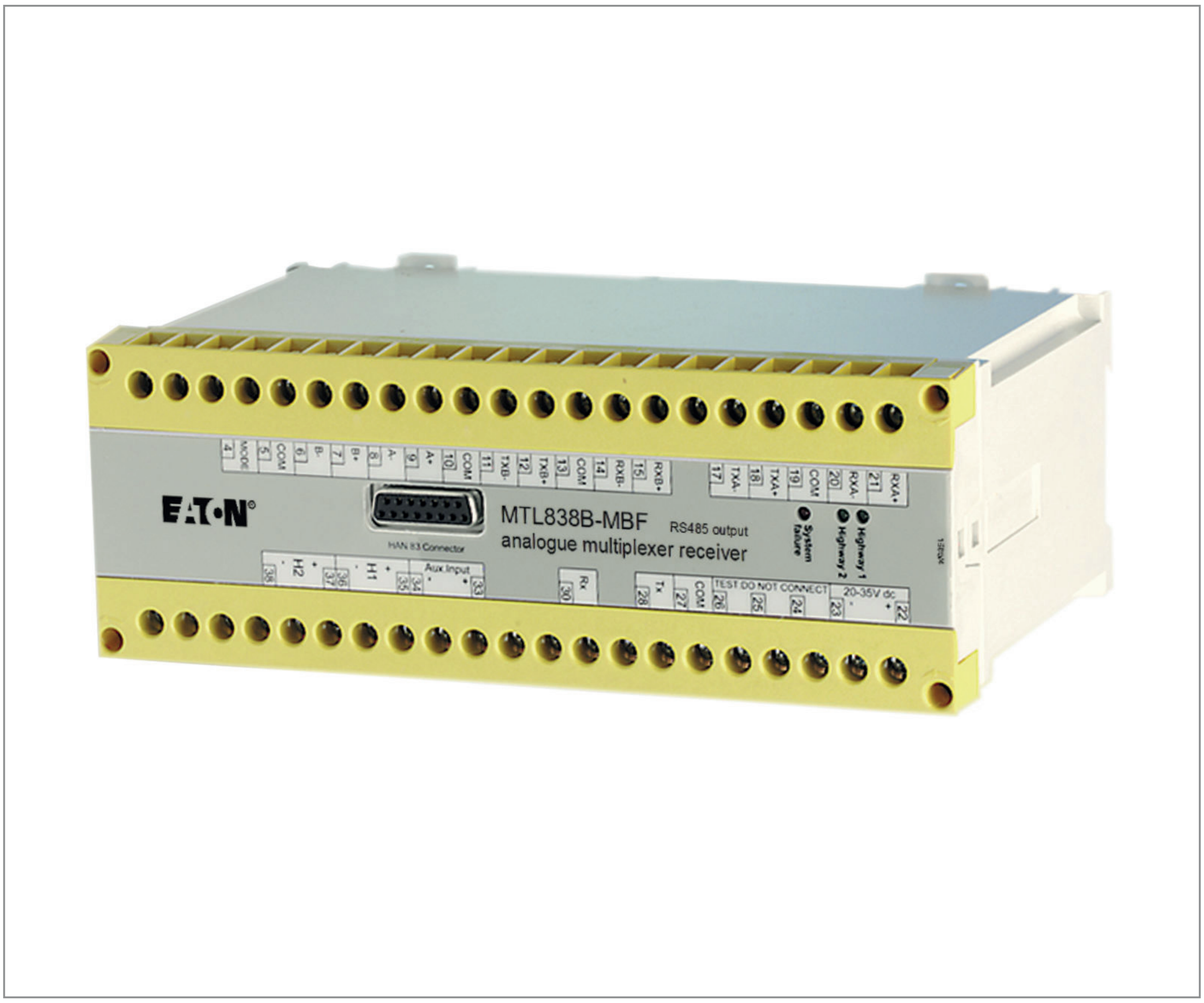
Technical data
MTL intrinsic safety solutions

February 2017
INM838B-MBF Rev 7

CROUSE-HINDS
SERIES

MTL838B-MBF

Modbus protocol manual



Powering Business Worldwide

Contents

Contents iii

INTRODUCTION 1

QUICKSTART GUIDE 3

BACKGROUND TO MODBUS 7

What is Modbus? 7
Modbus Transactions 7
The query-response cycle 9
  The query 9
  The response 9
The transmission modes 10
  ASCII mode 10
  RTU mode 10
  Comparison of RTU and ASCII modes 11
Modbus message framing 11
  ASCII message framing 11
  RTU message framing 12
The message fields 12
  The address field 12
  The function field 12
  The data field 12
  Byte count data 13
  The error check field 13
Data Encoding and Scaling 13
Modbus concepts and nomenclature 14
  The register concept 14
  Register and flag organisation and addressing 14
Modbus functions and exception responses 15
  Modbus functions 15
  Modbus exception responses 16
The Modbus physical layer 17
  The RS485 serial interface standard 17
  2- and 4-wire interconnection 18
  Point-to-point and multi-drop connection 18
  Data converters 19

MODBUS FUNCTIONS SUPPORTED BY THE MTL838B-MBF 20

READ COIL STATUS (function 01) 20
READ INPUT STATUS (function 02) 22
READ HOLDING REGISTERS (function 03) 23
READ INPUT REGISTERS (function 04) 24
FORCE SINGLE COIL (function 05) 25
PRESET SINGLE REGISTER (function 06) 26
READ EXCEPTION STATUS (function 07) 27
DIAGNOSTICS (function 08) 28
  RETURN QUERY DATA (subfunction 00 00) 28
  RETURN DIAGNOSTIC REGISTER (subfunction 00 02) 29
  CLEAR COUNTERS AND DIAGNOSTIC REGISTERS (subfunction 00 10) 30
  RETURN BUS MESSAGE COUNT (subfunction 00 11) 30
  RETURN BUS COMMUNICATION ERROR COUNT (subfunction 00 12) 31
  RETURN BUS EXCEPTION ERROR COUNT (subfunction 00 13) 32
PRESET MULTIPLE REGISTERS (function 16) 33

EXCEPTION RESPONSES SUPPORTED BY THE MTL838B-MBF 34

Construction of exception responses 34
ILLEGAL FUNCTION (exception code 01) 35
ILLEGAL DATA ADDRESS (exception code 02) 35
ILLEGAL DATA VALUE (exception code 03) 36
SLAVE DEVICE FAILURE (exception code 04) 36

NEGATIVE ACKNOWLEDGE (exception code 07)..36

BACKGROUND TO THE MTL838B-MBF 37

The analogue-input multiplexer system 37
Configuring the MTL838B-MBF 37
On-line 37
Off-line 38
Hardware configuration of the MTL838B-MBF..39
Switch 101 40
Switch 102 40
Switch 301 41
Switch 302 41
Interconnection of the MTL838B-MBF 42
Initialisation mode 43
Slave, Transmitter and Input addressing 44
Addressing MTL838B-MBF slaves 44
Addressing the transmitters of each MTL838B-MBF 44
Allocation of addresses to individual inputs...44

INPUT STATUS FLAGS AND REGISTERS 46

Mapping of input status flags and input registers...46
Mapping for IEEE data format 46
Mapping for non-IEEE data format 47
Revision number of HCA software...48
Revision number of IPP software...48
MTL838B-MBF status information 48
'Error flag'..49
'Configuration changing' 50
'Configuration rejected - due to internal MTL838B-MBF fault' 50
'Invalid database' 50
'Auxiliary input open' 50
'Highway 1 OK' and 'Highway 2 OK' 50
'Transmitter failed'..50
'Open circuit -', 'Low alarm -' and 'High alarm detected on any input'.......................51
Configuration checksum 51
High alarm status register 51
Low alarm status register..51
Open alarm status register..52
Scaled analogue input value 52
Cold junction temperature of MTL831Bs 52

COIL STATUS FLAGS 54

Mapping of coil status flags...54
Store the current checksum value 54
Set factory defaults for mV inputs...54
Confirm database correctly configured...55
Set factory defaults for mV inputs, leaving DATAFORMAT unchanged 55

HOLDING REGISTERS..56

Configuration checksum reference 57
Unused holding registers 57
Data format selection 57
Tag field 59
Number and type of transmitters...59
Units of temperature..59
Line frequency of power supply...60
Configuration test method 60
Input type and safety drive 60
Input zero with offset - for scaling output measurements 62
Gain - for scaling output measurements 63
High alarm level 63
Low alarm level 63
Output zero offset..63

MTL838B-MBF EXCEPTION RESPONSES..64

Following 'READ COIL STATUS' queries 64
Following 'READ INPUT STATUS' queries 64
Following 'READ HOLDING REGISTERS' query 64
Following 'READ INPUT REGISTERS' query 65
Following 'FORCE SINGLE COIL' queries 65
Following 'PRESET SINGLE REGISTER' queries 65
Following 'READ EXCEPTION STATUS' queries 65
Following 'DIAGNOSTICS' queries 65
Following 'PRESET MULTIPLE REGISTERS' queries 66
Following queries not supported by the MTL838B-MBF 66

SCALING 67

Background to scaling input data 67
Calculation of scaling parameters - in practice 68
Self calibration of input measurements 69
Sensor Input Processing 69
  Thermocouple inputs 69
  Resistance inputs 69
  RTD inputs 69
  mV inputs 70
Data timing 70

Configuring with PCS83 software 71

Computer system requirements 71
Installing software 71
  Installing to a Hard Disk 71
  Installing to a Floppy Disk 72
Before you start 72
Serial link cable connections 72
Troubleshooting 73
Receiver DIL switch settings 73
Using the program 74
Option 1 : Upload Data 75
Option 2 : Configure Rx 76
  Reset Rx 77
  Tag Name 77
  Data Format 77
  Reset Scales 77
  Number of Transmitters 77
  Temperature Units 77
  Line Rejection 77
  Checksum Mode 77
  Checksum 77
  Address and Communications Parameters 77
  Diagnostics 77
Option 3: Configure Tx 78
  Tx Number 79
  Tx Type 79
  Channel Number 79
  Channel Type 79
  Safety Drive 79
  High and Low Alarm 79
  Input Zero, Output Zero and Gain factor 80
  Calculate Scalings 80
Option 4: Copy 80
Option 5: Download 81
Option 6: Load 81
Option 7: Save 81
Option 8: Signoff 81
Option 9: Logged Drive 81
Option 10: Programming 82
Option 11: Serial Port 82
Option 12 : Input Monitor 82
Option 13: Tx. Monitor 82

Option 14: Hard Copy..83
Appendix A 85
ERROR CHECKING TECHNIQUES IN MODBUS 85
Calculation of the LRC in ASCII transmission mode 85
Calculation of the CRC in RTU transmission mode..85
Appendix B 86
IEEE single precision data format 86
Appendix C 87
Non-IEEE data format...87
Appendix D 89
Faultfinding on the MTL830 System 89
Single highway mode..89
Dual highway mode 89
Appendix E...92
Maintenance & Disposal 92
Index..93

INTRODUCTION

This protocol manual is principally intended for instrumentation engineers and technicians who need to configure the communications between Modbus system hosts and MTL838B-MBF multiplexer receivers. It also includes information for the end user on the use of tools that are available for configuring the MTL838B-MBF.

It provides comprehensive information on the Modbus* protocol, describes the communication between the MTL838B-MBF and the host, and provides detailed information relating to the functions of the MTL838B-MBF. No previous knowledge of Modbus is assumed.

The JBUS* protocol is also supported by the MTL838B-MBF. JBUS is virtually identical to Modbus apart from a slight difference in the addressing of slaves, and this manual may be used for both protocols. The difference in slave addressing is explained in the relevant section.

The manual is divided into chapters which can be summarised as follows:

QuickStart Guide

This describes the commissioning of a simple system with the most commonly used settings.

Background to Modbus

This is an introduction to Modbus and describes the design and maintenance of a communications link between an MTL838B-MBF and a Modbus master.

Modbus functions supported by the MTL838B-MBF

A detailed description of the Modbus functions recognised by the MTL838B-MBF. This is to enable users to select the most appropriate function for the Modbus master.

Exception responses supported by the MTL838B-MBF

This covers a range of diagnostics for the more advanced user.

Background to the MTL838B-MBF

Essential information for configuration and maintenance of MTL830 Multiplexer Systems including DIL switch settings.

Input Status Flags and Registers

Input status flag and register location required for configuration of Modbus master.

Coil Status Flags

Mainly for advance users considering configuration of the MTL838B-MBF from the Modbus master - a method that is not really recommended.

Holding Registers

Also for advance users considering configuration of the MTL838B-MBF from the Modbus master - a method that is not really recommended.

MTL838B-MBF Exception Responses

Interpretation of exception responses for advanced users.

Scaling

Points to consider for selecting scaling parameters within the MTL838B-MBF.

Configuring with the PCS83 software

Description of the recommended configuration method.

Modbus is a trademark of Schneider Automation Inc., North Andover, MA
JBUS is a trademark of April.

QUICKSTART GUIDE

This quickstart guide is written for an MTL830 system based on an MTL831B temperature input multiplexer transmitter with an MTL838B-MBF multiplexer receiver. For systems based on the MTL832 or MTL832EXE – 4-20mA multiplexer transmitters – this guide can still be used by also referring to INM883 for any differences between the MTL832 and the MTL831B.

Before actual installation, it is recommended that new users initially set up a simple system on the bench to become familiar with the MTL830 system. The minimum hardware required for a test system is as follows:

MTL831B	Multiplexer transmitter
MTL838B-MBF	Multiplexer receiver
MTL3052	Isolator (for hazardous area installations only).

In order to run a test the following equipment will be required:

A PC loaded with PCS83 software

Power supply 20 - 35V @ 500mA,

together with suitable cabling for the following requirements:

Data highway connections (see INS831B / INS838B)

Power supply connections

PCS83 serial link cable (see page 72 for connections).

The user will also need the following documentation for wiring information:

INS831B	MTL831B instruction sheet
INS838B	MTL838B-MBF instruction sheet.

1. Connect the MTL831B transmitter, MTL3052 isolators (hazardous area installations only) and MTL838B-MBF receivers as described in the instruction sheets INS831B and INS838B.
2. Configure the MTL831B transmitter addresses (see INS831B) and the terminal strip to normal or 3-wire RTD mode (see INS838B).
3. Determine the Modbus slave address and communication parameters to be used for the Modbus link. It is recommended that these are setup as 'hardware defined' not 'software defined', so that in the event of RAM corruption in the receiver the settings are not lost, and the unit continues to communicate with the master.

Remove both terminal strips from the MTL838B-MBF and set the DIL switches as required. The most commonly used settings are detailed in Figure 1 overleaf.

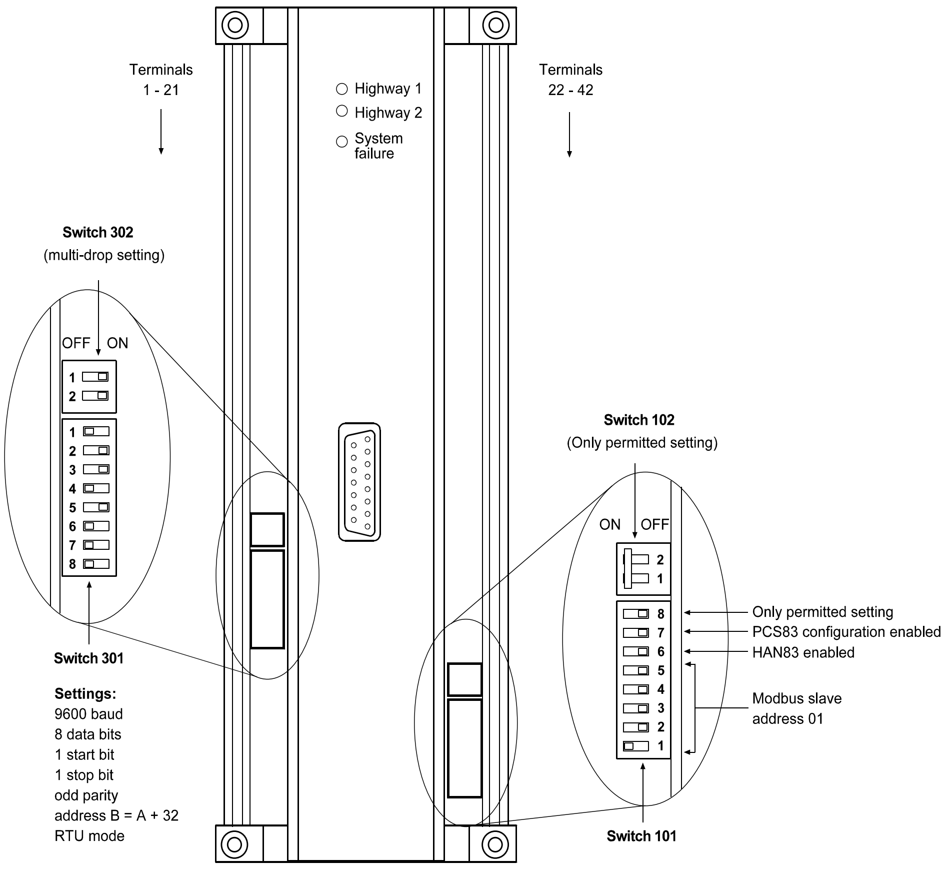


Figure 1 - QuickStart DIL Switch settings

4. Reconnect the terminal strips. Power up the MTL838B-MBF and check the status of the monitor LEDs as detailed in the table below.

Highway mode	* Highway LEDs		System failure LED
	H1	H2	
Single highway	ON	OFF	OFF
Dual highway	ON	ON	OFF

* This table is based upon using Highway 1. ON/OFF states are reversed when using Highway 2.

5. A software configuration file can be, and is often, created on a PC before establishing communications between the PC and the MTL838B-MBF. To upload data from the receiver and then amend it go to step 6, otherwise run the software as follows:

```
C:        <enter>
CD\MTL    <enter>
MTLMOD    <enter>
```

Select NO at the prompt to “Upload data?”. Select the Programming option and chose “Off-line”. Go to step 7 to continue the off-line setup.

6. Power-down the MTL830 system, connect the RS232 cable link to the PC that will run the PCS83 software, connect a wire link between terminal 4 (MODE) and terminal 5 (COM.) and then power-up the system again. The system will now start up in configuration mode. (See page 72 for cable link details).

On the PC, in DOS mode, run the software as follows:

```
C:        <enter>
CD\MTL    <enter>
MTLMOD    <enter>
```

Select YES at the prompt to “Upload data?”

If communications are established, “Uploading Configuration” at the bottom right of the screen is replaced by “Rx 0   Tx 1   Ch 1” which then counts up through the channels and transmitters connected.

If communications cannot be established an error message “RS232 Timeout” will be displayed. If this occurs, recheck the wiring and see the **PCS83 Software** section of this manual for further details.

For a completely new configuration it is recommended that the configuration is first set to a default condition by selecting “Reset Rx” - at which point the user must specify MTL831B or MTL832.

7. Select Configure Rx - see page 76.

   Select Data Format to be used by the Modbus host, number of transmitters and temperature units. Check other settings and change if required.

   Note: Address and communications parameters can only be changed if DIL switches are set to “software configurable”, and this is not recommended.

8. Select “Configure Tx.”.

   Select “Tx. No.”.

   Select “Tx. Type” - this is “831” for MTL831B

   Select “Channel No.”.

   Select “Input Type” and “Safety Drive” for this channel.

   Select “Calc. Scalings” - set required Signal Zero and Signal FSD, and set Output Zero and Output FSD values that will represent the signal inputs in the control system.

When <Esc.> is pressed the system will calculate the Gain factor which is then displayed on the Configure Tx. screen.

Note: If the Calc. Scalings is selected again for the same channel, the Signal FSD returns to the default of “100” and Output FSD displays the equivalent output.

Set High and Low Alarms in output units if required. Repeat this procedure for each channel.

Note: Use of the “Copy” function is recommended for similar inputs - the other parameters can then be edited.

Repeat the procedure for each transmitter.

9. If the setup has been carried out “On-line” to the MTL838B-MBF go to step 10. If a setup file has been created “Off-line” on the PC, use the ‘Save’ option to retain the settings; then connect the PC to the MTL838B-MBF as described in step 6, and use the ‘Download’ option to send the settings to the receiver.

10. Test operation of the multiplexer transmitter by selecting ‘Tx. Monitor’ - check that the correct outputs are displayed.

Note: If no inputs are connected to an MTL831B, then set an input type to “K-type Thermocouple” and connect a wire link between the “+” and “–” of the chosen input.

The scaling parameters should default to the following:

Signal Zero	0°C
Signal FSD	100°C
Output Zero	0°C
Output FSD	100°C

. . . but confirm these values. The channel will now display the cold junction temperature of the MTL831B.

11. When the configuration, or part of it, has been completed it can be saved to a PC file by selecting 'Save' - see page 81 for full detail. This may be useful if the configuration needs to be restored later.
12. Select 'Signoff' and follow screen instructions to remove the MODE/COM wire link. The MTL838B-MBF will exit congifuration mode and function as a Modbus slave.
13. Configure the Modbus master to read the required registers from the MTL838B-MBF.

Modbus Function 04 - READ INPUT REGISTERS - is usually used. This must specify the slave address, the starting register location and the number of registers to be read (max. 60).

	Input Register Location	
Parameter	**IEEE Data format**	**non- IEEE Data format**
Status & diagnostics	30005 - 30014	30005 - 30014
Tx. 1 inputs 1 - 16	30015 - 30046	30015 - 30030
Tx. 2 inputs 17 - 32	30047 - 30082	30031 - 30046

See page 24 for further details.

A typical configuration would be:

Slave address:	01
Function:	04
Starting register:	30015
Number of registers:	32

This command would read 32 inputs in non-IEEE data format, or 16 inputs in IEEE data format, from the MTL838B-MBF - Slave address 01 - and write the information into registers 30015 to 30046 of the Modbus master.

BACKGROUND TO MODBUS

What is Modbus?

Modbus is a communication protocol developed by AEG-Modicon, and was devised initially for use with their own Programmable Logic Controllers. It has, subsequently, become widely accepted as a communications standard, and many products have now been developed which use this protocol.

The protocol specification is maintained by the originators, AEG Modicon, independently of any professional body or industry association. Consequently, there is no formal process by which a product may be certified to be ‘Modbus compatible'. The onus is on the manufacturer of the product to confirm that their products are, and continue to remain, compatible with other Modbus devices.

The protocol defines a message structure and format, and determines how each slave will recognise messages sent to it, and how it should decode the information contained in the message. Standardising these elements allows Modbus devices from a number of different sources to be interconnected, without the need to write specialised software drivers for each component. This is a significant benefit to the end user and is one of the reasons that Modbus has been so successful and popular.

Modbus Transactions

Modbus controllers communicate using a master-slave technique, in which only one device (the master) can initiate a communication sequence.

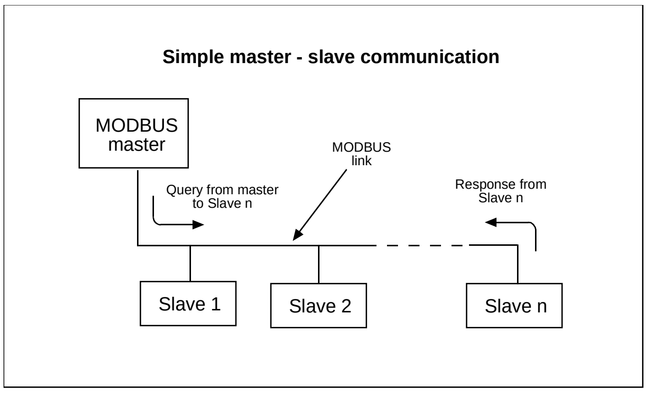


Figure 2 - Master - slave communication

The sequence begins with the master issuing a request or command on to the bus (a 'query'), which is received by the slaves.

The slaves respond by:

a) taking appropriate action,

b) supplying requested data to the master or

c) informing the master that the required action could not be carried out.

The master can address individual slaves or can transmit a message to be received by all slaves - through a 'broadcast' message (Figure 3).

When a slave receives a message addressed specifically to that slave, it will return a message to the master called a 'response'.

The response confirms:

- that the message was received, understood and acted upon, or
- it informs the master that the action required could not be carried out.

If the 'query' requests data from the slave, this will be returned as part of the response.

Messages 'broadcast' to all slaves do not require responses.

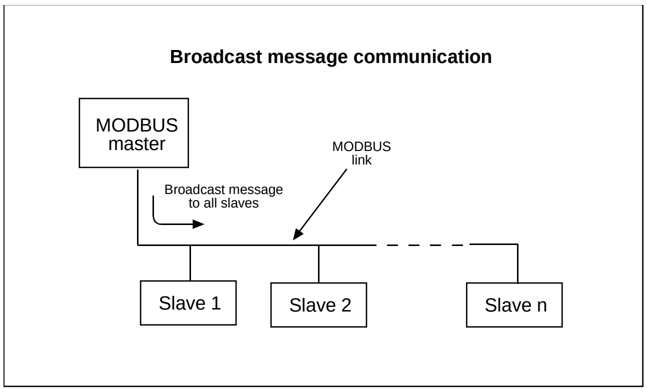

Broadcast message communication
MODBUS master
MODBUS link
Broadcast message to all slaves
Slave 1
Slave 2
Slave n

Figure 3 - Broadcast communication

Modbus slaves will only transmit on to the network when required to do so by the master. Slaves never transmit unsolicited messages.

If the slave cannot carry out the requested action, then it will respond with an error message. This error message, known as an **exception response**, indicates to the master the address of the responding slave, the action it was requested to carry out and an indication of why the action could not be completed.

If an error occurs in receipt of the message, the message will be ignored. This ensures that a slave does not take action that was really intended for another slave and does not carry out actions other than those that it is required to do. Should the message be ignored, the master will know that it's query has not been received correctly as it will not receive a response.

Modbus does not define the encoding of numerical data within the message. This is established by the manufacturer - see **Data Encoding and Scaling** on page 13.

Modbus ports frequently employ RS232C compatible serial interfaces, though RS422 and RS485 interfaces are also used. The type of interface used defines the connector pinouts, the cabling and the signal levels; these are not defined in Modbus. Similarly, transmission rates and parity checking are not defined in Modbus and will depend on the serial interface used and the options made available by the manufacturer of each Modbus component. Converters are available which allow components employing (say) an RS485 interface to be connected to a Modbus controller with a standard RS232 port. See **Data converters** on page 19 for more information.

Modbus will support up to 247 slaves from addresses 1 to 247 (JBUS 1 to 255) - address 0 is reserved for broadcast messages. In practice, the number of slave addresses that can be used is determined by the communications link that is chosen. For example, RS485 is limited to a total of 31 slaves.

The query-response cycle

The query-response cycle forms the basis of all communication on a Modbus network. In all situations it is the master that initiates the query and the slave that responds.

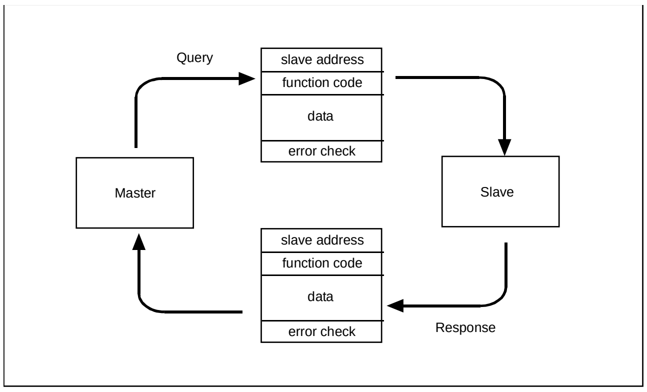


Figure 4 - The query / response cycle

The query

The query is made up of four parts: the device address; the function code; eight bit data bytes; and an error check.

The device address - uniquely identifies a particular slave or indicates that the message is a 'broadcast' addressed to all slaves.

The function code - tells the slave what type of action to perform.

The data - bytes contain any data that the slave will require to carry out the requested function (this may be a register address within the slave, a value to be used by the slave, etc.).

The error check - field allows the slave to confirm the integrity of the message received from the master. If an error is detected, the slave ignores the query and waits for the next query to be addressed to that slave.

The response

A slave will normally be required to provide a response (when a query has been addressed to that slave specifically, and not broadcast to all slaves), which will have the same overall structure format as was used for the query: a device address; a function code; eight bit data bytes; and an error check.

The device address - in the response is that of the addressed slave. This indicates to the master which slave is replying to it's query, and allows it to confirm that the correct slave is responding.

