Windows Standard

Serial Communications

for PowerBASIC

Programmer's Manual

(WSC_4PB)

Version 7.0

October 7, 2019

This software is provided as-is.
There are no warranties, expressed or implied.

Copyright (C) 2019
All rights reserved

MarshallSoft Computing, Inc.
Post Office Box 4543
Huntsville AL 35815

Email: info@marshallsoft.com
Web: www.marshallsoft.com

MARSHALLSOFT is a registered trademark of MarshallSoft Computing.

TABLE OF CONTENTS

1 Introduction

The **Windows Standard Serial Communications Library for PowerBASIC (WSC4PB)** is a toolkit that allows software developers to quickly develop serial communication applications in PowerBASIC.

The **Windows Standard Serial Communications Library** (WSC) is a component DLL library used to create serial communications programs that access data from a serial port using RS232 or multi-drop RS422 or RS485 ports. **WSC** also supports virtual serial ports using Bluetooth serial and USB to serial converters. The **WSC** component library uses the Windows API for all communication and can be used to easily write applications to control serial devices such as barcode scanners, modems, lab instruments, medical devices, USB serial devices, scales, GPS navigation, etc.

The **Windows Serial Communications Library for PowerBASIC (WSC4PB)** component library supports and has been tested with all versions of the PowerBASIC Console Compiler (PBCC), PowerBASIC DLL Compiler (PBDLL), and PowerBASIC Windows Compiler (PBWIN). WSC4PB includes numerous PowerBASIC example programs with source that demonstrate serial port communications functions.

The **WSC SDK** runs under 64-bit and 32-bit Windows (through Windows 10). A Win32 DLL is provided with WSC4XB. A Win64 DLL is available. The **Windows Standard Communications Library SDK** DLLs (WSC64.DLL and WSC32.DLL) can also be used from any development environment (Visual Basic, C++, Delphi, COBOL, Visual FoxPro, dBase, PowerBASIC, etc.) capable of calling the Windows API.

When comparing the **Windows Standard Serial Communications Library** against our competition, note that:

1. WSC4PB is a standard Windows DLL (NOT an OCX or ActiveX control) and is much smaller than a comparable OCX or ActiveX control.
2. A WIN32 DLL is included.
3. WSC does NOT depend on ActiveX or Microsoft Foundation Class (MFC) libraries or similar "support" libraries.
4. The WIN32 version of WSC is fully thread safe.
5. The WSC functions can be called from applications not capable of using controls.

MarshallSoft also has versions of the **Windows Standard Serial Communications Library** for Visual C/C++ (WSC4C), Delphi (WSC4D), dBASE (WSC4DB), Visual FoxPro (WSC4FP), Visual Basic (WSC4VB), and Xbase++ (WSC4XB). All versions of WSC use the same DLLs (WSC32.DLL and WSC64.DLL). However, the examples provided for each version are written in the specified computer programming environment.

The latest versions of the **Windows Standard Serial Communications Library (WSC)** can be downloaded from our web site at

http://www.marshallsoft.com/serial-communication-library.htm

Our goal is to provide a robust serial communication library component that you and your customers can depend upon. A fully functional evaluation version is available. Contact us if you have any questions.

1.1 Features

Some of the many features of the **Windows Serial Communications Library for PowerBASIC** are:

- Comes with a 32-bit DLL. (64-bit DLL is available)
- Can control any serial device (scale, barcode reader, etc.) connected to the serial port.
- Can be used from GUI mode or console mode programs.
- Can control up to 256 ports simultaneously.
- Can be used with virtual serial ports using Bluetooth serial or a USB to serial converter.
- Includes 52 functions plus modem control.
- Comes with ANSI emulation and ASCII, XMODEM and YMODEM.
- Supports RS232, and multidrop RS422, and RS485 ports.
- Supports hardware and software flow control.
- Supports any baud rate (32-bit version).
- Ability to specify the parity, word size, and number of stop bits.
- Supports binary and text data transfer.
- Port re-entrant.
- Is fully thread safe.
- Supports character peek (**SioEventChar**).
- Supports transmit and receive timeouts.
- Can send Windows messages on completion of events (incoming character, etc.)

