Smart TLM

Table of contents

1 Overview

1.1 Overview

1.1.1 Introduction

The Smart CPU is a very flexible, communications rich, cost effective platform for driving SoftPLC Corporation's Tealware I/O architecture while running the SoftPLC runtime software or I/O adapter software. This document explains how to configure the I/O driver for the Tealware I/O when attached to a Smart CPU running the runtime software. A driver under the SoftPLC runtime is called a TOPDOC Loadable Module **(TLM)** . There can be several TLMs running concurrently under the runtime. This particular TLM is written in C ++ and implements a capable architecture which can manage up to 3072 digital I/O points attached to a single CPU. (Remember that the runtime can have other kinds of TLMs driving other kinds of I/O concurrently also.) This document describes the installation, usage, and functionality of the Smart TLM.

The Smart TLM may be used to monitor and control up to 12 racks of Tealware I/O modules. A rack may have 4, 6, or 8 I/O module slots in it. Alternatively, there is also a 3 slot backplane that can be attached directly to the CPU to control 3 I/O modules.

1.1.2 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 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.

1.1.3 Features

1.1.3.1 I/O Forcing

The TLM supports input and output forcing on I/O that are digital.

1.1.3.2 Hot Swap Digital I/O

Digital I/O modules may be replaced under power with a module of the same kind.

1.1.3.3 Reconfiguration Without Power Cycling

The TLM will re-read its configuration file on any PROGRAM mode to PROGRAM mode transition.

1.1.3.4 Auto Configuration

The TLM will automatically detect all I/O modules present if the configuration file is not present, and create a new configuration file.

1.1.3.5 Frequency Mode for High Speed Counter Modules

The HSC11 module is a high speed counter module which normally only counts. This TLM can work with that module to measure frequency of a digital pulse train.

2 Warranty

2.1 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: August, 2010
Latest Printing: August, 2010

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

3 Scanning Operation

3.1 Scanning Operation

3.1.1 Operating Modes and States

The SoftPLC runtime engine is always in one of the following states, called Operating Modes.

Mode	Description
Program or Remote Program	Logic is not being solved and the outputs are in an **idle** state. Normally idle state means "turned off or zeroed".
Run or Remote Run	Logic is being solved and the outputs are active and under the control of the logic program. They are not idle. The logic program makes its decisions based on the current state of each input, all of which are actively scanned.
Test or Remote Test	Logic is being solved but the outputs are idle. The logic program makes its decisions based on the current state of each input, all of which are actively scanned.
Faulted	Logic is not being solved and the outputs are idle. This mode is entered automatically if you have an error in your program or in one of your driver configurations.

Table 1: SoftPLC Operating Modes

4 Configuration

4.1 Configuration Basics

4.1.1 Smart Configuration File

The configuration file is a XML text file and is best edited with the dedicated editor built into TOPDOC NexGen, but may also be edited with any text editor.

4.1.2 Configuration Elements

This TLM is configured using a special configuration editor which is built into TOPDOC NexGen. The configuration file is XML text, is hierarchical with the following XML eelements. Elements are listed below with one of the following characters appended. The appendage is used to indicate howmany times the element may occur in any given context. The appended character and its meaning is as follows:

- ? Question Mark => Optional (zero or one)
- * Asterisk => Zero or more
- + Plus Sign => One or more
- None (no suffix) => exactly once

Element Name	Description	Sub Element(s)
SmartTLM	Top most element, holds all other elements	bus*
bus	References and configures a communications channel which talks to racks.	rack*
rack	Holds 3, 4, 6, or 8 I/O modules.	module*
module	Identifies a I/O module by its slot position within a rack, and its module type.	in? out? CDM? hz?
in	Present only for input modules, identifies where in the SoftPLC datatable the module's input scan data will be placed. For **digital** input modules this must be in the I: section of the datatable.	
out	Present only for output modules, identifies where in the SoftPLC datatable the module's output scan	

