WAVETEK

USER'S HANDBOOK

# WaveForm DSP2

## Arbitrary Waveform Software

# User's Handbook

for

# WaveForm DSP2
# Arbitrary Waveform Software

850322 Issue 1.0 (September 1998)
For use with WaveForm DSP2 Version 2.0

**ISO 9002**
**Wavetek Ltd**
CERTIFICATE
No. FM 29700

For any assistance contact your nearest Wavetek Sales and Service Center.
Addresses can be found at the back of this handbook.

## Wavetek Limited Software Warranty

Wavetek Ltd. warrants these components of WaveForm DSP2 for 90 days from the date of purchase:

- The WaveForm DSP2 disks are free from defects in material and workmanship, assuming normal use, for the warranty period.
- The manual contains the information that Wavetek Ltd. deems necessary to operate the software.
- The software functions substantially as described in the manuals.

Except as provided above, Wavetek Ltd. provides no warranties, expressed or implied, including, but not limited to, any implied warranty of merchantability or fitness for a particular purpose. Wavetek Ltd. shall not be liable for any direct, indirect, incidental, or consequential damages, including, without limitation, loss of anticipated profits or benefits resulting from the use of, or inability to use, the product, or arising out of the breach of any warranty even if Wavetek Ltd. has been advised of the possibility of such damages. In no event shall such liability or damages exceed the purchase price of the product.

## Copyright Notice



## Trademark Acknowledgements

Wavetek is a registered trademark of Wavetek Corp.
Windows is a trademark of Microsoft Corp.
MathViews is a registered trademark of The MathWizards, Inc.
IBM is a registered trademark of International Business Machines Corp.

# Contents

# 1 Introduction

Your new WaveForm DSP2 Arbitrary Waveform Generator Software Package provides a friendly, convenient interface for compatible arbitrary waveform generators (ARBs). With WaveForm DSP2 you will find it easier to manage, modify, and create waveforms.

You can use Waveform DSP2 to create new waveforms in six different ways:

- by entering their mathematical representations, for interpretation by the program's equation parser
- by uploading them from compatible DSOs
- by drawing them with the programs graphic interface
- by combining all or parts of existing waveforms using simple editing commands
- by mathematically combining the expressions of existing waveforms
- by combining all of the above methods

Your editing tools include deleting, splicing, adding, subtracting, multiplying, and concatenating. All of these operations are performed at the computer.

The program supports external ARBs and DSOs through an IEEE488 interface, an RS232 interface, or a VXI-Visa interface.
Waveform DSP2 uses a number of editable script files for the control of data transfers with the selected DSO and/or ARB, thus other DSOs may be added to the list of supported instruments by creating a script file and adding it to the list in the ASCII configuration file 'dsp2u.cfg'.

Waveforms can also be moved to and from other applications e.g. Microsoft Excel using the Windows clipboard.

WaveForm DSP2 runs under the Windows 95 and Windows NT operating systems.

## 1.1 Recommended PC Specification

We recommend a computer that meets or exceeds these specifications for running WaveForm DSP2:

**Host Computer:** 100% IBM PC compatible with 486 processor 16 Mbytes of memory.
**Operating System:** Microsoft Windows 95 or Windows NT Version 4.0.
**Hard Disk:** 8 Mbyte unused capacity.
**Floppy Disk Drives:** One 3.5-inch, 1.44 Mbyte capacity drive.
**Monitor:** SVGA colour graphics.

**IEEE488 Interface Card:** The National Instruments PCII/PCIIA

The recommended communication with ARBs or DSOs is via an IEEE488 interface card in the PC, although provision has been made for RS232 communication with benchtop ARBs where applicable.

For communication with VXI ARBs the PC listed above may be embedded in the VXI rack. Alternatively an IEEE488 to VXI card may be used in the rack with an IEEE488 interface card in the PC, or an MXI to VXI card may be used in the rack with a MXI interface card in the PC.

When WaveForm DSP2 is started it detects the presence of a GPIB.DLL and VISA.DLL to establish the communication capabilities.

## 1.2 Compatible ARBs

The following ARBs are currently supported by the WaveForm DSP2 software:-

| Manufacturer | Model |
|---|---|
| Wavetek | Model 29 |
| | Model 39 |
| | Model 295 (32K, 128K or 512K) |
| | Model 296 (128K, 512K or 2M) |
| | Model 395 (64K or 256K) |
| | Model 1375 (32K or 128K) |
| | Model 1385 (128K or 512K) |
| | Model 1395 (32K or 128K) |
| | Model 1396 (128K, 512K or 2M) |

## 1.3 Compatible DSOs

The following Digital Storage Oscilloscopes (DSOs) are currently supported by the WaveForm DSP2 software:-

| Manufacturer | Model |
|---|---|
| Tektronix | Model TDS 410 |
| | Model TDS 460 |
| | Model TDS 520 |
| | Model TDS 540 |
| | Model TDS 620 |
| | Model TDS 640 |
| | Model TDS 644 |
| | Model TDS 680 |
| | Model TDS 684 |
| | Model TDS 724 |
| | Model TDS 744 |
| | Model TDS 782 |
| | Model TDS 784 |

Waveform DSP2 uses editable script files for the control of data transfers from the selected DSO, and thus other non-supported DSOs may be added by creating a script file and adding it to the list in the ASCII configuration file 'dsp2u.cfg'.

# 2 Getting Started

## 2.1 Software Installation

To complete this installation you will need the disks supplied.

## 2.2 Software Requirements

The following software is required for the operation of the WaveForm DSP2 application:-

- Microsoft Windows 95 or Windows NT Version 4.0
- GPIB and / or VISA Dynamic Link Libraries (DLL) for communication with ARBs or DSOs. The GPIB.DLL is usually supplied with the IEEE488 interface card as part of the installation software. The VISA.DLL will be supplied with VXI Embedded Controllers or VXI Slot 0 Controllers.
  *NOTE: When WaveForm DSP2 is started it searches for the these DLLs to determine the communication capability of the application.*

## 2.3 Installation Procedure

Insert the WaveForm DSP2 installation Disk 1 in the floppy drive (A).
From the Windows Start menu select Run and type in the following program path and name **A:\setup.** This will start the installation process which contains instructions for the installer to follow to completed the installation succesfully.

On completion the target folder, nominated in the installation process (default = c:\Program Files\WaveForm DSP2), will contain the following files:-

This group of files are necessary for operation of the application.

| File Name | Description |
|---|---|
| Wavedsp2.exe | |
| Wavedsp2.ini | |
| wave_org.mlb | standard library of functions |
| mathsrv1.dll | |
| mathsrv1.ini | |
| waveform.wlb | waveform library |
| wavedsp2.cfg | download configuration. May not be edited. |
| dsp2u.cfg | upload configuration. May be edited to include new DSOs. Follow the instructions in the file. |
| waveform.hlp | Help file |
| waveform.cnt | Help contents |
| DelsL1.isu | |

This group of editable script files are used by download and upload script files and should not be removed or altered without extreme caution.

| File Name | Description |
|---|---|
| str2num.m | ASCII to number conversion |
| waitfor.m | reading - wait for |
| pi.m | Constant |
| readto.m | reading - up to |
| noise.m | random noise |
| Startup.m | Location of system files and Library loader |
| sgn.m | Older version of SIGN built-in function |
| rept.m | REPT built-in function |
| W_info.m | Wavetek information |
| expand.m | EXPAND built-in function |
| indx.m | INDEX built-in function |
| | |

This group of editable script files are used for upload from DSOs and ARBs. They may be modified if changes to the default operation are required.
*New files may be created, using these files as templates, for additional upload capability. Any new files must be added to the upload configuration file to become available for use by the application.*

| File Name | Description |
|---|---|
| G1600_ul.m | Untested — Gould models 1602,1604,1624,2608 |
| G4070_ul.m | Untested — Gould models 4072, 4074, 4082,4084 |
| H5454_ul.m | Untested — Hewlett Packard 54540A |
| H5475_ul.m | Untested — |
| H6075_ul.m | Untested — Hitachi models 6075, 6175, 6275 |
| K7101_ul.m | Untested — Kikusui models 7061, 7101, 7201 |
| T2232_ul.m | Untested — Tektronix model 2232 |
| T2430_ul.m | Untested — Tektronix models 2430, 2440 |
| T540_ul.m | Used for all Tektronix models TDS410, TDS 460, TDS 520, TDS540, TDS620, TDS640, TDS644, TDS680, TDS684, TDS724, TDS744, TDS782, TDS784 |
| Y1200_ul.m | Untested — Yokogawa model DL1200A |
| W29_ul.m | Wavetek Model 29 |
| W39_ul.m | Wavetek Model 39 |
| W295_ul.m | Wavetek Model 295 |
| W296_ul.m | Wavetek Model 296 |

| | |
|---|---|
| W395_ul.m | Wavetek Model 395 |
| W1375_ul.m | Wavetek VXI Model 1375 |
| W1385_ul.m | Wavetek VXI Model 1385 |
| W1395_ul.m | Wavetek VXI Model 1395 |
| W1396_ul.m | Wavetek VXI Model 1396 |
| H5454err.m | May be used to assist detection of instrument error during script development. |
| T540err.m | ditto |
| W395err.m | ditto |
| W39err.m | ditto |

This group of editable script files are used for download to ARBs.
They may be modified if changes to the default operation are required.

| **File Name** | **Description** |
|---|---|
| W29_dl.m | Wavetek Model 29 |
| W39_dl.m | Wavetek Model 39 |
| W295_dl.m | Wavetek Model 295 |
| W296_dl.m | Wavetek Model 296 |
| W395_dl.m | Wavetek Model 395 |
| W1375_dl.m | Wavetek VXI Model 1375 |
| W1385_dl.m | Wavetek VXI Model 1385 |
| W1395_dl.m | Wavetek VXI Model 1395 |
| W1396_dl.m | Wavetek VXI Model 1396 |

Examples Folder contains

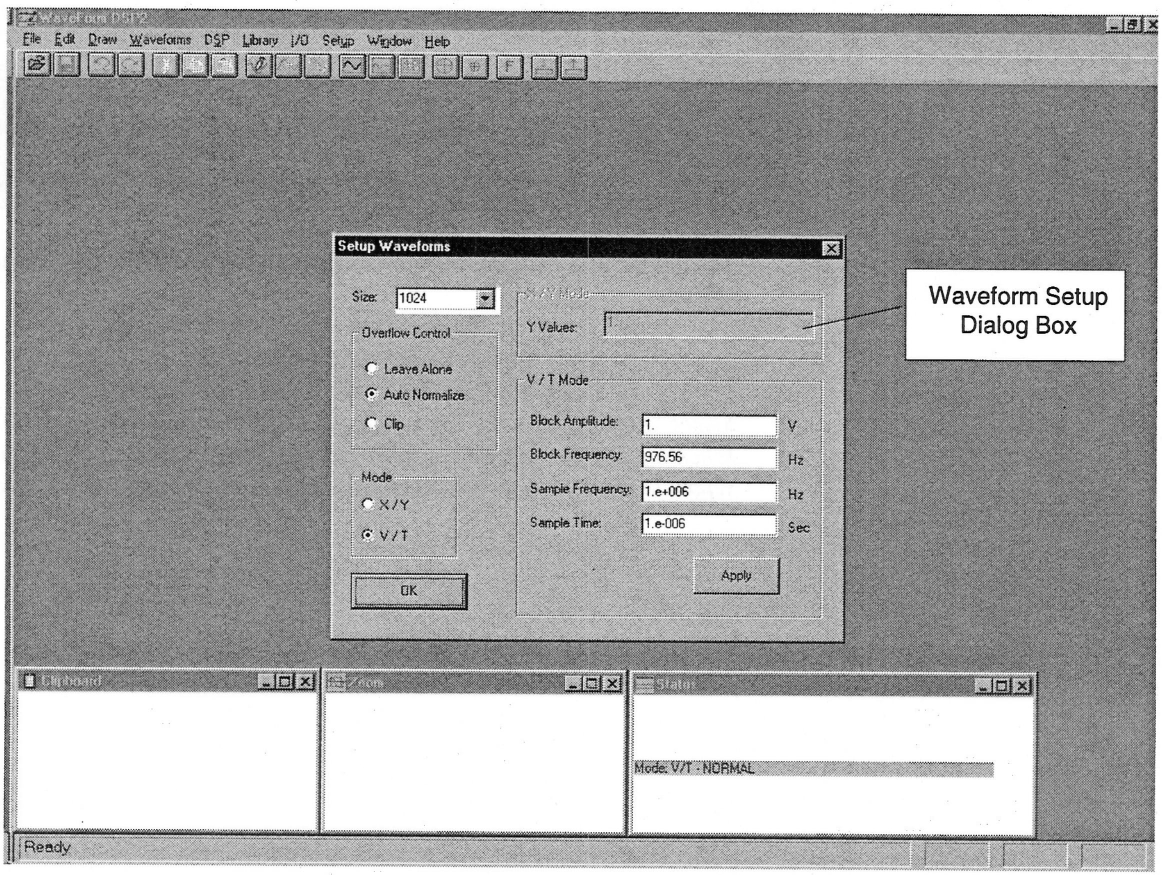
| File Name | Description |
|---|---|
| comp.wfm | demonstration file |
| Demo.wfm | demonstration file referenced in Help and Manual |
| Demo_ops.wfm | demonstration file referenced in Help and Manual |
| Examp1.wfm | as Example #1 in users manual |
| Examp2.wfm | as Example #2 in users manual |
| Examp3.wfm | as Example #3 in users manual |
| Examp4.wfm | as Example #4 in users manual |
| Examp5.wfm | as Example #5 in users manual |
| Examp5A.wfm | as Example #5A in users manual |
| Examp5B.wfm | as Example #5B in users manual |
| Examp6.wfm | as Example #6 in users manual |
| Examp7.wfm | as Example #7 in users manual |
| Examp8.wfm | as Example #8 in users manual |
| Examp9.wfm | as Example #9 in users manual |
| Examp9A.wfm | as Example #9A in users manual |
| templ_ul.m | template for an upload script |
| Wxxx_ul.m | template for a Wavetek instrument upload script |
| WxxxERR.m | template for an Wavetek Error function |

## 2.4 Starting WaveForm DSP2

You can use either the mouse or keyboard to move around the WaveForm DSP2 display and make the desired selections. WaveForm DSP2 follows the standard Windows procedures in most dialog boxes.

For a standard installation select Windows Start, then select from Programs the WaveForm DSP2 folder. Click on the WaveForm DSP2 icon or highlight using the cursor keys and activate with the <Enter> key.

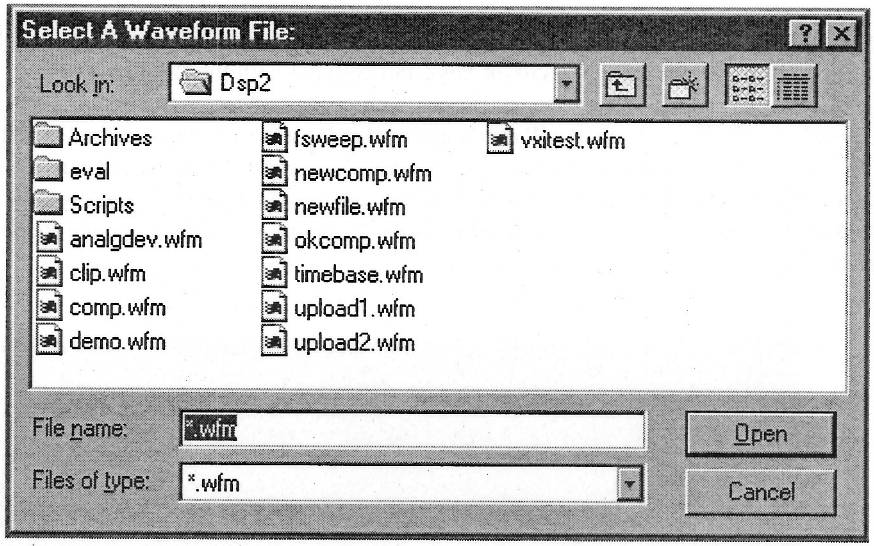
As WaveForm DSP2 starts up, it will momentarily display a copyright notice and then the Setup Waveforms dialog box. This allows you to enter the preferences that WaveForm DSP2 will apply to the waveforms you create and download. You must press the <Enter> key or press the OK button with your mouse to get past this dialog. Detailed information on setting these parameters is contained in the next section. This setup dialog can be opened at any time to make changes by selecting the Setup menu and selecting Waveforms.



## 2.5 Activating an Existing Waveform

I. Select the **File** menu.

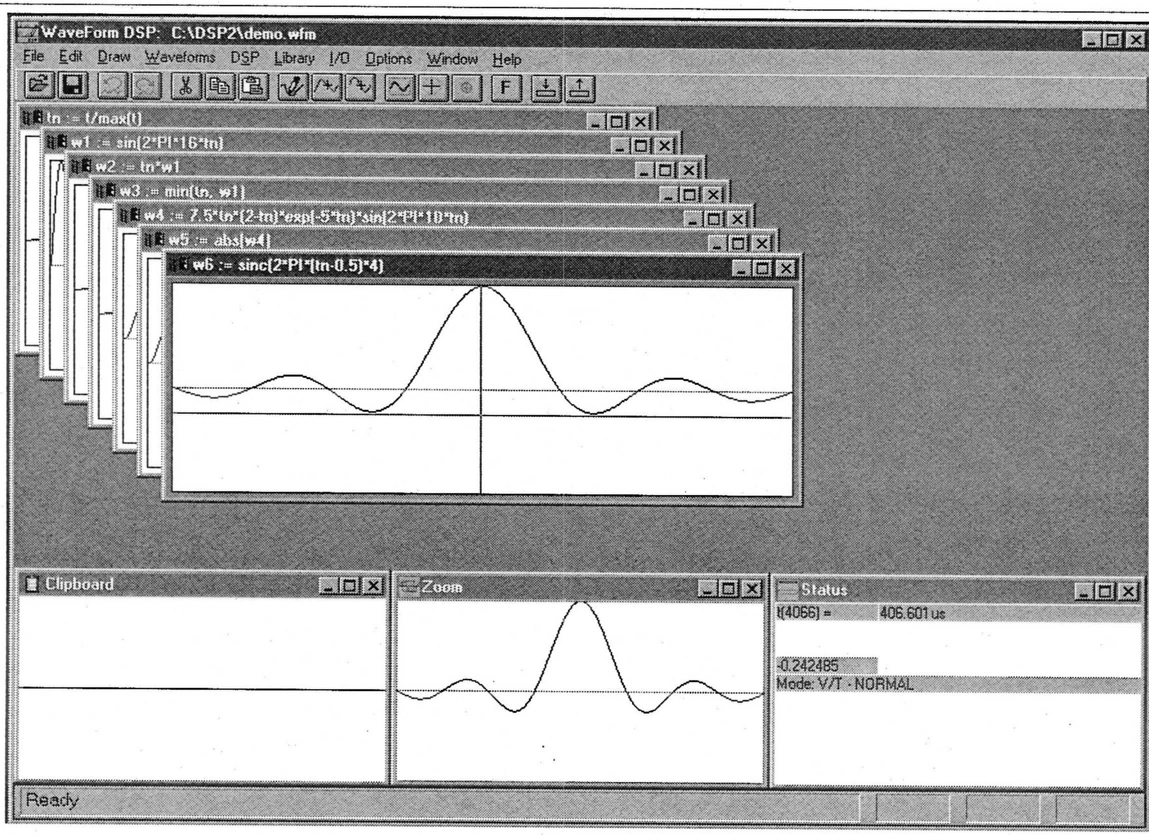
II. Select the **Open** option




III. Select the file containing the waveform you want to work with. You have three options:

A. Select the file name from the list of filenames
B. Type the File name into the dialog box
C. Look in a different folder for files of the required type.

*Select "demo.wfm" and "demo_ops.wfm" for a look at a typical file.*
*When a file is opened, one or more waveforms appear. Each waveform is in a separate window cascaded to allow selection of the required window.*




IV. Select the window with the waveform you want to work with. Do this by clicking on an exposed portion of the window. *If the window you want is hidden, select the **Windows** menu **Tile** option.*

## 2.6 Creating Waveforms

There are many ways to create waveforms with WaveForm DSP2, but basically they can be divided into two main methods: equations and graphics. The two methods are described in the sections titled *Working With Equations* and *Working With the Editor.*

As a quick example you can create an additional waveform in the demo.wfm waveform file using two of the other waveforms already created.

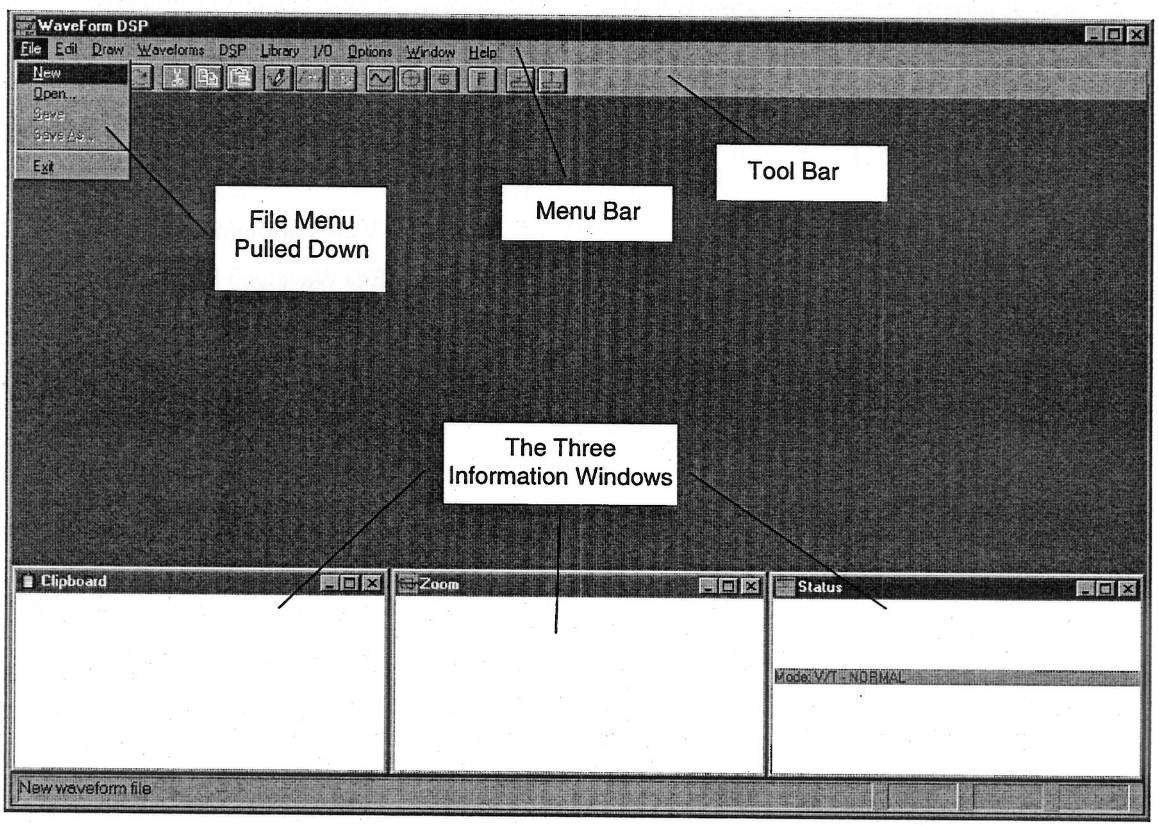
Open up a new waveform window by typing **^N** (Ctrl + N) or by using the mouse to select the **Create** option under the **Waveforms** menu. A dialog box pops up asking you to enter a name of the new waveform. Type in **w5** then `<Enter>`. Now select the **Expression** option from the **Waveforms** menu or type **^E** which will provide a dialog box for expression entry. Type in **w1*w4** and `<Enter>`. Observe all of the *demo.wfm* waveforms by selecting the **Tile** option under the **Windows** menu. You can save the new waveform by selecting File and Save. Your *demo.wfm* will now have the additional waveform.

# 3 User Interface

This section contains an illustration and a brief description for each of the main Waveform DSP2 windows and dialog boxes. It also shows you how to select and activate items in the displays, using both the mouse and the keyboard.

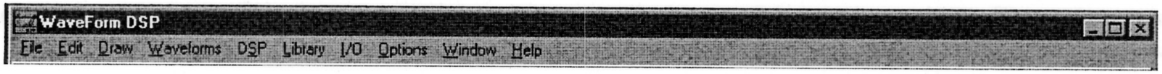
## 3.1 Window Format

The main WaveForm DSP2 screen has the standard Windows format. A title bar at the very top contains the name of the currently active file. In the upper left-hand corner there is a control menu button. In the upper right-hand corner there are the windows sizing and close buttons. A menu bar, directly under the title bar, contains the names of the WaveForm DSP2 menus. A toolbar, directly under the menu bar, contains short-cut buttons for common operations.



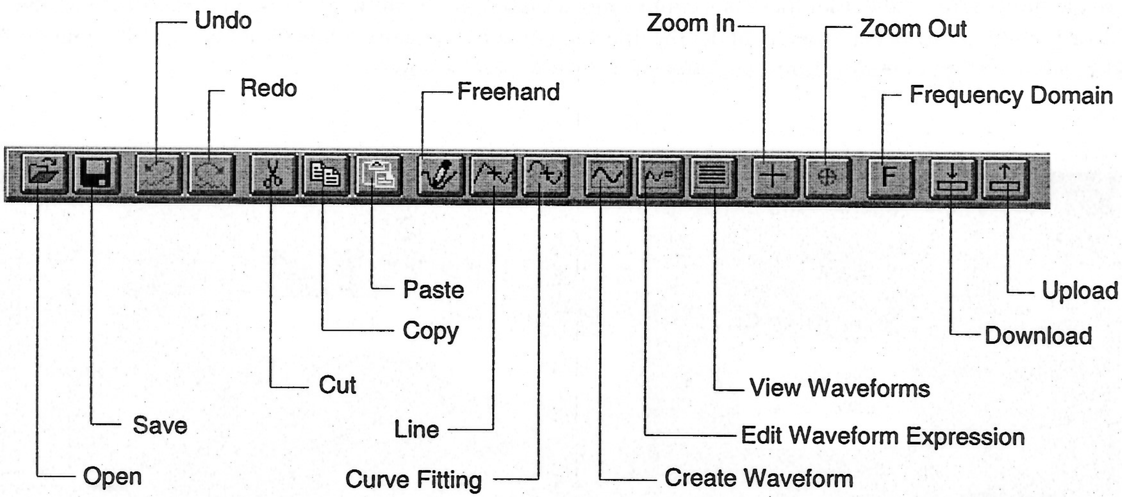
## 3.2 Menu Bar

The menu bar appears directly beneath the title bar:

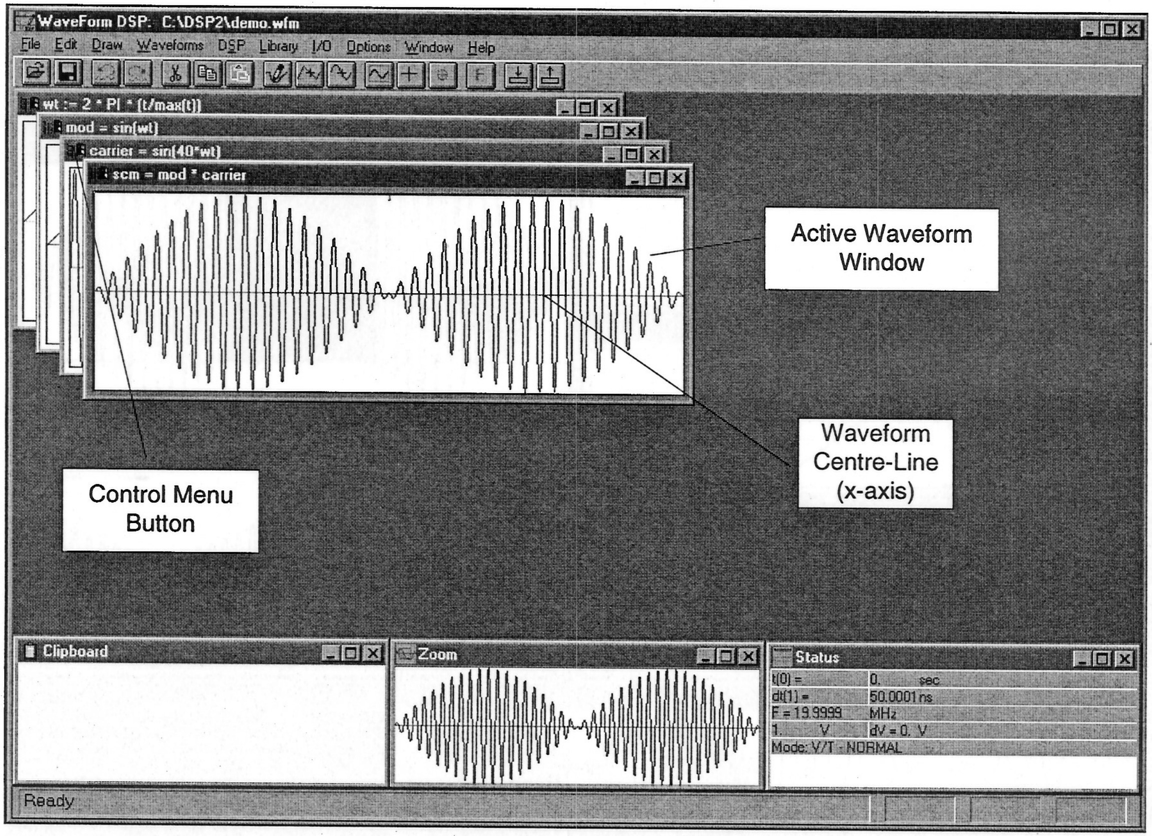



## 3.3 Toolbar

The toolbar offers short-cut buttons for common operations:



# 3.4 The Waveform Window

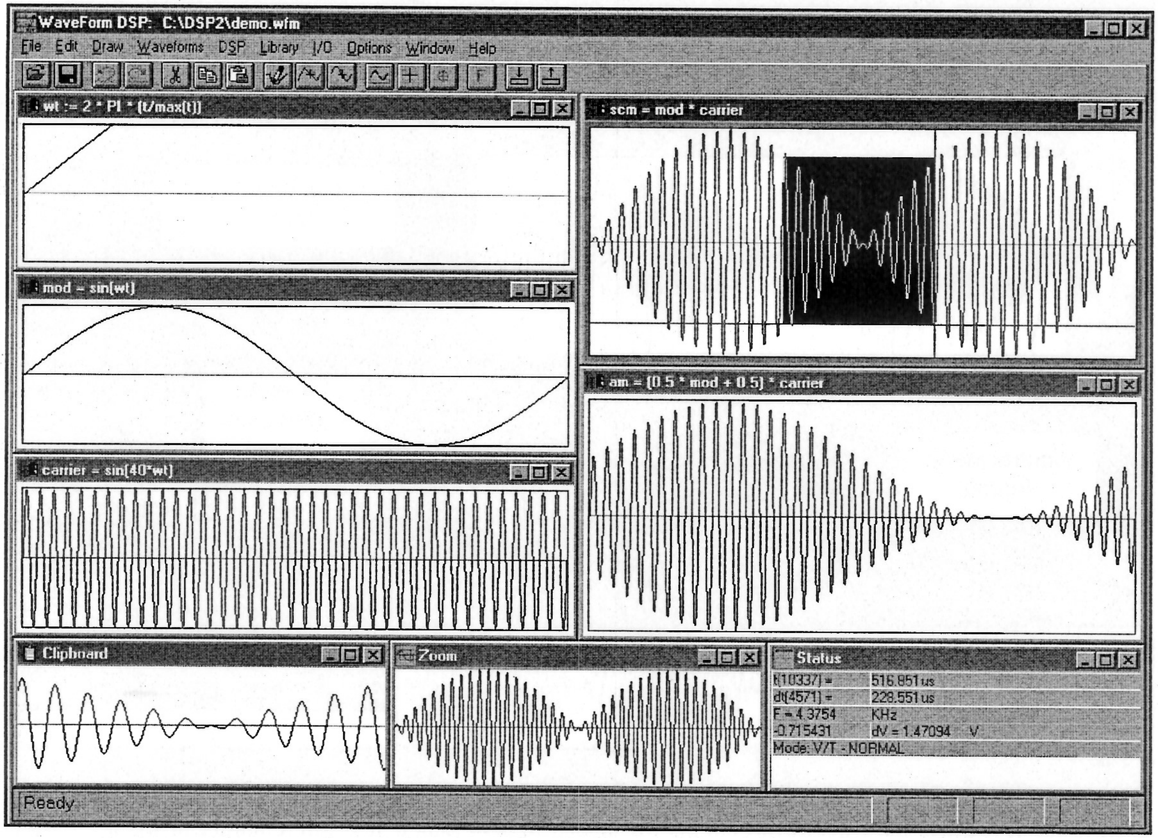



When a waveform file is opened, the waveform(s) in the file appear in their own waveform windows inside the main window. In the illustration above, a four waveform file is shown with the individual waveform windows cascaded. The top window is the currently active waveform. The waveform windows also have the standard Windows format. A title bar at the top of the window contains the name of the waveform, and there is a control-menu button in the upper left corner.

*Waveforms which were created with equations have the equation appended to their name.*

When the entire waveform is displayed in the waveform window, the window has no scroll bars. When a portion of the waveform is selected and zoomed in, scroll bars appear so that the segment being zoomed can be moved around the waveform. The waveform window can be resized, moved, opened and closed. In addition to the waveform, the window has a horizontal line corresponding to the x-axis centre-line of the waveform.

# 3.5 Information Windows




The three information windows appear at the bottom of the main window when WaveForm DSP2 is opened. If closed, they can be activated through the Window menu. These information windows give the user visual indication of the effect of various operations, such as enlarging the view of a portion of the waveform, and cutting portions of the waveform to the clipboard.

The status window provides a read-out of waveform data, using the cursor position within the currently active waveform window. In the example above, the Mode is "V/T - NORMAL" (not zoomed). The "scm" waveform has a segment highlighted (the black box), and the dimensions of the highlighted area is given as dV and dT in the Status window. The highlighted area has also been copied, and it appears as a waveform segment in the Clipboard. An ASCII data file of data points of the same waveform segment is also copied to the external Windows clipboard. Since the active waveform has not been zoomed in, the Zoom window contains the entire active waveform.

## 3.6 Cursor

Most of the time the WaveForm DSP2 cursor appears as a standard single-headed or double-headed arrow. In the Draw and Library modes, before pressing the mouse button, the cursor has the form of a pair of short cross hairs, which are placed at the point where you begin to draw or insert waveforms.

In the active waveform window, with the left mouse button pressed, the cursor takes the form of a horizontal line and a vertical line the entire window. (To see them you may have to move the edges of the screen). The vertical line follows the horizontal movement of the mouse or arrow keys, and the horizontal line follows movement of the mouse or arrow keys.

## 3.7 Moving around the display

You can use either the mouse or the keyboard to move around display and make the desired selections. WaveForm DSP2 follows the standard Windows procedures.

If you are unfamiliar with Windows, it may be helpful to review the Windows documentation before using Waveform DSP2.

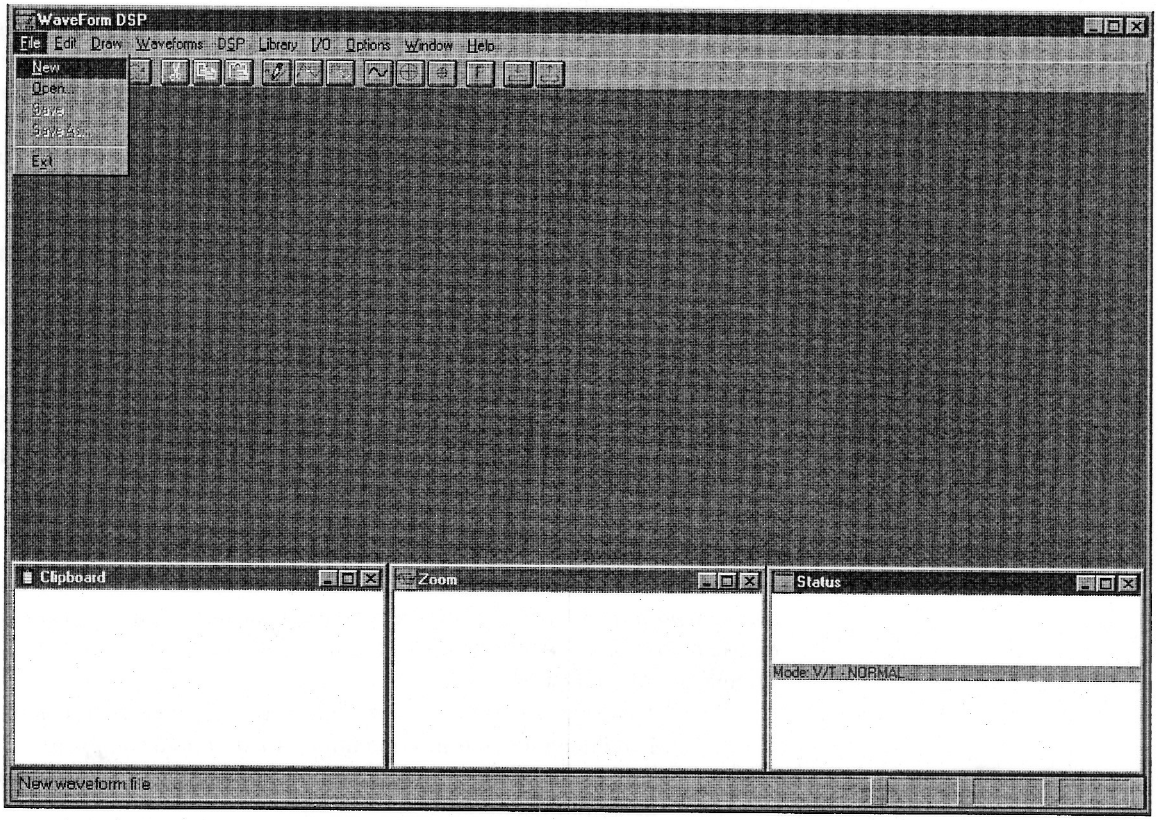
### Using the Mouse

The screen's cursor arrow follows the mouse. Items are selected by clicking the left mouse button in the standard manner. To click means to quickly press and release the mouse button, keeping the cursor on the selected item.

To Select a Menu Option

1. Move the cursor to the name of the menu (in the selection top of the screen).
2. Hold down the left mouse button. *A list of options appears.*
3. While still holding down the left mouse button, drag the cursor down to the name of the option
4. Release the mouse button.

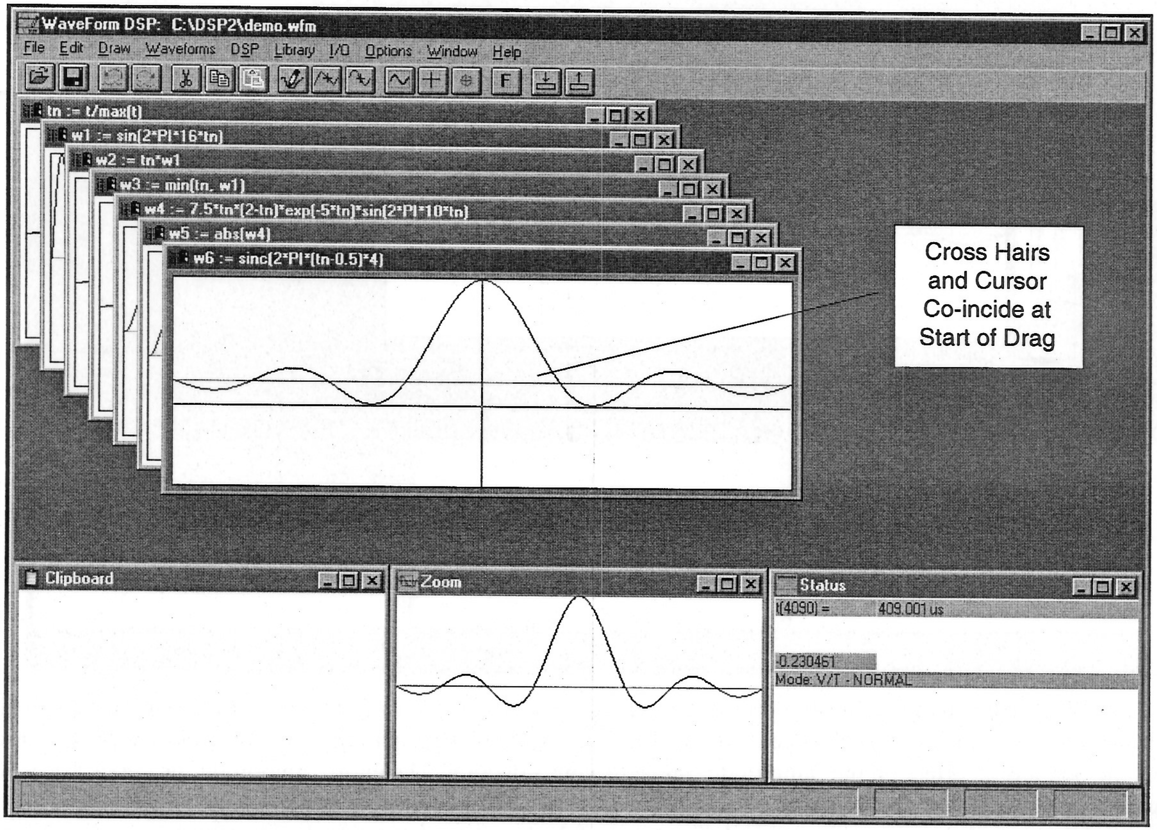
*You can also click on the name of a menu. Its list of options remains until you click on the name of an option.*


WaveForm DSP
File
Edit
Draw
Waveforms
DSP
Library
I/O
Options
Window
Help
New
Open...
Save
Save As...
Exit
Clipboard
Zoom
Status
Mode: V/T - NORMAL
New waveform file

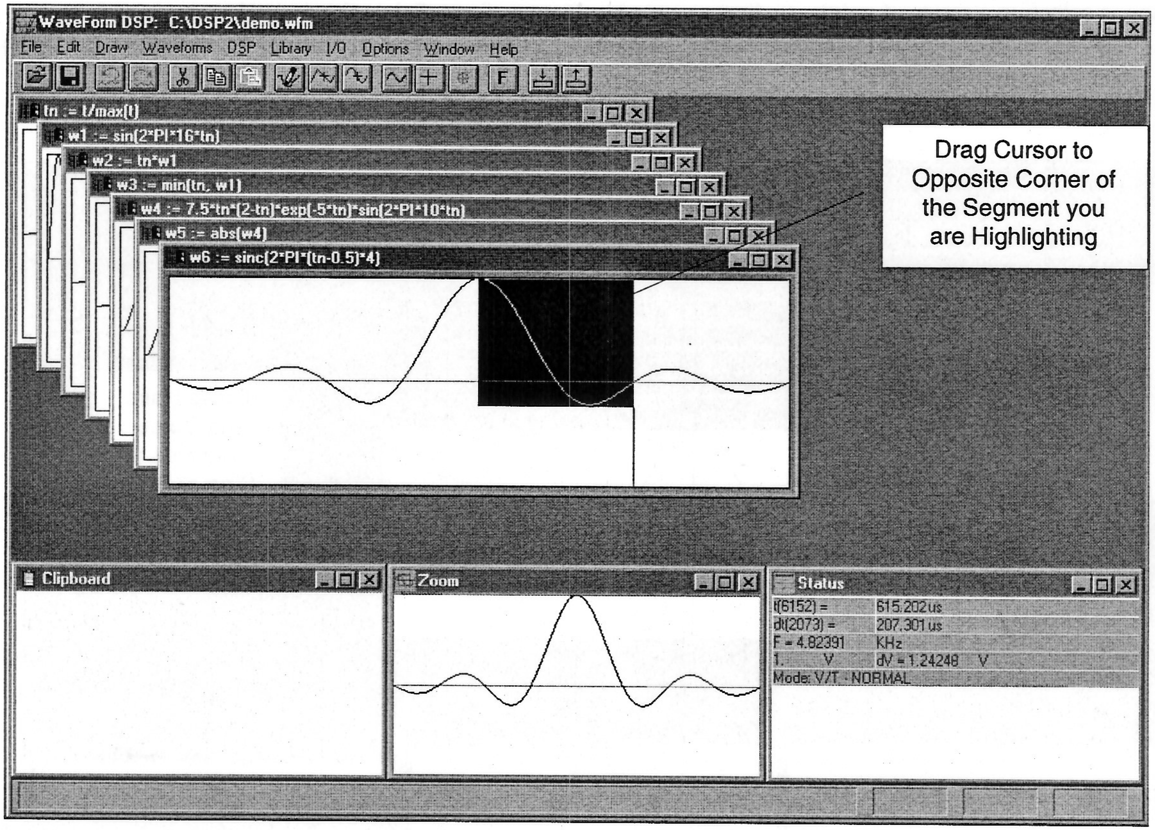
*To Highlight*

Highlighting is selecting a series of points, such as a segment of a waveform. Select the desired segment as follows:

1. With the mouse, position the cursor on the point at one end of the desired portion. Then depress and do not release the mouse button.




2. Drag the cursor across the waveform to the point at the other end of the desired portion, and then release the mouse button.




Your starting point and ending point define the opposite corners of a box, which is shown in reverse video. The horizontal dimension of the box determines the 'time' or x-axis range of your selection. If you are highlighting in order to cut or copy, then the vertical dimension of the box has no effect on the 'amplitude' range of your selection. The entire 'amplitude' or y-axis range within the selected segment is included in your selection.

If you are highlighting in order to zoom, the vertical dimension of the box does determine the 'amplitude' or y-axis range of your selection, and horizontal dimension still determines the 'time 'or x-axis range of your selection.

*To Escape*

If your mouse has more than one button, clicking the right mouse button is the same as striking the ESCAPE key on the keyboard.

## Using the keyboard

In most dialog boxes, WaveForm DSP2 follows the standard Windows conventions on the use of the Alt, ENTER, TAB, and arrow keys in place of the mouse.

*To Select a Menu Option*

There are three ways to select a menu option:

1. By opening the menu *(1)*:
   a) Hold down the Alt key and type the underlined letter of the menu name as it is shown in the Menu Bar.
2. A list of options appears under the menu name.
   a) Use the arrow keys to move the cursor down to the name of the option, and strike the ENTER key.
3. By opening the menu *(2)*:
   a) Hold down the Alt key and type the first letter of the menu name.
4. A list of options appears under the menu name.
   a) Each of the menu options has one letter underlined, strike the key for that letter.
5. By using the keyboard accelerators described under Accelerators below.

To deselect the window, strike just the Alt key.

*To Highlight*

Highlighting is selecting a series of points, such as a segment of a waveform. Select the desired segment as follows:

1. Use the arrow keys to move the cursor to the point at one end of the desired portion of the waveform.
2. While holding down the Shift key, use the arrow keys to move the cursor to the point at the opposite end of the desired portion.

   *Your starting point and ending point define the opposite corners of a box, which is shown in reverse video. The horizontal dimension of the box determines the 'time' or x-axis range of your selection.*

   *If you are highlighting in order to cut or copy, then the vertical dimension of the box has no effect on the "amplitude" range of your selection. The entire "amplitude" y-axis range within the selected "time" segment is included in your selection.*

   *If you are highlighting in order to zoom, the vertical dimension of the box does determine the "amplitude" or y-axis range of your selection, and horizontal dimension still determines the "time" or x-axis range of your selection.*

### Accelerators

There are options in the Edit and Waveform menus that have shorter keyboard equivalents. The keyboard equivalents are called accelerators because they are usually simpler to use than the mouse or the "underlined key" combinations. These accelerators are shown in the pull-down menus.

Accelerators are shown in the pull-down menus beside their option names. They are listed here for ready reference. The plus sign (+) is used to indicate that you hold down the first key while striking the second. When using accelerators, you do not specify the menu they belong to.

| | |
|---|---|
| Cut | Ctrl + X |
| Copy | Ctrl + C |
| Delete Selection | Highlight selection; strike DEL |
| Delete Waveform | Ctrl + D |
| Enter Expression | Ctrl + E |
| Create Waveform | Ctrl + N |
| Paste | Ctrl + V |
| Select All | Ctrl + A |
| Redo | Ctrl + Y |
| Undo | Ctrl + Z |
| Zoom In | Ctrl + I |
| Zoom Out | Ctrl + O |

### Activating Windows

Clicking on any window makes it the active window (the window that accepts input from the operator). An individual information window can be opened, closed, moved and resized according to Windows format.

## 3.8 Control Menu

This is a standard Windows feature, and is only briefly described here. See Windows documentation for details.

The following window controls are available from the window control button that appears in the top left corner of the main Waveform window.

### Move

Changes position of window. Any window can also be moved by clicking the mousebutton on the title bar and holding the mouse button down as you move the mouse.

### Window Size

Changes the size of the window. The shape and size of any window can also be changed by moving the arrow cursor to an edge or corner of a window where the arrow becomes bi-directional, then holding down the mouse button and dragging the edge or corner to the desired position.

### Minimize

Reduces window to an icon. (Run application "in background"). Active only when running an application inside a window.

### Maximize

Enlarges window to fill screen. Active only when running an application inside a window.

### Close

Deactivates and removes window from view.

## 3.9 File Menu

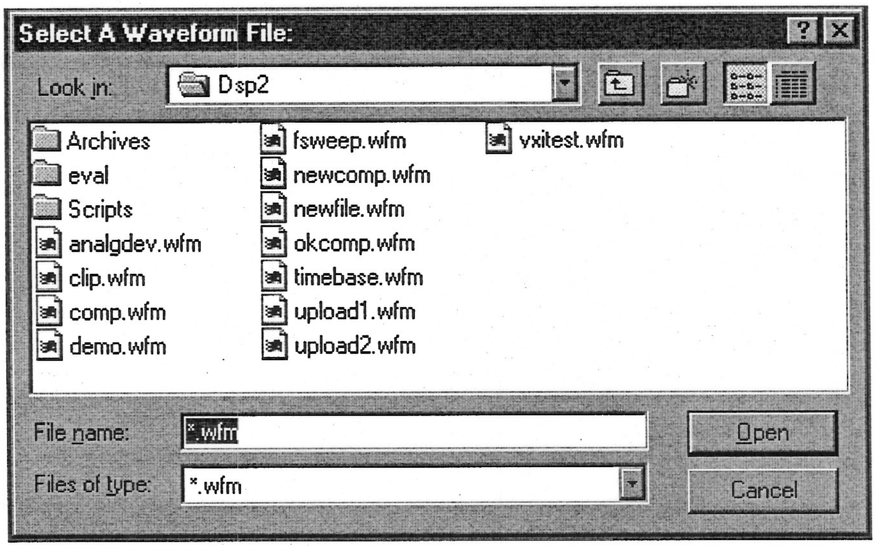
Open, Close, Create or Rename files.