- **Free** technical support for one year.
- License covers all programming languages.
- Royalty free distribution with a compiled application. There are no run time fees.
- Evaluation versions are fully functional. No unlock code is required.
- Can be used from GUI mode or console mode programs.
- Supports Windows XP through Windows 10.
- Implemented as a **standard** Windows DLL, which will work with all versions of Windows.
- Will run on machines with or without .NET installed
- Supports the PowerBASIC Console Compiler (PBCC), PowerBASIC DLL Compiler (PBDLL), and PowerBASIC Windows Compiler (PBWIN)
- Can also be used with all versions of Visual Basic through VB.Net.
- Does **not** depend on support libraries. Makes calls to core Windows API functions only.
- Can be used with any program (in any language) capable of calling Windows API functions such as Visual C++, Visual C++ .NET, Visual FoxPro, Delphi, Xbase++, dBASE, COBOL, Access and Excel.
- Can be purchased with or without ANSI C source code to the WSC DLL.
- Purchase a developer license for WSC4PB and use the DLLs with any other development environment (C++, Visual FoxPro, etc).
- Updates are **free** for one year (updates to source code are separate).
- Documentation online as well as in printable format.

Also see WSC versions for other supported languages:

- http://www.marshallsoft.com/wsc4c.htm (C/C++,C#,.NET)
- http://www.marshallsoft.com/wsc4d.htm (Delphi)
- http://www.marshallsoft.com/wsc4vb.htm (Visual Basic, VB.Net, VBA languages)
- http://www.marshallsoft.com/wsc4fp.htm (Visual FoxPro)
- http://www.marshallsoft.com/wsc4db.htm (dBase)
- http://www.marshallsoft.com/wsc4xb.htm (Xbase++)

Several PowerBASIC example programs with full source code are included. Refer to Section 4 for more details on each of the example programs.

```
DEVICE   : Sends text string to serial device.
DIALER   : Modem dialer.
ECHOPORT : Multi-threaded console mode program echoes all input.
FINDER   : Finds a modem connected to one of your serial ports.
LISTER     :  Lists all serial ports.
HELLO    : PowerBASIC Windows compiler (PBWIN) version of WSCVER.
PORTS    : PowerBASIC DLL Compiler (PBDLL) example program finds ports.
RS485    : RS485 example terminal program.
SELFTEST : Performs COM port functionality testing.
SIMPLE   : A simple terminal emulator.
SIMPLE2  : Same as SIMPLE, but uses WSC32PTR.PBI.
THREAD   : Same as SIMPLE, but implemented using a Win32 thread.
WSCVER   : Display WSC version number.
XMR      : Receives files via XMODEM protocol.
XMS      : Sends files via XMODEM protocol.
YMR      : Receives files via YMODEM protocol.
YMS      : Sends files via YMODEM protocol.
EventW   : Terminal program that uses SioEventWait to efficiently wait.
W_Hello  : PBWin version of WSCVER.BAS (WSC Version)
W_Device : PBWin version of DEVICE.BAS
W_XMS    : PBwin version of XMS.BAS
W_XMR    : PBWin version of XMR.BAS
SCALE    : Reads data from digital scale.
ReadGPS  : Reads NMEA 183 GPS data
```

WSC4PB contains 52 functions and modem control. All functions return a negative number if an error condition is detected. For more details, consult the WSC Reference Manual (WSC_REF.PDF) and the RS232/422/485 Serial Communications User's Manual (SERIAL.PDF). Both manuals can be accessed online at

http://www.marshallsoft.com/wsc4pb.htm

1.2 Documentation Set

The complete set of documentation consists of four manuals in Adobe PDF format. This is the first manual (WSC_4PB) in the set.

- WSC_4PB Programmer's Manual (WSC_4PB.PDF)
- WSC User's Manual (WSC_USR.PDF)
- WSC Reference Manual (WSC_REF.PDF)
- SERIAL User's Manual (SERIAL.PDF)

The WSC_4PB Programmer's Manual is the programming language specific (PowerBASIC) manual and provides information needed to compile your programs in a PowerBASIC environment.

