

MFT B-Series Serial Communications

The Kurz Instruments MFT B-Series (Mass Flow Transmitter) Models have three modes of data communications using the USB or RS-485 interface. They are available on the MFTB firmware version 1.x or later.

The three communication modes are as follows:

1. Remote Terminal
2. Data Logging
3. Modbus Protocol, ASCII and RTU

The Remote Terminal and Data Logging modes of data communication use a common serial port of the microprocessor therefore only one mode can be used at time.

The Remote Terminal mode is used with the USB interface and requires the Kurz [USB Driver](#) to communicate with a computer terminal (PC or Laptop). This mode is usually used to setup the configuration of the MFT B-Series and to Upload or Download the configuration files.

The [Data Logging mode](#) is used with the USB interface and requires the Kurz USB Driver to log data from the MFT B-Series. Log output is in a comma separated variable (CSV) format for ease of importing into a spreadsheet program and can be setup to log with a periodic time interval or forced to log using the command 'I' (PC) or 'L' (MFT B-Series keypad). It is advisable to turn off the terminal echo to prevent undesirable data from being transmitted.

The Modbus mode is used with RS-485 (point to point or multi-drop). The RS-485 can only be configured as Half Duplex. Modbus is a network communication protocol that was originally developed then released in 1979 by Modicon (*owned by Schneider Electric*) for their controllers. The controllers can communicate using master-slave technique, in which only one device (*the master*) can initiate transactions (*called the queries*). The other devices (*the slaves*) respond by supplying the requested data to the master, or by taking the action requested in the query. The master can address individual slaves, and the slave will respond by returning a message (*called response*) to the query that is individually addressed to it. The Modbus protocol establishes the format for the master's query by placing the following into the query: device address, a function code defining the requested action, any data to be sent and an error-checking field. The slave's response message is also constructed using Modbus protocol. It contains a field confirming the action taken, any data to be returned and an error-checking field. If an error occurred in receipt of the message or the slave is unable to perform the requested action, the slave will construct an error message and send it as its response. The ASCII or RTU transmission modes are available in the MFT B-Series models.

Communication Setup

Setup for Remote Terminal Mode

Remote Terminal communication mode is a default. This allows the user to communicate with the MFT B-Series via a computer. When the communication is established, the MFT B-Series will echo all the information that is being displayed to the local LCD display to the terminal emulator program on the computer. It also allows the user to key in the keyboard command from the computer keyboard like the 'p' ('P' key from the local keypad) for programming, or 'd' ('D' key from the local keypad) for display. See below for more information on allowable inputs. Note that the PC commands must be in lower case and '+' is used to toggle the terminal echo on or off.

Parts of this section are also located in the [Installation section](#) under USB Serial Communications. Also, see [MFT B-Series Screen Diagrams \(324042\)](#) for more information on the menus and allowable inputs.

Any serial communications program may be used to act as a remote terminal to the MFT B-Series. HyperTerminal, which comes with all Windows versions except for Vista, is sufficient but has known issues with correctly opening the port and supports Xmodem for transferring and storing the MFT B-Series configuration files. We recommend using Tera Term 4.62 or newer. This takes about five seconds at 9600 baud, the only supported USB driver baud rate. The character format is 1 start bit, 1 stop, 8 data, and no parity. The program [KzComm](#) may also be used if you only need to save/print the configuration files, but it is not a terminal emulator.

Tera Term, our recommended terminal emulator, is freely available for download at the following websites: Open source version 4.62 at the time of writing can be found at <http://tssh2.sourceforge.jp/> and version 3.13, which is limited to COM ports 1 through 4, can be found at <http://www.ayera.com/teraterm>. The open source version does not have the COM port limitation. Tera Term works with Windows 95 through Vista and has English, Japanese, and German versions. After installing the program configure it as described in the previous paragraph. You may want to put a shortcut to this program in your Start menu and on your desktop. Tera Term also auto configures the port based on the COM ports settings.

If HyperTerminal is not installed on your computer you may add it by going to the Add/Remove Programs within the Control Panel and clicking the Windows Setup tab. Then choose the Communication group and within it you will find the button to select HyperTerminal. After this you may be asked for your Windows disk so this may be installed. Refer to your Windows documentation if you need more information on HyperTerminal. Once the program is installed, you start

HyperTerminal which will prompt you for a name of the connection, e.g. Kurz MFT B-Series, then an icon. Next you select the communications port, e.g. COM 4 and finally you press Configure and set the baud rate to 9600, data bits to 8, parity to none, stop bits to 1 and flow control to none. Once you have verified this works, save the configuration under File, Save.

