License Agreement:

TAL Technologies, Inc.
2027 Wallace Street
Philadelphia, PA 19130 U.S.A.
Tel: (215)-763-5096 Fax: (215)-763-9711
email: tal1@taltech.com
Web: http://www.taltech.com

What is the Software Wedge?

The Software Wedge is an extremely powerful utility that adds serial communications capabilities to any DOS application program. The main function of the Software Wedge is to allow you to connect any serial device (measuring instrument, electronic scale, bar code reader, PLC, PDT, etc.) to a PC and be able to read or write data to or from the device directly from within any DOS application. The Software Wedge eliminates the need for special serial device drivers or custom application programs. The Wedge is best described as a user customizable serial device driver that transfers serial data to your foreground applications via the keyboard buffer. Thus, it causes your PC to behave as if data from your serial device were being typed in on your keyboard.

The Software Wedge transfers serial data to *any* DOS application by sending data as keystrokes through the DOS keyboard buffer. You can even set up the "Wedge" to issue additional keystrokes or commands along with serial data, and force the receiving application to perform actions after each input from your serial device. For example you could issue additional keystrokes that would cause an application to update a graph or chart in real time or perform calculations on data received through a serial port. To send data out the serial port, you can use one of the special utility programs included with the Software Wedge to transmit character strings or you can define special hot key activated output strings or even automatic - timer controlled output strings.

The Software Wedge offers a number of features that allow you to manipulate or pre-process incoming serial data before sending it to an application program. These include the ability to parse and filter data received through the serial port as well as add additional keystrokes or data and/or date and time stamps to the incoming data. Two translation tables allow you to translate incoming characters to other characters or even to specific PC keystrokes.

All of these features make it possible to create extremely sophisticated serial device interfaces for any DOS application program.

The Professional Edition of Software Wedge can also be used to capture serial data directly to a disk file in the background, while you work with other programs in the foreground! A program called **File Wedge** is also provided that allows you to convert data from a disk file to keystrokes. Thus you could capture data to disk with the Software Wedge and then play the data back to an application later using File Wedge. You could also use File Wedge to force data from virtually *any* disk file into *any* DOS application.

The best way to understand the complete capabilities of the Software Wedge is to read this manual thoroughly.

Before You Begin

We at TAL Technologies would like to take this opportunity to thank you for purchasing the Software Wedge. We sincerely hope that it proves to be the perfect solution for all your serial I/O applications. We also hope that you find this product both easy and enjoyable to use. If there is a particular feature that you would like to see in a future version or perhaps a change in either the software or the documentation that would make this software easier to use, please contact us. Your feedback will be greatly appreciated.

We would also like to take this opportunity to remind you to read the copyright notice and license agreement on page two of this manual thoroughly. You have purchased a license to use this software in a single PC and thus you may not use it on more than one PC at a time without purchasing additional licenses. Please help us to protect your investment by adhering to the terms of this agreement and by reporting copyright violations directly to:
TAL Technologies, Inc.
Tel: 800-722-6004 or 215-763-5096
Fax: 215-763-9711
Email: tal1@taltech.com
Web: http://www.taltech.com

Again, thank you for your support.

System Requirements.

The Software Wedge Professional Edition will run on any IBM or compatible PC running MS or PC DOS version 2.0 or above. A mouse and a hard disk with approximately 800kb free disk space are highly recommended although not entirely necessary.

Devices Compatible with the Software Wedge

The Software Wedge was designed primarily as an interface to most commonly used serial data collection devices including Electronic Scales, Bar Code Readers, Magnetic Stripe Readers, Electronic Measuring Instruments, Medical, Industrial and Laboratory Instruments, PLCs, etc.. Any type of serial device whose output data is transmitted in a somewhat well-defined format is a good candidate for use with the Software Wedge. The Professional Edition is designed to support even the most sophisticated serial instruments.

What's New In Version 5.0 - Professional Edition

Version 5.0 of The Software Wedge adds several new capabilities to previous versions including: A completely redesigned and much easier to use configuration program, support for the latest generation of PC serial adapters (including full support for 16550 UARTs), a Pre-Input Character Translation Table, a Disk File Logging Mode, additional data parsing and filtering capabilities, a Timed Automatic Serial Output Function and up to 26 user definable hot keys that invoke any of 12 different "actions". A powerful set of function calls allow advanced users and programmers to create custom applications using the Software Wedge as a serial I/O driver. All new features were designed to support a wider range of serial data generating instruments and I/O data formats.

The Professional version of the Software Wedge also comes complete with a new TSR utility program called "File Wedge" that allows you to input data from a disk file directly into any DOS application by porting the data through the keyboard buffer. File Wedge is almost identical to the Software Wedge except that it takes its input from a disk file instead of from a serial port. This powerful program makes it possible to read data from almost any source directly into any DOS application program! Because the Software Wedge Professional can log data directly to a disk file in the background, you can use File Wedge to read previously logged data into an application whenever convenient.

Why Is It Called the *Software Wedge*™ ?

A *Wedge* (or *keyboard wedge*) is normally a hardware device that connects between a computer and it's keyboard. The purpose of a wedge is to provide a way to input data from a serial device into a PC as if it was being entered via the keyboard.

Before the development of the *Software Wedge*, hardware wedges were the only available solution if you needed to input serial data directly into an off the shelf program like Lotus 123 or dBase. Unfortunately, hardware wedges are severely limited in capability, are painfully slow, and they cannot support full two way serial I/O. They are also expensive, prone to failure and can be difficult or impossible to install; especially on a Laptop that does not have a keyboard port.

The Software Wedge is a highly advanced *software only* alternative to a keyboard wedge that allows you to connect a serial device directly to your PC's serial port with no need for any additional hardware. Thus the name: *Software Wedge*™

Getting Started

Installing the Software Wedge on a Hard Disk

The Software Wedge has an installation program called "INSTALL.EXE" that will copy all Software Wedge program files to a directory on your hard disk that you specify.

To install the Software Wedge, place the Software Wedge distribution diskette into drive A: (or B:) then at the DOS prompt, type the command "A:INSTALL". The INSTALL program takes approximately two minutes to copy all files.

You may also install the Software Wedge by copying all files from the Software Wedge distribution diskette(s) to your hard drive using the DOS COPY command. We recommend creating a directory called "WEDGE" and copying all files to this directory.

Parts of the Software Wedge Package

The Software Wedge consists of two main programs: SWCONFIG.EXE and WEDGE.COM. Also included with the Software Wedge are several utility programs that provide special functions for use along with or instead of the Wedge. For more information on all of the utility programs that are provided with the Software Wedge, please refer to the Utilities section of this manual.

WEDGE.COM is the actual Wedge program that does all the serial I/O work. WEDGE.COM is a TSR (Terminate and Stay Resident) program which means that it must be installed into memory as an extension to the operating system. Similar to a device driver, WEDGE.COM sits in the background and does its job while you run other programs in the foreground on your PC.

Before you can install the Wedge, you must first create a configuration file for it. SWCONFIG.EXE is mainly a front end for the Wedge that is used to create configuration files that are read in by WEDGE.COM when it is loaded into memory. SWCONFIG.EXE allows you to customize the operation of the Wedge for your particular serial device and application and then store your configurations to a disk file that can be retrieved later.

See Also: Loading Configuration Files With Command Line Arguments

SWCONFIG.EXE and Software Wedge Configuration Files

The Software Wedge Configuration program SWCONFIG.EXE allows you to create and edit configuration files. When you configure the Software Wedge for use with a particular serial device and a specific application program, you can save your configuration to a disk file that can be reloaded later. On the main menu of the SWCONFIG program there is a "FILE" option that allows you to save and load configuration files.

Loading Configuration Files With Command Line Arguments

The SWCONFIG program supports one optional command line argument using the following syntax:

SWCONFIG *filename*

If a filename is specified on the command line, it must be the name of a valid Software Wedge configuration file. This filename must be complete with the filename extension and also the drive and directory path for the file if it is not in the current directory. If a filename is supplied on the command line, it causes SWCONFIG to pre-load the specified configuration.

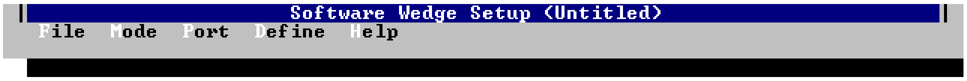
How To Obtain Technical Support

If you experience difficulty configuring the Software Wedge after you have read this manual thoroughly, please call TAL Technologies, Inc. at (215)-763-5096 for technical support or visit our web site at http://www.taltech.com. Only registered users of the Software Wedge are eligible for technical support. Please have your original Software Wedge diskette(s) and this users manual available when calling for support. See Also: Troubleshooting

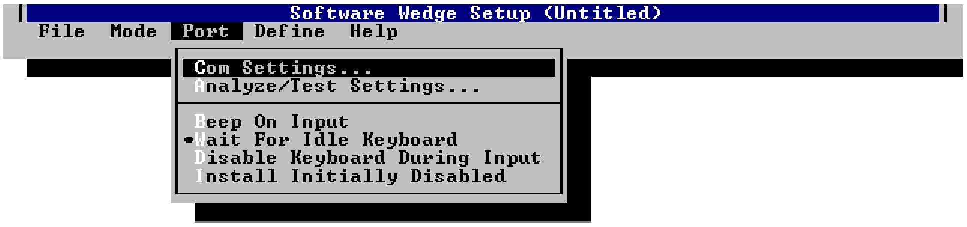
If you have not already done so, please register your copy of the Software Wedge immediately to ensure your eligibility for technical support and upgrade notification. You can also register on line at http:/www.taltech.com in the support section of the web site.

Navigating Menus & Dialog Boxes in SWCONFIG.EXE

When you run SWCONFIG.EXE, you are presented with a main menu as shown below. The main menu has five sub-menus some of which provide access to additional dialog boxes or other options. To select main menu items you may either click on the desired item with a mouse or you may press the ALT key and the first letter of the desired item.



For example, to select the Port menu, you would press the ALT key followed by the letter "P". The Port sub-menu will drop down as shown below:

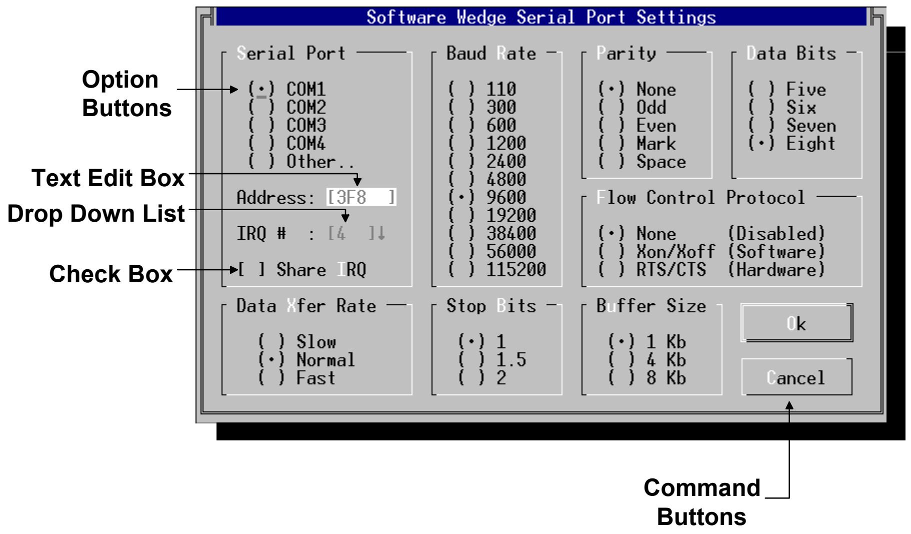

Software Wedge Setup (Untitled)
File Mode Port Define Help
Com Settings...
Analyze/Test Settings...
Beep On Input
•Wait For Idle Keyboard
Disable Keyboard During Input
Install Initially Disabled

Sub menu items are selected by either clicking on the desired item with a mouse or by pressing the key corresponding to the hi-lighted character in the desired sub menu item's caption. You may also select sub menu items using the up/down arrow keys and then pressing the ENTER key when a desired item is hi-lighted. If you make a mistake, press the ESCAPE key to back out of a selection.

Note: Sub menu items that have three periods following their caption cause a dialog box to appear. All other items either perform an immediate action or are toggle options. Toggle options are toggled ON and OFF by selecting them. When a sub menu toggle option is "ON", a dot or check mark will appear to the left of the option in the menu.

Dialog Boxes

Dialog boxes in SWCONFIG.EXE may contain one or more types of "Controls". Controls include Text Input Boxes, Command Buttons, Check Boxes, Option Buttons, List Boxes, and Drop Down Lists.



```
                     Software Wedge Serial Port Settings
 Serial Port        Baud Rate     Parity          Data Bits
 (•) COM1           ( ) 110       (•) None        ( ) Five
 ( ) COM2           ( ) 300       ( ) Odd         ( ) Six
 ( ) COM3           ( ) 600       ( ) Even        ( ) Seven
 ( ) COM4           ( ) 1200      ( ) Mark        (•) Eight
 ( ) Other..        ( ) 2400      ( ) Space
                    ( ) 4800
 Address: [3F8  ]   (•) 9600      Flow Control Protocol
                    ( ) 19200
 IRQ #  : [4   ]↓   ( ) 38400     (•) None     (Disabled)
                    ( ) 56000     ( ) Xon/Xoff (Software)
 [ ] Share IRQ      ( ) 115200    ( ) RTS/CTS  (Hardware)

 Data Xfer Rate     Stop Bits     Buffer Size
                                                   Ok
   ( ) Slow           (•) 1         (•) 1 Kb
   (•) Normal         ( ) 1.5       ( ) 4 Kb
   ( ) Fast           ( ) 2         ( ) 8 Kb       Cancel
```

To navigate between controls in a dialog box, you may either click your mouse on a desired control or you may use the TAB key to switch the input focus from one control to the next. If the caption for a control has a single hi-lighted character, you may select the control by holding the ALT key down while pressing the hi-lighted character. If a control appears grayed out, it is not available for selection.

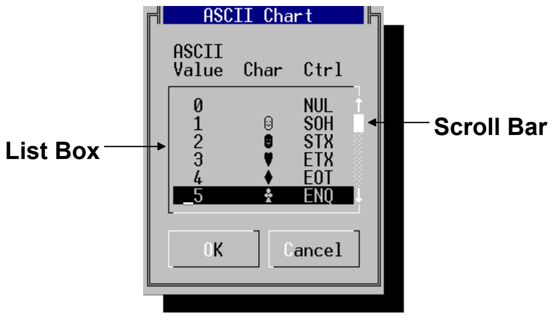
Option Buttons are typically grouped together to allow selection of a single option in the group. To select an option in a group, either click your mouse on the desired option or press the TAB key until the option group has the focus and then use the up/down arrow keys until the desired option is selected.

To edit text in a Text Edit Box, you typically use the cursor control keys (backspace, right & left arrows, delete, etc.) to edit typed in data.

Drop Down Lists and List Boxes allow selection of a single item in a list. To select from a Drop Down List, either click your mouse on the arrow to the right of the list or hold the ALT key down while pressing the DOWN arrow key. This will cause the list to drop down similar to a normal List Box or sub menu so that you can use your mouse or arrow keys to select a particular item in the list. If a list contains a large number of items, a Scroll Bar will appear to the right of the list so that you can use your mouse to scroll up and down through the list. If a scroll bar appears in a list box then you may also use your cursor control keys (Home, End, PgUp, PgDn and Up/Down Arrows) to scroll through the list.

Check Boxes allow an option to be toggled On or Off as indicated by a check mark in square brackets. To toggle a check box on or off, either click on the check box with your mouse or use the TAB key to switch the focus to the check box and press the space bar to toggle the option.

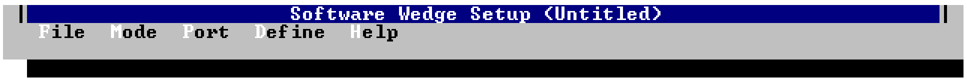
Command Buttons may be selected either by clicking with your mouse or by using the TAB key to switch the focus to the button and pressing the space bar.



ASCII Value	Char	Ctrl
0		NUL
1	☺	SOH
2	☻	STX
3	♥	ETX
4	♦	EOT
5	♣	ENQ

The Main Menu

When you run SWCONFIG you are presented with the following "Main Menu":



The title bar of the main menu indicates that you are in setup mode and it also displays in parentheses the filename of any currently selected configuration file.

The five main menu options and their functions are :

File
Allows opening and saving of configuration files and installation of the Software Wedge. See Also: The File Menu

Mode
Allows selection of the method used to transfer serial input data from the Software Wedge to another application or disk file & optionally specify a disk file that is to receive the data. See Also: The Mode Menu

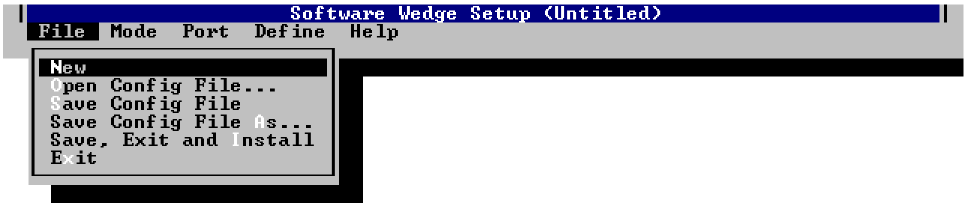
Port
Allows selection and testing of serial port communications parameters and several other options that affect the operation of the Wedge while it is running. See Also: The Port Menu

Define
The define menu contains selections that allow you to customize the operation of the Software Wedge. The different sub-menus allow the definition of the input data record structure, parsing instructions & filters to be applied to incoming data, keystroke macros or additional data that is to be added to the serial input data, translations of incoming characters, definition of serial output data strings, hot keys and hot key actions. See Also: The Define Menu

Help
Provides on-line help for the Software Wedge. See Also: Accessing On-Line Help

The File Menu

Selecting "File" from the Software Wedge main menu presents the sub-menu shown below:



File sub-menu options and their functions are as follows:

New
Unloads any currently loaded configuration file and resets SWCONFIG to its default configuration.

Open Config File...
Opens a dialog box that allows selection of a previously saved Software Wedge configuration file.