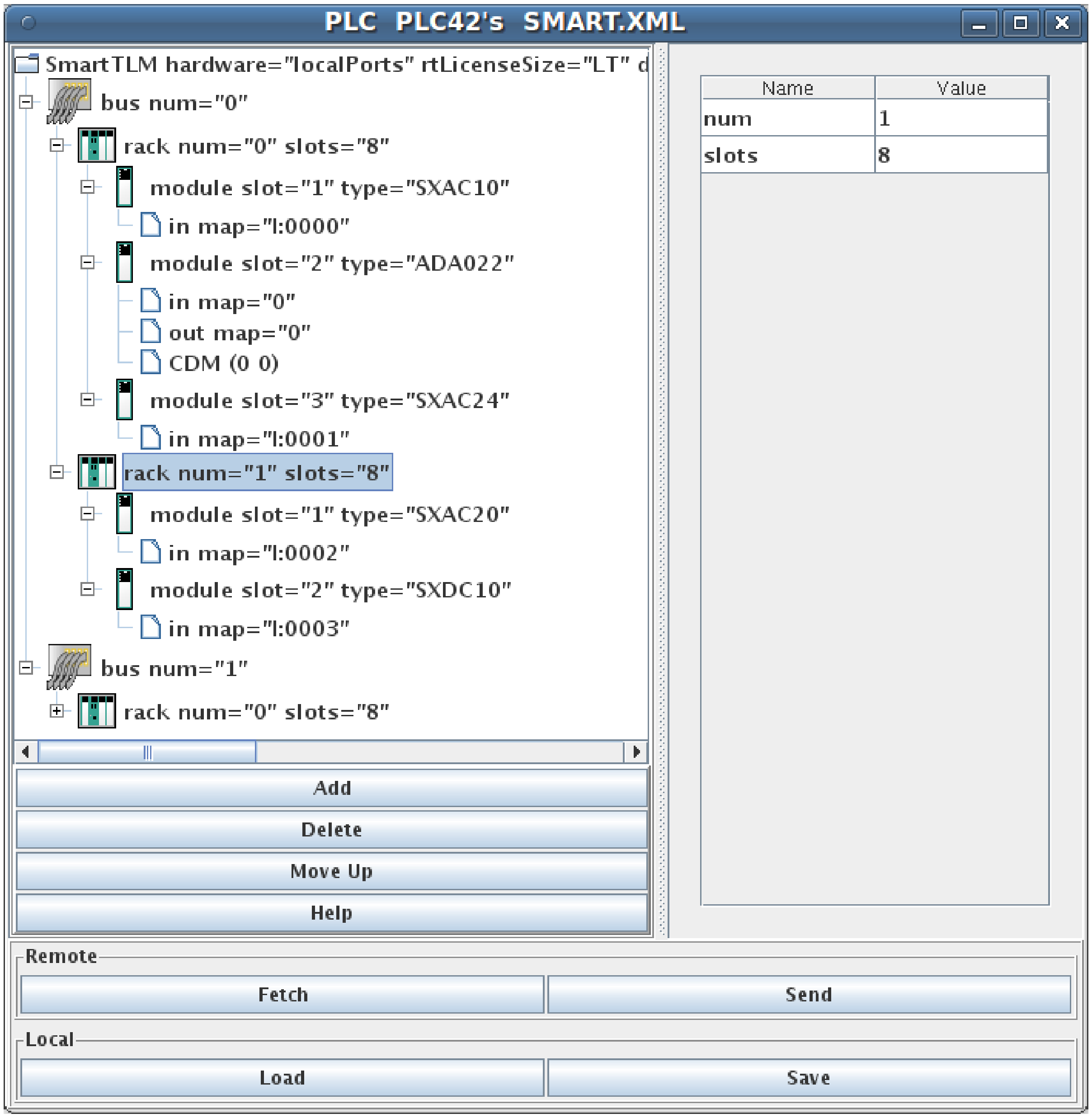
Element Name	Description	Sub Element(s)
	data will come from. For **digital** output modules this must be from the O: section of the datatable.	
CDM	Present only for intelligent modules, identifies a block of single shot inline configuration data that the TLM will download to the module for configuration purposes on any transition to a RUN mode.	
hz	Allowed only on HSC11 high speed counter modules, and when present enables the TLM to calculate a frequency for each of the 3 module channels via the HZ ladder instruction.	

Table 1: Elements and their Allowed Sub-Elements

Note:

When using NexGen to edit the configuration file, its TLM specific configuration editor takes care of enforcing the rules of the configuration file.

Here is a sample screen from the configuration editor showing a few of the elements from the above table. Notice how they are arranged hierarchically and that each element can "contain" other elements. (The rules of containment are given in the table Elements and their Allowed Sub-Elements.)