Windows Vista machines do not include HyperTerminal so you will need to download and install it or another terminal emulator program. HyperTerminal is available for purchase and download from its designer at the following web site: <http://www.hilgraeve.com/htpe>.

Allowable inputs via the terminal emulation program are shown in the following table.

Key	Function
c	Clear the data entry screen or escape one level up on the menu.
c c	Safe escape commanded by pressing c twice. Escape to the next menu item in <i>Program Mode</i> .
d	Command to <i>Display Mode</i> or scroll through Display Mode and Backspace during data entry.
Enter or return	Enter to complete the data entry. (E on the Kurz LCD/keypad)
h	Command to Hold the current display.
h h	Command for Help Mode caused by pressing h twice.
l	Command to Log data to the USB Port.
p	Command to Program Mode, use also to see program data without change.
v	Scroll Down or answer No.
^	Scroll Up or answer Yes.
+	Toggles On/Off Terminal Echo, stops the transmission of data to the remote terminal which would be seen on the Kurz LCD/keypad. This is helpful when doing manual data logging or ASCII commands via the terminal interface (USB port)
-	Dash or Minus sign
.	Period
0	Zero, some menus permit numeric input, If not the keys are ignored.
1	One
2	Two
3	Three
4	Four
5	Five
6	Six
7	Seven
8	Eight
9	Nine

MFT B-Series ASCII Commands

The MFT B-Series allows the user to access internal data remotely using a PC or Laptop computer with a USB Port and Kurz USB Driver ([DCN 451035](#)). With terminal emulation software like HyperTerminal, data can be accessed from the MFT B-Series by using ASCII commands. The format of the command is as follows: start with Escape character (0x1B) followed by the ASCII command and terminated by Return character (0x0D). The response start with Greater than character (0x3E) followed by the data and terminated by Return character (0x0D). On the other hand, the UPLOAD and DOWNLOAD commands respond with a start prompt then begin the XMODEM protocol. The table below is the list of supported ASCII commands. Note that the PC commands must be in lower case and '+' is used to toggle the terminal echo on or off.

ASCII Command	Description
download	Download a configuration file from PC to the MFT B-Series.
upload	Upload the configuration file from the MFT B-Series to the PC.
qvel	Query the Velocity
qflow	Query the Flow Rate
qtemp	Query the Temperature
qmeterid*	Query the Flow Meter's Identification
qsnumber*	Query the Sensor Serial Number
qmeter1*	Query the Meter 1 Display Mode Information
qmeter2*	Query the Meter 2 Display Mode Information
qai1*	Query the External Input Display Mode Information

* These features are only available in firmware versions 1.05 and newer.

The command will work only when the MFT B-Series is in the *Executive Mode*. Refer to the topic "Setup for Remote Terminal Mode" for the setup. Here is an example of how to query the velocity of the MFT B-Series from a PC running with a terminal emulator program. In this example the HyperTerminal emulator program was used. Once the communication between the MFT B-Series and the PC has been established, if the terminal echo is ON (HyperTerminal is receiving data that is displayed by the MFT B-Series) turn it OFF by typing the plus '+' key on the keyboard for a cleaner screen. The terminal echo will stop after the MFT B-Series receives the command. Carefully type in the command because you will not see it displayed on the screen. Start with Escape key followed by **qvel** and terminate with Enter. Once the MFT B-Series receives the command it will respond to the velocity reading at that time. The command and response look like the line below;

Command	Response
<i>EscqvelEnter</i>	1000.00

The format for the qmeter1, qmeter2, and qai1 outputs are in a comma separated variable format.

qmeter1:

flow meter ID, runtime (hrs), flowrate, MUNIT, totalizer, TOTUNIT, elapsed time (min), velocity, VUNIT, ref. density, density unit, flow area, Area unit, correction factor, PRP or IRP, PRP/IRP value, PRP/IRP units, raw flow or velocity, raw flow or velocity unit

qmeter2:

temperature meter ID, runtime (hrs), temperature, TUNIT, correction factor

qai1:

One of the following formats will be used depending on the External Input usage:

current (mA)

current, current units, engineering scaled value, engineering units

THE UPLOAD and DOWNLOAD Commands

These two commands will allow the user to save the configuration of the MFT B-Series to the PC disk drive and update the configuration of the MFT B-Series from the PC using the configuration file. These two ASCII commands are associated with XMODEM protocol to transfer the file. The following are examples of how to save and update the configuration of the MFT B-Series using the UPLOAD and DOWNLOAD command respectively. These examples assume that the MFT B-Series and the PC or Laptop have already established communications and use the HyperTerminal emulation program.