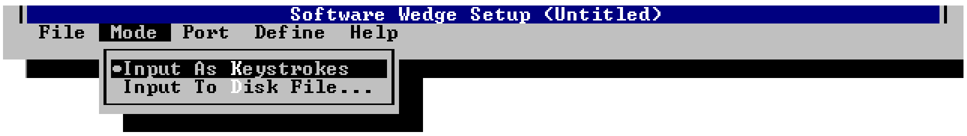
Save Config File
Saves changes to the currently open Software Wedge configuration file.

Save Config File As...
Opens a dialog box that allows you to save the current configuration with a filename that you specify.

Save, Exit and Install
Saves changes to the currently open Software Wedge configuration file, exits from the SWCONFIG program and installs the Software Wedge using the current configuration parameters. See Also: Loading WEDGE.COM Into Memory

Exit
Exits the SWCONFIG program without saving the current configuration and without installing the Software Wedge.

The Mode Menu



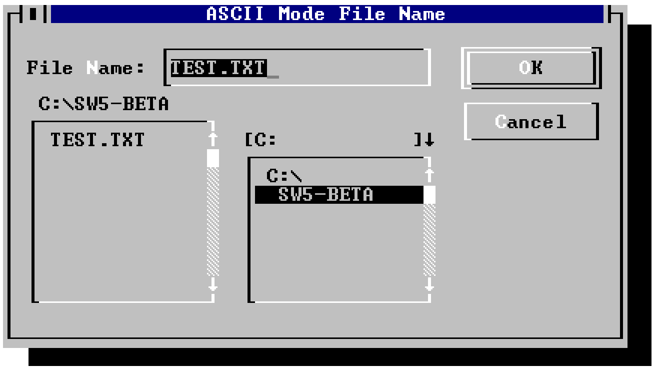
The Software Wedge can operate in one of two modes that specify how to transfer data from your serial device into your PC. The Software Wedge can either transfer incoming serial data to an application by converting incoming serial data to keystrokes and then sending these keystrokes directly to the DOS keyboard buffer or it can store incoming serial data to a disk file.

Input As Keystrokes Mode

In "Input As Keystrokes" mode, the Software Wedge will convert serial input data to keystrokes and then send these keystrokes directly to the DOS keyboard buffer. When operating in "Input As Keystrokes" mode, the Wedge essentially causes your PC to act as if it had a second keyboard attached to it with data from your serial device appearing as if it were being typed in directly on your keyboard.

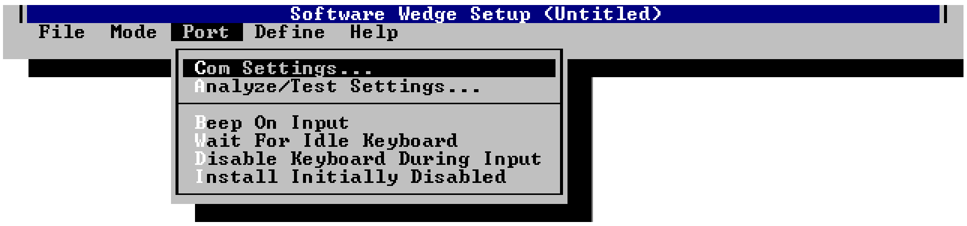
Input To Disk File Mode

When configured in "Input To Disk File" mode, incoming serial input data is written to a disk file instead of being sent to the DOS keyboard buffer. If you select "Input To Disk File" from the Mode menu, the dialog box shown below will appear where you may specify the disk file that is to receive serial input data. When specifying a file name, you should supply the complete path name including the drive and directory for the file where data is to be stored.



Input To Disk File mode is useful in situations where you would like to capture serial data directly to a disk file for later use instead of transferring the data directly to an application program. After capturing data to a disk file, you could then read the data into another program at a later time using either the FILE WEDGE utility provided with the Software Wedge or by using any file import facilities that the program that you want to receive the data offers. Because the Software Wedge provides a wide array of parsing and filtering functions, you could use the Software Wedge to pre-format serial data so that it will be in a format that is most easily imported into your particular application.

The Port Menu


Software Wedge Setup (Untitled)
File Mode Port Define Help
Com Settings...
Analyze/Test Settings...
Beep On Input
Wait For Idle Keyboard
Disable Keyboard During Input
Install Initially Disabled

The six PORT menu options and their functions are:

Com Settings

Displays a dialog box that allows you to select all serial communications parameters for your particular serial device, (i.e. port, baud rate, parity, flow control, buffer size, etc.)
See Also: The Port Settings Dialog Box

Analyze

The Analyze option allows you to test all serial communications parameters and also view or analyze the structure and contents of data from your serial input device. The Analyze feature is extremely useful for testing the communications link between your PC and your serial device as well as for determining the structure of all data received from the device.
See Also: The Port Analyze Option

Beep On Input

This option is a switch that instructs the Software Wedge to beep your PC's speaker when data has been received and is about to be transferred to the keyboard buffer or a disk file. Selecting this option toggles this function on or off where "on" is indicated by a check mark to the left of this menu selection.

Wait For Idle Keyboard

The "Wait For Idle Keyboard" option is a switch that instructs the Software Wedge to wait until the keyboard is idle for at least two seconds before stuffing any data into the keyboard buffer. This can help to insure that data from the serial port does not interfere with data that you are typing on your keyboard.

Disable Keyboard During Input

The "Disable Keyboard During Input" option is a switch that instructs the Software Wedge to disable the keyboard while it is stuffing data into the keyboard buffer. Similar to the "Wait For Idle Keyboard" option, this helps to insure that data from the keyboard does not interfere with data that is being entered by the Wedge. Note: While the keyboard is disabled, the Software Wedge will still recognize any hot keys that you define.
See Also: Defining Hot Keys

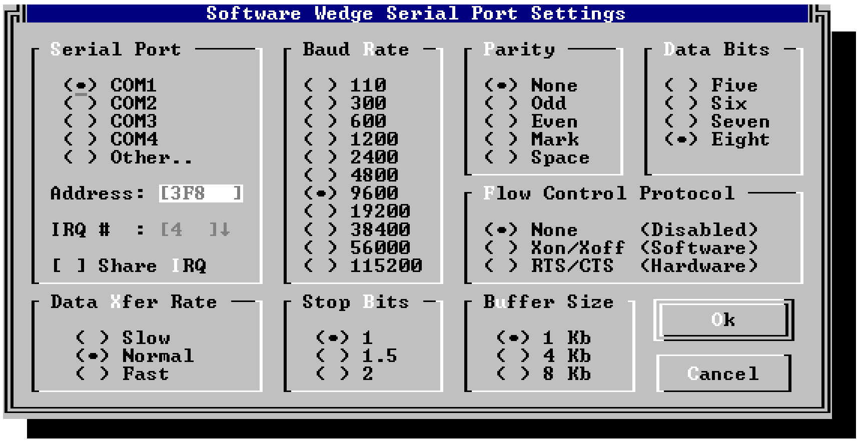
Install Initially Disabled

The "Install Initially Disabled" option causes the Wedge to be disabled when it is initially loaded into memory. SWCONFIG also allows you to define Hot Keys that can be used to Enable and Disable the Wedge after it is loaded into memory. If you install the Wedge initially disabled, you should also define a hot key that can be used to enable it later on. This feature can be useful for controlling when and where to input serial data.
Note: The Software Wedge will still accept serial input while it is disabled so that you do not lose any data. The data is stored in a buffer until the Wedge is enabled by pressing your hot key. See Also: Defining Hot Keys

The Port Settings Dialog Box

The Port Settings dialog box allows you to select all serial parameters for your particular serial device including the Serial Port, Baud Rate, Parity, Number of Data Bits, Stop Bits, the Flow Control Protocol to use, the Input Buffer Size and the rate at which the Software Wedge transfers data from the serial buffer to the keyboard buffer.

Note: All communications parameters selected in the Port Settings dialog box must exactly match the settings for the serial device that you will be using. If you do not know the correct settings for your serial device, then you should contact its manufacturer for this information. Please do not call for technical support if you do not know the correct parameters for your particular device. See Also: Troubleshooting


```
Software Wedge Serial Port Settings

Serial Port            Baud Rate     Parity       Data Bits
 (•) COM1              ( ) 110       (•) None     ( ) Five
 ( ) COM2              ( ) 300       ( ) Odd      ( ) Six
 ( ) COM3              ( ) 600       ( ) Even     ( ) Seven
 ( ) COM4              ( ) 1200      ( ) Mark     (•) Eight
 ( ) Other..           ( ) 2400      ( ) Space
                       ( ) 4800
Address: [3F8   ]      (•) 9600      Flow Control Protocol
                       ( ) 19200
IRQ #   : [4   ]↓      ( ) 38400     (•) None     (Disabled)
                       ( ) 56000     ( ) Xon/Xoff (Software)
[ ] Share IRQ          ( ) 115200    ( ) RTS/CTS  (Hardware)

Data Xfer Rate         Stop Bits     Buffer Size
  ( ) Slow             (•) 1         (•) 1 Kb        Ok
  (•) Normal           ( ) 1.5       ( ) 4 Kb
  ( ) Fast             ( ) 2         ( ) 8 Kb        Cancel
```

The **Serial Port** option allows you to select the serial port on your PC that you will be using for input. The Software Wedge supports the standard COM1 thru COM4 serial ports or you may specify a serial port by choosing "Other" for the Serial Port parameter and then entering a port "Address" (in hexadecimal) and Interrupt Request or IRQ number. The Port Address and IRQ number specifically identifies an installed serial adapter. Values for Port Address and IRQ should be documented in the literature provided with your serial adapter hardware. If you do not know the correct Address and IRQ values, you can also use the utility program **ShoPorts.Com** provided with the Software Wedge to determine the correct addresses and IRQs for all serial adapters installed in your PC. Please refer to the Utilities section of this manual for instructions on using ShoPorts.

The **Share IRQ** option allows you to instruct the Wedge to allow Interrupt Request Line sharing. Interrupt Request lines are hardware lines that are used by peripheral devices to signal the attention of the CPU. Normally each device that uses an IRQ line (serial ports, mice, sound cards, network adapters, etc.) must have exclusive control over the IRQ line that it uses. The Software Wedge will check a chosen IRQ line before installing itself into memory and if the IRQ line is being used by another device, the Wedge will display a message informing you of an IRQ conflict. Some newer PC's including EISA and Micro Channel models allow two or more devices to share an IRQ line. Also, some multi-port serial adapters are available that multiplex interrupts such that two or more serial ports can use the same IRQ. If you are using an EISA or Micro Channel PC or if you are using a multi port serial card that supports "Interrupt Multiplexing", you can safely select this option otherwise, it should be left unchecked. Note: If you share IRQs on a system that does not support it, you risk an almost certain crash. See Also: Troubleshooting

The **Buffer Transfer Rate** option allows you to select the rate of speed that the Software Wedge transfers data from its serial input buffer to the DOS keyboard buffer. If your application programs cannot tolerate fast filling of the keyboard buffer then you should select SLOW for this option. Symptoms of this problem are that characters appear to be randomly lost from the input. The FAST option will transfer data at the fastest possible speed and may usually be used safely if your PC's clock rate is higher than 8MHz. Otherwise you should select NORMAL. Note: Even at the fastest rate of speed, the keyboard buffer can only be "stuffed" at a rate of about 270 characters per second. This speed limit is due to the fact that the DOS keyboard buffer can only hold 15 keystrokes and because DOS only allows the Wedge the opportunity to stuff the keyboard buffer 18 times a second. This does not mean that you have to worry about losing data because the Wedge buffers all incoming serial data before transferring it to the keyboard buffer.

The **Buffer Size** option allows you to specify the size of the receive buffer. The receive buffer is an area in memory that is used to store incoming serial data that comes in faster than the Wedge can stuff the keyboard buffer or write to disk. The default input buffer size of 1024 bytes should be sufficient for most devices. If you need to input large amounts of data at baud rates above 1200 baud with no flow control, you may have to increase the input buffer size to avoid losing data. The receive buffer must also be large enough to hold one complete data record from your device. See Also: Defining the Record Structure.

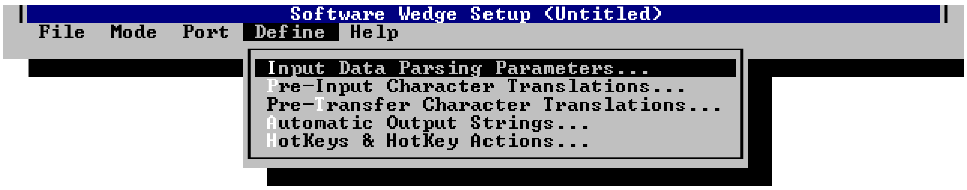
The size of the receive buffer is directly related to the amount of memory that the Wedge will occupy when it is installed. If you select a 1K buffer, the Wedge will use approx. 13Kb of memory. Selecting a 4K buffer will cause the Wedge to use approx. 18Kb and an 8K buffer will swell the Wedge to about 33Kb in size. Because the Wedge can write data to disk much faster than it can stuff the keyboard buffer, small buffer sizes are preferable when the Wedge is in "Input To Disk File" mode. The transmit buffer size is fixed at 1024 bytes.

The Port Analyze Menu Option

This option allows you to test the serial port settings that you chose in the Port Settings dialog box and also view or analyze data received from your serial device. This feature is designed to help diagnose serial I/O problems and to make it easier to configure the Software Wedge for your particular serial device and application program.

The Port Analyze option actually invokes the utility program COMSHOW.EXE that is provided with the Software Wedge. COMSHOW is essentially a simple terminal program with some very powerful features. For more information on using COMSHOW.EXE, refer to Utilities section of this manual.

The Define Menu


Software Wedge Setup (Untitled)
File Mode Port Define Help
Input Data Parsing Parameters...
Pre-Input Character Translations...
Pre-Transfer Character Translations...
Automatic Output Strings...
HotKeys & HotKey Actions...

The five options in the DEFINE menu and their functions are:

Input Data Parsing Parameters

Presents a series of dialog boxes that allow you to define the basic structure of input data from your serial device. The dialogs presented also allow you to specify how the Software Wedge should parse and filter data records before transferring them to another application or disk file. When you define the input record structure, you can also define additional data or keystrokes to include before or after each data field in an input record. See Also: Defining The Record Structure

Pre-Input Character Translations

Displays a translation table that allows you to translate incoming ASCII characters to other characters before they are acted upon by the Software Wedge. This is an extremely powerful concept that allows a great deal of flexibility in the way incoming data is parsed or filtered by the Software Wedge. See Also: Translation Tables

Pre-Transfer Character Translations

Displays a translation table for translating characters to keystrokes when the Software Wedge is in *Input As Keystrokes* mode or for translating characters to other characters when the Software Wedge is in *Input To Disk File* mode. Pre-Transfer translations are performed after the Software Wedge has parsed and filtered an incoming data record, just before the data is transferred to the DOS keyboard buffer or to a disk file.
See Also: Translation Tables

Automatic Output Strings

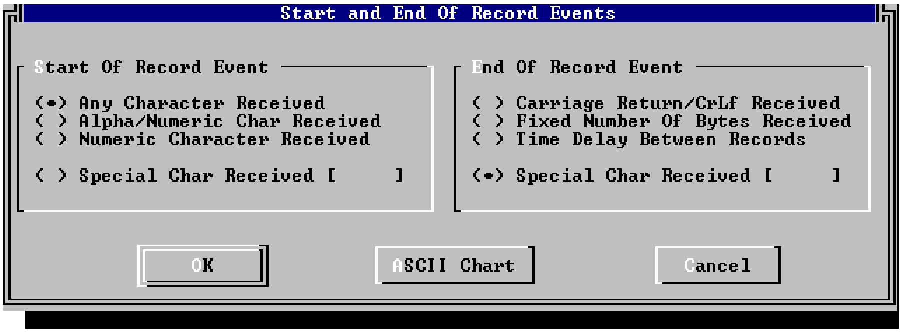
Opens a dialog box where you can pre-define character strings to be transmitted out to your serial device automatically by the Software Wedge. Several output strings may be defined including an initialization string, an acknowledgment string and one "timer controlled" output string. See Also: Defining Automatic Output Strings

Hot Keys and Hot Key Actions

Opens a dialog box where you can define up to 26 different hot keys that can be used to perform one of twelve different actions including transmitting data out the serial port, changing the state of serial port hardware lines, resetting the Wedge, enabling and disabling the Wedge, etc.. See Also: Defining Hot Keys

Defining the Record Structure

Because the Software Wedge has no way of knowing how the data from your serial device is structured or how to determine what parts of the data are important to your application, you must first define the structure of the incoming data. This option also allows you to define how the Software Wedge should parse and/or filter incoming data records for a particular application. The Software Wedge treats all incoming serial data as a stream of consecutive data *records* containing one or more *fields* of data. Therefore, several basic descriptors must be specified, the first being the events that determine the start and end of each record. When you select "Input Record Structure" from the Define menu, the following dialog box will appear:


Start and End Of Record Events

Start Of Record Event
(•) Any Character Received
( ) Alpha/Numeric Char Received
( ) Numeric Character Received
( ) Special Char Received [ ]

End Of Record Event
( ) Carriage Return/CrLf Received
( ) Fixed Number Of Bytes Received
( ) Time Delay Between Records
(•) Special Char Received [ ]

The Start Of Record Event

The *Start Of Record Event* options allows you to specify the event that triggers the Software Wedge to start accepting characters as part of a each data record (for the first data record and also after the end of each record).

If you select "Any Character Received" as the start of record event, the first character received will be considered to be the start of each data record. All subsequent characters received will be treated as part of the record until an "End Of Record Event" occurs.

The *Alpha/Numeric Char Received* option instructs the Software Wedge to ignore input until a valid Alpha/Numeric character is received (i.e. non control characters or chars with ASCII codes greater than 31 and less than 127).

