ModbusIP Slave for SoftPLC® Runtime

Version 1.2

Table of Contents

Chapter 1. Overview

1.1. Introduction

This document describes the intallation, usage, and functionality of a TOPDOC Loadable Module **(TLM)** for SoftPLC version 4.x and later. The TLM implements the slave (or server) side of the Modbus TCP protocol. As an extention of the standard, it also implements the same protocol on UDP/IP.

It may be used to talk to an MMI Graphical User Interface station or it may be used to talk between other controllers on an ethernet.

1.2. Concepts

The SoftPLC runtime software supports TLMs, which are shared library extensions to SoftPLC. A TLM may be loaded either as a **DRIVER** or as a **MODULE**. The difference between a DRIVER and a MODULE is that a DRIVER is called once per SoftPLC scan, and optionally an additional number of times per scan. A MODULE is only called when the control program decides to call it. TLMs are made known to SoftPLC in the MODULES.LST file which may be edited by TOPDOC NexGen by traversing to: PLC | Modules.

1.3. Features

In order to use the modbus IP slave TLM you need a working ethernet connection. This TLM listens on the standard TCP and UDP port 502 and acts as a "slave" or a "server". On the other end of any Modbus conversation is a "master" or a "client". Because the slave implemented by this TLM supports both TCP and UDP carriers for the Modbus protocol, the master may choose which it will use. More than one master may talk to this slave at once, and each master may chose independently to do this on either TCP or UDP, or both.

When TCP is used by the master, there is a TCP connection established. When UDP is used by the master, the request response sequence takes place in a "connection-less" fashion. Because UDP is not a guaranteed delivery service, any Modbus master using UDP instead of TCP to carry the modbus requests and responses should implement timeout and retry logic. On most hardware platforms, SoftPLC will respond to a request within a millisecond or two, and knowledge of this can be used to pick reasonable timeouts for the required UDP logic.

This slave TLM supports 32 simultaneous Modbus/TCP connections and an unlimited number of Modbus/UDP sessions. SoftPLC 4.x's integrated firewall may be used to restrict which network nodes can communicate to this TLM.

SoftPLC also provides a Modbus IP Master TLM, which is documented here. A single SoftPLC machine can be both a master and a slave. This capability gives the systems designer the power and flexibility to develop very powerful, fast and flexible distributed control systems. Obviously a SoftPLC Modbus master can talk to a SoftPLC Modbus slave as well as third party slaves.

1.4. Requirements

- A working ethernet or PPP link.
- Version 4.x SoftPLC or later.

Chapter 2. Terms of Use

Because of the variety of uses of the information described in this manual, the users of, and those responsible for applying this information must satisfy themselves as to the acceptability of each application and use of the information. In no event will SoftPLC Corporation be responsible or liable for its use, nor for any infringements of patents or other rights of third parties which may result from its use.

SOFTPLC CORPORATION MAKES NO REPRESENTATIONS OR WARRANTIES WITH RESPECT TO THE CONTENTS HEREOF AND SPECIFICALLY DISCLAIMS ANY IMPLIED WARRANTIES OF MERCHANTABILITY OR FITNESS FOR ANY PARTICULAR PURPOSE.

SoftPLC Corporation reserves the right to change product specifications at any time without notice. No part of this document may be reproduced by any means, nor translated, nor transmitted to any magnetic medium without the written consent of SoftPLC Corporation.

SoftPLC, and TOPDOC are registered trademarks of SoftPLC Corporation.

First Printing March, 2005

Latest Printing June, 2005

SoftPLC Corporation 25603 Red Brangus Drive Spicewood, Texas 78669
USA Telephone: 1-800-SoftPLC

Fax: 512/264-8399
URL: http://softplc.com
Email: support@softplc.com

Chapter 3. Installation

The TLM is named mbipslav.tlm.so and is found as part of the standard SoftPLC 4.x installation in the /SoftPLC/tlm directory. To use it you merely have to enable it in NexGen's PLC | MODULES editor. Then you must edit the text file MBIPSLAV.LST which is the TLM's configuration file.

