel	Function	Element	EAC-300 Description
1	1201	80	1	EAC3 F80 E01
2	1202	80	2	EAC3 F80 E02
3	1203	80	3	EAC3 F80 E03
4	1204	80	4	EAC3 F80 E04
5	1205	80	5	EAC3 F80 E05
6	1206	80	6	EAC3 F80 E06
7	1207	80	7	EAC3 F80 E07
8	1208	80	8	EAC3 F80 E08
9	1209	80	9	EAC3 F80 E09
10	1210	80	10	EAC3 F80 E10
11	1211	80	11	EAC3 F80 E11
12	1212	80	12	EAC3 F80 E12
13	1213	80	13	EAC3 F80 E13
14	1214	80	14	EAC3 F80 E14
15	1215	80	15	EAC3 F80 E15
16	1216	80	16	EAC3 F80 E16
17	1217	80	17	EAC3 F80 E17
18				

Field	Description
Channel	Channel number to update (Analog in; Digital in; Analog out; Digital out)
Function	function definition
Element	element from function
EAC-300 Description	Description, information

Remarks:

For faster configuration it is possible to use CTRL+D:

- configure one complete row
- select one column and multiple empty rows
- press CTRL+D, automaticially a default will filled in

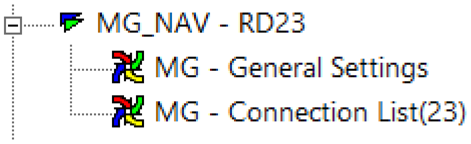
By double-clicking on channel column header sort action is performed.

See also:

EAC3 Plugin

MG – Plugin

After selecting ‘Plugins’ and LineIn



MG_NAV - RD23
MG - General Settings
MG - Connection List(23)

There are several items of this plugin:

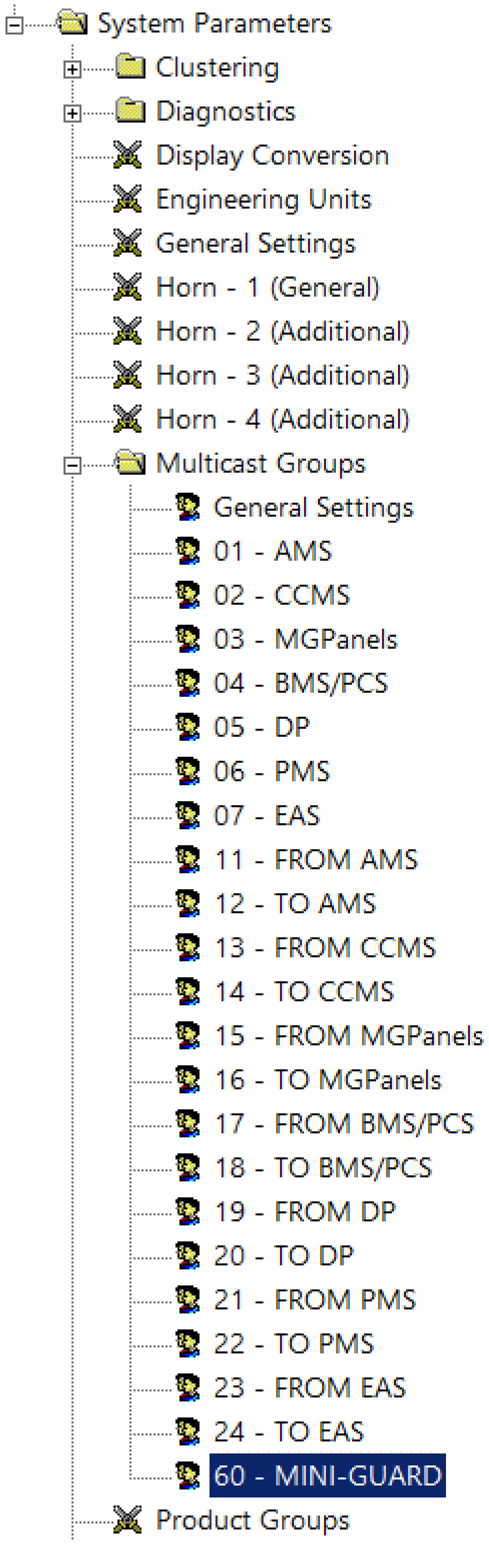
1. MG - General Settings

2. MG - Connection List

MG Protocol runs only XP processor

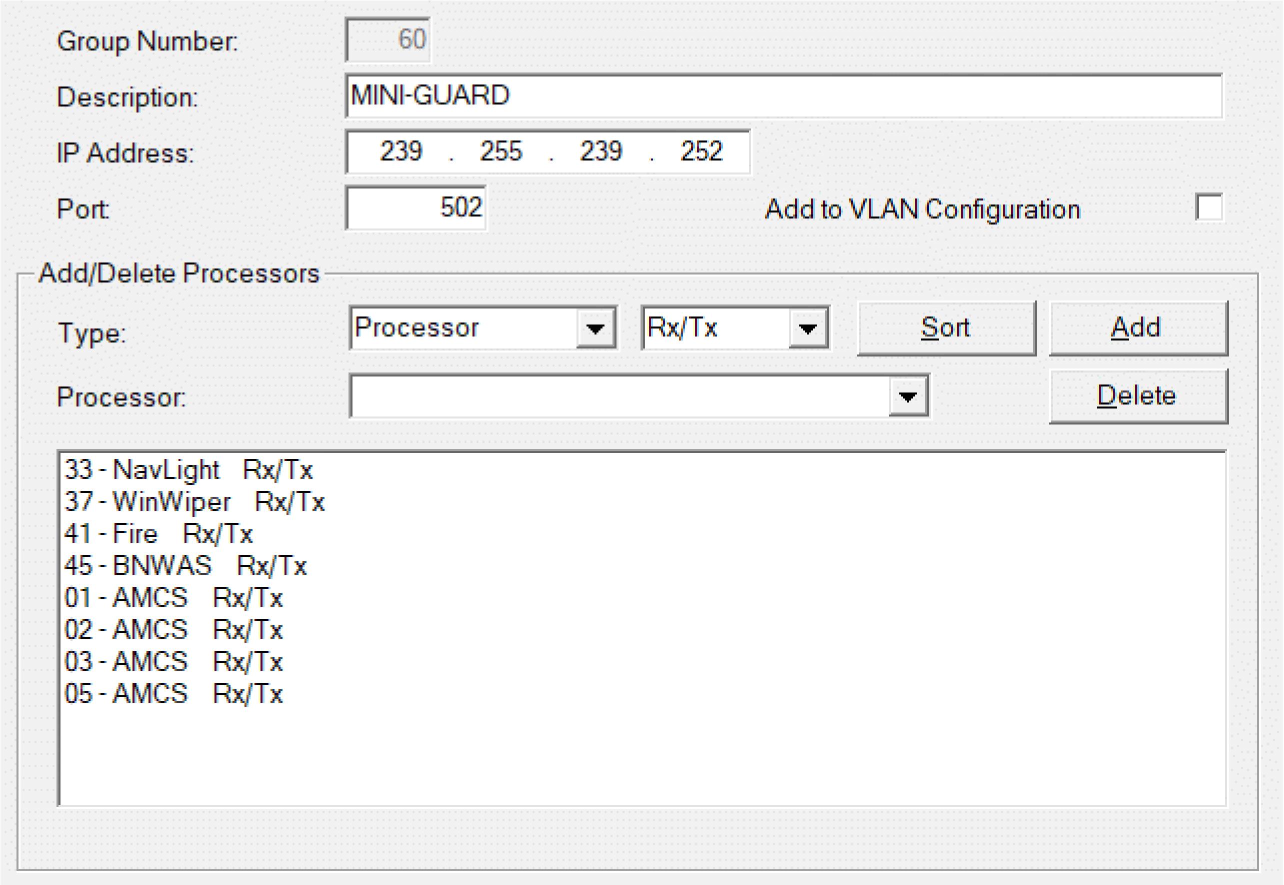
MG = MEGA-Guard
It connects source channels from MEGA-Guard (Stand-Alone / Mini-Guard) Panels to local XP channels by sending/receiving channel information over ethernet.

It uses a special multicast group (=60) to accomplish this:



- System Parameters
  - Clustering
  - Diagnostics
  - Display Conversion
  - Engineering Units
  - General Settings
  - Horn - 1 (General)
  - Horn - 2 (Additional)
  - Horn - 3 (Additional)
  - Horn - 4 (Additional)
  - Multicast Groups
    - General Settings
    - 01 - AMS
    - 02 - CCMS
    - 03 - MGPanels
    - 04 - BMS/PCS
    - 05 - DP
    - 06 - PMS
    - 07 - EAS
    - 11 - FROM AMS
    - 12 - TO AMS
    - 13 - FROM CCMS
    - 14 - TO CCMS
    - 15 - FROM MGPanels
    - 16 - TO MGPanels
    - 17 - FROM BMS/PCS
    - 18 - TO BMS/PCS
    - 19 - FROM DP
    - 20 - TO DP
    - 21 - FROM PMS
    - 22 - TO PMS
    - 23 - FROM EAS
    - 24 - TO EAS
    - 60 - MINI-GUARD
  - Product Groups

Treeview from the PAL



Group Number: 60

Description: MINI-GUARD

IP Address: 239 . 255 . 239 . 252

Port: 502

Add to VLAN Configuration □

Add/Delete Processors

Type: Processor Rx/Tx

Processor:

```
33 - NavLight   Rx/Tx
37 - WinWiper   Rx/Tx
41 - Fire   Rx/Tx
45 - BNWAS   Rx/Tx
01 - AMCS   Rx/Tx
02 - AMCS   Rx/Tx
03 - AMCS   Rx/Tx
05 - AMCS   Rx/Tx
```

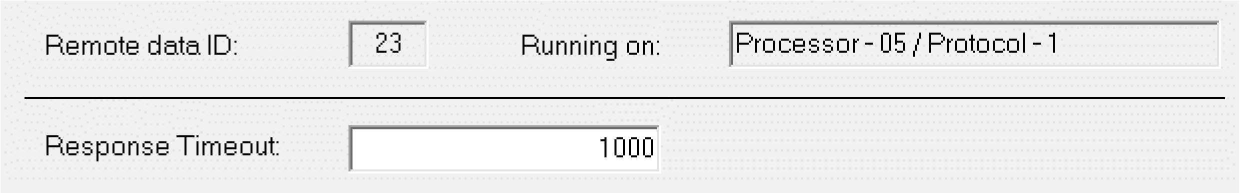
Example: Multicast group 60 configured.