The *Numeric Char Received* option causes the Wedge to wait until a numeric character is received (i.e. 0123456789.-)

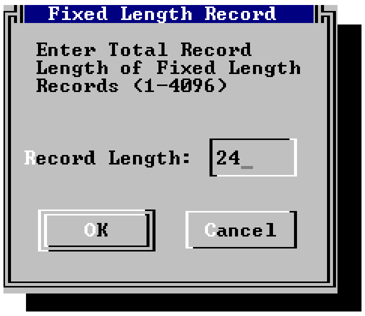
The *Special Char Received* option allows you to specify which characters are to signal the start of a record. If you use the Special Char Received option, you may enter one or more characters in the text box next to this option. To enter control codes in the text box, press the *ASCII Chart* button and choose specific characters from the displayed ASCII Chart. If you enter more than one character in the "Special Char Received" text box, the reception of any of the characters entered will trigger the start of a record. For example, if you specify "AB", either an "A" <u>or</u> a "B" will signal the start of a record (not the complete string "AB"). See Also: Pre-Input Character Translation Table

The End Of Record Event

Similar to the Start Of Record Event, the End Of Record Event options allow you to specify the event that will signal the end of each input record. The Software Wedge will not transfer any data to a foreground application or disk file until the selected End Of Record Event occurs. Therefore, you should select the "End of Record Event" that will always reliably determine when each serial input record ends.

If each data record is terminated by a carriage return or carriage return linefeed, the option "Carriage Return or CrLf Received" should be chosen. Note: Carriage return and line feed characters are **not** automatically removed if this option is chosen. If you do not want these characters as part of your data, use the "Pre-Transfer Character Translation Table" to remove them by translating them to "Nul". See Also: Translation Tables

If your data records always consist of a fixed number of bytes with one or more fixed length data fields, then the option "Fixed Number of Bytes Received" should be chosen. If you choose this option, you will be prompted with the following dialog box to enter the complete record length for your data records. See Also: Pre-Input Character Translation Table & The Port Analyze Menu Option

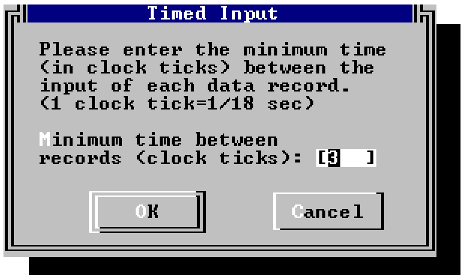


Fixed Length Record

Enter Total Record Length of Fixed Length Records (1-4096)

Record Length: 24

If the serial device that you are using does not transmit data records with a specific character marking the end of a record and the records from the device are not always the same length but there is always a time delay between inputs, then "Time Delay Between Records" should be chosen. A typical example of a device that may fit this criteria is a bar code reader or magnetic stripe reader. If you select this option, you will be prompted to enter the minimum amount of time (in 1/18 second increments) between the arrival of each new data record.

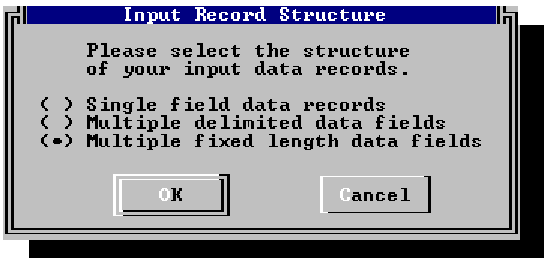

Timed Input

Please enter the minimum time
(in clock ticks) between the
input of each data record.
(1 clock tick=1/18 sec)

Minimum time between
records (clock ticks): [3   ]

The "Special Char Received" option allows you to specify characters that are to signal the end of an input record. If you choose this option, you may enter one or more characters in the text box next to this option. To enter control codes in this text box, press the "ASCII Chart" button and choose specific characters from a displayed ASCII Chart. The reception of any of the characters entered will signal the end of an input data record. NOTE: Special characters are **not** automatically removed if this option is chosen. If you do not want these characters to appear as part of your data, then you can use the "Pre-Transfer Character Translation Table" to remove them by translating them to "Nul" characters. See Also: Pre-Transfer Character Translation Table

If you choose either "Carriage Return or CrLf Received", "Time Delay Between Records" or "Special Char Received" as the End of Record Event, you will also be prompted to select a "record structure" for your data from the following dialog box:


Input Record Structure

Please select the structure
of your input data records.

( ) Single field data records
( ) Multiple delimited data fields
(•) Multiple fixed length data fields

The three possible record structures and their meanings are listed below:

Single field data records

Single field data records means that each entire data record should be considered as a single entity thus no parsing should be performed on the data. Note: The maximum length for a single data field is 4095 bytes.

NOTE: Some types of serial devices produce an output that may be too complex or too inconsistent to be parsed into a set number of "fields" by the Software Wedge. For these types of devices, it may be easier to define the entire output from the device as a single data field and then use the capabilities of the destination application (the macro language in Lotus 123 for example) to parse the data received from the Software Wedge.
See Also: Understanding the Data From Your Serial Device & Translation Tables

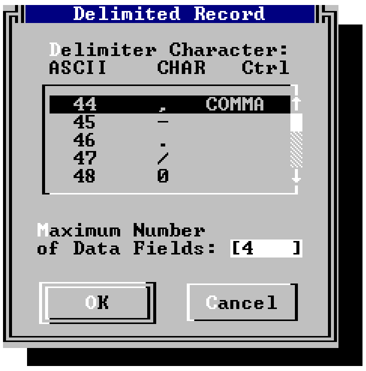
Multiple delimited data fields

This record structure option specifies that your data records consist of two or more unique "Fields" that should be parsed based on the position(s) of a delimiter character within each record.

For example the following record consists of four fields delimited by commas and with a carriage return linefeed at the end of the record: 12345, 9090, Value=123, 98765<CrLf>

If you select this option, you will be prompted with the dialog box shown below to specify the delimiter character used to separate data fields and also the maximum number of data fields that are contained in a single data record.

The Software Wedge supports records with up to 40 data fields. If more than the specified number of fields are received for a particular record, the additional data is discarded. If less than the specified number of fields are received, the remaining fields will be nul.


Delimited Record

Delimiter Character:

ASCII	CHAR	Ctrl
44	,	COMMA
45	-	
46	.	
47	/	
48	0	

Maximum Number of Data Fields: [4]

Special Considerations Regarding Delimiter Characters

All delimiter characters will be removed from the input data and thus will not appear as part of each data field. If you choose a space character as your delimiter, the Software Wedge will treat strings of consecutive spaces as a single delimiter. Also, if a chosen delimiter character matches a character that signals the end of an input record, (for example: you choose a carriage return as your delimiter character and you also choose "Carriage Return or CrLf Received" as your End of Record Event), the Wedge will treat the character as the delimiter until the specified number of fields have been received after which the next delimiter/end of record character received will signal the end of the record.

In cases where your input data records contain two or more different delimiter characters, you can use the "Pre-Input Translation Table" to convert all delimiters to a single character. For example, consider the following record:

```
Instrument# 123, Reading#3=23.45<CrLf>
```

Normally this record would be interpreted to contain two fields: "*Instrument# 123*" and "*Reading#3=23.45*", with a single comma delimiter between the two. If we use the Pre-Input Character Translation Table to convert the pound signs (#) and the equal sign (=) to commas, then after the translation, the record would appear to the Software Wedge as:

```
Instrument, 123, Reading,3,23.45<CrLf>
```

The translated record thus now contains five individual, comma delimited, data fields because of our creative use of the Pre-Input Translation Table.
See Also: Pre-Input Translation Table

Multiple fixed length data fields

Multiple fixed length data fields should be chosen as the record structure if you need to parse your data records based on the length of each data field. If you use this option, you will later be able to specify the exact length (number of characters) contained in each data field. This option should be used only in cases where you can always rely on each data field to contain a fixed number of bytes. This option is the default if you select "Fixed Number of Bytes Received" as the End of Record Event.
See Also: Pre-Input Character Translation Table

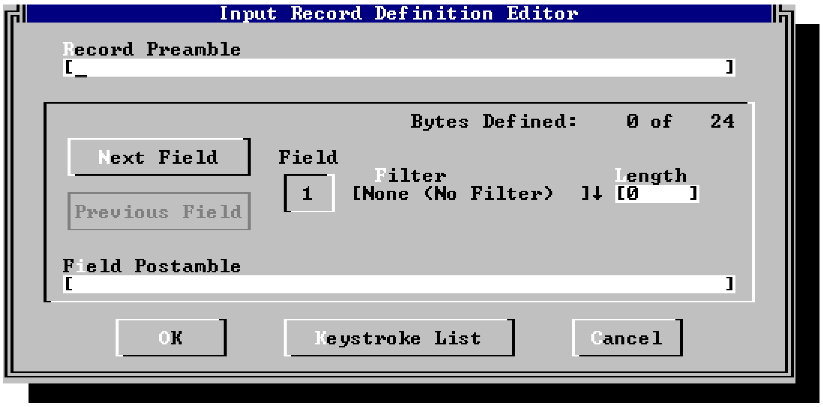
Specifying Filters and Field Lengths

After you specify the Start and End of Record Events and choose the basic structure of your data records, an "Input Record Definition Editor" dialog box will appear. This dialog box is where you specify a filter to be applied to each field and also the length of each field if you chose "Multiple fixed length data fields" as the record structure.

This dialog box also allows you to specify a record *Preamble* and field *Postambles* that perform different functions depending on the mode that you chose for transferring data. In "Input As Keystrokes" mode, the preamble and postambles are additional keystrokes or *macros* that are issued before or after each data field is sent to another program. In "Input To Disk File" mode, the preamble and postambles are additional ASCII characters or control codes that you want written to disk along with the serial input data.

See Also: Specifying Preambles & Postambles (Input As Keystrokes Mode)
Specifying Preambles & Postambles (Input To Disk File Mode)

Note: Some of the controls in the Input Record Definition Editor may not be displayed or may have different captions than the example below depending on the chosen record structure and data transfer mode.



The controls in the middle of the "Input Record Definition Editor" dialog box are where you specify a "Filter" to be applied to each data field in your input records as well as a "Field Postamble" and in cases where each data field consists of a fixed number of bytes, you must also specify a field "Length".

Under the label "Field" is a status box indicating the number of the current data field that you are defining a filter, a length and a Field Postamble for. If you specified that your data records consist of more than one data field, there will be two buttons marked "Next Field" and "Previous Field" in the dialog box. These buttons are used to scroll foreword and backward through the parameters for each data field, (the first field is Field 1; selecting the "Next Field" button will display parameters for Field 2, etc.)

Selecting Filters

There are four choices for the "Filter" that can be applied to each data field: None, Numeric Data Only, Ignore This Field, and Convert To Hex.

Specifying "None" for the filter means that no filtering is to be performed on the data for the current field, thus, all characters are to be accepted as valid data.

Specifying "Numeric Data Only" causes all alpha and control characters to be filtered out and only allows chars 0123456789-. to be accepted as valid data. The Numeric filter is especially useful when you are reading numbers into a spreadsheet. Any leading spaces or non-numeric characters would cause the spreadsheet to interpret the data as a label instead of a number.

If you select "Ignore This Field", then no data is accepted and the entire field is discarded. The Ignore filter is useful when you would like to remove data fields that are not required by your application.

The "Convert To Hex" filter causes the Software Wedge to perform a Binary to Hexadecimal conversion on the data contained in the field. Each character in the field is converted to a two byte string containing the hexadecimal value for the character. For example, an ASCII character 255 would be converted to "FF" and an ASCII 0 character would be converted to "00". The Convert To Hex filter allows the Wedge to work with raw binary data and not just ASCII text data.

Specifying Field Lengths

The text box labeled "Length" is where you specify the exact number of bytes that will be received for a particular field. The "Length" text box will only appear if you selected either "Fixed Number of Bytes Received" as the "End Of Record Event" or "Multiple Fixed Length Data Fields" as the record structure. The field length thus specifies the exact number of bytes in each specific field in the input record. Field lengths may range from 1 to 4095 bytes. For fixed length records, the sum total of all field lengths entered should equal the total length of the input record. See Also: Pre-Input Character Translation Table

In the middle, toward the right of the Input Record Definition Editor, are two status boxes labeled "Bytes Defined" that show the total of all specified field lengths as well as the total length of your data records (specified earlier for fixed length records). This information is only visible when the end of record event is: Fixed Number of Bytes Received.

Specifying Preambles & Postambles (Input As Keystrokes Mode)

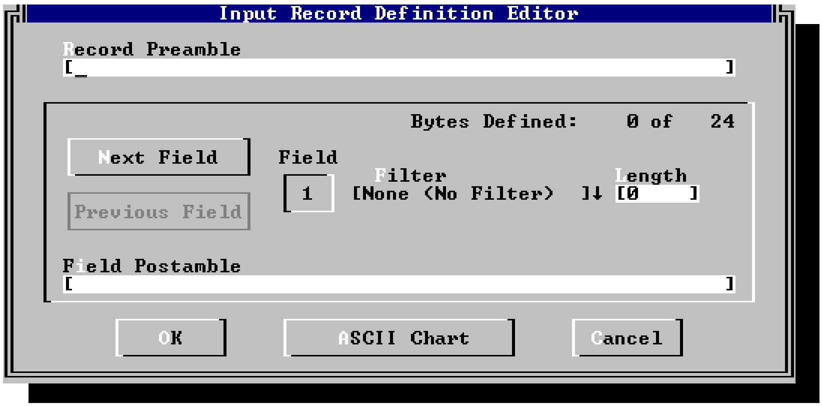
When the Software Wedge is in "Input As Keystrokes" Mode, the "Record Preamble" is a series of keystrokes that are to be issued immediately before all data fields from your serial device are transferred to your application program. For example, the Record Preamble could be used to move the cursor to a specific location in the foreground application just before the serial input data is transferred to it.

Likewise, the "Field Postambles" are keystrokes that are issued immediately following the data for each particular data field. Field postambles are usually used to navigate around in the application that is receiving the keystroke data. For example, if you are reading data from a device into a spreadsheet and you want successive readings to be entered into a single column, you could issue a field postamble macro that consists of a {DOWN} arrow key press. This would then cause the cursor to move to the next cell down in the spreadsheet after each reading from the device. You define macros to mimic the actual keystrokes that you would type if you were entering the data by typing it at the keyboard.

When specifying keystroke macros, you simply type the required keystrokes. If a required keystroke does not appear when typed because it is reserved for navigating menus and dialog items in SWCONFIG, a menu containing all possible DOS keystrokes and other special Date or Time stamp functions is available by pressing the button marked *Keystroke List*. The keystroke list allows you to select individual keystrokes or date and time stamp functions from a menu while editing any preamble or postamble.
See Also: Date and Time Stamps In Preambles & Postambles

Specifying Preambles & Postambles (Input To Disk File Mode)

When using the Software Wedge in "Input To Disk File Mode", the Input Record Definition Editor dialog box will appear as below.


Input Record Definition Editor

Record Preamble
[_]

Bytes Defined: 0 of 24

Field
1

Filter
[None (No Filter)]↓

Length
[0]

Field Postamble
[ ]

Most of the controls in this dialog box are the same as for "Input As Keystrokes" mode except that in "Input To Disk File" Mode, an ASCII chart is available instead of a Keystroke List. Because the Software Wedge will be writing all output to a disk file instead of to the DOS keyboard buffer, the Record Preamble and Field Postambles may only consist of additional ASCII characters that are to be written to disk along with the field data. For example, you may wish to write a carriage return - linefeed pair after each individual data field as they are being saved to disk. To specify preamble or postamble characters that cannot be typed on your keypad, the ASCII chart is available by clicking on the button marked "ASCII Chart".

Note: Preambles and Postambles in *Input To Disk File* Mode may also contain special date and/or time stamps. All Date and Time Stamp functions can be found at the bottom of the ASCII chart list box. See Also: Date and Time Stamps In Preambles & Postambles

Date and Time Stamps In Preambles & Postambles

The Software Wedge supports the following six date and time stamp functions that can be embedded in any preamble or postamble. Date and time stamp functions can be found in either the "Keystroke List" or "ASCII Chart" that is available when editing preambles or postambles.

Function	Return Value	Range
{Year}	Inserts the current year	00 - 99
{Month}	Inserts the current month	01 - 12
{Day}	Inserts the current day	01 - 31
{Hour}	Inserts the current hour	00 - 24
{Minute}	Inserts the current minute	00 - 59
{Second}	Inserts the current second	00 - 59

All date & time values are issued as a two byte string padded with a zero on the left if the value is less than ten.

Translation Tables

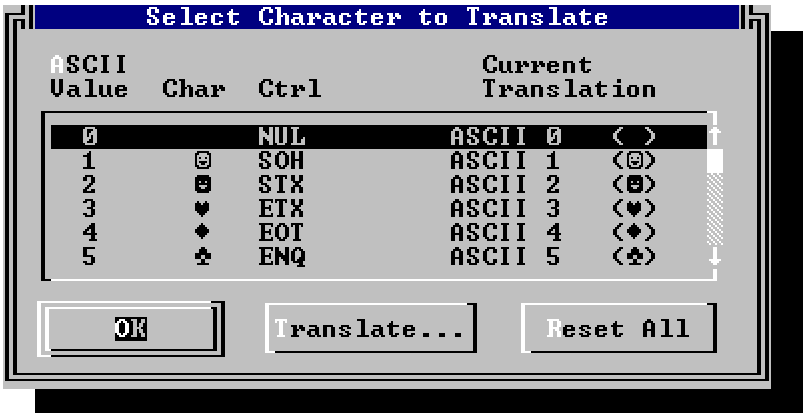
The Software Wedge provides two translation tables; a *Pre-Input Character Translation Table* and a *Pre Transfer Character Translation Table*. The two translation tables allow you to translate characters received from your serial device at two different points, - before the data is acted upon by the Wedge, and just before data is transferred to another application program or disk file, after the data has been parsed and filtered.

Pre-Input Character Translation Table

The Pre-Input Character Translation Table allows you to translate incoming ASCII characters to other ASCII characters before they are acted upon by the Software Wedge. This is an extremely powerful concept because it can dramatically alter the operation of the Software Wedge as well as the way that you think about your serial data.