### New

Creates an unnamed new file in which to define new waveforms. This option closes any open files, clearing the screen of any active waveforms. You can then move to the Waveform menu or toolbar to create a new waveform. There is no dialog box, because you name the file when you first save.

### Open...

Selects file containing desired waveform(s).




Select the file by typing its name in the dialog box, or by selecting its name from the list of files. You can also look in another folder and select a file from it, if an appropriate file exists there.

*Only files with the '*.wfm' extension are displayed.*

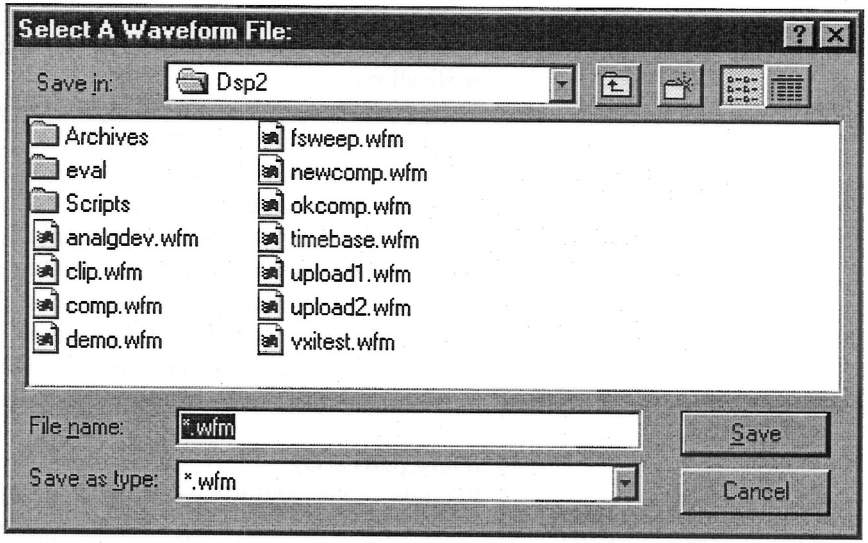
### Save

Saves waveform(s) to original file and over-writes original (whether changed or not). If the file has not previously been saved, a file-name dialog box opens (as for the Open option). If the file has been saved already, the box does not open. A small hour glass appears until the file is saved. The convention for waveform file extensions is '*.wfm'.

**Save As ...**

- Saves new waveform(s) to new file name.
- Copies an existing file to a different file name (whether changed or not).
- Copies an existing file to an entirely new path name.

*The file-name dialog box opens (as for the Open...option)*




**Exit**

Terminates the WaveForm session, and returns you to Windows. If you have changed the file and not saved the changes, a dialog box appears. You must select 'Yes' (save changes and exit), 'No' (abandon changes and exit), or 'Cancel' (do not exit).

## 3.10 Edit Menu

Cut, insert, copy, and delete waveform segments.

**Undo (Ctrl + Z)**

Cancels last command or keystroke.

**Redo (Ctrl + Y)**

Repeats last command or keystroke.

### Cut (Ctrl + X)

Removes the highlighted portion from the currently active waveform and moves that portion to the clipboard.

The removed portion appears in both the Windows clipboard and the local WaveForm clipboard.

The copy in the Windows clipboard is in ASCII format. The Y-axis value of each data point is an ASCII character. The X-axis values are not required to reconstruct the waveform, and they are not saved. This copy can be exported to other Windows applications.

The copy in the local WaveForm clipboard is a graphic representation of the ASCII data.

The copies in each clipboard remain in that clipboard (even if you close the file) until something else is saved to that particular clipboard. As long as it remains in the Windows clipboard, it can be pasted into any file and into any part of any waveform.

*If a curve from another application is imported to WaveForm from the Windows system clipboard, it DOES NOT update the WaveForm clipboard.*

*In the WaveForm clipboard, the cut portion horizontally fills the clipboard window, regardless of the number of points it contains. Therefore, if just one point is cut, it shows up on the clipboard as a straight horizontal line at the amplitude 1/2 the cut point. However, it will paste as a point.*

### Copy (Ctrl + C)

Reproduces the highlighted portion of the currently active waveform and saves it to the clipboard, without removing it from the active waveform.

The copied portion appears in both the Windows clipboard and the local WaveForm clipboard.

The copy in the Windows clipboard is in ASCII format. The Y-axis value of each data point is an ASCII character. The X-axis values are not required to reconstruct the waveform, and they are not saved. This copy can be exported to other Windows applications.

The copy in the local WaveForm clipboard is a graphic representation of the ASCII data.

The copies in each clipboard remain in that clipboard (even if you close the file) until something else is saved to that particular clipboard. As long as it remains in the Windows clipboard, it can be pasted into any file and into any part of any waveform.

*If a curve from another application is imported to WaveForm from the Windows system clipboard, it DOES NOT update the Waveform Clipboard.*

*In the WaveForm clipboard, the copied portion horizontally fills the clipboard window, regardless of the number of points it contains. Therefore, if just one point is copied, it shows up on the clipboard as a straight horizontal line at the amplitude of the copied point. However, it will paste as a single point.*

### Paste (Ctrl + V)

Reproduces the series of data points currently residing in the Windows clipboard into the currently open... file. The pasted segment starts at the point where the cursor is in the target window in the target file.

If there is a waveform already in the window, but there is no highlighting on the waveform, the pasted segment is inserted into the waveform beginning at the point where the cursor lies.

If there is a waveform already in the window; and it is highlighted, the pasted segment replaces the highlighted portion of the waveform.

> *If a curve from another application is pasted to a WaveForm file from the Windows system clipboard it DOES NOT update the WaveForm Clipboard.*
> *If any pasting causes the resulting waveform to grow beyond the maximum number of points for the window, the resulting waveform is truncated at the right-hand limit.*

### Delete (Del)

Removes the highlighted portion of the waveform, and then closes up and re-connects the remaining portions of the waveform.

### Select All (Ctrl + A)

Highlights the entire currently active waveform. The entire waveform can then be cut, copied , pasted or deleted.

## 3.11 Draw Menu

Puts the selected Waveform into the draw mode.

The ***cursor*** changes to short cross hairs when inside the waveform window while in draw mode. To turn the draw mode off, click the right mouse button, or press the ESCAPE key, returning the ***cursor*** to an arrow.

### Curve Fitting

The curve fitting method used in WaveForm fits the generalised third degree equation:

*(at3+bt2+ct+d)*

through the points you select in a waveform window. This is commonly referred to as a spline function.

Position the short crosshairs at the start of your waveform or segment. Click the mouse button, this changes the cursor to the window wide crosshairs. Repeat from left to right for each of the selected points in turn.

The curve is fitted to three points at a time. When you select a series of more than three points, the program looks back at the last two points selected. A new curve is then fitted to the latest three points selected. The process continues until the last point is selected. The session is ended by clicking the right mouse button or striking the ESCAPE key.

*If the resulting waveform exceeds the full-scale vertical value of the waveform window, the waveform may be automatically re-scaled or clipped when the Curve Fitting mode is escaped (see Setup menu Set-up Waveforms option)*

Curve fitting is most useful when it is extremely difficult to characterise the waveform mathematically, but data points are obtainable.

In this mode, any movement of the cursor inside the waveform window, even after releasing the mouse button, changes the shape of the waveform.

Strike the ESCAPE key or click the right mouse button to stop drawing and turn the draw mode off returning to the arrow cursor.

### Freehand

Uses mouse or arrow keys to create or change a waveform in the waveform window of the currently active file. Drawing exactly matches the motion of the mouse, except that only left-to-right motion is included. The software does not help by smoothing or selecting the closest smooth line (as Line and Curve Fitting options).

Position the short crosshairs at the start of your waveform or segment. Hold down the mouse button to draw, this changes the cursor to the window wide crosshairs. Release the mouse button to stop drawing. Strike the ESCAPE key or the right mouse button to turn the draw mode off, returning to the arrow cursor.

### Line

Draws a straight line, which is aligned between the point the cursor was on when you started drawing, and the point the cursor was on when you stopped drawing. Since each x-position can have only one y-value, a vertical line will actually lean by one x point. Position the short crosshairs at the start of your waveform or segment. Click the left mouse button to start the draw, this changes the cursor to the window wide crosshairs. Move the cursor to the desired position.

*A line is drawn to connect the cursor with the previous point.*

Click the left mouse button to confirm this point and move on to the next point

*In this mode, any movement of the cursor left or right inside the waveform window, can change the shape of the waveform if a left mouse click (confirm) is used.*

Strike the ESCAPE key or the right mouse button to stop drawing and turn the draw mode off, returning to the arrow cursor.

## 3.12 Waveforms Menu

Create, delete, view, set amplitude of waveforms.

### Create

Creates a new empty waveform window in an existing file. When Create is selected, a dialog box appears in which to name your new waveform.
The name given to the waveform must be unique.
If the name you choose for your new waveform already belongs to a window in the current file, an error message and dialog box appear, prompting you to use a different name.

Names given to waveforms created by using the mathematical expression for the waveform, have the following additional constraints:

- The first character must be alphabetical or the underline.
- After the initial alpha or underline, characters can be numerical, alphabetical, or the underline.
- There can be no spaces or special characters.
- There can be no more than 31 characters.
- Upper and lower case letters are not recognised as different.
- The name cannot be the same as any of the built-in function names.

After the waveform is named using the dialog box, a window appears in which to create the new waveform.

*Initially, the window contains a straight horizontal line at zero volts (or y = 0) which is hidden by the x-axis line.*

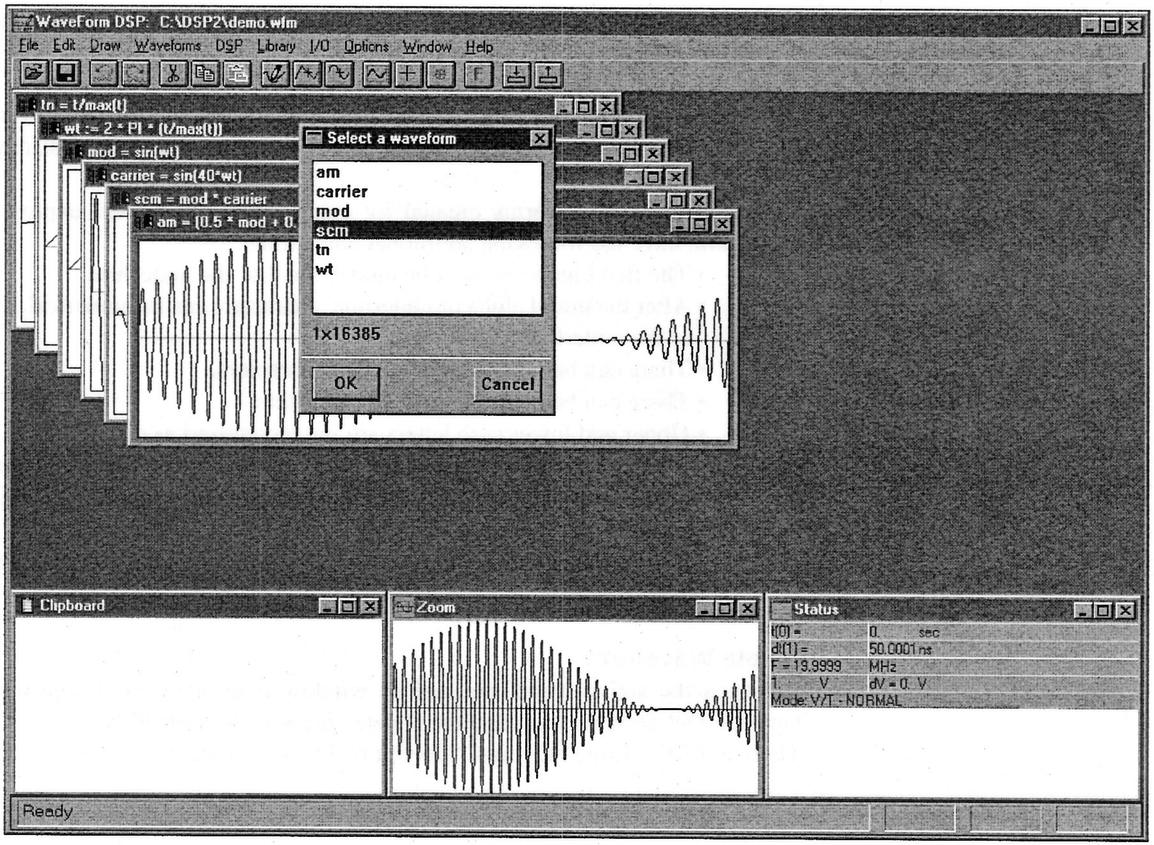
### Delete Waveform

Removes the active waveform and its window from memory. (Note the difference between Delete Waveform and the Delete option in the Edit Menu).
There is **NO** warning message, 'Do you really want to delete waveform <name>?'

## View

Selects an existing, but not active, waveform from an alphabetical Waveforms list and makes it the currently active waveform.

*This is a way to select a waveform when its window is closed or hidden from view and cannot be reached by the cursor.*



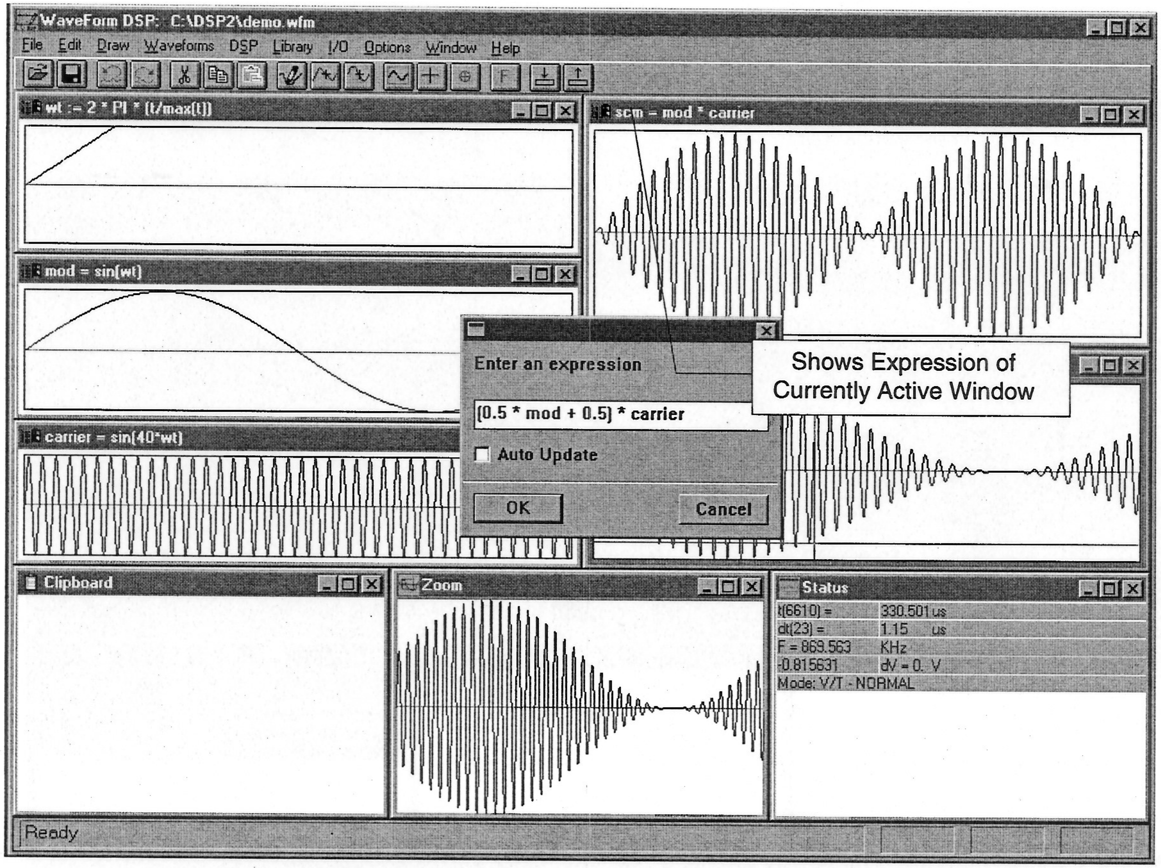
## Expression

Creates a waveform from its mathematical expression when the expression is typed into the dialog box.

The expression may reference the name of any other waveforms already created and loaded. Changing or replacing the expression in the dialog box of an existing waveform, changes the existing waveform. If a waveform file has been resized (*see the Setup menu, Set-up Waveforms option*), any waveform defined with an expression can be reset to fill the new block size by opening and then closing its expression dialog box.

The expression is checked by an equation parser, and if valid, it appears in the title bar of the waveform window. The waveform can subsequently be changed by any of the non-mathematical editing methods available, such as cutting, pasting, or drawing. However, the expression in the title bar then disappears, because it no longer represents the waveform.

An expression is any series of symbols that MathView* uses to produce a value. The simplest expressions are constants and variable names, which have no operator and generate a value directly.



## Zoom-In (Ctrl + I)

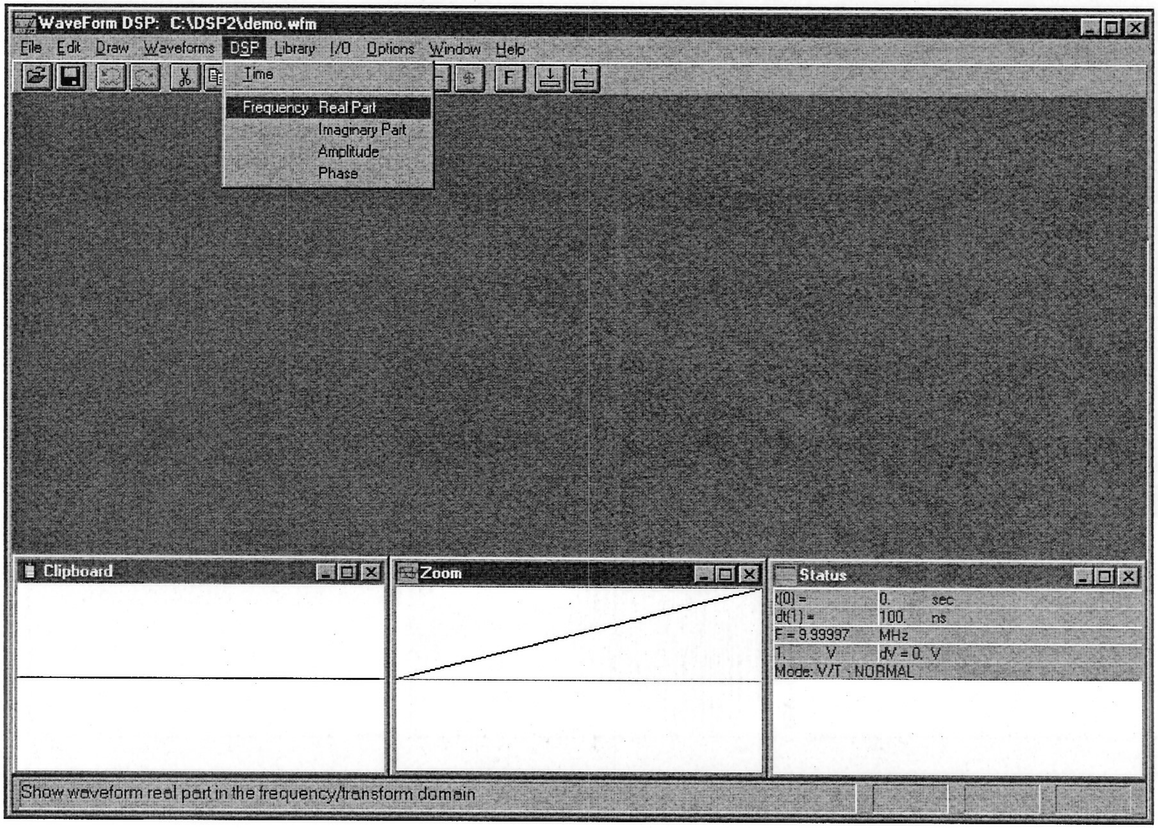
Scaled up (close-up) view of the current waveform.
The portion of the waveform that you highlight fills the window with an enlarged view of that portion. A zoomed-in window can be zoomed-in repeatedly for closer views.

## Zoom-Out (Ctrl + O)

Normal view of the current waveform.

# 3.13 DSP Menu

Once a function is created in the time domain window, by expression or by the editor, its corresponding frequency-domain function can be displayed by selecting the Frequency option from this menu.



### Time Window

Displays the time-domain curve of the function in the currently active window. This is the default selection.

### Frequency - &Real Part

This window displays the real portions of the complex spectrum.
WaveForm appends "_R" for real-parts to the name of the original time-domain signal to create the name for the transform.

### Frequency - &Imaginary Part

This window displays the imaginary portions of the complex spectrum.
WaveForm appends "_I" for imaginary-parts to the name of the original time-domain signal to create the name for the transform.

### Frequency - &Amplitude

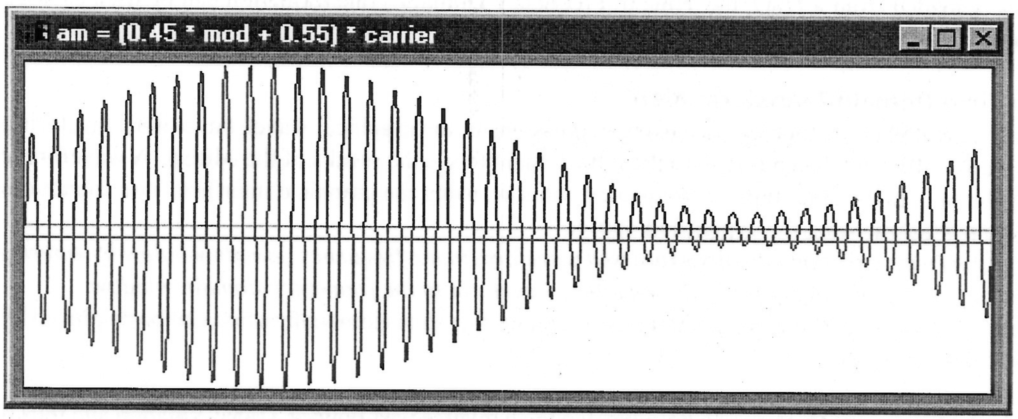
This windows displays the amplitude properties of the complex spectrum.
WaveForm appends "_A" for amplitude to the name of the original time-domain signal to create the name for the transform.

### Frequency - &Phase

This window displays the phase properties of the complex spectrum.
WaveForm appends "_P" for phase to the name of the original time-domain signal to create the name for the transform.

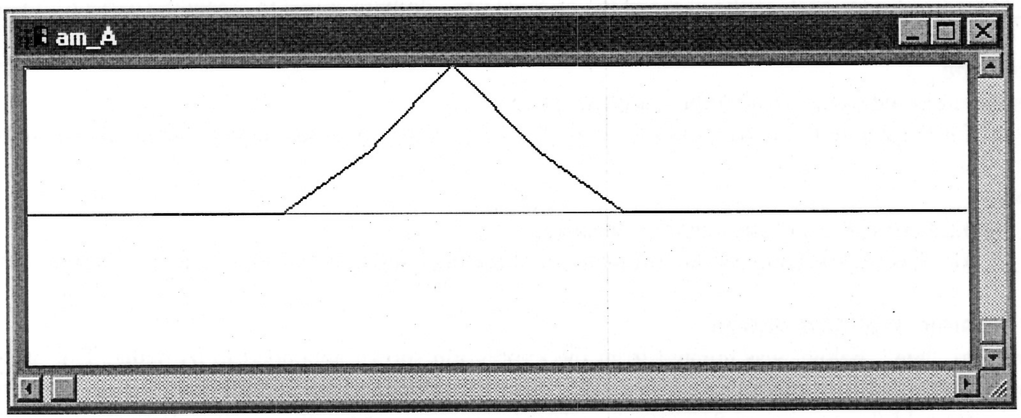
### Time to Frequency Domain Transformation

The time domain waveform, "am", below, was created from the expression shown appended to its name. The waveform is a sinusoidal 40 kHz carrier, 90% amplitude modulated with a 1kHz sine. If you opened the Setup Menu, Set-up Waveform option dialog for this waveform, it would show a size of 8192 points, a Block Frequency of 1000Hz, a Sample Frequency of 8192 MHz and a Sample Time of 122ns.




When the "am" waveform is translated to the frequency domain, a Fast Fourier Transform (FFT) is run on the waveform data file. Once the transform data file is built, WaveForm can quickly switch between the time representation and the four frequency representations. The frequency domain waveform data file uses the lowest possible periodic frequency (the Block Frequency) to calculate successive spectral components (dF) and includes all spectral components out to the Nyquist point of the Sample Frequency in the waveform. Negative frequency is not displayed, so the frequency domain waveform widow is created with 0Hz at the left, Sample Frequency divided by 2 at the right, and a frequency resolution equal to the block frequency.

The frequency domain window below, "am_A", contains the amplitude vs. frequency spectrum of "am" from DC to 4.096MHz, with a resolution of 1kHz. The window below has been zoomed in to show approximately the first 60kHz of the spectrum. This is why the window has scroll bars.




If a Waveform is uploaded from a DSO, the Time to Frequency Domain Transformation can be used as a spectrum analyser to examine harmonics down to the sensitivity of the DSO.

### Frequency to Time Domain Transformation

A waveform can be created in the time domain using expressions or the graphical waveform editor, transformed to the frequency domain, edited graphically, and then returned to the time domain to see the results of the editing. A waveform can also be created in the frequency domain, in the amplitude vs. frequency window, using the graphical interface. You cannot create a waveform in a frequency-domain window using the expression option.

> *You can start with a mathematically expressed waveform in the frequency domain window by entering the expression in a time-domain window and pasting the resulting waveform to the frequency-domain window.*

Once a function is created in the frequency domain, its corresponding time-domain function can be displayed by selecting the time domain window from this menu.

Any waveform in the Frequency Domain can be subjected to an inverse Fast Fourier Transformation and thus returned to the time domain, by selection from the DSP menu.

Any editing in the Frequency Domain is preserved, so it is easy to change the harmonic content of a waveform by using Draw, then transforming to the Time Domain to see the resulting waveform shape.

## FFT Details and Windowing

In general, Fourier analysis results in describing a time domain waveform in the form of the real and imaginary parts of the complex frequency domain. These real and imaginary parts can be easily converted into the more familiar frequency domain amplitude and phase components.

Continuous Fourier analysis requires that the time domain waveform be completely described as mathematical equations. However, any time domain waveform can be sampled and digitized. This results in turning the waveform into an array of values. WaveForm DSP2 creates such an array of values for each waveform stored in a *.wfm file. The Discrete Fourier Transform (DFT) can then transform the array into complex frequency domain values. The advantage of the DFT is that it does not require mathematical equations for the original time domain waveform.

The Fast Fourier Transform (FFT) is a computer algorithm developed for quickly computing the DFT.

After much manipulation, the Continuous Fourier Transform has the form:

$$X(f) = \int x(t) e^{-j2\pi f(t)} dt = \mathrm{Re}(f) + j\,\mathrm{Im}(f)$$

Where the integral value is evaluated from -∞ to +∞ and the result is the real and imaginary values in the frequency domain. The results can be converted into magnitude and phase through the following rectangular to polar coordinate subversion:

$$|X(f)| = \left[\mathrm{Re}(f)^2 + \mathrm{Im}(f)^2\right]^{\frac{1}{2}}$$

$$\varnothing(f) = \tan^{-1}\left[\mathrm{Im}(f) / \mathrm{Re}(f)\right]$$

The DFT is derived from the continuous form as follows:

$$DFT = x_d(n\Delta f) = \Delta t \bullet \sum x(n\Delta t) e^{-2\pi k \Delta f n \Delta t}$$

$$= x_d(k) = \sum x(n) e^{-2\pi kn/N}$$

$$= 1/N\left[\sum x(n)\cos(2\pi kn/N) - jx(n)\sin(2\pi kn/N)\right]$$

Where the summation is evaluated from $n$=0 to $n$=($N$-1) and:

$N$ = **Size** (number of samples)
$\Delta t$ = **Sample time**
$N\Delta t$ = Block period
$\Delta f$ = **Block frequency** ($1/N\Delta t$)
$n$ = summation index (0 to N-1)
$x(n)$ = value of time waveform at $n_{th}$ sample

The final form of the DFT equation lends itself to iterative solution using the digital computer. This form results in the real and imaginary components which are converted to the magnitude and phase components as before. The computer algorithm to compute DFT requires $N^2$ operations. The FFT is a more efficient DFT algorithm that requires only $N\log_2 N$ operations to complete a transform. The FFT is able to realize this simplification by taking advantage of symmetry and periodicity obtained by considering the time domain waveform array as a rectangular "window" taken from a continuous waveform. The continuous waveform can be realized by repeating the window. The FFT gains further efficiency by making a value of "*N*" (size ) a power of two. An additional simplification is to realize that a certain point of a sinusoid is indistinguishable from points $2\pi n$ radians away in phase. Thus phase is calculated in the $2\pi$ interval of $0\pm\pi$, resulting in modulo $2\pi$ FFT.

Since the FFT assumes that the digitised time domain waveform data is taken from a rectangular window of a continuous wave-train, some care must be taken in defining the way the original waveform fits in the window. It helps if you visualise DSP's waveform window as a cylinder with the x-axis wrapped around it (like a rubber band around a rolled-up newspaper).

The final point in your waveform has to connect smoothly to the first point or there will be a phase discontinuity. If you are doing FFT analysis it is best to define integer numbers of cycles within the window.

## 3.14 Library Menu

The Library is a place to build up a non-volatile store of waveforms or waveform segments. The waveforms stored in the library can be copied to any waveform window or spliced into any existing waveform.

Waveforms are copied to the library via the clipboard. However, copying from the library is not the same as copying from another file.

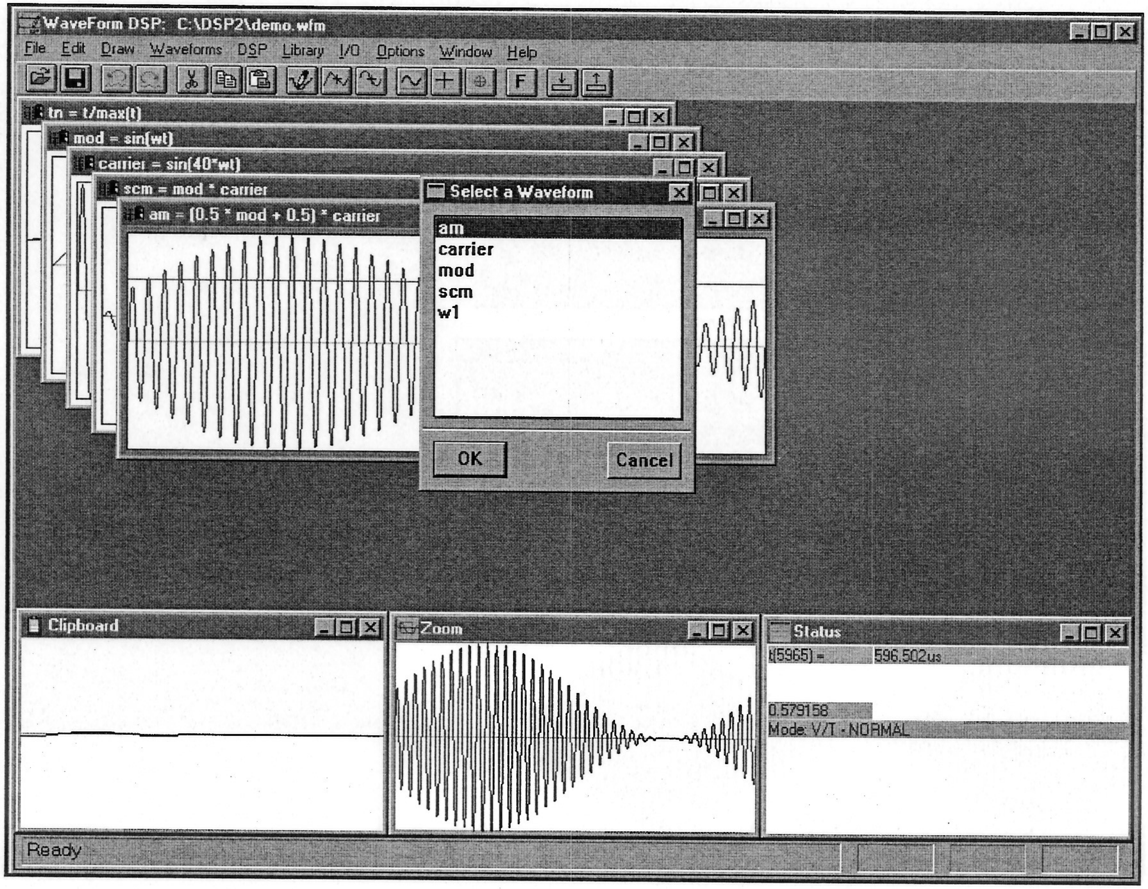
The method of splicing from the library gives you more control over the horizontal and vertical dimensions of the copied segment. This is a way to stretch a waveform to fill a window or to span a specific portion of a window.

Selecting a waveform from the library menu puts you in the library mode of operation (changes cursor to a cross-hair), which you must exit by striking the ESCAPE key (or clicking the right mouse button, if available).

- Select... Select an existing waveform from within the library.
- Add... Add an existing waveform to the library.
- Delete... Delete an existing waveform from within the library.

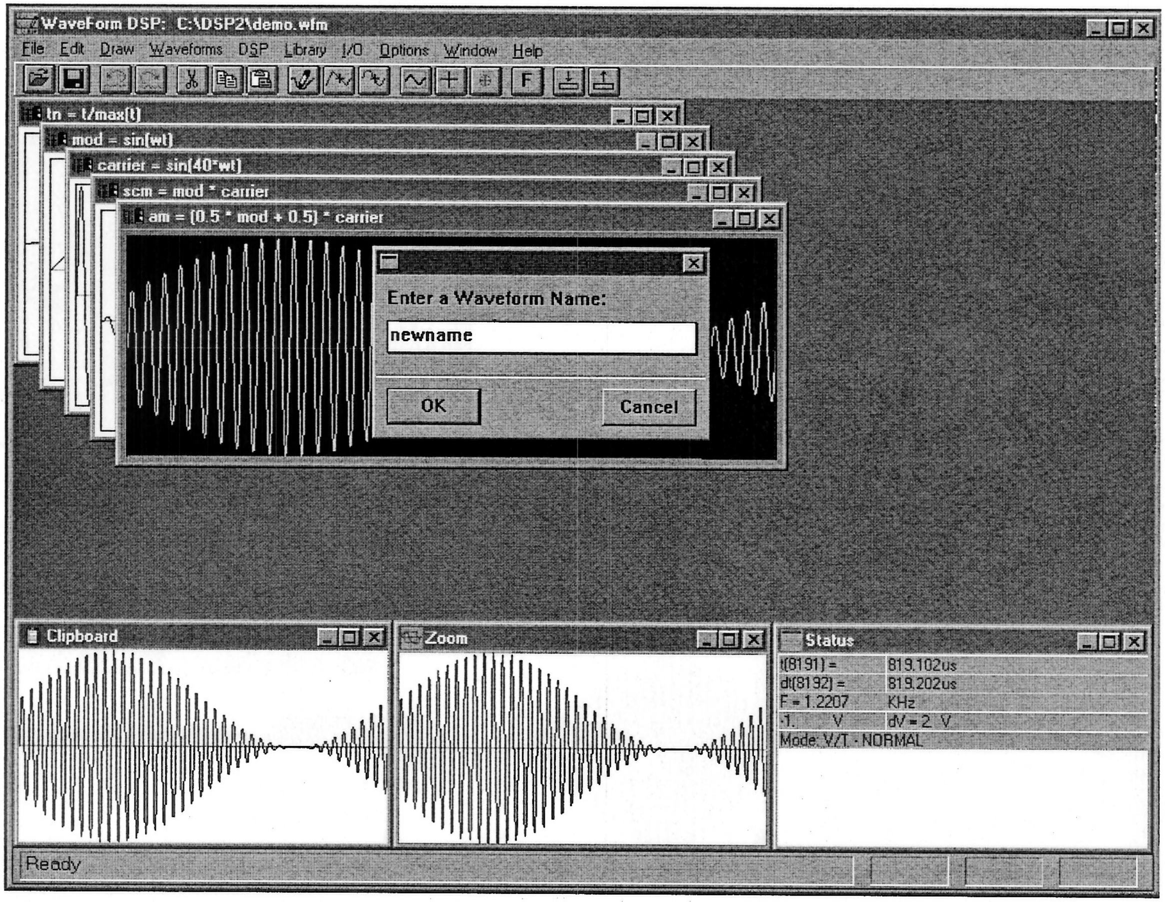
## Select...

Selects a library waveform from a list in the dialog box. You must have the target waveform (or empty waveform window) activated, before opening the library menu.



### Add...

Adds the selected waveform to the library. The waveform you are adding must be in the clipboard before you open the library menu.




### Delete...

Selects a waveform from a list in the dialog box, and deletes it from the library. The dialog box is the same as used in library waveform selection.

## 3.15 I/O Menu

Executes upload from, or download to a target instrument that you have selected. Specific parameters for the download or upload must be determined within the set-up dialog boxes in the **Upload...** and **Download...** options under the **Setup** menu.

### Execute Download

Executes the download procedure for the Arbitrary Waveform Generator selected from the pick list displayed in the **Download...** option of the **Setup** menu.

The download destination is one of the communication options (GPIB, VXI , COM , File) and is dependent on target and hardware selection.

### Execute Upload

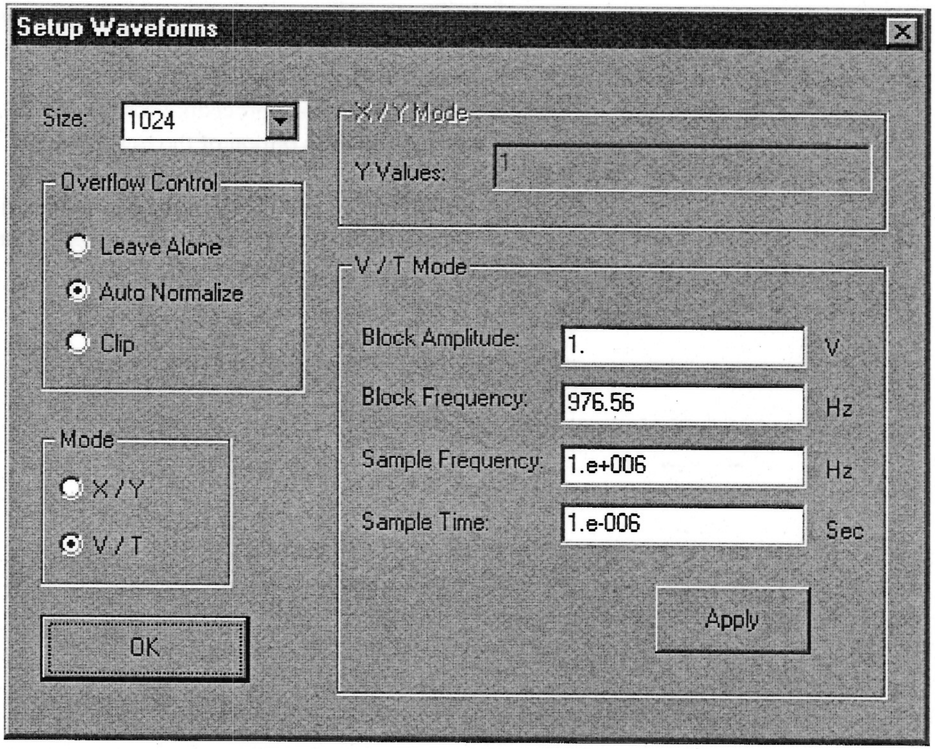
Execute the upload procedure for the Arb or DSO selected from the pick-list displayed in the **Upload...** option of the **Setup** menu.

The upload source is one of the communication options (GPIB, VXI , COM , File) and is dependent on target and hardware selection.

## 3.16 Setup Menu

### Waveforms...

Brings up the dialog box illustrated below. The setup dialog is repeated here for the following discussion. The settings shown in the figure are WaveForm DSP2's start-up defaults.




The Setup Waveforms dialog is brought up each time Waveform DSP2 is started. It can also be brought up at any time with the **Setup** menu **Waveforms... option**. The dialog allows you to set up or to change your preferences for creating, viewing and downloading waveforms.

## Size

The Size parameter determines the number of data points (horizontal resolution) in the waveform you will create . This value defaults at 1024, but may be selected at any power of two value from 8 to 524,288. You can select size using your keyboard or mouse per the standard Windows interface, except when using the keyboard, just enter the first digit of the numeric value you want until the value you want is in the size box.

## Overflow Control

The Overflow Control selections determine how WaveForm will handle creating a waveform when the maximum Block Amplitude or Y-Value is exceeded. This can happen under special conditions (for example, entering too steep a slope when using the Curve Fitting spline algorithm to manually draw a waveform). When you select "Leave Alone" option the "overdriven" values will be stored without change in the Waveform's data file. The default "Auto Normalize" option causes the largest data value in the waveform to be re-scaled equal to the largest allowable vertical value, and each data point in the waveform will be re-scaled accordingly. Finally, the "Clip" option limits data values to the maximum vertical value (clips them off without re-scaling).

## Mode

You can set WaveForm DSP2's waveform data read-out mode to "X-Y co-ordinates" (X/Y) or to "amplitude as a function of time" (V/T) by selecting the appropriate button in the Mode box. Essentially, these are programming models to help you visualize the waveform block in terms you are most accustomed to.

When the default V/T mode is taken, the X/Y mode settings box is "grayed out" so that you cannot make changes. Likewise, when the X/Y Mode is selected, the V/T Mode settings box is 'grayed out'. In the SETup Menu Set-up Waveforms option dialog box with the default settings, the V/T Mode settings are displayed and you may edit any values. The default Block Amplitude of 1 means that the full vertical scale of the waveform block corresponds to a waveform amplitude of plus and minus 1 volt around the x-axis, which is 1 Vp or 2 Vpp delivered by the Arb into its specified termination. The horizontal axis has units of time, which is controlled by Block Frequency, Sample Frequency, or Sample Time. These three parameters are tied together and to the Size value. When you change one of these three values, the other two update their values accordingly when you click "Apply" or make another selection. When you change Size, the Sample Frequency and Sample Time values update. Block Frequency is the rate at which the waveform block is to be "played back" by an Arb. Sample Frequency is the frequency at which the waveform advances from point to point, and equal to Block Frequency multiplied by the Size setting. Sample Time is the time spent at each data point, and is the inverse of Sample Frequency.

When you select X/Y mode, the concept of time and amplitude is removed from waveforms you create, and you are simply creating a block of data with the number of data values equal to the Size setting. The vertical values range from zero at the waveform centre line to plus and minus "Max Y Value", which defaults to 1. You may change this value to anything you are comfortable with. For example, if you will be downloading to an Arb which uses a 12-bit DAC, you might want to set Max Y Value to 2047 so that your waveform read-out is the same as the Arb waveform editor's read-out.

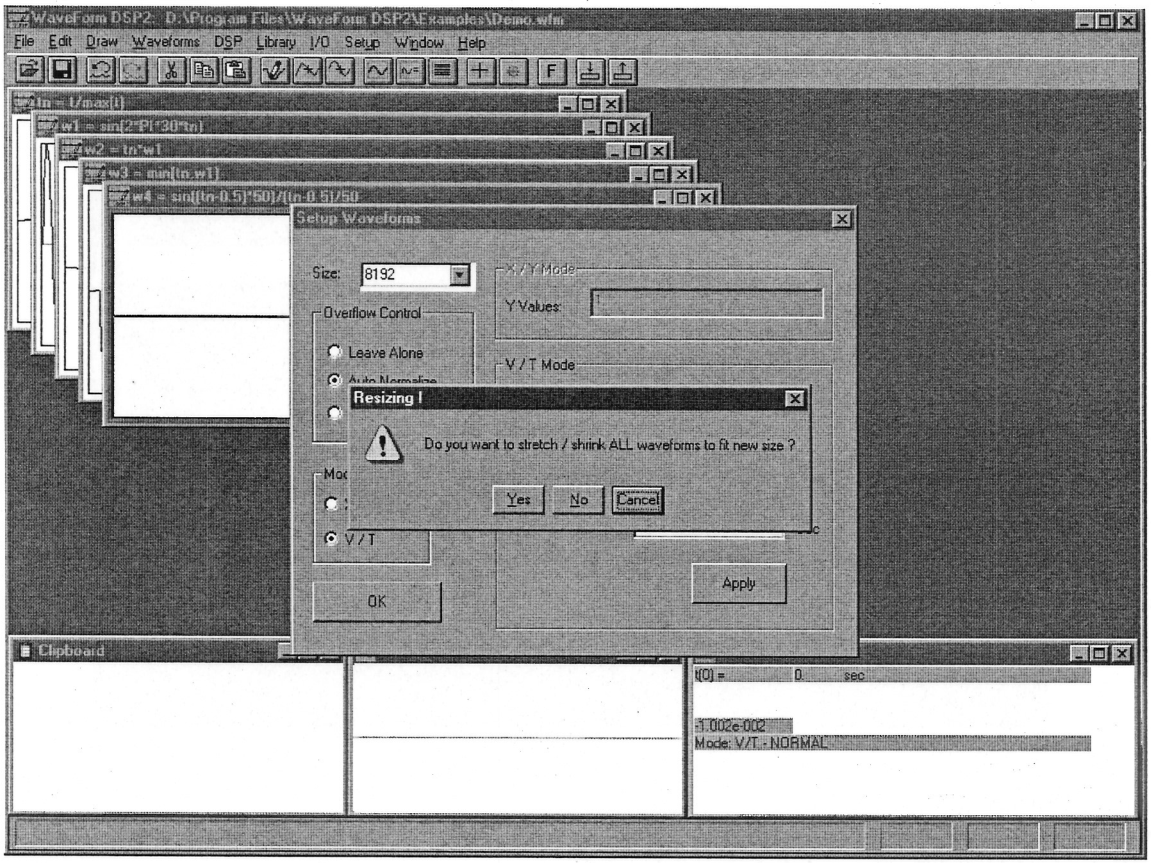
## Using the Size setting to resize Waveforms

You are asked to define the size setting for your waveforms each time you start Waveform DSP2. Any waveform files you create after making this selection will have this size value as the number of points in each waveform. This has a direct relationship on the size of the file when you save it. If you choose to open... an existing .wfm waveform file that you had created at another time and with another size setting.

WaveForm DSP2 does not automatically resize the waveforms in the file to your new setting.

Instead, the size setting, as well as the other preferences selected in the **Setup** menu **Waveforms...** option dialog, will be over-written by the values that had been used to create the file. If you want to change waveform preferences, bring up the dialog box again by selecting the **Waveforms...** option under the **Setup** menu.

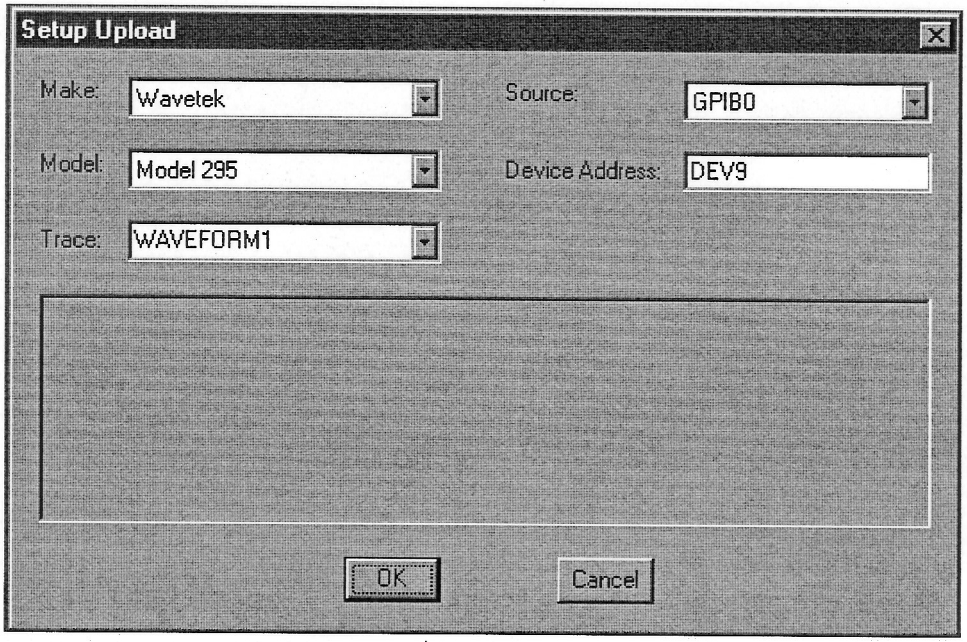
In the following window, the default settings have been changed and size has been set to 8192, "Auto Normalize" has been selected, and the Mode is X/Y.




After selecting "OK" to choose the new settings and close the window, if the size value has been changed from its last selected value, the warning box pops up to inform you that currently open... waveform files (or previously saved waveform files you have open) will be resized horizontally to the new size setting.

- Selecting 'Cancel' allows you to go back to the set-up dialog and make changes.
- Selecting 'Yes' causes the open waveform file to be resized using a "stretch/shrink" process to make the waveforms fit their new block size. Because the number of data points in waveforms files are powers of two values, the stretch or shrink to fit process is very smooth. Shrinking involves removing alternate data points, whereas stretching involves inserting new data points using linear interpolation between existing data points.
- Selecting 'No' sets the new block size, but does not allow the "stretch/shrink"process. Now the waveform data file will be mapped point-by-point into the new block size. If the block size is larger, your old waveforms will start at the beginning of the block, followed by zero data values out to the end of the block. If the new block size is smaller, your old waveforms will be truncated.