The function code - in the response is normally an exact copy of the function code in the query, and will only vary if the slave is unable to carry out the requested function. In such circumstances the function code returned is a modified form of the original code - this then indicates which function the slave was unable to perform.

The data - contain any data requested in the query.

The error check - allows the master to confirm the integrity of the message received from the slave - if the error check is not correct, the response is ignored.

The transmission modes

Two modes are available for transmitting serial data over a Modbus network, but they cannot be used together. The mode that is chosen must be adopted by all the Modbus components throughout the network.

The two transmission modes available are ASCII and RTU, and they differ in a number of ways. The way that the bit contents of the messages are defined, the way that information is packed in to the message fields, the way it will be decoded and the speed of operation at a given baud rate.

ASCII mode

When Modbus components are set up to communicate using ASCII (American Standard Code for Information Interchange) mode:

- 8-bit data is encoded in the message as two hex ASCII characters.
- The transmitted characters are printable.
- Each message begins and ends with specified characters.
- Time intervals between characters of up to 1 second are allowed, without causing an error.
- The error checking employed is a Longitudinal Redundancy Check (LRC

This transmission mode is the slower of the two techniques (as two characters must be transmitted for every 8-bits of data) and it has a less robust error check. The ability to print the transmitted characters and the ability to continue, despite delays in transmission, may only be a significant advantage in a limited number of applications.

The format for each byte in ASCII mode is:

Coding system:	Hexadecimal, ASCII characters: 0-9, A-F
	Two ASCII characters per 8-bit data
Bits per byte:	1 start bit, 7 data bits (least significant bit sent first),
	1 bit for even/odd parity (no bit for no parity) and 1 or 2 stop bits
Error check field:	Longitudinal redundancy check (LRC)

RTU mode

When Modbus components are set up to communicate in RTU (Remote Terminal Unit) mode:

- 8-bit data is encoded in the message as one 8-bit hex character.
- The transmitted messages are not printable.
- The start and end of each message is signalled by 'gaps' in transmission.
- Transmission of data within the message must be continuous.
- The error checking employed is a Cyclical Redundancy Check (CRC).

RTU mode is faster and more robust than ASCII.

The format for each byte in RTU mode is:

Coding system:	8-bit binary, comprised of two 4-bit words
	Each word equivalent to one hexadecimal character
Bits per byte:	1 start bit, 8 data bits, least significant bit sent first,
	1 bit for even/odd parity; no bit for no parity and 1 or 2 stop bits
Error check field:	Cyclical Redundancy Check (CRC)

Comparison of RTU and ASCII modes

The table below shows the encoding of the following message in the two modes:

	Message	**ASCII**	**RTU**
Start of frame	-	3A i.e. (:)	> 3 chars
Slave address	01	30 31	01
Function	04	30 34	04
Data	04 01	30 34 30 31	04 01
Error check	-	46 37	F1 C9
End of frame	-	0D 0A i.e. (CRLF)	-

The message above requests that the slave with address '01' reads a holding register (function '04') - and returns the values to the master. Note that although the data in ASCII mode is shown as hexadecimal characters, it would of course be transmitted as a binary stream. (Further, for simplicity, start, stop and parity bits have been omitted.)

A further difference between ASCII and RTU modes is the possibility of using either 7-bit or 8-bit data. As RTU mode requires the use of 8 bit characters, the 7-bit option cannot be used. ASCII only requires 7-bits for each character of the pair and can therefore use this option. It will work equally well with 8-bit communication, but the most significant bit will always be set to '0'.

The MTL838B-MBF protects against trying to set 7-bit data with RTU mode by always adopting 8-bit data with RTU, irrespective of the switch setting for the number of data bits.

Modbus message framing

Modbus messages must be structured (or 'framed') so that the different Modbus components can detect the start, content structure and end point of a message. It also allows any errors to be detected.

The framing used depends on the transmission mode chosen - ASCII or RTU.

ASCII message framing

In ASCII mode, messages start with a 'colon' (:), which in hex is '3A'. The message end is shown by 'carriage return/line feed' (CRLF) or 'OD OA' in hex .

The allowable characters for all other fields are hexadecimal 0-9, A-F. Networked components monitor the bus continuously for the 'colon' character and when one is received, they decode the next field (the address field) to find out if the address is for that slave. If the address is for another slave, then no action is taken, and the slave returns to monitoring for the 'colon' character. If the field following the colon is the address of the slave in question, then the slave continues to read the message and to act on it's contents.

Intervals of up to one second can elapse between characters within the message, but if an interval is greater than this, then the device assumes that an error has occurred. If the delay occurs in the 'query' to a slave, then the addressed slave will discard the message received up to that point and wait till the next message (marked by the colon start character) is received.

RTU message framing

In RTU mode, the message begins with a gap in transmission of at least 3.5 character periods. Network components monitor the bus continuously and when a 'silent' period of more than 3.5 character periods is detected, the first character following the transmission gap is translated to determine if it corresponds to the device's own address.
The end of the transmitted message is marked by a further interval of at least 3.5 character periods duration. An new message can only begin after this interval.

The entire message field must be transmitted as a continuous stream. If an interval of more than 1.5 character periods is detected during transmission of the message, then the message is assumed to be incomplete and the device returns to waiting for the next device address. The action taken on receipt of an incomplete message is as for receipt of an incorrect message, and it is ignored.

If a new message begins within 3.5 characters periods of the end of the previous frame, the device again ignores the message.

The message fields

The address field

Slave addresses may be in the range 1 to 247 with Modbus (1 to 255 with JBUS). A slave is addressed by the master placing the relevant address in the address field of the query message. When the slave sends its response, it places its own address in the message field to indicate to the master that the correct slave is replying.

Address '0' is used for 'broadcast' messages. All suitable slaves read them, but do not provide responses to such query messages.

The function field

Function codes may be in the range 1 - 255, though not all functions will be supported by all devices. When a message is sent from a master to a slave, the function code defines the action that is required from the addressed slave. Examples of action requested by the various function codes include: read input status; read register content; change a status within the slave; etc..

When the slave sends its response to the master, it will repeat the function code received, to indicate that the slave has understood the query and acted accordingly. If the query instruction could not be carried out by the slave, an 'exception response' is generated and the function code and data fields are used to inform the master of the reason for the exception.

The exception response is generated by returning the original function code from the master, but with its most significant bit set to '1'. Further information regarding the exception response is passed to the master via the data field of the response message. This tells the master what kind of error occurred and allows it to take the most appropriate action - either to repeat the original message, to try and diagnose what has happened to the slave, to set alarms or to take whatever action is most appropriate.

The data field

The data field transmits a number of hexadecimal values, each in the range 00 to FF. In ASCII transmission mode this is made up of a pair of characters, in RTU it is a single character.

A significant aspect of the communication between the master and it's slaves, that is not defined by Modbus, is the encoding of numerical data. Modbus allows the manufacturers of devices to determine which data encoding techniques are available to users of the device. The encoding of data is discussed on page 13.

The data field is used to provide the slave with any additional information needed to perform the function requested in the query. This would typically be a register address, a register range or a value. With some functions, the data field is not required and will not be included in the query.

If no errors occur, the data field of the response is used by the slave to pass data back to the master.

If an error occurs, the data field is used to pass more information to the master relating to the nature of the fault detected.

Byte count data

The responses to a number of queries require the slave to inform the master of the number of data bytes that are being returned in the response, and this requires a special implementation within the data field.

A typical example of this would be when the master has requested the slave to communicate the status of a range of registers. The slave responds by repeating the function code and it's own address, followed by the data field. The first byte of the data field identifies the number of bytes that are being returned that contain the register status information.

As was mentioned earlier, ASCII mode requires two 8-bit bytes to communicate a single register content, compared to RTU which only requires a single 8-bit byte. This difference is ignored when the byte count field is calculated, and the number of bytes indicated is identical to the number of bytes communicated in RTU mode, but is half the actual number of bytes communicated in ASCII mode.

The error check field

The error checking technique employed on the Modbus network depends on the transmission mode selected. With ASCII the technique used is based on an LRC (Longitudinal Redundancy Check) and with RTU a CRC (Cyclical Redundancy Check).

In both cases, the characters transmitted in the error check field are calculated by the transmitting device and included in the resulting transmission. The receiving device calculates what the error check field should contain, on receipt of the message, and compares it with the error check field in the received message. If these two values do not match exactly, then the receiving device knows that it has not received the message correctly, and disregards it.

In both modes, parity checking can be optionally selected.

A fuller explanation of the error checking techniques used by Modbus is given in Appendix A.

Data Encoding and Scaling

As has been mentioned earlier, an important area of the communication along the network, that is not defined by the Modbus protocol, is the encoding of numerical data. A related problem is the adoption of a scaling system for the data once it has been encoded. (Note: this is an area which requires careful consideration by users of the MTL838B-MBF.)

There is no problem here for manufacturers who are supplying complete systems, based on the Modbus network, as they can select a data encoding and scaling system appropriate to their needs. However, for manufacturers who are supplying products for general use, there is no possibility that they will be able to determine which data encoding system will be used by their customers, and they must allow the data encoding technique to be user selectable.

Three data encoding techniques are the most popular - IEEE, 16-bit unsigned and 16-bit offset.

A further area of difficulty associated with the encoding of data is the way in which the data is scaled - to provide a resolution of the measured value appropriate to the requirements of each application.

Users attempting to configure and scale analogue units via the Modbus master and network may find considerable difficulty with this issue. If specifically designed software is available for configuration and scaling of Modbus devices, this may be the simplest and most convenient method of scaling and encoding data. An example of this is the PCS83 software, which is available from Eaton's MTL product line for configuring the MTL838B-MBF. This makes encoding and scaling decisions transparent to the user.

The difficulties of implementing an encoding and scaling regime for the MTL838B-MBF via the Modbus host are discussed in depth on page 37. Users are strongly recommended to read this section before selecting the configuration method to be used with the MTL838B-MBF.

Modbus concepts and nomenclature

Modbus was originally written as a communication protocol for use with Modicon's own PLCs, and the nomenclature and concepts within Modbus reflect this early intention.

The register concept

The Modbus protocol is based on the concept that Modbus slaves hold their data in a series of defined status flags and registers, with defined addresses. A number of flags and registers are set aside for a number of purposes: single bit input/output; single bit output; multi-bit input/output; and multi-bit output.

Register and flag organisation and addressing

The registers and flags within Modbus devices are normally grouped as below:

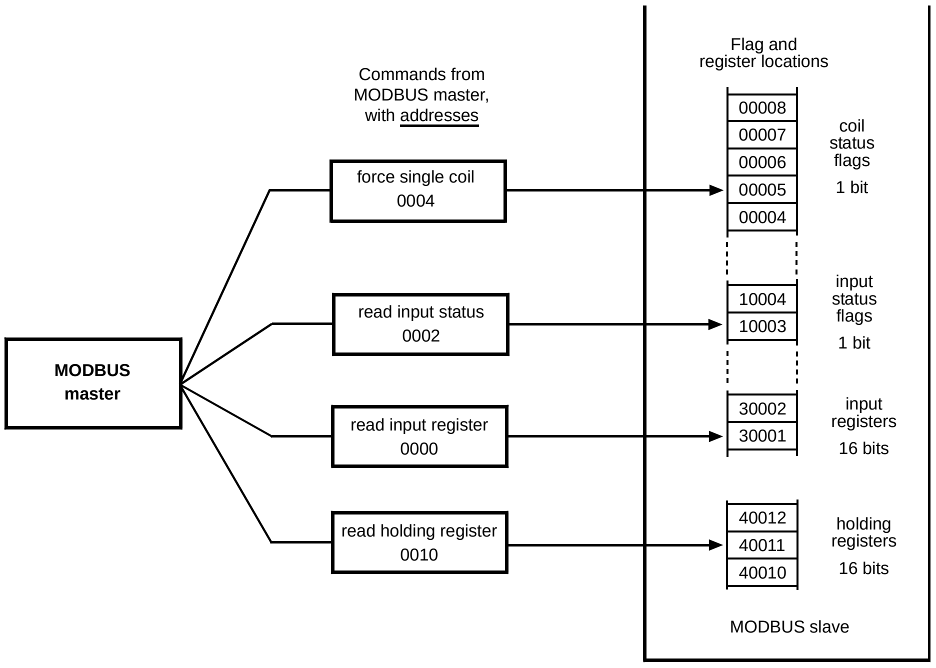


Figure 5 - Addressing Modbus registers

It is not necessary for manufacturers of Modbus devices to follow the register numbering conventions adopted by Modicon's PLCs, but it is normal to do so wherever possible.

COIL STATUS FLAGS

The single bit 'COIL STATUS' flags conventionally have an address of the format 0XXXX, and are used to store digital output information (outputs being *from* the slave *to* the field). These flags can be read from and written to by the Modbus master. Though the Modbus slave may not be capable of driving a digital output, these flags may still be used - for example, to request particular functions, such as to reset to original configuration.

Note: The term 'coil' comes from the original PLC application, in which the outputs from the PLC were set by controlling the coils of the output relays.

INPUT STATUS FLAGS

The single bit 'INPUT STATUS' flags conventionally have an address of the format 1XXXX. They are used to store the status of digital inputs to the Modbus slave and can only be read by the master. There is no facility, nor any need, for the master to write to these locations.

INPUT REGISTERS

The 16-bit 'INPUT REGISTERS', conventionally, have addresses of the format 3XXXX.
They are used to store data which has been collected by the Modbus slave. These registers can only be read by the Modbus master.

HOLDING REGISTERS

The 16-bit 'HOLDING REGISTERS' conventionally have an address of the format 4XXXX. They are used to hold general purpose data within the slave. The master can both read from and write to these registers. A typical application for these registers would be to hold configuration data.

The addressing of flag and register locations within a Modbus slave is simplified by the organisation outlined above. Because the Modbus master assumes that data will be stored in a suitable area, the most significant bit of the address is not sent - it is implicitly given by the function code. For example the instruction to 'FORCE COIL STATUS' would not be followed by an address of the format '1XXXX' as the '1' is presumed by way of the function itself.

IMPORTANT NOTE: A particular anomaly exists with the addressing employed by Modbus.

The locations of a slave's flags and registers begin from X0001, but the addresses given out by the Modbus master begin at X0000.

This means that to force the coil status in flag 10001, the master must send the “force the coil” instruction to address 0000 NOT 0001. To avoid confusion, throughout this manual a distinction is always made between the **'location'** and the **'address'** of flags and registers.

The addressing of the flags and registers of Modbus slaves may be better understood by reference to Figure 5, where the location number in the slave can be seen to be ‘one higher’ than the address issued by the master

When the master wishes to read from or write to a number of flags or registers, Modbus requires that this information is passed to the slave by specifying the address of the first register to be read from, or written to, followed by the number of subsequent registers that must be read. (i.e. To read the values in registers '02' to '04', the master would supply the address for register '02' and ask for the status of three registers.) It follows that if the master wishes to read or write to a number of registers, they must either be sequential - or the master must send more than one query.

Modbus functions and exception responses

Modbus functions

A total of 24 function codes are defined in revision 'E' of the Modbus specification. Modbus slaves will frequently not support all of the functions defined, and the MTL838B-MBF is typical of this. The following table summarises all of the functions that are available in revision 'E' and the ♦ indicates which are supported by the MTL838B-MBF.

CODE	DESCRIPTION

01 ♦	read coil status
02 ♦	read input status
03 ♦	read holding registers
04 ♦	read input registers
05 ♦	force single coil
06 ♦	preset single register
07 ♦	read exception status
08 ♦	diagnostics
09	program 484
10	poll 484
11	fetch comm. event ctr.
12	fetch comm. event log
13	program controller
14	poll controller
15	force multiple coils
16 ♦	preset multiple registers
17	report slave ID
18	program 884/M84
19	reset comm. link
20	read general reference
21	write general reference
22	mask write 4X register
23	read/write 4X registers
24	read FIFO queue

When a slave is unable to carry out a command that has been read correctly (i.e. the received message passes the error and parity checks), the slave will return a response which indicates the function it could not carry out and it will also transmit one of a number of 'exception responses' in the data field.

Modbus exception responses

The exception responses defined in revision 'E' are listed below. Those exception responses supported by the MTL838B-MBF are highlighted with a ♦ symbol.

CODE	DESCRIPTION
01	**ILLEGAL FUNCTION ♦** The function code received in the query is not an allowed action for the slave.
02	**ILLEGAL DATA ADDRESS ♦** The data address received in the query is not an allowable value for the slave.
03	**ILLEGAL DATA VALUE ♦** A value contained in the query data field is not an allowable value for the slave.
04	**SLAVE DEVICE FAILURE ♦** An unrecoverable error occurred while the slave was attempting to perform the requested action.

05	**ACKNOWLEDGE** The slave has received and accepted a request, but it will require a long period of time to complete the task. This prevents a time-out error occurring in the master.
06	**SLAVE DEVICE BUSY** The slave is engaged on a long-duration task. The master should re-transmit the request.
07	**NEGATIVE ACKNOWLEDGE ♦** The slave cannot perform the program function requested.
08	**MEMORY PARITY ERROR** The slave detected a parity error when reading extended memory.

The exception responses issued by a slave may be tailored by the component manufacturer to suit each slave, and events other than those defined in the table above can be used to trigger exception responses. The MTL838B-MBF is typical of this and uses the exception response '04' to indicate that it's configuration database has become corrupted.

The Modbus physical layer

The serial interface used by most Modbus masters is RS232C. This interface does not allow the Modbus network to extend beyond 10 to 20 metres in length, hence, many manufacturers of Modbus slaves have used other serial interfaces - with longer network capabilities.

We use the RS485 serial interface standard for its Modbus multiplexers. This includes tri-state operation, allows network lengths of up to 1000m and operates with data rates between 300 baud and 19.2 kbaud when used with MTL838B-MBF. RS485 also allows simple parallel connection of a number of units.

Note: when a non-RS232 interface is used with an RS232 master, a data converter must be inserted in to the network.

The RS485 serial interface standard

The RS485 standard defines the characteristics of the drivers and receivers that are connected to the bus. It does not define the cabling or connectors used, nor does it specify a particular data rate or signal format.

RS485 employs differential signalling and therefore requires at least two connectors per signal and a 'common' line connected between all devices.

RS485 specifies that the two differential signal lines should be marked 'A' and 'B', but this is not always followed. Eaton mark their multiplexers with '+' and '-', which describes the relative voltages of the signalling lines in their quiescent state. Note: with RS485, no damage will occur if the signalling lines are connected with the wrong polarity - but the system will not operate.

RS485 effectively limits the number of addresses supported to 32 units, which in Modbus corresponds to 31 slaves (with 1 master).

Terminations

RS485 interfaces should ideally be provided with a ‘matched’ termination to prevent reflections and 'ringing' of the signal on the bus cabling. The termination will normally be a simple resistive terminator, with an impedance that matches the characteristic of the bus - this will normally be close to 100Ω. In practice, with low data rates and relatively short networks, it is often unnecessary to terminate the bus.

Biasing

When no communication is taking place, the bus is in an undefined, floating state - so that noise on the bus may be decoded as real characters. Well written software should discard most of these characters, but the system may be further protected by *biasing* the bus to a known state and thereby prevent reception of ‘false’ characters.

Modbus multiplexers from Eaton do not feature any facilities for terminating or biasing the network, which must be provided by the user.

2- and 4-wire interconnection

When using RS485, the user can select either two-wire or four-wire interconnection. The two systems are shown diagramatically below.

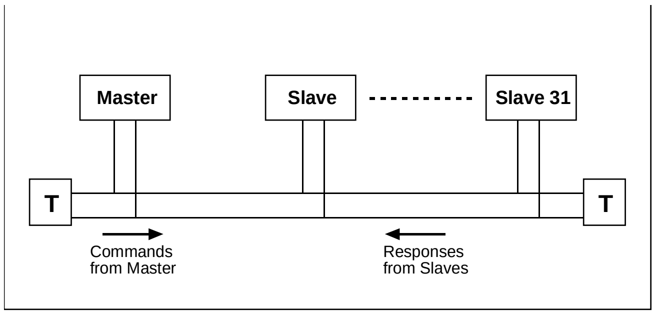


Figure 6 - 2 wire interconnection

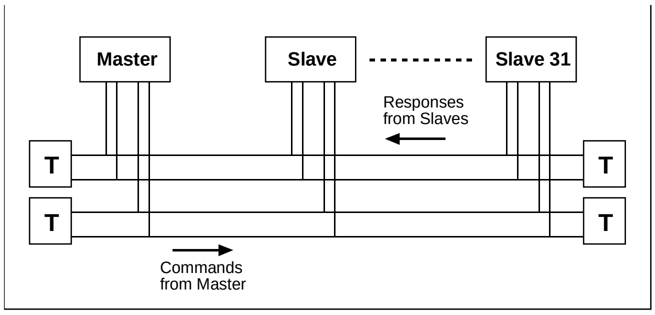


Figure 7 - 4 wire interconnection

In simple terms, two-wire uses the same pair of wires to transmit queries from the master and responses from the slave. With four-wire, queries are sent out on one pair of wires and responses are returned on another set. Note, the nomenclature 'two-' and 'four-wire' ignores the common wire, which is normally used to connect to all devices, but it is not generally shown on interconnection diagrams.

With some protocols, with the adoption of 'four-wire' interconnection, the system can be made to run more quickly, by allowing simultaneous communication of commands and responses. This is not possible with Modbus, as the 'Query-Response' cycle ensures that simultaneous communication by both the master and the addressed slave can never occur. When using Modbus, the decision to use 'two-' or 'four-wire' interconnection is generally made according to the type of serial data interface that is made available on the host. If an RS232 interface is used, then it is simplest to implement 'four-wire' interconnection, if the interface is RS422, then 'two-wire' interconnection is the simplest to implement.

Point-to-point and multi-drop connection

“Point-to-point” and “multi-drop” are methods of forming the network link between the master and its slave or slaves. Point-to-point describes the connection between a master and a single slave, whereas multi-drop allows a number of slaves to be connected to one master.

Linear BUS and daisy chain connection are multi-drop techniques for connecting a number of slaves to a single master. The two methods are shown below, where the difference between the two methods is apparent.

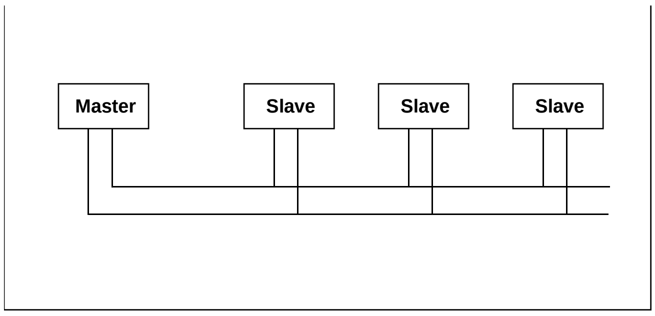


Figure 8 - Linear bus connection

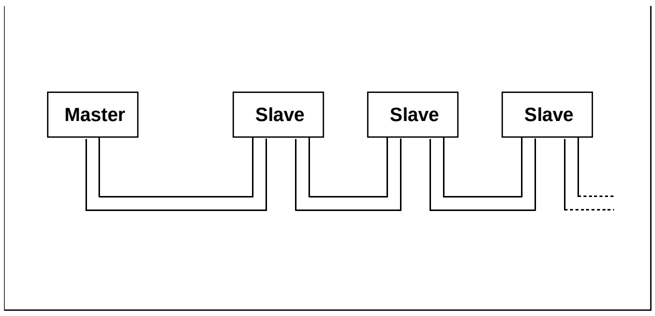


Figure 9 - Daisy-chain connection

Linear BUS connection has two advantages over daisy chain. Firstly, if any particular slave fails, then only that slave is lost, and secondly, slaves can be added to the system at any point along the network. With daisy chain connection, it is possible to lose communication with all slaves connected 'down stream' of any failed device, and it is necessary to gain access to at least one other slave in order to connect a new unit.

With some interfaces, linear BUS connection is not supported, and the only multidrop connection that is possible is daisy chain. One of the significant advantages of the MTL838B-MBF is that it allows multi-drop connections.

Note on RS422: the only direct connection method that is theoretically allowed when using a Modbus host with an RS422 serial interface is point-to-point. The specification for RS422 indicates that it would be necessary to employ an RS422 to RS485 converter to connect more than one Modbus slave. However, in practice, RS422 is found to perform well when RS485 slaves are linear BUS connected to the RS422 host.

The theory maintains that since RS422 has no tri-state facility it cannot be connected to more than one slave. However, since the Tx and Rx lines of the host are always ready to transmit to or receive from a slave, there is no need for the host to adopt a tri-state mode. This then allows linear BUS connection on an RS422 host with RS485 slaves.

Data converters

Most Modbus hosts will not provide an RS485 serial interface as standard, with RS232 and RS422 being much more common. Apart from the possibilities of using RS422 as outlined above, in many applications it will therefore be necessary to use a data converter to allow the connection of the Modbus RS485 interface to the host's RS232 or RS422 interface.

MODBUS FUNCTIONS SUPPORTED BY THE MTL838B-MBF

The following section exclusively describes the Modbus functions supported by the MTL838B-MBF:

FUNCTION NAME	FNCN. No.
READ COIL STATUS	01
READ INPUT STATUS	02
READ HOLDING REGISTERS	03
READ INPUT REGISTERS	04
FORCE SINGLE COIL	05
PRESET SINGLE REGISTER	06
READ EXCEPTION STATUS	07
DIAGNOSTICS	08
PRESET MULTIPLE REGISTERS	16

All other functions in the range 0 to 127 will not be acted upon or will be ignored. In some cases, when functions that are not supported, the MTL838B-MBF will respond with an appropriate exception response.

Important Note: This chapter contains a number of detailed tables that demonstrate the construction of messages passed along the Modbus network. However, most Modbus masters will have a user-interface that “shelters” the user from most of these details, and will only require the slave address, the function code, the initial coil or register location and the number of coils or registers to be read. The reader need not concern themselves with much of the detail presented here.

Some of the values are shown as hex some as binary. The hex values are given to describe the code or value that must be sent in the query and the response.

For ASCII mode, the communication is shown as hex coded ASCII, which demonstrates the additional level of character transmission required in this mode - actual communication is, of course, as a binary signal. The binary code for transmission in RTU mode is given directly, and it will be seen that this is a simple encoding of the equivalent hex value.

In the body of the text, decimal values are used, so as to be consistent with the numbering of the function codes in Revision 'E' of the Modbus specification. Where the encoding of these decimal values in to hex makes them appear differently in the table, the hex value is given in parenthesis - e.g. function code 10 ('OA' in hex). For simplicity, start, stop and parity bits are ignored throughout.

READ COIL STATUS (function 01)

The READ COIL STATUS function requests that the slave reads the status of a specified range of it's single bit input/output flags and returns these to the master. The range of flags to be read is given in the query, by the master indicating the address of the first flag to be read and then total number of subsequent flags - including the first.

The example in the following table shows the query required to read the status of flags 10001 to 10009 of slave number 20 (14 in hex.). The start address and number of flags to be read are always transmitted as two bytes - most significant bits (MSB) first, followed by the least significant bits (LSB):

FIELD NAME	HEX. VALUE	ASCII	RTU
slave address	14	31 34	0001 0100
function	01	30 31	0000 0001
starting address MSB	00	30 30	0000 0000
starting address LSB	00	30 30	0000 0000
no. of locations MSB	00	30 30	0000 0000
no. of locations LSB	09	30 39	0000 1001
error check	-	LRC	CRC

Note: due to the anomaly in the address and flag locations in Modbus, the address is always 1 less than the flag location. Thus flag '10001' is addressed by the hex. value '00 00'. See also the ‘Important Note’ on page 20.

The normal response to a READ COIL STATUS query contains the slave address, the repeated function code, the number of data bytes that are being transmitted in the response, the data bytes themselves and the error check.

The data bytes encode the status of the flags so that the status of the first flag to be read forms the LSB of the first data byte. Subsequent flag states form the next most significant bits of the first byte - thus if the master had requested the status of 8 flags, the data would be transmitted in a single data byte, with the LSB being the status of the first flag and the MSB being the status of the eighth. This is continued so that the status of the ninth flag requested forms the LSB of the second data byte.

If the master requests the status of a number of flags so that it is not possible to return 'complete' 8-bit data bytes (e.g. if the master requests the status of 9 flags, as above, which would require one complete 8-bit byte and a single bit), then the last data byte to be transmitted is 'packed' with '0's in it's MSBs.