Saving the Configuration to the PC disk drive

If the terminal echo is ON, press the plus '+' key on the keyboard to turn OFF. Type in the command *EscuploadEnter*, the MFT B-Series will respond with a prompt;

>MFT-B Ready to Transmit File

>XMODEM Receive File from MFT-B

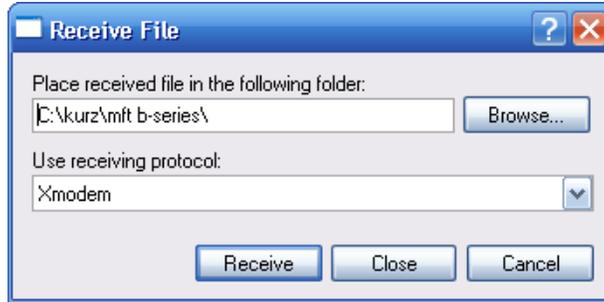


Figure 1: Receive File

From this point, click Transfer on the HyperTerminal menu bar and click Receive File, Figure 1 will be displayed.

Enter or browse to the folder where you want to save the configuration file. Select the Xmodem protocol. Click the Receive button when you are done. A new window, as shown in Figure 2, will be displayed. Enter the filename of the configuration file with a .cf extension (e.g. FT20001A.cf). Press the OK button to continue.

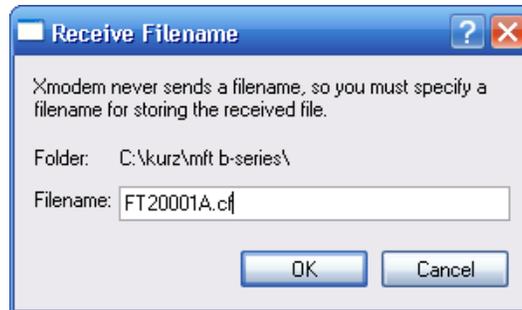


Figure 2: Receive Filename

At this point the Xmodem protocol is receiving the file as seen on the Xmodem activity window shown in Figure 3.

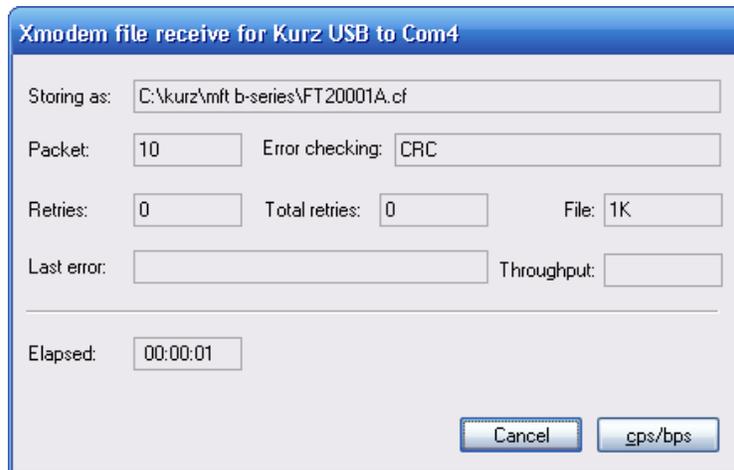


Figure 3: Xmodem Receive Activity Window

Updating the MFT B-Series Configuration from a Configuration File

If the terminal echo is ON, press the plus '+' key on the keyboard to turn OFF. Type the command *EscdownloadEnter*, the MFT B-Series will respond with a prompt;

```
>MFT-B Ready to Receive File  
>XMODEM Transmit File to MFT-B
```

From this point, click Transfer on the HyperTerminal menu bar and click Send File, Figure 4 will be displayed.

