Advanced Serial Port Monitor

Watch It!

PRINTED MANUAL

Advanced Serial Port Monitor

Printed: Апрель 2016

Publisher

AGG Software

Production

2003, AGG Software
http://www.aggsoft.com

Table of Contents

1 Introduction

1.1 About software

Today we are all living in a completely new era. Utilizing digital technologies is as natural as breathing and consuming food, these days. That is why so many businesses turn towards digital products and solutions. Number of devices that can be connected to a home or office PC grows exponentially. PC becomes the control center of various pieces of hardware: from MP3-players to industrial machinery.

One of the industry-standard device communications protocols is RS232/RS485/RS422 serial interface or COM port for short. Creating devices that communicate with PC via serial port is very cost-effective, not mentioning the amazing hardware compatibility. Most computers are equipped with at least one free serial port. However, developing hardware and software solutions that are built around COM port communications protocol is rather a challenging task without proper tools.

Introducing Advanced Serial Port Monitor (ASPM) by AGG Software, the unique serial port monitoring and data handling solution that allows developers focus on their project instead of battling hardware.

Advanced Serial Port Monitor can operate in manual, automatic and spy modes providing developers with unique opportunity to monitor all data received from device connected to COM port as well as send data directly to a serial port either manually or automatically on given time intervals. The application supports full duplex mode, which means you can use it anytime without having to close or restart programs already communicating with a device via serial port.

Advanced Serial Port Monitor supports all possible baud rates, data bits number, stop bits number, parity and flow control types. Besides flexible configuration, the software supports plug-ins used to emulate various devices and expand software functionality.

The application comes with comprehensive help system and intuitive interface to make it easy to start using it right after installation.

ASPM has the following capabilities:

- transmit of any data in both directions (from and to computer) in duplex mode (receive and transmit at the same time);
- work in manual control mode;
- work in spy mode, when do with other program data exchange;
- any data source (input string or file);
- output of received data to file;
- flexible connection settings (baud rate, data bits, etc.);
- switch of plug-ins;
- friendly, completely setup interface.

1.2 Glossary

Baud Rate - The transmission speed of data through an asynchronous channel. Often confused with BPS (bits per second), baud rate actually refers to the number of signals per second. Because each signal can represent more than one bit of data, the number of bits per second is usually higher than the baud rate. For example, 2400 bps is typically sent at a rate of 600 baud.

Binary File - A file that contains data or program instructions written in ASCII and extended ASCII characters.

ASCII - An acronym for American Standard Code for Information Interchange. ASCII files are plain, unformatted text files that are understood by virtually any computer. Windows Notepad and virtually any word processor can read and create ASCII files. ASCII files usually have the extension .TXT (e. g., README.TXT).

Bytes - A collection of eight bits that represent a character, letter or punctuation mark.

COM port - Short for a serial communication port. Most DNC software communicate with a computer through a communication port, and most IBM and IBM-compatible computers support up to four serial ports COM1, COM2, COM3 and COM4. Additional ports can be added by adding additional hardware.

Data bits - A group of bits (1's and 0's) that represent a single character or byte. Typically, there are seven or eight data bits. During an asynchronous communication (e.g., BitCom connecting to CompuServe), each side must agree on the number of data bits. Data bits are preceded by a start bit and followed by an optional parity bit and one or more stop bits.

Flow control - A method of controlling the amount of data that two devices exchange. In data communications, flow control prevents one modem from "flooding" the other with data. If data comes in faster than it can be processed, the receiving side stores the data in a buffer. When the buffer is nearly full, the receiving side signals the sending side to stop until the buffer has space again. Between hardware (such as your modem and your computer), hardware flow control is used; between modems, software flow control is used.

Handshaking - Is the way in which the data flow between computers/hardware is regulated and controlled. Two distinct kinds of handshaking are described: Software Handshaking and Hardware Handshaking. An important distinction between the kinds of signals of the interface is between data signals and control signals. Data signals are simply the pins which actually transmit and receive the characters, while control signals are everything else.

Parity - In data communications, parity is a simple procedure of checking the integrity of transmitted data. The most common type of parity is Even, in which the number of 1's in a byte of data add up to an even number, and None, in which a parity bit is not added.

PC - abbreviation for a Personal Computer.

RS232, RS423, RS422 AND RS485 - The Electronics Industry Association (EIA) has produced standards for RS232, RS423, RS422, and RS485 that deal with data communications. EIA standards where previously marked with the prefix "RS" to indicate the recommended standard. Presently, the standards are now generally indicated as "EIA" standards to identify the standards organization.

Electronic data communications will generally fall into two broad categories: single-ended and differential. RS232 (single-ended) was introduced in 1962. RS232 has remained widely used, especially with CNC control builders. The specification allows for data transmission from one transmitter to one receiver at relatively slow data rates (up to 20K bits/second) and short distances (up to 50' @ the maximum data rate). This 50' limitation can usually be exceeded to distances of 200' or more by using low capacitance cable and keeping the data rates down to 9600 baud and lower.

RTS/CTS Hardware handshaking - uses additional wires to tell a sending device when to stop or start sending data. DTR and RTS refer to these Hardware handshaking lines. you can select whether you need to use DTR or RTS individually, or use both lines for hardware handshaking. See also Xon/Xoff.

Stop bits - In data communication, one or two bits used to mark the end of a byte (or character). At least one stop bit is always sent.

2 License, Registration and technical support

2.1 License

Copyright © 1999-2016 AGG Software.
All Rights Reserved

SOFTWARE LICENSE

Trial Limited Version

The trial limited version of this software may be used for evaluation purposes at the user's own risk for a trial period. At the end of the trial period, the user must either purchase a license to continue using the software, or remove it from his/her system.

The trial limited version may be freely distributed, provided the distribution package is not modified. No person or company may charge a fee for the distribution of Advanced Serial Port Monitor without written permission from the copyright holder.

Licensed Version

On payment of the appropriate license fee, the user is granted a non-exclusive license to use Advanced Serial Port Monitor on one computer (i.e. a single CPU), for any legal purpose, at a time. The registered software may not be rented or leased, but may be permanently transferred, if the person receiving it agrees to terms of this license. If the software is an update, the transfer must include the update and all previous versions.

Whilst every care has been taken in the construction and testing of this software, it is supplied subject to the condition that the user undertakes to evaluate the suitability of the control for his/her purposes. AGG Software makes no representation of the software's suitability for any purpose, and the user agrees that AGG Software has no responsibility for any loss or damage occasioned by the use of this software.

TO THE MAXIMUM EXTENT PERMITTED BY APPLICABLE LAW, THE SOFTWARE AND DOCUMENTATION ARE PROVIDED "AS IS" AND AGG SOFTWARE DISCLAIMS ALL OTHER WARRANTIES AND CONDITIONS, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, CONFORMANCE WITH DESCRIPTION, TITLE AND NON-INFRINGEMENT OF THIRD PARTY RIGHTS.

TO THE MAXIMUM EXTENT PERMITTED BY APPLICABLE LAW, IN NO EVENT SHALL AGG SOFTWARE BE LIABLE FOR ANY INDIRECT, INCIDENTAL, CONSEQUENTIAL, SPECIAL OR EXEMPLARY DAMAGES OR LOST PROFITS WHATSOEVER (INCLUDING, WITHOUT LIMITATION, DAMAGES FOR LOSS OF BUSINESS PROFITS, BUSINESS INTERRUPTION, LOSS OF BUSINESS INFORMATION, OR ANY OTHER PECUNIARY LOSS) ARISING OUT OF THE USE OR INABILITY TO USE THE SOFTWARE PRODUCT, EVEN IF AGG SOFTWARE HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES. IN ANY CASE, AGG SOFTWARE'S CUMULATIVE AND ENTIRE LIABILITY TO YOU OR ANY OTHER PARTY FOR ANY LOSS OR DAMAGES RESULTING FROM ANY CLAIMS, DEMANDS OR ACTIONS ARISING OUT OF OR RELATING TO THIS AGREEMENT SHALL NOT EXCEED THE PURCHASE PRICE PAID FOR THIS LICENSE.

Should any term of these terms and conditions be declared void or unenforceable by any court of competent jurisdiction, such declaration shall have no effect on the remaining terms hereof.

If you do not agree to these conditions you should not install this software.

2.2 Limitations

Program is distributed on shareware terms. This means limited and unavailable secondary program possibilities, which become valuable or available after program registration. To register the program read here.

In trial version of our program are the following limits:

- Trial period is limited by 15 days. After that time program won't work until it is registered.
- Continuous program work time is limited. After set period a message will be displayed and program stops its work;
- In spy mode you can receive only first 1024 bytes.

2.3 How to register

The program is distributed on shareware terms. This signifies limited or unavailable secondary capabilities of the program, getting of full value or available after program registration.

If you'd like to be a registered user, to get information about the release of new versions, to use technical support and, at last, to get access to disabled functions of the program, register your program version. For registration read license agreement.

To buy a program trough Internet visit registration page of our site. On this page you can get the newest information about registration process, and also find link to order registration. After you've have the form of order registration. Enter your personal particulars and choose the most convenient payment method for you. Further you'll get notification and follow the notes in it.

More information about license and payment terms you may find on our registration page of our site.

2.4 Support

Technical questions	support@aggsoft.com
Common questions	support@aggsoft.com
Sales questions	sales@aggsoft.com

3 Installation

3.1 Installation process

Run aspmon4.exe.

If any beta-version was installed on your computer, remove it.

Quit of the working Advanced Serial Port Monitor on installation time.

By default, Advanced Serial Port Monitor will be installed to the directory "/Programs Files/Advanced Serial Port Monitor" of your system disk, but you can change this path.

In standard distributive of Advanced Serial Port Monitor are no additional modules files, which you can download from our site.

3.2 System requirments

Windows 2000+ (x86 or x64) (you need administrator rights to use the spy mode).
Windows Server 2003+ (x86 or x64) (you need administrator rights to use the spy mode).

