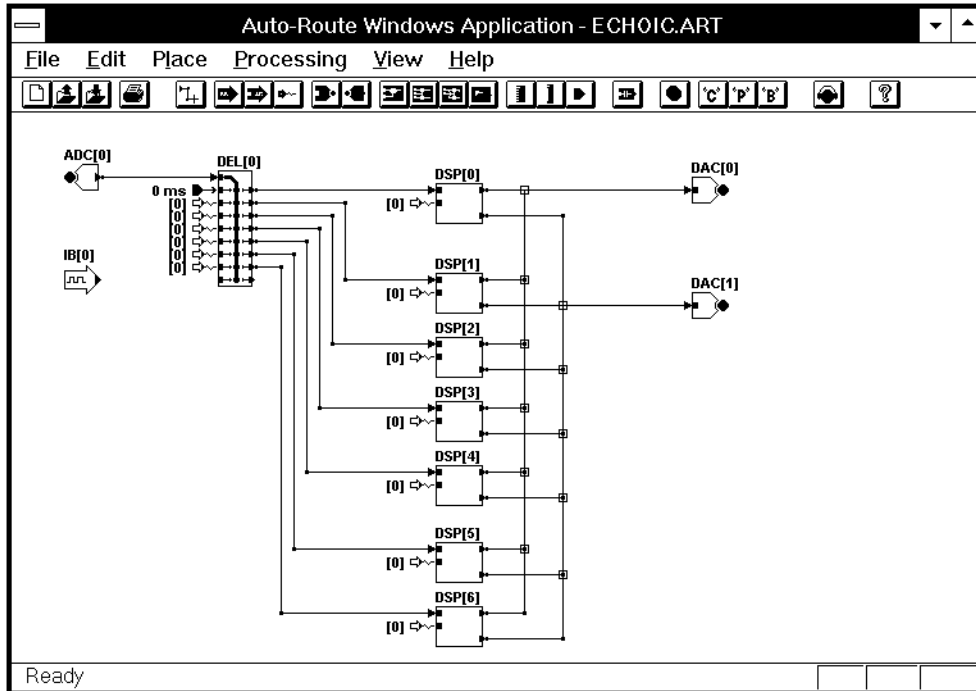


# Part 1

# AutoRoute





*AutoRoute User's Guide – Version 1.0*

**Copyright**

© 1994 TDT. All rights reserved.

No part of this manual may be reproduced or transmitted in any form or by any means, electronic or mechanical, for any purpose without the express written permission of TDT.

**Licenses and Trademarks**

Microsoft, MS-DOS, and Windows are registered trademarks of Microsoft Corporation.





# Contents

## Preface

*Software Philosophy* ix

## Organization of the Manual

*General Information* xi

*Basic Information* xi

*Designing a Routing Schedule* xi

## Chapter 1 Introduction

***What Is AutoRoute?*** 1-1

AutoRoute for Windows 1-1

Hardware Support 1-1

***Who Can Use AutoRoute?*** 1-1

***Routing Schedule Basics*** 1-2

What Is the Real Time Router? 1-2

What Is a Routing Schedule? 1-2

What Is a PD1 Resource? 1-2

What is an AutoRoute Routing Schedule Display? 1-2

***How to Design an AutoRoute Routing Schedule*** -3

***How to Use AutoRoute Routing Schedules*** 1-4

Routing Schedule Files 1-4

Using AutoRoute Files 1-5

Applications 1-5

***Before You Begin*** 1-6

What You Need 1-6

***Installing the Software*** 1-6

Requirements 1-6

Installation 1-7

## **Chapter 2 Learning the Basics**

### ***Getting Started 2-1***

Starting AutoRoute 2-1

### ***Getting to Know the AutoRoute Main Window 2-2***

Using the AutoRoute Menus 2-2

The Toolbar 2-7

### ***Working with AutoRoute Files 2-8***

Creating a New Routing Schedule 2-8

Opening an AutoRoute File 2-9

Saving Routing Schedules 2-10

Deleting Files 2-11

### ***Customizing Your Display 2-12***

Customizing the Display Text Font 2-12

Customizing Colors 2-13

### ***Basic Drawing and Editing 2-15***

Resource and Data Objects 2-15

Routing Wires 2-18

Editing a Display 2-21

## **Chapter 3 Designing a Routing Schedule**

### ***Display Objects 3-1***

Resources 3-1

Routes 3-6

### ***Routing Rules 3-9***

Route Order 3-9

Inbound Data Streams 3-11

### ***Modeling Acoustic Environments 3-11***

Specifying the Routing Schedule 3-12

Dynamic Updating through Inbound Data Streams 3-13

## **Chapter 4 Evaluating a Routing Schedule**

### ***Processing 4-1***

What Happens in Processing? 4-1

Types of Processing 4-1

### ***Implementing 4-2***

## ***Chapter 5*** Using Routing Schedule Files

### ***Binary Files*** 5-1

Binary File Format 5-1

Saving Binary Files 5-1

### ***'C' Files*** 5-2

Saving 'C' Text Files 5-2

### ***Pascal Files*** 5-3

Saving Pascal Text Files 5-3



# Preface

## ***Software Philosophy***

With the release of its first Window-based signal processing tool, SigGen, TDT has demonstrated a commitment to high quality, high level design tools. This commitment has lead to the development of yet another Windows-based design tool, *AutoRoute*. As with SigGen, AutoRoute provides a powerful, yet easy-to-use, graphical signal processing tool. AutoRoute offers a circuit design environment that is above the level of 'C' or Pascal programming. It is a complete circuit design application and an integral part of TDT's PD1 POWER SDAC system.

Many complex circuit models can be generated with ease when using AutoRoute. Because AutoRoute's design incorporates basic Windows principals, users can quickly learn to use AutoRoute to design complex signal processing circuits. In addition to its powerful design features, AutoRoute offers the capability of generating line by line circuit design instructions for use in your own customized applications written in 'C' or Pascal.

As with all software applications, AutoRoute will undoubtedly have some limitations. TDT is sure, however, that users will find AutoRoute an invaluable tool for use in the modeling of acoustic environments.



# Organization of the Manual

The *AutoRoute User's Guide* presents the user with all the information necessary to begin designing circuits using AutoRoute. This document guides you through the process of designing signal processing models, or *routing schedules*, with AutoRoute. This document also serves as a general reference tool. Information provided includes:

- General information about AutoRoute
- Basic information about AutoRoute's features
- A detailed explanation of routing schedule design using AutoRoute

## ***General Information***

General information includes the purpose and uses of AutoRoute, AutoRoute installation, and basic AutoRoute concepts. General information is presented in:

- *Chapter 1* Introduction

## ***Basic Information***

Basic information about AutoRoute's features is described in the following section:

- *Chapter 2* Learning the Basics

## ***Designing a Routing Schedule***

The steps required to design a routing schedule are presented below, along with their associated chapter.

<b>Step</b>	<b>Chapter</b>
Design the routing schedule	<i>Chapter 3</i> Designing a Routing Schedule
Evaluate the routing schedule	<i>Chapter 4</i> Evaluating a Routing Schedule
Save the routing schedule	<i>Chapter 5</i> Using Routing Schedule Files



# Chapter 1 Introduction

Welcome to AutoRoute, TDT's Windows-based signal circuit design application.

## What Is AutoRoute?

AutoRoute was designed to work with TDT's POWER SDAC high performance convolver. AutoRoute is an easy to use, graphical signal processing design tool. By using AutoRoute, you can logically 'wire-up' signal processing circuits, or *routing schedules*, by graphically linking various PD1 *resources*.

## AutoRoute for Windows

AutoRoute for Windows was created for today's preferred PC environment. It combines the advantages of Windows support with the advanced capabilities of TDT's PD1 and other System II hardware. Because Windows provides a graphical user interface, learning new applications is a highly intuitive process. After having mastered one Windows application, the user becomes armed with the basic tools necessary to use any other.

AutoRoute uses standard Windows techniques for saving and loading files, graphically designing circuits, and printing. It does not require you to learn sophisticated commands, unlike many non-Windows programs.

## Hardware Support

AutoRoute supports TDT's System II architecture and was specifically designed to work with TDT's PD1 system.

## Who Can Use AutoRoute?

Anyone with a PC, Windows 3.1, and TDT's PD1 instrumentation can use AutoRoute. Users included:

- Anyone creating signal processing models
- Auditory scientists
- Architectural acousticians
- Speech Scientists
- Musical acousticians

## Routing Schedule Basics

PD1 *resources* may be logically connected through the design of *routing schedules*. These routing schedules are used to provide the *Real Time Router* with the routing instructions necessary for building the desired PD1 circuit.

### What Is the Real Time Router?

The *Real Time Router* (RTR) consists of a single DSP processor and some special addressing hardware. The RTR is the heart of the PD1 system. It is the RTR that handles the routing of data to various PD1 resources.

