SendMail for SoftPLC® Runtime

Version 1.0

Table of Contents

Chapter 1. Overview

1.1. Introduction

The Send Email TLM (Cat No SM-MSG) is an add-on firmware option for SoftPLC version 4.x and later, which can be purchased for each system needing the capability to send emails or text messages. This capability is implemented as a TOPDOC Loadable Module **(TLM)**, written in C++ and implements a ladder instruction (TLI) which sends email using the well known SMTP protocol. This document describes the installation, usage, and functionality of the sendmail TLM.

This TLM may be used to send email by way of a ladder logic instruction, giving the user full programatic control over the trigger conditions. Queuing is implemented. Therefore a number of outbound emails can be triggered within a very small window of time and the TLM will queue up the requests on a first in first out basis in the same way that an alarm annunciator would. The queued emails will be sent out as fast as possible, depending on the speed of the connection path to the SMTP server and SMTP server speed.

A number of different SMTP authentication methods are supported, include the one used by Google's well known gmail. Provided you have a gmail account, the GMail SMTP server can be used to forward messages to your cell phone. (Gmail accounts are readily attainable for free, making this capability available to all SoftPLC users with Internet access.)

1.2. Definitions

- An email message consists of the following parts: recipient list, sender identity, message body, and optional attachment.
- An SMTP server is provided by your Internet Service Provider, or you may use any for which you can provide authentication information such as a username and password. This TLM must be given authenticated access to an SMTP server. That is, all email is sent **through** an SMTP server on the Internet or in house.

1.3. Concepts

The **SoftPLC runtime engine** 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 and not as an inherent part of the scan. TLMs are made known to SoftPLC in the MODULES.LST file which may be edited by TOPDOC NexGen by traversing to: PLC | Modules.

This SENDMAIL TLM is to be used as a MODULE, not as a DRIVER, and it provides a Topdoc Loadable Instruction (TLI) called SENDMAIL that you call from your ladder program. A new email is sent on any false to true rung condition edge. The SENDMAIL TLI takes a CONTROL datatable element as its first parameter and uses this to keep track of edge triggering and email sending progress. Each TLI used in your ladder program must have its own dedicated unique CONTROL element.

1.4. Features

In order to use the SENDMAIL TLI you need a working ethernet connection, DNS server, and default gateway. This TLM takes a configuration file named SENDMAIL.LST which holds information about one or more SMTP servers that you intend to use, and about your SMTP account on each identified server.

Up to 5 email messages may be queued at one time in RAM before the SENDMAIL TLI begins to report that the queue is full. As soon as the first queued up email message has been sent, this frees up another slot and then the TLI is ready for another rising edge trigger condition. (The limit of 5 is arbitrary. If you have enough RAM relative to the size of your email messages, this limit can be expanded upon special request.)

An email message can consist of a text file that you prepare in advance on a SoftPLC runtime disk. Optionally, a second file can be included as an attachment to the email. The attachment is another disk file, and if provided, can be either text or binary in nature.

A few different levels of verbosity are supported regarding debug logging of the SMTP transactions.

A command line program is provided allowing you to debug your configuration and SMTP client setup outside of the SoftPLC runtime, yet on the same box using the Linux command prompt.

1.5. Requirements

- A working link to the Internet, using either ethernet, modem, or PPP.
- A DNS server defined in your /etc/NETWORK(S).LST file.
- A default gateway defined in your /etc/NETWORK(S).LST file.
- Version 4.x SoftPLC of 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.

© Copyright 2010-2011 SoftPLC Corporation ALL RIGHTS RESERVED

First Printing	May, 2010
Latest Printing	Jan, 2011

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. Configuration

3.1. Email Components

An email message as supported by this TLM is assembled from various parts. Listed below are the parts that the TLM needs or uses and the source of the part.