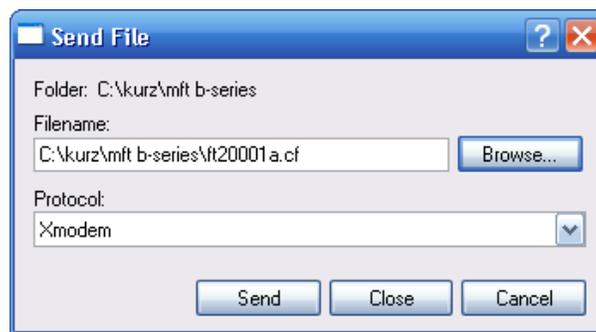


Figure 4: Send File

Enter or browse the filename of the configuration file to be sent and select Xmodem protocol. Click the Send button finished.

Note:

During this process the command may time out and Xmodem can not successfully send the file. If this happens, try it again, this time it will be faster because the folder for your file is already setup.

After the Send button is clicked, the Xmodem activity window will popup, as seen in Figure 5.

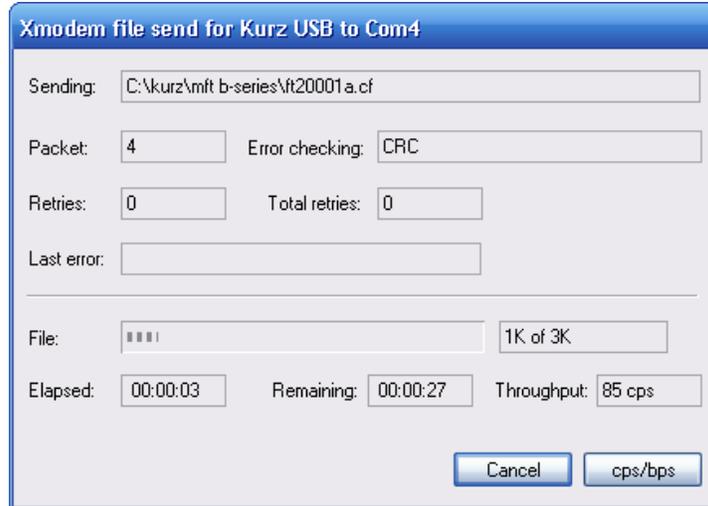


Figure 5: Xmodem Send Activity Window

The other way to do this is to use the KzComm program that is supplied with the MFT B-Series unit. The User's Guide is Kurz Instruments [DCN: 280128](#).

USB to PC Communication Problems

An issue between how windows handles the virtual COM port created by the Kurz USB Device Driver for the PC can cause the terminal emulator program to display gibberish characters. This can sometimes be caused by not disconnecting from the terminal emulator program before disconnecting or powering down the MFT B-Series unit connect to the PC via the USB cable. The following solutions can be used to remedy this problem.

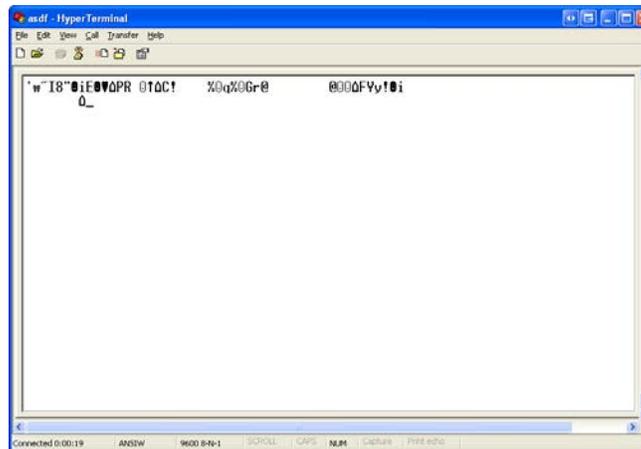


Figure 6: Terminal Emulator Displaying Gibberish

General Notes: Always disconnect from the terminal before disconnecting or powering down the MFT B-Series unit.

KzComm: See Reset XMODEM COM Port in the KzComm Help or Users Guide.

Terminal Emulator:

Solution 1: Use [Tera Term 4.62](#) or latest (Supports COM1 through COM15).

1. Open Tera Term.
2. Connect
 - a. Tera Term only allows connections to COM1 through 15. Use the Device Manager (type **devmgmt.msc** from the **Start->Run** window) to change the COM port to a lower number.
 - b. Only the COM port needs to be chosen.
3. If gibberish is seen then select **Control->Reset** port.
4. Save the setup by selecting **Setup->Save Setup...**
5. Create a shortcut to Tera Term on your desktop.

Solution 2: Use [Tera Term Pro 3.1.3](#) (Only supports COM1 through 4).

See Solution 1, but only supports COM1 through 4.