PLC PLC42's SMART.XML

```
SmartTLM hardware="localPorts" rtLicenseSize="LT" d
bus num="0"
  rack num="0" slots="8"
    module slot="1" type="SXAC10"
      in map="I:0000"
    module slot="2" type="ADA022"
      in map="0"
      out map="0"
      CDM (0 0)
    module slot="3" type="SXAC24"
      in map="I:0001"
  rack num="1" slots="8"
    module slot="1" type="SXAC20"
      in map="I:0002"
    module slot="2" type="SXDC10"
      in map="I:0003"
bus num="1"
  rack num="0" slots="8"
```

Name	Value
num	1
slots	8

In the above panel, the element name is at the far left of each tree row. To the right of the element name, still within the tree row, is a list of **attributes**. That element's attributes are elaborated on within the table at the far right of the panel. That table is dynamic (depends on the selected element), and has one row for each attribute. The following section has a table listing all the allowed attributes for each element type.

4.1.3 Attributes of Elements

The table below gives the allowed **XML attributes** which may be attached to each **XML element** in the configuration file. Only these attributes may be attached to the corresponding element.

Element	Attribute	Value	Required
SmartTLM	debug	0, 1, or 2, meaning "enable none, some, or all debugging print statements"	no, defaults to 0
	hardware	Type of connectivity to the Tealware I/O: 'localPorts' or 'backplane3'. One architecture provides only 3 slots on a single bus (backplane3), and the other provides 3 racks on each of 4 buses (localPorts). See here.	yes
	rtLicenseSize	Runtime license size: LT, 1K, 2K, or 8K, and pertains to the I/O capacity of the runtime license. Setting this correctly allows the editor help you stay within limits that are imposed by the runtime software later when it loads the configuration file.	yes
	watchdog	The Tealware I/O modules each have a watchdog timer in them. The watchdog setting is a value sent to all of them on a transition to RUN mode that controls how long to wait during a quiet time before a module is to turn off its outputs. Range:	yes, e.g. 7 equates to 7/10ths of a second.

Element	Attribute	Value	Required
		1-14 deci-seconds. Additionally the special value 0xf0 means do not use the watchdog.	
	digInStart	Is used to establish the starting I: address used during the allocation of input image table required by digital input modules. This controls both a) the manual (module at a time new entry) allocation and b) the full configuration auto allocation which is available by selecting the top most element and calling up the popup menu with a right click.	yes
	digOutStart	Is used to establish the starting O: address used during the allocation of output image table required by digital output modules. This controls both a) the manual (module at a time new entry) allocation and b) the full configuration auto allocation which is available by selecting the top most element and calling up the popup menu with a right click.	yes
	regInStart	Is used to establish the datatable file for all analog input data. The word component must be zero, but any available N: file may be used. The word element for any analog or intelligent input module is then	yes

Element	Attribute	Value	Required
		calculated by using the associated **in** element's 'map' attribute as a word offset. For example, if regInStart is N17:0, and a module's <in map="14">, then the module's analog data will be moved into a block starting at N17:14 during the I/O scan.	
	regOutStart	Is used to establish the datatable file for all analog output data. The word component must be zero, but any available N: file may be used. The word element for any analog or intelligent output module is then calculated by using the associated **out** element's 'map' attribute as a word offset. For example, if regOutStart is N7:0, and a module's <out map="14">, then the module's analog data will be sourced from a block starting at N7:14 during the I/O scan.	yes
bus	num	The bus number, 0, 1, 2 or 3	yes
rack	num	The rack number, 0, 1 or 2.	yes
	slots	The number of slots for the rack: 3, 4, 6, or 8.	yes
module	slot	The slot number, which starts at 1, with a range of 1-8	yes
	type	The type of Tealware I/O module, picked from	yes

Element	Attribute	Value	Required
		a menu from within NexGen.	
in	map	One of two kinds of datatable references, either absolute or relative. The absolute form is an actual datatable address and is used for digital input modules. The relative form is a zero based offset from the absolute starting address given by element SmartTLM's regInStart, and is used for non-digital input modules.	required and present only for input modules
out	map	One of two kinds of datatable references, either absolute or relative. The absolute form is an actual datatable address and is used for digital output modules. The relative form is a zero based offset from the absolute starting address given by element SmartTLM's regOutStart, and is used for non-digital output modules.	required and present only for output modules
hz	window	The number of samples to use in a sliding window filter, 2-4096. Frequency is calculated by subtracting the oldest count sample from the newest count sample and dividing by the elapsed time between the two samples. The oldest sample is disposed of	yes