### What Is a Routing Schedule?

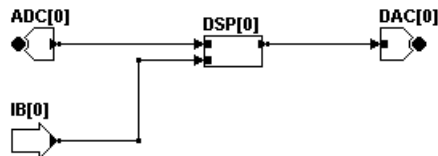
The Real Time Router may be instructed to logically 'wire-up' circuits between PD1 resources. These circuits are specified within the *routing schedule*. Routing schedules may be graphically designed through AutoRoute.

### What Is a PD1 Resource?

The PD1 module contains a number of signal processing elements called *resources*. PD1 resources include D/A converters, A/D converters, convolving DSPs, delay processors, and inbound and outbound data streams. These resources can be logically connected through a routing schedule.

### What is an AutoRoute Routing Schedule Display?

An AutoRoute routing schedule display is a graphical representation of a routing schedule. You design the routing schedule by placing icons representing resources, or *resource objects*, in the AutoRoute display area. These resource objects are then linked by drawing *routes* indicating the direction of signal flow.



## How to Design an AutoRoute Routing Schedule

Once you have familiarized yourself with the basic features of your PD1 system, you will be ready to design your signal processing model, or routing schedule. Design of an AutoRoute routing schedule is relatively easy.

If the PD1 system is logically compared to a typical system consisting of separate physical hardware devices, then PD1 resources are analogous to individual hardware devices and *routing wires* are analogous to connections between devices. In AutoRoute, you simply 'wire-up' your routing schedule in much the same way as you would install and connect a system consisting of individual hardware devices.

The process is simple:

1. Place the icons for each desired resource.
2. 'Wire-up' the routing schedule by drawing connections between the resources.
3. Save the graphical routing schedule design as an AutoRoute file (.art file).
4. Save the routing schedule for use with other signal processing applications such as TDT's Sound Stage. (See the next section, "How to Use AutoRoute Routing Schedules.")

### Example: Designing a Simple Routing Schedule

You may want a system that does the following:

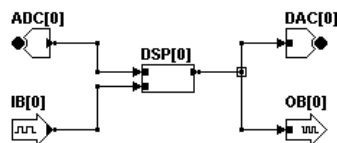
1. Converts an analog signal to a digital signal.
2. Sends that signal to a convolving DSP where it is filtered according to filter coefficients being dynamically updated via an inbound data stream.
3. Outputs the resulting signal to a digital to analog converter and to a file on disk via an outbound data stream.

#### To design the routing schedule

1. Place icons for each resource.



2. 'Wire-up' the routing schedule by drawing connections between the resources.



## ***How to Use AutoRoute Routing Schedules***

AutoRoute provides an easy, graphical means for designing routing schedules. Designing a routing schedule, however, is just the first step in a larger process. The real power of AutoRoute lies in its ability to generate files that can be used by other signal processing applications.

### **Routing Schedule Files**

Routing schedules designed in AutoRoute may be saved in a variety of formats:

- Binary file format
- Text file format

#### **Binary File Format**

Routing schedules designed with AutoRoute may be saved as binary files for use with TDT's Sound Stage acoustic environment design tool or with your own programs written in 'C'. The format of this binary file is discussed in a later section. Binary routing schedule files are saved with the default extension, *rs*.

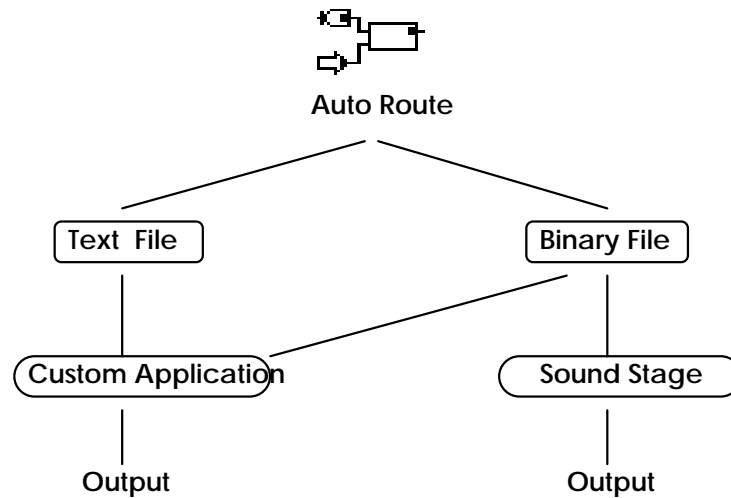
#### **Text File Format**

You may wish to use an AutoRoute routing schedule with custom-designed applications written in 'C' or Pascal. AutoRoute allows you to generate either 'C' or Pascal code containing the line by line instructions needed to program the routing schedule. 'C' text files are saved with the default extension, *.c*. Pascal text files are saved with the default extension, *.pas*.

## Using AutoRoute Files

AutoRoute routing schedule files may be used as follows:

1. Design the routing schedule with AutoRoute.
2. Save the routing schedule to a file.
3. Use the file as input to an application.
4. Process output.



## Applications

AutoRoute may be used by scientists and researchers from a variety of disciplines. Applications include:

- Sound localization research
- Binaural hearing aid research
- 3D sound generation
- Architectural acoustics
- Transducer correction
- DSP simulation and development

## Before You Begin

With a bit of preparation, circuit design with AutoRoute is quick and easy.

### What You Need

See your Microsoft Windows documentation.

See **Error! Reference source not found.**, Chapter 1.

See Chapter 2, *Routing Schedule Basics* or *Programming the PD1 POWER SDAC*.

See *Programming the PD1 POWER SDAC*.

- Windows fundamentals  
You should be comfortable with Windows basics: starting Windows; using the mouse; manipulating windows; opening, closing, and saving files.
- Signal processing  
A basic knowledge of signal processing is necessary.
- Basic AutoRoute concepts  
You should recognize the terms *real time router*, *routing schedule*, and *resource*.
- PD1 Basics  
You should review PD1 basics. In particular, you should be familiar with *routing rules*.

## Installing the Software

### Requirements

In order to run AutoRoute, you must have the following:

- TDT's AP2 Array Processor
- APOS ONBOARD software (latest version available)
- Microsoft Windows 3.1
- Super VGA (1024 x 768) resolution graphics
- TDT's XBUS hardware
- TDT's PD1 system hardware and software

## Installation

### *To install AutoRoute*

1. Make sure your TDT hardware is installed and functioning properly.  
Refer to the *System II Installation Guide*.
2. Update your AP.OBJ file, if necessary.  
Compare the date of your current AP.OBJ to the one found on the AutoRoute diskette. If your version is the older of the two, update it with the file on the AutoRoute diskette.
3. Copy files from the AutoRoute diskette.
  - a. Create the **Router** directory on your hard drive.
  - b. Copy all files from the AutoRoute diskette into the **Router** directory.
4. Add the AutoRoute Icon.
  - a. If necessary, create the **TDT** program group.  
From the File menu of Program Manager, select New... .  
From the New Program Object dialog box, select New Program Group and click OK.  
From the Program Group Properties dialog box, type **TDT** in the Description field and click OK.
  - b. From File Manager, drag the file **router.exe** to the TDT program group displayed in Program Manager.



# ***Chapter 2* Learning the Basics**

## ***Getting Started***

### **Starting AutoRoute**

#### ***To start Windows***

- Type **win** at the DOS prompt.



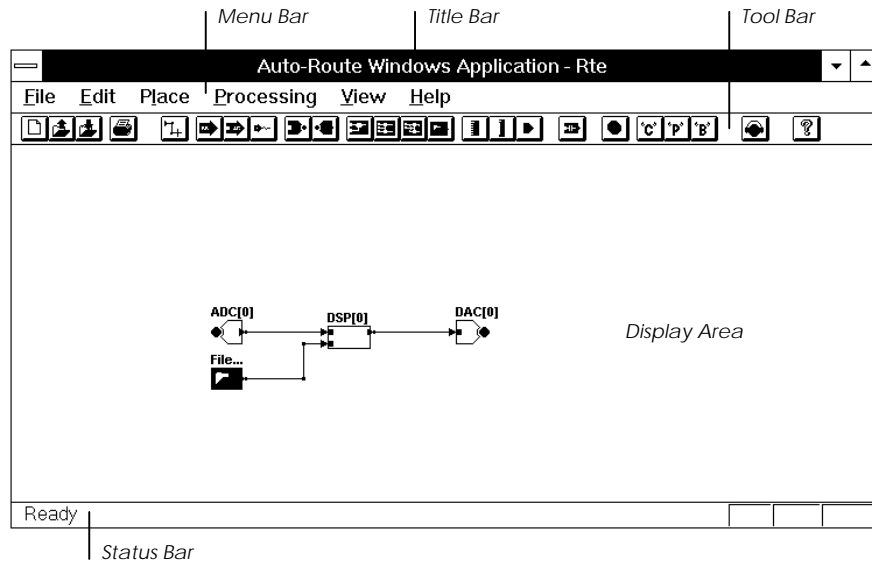
#### ***To start AutoRoute***

- Double-click the AutoRoute icon.  
The AutoRoute main window opens.

You are now ready to begin designing routing schedules with AutoRoute.

## Getting to Know the AutoRoute Main Window

Upon starting AutoRoute, you will be presented with the AutoRoute main window. It is from this window that you will be able to graphically design routing schedules.