Shortcuts

Icon:

Place: Plugins\

MG – General Settings



Remote data ID: 23

Running on: Processor - 05 / Protocol - 1

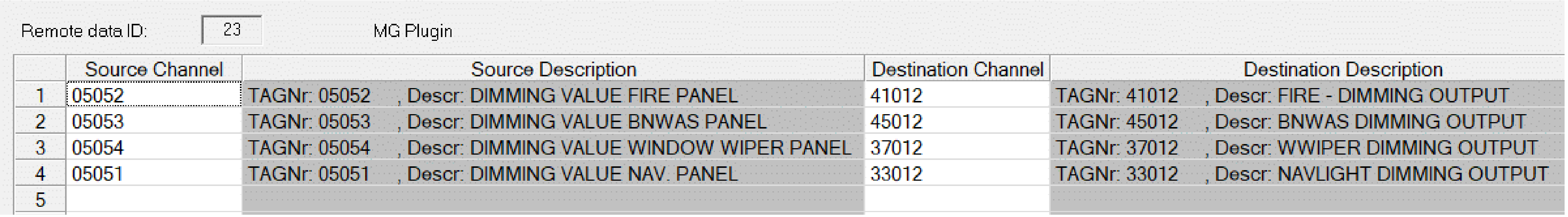
Response Timeout: 1000

Setting	Description	Default
Response Timeout	Timeout before an error will be triggered	1000

See also:

MG Plugin

MG – Connection List



Remote data ID: 23 MG Plugin

	Source Channel	Source Description	Destination Channel	Destination Description
1	05052	TAGNr: 05052 , Descr: DIMMING VALUE FIRE PANEL	41012	TAGNr: 41012 , Descr: FIRE - DIMMING OUTPUT
2	05053	TAGNr: 05053 , Descr: DIMMING VALUE BNWAS PANEL	45012	TAGNr: 45012 , Descr: BNWAS DIMMING OUTPUT
3	05054	TAGNr: 05054 , Descr: DIMMING VALUE WINDOW WIPER PANEL	37012	TAGNr: 37012 , Descr: WWIPER DIMMING OUTPUT
4	05051	TAGNr: 05051 , Descr: DIMMING VALUE NAV. PANEL	33012	TAGNr: 33012 , Descr: NAVLIGHT DIMMING OUTPUT
5				

Field	**Description**
Source Channel	source Channel number, which value needs to be sent
Source Description	channel tag and channel description
Destination Channel	destination Channel number where received value is stored and used
Destination Description	channel tag and channel description

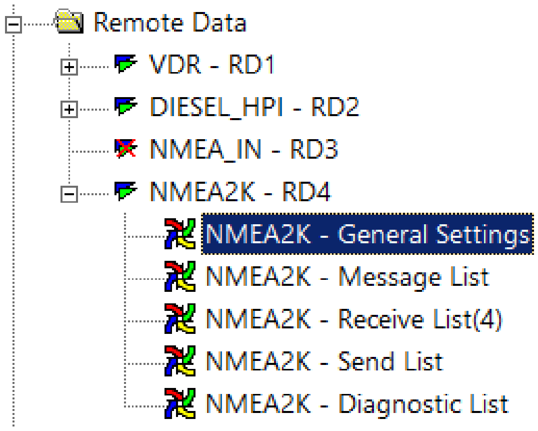
Example shows AMS channels from XP05, which are sent to serveral panels, to create a central dimming function.

See also:

MG Plugin

NMEA2K – Plugin

After selecting 'Plugins' and NMEA2K – RD xx



There are several items of this plugin:

1. NMEA2K - General Settings
2. NMEA2K - Message List
3. NMEA2K - Connection List
4. NMEA2K - Send List
5. NMEA2K - Diagnostic List

Introduction (from Internet)

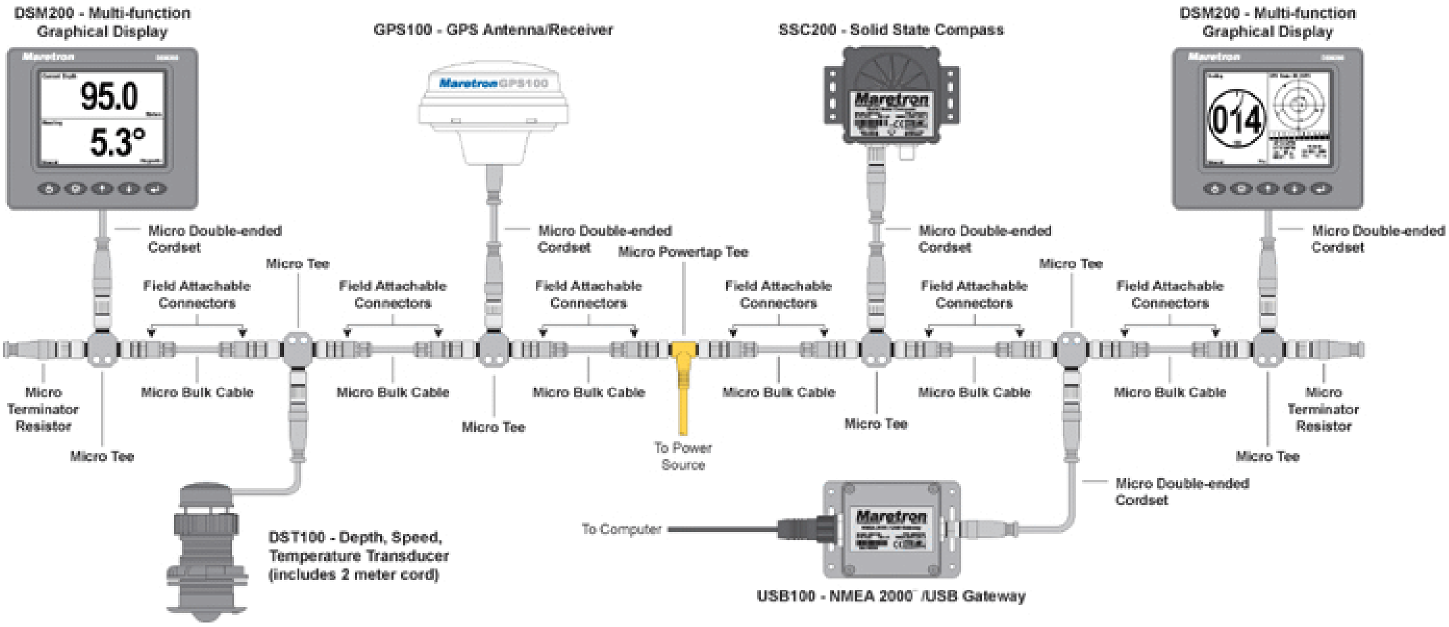
NMEA 2000, abbreviated to NMEA2k or N2K and standardised as IEC 61162-3, is a plug-and-play communications standard used for connecting marine sensors and display units within ships and boats.
Communication runs at 250 kilobits-per-second and allows any sensor to talk to any display unit or other device compatible with NMEA 2000 protocols.
Electrically, NMEA 2000 is compatible with the Controller Area Network ("CAN Bus") used on road vehicles and fuel engines.
The higher-level protocol format is based on SAE J1939, with specific messages for the marine environment.

It takes a backbone

The key to a NMEA 2000 network is the "backbone," (or "trunk") a central cable running throughout the vessel.
The backbone is connected to the boat's power and ground, and your NMEA 2000 devices are connected to the backbone
via special T-connectors, sometimes referred to as "drop tees."

The beauty of this networking system is that it allows different devices (even ones from different manufacturers) to communicate with one another.
It's ahead of the stacks of various devices found on so many ships, and it's a lot easier than wiring all this stuff in one by one.

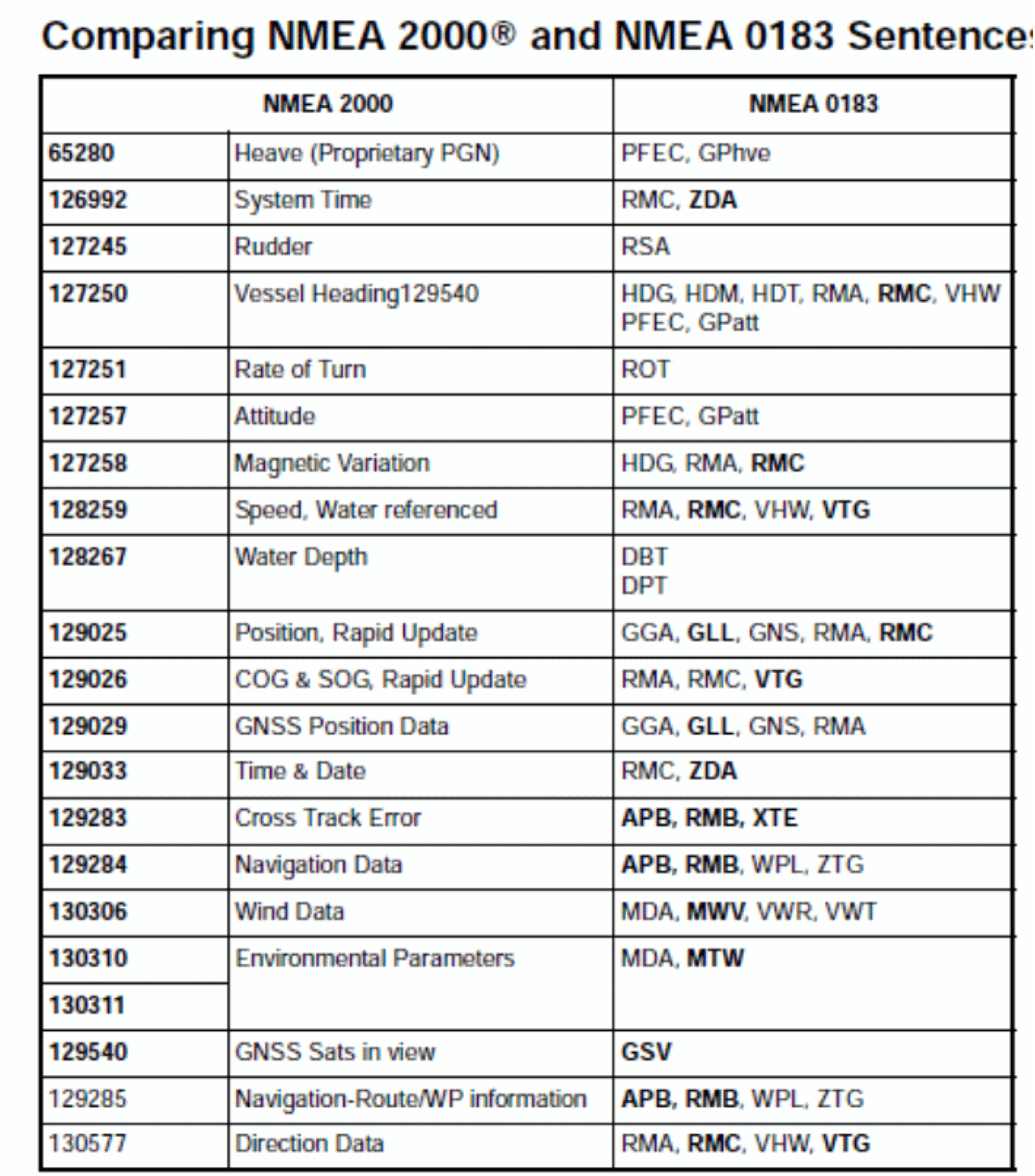
We could use the phrase "plug and play," which is a reasonably accurate description of the connection itself.
The actual installation will probably involve some challenges,
but that really depends on your ship and the network you're trying to build.
Several marine electronics manufacturers have networks of their own. Some of them work with NMEA 2000,
while others are proprietary, and require adapters to plug into a backbone.
When you see the phrase "NMEA 2000 Certified" on a piece of electronic gear, from,
let's say, FUSION, you'll know that the device will work with your existing NMEA 2000 network.


