Windows Standard

Serial Communications

Library for Visual dBase

Programmer's Manual

(WSC4DB)

Version 6.0

March 24, 2017

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

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

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

TABLE OF CONTENTS

1 Introduction

The **Windows Standard Serial Communications Library for Visual dBASE (WSC4DB)** is a toolkit that allows software developers to quickly develop serial communication applications in Visual dBASE or dBASE Plus.

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 Visual dBASE (WSC4DB)** component library supports all Win32 (dBASE 7.0, 7.5 and dBASE Plus) versions of dBASE.. **WSC4DB** runs under Windows XP through Windows 10, and includes numerous example programs that demonstrate serial port communications functions.

The **Windows Standard Communications Library** DLLs can also be used from any development environment (C/C++, Delphi, Visual Basic, PowerBASIC, Visual FoxPro, Visual dBase, XBase++, COBOL etc.) capable of calling the Windows API.

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

1. WSC4 is a standard Windows DLL (NOT an OCX or ActiveX control) and is much smaller than a comparable OCX or ActiveX control.
2. WSC4 is implemented as a Win32 DLL.
3. WSC does NOT depend on ActiveX or Microsoft Foundation Class (MFC) libraries or similar "support" libraries.
4. The Win32/Win64 versions of WSC are 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), PowerBASIC (WSC4PB), Visual FoxPro (WSC4FP), Visual Basic (WSC4VB), and Xbase++ (WSC4XB). All versions of WSC use the same DLLs (WSC32.DLL or 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 Visual dBASE (WSC4DB)** are:

- Comes with 32-bit DLL. A 64-bit DLL is available (although dBase is a 32-bit app).
- Can control any serial device (scale, barcode reader, etc.) connected to the serial port.
- Can control up to 256 ports simultaneously.
- Can be used with virtual serial ports using Bluetooth serial or a USB to serial converter.
- Includes over 46 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.
- 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 all versions of 32-bit Visual dBase and dBase Plus.
- 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++, Visual Basic, COBOL, Access and Excel.
- Can be purchased with or without ANSI C source code to the WSC DLLs.
- Purchase a developer license for WSC4DB 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.

A good selection of dBase programs with full source code is included. Refer to Section 4 for more details on each of the example programs.

```
[PROGRAM]     [DESCRIPTION]
WSCVER      :  Program that displays the WSC version number.
SIMPLE      :  A simple terminal emulator.
SIMPLE2     :  A simple terminal emulator (alternate version)
FINDER      :  Finds a modem connected to one of your serial ports.
LISTER      :  Lists all serial ports.
XMS         :  XMODEM send program.
XMR         :  XMODEM receive program.
YMS         :  YMODEM send program.
YMR         :  YMODEM receive program.
DEVICE      :  Program sends string to serial device.
DIALER      :  Dials up host (or BBS).
ATOK        :  Forms based program that sends "AT" to modem.
SELFTEST    :  Performs port functionality testing.
PROXR       :  Reads relays on ProXR device.
ReadGPS     :  Read NMEA 183 GPS data.
```

WSC4DB contains 49 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) and the RS232/422/485 Serial Communications User's Manual (SERIAL).

1.2 Documentation Set

The complete set of documentation consists of four manuals:

- WSC_4DB Programmer's Manual (WSC_4DB.PDF)
- WSC User's Manual (WSC_USR.PDF)
- WSC Reference Manual (WSC_REF.PDF)
- SERIAL User's Manual (SERIAL.PDF)

The WSC_4DB Programmer's Manual is the programming language specific (dBASE) manual and provides information needed to compile your programs in a dBase 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 Manual (SERIAL) contains background information on serial port hardware.

Documentation can be found in the \WSC4DB\APPS folder and also online at
http://www.marshallsoft.com/wsc4db.htm

1.3 Example Program

The following example program segment to set DTR (Data Terminal Ready) demonstrates the use of some of the library functions:

```
*
* SETDTR.PRG
*
#INCLUDE WSC32.CC
#INCLUDE KEYCODE.CC

? "WSCVER 5.3"
?
* pass the key code
if SioKeyCode(WSC_KEY_CODE) < 0
  ?"ERROR: Bad Key Code!"
  return
endif
* open port COM1
Code = SioReset(COM1, 512, 512)
If Code < 0
  ?"ERROR: Cannot open port"
  return
endif
* set DTR
Code = SioDTR(COM1, ASC("S"))
* close port
Code = SioDone(COM1)
```

1.4 Installation

(1) Before installation of WSC4DB, your dBase compiler should already be installed on your system and tested.

(2) Unzip WSC4DB60.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 WSC4DB files including copying WSC32.DLL to the Windows directory.

Note that no DLL registration is required..

1.5 Uninstalling

Uninstalling WSC4DB is very easy.