Solution 3: Use HyperTerminal

1. Setup HyperTerminal with the correct COM port, 9600 baud rate, 8 data bits, no parity, 1 stop bit, and no flow control.
2. **File->Save As** and save it to your desktop.
3. Open the connection. If gibberish disconnect and exit HyperTerminal and repeat until the 3rd failure.
4. If gibberish, disconnect and close HyperTerminal. Open the Device Manager (type **devmgmt.msc** from the **Start->Run** window) to change the COM port baud rate to any other value not previously used and exit the COM properties window. Disconnect the USB from the Computer wait a few seconds and reconnect.
5. Go to Step 3 and repeat until working.

Setup for Modbus Mode

The Modbus communication mode is a default. It can be selected as either Modbus ASCII or Modbus RTU. RTU is set as the factory default. The MFT B-Series must be used only as a slave device and it will work only if used with a master device. RS-485 Interface can be used to communicate with the MFT B-Series via the Modbus protocol. The RS-485 can only be used as Half Duplex and can be used in point to point or multi-drop (1 Master and 1 or more Slaves) operations. Each slave devices must have a unique device address within the network. The individual slave device address can be assigned in the range of 1-247.

WARNING

Do not connect the MFT B-Series to the Modbus Network until it is properly configured.

The MFT B-Series may be setup for Modbus communications. Instructions to setup for Modbus operation are as follows:

1. Press **P** on the local keypad, the MFT B-Series will prompt you to enter the access code.
2. Enter the tech access code, the default is “**654321**”, then press **E** on the local keypad.
3. Press **P**, **^**, or **v** key repeatedly until the “PRESS E TO SET MODBUS COMM” is displayed.
4. Press **E** and enter the slave address (valid range 1-247) and press **E to accept**. If the MFT B-Series will be used in a multi-drop network, make sure that you assigned an address unique to all of the devices connected to the network. The default address set in the factory is 1.
5. Select the “MODBUS ASCII” or “MODBUS RTU” from the selection. Press **E** to accept. The communication parameters for the Modbus ASCII are as follows:

7 Databits
No Parity
2 Stop bits

The communication parameters for the Modbus RTU are as follows:

8 Databits
No Parity
1 Stop bit

6. Using **^** or **v** choose the baud rate. Press **E** to select the displayed baud rate. The factory default is 38400 baud.
7. Use **^** or **v** to choose the byte order. Press **E** to select the displayed byte order. The factory default is 1 2 3 4.
8. Press **C** to return to the *Executive Mode*.
9. Refer to the MFT B-Series Field Wiring [Diagram](#).

Note: If trouble occurs with reading floating point numbers try changing the byte order.

MFT B-Series Modbus Command and Registers

The MFT B-Series firmware version 1.00 and later support the reading and writing of coils, and registers. Firmware version 1.03 and later support device identification. Coils are represented by a single bit. A register is defined as a 16-bit storage of data. This means that a 16 bit integer number will occupy one register, 32 bit floating point numbers will occupy two registers, and two ASCII 8 bit characters are stored in one register. Note that registers with ASCII characters are represented as a char array and need to be null character terminated. The MFT (A-Series) supported reading data via modbus, but only using Function 03 -Read Hold Registers. That data can now be read with the MFT B-Series using the Function 04 – Read Input Registers.

Modbus functions operate on memory mapped to registers. Modbus registers are organized into the following reference types identified by the leading number of the reference address:

Modbus Register Reference Address	Description
0xxxx	<u>Read/Write Discrete Outputs or Coils.</u> A 0x reference address is used to drive output data to a digital output channel.
1xxxx	<u>Read Discrete Inputs.</u> The ON/OFF status of a 1x reference is controlled by the corresponding digital input.
3xxxx	<u>Read Input Registers.</u> A 3x reference register contains a 16-bit number received from an external source – e.g. an analog signal.
4xxxx	<u>Read/Write Output or Holding Registers.</u> A 4x register is used to store 16-bits of numerical data (binary or decimal), or to send data from the CPU to an output channel.

Register Map Notes:

1. The “x” following the leading character represents a four-digit address location in memory. The leading character is generally implied by the function code and omitted from the address specifier for a given function. The leading character also identifies the I/O data type.
2. The ON/OFF state of discrete inputs and outputs is represented by a 1 or 0 value assigned to an individual bit in the 16-bit data word. With respect to mapping, the LSB of the word maps to the lowest numbered coil of a group and coils numbers increase sequentially as you move towards the MSB. Unused bits are set to zero.

