



# Core Users Guide and Reference Virtual Serial Port

for Windows XP, Windows 2000, and Windows NT  
Constellation Data Systems, Inc.

[www.VirtualPeripherals.com](http://www.VirtualPeripherals.com)

Copyright © 2003, 2004 Constellation Data Systems, Inc ("CDS"). All rights reserved. Consult your software license agreement. Brand and product names are trademarks of their respective holders. Portions of this manual are © Microsoft Corporation, and are used by permission of the MSDN.

## Table of Contents

1.	Introduction and Overview .....	3
1.1	Virtual Serial Port Core Capabilities.....	3
1.2	The VSP Device Driver.....	4
1.3	VSP Device Enumerator.....	4
1.4	VSP Utilities.....	4
2.	Installation Procedure .....	5
2.1	Step by Step Instructions.....	5
2.2	Quick Installation Verification using VSP Enumerator .....	12
3.	Removal (Uninstall) Procedure.....	13
4.	Demonstration Period .....	14
5.	VSP Utilities.....	15
5.1	Syntax Conventions.....	15
5.2	<i>Virtual To Virtual</i> Utility .....	16
5.2.1	Data Flow of <i>Virtual To Virtual</i> .....	16
5.2.2	Command Line Parameters of <i>Virtual To Virtual</i> .....	16
5.2.3	Demonstration Using <i>Virtual To Virtual</i> .....	18
5.3	<i>Virtual To Physical</i> Utility .....	19
5.3.1	Data Flow of <i>Virtual To Physical</i> .....	19
5.3.2	Command Line Parameters of <i>Virtual To Physical</i> .....	20
5.3.3	Demonstration Using the <i>Virtual To Physical</i> Utility .....	25
5.3.4	Improper Connections using <i>Virtual To Physical</i> .....	27
5.4	<i>Multi Virtual to Physical</i> Utility.....	28
5.4.1	Data Flow of <i>Multi Virtual To Physical</i> .....	28
5.4.2	Command Line Parameters of <i>Multi Virtual To Physical</i> .....	28
5.4.3	Demonstration Using <i>Multi Virtual To Virtual</i> .....	34
5.5	<i>Add Port</i> Utility .....	34
5.5.1	Command Line Parameters of <i>Add Port</i> .....	34
5.5.2	Demonstration Using <i>Add Port</i> .....	34
5.6	<i>Delete Port</i> Utility .....	35
5.6.1	Command Line Parameters of <i>Delete Port</i> .....	36
5.6.2	Demonstration Using <i>Delete Port</i> .....	36
5.7	<i>Enum Ports</i> Utility .....	36
5.7.1	Command Line Parameters of <i>Enum Ports</i> .....	36
5.7.2	Demonstration Using <i>Enum Ports</i> .....	37
5.8	<i>Serial Number Entry</i> Utility.....	37
5.8.1	Command Line Parameters of <i>Serial Number Entry</i> Utility .....	37
5.8.2	Demonstration Using <i>Serial Number Entry</i> Utility.....	37
6.	Detailed Installation Verification Procedures .....	38
6.1	Verification of Port Names and Version Information .....	38
6.2	Installation Verification using <i>VirtualToVirtual</i> .....	39
7.	Notices.....	41
8.	Index of Acronyms and Abbreviations .....	42

# 1. Introduction and Overview

This manual describes the Virtual Serial Port (VSP) core components, and their installation, use and operation.

## 1.1 Virtual Serial Port Core Capabilities

The Virtual Serial Port (VSP) is a product of Constellation Data Systems, Inc (CDS). The VSP is a development accelerator, which can cut months or years from a development project, which requires a virtualized serial or communications resource. The VSP Core is capable of the following powerful activities:

- Hardware-less serial port interface emulation
- Easy capture of data from a serial port data.
- Easy generation of data into a serial port.
- High speed data transfers / transmissions.
- Multiplexing multiple data sources on a single serial port.
- Splitting data from a single source onto multiple serial ports.
- Serial port data redirection
- Serial port device simulation
- Dynamic Port Creation

The VSP is also extensible. Should you have custom requirements, the VSP *Software Development Kit* (SDK), is available. This SDK enables rapid development of custom VSP applications.

The VSP Core, as downloaded from our Web Site, is provided as a Demo. As a demo, the unit is fully functional with the exception that the Startup Screen will keep re-appearing periodically, requiring the user to obtain a serial number. Once a serial number has been acquired from Constellation Data Systems, Inc., the Startup Screen will cease to keep re-appearing. In the Demo Mode when this screen appears, the user must acknowledge it by pressing on "OK". This will take the user back to the License Agreement Dialog box requiring the user to accept the terms and conditions of the agreement. Once the VSP software has been registered, the Splash Screen will only appear for 10 seconds, at which time it will automatically disappear. The Startup Screen, in the registered user mode, can also be frozen to allow the user to view its contents for extended periods of time. See Section 2.2

## Description of VSP Core Components

### 1.2 The VSP Device Driver

The heart of the VSP core is the *VSP Device Driver*. This driver is a fully functional Windows serial port device driver. This driver presents the following interfaces:

1. A standard serial port interface to the Windows Operating System. This interface standard is known as the *WIN32 Communications API*. All serial port accesses, irrespective of programming paradigm (C/C++, Visual Basic, MFC, etc.), make use of this interface to access serial ports. Through this interface a Virtual Serial Port looks like any other serial port on a system.
2. An interface, unique to the VSP, which allows software control of serial port functions which were formerly only possible with physical hardware. This interface is described in "*VSP Applications Programming Interface Reference*" document.

### 1.3 VSP Device Enumerator

Another core component of the VSP Product is the *Device Enumerator*. This enumerator is typically run at system logon time. This enumerator presents the operator with a synopsis of serial port devices, both Virtual and Physical. The Virtual devices are enumerated in a dialog box, and shown on the left hand side of that box along with the underlying VSP driver components versioning information. The Physical devices are shown on the right hand side of the dialog box, along with some additional device specific information.

After user logon, the *Device Enumerator* remains running in the background, continuously monitoring the status of the system's Virtual Serial Ports.

See section 2.2, *Quick Installation Verification using VSP Enumerator* for related information.

### 1.4 VSP Utilities

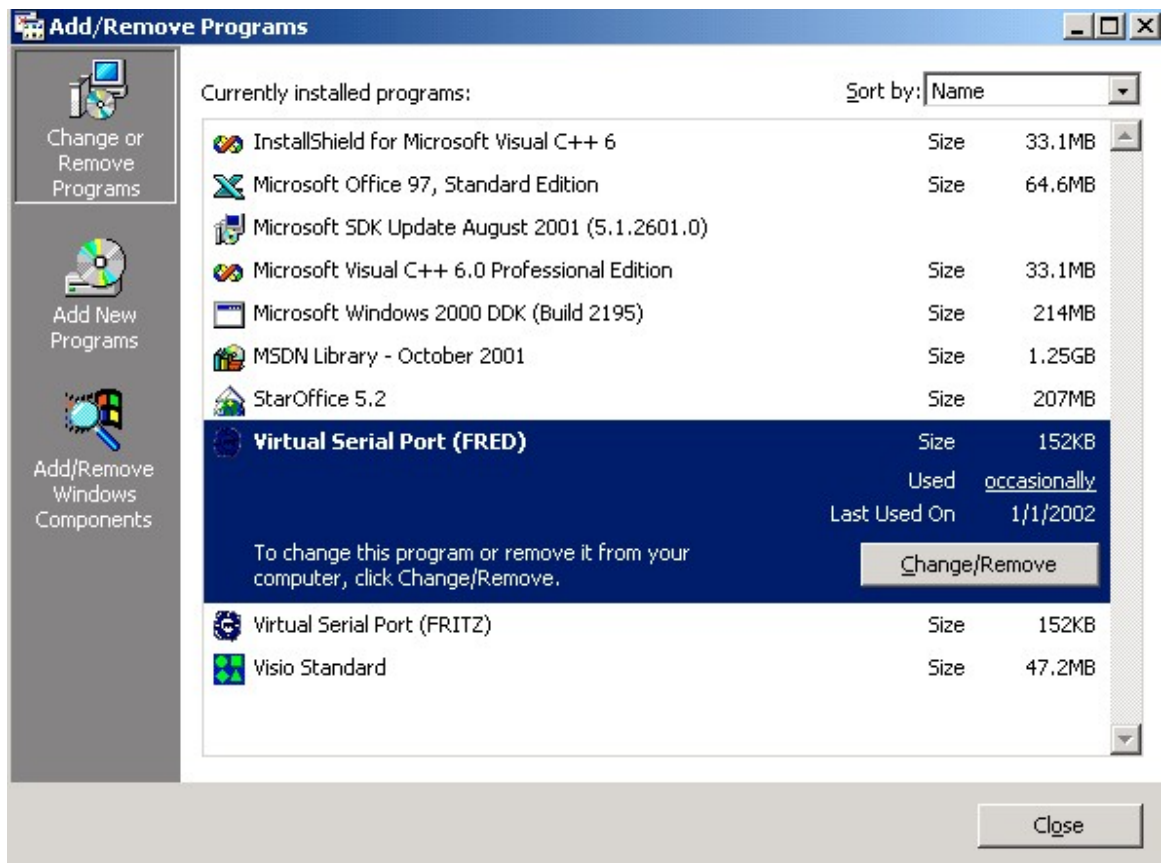
The VSP Utilities are powerful pre-built VSP applications. Many users will be able to use the VSP Utilities in stand-alone form. Other users, with more complex requirements may wish to consider using the VSP Software Development Kit (VSPSDK) to implement custom requirements. Consult Section 5, *VSP Utilities* for related information.

## 2. Installation Procedure

### 2.1 Step by Step Instructions

It is recommended that the VSP be installed on a clean installation of Windows. This may require you to reinstall the operating system.

1. **Log on to the target machine as the system administrator.**
2. Remove any existing VSP installations of a version prior to the target version. While multiple VSP installations may be made on a single PC, it is recommended that they all be of the same version. Go to the Control Panel function; Add/Remove programs, and identify any VSP installations of prior of prior version. Consider the following “Add/Remove Programs” dialog from a Windows 2000 installation:



In this example there are 2 virtual serial ports which should be removed. They were installed using the arbitrary port names “FRED” and “FRITZ”. Of course the port names will differ on your system. Remove all previous VSP

versions from the target system. The registry key “HKEY\_LOCAL\_MACHINE\SOFTWARE\Constellation Data Systems” will need to be removed if it exists, as will Virtual Serial Port entries in “HKEY\_LOCAL\_MACHINE\HARDWARE\DEVICEMAP\SERIALCOMM”. After successfully removing entries from the registry, reboot the target machine.

3. The setup executable is named using the following nomenclature: “VirtualSerialXXX.exe”, where XXX is a short version description embedded in the file name. For example, version 2.26 of the Core VSP, XXX would be 226, and the entire module would be named “VirtualSerial226.exe”.