## Upload...

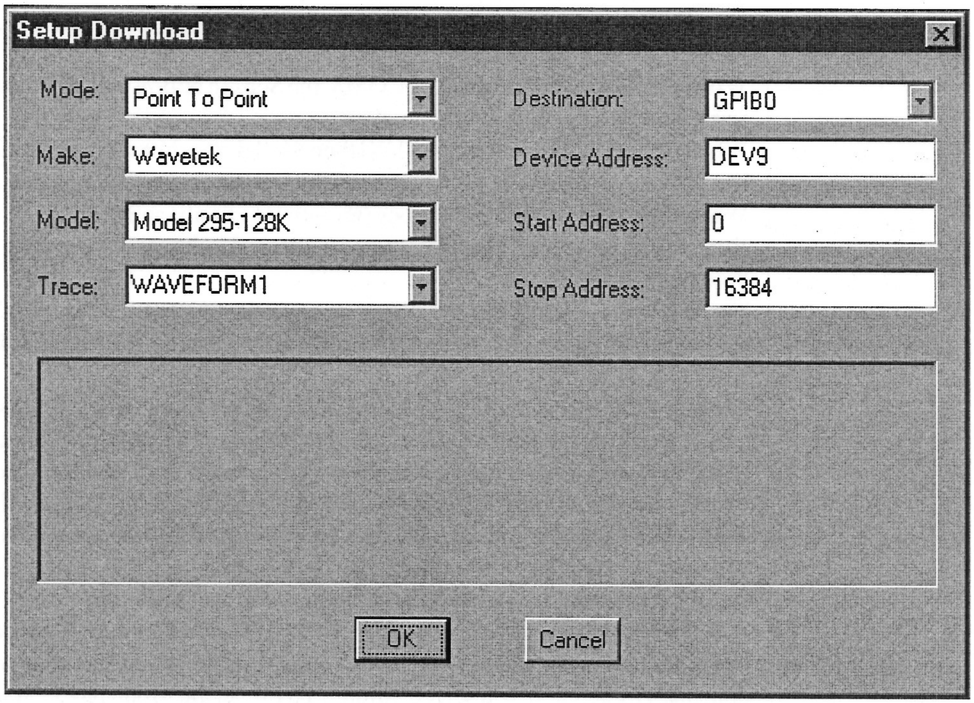



Provides a dialog box for specifying:

- Instrument Make(from a pick-list provided in dialog box)
- Instrument Model (from a pick-list provided in dialog box)
- The location (Trace) of the source instrument trace data.
- Source communication (from pick-lists provided in dialog box).
- Device Address or setting (dependent on Source communication selected)

Select the Manufacturers name of your DSO or ARB from the list under Make. Then select the DSO or ARB model number under Model. Then under Trace, the selections presented match what is available in your DSO or ARB.
The communications selections are interdependent and will default to those saved in the desktop file when a model is selected.
When the Source selected is either GPIB0 or GPIB1 enter the GPIB Address that your DSO or ARB is set to into the Device Address block as "DEVn", where "n" is the GPIB address from 0 to 30.
When the Source selected is COMn pressing Setting will allow modification of the serial port communication parameters.
When the Source selected is VXI then the Address settings available are displayed in an associated pick list obtained from the VXI resource manager which runs when the WaveForm DSP2 application goes through its initialization.
When the Source selected is "File" it allows the upload of a file previously saved by a download file operation These files have a "*.arb" file extension. The selections for Make, Model and Trace are not necessary for file transfers.

## Download...




Provides a dialog box for specifying:

- Instrument Make (from a pick-list provided in dialog box)
- Instrument Model (from a pick-list provided in dialog box)
- Destination communication (from pick-lists provided in dialog box)
- Device Address or setting (dependent on Destination communication selected)
- Start and Stop Addresses (in the Arbitrary Waveform Generator) for the downloaded waveform.
- The location (Trace) and name for the destination of the waveform data.
- Mapping style (Mode) of data from wavefrom to destination.

Select the manufacturer under Make and the model number under Model. Be sure to include an extended RAM option in the model number as shown in the figure. If the Arb is multi-channel or has multi-waveform capability, the appropriate selections for that model will then appear in the Trace block.

The communications selections are interdependent and will default to those saved in the desktop file when a model is selected.

When the Destination selected is either GPIB0 or GPIB1 enter the GPIB Address that your DSO or ARB is set to into the Device Address block as 'DEVn', where 'n' is the GPIB address from 0 to 30.

When the Destination selected is COMn, the Setting button will allow modification of the serial port communication parameters.

When the Destination selected is VXI then the Address button will display those available in an associated pick list obtained from the VXI resource manager which runs when the WaveForm DSP2 application goes through its initialization.

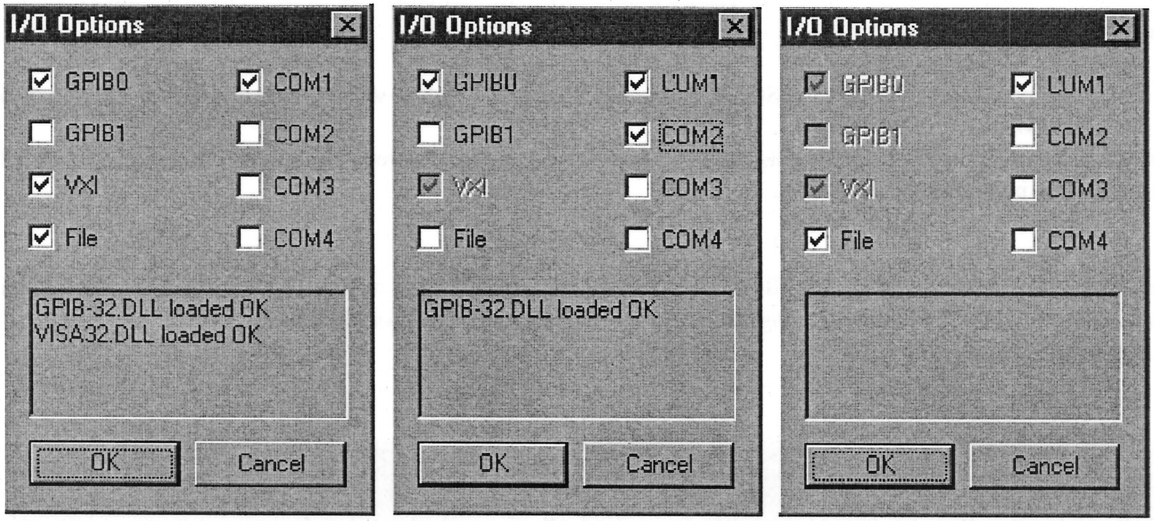
When the Destination selected is "File" it allows the download waveforms to a file, rather than to an Arb.

The file has a "*.arb" file extension and can then be read by another application. The selections for Make, Model and Trace are not necessary for file transfers.

For non-SCPI Arbs, the Start Address defaults to '0' and Stop Address defaults to (size - 1). If required, the Start Address may be changed to any value within the Arb's address space to avoid overwriting existing waveforms in the Arb. The Stop Address will move out to (start + size - 1) if that is within the Arb's address range. If you wish to set the Stop Address to any other setting, note that the number of points you will be downloading is not equal to the Size of the waveform. In this case you will want to look at the Mode setting. The default, "Point To Point", maps the waveform data directly into the Arb's waveform memory. If the value (stop - start +1) is smaller than Size, the waveform will be truncated. If the value (stop - start +1) is greater than Size, the waveform will be appended by zeros. If you select "Stretch To Fit" in the Mode block, your waveform data will be compressed or stretched, as appropriate, to exactly fill the span from start to stop address. The resizing process uses linear interpolation to change the waveform data during the download. It does not modify WaveForm DSP2's data file. Since waveform file sizes are powers of two values, you will get the smoothest results if you download powers of two.

For SCPI language Arbs, there is no fixed start and stop addressing, and WaveForm DSP2 sets the Size setting default to the size selected in the **Setup** menu, **Waveforms...** option. You may change this size setting to download different sized waveforms.

## Hardware...




The three presentations of the Hardware dialog box are given to indicate that the application will detect the presence of appropriate DLL files and load them, and from this infer the communication interface available. If the interface is not available the check boxes are grayed out and the DLL is not loaded. The user can select from the available tick boxes one or more of the available interfaces.

*Check carefully that the COM port(s) selected are available.*

## Load...

Retrieves any SETup menu preferences (this includes Set-up Waveforms, Set-up UPLoad and Set-up DOWnload preferences) that are saved in the desktop file. The retrieved set-up replaces any set-up that is active at the time of the loading.

A dialog box appears, which prompts: 'Are you sure you want to change the desktop file?' You must select the 'Yes' or the 'No' answer, as a precaution, before the load is executed.

## Save...

Saves your current SETup menu preferences (this includes Set-up Waveforms, Set-up UPLoad and Set-up DOWnload preferences) to the desktop file, so that you can move to a different set-up without harming it. You can return to it to complete what you were doing as long as you have not saved another set-up in the meantime.

A dialog box appears, which prompts: 'Are you sure you want to change the current set-up?'. You must select the 'Yes' or the 'No' answer, as a precaution, before the load is executed.

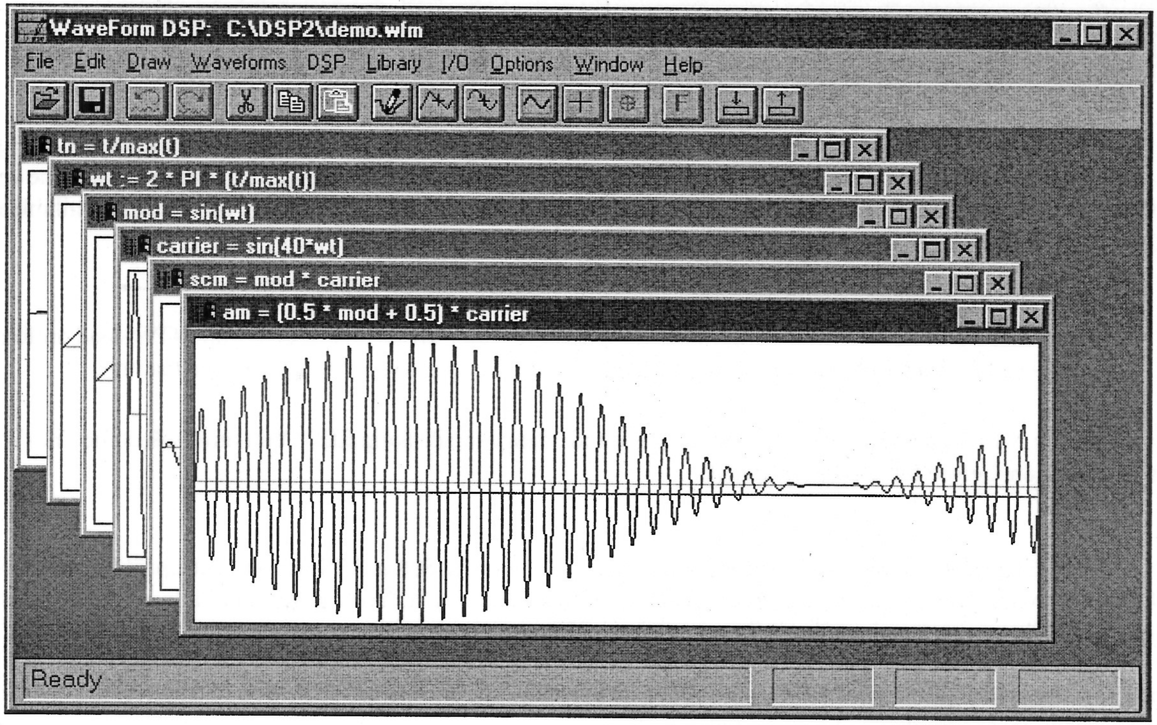
## 3.17 Window Menu

The Window Menu contains options that allow the user to organise the arrangement of Waveform windows within the main Waveform DSP2 screen.

Options in this menu may be used to turn the Status, Clipboard, Zoom and Debug windows on or off, and activate or de-activate the Status line and Toolbar.
Any option window can be closed by clicking its close box (as long as it is not completely hidden by another window), but you must return to the Window menu to turn it back on.

### Cascade

Displays all of the waveforms in the active file. Views are full size and they are overlapped, so that only the top waveform is completely visible. However the views are staggered, so that they are accessible for selection.
Cascade is the default view.



### Tile
Displays all of the waveforms in the active file. Views are small and side by side, so that all waveforms can be seen simultaneously.
Click on a tile to activate it.

### Close All
This will close all of the waveform windows. To restore any waveform window use the **View...** option in the **Waveforms** menu.

### Status
Turns the status window on or off. The Status window is very important:

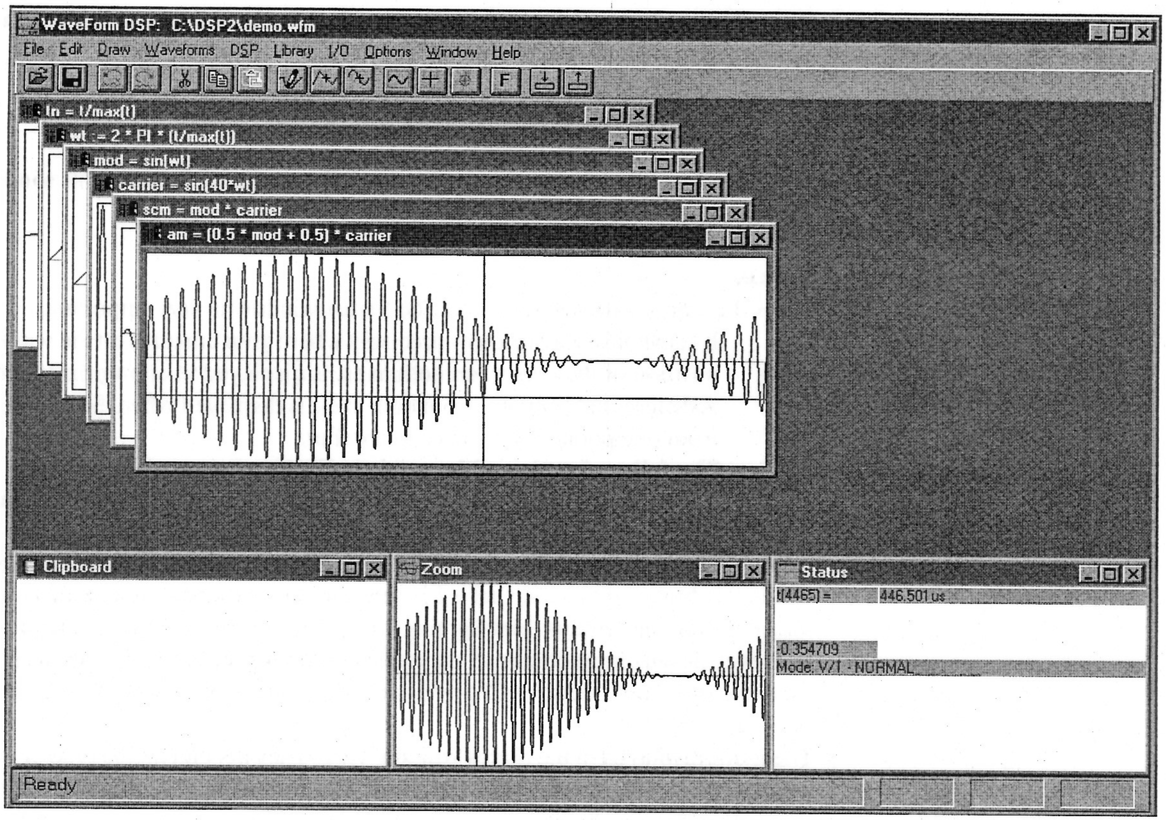
It is the only way to know precisely where your cursor is on the waveform. The position of the cursor, on both axes, is displayed in appropriate units.
A differential position, from both axes, is displayed when the mouse is dragged from one point on the curve to another.
The full-scale value of the y axis is displayed.
The zoom read-out mode can be activated or deactivated from the status window by selecting the Zoom-In button on the toolbar.

Additionally, the waveform is zoomed-in whenever a segment of waveform in the active window is high-lighted and Ctrl + I is selected. Once the read-out mode (zoom box) is selected, any subsequent highlighting of the waveform will automatically zoom in. However, deselecting zoom read-out mode by clicking the Zoom-In button on the toolbar does not zoom out (requires Ctrl + O), it only changes the read-out mode and disables automatic zooming.
The items displayed in the Status window vary. They depend on the type of information being displayed in the waveform window. The following seven figures are required to show all of the possibilities.

V/T Mode - Cursor position and highlighted read out as Voltage/Time co-ordinates
V/F Mode - Cursor position and highlight read-out as Voltage/Frequency co- ordinates
X/Y Mode - Cursor position and highlight read-out as X/Y co-ordinates

V/T Mode - Cursor position read-out

When you are in V/T Mode, the Status window can be used to indicate exact cursor position in appropriate V/T units. The dotted horizontal line through the centre of the waveform window is 0Vdc. Time values range from 0us at the left to 1000*s at the right. Amplitude values range from 0Vdc at the centre-line to -1 volt at the top and -1 volt at the bottom.

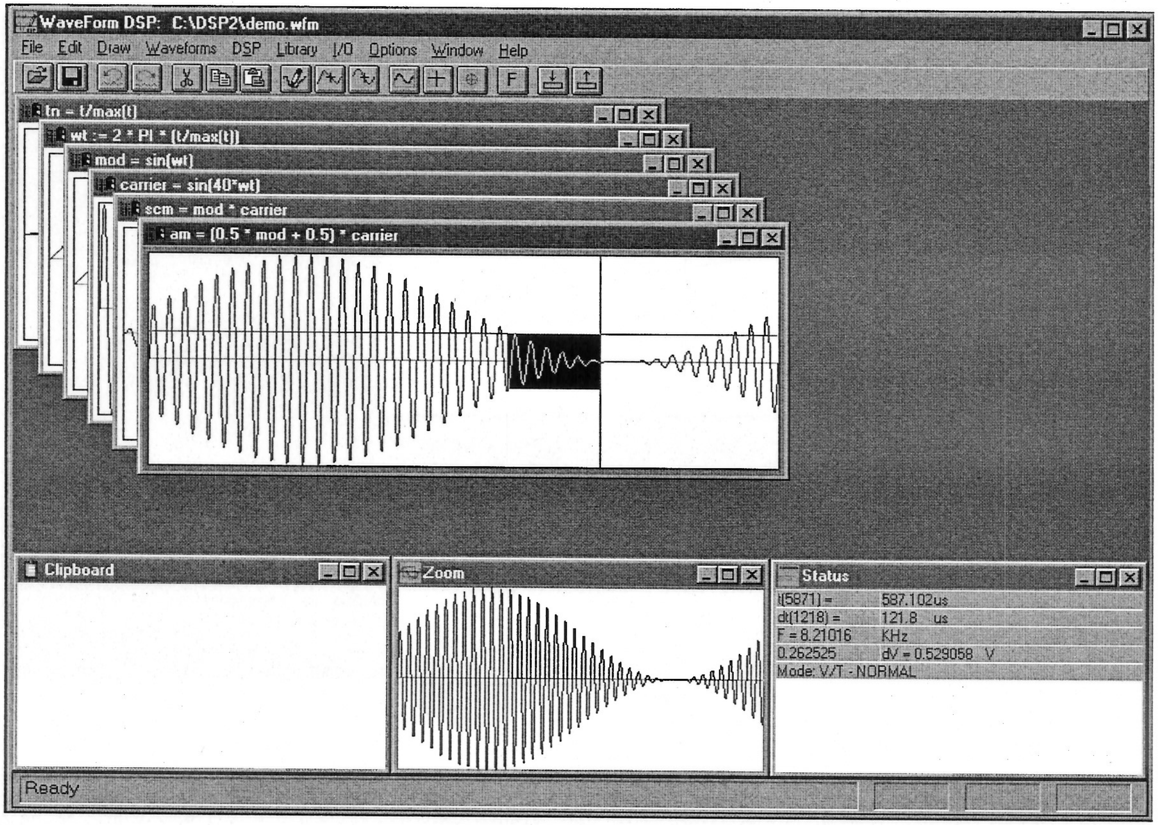



The cursor indicates where a "mouse click" has selected a point of interest on the waveform. A horizontal line and vertical line appear at the selection point and will remain there even when the cursor is moved, until another mouse operation selects a different area. The horizontal line indicates the amplitude value of the selection. The vertical line indicates the time value.

The Status window displays those time and amplitude values of the current cursor position. In this case time is t(4465) = 446.501*s and amplitude is -0.354709 volts.
Note that the Status window also indicates that the read-out is in V/T Mode and the view is Normal (not Zoomed).

## V/T Mode - Highlight read-out

When you are in V/T Mode, the Status window can be used to indicate the time and amplitude dimensions of a highlight as well as exact cursor position at the end of the "mouse drag" in appropriate V/T units.

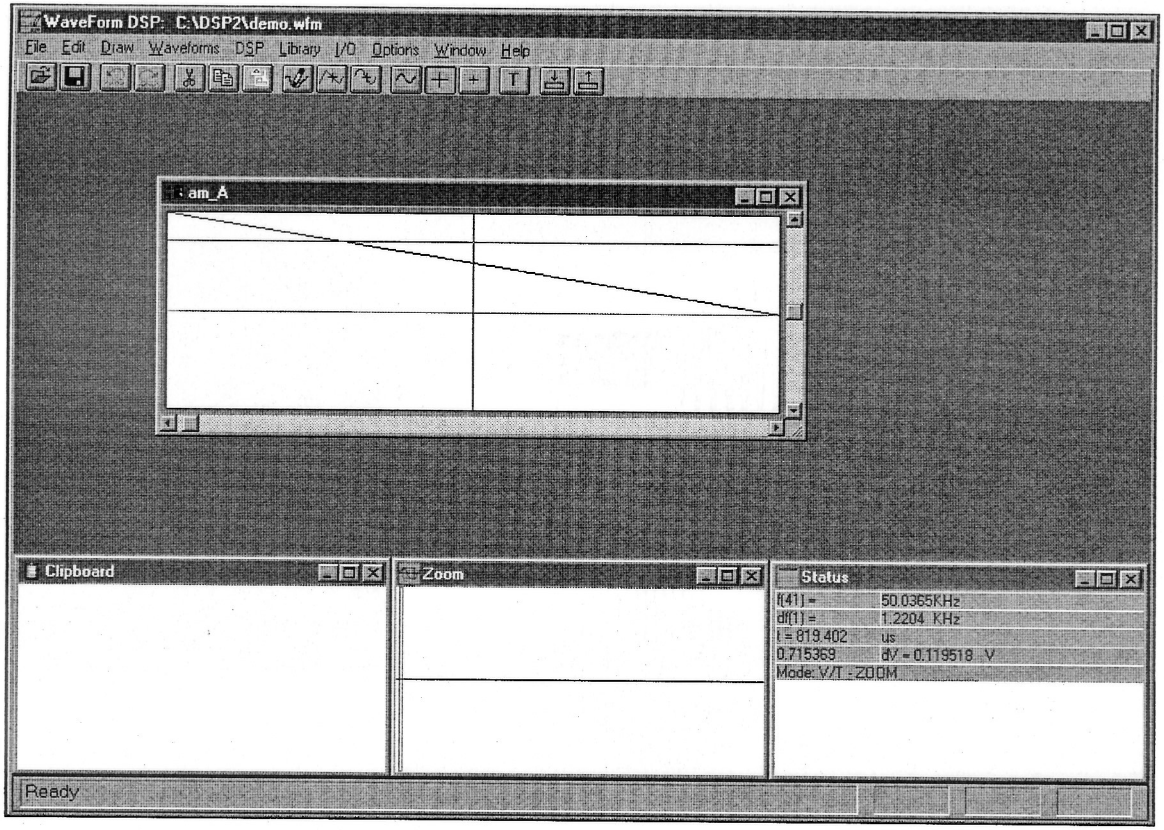



The cursor indicates the position of the end of a "click and drag" to highlight a portion of the waveform. A horizontal line and vertical line appear at the highlight end point and will remain there even when the cursor is moved, until another mouse operation selects a different area. The horizontal line indicates the amplitude value of the end point. The vertical line indicates the time value of the end point.

The Status window displays those time and amplitude values of the current cursor position. In this case time is t(5871) = 587.102*s and amplitude is 0.262525 volts. The length of the "drag" in the horizontal direction is dt(1218) = 121.8*s, and in the vertical direction is dV = 0.529059 volts. The dt value has a corresponding frequency of F = 0.821016MHz.

## V/F Mode - Cursor position and Zoom read-out

In this example, the sine waveform in the previous examples has been transformed into the frequency domain. Since the transformation includes frequencies from dc to the Nyquist point and "smooth" time domain waveforms usually do not contain such high frequency components, most amplitude vs. frequency spectra will have to be zoomed in.



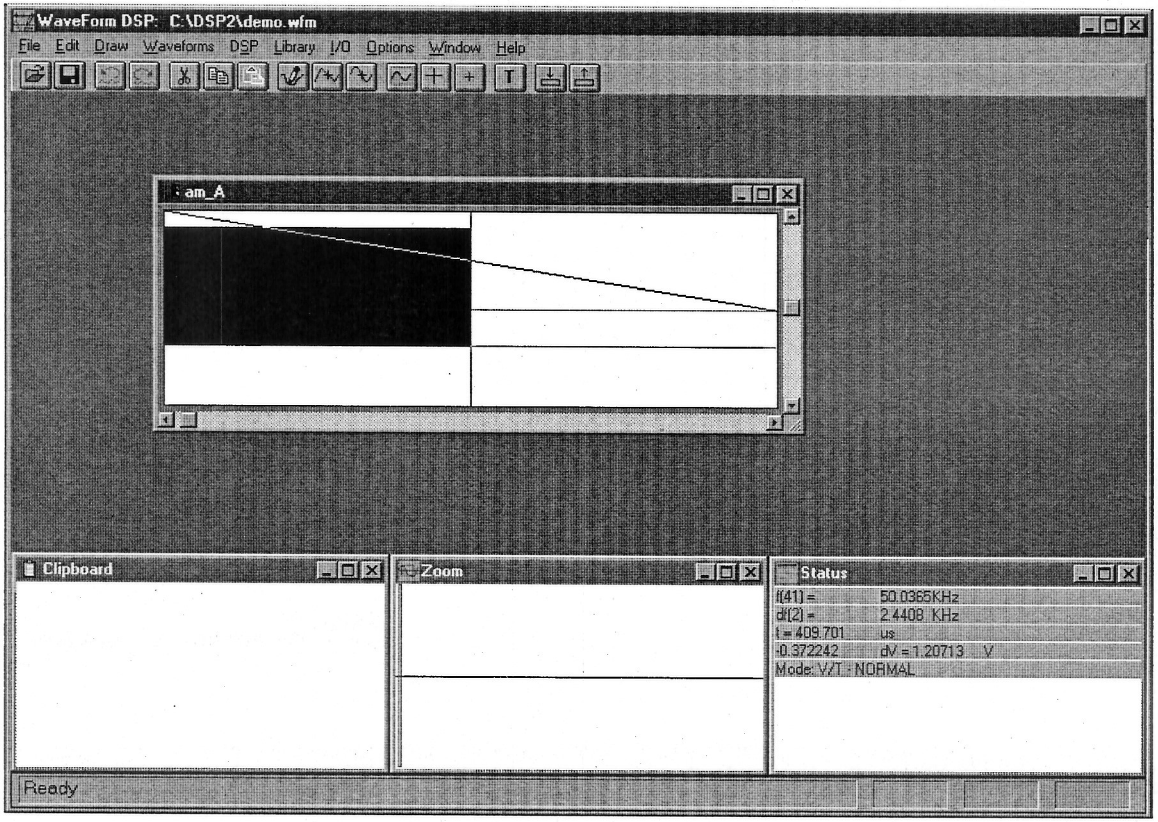

In the figure above, the Zoom window shows a representation of the complete waveform, and also indicates the zoomed-in area with a box. The Waveform window now shows the enlarged waveform segment, and scroll bars appear so that the zoom area position can be finely adjusted.

The Status window is in the Zoom read-out mode as long as the zoom button on the toolbar is depressed. The units are now frequency rather than time. When a "Phase" representation is chosen, the units will be phase in degrees. A mouse click sets the horizontal and vertical line for position read-out.
The vertical line indicates a frequency of f(41)=50.0365KHz and the horizontal line indicates an amplitude of 0.715369 volts. The vertical cursor line can be moved with a frequency resolution of 1kHz, which is the Block Frequency, and the lowest possible frequency difference between spectral lines.

The zoom box has a horizontal dimension of dF(1) = 1.2204KHz and a vertical dimension of dV = 0.119518 volts.

V/F Mode - Highlight read-out

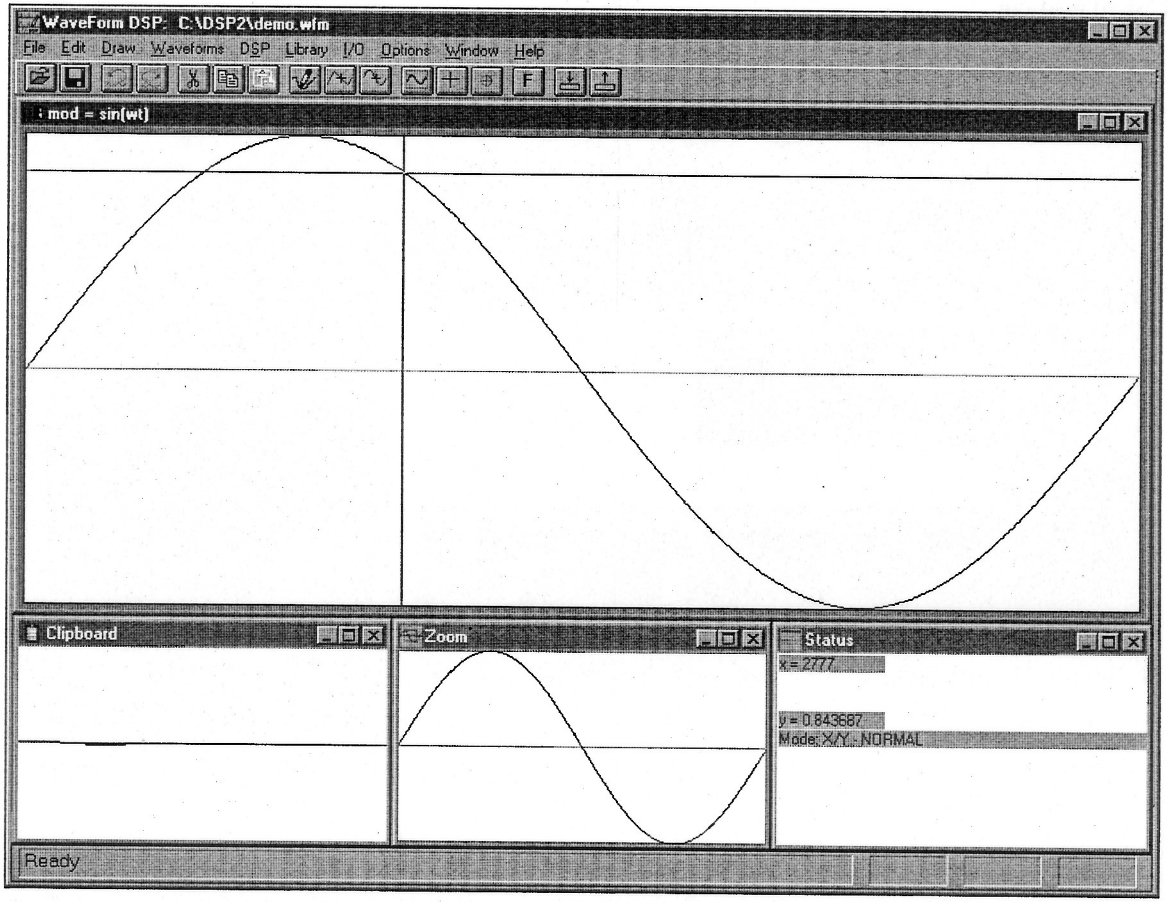
In this final example, the frequency domain sine waveform has an area of interest highlighted for read-out in the Status window. If you wish to highlight a portion of a 'zoomed-in' waveform in order to display parameters in the high-lighted area in the Status window, you must first click on the Zoom button in the toolbar. Otherwise it will zoom in to the new area rather than provide you with the desired read-out.




The Status window now shows the dimensions of the highlight (not the Zoom area as before) and the position at the upper right hand corner at the end of the mouse "drag".

## X/Y Mode - Cursor Position read-out

When you are in X/Y Mode, the Status window can be used to indicate exact cursor position in appropriate X/Y co-ordinates. The figure below shows a sine wave created in X/Y co-ordinates with 8192 points horizontally and Y max = 1. The dotted horizontal line through the centre of the waveform window is the x-axis. X values range from 0 at the left to 8191 at the right. Y values range from 0 at the x-axis to +1 at the top and -1 at the bottom.



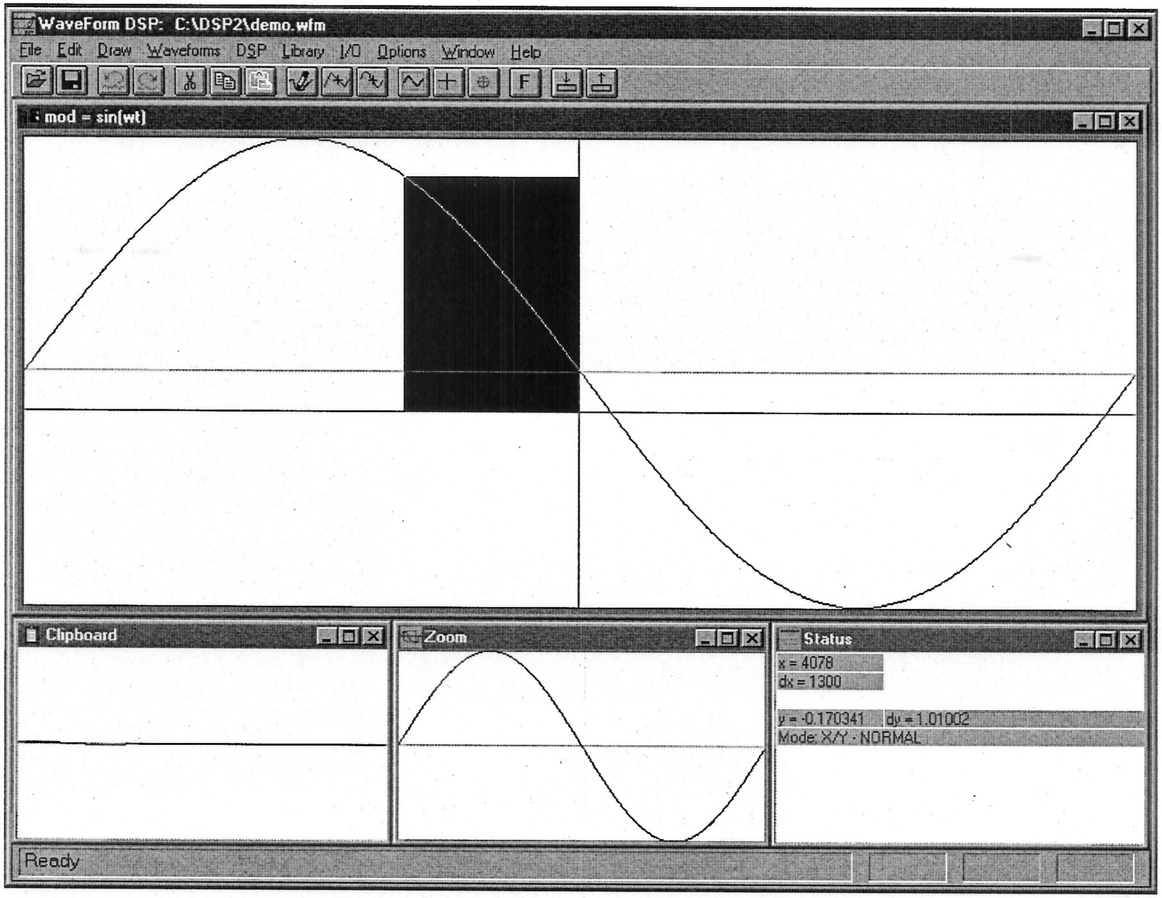

The cursor indicates where a "mouse click" has selected a point of interest on the waveform. A horizontal line and vertical line appear at the selection point and will remain there even when the cursor is moved, until another mouse operation selects a different area. The horizontal line indicates the vertical Y value of the selection. The vertical line indicates the horizontal X value.

The Status window displays those X and Y values of the current cursor position. In this case X = 2777 and Y = 0.843687. Note that the Status window also indicates that the read-out is in X/Y Mode and the view is Normal (not Zoomed).

## X/Y Mode - Highlight read-out

When you are in X/Y Mode, the Status window can be used to indicate the X and Y dimensions of a highlight as well as the exact cursor position at the end of a mouse drag.

The cursor indicates the position of the end of a "click and drag" to highlight a portion of the waveform. A horizontal line and vertical line appear at the highlight end point and will remain there even when the cursor is moved, until another mouse operation selects a different area. The horizontal line indicates the vertical Y value of the end point. The vertical line indicates the horizontal X value of the end point.



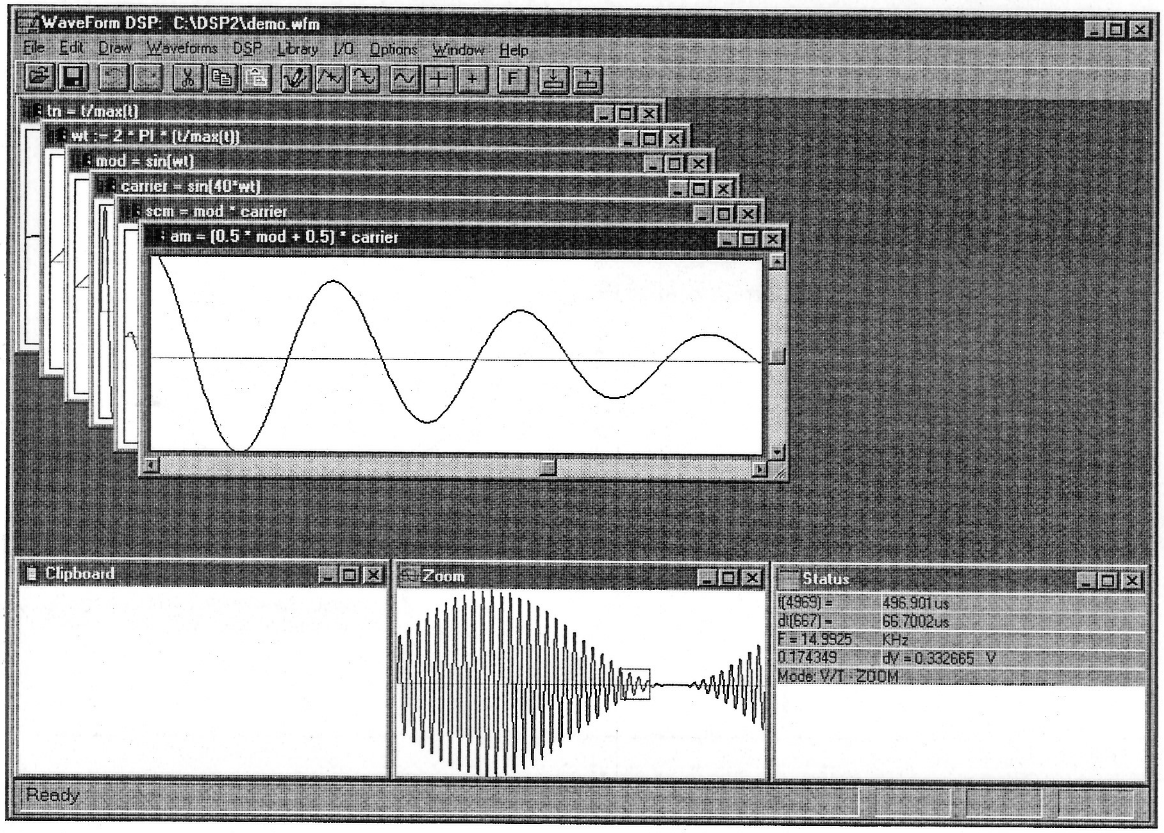

The Status window displays those X and Y values of the current cursor position. In this case X=4078 and Y=-0.170341. The length of the "drag" in the horizontal direction is dx = 1300 points, and in the vertical direction is dy = 1.01002.

## Zoom Mode read-out

In this example, the waveform in the Amplitude Modulation example has been "zoomed-in" to view a smaller feature in detail. To zoom in, you highlight the area and then either press Ctrl + I with the keyboard, select the Zoom-In option of the Waveforms menu with the mouse, or simply click the Zoom button inside the toolbar. Any of these actions zooms in the selected segment of the waveform and checks the zoom box. As long as the zoom button remains depressed, subsequent highlighting of portions of the waveform inside the waveform window will be zoomed in even further.

If you wish to highlight a portion of a "zoomed-in" waveform in order to display parameters of the highlighted area in the Status window, you must first click on the Zoom button in the toolbar. Otherwise it will zoom in to the new area rather than provide you with the desired read-out.

The Status window remains in the zoom read-out mode as long as the zoom button in the toolbar is depressed.



### Clipboard

Turns the Clipboard window on or off. All, or any portion of, any waveform in the WaveForm application files can be cut or copied to the Clipboard. The waveform in the clipboard can then be pasted to any waveform window in the WaveForm application. An ASCII data file of the pasted waveform segment is also sent to the external Windows clipboard for other applications. Changes made to the Windows clipboard do not update the waveform segment in the WaveForm DSP2 clipboard.

The content of the clipboard remains in the clipboard after a paste, and can be pasted repeatedly, until another save to the clipboard replaces it.

{bmc clipboard.bmp}

In the example above, the highlighted portion of the waveform "am" has been copied to the clipboard and then pasted back into the waveform "example". Note that the copied waveform segment completely fills the clipboard window, but it pastes into the new waveform with its original dimensions.

### Zoom Window

Turns the Zoom window on or off. The Zoom window always displays the entire currently selected waveform. With Zoom off, this is the same as the current waveform window. When a portion of the waveform is highlighted in the waveform window, and then Zoom-In is selected, the Zoom window shows a box around the zoomed portion displayed in the waveform window. When the waveform window zoom display is fine adjusted using the scroll bars, the box representing the zoomed area moves in synchronisation with the scroll. As long as the zoom mode is active (the zoom button is depressed in the toolbar), any subsequent highlighting of portions of the waveform will cause the display to zoom in even further.

### Debug Window

Turns the Debug window on or off. This window is used to provide debug data when developing new scripts for use mainly during upload and download of source or destination instrument data. The amount of data provided is dependent on the construction of the script. The semicolon (;) that appears at the end of the script statements disables the result of that statement from appearing in the debug window. Temporary removal of the semicolon will provide the statement results for perusal. It is advisable to experiment with a few lines initially as the amount of data that can be contained in this scrolling window is limited. Temporarily edit the script in question and then restart WaveForm DSP2. The scripts are interrogated during startup and thus any changes will only be recognised after a restart.

*It is useful to save the current setup before exiting WaveForm DSP2 for a revision of the script as the Waveform, Upload, and Download settings are reinstated from the Save during a restart.*

### Activate/deactivate the toolbar and status line

## 3.18 Help Menu

### About
Gets on-line information about WaveForm DSP2 versions and copyrights.

### Help
Display on-line help for Waveform DSP2 commands, features and functions.

# 4 Working with the Editor

The procedures described here can be performed in sequence, as a tutorial, or reviewed individually as reference material.

Most of the operations described in this section can be performed using either the mouse or other compatible pointing device, or the keyboard. however. some operations require a pointing device.

Where possible the procedures are described in terms applicable to both methods. For example, the word "select" means to click the mouse button with the cursor on the selection or to press the RETURN key with the cursor on the selection. Where necessary for the sake of clarity, procedures are described for the mouse only.

## 4.1 Creating a new Waveform

There are many ways to create a new waveform using the editor. In the following example we start with an existing waveform and change it using a variety of methods, including drawing portions of the waveform by moving the cursor in the Draw mode.

Describing waveforms and portions of waveforms mathematically is done in the expression mode. The expression mode is described in the section "Working with Equations".

When creating waveforms by mathematical expressions keep in mind that:

- The waveform may be normalized within the waveform window, independent of the full-scale value assigned to it in the Set-up Waveforms dialog box.
- The values from the Set-up dialog box are the basis for the value read-outs in the status window, and where applicable, the values downloaded to the Arb.
- When an expression is entered, it is evaluated. If it is valid, it is displayed. Otherwise a dialog box appears to notify you that an error has occurred. "Variable" errors refer to illegal waveform names, otherwise the error is in the expression.

## 4.2 Changing an existing Waveform

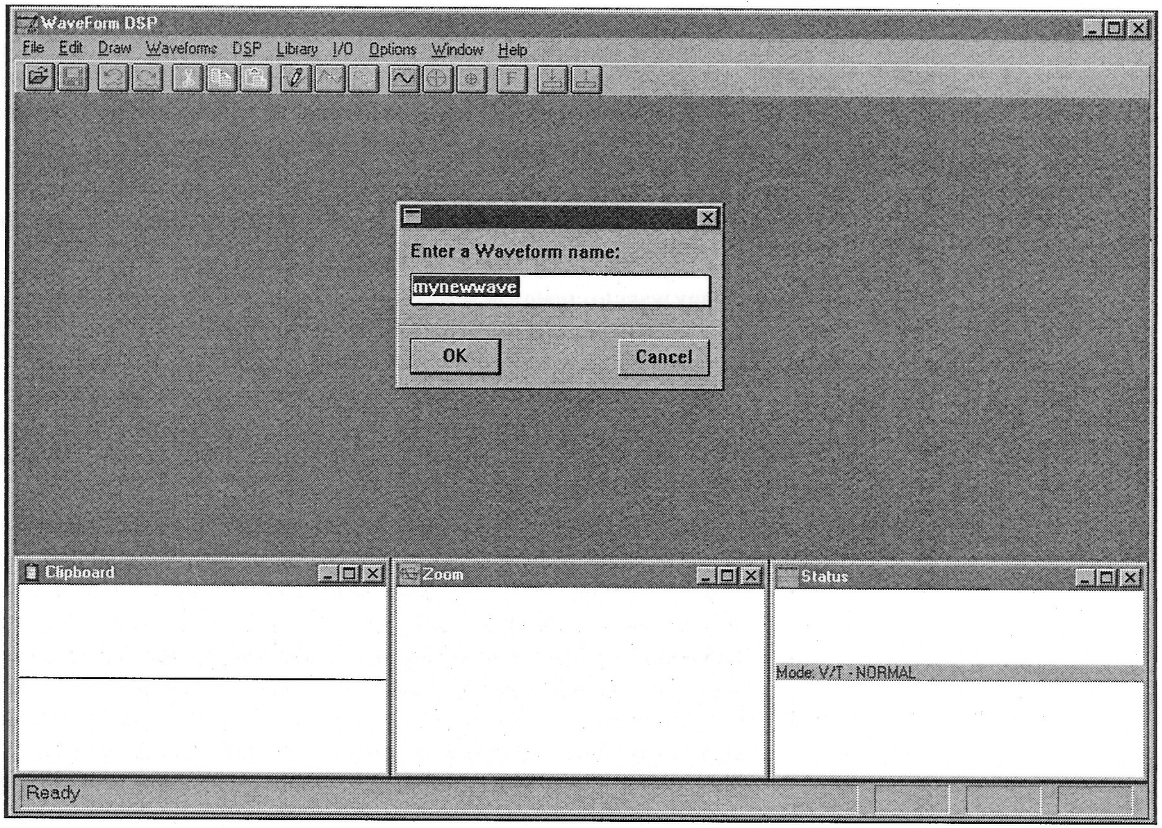
When changing an existing waveform, it is safer to copy it first to avoid accidentally damaging the original:

### Copying an existing Waveform as a starting point:

1.Select the **File** menu, **New** option to clear the main WaveForm window.

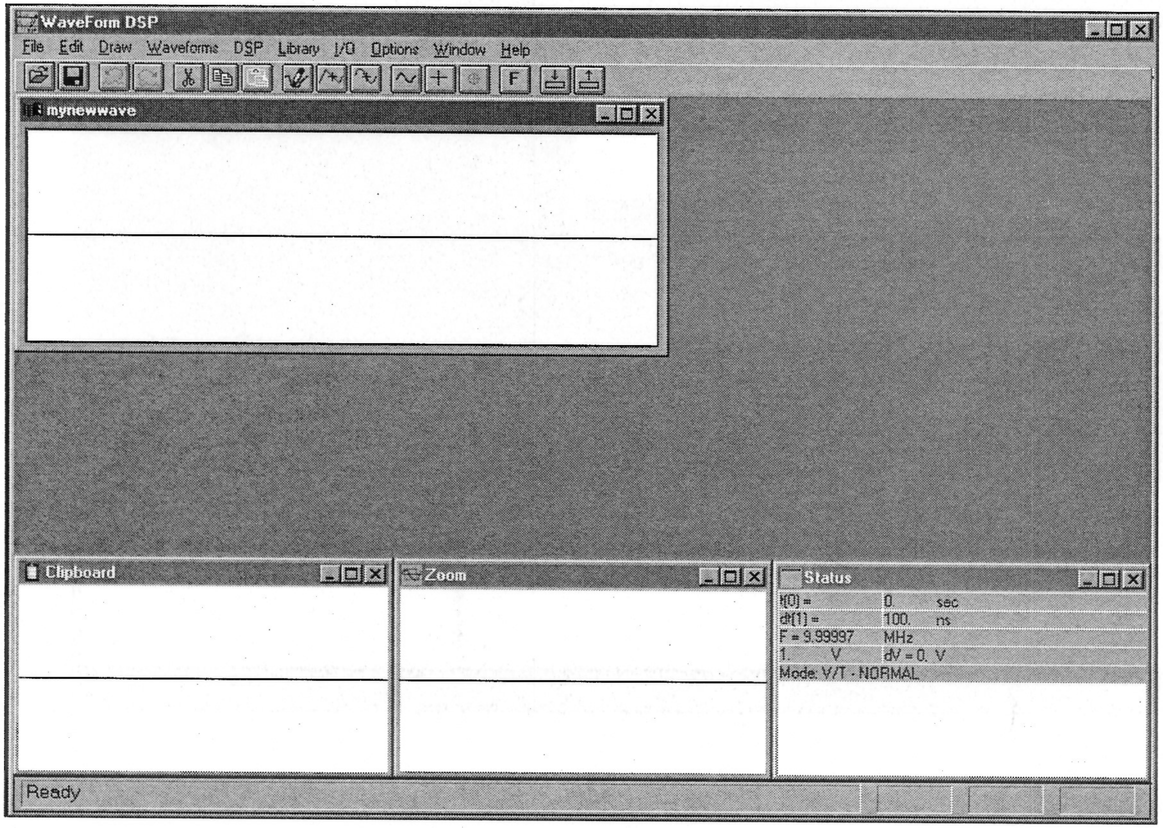
2.Select the create button from the toolbar  or select from the **Waveform** menu the **Create** option, or use the accelerator **Ctrl+N**.