Function 05 (0x05) – Write Single Coil Function 01 (0x01) – Read Coils

Modbus Register Reference Address 0xxxx	
Modbus Coil	Coil Description
0	Start the Zero Drift Check
1	Start the Mid-Span Drift Check
2	Start the Span Drift Check
3	Start the Drift Check Cycle
4	Abort On-Going Drift Check
5-7	Reserved
8	Start a Purge (Requires Purge Option)

Function 02 (0x02) – Read Discrete Inputs	
Modbus Register Reference Address 1xxxx	
Modbus Coil	Coil Description
0	Zero Drift Check started
1	Mid-Span Drift Check started
2	Span Drift Check started
3	Drift Check Cycle started
4-7	Reserved
8	Purge Cycle started (Requires Purge Option)
9-15	Reserved
16	Error - Rp Resistance Above High Limit
17	Error - Rp Resistance Below Low Limit
18	Error - Rtc Resistance Above High Limit
19	Error - Rtc Resistance Below Low Limit
20	Error - Wire Loop Resistance Above High Limit
21	Error - Rps Sensor Lead Open Circuit
22	Error - High Sensor or Wire Leakage
23	Error – Flow Rate Above Design Limit
24	Error - Meter Kick Out High
25	Error - Meter Kick Out Low
26	Error - ADC Failed to Convert Measurement
27	Error - Sensor Control Drive Stopped Responding
28	Error - Sensor Over-Voltage Crowbar Engaged
29	Error - Sensor Type Does Not Match Configuration
30	Error - Abnormal Sensor Node Voltages
31	Error - Unable to Write the Configuration File to the EEPROM
32	Error – Sensor Type Does Not Match Board Build ²
33	RESERVED ²
34	RESERVED ²
35	RESERVED ²
36	RESERVED ²

37	RESERVED ²
38	RESERVED ²
39	RESERVED ²
40	RESERVED ²
41	RESERVED ²
42	RESERVED ²
43	RESERVED ²
44	RESERVED ²
45	Sensor Leakage Warning: S-Gnd below 100 k ³
46	Logging Power On Event ²
47	Logging Change Made to the Configuration Event ²
48	Alarm 1 Triggered ¹
49	Alarm 2 Triggered ¹

¹ These features are only available in firmware versions 1.05 and newer.

² These features are only available in firmware versions 1.20 and newer.

³ These features are only available in firmware versions 1.10 and newer and have a factory configuration option to allow operation up to 600 °C for the FD2 Sensor and the warning code may be followed by the error at coil 22.

Function 04 (0x04) – Read Input Registers		
Modbus Reference Register Address 3xxxx		
Modbus Register	Register Description	Data Type C-Compilers
0-1	Flow Rate	Float
2-3	Velocity	Float
4-5	Temperature	Float
6-7	Total Flow	Float
8-9	Elapsed Time	Float
10-11	Flow Rate Correction Factor	Float
12-13	Temperature Correction Factor	Float
14-15	Density	Float
16-20	Serial Number	ASCII (char)
21-23	Velocity Unit	ASCII (char)
24-26	Flow Rate Unit	ASCII (char)
27-29	Total Flow Unit	ASCII (char)
30-32	Temperature Unit	ASCII (char)
33-34	Sensor Rp Current (IRP)	Float
35-36	Sensor Rp Power (PRP)	Float
37-38	Electronic Temperature	Float
39-40	Voltage Input for Zero Drift Check	Float
41-42	Voltage Output for Zero Drift Check	Float
43-44	Percent Difference of the Zero Drift Check	Float
45-46	Voltage Input for Mid Drift Check	Float
47-48	Voltage Output for Mid Drift Check	Float
49-50	Percent Difference of the Mid Drift Check	Float
51-52	Voltage Input for Span Drift Check	Float
53-54	Voltage Output for Span Drift Check	Float
55-56	Percent Difference of the Span Drift Check	Float
57-58	Current Runtime Counter (sec)*	Unsigned 32 bit Integer
59-60	AO1 Current (mA)*	Float
61-62	AO2 Current (mA)*	Float

*These features are only available in firmware versions 1.05 and newer.