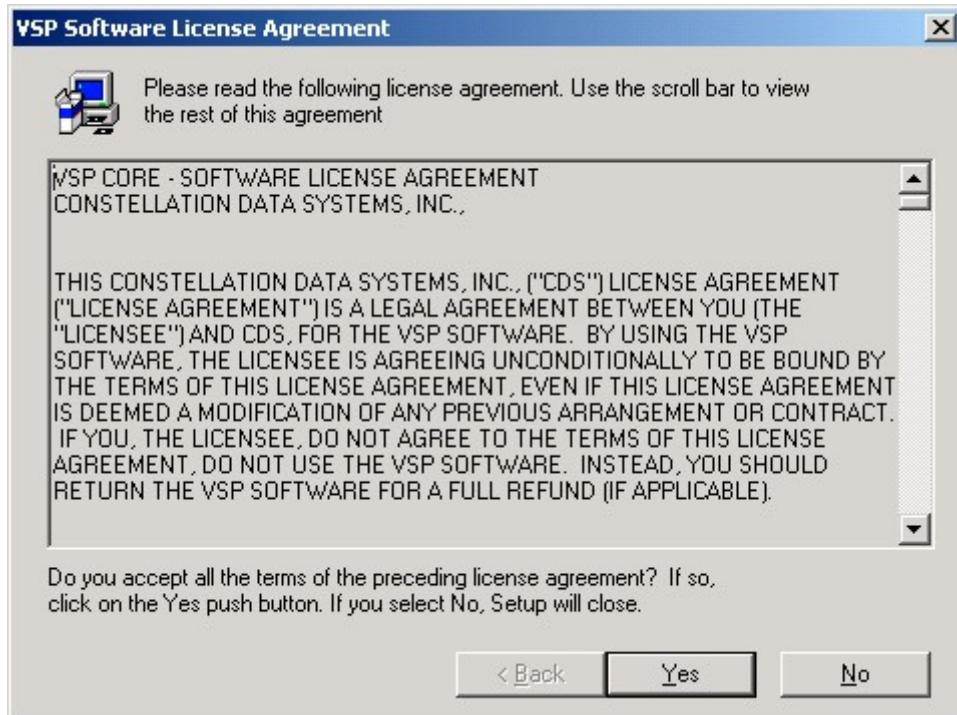
Run the setup executable. You will be prompted for the “Unzip To Folder”. The folder “c:\VirtualSerial” is recommended. The VSP Core installation files will then be unzipped into that directory. The VSP Core installation file set consists of the files shown in the following table. Note that while the modules may vary from one VSP version to another VSP version, the following modules should generally be present in the VSP installation directory (or installation disk).

<p>VspStartup.exe Os.dat _ISDEL.EXE SETUP.EXE Lang.dat _INST32I.EX_ _setup.dll setup.ins _sys1.cab DATA.TAG SETUP.INI _user1.cab layout.bin data1.cab setup.lid Vsp.sys License.txt Connector.bmp</p>	<p>VSP Core installation files, and install shield information.</p>
<p>Exe\VirtualToVirtual.exe Exe\VirtualToPhysical.exe Exe\MultiVirtualToPhysical.exe Exe\AddPort.exe Exe\DeletePort.exe Exe\EnumPorts.exe Exe\SerialNumEntry.exe</p>	<p>These are the VSP utilities. Consult Section 5, VSP Utilities, for more information</p>

4. Run "setup.exe" from the installation directory (generally "C:\VirtualSerial"). From the START button, select RUN, and then "c:\VirtualSerial\Setup.exe".

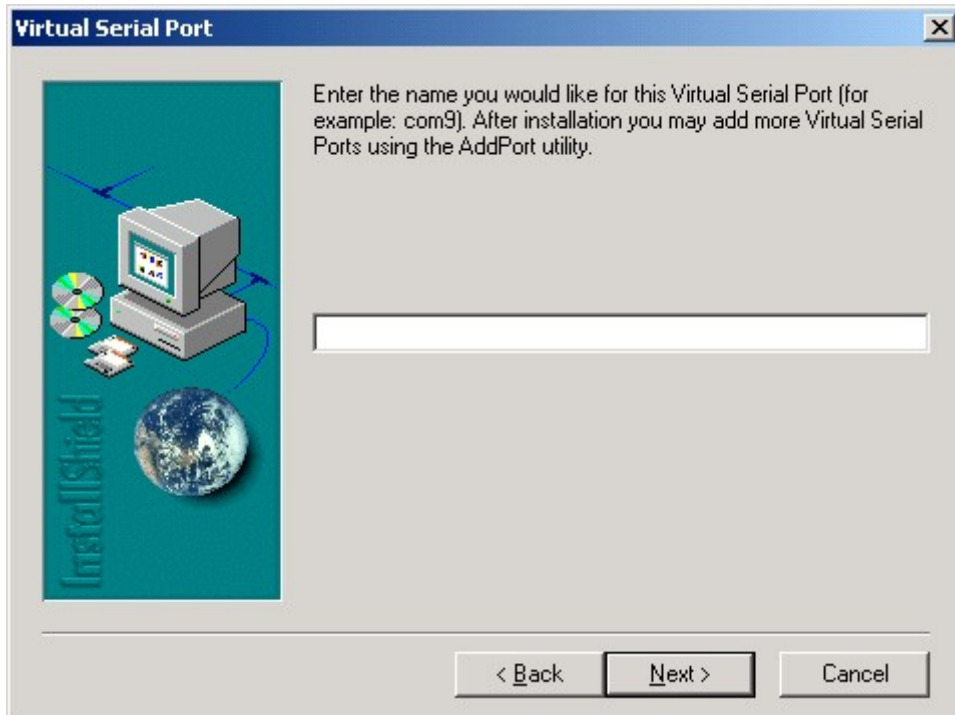
Should the installation device be a floppy disk, simply select the START button, then RUN "a:setup".

5. The VSP Software License agreement will then be displayed for your acceptance.



After reading the entire agreement, if you accept the Software License agreement's terms, indicate acceptance by selecting "Yes". Should you not be in agreement with the terms, select "No", and the installation procedure will terminate.

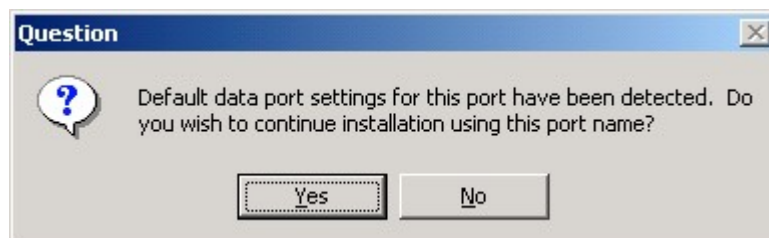
6. After the Software License dialog is displayed and accepted, the Install Shield Virtual Serial Port Dialog will be displayed:



Enter the name of the Virtual Serial Port that you would like to create.

This will be the same name that applications, such as HyperTerminal use to identify the VSP device. Avoid already defined port names, or common port names such as COM1 or COM2. Once a port name is chosen, such as ("Virtual1"), select "Next". The "Enter Product Serial Number" dialog box should then appear.

If a port with the selected name already exists, then a message box will appear which prompts you to select another name. You may also observe the following dialog:



This informative dialog simply indicates that a previous installation (using that port name) was detected, and that some residual port data still remains in the

registry. If the port name is correct, simply select YES, otherwise select NO and re-enter the port name.

7. The “Enter Product Serial Number” will allow you to enter the product serial number assigned to you by CDS. Users running the product for Demonstration purposes can simply enter “DEMO”. Should you choose to enter a Serial Number at a later date, you can do so through the *SerialNumEntry.exe* Utility; please refer to Section 5.8. Note: Product Serial Numbers should be protected in accordance with the terms of the Software License Agreement.

**Virtual Serial Port**

Please enter your name, the name of your company, and a Product Serial Number (assigned by vendor), OR for a 2 hour DEMO, enter 'DEMO' as a Serial Number below.

Name:

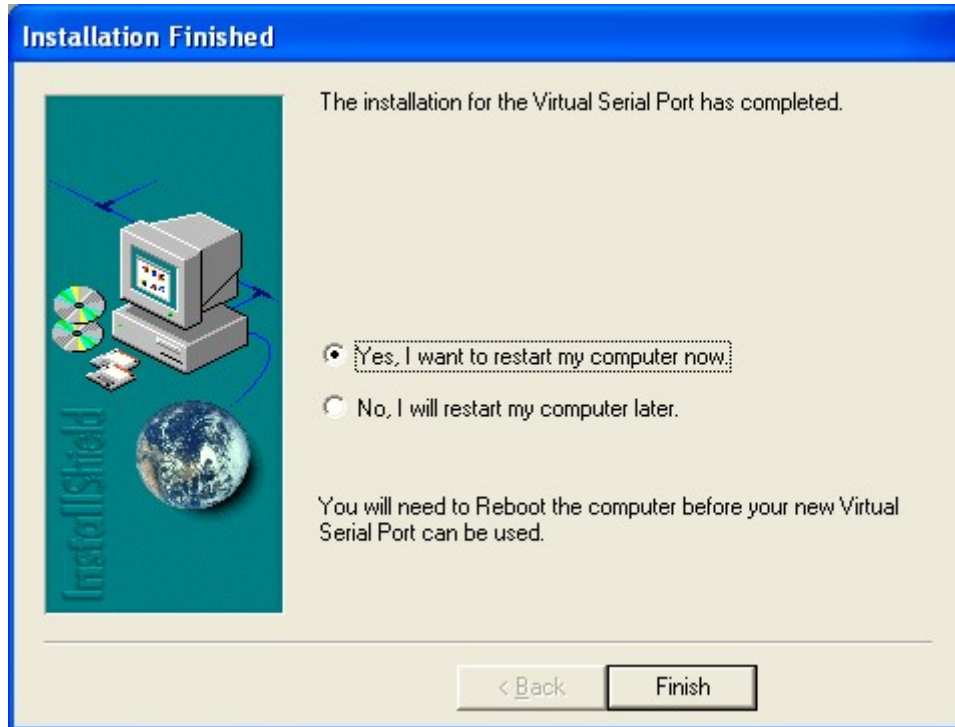
Company:

Serial:

< Back    Next >    Cancel

8. Once you enter a valid product serial number, the file copy phase of installation procedure will begin. Rather quickly the “Installation Finished” dialog box should appear.

9. The “Installation Finished” dialog box (shown below), prompts you to reboot the machine in order complete the installation process.

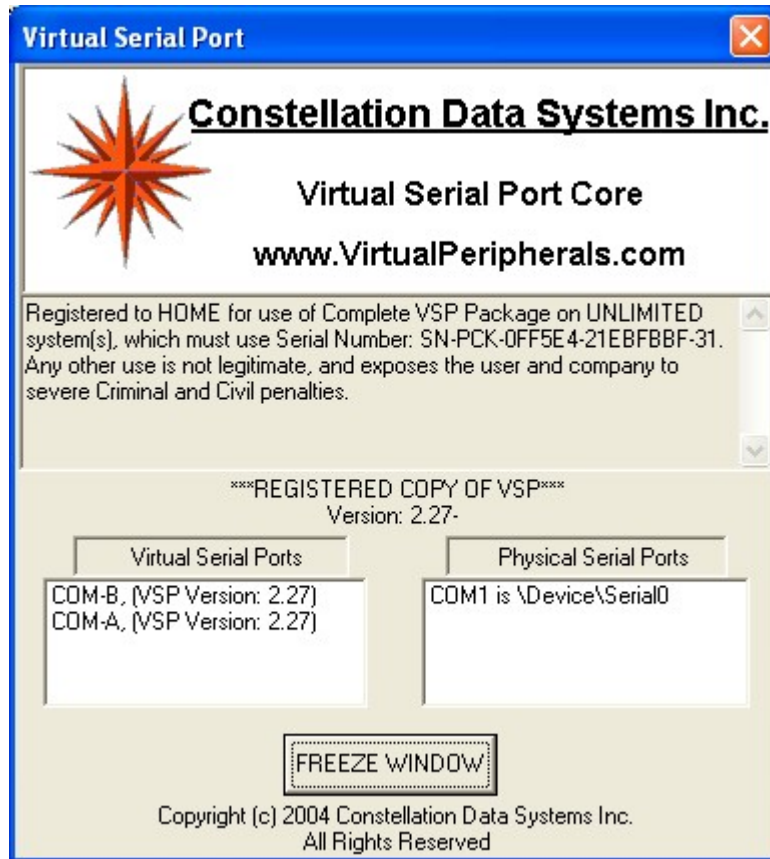


If at this time you wish to add another Virtual Serial Port, select the “No” option button and then select “Finish”. Repeat steps #7 through #9 in order for another Virtual Serial Port to be installed. Once all ports have been installed, it will be necessary to reboot the target machine.

Virtual Serial Ports can also be dynamically created outside of the Installation Procedure via the *Add Port Utility*. A feature of adding ports in this manner is that the target machine does not need to be rebooted in order for the newly installed Virtual Serial Ports to be operable. For more information, please refer to Section 5.5.

## 2.2 Quick Installation Verification using VSP Enumerator

Following reboot after successful installation, the following dialog box should appear at logon time. This dialog is referred to as the “Virtual Serial Port Enumerator”.