*A dialog box appears, prompting you to name the new waveform.*



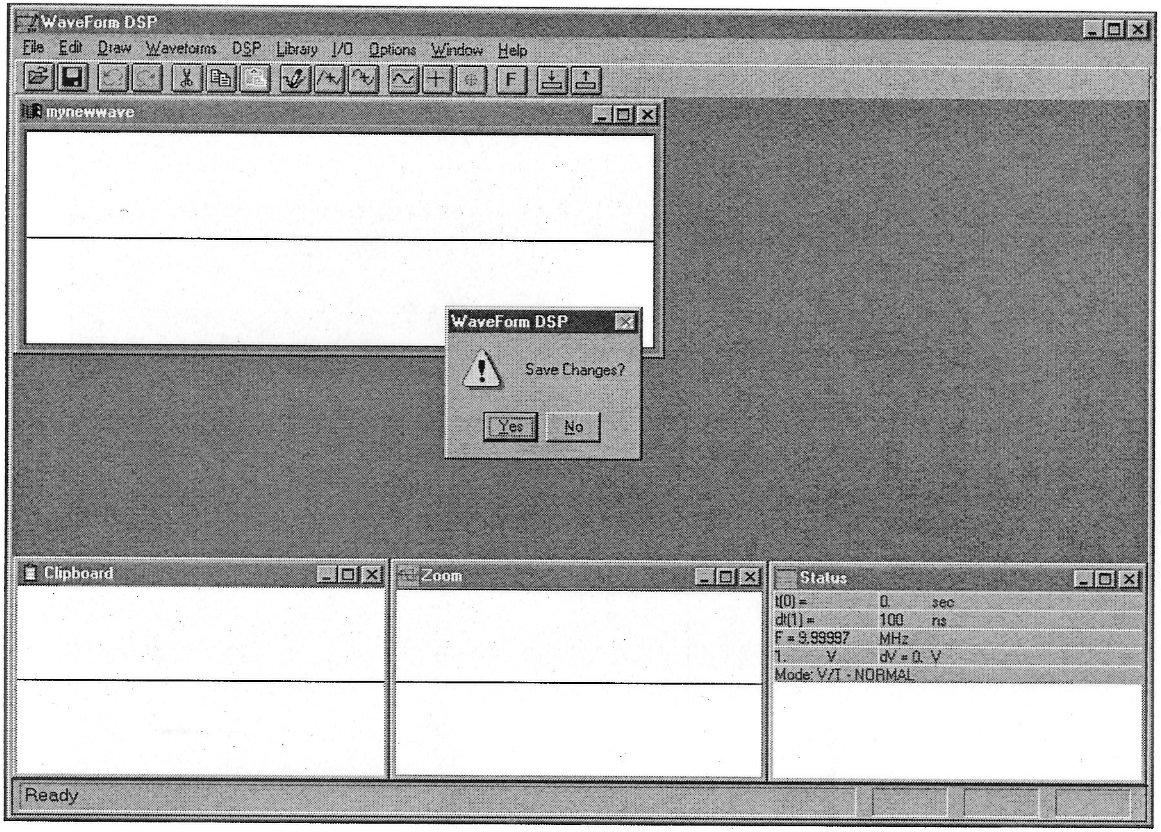

3. Type “mynewwav” into the box, and strike ENTER.

*An empty waveform window appears*

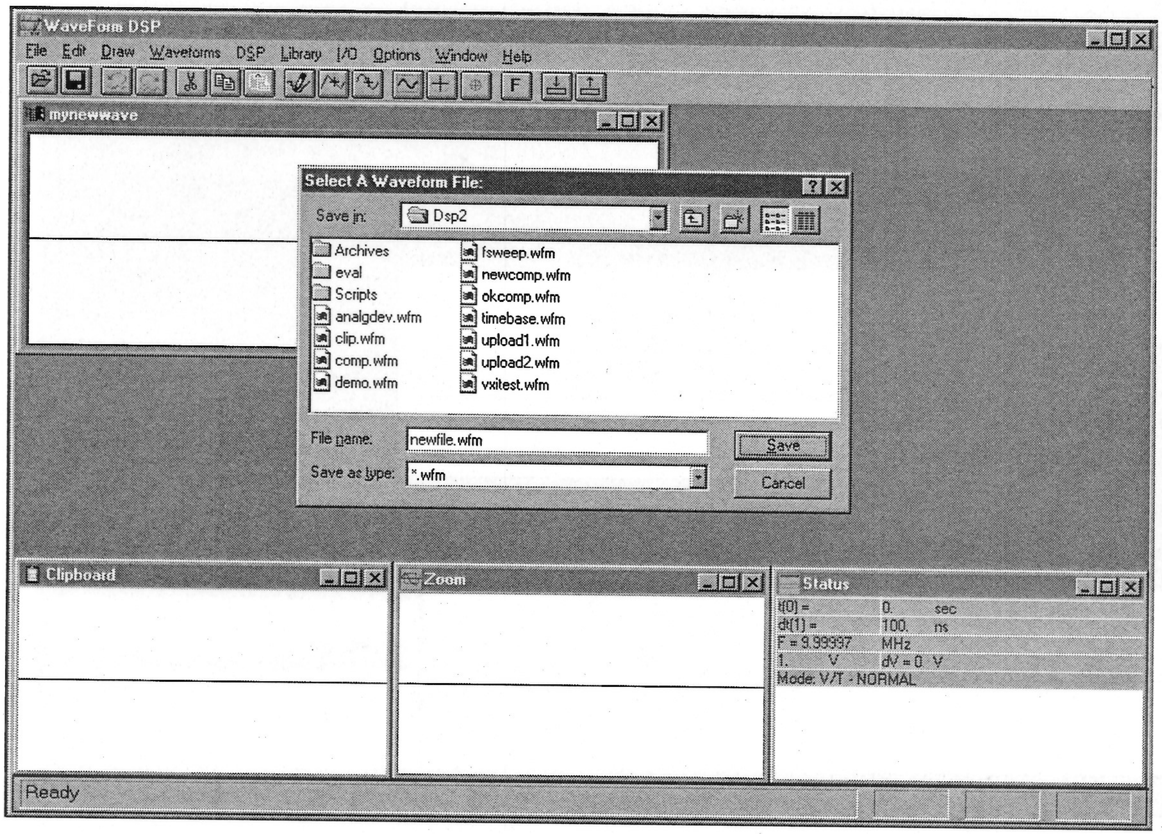



4. Select the Open button from the toolbar  or select from the **File** menu the **Open...** option.

   *A dialog box appears, prompting you to save your new waveform in a file.*

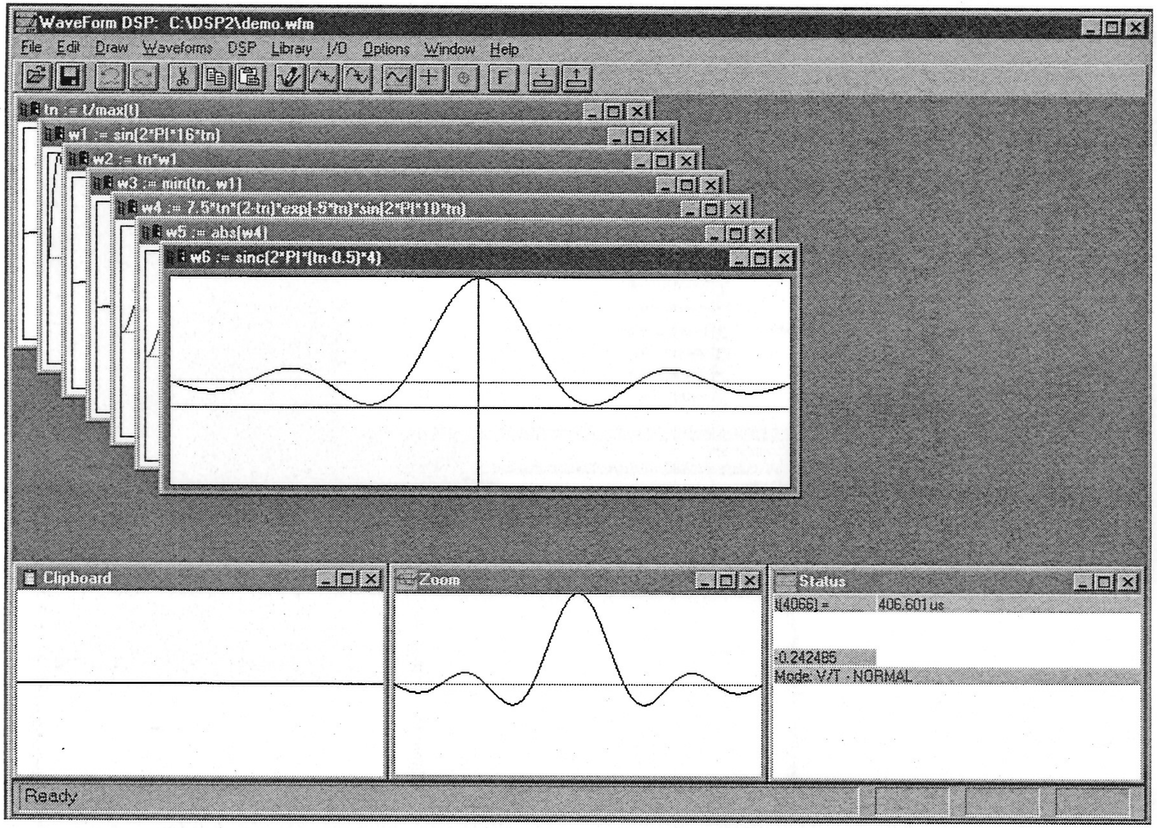



5. Select "Yes".
   *A dialog box appears, prompting you to name a .wfm file.*
6. If you are running a tutorial, type the new file name "newfile.wfm" into the filename box.

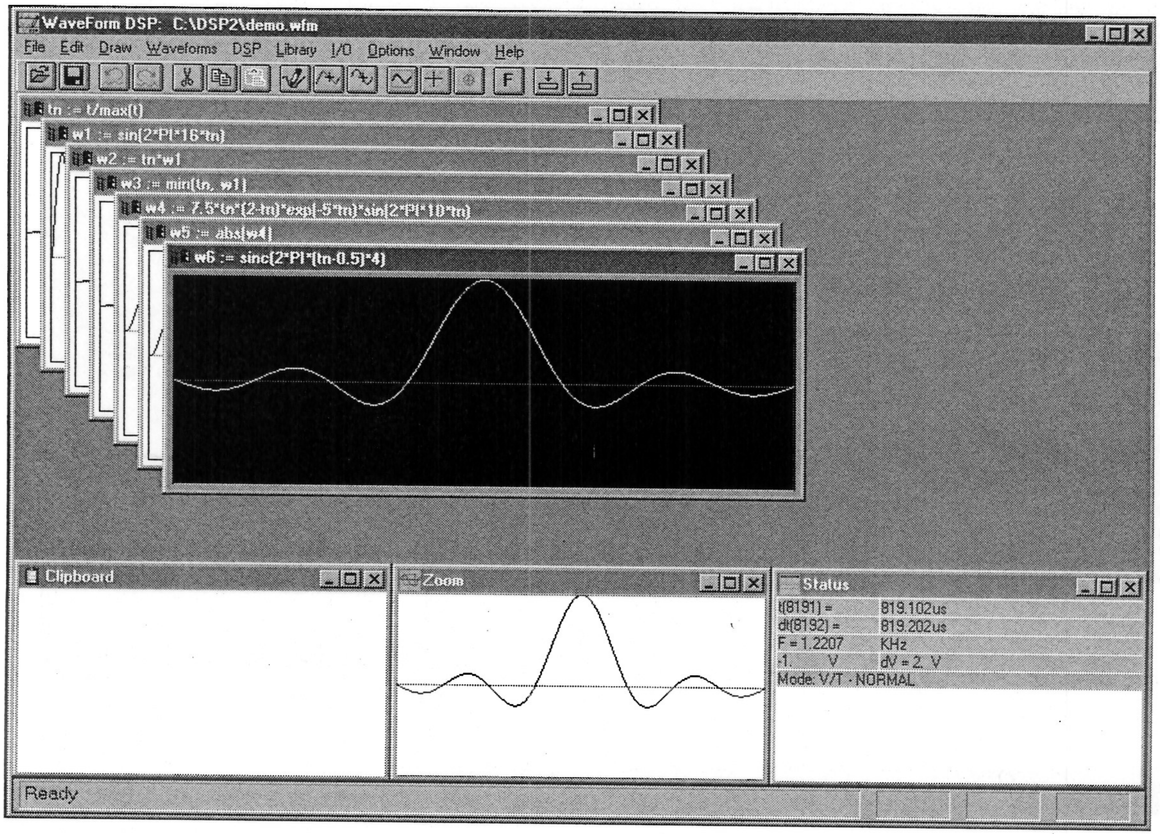



*Once you have saved your new file, the* ***File*** *menu,* ***Open...*** *option dialog box appears*

7. Select and open the file containing the waveform you want to copy. If you are running a tutorial session, select "demo.wfm" from the file list.
   *The file window opens showing all of the waveform windows*




8. Select the waveform you want to copy
9. Select from the **Edit** menu, the **Select All** option or the accelerator **Ctrl+A**.
   *The selected waveform window changes to a black background.*



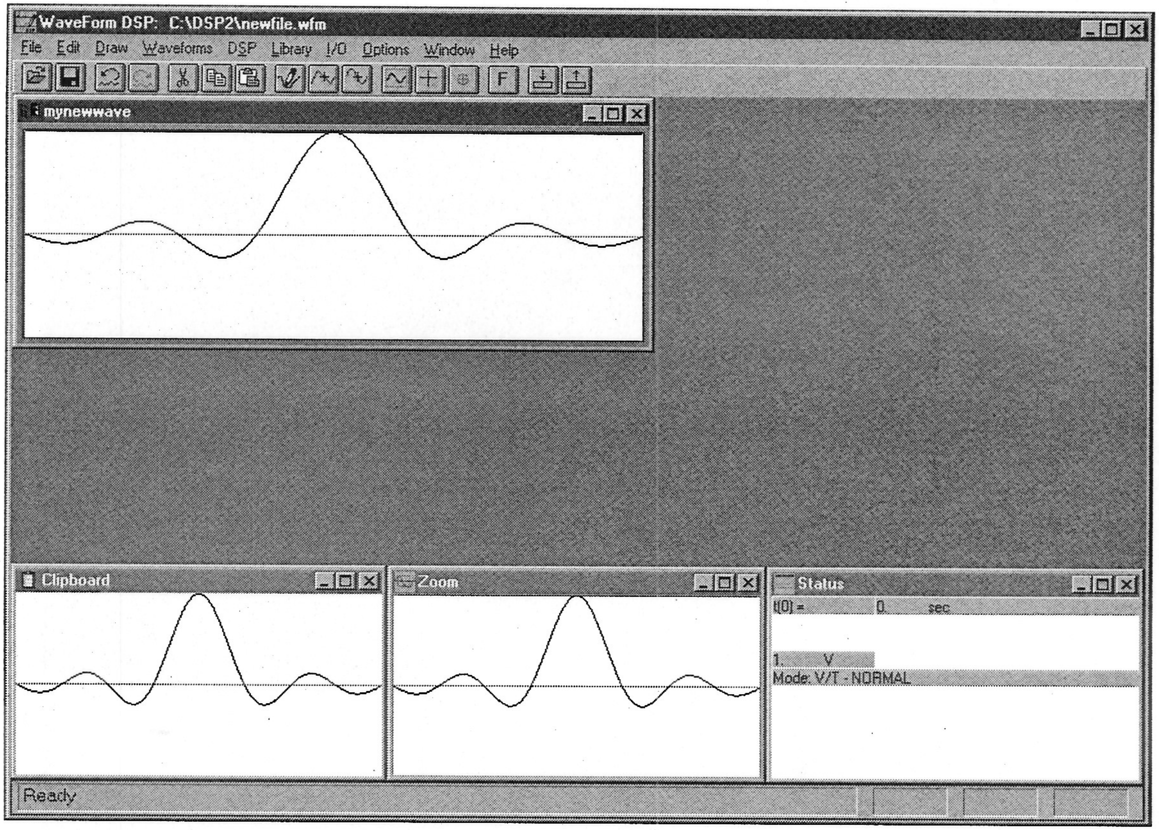

10.Select the Copy button from the toolbar or select from the **Edit** menu, the **Copy** option, or use the accelerator **Ctrl+C.**

*Notice that a copy of the waveform appears in the clipboard window. You can see it by clicking on the exposed portion of the clipboard (in lower left corner of display).*

11. Select the Open button from the toolbar or select from the **File** menu the **Open...** option.

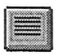
12. From the dialog box, select and open "newfile.wfm".

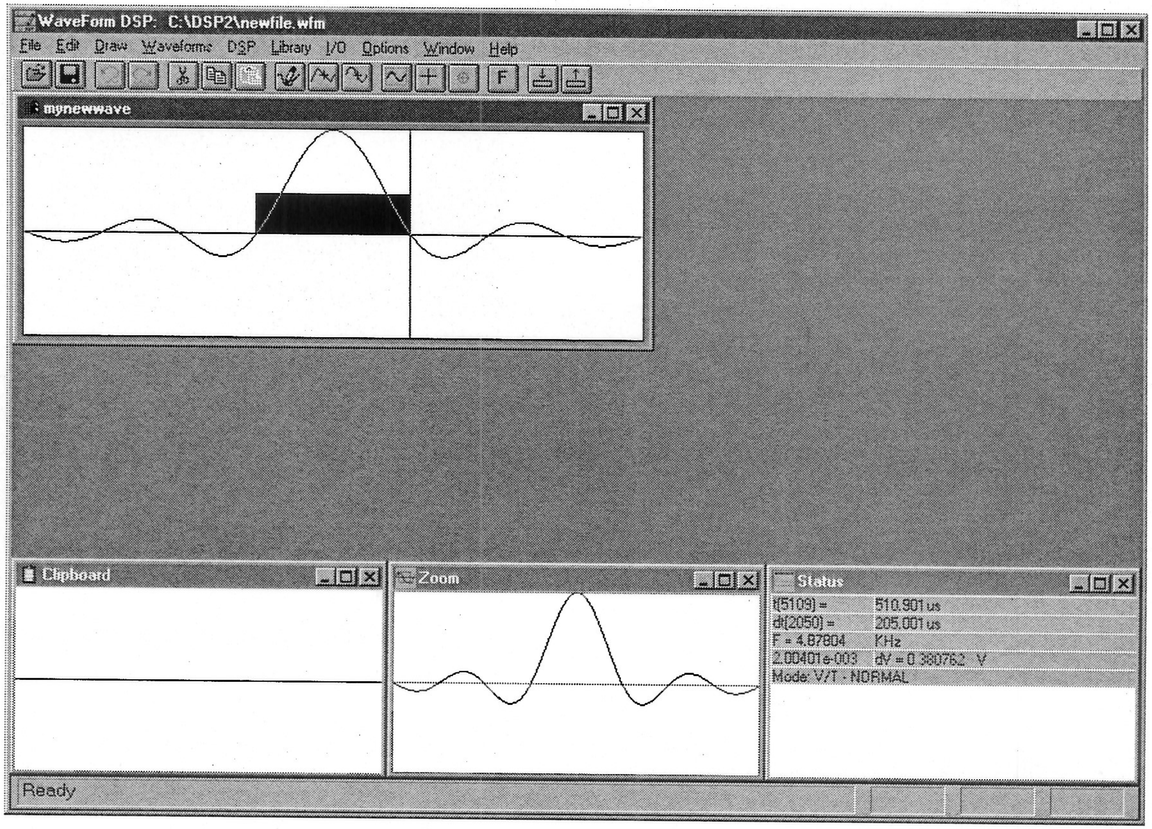
13. Select the Paste button from the toolbar or select the from **Edit** menu the **Paste** option, or use the accelerator **Ctrl+V.**




You now have a copy of the waveform under a new name. In this document we shall call the new waveform "mynewwav". You are now ready to edit. There are several methods of editing a waveform. Each method is described separately in the procedures below. Skip to the desired procedure, or perform them in order as a tutorial.

### Cut out part of a currently active Waveform

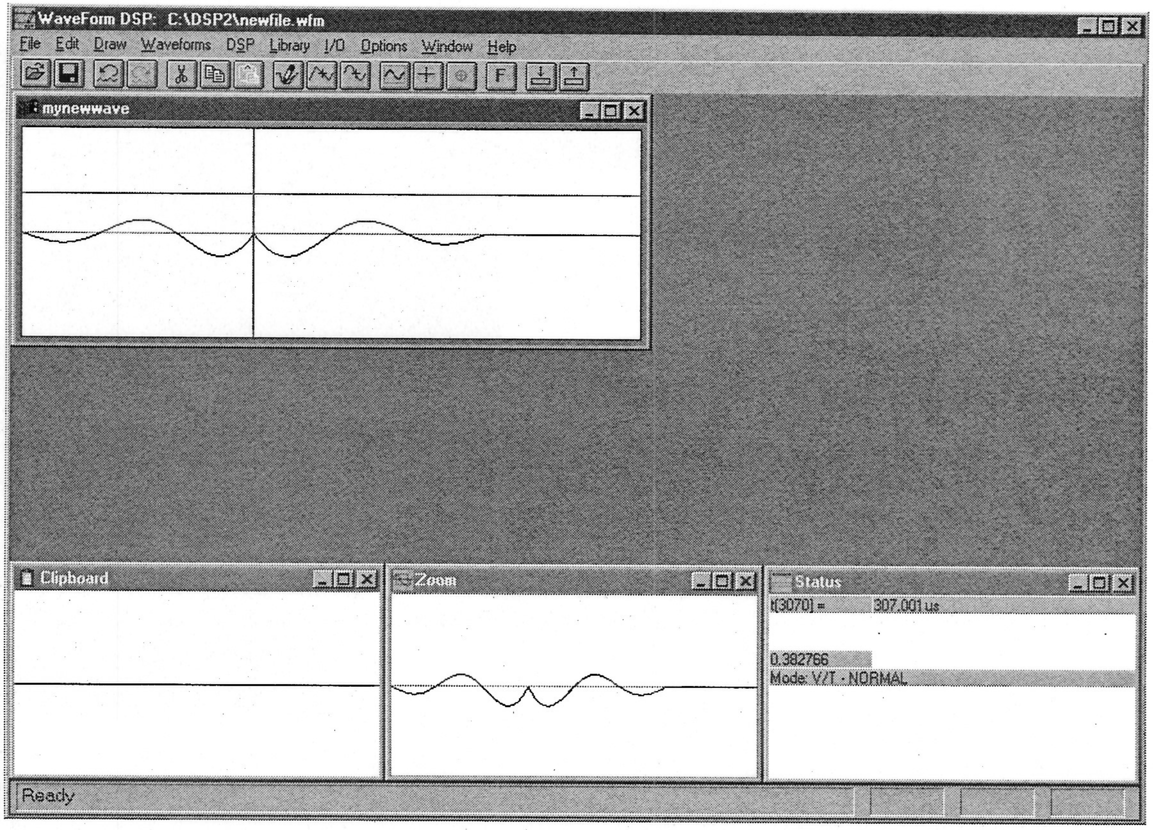
1. Select the View button from the toolbar  or select from the **Waveform** menu the **View...** option.
2. Select the file containing the waveform to be edited.
3. Select the waveform to be edited. If you are running a *tutorial*, select "mynewwav" which you created in "Copying an existing Waveform as a starting point" above.
4. Highlight the portion of the waveform to be eliminated.




*In the illustration, the horizontal cursor line was used to start the cut at the precise amplitude of the two side peaks of the waveform. You can check the status window to read the voltage level.*

5. Select the Cut button from the toolbar or select from the **Edit** menu the **Delete** option, or use the accelerator **Ctrl+X**.

*You have deleted the entire "time" segment, including the peak of the waveform, even though you did not highlight the peak. Note also that the waveform closes up and reconnects the ends created by the cut.*

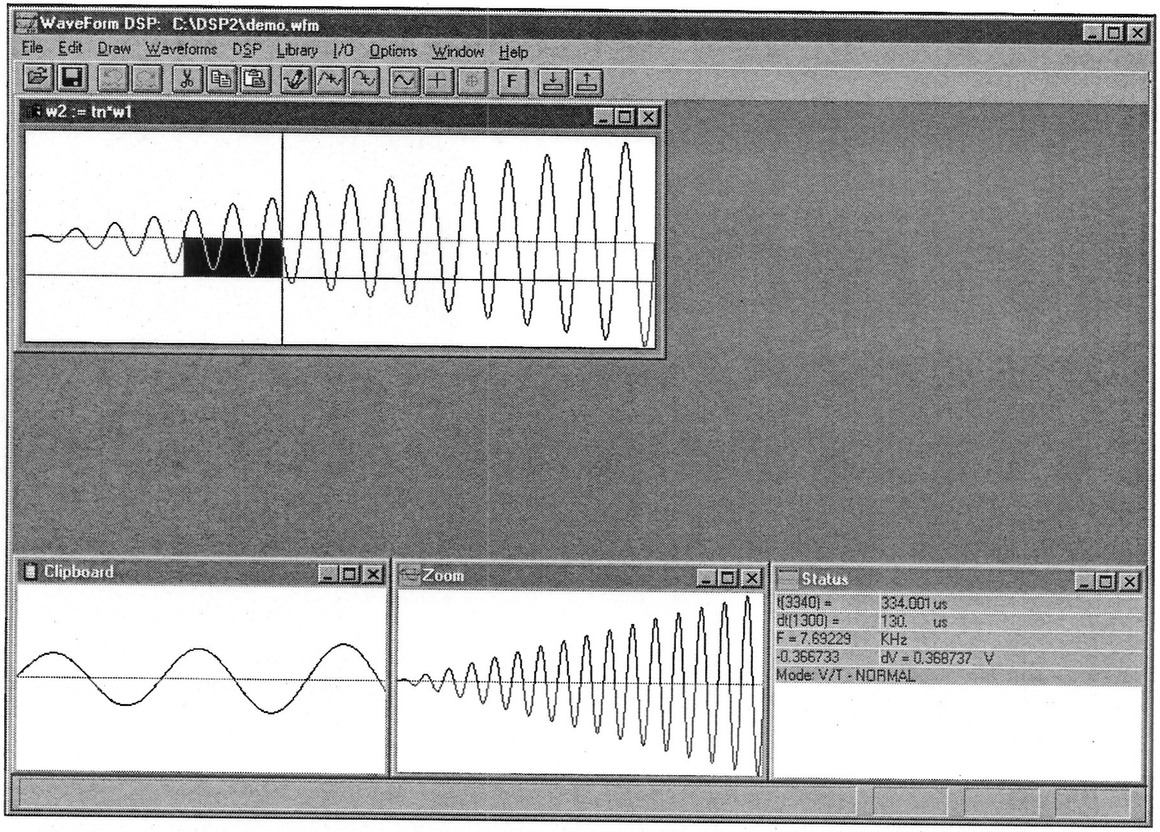



6. Save the new changes

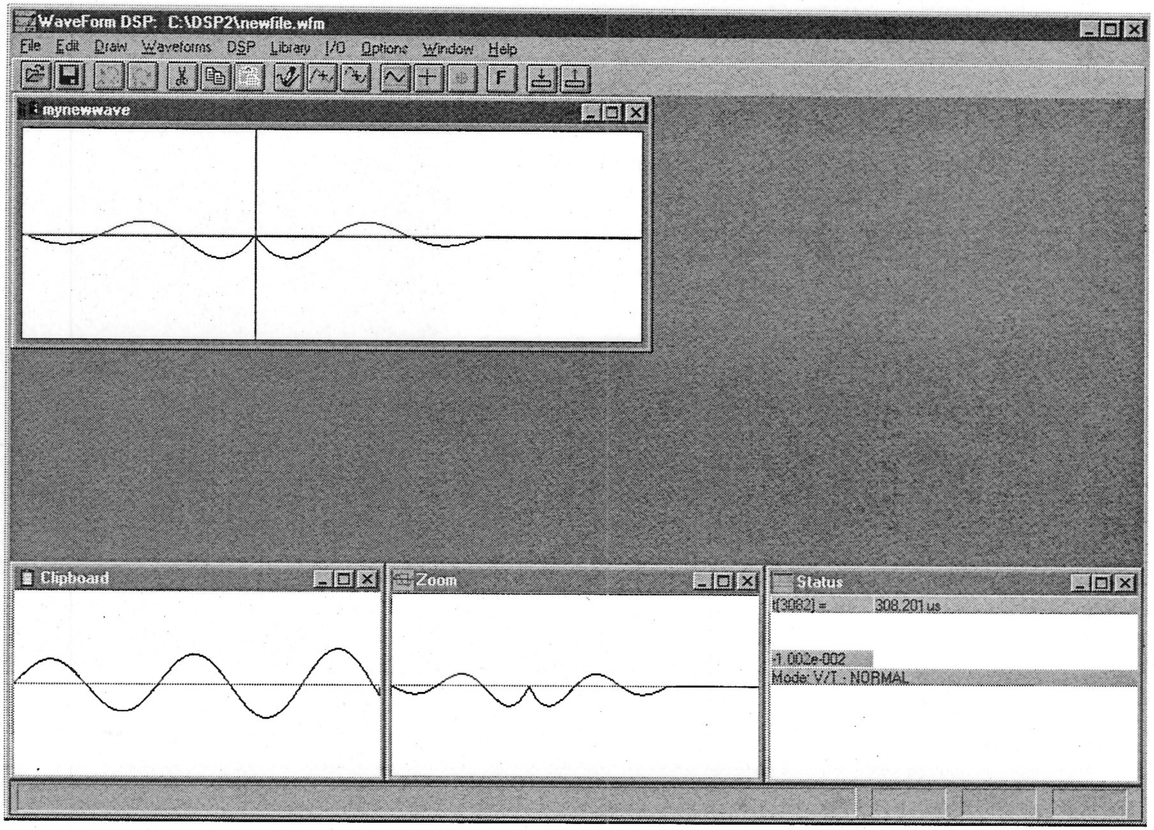
### Splice (Part of) a Waveform into another Waveform

1. Select the Open button from the toolbar  or select the **File** menu, **Open...** option.

   A dialog box appears asking you if you want to save the changes to the file you are leaving. Select "Yes" if you do, or if you are running a tutorial.
2. Open the file containing the waveform that you are going to copy from. If you are running a tutorial select "demo.wfm".
3. Select the waveform that you are going to copy from. If you are running a tutorial select the waveform shown in the following diagram.
4. Highlight the waveform, or the desired portion of it. To match the tutorial, highlight the portion shown in the following diagram.




5. Select the Copy button from the toolbar  or select from the **Edit** menu the **Copy** option, or use the accelerator **Ctrl+C**.
   *The copy you just made is now in the clipboard. You can see the clipboard by clicking on the exposed portion in the lower left corner of the display.*
6. Select the Open button from the toolbar  or select from the **File** menu the **Open...** option.
7. Select and open "newfile.wfm".
8. Do either (a) or (b).below.
   a) Place the cursor at the point in the new ("mynewwav") waveform where you want the waveform in the Clipboard to be added.
   b) Highlight the portion of the "mynewwav" waveform that you want the Clipboard copy to replace.
9. If you are running a tutorial, select the insertion point shown in the diagram.

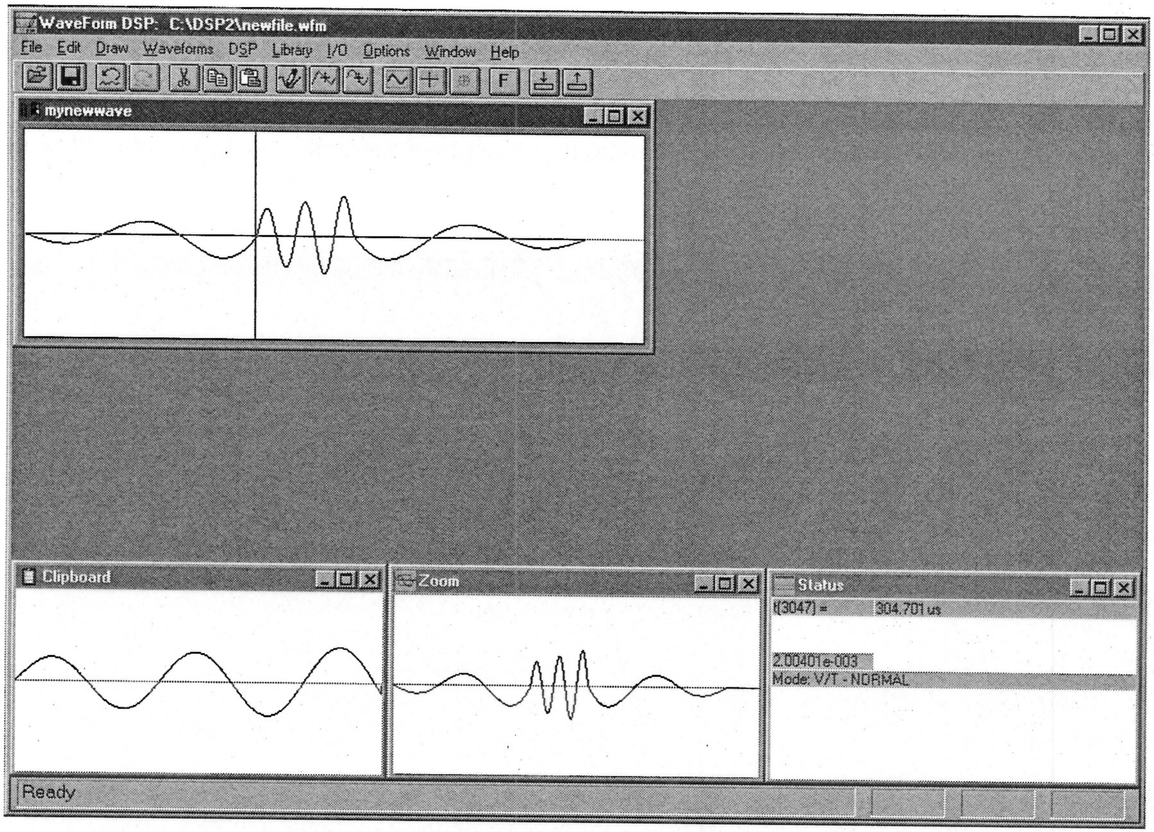



10. Select the Paste button from the toolbar or select from the **Edit** menu the **Paste** option, or use the accelerator **Ctrl+V**.

*You now have a copy of the Clipboard waveform spliced into the starting waveform. The highlighted portion of the waveform was replaced. Everything else moved over.*

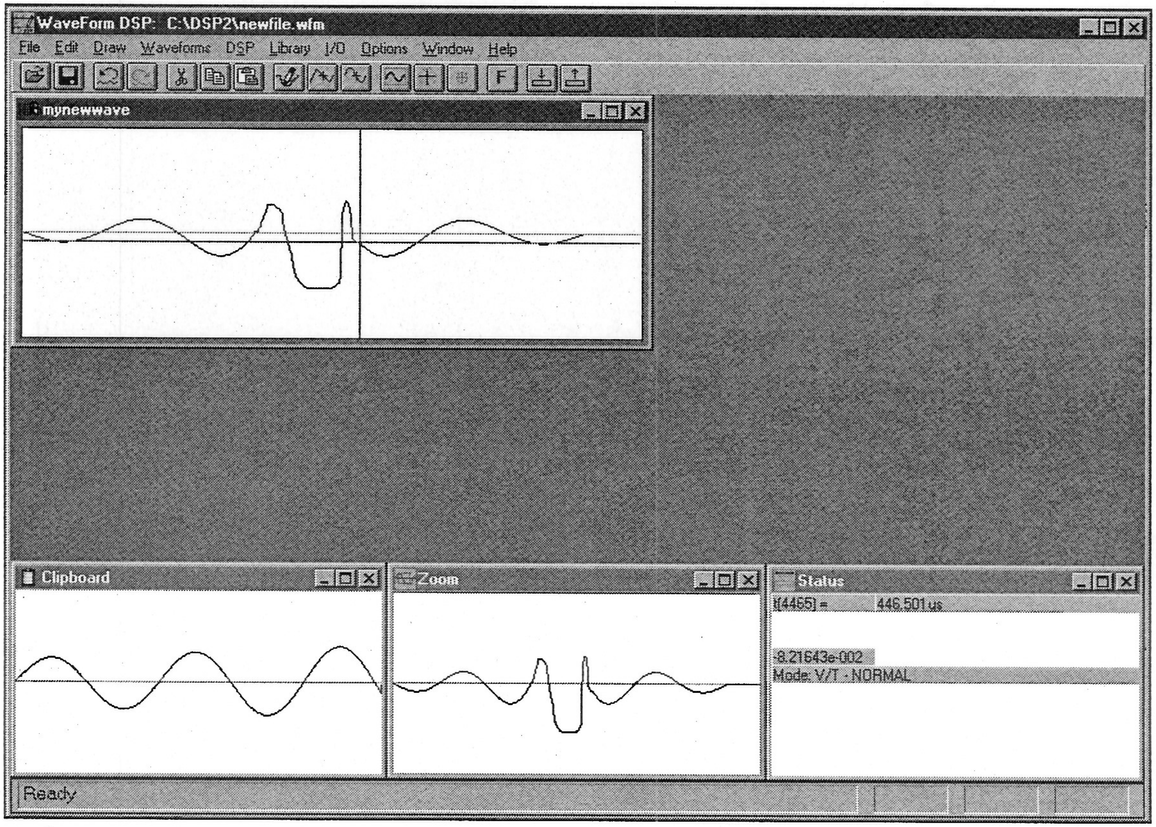
*If you had not selected either a point on, or a portion of, the "mynewwav" waveform, the Clipboard copy would have been added at the beginning of your new waveform.*

*If any editing causes the resulting waveform to grow beyond the maximum number of points, the resulting waveform is truncated at the right-hand limit.*


WaveForm DSP: C:\DSP2\newfile.wfm
File Edit Draw Waveforms DSP Library I/O Options Window Help
mynewwave
Clipboard
Zoom
Status
t(3047) =
304.701 us
2.00401e-003
Mode: V/T - NORMAL
Ready

## Graphically editing a Waveform

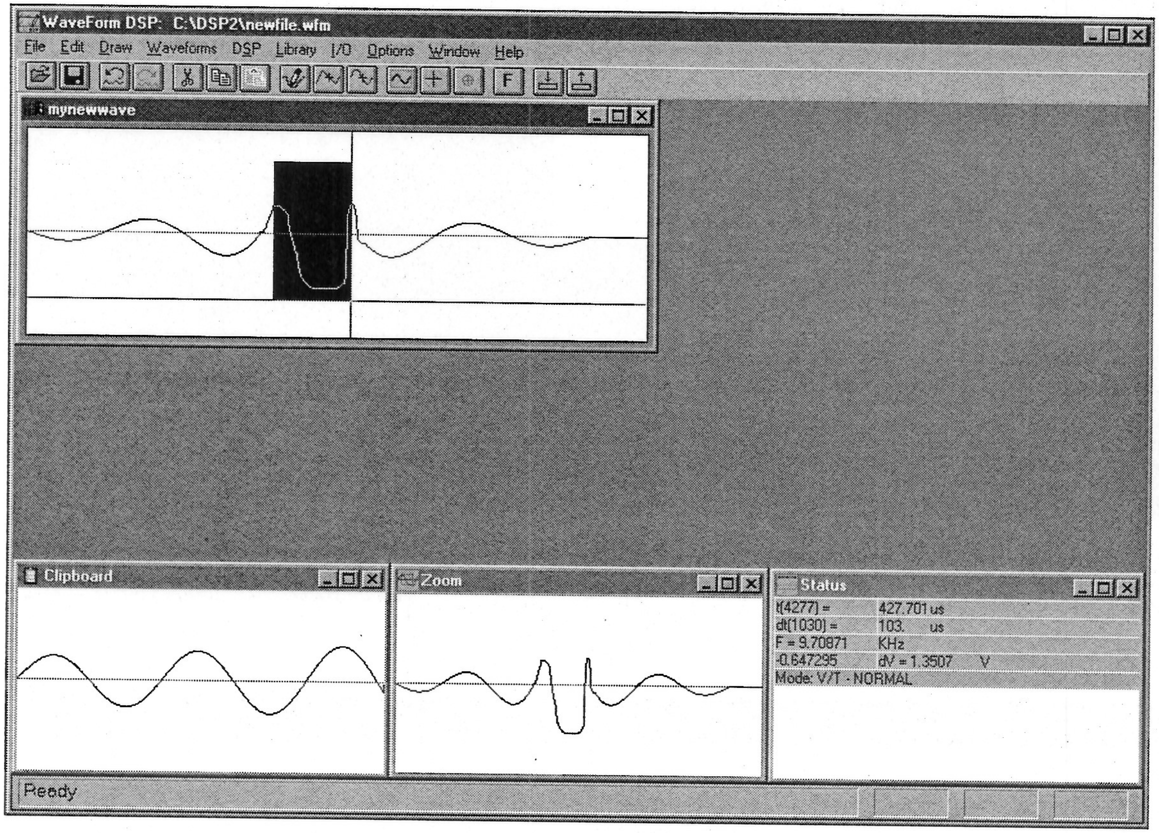
1. Activate the file containing the waveform that you are going to edit. Activation is described in "Changing an Existing Waveform", above. For the tutorial, stay with "mynewwav".
2. Select the Draw menu, Freehand option.
3. Note the beginning and end points of that portion of the waveform that you are going to change. For the tutorial, choose the section between the first and last peaks of the section you just spliced in.
4. Move the cursor to the first end point of the portion.
5. Drag it to the opposite end point through a path describing the new shape of that portion, as shown in the following diagram.

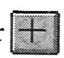


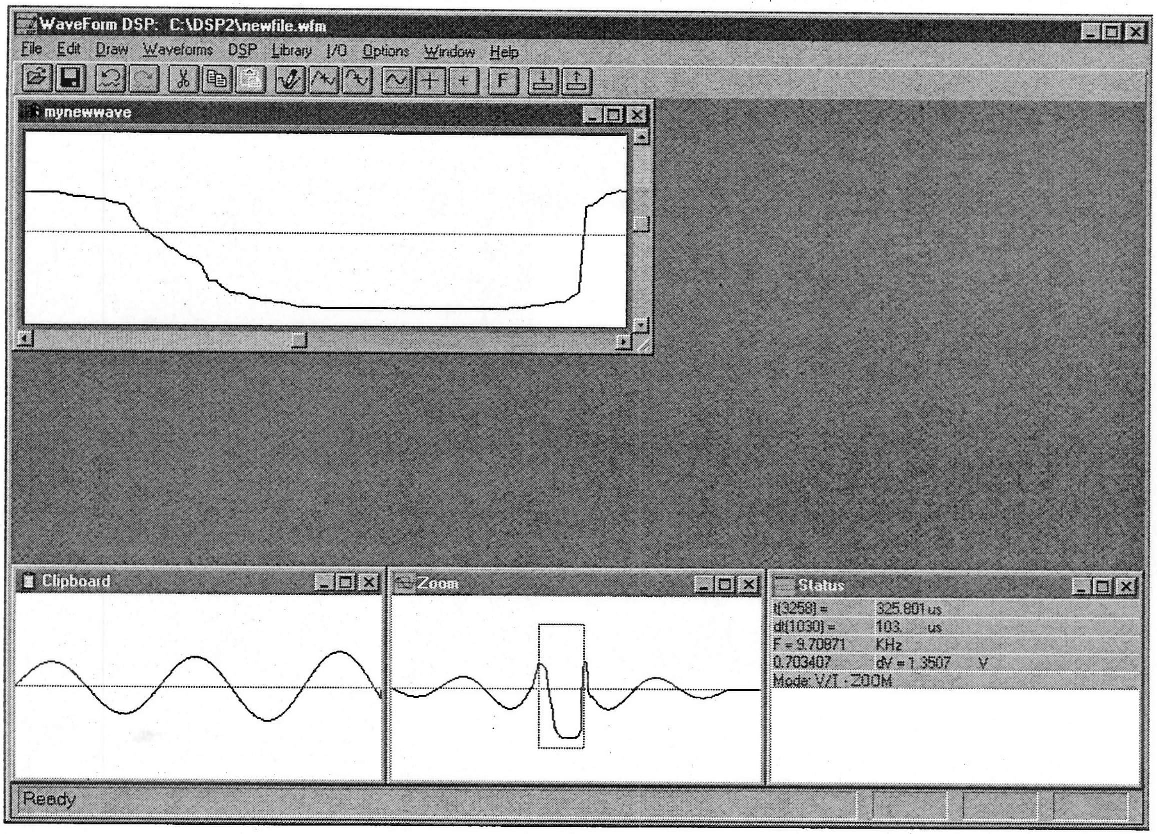

6. Strike the ESCAPE key or the right mouse button to leave the Draw mode.
   *Cursor returns to the arrowhead shape.*

Zoom in for finer detail

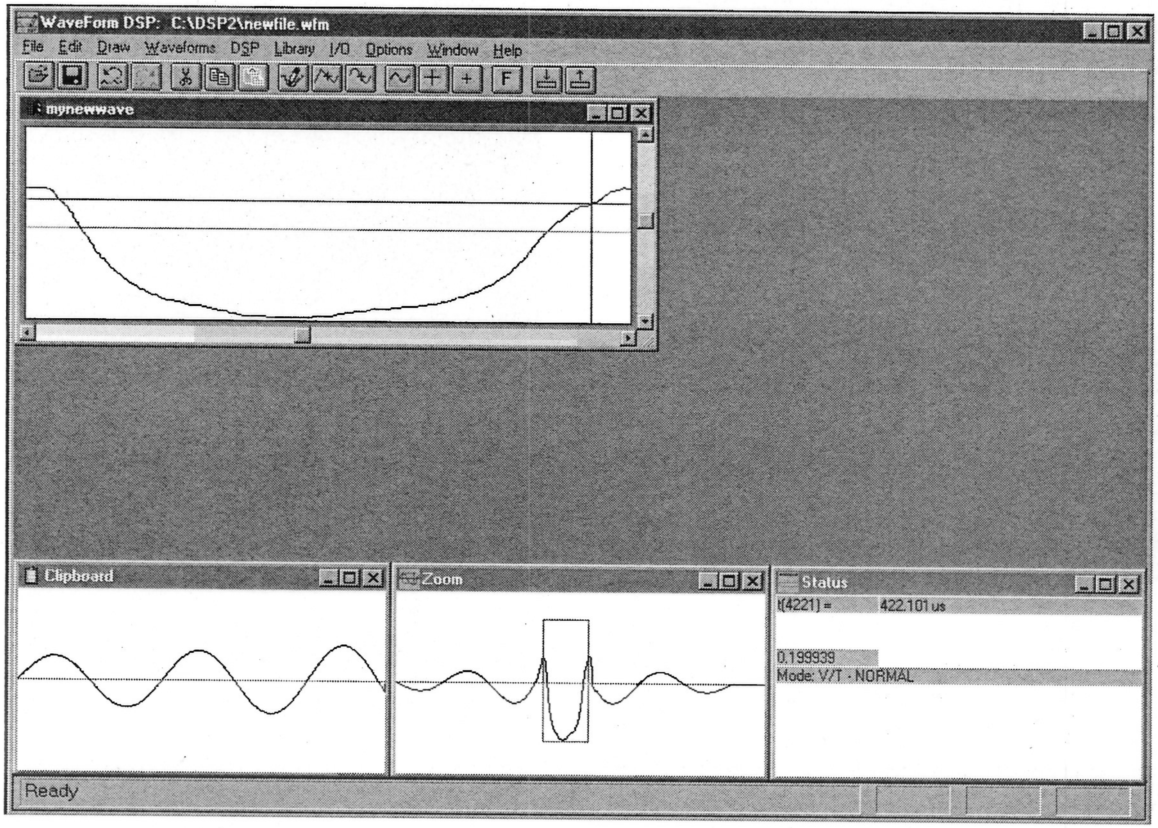
1. Highlight the waveform as illustrated



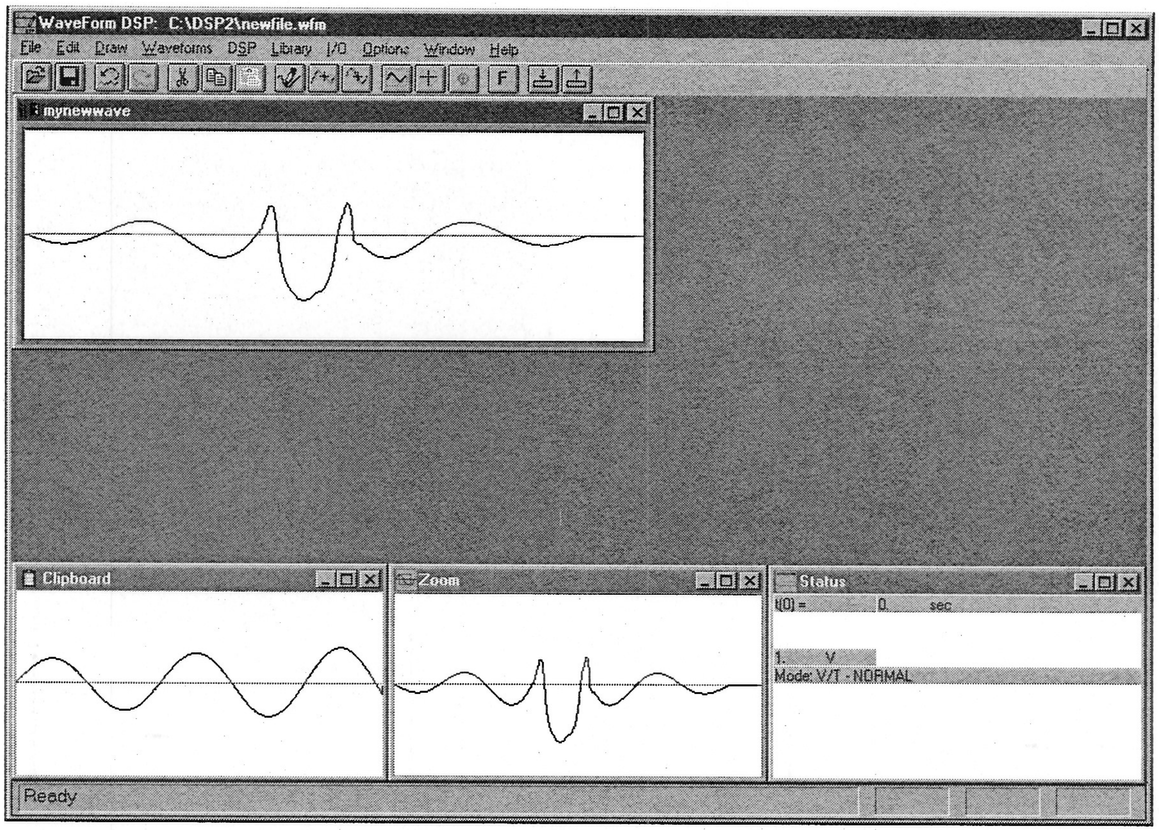

2.Select the Zoom In button from the toolbar  or select from the **Waveform** menu the **Zoom In** option, or use the accelerator **Ctrl+I**.




