Live Training Handout – Modpoll

Introduction

Modpoll allows you to test modbus devices. The documentation can be found on the producer's website. This document intends to provide some information related to the use with VPInstruments products. To save you time, finding the right commands…

Download

https://www.modbusdriver.com/modpoll.html

Usage

Start modpoll with --help : and get this:

```
Usage: modpoll [OPTIONS] SERIALPORT|HOST [WRITEVALUES...]
Arguments:
SERIALPORT    Serial port when using Modbus ASCII or Modbus RTU protocol
              COM1, COM2 ...                on Windows
              /dev/ttyS0, /dev/ttyS1 ...    on Linux
HOST          Host name or dotted IP address when using MDBUS/TCP protocol
WRITEVALUES   List of values to be written. If none specified (default) modpoll reads data.
General options:
-m ascii      Modbus ASCII protocol
-m rtu        Modbus RTU protocol (default if SERIALPORT contains a /)
-m tcp        MODBUS/TCP protocol (default otherwise)
-m udp        MODBUS UDP
-m enc        Encapsulated Modbus RTU over TCP
-a #          Slave address (1-255 for serial, 0-255 for TCP, 1 is default)
-r #          Start reference (1-65536, 100 is default)
-c #          Number of values to read (1-125, 1 is default)
-t 0          Discrete output (coil) data type
-t 1          Discrete input data type
-t 3          16-bit input register data type
-t 3:hex      16-bit input register data type with hex display
-t 3:int      32-bit integer data type in input register table
-t 3:mod      32-bit module 10000 data type in input register table
-t 3:float    32-bit float data type in input register table
-t 4          16-bit output (holding) register data type (default)
-t 4:hex      16-bit output (holding) register data type with hex display
-t 4:int      32-bit integer data type in output (holding) register table
-t 4:mod      32-bit module 10000 type in output (holding) register table
-t 4:float    32-bit float data type in output (holding) register table
-i            Slave operates on big-endian 32-bit integers
-f            Slave operates on big-endian 32-bit floats
-e            Use Daniel/Enron single register 32-bit mode (implies -i and -f)
-0            First reference is 0 (PDU addressing) instead 1
-1            Poll only once only, otherwise every poll rate interval
-l #          Poll rate in ms, (1000 is default)
-o #          Time-out in seconds (0.01 - 10.0, 1.0 s is default)
Options for MODBUS/TCP, UDP and RTU over TCP:
-p #          IP protocol port number (502 is default)
Options for Modbus ASCII and Modbus RTU:
-b #          Baudrate (e.g. 9600, 19200, ...) (19200 is default)
-d #          Databits (7 or 8 for ASCII protocol, 8 for RTU)
-s #          Stopbits (1 or 2, 1 is default)
-p none       No parity
-p even       Even parity (default)
-p odd        Odd parity
-4 #          RS-485 mode, RTS on while transmitting and another # ms after
```

Examples

Device specific examples should be documented in the page for the device itself, but here some easy examples for understanding the modbus command line. Speces between the argument and the value does not seem to matter.. -pnone or -p none seems to work both.

Read temperature from VPFlowScope:

```
#legacy device from linux (temperature from VPFlowScope in-line)
./modpoll -b38400 -pnone -m rtu -a 3 -r73 -t 4:float /dev/ttyS0
```

Read Dew Point from VP Dew Point Sensor:

```
Read dew point sensor on a linux system:

./modpoll -b19200 -pnone -m rtu -a 240 -r7 -t 4:float /dev/ttyS0
```