Note that this dialog enumerates all installed and functioning Virtual Serial Ports, as well as their version number. While it is possible that different versions of the VSP may be installed on the same system, configurations with mixed versions are not suggested. Review the VSP Device Enumerator after installation and verify that all Virtual Serial Ports are of the same version. Also verify that the “Startup Version” is of the same version number.

The physical serial devices and resources, which are generally either RS-232 ports or physical modems, are shown on the right hand side of the dialog box. Should you desire to see this information at any time after boot, it may be run from the “Start” bar, to “Programs”, and then “Startup”.

Note: an Error 87 shown after the port name indicates that the demonstration period has expired. Simply reboot to continue software demonstration for another period.

### 3. Removal (Uninstall) Procedure

Go to the Control Panel, and select the “Add/Remove Programs” function. Then select the VSP you wish to remove. A reboot after removal is recommended. Upon reboot the “Virtual Serial Port Startup Dialog” should no longer enumerate the VSP just removed.

Note: When the last VSP is removed from a system, the “Virtual Serial Port Startup Dialog” is not executed.

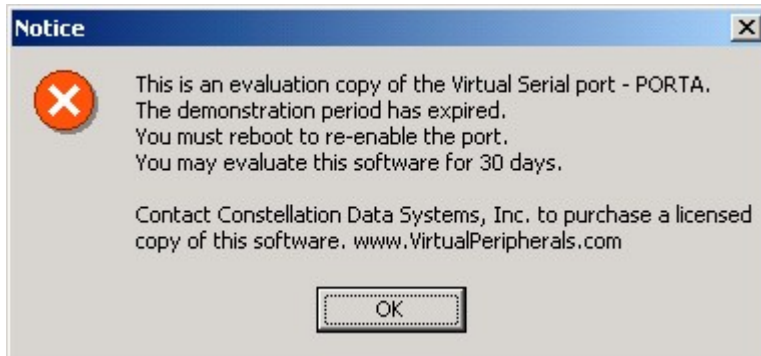
Errata NT4 - Install shield has been observed to not load uninstall data on certain Windows NT4 systems.	This interferes with the uninstall procedure. To remove the VSP from affected systems use the following procedure: <ol style="list-style-type: none"><li>1. Log on as the system administrator.</li><li>2. Use REGEDIT to remove all “VirtualSerial” subkeys from HKEY_LOCAL_MACHINE\System\CurrentControlSet\Services</li><li>3. Then right click on the START button and “explore all users”. Then delete all Virtual Serial Ports from the “Programs”, “Startup”.</li><li>4. Reboot the system for the changes to take affect.</li></ol>
--	--

The registry key “HKEY\_LOCAL\_MACHINE\SOFTWARE\Constellation Data Systems” will need to be removed if it exists.

Remove all Virtual Serial Port entries in:  
“HKEY\_LOCAL\_MACHINE\HARDWARE\DEVICEMAP\SERIALCOMM”.

## 4. Demonstration Period

The Virtual Serial Port provided for demonstration (“DEMO”) disables itself several hours following boot. Logons with disabled serial ports will cause the dialog box below to appear, and then those disabled ports will be enumerated at startup with “Error 87”.



Should continued evaluation be desired, simply reboot the target machine, and several hours of additional evaluation will be possible.

## 5. VSP Utilities

The VSP Utilities are powerful pre-built VSP applications. Many users will be able to use the VSP Utilities in stand-alone form. Other users, with more complex requirements may consider using the VSP SDK to implement custom requirements. This is available for free and can be downloaded from our website at [www.VirtualPeripherals.com](http://www.VirtualPeripherals.com).

### 5.1 Syntax Conventions

Syntax represents the command line format you must follow when using the VSP utilities. Unless otherwise indicated, you may enter commands and parameters in either upper or lower case. It is expected that you type the ENTER key at the end of a VSP utility command.

Consider the following mapping:

*utility* parameter1 {parameter 2|parameter2a} [parameter 3a | parameter 3b] ...  
parameter 'n' [switches]

<u>Element</u>	<u>Usage</u>
<i>Utility</i>	The name of the utility command (VirtualToVirtual, for example).
[ ]	Brackets indicate that an item is optional.
{ }	Braces indicate a choice of parameters, where one of the choices is mandatory.
	Separates two or more mutually exclusive choices.
...	Indicates that the preceding parameter may be repeated zero or more times.
[switches]	Zero or more optional command switches. A switch begins with a slash, "/p", for example.

While each VSP command will have its own syntax mapping, in the preceding VSP Syntax mapping:

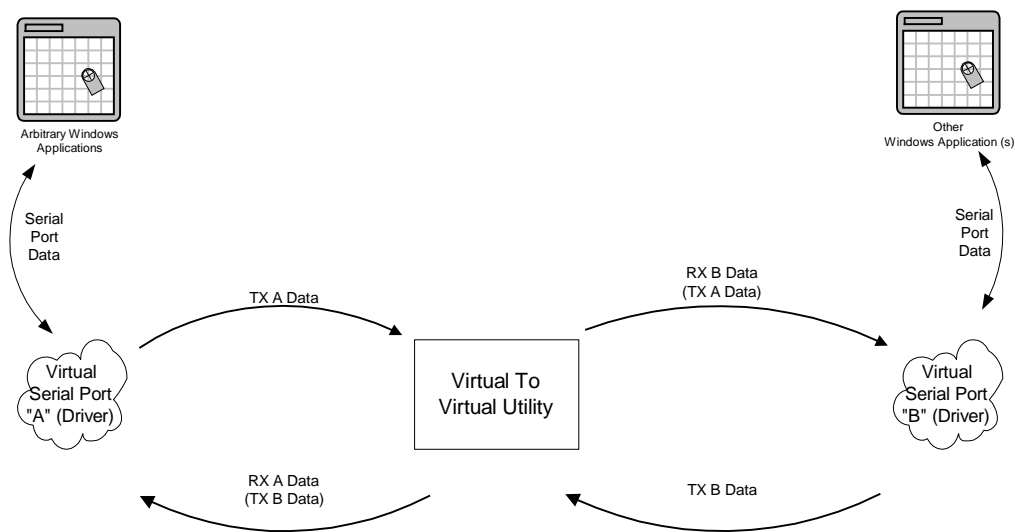
- *'utility'* is the name of the VSP Utility Command.
- Parameter 1 is mandatory
- Parameters 2a or 2b are mandatory, and are mutually exclusive.
- Parameters 3a and 3b are optional, and when specified are mutually exclusive.
- Additional parameters may follow the optional parameter 3, and are may be considered as parameters ... n-2, n-1, n.
- Optional command line switches as required.

## 5.2 Virtual To Virtual Utility

The *Virtual To Virtual* (“virtualtovirtual.exe”) utility sends data received from one Virtual Serial Port to another, and visa-versa. It is simply a “high tech” null modem for virtual serial devices.

### 5.2.1 Data Flow of *Virtual To Virtual*

Consider the following hypothetical data flow diagram, which illustrates using the utility to connect two arbitrary Windows serial port aware applications:



### 5.2.2 Command Line Parameters of *Virtual To Virtual*

Consider the following syntax mapping:

<p><b>Syntax</b></p>	<p><b>VirtualToVirtual</b> VIRTUALA VIRTUALB [switches]</p> <p>VIRTUALA ----- Virtual Serial Port VIRTUALB ----- Virtual Serial Port</p> <p>[switches] – Optional settings. All VALUE's are in decimal.</p> <p>/DelayWriteFileByteToByteA:VALUE /DelayWriteFileConstantA:VALUE /DelayReadFileByteToByteA:VALUE /DelayReadFileConstantA:VALUE /DelayWriteFileByteToByteB:VALUE /DelayWriteFileConstantB:VALUE /DelayReadFileByteToByteB:VALUE /DelayReadFileConstantB:VALUE</p>
<p><b>Parameters</b></p>	<p><b>VirtualA</b> – Data is read from VirtualA, and written to VirtualB. <b>VirtualB</b> – Data is read from VirtualB, and written to VirtualA.</p>
<p><b>Switches</b></p>	<p><b>/?</b> – Prints command line usage.</p> <p><b>/DelayWriteFileByteToByteA:VALUE</b> – Specifies the amount of time, in milliseconds, to delay between bytes when writing to VirtualA</p> <p><b>/DelayWriteFileConstantA:VALUE</b> - Specifies a constant amount of time, in milliseconds, to delay between bytes when writing to VirtualA</p> <p><b>/DelayReadFileByteToByteA:VALUE</b> - Specifies the amount of time, in milliseconds, to delay between bytes when reading to VirtualA</p> <p><b>/DelayReadFileConstantA:VALUE</b> - Specifies the amount of time, in milliseconds, to delay between bytes when writing to VirtualA</p> <p><b>/DelayWriteFileByteToByteB:VALUE</b> - Specifies the amount of time, in milliseconds, to delay between bytes when writing to VirtualB</p> <p><b>/DelayWriteFileConstantB:VALUE</b> - Specifies a constant amount of time, in milliseconds, to delay between bytes when writing to VirtualB</p> <p><b>/DelayReadFileByteToByteB:VALUE</b> - Specifies the amount of time, in milliseconds, to delay between bytes when reading to VirtualB</p> <p><b>/DelayReadFileConstantB:VALUE</b> - Specifies a constant amount of time, in milliseconds, to delay between bytes when reading to VirtualB</p>
<p><b>Runtime</b></p>	<p>'A' then 'ENTER' typed on the command line console causes the utility to cancel wait for changes on VirtualA.</p> <p>'B' then 'ENTER' typed on the command line console causes the utility to cancel wait for changes on VirtualB.</p>

	'Q' then 'ENTER' typed on the command line console causes the utility to quit (exit).
--	---

### 5.2.3 Demonstration Using *Virtual To Virtual*

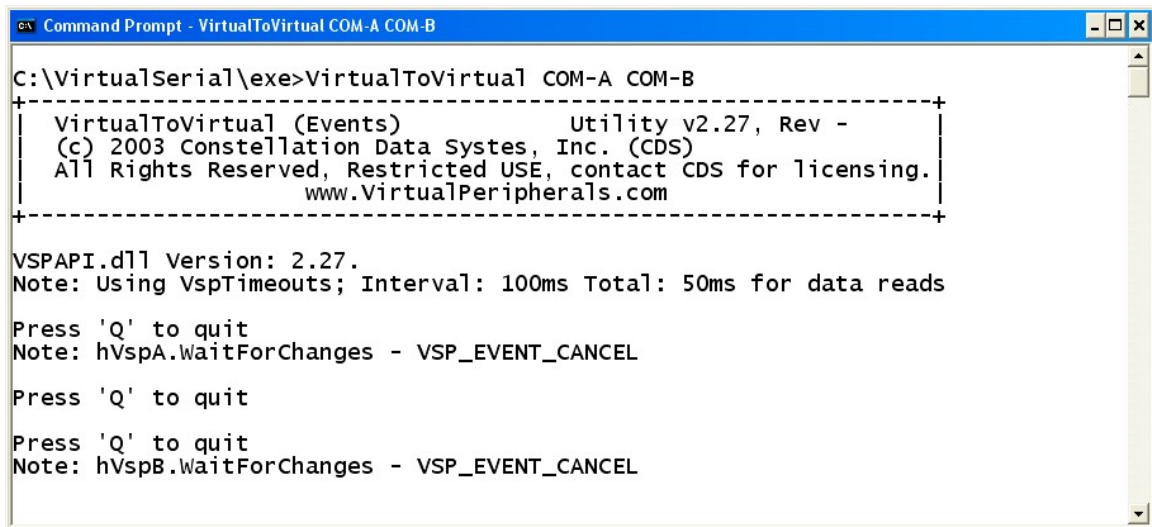
In the following example, the *Virtual To Virtual* utility has been used to connect the Transmit (TX) data of Virtual Serial Port "PORTA" with the Receive (RX) data of Virtual Serial Port "PORTB". Additionally, the Transmit (TX) data of Virtual Serial Port "PORTB" has been connected with the Receive (RX) data of Virtual Serial Port "PORTA". This is the normal "Full Duplex" usage of the *Virtual To Virtual* utility.

```
Command Prompt - VirtualToVirtual COM-A COM-B /DelayWriteFileByteToByteA:100 /DelayWriteFileConstantA:1000
C:\VirtualSerial\exe>VirtualToVirtual COM-A COM-B /DelayWriteFileByteToByteA:100 /DelayWriteFileConstantA:1000
+-----+
| VirtualToVirtual (Events)          Utility v2.27, Rev -          |
| (c) 2003 Constellation Data Syses, Inc. (CDS)                    |
| All Rights Reserved, Restricted USE, contact CDS for licensing.   |
|                               www.VirtualPeripherals.com          |
+-----+
VSPAPI.dll Version: 2.27.
Note: Using VspTimeouts; Interval: 100ms Total: 50ms for data reads
Press 'Q' to quit
```

Two *Hyperterminals* (standard Windows Accessory) may then be ran; one connected to "PORTA" (in this example), and the other connected to "PORTB". Data transfers may then be accomplished and monitored through the Hyperterminal consoles.