3. Select the Freehand button from the toolbar  or select from the **Draw** menu the **Freehand** option.
4. Again, try very carefully to draw the new shape.

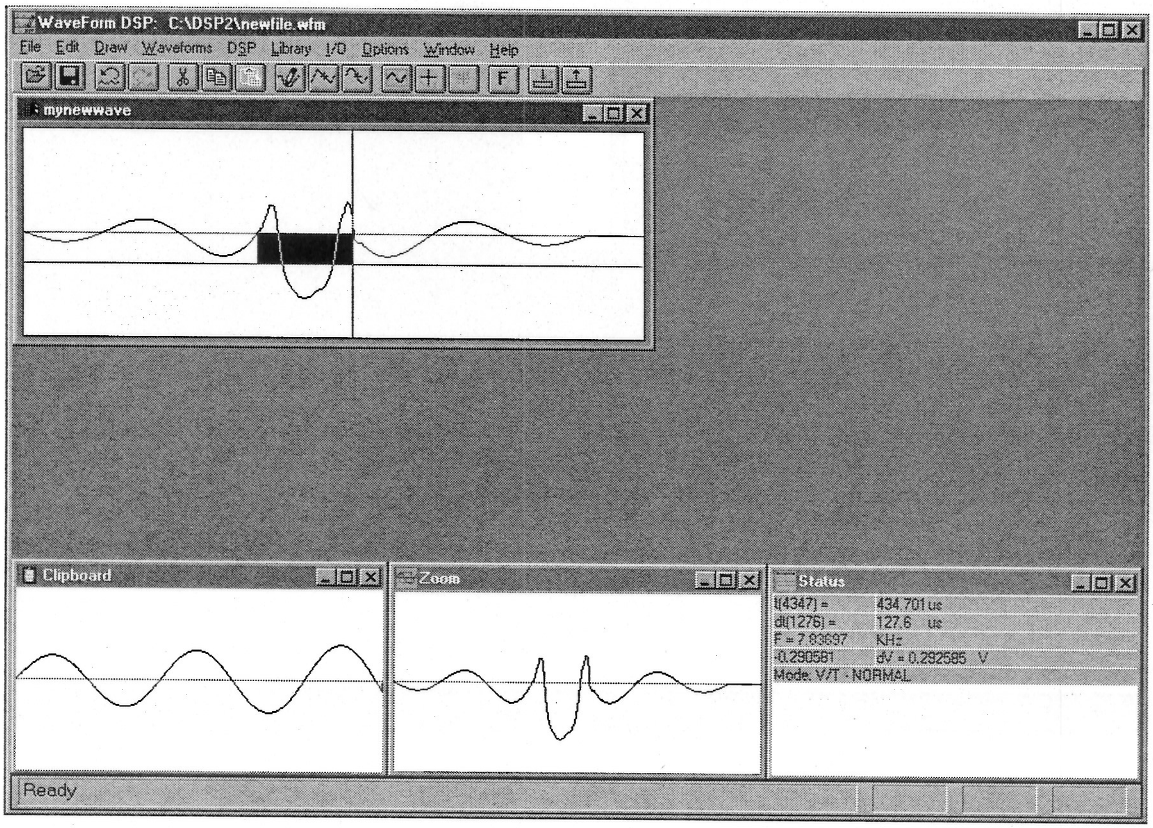



5. Strike the ESCAPE key, or click the right mouse button, to leave the Draw mode.
6. Select the Zoom Out button from the toolbar [+] or select from the **Waveform** menu the **Zoom Out** option, or use the accelerator **Ctrl+O**.
*Notice the improvement. You could also try the Curve Fitting option, to draw the curve a point at a time, on this portion of your waveform.*

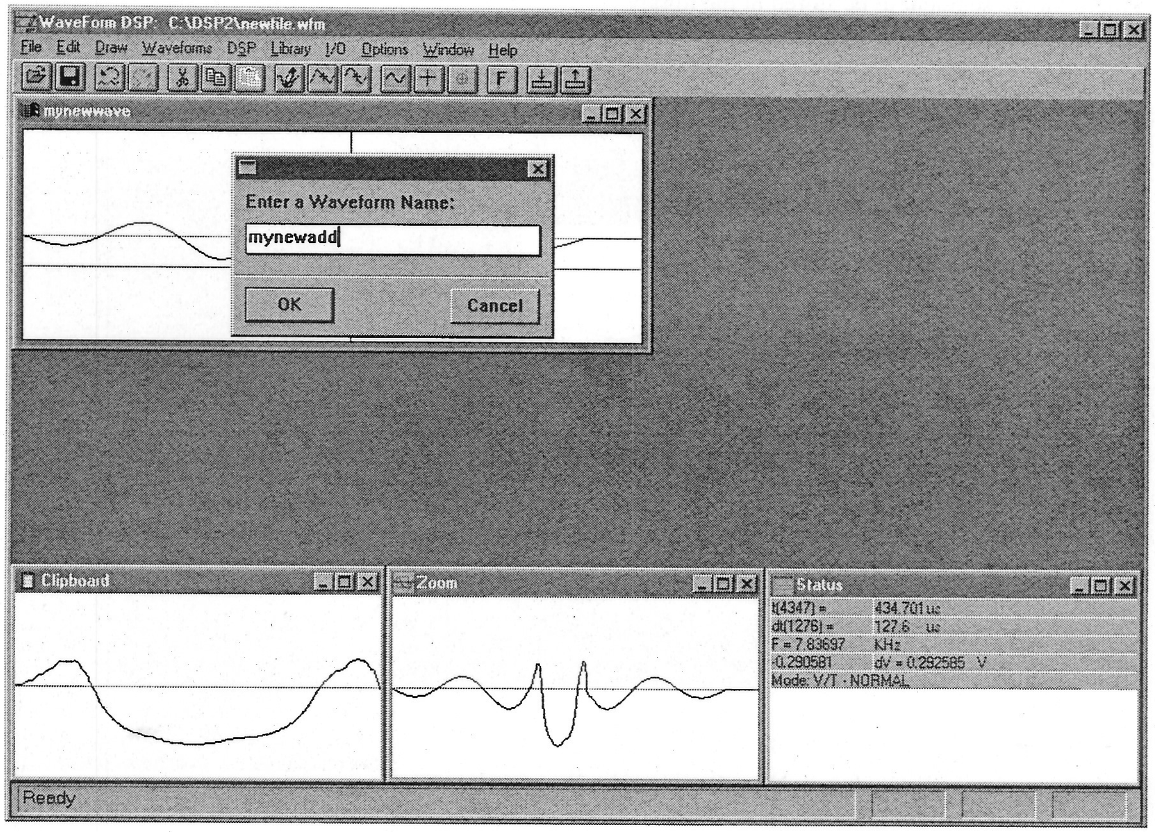

WaveForm DSP: C:\DSP2\newfile.wfm
File Edit Draw Waveforms DSP Library I/O Options Window Help
mynewwave
Clipboard
Zoom
Status
t(0) = 0. sec
1. V
Mode: V/T - NORMAL
Ready

## 4.3 Copying a Waveform to the Library

1. Highlight the segment to be stored in the library.

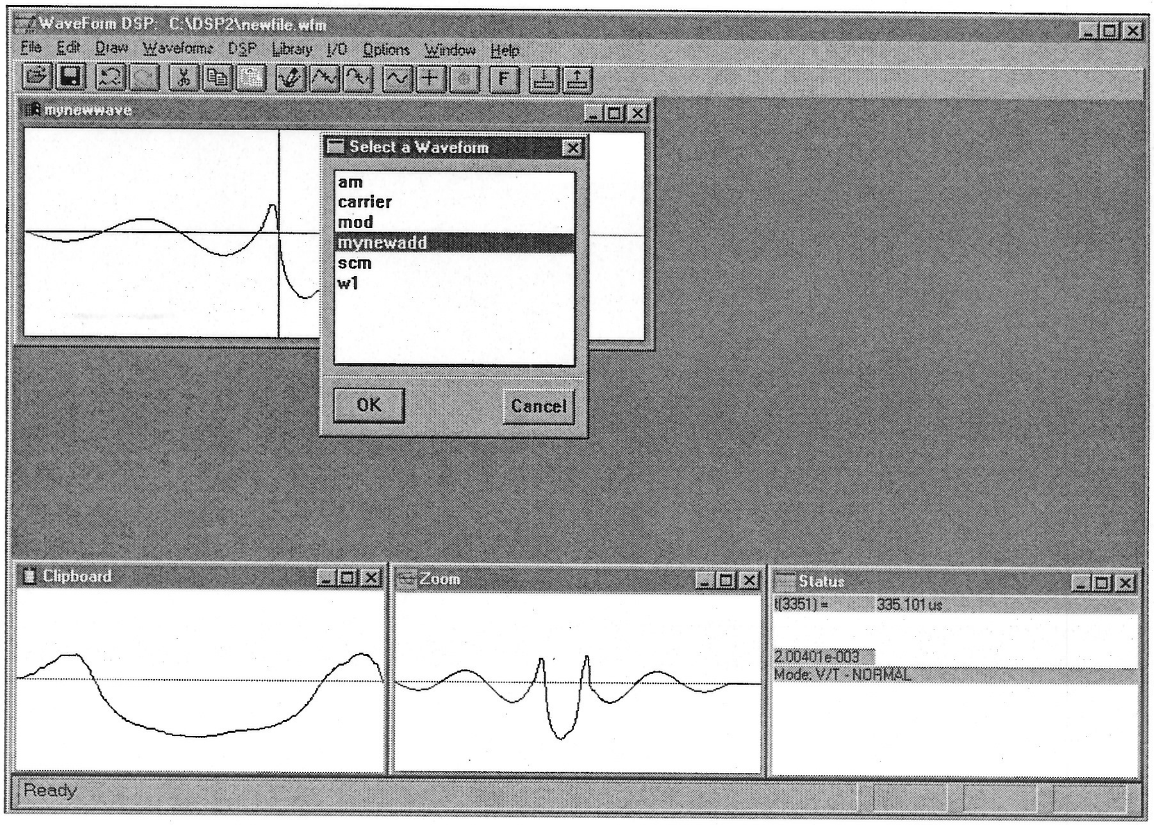



2. Select the Copy button from the toolbar  or select from the **Edit** menu the **Copy** option, or use the accelerator **Ctrl+C**.
   *The selection appears in the clipboard window*
3. Select from the **Library** menu the **Add...** option.
   *A dialog box appears*
4. Type in the name of the new library addition, and strike ENTER. For the tutorial, type in "mynewadd".

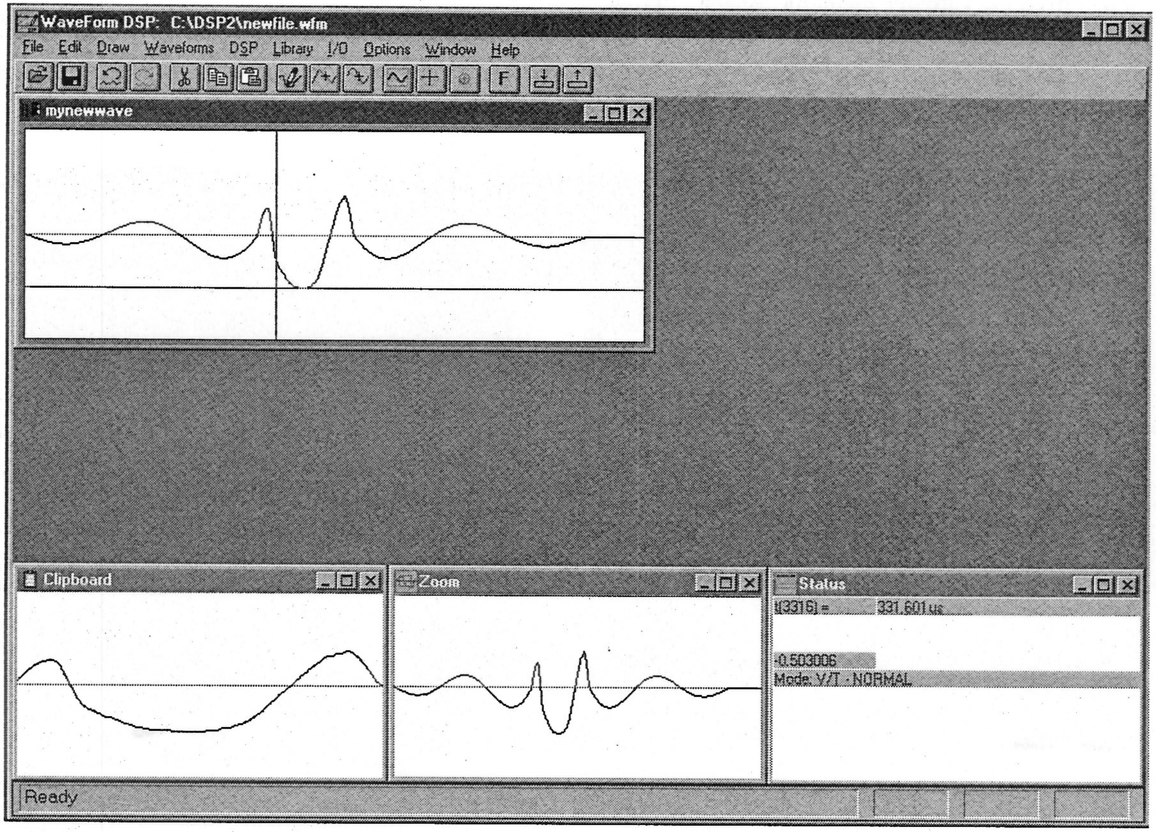

WaveForm DSP: C:\DSP2\newfile.wfm
File Edit Draw Waveforms DSP Library I/O Options Window Help
mynewwave
Enter a Waveform Name:
mynewadd
OK
Cancel
Clipboard
Zoom
Status
t(4347) = 434.701 us
dt(1276) = 127.6 us
F = 7.83697 KHz
-0.290581 dV = 0.292585 V
Mode: V/T - NORMAL
Ready

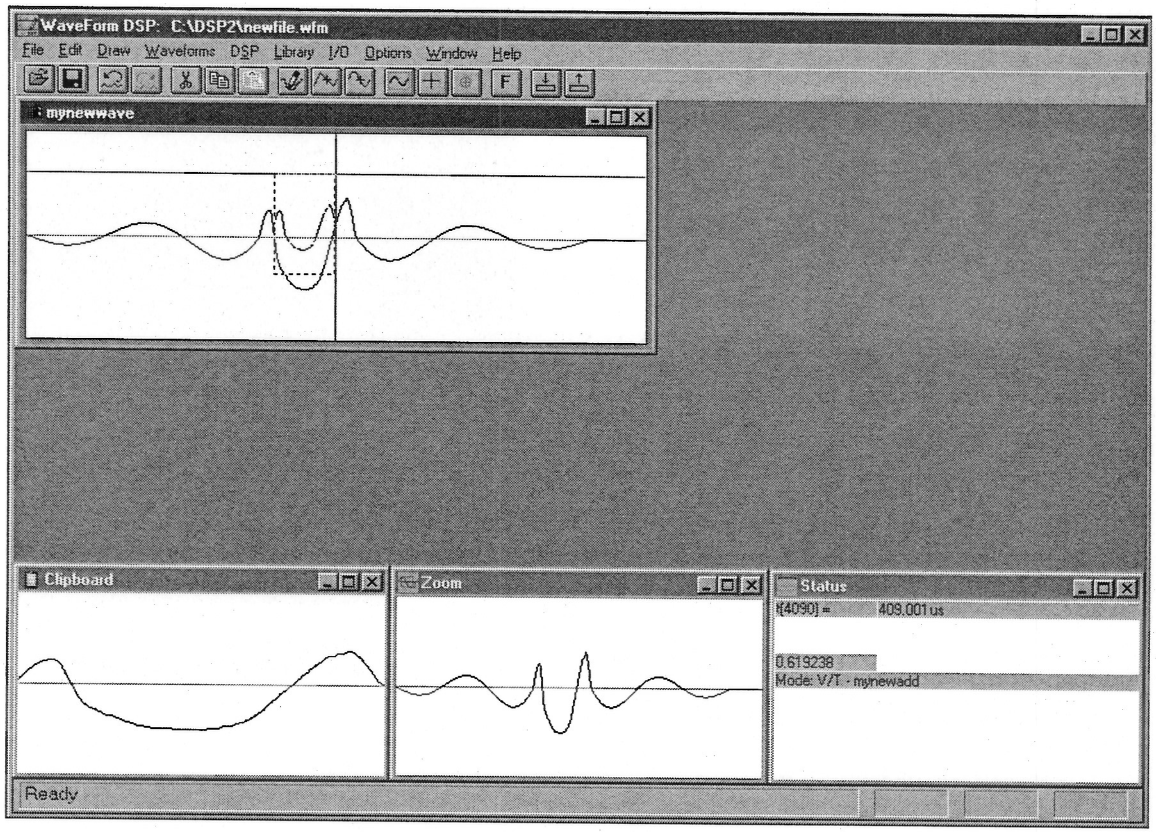
## 4.4 Copying a Waveform from the Library

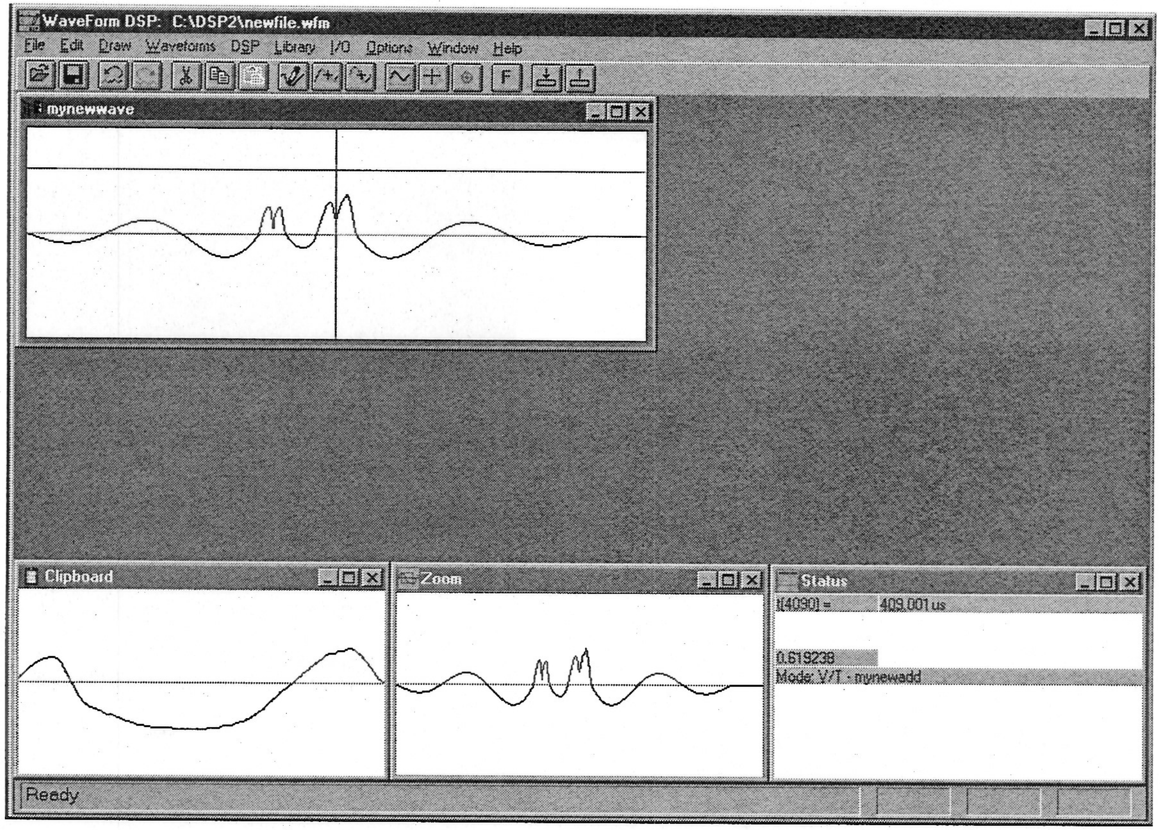
1. Select from the **Library** menu the **Select...** option.
   *A dialog box appears. Note that "mynewadd" appears in the list of waveforms.*




2. Select and open "mynewadd".
3. Drag the mouse from the lower left to the upper right corners of an imaginary box surrounding the area to be replaced. See the next three illustrations:


WaveForm DSP: C:\DSP2\newfile.wfm
File Edit Draw Waveforms DSP Library I/O Options Window Help
F
mynewwave
Clipboard
Zoom
Status
t[3316] = 331.601 us
-0.503006
Mode: V/T - NORMAL
Ready


WaveForm DSP: C:\DSP2\newfile.wfm
File Edit Draw Waveforms DSP Library I/O Options Window Help
mynewwave
Clipboard
Zoom
Status
t[4090] = 409.001us
0.619238
Mode: V/T - mynewadd
Ready




4.If the results of this operation are unsatisfactory, select the Undo button from the toolbar  or from the **Edit** menu the **Undo** option, and repeat steps 1 through 3.

5.Strike the ESCAPE key to quit library mode.

## 4.5 Frequency Domain

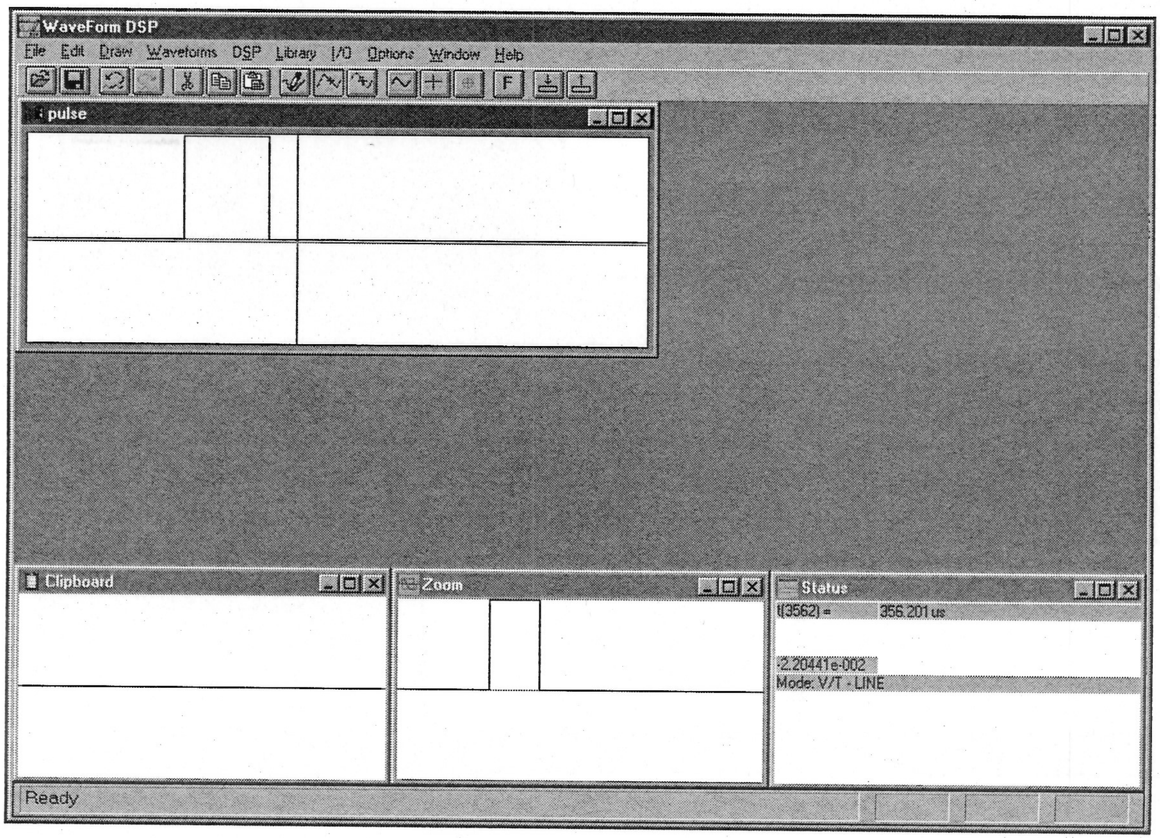
### Spectrum of a Time Function

We will demonstrate creating the frequency spectrum of a waveform using the simple, classic example of a pulse.

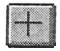
1. Create a new waveform window.
2. Select the Draw menu, Line option.
3. Draw a short horizontal line near the very top of the waveform, similar to the top of the waveform in the illustration below.

   *You can get help with the Draw mode in an earlier part of this section.*

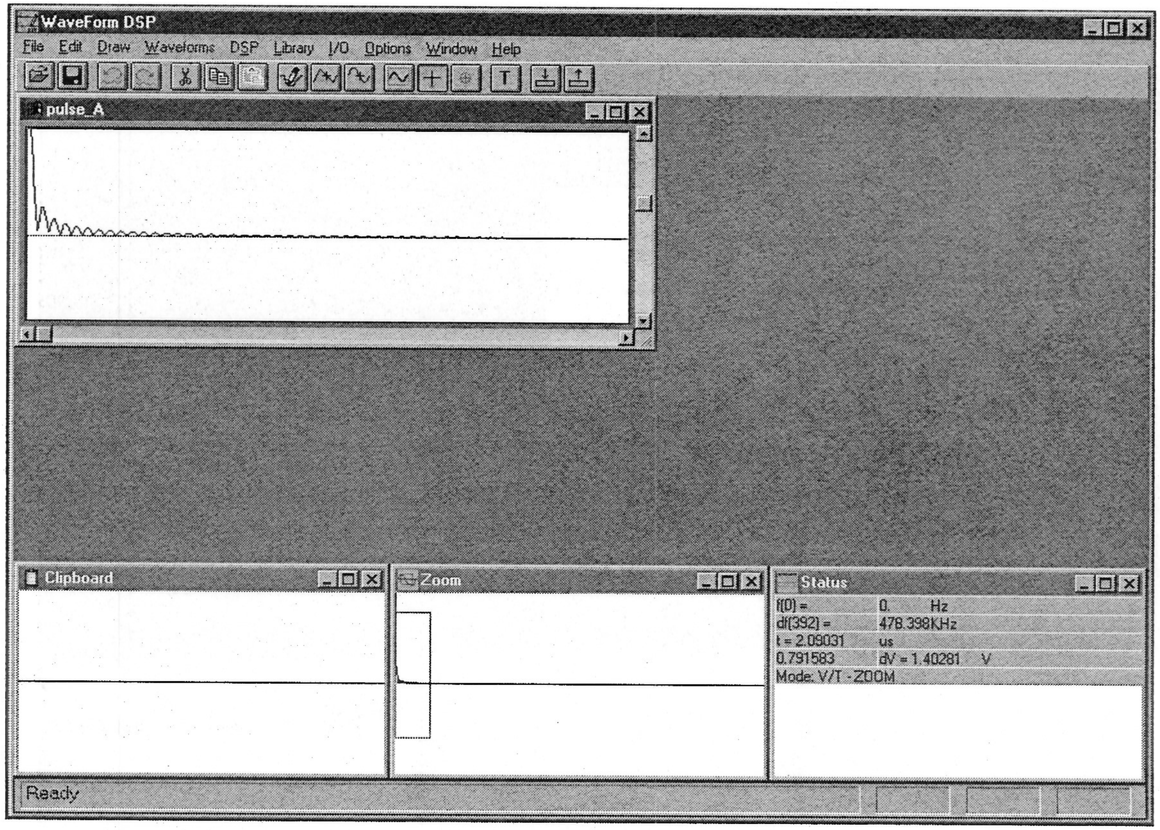
   *You should end up with a waveform similar to the illustration below.*



4.Select the DSP menu,
Frequency: Amplitude option.

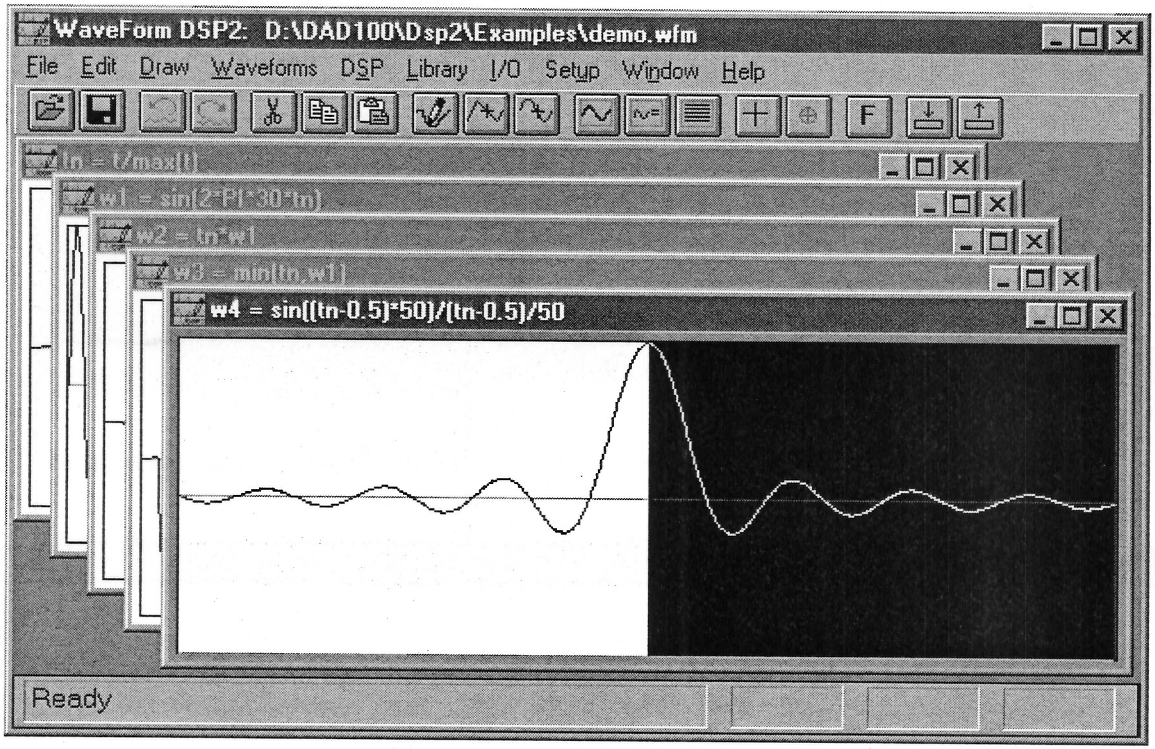
5.When the frequency-domain waveform appears, select the Zoom In button from the toolbar  or select from the **Waveform** menu the **Zoom In** option, or use the accelerator **Ctrl+I.**

6.Highlight the portion of the window you want a closer look at.
*Note that the frequency axis is a linear scale from waveform frequency to 0.5 * (sampling frequency).*
*When you release the mouse button, the view in the waveform window will expand. You may have to Zoom-in more than once to get a view similar to the one shown below.*



## Frequency Domain Editing

1. Open the "demo.wfm" file.
2. Activate the window [w4] containing the sin(x)/x function.
3. Highlight the right half of the sin(x)/x function.

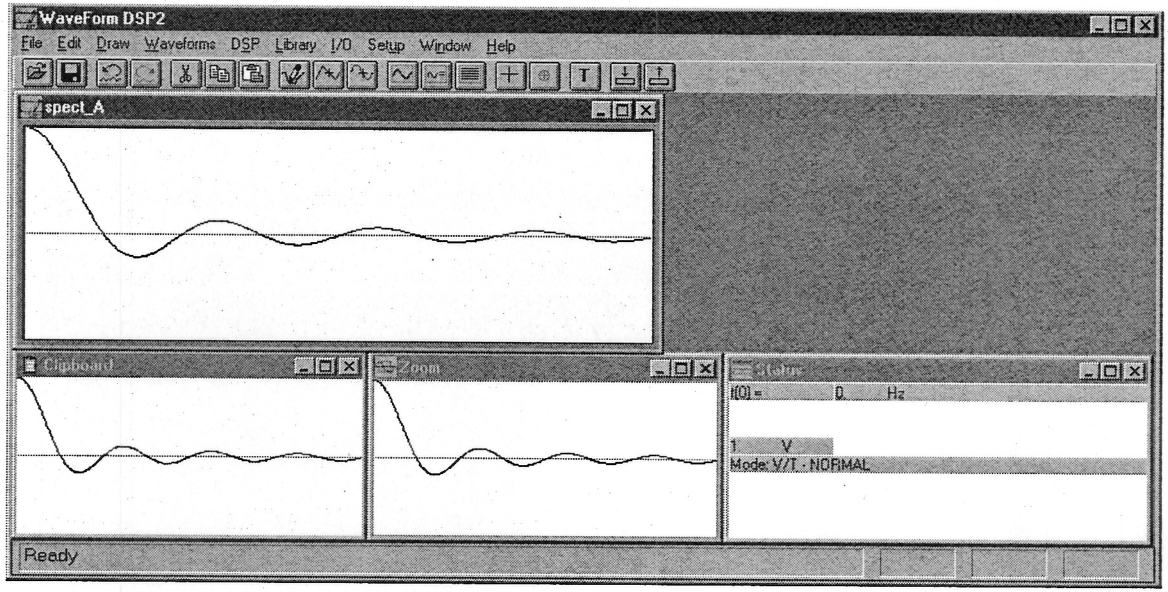



4. Select the Copy button from the toolbar or select from the **Edit** menu the **Copy** option, or use the accelerator **Ctrl+C**.
5. Close the "demo.wfm" and using the **New...** option from the **File** menu.
6. Open a new waveform window using the Create button from the toolbar or select from the **Waveform** menu the **Create** option, or use the accelerator **Ctrl+N**.
7. Leave the time domain window blank, and using the F button from the toolbar, or select from the **DSP** menu the **Frequency &Amplitude** option.

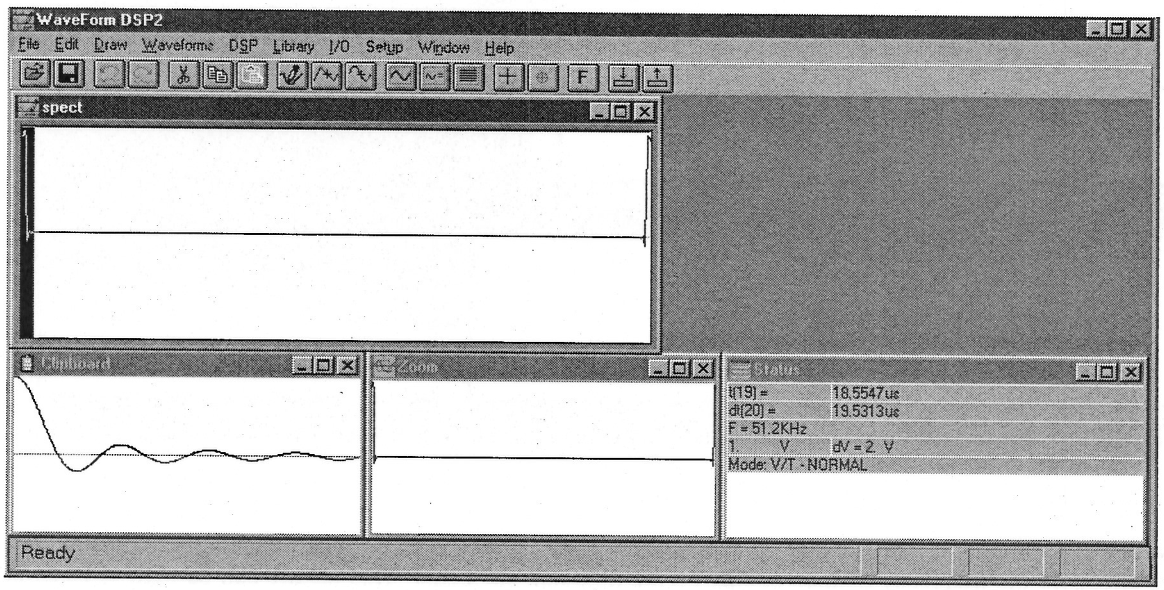
8. Place the cursor at the extreme left of the waveform window and on the horizontal axis.

9. Select the Paste button from the toolbar  or select from the **Edit** menu the **Paste** option, or use the accelerator **Ctrl+V**.

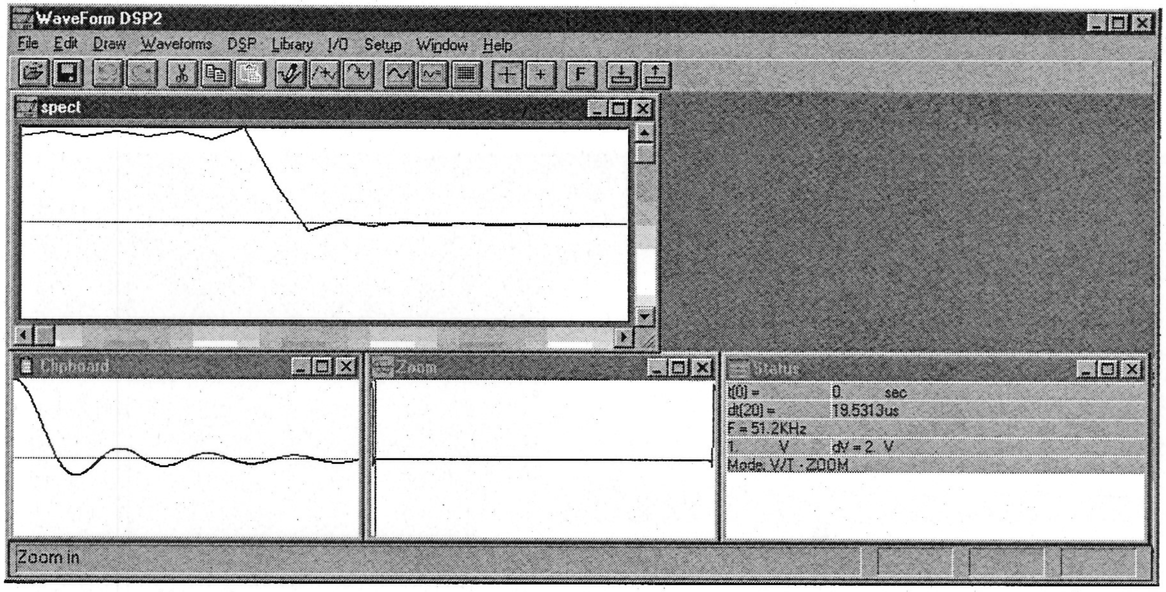
   *The waveform is now pasted into the frequency domain as shown.*



10. Now select the button from the toolbar, or select from the **DSP** menu the **Time** option.
    *The waveform in the time domain is probably very narrow.*
11. Zoom the pulse in, as shown.




*As expected, a pulse waveform, with some ringing, appears in the time domain window.*




You can use any of the editing tools to modify the frequency-domain waveform. When you switch back to the time domain, you then see the effects of the frequency-domain editing on the time-domain signal. This feature can be used to clean up a noisy waveform or to smooth a rough waveform by eliminating high frequency noise or harmonic content in the spectrum of the waveform before using the time-domain signal.

# 5 Working With Equations

You can add to your collection of waveforms by giving a WaveForm the mathematical expression for the curve required. For this, the expression mode may be used, which is accessed through the waveform menu.

The Expression mode feature of WaveForm uses an interactive equation-parsing program called MathViews™. It creates complex waveforms directly from equations entered from the keyboard. They can be algebraic/trigonometric or integral/differential equations. You can create more complex waveforms by adding, subtracting, multiplying and concatenating simpler waveforms together.

The mathematical expressions used must adhere to certain syntactical rules, which are determined by the MathViews™equation-parsing program. These rules are given in detail in this section.

MathViews™ uses a comprehensive set of operators and built-in functions to perform real array operations and data manipulation. The one-dimensional arrays that are created translate directly into digitized waveforms.

MathViews™ acts as an interpreter. Each command is actually compiled as it is entered rather than interpreted at run time. There is an immediate syntax check of the input.

All calculations are performed in double precision. Function keys are not used and workspace is limited only by the available memory.

## 5.1 Time Domain Windows

Vertically, the window is two dimensionless units high from -1 to +1. These values are multiplied by the amplitude (in volts) or the Y-max value specified by the **Setup** menu, **Waveforms...** option.

The amplitude is a double-precision value. When this value is downloaded to an Arb, it is truncated to the appropriate value for that model.

Size (in points) and Frequency may also be specified in the Setup Waveforms dialog. Changing amplitude or frequency affects the status window read-out of points selected in the waveform window, and they determine the frequency and amplitude settings downloaded to the Arb, but they do not effect the waveform data table.

## 5.2 Frequency Domain Windows

The default scale of the waveform window depends on the parameter selected for the original time domain expression. For example when you create a time domain function, you specify a **Block Frequency** and a **Size** or number-of-points of resolution. The incremental time between data points, $dt(1)$, which can be read from the status option window, is then:

$$dt(1) = 1 / (nf)$$

where $f$ is the frequency and $n$ is the number of points of resolution.

When you view the function in the amplitude vs. frequency window, the incremental frequency step, df(1), which can be read from the Status window, is:

$df(1) = f$     *(the frequency increment is the **Block Frequency**)*

and the full-scale frequency is:

$nf / 2$     *(the Nyquist point of **Sample Frequency**)*

The division by two reflects the fact that the number of points resolution the transform is one half the resolution of the time-domain function The reduced resolution is, in turn, a consequence of the fact that the transform consists of an equal number of real and imaginary parts. Only the real part or only the imaginary part are displayed.

Phase, in the phase/frequency window, is normalized to -180°, 0°, +180°.

*A waveform in a Frequency Domain Window can be edited using any selection from the Draw and Edit menus.*

*You cannot enter or modify mathematical expressions in a frequency-domain window, however, cut, copy and paste operations do not distinguish between the Time or Frequency domains that they are used in. To incorporate a mathematical function into a Frequency Domain waveform you must define it in the Time Domain and then use cut or copy operations to put the sequence of points onto the clipboard, and then paste the untransformed points into the Frequency Domain waveform.*

## 5.3 Procedures

The rules for entering equations into WaveForm DSP2 are presented next under *Equation Parsing Rules*, which is organized to serve as a reference section. There are also suggestions for real-world applications for some of the functions.

The section *Creating Waveforms* follows, and contains or references a number of sample procedures. You might find it useful to try the procedures described in that section before digging too deeply into the rules.

## 5.4 Equation Parsing Rules

The rules for entering equations into the Expression dialog box are as follows:

- Expressions perform calculations using variables, numbers, binary and unary operators, and built-in functions.
- Expressions are always entered in the time domain. However, the resulting waveform can be pasted to a frequency-domain window using the editor. (See *Working with the Editor* section).
- Trigonometric expressions are in radians.
- Variables store real scalars, real arrays, and several built-in constants. (Waveforms are MathViews™ variables with array arguments).
- Variables, functions and constants are not case sensitive.
- All calculations are performed in double precision.

## 5.5 Data Types

### Scalars

#### Real Numbers

Real numbers are integers and floating point numbers. Floating point numbers consist of an integer part, an optional decimal point, a fraction part, an "e" or "E," and an integer exponent. All floating point numbers are double precision. Some examples of real numbers are:

`3.14159` and `10E-6`

**Complex Numbers**

Complex numbers are formed as follows:

- A number followed by "i" or "j" to represent a purely imaginary number.
- A number followed by "D" to represent a complex number in polar format with unit amplitude at the specified angle in degrees.
- A number followed by "R" to represent a complex number in polar format of unit amplitude at the specified angle in radians.
- Using one of the built-in functions, such as `sqrt(-1)`, to create a complex number.

Some examples of complex numbers are:

```
2.5i or 3.4j
30D
0.55R
2+sqrt(-1)
```

## Variables

Variable names are composed of letters and digits. The first character must be a letter. The underscore, "_", is recognized in MathViews™ as a letter. MathViews™ is case sensitive, by default, so upper and lower case characters are different. In addition, control-flow keywords like **if**, **else**, **elseif**, **end**, **while**, **for**, **end**, and **break** are reserved.

Variable names that start with the underscore are reserved. You should avoid naming variables with an underscore as the first character.

## Built-In Variables

MathViews™ has several built-in variables, `i, j, pi and eps`. The `i` and `j` constants are the purely imaginary number. The `eps` constant is used as the system's floating point tolerance. It has a default value of the smallest double precision number.

## Strings

A string is a constant consisting of a series of ASCII characters enclosed in single quotation marks. The quotes are not part of the string.

Some examples of string variables are:

```
t = 'MathViews'
A = [ 'Here is a string';
'A:file1.m         ';
'c:\user\file.txt  ']
```

## Matrices

The basic unit of analysis in MathViews™ is a rectangular matrix. Scalars are 1-by-1 matrices and vectors are matrices with one row or column. The simplest way to enter a matrix is to enter an explicit list of elements. For example, to define a 2-by-3 matrix, enter:

```
A=[1 2 3;5 6 7]
```

The individual elements are separated by a comma or a space, and rows are separated by semicolons. Each element can be any MathViews™ expression, including other matrices. This allows complex matrices to be build up rather easily.

## Waveforms

Waveforms are one dimensional real matrices (arrays).

# 5.6 Expressions and Operators

## Expressions

An Expression is any series of symbols the MathViews™ uses to produce a value. The simplest expressions are constants and variable names, which have no operator and generate a value directly.

## Operators

Operators combine expressions and values to produce new values. They are classified in three ways: by type, associativity and precedence.

<u>Types</u>

Operators are of three types:

- Unary operators, which operate on a single operand.
- Binary operators, which combine two operands to perform a variety of arithmetic operations.
- Assignment operators: assign a value to a variable, and optionally perform an additional operation before the assignment takes place.

<u>Associativity</u>

Associativity; is either left to right or right to left.

- Left to right: Operates on the quantity or parenthesized expression to its right.
- Right to left: Operates on the quantity or parenthesized expression to its left.

<u>Precedence</u>

Precedence specifies the order in which the indicated operations are performed. The operation with highest precedence is performed first.

<u>Classification</u>

The following table shows the set of MathViews™ operators arranged by precedence. Operators with highest precedence appear at the top of the table, and those with the lowest precedence appear at the bottom. Operators having the same precedence appear in the same row.

| Precedence and Associativity of MathViews™ Operators | | | |
|---|---|---|---|
| | Operator | Associativity | Type |
| Matrix power | ^ | left to right | Binary |
| Array power | .^ | | |
| Pre and post increment | ++ | left to right | Unary |
| Pre and post decrement | — | | |

| | | | |
|---|---|---|---|
| Transpose | ‘ | left | Unary |
| Conjugate Transpose | .’ | | |
| Matrix multiplication | * | left to right | Binary |
| Array multiplication | .* | | |
| Matrix division | / | | |
| Array division | ./ | | |
| Matrix division | \ | right to left | |
| Array division | .\ | right to left | |
| Addition | + | left to right | Binary |
| Subtraction | - | | |
| Greater than | > | left to right | Binary |
| Greater or equal | >= | | |
| Less than | < | | |
| Less or equal | <= | | |
| Equal (logical) | == | | |
| Not equal | ~= | | |
| And | & | left to right | Binary |
| Or | \| | | |
| Not | ~ | | Unary |
| Rotate | <> | left to right | Binary |
| Shift | >> | | |
| Pipe | -> | left to right | Binary |
| Colon | : | | |
| Assignments | = | left to right | |
| | += | | |
| | -= | | |
| | *= | | |
| | /= | | |
| | := | right to left | |

Parentheses

Parentheses are used to change the associativity of MathViews™ operators. Parentheses can be nested arbitrarily deep. An example of a parenthesized expression is:

```
(1+4*(5+2))*3+10
(A & B) | C | ( ~ A)
(sqrt(2) + A)
```

## Reserved Keywords

MathViews™ provides facilities for block-structured conditional control-flow statements. Several statement formats are included in MathViews™ to perform loops, to select other statements to be executed, and to transfer control. The list of the reserved keywords is: **if, elseif, else, for, while, break, return, end**

## Arithmetic Operations

Negation

**Syntax:** `-e1`

- If `e1` is a scalar: The result is a scalar and is the arithmetic negation of `e1`.
- If `e1` is a matrix: The negation operates on *every* element of `e1`, and the result is a matrix with the same dimensions as `e1`.

The negation operator is defined for real and complex expressions.