DSM200 - Multi-function Graphical Display
GPS100 - GPS Antenna/Receiver
SSC200 - Solid State Compass
DSM200 - Multi-function Graphical Display
95.0
5.3°
014
Maretron
Micro Double-ended Cordset
Micro Double-ended Cordset
Micro Powertap Tee
Micro Double-ended Cordset
Micro Double-ended Cordset
Micro Tee
Micro Tee
Field Attachable Connectors
Field Attachable Connectors
Field Attachable Connectors
Field Attachable Connectors
Field Attachable Connectors
Field Attachable Connectors
Micro Terminator Resistor
Micro Bulk Cable
Micro Bulk Cable
Micro Bulk Cable
Micro Bulk Cable
Micro Bulk Cable
Micro Bulk Cable
Micro Terminator Resistor
Micro Tee
Micro Tee
To Power Source
Micro Tee
Micro Tee
Micro Double-ended Cordset
DST100 - Depth, Speed, Temperature Transducer (includes 2 meter cord)
To Computer
USB100 - NMEA 2000 /USB Gateway

So, how will NMEA 2000 work for me?

The size of the network you build will mostly depend on the size of the ship you have.
On a smaller ships, a receiver and a chartplotter are probably all you need,
but if you have a larger vessel or a frequent need to have remote controllers handy,
we can add more NMEA 2000-compatible devices.
The limit is 50, which is more than the average ship will need.

Next follows a table with NMEA 2000 vs NMEA 0183,
it gives some insight how some general formatters are converted to new PGN.(=Parameter Group Number)



Comparing NMEA 2000® and NMEA 0183 Sentence:

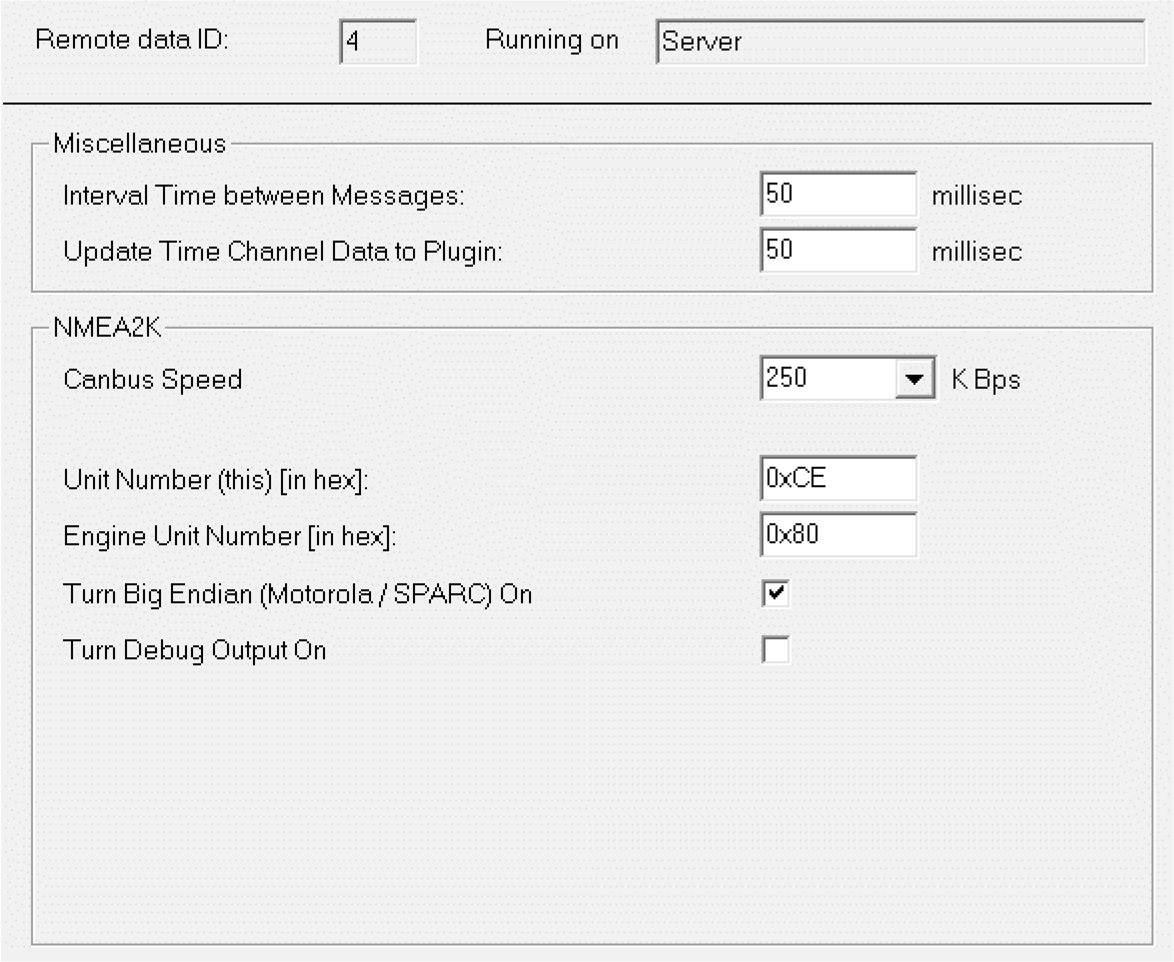
NMEA 2000		NMEA 0183
65280	Heave (Proprietary PGN)	PFEC, GPhve
126992	System Time	RMC, **ZDA**
127245	Rudder	RSA
127250	Vessel Heading129540	HDG, HDM, HDT, RMA, **RMC**, VHW PFEC, GPatt
127251	Rate of Turn	ROT
127257	Attitude	PFEC, GPatt
127258	Magnetic Variation	HDG, RMA, **RMC**
128259	Speed, Water referenced	RMA, **RMC**, VHW, **VTG**
128267	Water Depth	DBT DPT
129025	Position, Rapid Update	GGA, **GLL**, GNS, RMA, **RMC**
129026	COG & SOG, Rapid Update	RMA, RMC, **VTG**
129029	GNSS Position Data	GGA, **GLL**, GNS, RMA
129033	Time & Date	RMC, **ZDA**
129283	Cross Track Error	**APB, RMB, XTE**
129284	Navigation Data	**APB, RMB**, WPL, ZTG
130306	Wind Data	MDA, **MWV**, VWR, VWT
130310	Environmental Parameters	MDA, **MTW**
130311		
129540	GNSS Sats in view	**GSV**
129285	Navigation-Route/WP information	**APB, RMB**, WPL, ZTG
130577	Direction Data	RMA, **RMC**, VHW, **VTG**

Shortcuts

Icon:

Place: Plugins\

NMEA2K – General Settings



Remote data ID: 4 Running on Server

Miscellaneous

Interval Time between Messages: 50 millisec

Update Time Channel Data to Plugin: 50 millisec

NMEA2K

Canbus Speed 250 K Bps

Unit Number (this) [in hex]: 0xCE

Engine Unit Number [in hex]: 0x80

Turn Big Endian (Motorola / SPARC) On ☑

Turn Debug Output On □

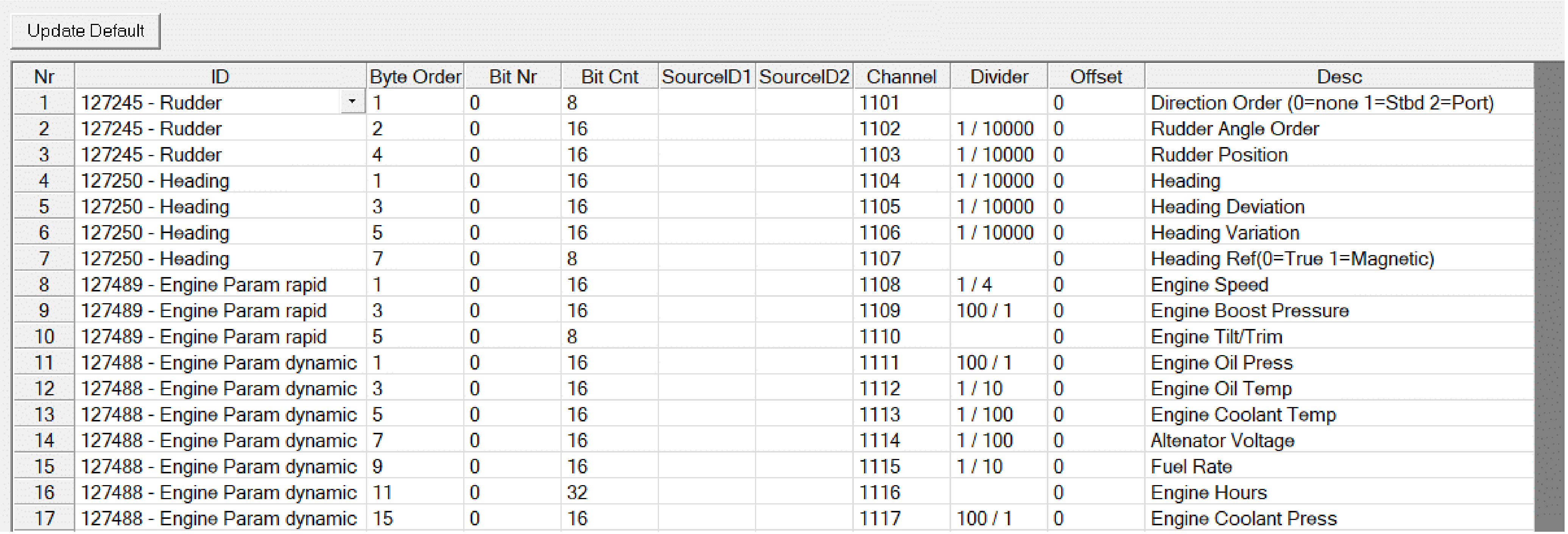
Setting	Description	Default
Interval Time between Messages	Minimal Time between two messages before reporting an error (50-100000 in milli-seconds	1000
Update Time channel Data to Plugin	When this protocol running on Server, IOServer uses this time to update channel values (50-100000 in milli-seconds	1000
Can Speed	Speed Setting for Canbus 125K, 250k, 500k 1M etc.	250k
Unit Number (this) [in hex]	Unit Address Number of Plugin	0xCE
Engine Unit Number [in hex]	Unit Address Number of Engine	0x80
Turn Big Endian (Motorola/SPARC) On	how to handle values greater than 1 byte - yes/no In computing, endianness refers to the order of bytes within a binary representation of a number.	yes
Turn Debug Output On	connect on comport1, text output what messages are received on Canbus - yes/no	no

See also:

NMEA2K Plugin

How to use Debug Output
the description is for CANOpen, but is same for NMEA2K.