First, run UINSTALL.BAT, which will delete WSC32.DLL from the Windows directory, typically C:\WINDOWS for Windows XP/2003-2012/Vista/Win7/Win8/Win10.

Next, delete the WSC project directory created when installing WSC4DB.

1.6 Pricing

A developer license for WSC4DB can be registered for $115 (or $195 with ANSI C source code to the library DLL's). Purchasing details can be found in the WSC User's Manual (WSC_USR), Section 1.3, "How to Purchase" (http://www.marshallsoft.com/wsc_usr.pdf). Also see INVOICE.TXT provided with the evaluation person or order directly on our web site at

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

1.7 Updates

When a developer license is purchased for WSC4DB, the developer will received a new a set of registered DLLs plus a license file (WSCxxxx.LIC). The license file is needed to download updates to the registered DLL's for a period of one year from purchase. Updates can be downloaded from

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

After one year, the developer license must be updated to be able to download updates. The developer license can be updated for:

- $30 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.
- $75 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/Win7/Win8/Win10.

The SETUP installation program will copy the DLLs to the Windows directory. Refer to Section 1.4 "Installation". After SETUP is run, the WSC4DB files are copied to the directory specified (default \WSC4DB). Three sub-directories are created, as follows:

```
DOCS – All documentation files
APPS – All example code
DLLS – All DLL's
```

2.1 Dynamic Link Libraries

The **WSC4DB serial communication library** component uses a 32-bit 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

WSC32.DLL has a keycode encoded within it. Your keycode is a 9 or 10 digit decimal number (unless it is 0), and will be found in the file KEYCODE.CC. The keycode for the evaluation version is 0. The developer will receive a new key code and set of DLL's after purchasing a developer license. The KEYCODE is passed to **SioKeyCode**.

If an error message (value -108) is received 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 them

2.3 Limitations on COM Ports

The 32-bit version of WSC4DB (WSC32.DLL) can use any port from COM1 to COM256, provided that the port is known to Windows 2003-2012/XP/Vista/Win7/Win8/Win10 and there is physical hardware present.

2.4 INCLUDE Files

INCLUDE files used by the example programs are:

```
WSC32.CC  - Serial communications constants & declarations.
MIO32.CC  - Modem I/O constants & declarations.
XYM32.CC  - X/Ymodem constants & declarations.
```

There are three recommended ways to handle these INCLUDE files in dBase programs.

1. Copy the INCLUDE files to your compiler's INCLUDE directory.
2. Edit the INCLUDE statements (in each program) with their physical location.
3. Replace the INCLUDE statements (in each program) by their contents.

2.5 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 most USB to serial port converts and with Bluetooth serial, although WSC doesn’t access USB directly.

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).

2.6 Dynamic Strings

The dBase programming language uses a technique known as "garbage collection" to manage string space at runtime, and may be called internally at any time by the dBase runtime, asynchronous to what you may be doing in your code.

When passing a string buffer to a DLL function into which text will be copied, it is strongly recommended that the local string be allocated immediately before use. For example:

```
Code = SioReset(Port, 1024, 1024)
if Code < 0
  ? "ERROR " + Str(Code)
  * allocate 'Buffer' immediately before use
  Buffer = SPACE(128)
  Code = SioWinError(Buffer, 128)
  if Code > 0
    ? Left(Buffer,Code)
  endif
endif
```

This technique is not necessary for passing a string to a DLL function, only when passing a buffer to a DLL into which data is to be placed by the DLL function.

2.7 Null Terminated Strings

All strings returned from WSC functions are null terminated. That is, the end of the string is delimited by a Chr(0) character. These strings may be converted for dBase by using the dBase `LEFT` function: For example,

```
Code = SioWinError(Buffer, 128)
if Code > 0
  Buffer = Left(Buffer,Code)
endif
```

2.8 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. 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 2.10, "SioEvent Logic".

2.9 Error Display

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

2.10 SioEvent Logic

SioEvent, **SioEventChar**, and **SioEventWait** will block (the application will sleep) until the specified event or timeout (**SioEventWait** only) 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. Descriptions of these functions can be found in the WSC Reference Manual.

Also see the SIMPLE2.PRG example program that uses **SioEventWait.**

3.0 Compiler Issues

3.1 Visual dBase 5.6 and 5.7

Support for Win16 was dropped beginning with version 5.2. Version 5.1 is still available (free when purchasing the current version) for those wanting Win16 support.

3.2 Visual dBase 7.0 and above

Visual dBase 7.0 and above create 32-bit applications and use the 32-bit WSC DLL's (WSC32.DLL, MIO32.DLL, and XYM32.DLL). Copy the 32-bit CC files (WSC32.CC, MIO32.CC and XYM32.CC) to your compiler's INCLUDE directory.