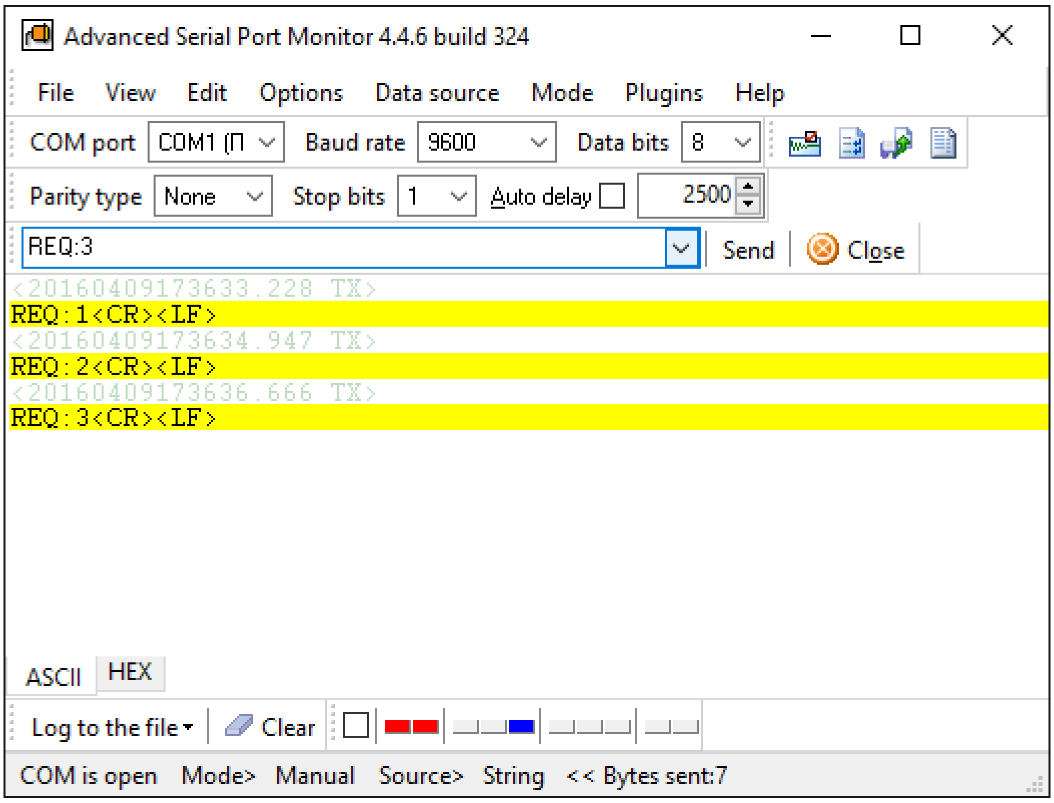
It is necessary to have at least one free COM port, not busy by any device (mouse, for example) to connect external device.

4 Program use

4.1 First run

Thanks for looking at our program and reading this manual. This chapter shows how quickly start working with the program and get the result. More detailed manual read in the next chapters.

On the first run, just after installation, You will get into the main window (pic.1).



COM port COM1 (П Baud rate 9600 Data bits 8
Parity type None Stop bits 1 Auto delay 2500
REQ:3

```
<20160409173633.228 TX>
REQ:1<CR><LF>
<20160409173634.947 TX>
REQ:2<CR><LF>
<20160409173636.666 TX>
REQ:3<CR><LF>
```

COM is open Mode> Manual Source> String << Bytes sent:7

Pic.1. Main window.

To start work with COM port at once execute the following actions:

1. Setup connection settings (baud rate, bits data number, stop bits number, etc.);
2. Select from the list of available COM ports the one you need;
3. Click "**Open**". After that COM port will be unavailable for other programs, because the program uses it in monopoly mode. If COM port was busy by other application before, warning message will be displayed;
4. Press "F4" key, setting up necessary data in the edit box and click "**Send**" in program window or press "F4" key;
5. Configure, enable or disable writing to file with "Write to file" drop-down button.

Now your data were sent to COM port. Received data will be viewed on the display.
To select file as data source set file name. For quick access to this option use button with the picture of diskette.

Switch between program work modes is done with menu item "**Mode**".

Data and window view can be edit by pop up menu in data window (right-click over data window and select item).

4.2 How to use

In the main window connection parameters must be set, such as baud rate, data bits number, stop bits number and etc. If you don't know what it is, first read information about COM ports and serial data flow method (see Internet links below).

Advanced protocol settings of data transmitting and data transmitting control methods are available in options. Program work is available in interface mode RS-485, which provides half duplex data transmit, in which the program controls RTS lines for selecting data flow direction.

Then click "Open" button. COM-port (exclusive) opening with set number will be done. In case of lucky opening , there will be a message in status string and COM-port becomes available for sending and receiving data.

Note: Under COM-port number is meant not physical device position but the number set to COM-port in properties of Windows. This number is connected with device address (UART-device) in this system. This address can be changed in BIOS/Setup settings.

Attention:If you don't know what is BIOS or UART, read and don't make any settings in Setup. It can give undesirable results.

Also in the main window are (look pic.1 in the previous chapter):

- Data output window;
- Data edit box and "Send" button (it is available in manual mode only);
- Serial port parameters control toolbars;
- Data display window clear button;
- Drop-down menu item, where you can enable or disable writing to protocol file;
- Serial line current state toolbar, some signals of which you can control manual;
- Buttons for quick access to frequently used options;

Select program work mode:

- Manual control - in this mode you can send and received data. Data are sent on pressing "Send" or automatically. Automatically ASPM sends data last entered in edit box, over interval, defined in "Delay" field in the main window;
- Spy - in this mode is data transmit over COM-port by other program. That's why in this mode communication parameter setting is not available, beside spy COM-port number;
- Plugins - in this mode control is given to the selected (additional) plug in module, which controls data receive and transmit. For more productive work you can run several plugins.

More detailed information read in the corresponding chapter. The selected mode makes available that or those control elements.

In manual and auto modes you can use file as data source. It can be text or binary file. Data source choice is made through menu item "Data source".

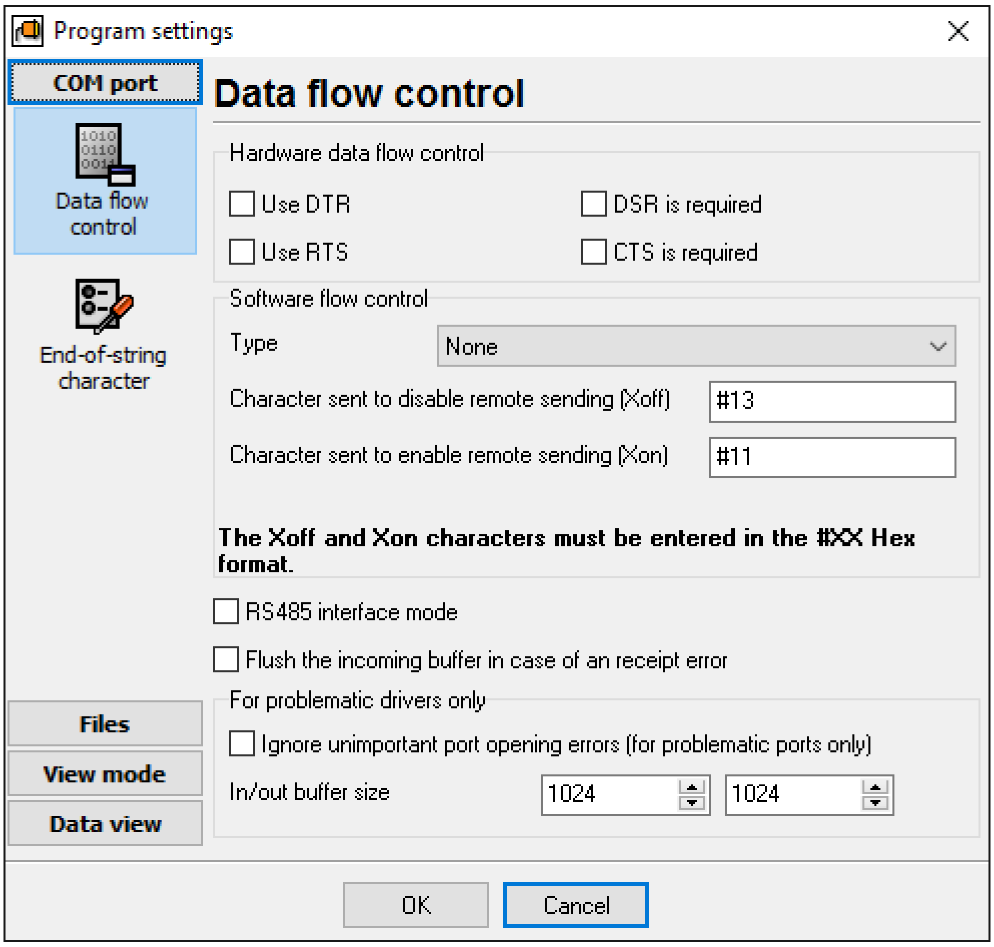
- Input string - for data sending use input string in the main window. You can make data packet, containing characters with code < #20h, sending of which will be done according to selected work mode;
- File - data are taken from file, which you selected in options. Until file is not set this mode is unavailable.

In detail about each data source read in corresponding chapter.

If there is also data receive in program work, they are displayed in data receive window. Format and data display method are set in options. There are many settings for more effective work.
Data receive window can be cleaned manual if you need or set the size of screen buffer in options.

Received and sent data can be saved in one or different files. Extended settings of file format and data separation are available in options.

To make complete program setting go to menu item "**Options/Setting**". In the pop up window (pic.1), going from one tab to other make the necessary changes.



Data flow control

Hardware data flow control

☐ Use DTR ☐ DSR is required

☐ Use RTS ☐ CTS is required

Software flow control

Type: None

Character sent to disable remote sending (Xoff): #13

Character sent to enable remote sending (Xon): #11

The Xoff and Xon characters must be entered in the #XX Hex format.

☐ RS485 interface mode

☐ Flush the incoming buffer in case of an receipt error

For problematic drivers only

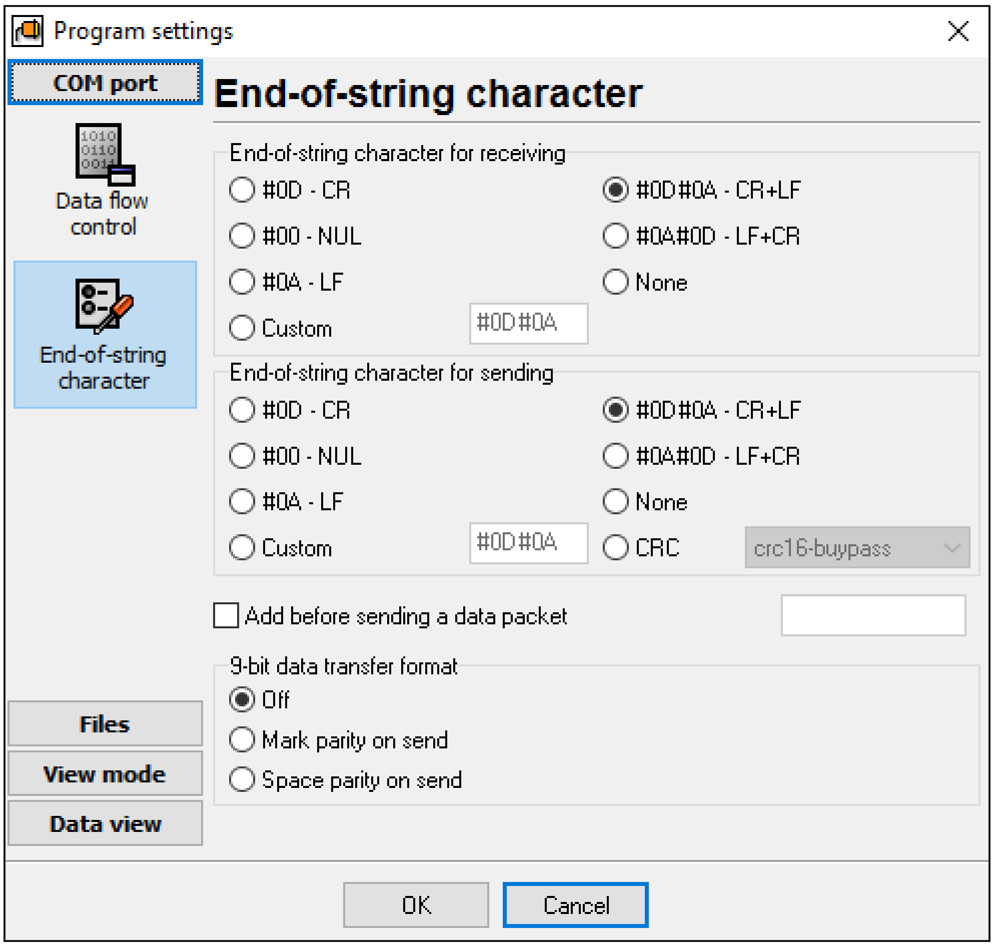
☐ Ignore unimportant port opening errors (for problematic ports only)

In/out buffer size: 1024 1024

Pic.1 Options window example

4.3 The sign of the end of string

For identification of transmitting data packets you can add to the packet end its end sign. The program will extract data packets from common incoming information stream if string on receive is set. These parameters are set up in Options on "**COM port/Signs of end of string**" tab (pic.1).