NMEA2K – Connection List



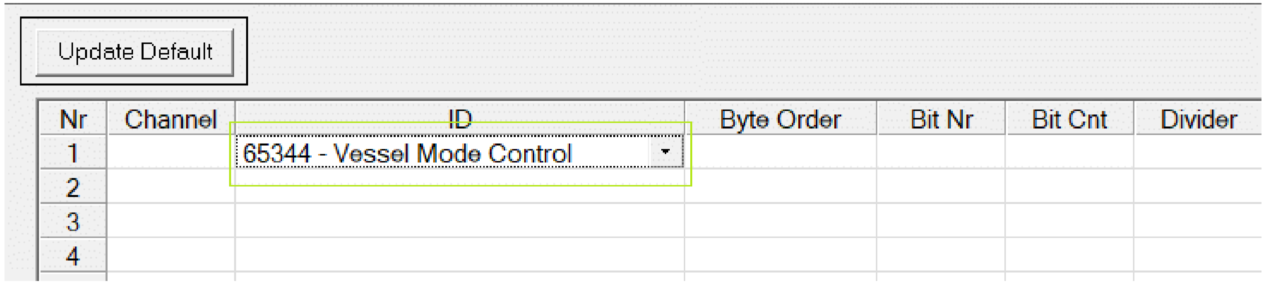
Nr	ID	Byte Order	Bit Nr	Bit Cnt	SourceID1	SourceID2	Channel	Divider	Offset	Desc
1	127245 - Rudder	1	0	8			1101		0	Direction Order (0=none 1=Stbd 2=Port)
2	127245 - Rudder	2	0	16			1102	1 / 10000	0	Rudder Angle Order
3	127245 - Rudder	4	0	16			1103	1 / 10000	0	Rudder Position
4	127250 - Heading	1	0	16			1104	1 / 10000	0	Heading
5	127250 - Heading	3	0	16			1105	1 / 10000	0	Heading Deviation
6	127250 - Heading	5	0	16			1106	1 / 10000	0	Heading Variation
7	127250 - Heading	7	0	8			1107		0	Heading Ref(0=True 1=Magnetic)
8	127489 - Engine Param rapid	1	0	16			1108	1 / 4	0	Engine Speed
9	127489 - Engine Param rapid	3	0	16			1109	100 / 1	0	Engine Boost Pressure
10	127489 - Engine Param rapid	5	0	8			1110		0	Engine Tilt/Trim
11	127488 - Engine Param dynamic	1	0	16			1111	100 / 1	0	Engine Oil Press
12	127488 - Engine Param dynamic	3	0	16			1112	1 / 10	0	Engine Oil Temp
13	127488 - Engine Param dynamic	5	0	16			1113	1 / 100	0	Engine Coolant Temp
14	127488 - Engine Param dynamic	7	0	16			1114	1 / 100	0	Altenator Voltage
15	127488 - Engine Param dynamic	9	0	16			1115	1 / 10	0	Fuel Rate
16	127488 - Engine Param dynamic	11	0	32			1116		0	Engine Hours
17	127488 - Engine Param dynamic	15	0	16			1117	100 / 1	0	Engine Coolant Press

Update Default button, updates fields when it is known to software (hard-coded), see below, same as for send list.

Field	Description
Nr	Number, row number
ID	ID Number + description, see Message List
Byte Order	start byte in message, 0..255
Bit Nr	start bit in message, 0..7
Bit Cnt	number of bits, 1 = digital, 8 = byte, 16 = word, 32 = dword
Source ID 1	first byte in message should equal to .. Leaving this field empty, it will not be used
Source ID 2	second byte in message should equal to .. Leaving this field empty, it will not be used
Channel	Channel to retrieve value/status from eq. Leaving this field empty, this will delete entry
Divider	received value with be calculated with this divider before placing it into a channel value

	1/100 or *50 Remark: *50 will be shown as 50 / 1
Offset	received value will be added to this offset before placing it into a channel value
Desc	Details description depends on contents on message

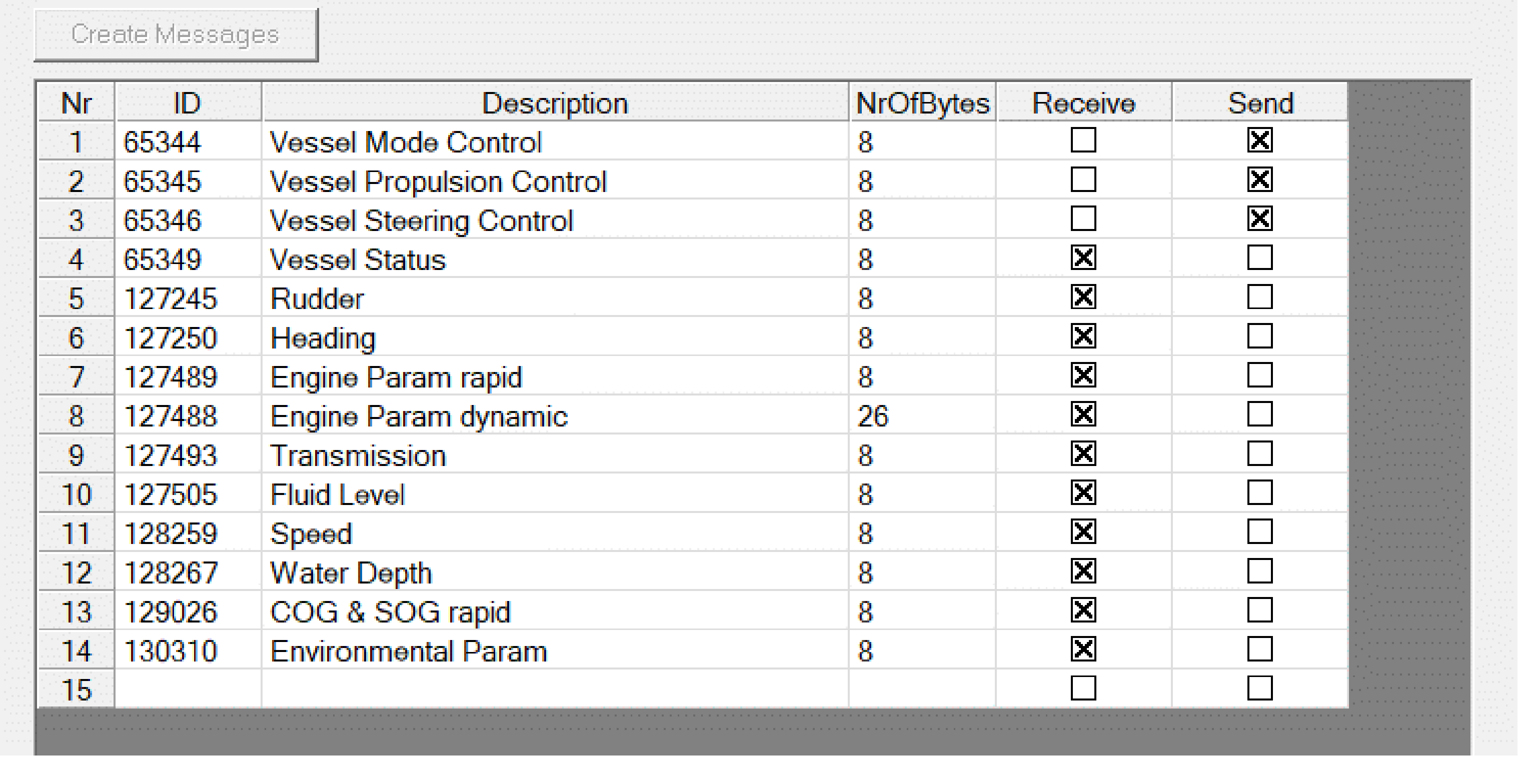
Update Default:
Add default configuration fields of a certain ID string.
First select ID on empty line and afterthat press on 'Update Default' button.
Fields (which are known by the plugin) are automaticially filled in.



Nr	Channel	ID	Byte Order	Bit Nr	Bit Cnt	Divider
1		65344 - Vessel Mode Control				
2						
3						
4						

See also: NMEA2K Plugin

NMEA2K – Message List



Nr	ID	Description	NrOfBytes	Receive	Send
1	65344	Vessel Mode Control	8	□	☒
2	65345	Vessel Propulsion Control	8	□	☒
3	65346	Vessel Steering Control	8	□	☒
4	65349	Vessel Status	8	☒	□
5	127245	Rudder	8	☒	□
6	127250	Heading	8	☒	□
7	127489	Engine Param rapid	8	☒	□
8	127488	Engine Param dynamic	26	☒	□
9	127493	Transmission	8	☒	□
10	127505	Fluid Level	8	☒	□
11	128259	Speed	8	☒	□
12	128267	Water Depth	8	☒	□
13	129026	COG & SOG rapid	8	☒	□
14	130310	Environmental Param	8	☒	□
15				□	□

Create Messages Button to create some messages for communication with NMEA2K bus.