The *Virtual To Virtual* utility continues to operate until 'Q' and ENTER are typed on the console. This causes a QUIT operation and *Virtual To Virtual* will gracefully shutdown and say "... Goodbye ...".

*Virtual To Virtual*, as distributed (beginning with Version 2.24), is setup with 100ms (0.1 second) timeouts between bytes, after the first byte is processed, as well as a 50ms (.05 second) total read timeouts. These values may be changed by rebuilding the corresponding *Reference Design* using the VSPSDK.



```
Command Prompt - VirtualToVirtual COM-A COM-B
C:\VirtualSerial\exe>VirtualToVirtual COM-A COM-B
-----+-----
VirtualToVirtual (Events)          Utility v2.27, Rev -
(c) 2003 Constellation Data System, Inc. (CDS)
All Rights Reserved, Restricted USE, contact CDS for licensing.
www.VirtualPeripherals.com
-----+-----
VSPAPI.dll Version: 2.27.
Note: Using VspTimeouts; Interval: 100ms Total: 50ms for data reads
Press 'Q' to quit
Note: hVspA.WaitForChanges - VSP_EVENT_CANCEL
Press 'Q' to quit
Press 'Q' to quit
Note: hVspB.WaitForChanges - VSP_EVENT_CANCEL
```

### 5.3 *Virtual To Physical* Utility

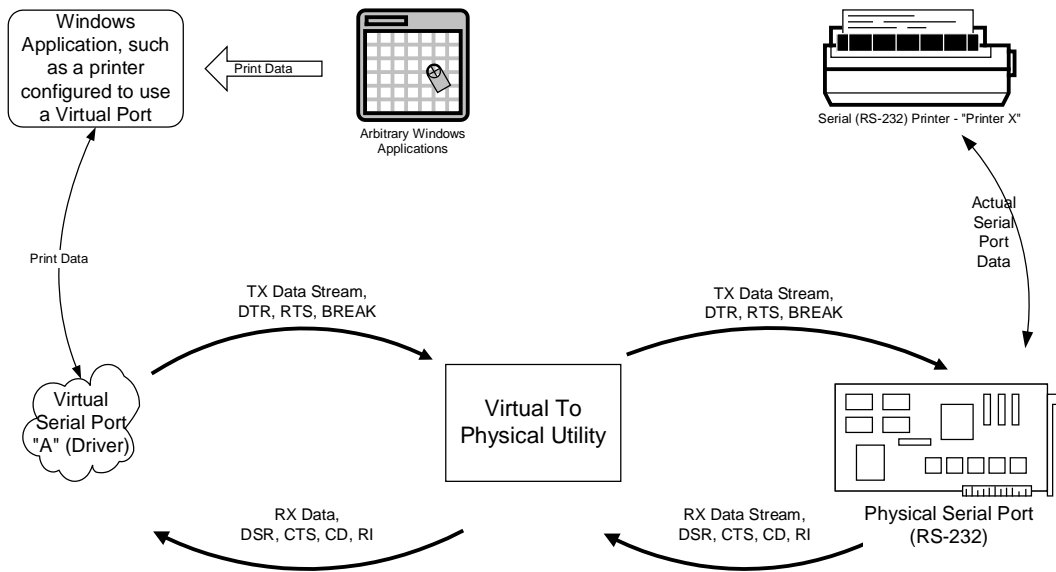
The *Virtual To Physical* (“virtualtophysical.exe”) utility allows demonstration of the Virtual Serial Port capabilities. This utility simply sends data, modem control and status lines, received from one Virtual Serial Port to a physical serial port (such as COM1), and visa-versa. In essence, this utility simply places a Virtual Serial Port endpoint on an actual serial port endpoint.

A typical use of this utility is to demonstrate proof of concept for using this utility’s source code (distributed as a “Reference Design” in the VSP SDK) as a starting point for a customer VSP application.

#### 5.3.1 Data Flow of *Virtual To Physical*

The following Diagram illustrates the data flow of a hypothetical use of the *Virtual To Physical* demonstrating printing through the utility to a physical printer through a physical communications (serial) port:

**Core Users Guide and Reference**  
**Virtual Serial Port**



### 5.3.2 Command Line Parameters of *Virtual To Physical*

Consider the following syntax mapping:

<b>Syntax</b>	<p><b>VirtualToPhysical</b> PHYSICAL VIRTUAL [switches]</p> <p>PHYSICAL ----- Physical port (COM1, COM2, etc.)          VIRTUAL ----- Virtual Serial Port</p> <p>[switches] – Optional settings of the physical port unless otherwise noted ('Virtual' nomenclature). All VALUE's are in decimal.</p> <pre> /Baud:VALUE /Parity:{EVEN MARK NO ODD SPACE} /RxParity:{ENABLE DISABLE} /StopBits:{ONE TWO ONEANDHALF} /OutCtsFlow:{ENABLE DISABLE} /DtrControl:{DISABLE ENABLE HANDSHAKE} /DsrSensitive:{ENABLE DISABLE} /TxContinueOnXoff:{ENABLE DISABLE} /OutX:{ENABLE DISABLE} /InX:{ENABLE DISABLE} /XonLim:VALUE /XoffLim:VALUE /XoffChar:VALUE /XonChar:VALUE /NullDiscard{ENABLE DISABLE} /RtsControl:{DISABLE ENABLE HANDSHAKE TOGGLE} /ErrorReplaceChar:VALUE </pre>
---------------	---

<p><b>Syntax (continued)</b></p>	<p>/ByteSize:VALUE /ReadIntervalTimeout:VALUE /VirtualReadIntervalTimeout:VALUE /VirtualReadTotalTimeout:VALUE /MonitorHex /MonitorAscii /DelayWriteFileByteToByte:VALUE /DelayWriteFileConstant:VALUE /DelayReadFileByteToByte:VALUE /DelayReadFileConstant:VALUE</p>
<p><b>Parameters</b></p>	<p><b>PHYSICAL</b> – The physical communications port to use. This device must be an industry standard physical communications device running the standard Windows serial port device driver. Generally these devices are named either COM1 or COM2. Data received at the indicated physical port is reproduced (written) to the VIRTUAL port specified.</p> <p><b>VIRTUAL</b> – Virtual Serial Port. Data received at the port is written to the PHYSICAL port.</p>
<p><b>Switches</b></p>	<p><b>/Baud:VALUE</b> – The valid decimal data rate VALUE in bits per second (pbs) for the physical port. Examples of valid data rates are 300, 600, 1200, 2400, 4800, 9600, 19200, 38400, and so on.</p> <p><b>/Parity:{EVEN MARK NO ODD SPACE}</b> - Parity scheme to be used in transmitting data and receiving data on the physical port.</p> <p><b>/RxParity:{ENABLE DISABLE}</b> – Indicates whether parity checking is enabled for data received on the physical port. If enabled, parity checking is performed.</p> <p><b>/StopBits:{ONE TWO ONEANDHALF}</b> - Number of stop bits to be used for data transmitted and received on the physical port.</p> <p><b>/OutCtsFlow:{ENABLE DISABLE}</b> - Indicates whether the physical port's CTS (clear-to-send) signal is monitored for output flow control. If enabled and CTS is turned off, output is suspended until CTS is sent again.</p> <p><b>/DtrControl:{DISABLE ENABLE HANDSHAKE}</b> – DTR (data-terminal-ready) flow control of the physical port. DISABLE causes the DTR line to disable (de-assert) when the physical device is opened and leaves it disabled. ENABLE causes the DTR line to assert when the physical device is opened and leaves it asserted. HANDSHAKE allows DTR handshaking on the physical device, if handshaking is enabled</p> <p><b>/DsrSensitive:{ENABLE DISABLE}</b> - Indicates whether the physical port's communications driver is sensitive to the state of the DSR signal. If enabled, the physical device's driver ignores any bytes received, unless the DSR modem input line is high (asserted).</p> <p><b>/TxContinueOnXoff:{ENABLE DISABLE}</b> - Indicates whether</p>

<p><b>Switches (continued)</b></p>	<p>transmission stops when the input buffer is full and the physical device's driver has transmitted the <b>XoffChar</b> character. If enabled, transmission continues after the input buffer has come within <b>XoffLim</b> bytes of being full and the driver has transmitted the <b>XoffChar</b> character to stop receiving bytes. If not enabled, transmission does not continue until the input buffer is within <b>XonLim</b> bytes of being empty and the driver has transmitted the <b>XonChar</b> character to resume reception.</p> <p><b>/OutX:{ENABLE DISABLE}</b> Indicates whether XON/XOFF flow control is used during transmission on the physical port. If enabled, transmission stops when the <b>XoffChar</b> character is received and starts again when the <b>XonChar</b> character is received</p> <p><b>/InX:{ENABLE DISABLE}</b> - Indicates whether XON/XOFF flow control is used during reception on the physical port. If enabled, the <b>XoffChar</b> character is sent when the input buffer comes within <b>XoffLim</b> bytes of being full, and the <b>XonChar</b> character is sent when the input buffer comes within <b>XonLim</b> bytes of being empty.</p> <p><b>/XonLim:VALUE</b> – On the physical port, the minimum number of bytes allowed in the input buffer before flow control is activated to inhibit the sender. Note that the sender may transmit characters after the flow control signal has been activated, so this value should never be zero. This assumes that either XON/XOFF, RTS, or DTR input flow control is specified using <b>/InX</b>, <b>/RtsControl</b>, or <b>/DtrControl</b>.</p> <p><b>/XoffLim:VALUE</b> – On the physical port, the maximum number of bytes allowed in the input buffer before flow control is activated to allow transmission by the sender. This assumes that either XON/XOFF, RTS, or DTR input flow control is specified in <b>/InX</b>, <b>/RtsControl</b>, or <b>/DtrControl</b>. The maximum number of bytes allowed is calculated by subtracting this value from the size, in bytes, of a buffer specific to the physical devices driver.</p> <p><b>/XonChar:VALUE</b> - Value of the XON character for both transmission and reception on the physical port.</p> <p><b>/XoffChar:VALUE</b> - Value of the XOFF character for both transmission and reception on the physical port.</p> <p><b>/NullDiscard{ENABLE DISABLE}</b> - Indicates whether null bytes are discarded. If enabled then null bytes are discarded when received on the physical port.</p> <p><b>/RtsControl:{DISABLE ENABLE HANDSHAKE TOGGLE}</b> - RTS (request-to-send) flow control of the physical port. DISABLE causes the RTS line to be de-asserted (disabled) when the device is opened and leaves it de-asserted (disabled). ENABLE asserts (enables) the RTS line when the device is opened and leaves it asserted (on). HANDSHAKE allows RTS handshaking. The physical port's driver asserts (raises) the RTS line when the "type-ahead" (input) buffer is less than one-half full and de-asserts (lowers) the RTS line when the buffer is more than three-</p>
--	--