Program settings

COM port

Data flow control

End-of-string character

Files

View mode

Data view

End-of-string character

End-of-string character for receiving

#0D - CR | [✓ #0D#0A - CR+LF]
#00 - NUL | #0A#0D - LF+CR
#0A - LF | None
Custom: #0D#0A

End-of-string character for sending

#0D - CR | [✓ #0D#0A - CR+LF]
#00 - NUL | #0A#0D - LF+CR
#0A - LF | None
Custom: #0D#0A | CRC: crc16-buypass

☐ Add before sending a data packet

9-bit data transfer format

[✓ Off]
Mark parity on send
Space parity on send

OK | Cancel

Pic.1. Sign of end of string

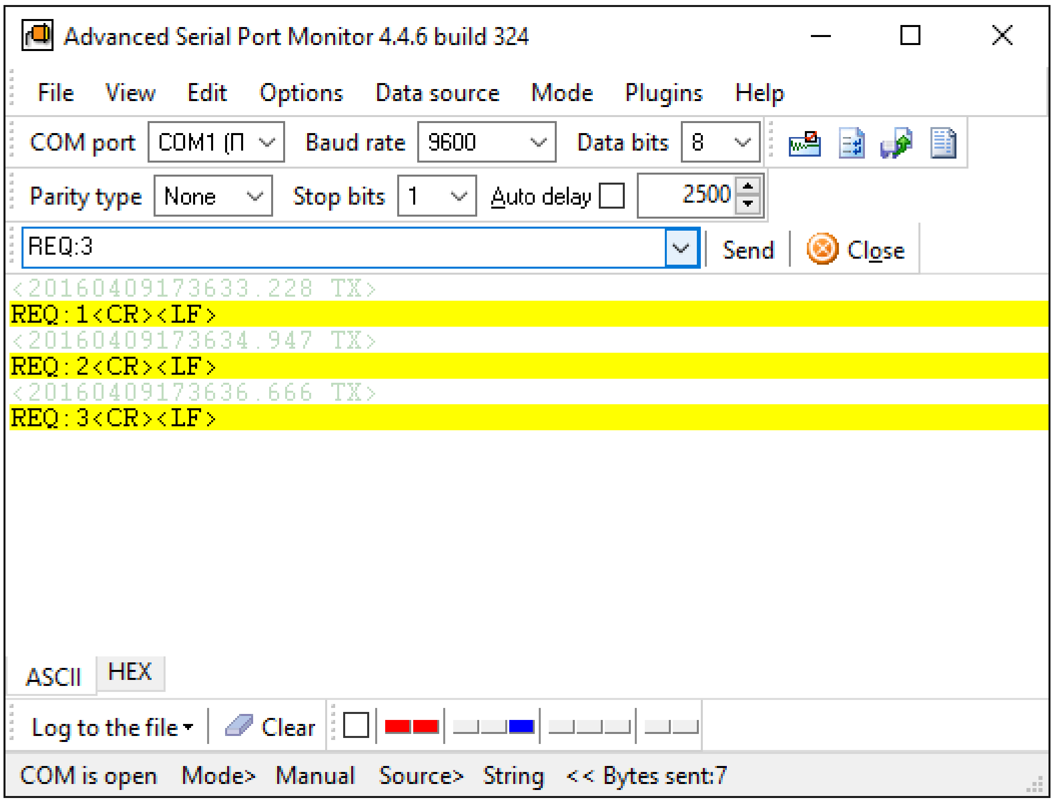
In time of program work, the selected character or string will complete a data packet. In output this string will be added to the text, entered in input string or read from ASCII file. On data receive this character is the sign, on which string length will be counted and new string on display will be formed. Also this character considers event for plug in on the next string receive.

It is possible to consider string and character(s) on receive and transmit. And besides these characters can be different. Also you can disable any of the given characters (or combinations) selecting "None".

You can specify your custom end character(s). Just type it in the "Custom" input field. For example, if you need to set up ETX as end character, then type #03. Here 03 is hexadecimal value of the ETX ASCII code.

You need set up these signs for formatted display in data window. For example, if your device sends strings, which terminates with 0D byte, then specify "#0D - CR" in the group "**String or character**

on receive" and you'll see in data window string by string as on pic.2.



COM port COM1 (П   Baud rate 9600   Data bits 8
Parity type None   Stop bits 1   Auto delay ☐ 2500
REQ:3

```
<20160409173633.228 TX>
REQ:1<CR><LF>
<20160409173634.947 TX>
REQ:2<CR><LF>
<20160409173636.666 TX>
REQ:3<CR><LF>
```

COM is open   Mode> Manual   Source> String   << Bytes sent:7

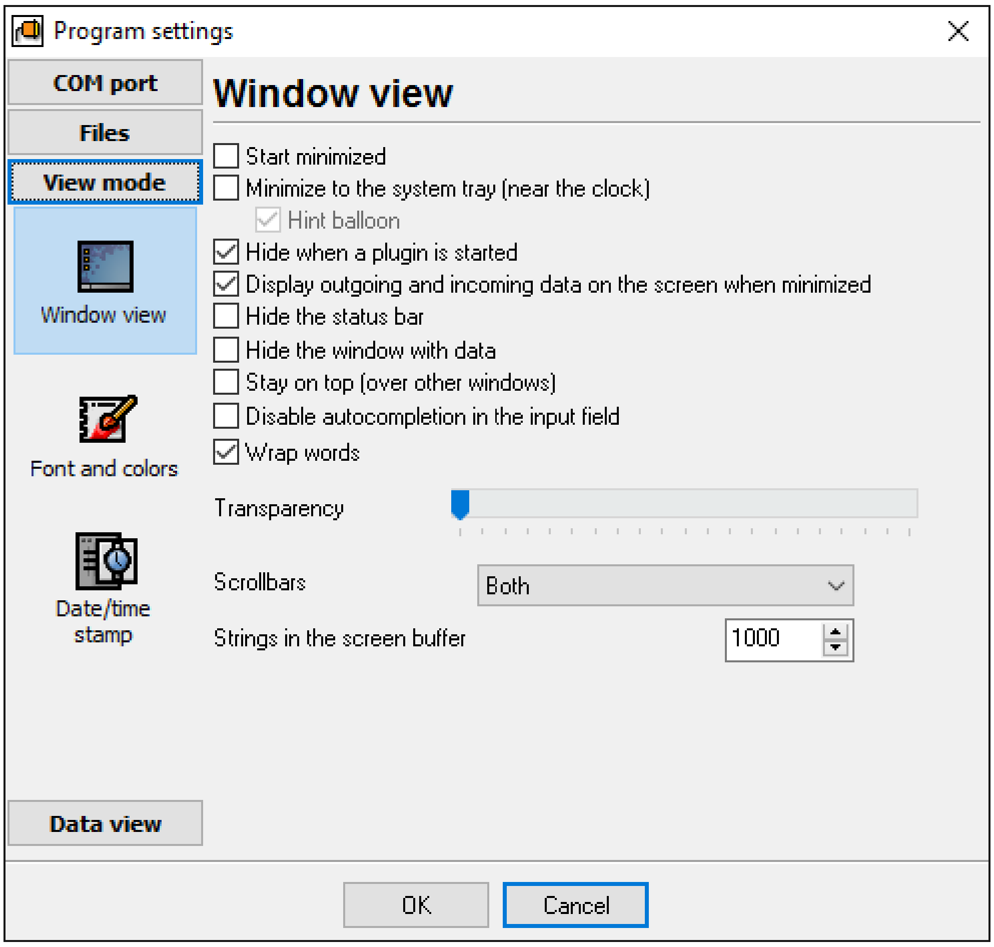
Pic.2 String by string

Some devices needs special framing characters in their requests. For example, common case is using STX and ETX ASCII characters as framing characters. Yes, of course, you can type this characters manually in each case, when you need send data to your device, but is more efficient specify #03 as terminate character on transmit and enable option "**Add before data packet on transmit**" and type #02 in a input field.

4.4 Window views and data display

Comfort and friendly interface is one of advantages of the program. You can set up many small interface details from background color to toolbar position in the main window. All settings are auto saved on exit and are restored on the next run.

In options on "**Other/Window view**" tab you can set up (numbered from top to bottom, pic.1):


Program settings

COM port | Files | View mode | Window view | Font and colors | Date/time stamp | Data view

Window view

- [ ] Start minimized
- [ ] Minimize to the system tray (near the clock)
  - [✓] Hint balloon
- [✓] Hide when a plugin is started
- [✓] Display outgoing and incoming data on the screen when minimized
- [ ] Hide the status bar
- [ ] Hide the window with data
- [ ] Stay on top (over other windows)
- [ ] Disable autocompletion in the input field
- [✓] Wrap words