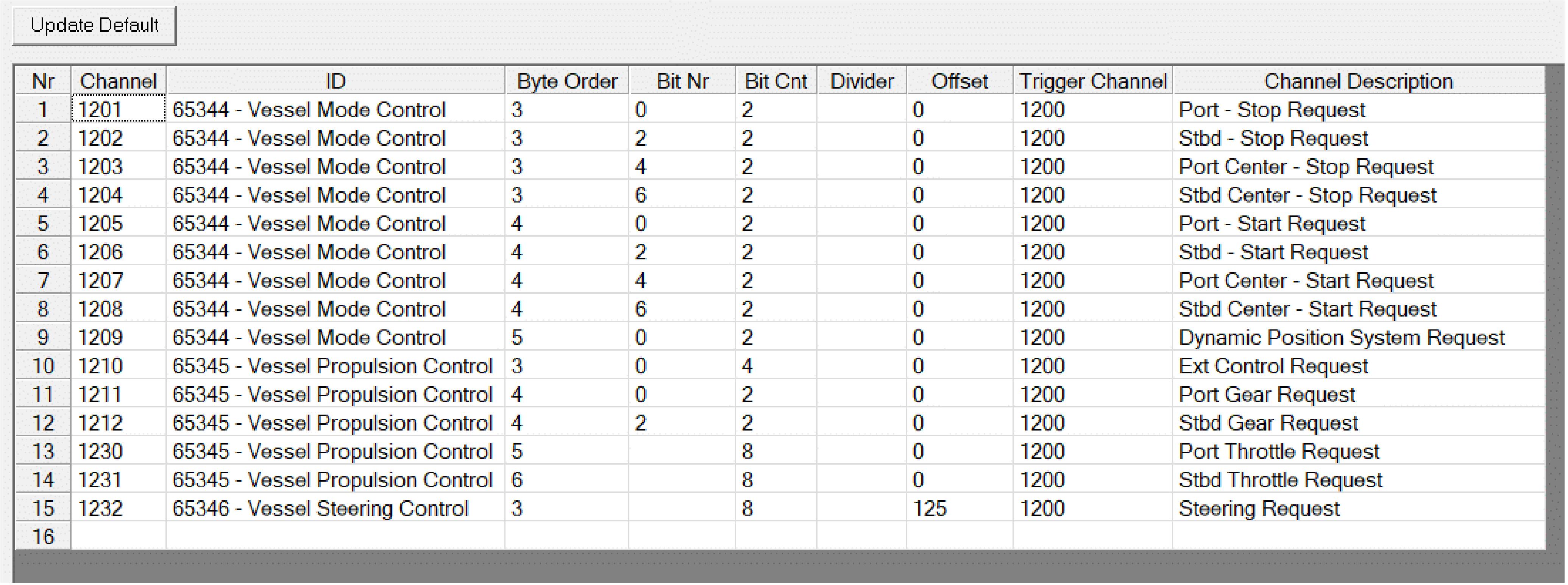
Field	Description
ID	indentifier of message which will be received or sent by this plugin
Description	description of indentifier
Nr Of Bytes	data size of message
Receive	this message will be received from external device like Engine, GPS or Sensor
Send	this message will be sent to external device

ID Enter your PGN in decimal.
Example: 127250 (Vessel Heading)

See also:

NMEA2K Plugin

NMEA2K – Send List

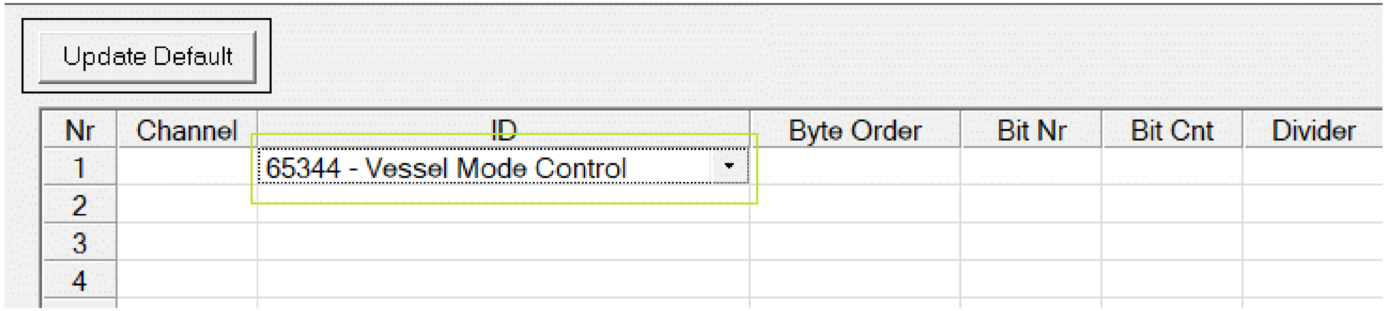


Nr	Channel	ID	Byte Order	Bit Nr	Bit Cnt	Divider	Offset	Trigger Channel	Channel Description
1	1201	65344 - Vessel Mode Control	3	0	2		0	1200	Port - Stop Request
2	1202	65344 - Vessel Mode Control	3	2	2		0	1200	Stbd - Stop Request
3	1203	65344 - Vessel Mode Control	3	4	2		0	1200	Port Center - Stop Request
4	1204	65344 - Vessel Mode Control	3	6	2		0	1200	Stbd Center - Stop Request
5	1205	65344 - Vessel Mode Control	4	0	2		0	1200	Port - Start Request
6	1206	65344 - Vessel Mode Control	4	2	2		0	1200	Stbd - Start Request
7	1207	65344 - Vessel Mode Control	4	4	2		0	1200	Port Center - Start Request
8	1208	65344 - Vessel Mode Control	4	6	2		0	1200	Stbd Center - Start Request
9	1209	65344 - Vessel Mode Control	5	0	2		0	1200	Dynamic Position System Request
10	1210	65345 - Vessel Propulsion Control	3	0	4		0	1200	Ext Control Request
11	1211	65345 - Vessel Propulsion Control	4	0	2		0	1200	Port Gear Request
12	1212	65345 - Vessel Propulsion Control	4	2	2		0	1200	Stbd Gear Request
13	1230	65345 - Vessel Propulsion Control	5		8		0	1200	Port Throttle Request
14	1231	65345 - Vessel Propulsion Control	6		8		0	1200	Stbd Throttle Request
15	1232	65346 - Vessel Steering Control	3		8		125	1200	Steering Request
16									

Field	Description
Nr	row index of this grid
Channel	Channel contains value to be sent
ID	NMEA2K ID for Message
Byte Order	byte number where to store value in message, 0-7
Bit Nr	bit number where value to start, 0-7
Bit Cnt	number of bits, 1 = digital, 8 = byte, 16 = word, 32 = dword
Divider	send value will be calculated with this divider before putting it into the message 1/100 or *50 Remark: *50 will be shown as 50 / 1
Offset	send value will be calculated with this offset before putting it into the message

	channel range -100 .. 100, send value range 0 - 200 (positive)
Trigger Channel	when digital channel is status on, message will be sent, leaving blank: message is always sent
Channel Description	Description of Channel

Update Default:
Add default configuration fields of a certain ID string.
First select ID on empty line and afterthat press on 'Update Default' button.
Fields (which are known by the plugin) are automaticially filled in.

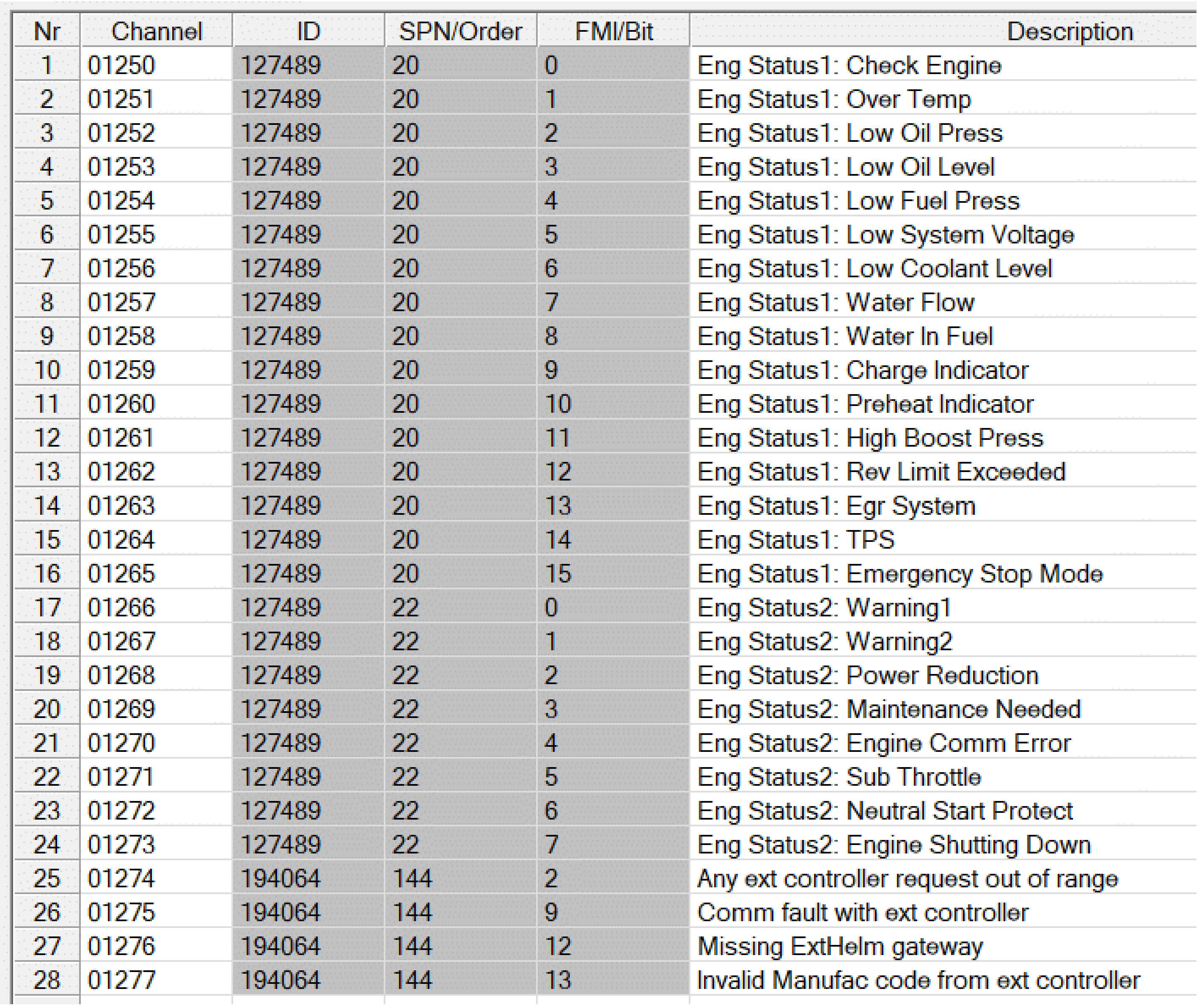


Nr	Channel	ID	Byte Order	Bit Nr	Bit Cnt	Divider
1		65344 - Vessel Mode Control				
2						
3						
4						

See also:

NMEA2K Plugin

NMEA2K – Diagnostic List



Nr	Channel	ID	SPN/Order	FMI/Bit	Description
1	01250	127489	20	0	Eng Status1: Check Engine
2	01251	127489	20	1	Eng Status1: Over Temp
3	01252	127489	20	2	Eng Status1: Low Oil Press
4	01253	127489	20	3	Eng Status1: Low Oil Level
5	01254	127489	20	4	Eng Status1: Low Fuel Press
6	01255	127489	20	5	Eng Status1: Low System Voltage
7	01256	127489	20	6	Eng Status1: Low Coolant Level
8	01257	127489	20	7	Eng Status1: Water Flow
9	01258	127489	20	8	Eng Status1: Water In Fuel
10	01259	127489	20	9	Eng Status1: Charge Indicator
11	01260	127489	20	10	Eng Status1: Preheat Indicator
12	01261	127489	20	11	Eng Status1: High Boost Press
13	01262	127489	20	12	Eng Status1: Rev Limit Exceeded
14	01263	127489	20	13	Eng Status1: Egr System
15	01264	127489	20	14	Eng Status1: TPS
16	01265	127489	20	15	Eng Status1: Emergency Stop Mode
17	01266	127489	22	0	Eng Status2: Warning1
18	01267	127489	22	1	Eng Status2: Warning2
19	01268	127489	22	2	Eng Status2: Power Reduction
20	01269	127489	22	3	Eng Status2: Maintenance Needed
21	01270	127489	22	4	Eng Status2: Engine Comm Error
22	01271	127489	22	5	Eng Status2: Sub Throttle
23	01272	127489	22	6	Eng Status2: Neutral Start Protect
24	01273	127489	22	7	Eng Status2: Engine Shutting Down
25	01274	194064	144	2	Any ext controller request out of range
26	01275	194064	144	9	Comm fault with ext controller
27	01276	194064	144	12	Missing ExtHelm gateway
28	01277	194064	144	13	Invalid Manufac code from ext controller

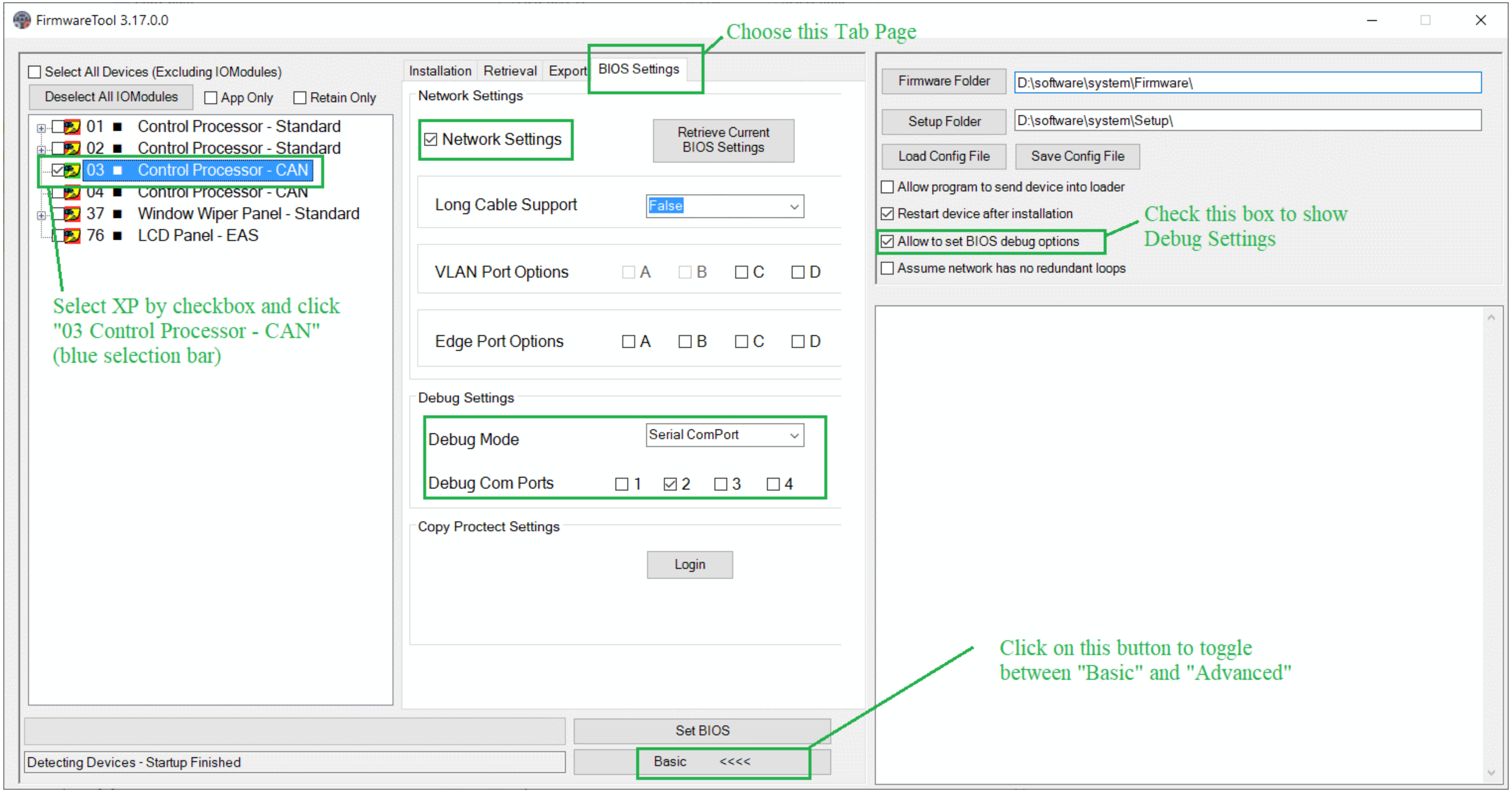
Field	Description
Nr	row index of this grid
Channel	Digital Channel which will be triggered into alarm/status high when fault occurs
ID	Parameter Group Number
SPN/Order	Suspect Parameter Number or Byte Order
FMI/Bit	Failure Mode Identifier or Bit Number which to be checked
Description	Alarm Description

See also:

NMEA2K Plugin

J1939 - How to use Debug Output with UDP Port

First XP must be set ready to put out debug messages.
Start FirmwareTool to Set BIOS settings for this XP.



☐ Select All Devices (Excluding IOModules)

☐ App Only ☐ Retain Only

- ☐ 01 ■ Control Processor - Standard
- ☐ 02 ■ Control Processor - Standard
- ☑ 03 ■ Control Processor - CAN
- ☐ 04 ■ Control Processor - CAN
- ☐ 37 ■ Window Wiper Panel - Standard
- ☐ 76 ■ LCD Panel - EAS

Network Settings

☑ Network Settings

Long Cable Support: False

VLAN Port Options ☐ A ☐ B ☐ C ☐ D

Edge Port Options ☐ A ☐ B ☐ C ☐ D

Debug Settings

Debug Mode: Serial ComPort

Debug Com Ports ☐ 1 ☑ 2 ☐ 3 ☐ 4

Copy Proctect Settings

Detecting Devices - Startup Finished

Firmware Folder: D:\software\system\Firmware\

Setup Folder: D:\software\system\Setup\

☐ Allow program to send device into loader

☑ Restart device after installation

☑ Allow to set BIOS debug options

☐ Assume network has no redundant loops

Choose this Tab Page

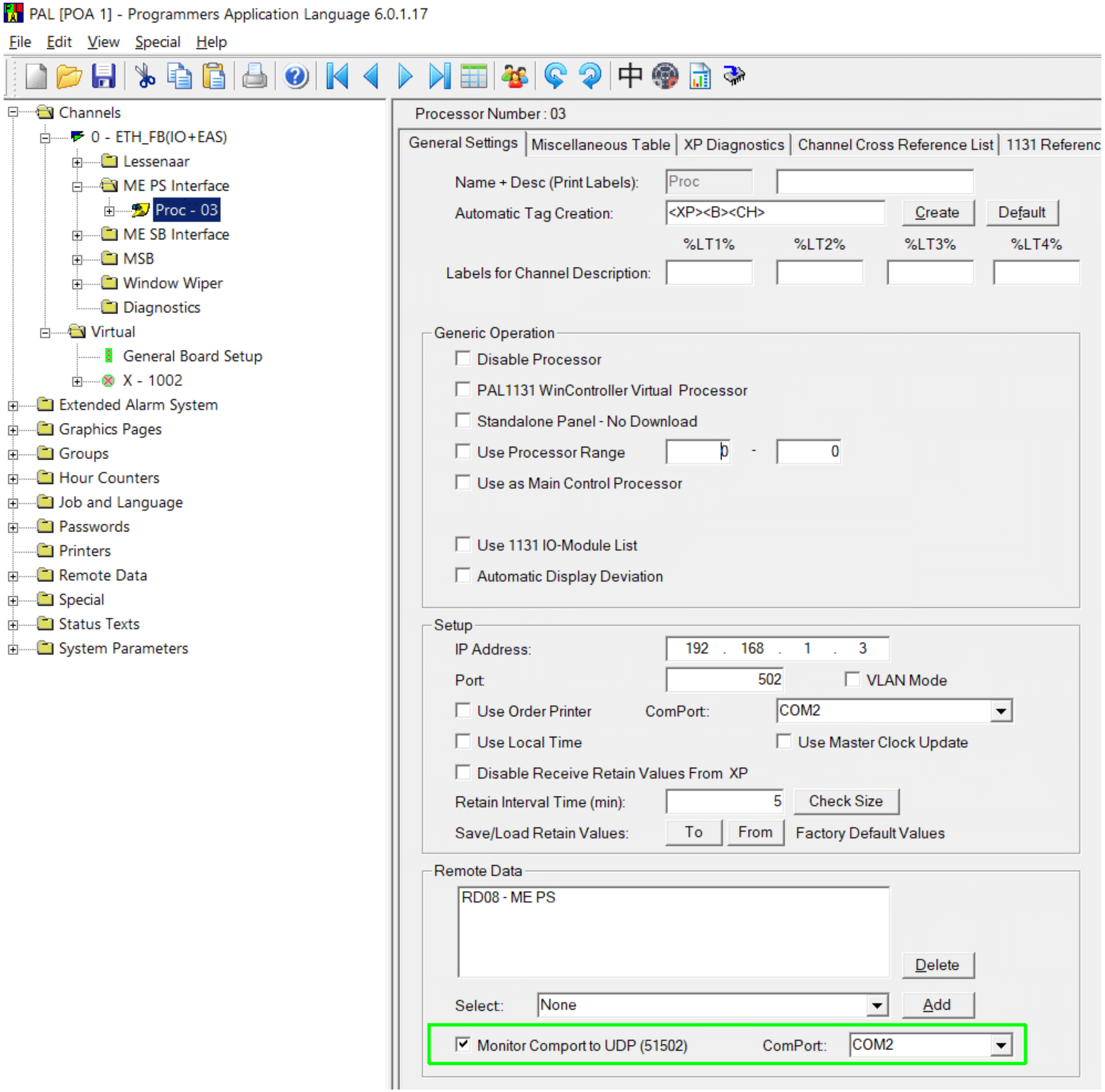
Select XP by checkbox and click "03 Control Processor - CAN" (blue selection bar)

Check this box to show Debug Settings

Click on this button to toggle between "Basic" and "Advanced"

After setting all selections, see screenshot from above.
Press "Set BIOS" button to upload the new settings to the XP.

