Sycon Serial Communication Protocol Library Manual
2007-10-19 REV1

## Introduction

The Sycon communication library (hereafter referred to as the SMDP control) is an ActiveX library that implements the following protocols:

- 'Sycon' protocol (STC200, STM-100, etc)
- SMDP (Sycon Multi Drop Protocol)

Note that there are products other than Sycon products that use the Sycon protocols. Also note that not all of Sycon's products use both protocols.

The SMDP control is built as an out-of-process server, and automatically manages multiple sessions with multiple programs. This means that all programs that use this control can share the communication link without interference or crosstalk. It is therefore possible to have multiple programs communicating with the same instrument. This feature is transparent to the user.

## Using

The SMDP control is a standard ActiveX control (so it will only work on Windows systems). Since all programming environments have different ways to integrate ActiveX controls, linking to the control will not be covered. Most programming environments can use ActiveX controls (Delphi, Visual Studio (vb,vc++, etc), LabVIEW, and many, many others). The only thing you will need to know is the name of the library (**SMDP_SVR is the name of the ActiveX control library**).

You should instantiate an instance of the control for each instrument you want to communicate with, then set the properties accordingly, and then call DoTransaction to do the communications. Do not worry about multiple instantiations on the same com port (the control handles this) or multiple instantiations for the same instrument (the control handles this, too). The control is designed to be easy to use, just instantiate, then set the properties for the instantiation, then call DoTransaction().

**Methods**

- Open() : Attempts to acquire COM port and allocate resources. This function does not need to be called; if the port was not opened it will automatically be opened when the DoTransaction function is called.
- DoTransaction(addr, SmdpCmd, Msg, Rsp, resetflag)
  This is the main function; it initiates the communication with the instrument, and waits for a reply from the instrument (or timeout) before returning.
  - In: addr is SMDP address of target instrument
  - In: SmdpCmd is SMDP command opcode
  - In: Msg is the message to send, may be a string or byte array
  - Out: Rsp is string, response from instrument
  - Out: Boolean resetflag is TRUE if instrument has been reset since the flag was acknowledged last.
- Close(): Frees up resources associated with the instance. Will also close the com port if there are no more instances communicating on the port. In most cases it is not necessary to call this functions since your programming environment will unload the control when it is finished, and the control automatically frees it's resources when it is unloaded.

**Communication Properties (independent for each instance)**

- **ComPortNo**: which PC com port to use for communication to instrument
- **Baud**: baud rate to use
- **Protocol**: which protocol to use (Sycon or SMDP)
- **DoPacketStamp**: Set TRUE to use SMDP-II protocol (should always be true unless communicating to STM-1)
- **TimeoutMS**: Sets the number of milliseconds to wait for the instrument to respond before the control gives up with a "timeout error" (during DoTransaction()
- **LastTimeTransSec**: The amount of time the last DoTransaction() call took to complete.

## Other properties

These properties are read only, and return the same value across all instances.

- **Numinstances**: The number of instantiations of the control. Put another way, number of users on communication server. Read only and same value for all instances.
- **Build**: The build number of the control (unique version identifier). Read only and same value for all instances.

**References**

See "Smdp Protocol.doc" for more information about the SMDP protocol.