Transparency

Scrollbars: Both

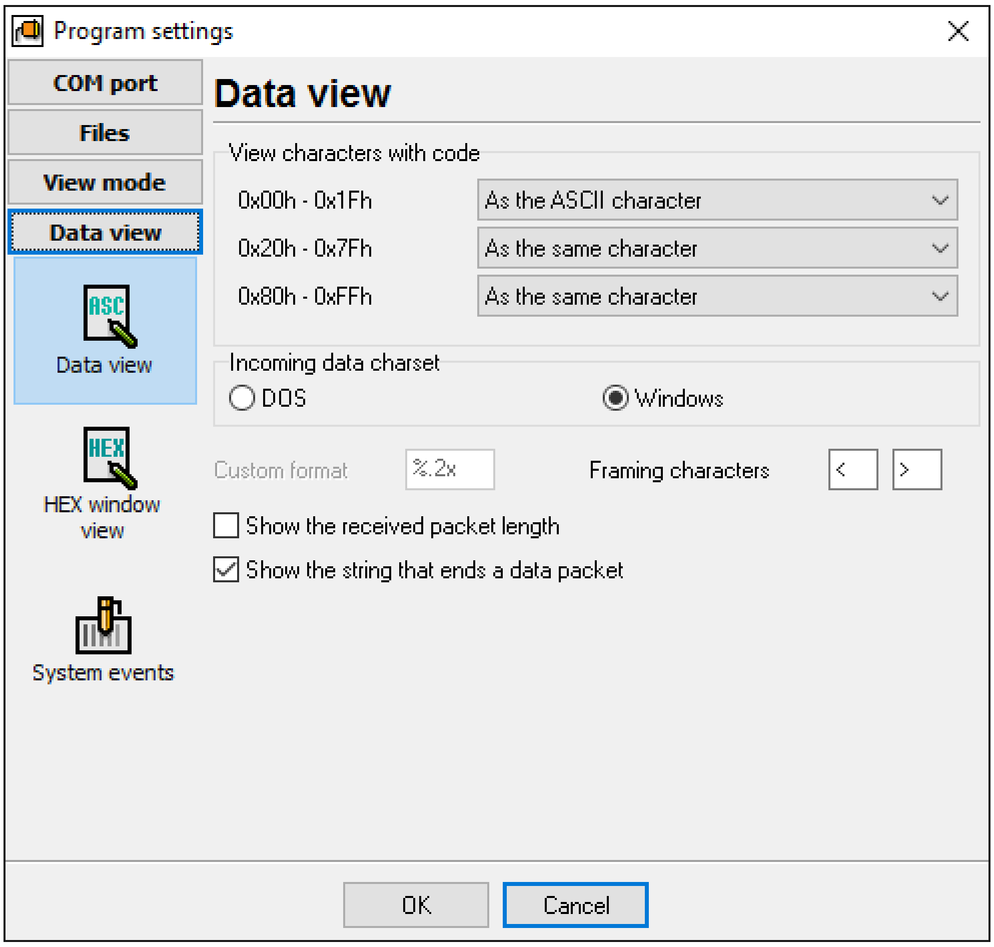
Strings in the screen buffer: 1000

OK | Cancel

Pic.1. Window view

1. **Start up in minimized state** - on program start, the main window is not displayed on the desktop, but puts its icon in the system panel;
2. **Minimize to Systray (a panel near clock)** - you may select that the program will place its icon on minimizing on the panel near clock (SysTray). Then, if you need, click left mouse button to restore former window position. On right mouse button click on icon, context menu will pop up. When cursor is over icon in Systray short Advanced Serial Port Monitor current state will pop up.
3. **Hide on plug start up** - main window will hide before plug in work and will show after closing the plugin window.
4. **Display data sent and received in minimized state** - data will be displayed only when main window is seen. In some cases it can minimize CPU load.
5. **Hide status bar, window with data** - You can change window view, extending it vertically and horizontally. Also you can delete status bar or data receive window from view.
6. **Screen view mode** - help you to set up font and font size, background color, characters color in data window. **Transparency** of main window help you to view state of other programs or desktop below ASPM window. **Wrap words** allows you display all data on screen at same time. You need it, if you specify "None" as end of string. ASPM will display all data received as one long string. And this options split long string to few others.
7. **Scroll bar** - view or not scroll bar in received data display window.
8. **Number of strings in screen buffer** - if string number, shown in this setting is exceeded, data window will be auto cleaned.

Also on tab "**Other/Data view**" (pic.2) you can set up data, displayed in program window:


Program settings

Data view

View characters with code

0x00h - 0x1Fh: As the ASCII character

0x20h - 0x7Fh: As the same character

0x80h - 0xFFh: As the same character

Incoming data charset: ( ) DOS [✓ Windows]

Custom format: %.2x

Framing characters: < >

☐ Show the received packet length

☑ Show the string that ends a data packet

Pic.2. Data view

1. **View characters with code < 20 Hex as** - since data are from bytes you can set up data byte view with code >= 20 Hex. These data doesn't have the corresponding character, that's why these data can be displayed only in hex, dec or ASCII code. But you can specify your own format of data output (please, see below)
2. **View characters with code >= 20 Hex as** - since received data are from bytes you can set up data byte display with code<20 Hex. You can set to display these data in hex and dec code or in character, corresponding to this code. You can specify your own format of data output too (please, see below).
3. **View received packet length** - packet length will be counted and output to data window;
4. **Highlight data sent on screen** - string with sent data will be highlighted by the set color.

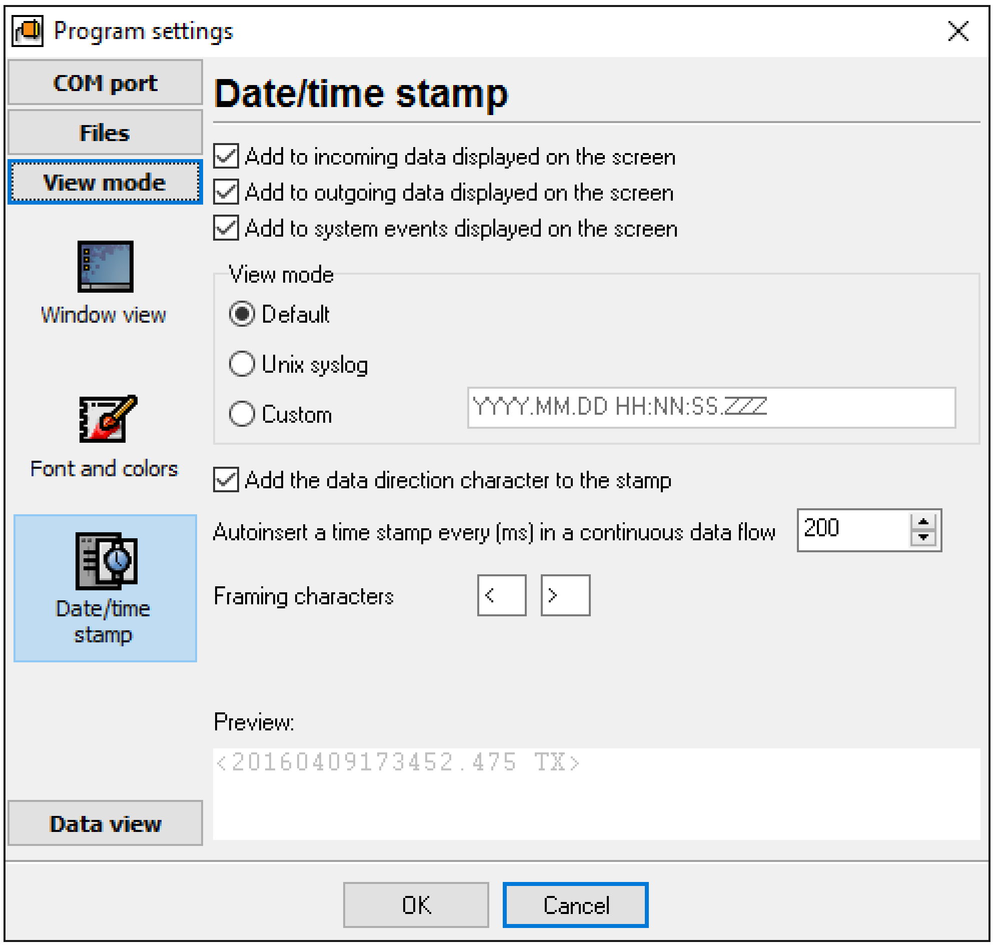
User's format of data view

You can set up own data display format. Directive %d in the "**User's format**" field shows to display decimal code. Directive %x - hexdecimal value. Directive %.2x - hexdecimal value with one leading zero before number (you can change number two to other). You can set other framing characters before and after these directives. For example, byte 12 (Dec) [%x] will output as [0D].

Fields "**Framing characters**" allows specify framing characters (as in the example above) but for predefined options in groups "**View characters with code < 20 Hex as**" and "**View characters with code >= 20 Hex as**".

4.5 Date/time stamp view

This tab allows you to add datetime stamp to display and file output. With options "**Add to display output for data received**" and "**Add to display output for data received**" you can enable stamp for each data direction. Then you can configure view mode of the stamp. Select necessary item from the "**View mode**" group and see results in the "Preview" box. The field "**Framing characters**" allows you to add character or string of characters to begin and end of stamp. With the "**Font color**" dialog window you can specify color of stamp on screen. If you'll analyze log file offline, possible, will useful for you, direction sign in the stamp. Add it with "**Add data direction sign to a stamp**" option.



Date/time stamp

[✓] Add to incoming data displayed on the screen
[✓] Add to outgoing data displayed on the screen
[✓] Add to system events displayed on the screen

View mode
● Default
○ Unix syslog
○ Custom YYYY.MM.DD HH:NN:SS.ZZZ

[✓] Add the data direction character to the stamp

Autoinsert a time stamp every (ms) in a continuous data flow 200

Framing characters < >

Preview:
<20160409173452.475 TX>

Pic.1. Datetime stamp customization

4.6 Command line parameters and scripting language

With command line parameters you can adjust ASPM options on start up. ASPM has only one command with following syntax.

aspmon -f script.txt

If you start the program with this command line you will specify ASPM to read commands from a script.txt file and execute it.

The short description of commands of script language is resulted in a separate help file.

4.7 Custom baud rates

You can specify any custom baud rates in the baud's list in the main window. Just open the file "custombaud.txt" that is being contained in the program folder with Notedpad or other text file editor and add your baud one per row. After editing, please, restart ASPM software.

File example:

```
; Please, add your custom baud rates here
; One baudrate per row. Please, use digits only
; After you finished, please, restart ASPM
28800
```

4.8 Files

4.8.1 Files

In Advanced Serial Port Monitor are two file types.

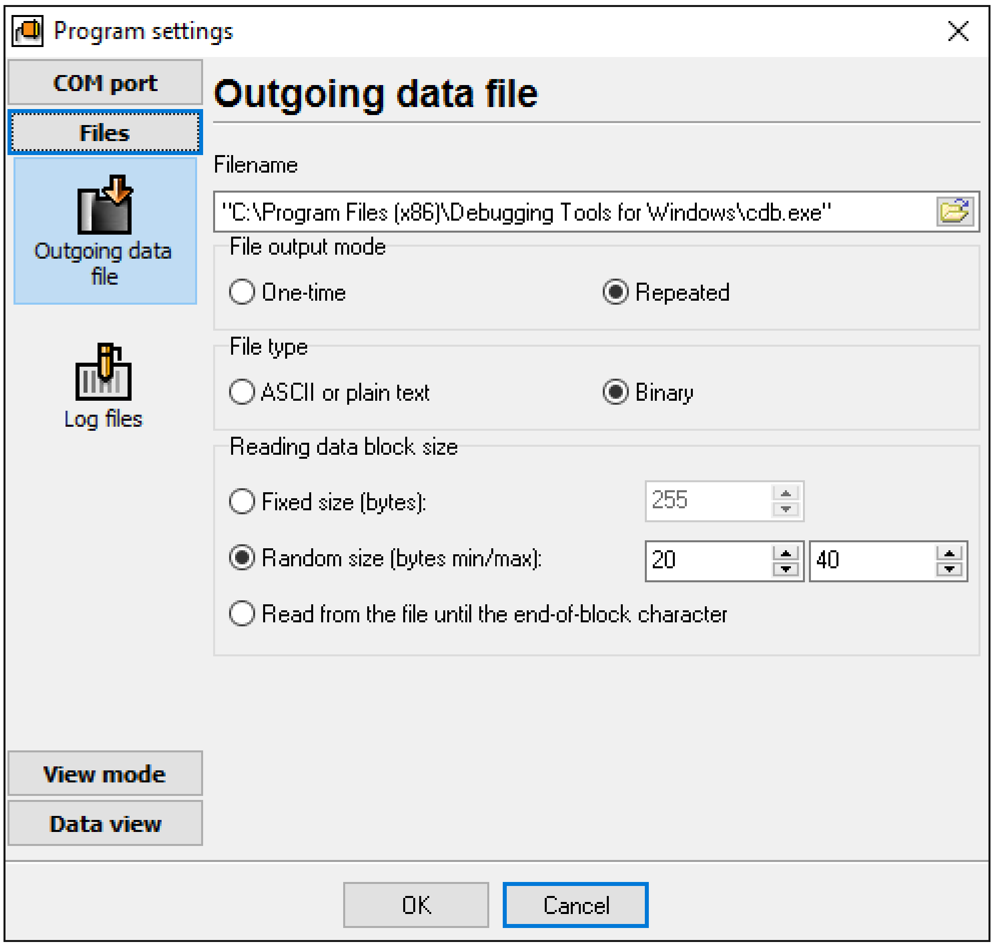
- Data source(sent) file - from this file data for transmit through serial port are read;
- Protocol files - used for data saving, received or sent through serial port.