Set option **'Monitor Comport To UDP 51502'** to on by checking this box (see: Proc, Tab Page General Settings)
and select ComPort COM2



Processor Number : 03

Name + Desc (Print Labels): Proc

Automatic Tag Creation: <XP><B><CH>

Labels for Channel Description: %LT1% %LT2% %LT3% %LT4%

Generic Operation

□ Disable Processor

□ PAL1131 WinController Virtual Processor

□ Standalone Panel - No Download

□ Use Processor Range 0 - 0

□ Use as Main Control Processor

□ Use 1131 IO-Module List

□ Automatic Display Deviation

Setup

IP Address: 192 . 168 . 1 . 3

Port: 502 □ VLAN Mode

□ Use Order Printer ComPort: COM2

□ Use Local Time □ Use Master Clock Update

□ Disable Receive Retain Values From XP

Retain Interval Time (min): 5

Save/Load Retain Values: Factory Default Values

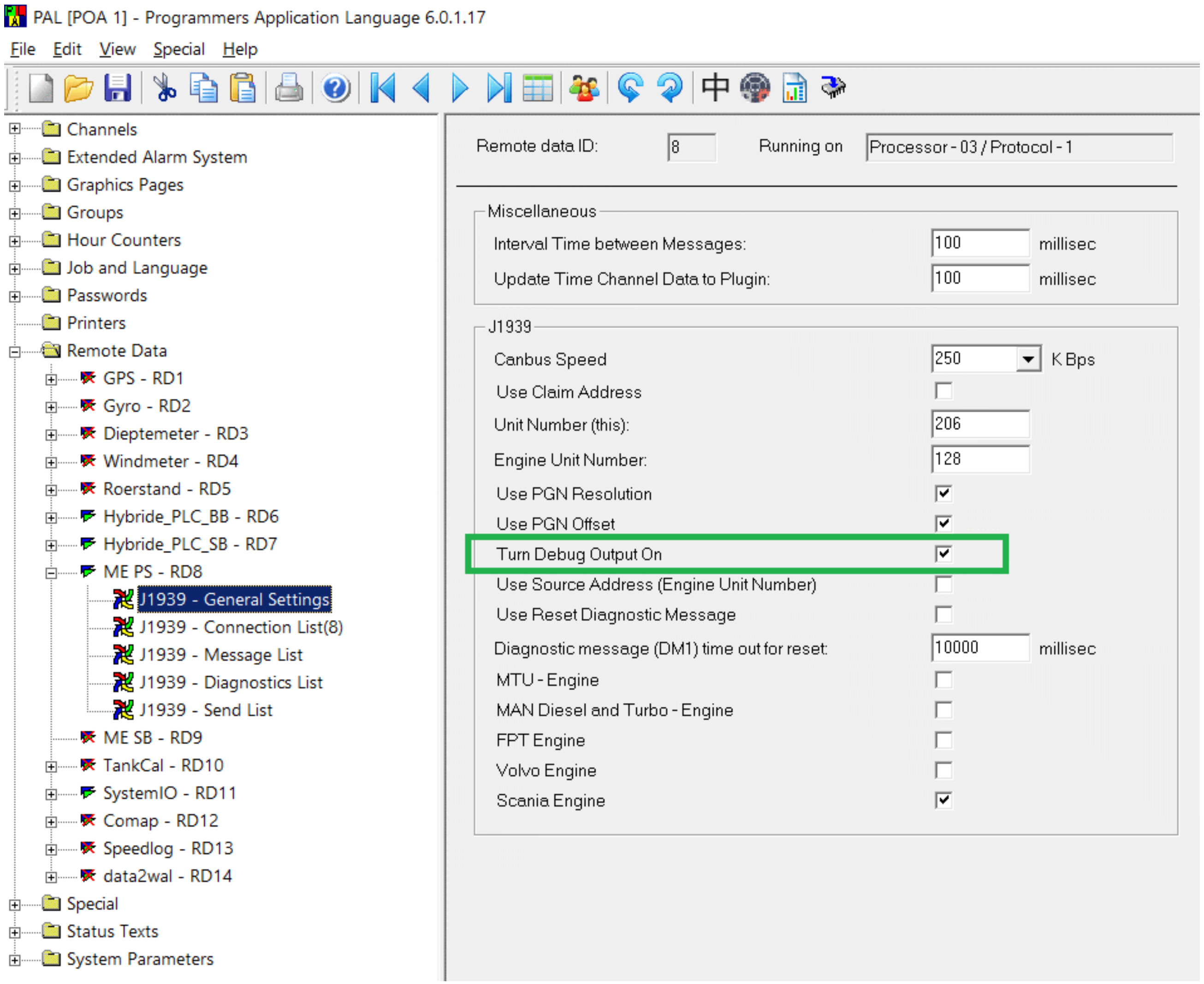
Remote Data

RD08 - ME PS

Select: None

☑ Monitor Comport to UDP (51502) ComPort: COM2

Set option **'Turn Debug Output on'** to on by checking this box (see: J1939 - General Settings)



PAL [POA 1] - Programmers Application Language 6.0.1.17

- Channels
- Extended Alarm System
- Graphics Pages
- Groups
- Hour Counters
- Job and Language
- Passwords
- Printers
- Remote Data
  - GPS - RD1
  - Gyro - RD2
  - Dieptemeter - RD3
  - Windmeter - RD4
  - Roerstand - RD5
  - Hybride_PLC_BB - RD6
  - Hybride_PLC_SB - RD7
  - ME PS - RD8
    - J1939 - General Settings
    - J1939 - Connection List(8)
    - J1939 - Message List
    - J1939 - Diagnostics List
    - J1939 - Send List
  - ME SB - RD9
  - TankCal - RD10
  - SystemIO - RD11
  - Comap - RD12
  - Speedlog - RD13
  - data2wal - RD14
- Special
- Status Texts
- System Parameters

Remote data ID: 8 Running on: Processor - 03 / Protocol - 1

Miscellaneous

Interval Time between Messages: 100 millisec

Update Time Channel Data to Plugin: 100 millisec

J1939

Canbus Speed: 250 K Bps

Use Claim Address □

Unit Number (this): 206

Engine Unit Number: 128

Use PGN Resolution ☑

Use PGN Offset ☑

Turn Debug Output On ☑

Use Source Address (Engine Unit Number) □

Use Reset Diagnostic Message □

Diagnostic message (DM1) time out for reset: 10000 millisec

MTU - Engine □

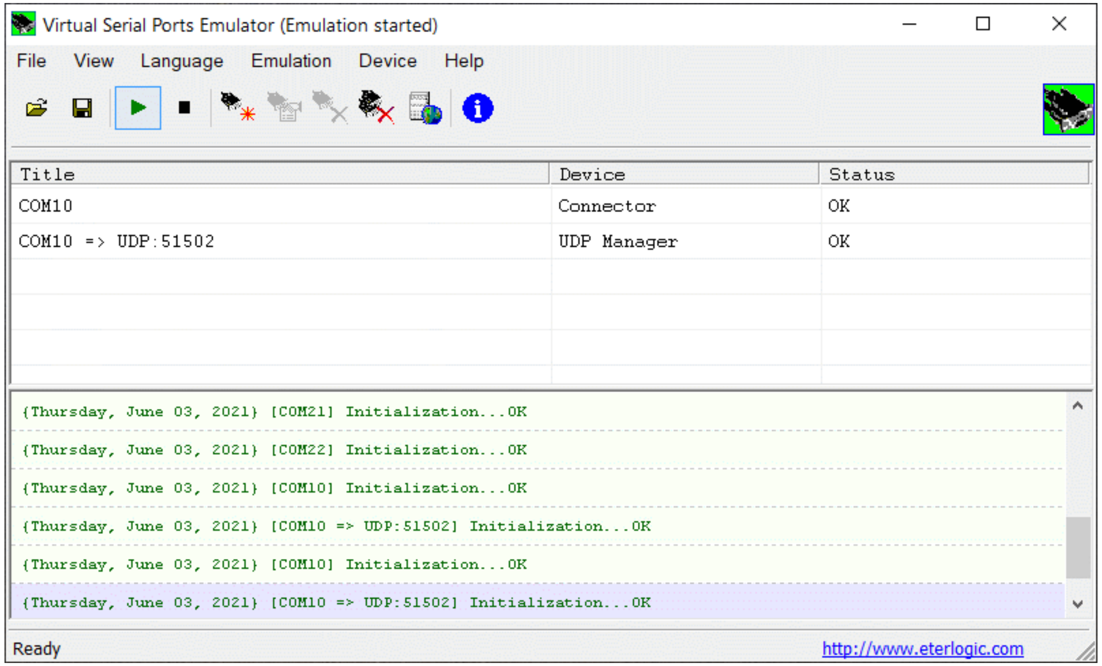
MAN Diesel and Turbo - Engine □

FPT Engine □

Volvo Engine □

Scania Engine ☑

Create a Virtual Comport on your PC or Laptop
There are many programs on internet to find to create a virtual comport.
For this example, we use the program "Virtual Serial Ports Emulator",
which can be obtained by internet http://www.eterlogic.com/



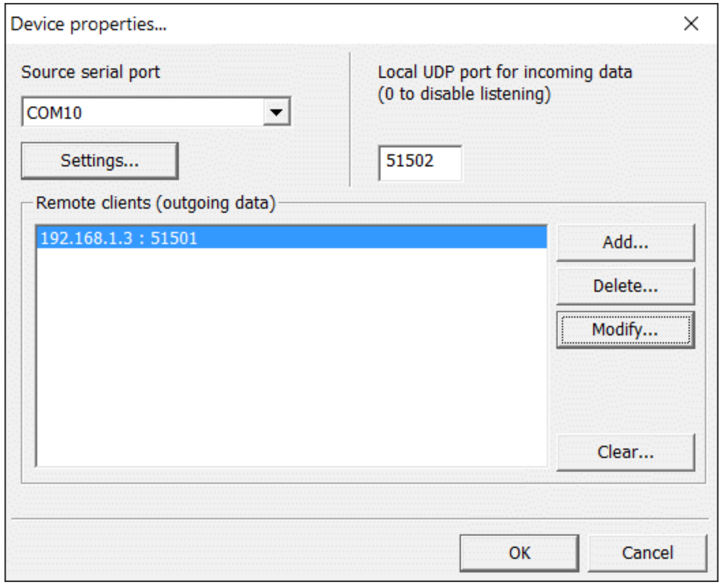
Title	Device	Status
COM10	Connector	OK
COM10 => UDP:51502	UDP Manager	OK

```
{Thursday, June 03, 2021} [COM21] Initialization...OK
{Thursday, June 03, 2021} [COM22] Initialization...OK
{Thursday, June 03, 2021} [COM10] Initialization...OK
{Thursday, June 03, 2021} [COM10 => UDP:51502] Initialization...OK
{Thursday, June 03, 2021} [COM10] Initialization...OK
{Thursday, June 03, 2021} [COM10 => UDP:51502] Initialization...OK
```

Ready

http://www.eterlogic.com

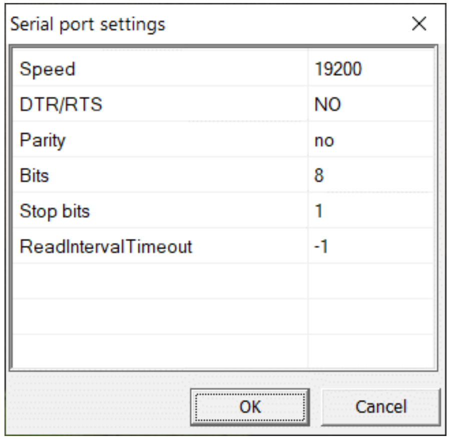
Example has virtual comport 10 and UDP connection to this comport.