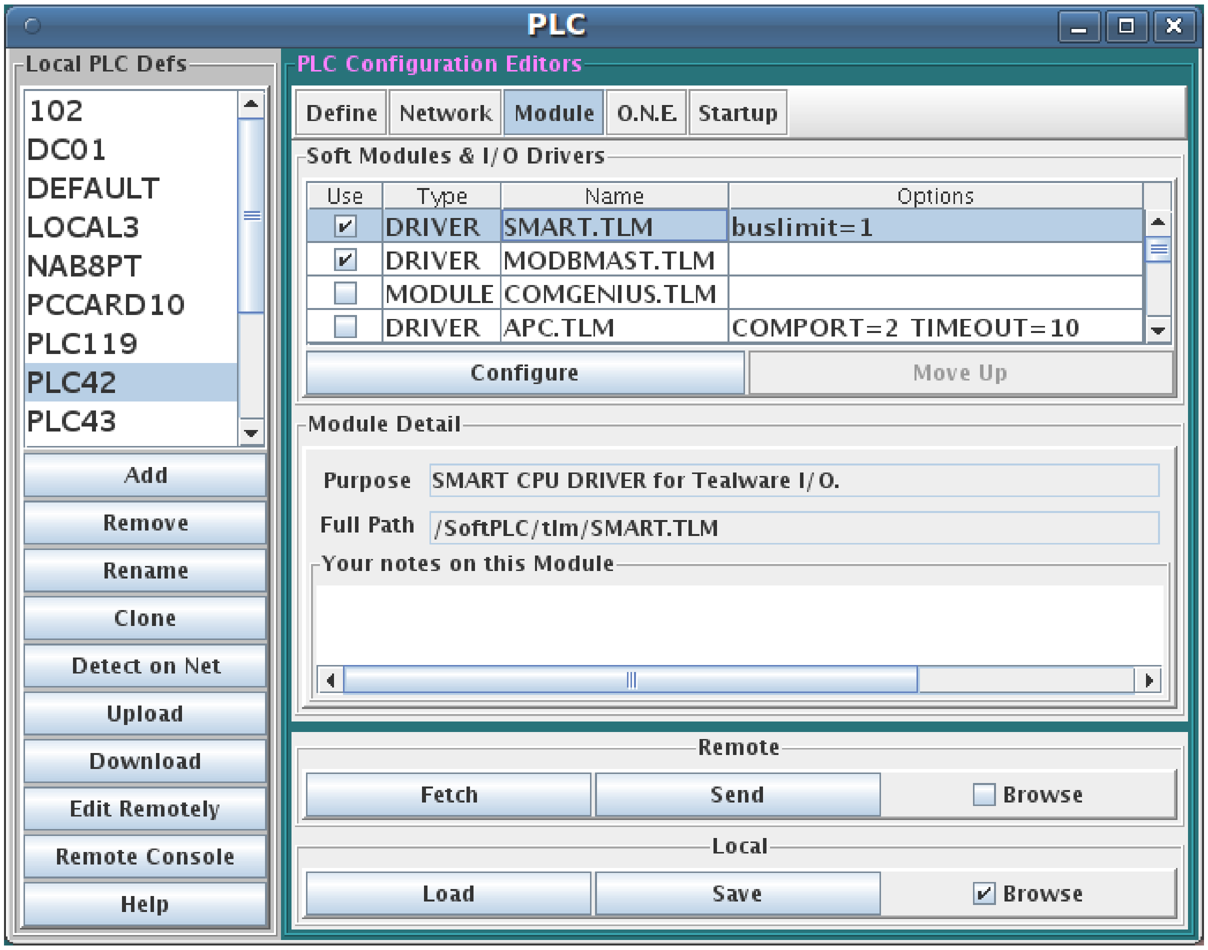
Element	Attribute	Value	Required
		when the newest sample is inserted into the sliding window. The sample rate is established by how often the **HZ** ladder instruction is energized for this module.	

5 Usage

5.1 Usage

5.1.1 Installation

The TLM is named smart.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 XML file SMART.XML which is the TLM's configuration file. There is an application specific editor for this SMART.XML file within NexGen. It is easy to edit the configuration file from the PLC | MODULES editor. Simply click on the ***Configure*** button after selecting and enabling ***Use*** in the same row as the SMART TLM.