The window contains the following sections:

**Title Bar** Displays "AutoRoute" and the name of the current AutoRoute file.

**Menu Bar** Contains a list of menus used for designing and saving routing schedules.

**Toolbar** Provides easy-to-use icons for the most common AutoRoute menu commands.

**Display Area** From within this area, you can graphically design routing schedules.

**Status Bar** Provides the user with information concerning the selected command.

## Using the AutoRoute Menus

AutoRoute provides the user with a full set of menu commands. Using these menu commands, you can create, open, and save routing schedules; build new routing schedules or edit existing routing schedules; print routing schedules; and make choices about your screen display.

The section presents a brief description of the AutoRoute menu commands.

## The File Menu

<b>File</b>	
<b>N</b> ew	Ctrl+N
<b>O</b> pen...	Ctrl+O
<b>S</b> ave	Ctrl+S
Save <b>A</b> s...	
<b>P</b> rint...	Ctrl+P
Print <b>P</b> review	
Print <b>S</b> etup...	
<b>1</b> MULTI.ART	
<b>2</b> TYPES.ART	
<b>3</b> SERIES.ART	
<b>4</b> SIMPLE.ART	
<b>E</b> xit	

The File Menu provides standard Windows commands for creating, saving, opening, closing, and printing files.

**New** Clears the display area. All display information will be lost.

**Open...** Opens and displays an existing routing schedule. All previous display information will be lost.

**Save** and **Save As...** These menu options save the graphical representation of a routing schedule in an AutoRoute file. Files saved in this format use the default extension *.art*.

**Print...** Prints the current routing schedule.

**Print Preview** Allows you to preview the routing schedule as it will be printed.

**Print Setup...** Allows you to select a printer, define the paper size, and choose a page orientation.

## The Edit Menu

Edit	
Undo	Ctrl+U
Select All	Ctrl+A
Delete	Ctrl+D
Move	Ctrl+M
Deselect	Ctrl+E
Setup Display	

The Edit Menu provides commands that allow you to manipulate the routing schedule design. It also provides a command that allows you to customize your routing schedule display.

**Undo** Cancels the most recent editing command and restores the routing schedule to its previous state.

**Select All** Causes all routing schedule objects to be selected.

**Delete** Deletes all selected routing schedule objects.

**Move** Enables you to move all selected objects as a unit.

**Deselect** Deselects all selected objects.

**Setup Display** Opens the Set Display Options dialog box, from which you can specify routing schedule display fonts and colors.

## The Place Menu

<b>Place</b>
<b>Inbound Data Stream</b>
<b>Outbound Data Stream</b>
<b>IB Patch</b>
<b>A/D Converter</b> <b>D/A Converter</b>
<b>Mono Convolver DSP</b> <b>Stereo Convolver DSP</b> <b>Mono In/Stereo Out DSP</b>
<b>8 Tap Delay</b> <b>16 Tap Delay</b>
<b>Routing Wire</b>
<b>Delay Constant</b> <b>Coef File</b> <b>Isolation Register</b>

In AutoRoute, PD1 resources are represented by graphical objects. The Place menu provides commands that allow you to place routing schedule objects in the display area. These objects are described in greater detail in the "Display Objects" section of *Chapter 3*.

**Inbound Data Stream** Places an icon representing an inbound data stream.

**Outbound Data Stream** Places an icon representing an outbound data stream.

**IB Patch** Places an icon representing a patch, or marker, for an inbound data stream.

**A/D Converter** Places an icon representing an A/D Converter.

**D/A Converter** Places an icon representing a D/A Converter.

**Mono Convolver DSP** Places an icon representing a single channel DSP.

**Stereo Convolver DSP** Places an icon representing a dual-channel DSP.

**Mono In/Stereo Out DSP** Places an icon representing a DSP with one input channel and two output channels.

**8 Tap Delay** Places an icon representing a delay with eight taps.

**16 Tap Delay** Places an icon representing a delay with 16 taps.

**Routing Wire** Draws the signal flow path between routing schedule objects.

**Delay Constant** Places an icon representing a specified delay constant.

**Coef File** Places an object representing a filter coefficient file.

**Isolation Register** Places an object representing an isolation register.

## The Processing Menu

<b>P</b> rocessing
<b>P</b> rocess
✓ <b>A</b> uto-Process
<b>I</b> mplement
Generate ' <b>C</b> ' Source
Generate <b>P</b> ascal Source
Generate <b>B</b> inary File

The Processing menu provides commands that control the processing of the routing schedule display, playback testing of the routing schedule, and the generation of routing schedule files.

**Process** Choosing this option causes AutoRoute to analyze the current routing schedule display.

**Auto-Process** Turning this option on causes AutoRoute to dynamically analyze the current routing schedule display.

**Implement** Instructs the PD1 to implement the routing schedule.

**Generate 'C' Source** Generates a text file containing the 'C' source code necessary to specify the routing schedule. Files are saved with the default extension, *.c*.

**Generate Pascal Source** Generates a text file containing the Pascal source code necessary to specify the routing schedule. Files are saved with the default extension, *.pas*.

**Generate Binary File** Generates a binary file specifying the routing schedule. This binary file may be used with TDT's Sound Stage acoustic environment modeling software or with your own programs written in 'C'.

## The View Menu

<b>V</b> iew
✓ <b>T</b> oolbar
✓ <b>S</b> tatus Bar
✓ Show <b>R</b> oute Type
Show <b>N</b> odes

From the View Menu, you can display or hide the Toolbar, Status Bar, or choose between a display of route types or a display of nodes.

**Toolbar** Shows/hides the toolbar.

**Status Bar** Shows/hides the status bar.

**Show Route Type** Causes route types to be displayed in distinctive colors. Choosing Show Route Type turns off Show Nodes.

**Show Nodes** Causes each node to be displayed in a distinctive color. Choosing Show Nodes turns off Show Route Type.

## The Help Menu














Help  
About Router...












From the Help Menu, you can access the About AutoRoute dialog box.

## The Toolbar

You can find quick point-and-click access to the most common commands on the AutoRoute Toolbar.



Click	To
	New
	Open
	Save
	Print
	Draw Route
	IB Stream
	OB Stream
	IB Patch
	DAC
	ADC
	Mono DSP
	Stereo DSP
	Mono to Stereo DSP
	Clear the Display Area
	Open an existing AutoRoute file
	Save an AutoRoute file
	Print the current routing schedule display
	Draw a signal route from one resource to another
	Place an inbound data stream icon
	Place an outbound data stream icon
	Place an inbound data stream patch icon
	Place a D/A converter icon
	Place a A/D converter icon
	Place a single channel DSP icon
	Place a dual channel DSP icon
	Place a single channel in/dual channel out DSP icon

Click		To
	Coef File	Place a coefficient file icon
	8 Tap Delay	Place an icon for an eight tap delay
	16 Tap Delay	Place an icon for a 16 tap delay
	Delay Constant	Place an icon for delay constant
	Isolation Register	Place an isolation register icon
	Process	Cause AutoRoute to analyze the routing schedule display
	'C' Text File	Generate a 'C' text file
	Pascal Text File	Generate a Pascal text file
	Binary File	Generate the binary file used with Sound Stage
	Implement	Tell the PD1 to implement the current routing schedule
	Help	Display the About AutoRoute dialog box

## ***Working with AutoRoute Files***

### **Creating a New Routing Schedule**

Routing schedules are designed in the Display Area of the AutoRoute main window. Upon starting AutoRoute, the display area is ready for routing schedule design. At any time, you may clear the display area for design of a new circuit.

#### ***To clear the Display Area***

- Choose New... from the File menu.

**Note:** When New is chosen, all display information will be lost. Be sure to save any desired routing schedule displays as AutoRoute files prior to choosing New from the File menu.

## Opening an AutoRoute File

The graphical display of an AutoRoute routing schedule may be saved for later use in an AutoRoute file (.art file).

### *To open an existing AutoRoute file*

1. Choose Open... from the File menu.
2. Click on the desired file name.
3. Click the OK button.

**Note:** When a file is opened, all previous display information will be lost. Be sure to save any desired routing schedule displays as AutoRoute files prior to choosing Open from the File menu.

**Note:** The selected AutoRoute file must be in AutoRoute format. These files are typically saved with the extension, .art.

### *To open a recently updated AutoRoute file*

1. Choose the File menu.  
AutoRoute displays the names of the last 4 AutoRoute files saved.
2. Select the desired file name.

## Saving Routing Schedules

Routing schedules may be saved in the following formats:

- **AutoRoute file format**

Files saved in the AutoRoute format contain the graphical information necessary to recreate the AutoRoute routing schedules display. These files may be read by AutoRoute only. AutoRoute files are saved with the default extension, *.art*.
- **Binary file format**

You can save the information necessary to program a routing schedule in a binary file format. This format can be read by TDT's acoustic environment modeling software, Sound Stage, or by your own programs written in 'C'. Binary routing schedule files are saved with the default extension, *.rs*.
- **Text file format**

You may wish to use an AutoRoute routing schedule with custom-designed applications written in 'C' or Pascal. AutoRoute allows you to generate either 'C' or Pascal code containing the line by line instructions needed to program the routing schedule. 'C' text files are saved with the default extension, *.c*. Pascal text files are saved with the default extension, *.pas*.