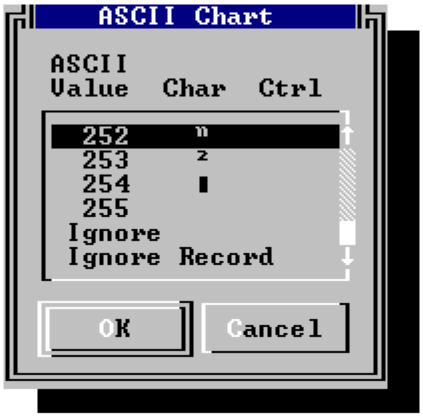
For example, when defining your input data record structure, you could specify that your data consists of multiple delimited fields with a particular delimiter character used to delimit each field. Suppose you choose a comma delimiter but your data records actually use two different characters as delimiters, both a comma and a tab character. In this case, you could simply translate all tabs to commas using the Pre-Input Character Translation Table and the Software Wedge would act on all tabs as if they were comma delimiters. See Also: Special Considerations Regarding Delimiter Characters

If you select *Pre-Input Character Translation Table* from the *Define* menu, the following dialog box containing the translation table will appear:



ASCII Value	Char	Ctrl	Current Translation
0		NUL	ASCII 0 ( )
1	☺	SOH	ASCII 1 (☺)
2	☻	STX	ASCII 2 (☻)
3	♥	ETX	ASCII 3 (♥)
4	♦	EOT	ASCII 4 (♦)
5	♣	ENQ	ASCII 5 (♣)

The translation table shows the ASCII values for all possible characters as well as their printable ASCII characters, ASCII control code mnemonics and the current translation for each character. To translate characters, select the character that you need to translate and then press the "Translate" button. This will cause the following ASCII Chart to be displayed where you can select a specific translation. The "Reset All" button in the translation table causes all translations to revert to their default settings.



At the bottom of the ASCII chart, you will find two choices "Ignore" and "Ignore Record" that do not represent actual characters but instead have special meanings when selected.

"Ignore" causes the translated character to be ignored by the Software Wedge. When a character translated to "Ignore" is received, it will be discarded and thus it will not appear in the input data. Also, because the character is being ignored, its presence is not counted when reading data into a fixed length record or a fixed length data field. Thus, *Ignore* removes the character from the input data before it reaches the Software Wedge.

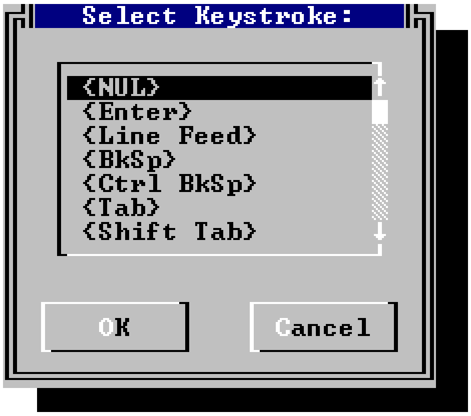
"Ignore Record" causes the Software Wedge to discard the entire current record if a character that has been translated to "Ignore Record" is present anywhere in an input data record. This feature can be useful in situations where you would like to reject certain outputs from a device. For example, suppose you had a device that always transmitted a special "initialization record" each time the device is turned on. If the initialization record contained any character that would never appear in any following record, you could use the "Ignore Record" translation to effectively remove the unwanted initialization record.

Pre-Transfer Character Translation Table

The Pre-Transfer Character Translation Table is used to translate characters *after* the Software Wedge has parsed and filtered your data and just before the data is sent to the DOS keyboard buffer or to a disk file.

The Pre-Transfer Translation Table is almost identical to the "Pre-Input Character Translation Table" except that this translation table will work differently depending on the current mode that the Software Wedge is set for. i.e. "Input As Keystrokes" mode, or "Input To Disk File" mode.

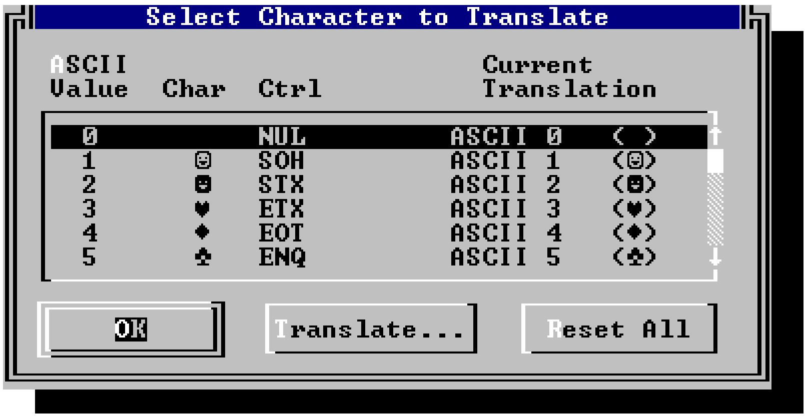
If the Software Wedge is in "Input As Keystrokes" Mode, then this translation table is used to convert individual characters to specific keystrokes. Because many ASCII characters do not correspond to a specific keystroke, the translation table provides a way to map characters to specific meaningful keystrokes. To translate characters, simply select the character that you need to translate and then click the "Translate" button. This will display the following "Keystroke Selection" dialog box that allows you to select a specific keystroke or key combination.


Select Keystroke:

{NUL}
{Enter}
{Line Feed}
{BkSp}
{Ctrl BkSp}
{Tab}
{Shift Tab}

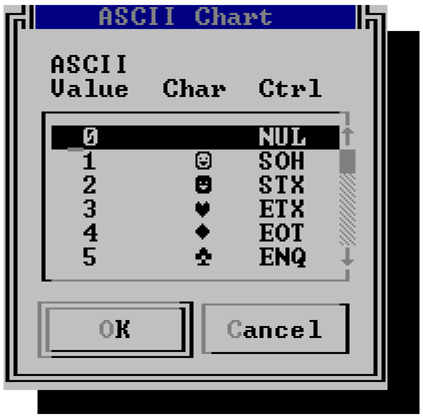
The keystroke selection dialog box contains a list of all DOS keystrokes as well as six special Date/Time stamp functions. The first item in the keystroke list, "NUL", does not represent an actual keystroke but instead is used to remove unwanted characters such as carriage returns and control codes before data is sent to a foreground application program.

If the Software Wedge is in "Input To Disk File" mode then the Pre-Transfer Character Translation Table will appear as shown below. In "Input To Disk File" mode this translation table is similar to the Pre-Input Translation Table in that it only allows you to translate characters to other ASCII characters (not to keystrokes).


Select Character to Translate

ASCII Value	Char	Ctrl	Current Translation
0		NUL	ASCII 0 ( )
1	☺	SOH	ASCII 1 (☺)
2	☻	STX	ASCII 2 (☻)
3	♥	ETX	ASCII 3 (♥)
4	♦	EOT	ASCII 4 (♦)
5	♣	ENQ	ASCII 5 (♣)

When you click on the "Translate" button to translate a character, the following ASCII Chart will appear allowing you to translate the selected character to another ASCII character. This ASCII Chart is similar to the one displayed for the Pre-Input Translation Table except that it does not have an "Ignore" or "Ignore Record" entry.

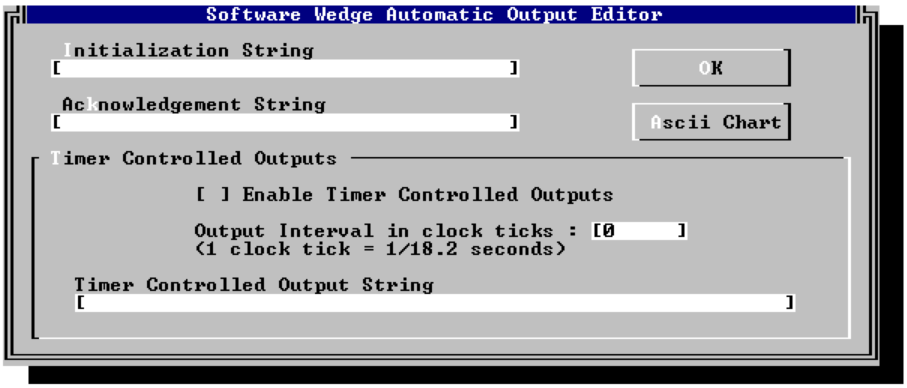


ASCII Value	Char	Ctrl
0		NUL
1	☺	SOH
2	☻	STX
3	♥	ETX
4	♦	EOT
5	♣	ENQ

NOTE: In Input To Disk File Mode, Nul characters (ASCII 0) and characters translated to "Nul" are written to disk along with the rest of your data.

Defining Automatic Output Strings

Selecting "Automatic Output Strings" from the Define menu opens the dialog box shown below where you may define an initialization string, an acknowledgment string and a timer controlled output string.



```
Software Wedge Automatic Output Editor

Initialization String
[                                    ]        OK

Acknowledgement String
[                                    ]        Ascii Chart

Timer Controlled Outputs

            [ ] Enable Timer Controlled Outputs

            Output Interval in clock ticks : [0      ]
            (1 clock tick = 1/18.2 seconds)

 Timer Controlled Output String
 [                                                      ]
```

The Initialization String is an output string that is automatically sent out the serial port when the Wedge is initially loaded into memory or when the Wedge is reset. The Initialization string is meant to be used to initialize a serial device (i.e. send configuration data or special commands to a device, dial a modem, etc.).

The Acknowledgment String is an output string that is automatically sent out the serial port after each complete data record is received from your serial device. The capability to send an Acknowledgment string was originally intended for those devices that require one as part of their serial protocol but it could also be used as a way to continually request data from a device that can be polled by sending it a character string.

A timer controlled output string may also be defined that is automatically transmitted at regular timed intervals. The timer interval value may range from 1 to 65535 clock ticks. (One clock tick equals **1/18.2** seconds i.e. 91 clock ticks=5 seconds, etc.). A check box allows you to specify whether or not timed automatic outputs are initially enabled when the Software Wedge is loaded into memory. Note: You may also define hot keys that will allow you to enable or disable timed automatic outputs. See Also: Defining Hot Keys.

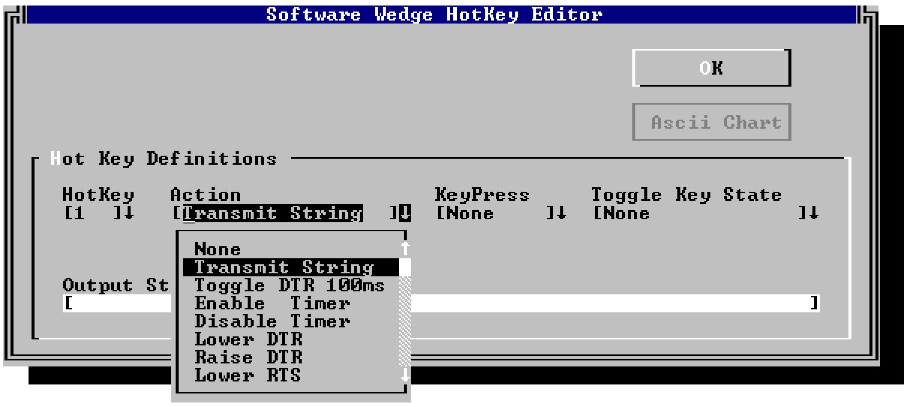
If you specify a nonzero time interval but do not actually enter any data for the timer controlled output string, the Wedge will toggle the DTR line for 100ms at regular intervals instead. Because some serial devices (Sylvac digital calipers for example) use the toggling of the DTR line as a signal to transmit a data reading, this feature allows you to poll these types of devices at regular intervals.

When editing output strings, you simply type in the string that you want sent. If you need to output ASCII characters that cannot be typed on your keyboard, an ASCII chart is available by either clicking your mouse on the command button labeled "ASCII Chart" or by holding the ALT key while you press the letter A. To select characters from the ASCII chart, scroll the chart with the arrow keys until the desired character is hi-lighted and then press the Enter key. The selected character will appear at the current cursor position in the output string that you are editing.

Defining Hot Keys

The Software Wedge allows you to pre-define up to 26 "Hot Keys" that perform any of a dozen different actions when pressed on your PC's keyboard while the Wedge is running. The hot key actions that are available are listed below:

	Hot Key Actions	**Description**
1.	Transmit String	Transmits a character string out the serial port
2.	Toggle DTR for 100ms	Lowers the serial port DTR line for 100ms
3.	Enable Timer	Enables timer controlled output string
4.	Disable Timer	Disables timer controlled output string
5.	Raise DTR	Raises the serial port DTR line
6.	Lower DTR	Lowers the serial port DTR line
7.	Raise RTS	Raises the serial port RTS line
8.	Lower RTS	Lowers the serial port RTS line
9.	Reset Wedge	Resets the Wedge & clears all buffers
10.	Disable Wedge	Disables the Wedge
11.	Enable Wedge	Enables the Wedge
12.	Enable Wedge for 1 Record	Enables the Wedge for one input record only


Software Wedge HotKey Editor

Hot Key Definitions
HotKey [1]
Action [Transmit String]
KeyPress [None]
Toggle Key State [None]
Output St [ ]

To define or edit hot keys, first select the hot key that you want to define or edit from the drop down list with the caption "Hot Key". Hot keys are simply numbered from 1 to 26 and the number that you use for each hot key is arbitrary. Next, select the desired hot key action from the drop down list labeled "Action". If you choose "Transmit String" for the hot key action, a text edit box will appear where you can edit the output string. Finally, you must select the PC keystroke and toggle key state that will invoke your hot key action. The "Keypress" is the actual key that invokes the action and the "Toggle Key State" specifies which, if any, toggle keys (i.e. Shift, Ctrl or Alt) must be held in combination with the keypress.

For example, to define a hot key that transmits a question mark out the serial port whenever you press the key combination Ctrl+F1, simply select "Transmit String" as the Action, "F1" as the Keypress, "Ctrl" as the Toggle Key State and in the text edit box for the output string, enter a question mark.

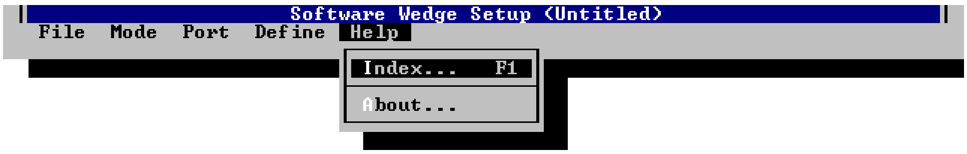
An ASCII chart is available when editing the output string so that you can select characters that cannot be typed on your keyboard. The ASCII chart is available by selecting the button labeled "ASCII Chart". To select characters from the ASCII chart, scroll the chart until the desired character is selected and then press the Enter key. The selected character will appear at the current cursor position in the output string.

When defining hot keys, you may not use the same "Keypress" + "Toggle Key State" combination for more than one hot key. If you try to define two hot keys that are both assigned the same combination of Keypress and Toggle Key State, only the first hot key action will be invoked when you press the hot key.

One typical use for a hot key is to prompt or reset a serial device. For example, many instruments can be instructed to transmit a data reading by sending them a prompt string. For this situation you could define "Transmit String" hot key with an output string containing the required prompt characters. Then whenever you press your hot key, the prompt string would be sent to the device causing it to send back a data reading.

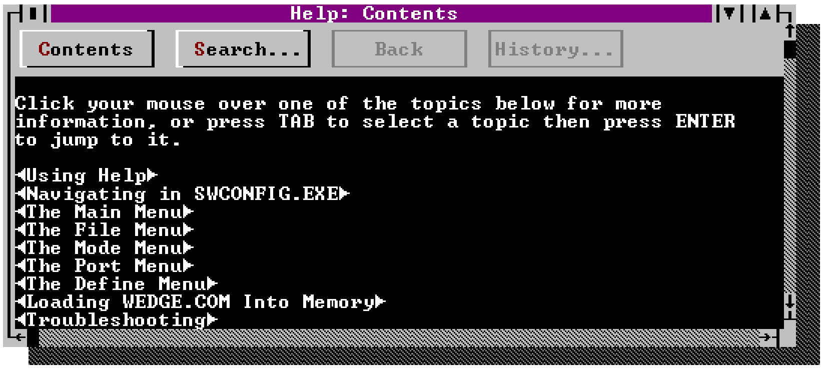
Hot keys used in conjunction with each other can make it possible to deal with all kinds of interesting situations. For example, suppose you had a device such as a weigh scale or temperature probe that continually transmitted data readings but you only wanted to capture a single reading at a time instead of having the data continually stream into your application program. You could define two hot keys, one that resets the Wedge and another that enables the Wedge for one data record. If you then install the Wedge initially disabled, whenever you would like to input a single data reading you could simply press the "Reset" hot key to flush the Wedge's input buffer and then press the "Enable 1 Record" hot key to allow only the next single data reading to be inputted.

Accessing On-Line Help



On-Line help is available from the main menu of the Software Wedge by selecting the Help sub menu item "Index". This option will display a help dialog box containing an index that allows access to help topics for the Software Wedge. On-Line help is only available from the main menu in SWCONFIG. Selecting the Help sub menu item "About" will display a copyright notice.

When you select "Index" from the Help menu, a dialog box similar to the one below will be displayed allowing you to select help topics. Help topics are listed in the help dialog box surrounded by small triangles. You can jump to a help topic by clicking your mouse on the topic text between triangles or you can press the TAB key to select a topic and then press the ENTER key to jump to the selected topic. The first topic in the table of content is "Using Help". Select the "Using Help" topic for instructions on using all other features available in the Help dialog box.



```
Click your mouse over one of the topics below for more
information, or press TAB to select a topic then press ENTER
to jump to it.

◄Using Help►
◄Navigating in SWCONFIG.EXE►
◄The Main Menu►
◄The File Menu►
◄The Mode Menu►
◄The Port Menu►
◄The Define Menu►
◄Loading WEDGE.COM Into Memory►
◄Troubleshooting►
```

Loading WEDGE.COM Into Memory

Wedge.Com is a TSR which means that it must be loaded into memory before it can do its job. You can install Wedge.Com either by selecting "Save, Exit and Install" from the File menu in SWCONFIG or you can enter the command: "WEDGE *filename"* at the DOS prompt where *filename* is the name of a configuration file. For example, if you create a configuration file for the Wedge and save it to a directory called "Wedge" with the name MyConfig.SW, you would issue the command: **WEDGE C:\WEDGE\MYCONFIG.SW** at the DOS prompt or from within a batch file (or AUTOEXEC.BAT) to load the Wedge into memory. (It is assumed above that Wedge.Com is in the current directory)

Note: If you install the Wedge from the main menu in SWCONFIG, a batch file named "SW.BAT" will be created in the current directory that contains the WEDGE command including the name of the currently open configuration file. After you install the Wedge once from within SWCONFIG, all you have to do to re-install it with the same configuration is to issue the command SW at the DOS prompt from within the directory where SW.BAT was created.

