

# *Mega\_Link*

## ***APPLICATION NOTE AN027*** ***Using the Fieldbus Port on COM3***

### **Summary**

In addition to transferring 'Real-World' physical data, *Mega\_Link* can also interface directly to a PLC, SCADA system or third party telemetry system over the *Fieldbus* serial port. This document outlines the supported options for accessing this information. More detailed information can be found in the *Mega\_Link Technical Manual*.

This document assumes that the user has already established the communications network of *Mega\_Link* modules.

For the sake of simplicity this document assumes each item of third party equipment is a PLC. However, it could equally be a SCADA system, a regional telemetry outstation or an intelligent instrument.

# 1. Overview

## 1.1 Introduction

Many PLCs have the ability to communicate with other devices via a serial link. This is generally done at one of two electrical interface standards – RS232 or RS485. The PLC manufacturer also defines the protocol (language) used for communication. The device to which it is linked, along with the data rate and format must match this.

Virtually all PLC protocols operate on a master/slave basis, whereby a master device can ‘talk’ to one or more slaves. All communication is instigated by the master, and each slave responds only to the commands addressed to it. Many PLCs operate only in slave mode. Numerous SCADA systems have been developed to communicate with PLCs, so these generally implement only the relevant master mode. Many regional telemetry outstations have the facility to link to PLCs, and they generally implement only master mode.

*Mega\_Link* has been designed to interface to any of these systems. A serial interface port is included within each *Mega\_Link*, and is named *Fieldbus* to signify that it is a general-purpose port for communication with various PLC busses. It can be configured to emulate either master or slave in any required protocol, data rate and format, and can use either RS232 or RS485 signal levels.

## 1.2 Configuration Process

The process to set up *Fieldbus* communications within *Mega\_Link* follows the following sequence:

### 1.2.1 Configure the Communications Network

Every network comprises a base-station and one or more outstations. Each outstation can be configured to also act as a repeater if necessary to access more distant outstations.

### 1.2.2 Decide whether the Mega\_Link is to be configured as a master or a slave

Overall system configuration is far simpler if the *Mega\_Link* is configured as the master. However, if the PLC can only implement master protocol the *Mega\_Link* can be configured as a slave.

### 1.2.3 Establish a hardware link between each PLC and Mega\_Link

The wiring between the two could use either RS232 or RS485. The user needs to connect the items together and prove that the link is functioning. This is most easily done by installing a simple configuration, then using DUCX Diagnostics to prove that communication is working.

### 1.2.4 Configure Mega\_Link and/or the PLC to pass the required data

Once communications are functioning the user can define the configuration necessary to pass the required data.

*The rest of this document expands on the process.*

## 2. Protocol and Interface

### 2.1 Protocols

Various PLC types are supported by *Mega\_Link*, as it can communicate in a number of different languages (protocols). The most commonly implemented is Modbus, as the majority of PLC and SCADA systems support Modbus in one form or another – either inherently, or by the addition of a Modbus card. However, should the user not wish to, or be able to, talk Modbus, a number of other PLC manufacturers’ own protocols are also supported.

#### 2.1.1 Modbus

Modbus is an open protocol based on master/slave architecture. It is popular, well established, relatively easy to implement and reliable.

Since it is so easy to implement, Modbus has gained wide market acceptance wherever automation or management systems need to communicate with other devices, and is probably the most widely implemented automation protocol.

The simplicity of Modbus RTU messages is a mixed blessing. On the one hand, the simple message structure ensures widespread, rapid and accurate implementation, but on the other hand, various companies have corrupted the basic 16-bit Modbus RTU register structure to pack in floating point, queues, ASCII text, tables and other types of non-Modbus data.

*Mega\_Link* is fully configurable and allow user adjustments to compensate for the above issues

Modbus comes in two varieties – RTU and ASCII – both of which are supported by *Mega\_Link*.

The Modbus RTU protocol is a fast, efficient binary protocol that uses all 8 bits of each character. The Modbus ASCII protocol uses two ASCII characters to send and receive each character of the Modbus message. While less efficient than its RTU counterpart, it is also widely accepted as a communication standard for remote telemetry applications.

#### 2.1.2 Mitsubishi

Mitsubishi PLC’s can be configured to either ‘dedicated protocol’ mode, or ‘no-protocol’ mode.

In ‘no-protocol’ mode, the PLC programmer must define the protocol in ladder logic, which is an onerous task.

*Mega\_Link* therefore supports the dedicated protocol, which is the ‘Melsec-A’ protocol.

Both ‘FX’ and non-‘FX’ types of Mitsubishi PLC are supported using this protocol, with the configuration requirements for each being detailed in the *Mega\_Link* Technical Manual.

Note that some Mitsubishi serial interface modules do not support dedicated protocol..

#### 2.1.3 Allen-Bradley

*Mega\_Link* supports the standard Allen Bradley protocol ‘DF1’.

Note that, although the protocol may be supported on various PLCs, *Mega\_Link* will only connect over a serial link.

This means it will not connect over any network based link, such as DH, DH+, DH485 or ControlNet, unless a suitable RS232 module is fitted in the network.

Also, *Mega\_Link* will only support the Half-Duplex method of transmission. This is however sufficient to connect up to 255 nodes simultaneously, when using the RS485 connection option on *Mega\_Link*.

## 2.2 Fieldbus Connections

The *Fieldbus* port on COM3 of the the *Mega\_Link* is used for communication with PLC’s, SCADA systems and larger telemetry schemes. It is configured through the DUCX software, and presented in either RS232C or RS485 format.

The RS232 and RS485 interfaces is provided through an 8-pin RJ45 socket on COM3.

The RS485 interface uses pins designated A, T and B. The line can be multi-dropped to 32 devices, and should be terminated at both ends with 100Ω.

Note that RS485 is polarity-conscious, so all A terminals must be joined together, as must all B terminals.

## 2.3 Diagnostics

An LED indicator is included on *Mega\_Link* to monitor the *Fieldbus* port. It flashes red when *Mega\_Link* is transmitting data and green when it is receiving.

Furthermore, *Mega\_Link* includes very powerful diagnostics via the DUCX USB port. This allows a PC to be connected to configure and monitor all aspects of its operation. When set to monitor the *Fieldbus* communications it can display every message sent to/from *Mega\_Link*, both in plain English and in hex code.

## 2.4 Use with a PLC Slave

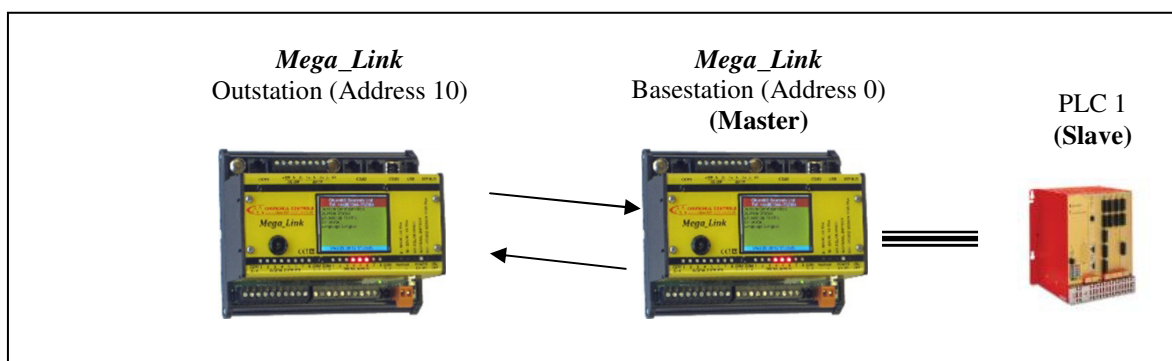
All PLC protocols work on a master-slave principle, whereby one unit on the PLC network is defined as a master and all others are slaves. The master can pass data to/from one or more slaves. Each slave must be given a unique address so the master can distinguish it.