The convention followed for the status is: 1 = ON; 0 = OFF.

A response to the query above would have the following format:

FIELD NAME	HEX. VALUE	ASCII	RTU
slave address	14	31 34	0001 0100
function	01	30 31	0000 0001
number of data bytes returned	02	30 32	0000 0010
first data byte	XX	XX XX	XXXX XXXX
second data byte	0X	30 XX	0000 000X
error check	-	LRC	CRC

Notes:

1. The seven most significant bits of the second data byte in RTU mode are zero, as the query only requested the status of 9 inputs. The seven zeros were packed in to the response to allow the slave to return complete 8-bit data bytes. The same packing of zeros takes place in ASCII mode, which will result in the ASCII characters returned being 30 XX.

2. The 'byte count' in the data field of the response shows the number of bytes returned in RTU mode, and half the number returned in ASCII. See page 13

3. The possible ranges for the elements of the query and response for the MTL838B-MBF are:

slave address	1 to 63
number of locations that may be read	1 to 512
number of data bytes returned	1 to 64

READ INPUT STATUS (function 02)

The READ INPUT STATUS function requests that the slave reads the status of a specified range of it's single bit output flags and returns these to the master. The range of inputs to be read is given in the query, by the master indicating the address of the first input to be read and then total number of subsequent flags - including the first.

The example below shows the query required to read the status of flags 10001 to 10030 of slave number 17 (11 in hex.). The start address and number of flags to be read are always transmitted as two bytes - most significant bits (MSB) first, followed by the least significant bits (LSB):

FIELD NAME	HEX. VALUE	ASCII	RTU
slave address	11	31 31	0001 0001
function	02	30 32	0000 0010
starting address MSB	00	30 30	0000 0000
starting address LSB	00	30 30	0000 0000
no. of locations MSB	00	30 30	0000 0000
no. of locations LSB	1E	31 45	0001 1110
error check	-	LRC	CRC

Note: due to the anomaly in the addresses and input status locations in Modbus, the address is always 1 less than the status location. Thus input '10001' is addressed by the hex. value '00 00'. See also the ‘Important Note’ on page 20.

The normal response to a READ INPUT STATUS comprises the slave address, the repeated function code, the number of data bytes that are being transmitted in the response, the data bytes themselves and the error check.

The data bytes encode the status of the inputs so that the status of the first input to be read forms the LSB of the first data byte. Subsequent input states form the next most significant bits of the first byte - thus if the master had requested the status of 8 inputs, the data would be transmitted in a single data byte, with the LSB being the status of the first input and the MSB being the status of the eighth. This is continued so that the status of the ninth input requested forms the LSB of the second data byte.

If the master requests the status of a number of inputs so that it is not possible to return 'complete' 8-bit data bytes (e.g. if the master requests the status of 9 inputs, which would require one complete 8-bit byte and a single bit), then the last data byte to be transmitted is 'packed' with '0's in it's MSBs.

The convention followed for the status is: 1 = ON; 0 = OFF.

A response to the query above would have the following format:

FIELD NAME	HEX. VALUE	ASCII	RTU
slave address	11	31 31	0001 0001
function	02	30 32	0000 0010
number of data bytes eturned	4	30 34	0000 0100
first data byte	XX	XX XX	XXXX XXXX
second data byte	XX	XX XX	XXXX XXXX
third data byte	XX	XX XX	XXXX XXXX
fourth data byte	XX	XX XX	00XX XXXX
error check	-	LRC	CRC

Notes:

1. The two most significant bits of the fourth data byte in RTU mode are zero, as the query only requested the status of 30 inputs. The two zeros were packed in to the response to allow the slave to return complete 8-bit data bytes. The same packing of zeros takes place in ASCII mode, but the value returned in ASCII cannot be determined without knowing the status of the last few data bits.

2. The 'byte count' in the data field of the response shows the number of bytes returned in RTU mode, and half the number returned in ASCII. See page 13

3. The possible ranges for the elements of the query and the response from the MTL838B-MBF are:

slave address	1 to 63
number of locations that may be read	1 to 512
number of data bytes returned	1 to 64

READ HOLDING REGISTERS (function 03)

The READ HOLDING REGISTERS function requests that the slave reads the binary contents of a specified range of its 16-bit holding registers and returns the values to the master. The range of registers to be read is given in the query, by the master indicating the address of the first register and the total number of subsequent registers to be read - including the first register.

The example below shows the query required to read the values held in holding registers 40108 to 40110 from slave 17 (108 and 17 are 6C and 11 in hex).

FIELD NAME	HEX. VALUE	ASCII	RTU
slave address	11	31 31	0001 0001
function	03	30 33	0000 0011
starting address MSB	00	30 30	0000 0000
starting address LSB	6B	36 4B	0101 1011
no. of registers MSB	00	30 30	0000 0000
no. of registers LSB	03	30 33	0000 0011
error check	-	LRC	CRC

Note: due to the anomaly in the addresses and register locations in Modbus, the address is always 1 less than the register location. Thus register '40108' is addressed as 0107 (hex. value '6B'). See also the ‘Important Note’ on page 20.

The normal response to a READ HOLDING REGISTERS query comprises the slave address, the repeated function code, the number of data bytes that are being transmitted in the response, the data bytes themselves and the error check.

The data bytes encode the contents of the holding registers as two bytes per register, with the binary contents right justified within each byte. For each register, the first byte contains the high order bits and the second byte the low order bits.

A response to the query above would have the following format:

FIELD NAME	HEX. VALUE	ASCII	RTU
slave address	11	31 31	0001 0001
function	03	30 33	0000 0011
number of data bytes returned	06	30 36	0000 0110
first data byte (MSB 40108)	XX	XX XX	XXXX XXXX
second data byte (LSB 40108)	XX	XX XX	XXXX XXXX
third data byte (MSB 40109)	XX	XX XX	XXXX XXXX
fourth data byte (LSB 40109)	XX	XX XX	XXXX XXXX
fifth data byte (MSB 40110)	XX	XX XX	XXXX XXXX
sixth data byte (LSB 40110)	XX	XX XX	XXXX XXXX
error check	-	LRC	CRC

Notes:

1. The 'byte count' in the data field of the response shows the number of bytes returned in RTU mode, and half the number returned in ASCII. See page 13

2. The possible ranges for the elements of the query and the response from the MTL838B-MBF are:

slave address	1 to 63
number of registers that may be read	1 to 60
number of data bytes returned (2 x the number of registers)	2 to 120

READ INPUT REGISTERS (function 04)

The READ INPUT REGISTERS function requests that the slave reads the binary contents of a specified range of its 16-bit input registers and returns the values to the master. The range of inputs to be read is given in the query, by the master indicating the address of the first register and the total number of subsequent registers to be read - including the first register.

The example below shows the query required to read the values held in input register 30009 from slave 31 (FF in hex).

FIELD NAME	HEX. VALUE	ASCII	RTU
slave address	FF	46 46	1111 1111
function	04	30 34	0000 0100
starting address MSB	00	30 30	0000 0000
starting address LSB	08	30 38	0000 1000
no. of points MSB	00	30 30	0000 0000
no. of points LSB	01	30 31	0000 0001
error check	-	LRC	CRC

Note: due to the anomaly in the addresses and register locations in Modbus, the address is always 1 less than the register location. Thus register '30009' is addressed by the hex. value '00 08'. See also the ‘Important Note’ on page 20.

The normal response to a READ INPUT REGISTERS query comprises the slave address, the repeated function code, the number of data bytes that are being transmitted in the response, the data bytes themselves and the error check.

The data bytes encode the contents of the input registers as two bytes per register. For each register, the first byte contains the high order bits and the second byte the low order bits.

A response to the query above would have the following format:

FIELD NAME	HEX. VALUE	ASCII	RTU
slave address	FF	46 46	1111 1111
function	04	30 34	0000 0100
number of data bytes returned	02	30 32	0000 0010
first data byte (MSB)	XX	XX XX	XXXX XXXX
second data byte (LSB)	XX	XX XX	XXXX XXXX
error check	-	LRC	CRC

Notes:

1. The 'byte count' in the data field of the response shows the number of bytes returned in RTU mode, and half the number returned in ASCII. See page 13

2. The possible ranges for the elements of the query and the response from the MTL838B-MBF are:

slave address	1 to 63
number of registers that may be read	1 to 60
number of data bytes returned (2 x the number of registers)	2 to 120

FORCE SINGLE COIL (function 05)

The FORCE SINGLE COIL function requests that the slave sets a specified input/output flag to a particular status. The address of the flag to be set is given in the query. The status to which the flag must be set is provided by two data bytes. If the flag is to be set to '1', then the data bytes sent are FF 00. If the flag is to be set to '0' the data bytes are 00 00.

The example below shows the query required to force the status of a flag with address 10065 of slave 18 to '1'. (65 and 18 are '41' and '12' in hex.)

FIELD NAME	HEX. VALUE	ASCII	RTU
slave address	12	31 32	0001 0010
function	05	30 35	0000 0101
flag address MSB	00	30 30	0000 0000
flag address LSB	40	34 30	0100 0000
force data MSB	FF	46 46	1111 1111
force data LSB	00	30 30	0000 0000
error check	-	LRC	CRC

The normal response to a FORCE SINGLE COIL query comprises the slave address, an echo of the function code, echoes of the flag address and status request, and an error check.

A response to the query above would have the following format:

FIELD NAME	HEX. VALUE	ASCII	RTU
slave address	12	31 32	0001 0010
function	05	30 35	0000 0101
flag address MSB	00	30 30	0000 0000
flag address LSB	40	34 30	0100 0000
force data MSB	FF	46 46	1111 1111
force data LSB	00	30 30	0000 0000
error check	-	LRC	CRC

The possible ranges for the elements of the query and the response from the MTL838B-MBF are:

slave address	1 to 63
coil address	0000 to 65535
number of data bytes returned	FF00 or 0000 - as query

PRESET SINGLE REGISTER (function 06)

The PRESET SINGLE REGISTER function requests that the slave writes specified data in to a particular register. The address of the register to be written to is given in the query. The data to be written is provided by two data bytes.

The example below shows the query required to 'pre-set' or write a register so that it holds the value 'FF FF'. The register location is 40003 of slave 1.

FIELD NAME	HEX. VALUE	ASCII	RTU
slave address	01	30 31	0000 0001
function	06	30 36	0000 0110
register address MSB	00	30 30	0000 0000
register address LSB	02	30 32	0000 0010
pre-set data MSB	FF	46 46	1111 1111
pre-set data LSB	FF	46 46	1111 1111
error check	-	LRC	CRC

The normal response to a PRESET SINGLE REGISTER query comprises the slave address, an echo of the function code, echoes of the register address and the pre-set data, and an error check.

A response to the query above would have the following format:

FIELD NAME	HEX. VALUE	ASCII	RTU
slave address	01	30 31	0000 0001
function	06	30 36	0000 0110
register address MSB	00	30 30	0000 0000
register address LSB	02	30 32	0000 0010
pre-set data MSB	FF	46 46	1111 1111
pre-set data LSB	FF	46 46	1111 1111
error check	-	LRC	CRC

The possible ranges for the elements of the query and the response from the MTL838B-MBF are:

slave address	1 to 63
register address	0 to 65535
data returned	0 to 65535 - as query

READ EXCEPTION STATUS (function 07)

The READ EXCEPTION STATUS function requests that the slave reads the contents of eight status bits within the slave and returns their values to the master. The registers that are used to store these exception status bits are pre-defined, so that the command itself is sufficient to locate the required locations.

In the MTL838B-MBF, the eight bits that are read correspond to the least significant bits of the STATUS input register 30005.

The example below shows the query required to read the exception status values for slave 10 (0A in hex.).

FIELD NAME	HEX. VALUE	ASCII	RTU
slave address	0A	30 41	0000 1010
function	07	30 37	0000 0111
error check	-	LRC	CRC

The normal response to a READ EXCEPTION STATUS query comprises the slave address, the repeated function code, a single data byte and the error check.

The data byte encodes the contents of the exception status bits in binary format, with the status of the lowest bit as the LSB of the byte.

A response to the query above would have the following format:

FIELD NAME	HEX. VALUE	ASCII	RTU
slave address	0A	30 41	0000 1010
function	07	30 37	0000 0111
exception status data	XX	XX XX	XXXX XXXX
error check	-	LRC	CRC

DIAGNOSTICS (function 08)

The DIAGNOSTICS function has a number of tests to check the communication link between the master and the slave. The subfunction code is transmitted as the first two bytes of data following an '08' function code in the query.

Some of the tests specified by the subfunctions require the slave to return data in the response to the query, others only require the slave to acknowledge receipt of the response in the normal way. Responses to diagnostic functions will return a repetition of the subfunction code as well as the '08' function.

Most of the diagnostic subfunctions supported by the MTL838B-MBF are defined so that the query must include two data bytes packed with zeros, immediately following the subfunction code.

A large number of are specified in revision 'E' of Modbus, not all of which are supported by the MTL838B-MBF. The ones supported are listed below:

CODE	DIAGNOSTIC SUBFUNCTION
00 00	return query data
00 02	return diagnostic register
00 10	clear counters and diagnostic registers
00 11	return bus message count
00 12	return bus comms. error count
00 13	return bus exception error count

The application of each of these subfunctions is discussed in detail in the following sections.

RETURN QUERY DATA (subfunction 00 00)

The diagnostic subfunction RETURN QUERY DATA requests the addressed slave to return (loop back) an exact copy of the data contained in the query, to the master, via the response.

An example of a query and response with this subfunction is given below. The master requests slave number 4 to return the hexadecimal data 'AA BB'.

The query:

FIELD NAME	HEX. VALUE	ASCII	RTU
slave address	04	30 34	0000 0100
function	08	30 38	0000 1000
subfunction MSB	00	30 30	0000 0000
subfunction LSB	00	30 30	0000 0000
data MSB	AA	41 41	1010 1010
data LSB	BB	42 42	1011 1011
error check	-	LRC	CRC

The response:

FIELD NAME	HEX. VALUE	ASCII	RTU
slave address	04	30 34	0000 0100
function	08	30 38	0000 1000
subfunction MSB	00	30 30	0000 0000
subfunction LSB	00	30 30	0000 0000
data MSB	AA	41 41	1010 1010
data LSB	BB	42 42	1011 1011
error check	-	LRC	CRC

RETURN DIAGNOSTIC REGISTER (subfunction 00 02)

The diagnostic subfunction RETURN DIAGNOSTIC REGISTER requests that the slave reads the contents of the diagnostic register and returns the binary data values to the master.

The query sends two zero data bytes, following the data bytes containing the subfunction code. The response returns two 8-bit data bytes containing the register data.

The contents of the diagnostic register may be defined by the manufacturer, according to the needs of each Modbus slave. For the MTL838B-MBF, the response to this query returns the content one of the registers that contain the STATUS data. The register returned is number 30005.

The following example shows the query and response generated when the master requests diagnostic subfunction 00 02 from the slave with address 12 ('0C' in hex.).

The query:

FIELD NAME	HEX. VALUE	ASCII	RTU
slave address	0C	30 43	0000 1100
function	08	30 38	0000 1000
subfunction MSB	00	30 30	0000 0000
subfunction LSB	02	30 32	0000 0010
data MSB	00	30 30	0000 0000
data LSB	00	30 30	0000 0000
error check	-	LRC	CRC

The response:

FIELD NAME	HEX. VALUE	ASCII	RTU
slave address	0C	30 43	0000 1100
function	08	30 38	0000 1000
subfunction MSB	00	30 30	0000 0000
subfunction LSB	02	30 32	0000 0010
data MSB	XX	XX XX	XXXX XXXX
data LSB	XX	XX XX	XXXX XXXX
error check	-	LRC	CRC

CLEAR COUNTERS AND DIAGNOSTIC REGISTERS (subfunction 00 10)

The diagnostic subfunction CLEAR COUNTERS AND DIAGNOSTIC REGISTERS requests that the slave clears a number of registers of their current values. In some slaves the function is as expected, and both counter and diagnostic registers are cleared. In some slaves (and the MTL838B-MBF is one of these) this subfunction only clears the counter registers and leaves the diagnostic registers untouched.

The query sends two zero data bytes, following the data bytes containing the subfunction code, and this is echoed in the response.

The following example shows the query and response generated when the master requests diagnostic subfunction 00 10 ('00 0A' in hex.) from the slave with address 02.

The query:

FIELD NAME	HEX. VALUE	ASCII	RTU
slave address	02	30 32	0000 0010
function	08	30 38	0000 1000
subfunction MSB	00	30 30	0000 0000
subfunction LSB	0A	30 41	0000 1010
data MSB	00	30 30	0000 0000
data LSB	00	30 30	0000 0000
error check	-	LRC	CRC

The response:

FIELD NAME	HEX. VALUE	ASCII	RTU
slave address	02	30 32	0000 0010
function	08	30 38	0000 1000
subfunction MSB	00	30 30	0000 0000
subfunction LSB	0A	30 41	0000 1010
data MSB	00	30 30	0000 0000
data LSB	00	30 30	0000 0000
error check	-	LRC	CRC

RETURN BUS MESSAGE COUNT (subfunction 00 11)

The diagnostic subfunction RETURN BUS MESSAGE COUNT requests that the slave returns to the master the contents of a register that is used to count the number of messages that the slave has detected on the system since its last restart, its last clear counters instruction or since power-up - whichever was the most recent.

The query sends two zero data bytes, following the data bytes containing the subfunction code. The response returns two 8-bit data bytes containing the register data.

The following example shows the query and response generated when the master requests diagnostic subfunction 00 11 ('00 0B' in hex.) from the slave with address 13 ('0D' in hex.).

The query:

FIELD NAME	HEX. VALUE	ASCII	RTU
slave address	0D	30 44	0000 1101
function	08	30 38	0000 1000
subfunction MSB	00	30 30	0000 0000
subfunction LSB	0B	30 42	0000 1011
data MSB	00	30 30	0000 0000
data LSB	00	30 30	0000 0000
error check	-	LRC	CRC

The response:

FIELD NAME	HEX. VALUE	ASCII	RTU
slave address	0D	30 44	0000 1101
function	08	30 38	0000 1000
subfunction MSB	00	30 30	0000 0000
subfunction LSB	0B	30 42	0000 1011
message count MSB	XX	XX XX	XXXX XXXX
message count LSB	XX	XX XX	XXXX XXXX
error check	-	LRC	CRC

RETURN BUS COMMUNICATION ERROR COUNT (subfunction 00 12)

The diagnostic subfunction RETURN BUS COMMUNICATION ERROR COUNT requests that the slave returns to the master the contents of a register that is used to count the number of CRC (or LRC) errors that the slave has detected on the system since its last restart, its last clear counters instruction or since power-up - whichever was the most recent.

The query sends two zero data bytes, following the data bytes containing the subfunction code. The response returns two 8-bit data bytes containing the register data.

The following example shows the query and response generated when the master requests diagnostic subfunction 00 12 ('00 0C' in hex.) from the slave with address 08.

The query:

FIELD NAME	HEX. VALUE	ASCII	RTU
slave address	08	30 38	0000 1000
function	08	30 38	0000 1000
subfunction MSB	00	30 30	0000 0000
subfunction LSB	0C	30 43	0000 1100
data MSB	00	30 30	0000 0000
data LSB	00	30 30	0000 0000
error check	-	LRC	CRC

The response:

FIELD NAME	HEX. VALUE	ASCII	RTU
slave address	08	30 38	0000 1000
function	08	30 38	0000 1000
subfunction MSB	00	30 30	0000 0000
subfunction LSB	0C	30 43	0000 1100
error count MSB	XX	XX XX	XXXX XXXX
error count LSB	XX	XX XX	XXXX XXXX
error check	-	LRC	CRC

RETURN BUS EXCEPTION ERROR COUNT (subfunction 00 13)

The diagnostic subfunction RETURN BUS EXCEPTION ERROR COUNT requests that the slave returns to the master the contents of a register that is used to count the number of exception errors (i.e. the number of times the slave has issued exception responses) that the slave has returned since its last restart, its last clear counters instruction or since power-up - whichever was the most recent.

The query sends two zero data bytes, following the data bytes containing the subfunction code. The response returns two 8-bit data bytes containing the register data.

The following example shows the query and response generated when the master requests diagnostic subfunction 00 13 ('00 0D' in hex.) from the slave with address 06.

The query:

FIELD NAME	HEX. VALUE	ASCII	RTU
slave address	06	30 36	0000 0110
function	08	30 38	0000 1000
subfunction MSB	00	30 30	0000 0000
subfunction LSB	0D	30 44	0000 1101
data MSB	00	30 30	0000 0000
data LSB	00	30 30	0000 0000
error check	-	LRC	CRC

The response:

FIELD NAME	HEX. VALUE	ASCII	RTU
slave address	06	30 36	0000 0110
function	08	30 38	0000 1000
subfunction MSB	00	30 30	0000 0000
subfunction LSB	0D	30 44	0000 1101
error count MSB	XX	XX XX	XXXX XXXX
error count LSB	XX	XX XX	XXXX XXXX
error check	-	LRC	CRC

PRESET MULTIPLE REGISTERS (function 16)

The PRESET MULTIPLE REGISTERS function requests that the slave writes specified data in to a range of registers. The range of registers to be written is identified by the master which indicates the location of the first register and then the total number of registers to be written - including the first.

The example below shows the query required to 'pre-set' or write two registers so that they both contain the value 'FF FF'. The first register location is 40003 of slave 1. Function 16 is '10' in hex.

FIELD NAME	HEX. VALUE	ASCII	RTU
slave address	01	30 31	0000 0001
function	10	31 30	0001 0000
register address MSB	00	30 30	0000 0000
register address LSB	02	30 32	0000 0010
register number MSB	00	30 30	0000 0000
register number LSB	02	30 32	0000 0010
1st pre-set data MSB	FF	46 46	1111 1111
1st pre-set data LSB	FF	46 46	1111 1111
2nd pre-set data MSB	FF	46 46	1111 1111
2nd pre-set data LSB	FF	46 46	1111 1111
error check	-	LRC	CRC

The normal response to a PRESET MULTIPLE REGISTERS query comprises the slave address, an echo of the function code, echoes of the register address and the pre-set data, and an error check.

A response to the query above would have the following format:

FIELD NAME	HEX. VALUE	ASCII	RTU
slave address	01	30 31	0000 0001
function	10	31 30	0001 0000
register address MSB	00	30 30	0000 0000
register address LSB	02	30 32	0000 0010
register number MSB	00	30 30	0000 0000
register number LSB	02	30 32	0000 0010
error check	-	LRC	CRC

The possible ranges for the elements of the response from an MTL838B-MBF are:

slave address	1 to 63
number of registers	1 to 60
data returned	0 to 65535 - as query

EXCEPTION RESPONSES SUPPORTED BY THE MTL838B-MBF

An MTL838B-MBF slave will issue one of five available exception responses if a message is received correctly (i.e. it passes the error checking), but the slave then finds it is unable to perform the required operation. The following section describes the construction of exception responses in general, and describes in detail those exception responses that are supported by the MTL838B-MBF.

The following exception responses are supported by the MTL838B-MBF:

ILLEGAL FUNCTION	response 01
ILLEGAL DATA ADDRESS	response 02
ILLEGAL DATA VALUE	response 03
SLAVE DEVICE FAILURE	response 04
NEGATIVE ACKNOWLEDGE	response 07

Note that if a slave receives a message which does not pass the error checking employed, it will discard the message and will not issue a response. This prevents a slave from carrying out operations that have either not been translated correctly or which were intended for another slave. The master employs a 'time-out' check, and if it has not received a response after a given time period, it will re-try or take other appropriate action.

Construction of exception responses

In an normal response, the slave exactly echoes the function code received from the master. In an exception response the slave returns to the master an echo of the function code received, but with it's MSB set to '1'. The master can therefore identify that an exception response is being returned, and identify the function code that was received by the slave. This is possible as there are less than 128 (or 80 hex.) function codes defined which, as binary 8-bit numbers, must always have a '0' as their MSB.

The example below shows the first few bytes of an exception response issued after slave 9 correctly received a function code '01' that it was then unable to perform.

FIELD NAME	HEX. VALUE	ASCII	RTU
slave address	09	30 39	0000 1001
function (as an exception for 01)	81	38 31	1000 0001

The exception response to function code '01' is perhaps most easily understood when examining the result in RTU mode, where it is clear that the MSB has been set to '1'.

Further data is passed to the master via the first bytes of the response's data field. The bytes returned are referred to as the 'exception code' and these are used to provide the master with additional information regarding the nature of the exception.

The exception code that is generated by the slave for any particular event, can be determined by the manufacturer of the device. It is normal, however, to try and use the exception codes so that the code name is as near as possible, in meaning, to the event that has caused the exception.

The example below shows the full exception response for the example used earlier - with the reason for the exception being identified as exception code '02'. This is the

'ILLEGAL DATA ADDRESS', which would typically be used if the master had requested the slave to read a non-existent status location.

FIELD NAME	HEX. VALUE	ASCII	RTU
slave address	09	30 39	0000 1001
function (as an exception for 01)	81	38 31	1000 0001
exception code	02	30 32	0000 0010
error check	-	LRC	CRC

The sections below describe the exception codes supported by the MTL838B-MBF in detail.

ILLEGAL FUNCTION (exception code 01)

The ILLEGAL FUNCTION exception code is used to inform the master that the function code received by the slave is not an allowable function for that slave.

An example of such a request would be the function code FORCE MULTIPLE COILS (function 15) sent to an MTL838B-MBF. This function is not supported by the MTL838B-MBF (because the design of the device does not require groups of coils to be set at a given moment). If the master were to send a request containing such a function code, an exception code '01' would be returned. The example below shows the exception response returned by such a slave, with address '04':

FIELD NAME	HEX. VALUE	ASCII	RTU
slave address	04	30 34	0000 0100
function (as an exception for 15)	95	39 35	1001 0101
exception code	01	30 31	0000 0000
error check	-	LRC	CRC

ILLEGAL DATA ADDRESS (exception code 02)

The ILLEGAL DATA ADDRESS exception code is used to inform the master that an address used in the query is not available within the slave.

An example of such a request would be the function code READ INPUT REGISTERS ('04') with the number of registers to be read given as 61, sent to an MTL838B-MBF. The MTL838B-MBF has a communication buffer that is only capable of containing sixty input registers, and a request to read more than this number could not be handled by the unit. The example below shows the exception response returned by such a slave, with address '09':

FIELD NAME	HEX. VALUE	ASCII	RTU
slave address	09	30 39	0000 1001
function (as an exception for 04)	84	38 34	1000 0100
exception code	02	30 32	0000 0010
error check	-	LRC	CRC

ILLEGAL DATA VALUE (exception code 03)

The ILLEGAL DATA VALUE exception code is used to inform the master that a value used in the query is not valid for the function requested form that slave.

An example of such a request would be the function code for DIAGNOSTICS with a diagnostic code of '01', sent to an MTL838B-MBF. The MTL838B-MBF does not support this diagnostic code, and would return the exception code above. The example below shows the exception response returned by such a slave, with address '06':

FIELD NAME	HEX. VALUE	ASCII	RTU
slave address	06	30 36	0000 0110
function (as an exception for 08)	88	38 38	1000 1000
exception code	03	30 33	0000 0011
error check	-	LRC	CRC

SLAVE DEVICE FAILURE (exception code 04)

The SLAVE DEVICE FAILURE exception code is used to inform the master that a error occurred in the slave whilst it was attempting to carry out the action required by the query.

An example of such a failure would be corruption of the configuration data stored by the MTL838B-MBF. The MTL838B-MBF would return a 'SLAVE DEVICE FAILURE' exception code if the configuration data was found to be corrupted and the master issued a request to READ INPUT REGISTERS '04' for registers that contained input status data. The example below shows the exception response returned by such a slave, with address '03':

FIELD NAME	HEX. VALUE	ASCII	RTU
slave address	03	30 33	0000 0011
function (as an exception for 04)	84	38 34	1000 0100
exception code	04	30 34	0000 0100
error check	-	LRC	CRC

NEGATIVE ACKNOWLEDGE (exception code 07)

The NEGATIVE ACKNOWLEDGE exception code is used to inform the master that the slave cannot perform the requested function.

An example of such a failure would be attempting to write data in to a register that was 'write disabled'. The MTL838B-MBF has a facility whereby the configuration data may be protected against over-writing. If this facility is used and the Modbus master attempts to write in to the configuration registers, then the exception code '07' will be returned. The following table shows the exception code returned by such a slave, with address 04.