Table 1. Parts of an Email Message According to SENDMAIL TLM

Part Name	Description	Source
Control	A datatable CONTROL element consisting of 3 words. Used to keep track of the progress of the email.	First TLI parameter
Result	A datatable STRING element that will hold explanatory text about the status of the send operation at completion.	Second TLI parameter
Identity	A datatable STRING element which names a record in the SENDMAIL.LST file. The named file record will provide the SMTP server, SMTP account username and password, and Reverse-Path. Reverse-Path is a formal designation of the "From:" address.	Third TLI parameter and the file /SoftPLC/tlm/SENDMAIL.LST
To	A datatable STRING element which holds a recipient list of email addresses, separated by a comma. No comma if only a single address. For example: "rbs@hme.com, another@hme.com" without the quotes.	Fourth TLI parameter.
File	A datatable STRING element which holds the filename of the main body of the email message.	Fifth TLI parameter and the named disk file. For example: "/home/root/mainbody.txt"
Attachment	A datatable STRING element which holds the filename of the attachment to the email message. An empty STRING is interpreted as no attachment.	Sixth TLI parameter and the named disk file. For example: "/tmp/processdata.csv" without the quotes.

Part Name	Description	Source
File MIME Type	A formal textual description of the MIME type for the **File** of the email message. If not provided, then "text/plain" is supplied automatically. A valid alternative would be "text/html" if your main body file is HTML.	Not required, but can be given preceeding the filename in the fifth TLI parameter. For example: "text/plain: /etc/NETWORK.LST" without the quotes. The colon space combo is the separator, and will be stripped away when grabbing the MIME type and filename.
Attachment MIME Type	A formal textual description of the MIME type for the **Attachment** of the email message. If the type is "application/octet-stream" or any other starting with "application/", then the attachment will be base64 encoded such that a binary attachment will be sent exactly. If not provided, then "application/octet-stream" is supplied automatically. A valid alternative would be "text/plain".	Not required, but can be given preceeding the attachment in the sixth TLI parameter. For example: "application/octet-stream: /tmp/ mydata.zip". The colon-space combo is the separator, and will be stripped away when grabbing the MIME type and attachment.

3.2. Sample Main Body Files

The main body file must be a text disk file, either simple ASCII or HTML.

3.3. Attachment Files

The single allowed attachment may be a text or binary disk file.

3.4. SENDMAIL.LST Configuration File

The configuration file for the SENDMAIL TLM is /SoftPLC/tlm/SENDMAIL.LST, and a sample is shown below.

Sample SENDMAIL.LST

```
# Configuration file for SoftPLC SENDMAIL TLM.
# Anything from # to end of line is a comment.



# The debug flag comes first, set to 0 to turn off debugging.
```