By far the simplest configuration when using *Mega\_Link* with a PLC is to configure the

*Mega\_Link* as the master and the PLC as a slave. The *Mega\_Link* will hence be responsible for initiating all transactions.

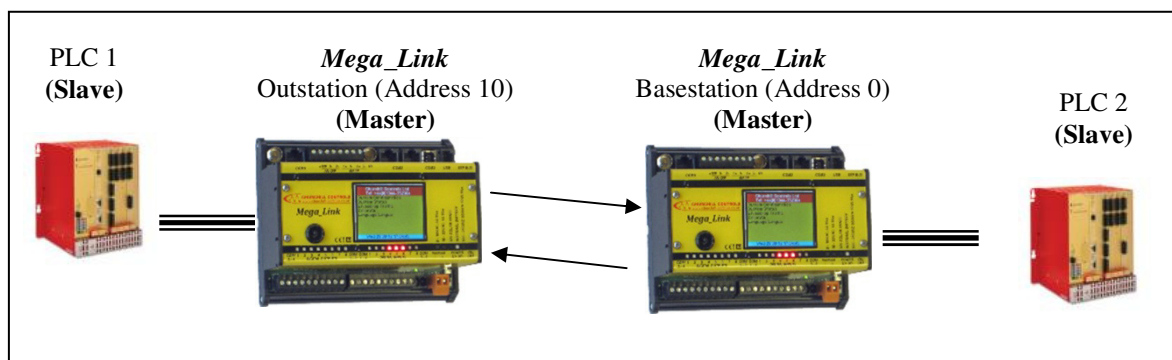
The user needs have no knowledge of the internal database structure of the *Mega\_Link*. All data routing is configured in the *Mega\_Link* Data Routing Table using simple descriptions to define the data source and destination.

**Example 1:** Transferring real world inputs & outputs to a PLC slave device via a remote base-station.



In this example the basestation's Data Routing Table would be configured to pass I/O from the outstation to/from specific registers in PLC 1. PLC 1 will have a defined address that must be used within the basestation Data Routing Table to identify the PLC. It should be apparent that more than one slave PLC could be connected to the *Mega\_Link* basestation, provided the electrical interface supports multi-drop usage.

**Example 2:** Two PLC slaves transferring register data.



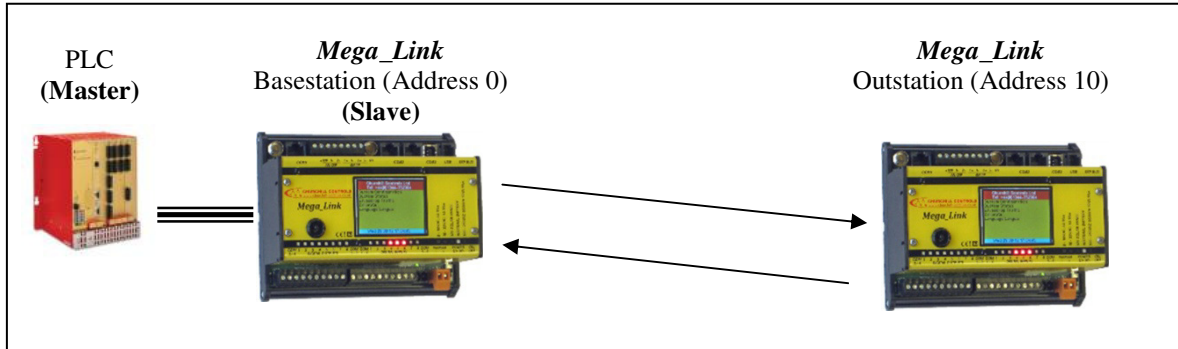
In this example the outstation Data Routing Table would be configured to copy data from PLC 1 to the outstation database, where it will appear as additional I/O. The base-station Data Routing Table can then be configured to pass the outstation's I/O to/from PLC 2.

## 2.5 Use with a PLC Master

If the PLC cannot be set as a Slave device for any reason, it is possible to set Mega\_Link as the Slave, allowing the PLC to be the Master.

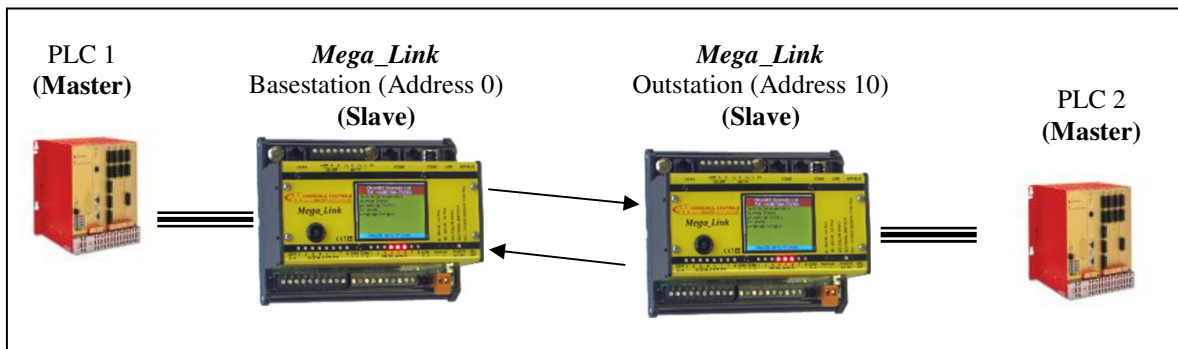
This is more complicated than previously, as the user needs have no knowledge of the internal database structure of the Data\_Link system.

**Example 3:** A PLC master transferring real world inputs & outputs to a remote outstation.



The outstation is configured with two independent addresses. One defines its address on the *Mega\_Link* network (10 in this example), and the other defines its address on the PLC network. The PLC master must be configured to pass data to/from specific database locations within the outstation. The user needs to understand the structure of the *Mega\_Link* database to determine which database locations to use.

**Example 4:** Two PLC masters transferring register data.



In this example PLC 1 will be configured to copy registers to/from its own database into the outstation database. Similarly PLC 2 will be configured to copy the same registers from/to its own database. The *Mega\_Link* protocol will automatically ensure that the relevant registers within the outstation and the base-station are kept in step.