Addition

**Syntax:** `e1 + e2`

- If `e1` and `e2` are scalars, then the standard addition is performed.
- If `e1` is a scalar and `e2` is a matrix, then `e1` is added to every element of `e2`. The result is a matrix with the same dimensions as `e2`. Similar operations are taken if the types of `e1` and `e2` are reversed.
- If `e1` and `e2` are matrices, then addition of matrices is performed on an element-by-element basis. The result is a matrix with the same dimensions as `e1` and `e2`. *Matrix addition is defined only for matrices with the same dimensions.*

The addition operator is defined for real and complex expressions.

Subtraction

**Syntax:** `e1 - e2`

The subtraction operator adds to `e1` the arithmetic negation of `e2`, i.e. `e1-e2=e1+(-e2)`. All the rules of addition apply to subtraction.

Matrix Multiplication

**Syntax:** `e1 * e2`

- If `e1` and `e2` are scalars, then the standard multiplication is performed.
- If `e1` is a scalar and `e2` is a matrix, then every element of `e2` is multiplied by `e1`. The result is a matrix with the same dimensions as `e2`. Similar operations are taken if the types of `e1` and `e2` are reversed.
- If `e1` and `e2` are matrices, then *the multiplication is only defined when the number of columns of e1 is the same as the number of rows of e2*. The standard matrix multiplication is performed. This yields a matrix whose numbers of rows correspond to the number of rows of `e1` and whose number of columns

corresponds to the number of columns of `e2`.

The matrix multiplication operator is defined for real and complex expressions.

Matrix Division (right)

**Syntax**: `e1 / e2` (right division)

- If `e1` and `e2` are scalars, then standard division is performed, (`e1` divided by `e2`).
- If `e1` is a scalar and `e2` is a matrix, then every element of `e2` is set to `e1` times the reciprocal of the element of `e2`. The result is a matrix with the same dimensions as `e2`.
- If `e1` is a matrix and `e2` is a scalar, then every element of `e1` is divided by `e2`. The result is a matrix with the same dimensions as `e1`.
- If `e1` and `e2` are matrices, then, if the inverse of `e2` exists, the result is `e1*inv(e2)`. The rules of matrix multiplication then apply to `e1*inv(e2)`.

It is an error to divide by 0. The matrix division operator is defined for real and complex expressions.

Matrix Division (left)

**Syntax**: `e1 \ e2` (left division)

- If `e1` and `e2` are scalars, then standard division is performed, i.e. `e2` divided by `e1`.
- If `e1` is a scalar and `e2` is a matrix, then every element of `e2` is divided by `e1`. The result is a matrix with the same dimensions as `e2`.
- If `e1` is a matrix and `e2` is a scalar, then every element of `e1` is set to `e2` times the reciprocal of the elements of `e1`. The result is a matrix with the same dimensions as `e1`.
- If `e1` and `e2` are matrices, then, if the inverse of `e1` exists, the result is `inv(e1)*(e2)`. The rules of matrix multiplication then apply to `inv(e1)*e2`.

It is an error to divide by 0.

The matrix division operator is defined for real and complex expressions.

Array Multiplication

**Syntax**: `e1 .* e2`

- If `e1` and `e2` are scalars, then the operation is the same as `e1*e2`.
- If `e1` is a scalar and `e2` is a matrix, then the operation is the same as `e1*e2`.
- If `e1` and `e2` are matrices, an element-by-element multiplication of `e1` and `e2` is performed. The result is a matrix with dimensions of `e1` and `e2` following the rule:

$$result_{ij} = e1_{ij} * e2_{ij}$$

*The dimensions of matrices e1 and e2 must be equal.*

The array multiplication operator is defined for real and complex expressions.

Array Division (right)

**Syntax:** `e1 ./ e2` (right array division)

- If `e1` and `e2` are scalars, then the operation is the same as `e1/e2`.
- If `e1` is a scalar and `e2` is a matrix, then the operation is the same as `e1/e2`.
- If `e1` is a matrix and `e2` is a scalar, then the operation is the same as `e1/e2`.
- If `e1` and `e2` are matrices, then an element by element division of `e1` by `e2` is performed, independent of the existence of the inverse of `e2`. The result is a matrix with dimensions of `e1` and `e2` following the rule:

$$result_{ij} = e1_{ij} / e2_{ij}$$

*The dimensions of e1 and e2 must be equal.*

It is an error to divide by 0. Right and left divisions are the same, except that the roles of the numerator and the denominator are switched.

The array division operator is defined for real and complex expressions.

Array Division (left)

**Syntax:** `e1 .\ e2` (left array division)

- If `e1` and `e2` are scalars, then the operation is the same as `e1\e2`.
- If `e1` is a scalar and `e2` is a matrix, then the operation is the same as `e1\e2`.
- If `e1` is a matrix and `e2` is a scalar, then the operation is the same as `e1\e2`.
- If `e1` and `e2` are matrices, then an element by element division of `e2` by `e1` is performed, independent of the existence of the inverse of `e1`. The result is a matrix with dimensions of `e1` and `e2` following the rule:

$$result_{ij} = e2_{ij} / e1_{ij}$$

*The dimensions of e1 and e2 must be equal.*

It is an error to divide by 0. Right and left divisions are similar, except that the roles of the numerator and the denominator are switched.

The array division operator is defined for real and complex expressions.

Exponentiation

**Syntax:** `e1 ^ e2`

- If `e1` and `e2` are scalars, then `e1` is raised to the `e2` power.
- If `e1` is a matrix and `e2` is a scalar, then `e1` is multiplied by itself `e2` number of times, following the rules of matrix multiplication. The result is a matrix with the same dimensions as `e1`. However, reversing the roles of `e1` and `e2` is an error.
- If `e1` and `e2` are both matrices, the operation is undefined and causes an error.

The matrix exponentiation operator is defined for real and complex expressions.

Array Exponentiation

**Syntax:** `e1 .^ e2`

- If `e1` and `e2` are scalars, then the operation is the same as `e1 ^ e2`.
- If `e1` is a matrix and `e2` is a scalar, then the operation is the same as `e1 ^ e2`.
- If `e1` and `e2` are matrices, then, element by element, each element of `e1` is raised to the power of the corresponding element of `e2`, following the rule:

$$result_{ij} = e1_{ij} \wedge e2_{ij}$$

*The dimensions of e1 and e2 must be equal.*

The array exponentiation operator is defined for real and complex expressions.

**Logical And Relational Operations**

MathViews™ provides a full set of relational and logical operators to support control flow statements. All the logical and relational operators are defined for *real* and *complex numbers*. However, when complex numbers are involved, only the real part of the complex number is used for operations. One exception is that the NOT operator does not support complex numbers.

The logical and relational operators yield a 0 if the specified relation is false and a 1 if the relation is true. Care must be exercised when comparing floating point numbers.

Greater Than

**Syntax:** `e1 > e2`

- If `e1` and `e2` are scalars, then the result is a 1 if `e1` is greater than `e2`; otherwise the result is a 0.
- If `e1` is a scalar and `e2` is a matrix, then the comparison will operate on every element of `e2` against the scalar `e1`, yielding a 1 if `e1` is greater than the element of `e2`, otherwise yielding a 0. The result is a logical matrix with dimensions of `e2`. The same operation is performed if `e1` and `e2` exchange positions.
- If `e1` and `e2` are matrices, then *the dimensions of e1 and e2 must be equal.*

Every element of e1 is compared to the corresponding element of `e2` following the rule:

$$result_{ij} = (e1_{ij} > e2_{ij})$$

The result is a logical matrix with the same dimensions as `e1` and `e2`.

Greater Than or Equal To

**Syntax:** `e1 >= e2`

- If `e1` and `e2` are scalars, then the result is a 1 if `e1` is greater than or equal to `e2`; otherwise the result is a 0.

- If `e1` is a scalar and `e2` is a matrix, then the comparison will operate on every element of `e2` against the scalar `e1`, yielding a 1 if `e1` is greater than or equal to the element of `e2`, otherwise yielding a 0.The result is a logical matrix with dimensions of `e2`.The same operation is performed if `e1` and `e2` exchange roles.
- If `e1` and `e2` are matrices, *the dimensions of e1 and e2 must be equal.* Every element of `e1` is compared with the corresponding element of `e2` following the rule:

$$result_{ij} = (e1_{ij} >= e2_{ij}).$$

The result is a logical matrix with the same dimensions as `e1` and `e2`.

Less Than

**Syntax**: `e1 < e2`

- If `e1` and `e2` are scalars, then the result is a 1 if `e1` is less than `e2`; otherwise the result is a 0.
- If `e1` is a scalar and `e2` is a matrix, then the comparison will operate on every element of `e2` against the scalar `e1`, yielding a 1 if `e1` is less than the element of `e2`, otherwise yielding a 0. The result is a logical matrix with the dimensions of `e2`. The same operation is performed if `e1` and `e2` exchange roles.
- If `e1` and `e2` are matrices, *the dimensions of e1 and e2 must be equal.* Every element of `e1` is compared with the corresponding element of `e2` following the rule:

$$result_{ij} = (e1_{ij} < e2_{ij}).$$

The result is a logical matrix with the same dimensions as `e1` and `e2`.

Less Than or Equal To

**Syntax**: `e1 <= e2`

- If `e1` and `e2` are scalars, then the result is a 1 if `e1` is less than or equal to `e2`, otherwise the result is a 0.
- If `e1` is a scalar and `e2` is a matrix, then the comparison will operate on every element of `e2` against the scalar `e1`, yielding a 1 if `e1` is less than or equal to the element of `e2`; otherwise it yields a 0. The result is a logical matrix with dimension of `e2`. The same operation is performed if `e1` and `e2` exchange roles.
- If `e1` and `e2` are both matrices, *the dimensions of e1 and e2 must be equal.* Every element of `e1` is compared with the corresponding element of `e2` following the rule:

$$result_{ij} = (e1_{ij} <= e2_{ij}).$$

The result is a logical matrix with the same dimensions as `e1` and `e2`.

### Equal To

**Syntax:** `e1 == e2`

- If `e1` and `e2` are scalars, then the result is a 1 if `e1` equals `e2`; otherwise the result is a 0.
- If `e1` is a scalar and `e2` is a matrix, then the comparison will operate on every element of `e2` against the scalar `e1`, yielding a 1 if `e1` equals the element of `e2`; otherwise it yields a 0. The result is a logical matrix with the dimensions of `e2`. The same operation is performed if `e1` and `e2` exchange roles.
- If `e1` and `e2` are both matrices, *the dimensions of e1 and e2 must be equal.* Every element of `e1` is compared with the corresponding element of `e2` following the rule:

$$result_{ij} = (e1_{ij} == e2_{ij}).$$

The result is a logical matrix with the same dimensions as `e1` and `e2`.

### Not Equal To

**Syntax:** `e1 ~= e2`

- If `e1` and `e2` are scalars, then the result is a 0 if `e1` equals `e2`; otherwise the result is a 1.
- If `e1` is a scalar and `e2` is a matrix, then the comparison will operate on every element of `e2` against the scalar `e1`, yielding a 0 if `e1` equals the element of `e2`, otherwise yielding a 1. The result is a logical matrix with dimension of `e2`. The same operation is performed if `e1` and `e2` exchange roles.
- If `e1` and `e2` are matrices, *the dimensions of e1 and e2 must be equal.* Every element of `e1` is compared with the corresponding element of `e2` following the rule:

$$result_{ij} = (e1_{ij} \sim= e2_{ij}).$$

The result is a logical matrix with the same dimensions as `e1` and `e2`.

### Logical Negation

**Syntax:** `~e1`

- If `e1` is a scalar, then the result is a 1 if `e1` is 0, and the result is 0 if `e1` is non zero.
- If `e1` is a matrix, then the logical negation operates on each element individually, following the rule stated above.

The logical negation operator does not support complex numbers and complex matrices.

### Logical And

**Syntax:** `e1 & e2`

- If `e1` and `e2` are scalars, then the result is a 1 if `e1` and `e2` are not 0; otherwise the result of the logical AND is a 0.
- If `e1` is a scalar and `e2` is a matrix, then the logical AND operates on each

element of e2 individually and logically compares them with the real value e1, following the rule stated above. The result is a logical matrix with the same dimensions as e2. The same operation is performed if e1 and e2 exchange positions.

- If e1 and e2 are matrices, *the dimensions of e1 and e2 must be equal.* Every element of e1 is AND'd with the corresponding element of e2 following the rule:

$$result_{ij} = e1_{ij} \;\&\; e2_{ij}.$$

The result is a logical matrix with the same dimensions as e1 and e2.

The logical negation operator does not support complex numbers and complex matrices.

### Logical Or

**Syntax:** e1 | e2

- If e1 and e2 are scalars, then the result is a 0 if e1 and e2 both are 0; otherwise the result of the logical OR is a 1.
- If e1 is a scalar and e2 is a matrix, then the logical OR operates on each element of e2 individually and logically compares them with the scalar value e1, following the rule stated above. The result is a logical matrix with the same dimensions as e2. The same operation is performed if the positions of e1 and e2 are exchanged.
- If e1 and e2 are matrices, *the dimensions of e1 and e2 must be equal.* Every element of e1 is OR'd with the corresponding element of e2 following the rule:

$$result_{ij} = e1_{ij} \;|\; e2_{ij}.$$

The result is a logical matrix with the same dimensions as e1 and e2.

The logical negation operator does not support complex numbers and complex matrices.

## Assignment Operations

MathViews™ has several assignment operators, all of which group right-to-left. The left side of an assignment must be a variable and/or named sub matrix. Assignment operators will release any memory previously allocated to a variable before dynamically allocating memory for the new assignment.

### Simple Assignment

**Syntax:** e1 = e2

The left side of the assignment operator, e1, must be a legitimate variable name, and the right side, e2, can be any legitimate MathViews™ expression or string. Any previous storage that had been allocated to e1 is released for future allocation, before the assignment takes place.

Array Assignment

**Syntax**: `e1(rV,cV) = e2`

`e1` is a **named** variable and must already exist prior to making the assignment. `e1(rV,cV)` references a sub matrix in `e1` that is replaced with `e2`, on an element by element basis. The sub matrix of `e1` must have the same dimensions, i.e., equal number of elements, as `e2`. `rV` and `cV` are vectors specifying the rows and columns of `e1` to be replaced by the elements of `e2`. Note that a scalar is a single-element vector.

Additive Compound Assignment

**Syntax**: `e1 += e2`

The additive compound assignment syntax is a short hand notation for `e1=e1+(e2)`.

Subtractive Compound Assignment

**Syntax**: `e1 -= e2`

The subtractive compound assignment syntax is a short-hand notation for `e1=e1-(e2)`.

Multiplicative Compound Assignment

**Syntax**: `e1 *= e2`

The multiplicative compound assignment syntax is a short-hand notation for `e1=e1*(e2)`.

Division Compound Assignment

**Syntax**: `e1 /= e2`

The division compound assignment syntax is a short-hand notation for `e1=e1/(e2)`. However the \= compound assignment is not supported.

Increment(++) and Decrement(--) Operators

**Syntax**: `++e1;  e1--;  e1++;  e1--`

MathViews™ implements the C operators for incrementing and decrementing variables. The increment operator `(++)` adds 1 to its operand while the decrement operator `(--)` subtracts 1 from its operand. These operators may be used either prefixed, as in `++x`, or postfixed, as in `x++`, but the two cases have different meanings. The variable `x` is incremented in both cases, but in the expression `++x`, `x` is incremented before its value is used, while `x++` increments `x` after using its value. When `x` is an array, each element of the array will be incremented by 1. You should be aware of the different side-effects of these two operators.

For complex numbers, only the real part is incremented or decremented. There is no space between the variable name and the `--` or the `++` operators.

Dependence Operator

**Syntax:** `e1 := e2`

This assignment makes variable `e1` dependent on variable `e2`. If the value of `e2` changes for any reason, `e1` is **automatically** updated to reflect the new value of `e2`. However, if `e1` is then reassigned, the old assignment is nullified.

You must be cautious when using the dependence operator. Circular relationships will trigger an error. For example, `e1 := e2, e2 := e3; e3 := e1.`

## Matrix Operations

Matrix operators in MathViews™ permit you to manipulate matrix elements. Matrix operators operate on any type of matrix element, either complex or real.

Vertical Concatenation

**Syntax:** `[e1; e2; ...]`

The number of rows of the resulting matrix is the sum of the number of rows of `e1` and `e2` and any other matrices specified. The number of columns remains constant.

The number of columns in all of the matrices, `e1, e2, ...,` must be equal for vertical concatenation.

Horizontal Concatenation

**Syntax:** `[e1,e2, ...]`

The number of columns of the resulting matrix is the sum of the number of columns of `e1` and `e2` and any other matrices specified. The number of rows remains constant.

The number of rows in all of the matrices, `e1, e2, ...,` must be equal for horizontal concatenation.

Linear Shifting

**Syntax:** `e1 >> e2`

Matrix rows and columns can be shifted in any direction, up, down, left, or right. The operand `e2` can be a real scalar or a 2-element real vector. 0's are padded in when the rows/columns are shifted out.

- If `e2` is scalar, then `e1` is shifted row-wise. If `e2` > 0, `e1` is shifted up, and if `e2` < 0, e1 is shifted down. If `e2` = 0, e1 remains the same.
- If `e2` is a vector, then the first element of `e2` specifies the number of rows to be shifted, and the second element specifies the number of columns to be shifted. A positive element shifts left or up; a negative element shifts right or down.
- If `e2` is a matrix or a vector of more than 2 elements, MathViews™ will only use the first two elements of the vector or the first two elements of the first column of the matrix.

Linear shifting is not bit shifting. The operation affects only whole elements of rows/columns of a matrix.

Circular Shifting

**Syntax**: `e1 <> e2`

Matrix rows and columns can be circularly shifted in any direction, up, down, left, or right. The operand `e2` can be a real scalar or a 2-element real vector.

- If `e2` is a scalar, then `e1` is shifted row-wise. If `e2` > 0, `e1` is circularly shifted up, and if `e2` < 0, `e1` is circularly shifted down. If `e2` = 0, `e1` remains the same.
- If `e2` is a vector, then the first element of `e2` specifies the number of rows to be circularly shifted and the second element specifies the number of columns to be shifted. A positive element circularly shifts left or up, and a negative element circularly shifts right or down.
- If `e2` is a matrix or a vector of more than 2 elements, MathViews™ will only use the first two elements of the vector or the first two elements of the first column of the matrix.

Circular shifting is not circular bit shifting. The operation affects only whole elements of rows/columns of a matrix.

Conjugate Transpose

**Syntax**: `e1'`

This unary operator takes the argument to its left and performs a conjugate transpose of the elements of `e1`.

Matrix Transpose

**Syntax**: `e1.'`

This unary operator takes the argument to its left and transposes the elements in `e1`.

Matrix Shaping

**Syntax**: `e1:optional_expr: e2`

This form of the colon operator will create a **row** vector, with the first element set to `e1`, and subsequent elements following the rule:

$$e_i = e_{i-1} + \text{optional\_ expr}$$

$$\text{where } i \geq 1,\ e_i \leq e2,\ e_0 = e1$$

`e1`, `e2`, and the `optional _expr` must evaluate to real scalars. If the `optional_expr` is not provided as in `e1:e2`, then `optional_expr` is assumed to be 1. If an undefined range is specified, as in `1:-1:5` then an empty matrix is returned.

If `e1`, `e2`, or `optional_expr` are complex, only the real part is used in MathViews™ calculations.

## Pipe Operation

Piping

**Syntax**: `expr -> function`

MathViews™provides a simple notation for a sequence of built-in function calls. The output of one function may be connected to the input of another function using the pipe operator. This notation is useful for replacing parentheses when the expression is deeply nested. It can also reduce the probability of syntax errors due to missing closing parentheses.

This syntax can only be used when the first function returns only **one** argument. For example:

```
5->sin->abs
ans =
        0.9589
```

## Statement Operations

Continuation

**Syntax**: `expr ...`

The continuation operator,"...", is used to continue from one line of text to another without breaking the flow of input. When processing the command or statement internally, MathViews™ treats the continuation operator as a mark indicating that the next line after the ellipsis actually belongs to the current line.

No Output

**Syntax**: `expr;`

When the semicolon operator appears **after an expression**, it is a message to the MathViews™internal engine to suppress any output resulting from the expression.

Comments

**Syntax**: `%comments`

A sequence of characters beginning with the percent operator (%) is treated as white space by the MathViews™ internal engine. A comment can contain any combination of characters up until the newline character. A newline character terminates the commenting sequence.

Multiple comment lines must have the % operator at the beginning of each sequence of commenting text.

## 5.7 Control Flow Statements

MathViews™ provides facilities for block-structured control flow that specifies the order in which computations are performed. The syntax and semantics of the MathViews™ control-flow facilities follow those of the MATLAB language.

If-Else Statement

**Syntax:**

**if** *expression*
    if statements
**elseif**
    elseif statements
**else**
    else statements
**end**

The **elseif** and **else** parts are optional. The *expression* is first evaluated, and if it is true, i.e. *expression* has a non zero value, the if statements are executed. Otherwise, if *expression* is false, i.e. the expression yields a 0 value, and if there is an **else** part, the else statements are executed.

While Loop

**Syntax:**

**while** *expression*
    while statements
**end**

The *expression* is evaluated and if it is true, i.e. non zero, the while statements are executed. This is repeated as long as the expression is true. The loop continues until the expression becomes false, at which point the execution resumes after the while statement.

For Loop

**Syntax:**

**for** *loop_variable=expression*
    for statements
**end**

This loop is executed N times, where N is the number of columns of *expression*. The *loop_variable* takes the value of each column of the *expression*.
For example, if expression is 1:5, the loop is executed five times with the loop_variable taking on the scalar values from 1 to 5.

## 5.8 User Defined Functions

Function Declaration

**Syntax:**

**[out1,out2,...,outM]**=funcname(**in1,in2,...,inN**)

function statements

A user-defined function resides in an external file whose name is the same as the function name and whose extension is '**.m**.' The file is referred to as an **M-file.** A function can have multiple arguments and may return multiple values.

Input arguments are passed by value, and output arguments are pass by reference. Both arguments have local scope. There are two run-time variables, `nargin` and `nargout`, which contain the number of input and output arguments that the function was invoked with.

Functions may be called recursively, but the function stack has limited depth. Functions can call any other functions in the same module or a different module. However, when calling other modules, the modules must be in MathViews™ path.

The square brackets can be omitted, but only if the function only returns one argument. It is best, as a standard practice, to include the square brackets at all times.

## 5.9 Built-In Functions

### ABS

Absolute Value Function.

**Syntax:** abs(expr)

The abs function computes the absolute value of its argument. If the argument is an array, the function operates on each element of the array, and the result is an array of the same dimension as the argument.

**Examples:**

abs(-3.2, 3.4)

3.200 34

abs(-1, 2, -3)

1.00 2.00 3.00

**Typical Application:** *Generate full wave-rectified version of a waveform. See the example waveform "full_wave_rectifier" from the "demo_ops.wfm".*

## ACOS

Arccosine Function.

**Syntax:** acos(expr)

The acos function computes the arccosine (inverse cosine) of its argument and returns a radian value in the proper range. If the argument is an array, the function operates on each element of the array, and the result is an array of the same dimension as the argument. The absolute value argument of this function should be less than 1.

**Examples:**

acos(0)

    1.570796

acos(1)

    0.000

**Typical Application:** *For more information, see Example #6 under* ***Creating Waveforms*** *later in this section.*

## ASIN

Arcsine Function.

**Syntax:** asin(expr)

The asin function computes the arcsine (inverse sine) of its argument and returns a radian value in the proper range. If the argument is an array, the function operates on each element of the array, and the result is an array of the same dimension as the argument. The absolute value argument of this function should be less than 1.

**Example:**

asin(1)

    1.5708

**Typical Application:** *For more information, see Example #6 under* ***Creating Waveforms*** *later in this section.*

## ATAN

Arctangent Function.

**Syntax:** atan(expr)

The atan function computes the arctangent (inverse tangent) of its argument and returns a radian value in the proper range. If the argument is an array, the function operates on each element of the array and the result is an array of the same dimension as the argument.

**Examples:**

atan(0)

    0.00

atan(1)*4

    3.141593

atan(1e9)

    1.570796

**Typical Application:** *Simulate signal distortion. See the example waveform "distorted_signal_2" from the "demo_ops.wfm".*

## COS

Cosine Function.

**Syntax:** cos(expr)

The cos function computes the cosine of its radian argument. If the argument is an array, the function operates on each element of the array, and the result is an array of the same dimension as the argument.

**Examples:**

cos(0)

1.00

cos(pi)

-1.00

**Typical Application:** *Generate sine and cosine based waveforms. For more information, see Example #4 under* ***Creating Waveforms*** *later in this section.*

## COSH

Hyperbolic Cosine Function.

**Syntax:** cosh(expr)

The cosh function computes the hyperbolic cosine of its argument. If the argument is an array, the function operates on each element of the array, and the result is an array of the same dimension as the argument.

**Examples:**

cosh(0)

1.000

cosh(1)

1.54308

**Typical Application:** *Generate complex amplitude or frequency modulation envelopes for sine and cosine based waveforms.*

## COSW

Sinusoidal Waveform Function.

**Syntax:** cosw(expr)

The cosw function is used to create sinusoidal waveforms. The argument should be a real scalar or a real array.

If the argument is a scalar, a single cycle sinusoidal array is created whose number of elements is equal to the value of the argument.

If the argument is an array, then:

1. the value of the first element of the array determines the number of elements of the waveform sequence

2. the value of the second element determines the number of cycles of the waveform
3. the value of the third element, when it is not zero, determines that frequency deviation of the waveform
4. the value of the forth element determines the initial phase of the waveform. For Example:

| You Enter | You Obtain |
|---|---|
| 1 | 360° (or 2π radians) |
| 0.5 | 180° or 0.5(2π radians) |
| 0.25 | 90° or 0.25(2π radians) |

If the array has less than four elements, the value of the undefined elements defaults to zero.

The cosw function is identical to the sinw() function except that the default value of the initial phase is 90 degrees rather than zero degrees.

**Examples:**

Let x = 128, 4

| | |
|---|---|
| sign(cosw(x)) | square wave |
| cosw(128,4,6) | linear sweep |

**Typical Application:** *Generate sinusoidal carrier or envelope waveforms, linear sweep, FM signal. For more information, see Example #3 under **Creating Waveforms** later in this section.*

## COT

Cotangent Function.

**Syntax:** cot(expr)

The cot function computes the cotangent of its radian argument. The value of cot(x) at its "singular points" is undefined. If the argument is an array, the function operates on each element of the array, and the result is an array of the same dimension as the argument.

**Examples:**

cot(pi/2)

0.00

cot(pi/4)

1.00

**Typical Application:** *Generate complex amplitude or frequency modulation envelopes for sine and cosine based waveforms.*

### CSUM

Cumulative Sum of an Array.

**Syntax:** csum(expr)

The csum function computes the cumulative sum of its argument. If the argument is not an array, it is left unchanged. When the argument is an array the function returns an array of the same dimension as the argument.

Examples:

Let x be equal to

1 2 3 4

then csum(x) is equal to

1 3 6 10

**Typical Applications:** Compute the integral of a waveform. Convert a square wave to a sine wave. *See the example waveform "integrated_signal" from the "demo_ops.wfm".*

### DECIMATE

Decimation of an Array.

**Syntax:** decimate(expr1 ,expr2)

The decimate function reduces the first argument by deleting elements from it at regular intervals. The first argument must be a named variable and the second argument, expr2, must be a scalar. If expr2 is scalar, every $(expr2\text{-}1)^{th}$ element will be deleted from expr1.

**Typical Application:** *Reduce the sampling rate of a slowly varying signal. This is the process involved in "shrink to fit" when resizing a waveform smaller.*

### EXP

Exponential Function.

**Syntax:** exp(expr)

The exp function computes the exponential function of its argument. If the argument is an array, the function operates on each element of the array, and the result is an array of the same dimension as the argument.

**Examples:**

exp(0)

1.000

exp(1)

2.718282

**Typical Application:** *Generate exponentially rising or decaying waveforms. For more information, see Example #7 under* ***Creating Waveforms*** *later in this section.*

## EXPAND

Expand an Array With Interleaving Zeros.

**Syntax:** expand(exprl ,expr2)

The expand function expands the first argument interleaving zeros between its elements. The first argument must be a named Variable and the second argument, expr2. must be a scalar. If expr2 is scalar, (expr2-1) zeros will be placed after each element of expr1.

**Examples:**

Let x be equal to

1 2 3

then

expand(x, 2) return

1 0 2 0 3 0

expand(x, 3) returns

1 0 0 2 0 0 3 0 0

**Typical Application:** *For more information, see Example #5b under* ***Creating Waveforms*** *later in this section.*

## FLIP

Mirror Image of an Array.

**Syntax:** flip(expr)

The flip function reverses the order of the elements of its argument. Scalar arguments are left unchanged by this function.

Examples:

Let x be equal to

1 2 3 4

then

flip(x) returns

4 3 2 1

**Typical Application:** Convert a rising ramp to a falling ramp. *For more information, see Example #2 under* ***Creating Waveforms*** *later in this section.*

## INDX

Linear ramp Waveform.

**Syntax:** indx(expr)

The indx function is used to create a linear ramp waveform, such as {VAL_0, VAL_0+D_VAL, VAL_0+2*D_VAL, ... ,VAL_1}. The argument should be a real scalar or a real array.

If the argument is a scalar K, the function returns an array whose VAL_0=0, VAL_1=K, and D_VAL is either +1, if K is positive or -1, if K is negative.

If the argument is an array:

1. VAL_0 is equal to the first element of the array,
2. VAL_1 is equal to the second element of the array, and
3. The absolute value of D_VAL is equal to
   a) the third element of the array if the argument has three elements
   b) if the argument has two elements.
4. The sign of D_VAL is equal to the sign of the expression (VAL_1 - VAL_0)

**Examples:**

indx(4)

| | | | | |
|---|---|---|---|---|
| 0 | 1 | 2 | 3 | 4 |

indx(-4)

| | | | | |
|---|---|---|---|---|
| 0 | -1 | -2 | -3 | -4 |

indx(2, 4)

| | | |
|---|---|---|
| 2 | 3 | 4 |

indx(3, 2, 0.5)

| | | |
|---|---|---|
| 3.0 | 2.5 | 2.0 |

**Typical Application:** *Generate linear ramps. For more information, see Examples #1 and #2 under* ***Creating Waveforms*** *later in this section.*

## LOG

Natural logarithm function.

**Syntax:** log(expr)

The log function computes the natural logarithm of its argument. If the argument is an array the function operates on each element of the array and the result is an array of the same dimension as the argument. The function is not defined for zero and negative values.

**Typical Application:** *Simulate compression For more information, see Example #7 under* ***Creating Waveforms*** *later in this section.*

## LOG10

Base-10 Logarithm Function.

**Syntax:** log10(expr)

The log10 function computes the base 10 logarithm of its argument. If the argument is an array, the function operates on each element of the array and the result is an array of the same dimension as the argument. The function is not defined for zero and negative values.

**Typical Application:** *Compute signal levels in dB. For more information, see Example #5a under* ***Creating Waveforms*** *later in this section.*

## MAX

Find the largest element of the Array(s).

**Syntax:** max(expr)

max(expr1, expr2)

The max function is used to:

1. Compute the maximum value of an array
2. Force all the elements of an array to be greater than a given threshold
3. Select the bigger elements of a two arrays on an element-by-element basis

When the max function is invoked with a single argument, the function returns a scalar which is the maximum value of its argument which may either a scalar or an array.

When the max function is invoked with two arguments, then:

Case 1: If both arguments are scalars, the function returns a scalar whose value is equal to the value of the bigger number.

Case 2: If both arguments are arrays of the same dimension, the function returns an array with the same dimension, whose elements are an element-by-element maximum value.

Case 3: If one argument is scalar and the second is an array, the function returns an array with the same dimension as the array argument and the same elements except those elements. which are smaller than the scalar. The elements which are less than the scalar are assigned the scalar value.

*This function is different from other built-in functions in that the comma is used for parameter passing and not for array concatenation.*

Examples:

Let x be equal to

| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| 33 | 16 | 72 | -24 | -10 | -57 | -8 | -4 | -17 |

and let y be equal to

| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| 3 | 61 | 27 | -42 | -1 | -57 | -18 | 4 | -7 |

then max(x) = 72 and max(x,10) is equal to

| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| 33 | 16 | 72 | 10 | 10 | 10 | 10 | 10 | 10 |

and max(x, y) is equal to

| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| 33 | 61 | 72 | -24 | -1 | -57 | -8 | 4 | -7 |

**Typical Application:** *Affix or simulate variable threshold in curves See the example waveform "half_wave_rectifier" from the "demo_ops.wfm". Can also be used to create the time function max(t) for trigonometric functions.*

## MIN

Find the smallest element of the Array(s).

**Syntax:** min(expr)

min(expr1, expr2)

The min function is used to:

1. Compute the minimum value of an array
2. Force all the elements of an array. to be smaller than a given threshold

3. Select the smaller elements of two arrays on an element-by-element basis
When the min function is invoked with a single argument, the function returns a scalar which is the minimum value of its argument which maybe either a scalar or an array. When the min function is invoked with two arguments then:-

Case 1: If both arguments are scalars, the function returns a scalar whose value is equal to the value of the smallest number.

Case 2: If both arguments are arrays of the same dimension, the function returns an array, with the same dimension, whose elements are an element-by-element minimum value.

Case 3: If one argument is scalar and the second is an array the function returns an array, with the same dimension as the array argument and the same elements except those elements which are bigger than the scalar. The elements which are bigger than the scalar are assigned the scalar value.

*This function is different from other built-in functions in that the comma is used for parameter passing and not for array concatenation.*

**Examples:**

Let x be equal to

| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| 33 | 16 | 72 | -24 | -10 | -57 | -8 | -4 | -17 |

and let y be equal to

| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| 3 | 61 | 27 | -42 | -1 | -57 | -18 | 4 | -7 |

then min(x) = -57 and min(x,10) is equal to

| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| 10 | 10 | 10 | -24 | -10 | -57 | 8 | -4 | -17 |

and min(x, y) is equal to

| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| 3 | 16 | 27 | -42 | -10 | -57 | -18 | -4 | -17 |

**Typical Application:** *Simulate clipping in an amplifier. See the example waveform "w3" from the "demo.wfm".*

## NOISE

Uniform noise Waveform.

**Syntax:** noise(expr)

The noise function returns an array whose elements are randomly distributed in the interface [-1, 1] and whose length equals the value of the argument. The argument should be a real scalar (typically the **Size** value).

*To seed the random number generator, use the rand function.*

**Typical Application:** *Add noise to a Waveform. For more information, see Example #8 under **Creating Waveforms** later in this section.*

## RAND

Random number generator, uniform distribution.

**Syntax:** rand(expr)
rand(expr1, expr2)
seed = rand('seed')
rand('seed', newseed)

The rand function generates random numbers.

If the argument is an integer, an expr by expr matrix is generated, with elements being random numbers in the range of (0,1).

If the argument is a matrix, the function will generate a matrix with dimensions equal to that of the argument; the resulting matrix contains randomly generated elements.

If two arguments are used to invoke the function, both arguments must be integers. The function will generate an expr1 by expr2 random elements matrix.

rand('seed') returns the current seed. rand('seed', 0) reset the seed to zero.

*The quality of any random number generator is a highly debatable issue. For code portability, MathViews™ calls the random number generator provided in the standard math library of the C compiler. rand() calls are handled automatically by the NOISE function.*

## RANDN

Random number generator, normal distribution.

**Syntax:** randn(expr)
randn(expr1, expr2)
seed = randn('seed')
randn('seed', newseed)

The randn function generates random numbers.

If the argument is an integer, an expr by expr matrix is generated , with elements being random numbers with normal distribution.

If the argument is a matrix, the function will generate a matrix with dimensions equal to that of the argument; the resulting matrix contains randomly generated elements.

If two arguments are used to invoke the function, both arguments must be integers. The function will generate an expr1 by expr2 random elements matrix.

randn('seed') returns the current seed. randn('seed', 0) reset the seed to zero

*Waveform values can exceed the range of (-1,+1).*

## REPT

Duplicate and concatenate array.

**Syntax:** rept(expr1, expr2)

The rept function duplicates the first argument and concatenates it. The second argument, expr2, must be a real scalar, and expr1 must be a named variable. Expr1 will be duplicated expr2 times.

**Examples:**
Let x be the array.
1 2 3
then
rept(x, 2) will return
1 2 3 1 2 3
**Typical Application:** *Create a periodic signal.*
*Example:*
Let signal2 = signal1[0,100]
then signal3 = rept(signal2,9)

## ROUND

Round integer numbers.
**Syntax:** round(expr)
The round function recalculates the expression for the Waveform, but with less precision. The effect is to produce a copy of the Waveform with less resolution. This simulates the degradation of the Waveform if downloaded to an Arb which cannot maintain the number of data points in the original Waveform.
**Examples:**
round(S1*2^(n-1))/2^(n-1)
S1-round(S1*2^(n-1))/2^(n-1)
**Typical Applications:** *Simulate an n-bit quantizer. Analyze quantization error. See the example waveform "digitized_signal_4bit" and then "digitized_error" from the "demo_ops.wfm".*
*Examples:*
round(S1*2.^(n-1))/2.^(n-1)
S1 - round(S1*2.^(n-1))/2.^(n-1)

## SIGN

Sign function.
**Syntax:** sign(expr)
The sign function computes the sign of its argument and returns +1 for a positive argument value and -1 for a negative argument value, and zero for an argument value of zero. If the argument is an array , the function operates on each element of the array, and the result is an array of the same dimensions as the argument.
**Typical Application:** *Create square waves. Simulate hard limiting receiver (1-bit quantizer). For more information, see Examples #8 under* ***Creating Waveforms*** *later in this section.*

### SIN

Sine function.

**Syntax:** sin(expr)

The sin function computes the sine of its radian arguments. If the argument is an array, the function operates on each element of the array, and the result is an array of the same dimensions as the argument.

**Typical Application:** *Generate sine and cosine based Waveforms. For more information, see Example #4 under* ***Creating Waveforms*** *later in this section.*

### SINH

Hyperbolic sine function.

**Syntax:** sinh(expr)

The sinh function computes the hyperbolic sine of its argument. If the argument is an array, the function operates on each element of the array, and the result is an array of the same dimensions as the argument.

**Typical Application:** *Generate complex amplitude or frequency modulation envelopes for sine and cosine based Waveforms.*

### SINW

Sinusoidal Waveform.

**Syntax:** sinw(expr)

The sinw function is used to create sinusoidal waveforms. The argument should be a real scalar or a real array.

If the argument is a scalar, a single cycle sinusoidal array is created whose number of elements is equal to the value of the argument.

If the argument is an array, then:

1. the value of the first element of the array determines the number of elements of the waveform sequence
2. the value of the second element determines the number of cycles of the waveform
3. the value of the third element, when it is not zero, determines that frequency deviation of the waveform
4. the value of the forth element determines the initial phase of the waveform. For Example:

| You Enter | You Obtain |
|---|---|
| 1 | 360° (or $2\pi$ radians) |
| 0.5 | 180° or 0.5($2\pi$ radians) |
| 0.25 | 90° or 0.25($2\pi$ radians) |

If the array has less than four elements, the value of the undefined elements defaults to zero.

The sinw function is identical to the COSW function except that the default value of the initial phase is zero degrees rather than 90 degrees.
**Examples:**
sinw(128, 4, 6) sweeps upwards in frequency
sinw(128, 4, -6) sweeps downwards in frequency
**Typical Application:** *Generate a linearly swept frequency. The third element in* the array argument is the change in frequency. Generate sinusoidal carrier or *envelope waveform, FM signal. For more information, see Example #3 under **Creating Waveforms** later in this section.*

## SIZE

Dimension of argument.
**Syntax:** [m,n] = size(expr)
m = size(expr, 1)
n = size(expr, 2)

If the argument is a scalar, the function returns the value unity.
If the argument is an array, the function returns the number of rows and the number of columns as a row vector of two elements.
The function size(A,1) returns the number of rows of A.
The function size(A,2) returns the number of columns of A.
**Examples:**

```
size(1.2)
          [1 1]
size([1 2 3 4 5 6])
          [1 6]
size([1 2 3 4 5 6],1)
          1
size([1 2 3 4 5 6],2)
          6
```

## SQRT

Square root function.
**Syntax:** sqrt(expr)
The sqrt function computes the square root of its argument. If the argument is an array, the function operates on each element of the array, and the result is an array of the same dimension as the argument.
It is an error to take the square root of a negative number.
**Typical Application:** *Convert energy or power into volts.*

### SUM

Sum of an array.

**Syntax:** sum(expr)

The sum function computes the sum of all the elements of its argument. The function returns a scalar.

**Example:**

Let x be equal to

| | | | | | |
|---|---|---|---|---|---|
| 1 | 2 | 3 | 4 | 5 | 6 |

then

sum(x) return 21

**Typical Application:** *Calculate the total energy of the signal.*

### TAN

Tangent function.

**Syntax:** tan(expr)

The tan function computes the tangent of its radian argument. The value of tan(x) at its "singular points" is undefined. If the argument is an array, the function operates on each element in the array, and the result is an array of the same dimension as the argument.

**Typical Application:** *Simulate signal distortion. See the example waveform "distorted_signal_1" from the "demo_ops.wfm".*

### WNORM

Normalize a Waveform.

**Syntax:** wnorm(expr)

The normalization function is defined as x/max(abs(x)). This ensures that the wnorm(x) peak absolute is less than or equal to unity

**Typical Application:** *Keep a Waveform from overshooting its window. This is the process that takes place when "Auto Normalize" is selected in the Set-up waveforms dialog. You might want to select "Leave Alone" in order to create a Waveform which intentionally overflows, such as "wt = 2*pi*wnorm(t)". You would use the wnorm function to prevent subsequent overflows on a Waveform-by-Waveform basis.*

## 5.10 Creating Waveforms

These examples use simple waveforms to demonstrate WaveForm DSP2 concepts and the user interface. When creating waveforms by mathematical expression keep in mind that:-

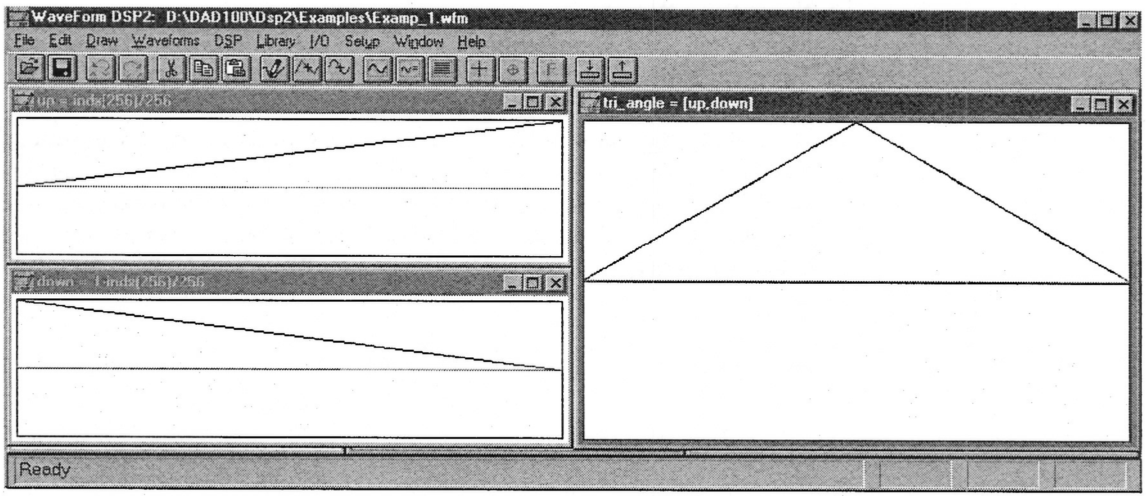
- The waveform may be normalized within the waveform window, independant of the full scale value assigned to it in the **Setup Waveforms...** dialog box.
- The values from the **Setup Waveforms...** dialog box are the basis for the value readout in the Status window, and where applicable the values downloaded to the ARB.
- When your expression is entered, it is evaluated. If it is valid it is displayed, otherwise a dialog box appears to notify you that an error occurred.
- For waveforms which are dependent on others then an auto update flag can be set in the Expression dialog box.

### Example #1 - Array Operators, Index Function

Open WaveForm DSP2 and accept the **Setup Waveforms...** defaults, except the **Size** is set to 256 and the **Mode** set **X/Y**.

1. Select **File** menu, **New** option.
2. Click to create a waveform and name it "up". Click and give it the expression "indx(256)/256".
3. Create a waveform "down = 1-indx(256)/256".
4. Create a waveform "triangle = [up,down]".

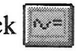
*Select **Tile** view. The waveforms should look like the illustration below.*



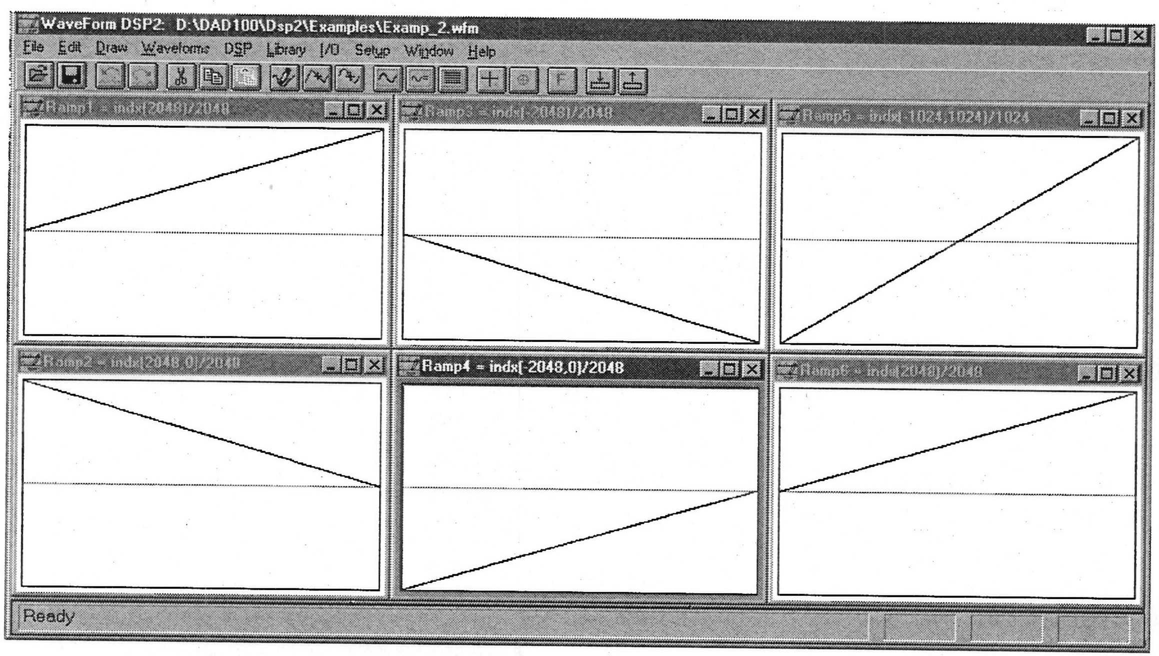

The function "up" was created usin the index function to create an array of 256 values from 0 to 255. Dividing the array by its maximum value normalizes it to the waveform block. You could do the same thing with the Setup Waveform... dialog Auto Normalize selection or on this waveform by using the WNORM function. The function "down" is created by subtracting the "up" array from the value "+1". Alternately, create the same waveform using the expression "flip(up)". The "triangle" is created by concatenating the "up" and "down" waveforms and produces a 512 point waveform.

### Example #2 - Index Function (Ramps)

Open WaveForm DSP2 and accept the **Setup Waveforms...** defaults, except the **Size** is set to 2048. Select **File** menu, **New** option.

1. Click  to create a waveform and name it "Ramp1". Click  and give it the expression "indx(2048)/2048".
2. Create a waveform "Ramp2 = indx(2048,0)/2048".
3. Create a waveform "Ramp3 = indx(-2048)/2048".
4. Create a waveform "Ramp4 = indx(-2048,0)/2048".
5. Create a waveform "Ramp5 = indx(-1024,1024)/1024".
6. Create a waveform "Ramp6 = indx(2048)/2048".

*Select **Tile** view. The waveforms should look like the illustration below.*




The index function can take up to three arguments but using just the one, as in Ramp1, sets up an array from 0 to 2048 with a positive step size of 1. To normalize it divide by 2048.
The two values in Ramp2 provide the start and stop numeric elements. The step size is again 1 but the sign of this step is derived by subtracting the first argument from the second.