The WSC User's Manual (WSC_USR) discusses language independent serial communications programming issues including modem control. Purchasing and license information is also provided.

The WSC Reference Manual (WSC_REF) contains details on each individual WSC function.

The Serial Communications User's Manual (SERIAL) contains background information on serial port hardware.

The documentation is also provided on our web site at

http://www.marshallsoft.com/wsc4pb.htm

1.3 Example Program

The following example program segment displays the WSC registration string.

```
 $INCLUDE "WSC32.PBI"
 $INCLUDE "KEYCODE.PBI"

FUNCTION PbMain() AS LONG
Dim Code As Integer
Dim Temp As Asciiz * 51
Code = SioKeyCode(%WSC_KEY_CODE)
IF Code < 0 THEN
  PRINT "SioKeyCode fails: Check value of %WSC_KEY_CODE"
  EXIT FUNCTION
END IF
' display registration string
Code = SioGetReg(Temp, 50)
IF Code > 0 THEN
  PRINT Left$(Temp, Code)
END IF
END FUNCTION
```

Refer to section 4.0 for complete examples with source.

1.4 Installation

(1) Before installation of WSC4PB, a PowerBASIC Console Compiler (or PowerBASIC DLL Compiler) should already be installed on your system and tested.

(2) Unzip WSC4PB70.ZIP (evaluation version) or WSCxxxxx.ZIP (registered version where xxxxx is your Customer ID) using any Windows unzip program.

(3) Run the installation program, SETUP.EXE, which will install all WSC4PB files, including copying WSC32.DLL to your Windows directory.

Note that no DLL registration is required.

1.5 Uninstalling

Uninstalling WSC4PB is very easy. First, delete the WSC4PB project directory created when installing WSC4PB. Second, delete WSC32.DLL from your Windows directory, typically C:\WINDOWS.

Running the UNINSTAL.BAT batch file will also delete WSC32.DLL, MIO32.DLL, XYM32.DLL, and ASD32.DLL as described above.

1.6 Pricing

A developer license for the Windows Standard Serial Communications Library can be purchased for $115 (or $195 with ANSI C source code to the library DLL's). Purchasing details can be found in Section 1.3 "How to Purchase" in the WSC User's Manual (WSC_USR.PDF). See

http://www.marshallsoft.com/wsc_usr.pdf

Also see INVOICE.TXT provided with the evaluation version or order directly on our web site at

http://www.marshallsoft.com/order.htm

1.7 Updates

When a developer license is purchased, the developer will receive a new set of registered DLLs plus a license file (WSCxxxxx.LIC). The license file is needed to download updates to the registered DLL'. Updates can be downloaded for a period of one year from purchase from:

http://www.marshallsoft.com/update.htm

After one year, your license must be updated if you want to be able to download updates. Your license can be updated for :

- $33 if the update is ordered within one year of the original purchase (or previous update).
- $55 if the update is ordered between one and three years of the original purchase.
- $77 if the update is ordered after three years of the original purchase.

If source code was previously purchased, updates to the source code can be purchased for $40 along with the DLL update.

Note that the registered DLL's never expire.

2 Library Overview

The **Windows Standard Serial Communications Library (WSC)** has been tested on multiple computers running Windows 2003-2012/XP/Vista/Windows 7/Windows 8/Windows 10.

2.1 Dynamic Link Libraries

The WSC4PB serial communication library component (WSC32.DLL) is implemented as a Win32 dynamic link library (DLL). A DLL is characterized by the fact that it need not be loaded until required by an application program and that only one copy of the DLL is necessary regardless of the number of application programs that use it. Contrast this to the traditional static library that is bound to each and every application that uses it at link time.

An important advantage that DLLs have over other "popular" library formats such as VBX or OCX is that DLLs are callable by all Windows applications. Since DLLs are the building blocks of the Windows Operating System, they will not be replaced by a "newer technology".