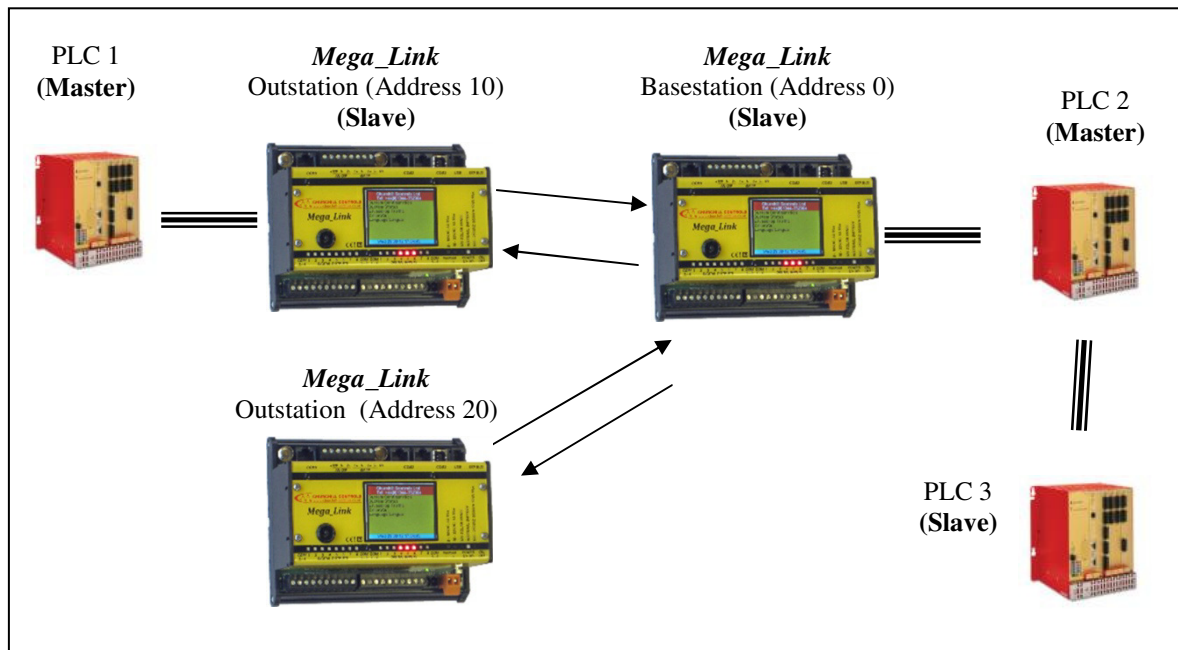
## 2.6 Use with a Combinations of the Above.

Referring to Example 2 above, it can be seen that there are three independent asynchronous communications subsystems:

- Serial Comms from PLC1 to *Mega\_Link* basestation
- Radio Comms from *Mega\_Link* basestation to *Mega\_Link* outstation
- Serial Comms from PLC2 to *Mega\_Link* outstation.

As these are each self-contained, it is possible to add to each of them as required, adding further PLCs and outstations as necessary.

Note that since *Mega\_Link* operates on a polling principle, there can only be one basestation on any given system, with one or more outstations. Similarly, since most PLC protocols also work on a polling principle, there can only be one PLC master on any given system, with one or more slaves:



It should be obvious that in this example the basestation and PLC 3 must be configured with different slave addresses.

## 2.7 Other Considerations

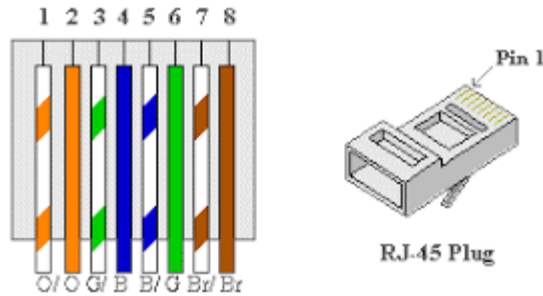
As well as maintaining within its database all the relevant I/O conditions, each *Mega\_Link* also monitors its own status. Certain database locations therefore store alarm flags (e.g. comms fail and battery low) and monitoring levels (e.g. battery voltage and radio receive signal strength). The user can include entries in the Data Routing Table to copy these to discrete outputs, but they can equally be copied to any required location within PLCs via the *Fieldbus* interface.

### 3. Hardware Connection

The *Fieldbus* port is used for communication with PLC's, SCADA systems and larger telemetry schemes. It is configured using the DUCX software, and presented in either RS232C or RS485 format.

#### 3.1 RS232C

The RS232 interface on COM3 is provided through an 8-pin RJ45 socket, configured as follows:



COM3 RJ-45 Pin	CAT5 Colour	Function wrt <i>Mega_Link</i>	Direction wrt <i>Mega_Link</i>
1	White/Orange	0V	<>
2	Orange	Do not connect	-
3	White/Green	Do not connect	-
4	Blue	Do not connect	-
5	White/Blue	TXD	O/P >
6	Green	RXD	I/P <
7	White/Brown	RTS	O/P >
8	Brown	CTS	I/P <

Typical connections to a 9-way D connector:

DCE (e.g. PC)			DTE (e.g. PLC)			D-type Pin
Colour	Function wrt to PC	Direction wrt to PC	Colour	Function wrt to PLC	Direction wrt to PLC	
-	Shield	-	-	Shield	-	1
Green	TXD	O/P >	White/Blue	RXD	I/P <	2
White/Blue	RXD	I/P <	Green	TXD	O/P >	3
Link to DSR	DTR	O/P >	Link to DSR	DTR	O/P >	4
White/Orange	0V	<>	White/Orange	0V	<>	5
Link to DTR	DSR	I/P <	Link to DTR	DSR	I/P <	6
White/Brown	CTS	I/P <	Link to RTS	CTS	I/P <	7
Brown	RTS	O/P >	Link to CTS	RTS	O/P >	8
-	-	-	-	-	-	9

A rule of thumb for DB9 connector is: If it's male it is likely to be a DTE, if it's female it's likely to be a DCE.

The most common problem when connecting to a serial device with *Mega\_Link* is the orientation of the TXD and RXD lines. *Mega\_Link* terminology regards TXD as the line through which the remote device accepts data, but some manufacturers regard it as the line on which *Mega\_Link* would receive data. The simplest way to identify the orientation is to measure the voltage on the relevant pin (relative to the 0V pin) when the units are not connected together. Each output line will measure a voltage in excess of 5V (it may be positive or negative), while each input pin will not measure any voltage. Obviously the output from one device must be connected to the input of the other.

Another problem that may occur relates to handshaking. *Mega\_Link* does not need any handshaking, since it is able to receive data at all times. Most third party products are the same, but some will raise RTS when they need to send data, and delay sending it until CTS is raised. This situation can be overcome by connecting together the RTS and CTS pins on the PLC. Furthermore, some devices raise DTR to signify that they are active, and will ignore all communication until DSR is returned. This situation can be overcome by connecting together the DTR and DSR pins on the PLC.

Note that RS232 is a point-point interface, so can only be used to connect *Mega\_Link* to a single PLC device.

### 3.2 RS485

RS485 is a two-wire interface that allows up to 32 devices to be interconnected via a single twisted pair of wires. Each end of the wire should be terminated with a 100Ω resistor. A termination resistor is fitted within *Mega\_Link*, but should only be connected if *Mega\_Link* is at one end of the wire. A 3-pin connector is therefore provided on *Mega\_Link*, with pins designated A, T and B. The line should be connected between A and B, and the termination resistor can be activated by linking T to B.

Note that RS485 is polarity-conscious, so all 'A' terminals must be joined together, as must all 'B' terminals. Some manufacturers may use a different designation, but a clear indication is provided on *Mega\_Link* to show the correct orientation:

If the A and B connections are reversed, the *Fieldbus* LED on *Mega\_Link* will show steady green. If this happens, simply reverse the connections.

COM3 RJ-45 Pin	CAT5 Colour	Function wrt <i>Mega_Link</i>	Direction wrt <i>Mega_Link</i>
1	White/Orange	0V	◊
2	Orange	RS485 B	◊
3	White/Green	RS485 T	Termination resistor by linking to RS485 A
4	Blue	RS485 A	◊
5	White/Blue	Do not connect	-
6	Green	Do not connect	-
7	White/Brown	Do not connect	-
8	Brown	Do not connect	-

### 3.3 Hardware Diagnostics

Before attempting to design the configuration in detail, the user is advised to install a very simple configuration first, to confirm that the communication path is working. A single line, copying a register from the *Mega\_Link* to the PLC, is adequate.