### Saving Files in AutoRoute Format

#### *To save an AutoRoute display*

- Choose Save or Save As... from the File menu.

The File Save As dialog box will be displayed. You may assign a name to your new routing schedule. It is best to use the default *.art* file extension.

#### *To save an existing AutoRoute file*

- Choose Save from the File menu.

The file will be saved with its current name.

### Saving Files in Binary Format

#### *To save a file in binary format*

- Choose Generate Binary File from the Processing menu.

The Specify Route Schedule File dialog box will be displayed. You may assign a name to your new routing schedule. It is best to use the default *.rs* file extension.

## Saving Files in Text Format

### *To save a 'C' text file*

- Choose Generate 'C' Source from the Processing menu.

The Specify Destination 'C' File dialog box will be displayed. You may assign a name to your new routing schedule. It is best to use the default *.c* file extension.

### *To save a Pascal text file*

- Choose Generate Pascal Source from the Processing menu.

The Specify Destination Pascal File dialog box will be displayed. You may assign a name to your new routing schedule. It is best to use the default *.pas* file extension.

## Deleting Files

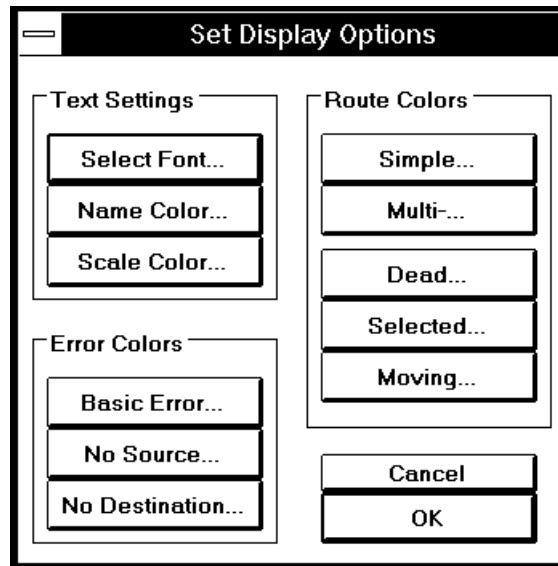
Files may be deleted from your disk by using DOS file commands or Windows File Manager. See the appropriate documentation for more information.

## Customizing Your Display

You may find it useful to alter the default AutoRoute screen configuration. Display settings can be customized from the Set Display Options dialog box.

### *To access the Set Display Options dialog box*

- Choose Setup Display from the Edit menu.



From the Display Setup dialog box you may configure the following:

- Display Text Font
- Colors

## Customizing the Display Text Font

You may customize the following features of the display text: font type, size, and style.

### *To customize the display text font*

- Click the Select Font... button.

The Font dialog box will be displayed. From this dialog box, you may select any available Windows font, font size, and style.

## Customizing Colors

AutoRoute uses color to help differentiate between types of connections, or routes. Color is also used to highlight errors or point out the status of routes. Colors may be divided into two types: error colors and route colors.

### Error Colors

When designing your AutoRoute routing schedule, it is possible to improperly connect resource objects. When Auto Processing is enabled or when you choose to manually process a routing schedule, AutoRoute will mark connections, or *routes*, that are in error through the use of color. There are three basic types of AutoRoute errors:

- **Basic Error**  
A basic error occurs when an inappropriate route is designed.
- **No Source**  
A No Source error occurs when a route has a destination resource object, but no source resource object.
- **No Destination**  
A No Destination error occurs when a route has a source resource object, but no destination resource object.

### Route Colors

Two types of routes may be created: simple routes and multi-routes. These two types of routes may be differentiated by color when Show Route Type is enabled.

- **Simple Route**  
A simple route connects one resource output port to one resource input port.
- **Multi Route**  
In a multi-route connection, multiple output ports are connected to a single input port.

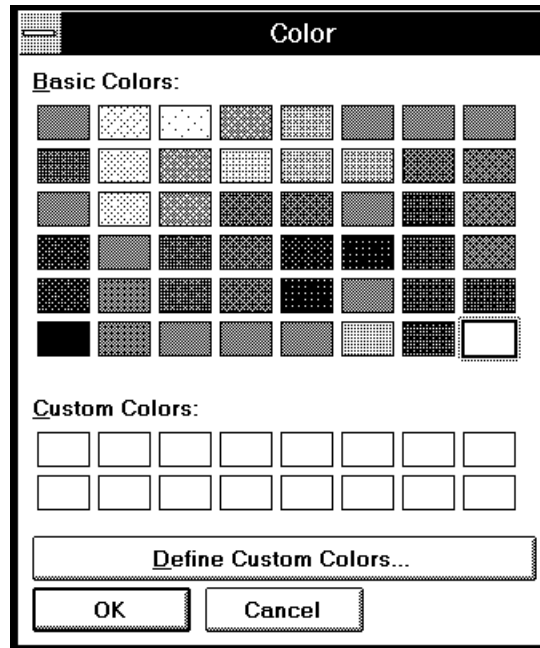
Routes may also exist in several states. These are also marked by color.

- **Dead**  
A Dead route is a non-functioning route.
- **Selected**  
A Selected route is one that has been marked by the user for some type of editing.
- **Moving**  
A route is Moving during a relocation of the route.

### *To access the Color dialog box*

- Click the appropriate Color button of the Display Setup box.

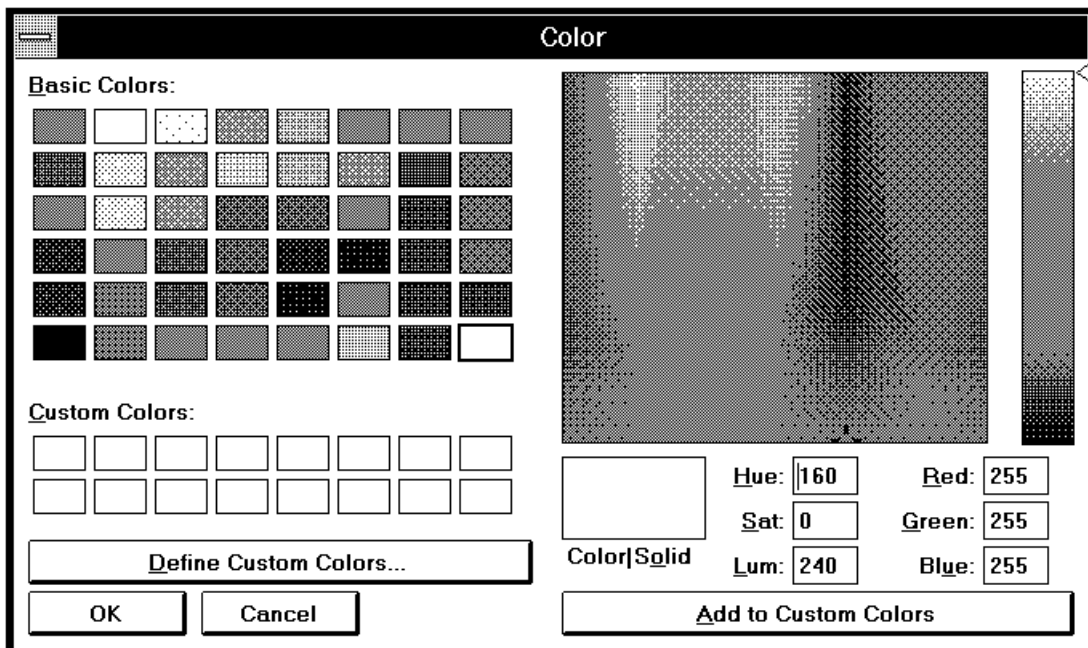
From the Color dialog box, you may select one of the available colors or design your own custom color.



*To design custom colors*

- Click the Define Custom Colors button.

The Color dialog box will expand to provide a color palette from which you can design your custom colors.



## Basic Drawing and Editing

This section describes the basics of drawing and editing a routing schedule. While the basic components of a routing schedule are mentioned, their specific functions are described in a later section.

In AutoRoute, all display elements are known as objects. Objects can be further defined by function into the following:

- Resource objects
- Data objects
- Routing wires

**Resource objects** These objects include: inbound and outbound data streams, D/A and A/D converters, convolving DSPs, delay taps, and isolation registers.

**Data objects** Data objects include: IB patches, coefficient files, and delay constants.

**Routing wires** Routing wires are used to connect all resource objects. Routing wires are also used to connect coefficient files to the coefficient input of a convolving DSP. IB patch objects and delay constant objects are connected directly to an input port of a resource object. IB patch objects and delay constant objects are never connected via a routing wire.

## Resource and Data Objects

AutoRoute resource and data objects are listed under the Place menu. All items in this list qualify as resource or data objects, with the exception of Routing Wire. Resource and data objects are also graphically represented by buttons in the toolbar (see *Chapter 3*, "Getting to Know the AutoRoute Main Window").