Mode setting with each mentioned file types is available in options.

In some modes is possible duplication of data source in the other file, for creation of data exchange protocol.

4.8.2 Source file

For easy process of great volume data transmit or repeated data blocks you can use source files (pic.1).



Outgoing data file

Filename: "C:\Program Files (x86)\Debugging Tools for Windows\cdb.exe"

File output mode: ( ) One-time [✓ Repeated]

File type: ( ) ASCII or plain text [✓ Binary]

Reading data block size:
( ) Fixed size (bytes): 255
[✓ Random size (bytes min/max): 20 40]
( ) Read from the file until the end-of-block character

Pic.1. Data source files

Program can work with the following file types:

- **Text (ASCII)** - data from this file type are read line by line. On reading string from file the existing end string sign is cut, on data transmitting is added (or not) end string sign, set in program settings;
- **Binary** - data from this file type are read by blocks. End block sign is end string sign, set in the program for sending data. On block reading it is completely and without any modifications is sent through COM port. When in binary file won't be found end block sing, the whole binary file will be sent as one block.

When transmitting data constantly, set file output mode. If you select **repeated** file output mode, at the file end data flow will continue on the next transmitting from the first file byte.

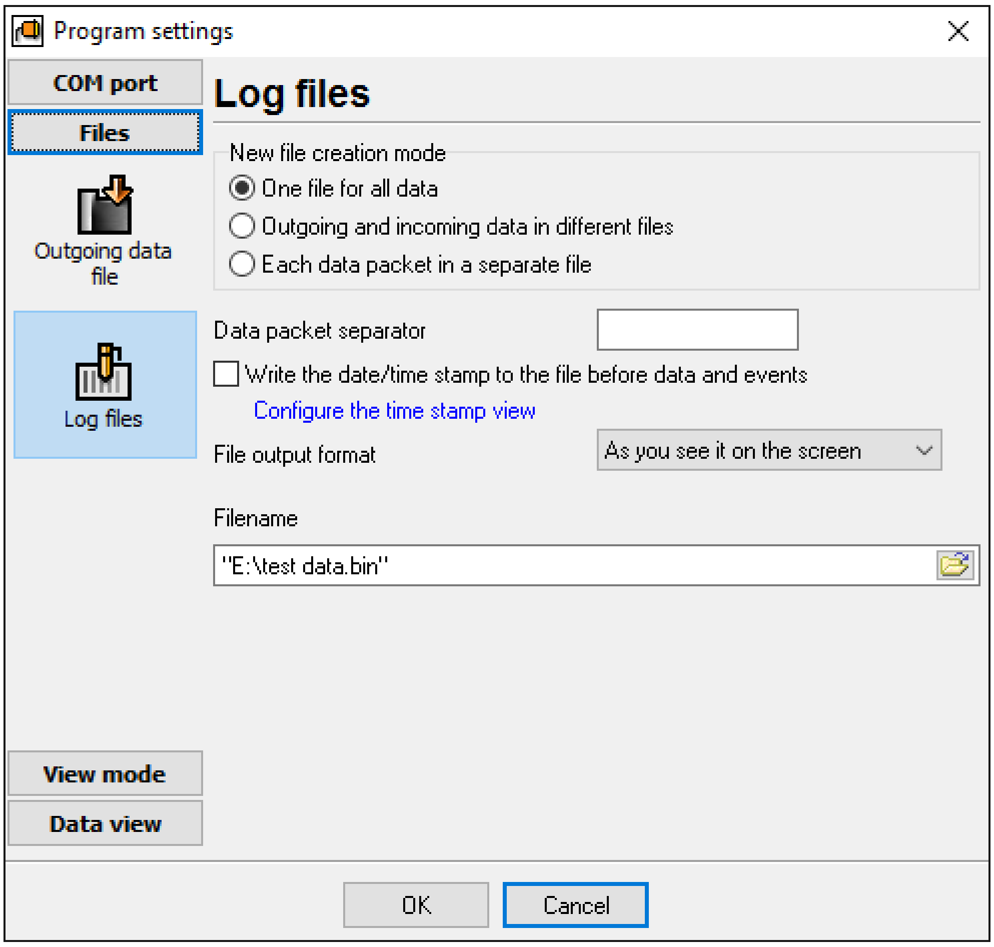
When you didn't set source file, you couldn't select file as data source.

Sometimes data flow protocoling is needed. Then set to output data source to file in menu

"Options" of the main program window. Protocol mode is described in the next chapter.

4.8.3 Protocol files

This file type let save all data exchange story between Advanced Serial Port Monitor and the receiving side. In the protocol file can be written both receiving and transmitting data. Data source write access to protocol file is set in menu "Options" of the main window (pic.1). For quick access to file forming setting window use button in the main window.



Log files

New file creation mode
[✓ One file for all data]
Outgoing and incoming data in different files
Each data packet in a separate file

Data packet separator

☐ Write the date/time stamp to the file before data and events

Configure the time stamp view

File output format: As you see it on the screen

Filename: "E:\test data.bin"

Pic.1 Protocol files configuration

There are some possibilities of protocol file forming:

- **One file for all data** - all data are protocoled in one file, name and position of which you show;
- **Data sent and received in different files** - sending and receiving data can be placed in separate files. You need to specify file path and name for each data direction;
- **Each sent and received packet in different file** - in this mode each sent and received data portion is written into separate file. Set the folder, where files will be created. File name is formed from prefix (which you can set up) and auto increasing number. Incoming data packets must be formed by sending side with frequency not less than 300ms. Otherwise receiving data will be written as one packet. Since file name is auto formed, such situation is possible when file with the name of already existing file will be formed. Then former file will be deleted and new

file will be written on its place. You can set up that new data will be written at the end of existing file.

Write activation into protocol file of receiving data is on pressing "hot key" or at file write access state change in the main window.

With field "**Separator between data packets**" you can specify character or string of characters, which ASPM will insert in the file between portions of data. This option helps you to analyze log files later, in the offline.

As separator you can use datetime stamp. Enable it with "**Write date/time stamp to file before writing data**" options. ASPM will insert in the file between portions of data too. Format of stamp you can configure with additional dialog window.

Attention: On write activation to file in the first two modes all data in it, if it existed would be saved, that' s why watch file size not to have unlimited volume growth!

Exists two modes of protocol files creation:

1. **Without any changes** - data receiving and sent are saved to file without any modification. View on screen and data in file will have different view. To analyze data in formed files you can using any other software (Notepad for example). Binary data can be analyzed with the help of any program for work with hex data. We recommend to use Total Commander as instrument combining all necessary functions;
2. **As you see it on screen** - data receiving and sent are saved to file with same format as on screen in the main program window (without colors).

I you need data in the file with text highlighting, then:

1. Set up high screen capacity;
2. When you'll receive and sent all necessary data right-click over data window and select "**Save screen to file**";
3. Select type of file: **RTF** or **HTML**. These file types supports text highlighting;
4. Type file name and click "Save";
5. Later, RTF file you can open with Wordpad or Microsoft Word, and HTML file with any web browser, Internet Explorer for example.

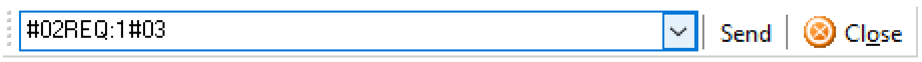
4.9 Modes

4.9.1 Manual control

In this mode data sending through COM port is made by clicking "Send". Input string or data file can be data source.

If data source was input string

Unempty string from the input field in the main window (pic.1) or last sent value will be send. Edit box will available only if you're opened serial port with "Open" button. Then you can enter any data and click "Send" button. How can you enter data in this box, please read below.



Pic.1. Edit box in the main window

If data source was file

the following data packet from the selected file. Data packet formation is done due to description, given before.

Also You can send data automatically. In this mode data from input string or file are sent over interval,set in the field "Delay".

If data source are taken from file, so for text (ASCII) files is used the next string, for binary files - data portion before the next end string sign on transmit. If end string sign is not set, the **WHOLE** file will be send.

Delay - waiting time in milliseconds before sending the next data portion from file or string from input field in auto mode. Delay is set in milliseconds due to requirements.

4.9.2 Spy mode

In this mode Advanced Serial Port Monitor doesn't send and receive any data, and only spies data exchange, made by other programs.

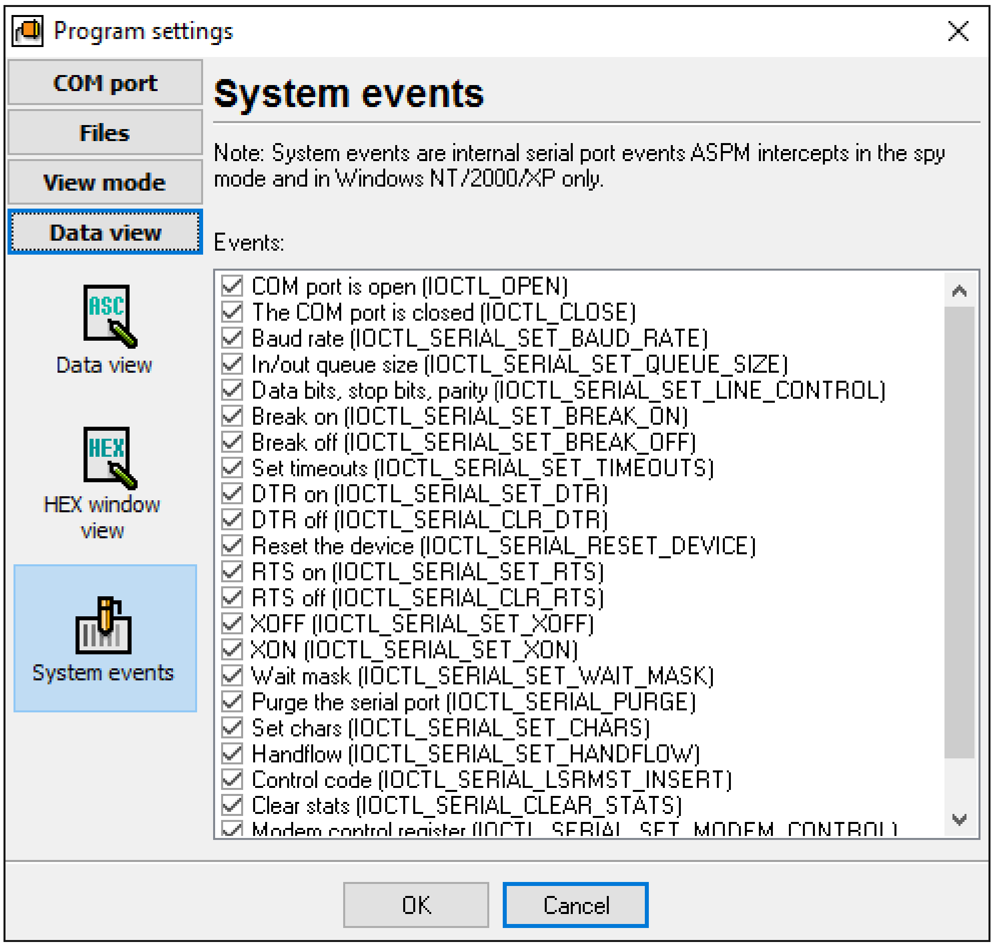
To spy received and sent data:

1. Select "Spy mode" from "Mode" menu in the main window;
2. Configure "Data view" mode in the options. Be sure, that you're enabled output of all characters types;
3. Don't forget to enable option in menu "**Options/Output data sent on screen**" to spy data sent by the given program (if necessary).
4. Open serial port **before** running the given program. It's very important. If you're opened serial port after start of the given program, then you will not get any output on the screen.