When you install the Wedge into memory, the first thing it does is read in the configuration parameters from the specified configuration file and then it checks to see if a copy of the Wedge is already installed for the selected serial port. If the Wedge is already installed for the chosen port, the configuration parameters are passed on to the installed Wedge. This allows you to re-configure the Wedge for a particular serial port without having to remove it from memory.

Note: The only parameter that cannot be changed while the Wedge is loaded in memory is the Buffer Size. If you need to change the buffer size after the Wedge has been installed, you must remove the Wedge and then re-install it with the new buffer size.
See Also: Removing the Wedge From Memory

If the Wedge is installed properly, you will receive a message similar to the one below indicating that the Wedge has been installed successfully:

```
Software Wedge Installation for COM1:1200,E,7,1
COMPLETED SUCCESSFULLY
```

(The Com port, baud rate, parity, number of databits and stopbits will be those that you specified in your configuration file and may differ from the example shown above.)

How To Test The Software Wedge

To test that the Software Wedge is working properly in "Input As Keystrokes" mode, simply enter data from the device that you have connected to your serial port. If the Software Wedge is installed properly, data entered from the device will appear as if it were typed in through the PC's keyboard.

If the Wedge is in "Input To Disk File" mode then you should be able to see all your data by using the DOS "TYPE" command or an editor to display the contents of the data file that you specified to store your input data.

If you get garbled data or characters that look like they are from another planet then the most likely cause is a mismatch in serial communications parameters between the Wedge and your serial input device (i.e. baud rate, parity, number of databits, etc.).

If you do not get any data at all, then refer to the Troubleshooting section of this manual.

Installing The Wedge For Multiple Serial Ports

The Software Wedge may be installed for as many serial ports as you like provided that you have the proper serial adapter hardware. If you have just COM1 & COM2 then you should have no problems on any machine using the Wedge with both ports simultaneously. Simply create two configuration files (one for each serial port) and install the Software Wedge twice.

EISA or Micro Channel PC's should not have any problems collecting data from multiple serial ports. If your PC has more than two serial adapters in use at a time and is not an EISA or Micro Channel based PC, then please read the section titled: Using More Than Two Serial Adapters At A Time.

Note: If you install the Wedge for more than one serial port, a separate copy of the Wedge will be loaded into memory for each different port.

Using More Than Two Serial Adapters At A Time

If you will be using the Software Wedge on a computer that is not a Micro Channel or EISA PC and more than two serial ports will be active at the same time, then you may have to change the configuration of your serial adapters for ports COM3 and above.

The standard bus architecture (ISA bus) for most IBM or compatible 80x86 PC's provide dedicated interrupt request lines for only two serial adapters (COM1 uses IRQ4 & COM2 uses IRQ3). Many serial adapters that provide COM3 & COM4 also use the same IRQ lines as COM1 & COM2. This is not a problem if polled serial I/O is used or if only one of two ports that use the same IRQ is active at a time, but when interrupt driven serial I/O is used, as by the Wedge, then two active serial ports will end up fighting with each other for control of the shared interrupt request line and one or both will lose. One adapter will try to drive the IRQ line "high" while the other adapter tries to drive it "low". EISA and Micro Channel bus PC's avoid this problem by using an edge triggered interrupt control mechanism instead of the original level triggered scheme used in the ISA bus.

One way around this problem is to take advantage of an unused IRQ line on your PC (usually IRQ 2, 5, 7,10,11,12 or 15) by configuring your COM3 or above adapter's jumper or switch settings to use one of these IRQ lines instead of IRQ4 or IRQ3. Most add-on serial adapters provide jumpers or switches just for this purpose. This must be done with care because many other peripheral devices including network cards, internal modems and sound cards may also use these IRQ lines. You can check to see which IRQ lines are free using the Microsoft Diagnostics program that comes with DOS. Type: MSD at the DOS prompt and when the menu appears, select "IRQ Status". IRQ lines that are available for use will be specified as "Reserved".

Another solution is to use a multi port serial adapter that provides an "interrupt multiplexing" scheme. DigiBoard Inc. (612)-943-9020 carries a complete line of multi port serial adapters and they also have excellent technical support for their products.

If you try to install the Software Wedge for a serial adapter that uses an IRQ line that is already being used by another hardware device, the Wedge will normally warn you that the chosen IRQ is in use and then prompt you to press ENTER to continue with the installation or press any other key to abort without installing the Wedge. You are generally safe to proceed if the PC you are using is an EISA or Micro Channel PC (IBM PS2) or if you are using a multi port serial adapter that supports interrupt multiplexing, otherwise you should abort the installation altogether. Note: You can override the "IRQ in use" error message by selecting "Share IRQ" in the Port Settings dialog box in SWCONFIG.

Removing the Wedge From Memory

The Software Wedge may be removed from your PC's memory by issuing the command "WEDGE/R" at the DOS prompt. This first disables the WEDGE and then frees up the memory used by it thus making it available to other application programs. The removal command also restores the serial port to the state it was in prior to loading the Wedge.

Because the Wedge may be installed for multiple serial ports, more than one copy of it can exist in your PC's memory (one copy for each serial port that it was installed for). If you install the Software Wedge for more than one port and then wish to remove all copies from memory, you will have to issue the removal command once for each installed copy. The most recently loaded copy will be removed first.

Troubleshooting

If you experience problems getting the Software Wedge to work on your PC, please go through the following steps to determine exactly what type of problem you are having.

If are not receiving any data all:

1. Make sure that the PC you are using is equipped with the serial port selected in the SWSETUP.EXE program. If you do not have the chosen serial port or if your serial port is configured improperly then you must correct the problem and try again.
Important: Make sure that the Port Address and IRQ number for your serial port match the values in the Port Settings dialog box in SWCONFIG.EXE. The Software Wedge comes with a utility called "ShoPorts.Com" that will display a list of all serial ports installed in your PC including their addresses and IRQs. See Also: ShoPorts

2. Check that you are connected to your PC using the proper RS232 cable for connection to a PC and that you are plugged into the serial adapter that you specified in SWCONFIG. Also make sure that your serial device is getting power and is turned on. You may also want to contact the manufacturer of the device to confirm that you are doing everything that is necessary to connect to a PC and also to get the device to transmit data.

3. Use the Port Analyze feature of the SWCONFIG.EXE program and try, at least, to get some data from the device to appear on your screen. If you cannot get any data at all to appear (even garbage) then either the device is not transmitting anything, or you need a "NUL Modem Adapter". A nul modem adapter is a small plug that connects between the serial cable from your device and serial port on your PC. Its purpose is essentially to cross the transmit and receive lines in the cable so that the transmit line from your serial device is connected to the receive line on your PC and vice versa. If you normally connect your serial device to a printer and you can successfully print to the printer, then you definitely need a NUL Modem Adapter. You can pick one up at any Radio Shack or computer supply store for about five dollars. Take the device or the cable to the store with you so that you can match the adapter to the cable.

4. Remove all other Terminate & Stay Resident programs from your PC and try running the Wedge again. You may have to edit your CONFIG.SYS or AUTOEXEC.BAT files and re-boot your computer to perform this step. Please read the section of this manual entitled: Running the Software Wedge along with other TSR's

If are receiving data but the data appears garbled or unreadable:

5. Check that the serial device is set up using exactly the same communications parameters as the Wedge, i.e. Baud rate, Parity, Data Bits, and Stop Bits. If you do not know the parameters used by your device, then you will need to either consult the users manual for the device or contact its manufacturer for this information. You can also use the Port Analyze feature in SWCONFIG to try different parameters until you get data that appears correct. Try different baud rates first using No Parity, Eight Data Bits, and One Stop Bit until you get data that looks partially correct. Finally, try different combinations of Parity, Number of Data Bits and Number of Stop Bits until all data is correct. Most devices use either seven data bits with Even or Odd parity or eight data bits with No Parity. One stop bit is also used more frequently than two.

If data appears using the Port Analyze feature but not at the DOS prompt:

6. Make sure that the Wedge is enabled and try reading in data at the DOS prompt. If no data appears at the DOS prompt, then you most likely have the Wedge configured incorrectly (either the Start or End Of Record Events are wrong or your parsing and filtering parameters are specified incorrectly). If you are inputting very large data records, you may also have to increase the size of the receive buffer. The receive buffer must be large enough to hold one complete data record. Re-configure the Wedge and try again.

If data appears at the DOS prompt but not in your application program:

7. Try using the Wedge with a different application program. If this works, then the problem may be with the application program that you are using. There are a small number of commercial programs that do not read their input from the DOS keyboard buffer. Instead, these applications read the keyboard directly and bypass DOS altogether. You can test if this is the case with a particular application simply by exiting from it. If data from the Wedge starts streaming in at the DOS prompt immediately after you exit the application, then the application is definitely the problem and the only thing that you can do about it is either call the publisher of the application and complain or find another program that does the same job without being so uncooperative. If experience this problem, please call TAL Technologies, Inc. and report the name of the application so that we can warn other Wedge users. A README.TXT file on the Software Wedge diskette contains a list of application programs that are known to have this problem.

If you cannot get the Wedge to operate properly after you have gone through the above procedures, then call or fax TAL Technologies, Inc. for assistance at: Tel: (215)-763-5096 or Fax:(215)-763-9711 Email support @taltech.com. Please have this manual and your original Software Wedge diskette(s) at hand and if possible, be at your PC when you call for support. Support is also available on line at http://www.taltech.com.

Memory Requirements And Technical Information

The TAL Technologies, Inc. Software Wedge was written entirely in assembly language and is therefore very compact as well as extremely fast and efficient. With the input buffer size set to 1Kb, the Professional Edition of the Software Wedge uses only 12800 bytes of memory after it has been installed and can operate reliably at data rates up to 115200 baud on a reasonably fast PC.

The Wedge uses fully double buffered, interrupt driven serial communications. This means that if data arrives at the serial port faster than it can be read by your application software, data will not be lost; it will be buffered until your application program is ready to accept it. Note: It is possible to overrun the serial input buffer by reading in huge quantities of data at high speeds but if your serial input device has the ability to transmit large amounts of data then it probably also supports some form of flow control that can be used of to insure that the serial buffer is never overrun. The Software Wedge supports all common types of flow control.

If you have serial adapters that use one of the newer 16550AF UARTS, then the Wedge will take advantage of the FIFO buffers provided by these advanced chips, thus allowing even faster and more reliable data throughput. The Wedge can also be used with the latest generation of serial adapters that support selection of IRQ's 8 thru 15 .This allows you to avoid IRQ conflicts and also allows you to install more serial adapters and other peripherals in your PC.

The Wedge will always check to see if it is already installed for a particular serial port each time it is run. This prevents having multiple copies of the Wedge in your computer's memory for the same serial adapter. If you need to change any parameters after the Wedge has been installed, you can simply re-install the Wedge using the new parameters. The new parameters will be used by the existing copy of the Wedge without placing a redundant copy in your computer's memory.

The TAL Technologies, Inc. Software Wedge is hands down the most powerful, most advanced, most reliable, smallest, fastest and easiest to use PC Serial I/O product on the market today. Tens of thousands of copies of the TAL Software Wedge are reliably being used daily in thousands of different "mission critical" data collection applications around the globe.

A Few Words About TSRs

TSR programming has long been viewed as a black art, even by many very experienced PC programmers. There is a common misconception that TSRs rely on undocumented DOS secrets hidden away in a deep, dark vault somewhere in Redmond, Washington. Many PC users are also afraid of TSRs and will try to avoid using them at all costs. Typically, the most common fear is that a PC will "lock up" unexpectedly because a TSR does something that "isn't supposed to be done". These fears are usually based on real experiences with poorly written TSR programs. Unfortunately, because of a few badly behaved programs, TSRs in general have gotten a bad name. Rest assured that TSR programming is a legitimate and legal science. There are no secrets or mysterious magic potions, only proper programming conventions. In fact, all device drivers and many parts of DOS itself are actually TSRs, without which, most PCs would be unable to function.

The SOFTWARE WEDGE strictly adheres to all standard TSR programming conventions and will therefore never lock up your PC or cause any unexpected disasters. It will not slow down your PC in any way and it may be used with full confidence along with all other well behaved PC application programs (including other TSRs).

Running The Software Wedge With Other TSRs

If you will be running the SOFTWARE WEDGE along with other TSR programs, then there is a small chance that you will experience difficulties because of a conflict with another TSR. The SOFTWARE WEDGE was written properly so that it will function along with other TSR's but unfortunately, a few other TSRs out there do not follow all the proper (and well documented) conventions for TSR programs.

Almost all TSR conflicts can be resolved by experimenting with the order in which you load each TSR program. We recommend that you load the SOFTWARE WEDGE first, before loading other TSR's. If the SOFTWARE WEDGE works before but does not work after you load a subsequent TSR, then you may have to switch the load order around in order to get it to work. The key word here is "experiment", especially before calling for technical assistance. Note: There are several shareware TSR management programs available that can help to resolve TSR conflicts, the most famous is called "Mark and Release" and can be obtained from Public Brand Software. (800) 426-3475

Other types of "terminate & stay resident" programs include keyboard enhancer programs, print spoolers, and "pop-up" utilities.

If you find a situation where the SOFTWARE WEDGE will not work at all or returns corrupted data because of a contention with another TSR, then please call TAL Technologies, Inc. and inform us of the situation.

NOTE: The SOFTWARE WEDGE has been found to work extremely well with most commercially available, and many shareware, keyboard enhancer programs. This opens up almost limitless data entry and data formatting possibilities. For example you could issue keystrokes from the SOFTWARE WEDGE that invokes another macro defined in your keyboard enhancer program. Macro chaining will also work with most spreadsheet programs (including Lotus 123) which will allow you to make logical decisions or chart graphs using data received from your serial input device. With a little creativity, all sorts of wonderful results can be obtained.

Running The Wedge Under DESQVIEW OR WINDOWS 3.x

When using either DESQVIEW or WINDOWS 3.x in Real Mode or Standard Mode, Wedge.Com should be installed before starting either of these programs. The Wedge will work properly when installed this way for all DOS application windows opened up by either program. You will also be able to switch windows and still use the SOFTWARE WEDGE for input in any DOS application window that you open or switch to.

With DESQVIEW, or Windows 3.x in Real or Standard mode, the foreground program will always receive the data from the Wedge no matter how it is installed. This also means that you cannot open up a window in DESQVIEW and run the Software Wedge with another (DOS) program and have it running in the background collecting data from your serial input device while you are working in the foreground with another program. This type of operation is only possible with WINDOWS 3.x in 386 Enhanced mode.

The SOFTWARE WEDGE should not be installed into memory before running WINDOWS 3.x in 386 Enhanced mode. Because of the way Windows creates "Virtual PC's", you must install Wedge.Com in each separate DOS window that will use the Wedge for input.

The recommended way of doing this is to use the Windows PIF Editor to create a PIF that invokes a batch file that first loads the Wedge into memory and then starts the application that will be using the Wedge. If you would like to have the application running in the background while you use another application in the foreground, you will need to check off the selection "Background" under the execution options in the main PIF Editor window and also clear the "Detect Idle Time" option from the "Multitasking Options" section of the PIF Editor "Advanced Options". You may also want to give the Wedge window a higher "Background Priority" setting to keep things running more smoothly.

If you want to run the Software Wedge from the "DOS Prompt" window in the "MAIN" program group of the Windows Program Manager and you would like to run the Wedge in the background in the DOS Prompt window, you will need to edit the PIF named "DOSPRMPT.PIF" and possibly the "_DEFAULT.PIF" so that the "Background" and "Detect Idle Time" options are set as described as above.

Note: For the Wedge to work properly in the background, you may have to edit the PIFs for any other DOS applications that will be running at the same time as the Wedge. You need to make sure that the "Exclusive" option in the "Multitasking Options" section of the other application's PIF is not checked.

Understanding The Data From Your Serial Device

Before you can correctly configure the Software Wedge to parse and filter data received from your serial device, you must first have a complete understanding of the data that your device transmits. You should also be able to recognize what parts of the data are important to you and also completely understand what features of the Software Wedge can be used to manipulate your data. To use the Software Wedge effectively, you must think of the input data from your device as a stream of *records* containing one or more specific *fields* of data. The next step is to decide what features of the Wedge can be used to consistently separate each data record from the next record and also what features can be used to separate individual data fields within a record from the next field. Finally you need to identify which fields within each record are important to your application and which fields should be removed or ignored.

The best place to start is to refer to the users manual for your serial device. The users manual should have a section that describes the structure and contents of all data that can be outputted from the device. If you do not have a users manual for your device or if the output data structure is not described, you can use the Port-Analyze feature in the Software Wedge or some other serial I/O program to analyze the output from your serial device by actually viewing the data that is transmitted from it.

When you configure the Software Wedge for a particular device you must define the input record structure to the Wedge by selecting "Input Record Structure" from the *Define* menu. When defining the input record structure, you specify a "start of record event", an "end of record event" and a general record structure for each data record (i.e. single field record, multiple delimited fields, or multiple fixed length fields). With some devices, the record structure may be immediately obvious. For example, the data record below contains three numeric data fields delimited by commas and terminated by a carriage return-linefeed.

110,250,801<CrLf>

For this record, you would probably select "Any Character Received" as the *Start Of Record Event* and "Carriage Return or CrLf Received" as the *End Of Record Event* and for the *Record Structure* you would choose "Multiple Delimited Data Fields" and specify that there are three data fields per record with a comma delimiter separating each field.