2.2 Keycode (License Key)

The WSC32.DLL has a keycode encoded within it. Your keycode is a 9 or 10 digit decimal number and will be found in the file KEYCODE.PBI. The keycode for the evaluation version is 0. You will receive a new keycode and a set of new DLL's when registering. The KEYCODE is passed to **SioKeyCode**.

If you get an error message (value -108) when calling **SioKeyCode**, it means that the keycode in your application does not match the keycode in the DLL. After registering, it is best to remove the evaluation version of the WSC32.DLL from the Windows search path or delete it.

2.3 Win32 STDCALL and DECLSPEC

WSC32 is written in ANSI C and is compiled using the STDCALL and DECLSPEC keywords. This means that WSC4PB uses the same calling conventions and file naming conventions as the Win32 API. . In particular, function names are NOT decorated. There are neither leading underscores nor trailing "@size" strings added to function names.

Any Windows application program capable of calling the Windows API provided the proper declaration file is used may call the WSC32.DLL functions.

2.4 Using Threads

WSC4PB is thread safe, and can be used from any Windows application capable of using threads. Refer to the THREAD and ECHOPORT example programs, which demonstrate the use of threads. ECHOPORT also demonstrates the use of the WIN32 Sleep() function.

The necessary Win32 thread functions are declared in the PowerBASIC INCLUDE file "WIN32API.INC"

2.5 Limitations on COM Ports

WSC4PB (WSC32.DLL) works with any real or virtual port from COM1 to COM256, provided that the port is known to Windows.

2.6 Error Display

The error message text associated with WSC error codes can be displayed by calling **SayError** (refer to Errors32.bas). Each sample program contains examples of error processing.

2.7 Waiting for New Serial Data

All serial data is moved from the UART's buffer to the receive queue in memory (by the Windows serial port driver) under interrupt control. Similarly, all out going serial data is moved to the transmit queue in memory.

There are several methods that can be used to receive incoming serial data. **SioMessage** can be used in PBWIN programs since they have a message loop. **SioGetc** and **SioGets** can be called directly. Note that if there is no input data available, **SioGetc** returns WSC_NO_DATA while **SioGets** returns zero. Also see Section 2.8.

2.8 SioEvent Logic

SioEvent, **SioEventChar**, and **SioEventWait** will block until the specified event occurs. If a call to **SioEvent**, **SioEventChar**, or **SioEventWait** is placed in a thread, then the thread will block but the application calling the thread will not.

2.9 Virtual Serial Ports

A “virtual” serial port is COM port that appears to be a real RS232 serial port to the Windows API (and thus to WSC), but is in reality a COM port emulator.

The two most common virtual ports are those created for USB/serial port converters and Blue Tooth. WSC will work with USB to serial port converters and with Bluetooth serial.

More information about Virtual serial ports can be found in the WSC User’s Manual, Section 2.12, “Virtual Serial Ports” (http://www.marshallsoft.com/wsc_usr.pdf).

2.10 Adding WSC to A PowerBASIC Program

In order to add WSC to a PowerBASIC program, add

```
$INCLUDE "WSC32.PBI"
$INCLUDE "KEYCODE.PBI"
```

after any other $INCLUDE statements. Then add

```
Code = SioKeyCode(%WSC_KEY_CODE)
If Code < 0 Then
  '... display error code to user here...
  EXIT FUNCTION
END IF
```

as the fist executed WSC function.

2.11 PowerBASIC Declaration Files

WSC constants are defined and all WSC functions are declared in the file WSC32.PBI. WSC32.PBI is included in all of the example programs except for SIMPLE2.BAS.

Several of the WSC functions have alternate declarations in the file WSC32PTR.PBI. The alternate declarations allow greater flexibility in how strings are handled in your PowerBASIC program. For an illustration, see the example program SIMPLE2.BAS.

3 Compiler Issues

The WSC4PB library supports and has been tested with the PowerBASIC Console Compiler (PBCC), PowerBASIC DLL Compiler (PBDLL), and PowerBASIC Windows Compiler (PBWIN).