If the given program sends or receives data over COM port, the whole data exchange process will be displayed in data receive window. Also it is possible to keep data exchange protocol.

To exit Advanced Serial Port Monitor close the given program or stop data exchange over COM port in it.

Advanced features for Windows NT4, Windows 2000 and Windows XP users are available:
- you can monitor modem connections. If a modem are installed in Windows OS, you may select your modem name from the port's list;
- you can monitor internal port events (system events), such as port opening, port closing, set handshake etc. You can select a list of events, that you'll monitor (fig. 1) and specify color for it.



System events

Note: System events are internal serial port events ASPM intercepts in the spy mode and in Windows NT/2000/XP only.

Events:

- ☑ COM port is open (IOCTL_OPEN)
- ☑ The COM port is closed (IOCTL_CLOSE)
- ☑ Baud rate (IOCTL_SERIAL_SET_BAUD_RATE)
- ☑ In/out queue size (IOCTL_SERIAL_SET_QUEUE_SIZE)
- ☑ Data bits, stop bits, parity (IOCTL_SERIAL_SET_LINE_CONTROL)
- ☑ Break on (IOCTL_SERIAL_SET_BREAK_ON)
- ☑ Break off (IOCTL_SERIAL_SET_BREAK_OFF)
- ☑ Set timeouts (IOCTL_SERIAL_SET_TIMEOUTS)
- ☑ DTR on (IOCTL_SERIAL_SET_DTR)
- ☑ DTR off (IOCTL_SERIAL_CLR_DTR)
- ☑ Reset the device (IOCTL_SERIAL_RESET_DEVICE)
- ☑ RTS on (IOCTL_SERIAL_SET_RTS)
- ☑ RTS off (IOCTL_SERIAL_CLR_RTS)
- ☑ XOFF (IOCTL_SERIAL_SET_XOFF)
- ☑ XON (IOCTL_SERIAL_SET_XON)
- ☑ Wait mask (IOCTL_SERIAL_SET_WAIT_MASK)
- ☑ Purge the serial port (IOCTL_SERIAL_PURGE)
- ☑ Set chars (IOCTL_SERIAL_SET_CHARS)
- ☑ Handflow (IOCTL_SERIAL_SET_HANDFLOW)
- ☑ Control code (IOCTL_SERIAL_LSRMST_INSERT)
- ☑ Clear stats (IOCTL_SERIAL_CLEAR_STATS)
- ☑ Modem control register (IOCTL_SERIAL_SET_MODEM_CONTROL)

Fig.1 System events.

4.9.3 Plugin mode

In this mode Advanced Serial Port Monitor can extend its capabilities with additional modules.

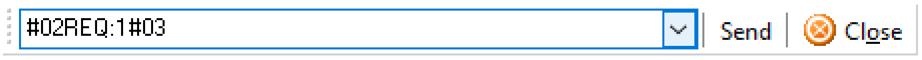
In simple case emulate device work, working over interface RS232 or RS485. Emulation process is defined by plug in. In plug in mode Advanced Serial Port Monitor on data receive gives control to plug in and also is in spy mode of commands from plug in. Received data in this mode are not displayed in data receive window. To stop work of plug in close its window.

You can develop plugin modules for ASPM. The interface is described in a separate SDK help file. Example of ASPM 's plugin with comments is available on our site.

4.10 Data source

4.10.1 Input string

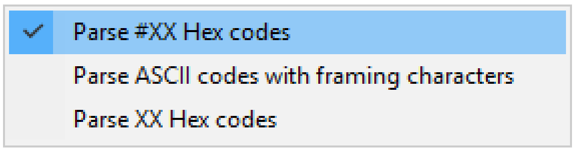
Input string (edit box, see pic.1) is for quick information transmit over COM port under user's control. When selecting this source, data packet on transmit is formed from character input string, only when the string is not empty, otherwise the last sent value will be taken. The edit box will available only when a serial port will open.



Pic.1. Edit box in the main window

If you need to enter special characters with byte code < 32 Dec (ASCII codes) you can do the following:

1. Go to "Options" -> "Special characters parsing" menu item and select (pic.2):



Pic.2 Parse mode selection

Parse #XX hex codes - then you need to enter special characters as #XX, where XX is hexadecimal value of character and # is character-prefix. If you need to sent character #, type it twice.

Example 1: You need to send character with code 128 (Dec) or 80 (Hex) then input string will be: #80.
Example 2: You need to send character with code 2 (Dec) or 2 (Hex) then input string will be: #02.
Example 3: You need to send string of the form #02MB then input string will be: ##02MB.

In this mode you can send byte with any code from 0 to 255. Just change XX to hexadecimal byte code.

Example 4: #02#20#21#22#03 will be sequence of bytes: 2, 32, 33, 34, 3

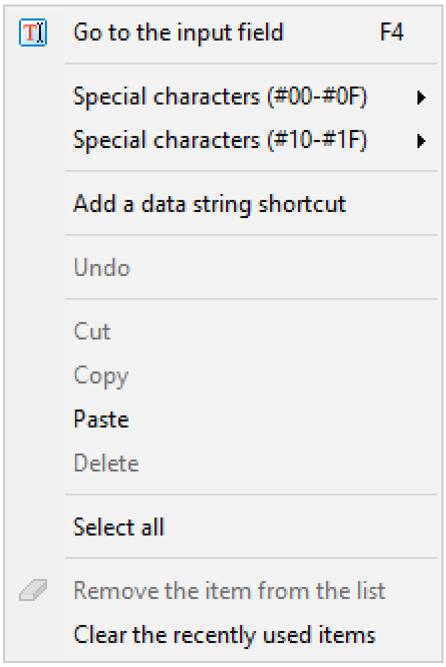
- **Parse ASCII codes with framing characters** - in this mode you can enter only valid ASCII codes from the list: NUL, SOH, STX, ETX, EOT, ENQ, ACK, BEL, BS, HT, LF, VT, FF, CR, SO, SI, DLE, DC1, DC2, DC3, DC4, NAK, SYN, ETB, CAN, EM, SUB, ESC, FS, GS, RS, US. Just type "<", then ASCII code above, ">" (without quotes) .

Example 5: <STX>Test<ETX>

Note: "<" and ">" is framing characters. You can change it in options, on "Data view" tab. If you'll change these characters, then you need to enter new characters in the edit box.

2. Right-click over edit box;
3. Popup menu will appear (pic.2);

4. Select special character from menu;
5. Special character selected will appear after cursor position.


Go to the input field F4
Special characters (#00-#0F)
Special characters (#10-#1F)
Add a data string shortcut
Undo
Cut
Copy
Paste
Delete
Select all
Remove the item from the list
Clear the recently used items

Pic.3 Popup menu

If you need to enter special characters with byte code >= 128 Dec (international characters) you can do the following:

1. Select "**Parse #XX hex codes"** as described above;
2. Type character byte code as #XX, as showed in examples 1-3.

To send the string, press "Enter" in input string or click "Send" button by mouse. Input string will be cleaned, sent value will be added to the list of sent values.

4.10.2 File

Look chapter "Data source file"

4.11 Scripts

4.11.1 Script Language

The basic syntax of the script language of Advanced Serial Port Monitor (ASPM) is:

```
  <command> <data1> <data2>          ;<comment>
```

where <command> describes the action to perform, <data1> and <data2> are optional arguments, and <comment> is an optional comment. The format of the arguments vary among commands. The various components of each line must be separated by at least one space or a comma. Additional spaces are permitted but ignored. Commands are not case-sensitive.

The following is a list of supported commands followed by brief descriptions and discussions of the relationships between various commands:

```
:<label>
;<comment>
INITPORT <COM1..COM99>
IF CONNECTED <label>
DONEPORT
SEND 'XXXXXX'
WAIT 'XXXX' <timeout in ms>
IF SUCCESS <label>
IF TIMEOUT <label>
IF FAIL <label>
WAITMULT 'XXX|YYY|ZZZ' <timeout in ms>
IF 1,2,3...127
GOTO <label>
DISPLAY 'XX XX'
SENDBREAK <duration in ms>
DELAY <duration in ms>
SET <option> <data>
RUN <command> <wait>
```

`:<label>`

A point in the script file that can be jumped to via a GOTO or IF instruction. A label name can be any type of string without embedded spaces. For example ':TopOfLoop', ':TOP_OF_LOOP' are both acceptable; ':top of loop' is not.

`;<comment>`

Any line that starts with a semicolon is considered a comment. Blank lines are also considered comments and may be freely added for readability.

`INITPORT` `<COM1..COM99>`

Opens the specified port. Only one port at a time may be opened.

DONEPORT

Closes a port previously opened with INITPORT.

SEND 'XXXXXX'

Transmits the string 'XXXXXX'. Control characters may transmitted by preceding a character with '^'. For example, a control C character is represented by ^C. You'll use this feature most often when sending carriage returns. For example, SEND 'myname^M' might be an appropriate response to a logon prompt where you would normally type your name and press <Enter>. NOTE: The control characters must be *inside* the quote marks, if quote marks are necessary. If the string does not contain any embedded blanks the beginning and ending quotes can be omitted. The quotes are required if the string has embedded blanks. Here are some examples to illustrate this point:

```
SEND ABC      sends ABC
SEND 'ABC'    sends ABC
SEND A B C    sends only the A ('B C' is considered a comment)
SEND 'A B C'  sends A B C
```

WAIT 'XXXXX' <timeout in ms>

Waits up to <timeout in ms> milliseconds for a particular received string. The string comparison is always case insensitive. However, the string comparison need not be complete. If, for example, a host returns the string 'Host XXXX ready' where XXXX might vary from session to session, the WAIT command should wait for 'ready' only. As with the SEND command, beginning and ending quotes are only required if the string contains embedded blanks. This command sets one of three conditions: SUCCESS, FAIL or TIMEOUT, which can be tested with the IF command. SUCCESS is set if the string is received before the timeout. TIMEOUT is set if the timeout expires before the string is received. FAIL is set if the timeout expires and all retries are exhausted.