Other devices may output data with a record structure that is less obvious. Consider the following two possible (different) output records:

```
Sample#1,*213**32****23**<CrLf>
Sample#2,*215**437**141**797**89**56<CrLf>
```

The two records are similar in that they both are terminated by a carriage return-linefeed and they both contain multiple delimited data fields. A minor problem with the above two records is that they both contain a different number of data fields. Another complication is that they seem to contain both a comma and one or more asterisks as delimiters. In the first record it appears that we have four data fields: "Sample#1", "213", "32" and "23"
In the second record it appears that we have seven data fields:
"Sample#2", "215", "437", "141", "797", "89" and "56"

There are actually several ways to configure the Wedge to correctly parse both records in the above example. The most elegant method is to use the "Pre-Input Character Translation Table" to modify the delimiters so that they are all the same. For example, we could translate all asterisks to commas to end up with the two records shown below:

```
Sample#1,,213,,32,,,,23,,<CrLf>
Sample#2,,215,,437,,141,,797,,89,,56<CrLf>
```

Now both records have multiple, delimited, data fields with a carriage return-linefeed at the end of each record. We can determine the maximum number of fields in each record by adding up the number of commas in each record and adding one. (i.e. the first record has 11 fields and the second has 13 fields therefore the maximum number of fields is 13)

Now, when you define the record structure in the Software Wedge you would select "Any Character Received" as the *Start Of Record Event* and "Carriage Return or CrLf Received" as the *End Of Record Event* and for the *Record Structure* you would choose "Multiple Delimited Data Fields" and specify that the maximum number of data fields per record is 13 with a comma delimiter separating each field.

When parsing records with multiple delimited data fields, if you choose a space as your delimiter, consecutive spaces would be treated as a single delimiter. Therefore, in the above example, if instead of translating asterisks to commas, you were to translate both commas and asterisks to *spaces*, and then chose a space as your delimiter, you would end up with the two records below with each field separated by a single (space) delimiter:

```
Sample#1 213 32 23 <CrLf>
Sample#2 215 437 141 797 89 56<CrLf>
```

Suppose you had a device that normally sent its output to a serial printer in the following manner with more than one line of data:

```
Device#         Height      Width    Length      <CrLf>
-----------------------------------------------------  <CrLf>
     1          23           12        24        <CrLf>
     2          27           13        8         <CrLf>
     3          27           13        9         <CrLf>
     4          27           13        8         <CrLf>
-----------------------------------------------------  <CrLf>
<FF>
```

(<CrLf> represents a carriage return-linefeed and <FF> represents a form feed character)

You can still think of the entire output shown above as a single record even though it contains several lines of data or you can think of each line as a complete record. You can even think of each word or number as a complete, single field, data record. Again there are hundreds of different ways to parse the entire output from the device using the different features available in the Software Wedge.

If you wanted to capture only the lines with numeric data and parse each of these lines into records with four fields, one way to accomplish this would be to specify the "Start Of Record Event" as "Numeric Character Received" and then specify the "End Of Record Event" as "Carriage Return or CrLf Received". Finally you would define the record structure as "Multiple Delimited Data Fields" with four fields per record and a space as the delimiter character. Because you chose "Numeric Character Received" as the "Start Of Record Event" and because there are no numeric characters in the first, second, seventh and eighth lines of data, these lines would be ignored.

If you wanted to capture the numeric data as above but with all 16 numbers in one record, you could again specify the "Start Of Record Event" as "Numeric Character Received" and then specify the "End Of Record Event" as "Special Character Received" and select the Dash ("-") character as the *special character* that signals the end of the record. Next, using the Pre-Input Character Translation Table, you would translate carriage returns and linefeeds to spaces and finally you would define the record structure as "Multiple Delimited Data Fields" with 16 fields per record and a space as the delimiter character.

Again, as long as you understand the data from your serial device and have a clear understanding of the full capabilities of the Software Wedge, you can transform almost any serial data collection problem into an extremely simple task.

Software Wedge Function Calls

Unfortunately, most programming languages provide little support for serial I/O but almost all languages (i.e. C, Pascal and QuickBASIC) incorporate an easy way to perform DOS function calls. Because of this, Wedge.Com was designed to hook into the DOS multiplex interrupt (int 2Fh) and provide a set of functions that can be called by other programs. This feature allows you to use the Wedge as a serial I/O device driver or simply control the Wedge from within your own custom applications. Function calls to the Wedge are made by putting appropriate values in CPU registers and then issuing an interrupt 2Fh.

Important: All function calls to the Wedge are made with the value CCh in the AH register, the serial port base address in BX and a function number in AL. Function 0 (Install Check) and Function 1 (transmit string) are the only functions that return a value and Function 1 (Transmit String) is the only function that requires additional parameters. If you call Wedge functions from an assembly language program or from inline assembly code in a C program, PUSH all registers onto the stack that you want to preserve before making a function call and then POP the registers off the stack after the call.

The following is a list of all functions that Wedge.Com supports:

Function 0	Install Check	Check if Wedge.Com is installed for a serial port
	Return Values:	AL=FFh if Wedge is installed otherwise AL=0
Function 1	Transmit String	Transmits a character string out the serial port
	Input Parameters:	ES:SI=String Address, CX=String Length
	Return Values:	AL=1 if string was transmitted otherwise AL=0

Note: The maximum length of an output string using Function 1 is 1023 bytes. If AL=0 on return, indicating failure, then the output buffer did not have room for the complete string. The recommended action after a failed transmit is to wait 1 second and try again.

Function 2	Toggle DTR	Lowers the serial port DTR line for 100ms
Function 3	Enable Timer	Enables the timer controlled output string
Function 4	Disable Timer	Disables the timer controlled output string
Function 5	Raise DTR	Raises the serial port DTR line
Function 6	Lower DTR	Lowers the serial port DTR line
Function 7	Raise RTS	Raises the serial port RTS line
Function 8	Lower RTS	Lowers the serial port RTS line
Function 9	Reset Wedge	Resets the Wedge & clears all buffers
Function 10	Disable Wedge	Disables the Wedge
Function 11	Enable Wedge	Enables the Wedge
Function 12	Enable 1 Record	Enables the Wedge for one input record only

FUNCTION CALL EXAMPLE FOR MS QuickBASIC:

```
'$INCLUDE: 'QB.BI'                    ' include file for QB INTERRUPT functions
```

```
DIM INREGS AS RegType             ' create CPU input register data structure
DIM OUTREGS AS RegType            ' create CPU output register data structure

INREGS.ax = &HCC00                ' AH=CCh and AL=0  (Install Check Function)
INREGS.bx = &H3F8                                 ' Put address of serial port in BX
CALL INTERRUPT(&H2F, INREGS, OUTREGS)             ' Call Interrupt 2Fh
IF OUTREGS.AX AND &HFF = 255 THEN                 ' If AL=FFh then Wedge is installed
   Test$ = "Hello World"                          ' Define string to send
   INREGS.ax = &HCC01                             ' AH=CCh, AL=1 - Transmit String
   INREGS.bx = &H3F8                              ' Put address of serial port in BX
   INREGS.cx = LEN(Test$)                         ' CX=Length of string to send
   INREGS.es = VARSEG(Test$)                      ' ES=Segment address of string
   INREGS.si = SADD(Test$)                        ' SI=Offset address of string
   CALL INTERRUPT(&H2F, INREGS, OUTREGS)          ' Call Int 2Fh to transmit the string
END IF
```

The Software Wedge also comes with a utility program called "SW-Func.Com" that performs Software Wedge function calls using command line arguments to specify a serial port and function number. The complete source code listing for SW-Func.Com (written in Assembly Language) can be found on the Software Wedge distribution diskette. See Also: SW-Func.Com and SW-Func.Asm

Software Wedge Utility Programs

The Software Wedge comes complete with the following utility programs:

File Wedge™ A version of the Software Wedge that inputs data from a disk file instead of a serial port. File Wedge consists of the two programs: FWCONFIG.EXE and FWEDGE.COM

Diagnostic Utilities

ComShow™	A Serial I/O diagnostics / Terminal / Data Logger program.
FileShow™	A powerful disk file browse utility - see *exactly* what's in a disk file.
ShoPorts™	Displays a list of all serial ports in a PC including Addresses & IRQs.

Serial I/O Utilities

SendOut™	A program that sends a text string out a serial port.
SendFile™	A program that sends a disk file out a serial port.
Set-Baud™	A program for setting a serial port baud rate to any possible value.
Set-RTS™	A program for setting the state of a serial port RTS line.
Set-DTR™	A program for setting the state of a serial port DTR line.

Serial I/O TSR Utilities

ComFile™	A TSR serial data logger (very small and fast).
Com-Key™	A TSR for creating hot keys that transmit strings out a serial port.
ComTimer™	A TSR that sends a string out a serial port at regular timed intervals.
RTS-Key™	A TSR for creating a hot key that controls a serial port RTS line.
DTR-Key™	A TSR for creating a hot key that controls a serial port DTR line.

SW-Func.Com & SW-Func.Asm

SW-Func.Com	A program that performs Software Wedge function calls using command line arguments as input parameters.
SW-Func.Asm	Assembly language source code for SW-Func.Com

File Wedge™

File Wedge is almost identical to the Software Wedge except that instead of taking input from a serial port, File Wedge reads its input from a disk file. Like the Software Wedge, the concept of File Wedge is extremely powerful because it allows you to take data from any disk file and pump the data directly into any DOS application via the keyboard buffer. Like the Software Wedge, File Wedge also has the ability parse, filter and translate data and also to write its output to a disk file instead of the DOS keyboard buffer. This means that you can even use File Wedge to convert data from one file format to another.

Just like the Software Wedge, File Wedge consists of two main programs: FWCONFIG.EXE and FWEDGE.COM.

FWCONFIG is used to create a configuration file for FWEDGE, and FWEDGE is a TSR that does the actual work of pumping the data from a disk file into the keyboard buffer or to another disk file. If you understand how to use the Software Wedge then you already know how to use File Wedge.

There are only a few minor differences between File Wedge and the Software Wedge as discussed below. For all other information about using FWCONFIG and FWEDGE, simply refer to the documentation for SWCONFIG and WEDGE.

Differences Between FWCONFIG.EXE and SWCONFIG.EXE

Since File Wedge reads input from a disk file and not from a serial port, instead of having a "PORT" sub menu in the main menu, FWCONFIG has a "INPUT" sub menu. Also, the "INPUT" sub menu contains the option "Source File Name" in place of a "Port Settings" option and instead of an "Analyze" feature, File Wedge provides a "Browse Source File" option. Selecting the "Source File Name" option opens a dialog box where you can select the name of the file containing the data that you want to read into another program or disk file. The "Browse Source File" option also invokes the utility program FileShow.Exe instead of running ComShow.Exe.

Since File Wedge does not have anything to do with serial ports, the "Automatic Output Strings" option is not available in the "Define" sub menu. Also, the only hot key actions that you can define for File Wedge are: "Reset FWedge", "Disable FWedge", "Enable FWedge" and "Enable 1 Record".

The only other major difference you will find between FWCONFIG and SWCONFIG is that there is no "Time Delay Between Records" option in the dialog box where you specify the start and end of an input record. Since data is being read from a disk file and not asynchronously through a serial port, this option would be totally meaningless.

Differences Between FWedge.Com and Wedge.Com

FWedge and Wedge are almost identical except that unlike Wedge.Com, you may only install one copy of FWedge.Com into memory at a time. This means that you can only read data from one source at a time.

The only other difference between FWedge and Wedge is that FWedge does not provide a set of function calls for programmers like the Software Wedge does. The reason that this feature was left out is because it was assumed that if you have the programming skills necessary to use function calls, then you probably have no need for File Wedge because you could simply open and read a disk file with standard DOS function calls instead.

Using File Wedge Effectively

Because File Wedge reads its input from a disk file, it does not have to wait for data from a serial device; the source data already exists in a disk file. When File Wedge is installed in "Input As Keystrokes" mode, it will start stuffing data into the DOS keyboard buffer immediately causing data to stream in at the DOS prompt right after you install it. Because of this problem, you will almost always want to install File Wedge initially disabled and then define a hot key that you can use to enable it after you have had enough time to start another application program and get to the point where you want data to be entered.

You may also want to define a hot key that you can use to disable File Wedge if things do not go as you expect. File Wedge will not stop stuffing the keyboard buffer until the entire file has been read in. If you specify a very large input file and the data is not flowing into your application program the way you want it to, you will most certainly want to have a quick and easy way to turn File Wedge off before too much damage is done.

Diagnostic Utilities

ComShow™

ComShow is a utility program similar to a terminal program that allows you to view data from any serial port in your PC. To run COMSHOW.EXE type: COMSHOW at the DOS prompt or select "Analyze" from the "Port" menu in SWCONFIG.EXE.

Like a terminal program, ComShow displays incoming serial data on your screen and provides a way to output data. Unlike a terminal program, all data received by ComShow is displayed as ASCII characters including control codes. Thus, carriage returns will appear as musical notes, etc.. This allows you to see exactly what and how many characters are being transmitted and also see their respective positions within the input stream. ComShow even has an ASCII chart that you can use to decipher the meanings of specific characters that appear in the input data.

A "settings" menu allows you to easily change COM parameters on the fly (i.e. port, baud rate, parity, etc.). This turns ComShow into a powerful serial I/O debugging utility.

ComShow provides a way to transmit data or control codes out the serial port as well as change the state of both the RTS and DTR lines. A status indicator continuously displays the state of all RS232 serial hardware lines so that you can see exactly what's going on. By enabling a "Disk File Logging" feature from within its "settings" menu, ComShow can also be used as serial data logger. If disk file logging is enabled, ComShow will automatically log all incoming serial data to a disk file named "ComShow .Log"

ComShow is extremely easy to use. Complete on-line help is available by pressing function key F1. Detailed prompts will guide you through most operations, therefore, further instructions are not necessary.

TECHNICAL NOTES:

ComShow creates a file called ComShow.INI that is used to "remember" the serial communications parameters between sessions. When you run ComShow from the DOS command line, it will first search for the ComShow.INI file and if found, the previously saved communications parameters will be used as the default at startup. If you invoke ComShow by selecting "Analyze" from the "Port" menu in SWCONFIG.EXE, ComShow will use the communications parameters that were selected in the "Port Settings" dialog box in SWCONFIG.EXE and will not allow you to directly change these settings.

ComShow also supports command line arguments to specify the communications parameters to be used at startup. The syntax for command line arguments is shown below. All command line arguments are optional and if omitted, the default value is used.

```
COMSHOW COMn:baud,parity,databits,stopbits,flowctrl  /L=filelog

where:
COMn     = COM1:, COM2:, COM3:, COM4: or &Haddr:IRQn     (default=COM1:)
baud     = 110,150,300,600,1200,2400,4800,9600 or 19200  (default=9600)
parity   = E (even), O (odd), N (none), M (mark) or S (space) (default=N)
databits = 5,6,7 or 8                                    (default=8)
stopbits = 1 or 2                                        (default=1)
flowctrl = XON, RTS    (enables flow control)            (default=None)
filelog  = ON          (enables file logging)   (default=Off)

EXAMPLE 1:   COMSHOW COM1:9600,E,7,1
EXAMPLE 2:   COMSHOW &H3F8:IRQ4,N,8,1,XON /F=ON
```

FileShow™

FileShow is a utility program that allows you to browse through the contents of a disk file. FileShow was designed for use with the File Wedge program, however it is also very useful for browsing through disk files in general. To run FILESHOW.EXE simply type: FILESHOW at the DOS prompt or select "Browse Source File" from the "Input" menu in FWCONFIG.EXE.

Unlike an editor or other file browsing programs, all data in a file is displayed as ASCII characters including control codes. Thus, carriage returns will appear as musical notes, etc.. This allows you to see exactly what and how many characters are in the file and also see their respective positions within the file. FileShow, like ComShow, has an ASCII chart that you can use to decipher the meanings of specific characters that appear in the file.

When you run FileShow from the DOS command line, FileShow allows you to enter or edit the file name and path for the file to browse by pressing function key F2.

When a file is displayed, you can scroll through the contents of the file using the cursor control keys (PgUp, PgDn, Home, End, etc.) and arrow keys.

Along with the contents of a disk file, FileShow also displays the total length of the file and the byte position of the character under the cursor as well as the ASCII value for the character under the cursor.

FileShow is extremely easy to use. Complete on-line help is available by pressing function key F1. Detailed prompts will guide you through most operations, therefore, further instructions are not necessary.

FileShow also supports an optional command line argument to specify the input file name to be browsed at startup. The syntax for command line argument is shown below. The file name command line argument is optional and if omitted FileShow simply displays nothing and waits for you to specify a file name.

```
FILESHOW filename      or      FILESHOW drive:path\filename
```

where:

filename	= the name of the file to browse
drive	= the disk drive letter where the file resides
path	= the directory path where the file resides

ShoPorts™

ShoPorts is a utility that will detect and display the true base port address (in hex) and interrupt request number (IRQ#) for each serial adapter installed in your PC.

When you run ShoPorts, the output will appear similar to that shown below:

```
  COM Port Found At Hex Address: 02F8 Tied To IRQ3,  (UART=16550A)
  COM Port Found At Hex Address: 03F8 Tied To IRQ4,  (UART=16550A)
```

The ROM BIOS stores the port addresses for the first four installed serial adapters in low memory at address 0000:0400 - 0407 however, these addresses are not always correct. Programs that "detect" serial ports by reading BIOS data may report incorrect addresses. They might also assume that your serial ports are configured using standard IRQs (i.e. COM1 tied to IRQ4, COM2 to IRQ3, COM3 to IRQ4 and COM4 to IRQ3). Again, this may be incorrect. The only real way to determine which IRQ a serial adapter uses is to actually force it to generate an interrupt and then record which interrupt occurs (as is done by ShoPorts). Also, the ROM BIOS only has room for the addresses of the first four serial ports whereas it is possible to have many more than that installed in your PC.

Note: The "Standard" addresses for COM1-4 are respectively: 3F8, 2F8, 3E8 and 2E8.

HOW SHOPORTS WORKS:

ShoPorts works by scanning the most likely serial port addresses and searching for the characteristic "signature" of an 8250,16450 or 16550 UART. If a UART is found, ShoPorts will trap IRQs 2,3,4,5,7,10,11,12 and 15 and then configure the UART to generate interrupts when a character is outputted (transmit holding register empty interrupt).
Note: ShoPorts will not trap IRQs that are currently active. See technical notes below.