Function 03 (0x03) – Read Holding Registers		
Function 06 (0x06) – Write Single Register		
Modbus Reference Register Address 4xxxx		
Modbus Register	Register Description	Data Type C-Compilers
0-5	Reserved	
6-7	Flow Area	Float
8-14	Flow Meter Identification	ASCII (char)
15-21	Temperature Meter Identification	ASCII (char)

22-23	Analog Out 1 - 4 mA Scale	Float
24-25	Analog Out 1 - 20 mA Scale	Float
26-27	Analog Out 2 - 4 mA Scale	Float
28-29	Analog Out 2 - 20 mA Scale	Float
30	Purge Width (ms)	Unsigned Integer
31	Purge Hold Mask (ms)	Unsigned Integer
32-33	Purge Interval (min)	Unsigned 32 bit Integer
34-35	Drift Check Zero Scale Value (%)	Float
36-37	Drift Check Mid Scale Value (%)	Float
38-39	Drift Check Span Scale Value (%)	Float
40	Drift Check Zero Check Duration (sec)	Unsigned Integer
41	Drift Check Mid Check Duration (sec)	Unsigned Integer
42	Drift Check Span Check Duration (sec)	Unsigned Integer
43	Drift Check Time Interval (hrs)	Unsigned Integer
44-45	PID Reference Data	Float

Read Device Identification			
Function 43 (0x2B) – Encapsulated Interface Transport			
MEI Type 14 (0x0E) – MODBUS Encapsulated Interface (MEI)			
Object Id	Object Name / Description	Type	Category
0x00	Vendor Name	ASCII String	Basic - Mandatory
0x01	Product Code	ASCII String	Basic - Mandatory
0x02	Major Minor Revision	ASCII String	Basic - Mandatory

Data Formats for Kurz Floating Point, Straight (Daniels Extensions) and Classic Modicon Modbus

Kurz has implemented the straight binary mapping of the 32 bit floating point variables into the Modbus registers. The Kurz byte order is 1 2 3 4. This method may also be known as Reverse-32 on some systems as the classic Modbus order uses 3 4 1 2 as the byte order. When viewing the Kurz variables like

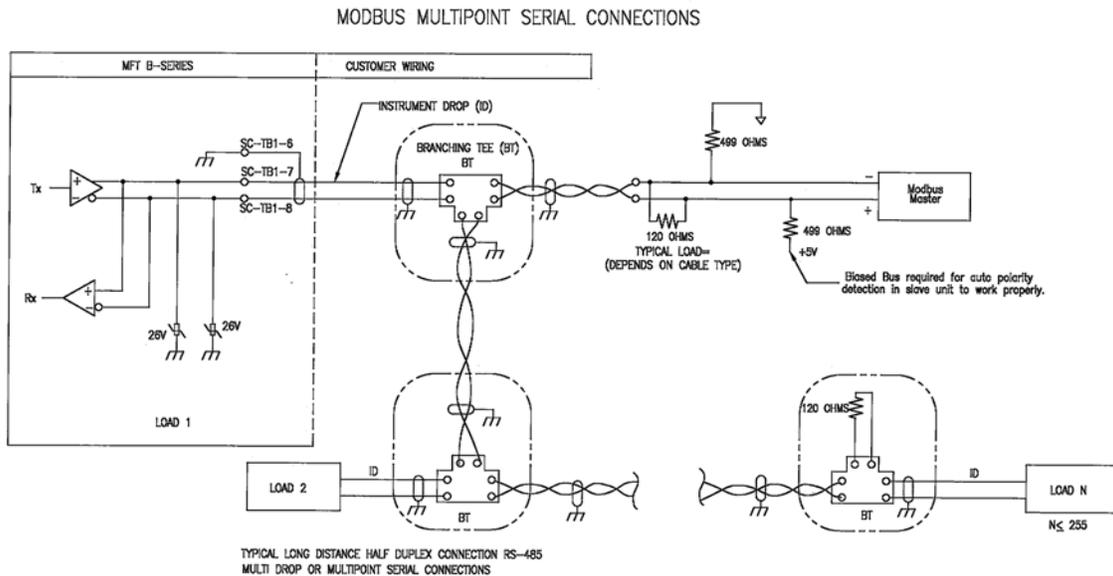
Temperature and Flow, if the numbers do not look normal, try the other formats for interpreting the data registers.