FIELD NAME	HEX. VALUE	ASCII	RTU
slave address	04	30 33	0000 0011
function (as an exception for 06)	86	38 36	1000 0110
exception code	07	30 37	0000 0111
error check	-	LRC	CRC

BACKGROUND TO THE MTL838B-MBF

The analogue-input multiplexer system

The MTL838B-MBF is an analogue multiplexer receiver that is used with the MTL831B hazardous area millivolt input multiplexer transmitter. The status of up to 32 analogue inputs may be communicated from the hazardous area to the safe area via two data highways, each comprised of a simple twisted pair - over distances up to 500m.

Each data highway must be protected by an MTL3052 digital isolator when the inputs are located in a hazardous area. The MTL831B is typically used with thermocouple and RTD inputs and is intrinsically safe. It can be mounted in a zone 0 hazardous area and will accept 16 inputs.

Up to two MTL831B transmitters can be combined on a single MTL838B-MBF receiver input - up to a total of 32 analogue inputs - as shown in Figure 10.

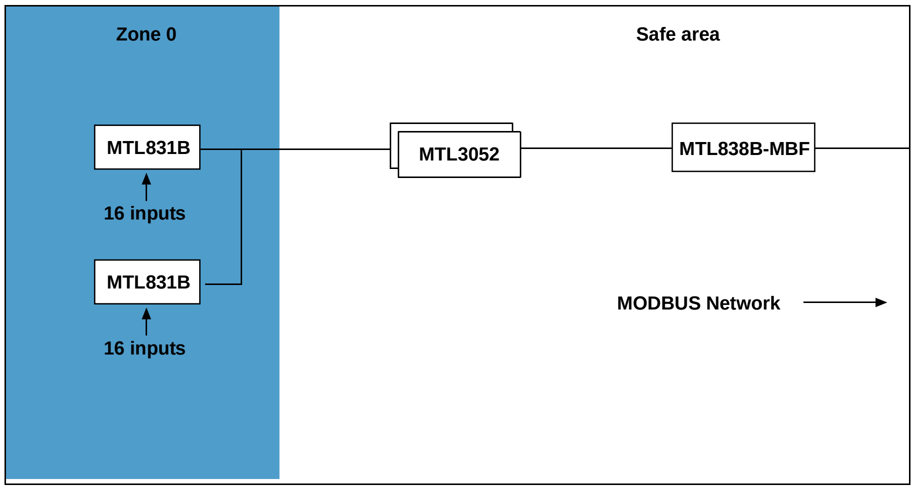


Figure 10 - Typical MTL838B-MBF configuration

The MTL838B-MBF acts as a Modbus slave. It may be connected into any standard Modbus network, with up to 31 MTL838B-MBF slaves on each network. If each unit has its full complement of 32 analogue inputs, the status of a total of 992 analogue inputs may be passed to a Modbus master using a single RS485 network.

Configuring the MTL838B-MBF

The MTL838B-MBF is configured by first setting some DIL switches within the unit and then setting the other configuration parameters using one of the following methods:

- on-line via the Modbus link, direct from the host
- off-line using specially written software (PCS83)

(The PCS83 software is the recommended method for initial configuration).

On-line

Configuring the unit via the Modbus master and the network might seem to be the simplest method at first sight, but there are a number of practical difficulties with this configuration technique. This approach means that the user must deal with a number of complex aspects which require a significant investment of the configurer's time, before they are understood fully. A further difficulty may be a lack of the necessary memory space within the Modbus master. If the configuration is likely to be changed frequently it could even be necessary for the system designer to design specific 'user interface' screens such as those used by the PCS83 software (see below), to allow changes to be made by operators. This would be a time consuming and costly task.

For most users, the attraction of being able to use the Modbus master to configure the unit is that the configuration can be re-sent if the slave's memory becomes corrupted. Whilst this is true, it is not possible to avoid the difficulties (and costs) outlined earlier and the decision to adopt a strategy of configuring via the Modbus master should be arrived at only after due consideration.

A cost effective compromise would be to perform the initial configuration via PCS83, and then read the configuration parameters stored in the MTL838B-MBF via the host. The stored parameters could then be re-written to the MTL838B-MBF should the configuration database ever become corrupt, or the battery back-up fail.

If a user intends to adopt the on-line configuration method, the calculation of configuration parameters for storage in the master can be simplified, and the possibility of 'human error' reduced, by using PCS83 to input the required data and data format, and then reading the stored values (encoded correctly in the required data format) back from the MTL838B-MBF via Modbus. The user should still realise that any subsequent alterations of the parameters will require further use of PCS83.

Off-line

Off-line configuration requires the use of either of the two methods described below. Once configured, the configuration parameters are stored in battery-backed memory within the unit. The battery will maintain the settings for at least 3 months (assuming storage at room temperature) - this can be ensured by leaving the unit powered up for 24 hours *before* configuration.

PCS83

By far the simplest method of configuring the MTL838B-MBF is using the PCS83 software. This software has been specifically designed to perform all of the complex calculations that must be carried out, in order to configure the unit. These calculations are transparent to the user, and this method provides a convenient and time efficient method. (See the section beginning at page 71 for full details.)

Alternatively, as explained before, the master could read the configured parameters *after* initial off-line configuration and these may then be stored within the host for use in the event of a database failure.

Hardware configuration of the MTL838B-MBF

Certain parameters are defined using DIL switches that are located within the unit.

To access the switches, the terminal strip on that side of the unit must first be removed. To access switch SW101, the terminal strip with terminals 22 to 42 (with connections to the highways and power supply) must be removed, to access switches SW302 and SW301, the terminal strip with terminals 1 to 21 (with connections to the RS485 outputs) must be removed.

The position of the switches within the unit is shown in the diagram below.

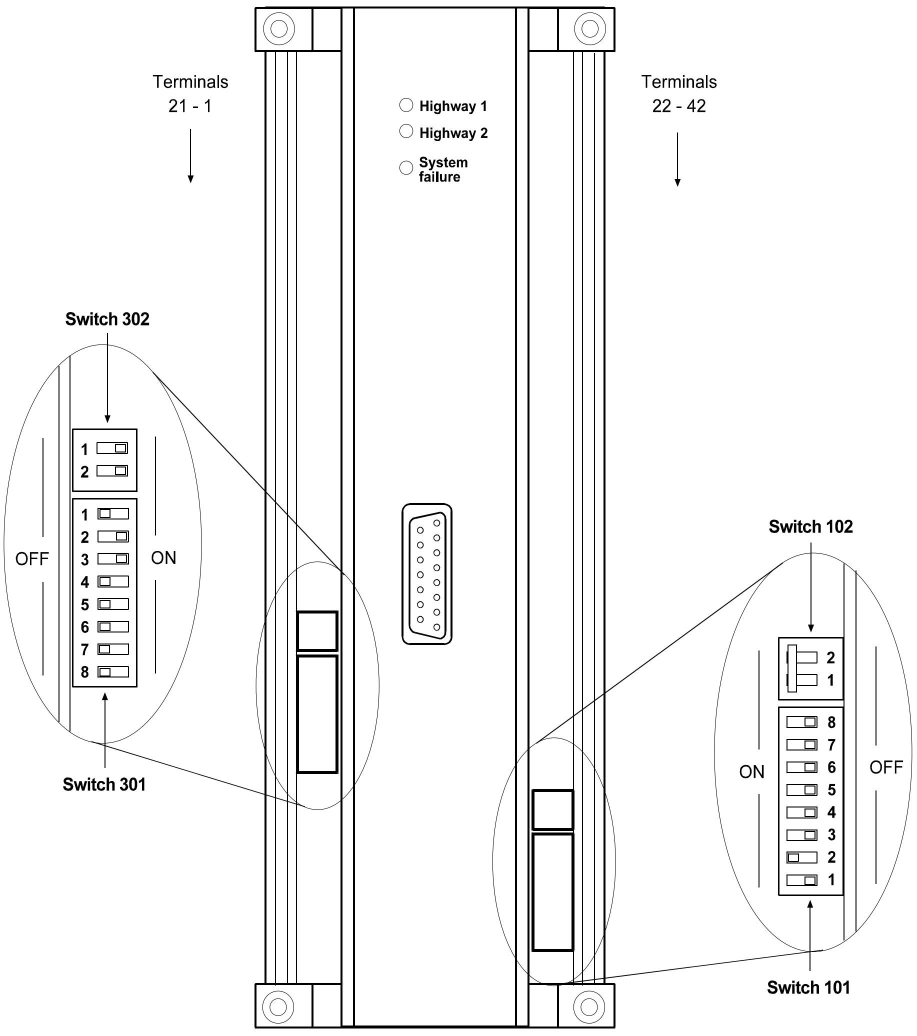


Figure 11 - DIL Switch locations

The parameters that must be defined before use are listed here.

- RS485 serial port communication parameters
- Modbus slave address (can also be software defined using PCS83, but not by the Modbus host)
- Transmission mode - ASCII or RTU
- Interconnection method - multi-drop or point-to-point
- Enable/disable configuration (via PCS83)
- Number of stop bits and parity

The tables below show the selections that are made by each lever of each switch.

Switch 101

LEVER & POSITION								SELECTION
1	2	3	4	5	6	7	8	
OFF	OFF	OFF	OFF	OFF	-	-	-	Address 0 - software defined
ON	OFF	OFF	OFF	OFF	-	-	-	Address 1
OFF	**ON**	OFF	OFF	OFF	-	-	-	Address 2
ON	**ON**	OFF	OFF	OFF	-	-	-	Address 3
OFF	OFF	**ON**	OFF	OFF	-	-	-	Address 4
ON	OFF	**ON**	OFF	OFF	-	-	-	Address 5
OFF	**ON**	**ON**	OFF	OFF	-	-	-	Address 6
ON	**ON**	**ON**	OFF	OFF	-	-	-	Address 7
OFF	OFF	OFF	**ON**	OFF	-	-	-	Address 8
ON	OFF	OFF	**ON**	OFF	-	-	-	Address 9
OFF	**ON**	OFF	**ON**	OFF	-	-	-	Address 10
ON	**ON**	OFF	**ON**	OFF	-	-	-	Address 11
OFF	OFF	**ON**	**ON**	OFF	-	-	-	Address 12
ON	OFF	**ON**	**ON**	OFF	-	-	-	Address 13
OFF	**ON**	**ON**	**ON**	OFF	-	-	-	Address 14
ON	**ON**	**ON**	**ON**	OFF	-	-	-	Address 15
OFF	OFF	OFF	OFF	**ON**	-	-	-	Address 16
ON	OFF	OFF	OFF	**ON**	-	-	-	Address 17
OFF	**ON**	OFF	OFF	**ON**	-	-	-	Address 18
ON	**ON**	OFF	OFF	**ON**	-	-	-	Address 19
OFF	OFF	**ON**	OFF	**ON**	-	-	-	Address 20
ON	OFF	**ON**	OFF	**ON**	-	-	-	Address 21
OFF	**ON**	**ON**	OFF	**ON**	-	-	-	Address 22
ON	**ON**	**ON**	OFF	**ON**	-	-	-	Address 23
OFF	OFF	OFF	**ON**	**ON**	-	-	-	Address 24
ON	OFF	OFF	**ON**	**ON**	-	-	-	Address 25
OFF	**ON**	OFF	**ON**	**ON**	-	-	-	Address 26
ON	**ON**	OFF	**ON**	**ON**	-	-	-	Address 27
OFF	OFF	**ON**	**ON**	**ON**	-	-	-	Address 28
ON	OFF	**ON**	**ON**	**ON**	-	-	-	Address 29
OFF	**ON**	**ON**	**ON**	**ON**	-	-	-	Address 30
ON	**ON**	**ON**	**ON**	**ON**	-	-	-	Address 31
-	-	-	-	-	OFF*	-	-	HAN83 configuration enabled
-	-	-	-	-	**ON***	-	-	HAN83 configuration disabled
-	-	-	-	-	-	OFF	-	serial configuration change enabled
-	-	-	-	-	-	**ON**	-	serial configuration change disabled
-	-	-	-	-	-	-	OFF	only allowed position for lever 8

* HAN83 configuration option is no longer available – set lever 6 to ON

Note: The convention used in the table indicates the switch lever position in relation to the PCB. Thus ' ON ' shows a switch lever set away from the PCB (in the 'UP' position) and ' OFF ' shows a lever set towards the PCB (in the 'DOWN' position).

Factory settings are: All positions – ‘OFF’

Switch 102

LEVER & POSITION		SELECTION
1	2	
OFF	OFF	Factory use only
ON	**ON**	Only user-allowed position for lever

Switch 301

LEVER & POSITION								RS485 PARAMETER SELECTION				
1	2	3	4	5	6	7	8	baud	data	start	stop	parity
OFF	OFF	OFF	OFF	-	-	-	-	300	8 bit	1	1	none
ON	OFF	OFF	OFF	-	-	-	-	1200	8 bit	1	1	none
OFF	ON	OFF	OFF	-	-		-	1200	8 bit	1	1	odd
ON	ON	OFF	OFF	-	-	-	-	1200	8 bit	1	1	even
OFF	OFF	ON	OFF	-	-	-	-	2400	8 bit	1	1	none
ON	OFF	ON	OFF	-	-	-	-	9600	8 bit	1	1	none
OFF	ON	ON	OFF	-	-	-	-	9600	8 bit	1	1	odd
ON	ON	ON	OFF	-	-	-	-	9600	8 bit	1	1	even
OFF	OFF	OFF	ON	-	-	-	-	9600	7 bit	1	1	none
ON	OFF	OFF	ON	-	-	-	-	9600	7 bit	1	1	odd
OFF	ON	OFF	ON	-	-	-	-	9600	7 bit	1	1	even
ON	ON	OFF	ON	-	-	-	-	19k2	8 bit	1	1	none
ON	ON	ON	ON	-	-	-	-	software definition of selection				
-	-	-	-	OFF	-	-	-	slave address channel B = channel A				
-	-	-	-	ON	-	-	-	slave address channel B = channel A + 32				
-	-	-	-	-	-	ON	-	ASCII transmission mode				
-	-	-	-	-	-	OFF	-	RTU transmission mode				

Notes:

1. The convention used in the table indicates the switch lever position in relation to the PCB. Thus ' ON ' shows a switch lever set away from the PCB (in the 'UP' position) and ' OFF ' shows a lever set towards the PCB (in the 'DOWN' position).

2. The selection of communication parameters by switches 1 to 4 only sets the parameters for communication via the RS485 outputs. The RS232 outputs that are used for configuration via PCS83 are always set to 9600 baud, 8 data bits, 1 start, 1 stop and no parity.

3. The selection of RTU transmission mode overrides any selection of 7-bit data, and sets the number of data bits to 8-bit.

4. Levers 6 and 8 are used for testing of the unit and have no other use.

5. Factory settings are: 1,3,5 – 'ON' 2,4,6,8 – 'OFF'

Switch 302

LEVER & POSITION		SELECTION
1	2	
ON	OFF	point-to-point RS485 connection (single MTL838B-MBF)
ON	ON	multi-drop connection (multiple MTL838B-MBFs)

Note: Factory settings are: Position 1 and 2 – 'ON'

Interconnection of the MTL838B-MBF

The MTL838B-MBF may be connected to a Modbus host in a number of ways—as was mentioned earlier it may be connected for multi-drop or point-to-point operation.

Two RS485 outputs, A and B, are provided on the MTL838B-MBF. As there are two outputs the unit can either be connected to a single Modbus master, with dual redundancy, or connected to two separate Modbus hosts.

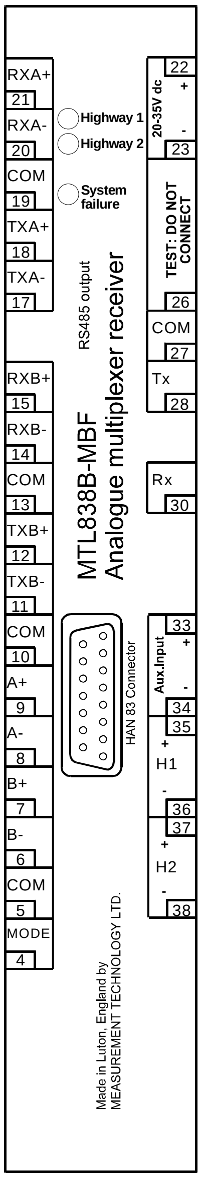


Figure 12 - Terminal arrangement of MTL838B-MBF

The MTL838B-MBF will respond on whichever RS485 connection the query is received, and there is no restriction placed on the simultaneous use of both interfaces. When using the second interface to communicate with a second Modbus master, the DIL switch that establishes the addressing of the second interface can be set to 'Add 32 for Port B addresses' if this is required. See page 41 for more detail.

Note: the RS232 terminals that are provided for configuration of the unit via PCS83 can NOT be used for Modbus communication and, similarly, the RS485 terminals cannot be used to configure the unit from PCS83. The D-connector on the top of the unit was used for the connection of the HAN83 configurator, **but this is no longer supported**.

The following diagrams show the terminal arrangement for the MTL838B-MBF and two methods of connection - 2-wire and 4-wire.

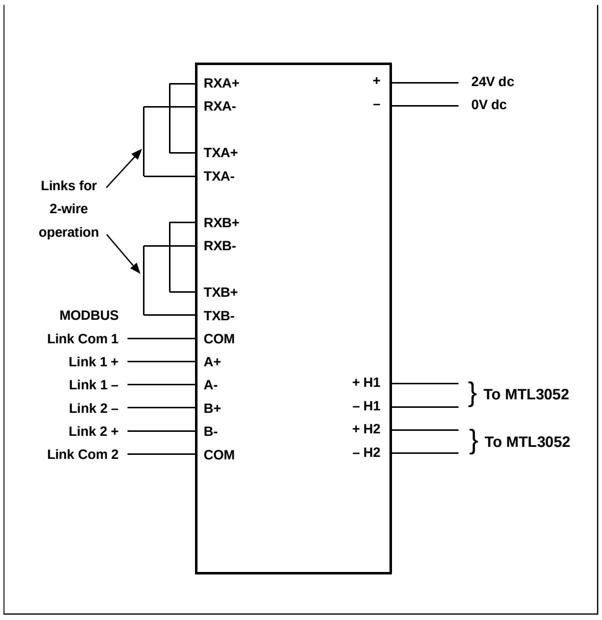


Figure 13 - 2-wire bus connection

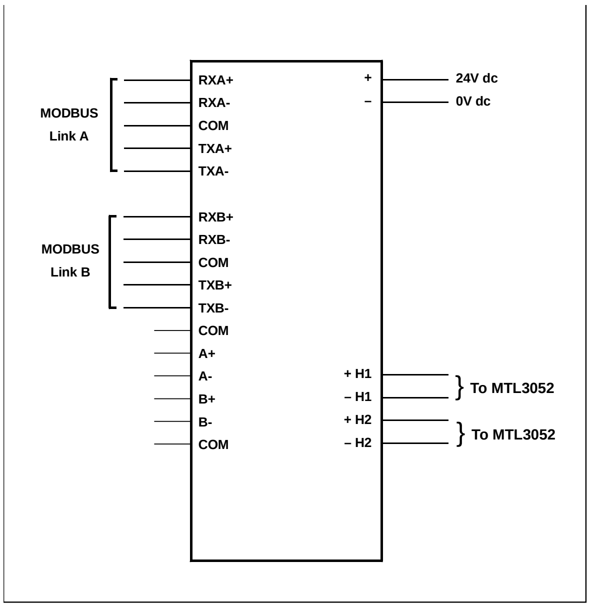


Figure 14 - 4-wire bus connection

There is no need with the MTL838B-MBF to disable an unused RS485 output. The unused connection can be left un-connected and floating.

Initialisation mode

The MTL838B-MBF has two distinct modes of operation - normal and initialisation.

It will always enter initialisation mode during power-up. It can also be triggered by the detection of internal hardware or software faults, or after receiving an instruction from the host to reset some or all of the configuration registers.

During initialisation, the unit will ignore all commands from the master.

The initialisation period will take several seconds to complete all the necessary operations and calculations. Following successful initialisation, the unit will automatically enter, or return to, normal operation mode.

If a corrupted configuration database is detected during initialisation the unit will revert to a set of default values, and on entering normal operation mode, will issue exception responses when requested by the host to read input values. Exception responses will continue to be issued until the unit is re-configured. The need to re-configure the unit will remain even if the MTL838B-MBF is powered down and back up.

If a corrupted configuration is detected, the slave address is not automatically reset. If the address was hardware defined then it can only be reset using the units DIL switches. If it is software definable, then the user has the choice of using a PC running PCS83 software.

Slave, Transmitter and Input addressing

The following discusses the allocation of addresses to the slaves on the Modbus network - including the MTL838B-MBF - and the allocation of addresses for the transmitters and inputs connected to each MTL838B-MBF.

Addressing MTL838B-MBF slaves

Modbus allows slave addresses in the range 1 to 247. JBUS allows slave addresses in the range 1 to 255. (This is the only difference between the two protocols). Since the MTL838B-MBF can only have addresses in the range 1 to 31, it will work equally well with either protocol.

The Modbus address for each MTL838B-MBF slave is set either via software (PCS83) or by hardware DIL switches within each unit. Using the hardware selection is recommended, as this allows the slave to identify it's own address even if all of the other configuration parameters are lost. The positioning of the DIL switches that set the slave address can be found on page 40. For the same reasons of security, it is not possible to set the address of the slave via the Modbus host.

Note: it is possible to use PCS83 to try and set the slave address to '0'. This is not allowed as a slave address as '0' is exclusively reserved for broadcast messages. If a '0' is written to a slave address it will be ignored and the slaves address will be set to '1'.

The hardware switches initially define the address of the first RS485 serial port of the unit (port 'A'). The address for port 'B' can either be identical to that for port 'A' or can be offset by 32, so that the same slave can be addressed as slave '1' via port 'A', or addressed as slave '33' via port 'B'. This facility allows the MTL838B-MBF to be connected to the same master twice, or to two different masters independently. (There is no restriction regarding simultaneous communication on both ports. The unit will respond via the port on which it received the query.)

Addressing the transmitters of each MTL838B-MBF

Each MTL831B transmitter accepts up to 16 sensor inputs.

The address of each transmitter connected to an MTL838B-MBF is set by DIL switches within each transmitter. There is no restriction on the allocation of addresses to the transmitters, except that each address may be used only once and that the addresses used must be contiguous from '1'. (i.e. if there are three transmitters, the addresses used must be '1', '2' and '3', but they can be allocated to the three slaves in any order.)

Note: the allocation of transmitter addresses controls the addressing of each input. See the next section.

Allocation of addresses to individual inputs

There are no facilities by which the user may define the address of individual inputs, other than by the allocation of transmitter addresses. This is structured so that the inputs to the first transmitter have addresses from 1 onwards, with input 'I1' having the address for 'input 1', input 'I2' having the address for 'input 2' etc.

The addressing of inputs continues sequentially through the inputs of subsequent transmitters.

INPUT STATUS FLAGS AND REGISTERS

The input status flags and input registers are used to store information that the master will want to read from the MTL838B-MBF. The data stored by the MTL838B-MBF is mapped twice, to the input status flags and to the input registers. The user can choose which of the two data stores is the simplest to read from, given the application in question.

Mapping of input status flags and input registers

The tables below show the mapping of the flag and register locations used by the MTL838B-MBF. The tables show the mappings with IEEE data format selected and with non-IEEE.

Mapping for IEEE data format

INPUT STATUS FLAG LOCATION	INPUT REGISTER LOCATION	NAME	DATA TYPE
10001 - 10032	30001 - 30002	HCA_REV	ASCII
10033 - 10064	30003 - 30004	IPP_REV	ASCII
10065 - 10096	30005 - 30006	STATUS	binary
10097 - 10128	30007 - 30008	CHECKSUM	binary
10129 -10160	30009 - 30010	HIGH_ALARM	binary
10161 - 10192	30010 - 30012	LOW_ALARM	binary
10193 - 10224	30013 - 00014	OPEN_ALARM	binary
10225 - 10256	30015 - 30016	INPUT_1	IEEE
10257 - 10288	30017 - 30018	INPUT_2	IEEE
10289 - 10320	30019 - 30020	INPUT_3	IEEE
10321 - 10352	30021 - 30022	INPUT_4	IEEE
10353 - 10384	30023 - 30024	INPUT_5	IEEE
10385 - 10416	30025 - 30026	INPUT_6	IEEE
10417 - 10448	30027 - 30028	INPUT_7	IEEE
10449 - 10480	30029 - 30030	INPUT_8	IEEE
10481 - 10512	30031 - 30032	INPUT_9	IEEE
10513 - 10544	30033 - 30034	INPUT_10	IEEE
10545 - 10576	30035 - 30036	INPUT_11	IEEE
10576 - 10608	30037 - 30038	INPUT_12	IEEE
10609 - 10640	30039 - 30040	INPUT_13	IEEE
10640 - 10671	30041 - 30042	INPUT_14	IEEE
10673 - 10704	30043 - 30044	INPUT_15	IEEE
10705 - 10736	30045 - 30046	INPUT_16	IEEE
10737 - 10768	30047 - 30048	INPUT_17	IEEE
10769 - 10800	30049 - 30050	INPUT_18	IEEE
10801 - 10832	30051 - 30052	INPUT_19	IEEE
10833 - 10864	30053 - 30054	INPUT_20	IEEE
10865 - 10896	30055 - 30056	INPUT_21	IEEE
10897 - 10928	30057 - 30058	INPUT_22	IEEE
10929 - 10960	30059 - 30060	INPUT_23	IEEE

(continued over page)

10961 - 10992	30061 - 30062	INPUT_24	IEEE
10993 - 11024	30063 - 30064	INPUT_25	IEEE
11025 - 11056	30065 - 30066	INPUT_26	IEEE
11057 - 11088	30067 - 30068	INPUT_27	IEEE
11089 - 11120	30069 - 30070	INPUT_28	IEEE
11121 - 11152	30071 - 30072	INPUT_29	IEEE
11153 - 11184	30073 - 30074	INPUT_30	IEEE
11185 - 11216	30075 - 30076	INPUT_31	IEEE
11217 - 11248	30077 - 30078	INPUT_32	IEEE
11249 - 11280	30079 - 30080	CJ1	IEEE
11281 - 11312	30081 - 30082	CJ2	IEEE

Note - when addressing the locations of the data given above, remember the anomaly that exists in Modbus, between the address passed by the function and the location within the slave. Thus locations 10001 - 10032 are addressed by the READ INPUT STATUS function with addresses 0000 - 0031, etc.

The contents of each location are explained more fully in the sections below.

Mapping for non-IEEE data format

INPUT STATUS FLAG LOCATION	**INPUT REGISTER LOCATION**	**NAME**	**DATA TYPE**
10001 - 10032	30001 -30002	HCA_REV	ASCII
10033 - 10064	30003 - 30004	IPP_REV	ASCII
10065 - 10096	30005 - 30006	STATUS	binary
10097 - 10128	30007 - 30008	CHECKSUM	binary
10129 -10160	30009 - 30010	HIGH_ALARM	binary
10161 - 10192	30011 - 30012	LOW_ALARM	binary
10193 - 10224	30013 - 30014	OPEN_ALARM	binary
10225 - 10240	30015	INPUT_1	non-IEEE
10241 - 10256	30016	INPUT_2	non-IEEE
10257 - 10272	30017	INPUT_3	non-IEEE
10273 -10288	30018	INPUT_4	non-IEEE
10289 - 10304	30019	INPUT_5	non-IEEE
10305 - 10320	30020	INPUT_6	non-IEEE
10321 - 10336	30021	INPUT_7	non-IEEE
10337 - 10352	30022	INPUT_8	non-IEEE
10353 - 10368	30023	INPUT_9	non-IEEE
10369 - 10384	30024	INPUT_10	non-IEEE
10385 - 10400	30025	INPUT_11	non-IEEE
10401 - 10416	30026	INPUT_12	non-IEEE
10417 - 10432	30027	INPUT_13	non-IEEE
10433 - 10448	30028	INPUT_14	non-IEEE
10449 - 10464	30029	INPUT_15	non-IEEE
10465 - 10480	30030	INPUT_16	non-IEEE
10481 - 10496	30031	INPUT_17	non-IEEE
10497 - 10512	30032	INPUT_18	non-IEEE
10513 - 10528	30033	INPUT_19	non-IEEE
10529 - 10544	30034	INPUT_20	non-IEEE
10545 - 10560	30035	INPUT_21	non-IEEE
10561 - 10576	30036	INPUT_22	non-IEEE
10577 - 10592	30037	INPUT_23	non-IEEE

10593 - 10608	30038	INPUT_24	non-IEEE
10609 - 10624	30039	INPUT_25	non-IEEE
10625 - 10640	30040	INPUT_26	non-IEEE
10641 - 10656	30041	INPUT_27	non-IEEE
10657 - 10672	30042	INPUT_28	non-IEEE
10673 - 10688	30043	INPUT_29	non-IEEE
10689 - 10704	30044	INPUT_30	non-IEEE
10705 - 10720	30045	INPUT_31	non-IEEE
10721 - 10736	30046	INPUT_32	non-IEEE
10737 - 10752	30047	CJ1	non-IEEE
10753 - 10768	30048	CJ2	non-IEEE

Note - when addressing the locations of the data given above, remember the anomaly that exists in Modbus, between the address passed by the function and the location within the slave. Thus locations 10001 - 10032 are addressed by the READ INPUT STATUS function with addresses 0000 - 0031, etc.