<p><b>Switches (continued)</b></p>	<p>quarters full. <b>TOGGLE</b>, which is valid in Windows NT/2000/XP, specifies that the RTS line will be asserted (high) if bytes are available for transmission. After all buffered bytes have been sent, the RTS line will be de-asserted (low).</p> <p><b>/ErrorReplace:{ENABLE DISABLE}</b> – Indicates whether bytes received on the physical port with parity errors are replaced with the character specified by the <b>/ErrorReplaceChar</b> value. If enabled, and the <b>/Parity</b> is enabled, then replacement occurs.</p> <p><b>/ErrorReplaceChar:VALUE</b> – Value of the character used to replace bytes received on the physical port, with a parity error.</p> <p><b>/ByteSize:VALUE</b> - Number of bits in the bytes transmitted and received on the physical port.</p> <p><b>/ReadIntervalTimeout:VALUE</b> - Maximum time, in milliseconds, allowed to elapse between the arrival of two characters on the physical port communications line. During a WIN32 <b>ReadFile</b> operation, the time period begins when the first character is received. If the interval between the arrival of any two characters exceeds this amount, the WIN32 <b>ReadFile</b> operation is completed and any buffered data is returned. A value of zero indicates that interval time-outs are not used.</p> <p><b>/VirtualReadIntervalTimeout:VALUE</b> - Maximum time, in milliseconds, allowed to elapse between the arrival of two characters on the virtual serial port. During a WIN32 <b>ReadFile</b> operation, the time period begins when the first character is received. If the interval between the arrival of any two characters exceeds this amount, the WIN32 <b>ReadFile</b> operation is completed and any buffered data is returned. A value of zero indicates that interval time-outs are not used.</p> <p><b>/VirtualReadTotalTimeout:VALUE</b> VALUE, in milliseconds, used to calculate the total time-out period for read operations on the Virtual Serial Port.</p> <p><b>/MonitorHex</b> – prints a data trace on the console in Hex.</p> <p><b>/MonitorAscii</b> – prints a data trace on the console in ASCII.</p> <p><b>/DelayWriteFileByteToByte:VALUE</b> - Specifies the amount of time, in milliseconds, to delay between bytes when writing to the Virtual Serial Port</p> <p><b>/DelayWriteFileConstant:VALUE</b> - Specifies a constant amount of time, in milliseconds, to delay between bytes when writing to the Virtual Serial Port</p> <p><b>/DelayReadFileByteToByte:VALUE</b> - Specifies the amount of time, in milliseconds, to delay between bytes when reading to the Virtual Serial Port</p> <p><b>/DelayReadFileConstant:VALUE</b> - Specifies a constant amount of time, in milliseconds, to delay between bytes when reading to the Virtual Serial Port</p>
--	---

<b>Runtime</b>	'Q' then 'ENTER', on the keyboard causes the utility to exit.
----------------	---

The utility ("VirtualToPhysical.exe") is found in the "exe" directory of the installation files. This utility can be run from either the command line, or from standard Windows shortcuts. Consider the following command line capture:



```
Command Prompt
C:\VirtualSerial\exe>VirtualToPhysical /?
-----
VirtualToPhysical Utility (Events)          v2.27, Rev -
(c) 2003 Constellation Data Systems, Inc. (CDS)
All Rights Reserved, restricted USE, contact CDS for licensing.
www.VirtualPeripherals.com
-----

Usage: VirtualToPhysical  PHYSICAL  VIRTUAL

    PHYSICAL - the physical port (COM1 or COM2)
    VIRTUAL  - the virtual serial port name

[switches] - optional settings of the physical port unless otherwise noted
('Virtual' nomenclature). All VALUE's are in decimal.

/Baud:VALUE
/Parity:{EVEN|MARK|NO|ODD|SPACE}
/RxParity:{ENABLE|DISABLE}
/StopBits:{ONE|TWO|ONEANDHALF}
/OutCtsFlow:{ENABLE|DISABLE}
/DtrControl:{DISABLE|ENABLE|HANDSHAKE}
/DsrSensitive:{ENABLE|DISABLE}
/TxContinueOnXoff:{ENABLE|DISABLE}
/OutX:{ENABLE|DISABLE}
/InX:{ENABLE|DISABLE}
/XonLim:VALUE
/XoffLim:VALUE
/XonChar:VALUE
/XoffChar:VALUE
/NullDiscard{ENABLE|DISABLE}
/RtsControl:{DISABLE|ENABLE|HANDSHAKE|TOGGLE}
/ErrorReplace:{ENABLE|DISABLE}
/ErrorReplaceChar:VALUE
/ByteSize:VALUE
/ReadIntervalTimeout:VALUE
/MonitorHex
/MonitorAscii
/DelayWriteFileByteToByte:VALUE
/DelayWriteFileConstant:VALUE
/DelayReadFileByteToByte:VALUE
/DelayReadFileConstant:VALUE

C:\VirtualSerial\exe>
```

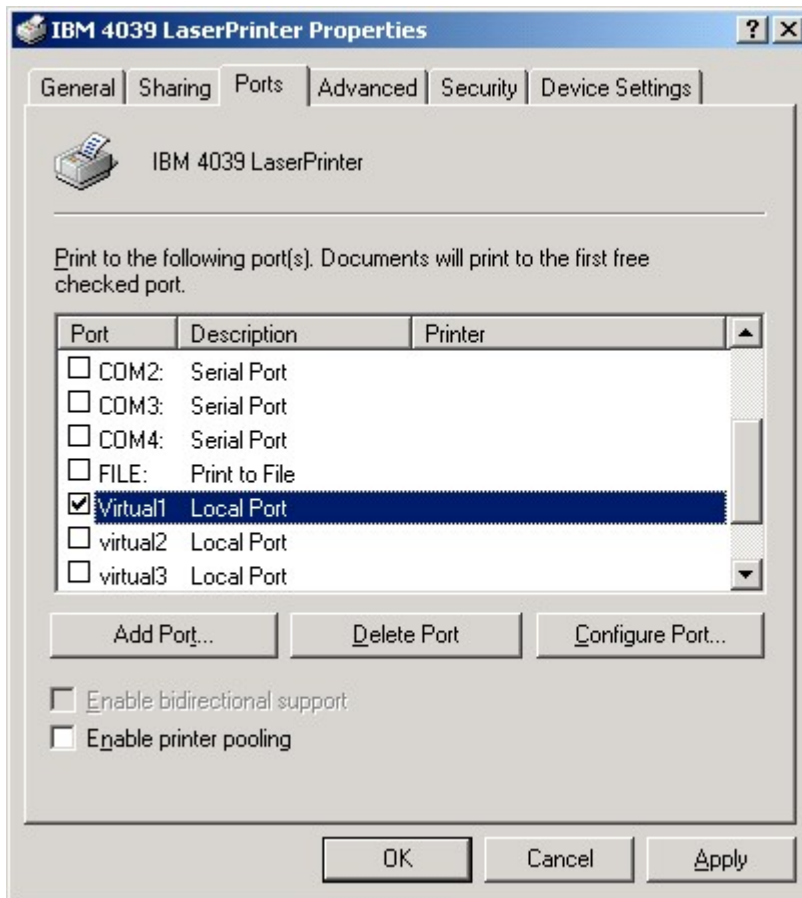
In this example, utility has been run from the command line with the "/" parameter. This parameter causes the utility to display its command line usage. Note that the utility has a version number which should match that of the underlying VSP drivers being used.

<b><u>Important:</u></b>	<p>The physical (“real”) com port is setup to use a default data rate of 2400bps, and 8 data bits, 1 stop and 1 start bit, and no parity. Handshaking is XON / XOFF. These settings may be overridden with <u>careful</u> use of the command line switches.</p> <p>The data rate and device settings on the VSP side are not relevant since the VSP transmits and receives data without regard to the device settings (baud/data bits, etc).</p>
--------------------------	--

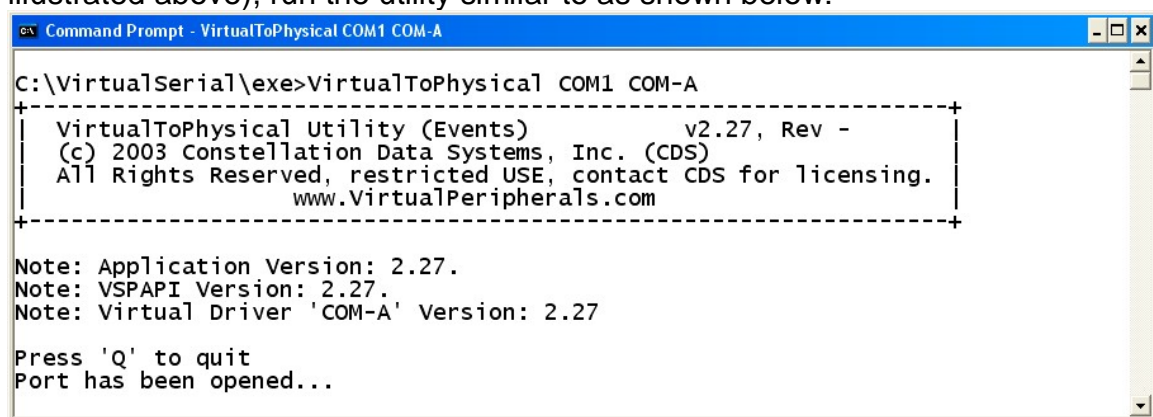
### 5.3.3 Demonstration Using the *Virtual To Physical* Utility

A helpful demonstration of VSP capabilities can easily be performed using the *Virtual To Physical* utility, a Virtual Serial Port, a physical serial port and a simple serial printer. In the following example, the *Virtual To Physical* utility has been used to connect the Transmit (TX) data of Virtual Serial Port “Virtual1” with the Receive (RX) data of Virtual Serial Port “COM1”, and visa-versa. A printer setup, such as the following, will allow print data to be delivered to a VSP, in this case, “Virtual1”. From the “Start” bar, run “Settings”, and “Printers”, and select the target printer. Then select “Properties”, and a dialog similar to the one shown below should be displayed.

<i>Tip</i>	<p>XON / XOFF handshaking is suggested for operations using the VSP and VSP utilities with a physical piece of hardware, such as a printer. This mode of handshaking will have to be setup in the <i>Printer’s Property Page</i> (“Configure Port”) as well as physically in the printer through a procedure specified by the manufacturer of the printer.</p>
------------	--



Setup the printer to use the desired Virtual Serial Port (“Virtual1” in the example above). After the printer has been setup to use a Virtual serial port (as illustrated above), run the utility similar to as shown below:

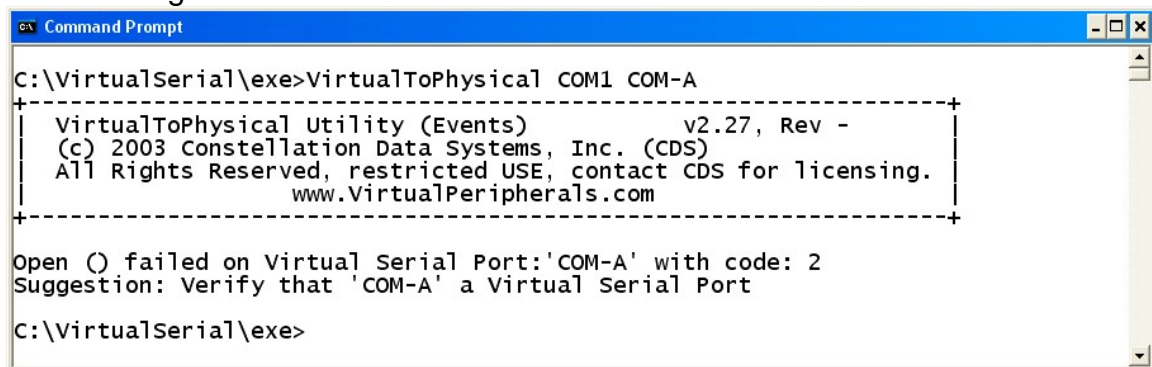


Observe also that the *Virtual To Physical* utility continues to operate until “Q” and “ENTER” are entered on the command line. This causes a QUIT operation and the utility will gracefully shutdown and say “.... Goodbye ....”.

Also notice that the version information of the underlying driver and the *Virtual To Physical* utility has been displayed.