If the communications are working the *Fieldbus* LED should be normally off, but flash red whenever *Mega\_Link* sends data, and green when it receives data. If the PLC includes a comms monitor LED it should also flash as data is passed.

If the LED's suggest no communication is taking place, the user should address the problem before proceeding. The most likely problem is the wiring, as described above. Once the wiring is correct, the next hurdle is to confirm that the *Mega\_Link* and the PLC understand each other, so the data rate and format must match, and the correct slave address must be used.

Once the user has convinced themselves that the wiring and the protocol are correct, they can run DUCX diagnostics (as described later in this document) to watch the data.

## 4. Software Configuration

### 4.1 General

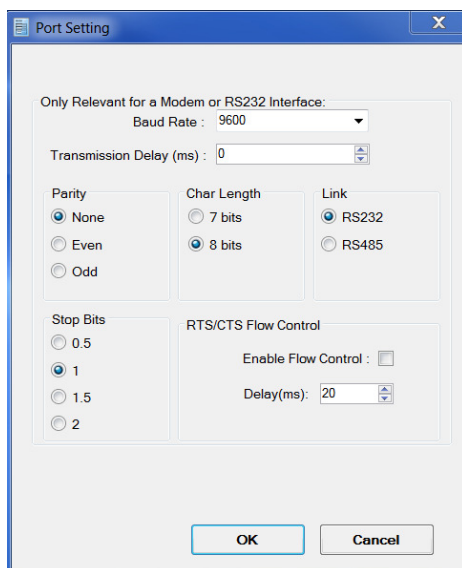
All software configuration of *Fieldbus* is carried out using the *Diagnostics, Configuration and Firmware Upgrade* (DUCX) software that is supplied on a memory stick with every *Mega\_Link* system. Alternatively it can be downloaded from our website on [www.churchill-controls.co.uk](http://www.churchill-controls.co.uk).

When DUCX is running, a configuration can be created either by opening an existing file, creating a new file or uploading the configuration from a *Mega\_Link*. Each configuration creates a new window.

### 4.2 Protocol

When linking the two pieces of equipment together, their data formats must be matched. That is, both units must be working on the same protocol, at the same data speed, using the same number of data bits, the same number of stop bits and the same parity.

These settings can be chosen by clicking on the 'COM3' button within the 'System Configuration' tab. This will open a dialog box called 'Port Setting' as follows:



The parameters shown should be self-explanatory, and must match those of the PLC.

### 4.3 Mega\_Link Database

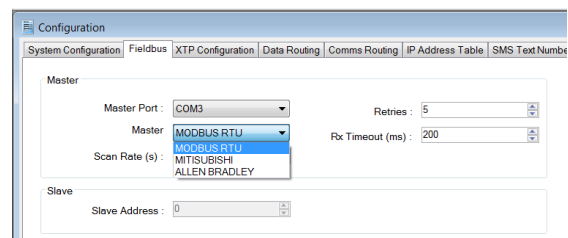
Each *Mega\_Link* unit maintains a database of 2000 input registers, 2000 output registers, 8000 digital inputs and 8000 digital outputs. *Mega\_Link* maps all the I/O from the basestation and outstations into this database in a well-defined, structured way. DUCX knows the structure and simplifies identification of each I/O point into simple plain English terminology, such as 'Outstation 10 Digital Input 1'.

If *Mega\_Link* is configured as a *Fieldbus* master, the user does not need any knowledge of the mapping, since they will be configuring the *Fieldbus* communications through DUCX. However, if *Mega\_Link* is configured as a *Fieldbus* slave, the user does need to know the database format so they can access the relevant registers.

It is therefore much easier to configure systems that use *Mega\_Link* as a master

### 4.4 Use with a PLC Slave

The *Mega\_Link* mode is configured using the Feildbus tab:



To configure a *Mega\_Link* as a Master, select the COM3 in the 'Master Port' selection box and the required protocol in the 'Master' selection box as shown. The parameters in the 'Master' area of the tab are only relevant when *Mega\_Link* is configured as a master:

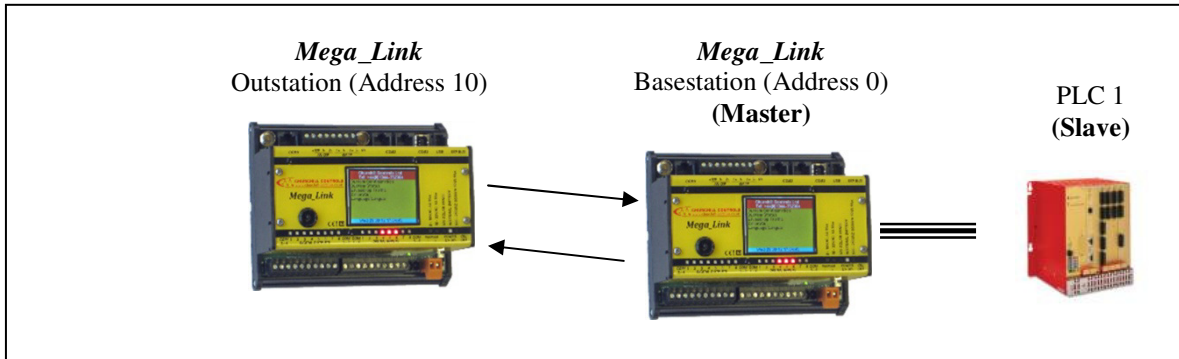
**Scan Rate:** The period at which the master polls the slave(s) for information.

**Rx Timeout:** This is the time for which the master will wait for a reply from the slave.

**Retries:** The maximum number of times the master will retry the same command to the slave.

## 4.5 Data Routing when using a PLC Slave:

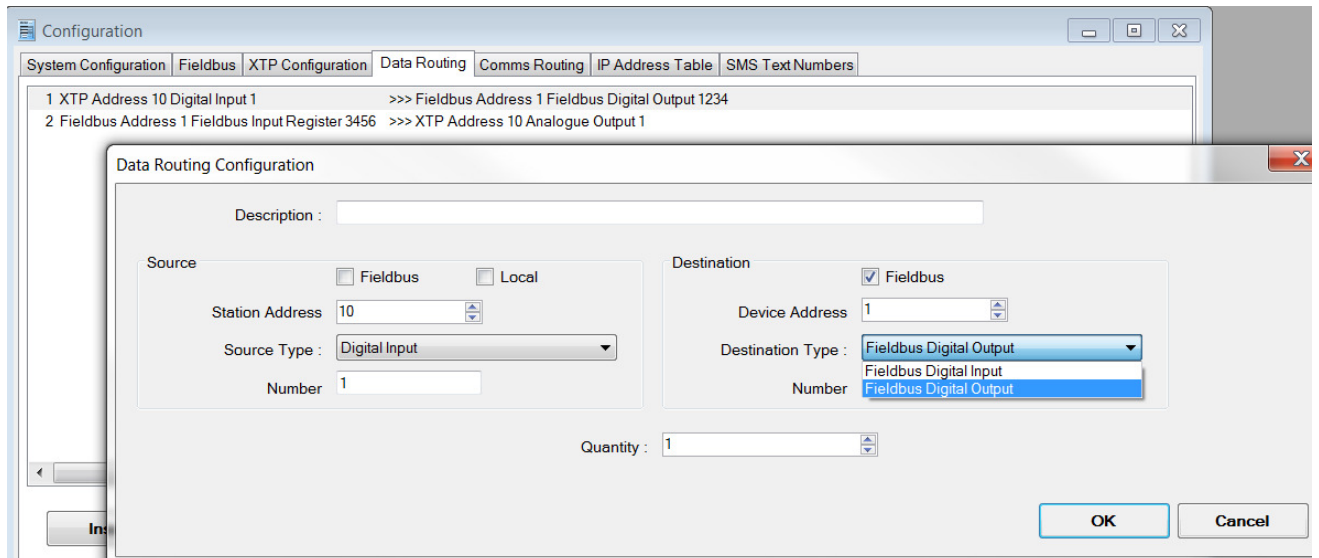
**Example 1:** Transferring real world inputs & outputs to a PLC slave device via a remote basestation.