### Placing a Resource or Data Object

Resource or data object icons may be placed anywhere within the Display Area.

#### *To place a resource or data object*

1. Select the desired object from the Place menu.

or

Click the appropriate resource object button in the Toolbar.

2. Data objects require the specification of data parameters. If a data object is chosen, a dialog box will appear.
  - a. Specify the necessary parameters.
  - b. Click the OK button to accept.

The resource or data object icon will appear beneath the mouse pointer whenever the pointer is in the Display Area.

3. Drag the resource object icon to the desired position in the Display Area.
4. Click the left mouse button to place the icon.

## Resource and Data Object Icons

Resource object icons may contain one or more input and/or output *ports*. Data objects have no input port and always have one output port. Input and output ports are graphically distinguished by the following:

- Location
- Shape



Type of Port	Location on Icon	Shape
Input	Left	Square
Output	Right	Triangle

Additionally, there are two types of input ports:

- Signal Data Input Port
- Auxiliary Input Port

Auxiliary Input Ports may receive a variety of types of data. For example, a convolving DSP may receive filter coefficients through an auxiliary coefficient input port. On the other hand, a delay may receive a delay constant through an auxiliary port.

Types of ports may also be distinguished by color.



Type of Port	Color
Signal Data Input	Green
Auxiliary Input	Yellow
Output	Red

## Resource Object Names

Resource object names are assigned automatically as resources are created. A resource name consists of a short abbreviation followed by a bracket containing an index number. For any type of resource, the index number is incremented by 1 as new objects are created. These index numbers originate at 0. Thus, if you have placed four A/D converters, they would be named: ADC[0], ADC[1], ADC[2], and ADC[3].

## Editing a Resource or Data Object

You may edit the name of a resource object. You also may edit the parameters of a data object.

### *To edit resource or data objects*

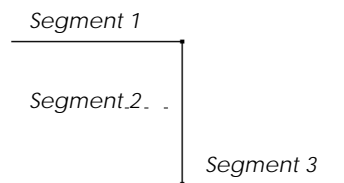
1. Double Click on the resource or data object icon.  
A dialog box displaying the resource object name or data object parameters appears.
2. Edit the desired data.
3. Click the OK button.

## Routing Wires

Signal flow routes between resource objects are specified by drawing *routing wires*.

### Drawing Routing Wires

Routing wires consist of at least one segment. Corners may be included in a routing wire by drawing multiple segments.



**Segment junctures** Segment boundaries are indicated by a small square. These boundaries are known as *segment junctures*.

### *To draw a routing wire*

1. Select Routing Wire from the Place menu.  
You are now in draw mode. The pointer becomes a cross-hair.
2. Place the cross-hair at the desired starting point.
3. Click the *left* mouse button to indicate the beginning of the segment.
4. Move the cross-hair to the desired destination.  
A line will be drawn as you move the mouse pointer.
5. Click the *left* mouse button to indicate the end of the segment.
6. If you are done drawing,
  - Click the *right* mouse button to indicate you are done drawing.  
You are no longer in draw mode. The pointer returns to an arrow.
 If you wish to draw another segment,
  - Repeat steps 4 through 6.

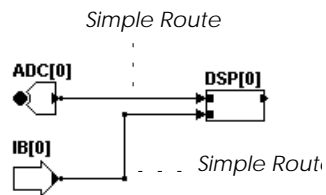
## Types of Routes

Routing wires represent two types of routes:

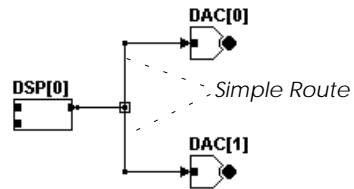
- Simple routes
- Multi-routes

**Simple Routes** A simple route connects one output port to one input port.

The diagram below illustrates two simple route connections between (1) the output port of ADC[0] and the data input port of DSP[0] and (2) the output port of IB[0] and the auxiliary input of DSP[0].

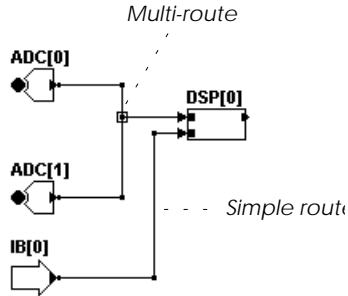


In the following diagram, there are also two simple route connections. The output from DSP[0] is connected via a simple route to the data input of DAC[0]. This output is also connected via a simple route to the data input of DAC[1].



**Multi-routes** A multi-route connects more than one output port to a single input port, in effect summing the input signals.

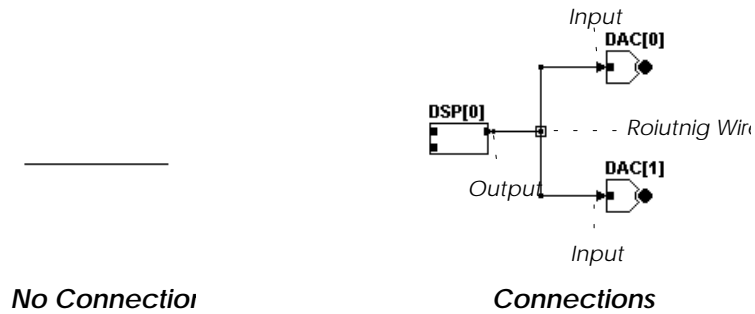
In the diagram below, the outputs from both ADC[0] and ADC[1] are connected to the data input of DSP[0] via a multi-route. A simple route connects the output port of IB[0] and the auxiliary input of DSP[0].



### Routing Wire Connections

When the end of a routing wire is connected to a resource, its appearance differs from that of an unconnected routing wire. You may use this visual difference to ensure that routing wires are connected.

Connection	Appearance
Output ports	Small square
Input ports	Small arrow
Routing wires	Box
None	No symbol



## Editing a Display

At any time you may edit the current routing schedule display. Objects, whether resource objects, data objects, or routing wires, may be moved or deleted. In addition, a routing wire may be edited by resizing the wire or moving the end of the wire.

Some editing functions require that objects be *selected* prior to performing the edit. These functions include: deleting objects, moving routing wires, moving multiple objects.

### Selecting/Deselecting

#### *To select/deselect a routing wire*

1. Place the mouse pointer over an end or segment juncture.
2. Shift-Click the left mouse button.

#### *To select/deselect a specific resource object*

1. Place the mouse pointer over the resource object icon.
2. Shift-click the left mouse button.

#### *To select all objects*

- ▶ Choose Select All from the Edit menu.

or

- ▶ Hit Ctrl-A.

#### *To deselect all selected objects*

- ▶ Choose Deselect from the Edit menu.

or

- ▶ Hit Ctrl-E.

### Moving

In some cases, objects must be selected prior to moving. These cases include: *moving a routing wire* and *moving multiple objects*.

#### *To move one end of a routing wire*

1. Place the mouse pointer over the end.
2. Hold down the left mouse button.
3. Drag to the desired position.
4. Release the left mouse button.

***To move a routing wire segment juncture***

1. Place the mouse pointer over the segment juncture.
2. Hold down the left mouse button.
3. Drag to the desired position.
4. Release the left mouse button.

***To move a resource object***

1. Place the mouse pointer over the resource object icon.
2. Hold down the left mouse button.
3. Drag to the new location.
4. Release the left mouse button.

***To move a routing wire or multiple objects***

1. Select the routing wire or the objects.
2. Follow steps 2 - 6 in "To move selected objects" below.

***To move selected objects***

1. Select all objects to be moved.
2. Choose Move from the Edit menu.

or

Hit Ctrl-M.

A box will appear around the objects to be moved.

3. Place the mouse pointer anywhere within the box.
4. Hold down the left mouse button.
5. Drag the object(s) to the new location.
6. Release the left mouse button.

**Resizing*****To resize a routing wire***

1. Place the mouse pointer over an end.
2. Hold down the left mouse button.
3. Drag until the desired length is achieved.
4. Release the left mouse button.

## **Deleting**

Objects to be deleted must first be selected.

### ***To delete selected objects***

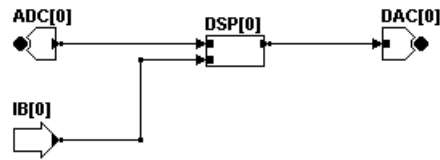
1. Select the object(s) to be deleted.
2. Choose Delete from the Edit menu.

or

Hit Ctrl-D.

# Chapter 3 Designing a Routing Schedule

Designing a routing schedule is fairly simple. Resource objects are placed within the display area and connected via routing wires, forming a signal processing circuit.



## Display Objects

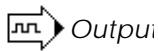
Display objects include resource objects and routing wires.

## Resources

The PD1 contains a number of signal processing elements called resources. These include inbound and outbound data streams, D/A converters, A/D converters, convolving DSPs, delay processors, and isolation registers. These resources are described below.

### Inbound and Outbound Data Streams

High speed data may be sent as input to or output from the PD1 through the use of inbound and outbound data streams.