### 5.3.4 Improper Connections using *Virtual To Physical*.

Should a physical serial port device be locked by another component, such as a printer connection already established on the physical serial port, then the connection using may fail. Note that when spooling is enabled on the print device, the print spooler may only “lock” the physical device during actual data transit periods. Should the device be locked, you will observe output similar to the following:



```
Command Prompt
C:\VirtualSerial\exe>VirtualToPhysical COM1 COM-A
+-----+
| VirtualToPhysical Utility (Events)          v2.27, Rev - |
| (c) 2003 Constellation Data Systems, Inc. (CDS)         |
| All Rights Reserved, restricted USE, contact CDS for licensing. |
| www.VirtualPeripherals.com                       |
+-----+
Open () failed on Virtual Serial Port: 'COM-A' with code: 2
Suggestion: Verify that 'COM-A' a Virtual Serial Port
C:\VirtualSerial\exe>
```

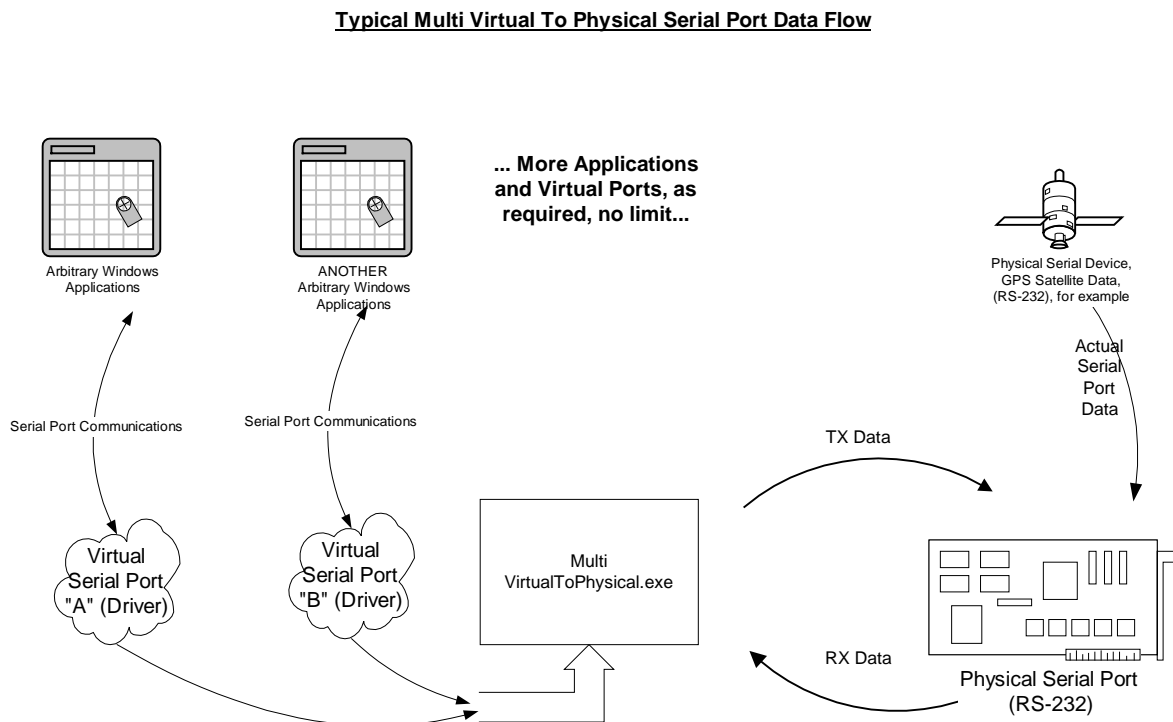
To clear this condition, free the device, in this case by simply removing the printer connection to the physical port.

## 5.4 Multi Virtual to Physical Utility

The *Multiple Virtual To Physical* utility reflects the multiple *Virtual Serial Ports* on a single physical serial port. An engineer may wish to use this utility for a GPS data splitter, data activity monitor, or a “data Y”.

### 5.4.1 Data Flow of *Multi Virtual To Physical*

Consider the following data flow diagram, which illustrates using the utility to connect GPS data from a serial port, to several arbitrary Windows applications:



### 5.4.2 Command Line Parameters of *Multi Virtual To Physical*

Consider the following syntax mapping:

<b>Syntax</b>	<p><b>MultiVirtualToPhysical</b> P1 V1 [V2 ...] [switches]</p> <p>P1 ----- Physical port (COM1, COM2, etc.)  V1, V2 . . . ----- One or more virtual serial ports</p>
---------------	--

<p><b>Syntax (continued)</b></p>	<p>[switches] – Optional settings of the physical port unless otherwise noted ('Virtual' nomenclature). All VALUE's are in decimal.</p> <p>/Baud:VALUE          /Parity:{EVEN MARK NO ODD SPACE}          /RxParity:{ENABLE DISABLE}          /StopBits:{ONE TWO ONEANDHALF}          /OutCtsFlow:{ENABLE DISABLE}          /DtrControl:{DISABLE ENABLE HANDSHAKE}          /DsrSensitive:{ENABLE DISABLE}          /TxContinueOnXoff:{ENABLE DISABLE}          /OutX:{ENABLE DISABLE}          /InX:{ENABLE DISABLE}          /XonLim:VALUE          /XoffLim:VALUE          /XoffChar:VALUE          /XonChar:VALUE          /NullDiscard{ENABLE DISABLE}          /RtsControl:{DISABLE ENABLE HANDSHAKE TOGGLE}          /ErrorReplaceChar:VALUE          /ByteSize:VALUE          /ReadIntervalTimeout:VALUE          /VirtualReadIntervalTimeout:VALUE          /VirtualReadTotalTimeout:VALUE          /MonitorHex          /MonitorAscii          /DelayWriteFileByteToByte:VALUE          /DelayWriteFileConstant:VALUE          /DelayReadFileByteToByte:VALUE          /DelayReadFileConstant:VALUE</p>
<p><b>Parameters</b></p>	<p><b>P1</b> – The physical communications port to use. This device must be an industry standard physical communications device running the standard Windows serial port device driver. Generally these devices are named either COM1 or COM2. Data received at the indicated physical port is reproduced (written) to the all the VIRTUAL port specified.</p> <p><b>V1, V2 ...</b> – One or more Virtual Serial Ports. Data received at these ports are written to the PHYSICAL port.</p>
<p><b>Switches</b></p>	<p><b>/Baud:VALUE</b> – The valid decimal data rate VALUE in bits per second (pbs) for the physical port. Examples of valid data rates are 300, 600, 1200, 2400, 4800, 9600, 19200, 38400, and so on.</p> <p><b>/Parity:{EVEN MARK NO ODD SPACE}</b> - Parity scheme to be used in transmitting data and receiving data on the physical port.</p> <p><b>/RxParity:{ENABLE DISABLE}</b> – Indicates whether parity checking is enabled for data received on the physical port. If enabled, parity checking is performed.</p>

<p><b>Switches (continued)</b></p>	<p><b>/StopBits:{ONE TWO ONEANDHALF}</b> - Number of stop bits to be used for data transmitted and received on the physical port.</p> <p><b>/OutCtsFlow:{ENABLE DISABLE}</b> - Indicates whether the physical port's CTS (clear-to-send) signal is monitored for output flow control. If enabled and CTS is turned off, output is suspended until CTS is sent again.</p> <p><b>/DtrControl:{DISABLE ENABLE HANDSHAKE}</b> – DTR (data-terminal-ready) flow control of the physical port. DISABLE causes the DTR line to disable (de-assert) when the physical device is opened and leaves it disabled. ENABLE causes the DTR line to assert when the physical device is opened and leaves it asserted. HANDSHAKE allows DTR handshaking on the physical device, if handshaking is enabled</p> <p><b>/DsrSensitive:{ENABLE DISABLE}</b> - Indicates whether the physical port's communications driver is sensitive to the state of the DSR signal. If enabled, the physical device's driver ignores any bytes received, unless the DSR modem input line is high (asserted).</p> <p><b>/TxContinueOnXoff:{ENABLE DISABLE}</b> - Indicates whether transmission stops when the input buffer is full and the physical device's driver has transmitted the <b>XoffChar</b> character. If enabled, transmission continues after the input buffer has come within <b>XoffLim</b> bytes of being full and the driver has transmitted the <b>XoffChar</b> character to stop receiving bytes. If not enabled, transmission does not continue until the input buffer is within <b>XonLim</b> bytes of being empty and the driver has transmitted the <b>XonChar</b> character to resume reception.</p> <p><b>/OutX:{ENABLE DISABLE}</b> Indicates whether XON/XOFF flow control is used during transmission on the physical port. If enabled, transmission stops when the <b>XoffChar</b> character is received and starts again when the <b>XonChar</b> character is received</p> <p><b>/InX:{ENABLE DISABLE}</b> - Indicates whether XON/XOFF flow control is used during reception on the physical port. If enabled, the <b>XoffChar</b> character is sent when the input buffer comes within <b>XoffLim</b> bytes of being full, and the <b>XonChar</b> character is sent when the input buffer comes within <b>XonLim</b> bytes of being empty.</p> <p><b>/XonLim:VALUE</b> – On the physical port, the minimum number of bytes allowed in the input buffer before flow control is activated to inhibit the sender. Note that the sender may transmit characters after the flow control signal has been activated, so this value should never be zero. This assumes that XON/XOFF, RTS, or DTR input flow control is specified using <b>/flnX</b>, <b>/RtsControl</b>, or <b>/DtrControl</b>.</p> <p><b>/XoffLim:VALUE</b> – On the physical port, the maximum number of bytes allowed in the input buffer before flow control is activated to allow transmission by the sender. This assumes that XON/XOFF, RTS, or DTR input flow control is specified in <b>/InX</b>, <b>/RtsControl</b>, or <b>/DtrControl</b>. The maximum number of bytes allowed is calculated by subtracting this value from the size, in bytes, of a buffer specific to the physical devices driver.</p>
--	---

<p><b>Switches (continued)</b></p>	<p><b>/XonChar:VALUE</b> - Value of the XON character for both transmission and reception on the physical port.</p> <p><b>/XoffChar:VALUE</b> - Value of the XOFF character for both transmission and reception on the physical port.</p> <p><b>/NullDiscard{ENABLE DISABLE}</b> - Indicates whether null bytes are discarded. If enabled then null bytes are discarded when received on the physical port.</p> <p><b>/RtsControl:{DISABLE ENABLE HANDSHAKE TOGGLE}</b> - RTS (request-to-send) flow control of the physical port. DISABLE causes the RTS line to be de-asserted (disabled) when the device is opened and leaves it de-asserted (disabled). ENABLE asserts (enables) the RTS line when the device is opened and leaves it asserted (on). HANDSHAKE allows RTS handshaking. The physical port's driver asserts (raises) the RTS line when the "type-ahead" (input) buffer is less than one-half full and de-asserts (lowers) the RTS line when the buffer is more than three-quarters full. TOGGLE, which is valid in Windows NT/2000/XP, specifies that the RTS line will be asserted (high) if bytes are available for transmission. After all buffered bytes have been sent, the RTS line will be de-asserted (low).</p> <p><b>/ErrorReplace:{ENABLE DISABLE}</b> – Indicates whether bytes received on the physical port with parity errors are replaced with the character specified by the <b>/ErrorReplaceChar</b> value. If enabled, and the <b>/Parity</b> is enabled, then replacement occurs.</p> <p><b>/ErrorReplaceChar:VALUE</b> – Value of the character used to replace bytes received on the physical port, with a parity error.</p> <p><b>/ByteSize:VALUE</b> - Number of bits in the bytes transmitted and received on the physical port.</p> <p><b>/ReadIntervalTimeout:VALUE</b> - Maximum time, in milliseconds, allowed to elapse between the arrival of two characters on the physical port communications line. During a WIN32 <b>ReadFile</b> operation, the time period begins when the first character is received. If the interval between the arrivals of any two characters exceeds this amount, the WIN32 <b>ReadFile</b> operation is completed and any buffered data is returned. A value of zero indicates that interval time-outs are not used.</p> <p><b>/VirtualReadIntervalTimeout:VALUE</b> - Maximum time, in milliseconds, allowed to elapse between the arrival of two characters on the virtual serial port. During a WIN32 <b>ReadFile</b> operation, the time period begins when the first character is received. If the interval between the arrivals of any two characters exceeds this amount, the WIN32 <b>ReadFile</b> operation is completed and any buffered data is returned. A value of zero indicates that interval time-outs are not used.</p> <p><b>/VirtualReadTotalTimeout:VALUE</b> VALUE, in milliseconds, used to calculate the total time-out period for read operations on the Virtual Serial</p>
--	---