After the IRQ trap is set up, a character is outputted (ASCII 0) and the corresponding interrupt, if it occurs, is recorded. Note: Because ShoPorts will output an ASCII 0 to all serial ports in your PC, you should either turn off or disconnect all serial devices (printers, networks, modems, etc.) before running ShoPorts. Serial mice are not affected by writing to them and therefore do not have to be disconnected.

ShoPorts starts scanning for serial ports at address 0100h and checks every eighth port address (i.e. addresses 0108, 0110, 0118, 0128, etc...) until it reaches address 0400. Several reserved port addresses are skipped including:

0320	Hex	Address reserved for fixed disk controller
03F0	Hex	Address reserved for floppy diskette controller
03D0 - 03D8	Hex	Addresses reserved for Color Graphics Adapter
03B0 - 03B8	Hex	Addresses reserved for Monochrome Adapter

Technical Note: When you run ShoPorts, it first performs a check to see if any of the IRQ lines that it tests are currently in use by installed TSRs or device drivers. If ShoPorts detects IRQs that are in use, it will not attempt to generate interrupts for these IRQs. It does this to avoid the possibility of an IRQ conflict and consequently a system crash. ShoPorts will also generate a message similar to the following if it detects active IRQs:

```
NOTE!  IRQ(s)  3,7,11  are currently in use on this PC.
ShoPorts will not be able to test COM ports that use these IRQs.
```

Suppose, for example, that IRQ 3 is being used by a mouse driver when you run ShoPorts. When ShoPorts tests each serial port to determine which IRQ it uses, it will not test for IRQ 3. Thus it will not be able to determine the IRQ for any ports configured for IRQ 3. If an IRQ cannot be found for a serial adapter because it uses a currently active IRQ, ShoPorts will display a question mark for the IRQ number. For this example, ShoPorts might display the following:

```
  COM Port Found At Hex Address: 02E8 Tied To IRQ?,  (UART=16550A)
  COM Port Found At Hex Address: 02F8 Tied To IRQ?,  (UART=16550A)
  COM Port Found At Hex Address: 03E8 Tied To IRQ4,  (UART=16550A)
  COM Port Found At Hex Address: 03F8 Tied To IRQ4,  (UART=16550A)
```

To free up all IRQs so that they are not in use when you run ShoPorts, simply remove all TSRs and device drivers and re-boot. The easy way to do this is to take a blank floppy and make it "bootable" with the DOS "SYS" command. Place the blank disk in drive A: and type 'SYS A:' at the DOS prompt. After DOS has been transferred to the diskette, re-boot with the disk still in drive A. This will start up your system with no device drivers or TSRs and you can re-run ShoPorts to determine the IRQs for all serial ports.

Warning: It is slightly possible for ShoPorts to lock up your PC because it reads port addresses that may be used by other peripheral devices. We highly recommend that you de-activate all peripherals and disconnect from networks before running ShoPorts. You should also exit from all programs including DOS Shell programs and multitasking environments (i.e. Windows), and run ShoPorts only from the native DOS prompt.

Serial I/O Utilities

SendOut^{TM}

SendOut is a utility that will send character strings out a serial port or toggle the serial port DTR line for 100ms. SendOut can be used to send prompts or configuration data to a serial device, dial out on a modem, send control codes to a serial printer, etc..

SendOut uses command line arguments to specify the serial port and the character string to send and thus it can be used at the DOS prompt, from within a batch file, or it can be called by other programs such as Lotus 123, dBase, etc.. All command line arguments are specified using the following syntax: (items in square brackets are optional)

```
SENDOUT COMn:[n,]['text'][,n] ,etc... or SENDOUT &Haddress:['text'][,n][,n][,'text'] ,etc...
```

The first argument "COMn" or "&Haddress" is the serial port that you want to use. The remaining part of the command line after the semicolon contains the data that you want sent out the serial port. If no data is specified and only a serial port is specified, then the serial port's DTR signal line will be toggled from "ON" to "OFF" for 100ms.

SendOut does not have the ability to set any communications parameters (i.e. baud rate, parity, databits, etc.) nor will it disturb any parameters that may be in effect when SendOut is run. If you need to set or change any communications parameters before using SendOut , you can always use the DOS 'MODE' command. See your DOS users guide for details on the MODE command.

Note: Do not use the DOS MODE command to set communications parameters for a serial port that the Software Wedge or ComFile is installed for. Both of these programs set communications parameters directly and using the MODE command will disable the Wedge or ComFile.

SPECIFYING THE SERIAL PORT

If you use the form "COMn:", you may specify either COM1:, COM2:, COM3: or COM4: The form "&Haddress" allows you to specify a serial adapter by supplying it's base port address in hexadecimal. For example to send data out a serial port at address 0100(hex), you would use the form &H0100: The semicolon in either of the above forms is required.

SPECIFYING OUTPUT DATA

All text to be sent out the port must be enclosed in single quotes. To send control codes or single quote marks (ASCII 39), you can specify the ASCII code number ("n") that represents the character that you want to send. For Example, to send a carriage return followed by a text string ("testing 123") followed by a carriage return linefeed, you would use the following example:

```
SENDOUT COM1:13,'testing 123',13,10
```

The next example shows how to send a string with embedded single quote marks.
(The ASCII value for a single quote character is 39)

```
SENDOUT COM1:39,'Four Score & Seven Years Ago',39
```

When sending multiple control codes or any combination of text and control codes, you must delimit each item from the next with a single comma. For example, to send two strings ('test1' & 'test2') with a carriage return after each string, use the following form:
(this example sends data to a serial port at the hex address 3F8)

```
SENDOUT &H3F8:'test1',13,'test2',13
```

SendFile™

SendFile is a program that allows you to send a disk file out any serial port on your PC. SENDFILE.COM uses command line arguments with the following syntax:
(items in square brackets are optional.)

```
SENDFILE COMn:filename [/C]    or     SENDFILE &Haddress:filename [/C]
```

The first command line argument "COMn" or "&Haddress" represents the serial port that you want to use. The filename parameter after the semicolon is the name of a disk file that contains data that you would like sent out the serial port. The /C parameter, if specified, enables RTS/CTS flow control. If no serial port or filename is specified, then SENDFILE simply exits without any error message.

SendFile does not have the ability to set any communications parameters (i.e. baud rate, parity, databits, etc.) nor will it disturb any parameters that may be in effect when SendFile is run. If you need to set or change any communications parameters before using SendFile , you can always use the DOS 'MODE' command. See your DOS users guide for details on the MODE command.

Note: Do not use the DOS MODE command to set COM parameters for a serial port that the Software Wedge or ComFile is installed for. Both of these programs set COM parameters directly and using the MODE command will disable the Wedge or ComFile.

SPECIFYING THE SERIAL PORT

If you use the form "COMn:", you may specify either COM1:, COM2:, COM3: or COM4: The form "&Haddress" allows you to specify a serial adapter by supplying its base port address in hexadecimal. For example to send data out a serial port at address 0100(hex), you would use the form &H0100: The semicolon in either of the above forms is required.

SPECIFYING THE FILE TO SEND

The filename specified can be either a simple file name indicating a file in the current directory or a complete path and filename as in the following examples:

```
SENDFILE COM1:MyFile.TXT
SENDFILE COM1:C:\DOS\DATA\MyFile.TXT /C
SENDFILE &H3F8:Serial.DAT
```

Set-Baud™

SET-BAUD is a utility program that allows you to set the baud rate for any serial port to any value from 50 to 115200 baud. SET-BAUD.COM uses command line arguments with the following syntax: (items in square brackets are optional)

```
SET-BAUD COMn:[divisor]    or    SET-BAUD &Haddress:[divisor]
```

The first command line argument "COMn" or "&Haddress" represents the serial port that you want to set the baud rate for. The remaining part of the command line after the semicolon contains an integer "divisor" used to specify the baud rate. The reason a divisor is used instead of an actual baud rate is because the UART chip in a serial adapter uses a divisor value and thus only supports a finite and discreet set of baud rate values based on the divisor.

SPECIFYING THE SERIAL PORT

If you use the form "COMn:", you may specify either COM1:, COM2:, COM3: or COM4: The form "&Haddress" allows you to specify a serial adapter by supplying it's base port address in hexadecimal. For example to send data out a serial port at address 0100(hex), you would use the form &H0100: The semicolon in either of the above forms is required.

SPECIFYING THE BAUD RATE DIVISOR

To calculate the divisor value, use the formula: divisor = 115200/desired baud rate

Note: Only integer values can be used for the divisor. If the desired baud rate does not divide evenly into 115200, truncate the resulting divisor to the closest integer value and use that value instead. If the divisor is not specified, the value 1 will be used by default.

Example values:

Baud Rate	Divisor	Baud Rate	Divisor	Baud Rate	Divisor
115200	1	9600	12	2400	48
56000	2	7200	16	2000	58
38400	3	4800	24	1800	64
19200	6	3600	32	1200	96

EXAMPLE:

To set COM1 to 19200 baud, use the command: SET-BAUD COM1:6
To set a port at address 2F8 to 50 baud, use the command: SET-BAUD &H2F8:2304

Set-DTR™ and Set-RTS™

Set-DTR.COM and Set-RTS.COM are utility programs that allows you to set the DTR line (Data Terminal Ready) and RTS (Request To Send) lines on or off for any serial port.

Set-DTR and Set-RTS use command line arguments with the following syntax:
(items in square brackets are optional)

```
SET-DTR COMn:[OFF]      &   SET-RTS COMn:[OFF]
```

or

```
SET-DTR &Haddress:[OFF]    &     SET-RTS &Haddress:[OFF]
```

The first command line argument "COMn" or "&Haddress" represents the serial port that you want control. The remaining part of the command line after the semicolon (if specified) indicates the state of the DTR or RTS line (either ON or OFF). If the OFF argument is not specified, it is assumed that you want to turn the line ON.

SPECIFYING THE SERIAL PORT

If you use the form "COMn:", you may specify either COM1:, COM2:, COM3: or COM4: The form "&Haddress" allows you to specify a serial adapter by supplying its base port address in hexadecimal. For example to send data out a serial port at address 0100(hex), you would use the form &H0100: The semicolon in either of the above forms is required.

EXAMPLES:

To set the DTR line ON for COM1, use the command: SET-DTR COM1:
To set the RTS line OFF for COM2, use the command: SET-RTS COM2:OFF
To set DTR OFF for a serial port at hex address &H2F8, use the command:
SET-DTR &H2F8:OFF

Serial I/O TSR Utilities

ComFile™

ComFile is a TSR utility that will capture incoming serial data and store it to a disk file or device while you run other programs on your PC. ComFile captures all serial data to disk exactly as it is received. All communications parameters and the name of the file or device to save data to are specified with command line arguments. Although the Software Wedge has the ability to log data to a disk file, ComFile uses about one third the memory of the Software Wedge because it does not have any parsing or filtering capabilities. In other words, ComFile is purely a background "data logger".

ComFile can not only be used to capture serial data to disk, but it can also be used to pass incoming serial data to another device including a printer or even another serial port. This means that you could even use ComFile as a simple printer sharing program allowing two or more PCs to share a single printer!

ComFile is a TSR or "memory resident program" which means that after you run ComFile, it stays in memory and continues to function in the background while you run other programs in the foreground. ComFile uses less than 4k of memory and it can be removed from memory with the command: ComFile/R

NOTE: Serial devices that normally send output to a printer (including other PC's) may require a "nul modem adapter" (available for about $5.00 at Radio Shack) if they are to be connected to a serial port on a PC. See Also: Troubleshooting

INSTALLING COMFILE

When you install ComFile you use command line arguments to specify the serial port, baud rate, parity, number of data bits, number of stop bits, type of flow control to use, and also the file or device name where serial input data is to be sent. The command line syntax is as follows:

```
COMFILE port:baud,parity,databits,stopbits [flow control] [/f=path]
```

Example: `ComFile COM2:9600,E,7,1 /X /F=C:\MyFile.DAT`

THE PORT PARAMETER

The 'port' parameter tells ComFile which serial adapter to use and can be specified in either the form COMn: where n represents the port number (i.e. COM1:, COM2:, COM3: or COM4:) or the form &Haddress:IRQn: where 'address' is the base port address in hexadecimal of the serial adapter that you would like to use and 'n' is the IRQ number for the serial adapter (ex. &H3F8:IRQ4:). Note: ComFile only supports IRQ's 2-5

THE BAUD PARAMETER

The 'baud' parameter specifies the baud rate to use. ComFile supports:
110, 150, 300, 600, 1200, 2400, 4800, 9600 and 19200 baud.

THE PARITY PARAMETER

The 'parity' parameter specifies the type of parity to use. ComFile supports No parity, Even parity, Odd parity, Mark parity and Space parity. Use the first letter of the parity type to specify this parameter. (i.e.. N = No parity, E = Even parity, etc..)

THE DATABITS PARAMETER

The 'databits' parameter specifies the number of data bits in each byte of serial data. ComFile supports 5,6,7 or 8 databits.

THE STOPBITS PARAMETER

The 'stopbits' parameter specifies the number of stop bits used to delimit each byte of serial data. ComFile supports 1 or 2 stop bits. Note: If you select 5 for the number of data bits, ComFile will automatically use 1.5 stopbits.

THE FLOW CONTROL PARAMETER

ComFile supports either No flow control, XON/XOFF flow control, or RTS/CTS flow control. If the flow control parameter is omitted, no flow control is used. To specify XON/XOFF, use the command line argument: /X. To specify RTS/CTS flow control, use the argument: /C.

THE PATH PARAMETER

The 'path' parameter is used to specify the full path name for the file that you want to save data to or it can be the name of an installed DOS device (i.e. PRN etc.) If you do not specify a file or device name, then ComFile will use the default path: \ComFile.LOG
NOTE: You must specify the compete path name for a disk file including the drive and directory path in order for ComFile to work correctly. If the drive or path name is left out, whenever you change directories with a foreground program, ComFile will use the new default directory and you will end up with a data file in each directory.

EXAMPLES:

```
ComFile COM2:1200,E,7,1 /C /F=C:\TEST
```

```
ComFile &H3F8:IRQ4:19200,N,8,1 /F=C:\DOS\COMMDATA.DAT
```

TECHNICAL DETAILS ABOUT ComFile

ComFile may be installed for each serial adapter available on your PC as long as the IRQ (interrupt request line) for each adapter is unique to the adapter. For example, COM1 and COM3 normally use the same IRQ number therefore you cannot install ComFile for both COM1 and COM3 at the same time. See Also: Troubleshooting

ComFile uses approx. 3900 bytes of memory for each serial port that you install it for.

ComFile uses a 1kb internal serial input buffer. Every 18th of a second, ComFile will check its input buffer to see if it contains any data. If it does not contain any data, then it simply does nothing. If there is any data in the buffer, ComFile will open the specified disk file, write the data from the buffer to the end of the file and then immediately close the file. If the specified file does not exist when ComFile tries to open it, ComFile will create it. Because ComFile does not keep the file open continually, you should be able to read from the file in a foreground program even while ComFile is writing to it.
See Also: The "SHARE" command in your DOS manual.

Com-Key^{TM}

COM-KEY is a TSR that allows you to define a character string to be sent out your serial port whenever you press a "hot key" on your PC's keyboard. All parameters including the serial port, hot key, and the data string are specified with command line arguments.

COM-KEY is a TSR or "memory resident program" which means that after you run COM-KEY, it stays in memory while you run other programs like Lotus 123 or dBASE, etc. thus when you run COM-KEY, you install it in your PC's memory and it is always available.

COM-KEY was designed for use with serial (RS232) devices that accept commands or prompt strings to control their operation. (i.e. modems, printers, electronic scales, measuring instruments, etc.) For example, you could use COM-KEY to dial a frequently called telephone number through your modem at the press of a key or you could use it to send control codes to a serial printer. Many electronic devices can be instructed to transmit a data reading by sending them a prompt string and some types of cash drawers can be triggered to open by sending them a character through a serial port.

COM-KEY does not have the ability to set any serial COM parameters (i.e. baud rate, parity, etc.) nor will it disturb any settings that are in effect when your hot key is pressed. If you need to set or change COM parameters before using COM-KEY, you can use the DOS 'MODE' command. See your DOS manual for details on the MODE command.

Note: Do not use the DOS MODE command to set COM parameters for a serial port that the Software Wedge or ComFile is installed for. Both of these programs set COM parameters directly and using the MODE command will disable the Wedge or ComFile.

INSTALLING COM-KEY

When you install COM-KEY you specify the serial port, hot key, and string to send on the DOS command line using the following syntax: COM-KEY port{hotkey} 'data string'
Each of the three parameters: port, hotkey, and 'data string' have variations in the way they can be specified as outlined below.

THE PORT PARAMETER

The 'port' parameter tells COM-KEY which serial adapter to use and can be specified in either the form COMn: where n represents the port number (i.e. COM1:, COM2:, COM3: or COM4:) or the form &Haddress where address is the base port address in hexadecimal of the serial adapter that you would like to use (ex. &H3F8).

THE HOTKEY PARAMETER

The hot key parameter has the following three variations:

{@char}, {^char} or {shiftmask,scancode}

The first variation {@char} specifies that the ALT key is to be held down while the specified character (char) is pressed. For example, specifying {@P} means that ALT-P is the desired hot key. NOTE: each variation must be enclosed in curly braces.

The second variation {^char} specifies that the CTRL key is to be held down while the specified character (char) is pressed. Thus specifying {^A} means that CTRL-A is the desired hot key. (The character '^' is a shifted '6' on most PC keyboards.)

The third variation {shiftmask,scancode} allows you to select virtually any key combined with any of the toggle keys: CTRL, ALT, Right-Shift or Left-Shift. The parameters shiftmask and scancode are entered as numbers separated by a comma. The shiftmask parameter represents a combination of toggle keys and the scancode represents the keypress to use in combination with the toggle keys. For details about specifying hot keys using shiftmask and scancode values, refer to the section: "Hot Key Look Up Tables". This section contains look up tables for both shiftmask values and scancode values.