The contents of each location are explained more fully in the sections below.

Revision number of HCA software

INPUT STATUS FLAG LOCATIONS: 10001 - 10032

INPUT REGISTER LOCATIONS: 30001 - 30002

Four ASCII characters are provided which identify the software revision number revision number: 'HCA_REV' of the EPROM installed on the HCA (Host Communications Adapter) board. This software controls the communication between the MTL838B-MBF and the Modbus master.

Revision number of IPP software

INPUT STATUS FLAG LOCATIONS: 10033 - 10064

INPUT REGISTER LOCATIONS: 30003 - 30004

Four ASCII characters are provided which identify the software revision number revision number: 'IPP_REV' of the EPROM installed on the IPP (Input Processor) board. This software controls the communication between the MTL838B-MBF and the transmitters (MTL831B and/or MTL832EXE).

MTL838B-MBF status information

INPUT STATUS FLAG LOCATIONS: 10065 - 10096

INPUT REGISTER LOCATIONS: 30005 - 30006

A total of 16 STATUS bits are provided which inform the master of the overall status of the MTL838B-MBF. The table below shows the meaning of each bit, with the input status flag and the input register locations shown for each bit.

INPUT STATUS FLAG LOCATION	INPUT REGISTER LOCATION + BIT	MEANING
10096	30006: bit 0	Error flag - set when any of the status bits '8' to '15' are set to '1'
10095	30006: bit 1	Configuration changing
10094	30006: bit 2	Configuration rejected - due to internal MTL838B-MBF fault
10093	30006: bit 3	Invalid database
10092	30006: bit 4	Reserved
10091	30006: bit 5	Auxiliary Input Open
10090	30006: bit 6	Highway 1 OK
10089	30006: bit 7	Highway 2 OK
10088	30006: bit 8	Transmitter 1 failed
10087	30006: bit 9	Transmitter 2 failed
10086	30006: bit 10	Transmitter 3 failed
10085	30006: bit 11	Transmitter 4 failed
10084	30006: bit 12	Unused
10083	30006: bit 13	Open circuit detected on any input
10082	30006: bit 14	Low alarm detected on any input
10081	30006: bit 15	High alarm detected on any input

All status bits are set to logic '1' for the 'true' condition.

The bits in flag locations 10081 to 10096 and in register 30006 are unassigned, and are set to '0'.

The functions 'READ EXCEPTION STATUS' (see also page 27) and 'RETURN DIAGNOSTICS REGISTER' (see also page 29) are defined so that they read part of the STATUS register. These functions may be more convenient methods of accessing status data.

'Error flag'

INPUT STATUS FLAG LOCATION:	10096
INPUT REGISTER LOCATION:	30006: bit 0

The 'Error flag' is set when any of the bits '8' to '15' are set to '1'. This single bit then shows that there is a fault with some part of the system. Monitoring this bit alone can allow the Modbus master to maintain a check on the correct operation of the slave and to monitor for any alarms on the slaves inputs, without absorbing a significant amount of communication time. If this 'Error flag' is monitored in this way, once an error has been detected, the master can quickly establish which areas to investigate by examining the whole of the 30006 status register. The master can then take appropriate action according to which of the other status bits has caused the 'Error flag' to be set.

'Configuration changing'

INPUT STATUS FLAG LOCATION:	10095
INPUT REGISTER LOCATION:	30006: bit 1

The 'Configuration change detected' flag is set when configuration changes are being implemented, but have not been completed.

'Configuration rejected - due to internal MTL838B-MBF fault'

INPUT STATUS FLAG LOCATION:	10094
INPUT REGISTER LOCATION:	30006: bit 2

An internal fault within the MTL838B-MBF has caused a configuration change to be rejected.

'Invalid database'

INPUT STATUS FLAG LOCATION:	10093
INPUT REGISTER LOCATION:	30006: bit 3

The 'Invalid database' flag is set when a fault is detected in the configuration database.

'Auxiliary input open'

INPUT STATUS FLAG LOCATION:	10091
INPUT REGISTER LOCATION:	30006: bit 5

The 'Auxiliary Input Open' bit indicates that there is no connection made between the 'Aux. Input' and 'COM' terminals. This is used with the earth leakage detector.

'Highway 1 OK' and 'Highway 2 OK'

INPUT STATUS FLAG LOCATIONS:	10089 - 10090
INPUT REGISTER LOCATIONS:	30006: bits 6 - 7

The 'Highway OK' bits indicate that communication is taking place successfully on each highway. This corresponds to the illumination of the 'Highway OK' LED of the unit.

'Transmitter failed'

INPUT STATUS FLAG LOCATIONS:	10085 - 10088
INPUT REGISTER LOCATIONS:	30006: bits 8 - 11

The 'Transmitter failed' bits are set when the MTL838B-MBF is not receiving data from a particular transmitter. The bits will only be set when a transmitter is identified as being present by the configuration parameter N_TY_MPX (see pages 56 & 60). This defines the number of Tx devices connected to the MTL838B.

'Open circuit -', 'Low alarm -' and 'High alarm detected on any input'

INPUT STATUS FLAG LOCATIONS:	10081 - 10083
INPUT REGISTER LOCATIONS:	30006: bits 13 - 15

The 'Open circuit -', 'Low alarm -' and 'High alarm detected on any circuit' flags will be set if any of the inputs from the field are, respectively, open circuit or showing low or high alarms.

Configuration checksum

INPUT STATUS FLAG LOCATIONS:	10097 - 10128
INPUT REGISTER LOCATIONS:	30007 - 30008

The CHECKSUM registers contain the checksum of the configuration database. The checksum is calculated from the contents of the holding registers. All of the holding registers that define the configuration of the unit (i.e. all holding registers except 'CSUMREF' and the unused 'SPARE' registers) are used to calculate the CHECKSUM. This value can then be used in a number of ways to ensure that any corruption of the configuration database is detected.

The calculation and use of checksums is discussed in more detail on page 60.

High alarm status register

INPUT STATUS FLAG LOCATIONS:	10129 - 10160
INPUT REGISTER LOCATIONS:	30009 - 30010

The 'HIGH_ALARM' flags and register bits indicate the presence of a high alarm on the inputs to the field transmitters. Each of the possible 32 inputs is allocated a bit within the 32 bits and two registers.

The bits are arranged so that a high alarm on input 1 will set the input status flag at location 10129 and the most significant bit of register 30009 and so on through the 32 inputs, with the most significant bit of register 30010 (and location 10145) corresponding to input 17.

If any of the above bits are set to '1', then the associated bit in the STATUS register will also be set to '1'.

Low alarm status register

INPUT STATUS FLAG LOCATIONS:	10161 - 10192
INPUT REGISTER LOCATIONS:	30011 - 30012

The 'LOW_ALARM' flags and register bits indicate the presence of a low alarm on the inputs to the field transmitters. Each of the possible 32 inputs is allocated a bit within the 32 bits and two registers.

The bits are arranged so that a low alarm on input 1 will set the input status flag at location 10161 and the most significant bit of register 30011 and so on through the 32 inputs, with the most significant bit of register 30012 (and location 10167) corresponding to input 17.

If any of the above bits are set to '1', then the associated bit in the STATUS register will also be set to '1'.

Open alarm status register

INPUT STATUS FLAG LOCATIONS:	10193 - 10224
INPUT REGISTER LOCATIONS:	30013 - 30014

The 'OPEN_ALARM' flags and register bits indicate the presence of an open circuit alarm on the inputs to the field transmitters. Each of the possible 32 inputs is allocated a bit within the 32 bits and two registers.

The bits are arranged so that an open alarm on input 1 will set the input status flag at location 10193 and the most significant bit of register 30013 and so on through the 32 inputs, with the most significant bit of register 30014 (and location 10209) corresponding to input 17.

If any of the above bits are set to '1', then the associated bit in the STATUS register will also be set to '1'.

Once an open alarm has been detected by the MTL838B-MBF, the safety drive for each input will be engaged to drive the input high or low. See also page 60.

Scaled analogue input value

With IEEE format data:

INPUT STATUS FLAG LOCATIONS:	10225 - 11248
INPUT REGISTER LOCATIONS:	30015 - 30078

With non-IEEE format data:

INPUT STATUS FLAG LOCATIONS:	10225 - 10736
INPUT REGISTER LOCATIONS:	30015 - 30046

The scaled analogue values of each input are stored in the registers 'INPUT_1' to 'INPUT_32'.

If an IEEE data format is chosen, the value of each input is stored in 32 sequential flag locations and is also mapped to two input registers.

If a non-IEEE format is chosen, then the system uses 16 sequential flag locations and a single input register to store the value of each input.

The convention adopted for mapping of data in to the flags and registers is a function of the data format selected. Data format '1' maps the most significant bits of the data value in to the least significant register location (and the least significant bit in to the highest register location). All other data formats map the most significant bit of the data value in to the lowest flag location (and the most significant bit of the lowest register location).

Cold junction temperature of MTL831Bs

With IEEE format data:

INPUT STATUS FLAG LOCATIONS:	11249 - 11312
INPUT REGISTER LOCATIONS:	30079 - 30082

With non-IEEE format data:

INPUT STATUS FLAG LOCATIONS:	10737 - 10768
INPUT REGISTER LOCATIONS:	30047 - 30048

The temperature of the cold junction in each of the MTL831Bs connected to the MTL838B-MBF is stored - in either IEEE or non-IEEE data format - in the flag and register locations shown above. The first flag location contains the most significant bit of the temperature of the first MTL831B (CJ1) and so on.

If a non-IEEE data format is selected, the CJ temperature is stored in tenths of degrees. Further, if the data format chosen is unsigned, an offset of 40^o is applied by the MTL838B-MBF, so that CJ temperatures down to -40^o can be reported. Hence, for non-IEEE unsigned data format, a stored value of 678 corresponds to a temperature of:

$$678 = 10\,((\,I/P + 40\,) - 0\,) + 0$$

$$67.8 = I/P + 40$$

$$I/P = 67.8 - 40 = \underline{27.8}^o$$

Note that the ‘degrees’ refer to whichever unit of temperature (°C, °F or °K) is specified in Holding Register 40030.

COIL STATUS FLAGS

A small number of single bit coil status flags are set aside for the Modbus master to read from and write to. The facility to write to these flags is not disabled by the internal DIL switches which disable the configuration parameters write facility.

The coil status flags are only of use when the configuration of the MTL838B-MBF is being done via the Modbus host.

Note: the flags that reset the unit to factory default values cause the unit to perform a significant number of internal operations. This process can take several seconds, and during this time the unit is unable to communicate with the master.

Mapping of coil status flags

The mapping of the six coil status flags within the MTL838B-MBF is shown below:

COIL STATUS FLAG LOCATION	NAME	FUNCTION
00001	CSTORE	store the current CHECKSUM value
00002	DFT831	configure to factory defaults, mV inputs
00003	DFT832	configure to factory defaults, mA inputs
00004	CONFIRM	confirms configuration completed correctly
00005	FMT831	as DFT831, without re-setting data format
00006	FMT832	as DFT832, without re-setting data format

Store the current checksum value

COIL STATUS FLAG LOCATION: 00001

Writing a '1' in to the 'CSTORE' location causes the MTL838B-MBF to copy the current value of CHECKSUM in to the holding register CSUMREF. This allows the MTL838B-MBF to monitor any changes to the value of it's CHECKSUM, without the need for the master to hold a copy of the value.

Whilst the value in CSTORE remains as '1', any alteration to the CHECKSUM value (because of re-configuration) will cause this new value to be copied in to CSUMREF. This updating will continue until a '0' is written in to CSTORE.

The use of checksum data is discussed in more detail on page 60.

Set factory defaults for mV inputs

COIL STATUS FLAG LOCATION: 00002

Writing a '1' into the coil status flag location DFT831 will cause the MTL838B-MBF to re-set itself and to install factory default values in to its configuration database, assuming that the field inputs are mV inputs to MTL831B units.

On receiving the instruction to force the coil status flag DFT831 to a logic '1', the MTL838B-MBF will issue a response confirming receipt of the instruction and then enter 'initialisation mode' for several seconds. During this time, the unit is unable to communicate with the master and any queries addressed to the unit will be ignored.

The default parameters are:

- 1 MTL831B transmitter
- Data format type IEEE754 (type 0)
- All inputs mV type
- IPZERO and OPZERO scaling parameters 0
- GAIN parameters set to '1', i.e. values read directly as mV and degrees C.

Confirm database correctly configured

COIL STATUS FLAG LOCATION: 00004

A hazard exists with the MTL838B-MBF, whereby it would be possible for the unit to become re-configured, and for the master to be unaware that this had taken place. This could arise following a 'power-up' sequence in which the MTL838B-MBF detects that it's stored CONFIGURATION DATABASE has become corrupted (so that the factory default values for configuration are used instead).

To protect against this risk, once such a re-configuration has occurred, the slave will respond to any READ DATA requests by issuing an EXCEPTION response. Only when the master writes a logic '1' to the CONFIRM flag location will the slave allow data to be read.

The requirement to write to the CONFIRM flag location CONFIRM remains, even if the unit is subjected to further power-down and power-up cycles.

A similar precaution must be taken to prevent the master reading data when it has instructed the slave to use a new DATAFORMAT, but before the CONFIGURATION DATABASE has been re-written in the new DATAFORMAT.

Again, to prevent the master reading data that is not configured correctly, any READ DATA queries will give rise to EXCEPTION responses, until the CONFIRM flag is set to '1'. The requirement to write to the CONFIRM flag location remains, even if the unit is subjected to further power-down and power-up cycles.

Note: Confirmation of a change in configuration database can also be achieved with PCS83. The 'sign-off' operation in PCS83, issues an instruction equivalent to 'CONFIRM'.

Set factory defaults for mV inputs, leaving DATAFORMAT unchanged

COIL STATUS FLAG LOCATION: 00005

Writing a logic '1' to status flag FMT831 performs the same operation as DFT831, but leaves the DATAFORMAT register unaltered. This allows the unit to be reset to factory default values, and then allows the master to write a known DATABASE CONFIGURATION in the required format of data.

HOLDING REGISTERS

The holding registers of the MTL838B-MBF are used almost exclusively to hold data regarding the configuration of the unit. A few unused registers are available for retaining other data if required. All configuration database parameters are stored in battery-backed RAM.

The layout of the holding registers is summarised in the table below:

HOLDING REGISTERS	NAME	DATA TYPE	DESCRIPTION
40001 - 40002	CSUMREF	B	CHECKSUM reference value
40003 - 40015	SPARE	A	unused registers
40016	DATAFMT	B	output data format
40017 - 40028	TAG	A	tag string - defined by user
40029	N_TY_MPX	B	number and type of transmitters
40030	UNIT	B	units of temperature to use
40031	POWER	B	frequency of power supply
40032	CFGTEST	B	configuration test method
40033	IPTYSF_1	DF	input type and safety drive, input 1
:	:	:	:
40064	IPTYSF_32	DF	input type and safety drive, input 32
40065 - 40066	IPZERO_1	DF	zero for input 1
:	:	:	:
40127 - 40128	IPZERO_32	DF	zero for input 32
40129 - 40130	GAIN_1	DF	gain for input 1
:	:	:	:
40191 - 40192	GAIN_32	DF	gain for input 32
40193 - 40194	HA_1	DF	high alarm for input 1
:	:	:	:
40255 - 40256	HA_32	DF	high alarm for input 32
40257 - 40258	LA_1	DF	low alarm for input 1
:	:	:	:
40319 - 40320	LA_32	DF	low alarm for input 32
40321 - 40322	OPZERO_1	DF	output zero for input 1
:	:	:	:
40383 - 40384	OPZERO_32	DF	output zero for input 32

Configuration checksum reference

HOLDING REGISTER LOCATION: 40001 - 40002

The configuration checksum reference CSUMREF, is used to store a checksum value that can then be used to detect any changes to the CHECKSUM of the configuration database.

CSUMREF is normally updated by copying the value of CHECKSUM to the holding register whenever the configuration - and thus the checksum - is changed. The reference value may be calculated and written to the CSUMREF by the master, but this is less common.

The use of checksum data is discussed in detail on page 60.

Unused holding registers

HOLDING REGISTER LOCATIONS: 40003 - 40015

A number of holding registers are provided that have no defined use. ASCII data can be written to and read from these registers, according to the needs of each user. A typical example of the use of these registers would be to store the last date of calibration check.

Data format selection

HOLDING REGISTER LOCATION: 40016

The DATAFMT register is used to select the format of the data stored by the MTL838B-MBF in those holding registers identified by 'DF' in the tables and all of its sensor input registers. (In the tables showing the contents of each register, those which are governed by DATAFMT are marked 'DF').

When a new value is written to the DATAFMT register, the scaling parameters become invalid as they conformed to the previously set data format. Any attempts to read data from these registers will cause the unit to issue an EXCEPTION response, until the CONFIRM flag is set to '1' (after the master has re-written the scaling parameters in the new data format).

The table below shows the decimal values that must be written (in binary) to the DATAFMT register to select each of the defined data formats:

DATA FMT	**DESCRIPTION OF FORMAT**	**value stored in register**	**decimal value**
0	IEEE single precision, floating point. Most significant data in lowest register address	0 to FFFFH	$-3.4x10^{38}$ to $+3.4x10^{38}$
1	IEEE single precision, floating point. Most significant data in highest register address	0 to FFFFH	$-3.4x10^{38}$ to $+3.4x10^{38}$
4	Unsigned 16-bit binary	0 to FFFFH	0 to 65535
5	Offset 16-bit binary	0 to FFFFH	-32768 to +32767
6	2's complement 16-bit binary	0 to FFFFH	-32768 to +32767
7	Signed 16-bit binary	0 to FFFFH	-32768 to +32767

(continued over page)

8	Unsigned 12-bit binary	0 to FFFH	0 to 4095
9	Offset 12-bit binary	0 to FFFH	-2048 to +2047
10	2's complement 12-bit binary	0 to FFFH	-2048 to +2047
11	Signed 12-bit binary	0 to FFFH	-2048 to +2047
12	Unsigned 4-decade BCD	0 to 9999 (BCD)	0 to 9999
13	Offset 4-decade BCD	0 to 9999 (BCD)	-5000 to +4999
14	10's complement 4-decade BCD	0 to 9999 (BCD)	-5000 to +4999
16	Unsigned 3-decade BCD	0 to 999 (BCD)	0 to 999
17	Offset 3-decade BCD	0 to 999 (BCD)	-500 to +499
18	Offset 10's comp. 3-decade BCD	0 to 999 (BCD)	-500 to +499

In many of the non-IEEE data formats specified in the table above, the encoding of the value in to the chosen format is not immediately apparent. The table below explains the encoding of each format. The table shows the decimal value of the binary, hexadecimal or BCD content of the register, and for each range of values for each data type, the formula for finding the 'represented value' is given.

DATAFMT	RANGE OF VALUES (dec. equivalent)	FORMULA FOR REPRESENTED VALUE
4	0 to 65535	RV = REG
5	0 to 65535	RV = REG - 32768
6	0 to +32767	RV = REG
	32768 to 65535	RV = REG - 65536
7	0 to +32767	RV = - REG
	32768 to 65535	RV = REG - 32768
8	0 to 4095	RV = REG
9	0 to 4095	RV = REG - 2048
10	0 to 2047	RV = REG
	2048 to 4095	RV = REG - 4096
11	0 to 2047	RV = - REG
	2048 to 4095	RV = REG - 2048
12	0 to 9999	RV = REG
13	0 to 9999	RV = REG - 5000
14	0 to 4999	RV = REG
14	5000 to 9999	RV = REG - 10000
16	0 to 999	RV = REG
17	0 to 999	RV = REG - 500
18	0 to 499	RV = REG
	500 to 999	RV = REG - 1000

Notes:

1. The conventions used in the table are that 'RV' is the represented value and 'REG' is the decimal equivalent of the registers contents (which will actually be in hexadecimal or binary).

2. The encoding of some parameters in non-IEEE format require further manipulation to be expressed as 'numerand and exponent'. See Appendix C.

3. The encoding of IEEE data is described in Appendix D.

Tag field

HOLDING REGISTER LOCATION: 40017 - 40028

The TAG holding register will contain the ASCII string 'MTL ANALOGUE MULTIPLEXER' as a default, which may be modified to a more suitable TAG by the master. Each TAG register will hold two ASCII characters, giving a maximum of 24 characters stored.

Number and type of transmitters

HOLDING REGISTER LOCATION: 40029

The holding register N_TY_MPX contains a binary value that is encoded to describe the number and type of transmitters connected to the data highways of any given MTL838B-MBF receiver. The information is encoded in to the binary value by a series of multiplications and additions. The result is a binary value that uniquely describes the number and type of receivers. The calculation of the value is shown below:

N_TY_MPX = MPX1_TYPE
+ (8 x MPX2_TYPE)
+ (64 X MPX3_TYPE)
+ (512 x MPX4_TYPE)
+ (4096 x NUM_MPX)

where: 'MPXn_TYPE' defines the type of transmitter 'n', and:

MPXn_TYPE = 1 for MTL831B

MPXn_TYPE = 2 for MTL832

'NUM_MPX' defines the number of transmitters, and

for one transmitter:	NUM_MPX = 1
for two transmitters:	NUM_MPX = 2
for three transmitters:	NUM_MPX = 3
for four transmitters:	NUM_MPX = 4

Units of temperature

HOLDING REGISTER LOCATION: 40030

The binary value stored in the UNIT register defines the units that are used for the temperature readings made by the MTL831B multiplexer receiver for thermocouples, RTDs and cold junctions. The value is stored as the binary equivalent of the decimal values:

1:	degrees Centigrade (^{o}C)
2:	degrees Fahrenheit (^{o}F)
3:	Kelvin (K)

Line frequency of power supply

HOLDING REGISTER LOCATION: 40031

The value that is placed in the holding register POWER identifies the frequency of the local a.c. power supply. This is then used to establish the line frequency which should be rejected. The frequency is identified by writing a the following decimal values in to the register:

0:	50Hz supply	1:	60Hz supply

Configuration test method

HOLDING REGISTER LOCATION: 40032

The value that is placed in holding register CFGTEST is used to determine if the CSUMREF value is updated automatically by the MTL838B-MBF, whenever the configuration database is changed, or if it is only updated when instructed to do so by the master. Automatic updating is selected by writing a '0' to the CFGTEST register, updating by command from the master is selected by a '1'.

The table below shows the various combinations of CSTORE and CFGTEST and describes the conditions under which CSTORE is re-written. A comparison is made between CSUMREF and CHECKSUM every few minutes.

TYPE	CSTORE	CFGTEST	Conditions for re-writing CSUMREF
Automatic transfer	0	0	CSUMREF is updated automatically each time the configuration database is changed
Master defines	0	1	the previous value of CSUMREF is maintained, unless the master writes a new value directly to the CSUMREF register
Controlled transfer	0-1-0	1	CSUMREF is updated whenever the CHECKSUM value changes

The conditions under which the CSUMREF will be re-written and the conditions under which a comparison will be made between CSUMREF and CHECKSUM are governed by CSTORE and CFGTEST (see page 54 for more information).

Input type and safety drive

HOLDING REGISTER LOCATIONS: 40033 - 40064

The holding registers INPTYSF_1 to INPTYSF_32 are used to store information regarding the type of field input to the multiplexer transmitter and the safety drive that is specified for each input. The actual contents of the register are binary encoded values that uniquely describe the input type and safety drive selected for each input.

The input type selected can be one of a wide range of inputs. The value that is used here is used in conjunction with the value calculated for N_TY_MPX, which defines the number and type of transmitters connected to each MTL838B-MBF.

The safety drive comes in to action on detection of an open circuit sensor (if open sensor detection is selected) or if a transmitter is found to have failed. The appropriate OPEN_ALARM and/or transmitter failed STATUS bit will be set and the input will be driven to it's full scale or lowest value (depending on the selection of safety drive). If HIGH_AL or LOW_AL are selected, these will also be triggered by the safety drive.

If no safety drive is selected, if a transmitter fails or an input becomes open circuit, the MTL838B-MBF will continue to supply the most up-to-date information it has received. This then allows the host to read values that may differ widely from the actual measured value in the field. It is the users responsibility to ensure that the data read from the MTL838B-MBF is valid, either by the judicious use of safety drives (which is recommended) and/or by continually monitoring that the unit and it's inputs are giving valid readings by way of the STATUS information.

By default, the upscale safety drive will be selected.

The value is calculated as follows:

INPTYSF_n = IPTYPE_n + (256 x SAFETY_n)

'IPTYPE_n' is the type of input connected to input 'n', as shown in the table below:

IPTYPE_n	MTL831B
0	mV voltage input (scaleable)
1	E-type THC temp without CJ comp.
2	J-type THC temp without CJ comp.
3	K-type THC temp without CJ comp.
4	N-type THC temp without CJ comp.
5	R-type THC temp without CJ comp.
6	T-type THC temp without CJ comp
7	E-type THC temp with CJ comp.
8	J-type THC temp with CJ comp.
9	K-type THC temp with CJ comp.
10	N-type THC temp with CJ comp.
11	R-type THC temp with CJ comp.
12	T-type THC temp with CJ comp.
13	RTD, resistance readout
14	RTD, temperature readout
15	Switch (not recommended)
16	S-type THC temp without CJ comp
17	S-type THC temp with CJ comp
18	E-type THC mV with CJ comp.
19	J-type THC mV with CJ comp.
20	K-type THC

	mV with CJ comp.
21	N-type THC mV with CJ comp.
22	R-type THC mV with CJ comp.
23	T-type THC mV with CJ comp.
24	S-type THC mV with CJ comp.
25	B-type THC mV with CJ comp.
26	B-type THC temp without CJ comp
27	B-type THC temp with CJ comp

'SAFETY_n' is the type of safety drive selected, as shown in the table below:

SAFETY_n	SAFETY SELECTION
0	no safety drive selected
1	upscale safety drive selected
2	downscale safety drive selected

The open sensor detection and the output value that will be given by the upscale and downscale drives depends on the input type selected. The table below shows the values that will be read on the outputs when driven upscale or downscale after the detection of an open circuit input. These values will be reached if the full scale values of the selected data format are sufficiently wide to include these values.

SENSOR TYPE	DOWNSCALE LIMIT	UPSCALE LIMIT
thermocouple	-120mV	+120mV
resistance	1200Ω	0Ω
RTD temperature	temp. equivalent to 1200Ω	temp. equivalent to 0Ω
millivolt	-120mV	+120mV

Input zero with offset - for scaling output measurements

HOLDING REGISTER LOCATIONS: 40065 - 40128

The two IPZERO_n registers for each input are used to hold the value of the input zero after the offset has been applied. Each IPZERO_n will be found from:

IPZERO_n = input zero + offset

Where:

input zero:	is the lowest input value that may be recorded by the field input
offset:	is the value included by the MTL838B-MBF to allow negative numbers to be represented by unsigned data formats.

Calculation of scaled output values is discussed in detail on page 68.

Note: as with many other database configuration parameters, the values stored in IPZERO are of the data format selected by the user. When non-IEEE data is stored, the two registers hold a 'numerand' and an 'exponent' of the required value. This allows the selected data format to provide a broader range of values than would otherwise be possible. The calculation of such values is discussed in Appendix C.

Gain - for scaling output measurements

HOLDING REGISTER LOCATIONS: 40129 - 40192

The two GAIN_n registers are used to hold a value termed 'gain' for each of the n inputs. This value is used in conjunction with the IPZERO, to give the required range (or span) of measurements. Gain for each input is normally calculated as below:

GAIN = (Output FSD - Output zero) / (Input FSD - Input zero)

Calculation of scaled output values is discussed in detail on page 68.