It is easy to edit the configuration file from the PLC | MODULES editor as well. Simply click on the "Configure" button after selecting and enabling the Modbus IP Slave TLM there.

Chapter 4. Configuration

Modbus commands were originally designed for a Modicon PLC. Therefore they assume 4 different types of memory regions. Words within these memory addresses are addressed using **Reference Numbers**, according to the following table. Basically, the first character of a reference number gives its region:

Table 1. Memory Regions and Reference Numbers

Memory Region	Reference Number Format
Input Discrete (boolean inputs)	1... e.g. 120438
Input Registers (16 bit words)	3...
Output Coils (boolean outputs)	0...
Output Registers (16 bit words)	4...

In order to make use of the Modbus commands, we must map the 4 required memory regions to datatable files within SoftPLC.

All the configuration is done in the single text file MPIPSLAV.LST. Below is a sample. Any part of a line of text to the right of a semicolon **(;)** is considered a comment and is ignored.

4.1. Sample Configuration File

Sample MBIPSLAV.LST

```
; This is the configuration file for the MODBUS IP Slave TLM.
; Anything after a semicolon is ignored during parsing.
; There are two sections: [DRIVER] and [SLAVES]:

; [DRIVER] contains global options, such as DEBUG and IOCHECK
;
; Set DEBUG to > 0 if you want diagnostic output temporarily.
;  DEBUG=0 gives no diagnostic output.
;  DEBUG=1 gives a nice Query Response trace.
;  DEBUG=2 gives what 1 does and more.
;
; Set IOCHECK to YES if you want the turn around time to be reduced by
;  having SoftPLC service the modbus commands more frequently than once per scan.
;  This is effective only when you have more rungs than STARTUP.LST's
;  oneCheckInterval setting.

[DRIVER]
DEBUG=0
IOCHECK=NO


; [SLAVES] should include one line for each group of files you want to expose.
; Each SLAVE is a ficticious internal Modicon device that is addressed by
```

```
; including the corresponding "Slave ID" in the modbus request packet.
;
; Each SLAVE line can provide access to up to 4 datatable files, one for each
; of these purposes:
; 1) Holding Registers, 2) Output Coils, 3) Input Registers, 4) Input States.
; Any or all of the 4 files can be the same as each other, for each SLAVE.
; The given files must start at element 0 and be of type INTEGER, BIT, INPUT,
; OUTPUT, or STATUS.
;
; A SLAVE with a SlaveID of -1 is special.  It will be used when the incoming
; request packets do not match any other SlaveID.  In many cases, you will
; only need a single SLAVE row and will want to use -1 as the SlaveID.

[SLAVES]
;SlaveID 0-255
;|  Holding Registers StartAddress:
;|  |        Output Coils StartAddress
;|  |        |        Input Registers StartAddress
;|  |        |        |         Input States StartAddress
;|  |        |        |         |
-1, N12:0,   N12:0,   N12:0,    N12:0

;0, N100:0,  N101:0,  N102:0,   N103:0
;2, N100:0,  B3:0,    N7:0,     I:0     spare, edit/copy, uncomment as needed

; Make sure you use TOPDOC to create the Datatable Files that you are
; referencing for each slave.

; Make sure you install this TLM as a DRIVER and not a MODULE.

; FIREWALL:
; If you want to restrict which network nodes can access this
; SoftPLC's MODBUS IP SLAVE functionality, then you should setup the internal
; Gatecraft Firewall with help from SoftPLC Corp.

;EOF
```

Modbus TCP protocol includes the original "slave id" field which was part of the modbus on serial line protocol. In the case of communications on ethernet, either via TCP or UDP, this field is no longer used to qualify the actual network node that will respond to a request. The reason for this is because the **IP Address** in the ethernet frame serves this purpose. Therefore the "slave id" field can be configured to use the slave id field as a "selector" in the mapping of the four Modbus memory regions to SoftPLC datatable files.

You may supply a single SLAVE row that has a -1 as the SlaveID. A SlaveID of -1 is special. This row, if supplied, is the default when the incoming SlaveID does not match any other row.