THE 'DATA STRING' PARAMETER

The 'data string' parameter specifies the character string to send out the serial port. The data string that you specify may contain both ASCII codes (for characters that cannot be typed on your keyboard) and character strings enclosed in single quotes. ASCII code values may range from 0 to 255. Any ASCII values in the data string must be separated from other ASCII values or quoted strings with commas or spaces. For example, the following command causes the string "Test 123" followed by a carriage return (ASCII 13) and a line feed (ASCII 10) to be sent out COM1: when the hot key CTRL+Q is pressed:

```
COM-KEY COM1:{^Q}'Test 123',13,10
```

NOTE: The 'data string' parameter is optional. If you do not specify a 'data string', COM-KEY will toggle the serial port Data Terminal Ready line (DTR) from "on" to "off" for approximately 100ms whenever you press your hot key. Toggling the DTR line for 100ms is interpreted by certain types of electronic instruments as a request for data.

TECHNICAL DETAILS ABOUT COM-KEY

COM-KEY may be installed as many times as you like and thus you may assign a different output data string to any possible number of hot key combinations. It uses approx. 700 bytes of memory for each hot key that you install it for.

COM-KEY can be used with all commercial DOS programs including other TSR programs and serial communications programs like The Software Wedge, ComShow, and ComFile.

REMOVING COM-KEY FROM MEMORY

To deactivate and remove COM-KEY from your computer's memory simply run COM-KEY using the command: COM-KEY/R

This will remove the most recently loaded copy of COM-KEY from memory and release that memory to DOS for use by other programs. If you loaded another hot key activated program after you loaded COM-KEY, the removal command will not work until you remove the other program from memory. If you load multiple copies of COM-KEY, you will have to issue the removal command once for each copy that you loaded to remove all copies. Multiple copies are removed from memory in the reverse order that they were loaded in originally.

ComTimer™

ComTimer is a TSR utility program that allows you to define a character string that is sent out a serial port at regular timed intervals or whenever you press a "hot key" on your PC keyboard. A typical use for ComTimer is to automatically prompt a serial device like a scale or measuring instrument at regular intervals thus causing the device to transmit data readings at regular timed intervals.

A "hot key" is used to activate and deactivate ComTimer. All parameters including the serial port to use, timer interval, hot key, and data string to send are fully selectable using command line arguments.

ComTimer also supports an inter-character delay function that can be used to cause a small time delay between the output of each character.

ComTimer is a TSR or "memory resident program" which means that after you run ComTimer, it stays in memory while you run other programs like Lotus 123 or dBASE, etc. When you run ComTimer, you essentially install it in your PC's memory and it is always available for use.

ComTimer can be set up to transmit data at intervals as fast as 18 times a second or as slow as once an hour.

ComTimer does not have the ability to set any serial COM parameters (i.e. baud rate, parity, etc.) nor will it disturb any settings that are in effect when your hot key is pressed. If you need to set or change COM parameters before using ComTimer, you can use the DOS 'MODE' command. See your DOS manual for details on the MODE command.

Note: Do not use the DOS MODE command to set COM parameters for a serial port that the Software Wedge or ComFile is installed for. Both of these programs set COM parameters directly and using the MODE command will disable the Wedge or ComFile.

INSTALLING COMTIMER

When you install ComTimer you specify the serial port, hot key, timing interval and string to send on the DOS command line using the following syntax:

```
COMTIMER port{hotkey,ticks,delay} 'data string'
```

The parameters: port, hotkey, and 'data string' have variations in the way they can be specified as outlined below. The "ticks" parameter must be specified as a number ranging from 0 to 65535. NOTE: the hotkey, ticks, and InterCharacterDelay parameters must be enclosed in curly brackets.

THE PORT PARAMETER

The 'port' parameter tells ComTimer which serial adapter to use and can be specified in either the form COMn: where n represents the port number (i.e. COM1:, COM2:, COM3: or COM4:) or the form &Haddress where address is the base port address in hexadecimal of the serial adapter that you would like to use (ex. &H3F8).

THE HOTKEY PARAMETER

The hotkey that you choose is used to enable and disable ComTimer. When ComTimer is installed, it is initially disabled until the specified hotkey is pressed. The same hotkey is also used to disable ComTimer. The hotkey parameter has the following three variations:

@char, ^char or shiftmask,scancode

The first variation @char specifies that the ALT key is to be held down while the specified character (char) is pressed. For example, specifying @P means that ALT-P is the desired hot key. The second variation ^char specifies that the CTRL key is to be held down while the specified character (char) is pressed. Thus specifying ^A means that CTRL-A is the desired hot key. (The character '^' is a shifted '6' on most PC keyboards.)

The third variation "shiftmask,scancode" allows you to select virtually any key combined with any of the toggle keys: CTRL, ALT, Right-Shift or Left-Shift. The parameters shiftmask and scancode are entered as numbers separated by a comma. The shiftmask parameter represents a combination of toggle keys and the scancode represents the keypress to use in combination with the toggle keys. For details about specifying hot keys using shiftmask and scancode values, refer to the section: "Hot Key Look Up Tables".

THE TICKS PARAMETER

The value that you supply for the "ticks" parameter controls how often ComTimer sends the data string out the serial port. The "ticks" parameter must be specified as an integer number between 0 and 65535 and represents the number of clock ticks between outputs. ComTimer uses your PC's system clock which registers 18.2 clock ticks per second. To set ComTimer to send a string out the serial port approximately once every second, you would specify 18 as the value for "ticks". Use the formula: ticks = 18.2 x (desired # of seconds between outputs). If you specify 0 for the ticks parameter, automatic output is disabled and ComTimer will only transmit the output string when you press the hot key.

THE DELAY PARAMETER

The value for this parameter specifies the amount of time that ComTimer should wait between the output of each character in the output string (the inter-character delay). The allowable range of values for this parameter is 0 to 18 and it also represents a number of clock ticks, thus 1 means wait 1/18th of a second between characters, etc... If you enter a number larger than 18, then the value 18 is used by default.

NOTE: The total number of characters in the output string times the value supplied for the InterCharacterDelay must be less that the value entered for the TICKS parameter in order for ComTimer to work correctly.

THE 'DATA STRING' PARAMETER

The 'data string' parameter specifies the character string to send out the serial port. The data string may contain both ASCII codes and character strings enclosed in single quotes. ASCII code values may range from 0 to 255. Any ASCII values in the data string must be separated from other ASCII values or quoted strings with commas. For example, the following command causes the string "Testing 123" followed by a carriage return (ASCII 13) and a line feed (ASCII 10) to be sent out COM1: once every 5 seconds (with an intercharacter delay of 1/18th of a second) after the hot key CTRL+Q is pressed to activate ComTimer:

```
COMTIMER COM1:{^Q,91,1}'Testing 123',13,10
```

The equivalent command using the shiftmask,scancode variation to specify ^Q as the hotkey is: COMTIMER COM1:{4,16,91,1}'Testing 123',13,10

NOTE: The 'data string' parameter is optional. If you do not specify a 'data string', ComTimer will toggle the serial port Data Terminal Ready line (DTR) from "on" to "off" for approximately 100ms at regular intervals specified by the "ticks" parameter. Toggling the DTR line for 100ms is interpreted by certain types of electronic instruments as a request for data. If DTR toggling is used, the minimum number for the ticks parameter is 4. This is because the DTR line is toggled for 100ms which is approximately 2-3 clock ticks.

TECHNICAL DETAILS ABOUT COMTIMER

ComTimer may be installed as many times as you like and thus you can have several different data strings being sent out either the same serial port or different serial ports at different timing intervals. If you load more than one copy of ComTimer with each copy using the same timing interval (either sending data out the same serial port or out different serial ports), the data strings for each installation of ComTimer will be sent out the serial port in the same order that each copy of ComTimer was installed. NOTE: You may not use the same hotkey to activate/deactivate more than one installation of ComTimer, thus each installation of ComTimer must use a different hotkey. When ComTimer senses its own hotkey, the keystroke is not passed on to any other programs that may also be looking for the same hot key thus you must take care to not use the same hotkey that is used by any other TSRs that you may be using including previously loaded copies of ComTimer, the Software Wedge or Com-Key.

ComTimer uses approximately 1200 bytes of memory for each copy that you install.

REMOVING COMTIMER FROM MEMORY

To remove ComTimer from your computer's memory simply run ComTimer using the command: COMTIMER/R
This will remove the most recently loaded copy of ComTimer from memory and release that memory to DOS for use by other programs.

If you load another hot key activated program after ComTimer, the removal command will not work until you remove the other program from memory. If you load multiple copies of ComTimer, you will have to issue the removal command once for each installed copy.

RTS-Key™ & DTR-Key™

RTS-KEY and DTR-KEY are TSR utilities that allows you to define a "hot key" on your PC's keyboard that can be used to toggle the RTS or DTR lines on a serial port on or off. RTS-KEY and DTR-KEY are identical programs except that RTS-KEY controls the RTS (Request To Send) line and DTR-KEY controls the DTR (Data Terminal Ready) line.

RTS-KEY & DTR-KEY simply toggle the state of their respective serial port hardware lines when you press the hot key assigned to them. If the line is on when the hot key is pressed, it will be turned off. Likewise, if the line is off when the hot key is pressed, it will be turned on. All parameters including the serial port to use and the hot key are fully selectable using simple command line arguments.

RTS-KEY & DTR-KEY are TSRs or "memory resident programs" which means that after you run them, they remain in memory while you run other programs like Lotus 123 or dBASE, etc.. Thus, when you run RTS-KEY or DTR-KEY, you essentially install them in your PC's memory and they are always available.

INSTALLING RTS-KEY or DTR-KEY

When you install RTS-KEY or DTR-KEY you specify the serial port and the hot key on the DOS command line using the following syntax:

```
RTS-KEY port{hotkey}     or     DTR-KEY port{hotkey}
```

Each of the parameters: port and hotkey have variations in the way they can be specified as outlined below.

THE PORT PARAMETER

The 'port' parameter tells RTS-KEY or DTR-KEY which serial adapter to use and can be specified in either the form COMn: where n represents the port number (i.e. COM1:, COM2:, COM3: or COM4:) or the form &Haddress where address is the base port address in hexadecimal of the serial adapter that you would like to use (ex. &H3F8).

THE HOTKEY PARAMETER

The hotkey parameter has the following three variations:

{@char}, {^char} or {shiftmask,scancode}

The first variation {@char} specifies that the ALT key is to be held down while the specified character (char) is pressed. For example, specifying {@P} means that ALT-P is the desired hot key. NOTE: each variation must be enclosed in curly braces.

The second variation {^char} specifies that the CTRL key is to be held down while the specified character (char) is pressed. Thus specifying {^A} means that CTRL-A is the desired hot key. (The character '^' is a shifted '6' on most PC keyboards.)

The third variation {shiftmask,scancode} allows you to select virtually any key combined with any of the toggle keys: CTRL, ALT, Right-Shift or Left-Shift. The parameters shiftmask and scancode are entered as numbers separated by a comma. The shiftmask parameter represents a combination of toggle keys and the scancode represents the keypress to use in conjunction with the toggle keys. For details about specifying hot keys using shiftmask and scancode values, refer to the section: "Hot Key Look Up Tables". This section contains look up tables for both shiftmask values and scancode values. Use these tables to obtain the necessary values for your desired hot key.

EXAMPLE:

The following command installs RTS-KEY for serial port COM1 and uses the hotkey combination Ctrl-R to toggle the RTS line on COM1: RTS-KEY Com1:{^R}

TECHNICAL DETAILS ABOUT RTS-KEY & DTR-KEY

RTS-KEY or DTR-KEY may be installed as many times as you like. Either program uses approximately 700 bytes of memory for each installed copy.

REMOVING RTS-KEY or DTR-KEY FROM MEMORY

To deactivate and remove RTS-KEY or DTR-KEY from your computer's memory simply run either program using the command: RTS-KEY/R or DTR-KEY/R

Hot Key Look Up Tables

Most of the TSR serial I/O utilities that are provided with the Software Wedge allow you to specify a hot key to perform specific actions. Hot keys for these utilities may be specified by supplying a ShiftMask value and a ScanCode value. Use the following tables to obtain the necessary values for a desired hot key.

SHIFTMASK LOOK UP TABLE

Use the following table to calculate the desired hot key shiftmask value:

Toggle Key Combination	Shiftmask Value
None (No Toggle Keys)	**0**
Right Shift	**1**
Left Shift	**2**
Right Shift + Left Shift	**3**
Ctrl	**4**
Ctrl + Right Shift	**5**
Ctrl + Left Shift	**6**
Ctrl + Right Shift + Left Shift	**7**
Alt	**8**
Alt + Right Shift	**9**
Alt + Left Shift	**10**
Alt + Right Shift + Left Shift	**11**
Alt + Ctrl	**12**
Alt + Ctrl + Right Shift	**13**
Alt + Ctrl + Left Shift	**14**
Alt + Ctrl + Right Shift + Left Shift	**15**

NOTE: shiftmask value must be in the range 1 to 15

KEYBOARD SCANCODE LOOK UP TABLE

Use the following table to determine a desired hot key's scan code:

Key	**ScanCode**	**Key**	**ScanCode**	**Key**	**ScanCode**	**Key**	**ScanCode**
ESC	**1**	**U**	**22**	**\ |**	**43**	**F6**	**64**
1 !	**2**	**I**	**23**	**Z**	**44**	**F7**	**65**
2 @	**3**	**O**	**24**	**X**	**45**	**F8**	**66**
3 #	**4**		**P**	**25**	**C**	**46**	**F9**
	67						
4 $	**5**	**[{**	**26**	**V**	**47**	**F10**	**68**
5 %	**6**	**] }**	**27**	**B**	**48**	**NUM**	**69**
6 ^	**7**	**ENTER**	**28**	**N**	**49**	**SCROLL**	**70**
7 &	**8**	**CTRL**	**29**	**M**	**50**	**HOME**	**71**
8 *	**9**	**A**	**30**	**, <**	**51**	**UP**	**72**
9 (	**10**	**S**	**31**	**. >**	**52**	**PGUP**	**73**
0)	**11**	**D**	**32**	**/ ?**	**53**	**GREY -**	**74**
- _	**12**	**F**	**33**	**R SHIFT**	**54**	**LEFT**	**75**
= +	**13**	**G**	**34**	*** PRTSC**	**55**	**CENTER**	**76**
BKSP	**14**	**H**	**35**	**ALT**	**56**	**RIGHT**	**77**
TAB	**15**	**J**	**36**	**SPACE**	**57**	**GREY +**	**78**
Q	**16**	**K**	**37**	**CAPS**	**58**	**END**	**79**
W	**17**	**L**	**38**	**F1**	**59**	**DOWN**	**80**
E	**18**	**; :**	**39**	**F2**	**60**	**PGDN**	**81**
R	**19**	**' "**	**40**	**F3**	**61**	**INS**	**82**
T	**20**	**` ~**	**41**	**F4**	**62**	**DEL**	**83**
Y	**21**	**L SHIFT**	**42**	**F5**	**63**		

TECHNICAL NOTES ABOUT HOT KEYS:

Your PC's keyboard sends scancode numbers to your PC whenever a key is either pressed or released. The scancode values above are sent when the key is pressed. The values sent when the key is released are simply the values shown above plus 128. Since DOS only generates a keystroke when a key is pressed and, (with the exception of toggle keys), ignores when it is released and because the release of a key is always preceded by a press, you can cause any of the TSR utilities to pass its own hot key to either a previously loaded TSR or to a foreground program by adding 128 to the scancode values in the table above. This causes the utility to react to the release of the hot key instead of the press and allows the press to pass through unaffected.

When any of the TSR utilities in the Software Wedge package detect their own hot key, they discard the hot key keystroke so that it is not passed on to the foreground application program or to any previously loaded TSR programs. All other keystrokes are passed on to previously loaded TSRs and/or the foreground program.

SW-Func.Com and SW-Func.Asm

SW-Func.Com is a utility program that can be used to perform Software Wedge function calls from the DOS command line, from a batch file or by running SW-Func from within an application program. SW-Func uses command line arguments to specify the serial port for an installed copy of the Wedge, the number of the function to be called and also an output string if the "Transmit String" function is called. SW-Func was designed to provide a way to execute Software Wedge function calls from within an application using a "Shell" or "Run" command provided by the application program. The complete source code listing for SW-Func is also provided in the file SW-Func.Asm.

All command line arguments are specified using the following syntax:
(items in square brackets are optional)

```
SW-FUNC COMn:{function} [n,]['text'][,n] ,...   or
SW-FUNC &Haddress:{function} ['text'][,n][,n][,'text'],...
```

The first argument "COMn" or "&Haddress" is the serial port that is being used by an installed copy of the Software Wedge. The "function" argument is the function number of the Software Wedge function call that you want to execute. The remaining part of the command line after the function number contains optional data that you want sent out the serial port if function number 1 (transmit string) is the desired function.

SPECIFYING THE SERIAL PORT

If you use the form "COMn:", you may specify either COM1, 2, 3 or 4. The form "&Haddress" allows you to specify a serial port by supplying it's base port address in hex. For example to specify a Wedge installed for a serial port at address 0100(hex), you would use the form &H0100: The semicolon in either of the above forms is required. Note: The Software Wedge must be installed for the specified serial port for SW-Func to work. If the Wedge is not installed, SW-Func will do nothing and report no error message.

SPECIFYING THE FUNCTION NUMBER

The function number must be specified inside curly brackets with no spaces between the brackets and the function number. SW-Func can be used to perform all Software Wedge function calls except function 0 (Install Check). For detailed information about all Software Wedge functions, refer to the section: Software Wedge Function Calls

SPECIFYING OUTPUT DATA FOR FUNCTION #1 (Transmit String)

All text to be sent out the port must be enclosed in single quotes. To send control codes or single quote marks (ASCII 39), you can specify the ASCII code number ("n") that represents the character that you want to send. For Example, to send a carriage return followed by a text string ("testing 123") followed by a carriage return linefeed, you would use the following example: SW-FUNC COM1:{1} 13,'testing 123',13,10

The next example shows how to send a string with embedded single quote marks.

```
SW-FUNC COM1:{1} 39,'Four Score & Seven Years Ago',39
```

When sending multiple control codes or any combination of text and control codes, you must delimit each item from the next with a single comma. For example, to send two strings ('test1' & 'test2') with a carriage return after each string, use the following form:

```
SW-FUNC &H3F8:{1} 'test1',13,'test2',13
```

INDEX