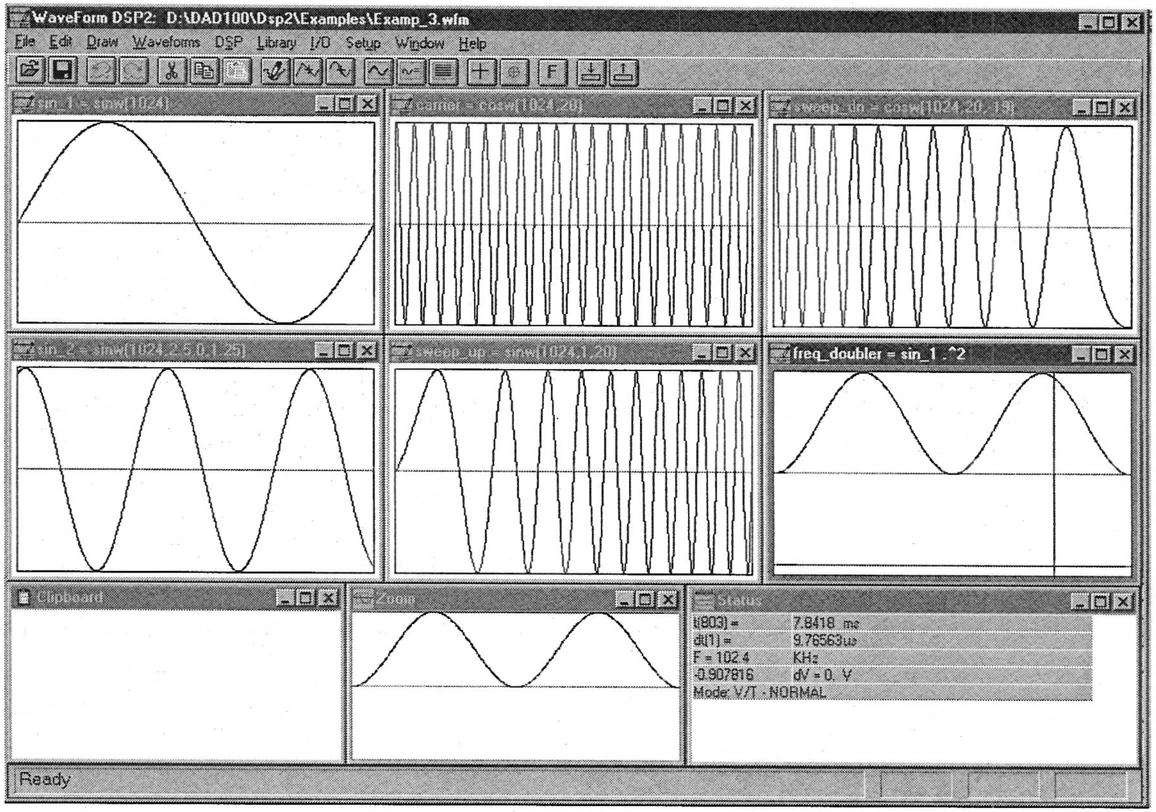
Ramps 3 and 4 are the negative values of Ramps 1 and 2.
Ramp 5 goes from a negative start point to a positive end point with a positive step of 1 but creates a waveform of 2048 points.

## Example #3 - Sinw, Cosw Functions

Open WaveForm DSP2 and accept the **Setup Waveforms...** defaults, except the **Size** is set to 1024. Select **File** menu, **New** option.

1. Click  to create a waveform and name it "sin_1". Click  and give it the expression "sinw(1024)".
2. Create a waveform "sin_2 = sinw(1024,2.5,0.0,0.125)".
3. Create a waveform "carrier = cosw(1024,20)".
4. Create a waveform "sweep_up = sinw(1024,1,20)".
5. Create a waveform "sweep_dn = sinw(1024,20,-19)".
6. Create a waveform "freq_doubler = sin_1 .^2".

*Select **Tile** view. The waveforms should look like the illustration below.*



NOTE

*Although the INDX, SINW and COSW functions are easy to use, they have one drawback. If you later resize the waveform file with waveforms created using these functions, you will have to resize them by modifying each expression. The arguments will have to be changed according to the change in* ***Size****.*

*If you think you might often be resizing your files, a better approach is to use a normalized time function, (t/max(t)) ot wnorm(t), which will be explained later as a horizontal indexing function. These have the advantage of being self-resizing.*

The simplest way to create a sinusoidal waveform is to use the SINW or COSW built-in functions. In the *sin_1* example the argument is a single value, which creates a single-cycle sinusoid array, whose number of elements is equal to the value of the argument. Since the number of elements equals the number of data points in the window, one cycle of the sine wave fits the window exactly.

The arguments (1024,2.5,0.0,0.125) for *sin_2* are interpreted as follows:

element 1: number of elements in the waveform sequence (1024)
element 2: number of cycles in the waveform sequence (2.5)
element 3: frequency deviation (swept frequency increase - None)
element 4: initial phase of the waveform (45∞).

The *carrier* has 20 cycles of cosine waves using all of the 1024 points

The arguments (1024,1,20) for *sweep_up* are interpreted as follows:

element 1: number of elements in the waveform sequence (1024)
element 2: number of cycles (at the start) in the waveform sequence (1)
element 3: frequency deviation (swept frequency increase - 20 cycles)
element 4: initial phase of the waveform (0 radians).

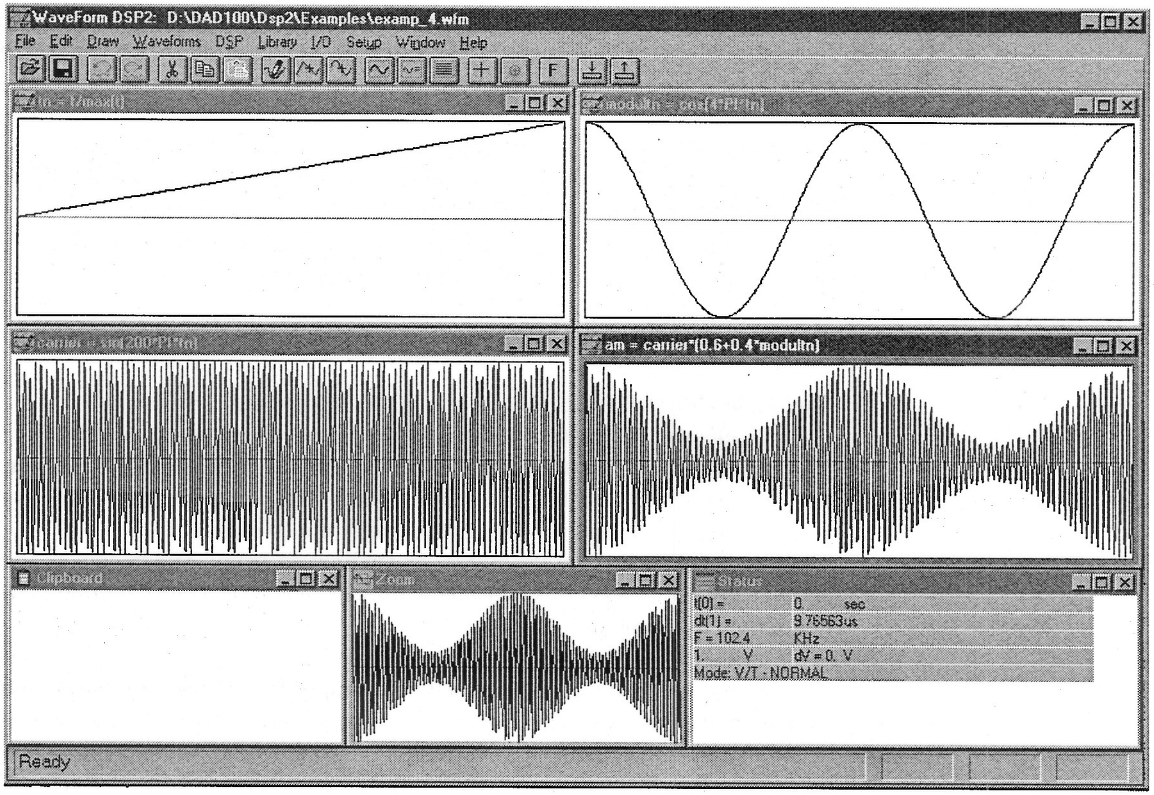
The waveforms *sweep_up* and *sweep_dn* are linearly swept. The frequency sweep starts at a waveform rate determined by element 2 and ends up at a rate determined by element 3 modified by element 3.

The final waveform, freq_doubler, in theis example demonstrates two things. First it uses the Exponentiation array operator ".^" to double frequency by squaring a sine wave. Secondly rather than typing the expression "sinw(1024).^2" the expression is simplified by referencing the first waveform by name. Waveforms within a single file can be referenced in this manner, but waveforms in other files cannot be referenced.

## Example #4 - AM Modulation

1. Select **File** menu, **New** option.
2. Create a waveform "tn = t/max(t)".
3. Create a waveform "carrier = sin(200*pi*tn)".
4. Create a waveform "modultn = cos(4*pi*tn)".
5. Create a waveform "am = carrier*(0.6+0.4(modultn))".

*Select **Tile** view. The waveforms should look like the illustration below.*




The function "*tn*" was created to simplify the expressions of the two following waveforms. It also creates a normalized ramp which may be used as the instantaneous time function, but it has no reference to the number of points in the waveform. Therefore waveforms created using "*tn*" are easily resized. Another way of creating the "*tn*" function is "*wnorm(t)*". A third method of creating a time scale is to create the function "*wt = 2*pi*wnorm(t)*". To create this waveform, you must set the **Overflow Control** in the **Setup**

**Waveforms..** dialog box to **Leave Alone**. This does eliminate the need for "2*pi" in each subsequent expression, and it leaves these expressions in a very recognizable form. However you must exercise care in the vertical scale of your other waveforms, or use the *wnorm* function to select the scale to your selected vertical size.

The function "*carrier*" could have used the convenient built-in function "sinw(2048,100)". However, by using "sin(2*pi*tn*100)" the waveform is now horizontally normalized and can be resized.

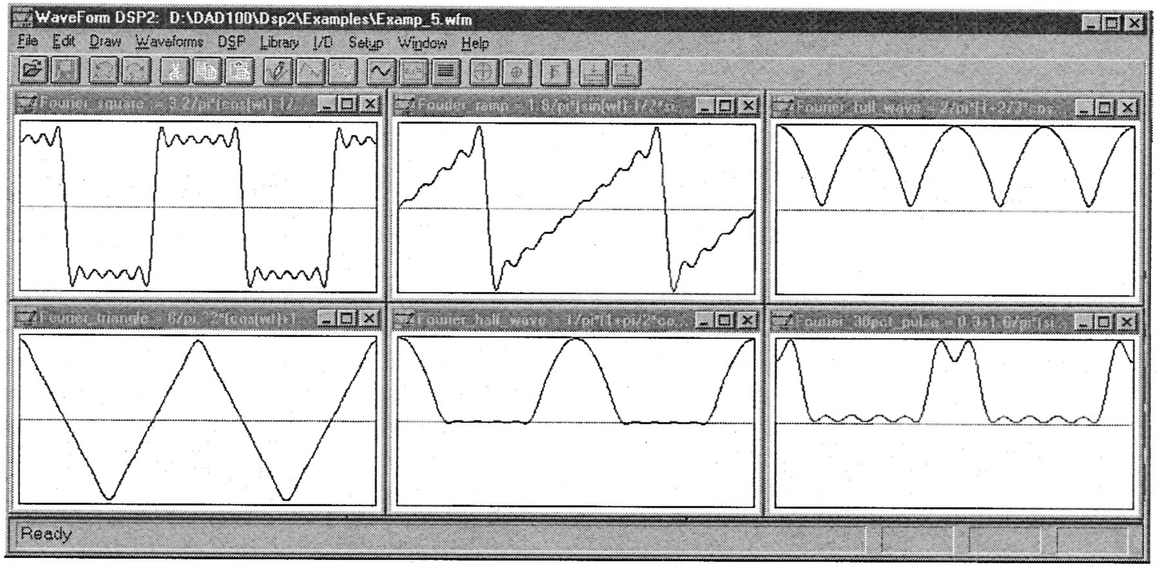
The function "*modultn*" also use the function "I" to simplify the expression.

Multiplying the carrier wave expression by the modulation wave expression results in the suppressed carrier modulation (SCM). For amplitude modulation (am), the modulation waveform must be offset (in this example, by 0.4) so as not to overmodulate the carrier. The offset value plus the amplitude value equals 1.0. The AM modulation index is twice the amplitude value (in this example 80%).

### Example #5 - Fourier waveforms

1. Select **File** menu, **New** option.
2. Create a waveform "wt = 2*pi*wnorm(t)" - **Overflow Control** set to **Leave Alone**.
3. Create a waveform "Fourier_square = 3.2/pi*(cos(wt)-1/3*cos(3*wt)+1/5*cos(5*wt)-1/7*cos(7*wt) +1/9*cos(9*wt)-1/11*cos(11*wt))".
4. Create a waveform "Fourier_triangle = 8/pi.^2*(cos(wt)-1/9*cos(3*wt)+1/25*cos(5*wt)-1/49*cos(7*wt))".
5. Create a waveform "Fourier_ramp = 1.8/pi*(sin(wt)-1/2*sin(2*wt)+1/3*sin(3*wt)-1/4*sin(4*wt) +1/5*sin(5*wt)-1/6*sin(6*wt) +1/7*sin(7*wt)-1/8*sin(8*wt) +1/9*sin(9*wt))".
6. Create a waveform "Fourier_half_wave = 1/pi*(1+pi/2*cos(wt)+2/3*cos(2*wt)-2/15*cos(4*wt)+2/35*cos(6*wt)-2/63*cos(8*wt))".
7. Create a waveform "Fourier_full_wave = 2/pi*(1+2/3*cos(2*wt)-2/15*cos(4*wt)+2/35*cos(6*wt)-2/63*cos(8*wt) +2/80*cos(9*wt)-2/99*cos(10*wt))".
8. Create a waveform "Fourier_30pct_pulse = 0.3+1.6/pi*(sin(0.9*pi)*cos(wt)+1/2*sin(0.6*pi)*cos(2*wt)+ 1/3*sin(0.9*pi)*cos(3*wt)+ 1/4*sin(1.2*pi)*cos(4*wt)+ 1/5*sin(1.5*pi)*cos(5*wt)+ 1/6*sin(1.5*pi)*cos(6*wt))".
9. Close the first waveform.

*Select **Tile** view. The waveforms should look like the illustration below.*

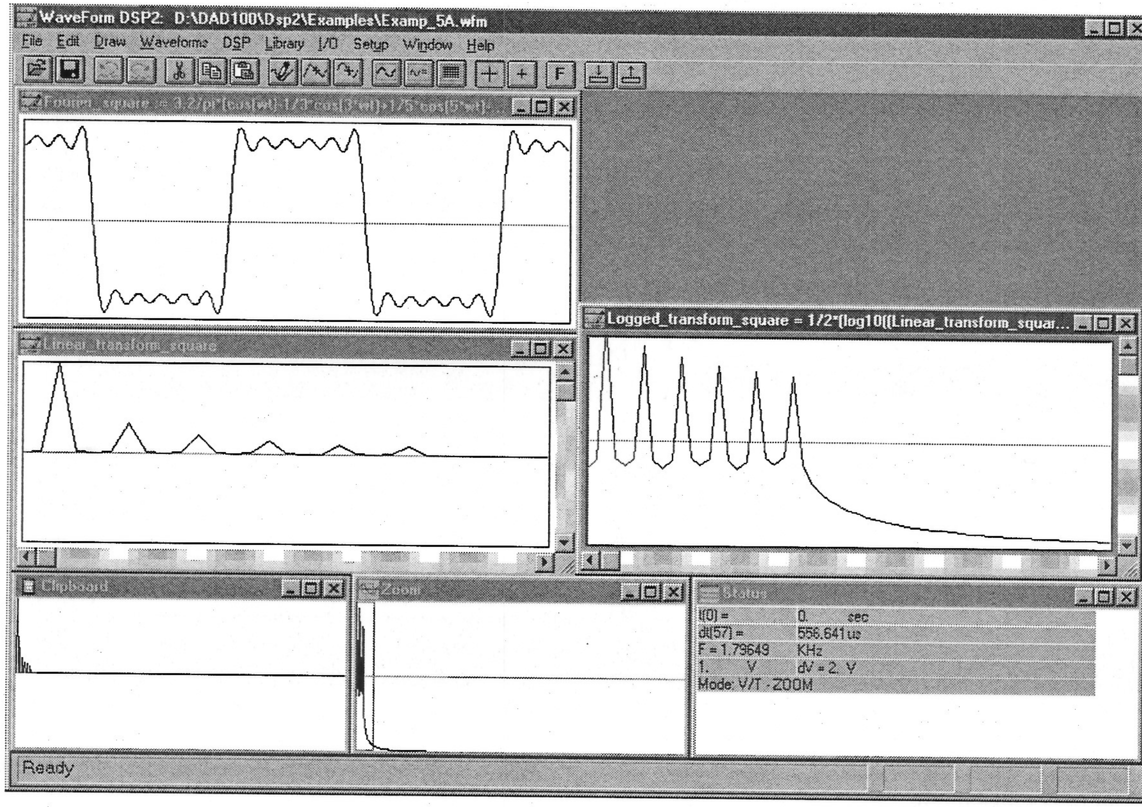



The function "wt" was created to simplify the expressions of the other waveforms. The advantage in using WaveForm DSP2 to create "Fourier" waveforms is that a waveform can be built up aterm or two at a time to demonstrate the Fourier series. With each addition of higher order harmonics, the waveform looks more like the desired final waveform. This also demonstrates how time domain waveforms might "roll off" in a bandwidth limited stage.

### Example #5A - Amplitude vs. Frequency, Log Scale Display of **_A

1. Do not close your file of Fourier waveforms created above.
2. Select the "*Fourier_square*" waveform. Click F in the toolbar to transform the waveform to the freqency &Amplitude domain. Zoom in on the display until you have isolated the spectral lines (one for each term in your time domain equation) as shown in the waveform "*Linear_transform_square*".
3. Ctrl_A will select all the points and Ctrl_C will copy them to the clipboard.Ctrl+n will allow creation of the new waveform "*Linear_transform_square*" and Ctrl+V will paste the clipboard data in this new waveform.
4. Create a new waveform "Logged_transform_square = _*(log10((Linear_transform_square*199.98)+0.01))".
5. Close all the waveforms except the "*Fourier_square*" and the two you have just created.

*Select **Tile** view. The waveforms should look like the illustration below.*

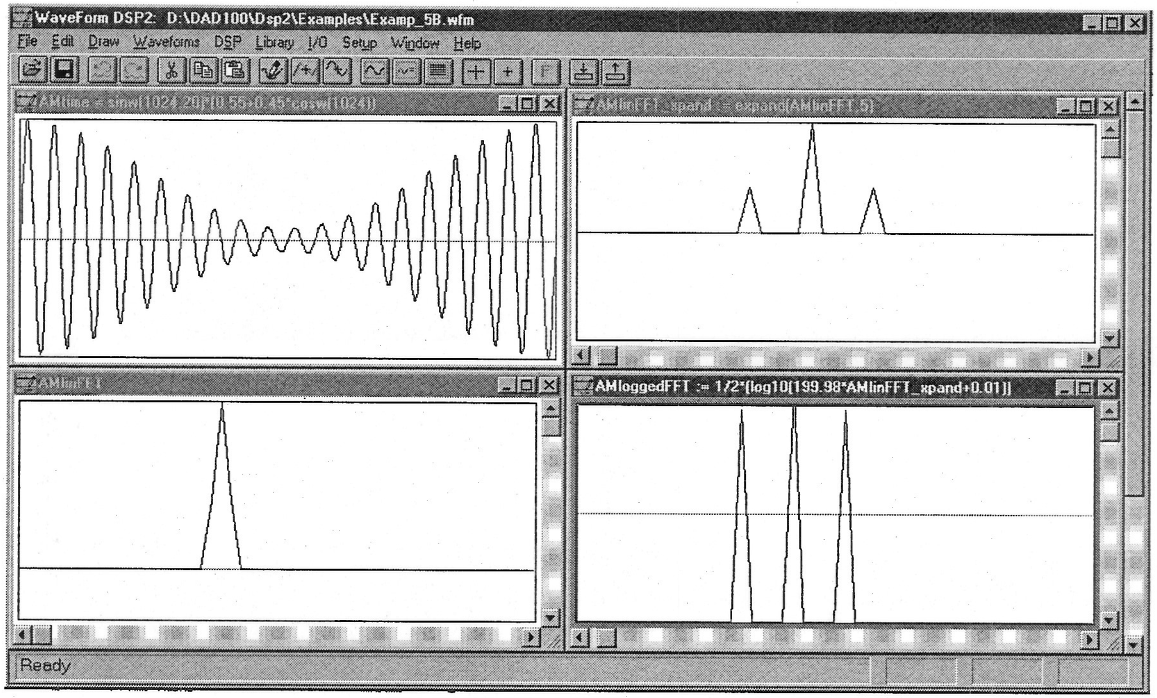



The Fourier series "square" was selected to transform into the frequency domain. Note that the number of odd harmonics corresponds to the number of terms in the series. The amplitude of the fundamental is "1" in the time domain and "0.5" in the frequency domain (the other 0.5 is in negative frequency). Relative to the fundamental, the harmonics should fall off in amplitude according to the series 1/3, 1/5, 1/7, 1/9... To get a better view of the spectrum we capture the frequency domain spectrum and return it to the time domain for post processing.

### Example #5B - Using expand function with ** _A

1. Select **File** menu, **New** option. Set **Size** to 1024.
2. Create a waveform "AMtime = sinw(1024,20)*(0.55_0.45*cosw(1024))".
3. Following the procedure in example 5A, capture Amtime_A and create the waveofrm "AMlinFFT".
4. Create a waveform "AMlinFFT_xpand = expand(AMlinFFT,5)".
5. Create a waveform "AMloggedFFT = _*(log10(199.98* AMlinFFT_xpand +0.01))".

*Select **Tile** view. The waveforms should look like the illustration below.*




The function "*AMtime*" was created without regard to how the waveform would appear in the frequency domain. In general, for good results with FFT you should:

1. Have more than one cycle of the lowest periodic frequency.
2. Have all periodic frequencies at integer relationships to the block.
3. If rule 2 cannot be met, or if non-periodic components are present , use Windowing Techniques (Example 9).

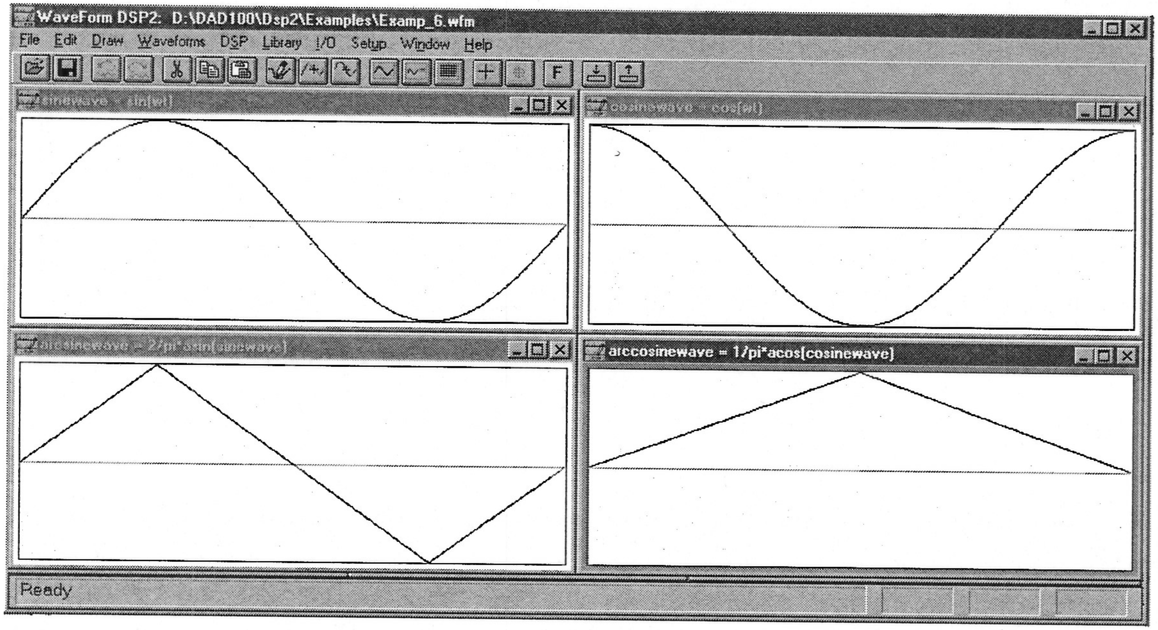
In the function "*AMtime*" the modulation envelope is at the block fequency. Therefore the AM sidebands will not be resolved. This could have been prevented by creating the waveform *"AMtime = sinw(2048,40)*(0.55_0.45*cosw(2048,2))"* using a **Size** setting of 2048.

Using the techniques in Example 5B , we have captured AMtime_A and copied it into a time domain waveform, *"AMlinFFT"*. The EXPAND function is used to resolve out the sidebands, and then the 80dB log function derived in Example 5B is used to give the spectrum dynamic range.

## Example #6 - ARCCOS, ARCSIN Functions

1. Select **File** menu, **New** option.
2. Create a waveform "wt = 2*pi*wnorm(t)" - **Overflow Control** set to **Leave Alone.**
3. Create a waveform "sinewave = sin(wt)".
4. Create a waveform "arcsinewave = 2/pi*asin(sinewave)".
5. Create a waveform "cosinewave = cos(wt)".
6. Create a waveform "arccosinewave = 1/pi*acos(cosinewave)".
7. Close the waveform "wt".

*Select **Tile** view. The waveforms should look like the illustration below.*

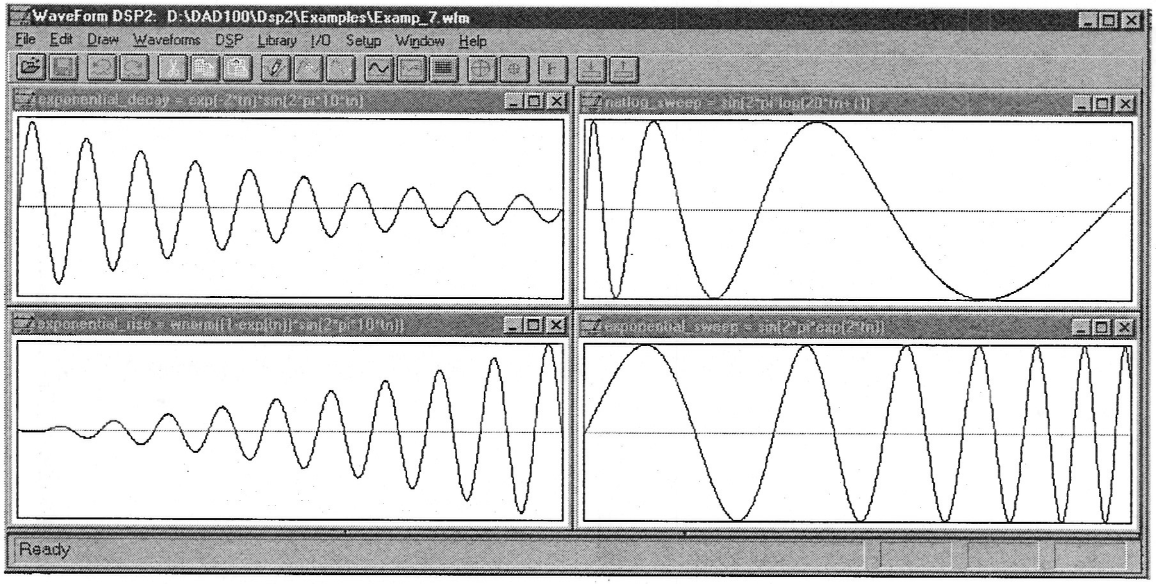



The "arc" waveforms do little else than demonstrate the ASIN and ACOS functions. However the "*arcsinewave*" does illustrate the unique way to create a triangle, and the "*arccosinewave*" could be used as a triangular Windowing function (see Example 9).

## Example #7 - Exponential and Log Functions

1. Select **File** menu, **New** option.
2. Create a waveform "tn = t/max(t)".
3. Create a waveform "exponential_decay = exp(-2*tn)*sin(2*pi*10*tn)".
4. Create a waveform "exponential_rise = wnorm(1-exp(tn))*sin(2*pi*10*tn)".
5. Create a waveform "exponential_sweep = sin(2*pi* exp(2*tn))".
6. Create a waveform "natlog_sweep = sin(2*pi* log(20*tn+1))".
7. Close the waveform "tn".

*Select **Tile** view. The waveforms should look like the illustration below.*

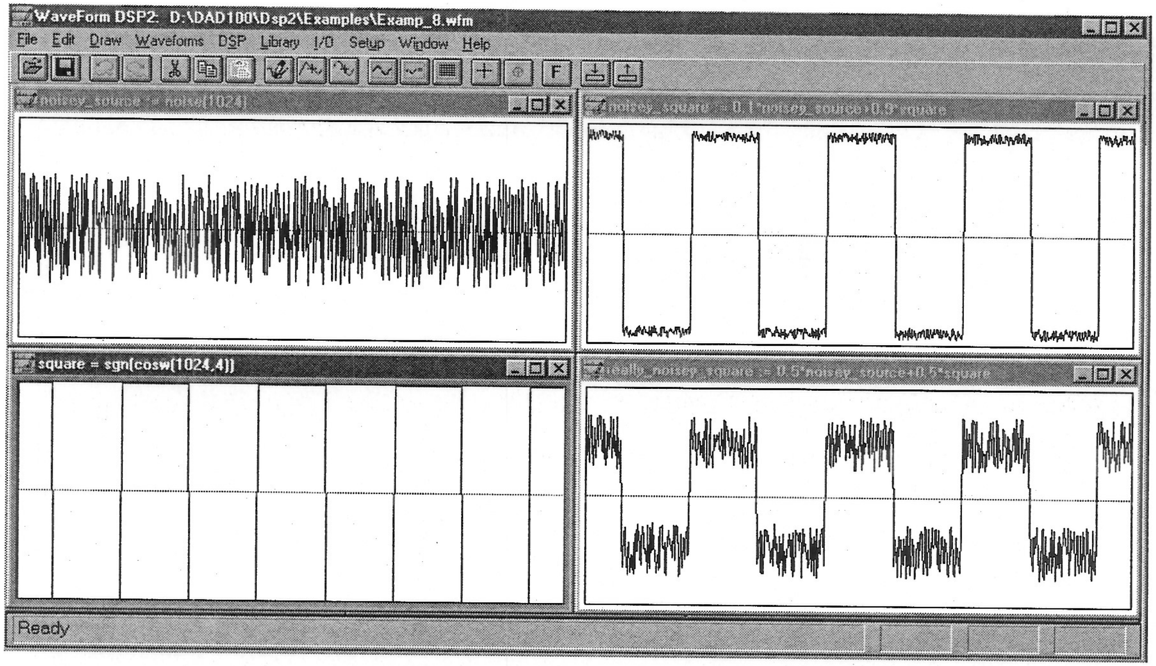



The function "*tn*" was created to simplify the expressions of the following waveforms. The first two waveforms are the product of the exponential (base2) function and a sinusoid. In the "*exponential_decay*" waveform, the EXP function is taken from t = 0 (where its value is unity) to t = -2 (where its value is near zero). A simple product with this exponential curve and a sine wave creates a very useful waveform, which can be copied to a library waveform and pasted into your waveforms.
In the "*exponential_rise*" waveform, the EXP function is taken from t = 0 (where its value is unity) to t = +1 (where its value is *e*). A curve from zero to approximately -1.72 is created by subtracting the exponential function from +1, and the result is multiplied by the sinewave. The WNORM function returns the resulting waveform to the correct vertical scale (unnecessary if the **Setup Waveforms..** dialog has **Auto Normalize** selected). This waveform could have been more simply created with the expression *"exponential_rise = flip(exponential_decay)"*.

## Example #8 - Noise Function

1. Select **File** menu, **New** option. Set **Size** to 1024.
2. Create a waveform "noisey_source = noise(1024)".
3. Create a waveform "square = sgn(cosw(1024,4))".
4. Create a waveform "noisey_square = 0.1* noisey_source+0.9*square".
5. Create a waveform "really_noisey_square = 0.5* noisey_source+0.5*square".

*Select **Tile** view. The waveforms should look like the illustration below.*

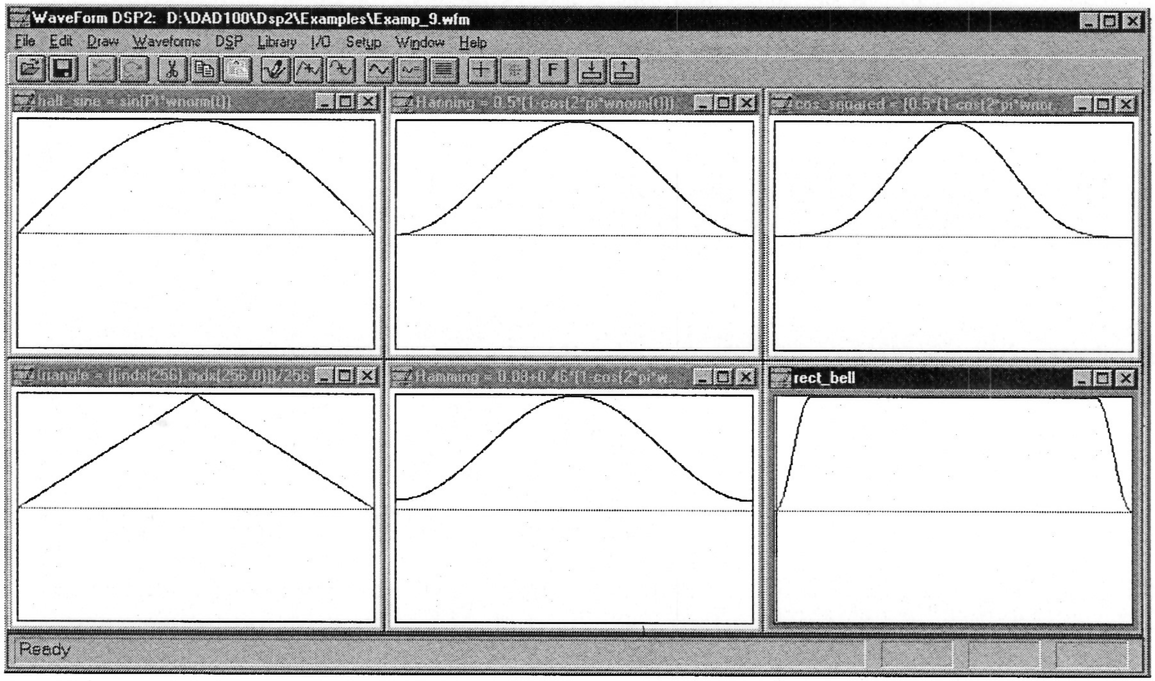


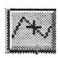

The NOISE function creates the "*noisey_source*" waveform. The next waveform is an example of how you can create a square wave using the SGN function and a sinusoid. The final two waveforms represent 20dB and 0dB Signal to Noise Ratios (S:N).

## Example #9 - Windowing Techniques

1. Select **File** menu, **New** option. Set **Size** to 512.
2. Create a waveform "half_sine = sin(pi*wnorm(t))".
3. Create a waveform "triangle = ([indx(256),indx(256,0)])/256".
4. Create a waveform "Hanning = 0.5*(1-cos(2*pi*wnorm(t)))".
5. Create a waveform "Hamming = 0.08+0.46*(1-cos(2*pi*wnorm(t)))".
6. Create a waveform "cos_squared = (0.5*(1-cos(2*pi*wnorm(t)))).^2".
7. Create a waveform "rect_bell = 0.5*(1-cos(10*pi*wnorm(t)))".

*Select **Tile** view. The waveforms (except the last) should look like the illustration below.*




Select the "*rect_bell*" waveform, and then select the line tool  from the toolbar. Draw a straight line from the top of the first cosine to the top of the last cosine. The rectangular bell windowing waveform should now look like the one in the illustration.

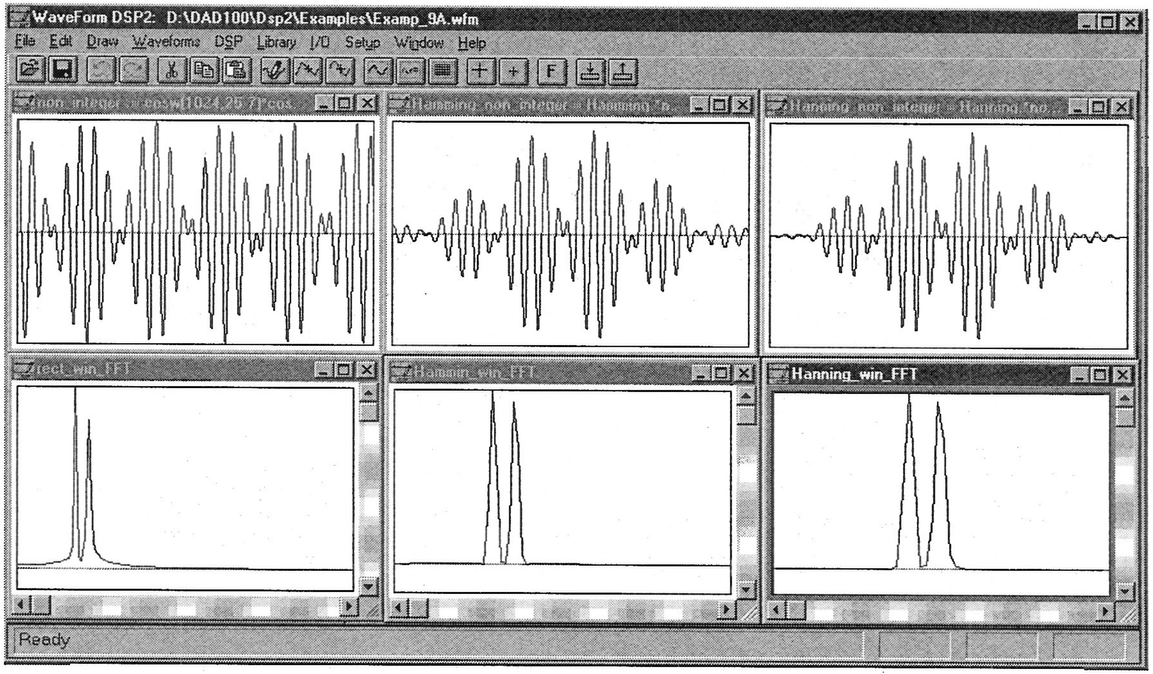
You should recall from the discussions following the DSP Menu in section 3 of this manual, that the FFT assumes that the rectangular "window" formed by the waveform block is periodic. The waveform in the block is repeated by concatenating the stop address to the start address. When the FFT is performed, this rectangular window contributes a *sin x/x* response to the resulting frequency spectrum. However if all waveforms components and the sampling frequency are integer multiples of the block frequency, the zero amplitude nodes of the sin x/x response line up at the spectral lines of the waveform. Therefore the sin x/x response of the rectangular "window" does not distort the spectrum of the waveform. However if the waveform is non-periodic or has non-integer related components, the spectral lines and the sin x/x nodes are off alignment. The spectrum will become distorted due to *leakage* from the sin x/x response. Another way of looking at the issue is to visualize your non-periodic waveform the way the FFT does. Wrap the stop point around to the start point, and there will be a discontinuity. This discontinuity adds energy to the spectrum. In fact the abrupt transitions act like square waves, creating the sin x/x response at the block frequency.

The "windowing" waveforms in this example are a means of minimizing the effect of non-periodic responses on the FFT. You can copy these windowing functions to a waveform library. When you find you have to create a non-periodic waveofrm and want an FFT with the sinx/x responses filtered out, you can accomplish this by multiplying your waveform by one of these windows before running the FFT. In effect this is converting from the rectangular window of the waveform block to a window which reduces the waveform amplitude in the area of the discontinuity.

## Example #9A - Windowing results

1. Add the following to the current file.
2. Create a waveform "non_integer = cosw(1024,25.7)* cosw(1024,2.6)".
3. Run the FFT on the waveform "non_integer" and select all and copy the "non_integer_A" waveform. Then select T from the toolbar to return to the Time domain for the original waveform. Create a new waveform "rect_win_FFT" and paste the contents of the clipboard to it.
4. Create a waveform "Hamming_non_integer = Hamming.*non_integer".
5. Following the procedure in step 3 above paste the transform of this waveform into a new waveform "Hamming_win_FFT".
6. Create a waveform "Hanning_non_integer = Hanning.*non_integer".
7. Following the procedure in step 3 above paste the transform of this waveform into a new waveform "Hanning_win_FFT".
8. Close all but the waveforms in this example 9A.

*Select **Tile** view. The waveforms should look like the illustration below.*



The function "*non_integer*" was created to demonstrate the effects of *leakage* on the FFT spectrum of a non-periodic waveform. The continuous waveform described by the expression "*non_integer*" is the familiar Suppressed Carrier Double Sideband Modulation (SCM) waveform. This produces a distinctive spectrum of two spectral lines at $f_c \pm f_m$. The rectangularily windowed transform "*rect_win_FFT*" indicates a very distorted SCM response. Note how non-symmetrical the sidebands are, and the amount of leakage around the base of the sidebands. The response obtained after multiplying with the Hamming window is much improved but *leakage* is just discernable. For the Hanning response, the *leakage* is no longer visable in this linear display. The log scaling developed in example 5A could be used to improve the dynamic range of the FFT responses.

# 6 Working with Arbs

When downloading to an Arb, WaveForm DSP2 sends information concerning the size (number of data points), frequency, and amplitude of the waveform, along with the waveform data. In some cases, specific start and stop addresses are also sent. You must give DSP2 this information so that it can be passed along to the target Arb. WaveForm DSP2 provides dialog boxes for this purpose.

It is very important that you be well acquainted with the characteristics of your target Arb. WaveForm DSP2's drivers can only make decisions on Arb-specific matters that are not optional. Other matters require decisions from the operator based on his knowledge of the task at hand and how that task relates to the characteristics of the Arb.

You should also be familiar with the *User Interface* and *Working with Equations* sections before attempting to download or upload waveforms.

The download procedures work for ARBs for which WaveForm DSP2 has access to download scripts. They require that a benchtop target ARB is connected to the computer running WaveForm DSP2 via either a GPIB or serial interface.
If the target is a VXI Arb this requires a VXI environment, either connected to an external host computer running WaveForm DSP2 or an embedded VXI computer running the same application.
The upload procedures work for instruments for which WaveForm DSP2 has access to upload scripts. A variety of popular Digital Storage Oscilloscopes have scripts available and additions can be made by creating a new script (using one of those supplied as a template along with file 'templ_ul.m') and modifying the file 'dsp2u.cfg' to include the new script.
The use of upload requires that the target instrument is connected to the computer running WaveForm DSP2 via either a GPIB or serial interface.

The GPIB or COMn computer interface must be properly set up and functioning. Be aware that the target Arb may also require setting up to conform with the GPIB Address or serial communication settings being used by the application.
During the initialization of WaveForm DSP2 the VXI resource manager is run to determine the instruments available and their addresses. The results are available in an Address pick list.

## 6.1 Setting the Waveform Parameters for an Arb

1. Select the  button from the toolbar or from the **File** menu select the **Open...** option.
2. Select the file containing the waveform you wish to work with.
3. Select the window with the waveform you wish to view by clicking on an exposed portion of that window.
   *If the window you want is hidden, or for some other reason you cannot determine which of the waveform windows to select, choose the Window menu, tile option. In the Tile mode, the waveforms are smaller but they are displayed side-by-side, and none are hidden.*
4. Select from the **Setup** menu the **Waveforms...** option.

The now familiar Setup Waveforms dialog box appears. The Set-up dialog will contain your settings from the waveform file. Look them over and see if that is what you want to be downloaded to the Arb.

**Size:** If you want a different number of points in the Arb waveform memory than in the waveform file, you can resize at the time of Arb download, leaving the file the same size. The only drawback is that the re-sizing algorithm uses linear interpolation rather than the waveform's expression to resize during download.
*If you choose to resize the waveform file, you will probably want to choose "Stretch / Shrink ALL waveforms" when you close the Set-up dialog. Be aware of possible loss of resolution if you are making the **Size** smaller. Also be aware of increased file size of the .wfm file on your system memory if you are making the **Size** larger.*

**Overflow Control:** - If you have "**Leave Alone**" checked for your current file, make sure that the waveform you are downloading does not exceed the vertical scale of the waveform window, or you will get errors from the Arb. "**Auto-Normalize**" can be used to normalize the waveform, or "**Clip**" will limit the waveform to the maximum Y Values.

**Mode:** - If **X/Y** is selected, only the data and the size (number of points) is editable. The frequency and amplitude values downloaded will be the current or default values stored in the waveform tile. In **X/Y Mode**, your **Y Values** selection is for your convenience, it does not effect the Arb. Your waveform will be scaled to the vertical resolution of the Arb. In **V/T Mode**, the waveform block is scaled to the vertical resolution of your Arb, as for **X/Y Mode** above. The Arb's Amplitude is programmed to give the same Vp (into its' specified termination) as set up in the dialog box.
*If your waveform fills the waveform window vertically and you type "1" in the Block Amplitude dialog box, your downloaded waveform amplitude will be 2Vpp (into specified termination, 4Vpp open circuit).*

*If your waveform fills half the waveform window vertically and you type "1" in the Block Amplitude dialog box, your downloaded waveform amplitude will be 1 Vpp (into specified termination).*

In **V/T Mode**, you can also specify waveform frequency. This is given in terms of Block Frequency, Sample Frequency or Sample Time. Make sure you have the size parameter set where you want it, then you can set or change the frequency by setting any one of these three frequency parameters. The other two will update themselves when you select "**Apply**" or select another parameter.

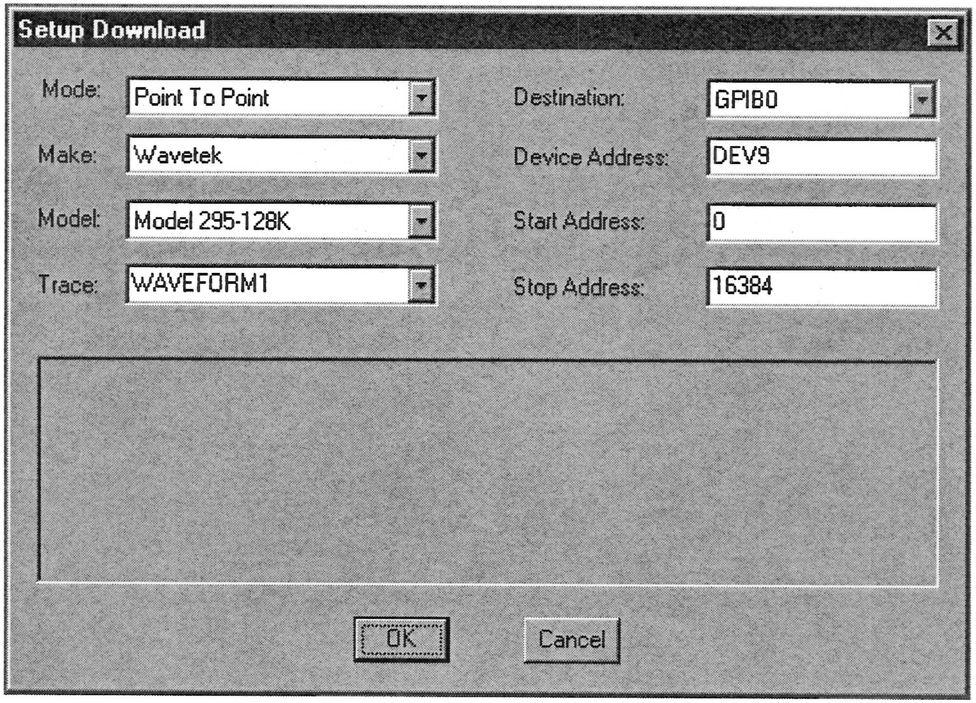
Block Frequency becomes the frequency that is output by the Arb.

## 6.2 Downloading Waveforms

After your instruments are connected, you should be ready to run the WaveForm DSP2 download. Proceed as follows:

1. Select the file containing the waveform you want to download.
2. Select the waveform window containing the waveform you want to download.
3. If you have not already done so, select the **Setup** menu, **Waveforms...** option, and type the Amplitude, Frequency, and Size of the waveform that you want to download and when complete select OK.
4. Select the **Setup** menu, **Download...** option.

A dialog box appears.




5. Proceed to the appropriate follow-on procedure, GPIB, COM, VXI, or File for your specific Arb.

## Downloading a Waveform (GPIB Interface)

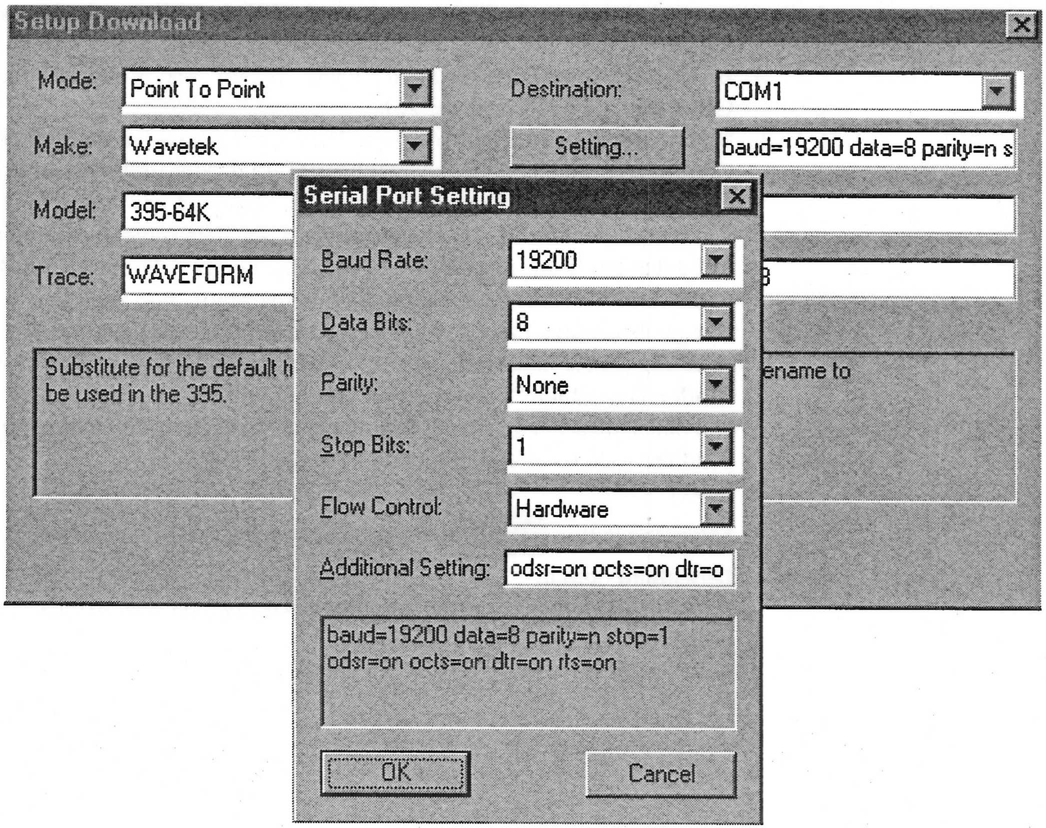
Because terminology differs between Arbs, the terms "trace", "function" and "channel" are used interchangeably in this section.

