FLUKE®

Fluke ScopeMeter® 125 helps solve a Modbus RS-485 timing problem

Application Note

Testing Functions
Case Study

Who: Ken Roach, Technical Consultant, Rockwell Automation

What: Fluke 125 ScopeMeter® Test Tool

Tests: Modbus RS-485 health test

Ken Roach is a technical consultant for Rockwell Automation. He works from the company's Seattle office and supports that office's sales engineers and managers in the installation of Rockwell Automation's advanced products. "They know how to sell them. They know how to price them. I know how to plug them in and make them work," Roach explains.

Roach recently solved a perplexing Modbus RS-485 communications-timing problem using a Fluke 125 ScopeMeter test tool. The Fluke 125 is a battery-powered, three-in-one instrument that combines a 40 MHz digital storage oscilloscope, two true-RMS digital multimeters and a dual input TrendPlot™ recorder. In particular, it includes an automatic network health test feature. "I often need to get down to the signal level on communication networks and take a close look at the bits and bytes that are going by on the electronic level", Roach reveals. "That's where we encountered the Fluke 125 and its features for troubleshooting industrial control and communications systems."

The problem

Roach describes the situation he faced this way: "I had an Allen-Bradley MicroLogix 1100 Controller (PLC) connected as a Modbus RTU (remote terminal unit) master to a compact third-party thermistor monitoring block. Troubleshooting should have been easy because the monitoring block has a three-wire RS-485 interface and uses totally normal Modbus Function 03 (Holding Register Read) functions for all the data."

The MicroLogix 1100 was equipped with a 1763-NC01 cable that connected its isolated three-wire RS-485 port to a terminal block. One hundred-twenty-five feet of Belden 9841 shielded cable was then connected to the thermistor monitoring block.

Connected through a 1716-NET-AIC isolator (an RS-232 to RS-485 signal converter), the PLC software utility that comes with the thermistor block could communicate with the block, but the MicroLogix 1100 could not. "We were certain that the Modbus addressing parameters were right in RSLogix 500 (the PLC programming software), so we tried a Modbus RTU master simulator (ModSim32). It worked fine," Roach reveals.

Next, Roach connected a passive, binary, serial analyzer, and the analyzer showed absolutely nothing wrong. "It showed, byte one, byte two, byte three, byte four, byte five…," Roach says. "The device was responding correctly, within the protocol, to the MicroLogix RTU master poll (request for information)." Still, the PLC continued to generate a "Timeout" error on the message instruction. In other words, the poll was received by the thermistor monitoring block, but the PLC was timing out during the response.

The solution

"My usual serial analysis tools didn't do the job," Roach explains, "so I turned to the Fluke 125. What it revealed that the serial analyzer had failed to reveal was a relatively long pause—about 7 milli-seconds—between byte three and everything else. It was something that the customer pointed to right away and said, 'What's that?'" (See Figure 1.)

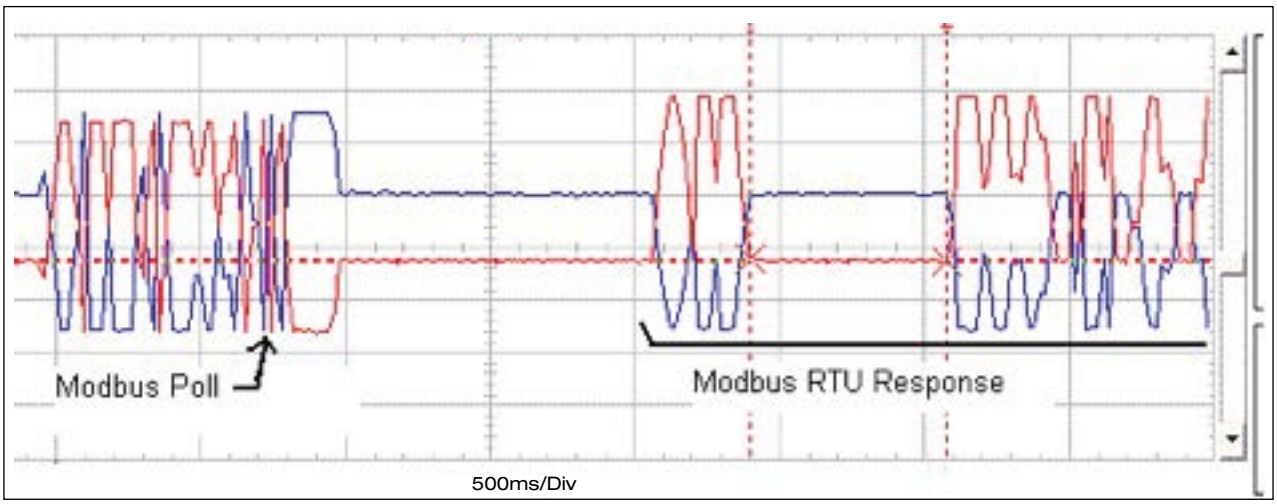

Figure 1. This data capture at 5 ms/div reveals both the poll and the response. Notice the delay of approximately 7 ms in the response.

Next Roach set the Fluke 125 to capture the waveform at 500 microseconds per division. This allowed a resolution at which he could see each individual byte going by and was able to capture the three bytes that were the header of the Modbus packet. (See Figure 2.)