Please examine the WSC32.PBI file. Note that %COM1 has numerical value zero, not one. The user must assume the responsibility for passing the correct information when calling WSC4PB functions

3.1 PowerBASIC Console Compiler (PBCC)

Most of the example programs are written for the PowerBASIC Console Compiler (PBCC). Be sure to read the comments at the beginning of each example program. Also see section 4.0 “Example Programs” below.

3.2 PowerBASIC DLL Compiler (PBDLL)

WSC functions can be called from programs written for the PowerBASIC DLL Compiler (PBDLL). The same DLL (WSC32.DLL) is used for both PowerBASIC and Visual Basic. WSC functions are defined in WSC32.PBI for PowerBASIC and in WSC32.BAS for Visual Basic. The Visual Basic version can be downloaded from www.marshallsoft.com/wsc4vb.htm. See the PORTS.BAS example program.

3.3 PowerBASIC Windows Compiler (PBWIN)

WSC functions can be called from programs written for the PowerBASIC Windows Compiler (PBWIN). There are no differences in how WSC functions are called in programs written in any of the Windows PowerBASIC compilers.

3.4 Compiling WSC Source

WSC32.DLL is written in standard ANSI C (WSC32.C), and has been compiled using Microsoft Visual C/C++ with the STDCALL and DECLSPEC compiler keywords. Source code for the WSC library can be purchased at the same time as a WSC developer license is purchased.

WSC32.C may also be compiled using Borland C/C++ or Watcom C/C++ compilers. If you recompile WSC32.C using Borland or Watcom compilers, the resulting WSC32.DLL can only be used by applications compiled with the same compiler, unless the STDCALL and DECLSPEC keywords are specified.

For more information on the C/C++ version of WSC, download the latest version of WSC4C from our web site at http://www.marshallsoft.com/wsc4c.htm

3.5 Compiling Example Programs

Compiling programs using the PowerBASIC Console Compiler is very easy. For example, to compile SIMPLE.BAS, type

```
PBCC SIMPLE.BAS
```

at the command prompt. If your program requires access to any Windows API functions, you must include the path on the command line. For example, assuming that you have installed PBCC at C:\PBCC, to compile the ECHOPORT.BAS program, type (the "-MT" is for Multi-Tasking support)

```
PBCC ECHOPORT.BAS -IC:\PBCC\WINAPI -MT
```

WSC4PB functions can also be called from PBDLL programs. See the PORTS.BAS example.

4 Example Programs

The example programs are designed to demonstrate the various capabilities of WSC4PB. The best way to become familiar with WSC4PB is to study and run the example programs.

4.1 WSCVER

WSCVER ("WSC Version") is a PBCC example program displays the WSC version number. This is the first program to compile and build since it versifies that WSC32.DLL is installed properly. Also see the HELLO.BAS (PBWIN) example program.

4.2 SIMPLE

SIMPLE is a very simple serial communications program using WSC4PB. Everything that is typed on the keyboard is sent to the serial port, and everything incoming from the serial port is displayed on the screen.

The easiest way to test SIMPLE is to connect to a modem. Typing 'AT' should result in an 'OK' being displayed.

A null-modem cable can also be used to connect two computers together with their serial ports. Run SIMPLE on both machines. Whatever is typed on one machine will be displayed on the other.

4.3 SIMPLE2

SIMPLE2 is the same program as SIMPLE, except that it uses WSC32PTR.PBI rather than WSC32.BPI, in which alternate definitions of SioPuts and SioGets are used.

4.4 SELFTEST

SELFTEST performs a serial port I/O functionality test. Either a pair of ports on the same computer (using a null modem cable) or a single port (using a loopback adapter) can be tested.

Refer to LOOPBACK.TXT for an explanation of how to make a loopback adapter (without tools!).

4.5 FINDER

The FINDER program searches for a connected modem. Your modem must be connected to one of COM1, COM2, COM3, ...,COM16, and must be turned on.

FINDER takes no arguments. After connecting your modem to one of your serial ports, type

```
FINDER
```

4.6 LISTER

The LISTER program lists all serial ports.

4.7 THREAD

The THREAD example program is similar to SIMPLE, except that it uses a Win32 thread. Note that the PowerBASIC thread function blocks awaiting serial input.