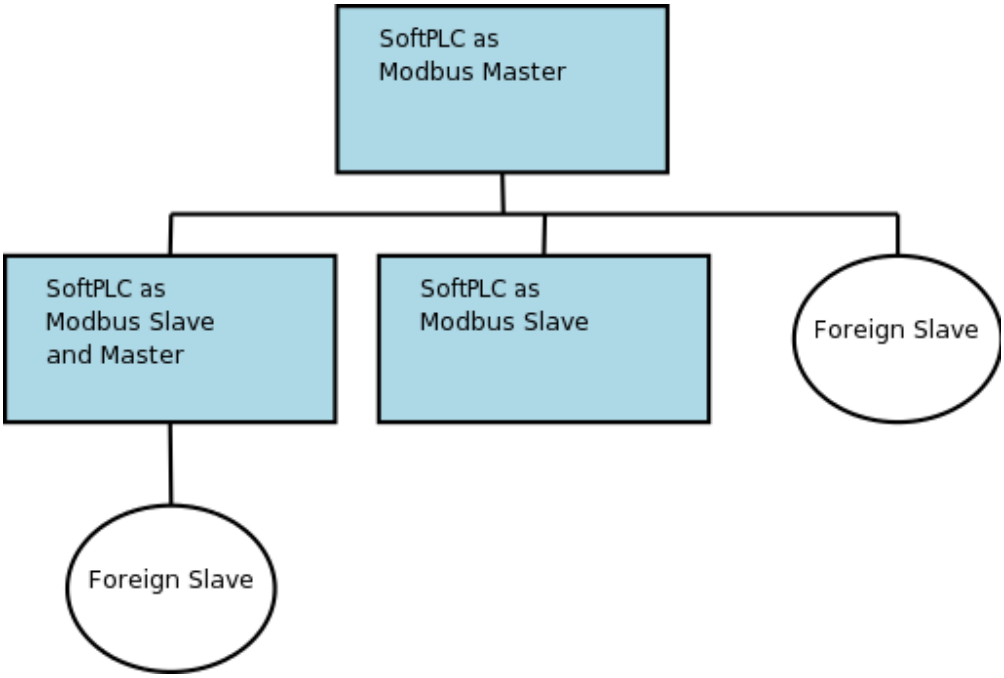
After using NexGen to edit the configuration file, **Send** it down to the SoftPLC. The next step is to

cycle power on the SoftPLC for the changes to take place. As an alternative to cycling power, you may enter "Remote Program" mode using NexGen, then select "Remote Program" a second time. This psuedo transition from Remote Program to Remote Program is a signal to the TLM that it should reload its configuration file. This way you can reconfigure without cycling power, although it does require you enter "Remote Program" mode (twice!).

Chapter 5. Usage

5.1. Typical System Architecture

- Distributed Control



5.2. Supported Modbus Commands

Modbus Function	Name	Supported?
1	Read Coils	Yes
2	Read Input Discretes	Yes
3	Read Multiple Registers	Yes
4	Read Input Registers	Yes
5	Write Coil	Yes
6	Write Single Register	Yes
7	Read Exception Status	No
15	Force Multiple Coils	Yes
16	Write Multiple Registers	Yes
20	Read General References	No
21	Write General Registers	No
22	Mask Write Register	Yes
23	Read Write Registers	Yes
24	Read FIFO Queue	No

Chapter 6. Debugging

This section gives tips on debugging problems on the Modbus network.

6.1. Isolating the Problem Slave Node

During startup or when troubleshooting a problem node it is usually best to isolate the problem node. This means look at it in isolation, by making it the only active slave on the network. You can keep the other slaves connected, but use a temporary configuration file to announce to the TLM only the node that you are troubleshooting. All other nodes/slaves will simply not be scanned.

Before you start debugging, you should use the configuration editor to Fetch and then Save your existing configuration. Then, on your development system (Windows computer), temporarily copy the file \SoftPLC\plc\<PLCNAME>\MBIPSLAV.LST to a safe place. Then, you can edit the configuration file temporarily and experiment freely. Later, restore by copying from the safe place back to \SoftPLC\plc\<PLCNAME>\MBIPSLAV.LST. Then, use the editor to Load then Send the file back down to the SoftPLC. Remember that you have to restart SoftPLC runtime after each configuration change.