Note: as with IPZERO_n, for non-IEEE data formats, the values for GAIN_n are stored as 'numerand' and 'exponent' according to the data format chosen. This is discussed further in Appendix C.

High alarm level

HOLDING REGISTER LOCATIONS: 40193 - 40256

The HA_n registers are used to store the level which should not be exceeded by the scaled output value. If the scaled output does exceed this level, the appropriate bit within the HIGH_ALARM input register will be set, and the 15th bit of the STATUS register will also be set.

Note: for non-IEEE data, the value for HA_n will be stored as a 'numerand' and 'exponent'. See Appendix C.

Low alarm level

HOLDING REGISTER LOCATIONS: 40257 - 40320

The LA_n registers are used to store the level below which the scaled output value should not go. If the scaled output does fall below this level, the appropriate bit within the LOW_ALARM input register will be set, and the 16th bit of the STATUS register will also be set.

Note: for non-IEEE data, the value for LA_n will be stored as a 'numerand' and 'exponent'. See Appendix C.

Output zero offset

HOLDING REGISTER LOCATIONS: 40321 - 40384

The value stored in the two OPZERO_n registers matches the lowest value of output that is required from the nth output - corresponding to the lowest value of the nth input.

Calculation of scaled output values is discussed in detail on page 68.

Note: as with IPZERO_n, for non-IEEE data formats, the values for OPZERO_n are stored as 'numerand' and 'exponent' according to the data format chosen. This is discussed further in Appendix C.

MTL838B-MBF EXCEPTION RESPONSES

The following section describes the exception responses that may be given to each type of query that may be received by the MTL838B-MBF.

Exception responses are only issued by Modbus slaves if a query that is received correctly (i.e. passes the error and parity checks) cannot be carried out by the slave. An exception response is constructed by returning the received function code to the master with it's MSB set to '1', followed by an exception code, passed back to the master as the first byte of the data field. See pages 16 and 34 for more detail.

Following 'READ COIL STATUS' queries

EXCEPTION	EXCEPTION RESPONSE
Address of first status to be read is outside the range 0000 - 0005	Code 02 (Only coil addresses 0000 to 0005 contain defined information)
Number of locations to be read is outside the range 1 - 5	

Following 'READ INPUT STATUS' queries

EXCEPTION	EXCEPTION RESPONSE
Address of first location to be read is outside the range 0000 - 1311	Code 02 (Only status flag addresses 0000 to 1311 for IEEE, 0000 to 0767 for non-IEEE contain defined information)
Number of locations to be read is outside the range 1 - 512	
Configuration database is not yet confirmed	Code 04

Following 'READ HOLDING REGISTERS' query

EXCEPTION	EXCEPTION RESPONSE
Address of first register to be read is outside the range 0000 - 0383	Code 02 (Only register addresses 0000 to 0383 contain defined information)
Number of registers to be read is outside the range 1 - 60	

Following 'READ INPUT REGISTERS' query

EXCEPTION	EXCEPTION RESPONSE
Address of first register to be read is outside the range 0000 - 0081	Code 02 (Only register addresses 0000 to 0081 contain defined information)
Number of registers to be read is outside the range 1 - 60	
Configuration database is not yet confirmed	Code 04

Following 'FORCE SINGLE COIL' queries

EXCEPTION	EXCEPTION RESPONSE
Address of coil to be forced is outside the range 0000 - 0005	Code 02 (only addresses 0000 to 0005 are defined)
Data value is neither FF00 hex ('1') nor 0000 hex ('0')	Code 03
Coil that was to be forced is 'write disabled'	Code 01

Following 'PRESET SINGLE REGISTER' queries

EXCEPTION	EXCEPTION RESPONSE
Address of register to be preset is outside the range 0000 - 0383	Code 02 (only addresses 0000 to 0383 are defined)
Data value is outside the range 0 - 65535	Code 03
Register that was to be preset is 'write disabled'	Code 01

Following 'READ EXCEPTION STATUS' queries

No exception responses can be generated by the MTL838B-MBF on correctly receiving a READ EXCEPTION STATUS query.

Following 'DIAGNOSTICS' queries

EXCEPTION	EXCEPTION RESPONSE
Diagnostic code not supported by the MTL838B-MBF	Code 03

Following 'PRESET MULTIPLE REGISTERS' queries

EXCEPTION	EXCEPTION RESPONSE
Address of first register to be preset is outside the range 0000 - 0383	Code 02 (only addresses 0000 to 0383 are defined)
Number of registers to be preset is outside the range 1 to 60	
Number of data bytes is outside the range 2 to 120	
Register data values are outside the range 0 to 65535	Code 03
Register that was to be forced is 'write disabled'	Code 01

Note: within the limits of the 'address of register to be preset' given above, it is possible for the MTL838B-MBF to accept a query that requires an undefined register to be preset. The MTL838B-MBF will accept the query and issue a confirming response, but it will not modify any registers.

Following queries not supported by the MTL838B-MBF

EXCEPTION	EXCEPTION RESPONSE
Function code is not supported by the MTL838B-MBF	Code 01

The 'broadcast' function is not supported by the MTL838B-MBF. The unit does not decode messages issued with the broadcast slave address '0', so that it does not subsequently issue an exception response.

SCALING

The inputs that are received by the MTL831B and MTL832EXE transmitters are processed by the MTL838B-MBF according to the type of input, and depending on the scaling parameters that have been selected by the user. The processing and scaling of input data is discussed here in detail. In practice, most users have a standard data format and standard zero and FSD values for their control system. This is easily accommodated using the PCS83 configuration software.

The self-calibration that is carried out by the multiplexer system and the timing of responses to requests and the speed of response of the overall system is also covered.

Background to scaling input data

This section describes the fundamentals of scaling input data that will need to be understood by those configuring the MTL838B-MBF via Modbus. If configuration via PCS83 is to be used, this section need not be understood in detail. The following section on practical calculations will be of more relevance to a PCS83 configurator.

In general terms, each input to the MTL838B-MBF will have an output given by:

output = gain x (input - input zero) + output zero

Where:

output:	is a digital value.
gain:	is a value provides the required output range for the specified input range. This must be calculated by the user and written to the unit.
input:	is the value of the field input (mV, mA, etc.)
input zero:	is the lowest value that the input will be expected to record.
output zero:	is the output value that corresponds to the lowest input value.

Note: It is important to understand the limitations of the output range that is defined by the data format that has been chosen. With IEEE format there is little need for concern because of the enormous range that is available ($>\pm10^{38}$) but with non-IEEE data formats the output zero and fsd values should be chosen to give the maximum resolution. As mentioned earlier, many users have site standards for zero and fsd values that have been selected for optimum performance.

In practice, the above equation must be modified. This is because the MTL838B-MBF must be able to represent negative numbers when using **unsigned** data formats. This means that the whole scale must be lifted, using an ‘offset’, to allow negative values to be represented by a ‘positive’ value. Specifically then, the equation must be:

output = GAIN_n x (input - IPZERO_n) + OPZERO_n

Where **IPZERO_n = input zero + offset**

All the parameters in upper case represent values written to the Holding Registers for each input number ‘n’

The level of offset varies according to the type of input selected:

INPUT TYPE	DATAFORMAT	OFFSET VALUE
All	signed	0
CJ temperature (0.1)	unsigned non-IEEE	40°C
mV	unsigned	100mV
Temperature	unsigned	500°C

Calculation of scaling parameters - in practice

Scaling parameters must be calculated for each input to the multiplexer system. This can be done by first completing a table as shown below:

ZERO VALUES	FULL SCALE VALUES	RANGES	GAIN (GAIN_n)
input zero	input FSD	(input FSD) - (input zero)	(output range) ÷ (input range)
OPZERO_n	output FSD	(output FSD) - (OPZERO_n)	

The table gives parameters that need to be entered in to PCS83. Those wishing to configure via Modbus, however, must follow the further working below.

The OPZERO_n and GAIN_n values are identical to the values written to the MTL838B-MBF. However, the 'input zero' value will need to be modified as described in the last section:

IPZERO_n = input zero + offset

As an example, consider an application using a thermocouple to measure temperature in the range -10°C to +40°C. The output data format will be unsigned 3 decade BCD, with a (decimal) range of 100 to 600. These values can be put in the table and the gain calculated:

ZERO VALUE	FULL SCALE VALUE	RANGE:	GAIN:
-10°C	+40°C	+40°C – –10°C = 50°C	500 / 50 =10
100	600	600 - 100 = 500	

The data format selected will require an offset of 500^{O}C and based on this therefore, the values written to the MTL838B-MBF would be:

IP_ZERO :	490 (found from -10°C + 500°C = 490°C)
OP_ZERO:	100
GAIN:	10

As an example, the output can be calculated for an input of +40°C?

output = GAIN x (input + offset - IPZERO) + OPZERO

= 10 x (40 + 500 - 490) + 100 = 500 + 100 = 600

Self calibration of input measurements

When using MTL831B transmitters, the MTL838B-MBF requests calibration data from the transmitter at least once every 8 seconds. The transmitted data allows the MTL838B-MBF to re-calibrate the inputs it receives and modify the outputs accordingly. This re-calibration is carried out input by input, up to the maximum of 32 inputs, with each input being calibrated in its own right.

Sensor Input Processing

The inputs to the transmitters are measured and processed in a number of different ways according to the type of input selected and a number of other factors. The processing of each input type is discussed in detail in the sections below:

Thermocouple inputs

The message received from the MTL831B is decoded to a mV measurement for each thermocouple input. Each measurement is then corrected according to the latest figures for calibration for that input.

If CJ compensation is selected, the mV measurement value is further corrected according to the CJ temperature of the associated transmitter.

Linearisation and conversion to a temperature reading is carried out by comparing the corrected mV value with the linearisation tables that are stored within the MTL838B-MBF. The result is a temperature measurement expressed in Kelvin, degrees Fahrenheit or degrees Centigrade according to the units selected.

The temperature value is converted to the required output according to the equation below, and depending on the scaling parameters selected:

output = GAIN_n x (temperature + offset - IPZERO_n) + OPZERO_n

The output is expressed in the required data format and is written to the input registers for 'INPUT_n', from where it may be read by the Modbus host.

Resistance inputs

The message received from the MTL831B is decoded to a mV measurement for each resistance input. Each measurement is then corrected according to the latest figures for calibration for that input.

The measurement for input 16 of each MTL831B is the mV signal measured across a precision 100Ω resistor, and this figure is used to establish the resistance seen across all the other inputs (which have the same current signal passing through them).

The mΩ value for each input is converted to the required output according to the equation below, and depending on the scaling parameters selected:

output = GAIN_n x (mΩ resistance + offset - IPZERO_n) + OPZERO_n

The output is expressed in the required data format and is written to the input registers for 'INPUT_n', from where it may be read by the Modbus host.

RTD inputs

The message received from the MTL831B is decoded to a mV measurement for each RTD input. Each measurement is then corrected according to the latest figures for calibration for that input.

The measurement for input 16 of each MTL831B is the mV signal measured across a precision 100Ω resistor, and this figure is used to establish the resistance seen across all the other inputs (which have the same current signal passing through them).

Linearisation and conversion to a temperature reading is carried out by comparing the corrected mΩ value with the linearisation tables that are stored within the MTL838B-MBF. The result is a temperature measurement expressed in Kelvin, degrees Fahrenheit or degrees Centigrade according to the units selected.

The temperature value is converted to the required output according to the equation below, and depending on the scaling parameters selected:

output = GAIN_n x (temperature + offset - IPZERO_n) + OPZERO_n

The output is expressed in the required data format and is written to the input registers for 'INPUT_n', from where it may be read by the Modbus host.

mV inputs

The message received from the MTL831B is decoded to a mV measurement for each resistance input. Each measurement is then corrected according to the latest figures for calibration for that input.

The mV value for each input is converted to the required output according to the equation below, and depending on the scaling parameters selected:

output = GAIN_n x (mV input + offset - IPZERO_n) + OPZERO_n

The output is expressed in the required data format and is written to the input registers for 'INPUT_n', from where it may be read by the Modbus host.

Data timing

The MTL838B-MBF requires approximately 1 second to scan the inputs of one MTL831B, so scanning two MTL831Bs will require around 2 seconds.

The delay between receiving a Modbus request and issuing a response will not exceed 15ms or 3 character periods, whichever is the longer.

The overall response time is largely dependent on the choice of baud rate and communication mode. The time taken to transmit each query and each response can be easily calculated by multiplying the chosen baud rate by the number of bits that are transmitted in each message.

Configuring with PCS83 software

The following describes the use of the PCS83 software package. It describes each of the options presented on screen and leads a user through the possible configurations of the MTL838B-MBF.

The software package runs on an IBM®, or compatible, PC and provides configuration, monitoring and debugging facilities for the MTL830 system using a simple menu system.

Computer system requirements

The following are the minimum requirements necessary to utilise all the facilities of the software:-

i) IBM ® PC, or 100% compatible

ii) Minimum 320k RAM

iii) DOS 2.0 or higher (excluding 2.1.1)

iv) Serial port (COM 1 or COM 2)*

v) CGA, EGA or VGA display

* The serial port is required for communicating with the MTL830 range receivers. The software can also be run on a standalone PC where a configuration may be defined and saved to disk.

The user is advised to run the software directly from DOS.

It can be run under Microsoft Windows™, but only if the DOS window is run ‘Full Screen’, and given exclusive use of the system, with no priority at all given to background tasks. Because of the specialised aspects of setting up the software to run in this way, the user is advised NOT to run the software in a DOS ‘window’.

Installing software

The software is not copy protected and it is recommended that the user copies the files to a ‘working disk’ before installation; the original disk should then be stored in a safe location.

Installing to a Hard Disk

Install the files from the working disk by copying them to a newly created subdirectory called MTL on the hard drive of the computer. For example:

To create a subdirectory called MTL on the C: drive, type:

```
MD C:\MTL <enter>
```

To copy the files from the A: drive to the C: drive, type:

```
COPY A:\*.* C:\MTL <enter>
```

where **`<enter>`** means ‘press the enter (or return) key to complete your instruction’. If the computer does not have a hard disk available, see the next topic.

Installing to a Floppy Disk

MSDOS ® should be installed on your floppy disk before copying the files of PCS83. To make a 'system disk', i.e. one that has the operating system on it, format the floppy disk using the '**/s**' switch. For example, to format a disk in the A: drive that also has the system files, type:

```
        FORMAT A: /S <enter>
```

The program files should now be copied onto the floppy disk.

Before you start . . .

In order to establish communications between the PC and the MTL838B-MBF an RS232C serial data link is required. Cable connections for this link depend upon the type of serial port connector available on the computer in use - the two most common options are shown in figures 16 & 17. Before using this link, levers 7 and 8 of DIL Switch 101 must be set in the OFF position (see page 73), a temporary wire link must be fitted between the MODE and COM terminals (4 & 5) and the unit powered down and up again. This process disables the MTL838B-MBF as a Modbus slave while configuration takes place.

When configuration is complete, the wire link should be removed and the "Signoff" command executed, this will reset the unit. Lever 7 of switch 101 can now be moved to the ON position to disable any further off-line configuration.

Serial link cable connections

The following diagrams illustrate the cable connections required between the MTL838B-MBF-MBF multiplexer receiver and the serial port of the PC. If the PC has a 25-pin D-type connector, then use Figure 15. If the PC has a 9-pin D-type connector, use the wiring shown in Figure 16 (but note the reversal of Tx & Rx pins!).

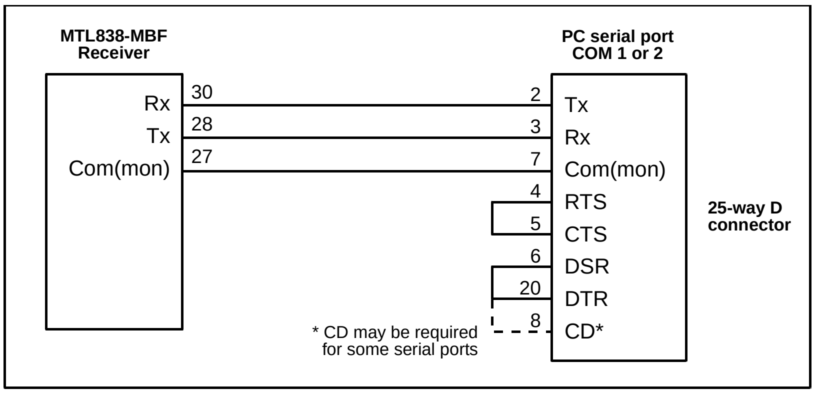


Figure 15 - 25-pin D-type connections

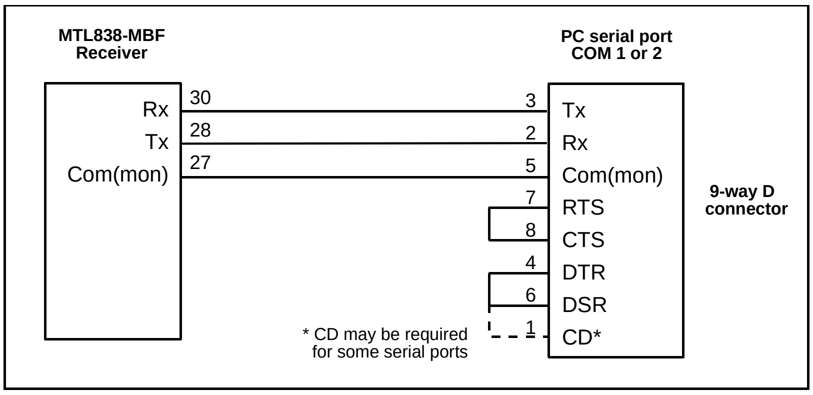


Figure 16 - 9-pin D-type connections

Troubleshooting

The software uses one of the computer's serial communications ports and any program that might interfere with the operation of this port should be disabled. Certain memory resident programs (TSRs) can affect the port, and any that *do* should be disabled. For instance, on some **laptop computers** a particular power management TSR called POWER.EXE is *known* to interfere with the PCS83 software.

Such TSR programs are often loaded automatically, during startup, by the CONFIG.SYS and/or the AUTOEXEC.BAT file. To prevent the loading of a specific TSR, identify the line that loads it in the CONFIG.SYS or the AUTOEXEC.BAT file and insert the letters REM, followed by a space, at the beginning of the line.

Note: This action may need to be reversed later if the TSRs are required for other software operation.

IF CHANGES ARE MADE TO THE CONFIG.SYS OR AUTOEXEC.BAT FILES THEN THE COMPUTER MUST BE REBOOTED FOR THE CHANGES TO TAKE EFFECT.

Receiver DIL switch settings

Figure 17 shows the locations of the DIL switches inside the MTL838B-MBF. The user must ensure that the following switch settings are in force before attempting to communicate with the receiver from the PC.

Switch 101	Levers 1 - 5	Set 'slave' address of Rx.
	Lever 7	Set OFF
	Lever 8	Set OFF (always left in this position)

For full details of all DIL switch settings see the section "Hardware configuration of the MTL838B-MBF" on page 39.

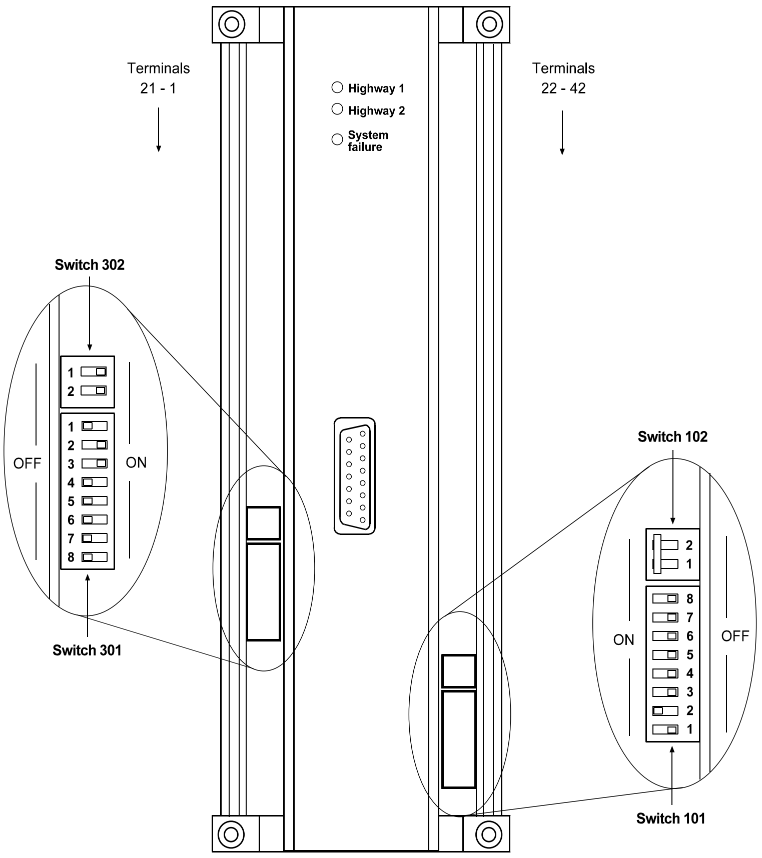


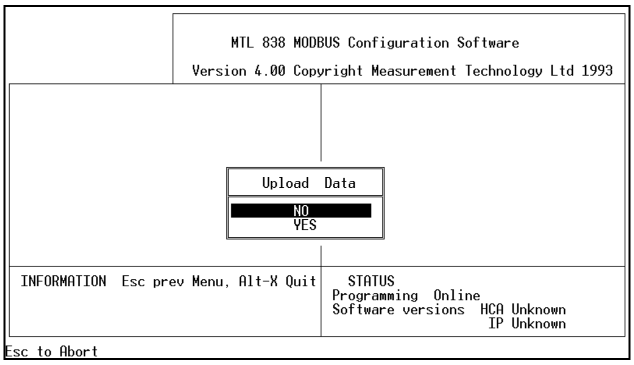
Figure 17 - DIL Switch locations

Using the program

The program expects all of the files it uses to be in the current working directory, so the following command sequence is required when running the program :-

```
C:  <enter>
CD\MTL <enter>
MTLMOD <enter>        (for a colour monitor)  OR
MTLMOD BW <enter>     (for a monochrome monitor)
```

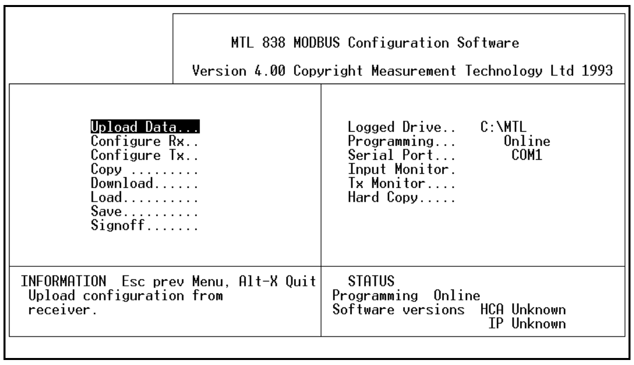
On start-up, the program asks if data needs to be uploaded from the database inside the MTL838B-MBF:



```
                    MTL 838 MODBUS Configuration Software
                  Version 4.00 Copyright Measurement Technology Ltd 1993

                              Upload  Data
                                  NO
                                  YES

INFORMATION  Esc prev Menu, Alt-X Quit    STATUS
                                        Programming  Online
                                        Software versions  HCA Unknown
                                                            IP Unknown
Esc to Abort
```

Figure 18 - Startup screen

For the purposes of this example 'NO' will be selected; the configuration screen will then appear, Figure 19.



```
                    MTL 838 MODBUS Configuration Software
                  Version 4.00 Copyright Measurement Technology Ltd 1993

          Upload Data...              Logged Drive..   C:\MTL
          Configure Rx..              Programming...      Online
          Configure Tx..              Serial Port...       COM1
          Copy .........              Input Monitor.
          Download......              Tx Monitor....
          Load..........              Hard Copy.....
          Save..........
          Signoff.......

INFORMATION  Esc prev Menu, Alt-X Quit    STATUS
 Upload configuration from              Programming  Online
 receiver.                              Software versions  HCA Unknown
                                                            IP Unknown
```

Figure 19 - PCS83 - Main Screen

The main section of the screen contains fourteen options from which the user may choose. To choose one, move the highlight bar using the arrow (cursor) keys, then select it by pressing <enter>, or <return>. The bottom left-hand corner of the screen contains help information about the option that has been highlighted.

Each of the options will lead on to another similar screen, or will generate a pop-up window containing further options. Use the <Esc> key to go back to the previous menu, and then to reach the opening screen. To quit the program, hold down the <Alt> key and then press the <X> key. You will be asked, at the bottom left of the screen, to confirm this choice.

The status window in the bottom right-hand corner of the screen gives information on the configuration of the serial port, the programming status and the software version of the multiplexer receiver.

If the programming status is "On-line" then most changes made during the configuration are sent immediately to the multiplexer. This function can be disabled by setting the programming to "Off-line" (see option 10), which allows configurations to be typed in and saved to disk without affecting the current settings of the multiplexer. This latter option is useful if setting up a configuration without an MTL838B-MBF connected.

If the program detects any errors during its operation, or is unable to carry out a command it will provide the user with an error message. The following list shows the possible messages and what they mean:

RS232 TIMEOUT	No response from the receiver
CRC ERROR	Corrupted message
DATA ERROR	Computer unable to decode message (e.g., incompatible software versions)
TX FAILED	Receiver has lost contact with transmitter
INVALID FILENAME	Unable to find file on disk
PROTOCOL ERROR	Message format incorrect or fields too long
WRONG PARAMETERS	Wrong number of parameters
TX/CH INVALID	Transmitter or channel number out of range
PARAM OVERRANGE	Other parameter out of range
CONTEXT ERROR	Illegal combination of options chosen
UNKNOWN ERROR	Transmitter number greater than that programmed

The convention adopted here is that those screens, or part screens, shown in white, are the currently active screens. The grey highlight bar on these active screens indicates the current menu choice, and the screen order reads from left to right.

The opening screen, Figure 19, shows the 14 principle options that are available when using the PCS83 program. Each of these options and their subsidiaries will be described in the order that they appear on screen.

***Note :** The following examples show data for a typical installation. When a system is configured for the first time, the program will show the system default values.*

Option 1 : Upload Data

This uploads the configuration database of the receiver, via the serial port, into the host computers memory. If this option is selected, the program will attempt to obtain the configuration from the battery-backed RAM memory within the MTL838B-MBF. This is illustrated in Figure 20 where the versions of the software start off as “Unknown” and the progress of the upload is shown in the bottom left corner of the screen.