#### Inbound Data Streams

Inbound data streams may consist of:

- Raw waveform data
- Data records

Four types of data records exist:

- Waveform Records
- Mono coefficient records
- Stereo coefficient records
- Delay time coefficients

See *Programming the PD1 POWER SDAC* for information concerning data record structure.

In AutoRoute, inbound data streams may be connected in the following ways:

- As input data to a convolving DSP
- As filter coefficient data to a convolving DSP
- As delay information to a delay tap



**Outbound Data Stream**

Data may be output from the PD1 through use of an outbound data stream.

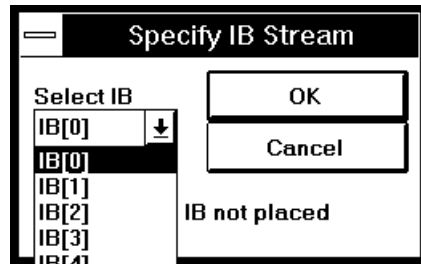


**IB Patch**

*See "Modeling Acoustic Environments" later in this chapter for more information.*

It may be desirable to send inbound data to more than one resource object. An IB patch is simply a visual marker that indicates that data from the specified inbound data stream will be used as input.

It is necessary to specify which IB data stream is associated with an IB patch. This association is indicated within the brackets preceding the IB patch icon.



An IB patch is connected directly to an input port. A routing wire connection is not required.

**A/D and D/A Converters**

The analog interface to the PD1 system consists of A/D and D/A converters as described below.



**A/D Converter**

You may connect as many as four analog to digital converters (ADCs).



**D/A Converter**

You may connect as many as four digital to analog converters (DACs).

## Convolving DSPs

The PD1 can be equipped with up to 28 convolvers. These convolvers are digital signal processors (DSPs) programmed to perform FIR filtering. Each convolver is able to run a 400 tap filter at 44KHz.

Each convolving DSP has one or two data input ports, one filter coefficient port, and one or two output ports.

Filter coefficients may be supplied to the coefficient port through two means:

- Inbound data streams
- Filter files



For more information, see "Using Inbound Data Streams" in [Programming the PD1 POWER SDAC](#).



**Inbound data stream** Filter coefficients may be specified through an inbound data stream by use of either a *mono coefficient record* or a *stereo coefficient record*. This is a special record format that may be used to dynamically update filter coefficients.

Inbound data streams are always connected via a routing wire.

**Filter files** Filter coefficients may be read from two types of files:

- Raw filter files
- HRTF files

**Raw filter files** Raw filter files consist of filter coefficient data saved in integer binary, floating-point binary, or text format. When specifying filter coefficients for two channels, separate filter files must be used for the right and left channels. Raw filter files are saved with the default extension, *.fir*.

**HRTF files** HRTF files contain Head Related Transfer Function (HRTF) information. Each record within the HRTF file consists of a stereo pair of HRTF filters recorded for a particular spatial position. In the currently supported HRTF file format, Type 1 file format, this record simply consists of the left ear filter coefficients followed by the right ear filter coefficients.

From more information, see "HRTF Files" in [Programming the PD1 POWER SDAC](#).

You may wish to specify one filter at a particular spatial position.

***To specify a filter at a specific spatial position***

1. Enter the azimuth in degrees in the Az field.
2. Enter the elevation in degrees in the El field.
3. Enter the distance in the Dist field.

The screenshot shows a dialog box titled "Coefficient File". It contains several sections for configuring filter parameters:

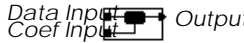
- File Type:** A sub-section containing "Raw Filters" with radio buttons for "Integer" (selected), "Float", and "Ascii"; and "HRTF Space File" with radio buttons for "Stereo", "Left", and "Right".
- Vector:** Three input fields labeled "Az:" (value 0.), "El:" (value 0.), and "Dist:" (value 1.).
- Left/Mono:** A section with a "File Name" field containing "Left.fir", a "Find File" button, and a "Scale Fact." field containing "1.".
- Right:** A section with a "Right File" field containing "Right.fir", a "Find File" button, and a "Scale Fact." field containing "1.".
- Error:** A text field at the bottom containing "none.".
- Buttons:** "Cancel" and "OK" buttons at the bottom right.

Filter files are always connected via a routing wire.

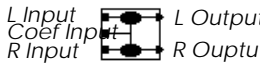
### Types of DSPs

Convolving DSPs may be of three types:

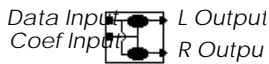
- Mono convolver DSP
- Stereo convolver DSP
- Mono in/stereo out DSP



**Mono Convolver DSP** The Mono Convolver DSP has one data input port, one coefficient input port, and one output port.



**Stereo Convolver DSP** The Stereo Convolver DSP has two data input ports, one coefficient input port, and two output ports.



**Mono In/Stereo Out DSP** The Mono In/Stereo Out DSP has one data input port, one coefficient input port, and two output ports.

### Delay Processors

The PD1 delay processor is an optional four channel delay line. Your routing schedule may include up to four delay resources, one for each delay channel.

Each delay resource has one data input port, multiple tap specification ports, and multiple output ports.

Delay tap ports may be specified in one of two ways:

- Inbound data stream
- Delay constant

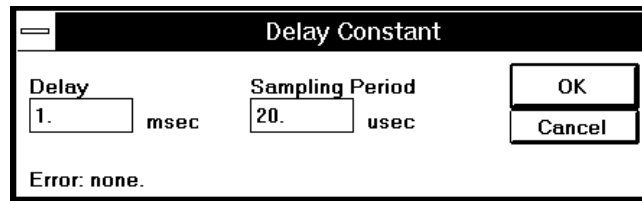


**Inbound data stream** Delay tap ports may be specified through an inbound data stream by use of a delay time record. This is a special record format that may be used to dynamically update a delay. Delay time records specify delay in terms of samples. (For more information, see "Using Inbound Data Streams" in *Programming the PD1 POWER SDAC*.)

Inbound data streams are always connected via a routing wire.



**Delay constant** If a delay value is to remain constant, it may be specified through the use of a delay constant object. Delay constants are specified in terms of delay in milliseconds and sampling rate in microseconds.

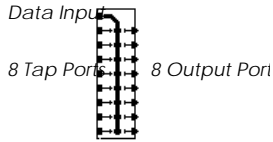


Delay constants are always connected directly to a tap specification port.

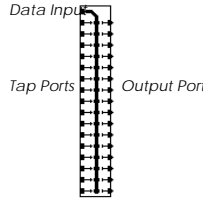
### Types of Delay Resources

Delay resources may be of two types:

- 8 Tap Delay
- 16 Tap Delay



**8 Tap Delay** The 8 tap delay has one data input port, 8 tap specification ports, and 8 output ports.



**16 Tap Delay** The 16 tap delay has one data input port, 16 tap specification ports, and 16 output ports.

### Isolation Registers

The Isolation Register is a temporary holding register. Isolation registers have many important uses. These uses are described in the sections "Routes" and "Routing Rules" later in this chapter.

## Routes

Together, *routes* comprise the heart of any routing schedule. In AutoRoute, routes are defined by drawing routing wires.

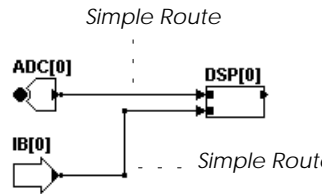
### Types of Routes

Routing wires represent two types of routes:

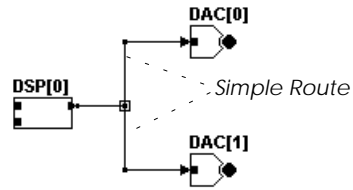
- Simple routes
- Multi-routes

**Simple Routes** A simple route connects one output port to one input port.

The diagram below illustrates two simple route connections between (1) the output port of ADC[0] and the data input port of DSP[0] and (2) the output port of IB[0] and the auxiliary input of DSP[0].

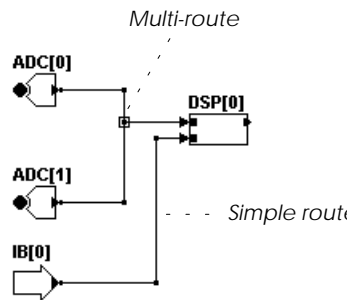


In the following diagram, there are also two simple route connections. The output from DSP[0] is connected via a simple route to the data input of DAC[0]. This output is also connected via a simple route to the data input of DAC[1].



**Multi-routes** A multi-route connects more than one output port to a single input port, in effect summing the input signals.

In the diagram below, the outputs from both ADC[0] and ADC[1] are connected to the data input of DSP[0]. A simple route connects the output port of IB[0] and the auxiliary input of DSP[0].



## Use of Multi-routes

Multi-routes combine data from more than one output port, in effect summing the data. Multi-routes provide an excellent means of performing summing operations on signal data in real time.

**Note:** You may not use a multi-route with inbound or outbound data streams. See "Using the Real Time Router" of *Programming the PDI POWER SDAC* and "Routing Rules" below for more information about this restriction.