Source serial port: COM10

Local UDP port for incoming data
(0 to disable listening): 51502

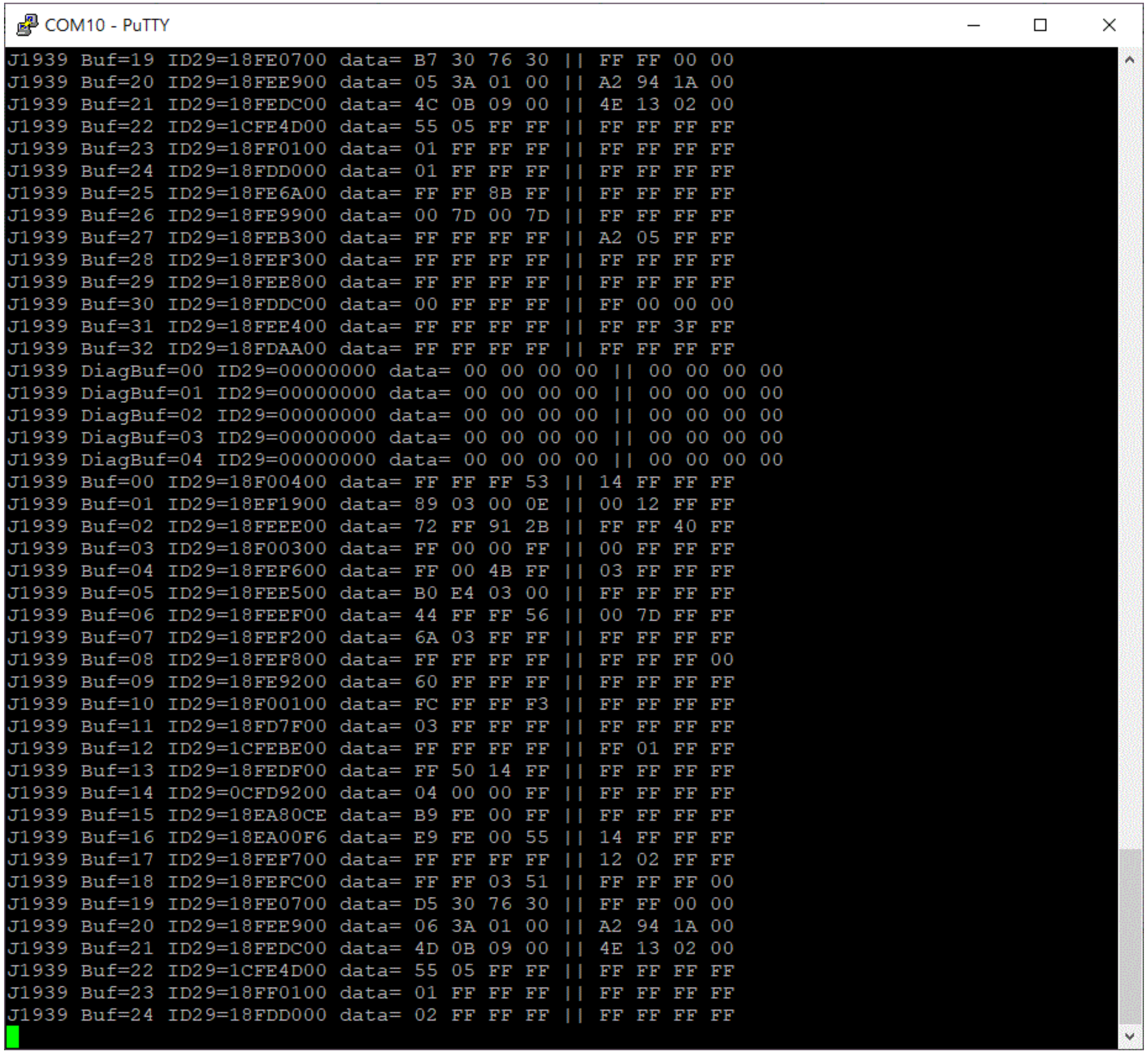
Remote clients (outgoing data):
192.168.1.3 : 51501


Serial port settings

Speed	19200
DTR/RTS	NO
Parity	no
Bits	8
Stop bits	1
ReadIntervalTimeout	-1

Example uses this comport settings (virtual comport)

By opening a putty client (serial port, 19200, 8, N, 1, flow control off) on PC
(settings must be same a virtual comport)
debug information can be retrieved for displaying it on a screen.
It will show text output which contains the canbus messages are received on Control Processor.
Debug output can be saved into .txt file.



```
J1939 Buf=19 ID29=18FE0700 data= B7 30 76 30 || FF FF 00 00
J1939 Buf=20 ID29=18FEE900 data= 05 3A 01 00 || A2 94 1A 00
J1939 Buf=21 ID29=18FEDC00 data= 4C 0B 09 00 || 4E 13 02 00
J1939 Buf=22 ID29=1CFE4D00 data= 55 05 FF FF || FF FF FF FF
J1939 Buf=23 ID29=18FF0100 data= 01 FF FF FF || FF FF FF FF
J1939 Buf=24 ID29=18FDD000 data= 01 FF FF FF || FF FF FF FF
J1939 Buf=25 ID29=18FE6A00 data= FF FF 8B FF || FF FF FF FF
J1939 Buf=26 ID29=18FE9900 data= 00 7D 00 7D || FF FF FF FF
J1939 Buf=27 ID29=18FEB300 data= FF FF FF FF || A2 05 FF FF
J1939 Buf=28 ID29=18FEF300 data= FF FF FF FF || FF FF FF FF
J1939 Buf=29 ID29=18FEE800 data= FF FF FF FF || FF FF FF FF
J1939 Buf=30 ID29=18FDDC00 data= 00 FF FF FF || FF 00 00 00
J1939 Buf=31 ID29=18FEE400 data= FF FF FF FF || FF FF 3F FF
J1939 Buf=32 ID29=18FDAA00 data= FF FF FF FF || FF FF FF FF
J1939 DiagBuf=00 ID29=00000000 data= 00 00 00 00 || 00 00 00 00
J1939 DiagBuf=01 ID29=00000000 data= 00 00 00 00 || 00 00 00 00
J1939 DiagBuf=02 ID29=00000000 data= 00 00 00 00 || 00 00 00 00
J1939 DiagBuf=03 ID29=00000000 data= 00 00 00 00 || 00 00 00 00
J1939 DiagBuf=04 ID29=00000000 data= 00 00 00 00 || 00 00 00 00
J1939 Buf=00 ID29=18F00400 data= FF FF FF 53 || 14 FF FF FF
J1939 Buf=01 ID29=18EF1900 data= 89 03 00 0E || 00 12 FF FF
J1939 Buf=02 ID29=18FEEE00 data= 72 FF 91 2B || FF FF 40 FF
J1939 Buf=03 ID29=18F00300 data= FF 00 00 FF || 00 FF FF FF
J1939 Buf=04 ID29=18FEF600 data= FF 00 4B FF || 03 FF FF FF
J1939 Buf=05 ID29=18FEE500 data= B0 E4 03 00 || FF FF FF FF
J1939 Buf=06 ID29=18FEEF00 data= 44 FF FF 56 || 00 7D FF FF
J1939 Buf=07 ID29=18FEF200 data= 6A 03 FF FF || FF FF FF FF
J1939 Buf=08 ID29=18FEF800 data= FF FF FF FF || FF FF FF 00
J1939 Buf=09 ID29=18FE9200 data= 60 FF FF FF || FF FF FF FF
J1939 Buf=10 ID29=18F00100 data= FC FF FF F3 || FF FF FF FF
J1939 Buf=11 ID29=18FD7F00 data= 03 FF FF FF || FF FF FF FF
J1939 Buf=12 ID29=1CFEBE00 data= FF FF FF FF || FF 01 FF FF
J1939 Buf=13 ID29=18FEDF00 data= FF 50 14 FF || FF FF FF FF
J1939 Buf=14 ID29=0CFD9200 data= 04 00 00 FF || FF FF FF FF
J1939 Buf=15 ID29=18EA80CE data= B9 FE 00 FF || FF FF FF FF
J1939 Buf=16 ID29=18EA00F6 data= E9 FE 00 55 || 14 FF FF FF
J1939 Buf=17 ID29=18FEF700 data= FF FF FF FF || 12 02 FF FF
J1939 Buf=18 ID29=18FEFC00 data= FF FF 03 51 || FF FF FF 00
J1939 Buf=19 ID29=18FE0700 data= D5 30 76 30 || FF FF 00 00
J1939 Buf=20 ID29=18FEE900 data= 06 3A 01 00 || A2 94 1A 00
J1939 Buf=21 ID29=18FEDC00 data= 4D 0B 09 00 || 4E 13 02 00
J1939 Buf=22 ID29=1CFE4D00 data= 55 05 FF FF || FF FF FF FF
J1939 Buf=23 ID29=18FF0100 data= 01 FF FF FF || FF FF FF FF
J1939 Buf=24 ID29=18FDD000 data= 02 FF FF FF || FF FF FF FF
```

See also:

J1939 - General Settings

Product Version Information

PRAXIS
AUTOMATION TECHNOLOGY

Zijldijk 24A
2352AB Leiderdorp
The Netherlands
Phone: +31 (0)71 5255 353
Fax: +31 (0)71 5224 947
E-mail: info@praxis-automation.nl
Home Page: http://www.praxis-automation.nl

PAL Help

product: MEGA-GUARD E-Series
versions: 6.0.1.17

It also possible to see version information inside Windows Explorer:

- go to Windows Explorer (Start button, Programs, Accessories, Windows Explorer)
- browse to an executable file like 'Pal.exe'
- select file by using left mouse button
- press right mouse button, a popup menu will be shown
- select properties
- go tab-page 'Version'
- click on 'Product Version' inside 'Item name' List Box
- Edit Box 'Value' shows current product version of executable file