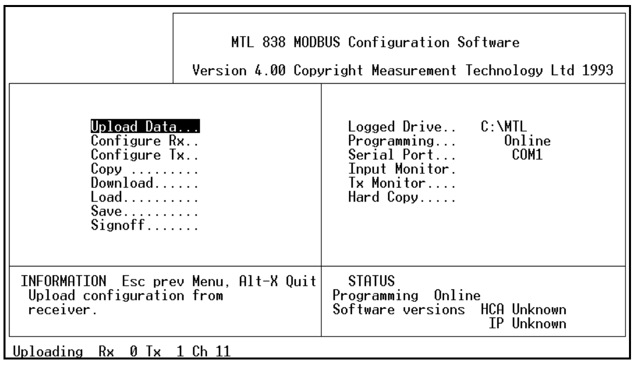
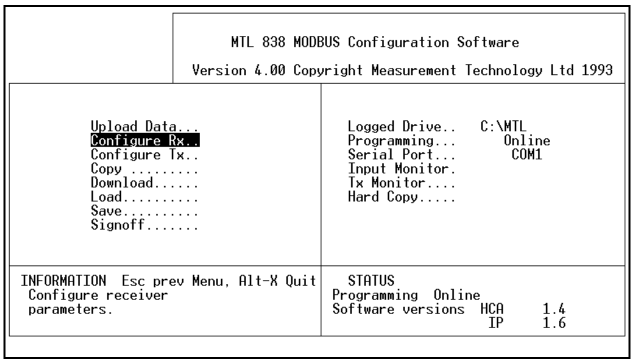

```
              MTL 838 MODBUS Configuration Software

      Version 4.00 Copyright Measurement Technology Ltd 1993

        Upload Data...           Logged Drive..   C:\MTL
        Configure Rx..           Programming...      Online
        Configure Tx..           Serial Port...       COM1
        Copy .........           Input Monitor.
        Download......           Tx Monitor....
        Load..........           Hard Copy.....
        Save..........
        Signoff.......

INFORMATION  Esc prev Menu, Alt-X Quit     STATUS
 Upload configuration from              Programming  Online
 receiver.                              Software versions  HCA Unknown
                                                            IP Unknown
Uploading  Rx  0 Tx  1 Ch 11
```

Figure 20 - ‘unknown’ software versions

In Figure 21, after the upload, the software version numbers can be seen to have been read for the HCA (Highway Comms Adapter) - version '1.4' - and, similarly, for the IP (Input Processor) - version '1.6'.



```
            MTL 838 MODBUS Configuration Software
  Version 4.00 Copyright Measurement Technology Ltd 1993

Upload Data...                  Logged Drive..   C:\MTL
Configure Rx..                  Programming...      Online
Configure Tx..                  Serial Port...      COM1
Copy .........                  Input Monitor.
Download......                  Tx Monitor....
Load..........                  Hard Copy.....
Save..........
Signoff.......

INFORMATION  Esc prev Menu, Alt-X Quit    STATUS
 Configure receiver                      Programming  Online
 parameters.                             Software versions  HCA     1.4
                                                            IP      1.6
```

Figure 21 - 'known' software versions

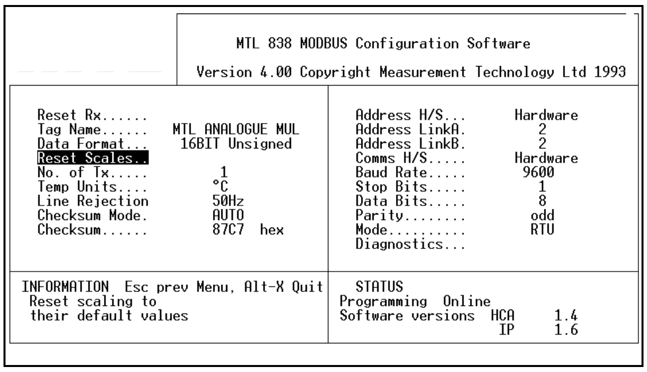
If this process fails, check the hardware, i.e.

- that the correct serial port is being used (COM 1 or COM 2),
- that the RS232C cable link is in place and
- that it is correctly wired to the D-connector.

Then retry the upload data command. If the upload still does not succeed refer to the section called Troubleshooting (page 73).

Option 2 : Configure Rx

Allows the configuration of the receiver via a further set of menu options which appear on selection of this choice. The screen is shown in Figure 22.



```
            MTL 838 MODBUS Configuration Software
  Version 4.00 Copyright Measurement Technology Ltd 1993

Reset Rx......                        Address H/S...     Hardware
Tag Name......  MTL ANALOGUE MUL      Address LinkA.        2
Data Format...   16BIT Unsigned       Address LinkB.        2
Reset Scales..                        Comms H/S.....     Hardware
No. of Tx.....        1               Baud Rate.....      9600
Temp Units....       °C               Stop Bits.....        1
Line Rejection       50Hz             Data Bits.....        8
Checksum Mode.       AUTO             Parity........       odd
Checksum......       87C7  hex        Mode..........       RTU
                                      Diagnostics...

INFORMATION  Esc prev Menu, Alt-X Quit    STATUS
 Reset scaling to                        Programming  Online
 their default values                    Software versions  HCA     1.4
                                                            IP      1.6
```

Figure 22 - Configure Rx screen

If the configuration was uploaded from the receiver, or loaded from a configuration file, then the screen displays the current settings. If the receiver has been reset then the values will show the default settings appropriate to the MTL831B or MTL832, whichever was chosen.

The options allow tailoring of the receiver’s configuration. Each selection brings into the screen a pop-up window in which:

1. a message appears, or
2. a choice should be made, or
3. a value entered.

A description of the functions offered is given next.

Reset Rx

This overwrites all the receiver settings with the default values for either:

- MTL831B or
- MTL832.

In addition, it sets all the receiver to **mV** and makes the dataformat **IEEE**.

Tag Name

This option permits the user to enter a unique name for a receiver, or its configuration/set-up. This is often used to uniquely identify a receiver within the overall site installation.

Data Format

This is used to set the required format for the scaling parameters and the data transferred via Modbus. For example, IEEE, Signed 16-bit binary, etc.

When a new selection has been made, the scaling parameters become invalid, and the 'Reset Scales' option must be used in order to start from a known default condition.

Reset Scales

This sets the default configuration for an MTL831B or MTL832 but, unlike the Reset Rx command above, it does not change the data format.

Number of Transmitters

This option specifies the number of transmitters that are connected to the receiver. Up to two MTL831B transmitters may be connected to a single MTL838B-MBF. If MTL832's are used, it permits up to four.

Temperature Units

The user can choose between degrees Centigrade, Fahrenheit or Kelvin.

Line Rejection

This is used to select the rejection frequency for the a.c. power in use - 50Hz or 60Hz.

Checksum Mode

The options are Auto or Manual. If Auto is selected the MTL838B-MBF will maintain the correct checksum despite changes being made to the configuration. If 'Manual' is selected the checksum must be entered by the Modbus Master. (This selection is normally set to 'Auto').

Checksum

The user may choose to have the checksum displayed in decimal or hexadecimal format.

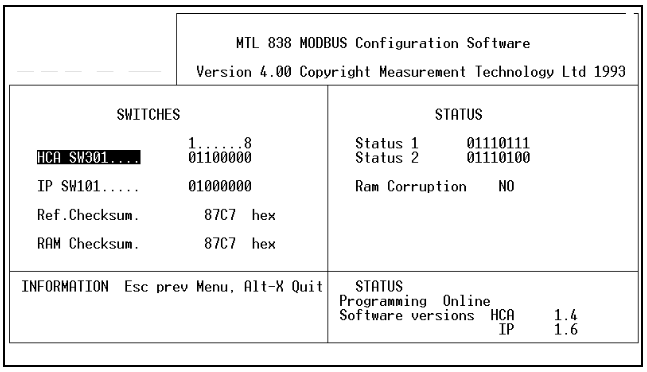
Address and Communications Parameters

The items on the right-hand side of the screen, from **Address H/S** through to **Mode,** can reflect hardware or software settings, which is indicated accordingly. If hardware set, then the parameters may not be accessed or modified using the software.

Diagnostics

The **Diagnostics** option, however, provides information on each of the DIL switches function, see

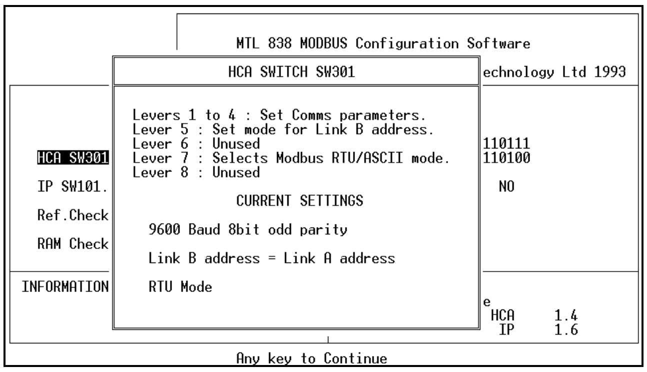
Figure 23. This screen also displays the database checksum values and alerts the user to RAM corruption. If corruption has occurred the database will require reconfiguration to clear the exception responses to input data requests.



```
              MTL 838 MODBUS Configuration Software
            Version 4.00 Copyright Measurement Technology Ltd 1993

        SWITCHES                                STATUS

                   1......8            Status 1      01110111
   HCA SW301....   01100000            Status 2      01110100

   IP SW101.....   01000000            Ram Corruption    NO

   Ref.Checksum.     87C7  hex

   RAM Checksum.     87C7  hex

INFORMATION  Esc prev Menu, Alt-X Quit    STATUS
                                       Programming  Online
                                       Software versions  HCA    1.4
                                                          IP     1.6
```

Figure 23 - Diagnostics screen

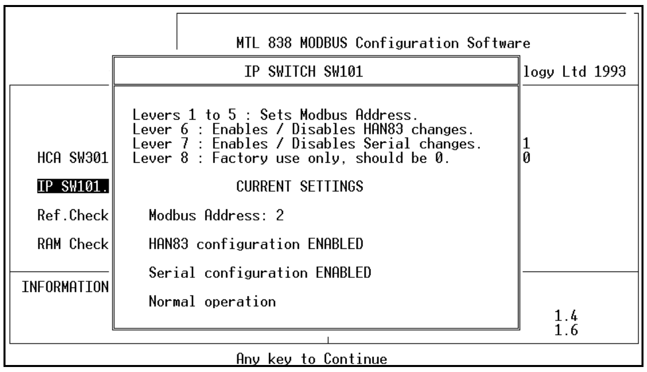
The first two options show the settings of the main DIL switches in the receiver. Press <Enter> for the first option to see the interpretation of the settings - Figure 24.



```
              MTL 838 MODBUS Configuration Software
               HCA SWITCH SW301                  echnology Ltd 1993

             Levers 1 to 4 : Set Comms parameters.
             Lever 5 : Set mode for Link B address.
             Lever 6 : Unused                         110111
   HCA SW301 Lever 7 : Selects Modbus RTU/ASCII mode. 110100
             Lever 8 : Unused
   IP SW101.             CURRENT SETTINGS               NO

   Ref.Check   9600 Baud 8bit odd parity

   RAM Check   Link B address = Link A address

INFORMATION    RTU Mode
                                                      e
                                                      HCA    1.4
                                                       IP    1.6
                        Any key to Continue
```

Figure 24 - HCA switch settings

Similarly, for the second option - the IP settings - the user can press <Enter> to obtain an interpretation of the switch settings.



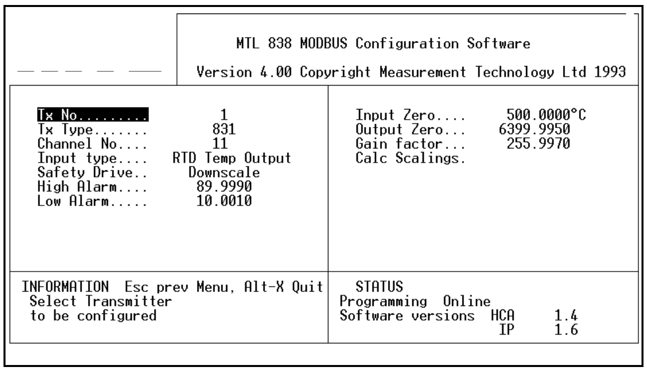
```
              MTL 838 MODBUS Configuration Software
                  IP SWITCH SW101                     logy Ltd 1993

             Levers 1 to 5 : Sets Modbus Address.
             Lever 6 : Enables / Disables HAN83 changes.
             Lever 7 : Enables / Disables Serial changes.    1
   HCA SW301 Lever 8 : Factory use only, should be 0.        0

   IP SW101.             CURRENT SETTINGS

   Ref.Check   Modbus Address: 2

   RAM Check   HAN83 configuration ENABLED

               Serial configuration ENABLED
INFORMATION
               Normal operation
                                                             1.4
                                                             1.6
                        Any key to Continue
```

Figure 25 - IP switch settings

Further information on these switch settings can be found on page 39.

Option 3: Configure Tx

This option is used to configure, or view, individual transmitters. Figure 26 shows the screen after choosing this option.



```
        MTL 838 MODBUS Configuration Software

  Version 4.00 Copyright Measurement Technology Ltd 1993

 Tx No.........        1            Input Zero....     500.0000°C
 Tx Type.......       831           Output Zero...    6399.9950
 Channel No....       11            Gain factor...     255.9970
 Input type....  RTD Temp Output    Calc Scalings.
 Safety Drive..    Downscale
 High Alarm....     89.9990
 Low Alarm.....     10.0010

INFORMATION  Esc prev Menu, Alt-X Quit     STATUS
 Select Transmitter                      Programming  Online
 to be configured                        Software versions  HCA     1.4
                                                             IP     1.6
```

Figure 26 - Configure Tx screen

Tx Number

Press <Enter> and choose from the list of transmitters that have been identified. The DIL switches in the transmitters must have been set correctly for them to be identified by the receiver.

Up to 32 inputs may be monitored on a single MTL838B-MBF; this could be four MTL832's,

Tx Type

This is used to define the type of transmitter to be used i.e. MTL831B

Channel Number

Select the number of the channel that is being defined.

Channel Type

Choose from the list of permissible input types. As there are more choices than can be shown on screen at one time, move the highlight to the "--more--" option and press <Enter> for further choices.

The "_ THC mV+CJ" types provide a mV input that is unlinearised, but has Cold Junction compensation.

Safety Drive

This allows the user to define the action of the channel if a fault occurs. The choices are Upscale, Downscale or Off. Upscale makes the input measurement ramp up to its maximum permissible value. Downscale makes the input measurement ramp down to its minimum value. Choosing Off means that no action will be taken. The purpose behind this is to trip an alarm point which will alert an operator to the fault condition. The maximum and minimum values differ with the input type selected:

INPUT TYPE	DOWNSCALE LIMIT	UPSCALE LIMIT
Thermocouple	-120mV	+120mV
Resistance	0Ω	1200Ω
RTD Temperature	Temperature equiv. to 0Ω	Temperature equiv. to 1200Ω
Millivolt	-120mV	+120mV

High and Low Alarm

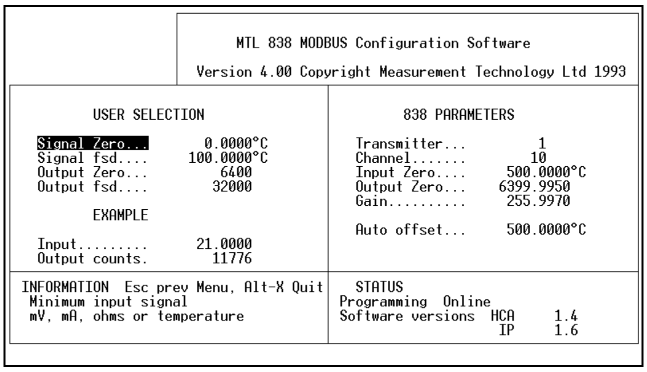
These allow the user to set thresholds for a high or low alarm condition. The default settings are equivalent to input values of ±60mV.

Input Zero, Output Zero and Gain factor

Values for these parameters may be entered directly but the easier route is using the next option - Calculate Scalings.

Calculate Scalings

Press <Enter> to see a screen similar to the following:



```
                MTL 838 MODBUS Configuration Software
              Version 4.00 Copyright Measurement Technology Ltd 1993

         USER SELECTION                        838 PARAMETERS

   Signal Zero...        0.0000°C      Transmitter...        1
   Signal fsd....      100.0000°C      Channel.......       10
   Output Zero...          6400        Input Zero....     500.0000°C
   Output fsd....         32000        Output Zero...    6399.9950
                                       Gain..........     255.9970
         EXAMPLE
                                       Auto offset...     500.0000°C
   Input.........       21.0000
   Output counts.         11776

 INFORMATION  Esc prev Menu, Alt-X Quit    STATUS
  Minimum input signal                   Programming  Online
  mV, mA, ohms or temperature            Software versions  HCA     1.4
                                                            IP      1.6
```

Figure 27 - Calculate Scalings screen

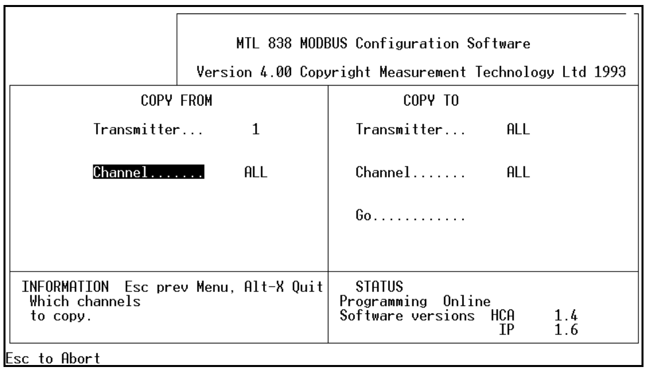
This enables the user to enter a input signal range and a receiver output range for the sensor being used. These ranges are specified on the left-hand side of the screen under User Selection.

The right-hand side shows the parameters that are calculated by the software for the MTL838B-MBF-MBF. As may be seen in the example above, the Input Zero may differ from the Signal Zero because of an offset, which will depend upon the type of input sensor chosen. The offset value is a read only parameter and is automatically applied by the software. The Example facility calculates the output for an input value that a user may enter.

If the input zero, output zero and gain factor are already known then these can be entered in the previous screen without having to go into **Calc Scalings**.

Option 4: Copy

This option will copy configurations from one transmitter to one or all others. Similarly, one or all channel configurations may be copied to one or all other locations. The screen is very simple and is shown in Figure 28.



```
                MTL 838 MODBUS Configuration Software
              Version 4.00 Copyright Measurement Technology Ltd 1993
              COPY FROM                           COPY TO

         Transmitter...      1          Transmitter...     ALL

         Channel.......     ALL         Channel.......     ALL

                                        Go............

 INFORMATION  Esc prev Menu, Alt-X Quit    STATUS
  Which channels                         Programming  Online
  to copy.                               Software versions  HCA     1.4
                                                            IP      1.6
Esc to Abort
```

Figure 28 - 'Copy' screen

This function saves a lot of time when configuring large systems.

From this screen a user may choose the transmitter(s) and the channel(s) which are to be copied, then choose the transmitter(s) and the channel(s) to which the configuration is to be sent. When the selections have been made select the Go option and press <Enter>.

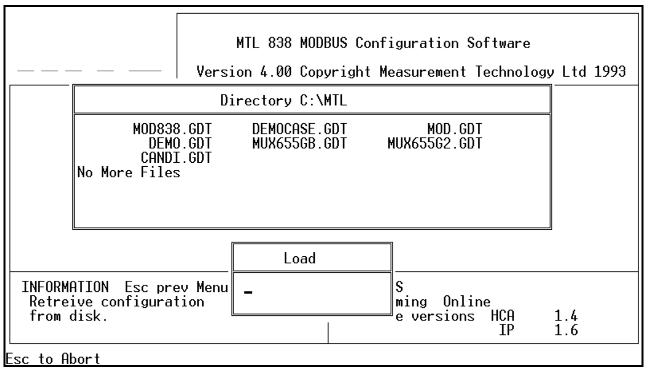
If a user chooses to copy **All** channels from the transmitter on the left, the Channel option on the right changes to All automatically.

Option 5: Download

This downloads the configuration data in the host computers memory to all of the receivers connected to the serial port. Each receiver, transmitter and channel is chosen in numerical order and the data downloaded.

Option 6: Load

This enables the user to load a configuration from a computer file. The file is loaded from the drive selected by the Logged Drive parameter, shown on the main screen. If a receiver configuration is already in the computer's memory it will be overwritten by the loaded file. On choosing this option the screen will look like the following:



```
                    MTL 838 MODBUS Configuration Software
                Version 4.00 Copyright Measurement Technology Ltd 1993
                    Directory C:\MTL
        MOD838.GDT     DEMOCASE.GDT          MOD.GDT
          DEMO.GDT     MUX655GB.GDT     MUX655G2.GDT
         CANDI.GDT
No More Files
                            Load
INFORMATION  Esc prev Menu  _          S
 Retreive configuration               ming  Online
 from disk.                           e versions  HCA     1.4
                                                  IP      1.6
Esc to Abort
```

Figure 29 - 'Load' screen

Figure 29 shows an example of the files available in the C:\MTL subdirectory. Type the name of the file to load - the file extension (.GDT) is not required - then press <Enter>. When the file has been selected, PCS83 will offer the choice of downloading it to the receivers via the serial link (**Yes** option) or just storing it in the PC memory (**No** option). Press <Esc> to cancel the option and do neither.

Option 7: Save

Saves the configuration currently in PC memory to disk. It will be saved on the drive selected by Logged Drive. Data is saved only for the receivers that have been configured, which makes it possible to merge files containing configurations for different receivers. Type a name for the file you wish to save to and then press <Enter>. Press <Esc> to cancel the option.

Caution: If you choose a file name that is the same as one already saved the earlier one will be overwritten.

Option 8: Signoff

This is the final act following an On-line configuration. Signoff confirms the settings currently in the receiver and forces a power-up reset.

If the MTL838B-MBF is required to boot up in Modbus mode, the MODE - COM link must be removed before executing this command. The user is given a warning to this effect before the command is executed

Option 9: Logged Drive

This allows the user to specify the name of the drive/directory that will be used by the Load and Save commands. Type the name of the drive/directory, then press <Enter>. Press <Esc> to cancel the option.

Option 10: Programming

Selects whether changes are automatically sent to the receiver or not. If it is set to **On-line**, changes are sent automatically, if it is set to **Off-line** they are not. Select the desired mode then press <Enter>. Press <Esc> to cancel the option.

Option 11: Serial Port

Use this to select the PC serial port that is, or will be, connected to the multiplexer - assuming there is more than one available.

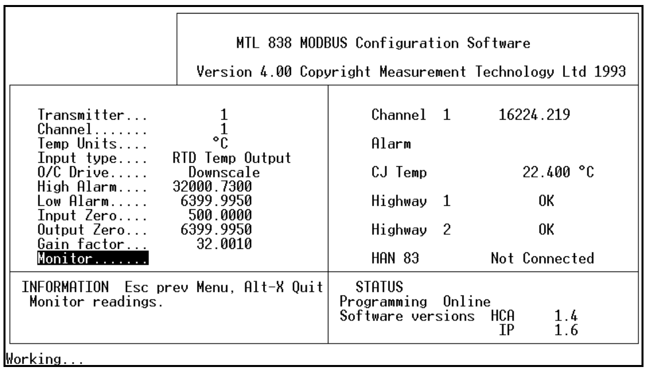
Select the desired serial port, then press <Enter>. Press <Esc> to cancel the option.

Option 12 : Input Monitor

This enables the configuration of one input to be viewed and/or modified, and the output value to be monitored in 'real time'. Each heading produces a pop-up menu, which offers the user a choice or the opportunity to enter a value.

When selected, the last option - **Monitor** - continually reads the specified input's value. This can be halted by pressing <F1> or <Esc>. <F1> allows the configuration to be changed and <Esc> returns the user to the opening screen. Figure 30 shows the current settings and values for channel 1.

Press <Esc> to return to the opening screen.



```
                     MTL 838 MODBUS Configuration Software

                 Version 4.00 Copyright Measurement Technology Ltd 1993

   Transmitter...          1               Channel  1      16224.219
   Channel.......          1
   Temp Units....         °C               Alarm
   Input type....   RTD Temp Output
   O/C Drive.....     Downscale            CJ Temp            22.400 °C
   High Alarm....   32000.7300
   Low Alarm.....    6399.9950             Highway  1           OK
   Input Zero....     500.0000
   Output Zero...    6399.9950             Highway  2           OK
   Gain factor...      32.0010
   Monitor.......                          HAN 83         Not Connected

 INFORMATION  Esc prev Menu, Alt-X Quit     STATUS
  Monitor readings.                       Programming  Online
                                          Software versions  HCA     1.4
                                                             IP      1.6
Working...
```

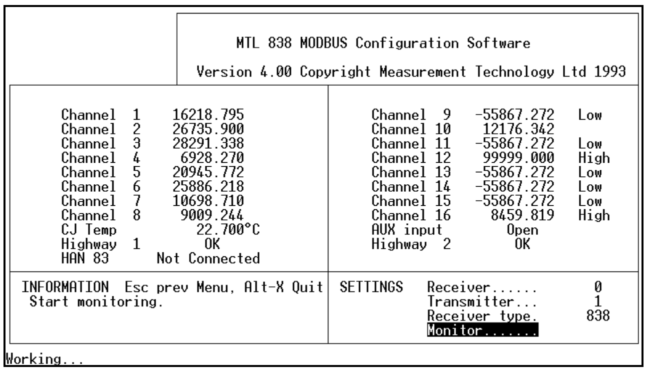
Figure 30 - Input Monitor screen

Option 13: Tx. Monitor

This allows ALL the inputs of a transmitter to be continually monitored. Transmitters connected to the system can be selected using the **Receiver**, **Transmitter** and **Receiver type** options at the bottom of the screen.

Select Monitor then press <Enter> to begin the monitoring. This is halted by pressing <Esc> and allows the selected transmitter to be changed. Pressing <Esc> again returns the user to the opening screen.

Figure 31 shows a number of channels being monitored.



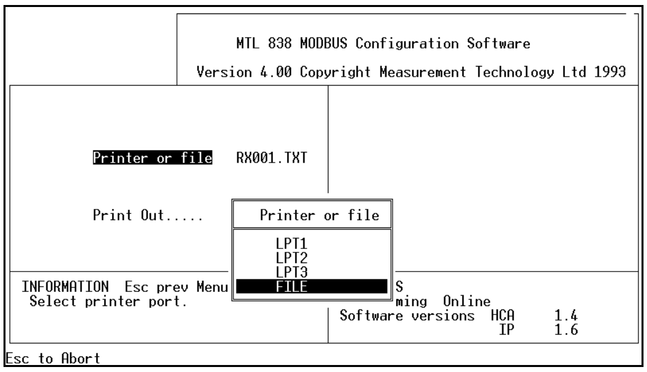
```
                    MTL 838 MODBUS Configuration Software
                 Version 4.00 Copyright Measurement Technology Ltd 1993

      Channel  1    16218.795              Channel  9   -55867.272   Low
      Channel  2    26735.900              Channel 10    12176.342
      Channel  3    28291.338              Channel 11   -55867.272   Low
      Channel  4     6928.270              Channel 12    99999.000   High
      Channel  5    20945.772              Channel 13   -55867.272   Low
      Channel  6    25886.218              Channel 14   -55867.272   Low
      Channel  7    10698.710              Channel 15   -55867.272   Low
      Channel  8     9009.244              Channel 16     8459.819   High
      CJ Temp           22.700°C           AUX input        Open
      Highway  1         OK                Highway  2        OK
      HAN 83      Not Connected

 INFORMATION  Esc prev Menu, Alt-X Quit  SETTINGS   Receiver......     0
  Start monitoring.                                 Transmitter...     1
                                                    Receiver type.    838
                                                    Monitor.......
Working...
```

Figure 31 - Tx Monitor screen

Channel 10 is a thermocouple input while channels 1 - 8 are RTDs. It also displays the CJ temperature, the status of the highways and whether, or not, the auxiliary input is connected.

Option 14: Hard Copy

This enables the user to direct the configuration information to a line printer. The user can select the receiver configuration and also the printer port, or file name, to which it will be sent - see Figure 32.



```
                    MTL 838 MODBUS Configuration Software
                 Version 4.00 Copyright Measurement Technology Ltd 1993


         Printer or file   RX001.TXT


         Print Out.....   Printer or file

                              LPT1
                              LPT2
                              LPT3
 INFORMATION  Esc prev Menu   FILE      S
  Select printer port.                  ming  Online
                                  Software versions  HCA     1.4
                                                      IP     1.6
Esc to Abort
```

Figure 32 - Hard Copy screen

If **File** is selected a file name is then required to be entered. The file will be placed in the “Logged drive” named on the main screen.

Press <Esc> to return to the Hard Copy screen, then select Print Out and press <Enter> to send the data to the printer or file.