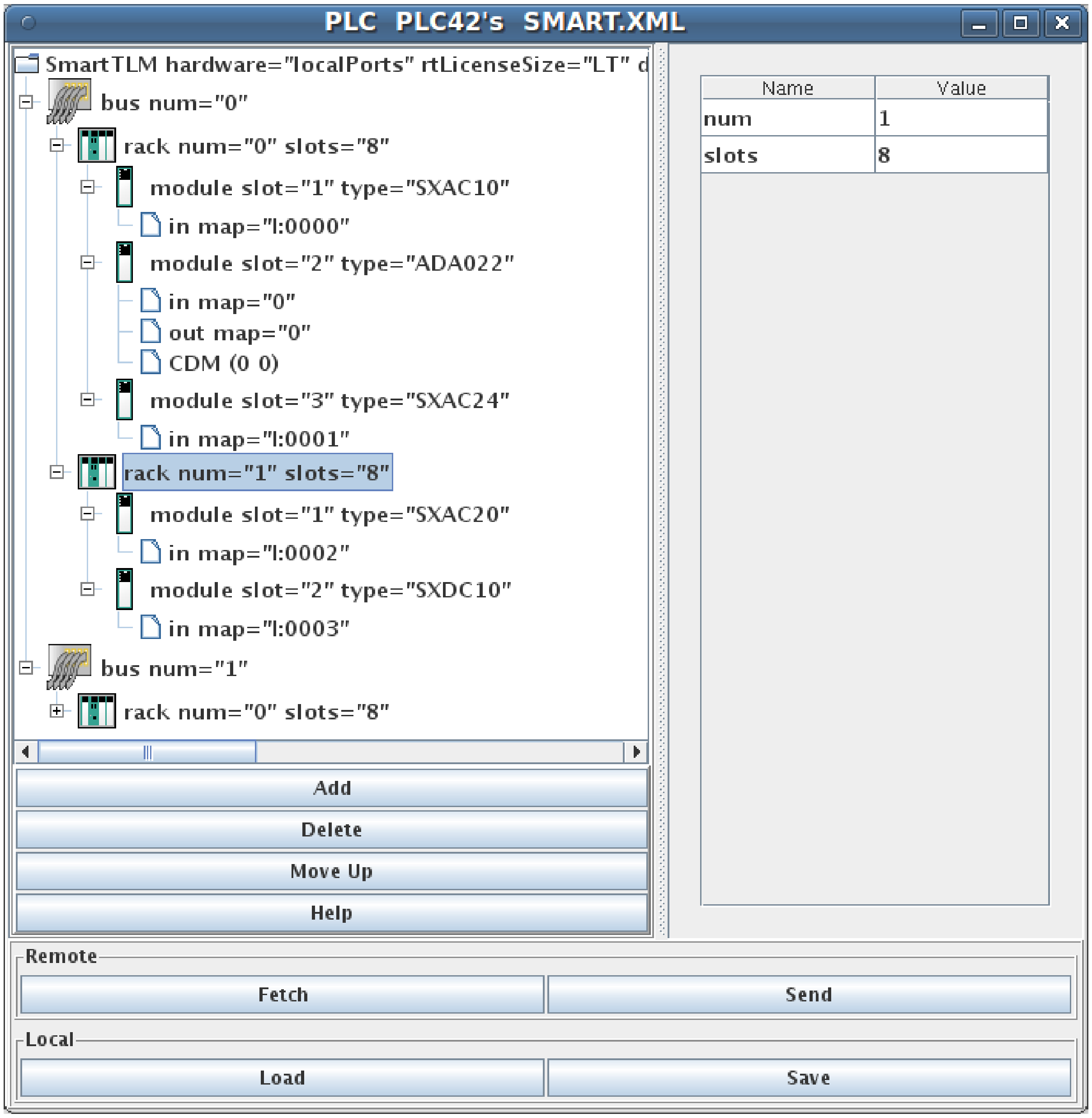


Use	Type	Name	Options
✓	DRIVER	SMART.TLM	buslimit=1
✓	DRIVER	MODBMAST.TLM	
	MODULE	COMGENIUS.TLM	
	DRIVER	APC.TLM	COMPORT=2 TIMEOUT=10

Purpose: SMART CPU DRIVER for Tealware I/O.