## Route Scale Factor

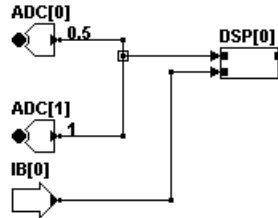
AutoRoute provides a means for scaling signal data summed with a multi-route, the *Route Scale Factor*. The Route Scale Factor is simply a multiplier that may be applied to the signal data prior to summing. Route Scale Factors may range from  $-0.9997$  to  $+1$ . By default, Route Scale Factors are set to  $+1$ .

Route scale factors may be used to:

- Attenuate one or more signal to be summed.
- Reverse the phase of one or more signal to be summed.

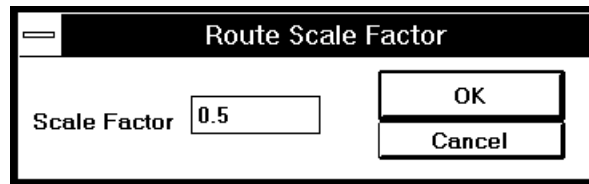
Reversing the phase can be accomplished by applying a negative multiplier.

Route Scale Factors are displayed immediately above the routing wire, as seen below:



In the above example, the signal output from ADC[0] is reduced to one-half its original amplitude and combined with the signal output from ADC[1] before being sent as input to DSP[0].

Route Scale Factors are set by entering a value in the Route Scale Factor dialog box.

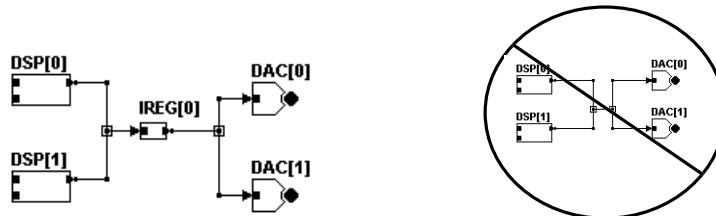


**To set a Route Factor Scale**

1. Double-click on the end of the routing wire or a segment juncture within the routing wire.  
The Route Scale Factor dialog box appears.
2. Type in a Scale Factor.
3. Click the OK button.

**Use of Isolation Registers with Multi-Routes**

You may wish to combine multiple signals with a multi-route and then send the summed signal as input to multiple resources. While it is possible to design such a route, it is extremely inefficient. The multi-route must be read for each destination resource. A better approach is to send the output of the multi-route to an isolation register. The output of the isolation register is then sent to multiple resources. By using this technique, the multi-route is read only once.



## Routing Rules

Designing a routing schedule is a fairly simple process. There are, however, a few rules to keep in mind when designing a routing schedule. These rules are discussed in detail in the "Using the Real Time Router" section of *Programming the PDI POWER SDAC*. You should take a moment to familiarize yourself with these rules before continuing with this section.

It is important to design routing schedules that conform to the routing rules. When using AutoRoute to design routing schedules, several things must be considered.

### Route Order

In general, routes are processed in the order in which they were specified, or in the case of AutoRoute, the order in which they were drawn. There are two exceptions to this general rule.

1. According to the Basic Ordering Rule, processing occurs as follows:
  - b. Multi-routes
  - c. Simple routes
  - d. Inbound and outbound data streams

This order is implemented by the *hardware* and overrides all software ordering.

2. In some cases, AutoRoute may automatically re-order routes so that they conform to basic route ordering rules.

**Note:** AutoRoute does not automatically correct all ordering errors. It is a good idea to assume that no automatic correction will be applied and draw all routes in the proper order, as described below.

### Route Order Rules

There are two main route order rules.

1. The Basic Ordering Rule (see above).

Remember that this ordering is implemented by the *hardware*.
2. Special Rule 1, which states that a read from a DSP should never occur after a write to the same DSP. This is known as a *write-before-read* error.

In most cases, write-before-read errors can be avoided by drawing all routes in reverse order of signal flow. However, in cases involving a multi-route input to a DSP followed by a simple route output from the same DSP, the implementation of the Basic Ordering Rule will cause a write-before-read error.

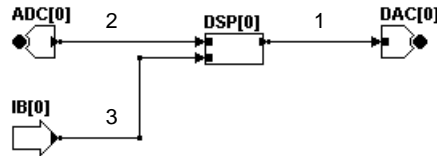
### Avoiding Write-before-read Errors

As you are designing routing schedules, keep in mind the two ordering rules above. Below are some techniques that help prevent route order rule violations.

#### Drawing Order

See Examples 1 and 2 in the "Avoiding a Write-Before-Read Problem" section of *Programming the PD1 POWER SDAC*.

Draw all routes in the reverse order of signal processing.



In the above example, a write-before-read error is avoided by connecting resources in the following order:

**Reads DSP[0], writes to DAC[0]**  
**Reads ADC[0], writes to DSP[0]**

1. Output from DSP[0] sent to the input of DAC[0].
2. Output of ADC[0] sent to the data input of DSP[0].
3. Output of IB[0] sent to the coefficient input of DSP[0].

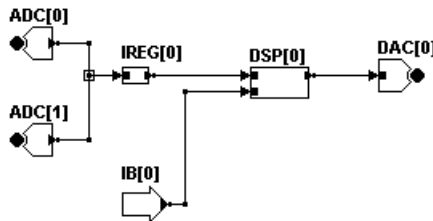
#### Use of Isolation Registers



See Example 3 in the "Avoiding a Write-Before-Read Problem" section of *Programming the PD1 POWER SDAC* for more information.

Isolation registers are a new feature of the PD1 and AutoRoute. An isolation register is a temporary holding register.

In cases where the Basic Ordering Rule causes a write-before-read error, such as a multi-route input to a DSP followed by a simple route output from the same DSP, write-before-read errors may be avoided by including an intermediary isolation register, providing that routes are drawn in the reverse order of signal flow. Such a case is illustrated below.



## Inbound Data Streams

According to Special Rule 2, multi-routes may not be used with inbound data streams.

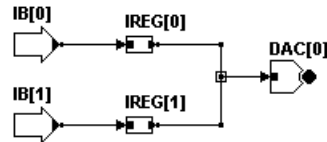
### Combining Inbound Data Streams

Multi-routes are used to sum data. In some cases, it may be desirable to sum two inbound data streams. However, multi-routes cannot be used with this type of data.

See the example in the "Using Simple Data Stream Routing" section of *Programming the PD1 POWER SDAC* for more information.

### Use of Isolation Registers

Isolation registers may be used to combine data from inbound data streams. Data from each stream is sent to its own isolation register. Output from each isolation register may then be combined via a multi-route.



## Modeling Acoustic Environments

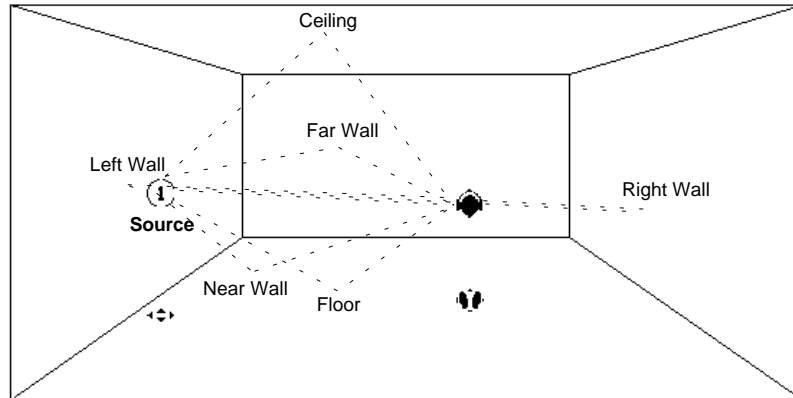
Two powerful modeling tools, AutoRoute and Sound Stage, are provided for use with the POWER SDAC system. Modeling acoustic environments using these tools is a two step process. The first step is the design of the signal processing circuit, or routing schedule. AutoRoute was designed for this purpose. The second step is the design of the acoustic environment to be modeled. Such modeling may be accomplished through use of TDT's Sound Stage software.

When designing the routing schedule, one must keep in mind how it will be implemented as part of a model of a physical environment. This section treats the development of a routing schedule as one part of the entire acoustic modeling process.

## Specifying the Routing Schedule

The process of modeling acoustic environments consists of the following:

1. Specifying sound sources and reflecting surfaces.
2. Determining how the original sound source and reflected signals should be filtered in order to simulate a free field sound received at a specific spatial position within the environment.



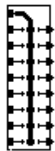
### Sound sources

Sound sources may originate in two manners:



- Analog sounds converted through use of an ADC
- Digital waveforms

Digital waveforms may be input to the PD1 via an inbound data stream.



### Reflections

When modeling the reflection of a source signal, it is necessary to specify a signal delay. This is accomplished through the use of a Delay resource.