IF SUCCESS/TIMEOUT/FAIL/RING/RTS/CTS/DTR/DSR <label>

Tests the condition set by the last command and, if the tested condition is true, script execution jumps to <label>. If the condition is not true then execution continues with the next statement.

RING/RTS/CTS/DTR/DSR condition tests corresponding pin state.

WAITMULTI 'XXX|ZZZ|YYY', <timeout in ms>

Waits up to <timeout in ms> milliseconds for one of several substrings. The bar character (|) separates the substrings. The comparisons are always case insensitive. The maximum length of the entire string is 255 characters. As with the SEND command, beginning and ending quotes are only required if the string contains embedded blanks.
This command sets a numeric condition result based on the substring received: '1' is set if the first substring is received, '2' is set if the second substring is received, and so on. If none of the strings are received then TIMEOUT is set; if all retries have been exhausted then FAIL is set.

Note: YYY is being tested first then ZZZ and XXX. ZZZ is being tested first then XXX. If YYY is a part of ZZZ or XXX, but a serial port received ZZZ or XXX, then YYY condition will be executed.

IF 1,2,3...127 <label>

Tests the condition set by the last WAITMULTI command and, if the tested condition is true, script execution jumps to <label>. If the condition is not true then execution continues with the next statement.

The following example sends a modem dial command, then waits for one of CONNECT, NO CARRIER, or BUSY responses. If none of the responses are received then control falls through to the GOTO statement:

```
send 'atdt260-9726^m'
waitmulti 'connect|no carrier|busy' 60000
if 1 HandleConnect
if 2 HandleNoConnect
if 3 HandleBusy
goto HandleTimeout
:HandleConnect
  ...proceed with session
:HandleNoConnect
  ...handle noconnect error
:HandleBusy
  ...handle busy error
...
```

`GOTO <label>`

Unconditionally jumps to <label>.

`DISPLAY 'Just did something'`

This can be used to monitor the progress of the script and to aid in debugging.

`EXIT`

Exit the program.

`SENDBREAK <duration in ms>`

Transmits a break of <duration in ms> milliseconds.

`DELAY <duration in ms>`

Delays for <duration in ms> milliseconds. The script doesn't yield during delays so keep the delays as short as possible.

`PLUGIN START <file name>`

Execute plugin module with `<file name>` file name. This plugin must exists in the Plugins folder of ASPM.

`PLUGIN STOP <file name>`

Stop `<file name>` plugin's execution. This plugin must exists in the Plugins folder of ASPM.

`PLUGIN SHOW <file name>`

Sets the plugin's main window to normal state.

`PLUGIN HIDE <file name>`

Sets the plugin's main window to minimized state.

`MODE <mode>`

Value:

- **manual** - manual mode;
- **spy** - spy mode.

`SOURCE <data source>`

Value:

- **string** - use the input string as a data source;
- **file** - use file as a data source.

CLEAR `- clear data in the screen.`

SET BAUD `<number>`

Sets the Baud property of ASPM.

SET DATABITS `<5,6,7,8>`

Sets the DataBits property of the ASPM. Allowable values are 5, 6, 7 or 8.

SET FLOW `<RTS/CTS,XON/XOFF,NONE>`

Sets flow control options for the ASPM. Allowable values are RTS/CTS for hardware flow control, XON/XOFF for software flow control, and NONE to turn off all flow control.

SET PARITY `<NONE,ODD,EVEN,MARK,SPACE>`

Sets the Parity property of the ASPM. Allowable values are NONE, ODD, EVEN, MARK or SPACE.

SET STOPBITS `<1,2>`

Sets the StopBits property of the ASPM. Allowable values are 1 and 2.

SET RETRY `<data>`

Sets an internal retry count that is incremented whenever WAIT or WAITMULTI result in a TIMEOUT condition. When <retry count> TIMEOUTs have occurred the FAIL condition is set. The default is 1, meaning no retries are attempted.

SET LEFT `<data>`
Sets the position of main window on screen.

SET TOP `<data>`
Sets the position of main window on screen.

SET WIDTH `<data>`
Sets the main window width in pixels.

SET HEIGHT `<data>`
Sets the main window height in pixels.

SET WINDOWSTATE `<data>`
Sets the main window state. 0 - maximized, 1 - minimized, 2 - normal state.

RUN `<command> <wait>`

Executes the specified command, batch file or program. <wait> can be *true* or *false*; and determines whether the script waits for the command to complete its execution. <wait> is *true* by default.

Note: the command should contain full path to batch file or program. If the path contains spaces then the command should be enclosed by "".

Command example: RUN "c:\Program Files\Advanced Serial Port Monitor\command.bat" false

Here's an example logon script showing how these commands might be used to log into a host or terminal server:

```
SET RETRY 10                  ;Try 10 times
:Again
SEND ^C                       ;Send an attention character
WAIT 'READY' 182              ;Wait 10 seconds for response
IF SUCCESS Logon              ;Got prompt, continue with logon
IF TIMEOUT Again              ;Try again if we timed out
IF FAIL, Done                 ;Give up after 10 tries
:Logon
SEND 'Name, password^M'       ;Send name and password

...

:Done
SEND 'Bye^M'
```

4.11.2 Additional set commands

Main window options

SET MAIN.OUTINPUTDATATOSCR <value> - Output data sent on &screen
Value: True, False

SET MAIN.AUTOSCROLL <value> - Auto-scroll monitor window
Value: True, False

SET MAIN.COM1VISIBLE <value> - Visibility of the COM port #1 toolbar
Value: True, False

SET MAIN.COM2VISIBLE <value> - Visibility of the COM port #2 toolbar
Value: True, False

SET MAIN.COMBOVISIBLE <value> - Visibility of the COM port management toolbar
Value: True, False

`SET MAIN.STATUSVISIBLE` <value> - Visibility of the line status toolbar
Value: True, False

`SET MAIN.BUTTONSVISIBLE` <value> - Visibility of the quick access toolbar
Value: True, False

`SET MAIN.STATESVISIBLE` <value> - Visibility of the additional toolbar
Value: True, False

`SET MAIN.SHORTCUTSVISIBLE` <value> - Visibility of the shortcuts toolbar
Value: True, False

`SET MAIN.PARSEXX` <value> - Parse #XX Hex codes
Value: True, False

`SET MAIN.PARSEASCII` <value> - Parse ASCII codes with framing characters
Value: True, False

`SET MAIN.DIFFERENTSETT` <value> - Use different settings for each program instance
Value: True, False

`SET MAIN.PORT` <value> - Avalaible COM port number
Value:
- **0** - COM1
- **1** - COM2
- etc..

`SET MAIN.SPEED` <value> - COM port baud rate
Value:
- **0** - 110
- **1** - 300
- **2** - 600
- **3** - 1200
- **4** - 2400
- **5** - 4800
- **6** - 9600
- **7** - 14400
- **8** - 19200
- **9** - 38400
- **10** - 56000

- **11** - 57600
- **12** - 115200
- **13** - 230400
- **14** - 460800
- **15** - 921600
- **16** - 15600 (custom baud)
- **17** - 28800 (custom baud)

`SET MAIN.BITS` <value> - Number of data bits
Value:
- **0** - 5
- **1** - 6
- **2** - 7
- **3** - 8

`SET MAIN.PARITY` <value> - Parity
Value:
- **0** - None
- **1** - Odd
- **2** - Even
- **3** - Mark
- **4** - Space

`SET MAIN.STOPBITS` <value> - Stop bits (1.5 will be exposed automatically for 5 data bits and 2 stop bits)
Value:
- **0** - 1
- **1** - 2

`SET MAIN.DELAY` <value> - Data output delay (in miliseconds)
Value: Data output delay (in miliseconds)

`SET MAIN.AUTO` <value> - &Auto delay
Value: True, False

`SET MAIN.COMMAND` <value> - Input string. Enter data and press 'Enter' (F4)
Value: Input string. Enter data and press 'Enter' (F4)

`SET MAIN.OUTRECEIVEDTOFILE` <value> - Output data received to a file
Value: True, False

`SET MAIN.OUTINPUTDATATOFILE` <value> - Output data sent to a file

Value: True, False

`SET MAIN.OUTSYSDATATOFILE` <value> - Output system events (in the spy mode) to a file
Value: True, False

Data flow control

Hardware flow control

`SET USEDTR` <value> - Use DTR
Value: True, False

`SET USERTS` <value> - Use RTS
Value: True, False

`SET REQUIREDSR` <value> - Require DSR
Value: True, False

`SET REQUIRECTS` <value> - Require CTS
Value: True, False

Software flow control

`SET FLOWCONTROL` <value> - Software flow control
Value:
- **0** - None
- **1** - On receiving
- **2** - On transmitting
- **3** - Both

`SET XOFFCHAR` <value> - Character that is sent to disable remote sending (Xoff). You should type string, where each #XX block will be replaced with character with XX hex code.
Value: One or more characters

`SET XONCHAR` <value> - Character that is sent to enable remote sending (Xon). You should type string, where each #XX block will be replaced with character with XX hex code.
Value: One or more characters

`SET RS485MODE` <value> - RS485 interface mode
Value: True, False

`SET FLUSHBUFFER` <value> - Flush incoming buffer on line error
Value: True, False

For problem drivers only

`SET IGNOREERR` <value> - Ignore non-important port opening errors (for problematic ports only)
Value: True, False

Signs of end of string

`SET INEND` <value> - Termination string or character while receiving
Value:
- **0** - #0D - CR
- **1** - #00 - NUL
- **2** - #0A - LF
- **3** - Custom
- **4** - #0D#0A - CR+LF
- **5** - #0A#0D - LF+CR
- **6** - None

`SET OUTEND` <value> - Termination string or character while transmitting
Value:
- **0** - #0D - CR
- **1** - #00 - NUL
- **2** - #0A - LF
- **3** - Custom
- **4** - #0D#0A - CR+LF
- **5** - #0A#0D - LF+CR
- **6** - None

`SET CUSTOMSEND` <value> - Custom termination string

Value: One or more characters

`SET CUSTOMREC` <value> - Custom termination string
Value: One or more characters

`SET GB9BIT` <value> - 9-bit data transfer format
Value:

- **0** - Off
- **1** - Mark parity on send
- **2** - Space parity on send

`SET ADDBEFORE` <value> - Add before a data packet while transmitting
Value: True, False

`SET ADDBEFORE` <value> - Send data before each data block. You should type string, where each #XX block will be replaced with character with XX hex code.
Value: One or more characters

File with source data

`SET SOURCEFILE` <value> - The name of a file with data, which will be sent over serial port
Value: 'drive:directory\file name'. If your path and/or file name contains spaces, then you should add double quotes before and after value: '"drive:directory\file name"'

`SET FILEMODE` <value> - File output mode
Value:

- **0** - Unitary
- **1** - Repeated

`SET FILETYPE` <value> - File type
Value:

- **0** - ASCII or plain text
- **1** - Binary

ASCII mode

`SET FILEINTERPRET` <value> - Interpret characters in the file like #XX or (depend on the corresponding option) as a character with same Hex code
Value: True, False

Binary mode

Reading data block size

`SET BLOCKFIXEDSIZE` <value> - Fixed size (bytes):
Value: True, False

`SET BLOCKRANDOMSIZE` <value> - Random size (bytes min/max):
Value: True, False

`SET BLOCKFIXEDSIZE` <value> - Reading data blocks from the source file with this size
Value: 1..1000000

`SET BLOCKRANDOMSIZEMIN` <value> - Reading data blocks from the source file with this min size
Value: 1..1000000

`SET BLOCKRANDOMSIZEMAX` <value> - Reading data blocks from the source file with this max size
Value: 1..1000000

`SET BLOCKUNTIL` <value> - Reading from a file until occurrence of a termination character
Value: True, False

Protocol files

`SET CREATEFILEMODE` <value> - Create new file mode
Value:
- **0** - One file for all data
- **1** - Data sent and received in different files
- **2** - Each sent and received packet in a different file

One file for all data

`SET OUTPUTFILE` <value> - File name
Value: 'drive:directory\file name'. If your path and/or file name contains spaces, then you should add double quotes before and after value: '"drive:directory\file name"'

Data sent and received in different files

`SET FILEFORSENT` <value> - File name for data sent
Value: 'drive:directory\file name'. If your path and/or file name contains spaces, then you should add double quotes before and after value: '"drive:directory\file name"'

`SET FILEFORRECEIVED` <value> - File name for data received
Value: 'drive:directory\file name'. If your path and/or file name contains spaces, then you should add double quotes before and after value: '"drive:directory\file name"'

Each sent and received packet in a different file

`SET DIRECTORYEDIT` <value> - Folder for files with data received and sent
Value: '"drive:directory\"'. If your path contains spaces, then you should add double quotes before and after value: '"drive:directory"'

`SET FILENAMEPREFIXSENT` <value> - File name prefix for data sent
Value: One or more characters

`SET FILENAMEPREFIXRECEIVED` <value> - File name prefix for data received
Value: One or more characters

`SET OVERWRITEFILES` <value> - Overwrite existing files
Value: True, False

`SET ADDNAMESTAMP` <value> - Add date/time stamp to a file name
Value: True, False

SET FILENAMEEXT <value> - File name extension
Value: One or more characters

SET COUNTERFMT <value> - File name counter format
Value: One or more characters

SET MAXCOUNTER <value> - Maximum counter value (-1 - no limitations)
Value: Any decimal number

SET FILENAMESUFFIXSENT <value> - File name suffix for data sent
Value: One or more characters

SET FILENAMESUFFIXRECEIVED <value> - File name suffix for data received
Value: One or more characters

SET COMMONCNT <value> - Common counter for received and sent files
Value: True, False

SET SEPARATOR <value> - Separator between data packets. You can type any characters here, where each #XX block will be replaced with character with XX hex code.
Value: One or more characters

SET ADDBODYSTAMP <value> - Write date/time stamp to a file before writing data and events
Value: True, False

SET FILEFORMAT <value> - Format of the file output
Value:
- **0** - Without any changes
- **1** - As you see it on screen

Window view

SET STARTMINIMIZED <value> - Start up in minimized state
Value: True, False

SET HIDETOSYSTRAY <value> - Minimize to Systray (a panel near clock)

Value: True, False

`SET HINTBALOON` <value> - Hint balloon
Value: True, False

`SET HIDEMAINFORM` <value> - Hide on plugin start up
Value: True, False

`SET SHOWINHIDE` <value> - Display data sent and received in minimized state
Value: True, False

`SET HIDESTATUS` <value> - Hide status bar
Value: True, False

`SET HIDESCR` <value> - Hide window with data
Value: True, False

`SET STAYONTOP` <value> - Stay on top (over other windows)
Value: True, False

`SET AUTOCOMPLETE` <value> - Disable autocompletion in the input string
Value: True, False

`SET WRAPWORDS` <value> - Wrap words
Value: True, False

`SET LINESCOUNT` <value> - Number of strings in a screen buffer
Value: 10..30000

Font and colors

`SET HIGHLIGHT` <value> - Highlight data sent on screen
Value: True, False

`SET COLORHIGHLIGHT` <value> - Data highlighting color

Value: Data highlighting color

`SET FONTLIST` <value> - Courier
Value: Courier

`SET FONTCOLOR` <value> - Main window a font color
Value: Main window a font color

`SET BACKCOLOR` <value> - Main window a background color
Value: Main window a background color

`SET COMBOCOLORPROFILE` <value> - Default
Value:

- **0** - Dark
- **1** - Default

`SET FONTBOLD` <value> - Bold
Value: True, False

`SET FONTITALIC` <value> - Italic
Value: True, False

HEX view

`SET COLGRIDLINES` <value> - Grid lines color
Value: Grid lines color

`SET COLEVENCOL` <value> - Grid lines color
Value: Grid lines color

`SET COLOFFSET` <value> - Grid lines color
Value: Grid lines color

Data view

`SET CHARSET` <value> - Charset of data received
Value:
- **0** - DOS
- **1** - Windows

View mode of characters with code

`SET SYM20HEX` <value> - 0x00h - 0x1Fh
Value:
- **0** - Don't display
- **1** - As Hex #XX code
- **2** - User's format
- **3** - As ASCII character code

`SET SYM80HEX` <value> - 0x20h - 0x7Fh
Value:
- **0** - Don't display
- **1** - As Hex #XX code
- **2** - As same character
- **3** - User's format

`SET SYMFFHEX` <value> - 0x80h - 0xFFh
Value:
- **0** - Don't display
- **1** - As Hex #XX code
- **2** - As same character
- **3** - User's format

`SET SHOWSTRLEN` <value> - View received packet length
Value: True, False

`SET SHOWEND` <value> - View string that completes a data packet
Value: True, False

`SET USERFORMAT` <value> - User view mode of characters with code < 20 Hex %d - decimal value of character code %x - hexdecimal value %.2x - hexdecimal value with leading zero
Value: One or more characters

`SET FRAMEBEGIN` <value> - Beginning characters of a frame for an ASCII code
Value: One or more characters

`SET FRAMEEND` <value> - Ending characters of a frame for an ASCII code
Value: One or more characters

HEX window view

`SET BYTESROW` <value> - Bytes per row
Value: 1..128

`SET BYTESCOL` <value> - Bytes per column
Value: 1..8

`SET HEXTRANS` <value> - Translation
Value:

- **0** - Windows
- **1** - ASCII
- **2** - BCD
- **3** - DOS
- **4** - Mac

`SET SHOWOFFSET` <value> - Show offset
Value: True, False

`SET HEXLOWER` <value> - Hex lowercase
Value: True, False

`SET SWAPNIBBLES` <value> - Swap nibbles
Value: True, False

`SET GRIDLINES` <value> - Grid lines
Value: True, False

`SET BYTESROWAUTO` <value> - Automatically calculate number of bytes per row
Value: True, False

Date/time stamp

`SET STAMPVIEW` <value> - View mode
Value:
- **0** - Default
- **1** - Unix syslog
- **2** - Custom

`SET CUSTOMSTAMPFMT` <value> - Custom view mode of a datetime stamp
Value: One or more characters

`SET ADDSCRSTAMPSYS` <value> - Add to display output for system events
Value: True, False

`SET ADDSCRSTAMPSEND` <value> - Add to display output for data sent
Value: True, False

`SET ADDSCRSTAMPREC` <value> - Add to display output for data received
Value: True, False

`SET STAMPFRAMEBEGIN` <value> - Beginnig characters of a frame for a datetime stamp
Value: One or more characters

`SET STAMPFRAMEEND` <value> - Ending characters of a frame for a datetime stamp
Value: One or more characters

`SET ADDDATADIR` <value> - Add data direction sign to a stamp
Value: True, False

`SET TIMEOUTINT` <value> - Auto-insert a stamp every (ms) in a continuous data flow
Value: 1..65535

System events

`SET EVENTS` <value> - 0|1|2|3|4|5|6|7|8|9|10|11|12|13|14|15|16|17|18|19|20|21|22|

Value: 0|1|2|3|4|5|6|7|8|9|10|11|12|13|14|15|16|17|18|19|20|21|22|

4.11.3 Control characters

Control characters may transmitted by preceding a character with '^'. For example, a control C character is represented by ^C. You'll use this feature most often when sending carriage returns. For example, SEND 'myname^M' might be an appropriate response to a logon prompt where you would normally type your name and press <Enter>. NOTE: The control characters must be *inside* the quote marks, if quote marks are necessary.

Control character	Dec	Hex
^@	0	0x00
^A	1	0x01
^B	2	0x02
^C	3	0x03
^D	4	0x04
^E	5	0x05
^F	6	0x06
^G	7	0x07
^H	8	0x08
^I	9	0x09
^J	10	0x0A
^K	11	0x0B
^L	12	0x0C
^M	13	0x0D
^N	14	0x0E
^O	15	0x0F
^P	16	0x10
^Q	17	0x11
^R	18	0x12
^S	19	0x13
^T	20	0x14
^U	21	0x15
^V	22	0x16
^W	23	0x17
^X	24	0x18
^Y	25	0x19
^Z	26	0x1A
^[	27	0x1B
^\	28	0x1C
^]	29	0x1D
^	31	0x1F

4.12 Useful advice

- Look hints on all window elements - it will help you to get idea about the function made by this element;
- Pushing "Send" button is analogues to "Enter" in input string;
- Baud rate, data bit number, stop bits number, parity type and others can be changed in program work without COM port close;
- Many main window elements have "hot" keys for quick access to its functions;

  - Ctrl+S - analogues to key"Send";
  - Ctrl+O - analogues to key"Open/Close;
  - Ctrl+W - on/off write to file;
  - Ctrl+C - analogues to button "Clear";
  - F4 - set focus to data input string;
  - Ctrl+T - activate mode, where data source is input string;
  - Ctrl+F - activate mode, where data source is file;
  - Ctrl+Alt+E - call options of data packet end string sign;
  - Ctrl+Alt+F - call options of protocol file forming;
  - Ctrl+Alt+C - call additional options of COM port configuration;

- On available COM port number change is not necessary to push 'Stop' button. Close of current and open of new com port will be automatic;
- You can change manual line RTS and DTR signals state by clicking not corresponding indicator.

5 Having problems

5.1 If program doesn't work or run

Make sure in the correct time installation on your computer, if you changed time after program installation, protection from the use of the program after trial mode activates.
Program won't work, if the debug SoftIce or other is run.
In other cases, please, inform the developers about your problems on address support@aggsoft.com
.

5.2 FAQ

Question: Why, when I push button "Open" COM-port doesn't open?
Answer: Possibly, some program have already used it (COM-port). It can be DOS-application for example.

Question: What to do?
Answer: Close application, using this COM port (for DOS - application must be closed and DOS window - session), or use another COM port.

Question: Why data are lost on some files transmit?

Answer: To avoid that select the corresponding file type in program settings.

Question: Why when I spy data exchange in spy mode I see nothing?
Answer: Run Advanced Serial Port Monitor and close COM port before the given program start.

Endnotes 2... (after index)