<p><b>Switches (continued)</b></p>	<p>Port.</p> <p><b>/MonitorHex</b> – prints a data trace on the console in Hex.</p> <p><b>/MonitorAscii</b> – prints a data trace on the console in ASCII.</p> <p><b>/DelayWriteFileByteToByte:VALUE</b> - Specifies the amount of time, in milliseconds, to delay between bytes when writing to the Virtual Serial Port(s)</p> <p><b>/DelayWriteFileConstant:VALUE</b> - Specifies a constant amount of time, in milliseconds, to delay between bytes when writing to the Virtual Serial Port(s)</p> <p><b>/DelayReadFileByteToByte:VALUE</b> - Specifies the amount of time, in milliseconds, to delay between bytes when reading to the Virtual Serial Port(s)</p> <p><b>/DelayReadFileConstant:VALUE</b> - Specifies a constant amount of time, in milliseconds, to delay between bytes when reading to the Virtual Serial Port(s)</p> <p>***** <b>IMPORTANT</b> *****</p> <p><b>NOTE: If file timing is specified, it applies to each and every Virtual Serial Port specified on the command line.</b></p>
<p><b>Runtime</b></p>	<p>'Q' then 'ENTER', on the keyboard causes the utility to exit.</p>

The utility (“MultiVirtualToPhysical.exe”) is found in the “exe” directory of the installation files. This utility can be run from either the command line, or from

standard Windows shortcuts. Consider the following command line capture:



```
Command Prompt
C:\VirtualSerial\exe>MultiVirtualToPhysical /?
-----
MultiVirtualToPhysical (Events)      Utility v2.27, Rev -
(c) 2003 Constellation Data Syses, Inc. (CDS)
All Rights Reserved, Restricted USE, contact CDS for licensing.
www.VirtualPeripherals.com
-----

Usage: VirtualToPhysical  PHYSICAL  VIRTUAL

    PHYSICAL - the physical port (COM1 or COM2)
    VIRTUAL  - the virtual serial port name

[switches] - optional settings of the physical port unless otherwise noted
              ('Virtual' nomenclature). All VALUE's are in decimal.

    /Baud:VALUE
    /Parity:{EVEN|MARK|NO|ODD|SPACE}
    /RxParity:{ENABLE|DISABLE}
    /StopBits:{ONE|TWO|ONEANDHALF}
    /OutCtsFlow:{ENABLE|DISABLE}
    /DtrControl:{DISABLE|ENABLE|HANDSHAKE}
    /DsrSensitive:{ENABLE|DISABLE}
    /TxContinueOnXoff:{ENABLE|DISABLE}
    /OutX:{ENABLE|DISABLE}
    /InX:{ENABLE|DISABLE}
    /XonLim:VALUE
    /XoffLim:VALUE
    /XonChar:VALUE
    /XoffChar:VALUE
    /NullDiscard{ENABLE|DISABLE}
    /RtsControl:{DISABLE|ENABLE|HANDSHAKE|TOGGLE}
    /ErrorReplace:{ENABLE|DISABLE}
    /ErrorReplaceChar:VALUE
    /ByteSize:VALUE
    /ReadIntervalTimeout:VALUE
    /VirtualReadIntervalTimeout:VALUE
    /VirtualReadTotalTimeout:VALUE
    /MonitorHex
    /MonitorAscii
    /DelayWriteFileByteToByte:VALUE
    /DelayWriteFileConstant:VALUE
    /DelayReadFileByteToByte:VALUE
    /DelayReadFileConstant:VALUE

C:\VirtualSerial\exe>
```

In this example, utility has been run from the command line no parameters. This causes the utility to display its command line usage. Note that the utility has a version number which should match that of the underlying VSP drivers being used.

<b><u>Important:</u></b>	<p>The physical com port is setup by default to use a data rate of 2400bps, and 8 data bits, 1 stop and 1 start bit, and no parity. Handshaking is XON / XOFF. These settings may be overridden with <u>careful</u> use of the command line switches.</p> <p>The data rate and device settings on the VSP side are not relevant since the VSP transmits and receives data without regard to the device settings (baud/data bits, etc).</p>
--------------------------	--

### 5.4.3 Demonstration Using *Multi Virtual To Virtual*

A simple demonstration may be accomplished by using *Hyperterminal* (standard Windows Accessory) to connect several virtual ports to a single physical port. Use a procedure similar to the one outlined in section 5.3.3, *Demonstration Using the Virtual To Physical Utility*.

## 5.5 Add Port Utility

The *Add Port* ("addport.exe") utility creates a Virtual Serial Port. The VSP name will be the same name that applications, such as HyperTerminal use to identify the VSP device. Avoid already defined port names, or common port names such as COM1 or COM2.

If a port with the selected name already exists, a message will appear which explains why the Virtual Serial Port could not be created.

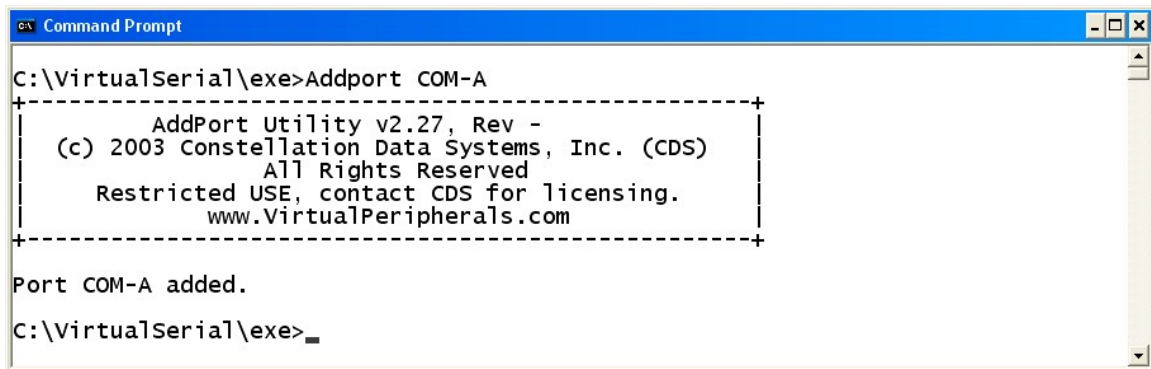
### 5.5.1 Command Line Parameters of *Add Port*

Consider the following syntax mapping:

<b>Syntax</b>	<b>AddPort</b> PortA
<b>Parameters</b>	PortA – Name of the Virtual Serial Port that is to be created.
<b>Switches</b>	<i>/?</i> – Prints command line usage.

### 5.5.2 Demonstration Using *Add Port*

In the following example, the *Add Port* utility has been used to create/add a Virtual Serial Port "PORTA".

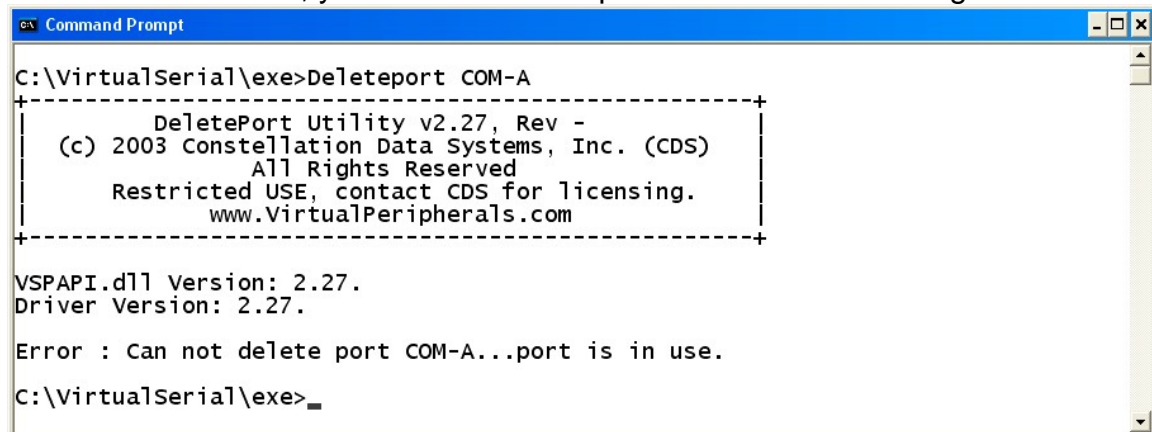


```
Command Prompt
C:\VirtualSerial\exe>Addport COM-A
+-----+
|           AddPort Utility v2.27, Rev -           |
| (c) 2003 Constellation Data Systems, Inc. (CDS) |
|           All Rights Reserved                   |
|           Restricted USE, contact CDS for        |
|           licensing.                             |
|           www.VirtualPeripherals.com            |
+-----+
Port COM-A added.
C:\VirtualSerial\exe>
```

## 5.6 Delete Port Utility

The *Delete Port* (“deleteport.exe”) utility deletes a Virtual Serial Port. The VSP name will be the same name that applications, such as HyperTerminal, use to identify the VSP device.

If a port with the selected name is currently in use, a message will appear which explains why the Virtual Serial Port could not be deleted. Should a virtual serial port device be locked by another component then the utility may fail. Should the device be locked, you will observe output similar to the following:



```
Command Prompt
C:\VirtualSerial\exe>Deleteport COM-A
+-----+
|           DeletePort Utility v2.27, Rev -           |
| (c) 2003 Constellation Data Systems, Inc. (CDS) |
|           All Rights Reserved                   |
|           Restricted USE, contact CDS for        |
|           licensing.                             |
|           www.VirtualPeripherals.com            |
+-----+
VSPAPI.dll Version: 2.27.
Driver Version: 2.27.
Error : Can not delete port COM-A...port is in use.
C:\VirtualSerial\exe>
```

To clear this condition, free the device, in this case by simply removing the connection to the Virtual Serial Port.

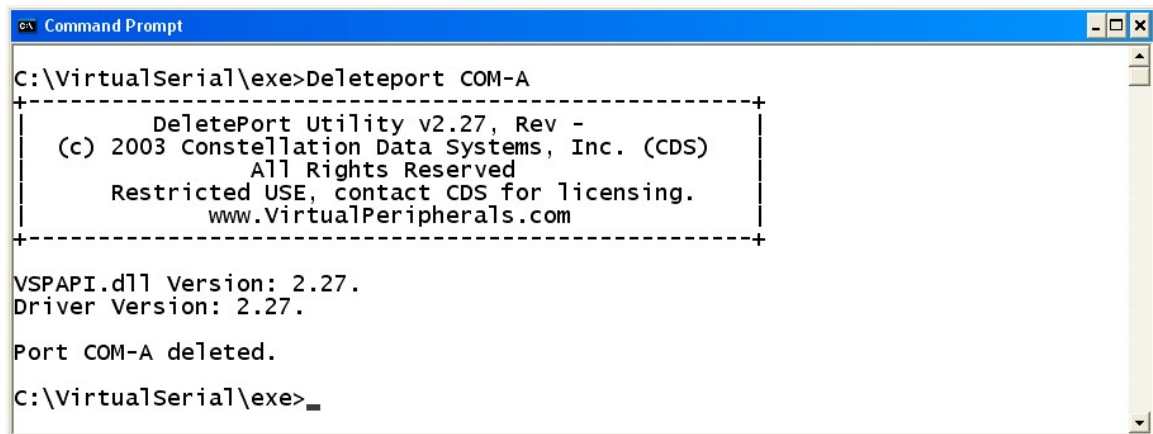
### 5.6.1 Command Line Parameters of *Delete Port*

Consider the following syntax mapping:

<b>Syntax</b>	<b>DeletePort</b> PortA
<b>Parameters</b>	PortA – Name of the Virtual Serial Port that is to be deleted.
<b>Switches</b>	<i>/?</i> – Prints command line usage.