As stated above, when *Mega\_Link* is configured as a master device, all data routing is configured via DUCX.

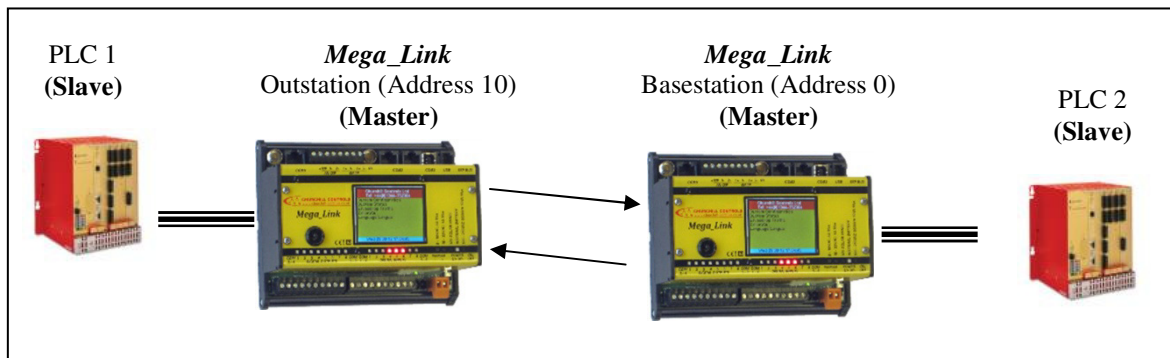
A sample Data Routing table is shown for the basestation:

### **Fieldbus Master Basestation:**



This will copy the state of digital input 1 at the outstation to digital output register 1234 in the PLC, and copy output register 3456 from the PLC to the first analogue output at outstation 10. Note that references to registers in the PLC use absolute addressing (i.e. the definitions at the top of the pull-down menu, prefixed with \*), since PLC's are not aware of the data structure used in *Mega\_Link*.

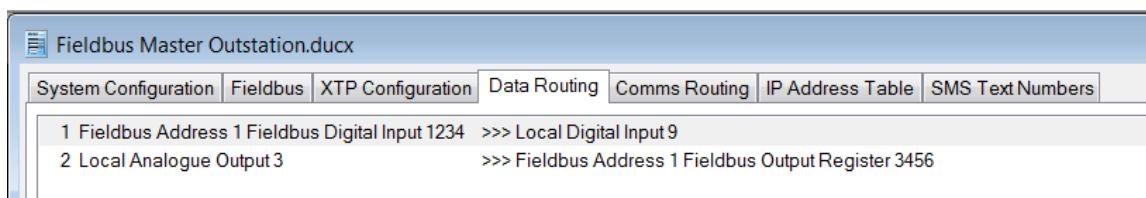
**Example 2:** Two PLC slaves transferring register data.



All *Mega\_Link* XTP transfers are configured in the basestation, so inputs/outputs should be viewed from that end. The basestation reads from local inputs and/or outstation inputs and writes to outputs. The *Fieldbus* configuration then has to follow suit.

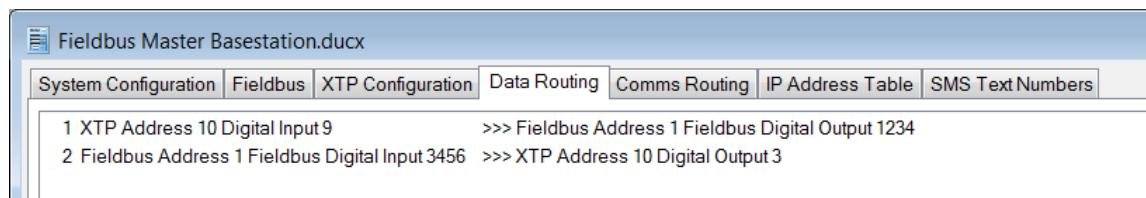
This requires entries in the Data Routing Table at both the outstation and the basestation:

**Fieldbus Master Outstation:**



Because data is copied from the PLC to digital input 9, this outstation will appear to the base-station as if it has a total of 9 digital inputs (i.e. the 8 inputs that are integral to *Mega\_Link*, plus one derived from the PLC). It is not relevant to the basestation whether the extra inputs come from expansion modules at the outstation or from *Fieldbus*. Similarly, the outstation will appear to have an extra analogue output in addition to the two which are integral to *Mega\_Link*.

**Fieldbus Master Basestation:**



The overall effect of this will be to copy digital input 1234 of the PLC at the outstation to digital output register 1234 of the PLC at the base-station, and to copy input register 3456 of the PLC at the basestation to output register 3456 of the PLC at the outstation.

**4.6 Data Routing when using a PLC Master:**

If the PLC cannot be set as a slave device, *Mega\_Link* can be configured as the slave, allowing the PLC to be the master. This is more complicated than previously, as the user needs have knowledge of the internal database structure of the *Mega\_Link*.

A *Mega\_Link* basestation effectively maintains a database of 2000 input registers, 2000 output registers, 8000 digital inputs and 8000 digital outputs. This can be more conveniently viewed as 250 input data blocks and 250 output datablocks.

A *Mega\_Link* outstation or basestation equipped with up to one digital input expansion module and/or one digital output expansion module also occupies only the root data block:

### Mega\_Link Root Data Block Usage

INPUT DATA BLOCK				OUTPUT DATA BLOCK			
	Digital	Register		Digital	Register		
0	Outstation Comms Fail	Count 1	0	-			
1	Battery Low Alarm						
2	Basestation Comms Fail						
3	Fieldbus Comms Fail						
4	Total b/s Comms Fail	Count 2	1	-			
5	Mains Fail						
6	Primary Comms Fail						
7	Secondary Comms Fail	Count 3	2	-			
8	Digital Input 1						
9	Digital Input 2						
10	Digital Input 3						
11	Digital Input 4	Count 4	3	-			
12	Digital Input 5						
13	Digital Input 6						
14	Digital Input 7						
15	Digital Input 8	Battery Volts	4	-			
16	Digital Input 9 <sup>1</sup>						
17	Digital Input 10 <sup>1</sup>						
18	Digital Input 11 <sup>1</sup>						
19	Digital Input 12 <sup>1</sup>	RSSI	5	-			
20	Digital Input 13 <sup>1</sup>						
21	Digital Input 14 <sup>1</sup>						
22	Digital Input 15 <sup>1</sup>						
23	Digital Input 16 <sup>1</sup>	Analogue Input 1	6	-			
24	Digital Input 17 <sup>1</sup>						
25	Digital Input 18 <sup>1</sup>						
26	Digital Input 19 <sup>1</sup>	Analogue Input 2	7	-			
27	Digital Input 20 <sup>1</sup>						
28	Digital Input 21 <sup>1</sup>						
29	Digital Input 22 <sup>1</sup>						
30	Digital Input 23 <sup>1</sup>	Analogue Output 1		-			
31	Digital Input 24 <sup>1</sup>						
				Digital Output 1			
				Digital Output 2			
				Digital Output 3			
				Digital Output 4			
				Digital Output 5			
				Digital Output 6			
				Digital Output 7			
				Digital Output 8			
				Digital Output 9 <sup>2</sup>			
				Digital Output 10 <sup>2</sup>			
				Digital Output 11 <sup>2</sup>			
				Digital Output 12 <sup>2</sup>			
				Digital Output 13 <sup>2</sup>			
				Digital Output 14 <sup>2</sup>			
				Digital Output 15 <sup>2</sup>			
				Digital Output 16 <sup>2</sup>			
				Digital Output 17 <sup>2</sup>			
				Digital Output 18 <sup>2</sup>			
				Digital Output 19 <sup>2</sup>			
				Digital Output 20 <sup>2</sup>			
				Digital Output 21 <sup>2</sup>			
				Digital Output 22 <sup>2</sup>			
				Digital Output 23 <sup>2</sup>			
				Digital Output 24 <sup>2</sup>			

All alarm bits are '1' in the normal state and '0' when in fail.

### NOTES

<sup>1</sup> Only available if a digital input expansion module is fitted, or data is mapped to fieldbus device, otherwise read '0'.

<sup>2</sup> Only available if a digital output expansion module is fitted, or data is mapped to fieldbus device, otherwise ignored.

If more than one digital input expansion module is fitted it will 'spill over' into the next input data block, as will any analogue input expansion modules. Similarly, if more than one digital output expansion module is fitted it will 'spill over' into the next output data block, as will any analogue output expansion modules. A *Mega\_Link* base-station or outstation could therefore occupy up to 33 consecutive data blocks in the worst case (i.e. if it is fitted with a full compliment of 32 analogue input modules).

Any given register or digital can be identified by an absolute address or a relative address from a reference Data Block.

For example, each of the following refers to the same register:

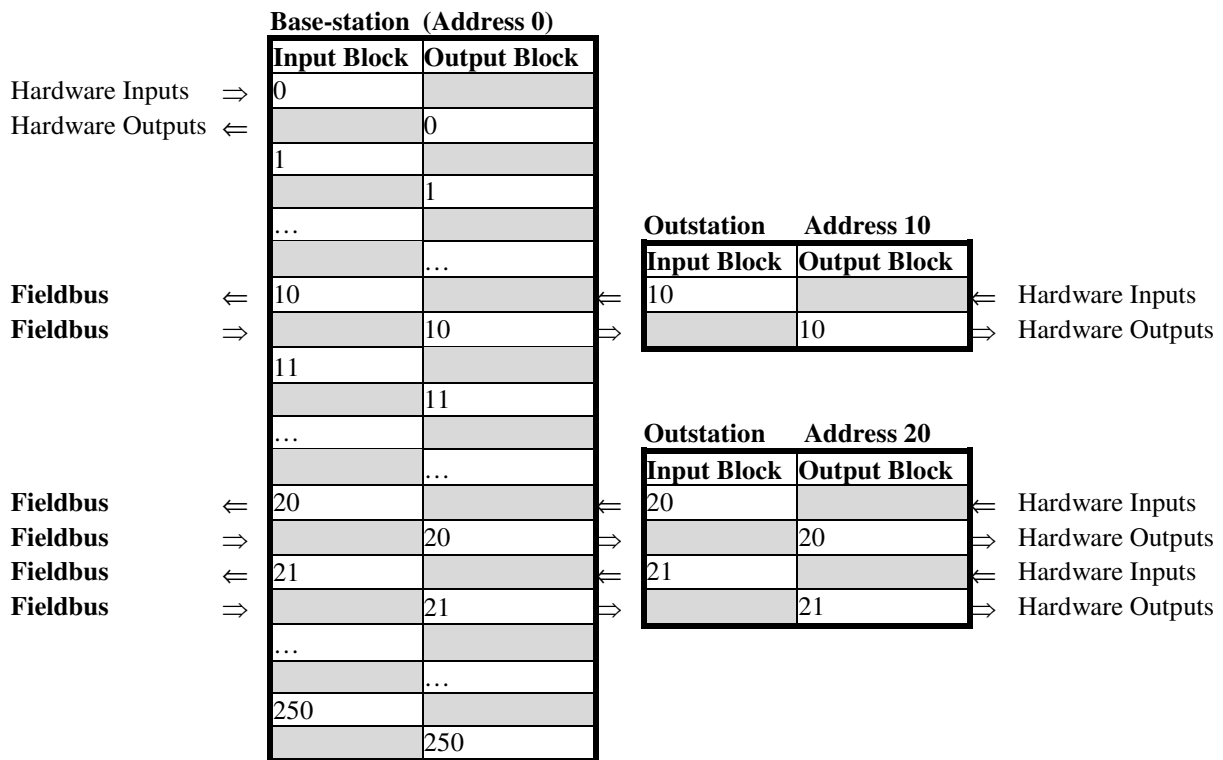
- a) Input Register 172
- b) Input Block 0, Register 172
- c) Outstation Address 21, Battery Volts ( $21 \times 8 + 4$ )
- d) Outstation Address 20, Analogue Input 6 ( $20 \times 8 + 12$ )

Absolute addressing (format (a)) is the format used by PLC's and SCADA systems communicating via serial protocols such as Modbus.

All the other formats identify the register relative to a given start point. Relative addressing is convenient for configuring *Mega\_Link*, since the Root Data Block can be used to identify the relevant outstation address, with the register number identifying the relevant digital or register within the outstation.

Note that the *Mega\_Link* numbering convention always starts from 0. Some PLC protocols (e.g. Modbus) start counting from 1, so the register in the above example would be regarded by some Modbus systems as register 173.

A *Mega\_Link* base-station effectively maintains a complete database of 2000 input registers, 2000 output registers, 8000 digital inputs and 8000 digital outputs. This can be more conveniently viewed as 250 input data blocks and 250 output datablocks. *Mega\_Link* protocol automatically transfers the relevant parts of the base-station database to/from the corresponding areas within the database at each outstation:

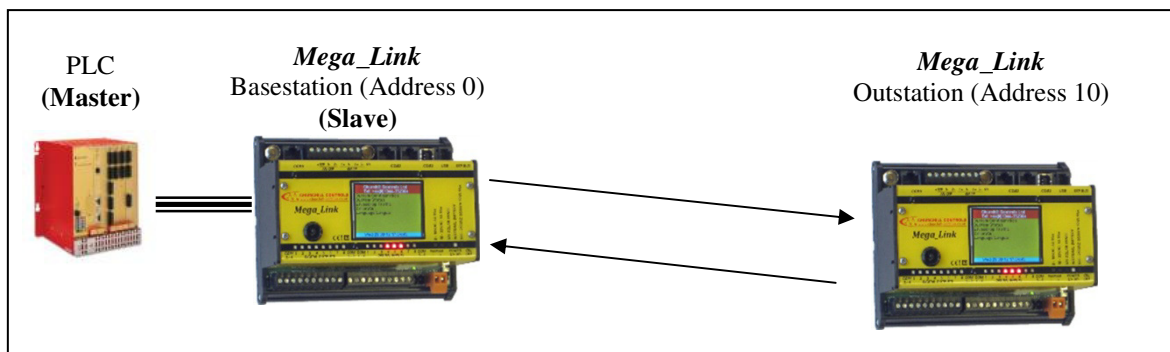


The PLC master will need be configured to identify which data it requires to access, using absolute addressing.