4.8 DEVICE

The DEVICE example program is designed to send a text string to a serial device. A carriage return is appended to the end of the string.

The DEVICE program can be used to send commands to serial devices which use ASCII commands, such as bar code readers, XY-plotters, etc.

Also see W_DEVICE below.

4.9 XMS and XMR

XMS (XMODEM Send) and XMR (XMODEM Receive) are programs that send and receive files using the XMODEM protocol.

Also see W_XMS and W_XMR below.

4.10 YMS and YMR

YMS (YMODEM Send) and YMR (YMODEM Receive) are programs that send and receive files using the YMODEM protocol.

4.11 ECHOPORT

The ECHOPORT program echoes characters incoming on COM1 and COM2 using threads. To use,

1. Connect COM1 or COM2 to another computer using a NULL modem cable.
2. Start SIMPLE on the other computer(s) at 38400 baud.
3. Start ECHOPORT. Anything transmitted from the other computer is echoed back by ECHOPORT.

4.12 RS485

The RS485 program is a simple terminal emulator using a RS485 port in which RTS is raised before transmitting and dropped afterwards.

4.13 PORTS

The PORTS example program demonstrates how to call DLL's created by the PowerBASIC DLL compiler from (32-bit) Microsoft Visual Basic. PORTS.BAS is compiled by

```
PBDLL PORTS.BAS
```

You must also compile the Visual Basic program PORTS.FRM by loading PORTS.VBP in Visual Basic version 4.0 or above.

4.14 EventW

EventW is a PBCC terminal program. It works exactly like SIMPLE and SIMPLE2 except that **SioEventWait** is used to efficiently wait for new incoming data.

4.15 W_HELLO

W_HELLO is a PowerBASIC for Windows (PB/WIN) example program similar to WSCVER. It displays the WSC version number.

4.16 W_DEVICE

W_DEVICE is a PowerBASIC for Windows (PB/WIN) example program similar to DEVICE.

The W_DEVICE example program is designed to send a text string to a serial device. A carriage return is appended to the end of the string.

The W_DEVICE program can be used to send commands to serial devices which use ASCII commands, such as bar code readers, XY-plotters, etc.

4.17 W_XMS and W_XMR

W_XMS (XMODEM Send) and W_XMR (XMODEM Receive) are PowerBASIC for Windows (PB/WIN) programs that send and receive files using the XMODEM protocol.

4.18 SCALE

The Scale example program ransmits an ASCII command string to a scale, then reads incoming (response) data until a 100 ms data pause (Quiet) is detected.

4.19 ReadGPS

The ReadGPS example program read lines from a device that is outputting NMEA 183 GPS sentences, although this program will read complete lines from any serial device that outputs such lines.

5 Revision History

NOTE: Version 2.3 was the first PowerBASIC version.

Version 2.3: August 3, 1998.

Initial release of PowerBASIC version.

Version 2.4: May 24, 1999

- SioEvent function efficiently blocks waiting for event.
- XYDRV can send multiple files in YMODEM. XYDRV is thread safe.
- Specify default DTR and RTS behavior with SioReset.
- New example programs (RS485,XMS,XMR,YMS, and YMS).

Version 3.0: September 8, 2000

- Increased default to 32 ports.
- Added several new example programs (WSCVER, THREAD, and DEVICE.).
- Added WORD and HTML documentation.

Version 3.1: May 15, 2001.

- RESETDEV Win API call not called (allows USB/serial converters).
- SioPutc and SioPuts return immediately (optional).
- XYM (XMODEM/YMODEM) allows local upload/download directory to be specified.
- Updated XMODEM/YMODEM supports local upload/download directory.

Version 3.2: August 8, 2002.

- Added PBDLL example program PORTS.BAS.
- Default for RESETDEV is "not called". SioDebug(ASC("R")) to enable.
- SioGetc & SioGets zero unused bits (DataBits 5,6,7).
- Corrected problem with SioBaud(-1, BaudRateCode).
- SioDebug returns -1 if no match.
- Added SioDebug(ASC("W")) toggle SioPuts wait for I/O completion.
- Added code to detect active threads & to close thread handles.
- Added USE_THREADS, so can compile version of WSC32.C without threads.
- Comm handle not saved in SioReset unless it is good.
- SioEvent returns mask that caused the event.
- Added SioInfo(ASC("B")) to get build number.