6.2. Enable Debugging

The SoftPLC runtime engine constantly monitors its processes, and 'logs' these observations as process output. By default, these logs are minimal. However, for troubleshooting purposes, the logs can provide greater detail.

In the configuration file (MBIPSLAV.LST), there is the section **[Driver]** and its option **[DEBUG]**.

- A debug value of "0" offers no diagnostic output.
- A debug value of "1" offers a Modbus Query/Response trace.
- A debug value of "2" offers additional details to the lower level values.

6.3. View Debugging

Viewing these logs shall be completed at the command prompt of the SoftPLC system. To access the command prompt, log into the SoftPLC by either:

- (from Windows) use third-party 'PUTTY' application; or similar
- (from Linux) use SSH from Terminal application
- (TOPDOC 5.x) use Remote Console feature in the 'PLC' window

Default login credentials are as follows:
user: root
password: softplc

Once logged in, the logs can be viewed by executing one of the following:
(the '#' represents the prompt, and is not typed)

- For SoftPLC firmware 4.x
  - # logread
- For SoftPLC firmware 5.x
  - # journalctl -u softplc
    - You may need to use the arrow keys to scroll down to the end of the logs. The last logs are the most recent.

6.4. Direct Debugging to Text File

The previous sections have shown how to view the logs from the command prompt. However, recording the logs to text file format is, in the least, efficient for receiving support. Accomplishing this, much like viewing the logs, is firmware dependent (see following sections). Once the text file is created, it can be transferred via (S)FTP to the TOPDOC machine. A detailed explanation of (S)FTP transfers can be found in the TOPDOC User's Guide.

6.4.1. Direct Debugging output into a text file (SoftPLC 4.x)

1. Log into SoftPLC using either a) PUTTY from Windows or b) using ssh from Linux or c) at the command prompt of the SoftPLC system.
2. Run this command:
   # /etc/init.d/softplc.sh stop
3. Change into the /SoftPLC/run directory:
   # cd /SoftPLC/run
4. You can run SoftPLC from the command prompt now and redirect its output to an arbitrary file (named 'out.txt' here). We put that file into the RAM disk which is anchored in the /tmp directory.
   # ./runsplc > /tmp/out.txt
5. Let this run for 10-60 seconds, then press control-C. Now you have the output captured in file '/tmp/out.txt', each request-response transaction will be captured in that file.
6. You can look at the file using the program named "less".
   # less /tmp/out.txt
   You can look at this output with the Modbus TCP Specification, and the manual for your I/O module in hand. Press ESC when done.
7. You can make configuration file changes and Send them down to SoftPLC. Then merely repeat the part of this process starting at step 4 above.
8. When done, remember to set debug back to "0", then you can start SoftPLC as a daemon either by a) power cycling the box or b) doing the following:
   # /etc/init.d/softplc.sh start

6.4.2. Direct Debugging output into a text file (SoftPLC 5.x)

1. Log into SoftPLC using either a) the remote console feature in TOPDOC's PLC window, b) PUTTY (from Windows), or c) using ssh (from Linux).
2. Run this command:
   # systemctl stop softplc
3. Change into the /SoftPLC/run directory:
   # cd /SoftPLC/run
4. You can run SoftPLC from the command prompt now and redirect its output to an arbitrary file (named 'out.txt' here). We put that file into the RAM disk which is anchored in the /tmp directory.
   # ./runsplc > /tmp/out.txt
5. Let this run for 5-60 seconds, then press control-C. Now you have the output captured in file '/tmp/out.txt', each request-response transaction will be captured in that file.
6. You can look at the file using the program named "less".
   # less /tmp/out.txt
   You can look at this output with the Modbus TCP Specification, and the manual for your I/O module in hand. Press ESC when done.
7. You can make configuration file changes and Send them down to SoftPLC. Then merely repeat the part of this process starting at step 4 above.
8. When done, remember to set debug back to "0", then you can start SoftPLC as a daemon either by a) power cycling the box or b) doing the following:
   # systemctl start softplc