Modbus Biasing

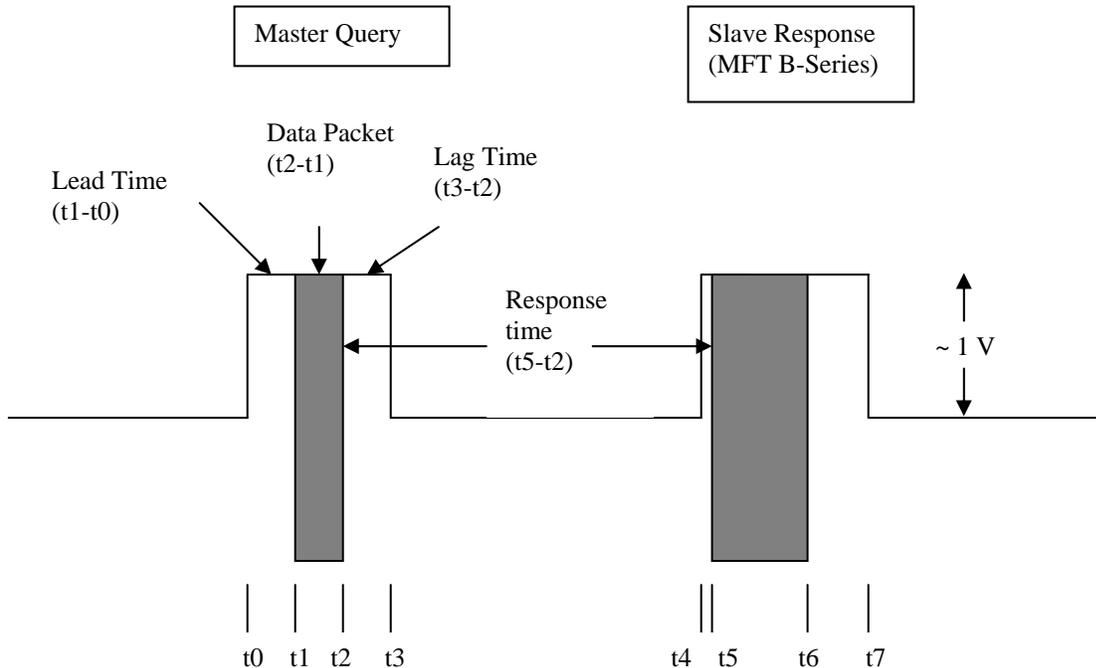
The MFT B-Series slave device will auto detect the polarity of its wiring and correct a polarity error in wiring, but it requires a biased bus for this functionality to work properly. See the [MFT B-Series Field Wiring Diagram](#) for a larger version of the following diagram.

Modbus ASCII Compatibility Issues

The MFT B-Series MODBUS setup for ASCII transmission framing is not supported by [KzComm](#). If KzComm is to be used over MODBUS, then the RTU transmission framing must be used.



Half-Duplex Modbus Serial Communications Example on an RS-485 Bus



The Modbus master issues a command with the slave responding. Note that between the transmissions we have no signal. RS-485 is a tri-state (logic low, high and off). The delay between asserting the RS-485 bus and the serial data is known as the lead time (t_1-t_0) or (t_5-t_4). The data packet length (t_2-t_1) or (t_6-t_5) depends on the message and the baud rate. The delay between the packet and release of the RS-485 bus is the lag time (t_3-t_2) or (t_7-t_6). The delay between the master and slave packets is the response time (t_5-t_2).

MFT B-Series Specifications (RTU protocol, 38.4 kbits/s)

Lead time:	184 us
Lag time:	14 ms max
Response time:	18 ms max

Many of the master devices are programmable for their baud rate, silent interval, timeout times, number of retries, framing, and protocol. Note that if the master lag time is too long, this transmission will collide with the slave response and will increase the data transmission errors.

Kurz recommends the following values:

Baud Rate:	38.4 kbaud
Silent Interval:	35 ms
Timeout Interval:	100 ms
Number of Retries:	2
Framing:	Remote Terminal Unit (RTU)

Because of these lead/lag and response times, the MFT B-Series is limited to approximately 15 transactions/sec at 38.4 kbaud. For large Multipoint systems this still allows a complete 12 point duct measurement to be made in under a

second. Because the CPU in the MFT B-Series is used for many different functions, it may not always be ready to respond to a command from the Modbus master. This is a common limitation for Modbus devices. For this reason it is recommended that you pole it multiple times or just skip this address if it does not respond during an address scan of the devices on the Modbus network. Many of the Modbus drivers are designed to pole a device multiple times or just skip it on a polling cycle and not report an error unless it does not respond after X tries.