Version 4.0: December 2, 2003.

- Can now order either with or without source code to the DLLs.
- Added SioSetInteger function to set port specific integer parameters.
- Added SioKeyCode function to pass the key code to the DLL.
- Added SioGetReg function to return the registration string.
- Added "Burst Size" parameter for setting the TX burst size.
- Added ability to signal blocked thread that was blocked by SioEvent.

Version 4.1: August 12, 2004

- Fixed problem with SioTxClear.
- Added overlapped I/O (for non-Win95) so can signal threads to exit w/o killing them.
- Increased default burst size to 256.
- SioFlow returns WSC_RANGE if cannot recognize parameter.
- Added PBWIN example.
- Adjusted XModem/YModem timing for faster transfers.

Version 4.2: March 28, 2006.

- SioFlow returns 1 if OK.
- SioSetInteger(Port, ASC("S"), 1) always forces SioEvent to unblock.
- Event mutex code added to EventThread() to prevent race conditions.
- Message box displays error if SioWinError(Buffer, 0) called.
- Major change in overlapped I/O
- Fixed problem: SioEvent returning wrong code.
- SioRxClear clears byte saved by SioUnGet.
- Number of supported ports increased to a maximum of 256.
- Added SioEventChar() and SioEventWait() functions.
- Added EventW example program.

Version 4.3: October 2, 2007.

- Fixed problem with SioTxQue returning wrong values.
- Changed SioParms so it checks the range of passed arguments.
- Port is verified in SioEventChar.
- SioStatus returns -1 if port is not functioning (USB/serial port disconnected).
- Added SioByteToShort and SioShortToByte (WSC32 only).

Version 4.4: February 5, 2009

- Added SioTimeouts() function (sets TX and RX time-outs).
- Provide documentation files in Adobe PDF format.
- Added W_XMS and W_XMR example programs.

Version 5.0: January 18, 2010

- Added SioHexView() function.
- Added SioSleep() function
- Supports 64-bits (WSC64.DLL).

Version 5.1: September 20, 2011

- Added SioRxWait function
- Added LISTER example program.

Version 5.2: July 8, 2012

- Added function SioQuiet()
- Added function SioWaitFor()
- Added example program ProXR.bas

Version 5.3: November 26, 2013

- Added SioLRC() that computes the "longitudinal redundancy check" per ISO 1155.
- SioQuiet() and SioWaitFor() verify the passed port number.
- SioWaitFor() verifies that the passed baud rate is > 0.
- SioSetInteger() no longer requires an open port for global (all ports) parameters.
- Modified SioReset() to make it more tolerant opening slow virtual ports.

Version 5.4: September 23, 2015

- Added SioCRC16() that computes 16-bit CCITT CRC (polynomial 1021 hex).
- Added SioCRC32() that computes 32-bit CCITT CRC (polynomial 04C11DB7 hex).

Version 6.0: March 30, 2017

- Added additional error codes: WSC_BUFFER_RANGE,WSC_BUFLEN_RANGE,WSC_BAD_CMD
- Added additional error codes: WSC_BAD_PARITY.WSC_BAD_STOPBIT,WSC_BAD_WORDLEN
- Added SioErrorText() that returns text associated with specified error codes.
- Added SioPortInfo() that returns baud in BPS (bits per sec) and the theoretical port CPS (char per sec).
- Added SioGetsC() that receives an entire line through the stop (end-of-line) character (usually CR).
- Added ReadGPS example program.

Version 7.0: October 7, 2019

- Fixed: SioErrorText() now returns text length from call to SioWinError()
- Fixed: SioGets() would never timeouts when overlapped I/O was enabled.
- Added: function SioOpen - same as SioReset(Port, 1280, 1280)
- Added: function SioClose - same as SioDone.
- Added: function SioGetsQ - reads port until no incoming data for specified "quiet" time.
- Added Scale.bas program.