3.3 Compiling dBase Programs

Visual dBase programs end with the extension ".PRG". Before compiling any of the example programs, edit each file with the port number and baud rate. Programs can be edited within any text editor, and compiled from the VDB (Visual dBase) command window with the COMPILE command (e.g.: COMPILE SIMPLE.PRG) or executed from the VDB command window with the DO command (e.g.: DO SIIMPLE.PRG).

To open a program within the Visual dBase source editor, choose "File", then "Open". When the "Open File" dialog box appears, choose "Programs" for "Files of Type", then choose the program (*.PRG) to open. Lastly, choose "Open in Source Editor" for "Action" and push the "Open" button.

After editing the source program, you are ready to compile. From the dBase menu bar, choose "Build", then "Compile". To run choose, "Run". The VDB command window must be displayed in order to see the output.

3.4 Compiling dBase Projects

dBase projects consist of several types of files such as forms, reports, data modules, etc. The project file itself ends with the extension of ".PRJ".

There is one example dBase project: ATOK32. Open ATOK32 by choosing "File", then "Open Project" from the dBase menu bar. To compile ATOK32, choose "Build" from the menu bar, then "Rebuild All". This will create ATOK32.EXE, which can be executed by choosing "Execute atok32.exe" from the "Build" menu bar pulldown, or from the Windows command line prompt.

3.5 Compiling to an Executable

dBase programs end in ".PRG". They can be compiled to an executable using the dBase BUILD command.

For example, to create WSCVER.EXE from WSCVER.PRG in the C:\WSC4DB\APPS directory, type the following in the dBase command window:

```
BUILD PROJECT C:\WSC4DB\APPS\WSCVER FROM C:\WSC4DB\APPS\WSCVER
BUILD EXE C:\WSC4DB\APPS\WSCVER FROM C:\WSC4DB\APPS\WSCVER
```

4 Example Programs

All example programs end with ".PRG", except for the FORM example (ATOK), which ends with ".WFM".

4.1 WSCVER

The WSCVER ("WSC Version") example program displays the WSC version number. This is the first program to compile and build since it verifies that WSC32.DLL is installed properly.

4.2 SIMPLE

SIMPLE is a very simple communications program using WSC4DB. 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 calls **SioPuts** rather than **SioPutc**, and **SioGets** rather than **SioGetc**. **SioPuts** and **SioGets** send and receive strings instead of characters. SIMPLE2 also calls **SioEventWait** when blocking (efficiently waiting) for new serial input.

4.4 XMS and XMR

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

4.5 YMS and YMR

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

4.6 FINDER

The FINDER program is a program that searches for a connected modem. Your modem must be connected to one of COM1 through COM256, and the modem must be turned on. Note that FINDER uses the MIO module.

4.7 LISTER

The LISTER program lists all serial ports.

4.8 DIALER

The DIALER program dials up a HOST (or BBS) program, connects, and then operates like SIMPLE; in which all serial input is copied to the screen and all keyboard input is sent out the serial port.

4.9 DEVICE

The DEVICE program is designed for talking to an arbitrary serial device. Use this program as a guide when communicating with all serial devices other than modems and other computers.

4.10 ATOK

The ATOK programs demonstrate how to call WSC functions from within a dBase FORM. They transmit the string "AT" and expect an "OK" back.

4.11 SELFTEST

SELFTEST performs basic serial port I/O functionality testing using a loopback adapter. Refer to LOOPBACK.TXT for an explanation of how to make a loopback adapter (without tools!).

4.12 ProXR

The ProXR example program demonstrates how to control the ProXR relays boards manufactured by ControlAnything.com

4.13 ReadGPS

The ReadGPS console mode 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

The WSC DLL (WSC32.DLL) is written in ANSI C. All language versions of WSC (C/C++, Delphi, Visual Basic, PowerBASIC, Visual FoxPro, Visual dBase, Xbase++, and COBOL) use the same identical DLLs.

Version 3.0: August 21, 2000.

- The initial release of the Visual dBase version of WSC.

Version 3.1: May 29, 2001.

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

Version 3.2: August 23, 2002.

- Added SELFTEST example program.
- 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: January 13, 2004.

- 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, which 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.
- Adjusted XModem/YModem timing for faster transfers.

Version 4.2: April 6, 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.

Version 4.3: October 10, 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 4, 2009

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

Version 5.0: January 8, 2010

- Supports 64-bits
- Added SioHexView function
- Added SioSleep function

Version 5.1: September 15, 2011

- Added SioRxWait function
- Added LISTER example program.

Version 5.2: July 14, 2012

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

Version 5.3: November 13, 2013

- Added SioLRC() that computes the "longitudinal redundancy check" per ISO 1155.
- SioQuiet() and SioWaitFo() 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 2, 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 22, 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.