For example, to read outstation 20 battery volts the SCADA system should read Output Register 164 at the base-station ( $20 \times 8 + 4$ ).

Any register or digital can be copied to any destination(s) via **Fieldbus** and/or internal transfers. However, any given output register or digital must only be fed data from one source.

**Example 3:** A PLC master transferring real world inputs & outputs to a remote outstation.

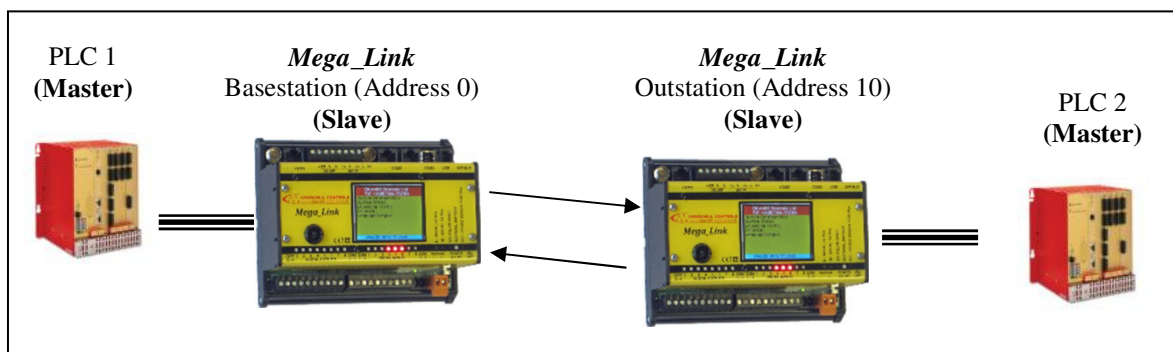


No data routing table is required for the basestation since the PLC Master will initiate all communications, by interrogating the desired registers within the Data\_Link database. Once the basestation sees a request from the PLC, if it is required to get the required information from a remote outstation it will automatically begin polling that outstation. It will continue polling the outstation until it is powered down or its configuration is changed, regardless of whether or not the PLC continues to poll it.

For example, to read Digital Inputs 1...8 at the outstation, the PLC must read digital inputs 328...335 from the basestation. (Since there are 32 digitals per data block, outstation 10 uses digitals 320...351. Inputs 320...327 are alarm flags, so the outstation inputs start at digital 328.)

Similarly, to write to Outstation 10 Analogue Output 1, the PLC must write to output register 86. (Since there are 8 registers per data block, outstation 10 uses registers 80...87. The first 4 registers are totalised counts on the first 4 digital inputs, followed by the outstation battery volts, then the outstation RSSI, then the 2 internal analogue inputs.)

**Example 4:** Two PLC masters transferring register data.



Data routing tables are not needed in either of the *Mega\_Link*'s. Each PLC must copy data to/from registers within the *Mega\_Link* that are not used for hardware I/O. In this example only data blocks 0 and 10 are used by hardware, so the PLC's could theoretically map data to any registers outside of these. However, the normal convention is to map data to the registers immediately following those used by the outstation. This ensures that there is minimal risk of duplication of register allocations if the system is subsequently expanded.

## 5. Comms Fail and Other Alarms

It is usually desirable to have some way of determining whether the Radio (or Line) link between the base-station and outstation is healthy.

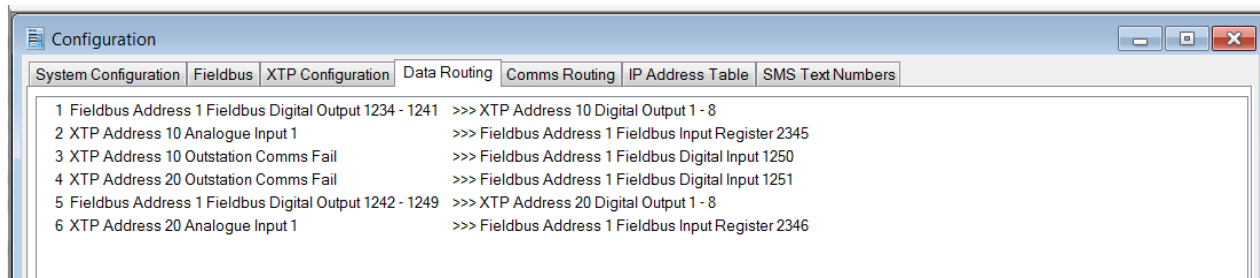
The standard way of achieving this is to map the 'Comm Fail' flags from each outstation to local digital outputs on the base-station. Alternatively, the same flags may be accessed via Fieldbus.

Referring to the data block definitions given above, the alarm flags for each outstation occupy the first 8 digital input registers. For example, to read Comms Fail for Outstation 10, the PLC must read digital input register 320 (Since there are 32 digital per data block, outstation10 starts at 320. Outstation Comms Fail is the first digital in the data block.)

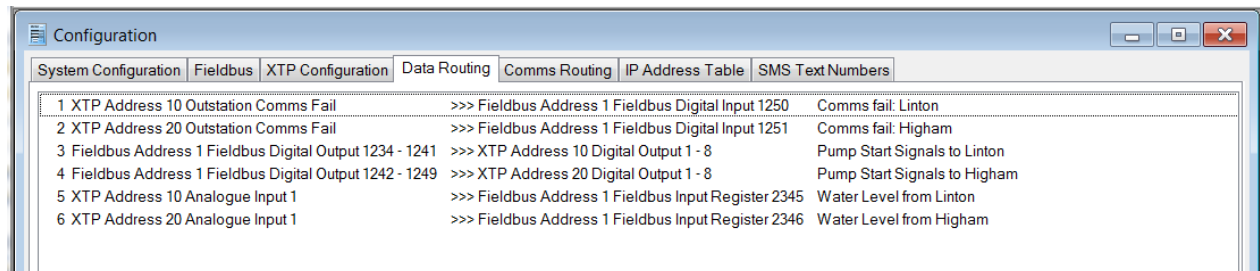
## 6. Data Routing Table Conventions

When compiling a long data routing table consisting of multiple register transfers, it is usually desirable to ensure that the right hand column follows a defined sequence. This makes it immediately apparent if two lines write conflicting data to a given output. It is also good practice to use the description field to annotate each table entry, again for ease of debugging later on.

Below are shown two Data Routing tables. These will both function identically, however the second table is easier to read and manage.