Full Path: /SoftPLC/tlm/SMART.TLM

5.1.2 Editor Usage



```
SmartTLM hardware="localPorts" rtLicenseSize="LT" d
  bus num="0"
    rack num="0" slots="8"
      module slot="1" type="SXAC10"
        in map="I:0000"
      module slot="2" type="ADA022"
        in map="0"
        out map="0"
        CDM (0 0)
      module slot="3" type="SXAC24"
        in map="I:0001"
    rack num="1" slots="8"
      module slot="1" type="SXAC20"
        in map="I:0002"
      module slot="2" type="SXDC10"
        in map="I:0003"
  bus num="1"
    rack num="0" slots="8"
```

Name	Value
num	1
slots	8

Add button will insert a new element within the selected element. First select the element you wish to insert into.

Delete button will deleted the selected element. It is only enabled when you are allowed to delete the selected element.

Move Up button will move the selected element up in the current containment list.

Fetch, Send, Load, and Save all have the same meaning as they do in the NexGen Module editor. You can see the helpfile for that editor by going to that editor and clicking on Help.

Use ***Send*** to transfer the configuration 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!).

5.1.2.1 Popup Menus

There are a number of right click invoked popup menu choices that are available when there is a tree row selected.

One such choice that comes available when the top most element is selected, is called **Delete Configuration on Smart** . This choice will attempt to delete the SMART.XML file within the associated actual SoftPLC CPU. When that file is missing at the time of power up, then the TLM will automatically create one using a module detection scan loop through each rack and slot.

Another helpful menu choice **Re-allocate All Datatable Addresses** , which will change the in and out map attributes in an ascending sequence with no gaps.

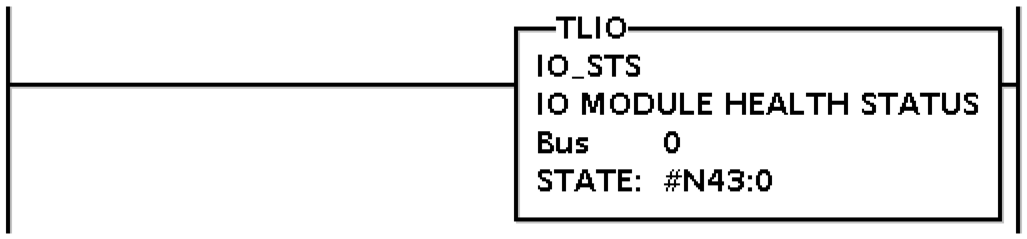
There are other menu choices to sort slots or racks or buses, should you wish to renumber them manually, then have the menu option sort them into order again.

5.1.3 Ladder Instructions

This TLM implements a number of ladder instructions, one of communications health monitoring, another for talking to an HSC11 module in a special frequency measurement mode, and still others to transfer blocks of data to specialty I/O modules on demand.

5.1.3.1 IO_STS

This instruction can be used to monitor the health of the communications to each and every I/O module within a local Tealware bus. You can program one of these instructions for each Tealware local bus that you have. It returns 4 words which are bitmapped with individual module status information for each slot on that bus. The first 24 bits of the word block are mapped to the 24 slots on the bus (3 racks x 8 slots max per rack), starting with the first 8 slots available to rack 0, then the next 8 slots for rack 1, followed by the last 8 slots for rack 2. Bits are allocated starting from bit 0 in the first word and continue into the least significant 8 bits of the 2nd word. The 3rd and 4th words are not used at this time but will be set to zeros.



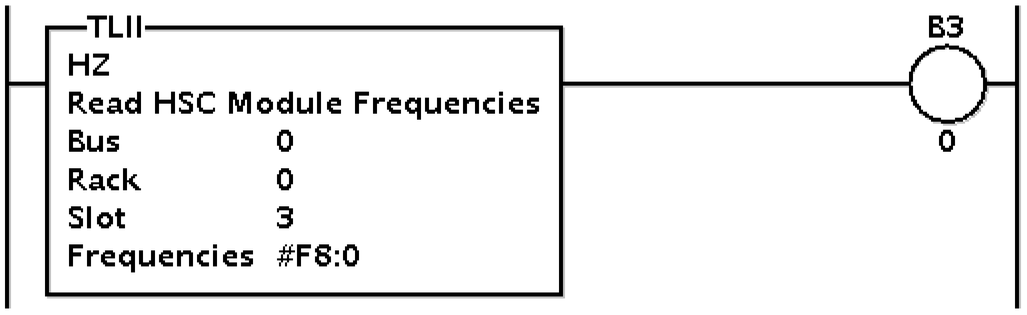
Parameter	Meaning
Bus:	An integer Tealware local bus number, 0-3.
State:	The address of an integer datatable block. The block's length is 4 words, and these 4 words receive the health status bits as decribed above, whenever the instruction is energized.