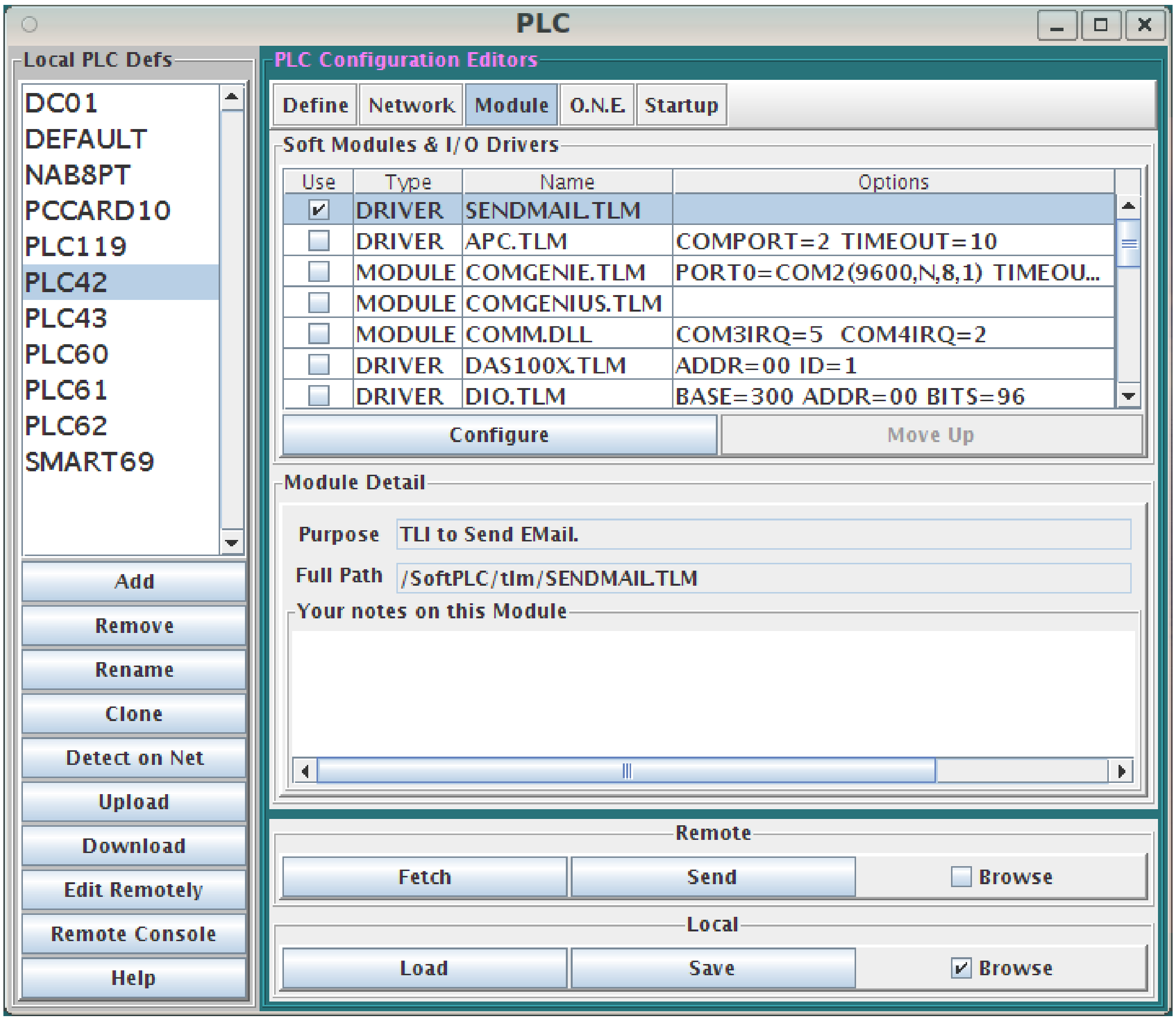
```
#define DEBUG_SHOW_ARGS         (1 << 0)    //< print the TLI arguments
#define DEBUG_TRANSACTIONS      (1 << 1)    //< print transactions with mail server

debug = 3
#debug = 0


# Next comes one or more identities.  An identity consists of
# identity = name
# and a number of trailing attribute/value pairs that follow on separate lines.
# The identity name must match your TLI "Identity" ladder instruction parameter.
# Then comes server, reverse_path, and optionally: username, password, starttls, helo.

#########################################################################

identity = birch.net

    # server: set to the name and port number of the SMTP server. Be sure and
    # provide the port number, either 25 or 587 after the server name.
    server = mailserver.birch.net:25

    # reverse_path: is tantamount to the "From:" email header, but is called
    # reverse_path because of RFC 2821. Think of this as a from field for the
    # email.
    reverse_path = dick@softplc.com

    # helo: set the hostname to be identify as when sending HELO or EHLO commands.
    # This is a per identity option.  It should be the name you are seen as
    # from the answering server.  If none given, then this machine's hostname
    # is used (and that is usually sufficent).
    helo     = plc42

    # username = plc42@factoryname.com
    # password = secretsmyfriend

    starttls = disabled

#########################################################################

identity = gmail
    server = smtp.gmail.com:587
    # helo     = plc42
    username = smtpuser@gmail.com
    password = passWord4
    starttls = required
    reverse_path = plc42@factoryname.com
```

From the sample configuration file above, you can see that after the **debug** setting, then comes the **identity** record(s). You only need one identity record if you only plan on using only one SMTP account. You may omit the 2nd identity record. The comments in the sample file should be enough to get you going.