System Configuration	Fieldbus	XTP Configuration	Data Routing	Comms Routing	IP Address Table	SMS Text Numbers
1	Fieldbus Address 1	Fieldbus Digital Output 1234 - 1241	>>> XTP Address 10 Digital Output 1 - 8			
2	XTP Address 10	Analogue Input 1	>>> Fieldbus Address 1	Fieldbus Input Register 2345		
3	XTP Address 10	Outstation Comms Fail	>>> Fieldbus Address 1	Fieldbus Digital Input 1250		
4	XTP Address 20	Outstation Comms Fail	>>> Fieldbus Address 1	Fieldbus Digital Input 1251		
5	Fieldbus Address 1	Fieldbus Digital Output 1242 - 1249	>>> XTP Address 20	Digital Output 1 - 8		
6	XTP Address 20	Analogue Input 1	>>> Fieldbus Address 1	Fieldbus Input Register 2346		

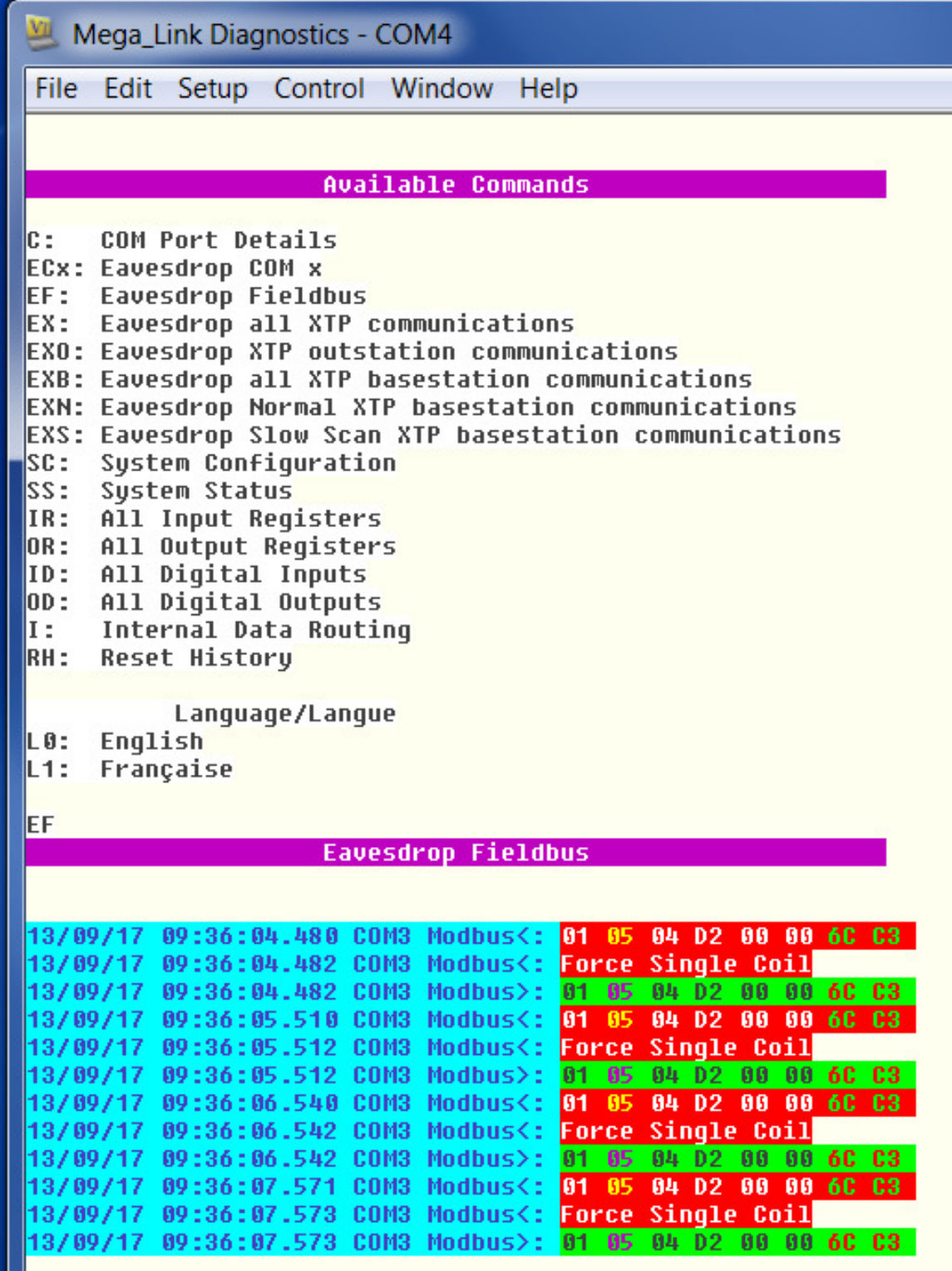


System Configuration	Fieldbus	XTP Configuration	Data Routing	Comms Routing	IP Address Table	SMS Text Numbers
1	XTP Address 10	Outstation Comms Fail	>>> Fieldbus Address 1	Fieldbus Digital Input 1250		Comms fail: Linton
2	XTP Address 20	Outstation Comms Fail	>>> Fieldbus Address 1	Fieldbus Digital Input 1251		Comms fail: Higham
3	Fieldbus Address 1	Fieldbus Digital Output 1234 - 1241	>>> XTP Address 10	Digital Output 1 - 8		Pump Start Signals to Linton
4	Fieldbus Address 1	Fieldbus Digital Output 1242 - 1249	>>> XTP Address 20	Digital Output 1 - 8		Pump Start Signals to Higham
5	XTP Address 10	Analogue Input 1	>>> Fieldbus Address 1	Fieldbus Input Register 2345		Water Level from Linton
6	XTP Address 20	Analogue Input 1	>>> Fieldbus Address 1	Fieldbus Input Register 2346		Water Level from Higham



## 7. Diagnostics

If a PC running DUCX is plugged into the DUCX port on the *Mega\_Link*, the user can start diagnostics by clicking on the button that has a picture of an oscilloscope display and hint "Diagnostic Terminal" to open a terminal emulator. Pressing 'Return' will give a menu and pressing 'EF' followed 'Return' will start *Fieldbus* diagnostics. If all is well the computer should display something like:



```
Mega_Link Diagnostics - COM4
File Edit Setup Control Window Help

Available Commands

C:  COM Port Details
ECx: Eavesdrop COM x
EF: Eavesdrop Fieldbus
EX: Eavesdrop all XTP communications
EX0: Eavesdrop XTP outstation communications
EXB: Eavesdrop all XTP basestation communications
EXN: Eavesdrop Normal XTP basestation communications
EXS: Eavesdrop Slow Scan XTP basestation communications
SC: System Configuration
SS: System Status
IR: All Input Registers
OR: All Output Registers
ID: All Digital Inputs
OD: All Digital Outputs
I:  Internal Data Routing
RH: Reset History

Language/Langue
L0: English
L1: Française

EF

Eavesdrop Fieldbus

13/09/17 09:36:04.480 COM3 Modbus<: 01 05 04 D2 00 00 6C C3
13/09/17 09:36:04.482 COM3 Modbus<: Force Single Coil
13/09/17 09:36:04.482 COM3 Modbus>: 01 05 04 D2 00 00 6C C3
13/09/17 09:36:05.510 COM3 Modbus<: 01 05 04 D2 00 00 6C C3
13/09/17 09:36:05.512 COM3 Modbus<: Force Single Coil
13/09/17 09:36:05.512 COM3 Modbus>: 01 05 04 D2 00 00 6C C3
13/09/17 09:36:06.540 COM3 Modbus<: 01 05 04 D2 00 00 6C C3
13/09/17 09:36:06.542 COM3 Modbus<: Force Single Coil
13/09/17 09:36:06.542 COM3 Modbus>: 01 05 04 D2 00 00 6C C3
13/09/17 09:36:07.571 COM3 Modbus<: 01 05 04 D2 00 00 6C C3
13/09/17 09:36:07.573 COM3 Modbus<: Force Single Coil
13/09/17 09:36:07.573 COM3 Modbus>: 01 05 04 D2 00 00 6C C3
```