Table 1: Instruction Parameters

5.1.3.2 HZ

This instruction is only helpful when you have an HSC11 module and want to get frequency from it. The module maintains counts, not frequency, but with this instruction accurate timing information is applied to the counts to calculate frequency. To use this instruction, each of the module's 3 high speed counter channels must be put into the "continuous count up" mode. This instruction will convert each of those 3 increasing counts into a separate frequency, 3 separate frequencies per module. It is not possible to use less than all 3 channels on any given HSC11 in a non frequency mode. It is all or none.

To put each channel into the "continous count up" mode, it is a two step process. Firstly, the configuration file has to have a special XML element attached to it for any HSC11 module that you wish to operate in frequency mode. This will establish the continous count mode for all 3 channels on that module in software, but not the counting direction (up or down). The counting direction must be set by hardware separately for each channel by wiring a high signal to one of the channel's wiring terminals. Without this the frequencies will be negative.



Parameter	Meaning
Bus:	The Tealware bus number, 0-3.
Rack:	The Tealware rack on said bus, 0-2.
Slot:	The Tealware slot number within said rack, 1-8.
Frequencies	The datatable address of a block of 3 floats which will receive the 3 channel frequency calculations in units of HZ (counts/second).

Table 1: Instruction Parameters

Energizing the instruction causes both a new set of samples to be stored into each of the 3 sliding windows, and also the calculations to be performed and returned. The instruction might not need be energized on every scan.

Note:

A channel's frequency is calculated by subtracting the oldest sliding window count sample from the newest (higher count) and dividing by the elapsed time between the two. This is an average over the sliding window. By using only these two points the calculation is fast and allows dividing by a larger delta time, leading to less signal noise. The sliding window average is a digital filter, and you have the tuning knobs to control how it works based on a) the window size and b) the rate at which you energize the instruction. This algorithm should work fine as long as the change in counts between oldest and newest samples does not exceed 2 billion.

For example, with a window size of 200, and the instruction energized every 100 msecs, this is a total span of time of 200 x .1 sec = 20 secs. So the calculation (frequency measurement) would give you an average frequency across the last 20 seconds, but do so every 100 msecs. If the window size was reduced down to 40 samples, then 40 x .1 sec = 4 seconds window width. If you wanted something even more responsive, say an average of the last 250 msecs, (a reasonable scenario in fluid flow metering application), and you had a program scan time of about 10 mescs, then you could set the window size at 250 and simply leave the instruction energized for every scan. Then 25 x .01 seconds = 0.250 seconds sliding window time span.

6 Debugging

6.1 Debugging Tips

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

6.1.1 Enabling Debug Prints

In the configuration file there is the top most element **SmartTLM** and its attribute **debug** . The first several bits of this integer value enable various categories of print statements in the TLM. Remember a debug value of "0" means no debugging, and a debug value of -1 turns on all bits.

Bit Number	Debug Print Category
0	Configuration File Parsing.
1	TLI Parameters. This feature will print any detected problems with the parameters that you are passing to the ladder instructions.
2	This will print out anything related to the I/O modules which are detected in the racks.

Table 1: Bit Mapped Debug Categories

On version 4.x SoftPLC, all process output from the SoftPLC runtime engine is normally directed to the syslog, because SoftPLC runs as a daemon normally. The syslog can be configured in a number of different ways, but the default uses a small RAM resident FIFO and eventually will run out of space and wrap back around on itself. Rather than reconfiguring the syslog, there is an easier way.

Following is a procedure to get the debugging output into a text file.

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 awhile, say 5-60 seconds, then press `control-C`. Now you have the output captured in file `/tmp/out.txt`, each debug print statment will be captured in that file.
6. You can look at the file using the program named "less".
   ```
   # less /tmp/out.txt
   ```
   Press ESC when done. You can also FTP this file up to your developement computer and look at it with a text editor.
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
   ```

7 All