Chapter 4. Usage

4.1. Installation

The TLM is named sendmail.tlm.so, and the configuration file is SENDMAIL.LST. You will need to copy these onto the SoftPLC flash in the /SoftPLC/tlm directory. Then, to use it you have to enable it in NexGen's PLC | MODULES editor. Simply click on the ***Configure*** button after selecting and enabling ***Use*** in the same row as the SENDMAIL TLM.



Local PLC Defs: DC01, DEFAULT, NAB8PT, PCCARD10, PLC119, PLC42, PLC43, PLC60, PLC61, PLC62, SMART69

Use	Type	Name	Options
✓	DRIVER	SENDMAIL.TLM	
	DRIVER	APC.TLM	COMPORT=2 TIMEOUT=10
	MODULE	COMGENIE.TLM	PORT0=COM2(9600,N,8,1) TIMEOU...
	MODULE	COMGENIUS.TLM	
	MODULE	COMM.DLL	COM3IRQ=5 COM4IRQ=2
	DRIVER	DAS100X.TLM	ADDR=00 ID=1
	DRIVER	DIO.TLM	BASE=300 ADDR=00 BITS=96

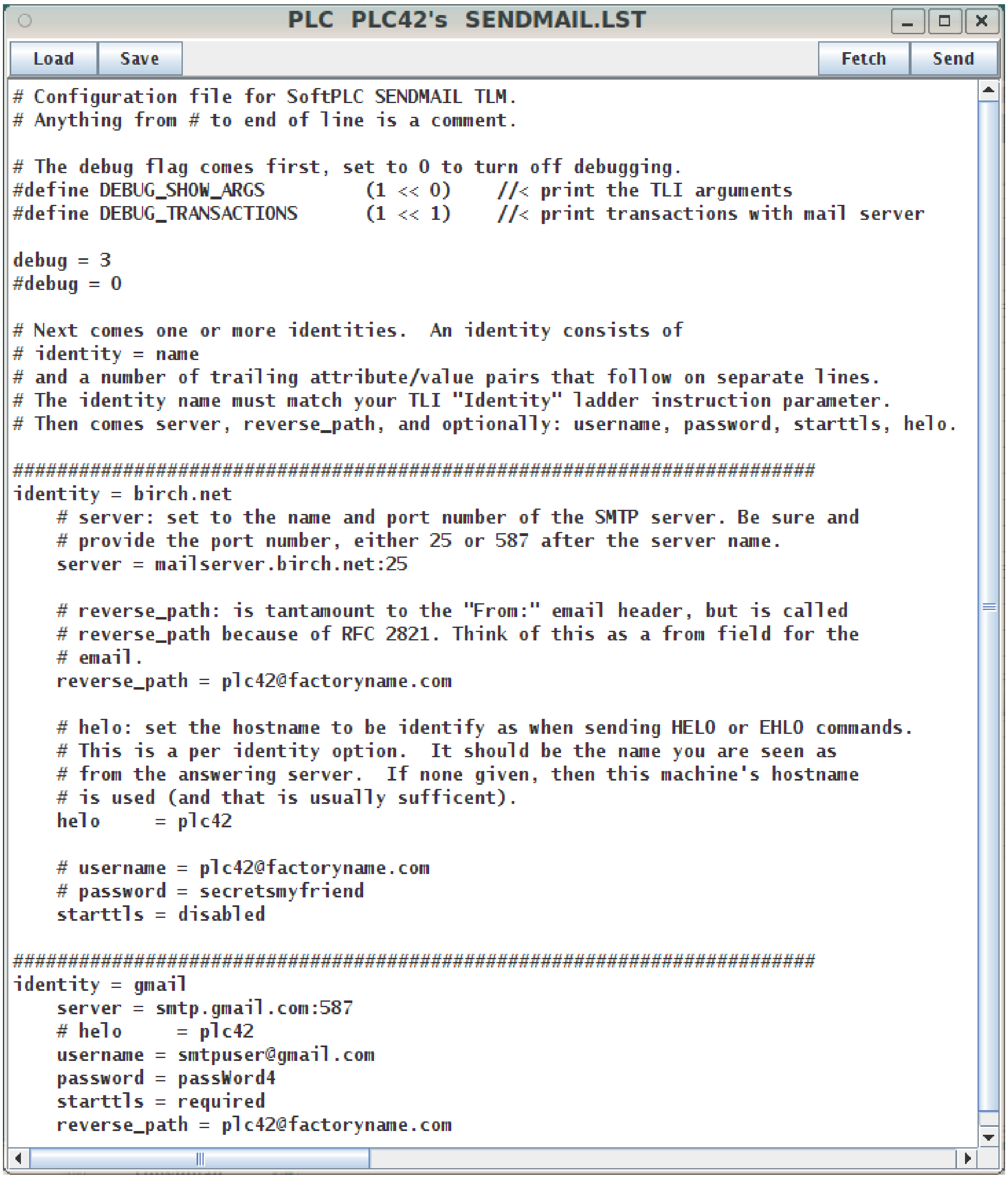
Purpose: TLI to Send EMail.