### 5.6.2 Demonstration Using *Delete Port*

In the following example, the *Delete Port* utility has been used to delete a Virtual Serial Port “PORTA”.



```
Command Prompt
C:\VirtualSerial\exe>Deleteport COM-A
+-----+
DeletePort Utility v2.27, Rev -
(c) 2003 Constellation Data Systems, Inc. (CDS)
All Rights Reserved
Restricted USE, contact CDS for licensing.
www.VirtualPeripherals.com
+-----+
VSPAPI.dll Version: 2.27.
Driver Version: 2.27.
Port COM-A deleted.
C:\VirtualSerial\exe>
```

## 5.7 Enum Ports Utility

The *Enum Ports* (“enumports.exe”) utility enumerates all Virtual and Physical Serial Ports on a system.

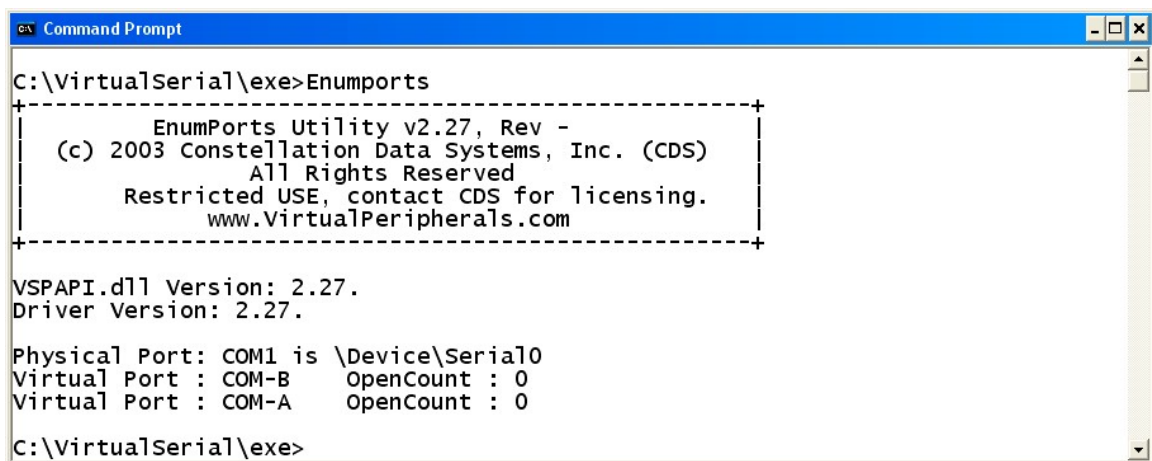
### 5.7.1 Command Line Parameters of *Enum Ports*

Consider the following syntax mapping:

<b>Syntax</b>	<b>EnumPorts</b>
<b>Parameters</b>	This utility does not take any parameters.
<b>Switches</b>	This utility does not have any options available.

### 5.7.2 Demonstration Using *Enum Ports*

In the following example, the *Enum Ports* utility has been used to enumerate all Virtual and Physical Serial Ports.



```
Command Prompt
C:\VirtualSerial\exe>Enumports
-----
EnumPorts Utility v2.27, Rev -
(c) 2003 Constellation Data Systems, Inc. (CDS)
All Rights Reserved
Restricted USE, contact CDS for licensing.
www.VirtualPeripherals.com
-----
VSPAPI.dll Version: 2.27.
Driver Version: 2.27.

Physical Port: COM1 is \Device\Serial0
Virtual Port : COM-B   OpenCount : 0
Virtual Port : COM-A   OpenCount : 0

C:\VirtualSerial\exe>
```

## 5.8 Serial Number Entry Utility

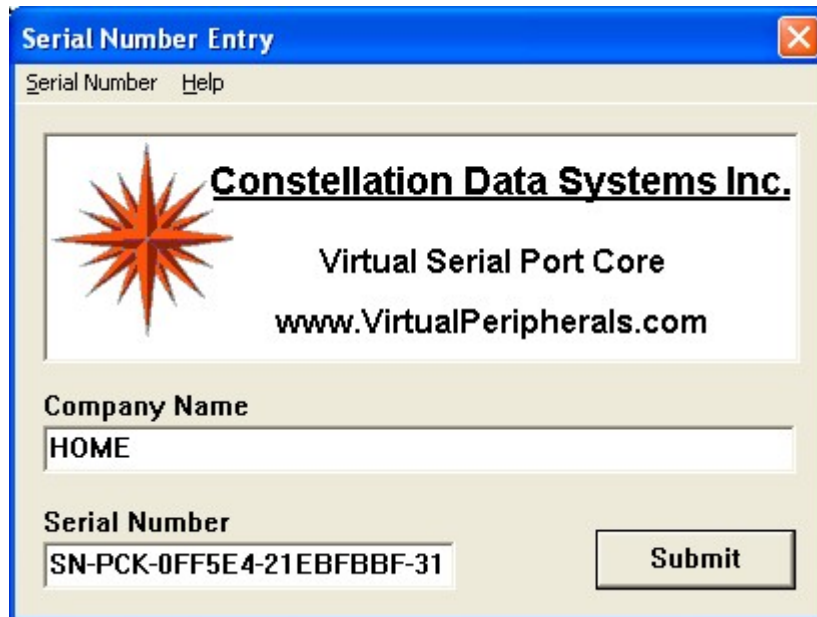
The Serial Number Entry utility (“SerialNumEntry.exe”) gives the user the ability to register VSP software without having to re-run setup and going through unnecessary steps. The target machine does not need to be rebooted after successfully entering a serial number with SerialNumEntry.exe; the VSP software can be used immediately.

### 5.8.1 Command Line Parameters of Serial Number Entry Utility

The Serial Number Entry utility is GUI (Graphical User Interface) based, so there are no command line parameters to deal with. Simply run the application, and enter the pertinent information in the appropriate boxes.

### 5.8.2 Demonstration Using Serial Number Entry Utility

In the following example, the Serial Number Entry Utility is being used to register a company to use VSP software.

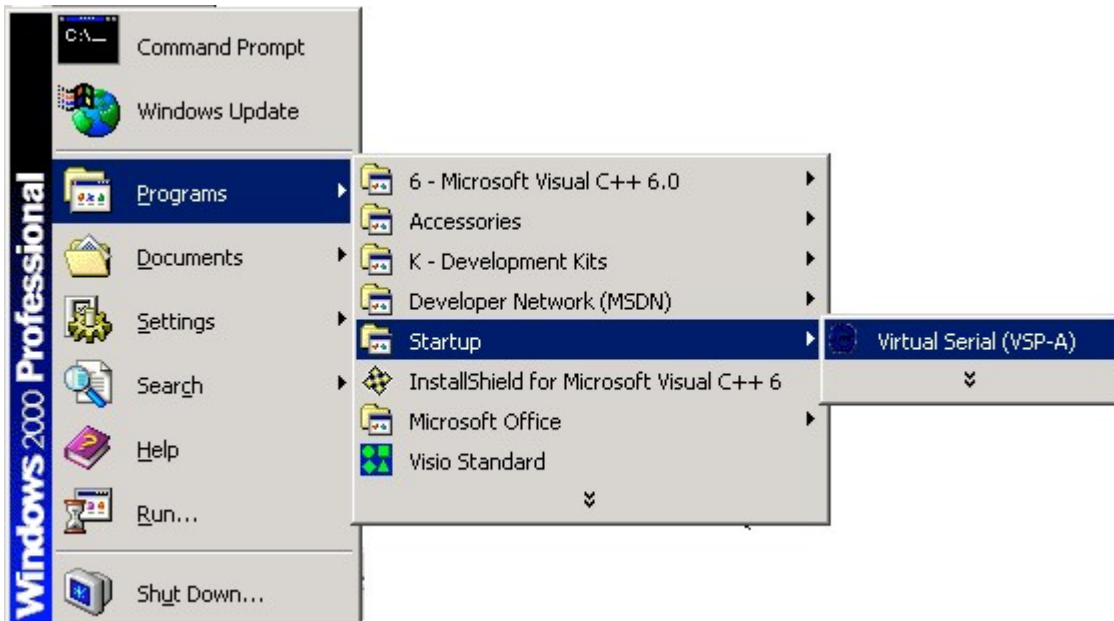


After successfully submitting the designated Serial Number, it is necessary to reboot the target system. This allows the newly entered Serial Number to be correctly configured/registered with the software.

## 6. Detailed Installation Verification Procedures

### 6.1 Verification of Port Names and Version Information

This information may be easily gathered by running the *VSP Startup*. From the start button, simply select Programs, Startup, and then selecting a Virtual Serial Port from the menu (see below);

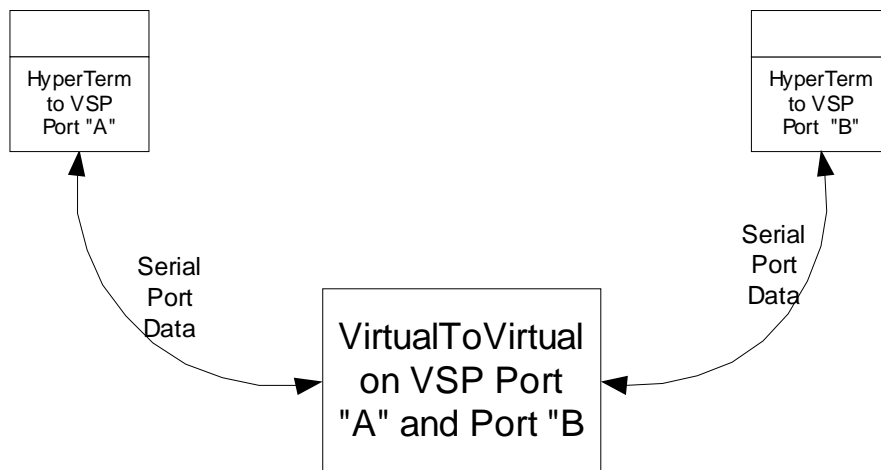


The *VSP Startup* is then displayed for 10 seconds. See section 2.2, *Quick Installation Verification* for more information.

## 6.2 Installation Verification using VirtualToVirtual

Installation verification can be done by connecting two Virtual Serial Ports, through the VirtualToVirtual Utility. These two ports can then be connected to individual Hyperterminal sessions. Hyperterminal is a standard windows communications accessory. Consider the following Data Flow Diagram:

## Hyperterminal <=> Hyperterminal Connection using VirtualToVirtual Utility



Data typed on the console of one hyper terminal session should appear on the console of the other Hyperterminal session. Additionally, file transfers may be performed using protocols such as YMODEM, ZMODEM etc.

Tip	Some versions of Hyperterminal have an issue where changes to COM port selection or data format, <b>do not</b> take effect until those settings are saved, and then Hyperterminal is reloaded using those settings. It is suggested that Hyperterminal setting changes always be saved to disk, and then have Hyperterminal restarted.
-----	--

## 7. Notices

Use of this software, information, or technology in a system, or as a component of a system, which can through action or inaction, cause damage to life, limb, property, or the environment is not authorized. Use of this software is also subject to the terms and conditions of the Software License Agreement with CDS that you accepted at time of installation.

This manual, information, technology and software is protected by copyright law and international treaties. Unauthorized reproduction or distribution may result in severe civil and criminal penalties, and will be prosecuted to the maximum extent.

## 8. Index of Acronyms and Abbreviations

API	Applications Programming Interface
BPS	Bits per Second (“baud”)
CDS	Constellation Data Systems, Inc.
CTS	Clear To Send (modem status line)
DOS	Disk Operating System
DLL	Dynamic Link Library
DSR	Data Set Ready (modem status line)
DTR	Data Terminal Ready (modem control line)
GPS	Global Positioning System
HyperTerminal	Standard Windows Communications Application
MS	Microsoft
MSDN	MS Developers Network
PCR	Physical Communications Resource (Such as a UART)
RD	Receive Data line
RI	Ring Indicate (modem status line)
RS-232	Recommended Standard 232 for data communications of the Electronics Industry Association
RTS	Request To Send (modem control line)
RX	Receive
SDK	Software Development Kit
TD	Transmit Data line
TLA	Three Letter Acronym
TX	Transmit
UART	Universal Asynchronous Receiver / Transmitter (Hardware)
VSP	Virtual Serial Port
VSPAPI	Virtual Serial Port Applications Programming Interface
WIN16	Windows 16 Bit Programming Paradigm (Arguably Obsolete)
WIN32	Windows 32 Bit Programming Paradigm
XON	Transmit On
XOFF	Transmit Off