1. **Make:** Select the manufacturer of the ARB (Wavetek is the default).
2. **Model:** Select the model number of your Arb from displayed pick-list.
3. **Trace (or Function, or channel):** Click on down-arrow button, and select from displayed pick-list. For ARBs with multi-channel capability the list will contain the possible channels (indicated by CHANNEL1— CHANNEL4) which should be selected first and then this should be over-typed with the trace name required. Failure to overtype will result in a trace name of "CHANNEL1". For single channel ARBs the default selection is the trace name WAVEFORM which will be used if the overtyping is omitted.
4. **Destination:** click on the down-arrow button, and select the interface you are connected to from the list. A single card GPIB is GPIB0 and addresses DEV0-DEV30.
5. **Device Address:** Enter the device name "DEVn" that is assigned the same GPIB primary address as the Arb.
6. **Start Address and Stop Address:** The Start Address has an initial (default) value of zero, and the Stop Address has an initial value of the size of the waveform minus one. You can, instead, start your waveform at any address in the Arb waveform memory. If you change either to another value, the number of points in the waveform file may no longer equal the number of points being downloaded to the Arb. You will then want to look at the Mode box and see if you want to change its setting. You can load larger waveforms into the same Trace name by selecting appropriate start and stop addresses for each waveform segment.
7. **Mode:** The default "Point to Point" setting will truncate files longer than size, and pad shorter files with 0Vdc at the end. "Stretch to Fit" will compress or expand the waveform by interpolation to satisfy size.
8. Select "OK".
9. Select the button  from the toolbar or from the **I/O** menu select the **Exec Download** option.
   *If a downloading error occurs, a dialog box appears with details of the error.*

## Downloading a Waveform (Serial Interface)

Because terminology differs between Arbs, the terms "trace", "function" and "channel" are used interchangeably in this section.

1. **Make:** Select the manufacturer of the ARB (Wavetek is the default).
2. **Model:** Select the model number of your Arb from displayed pick-list.
3. **Trace (or Function, or channel):** Click on down-arrow button, and select from displayed pick-list. For ARBs with multi-channel capability the list will contain the possible channels (indicated by CHANNEL1 — CHANNEL4) which should be selected first and then this should be overtyped with the trace name required. Failure to overtype will result in a trace name of "CHANNEL1". For single channel ARBs the default selection is the trace name WAVEFORM which will be used if the overtyping is omitted.
4. **Destination:** Click on the down-arrow button, and select the interface you are connected to from the list. Select COM 1 through 4 to control an Arb via the PC's serial port.
   *The button **Setting** appears allowing the setup of the serial interface parameters.*



5. **Setting:** Enter the Serial Port settings consistent with the target ARB.
*Flow Control: can be either None, Xon/Xoff, or Hardware. When Hardware selected the Additional Settings is active allowing selections of the following parameters:-*

odsr=on|off
Specifies whether output handshaking that uses the Data Set Ready (DSR) circuit is on or off.

octs=on|off
Specifies whether output handshaking that uses the Clear To Send (CTS) circuit is on or off.

dtr=on|off
Specifes whether the DTR circuit is on or off.

rts=on|off|hs|tg
Specifies whether the RTS circuit is set to on, off, handshake, or toggle.

idsr=on|off
Specifies whether the DSR circuit sensitivity is on or off.

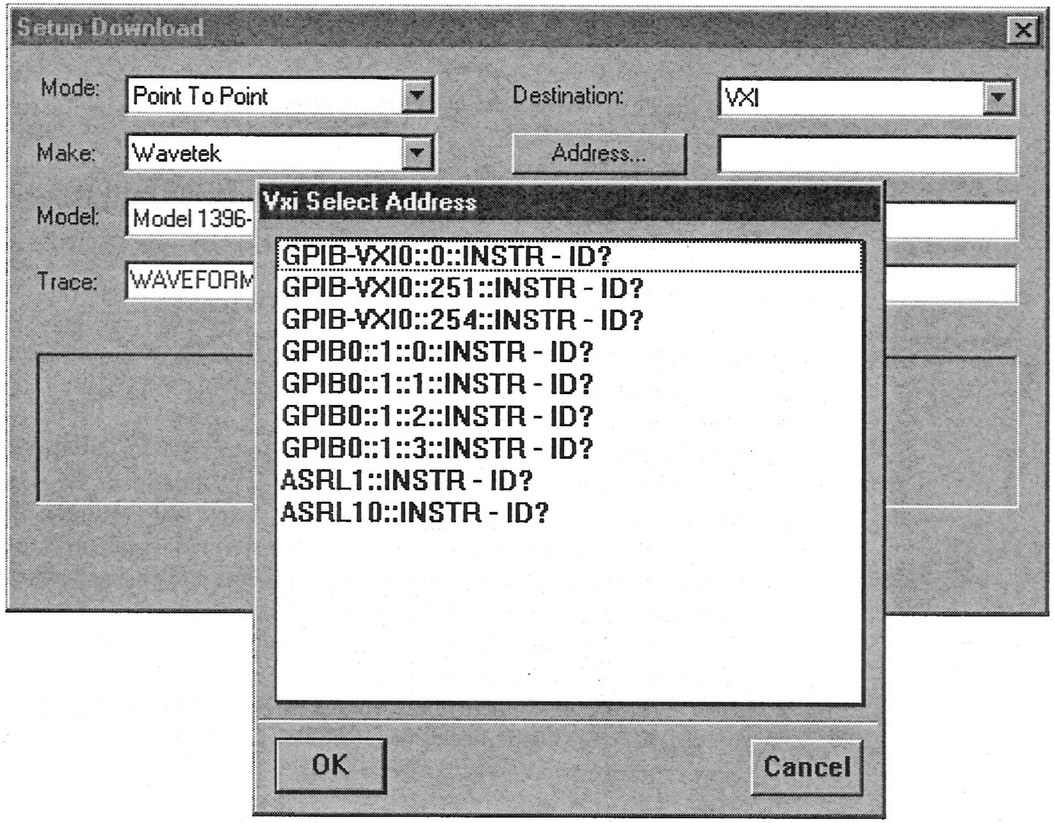
For the Wavetek Models 295, 296, 395 you should use the following settings if Hardware handshaking is enabled: odsr=on octs=on dtr=on rts=on

1. **Start Address and Stop Address:** The Start Address has an initial (default) value of zero, and the Stop Address has an initial value of the size of the waveform minus one.You can, instead, start your waveform at any address in the Arb waveform memory. If you change either to another value, the number of points in the waveform file may no longer equal the number of points being downloaded to the Arb. You will then want to look at the Mode box and see if you want to change its setting. You can load larger waveforms into the same Trace name by selecting appropriate start and stop addressesfor each waveform segment.
2. **Mode:** The default "Point to Point" setting will truncate files longer than size, and pad shorter files with 0Vdc at the end. "Stretch to Fit" will compress or expand the waveform by interpolation to satisfy size.
3. Select "OK".
4. Select the button  from the toolbar or from the **I/O** menu select the **Exec Down-load** option.
*If a downloading error occurs, a dialog box appears with details of the error.*

## Downloading a Waveform (VXI Interface)

The Models 1375, 1395, 1385, 1396 are designed to operate in a VXI rack, because of this, there are differences in the way the download is set up.

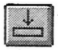
1. **Make:** Select the manufacturer of the ARB (Wavetek is the default).
2. **Model:** Select the model number of your Arb from displayed pick-list.
3. **Trace:** The default trace will be WAVEFORM and this should be overtyped with the trace name required.
4. **Destination:** Click on the down-arrow button, and select the VXI from the displayed pick-list. If no VXI is displayed then WaveForm DSP2 has not detected the existence of the VISA dll on initialization.
   *The button* ***Address*** *appears allowing the selection of one of the addresses detected by the VXI resource manager.*



5. **Address:** Select from the pick list the address of the target ARB. Note that the display of the address is dependent on the response from the VXI resource manager and will vary.

*In the above display GPIB-VXIO::0::INSTR - ID? is the slot0 resource manager with a second address representation of GPIB0::1::0::INSTR - ID? An ARB has the address GPIB-VXIO::251::INSTR - ID? wth another address identification of GPIB0::1::1::INSTR - ID? The ASLR1 is the detected serial port and the ASLR10 is the detected parallel port.*

6 **Start Address and Stop Address:** The Start Address has an initial (default) value of zero, and the Stop Address has an initial value of the size of the waveform minus one. You can, instead, start your waveform at any address in the Arb waveform memory. If you change either to another value, the number of points in the waveform file may no longer equal the number of points being downloaded to the Arb. You will then want to look at the Mode box and see if you want to change its setting. You can load larger waveforms into the same Trace name by selecting appropriate start and stop addresses for each waveform segment.

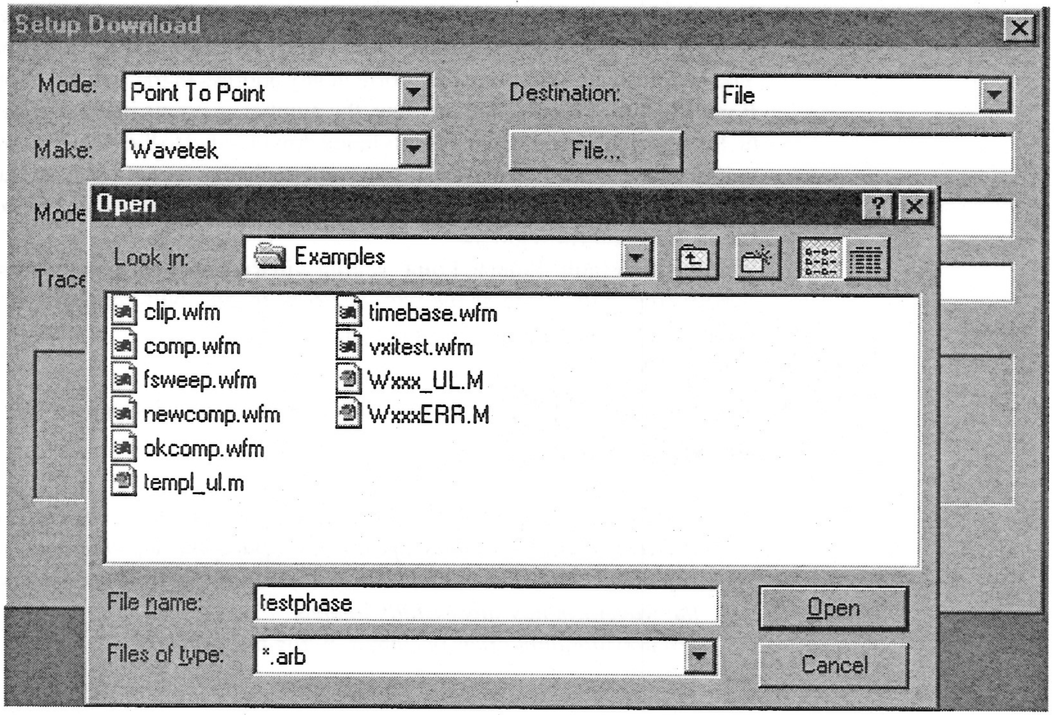
7. **Mode:** The default "Point to Point" setting will truncate files longer than size, and pad shorter files with 0Vdc at the end. "Stretch to Fit" will compress or expand the waveform by interpolation to satisfy size.
8. Select "OK".
9. Select the button  from the toolbar or from the **I/O** menu select the **Exec Download** option.

*If a downloading error occurs, a dialog box appears with details of the error. If you are using an oscilloscope on the output of the Arb, then a few seconds after you select "Exec Download" you will see an output from the Arb, indicating that the waveform has successfully loaded.*
*If a downloading error occurs, a dialog box appears.*

## Downloading a Waveform (File)

When File is selected as the destination in the download dialog box, a button is provided to allow access to a familiar Windows dialog box which allows definition of the filename (use the *.arb extension) and selection of the destination drive. After this dialog is closed, the Device Address block changes to a file name block, and the path and file name you selected for your file appear in the block Selecting File causes the download to go to a file rather than through the communications interface to your Arb.

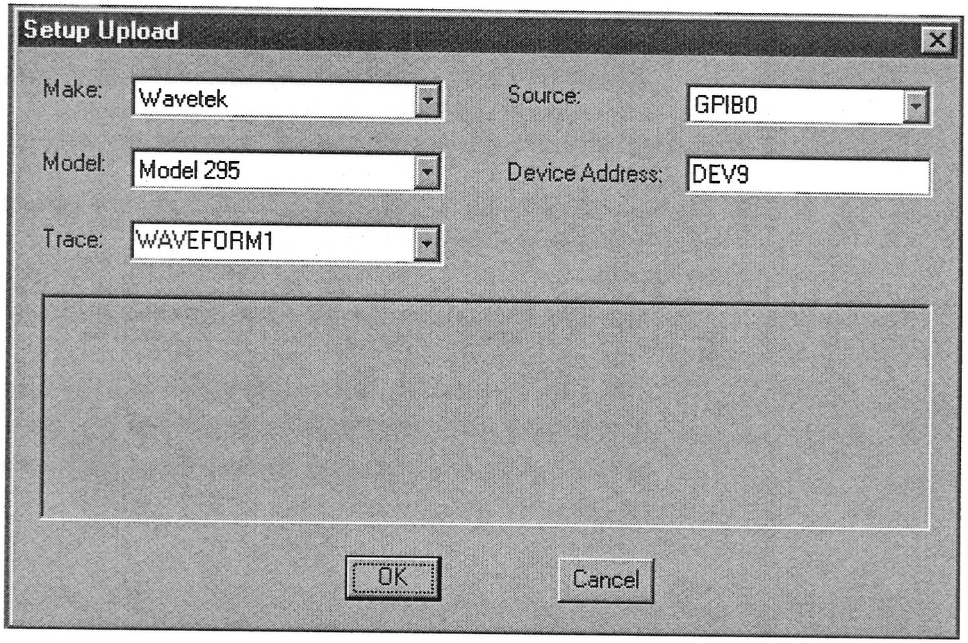
The file format used is specific for Models 295 & 296 and can also be used by the VXI drivers for Models 1375, 1385, 1395, & 1396. For other models this transfer option is either not available or is flagged as an error.



## 6.3 Uploading Waveforms

After your instruments are connected, you should be ready to run the WaveForm DSP2 download. Proceed as follows:

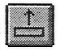
1. Select or create a waveform window to receive the waveform you are about to upload.
2. Select the **Setup** menu, **Upload…** option.
   *A dialog box appears.*




Proceed to the appropriate follow-on procedure, GPIB, COM, VXI, or File for the source instrument.

## Uploading a Waveform (GPIB Interface)

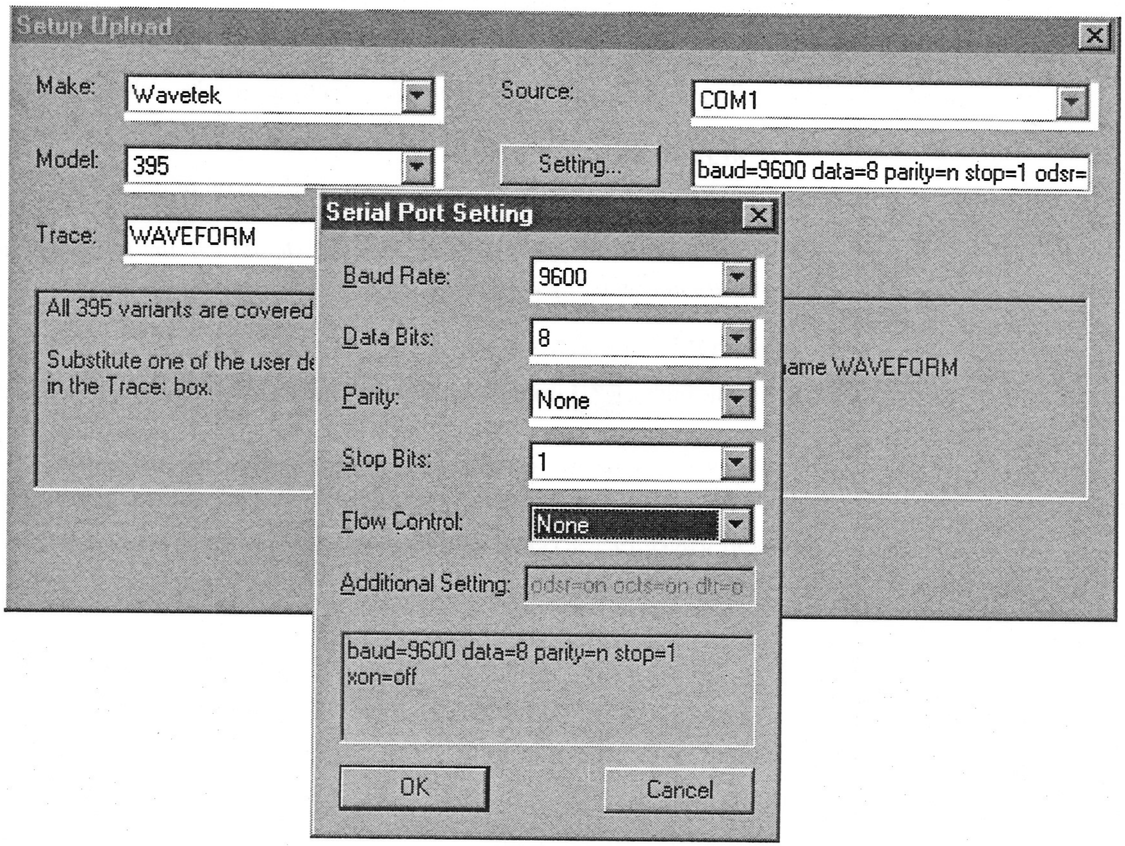
1. **Make:** Click on the down-arrow button, and select the manufacturers name of your source from the displayed pick-list.
2. **Model:** Click on down-arrow button, and select the model number of your source from the displayed pick-list.
3. **Trace (Function or Channel):** Click on down-arrow button and select the desired function from the displayed pick-list. The selection available is dependent on the facilities available in the DSO or ARB.

   *For Wavetek multi-channel ARBs you must first select the channel number from the pick list and then overtype this with the name of the trace to be uploaded. For Wavetek single channel ARBs (except Model 29) you must overtype the default trace name WAVEFORM with the desired trace name.*
4. **Source:** Click on the down-arrow button, and select the source device from the displayed pick-list.
5. **Device Address:** Enter the device name "DEVn" that is assigned the same GPIB primary address as the source instrument.
6. Select "OK".
7. Select the button  from the toolbar or from the **I/O** menu select the **Exec Upload** option.

   *After a few seconds the waveform will appear on the computer screen.*
   *If you have not selected an empty waveform for this upload and have overwritten an existing waveform you can us the undo button*  *to retrieve the original waveform.*

## Uploading a Waveform (Serial Interface)

1. **Make:** Click on the down-arrow button, and select the manufacturers name of your source from the displayed pick-list.
2. **Model:** Click on down-arrow button, and select the model number of your source from the displayed pick-list.
3. **Trace (Function or Channel):** Click on down-arrow button, and select the desired function from the displayed pick-list. The selection available is dependent on the facilities available in the DSO or ARB.
   *For Wavetek multi-channel ARBs you must first select the channel number from the pick list and then overtype this with the name of the trace to be uploaded. For Wavetek single channel ARBs (except Model 29) you must overtype the default trace name WAVEFORM with the desired trace name.*
4. **Source:** Click on the down-arrow button, and select the interface you are connected to from the list. Select COM 1 through 4 to control an Arb via the PC's serial port.
   *The button **Setting** appears allowing the setup of the serial interface parameters.*



5. **Setting:** Enter the Serial Port settings consistant with the target ARB.
*Flow Control: can be either None, Xon/Xoff, or Hardware. When Hardware selected the Additinal Settings is active allowing selections of the following parameters:-*

odsr=on|off
Specifies whether output handshaking that uses the Data Set Ready (DSR) circuit is on or off.

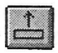
octs=on|off
Specifies whether output handshaking that uses the Clear To Send (CTS) circuit is on or off.

dtr=on|off
Specifes whether the DTR circuit is on or off.

rts=on|off|hs|tg
Specifies whether the RTS circuit is set to on, off, handshake, or toggle.

idsr=on|off
Specifies whether the DSR circuit sensitivity is on or off.

*For the Wavetek Models 295, 296, 395 you should use the following settings if Hardware handshaking is enabled: odsr=on octs=on dtr=on rts=on*

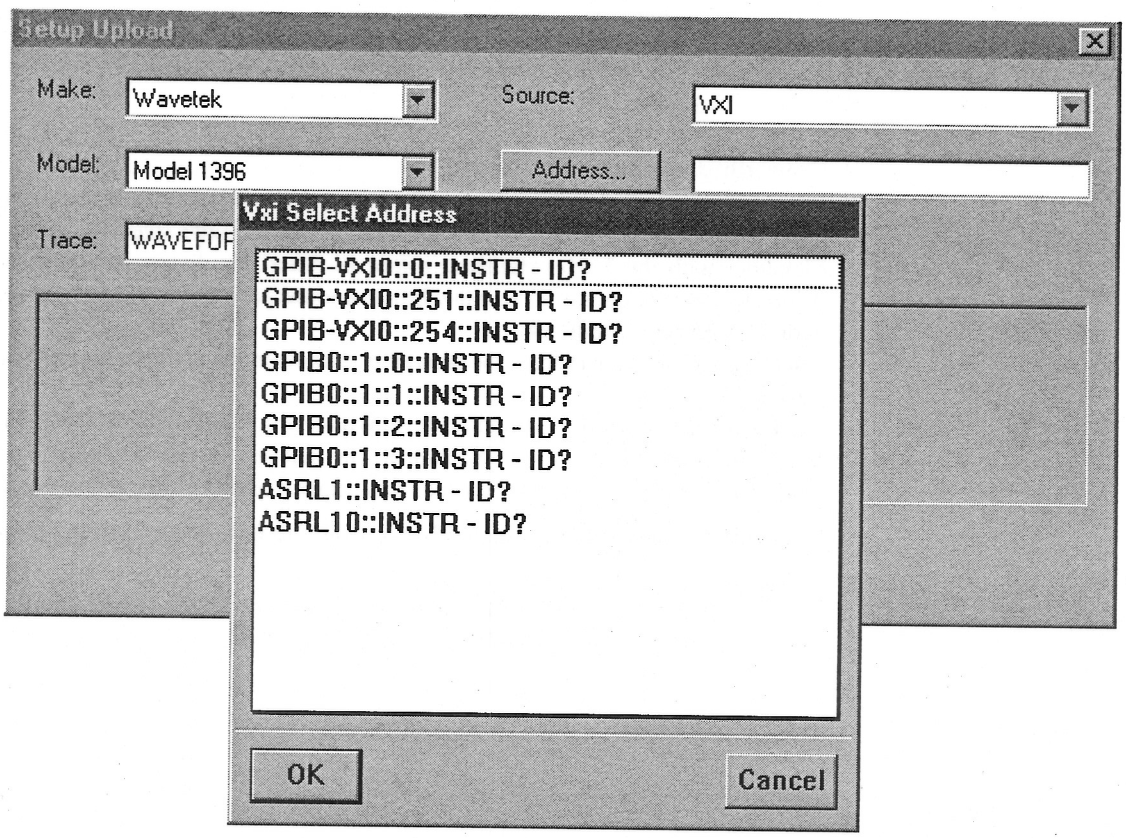
6. Select "OK".
7. Select the button  from the toolbar or from the **I/O** menu select the **Exec Upload** option.
*After a few seconds the waveform will appear on the computer screen.*
*If you have not selected an empty wave form for this upload and have overwritten an existing waveform you can us the undo button*  *to retrieve the original waveform.*

## Uploading a Waveform (VXI Interface)

1. **Make:** Click on the down-arrow button, and select the manufacturers name of your source from the displayed pick-list.
2. **Model:** Click on down-arrow button, and select the model number of your source from the displayed pick-list.
3. **Trace (Function or Channel):** Click on down-arrow button, and select the desired function from the displayed pick-list. . The selection available is dependent on the facilities available in the DSO or ARB.

   *For Wavetek single channel ARBs (except Model 29) you must overtype the default trace name WAVEFORM with the desired trace name.*
4. **Source:** Click on the down-arrow button, and select the VXI from the displayed pick-list. If no VXI is displayed then WaveForm DSP2 has not detected the existence of the VISA dll on initialization.

   *The button **Address** appears allowing the selection of one of the addresses detected by the VXI resource manager.*



5. **Address:** Select from the pick list the address of the target ARB. Note that the display of the address is dependent on the response from the VXI resource manager and will vary.

   *In the above display GPIB-VXIO::0::INSTR - ID? is the slot0 resource manager with a second address representation of GPIB0::1::0::INSTR - ID? An ARB has the address GPIB-VXIO::251::INSTR - ID? wth another address identification of GPIB0::1::1::INSTR - ID? The ASLR1 is the detected serial port and the ASLR10 is the detected parallel port.*
6. Select "OK".
7. Select the button from the toolbar or from the **I/O** menu select the **Exec Upload** option.

   *After a few seconds the waveform will appear on the computer screen.*
   *If you have not selected an empty waveform for this upload and have overwritten an existing waveform you can us the undo button to retrieve the original waveform.*

### Uploading a Waveform (from File)

When File is selected as the source in the Set-up UPLoad menu, a button is provided to allow access to a familiar Windows dialog box which allows selection of the drive and filename (with the *.arb extension) required. After this dialog is closed, the Device Address block changes to a file name block, and the path and file name you selected for your file appear in the block.
Selecting File loads data from the file rather than from the communications interface to your Arb.
The file format is specific to transfers from WaveForm DSP2 to the Wavetek Models 295 & 296 via floppy disk and to VXI Plug & Play Drivers for the Wavetek Models 1375, 1385, 1395, & 1396. For other sources this transfer option is flagged as an error.

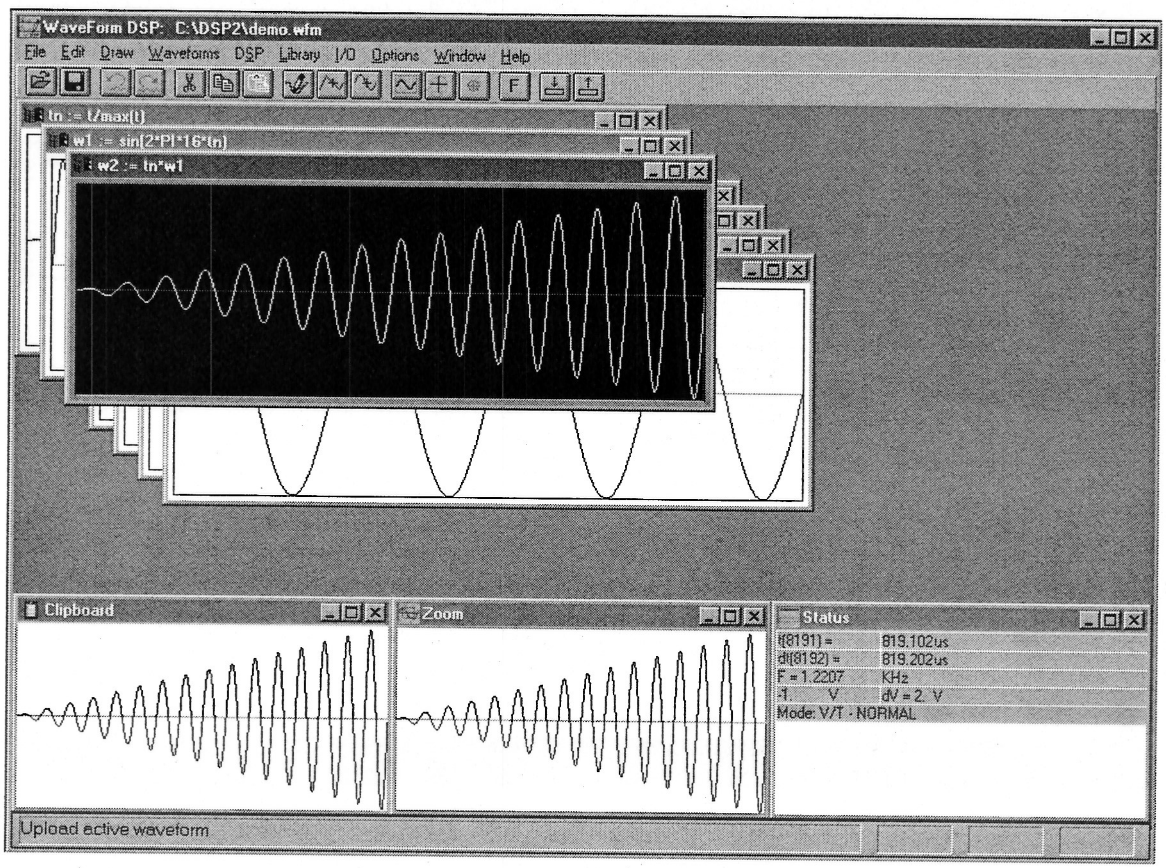
Select the button from the toolbar or from the **I/O** menu select the **Exec Upload** option.

*After a few seconds the waveform will appear on the computer screen.*
*If you have not selected an empty waveform for this upload and have overwritten an existing waveform you can us the undo button to retrieve the original waveform.*

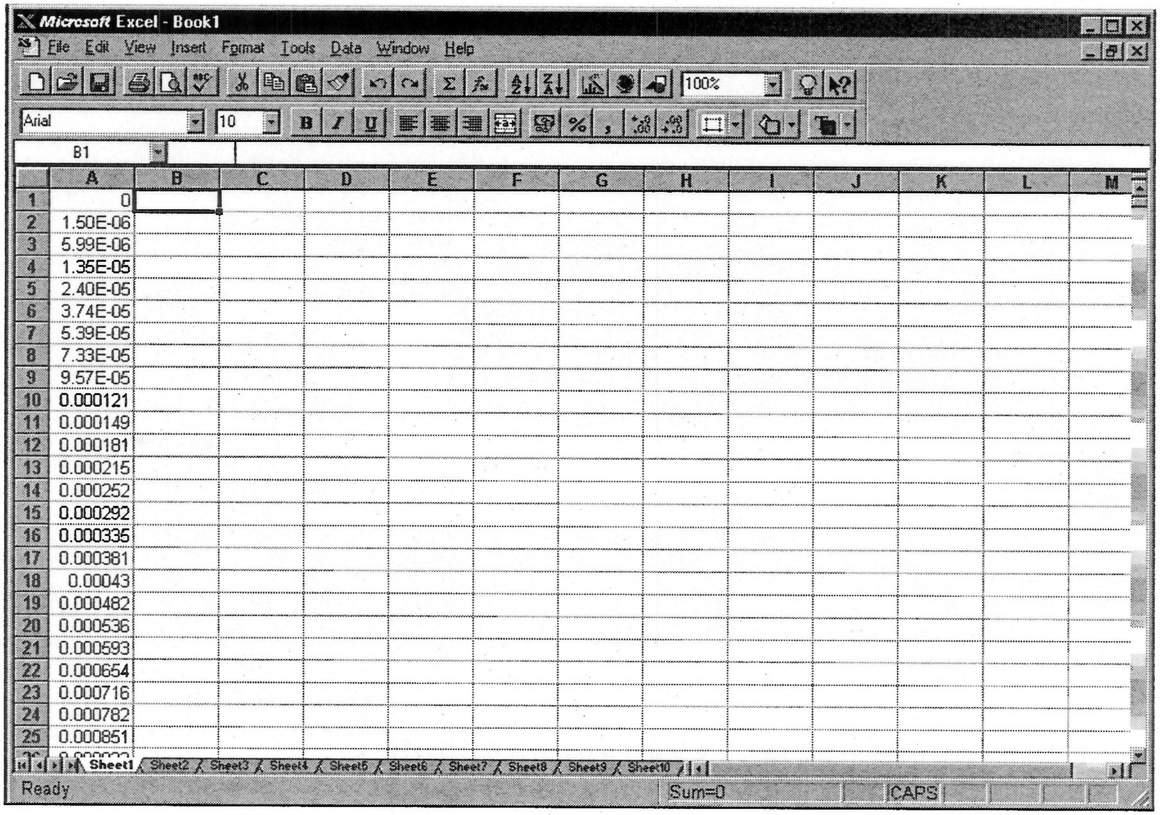
# 7 Working with Excel

When transferring a WaveForm DSP2 waveform into Microsoft Excel through the Windows clipboard, remember that the file as it is stored in the Windows clipboard is in ASCII format. Therefore, the file must be loaded into a spreadsheet. It will NOT paste to a graphics window directly as with "PICTURE" commands. Use the proper Excel commands as explained in the Excel manuals.

1. In WaveForm DSP2, Activate the waveform you want to Transfer.
2. Select the Edit menu Select All Option.
   *The window background turns black.*



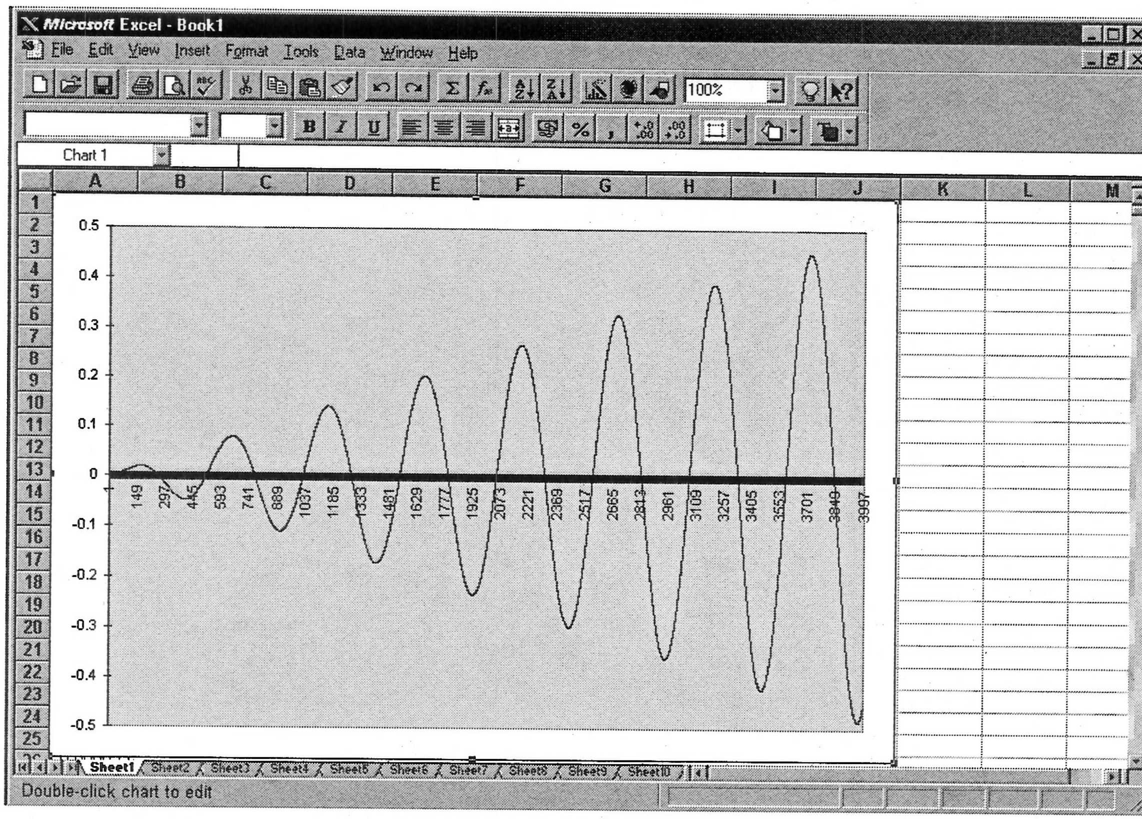
3. Select the Edit menu Copy option. Your waveform now appears in WaveForm DSP2's internal clipboard window in graphics mode. At the same time, the ASCII data is transferred to the external Windows clipboard.
4. Select the WaveForm DSP2 window Minimize button. You will probably see a portion of the Windows clipboard showing some of the ASCII code.
5. Click on the Excel icon to open the Excel application, if it is not already open.
6. In Excel, Open a fresh spreadsheet using the File menu. Remember, you are pasting NUMBERS, not graphics.
7. Use the Excel Edit menu, Paste option, to paste the data from the Windows clipboard to the spreadsheet. The waveform data should paste into the spreadsheet similar to the following figure.
   *Note that the cell values should fall in the range -1.0 to +1.0*




8. Follow Excel instructions to manipulate or display the data in the desired form.

9. After you have your spreadsheet, you may want to chart it in Excel. One reason for using an Excel Chart would be to use Excel to print your waveform. This is one way to have hard copy (in graphics mode) of the waveforms you have created in WaveForm DSP2. If you have made changes to the waveform data while in the Excel spreadsheet, Select All and Copy the data to the clipboard. If you have made no changes, the data is already in the Windows Clipboard.

10. To create an Excel Chart, select the Excel Chart Wizard button  from the Excel toolbar and follow the instructions that are presented.
*It should appear like the following figure.*




11. Follow Excel instructions to manipulate or display the chart in the desired form. When your waveform is displayed in graphics mode to your satisfaction, you may Print it from Excel.

12. Let us assume that you have made some changes to the waveform data while in the Excel spreadsheet and want to get those changes back into WaveForm DSP2. First, get back into the spreadsheet window (if you are in the Chart window) by selecting your spreadsheet from Excel's Window menu.
13. The data in this spreadsheet should either be normalized to be with the range of +/-1.0, or you have already selected the normalization function from within WaveForm DSP2's Set-up Waveform dialog.
14. Select the cells which you want to copy. Use Excel's Edit menu, Copy option to copy the cell data back into the external Windows Clipboard.
15. Minimize Excel and Maximize WaveForm DSP2. Select the waveform window which you will be loading the waveform data (or select Waveforms menu, New option).
16. Use the Edit menu, Paste option, to paste the data into your waveform window.

## WAVETEK TEST & MEASUREMENT DIVISION — SALES AND SERVICE REPRESENTATIVES WORLDWIDE

**ARGENTINA**
**Reycom Electronica SA**
B de Irigoyen 972 2DO Piso C,
1304 Buenos Aires
*Tel:* 5 41 300 2013 *Fax:* 5 41 300 2010

**AUSTRALIA**
**Scientific Devices Australia Pty. Ltd**
118 Atkinson Street, Oakleigh,
PO Box 163, Victoria 3166
*Tel:* 61 3 9569 1366 *Fax:* 61 3 9563 4728

**AUSTRIA**
**Wavetek GmbH**
Pharos Haus,
Nordbahnstrasse 36/Top 1.4,
A-1020 Vienna, Austria
*Tel:* 43 1 214 51 10 *Fax:* 43 1 214 51 09

**BANGLADESH**
**Sisco International Ltd.**
13/2 Toyenbee Circular Road,
Mitijheel Commercial Area,
GPO Box 2545
*Tel:* 880 2 241 491 *Fax:* 880 2 833 041

**BAHRAIN**
**Intercol**
Science & Industry Division,
PO Box 584,
Manama 301
*Tel:* 973 727 177 *Fax:* 973 727 228

**BENELUX**
**Air Parts International BV**
PO Box 255, Kalkovenweg 12,
2400 AG Alphen aan den Rijn
*Tel:* 31 172 422 455 *Fax:* 31 172 421 022

**BOLIVIA**
**Electronic Marconi SRL**
Calle Yanacocha 337,
Cajon Postal 143, La Paz
*Tel:* 591 2 352 574 *Fax:* 591 2 314 540

**BRAZIL**
**Sistronics Instrumentacao**
**E Sistemas LTDA**
Av. Adolfo Pinheiro 1000-9.o Andar,
CEP 04734-002, Sao Paulo - SP
*Tel:* 55 11 247 5588 *Fax:* 55 11 523 8457

**CHILE**
**Avantec Ltda**
PO Box 1087,
Fidel Oteiza 1921 - Of. 1106,
Providencia - Santiago
*Tel:* 56 2 341 1021 *Fax:* 56 2 341 1020

**CHINA**
**Wavetek Corporation**
Room 2701, Citic Building,
No. 19 Jianguomenwai Dajie,
Beijing 100004, PRC
*Tel:* 86 10 6592 8044 *Fax:* 86 10 6500 8199

**DENMARK**
**Teleinstrument A/S**
Christiansholmsgade 16,
DK-8700 Horsens
*Tel:* 45 76 251 818 *Fax:* 45 75 615 658

**ECUADOR**
**Dimatronic SA**
Junin 430 Y Cordova,
Guayaquil
*Tel:* 593 4 314 950 *Fax:* 593 4 563 664

**EASTERN EUROPE**
**Wavetek GmbH**
Pharos Haus,
Nordbahnstrasse 36/Top 1.4,
A-1020 Vienna, Austria
*Tel:* 43 1 214 51 10 *Fax:* 43 1 214 51 09

**EGYPT & MIDDLE EAST**
**Computek Integrated Systems**
Test & Measurement Instruments Div.,
5 Messaha Square, Dokki,
Giza, Cairo.
*Tel:* 202 338 4831 *Fax:* 202 349 2680

**FINLAND**
**Teleinstrument Nordic Group**
Div. Tecono,
Sinikalliontie 11, Box 51,
FIN-02631 Espoo
*Tel:* 358 9 2532 0000 *Fax:* 358 9 2532 0022

**FRANCE**
**M. B. Electronique**
606 Rue Fourny, Z.I. Centre, B.P. 31,
78533 BUC CEDEX
*Tel:* 33 1 3967 6767 *Fax:* 33 1 3956 5344

**GERMANY**
**Wavetek GmbH**
Gutenbergstraße 2-4,
85737 Ismaning
*Tel:* 49-89-996-410 *Fax:* 49-89-996-41160

**GREECE**
**American Technical Enterprises SA**
PO Box 3156,
Agiou Konstantinou 39,
Athens 10210
*Tel:* 30 1 524 0620 *Fax:* 30 1 524 9995

**GUATEMALA**
**Wandel & Goltermann**
De Centro America y el,
Caribe, PO Box 2761, 01901,
Guatemala
*Tel:* 502 331 8065 *Fax:* 502 331 8682

**HONG KONG**
**Euro Tech (Far East) Ltd**
18/F., Gee Chang Hong Centre,
65 Wong Chuk Hang Road,
Hong Kong
*Tel:* 852 2814 0311 *Fax:* 852 2870 0479

**ICELAND**
**Rafhus HF**
Fiskislod 94,
101 Reykjavik
*Tel:* 354 1 621 616 *Fax:* 354 1 627 366

**INDIA**
**Technical Trade Links**
354, 1st Floor, 2nd Cross, 13th,
'H' Main Road, H.A.L. II Stage,
Bangalore 560 008
*Tel:* 91 80 526 0646 *Fax:* 91 80 529 1285

**INDONESIA**
**Mecomb Tehnik**
Kawasan Komersial Cilandak Ged,
#304 JI. KKO Raya, Cilandak,
Jakarta 12560
*Tel:* 62 21 780 8008 *Fax:* 62 21 780 1040

**IRELAND**
**Euro Electronic Systems**
Unit 1, Palmerstown Business Park,
Palmerstown, Dublin 20
*Tel:* 353 1 623 0732 *Fax:* 353 1 623 0762

**ISRAEL**
**DAN-EL Technologies Ltd.**
PO Box 13144,
Tel-Aviv 61131
*Tel:* 972 3 647 8770 *Fax:* 972 3 647 8771

**ITALY**
**DELO Instruments**
Via Piemonte 14,
20090 Fizzonasco Pieve E, Milano
*Tel:* 39 2 907 22441 *Fax:* 39 2 907 22742

**JAPAN**
**Toyo Corporation**
26-9, Yushima 3 Chome,
Bunkyo-ku, Tokyo 113
*Tel:* 81 3 5688 6800 *Fax:* 81 3 5688 6900

**KENYA**
**GES Engsales Kenya Ltd.**
P O Box 46658,
Waiyaki Way, ABC Place,
Niarobi
*Tel:* 254 2 441 209 *Fax:* 254 2 448 815

**KOREA**
**Myoung Inc.**
Rm 407, Dorim Doosan Bears Tower,
110-4 Kuro 5 Dong, Kuro-Ku,
Seoul
*Tel:* 82 2 856 7942 *Fax:* 82 2 856 7943

**KUWAIT**
**Tareq Company**
PO Box 20506 Safat,
13066 Safat
*Tel:* 965 243 6100 *Fax:* 965 243 7700

**LEBANON**
**Mabek**
PO Box 13-5657,
Beirut
*Tel:* 961 1 603 541 *Fax:* 961 1 603 129

**MALAYSIA**
**CNN Sdn Bhd**
17D, 2nd Floor, Lebubraya Batu,
Lanchang,
11600 Penang
*Tel:* 60 4 657 9584 *Fax:* 60 4 657 0835

**MEXICO**
**Electronica 2000**
Blvd. A. Lopez Mateos 2016,
Col. Tlacopac.,
Del. Villa Alvaro Oregon,
01010 Mexico, D.F.
*Tel:* 525 662 8800 *Fax:* 525 662 5862

**MOROCCO**
**Minhol SA**
64 Rue El Mortada,
Casablanca 02
*Tel:* 212 225 5292 *Fax:* 212 225 4992

**NEW ZEALAND**
**Nilsen Technologies (NZ) Ltd**
Unit 4, Ambury Court,
1 Porters Ave, Eden Terrace,
Auckland
*Tel:* 64 9 309 2464 *Fax:* 64 9 309 2968

**NIGERIA**
**Boladel Doksfad Nigeria Ltd.**
8 Eleyele Street,
Eleyele Lay-Out,
PO Box 407,
Ile-Ife, Osun State
***Tel:*** 234 36 232 077

**NORWAY**
**Teleinstrument A/S**
Slynga 2,
Postboks 3,
N-2001 Lillestrom
***Tel:*** 47 6383 5700 ***Fax:*** 47 6383 5710

**PAKISTAN**
**Mushko Electronics (Pvt) Ltd.**
Oosman Chambers,
Abdullah Haroon Road,
Karachi 74400
***Tel:*** 92 21 524 131 ***Fax:*** 92 21 568 2298

**PERU**
**BMP Ingenieros SA**
Ave Del Parque Sur No 552,
Lima 41
***Tel:*** 51 1 475 5737 ***Fax:*** 51 1 475 7026

**PHILIPPINES**
**Spark Electronics Corp.**
3rd Floor, Harmony Building,
807 S. Laurel Street,
Mandaluyong City
***Tel:*** 63 2 724 062 126 ***Fax:*** 63 2 723 2669

**PORTUGAL**
**Decada Medida SA**
Taguspark,
Edificio Tecnologia III, 54,
2780 Oeiras
***Tel:*** 351 1 421 0111 ***Fax:*** 351 1 421 1213

**SAUDI ARABIA**
**Electronic Equipment Marketing Co.**
Suite 317, 2nd Floor, Al-Khobar Mall,
PO Box 3750,
Riyadh 11481,
***Tel:*** 966 3 859 1888 ext 717
***Fax:*** 966 3 859 1888 ext 617

**SINGAPORE**
**O'Connor's Singapore Pte Ltd**
O'Connor's House,
98, Pasir Panjang Road,
Singapore 118516
***Tel:*** 65 473 7944 ***Fax:*** 65 472 4508

**SOUTH AFRICA**
**Arrow Altech Distribution Pty Ltd**
PO Box 69,
Isando 1600
***Tel:*** 27 11 392 3804 ***Fax:*** 27 11 974 9625

**SPAIN**
**Equipos y Systemas SA**
Avda. De Fuencarral, km 15,700,
Edificio Europea, 2°- 5°,
28108 Alcobendas, Madrid
***Tel:*** 34 1 657 2215 ***Fax:*** 34 1 657 2222

**SWEDEN**
**Teleinstrument Nordic Group**
Div. Kaliber,
PO Box 700, Kung Hans vag 3,
SE-19127 Sollentuna
***Tel:*** 46 8 626 4150 ***Fax:*** 46 8 626 4151

**SWITZERLAND**
**Computer Controls AG**
PO Box 314,
Probusweg 2, CH-8057 Zurich
***Tel:*** 41 1 308 6666 ***Fax:*** 41 1 308 6655

**TAIWAN**
**Quatek Co., Ltd.**
2nd Fl., 32, Lane 583,
Jui-Kwang Road,
Nei-Hu, Taipei
***Tel:*** 886 2 2797 3357 ***Fax:*** 886 2 2797 3957

**THAILAND**
**Schmidt Scientific (Thailand) Ltd.**
Government Housing Bank Bldg.,
Tower 11, 9th Floor, 212 Rama 9 Road,
Huay Kwang, Bangkok 10320
***Tel:*** 66 2 643 1330 ***Fax:*** 66 2 643 1340

**UNITED ARAB EMIRATES**
**Tek Signals**
T&M and Communications Div.,
PO Box 47520, Abu Dhabi
***Tel:*** 971 2 272 701 ***Fax:*** 971 2 272 702

**UNITED KINGDOM**
**Wavetek Test & Measurement Division**
52 Hurricane Way,
Norwich Airport,
Norwich, Norfolk NR6 6JB, England
***Tel:*** 44 1603 256 600 ***Fax:*** 44 1603 483 670

**UNITED STATES OF AMERICA**
**North American**
**Sales & Service**
9145 Balboa Avenue,
San Diego, CA 92123
***Tel:*** (619) 279-2200 ***Fax:*** (619) 450-0325

**VENEZUELA**
**Equilab, CA**
Apartado 76609,
Caracas 1070A
***Tel:*** 58 2 239 3233 ***Fax:*** 58 2 238 0275

**WEST INDIES**
**WISCO International Ltd.**
690 West 28th Street,
Hilea FL 33010, USA
***Tel:*** 305 888 1676 ***Fax:*** 305 887 1624

**FOR CUSTOMERS IN COUNTRIES NOT LISTED, PLEASE CONTACT WAVETEK TEST AND MEASUREMENT DIVISION IN THE UNITED KINGDOM:**

**Wavetek Test & Measurement Division**
52 Hurricane Way,
Norwich Airport,
Norwich, Norfolk NR6 6JB, England
***Tel:*** 44 1603 256 600 ***Fax:*** 44 1603 483 670

**WEB SITE**
www.wavetek.com

**WAVETEK CORPORATION**
**Corporate Office**
11995 El Camino Real,
Ste. 301, San Diego, CA 92130, U.S.A
***Tel:*** 1 619 793 2300 ***Fax:*** 1 619 793 2310