A typical printout might look like the following (which continues on the next page):

```
                  Configuration Report MTL 838 Modbus Multiplexer

               ====================================================

        Tag Name: MTL ANALOGUE MULTIPLEXER                 Date: 3/6/1997

        Software Versions: HCA     1.4

                            IP     1.6

        Baudrate:     9600                    Data Format: 16BIT Unsigned

        Databits:      1                Temperature Units:  Centigrade

          Parity:     odd

        Stopbits:      1

  Line Rejection:    50Hz

     Transmitter:   1 of 1     Type: MTL 831
```

```
-----------------------------------------------------------------------------
 Ch No  Input Type          Alarms              Scaling Parameters
-----------------------------------------------------------------------------
 1  RTD Temp Output   High :32000.7300   I/P Zero:  500.0000   Gain:   32.0010
Sfty Drive    N/A      Low : 6399.9950   O/P Zero: 6399.9950

 2  RTD Temp Output   High :32000.7300   I/P Zero:  500.0000   Gain:   32.0010
Sfty Drive    N/A      Low : 6399.9950   O/P Zero: 6399.9950

 3  RTD Temp Output   High :32000.7300   I/P Zero:  500.0000   Gain:   32.0010
Sfty Drive    N/A      Low : 6399.9950   O/P Zero: 6399.9950

 4  RTD Temp Output   High :32000.7300   I/P Zero:  500.0000   Gain:   32.0010
Sfty Drive    N/A      Low : 6399.9950   O/P Zero: 6399.9950

 5  RTD Temp Output   High :32000.7300   I/P Zero:  500.0000   Gain:   32.0010
Sfty Drive    N/A      Low : 6399.9950   O/P Zero: 6399.9950

 6  RTD Temp Output   High :32000.7300   I/P Zero:  500.0000   Gain:   32.0010
Sfty Drive    N/A      Low : 6399.9950   O/P Zero: 6399.9950

 7  RTD Ohmic Output  High :32000.7300   I/P Zero:    0.0000   Gain:   21.3330
Sfty Drive    N/A      Low : 6399.9950   O/P Zero: 6399.9950

 8  RTD Ohmic Output  High :32000.7300   I/P Zero:    0.0000   Gain:   20.3290
Sfty Drive    N/A      Low : 6399.9950   O/P Zero: 6399.9950

 9  RTD Temp Output   High :   89.9990   I/P Zero:  500.0000   Gain:  255.9970
Sfty Drive    N/A      Low :   10.0010   O/P Zero: 6399.9950

10   K  THC with CJ   High :32000.7300   I/P Zero:  500.0000   Gain:  255.9970
Sfty Drive    N/A      Low : 6399.9950   O/P Zero: 6399.9950

11  RTD Temp Output   High :   89.9990   I/P Zero:  500.0000   Gain:  255.9970
Sfty Drive    N/A      Low :   10.0010   O/P Zero: 6399.9950

12  RTD Temp Output   High :   89.9990   I/P Zero:  500.0000   Gain:  255.9970
Sfty Drive    N/A      Low :   10.0010   O/P Zero: 6399.9950

13  RTD Temp Output   High :   89.9990   I/P Zero:  500.0000   Gain:  255.9970
Sfty Drive    N/A      Low :   10.0010   O/P Zero: 6399.9950

14  RTD Temp Output   High :   89.9990   I/P Zero:  500.0000   Gain:  255.9970
Sfty Drive    N/A      Low :   10.0010   O/P Zero: 6399.9950

15  RTD Temp Output   High :   89.9990   I/P Zero:  500.0000   Gain:  255.9970
Sfty Drive    N/A      Low :   10.0010   O/P Zero: 6399.9950

16    RTD Sense       High :     N/A     I/P Zero:     N/A     Gain:     N/A
Sfty Drive    N/A      Low :     N/A     O/P Zero:     N/A
```

Appendix A

ERROR CHECKING TECHNIQUES IN MODBUS

Calculation of the LRC in ASCII transmission mode

The Longitudinal Redundancy Check (LRC) used in the ASCII transmission mode is one byte long and contains an 8-bit binary value. It is calculated by the transmitting device and added to the message. It is also calculated by the receiving device, from the contents of the message, and then compared with the value contained in the message. If the two values are not equal, then an error has occurred in either the transmission or the reception of the message.

The LRC is calculated by adding together the successive 8-bit bytes of the message, discarding any carries and then performing a two's complement operation on the result.

When the sum of the bytes reaches 1111 1111, the addition of any subsequent bytes causes the data to reach a 9-bit value, which cannot be expressed in an 8-bit word. For the purpose of the LRC, the ninth bit is simply discarded so that, for example, the addition of the 8-bit byte 0000 1110, to the value above would give 0000 0111.

The two's complement operation is carried out by subtracting the final sum from 1111 1111, which would give 1111 1000 in our example, and then adding 1. This would give a value of 1111 1001 ('F9' in hex.) to be transmitted as the LRC.

The LRC is actually transmitted as two ASCII characters, equivalent of the LRC value obtained, so for our example, the actual characters transmitted would be '46 39'. The high order character being transmitted first, followed by the low order character.

Note: the start and stop characters that are used in ASCII transmission (':' and 'CR/LF') are excluded from the LRC calculation.

Calculation of the CRC in RTU transmission mode

The Cyclical Redundancy Check (CRC) used for RTU transmission mode is 16-bits long and is transmitted as two 8-bit bytes. It is calculated by the transmitting device and added to the message. It is also calculated by the receiving device, from the contents of the message, and then compared with the value contained in the message. If the two values are not equal, then an error has occurred in either the transmission or the reception of the message.

The CRC is calculated by carrying out a process of exclusive OR operations. The first byte is exclusive OR'ed with the value 1111 1111 1111 1111, and the result is used for any subsequent exclusive OR operations that are required. Only the eight bits of the message data are used for this operation. Start and stop bits and the parity bit - if one is selected - are not included in this operation.

Once the first byte has been exclusive OR'ed with the register contents, the least significant bit is removed, and the register contents are shifted towards the least significant bit position, with a zero placed in to the vacant position of the most significant bit.

The least significant bit which is extracted is examined and, depending on it's value, one of two operations is carried out. If the extracted LSB is a '1', the register is exclusive OR'ed with the hex. value 'A0 01'. If the LSB is a '0', the register contents are left as they are and shifted again. This shifting and extracting process is repeated until eight shifts have occurred.

The second data byte is exclusive OR'ed with the register contents and the process of shifting, examining and exclusive OR'ing is repeated another eight times. This process continues until all the data bytes have been through the operation. The resulting 16-bit CRC is transmitted as two 8-bit bytes, with the high order bytes transmitted first.

Appendix B

IEEE single precision data format

This appendix describes the encoding of IEEE single precision data. The table below shows the composition of the four 8-bit bytes required to describe a value in IEEE format. (Strictly, the data format is termed the IEEE754 single precision data format.) The most significant byte is transmitted first.

BYTE SIGNIFICANCE	**BIT NUMBER**							
	7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
most significant byte	s	e7	e6	e5	e4	e3	e2	e1
2nd most significant byte	e0	f22	f21	f20	f19	f18	f17	f16
3rd most significant byte	f15	f14	f13	f12	f11	f10	f9	f8
least significant byte	f7	f6	f5	f4	f3	f2	f1	f0

where:

s = sign bit

e = exponent

f = significand

The value to be encoded is given by the relevant entry in the table below:

e	**f**	**v**
0 < e < 255	all	$v = (-1)^{s} \times 2^{(e-127)} \times 1.f$
e = 0	f ¹ 0	$v = (-1)^{s} \times 2^{(e-126)} \times 0.f$
e = 0	f = 0	v = 0
e = 255	f = 0	$v = (-1)^{s} \times$ infinity
e = 255	f ¹ 0	v = non-allowed number

For example:

s = 0

e = 128

f = 5

$v = (-1)^{s} \times 2^{(e-127)} \times 1.f$

$= (-1)^{0} \times 2^{(128-127)} \times 1.5$

$= 1 \times 2 \times 1.5$

$= 3$

Appendix C

Non-IEEE data format

When non-IEEE data formats are used, some of the scaling parameter values used by the MTL838B-MBF must be expressed as a numerand and an exponent. This gives much greater flexibility to the values that may be used - especially when very large or very small numbers are required. This issue need only be considered if the user intends to configure the MTL838B-MBF via the Modbus host. If the PCS83 is used to configure the unit there is no need to consider the encoding of data in this way, as both configuration tools make these calculations automatically. The process becomes totally transparent to the user.

The parameters that must be expressed as numerand and exponent are GAIN_n, OPZERO_n, IPZERO_n, HA_n and LA_n.

Numerand and exponent

The exact process for expressing a value as a numerand and exponent will vary with the type of data format selected, but the overall principle behind the expression remains the same irrespective of the data format selected.

The value must be expressed in the general form:

$$V = n \times 10^{e}$$

where:

V = the value to be expressed

n = the 'numerand', with $-1 < n < 1$

e = the 'exponent', with $e < 6$

Expressing 'n' and 'e' with non-IEEE data formats

The equation for the calculation of 'n' and 'e' is modified slightly from that shown above, by the introduction of another factor:

$$V = (x / N) \times 10^{e}$$

where:

x = an integer value expressed in the chosen data format

N = the maximum value that can be expressed in the chosen data format

with:

$$|x / N| < 1$$

The value of N varies with the chosen data format as shown in the table below:

DATA FORMAT	EXPRESSION
unsigned 16-bit binary	$V = (x / 65535) \times 10^{e}$
other 16-bit formats	$V = (x / 32768) \times 10^{e}$
unsigned 12-bit binary	$V = (x / 4095) \times 10^{e}$
other 12-bit formats	$V = (x / 2048) \times 10^{e}$
unsigned 4-decade BCD	$V = (x / 9999) \times 10^{e}$
other 4-decade BCD formats	$V = (x / 5000) \times 10^{e}$
unsigned 3-decade BCD	$V = (x / 999) \times 10^{e}$
other 3-decade BCD	$V = (x / 500) \times 10^{e}$

Once the values of 'x' and 'e' have been calculated, they must be adjusted by the offset for each data format given in the table of section 9.3.

Register content convention

The values of 'x' and 'e', after the offset has been applied, are written to the two registers that contain the required scaling value. The upper register contains the numerand and the lower register the exponent.

Example

To represent -399.9 in offset 16-bit format. (Note that it would not be possible to encode the value directly in the chosen data format, the closest value that the format could represent is -400).

First 'normalise' the numerand to give a value of n < 1:

$$V = n \times 10^{e} = -0.3999 \times 10^{3} = -399.9$$

For this data format, the value of 'N' is 32768, thus:

$$V = (x / N) \times 10^{e} = x / 32768 \times 10^{3} = -13104/32768 \times 10^{3} = -399.9$$

Thus:

$$x = -13104 \text{ and } e = 3$$

In the offset 16-bit format, the figure to be written to the register is found from:

$$RV = REG - 32768$$

thus:

$$REGx = x + 32768 = -13104 + 32768 = 19664$$

$$REGe = e + 32768 = 3 + 32768 = 32771$$

These two values REGx and REGe can then be written to the two registers for the scaling parameter -399.9 with 'Offset 16-bit' data format selected. The value for REGx is written to the upper of the two registers, REGe is written to the lower.

Appendix D

Faultfinding on the MTL830 System

If the system does not operate, make sure that all inter-connections have been made correctly as described in INS831B and INS838B instruction sheets for the MTL831B and MTL838B-MBF, or the INS832 instruction sheet and INM838 instruction manual for the MTL832 and MTL832EXE. If the system still does not work correctly, try to identify which part of the system is malfunctioning by using the fault diagnoses set out below.

If it is suspected that a unit is faulty, DO NOT try to make any repairs or modifications since this may adversely affect operational and safety parameters. Faulty units should be returned to Eaton's MTL product line or our representative who supplied them.

Use one of the following tables, which are based on the unit's LED displays, to diagnose the reason for the system's behaviour, then refer to the numbered test procedures to identify the course of action.

Single highway mode

Highway 1 LED	Highway 1 LED	System failure LED	Comments
OFF	OFF	OFF	Check power supply - **Test 1**
OFF	OFF	ON	Check Highway - **Test 2** Check Tx. configuration - **Test 3**
ON*	OFF*	OFF	No data or invalid data being returned to Modbus master - check system configuration - **Test 4**

* Based on using Highway 1 - the order is reversed when using Highway 2

Dual highway mode

Highway 1 LED	Highway 1 LED	System failure LED	Comments
OFF	OFF	OFF	Check power supply - **Test 1**
OFF	ON	OFF	Check Highway 1 - **Test 2**
ON	OFF	OFF	Check Highway 2 - **Test 2**
OFF	OFF	ON	Check Highway 1 & 2 - **Test 2** Check Tx. configuration - **Test 3**
ON	ON	OFF	No data or invalid data being returned to Modbus master - check system configuration - **Test 4**

Test 1 - Power Supply

Using a voltmeter, ensure that 20 - 35V dc is present between receiver terminals 22 (+ve) and 23 (–ve).

Test 2 - Highways

The highway LED will be OFF if the highway is open circuit; if it has a short circuit; if there are one, or more, earth faults on the highway or if one or more configured transmitters are not responding.

Similarly, check that a highway voltage of approximately 12V dc is present across the receiver's data highway terminals:

Highway	Terminals
1	35 (+ve) and 36 (–ve)
2	37 (+ve) and 38 (–ve)

If this voltage is not present the data highway cabling may be faulty and should be checked thoroughly for earth faults and short circuits etc.

For IS installations, check that the 12V dc signal is present at the MTL3052 isolator's safe area terminals 2 (+ve) and 3 (–ve). If it is not, the data highway cabling may be faulty and should be checked thoroughly for earth faults and short circuits etc.

On IS installations, the highway signal at the transmitter cannot be checked unless an IS voltmeter is used. An alternative method to checking continuity in this situation is shown below. If the LED does not illuminate, but the highway signal is present at the receiver, then the MTL3052 isolator may be faulty (but check that the fuse in the MTL3052 has not blown).

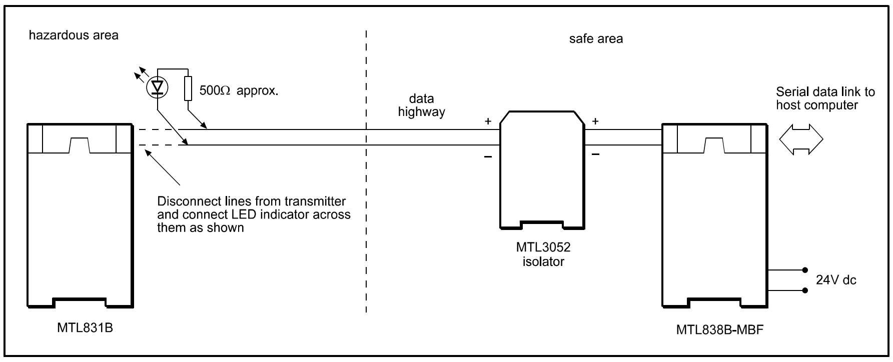

hazardous area
safe area
500Ω approx.
data highway
Serial data link to host computer
+
–
+
–
Disconnect lines from transmitter and connect LED indicator across them as shown
MTL3052 isolator
24V dc
MTL831B
MTL838B-MBF

Figure 33 - Checking highway continuity in an IS installation

If the highway signal is present up to the transmitter but the system still will not work, the configuration may be incorrect or the transmitter may be faulty. See Test 3 to check the system configuration.

Test 3 - Transmitter configuration

Check each transmitter's operation using one of the following methods.

a) the diagnostics communicated to the Modbus master,

b) the PCS83 configuration software

a) Modbus master diagnostics

The Modbus master can be used to investigate the STATUS register (see page 48) This will indicate faults for any transmitter that has been configured but is not responding.

b) PCS83 configuration software

With the software running and communicating with the MTL838B-MBF select the “Tx. Monitor” option. Starting with Transmitter 1 select “Monitor”. The message “Working” or “Tx fault” will be displayed in the bottom right corner of the screen to indicate the status of the transmitter.

Test 4 - System configuration

1. Check that the COM to MODE wire link, between terminal 4 and 5 on the MTL838B-MBF, has been removed following configuration with the PCS83.

If the link is in position, the configuration file should be downloaded again from the PCS83, and the Signoff procedure carried out to force a power-up reset of the MTL838B-MBF. The PCS83 will prompt for the configuration link to be removed.

2. Run the PCS83 software, with the PC linked to the MTL838B-MBF, and select the Tx. Monitor option. Choose Tx. No. 1 and select Monitor. Confirm that the cold junction temperature reading reflects, approximately, the temperature of the MTL831B - if not, replace the failed transmitter with one that is known to be functional. Set its address switches and the normal/3-wire RTD mode switches, load the required configuration and repeat the tests.

Check that the scaling values set for the input channel are in the same range as those of the Modbus master.

Repeat the same process for all of the transmitters.

3. Run the PCS83 software, with the PC linked to the MTL838B-MBF, and select the Configure Rx. option. Select Diagnostics. Check the “Ram Corruption” response on the right-hand Status screen. If this reads “YES” then use “Reset Rx” to download the configuration file to the receiver using the PCS83.
4. Run the PCS83 software, with the PC linked to the MTL838B-MBF, and select the Configure Rx. option. Check the right-hand half of the screen to ensure that the Link A and Link B addresses, the communication parameters and the Mode settings are the same as the Modbus master configuration.
5. Remove the terminal strip from the MTL838B-MBF to obtain access to Switches 101 and 102 (see page 39) and confirm that:

   Switch 101 - lever 8 - is OFF

   Switch 102 - levers 1 and 2 - are ON

If an MTL838B-MBF has been identified as potentially faulty, replace the unit with one that is known to be functional. Set its DIL switches to the required settings, download the configuration file using the PCS83 software and repeat the above tests.

Appendix E

Maintenance & Disposal

It is advisable to check the general condition of the installation from time to time to make sure no deterioration has occurred, and that no unauthorised modifications have been made. The following should be checked at intervals of not more than two years, and more frequently for systems used in particularly harsh environments.

Check that . . .

a) units are of the types specified in the relevant documentation.

b) unit labelling and tagging is clearly legible, and the details given comply with the relevant documentation.

c) units and enclosures are securely mounted.

d) there are no signs of damage or corrosion affecting the installation.

e) interconnecting cables are of the specified type and ratings are correctly routed and segregated, and are not frayed or otherwise damaged.

f) all connections are properly made.

g) the locations in which the units are mounted have not been degraded by the introduction of harmful materials.

h) access lids and doors to protective enclosures and cabinets are correctly secured.

Battery Replacement

The MTL838B-MBF contains a NiMH battery and is not a serviceable part. Therefore the instrument should be returned to Eaton, or their local representative for battery replacement.

Product - End of Life

The crossed-out wheeled bin means that within the European Union the product must be recycled in accordance with the WEEE directive and local environmental regulations.

CAUTION: DO NOT THROW BATTERIES INTO MUNICIPAL WASTE. DISPOSAL OF USED BATTERIES MUST BE SAFELY RECYCLED IN ACCORDANCE WITH LOCAL ENVIRONMENT REGULATIONS.

Index

address field, 12
address/location anomaly, 15, 47, 48
addressing
  inputs, 44
  slaves, 44
  transmitters, 44
ASCII mode, 10, 20
  comparison with RTU, 11
auxiliary input open, 50
broadcast, 7
byte count data, 13
cable - PCS83 serial link, 72
calibration, 69
CHECKSUM registers, 51
CLEAR COUNTERS AND DIAGNOSTIC REGISTERS, 30
COIL STATUS, 15
coil status flags
  mapping, 54
COIL STATUS FLAGS, 54
cold junction temperature, 52
colon, 11
configuration changing, 50
configuration rejected, 50
Converters, 8
CRC, 10, 13, 31
  calculation of, 85
CSUMREF, 57
Cyclical Redundancy Check. *See* CRC
daisy chain, 19
data
  query, 9
  response, 9
data converters, 19
data field, 12
data format
  IEEE, 86
  non-IEEE, 87
data timing, 70
device address
  query, 9
  response, 9
DIAGNOSTICS, 28
  exception responses, 65
DIL switches
  locations, 39
  settings, 40
  settings for PCS83, 73
encoding of data, 12, 13
error, 8
error check
  field, 13
  query, 9
  response, 9
error flag, 49
error messages - PCS83, 75
exception code
  ILLEGAL DATA ADDRESS, 35
  ILLEGAL DATA VALUE, 36
  ILLEGAL FUNCTION, 35
  NEGATIVE ACKNOWLEDGE, 36
  SLAVE DEVICE FAILURE, 36
exception responses, 8, 16
  code list, 16
  construction, 34
  interpretation, 64
  supported by the MTL838B-MBF, 34
flag
  organisation, 14
flags
  COIL STATUS, 15
    CONFIRM, 55
    CSTORE, 54
    DFT831, 54
    FMT831, 55
  HOLDING REGISTERS, 15
  INPUT REGISTERS, 15
  INPUT STATUS, 15
FORCE SINGLE COIL, 25
  exception responses, 65
function code

query, 9
response, 9
function codes, 15
function field, 12
HAN83
connector, 42
HCA software
revision number, 48
high alarm, 51
highway OK, 50
holding register
CFGTEST, 60
CSUMREF, 57
DATAFMT, 57
GAIN, 63
high alarm, 63
input type, 60
IPZERO, 62
line frequency, 60
low alarm, 63
no. of Txs, 59
OPZERO, 63
safety drive, 60
TAG field, 59
temp units, 59
type of Tx, 59
unused, 57
HOLDING REGISTERS, 15, 56
IEEE data format
description, 86
IEEE format
mappings, 46
ILLEGAL DATA ADDRESS, 35
ILLEGAL DATA VALUE, 36
ILLEGAL FUNCTION, 35
initialisation mode, 43
input offset, 67
input processing
mV, 70
resistance, 69
RTDs, 69
thermocouples, 69
input registers, 46
mapping, 46
INPUT REGISTERS, 15
INPUT STATUS, 15
input status flags, 46
mapping, 46
invalid database, 50
IPP software
revision number, 48
linear BUS, 19
location 'offset', 15
Longitudinal Redundancy Check. *See* LRC
low alarm, 51
LRC, 10, 13, 31
calculation of, 85
message fields, 12
message framing
ASCII, 11
MODBUS, 11
RTU, 12
MODBUS
concepts, 14
functions, 15
MTL838B-MBF
background to, 37
configuring, 37
interconnection, 42
modes of operation, 43
status register, 48
multi-drop, 18
NEGATIVE ACKNOWLEDGE, 36
no. of slaves
JBUS, 8
non-IEEE data format
description, 87
non-IEEE format
mappings, 47
offset - input, 67
open alarm, 52
open circuit, 51
parity checking, 8
PCS83, 38
configure Rx, 76
configure Tx, 78

configuring with, 71
copy configuration, 80
download configuration, 81
installing, 71
laptop computers, 73
load configuration, 81
logged drive, 81
monitor all inputs, 82
monitor an input, 82
print configuration, 83
programming, 82
running, 74
save configuration, 81
serial link cable, 72
serial port select, 82
signoff, 81
troubleshooting, 73
upload data, 75
PCS83 - error messages, 75
PCS83 software, 14, 37
physical layer, 17
point-to-point, 18
PRESET MULTIPLE REGISTERS, 33
exception responses, 66
PRESET SINGLE REGISTER, 26
exception responses, 65
query, 7, 9
query-response cycle, 9
READ COIL STATUS, 20
exception responses, 64
READ EXCEPTION STATUS, 27
READ EXCEPTION STATUS'
exception responses, 65
READ HOLDING REGISTERS, 23
exception responses, 64
READ INPUT REGISTERS, 24
exception responses, 65
READ INPUT STATUS, 22
exception responses, 64
redundancy, 42
register
concept, 14
organisation, 14
request, 7
response, 8
RETURN BUS COMMUNICATION ERROR COUNT, 31
RETURN BUS EXCEPTION ERROR COUNT, 32
RETURN BUS MESSAGE COUNT, 30
RETURN DIAGNOSTIC REGISTER, 29
RETURN QUERY DATA, 28
revision number
HCA software, 48
IPP software, 48
RS232, 8, 17, 18, 19, 41
limitations, 42
non RS232 interface, 17
RS232C. *See* RS232
RS422, 8, 18, 19
RS485, 8, 17, 19, 37, 39, 41
2 or 4-wire, 18, 43
limitations, 42
port parameters, 39
unused, 43
RTU mode, 10
comparison with ASCII, 11
scaled analogue value, 52
scaling parameters
background, 67
practical, 68
scaling system, 13
scan speed of inputs, 70
self calibration, 69
sensors
signal processing, 67
signal processing
sensors, 67
silent period, 12
SLAVE DEVICE FAILURE, 36
switches - DIL. *See* DIL switches
termination of network, 17
transmission
modes, 10
rates, 8
transmitter failed, 50
wire link - MODE to COM, 72

CROUSE-HINDS
SERIES

AUSTRALIA
MTL Instruments Pty Ltd,
10 Kent Road, Mascot, New South Wales, 2020, Australia
Tel: +61 1300 308 374 Fax: +61 1300 308 463
E-mail: mtlsalesanz@eaton.com

BeNeLux
MTL Instruments BV
Ambacht 6, 5301 KW Zaltbommel
The Netherlands
Tel: +31 (0)418 570290 Fax: +31 (0)418 541044
E-mail: mtl.benelux@eaton.com

CHINA
Cooper Electric (Shanghai) Co. Ltd
955 Shengli Road, Heqing Industrial Park
Pudong New Area, Shanghai 201201
Tel: +86 21 2899 3817 Fax: +86 21 2899 3992
E-mail: mtl-cn@eaton.com

FRANCE
MTL Instruments sarl,
7 rue des Rosiéristes, 69410 Champagne au Mont d'Or
France
Tel: +33 (0)4 37 46 16 53 Fax: +33 (0)4 37 46 17 20
E-mail: mtlfrance@eaton.com

GERMANY
MTL Instruments GmbH,
Heinrich-Hertz-Str. 12, 50170 Kerpen, Germany
Tel: +49 (0)22 73 98 12- 0 Fax: +49 (0)22 73 98 12- 2 00
E-mail: csckerpen@eaton.com

INDIA
MTL India,
No.36, Nehru Street, Off Old Mahabalipuram Road
Sholinganallur, Chennai- 600 119, India
Tel: +91 (0) 44 24501660 /24501857 Fax: +91 (0) 44 24501463
E-mail: mtlindiasales@eaton.com

ITALY
MTL Italia srl,
Via San Bovio, 3, 20090 Segrate, Milano, Italy
Tel: +39 02 959501 Fax: +39 02 95950759
E-mail: chmninfo@eaton.com

JAPAN
Cooper Crouse-Hinds Japan KK,
MT Building 3F, 2-7-5 Shiba Daimon, Minato-ku,
Tokyo, Japan 105-0012
Tel: +81 (0)3 6430 3128 Fax: +81 (0)3 6430 3129
E-mail: mtl-jp@eaton.com

NORWAY
Norex AS
Fekjan 7c, Postboks 147,
N-1378 Nesbru, Norway
Tel: +47 66 77 43 80 Fax: +47 66 84 55 33
E-mail: info@norex.no

RUSSIA
Cooper Industries Russia LLC
Elektrozavodskaya Str 33
Building 4
Moscow 107076, Russia
Tel: +7 (495) 981 3770 Fax: +7 (495) 981 3771
E-mail: mtlrussia@eaton.com

SINGAPORE
Cooper Crouse-Hinds Pte Ltd
No 2 Serangoon North Avenue 5, #06-01 Fu Yu Building
Singapore 554911
Tel: +65 6645 9864 / 6645 9865 Fax: +65 6 487 7997
E-mail: sales.mtlsing@eaton.com

SOUTH KOREA
Cooper Crouse-Hinds Korea
7F. Parkland Building 237-11 Nonhyun-dong Gangnam-gu,
Seoul 135-546, South Korea.
Tel: +82 6380 4805 Fax: +82 6380 4839
E-mail: mtl-korea@eaton.com

UNITED ARAB EMIRATES
Cooper Industries/Eaton Corporation
Office 205/206, 2nd Floor SJ Towers, off. Old Airport Road,
Abu Dhabi, United Arab Emirates
Tel: +971 2 44 66 840 Fax: +971 2 44 66 841
E-mail: mtlgulf@eaton.com

UNITED KINGDOM
Eaton Electric Ltd,
Great Marlings, Butterfield, Luton
Beds LU2 8DL
Tel: +44 (0)1582 723633 Fax: +44 (0)1582 422283
E-mail: mtlenquiry@eaton.com

AMERICAS
Cooper Crouse-Hinds MTL Inc.
3413 N. Sam Houston Parkway W.
Suite 200, Houston TX 77086, USA
Tel: +1 281-571-8065 Fax: +1 281-571-8069
E-mail: mtl-us-info@eaton.com

EATON
Powering Business Worldwide

Eaton Electric Limited,
Great Marlings, Butterfield, Luton
Beds, LU2 8DL, UK.
Tel: + 44 (0)1582 723633 Fax: + 44 (0)1582 422283
E-mail: mtlenquiry@eaton.com
www.mtl-inst.com

© 2017 Eaton
All Rights Reserved
Publication No. INM838B-MBF Rev 7 020217
February 2017

EUROPE (EMEA):
+44 (0)1582 723633
mtlenquiry@eaton.com

THE AMERICAS:
+1 800 835 7075
mtl-us-info@eaton.com

ASIA-PACIFIC:
+65 6 645 9888
sales.mtlsing@eaton.com