Full Path: /SoftPLC/tlm/SENDMAIL.TLM

Remote: [ ] Browse

Local: [✓] Browse

4.2. Editor Usage



```
# Configuration file for SoftPLC SENDMAIL TLM.
# Anything from # to end of line is a comment.

# The debug flag comes first, set to 0 to turn off debugging.
#define DEBUG_SHOW_ARGS         (1 << 0)    //< print the TLI arguments
#define DEBUG_TRANSACTIONS      (1 << 1)    //< print transactions with mail server

debug = 3
#debug = 0

# Next comes one or more identities.  An identity consists of
# identity = name
# and a number of trailing attribute/value pairs that follow on separate lines.
# The identity name must match your TLI "Identity" ladder instruction parameter.
# Then comes server, reverse_path, and optionally: username, password, starttls, helo.

########################################################################
identity = birch.net
    # server: set to the name and port number of the SMTP server. Be sure and
    # provide the port number, either 25 or 587 after the server name.
    server = mailserver.birch.net:25

    # reverse_path: is tantamount to the "From:" email header, but is called
    # reverse_path because of RFC 2821. Think of this as a from field for the
    # email.
    reverse_path = plc42@factoryname.com

    # helo: set the hostname to be identify as when sending HELO or EHLO commands.
    # This is a per identity option.  It should be the name you are seen as
    # from the answering server.  If none given, then this machine's hostname
    # is used (and that is usually sufficent).
    helo     = plc42

    # username = plc42@factoryname.com
    # password = secretsmyfriend
    starttls = disabled

########################################################################
identity = gmail
    server = smtp.gmail.com:587
    # helo     = plc42
    username = smtpuser@gmail.com
    password = passWord4
    starttls = required
    reverse_path = plc42@factoryname.com
```

Load button will load the configuration file from the development system’s disk.

Save button will write the configuration file to the development system’s disk.

Fetch button will load the configuration file from the runtime system’s disk.

Send button will write the configuration file to the runtime system’s disk. 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!).

4.3. Programming the TLI

The TLI is low to high edge triggered, and this edge will queue up another copy of an outbound email with the makeup given by the TLI parameters. Currently the queue size is arbitrarily set to 5 emails. A sixth attempt would not succeed until the first email was completely sent.

Upon completion, either the DN bit or ER bit will be set:

- Success: The DN (done) bit is set within the CONTROL, the POS field of the CONTROL will increment by 1, and the email will have been sent OK.
- Failure: The ER (error) bit will be set, the LEN field within the CONTROL will be set to a non-zero error code, and the Result: STRING will tell you what went wrong. The POS field of the CONTROL will increment by 1.

Chapter 5. Debugging

This section gives tips on debugging problems with the TLI.

5.1. Using the debug Configuration Setting

Set **debug** to 3 in the configuration file while testing out a new ISP’s SMTP server. Run softplc from the command line, this will tell you something about what is happening when the ladder instruction is triggered.

5.2. Using the Command Line Program mailtest

Run the program /SoftPLC/run/mailtest without any arguments and it will tell you how to send an email from the command line. Use this to hone your email components for the TLI.