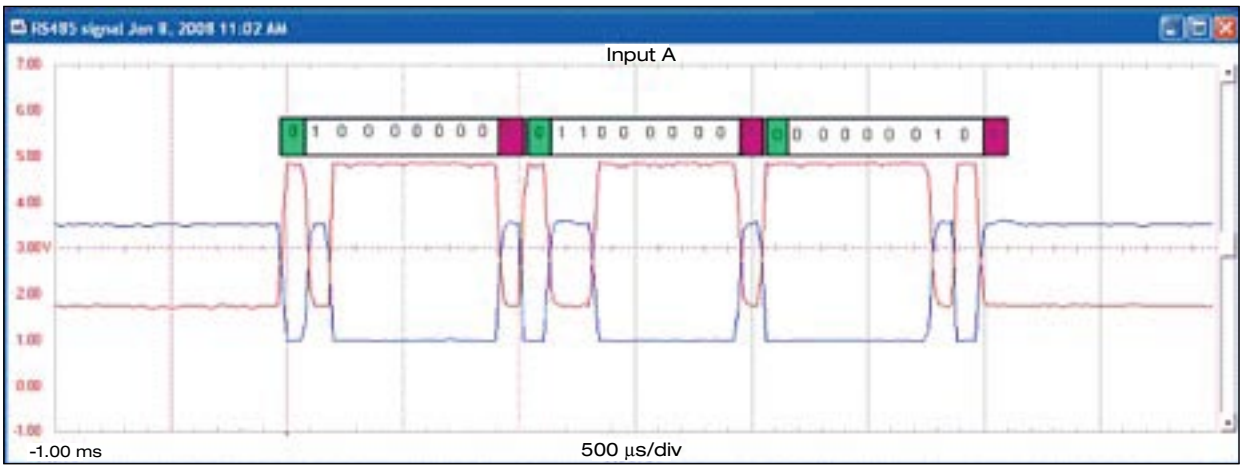

Figure 2. The data capture of the first three bytes of the Modbus RTU response at 500 µs/div. This representation includes a decoding of the response signal to clarify the slave node address, the function code and response data length in bytes (represented by the binary logic manually superimposed above the waveform).

FLUKE®

Conclusion

A Modbus RTU poll response begins with the slave node address, the function code, and the response data length in bytes. Typically, these data are followed by the response data and the CRC checksum. A close look at the data packet in Figure 2 allows one to see that this first short data packet consisted of the values 1, 3, and 64. That is the correct response before the actual data values for the polling of Modbus Node 1 with function code 3 for 32 words (64 bytes).

Roach conjectures that the thermistor monitoring block lacked sufficient CPU power to make a complete response that included response data and the CRC checksum, without pausing to "catch its breath." If that is the case, the little device might respond to the Modbus RTU poll with the first three bytes, then pause seven milliseconds before it sends the data itself.

Roach notes that the Modbus RTU specification says that any idle space of more than 3.5 byte widths is considered the end of the frame and the beginning of another. Clearly, the response was timing out.

Roach further explains that the MicroLogix serial port in Modbus RTU master mode has a settable inter-character time value. The default value of zero applies the Modbus RTU 3.5 byte time to the port. "We adjusted this value," he reveals, "and found that a value of 10 milliseconds made communications between the MicroLogix 1100 and the analog block device work perfectly."

In summary, Roach says, "I was pretty sure I could solve almost any Modbus RTU or DF1* issue with my serial analyzers, but their timestamps aren't accurate enough to find this kind of delay inside a serial data string. The Fluke 125, which did a good job showing the serial rate as well as the max-min waveform values, was an excellent tool for trouble-shooting this situation."

* DF1 is a Serial Communications Protocol used by most of Rockwell Automation's (Allen-Bradley) programmable controllers.

Bytes, bits and waveforms

In Figure 2, the waveform at 500 microseconds per division, the oscilloscope displayed left-to-right the sequence in which these bytes were coming down the wire. What an analyst must remember, especially an analyst who is not completely conversant in how bytes and bits tie to waveforms, is that the usual way of decoding these things on paper is to write "the least significant bit" on the right of the page. However, when looking at a byte on the oscilloscope, remember that it's the other way around—left to right.

In addition, one must understand also "serial framing." Even though a byte that comes across a serial line has only eight bits of data, there are actually ten bits on the line because there's a stop bit and a start bit. That's easy to forget, because there is no space or time between the stop and start bits.

Fluke. *Keeping your world up and running.®*

Fluke Corporation
PO Box 9090, Everett, WA 98206 U.S.A.

Fluke Europe B.V.
PO Box 1186, 5602 BD
Eindhoven, The Netherlands

For more information call:
In the U.S.A. (800) 443-5853 or
Fax (425) 446-5116
In Europe/M-East/Africa +31 (0) 40 2675 200 or
Fax +31 (0) 40 2675 222
In Canada (800)-36-FLUKE or
Fax (905) 890-6866
From other countries +1 (425) 446-5500 or
Fax +1 (425) 446-5116
Web access: http://www.fluke.com

Printed in U.S.A. 3/2008 3306965 A-EN-N Rev A