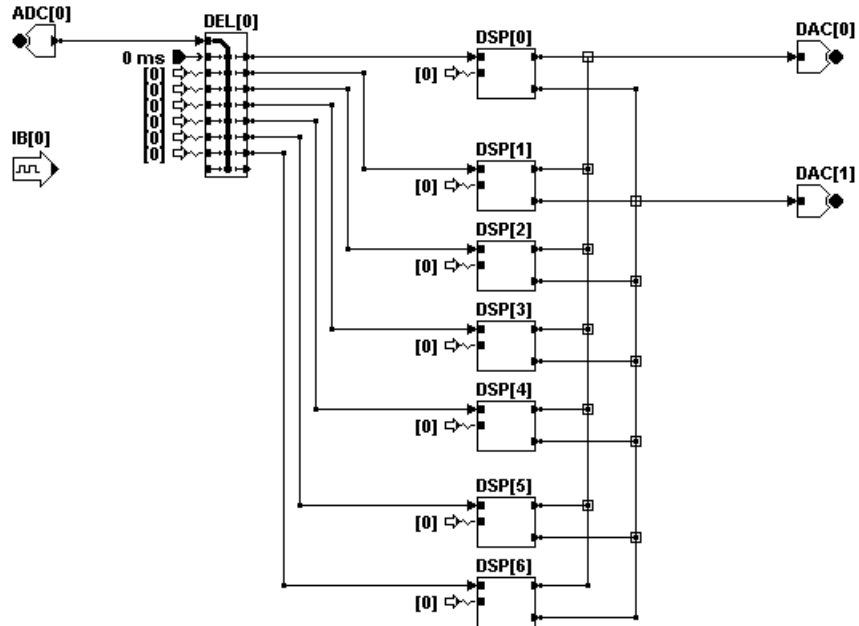


### Filters

The sound source and its reflections may be filtered to simulate the sound as it would be received by the ears. To accomplish this, each sound source or reflection is filtered according to its corresponding head related transfer function (HRTF). This filtering is accomplished through use of a DSP.

**Example: Designing a Single Sound Source Environment**

Below is an example of a stereo model that includes a single sound source. Signals emitted from this sound source are to be modeled as if they were reflected from the following: floor, ceiling, and all four walls.



In this model, the sound source signal originates from ADC[0]. The reflection of the sound source off the floor, ceiling, and four walls is simulated by delaying the reflected signal relative to the sound source. This is accomplished by including an 8-Tap Delay, DEL[0]. The first delay input of DEL[0], known as TAP[0], specifies a 0 millisecond delay of the source signal, simulating a direct path to the ears. Delays due to reflection are simulated through use of TAP[1] through TAP[6] of DEL[0]. The source signal and its six reflections are then filtered to simulate sound received at the ears. Filtering of the direct path to the ears is accomplished by DSP[0]. Filtering based on reflection off the floor, ceiling, and four walls is accomplished using DSP[1] through DSP[6]. Each DSP outputs a right and left channel. All left channel output signals are summed and sent to DAC[0]. Right channel signal output signals are summed and sent to DAC[1].

## Dynamic Updating through Inbound Data Streams

See the *PD1 User's Guide*, "Using Inbound Data Streams" for more information.

The determination of filter coefficients and delay times is dependent on the head related transfer function (HRTF). This function may be updated in real time, requiring dynamic update of all coefficients and delay times. This dynamic updating is accomplished through the use of inbound data streams. (see *Programming the PD1 POWER SDAC*, "Using Inbound Data Streams" for more information.)

### Specifying Inbound Data Streams

AutoRoute is a signal processing design tool. As such, it is useful for designing abstract signal processing circuits. AutoRoute does not know how elements in these circuits correspond to real world objects such as sound sources and reflective surfaces. Therefore, it is impossible for AutoRoute to know how to dynamically update coefficient and delay data. It is through the Sound Stage application that you may associate coefficient or delay data with real world objects such as a sound sources or reflective surfaces.

However, AutoRoute does allow the specification of an IB Patch. The IB Patch indicates that coefficients will be obtained in some manner from the associated inbound data stream. The association of an inbound data stream to a resource input port is stored in the routing schedule file and read by TDT's Sound Stage application. Through Sound Stage, the exact source of the inbound coefficient or delay data may be defined.

The use of IB Patches to indicate that data will be obtained from an inbound data stream is illustrated in the above example. In this example, one inbound data stream, IB[0] is specified. Filter coefficients and delay data will be dynamically updated through use of this inbound data stream. This is indicated by assigning an IB patch associated with IB[0] to each coefficient and delay input.



# Chapter 4 Evaluating a Routing Schedule

AutoRoute allows you to design complex signal processing circuits. It may be necessary to evaluate these circuits to determine whether or not they contain any logical errors. It may also be nice to listen to the output of the circuit. These two evaluations may be performed through:

- Processing
- Implementing

## Processing

AutoRoute is capable of evaluating the routing schedule to determine whether or not it contains any logical errors. This is accomplished by analysis of each routing wire connection. This analysis is termed *processing*.

### What Happens in Processing?

After processing, routing errors are indicated through use of color. Routing wires that are connected erroneously are marked in certain user-defined colors. See *Chapter 2, "Customizing Colors"* for more information about customizing error colors.

Additionally, correctly connected routing wires are also marked by color during processing. You may choose to mark correct routes in one of the following manners:

- By route type  
User-defined color is used to indicate simple routes and multi-routes.
- By individual node  
Each route, or node, is marked with a distinctive color assigned by AutoRoute.

#### *To color routes by route type*

- Choose Show Route Type from the View menu.

#### *To color individual routes*

- Choose Show Node from the View menu.

### Types of Processing

There are two methods of processing:

- Automatic Processing
- Manual Processing

## Automatic Processing

When Automatic Processing is enabled, AutoRoute evaluates the routing schedule each time an edit is completed or a new routing wire connection is drawn.

Automatic Processing may not be appropriate in some cases. If the model is large, analysis of routing wire connections following each edit may dramatically slow performance. In such cases, it is better to disable Automatic Processing. Processing can then be performed manually after drawing is completed.

### *To enable/disable Automatic Processing*

- Click Auto Process in the Processing menu.

When Automatic Processing is enabled, a check appears in front of the Auto Process item in the Processing menu. This check is absent when Automatic Processing is disabled.

## Manual Processing

Routing schedules may be manually processed at any time.

### *To manually process a routing schedule*

- Click Process in the Processing menu.

or

- Click the Process icon.

## Implementing

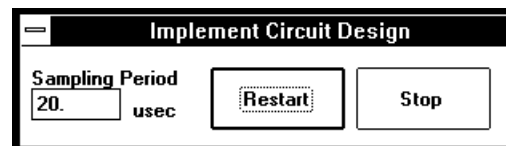
It may be desirable to listen to the output of a routing schedule. Provided that your system includes all the required hardware, you may listen to output by *implementing*.

### *To listen to the routing schedule output*

- Choose Implement from the Processing menu.

or

- Click the Implement icon.
  - a. Enter the sampling period in the Sampling Period field.
  - b. Click Restart to continue or Stop to terminate.



# Chapter 5 Using Routing Schedule Files

Routing schedule design with AutoRoute is often just the first step in a much bigger signal modeling process, such as acoustic modeling. Routing schedules may be used to 'wire up' PD1 circuits in TDT's Sound Stage as well as a variety of custom applications. This is accomplished by saving routing schedules as files. These files can then be used by other programs.

Routing schedules may be saved in the following file formats:

- Binary file format
- 'C' text file format
- Pascal text file format

## Binary Files

Routing schedules designed with AutoRoute may be saved as binary files for use with TDT's Sound Stage acoustic environment design tool or with your own programs written in 'C'.

Binary routing schedule files are saved with the default extension, *rs*.

## Binary File Format

To be included in a future version of the document.

## Saving Binary Files

*To save a routing schedule in binary format*

- Choose Generate Binary File from the Processing menu.

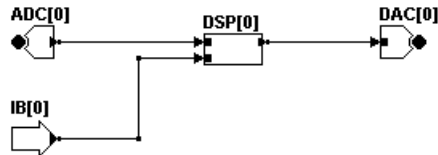
## 'C' Files

See the [XBDRV Software Reference Guide](#) for information concerning software calls.

You may wish to use an AutoRoute routing schedule with custom-designed applications written in 'C'. AutoRoute allows you to generate 'C' code containing the line by line instructions needed to program the routing schedule.

'C' text files are saved with the default extension, .c.

A simple routing schedule and the 'C' code it generates are presented below.



**Note:** This code illustrates the principle of **reverse ordering**. As discussed in Chapter 3, reverse ordering can avoid a **write-before-read** error. In this case, the read from DSP[0] occurs before the write to DSP[0].

See Examples 1 and 2 in the "Avoiding a Write-Before-Read Problem" section of [Programming the PD1 POWER SDAC](#).

```

/*****
/* The following dynamic stream assignments are specified... */
/*****
#include "xbdrv.h"
#include "pdl_sup.h"

/*****
/* Call this procedure to auto route the PD1.          */
/*****
int RouteSched(int din)
{
    int src[32];
    float sf[32];

    PDLclrsched(din);

    PDLaddsimp(din, DSPout[0], DAC[0]);
    PDLaddsimp(din, ADC[0], DSPin[0]);
    PDLspecIB(din, IC[0], COEF[0]);
    return(1);
}

/*****
/* Call this procedure to setup delay processor.      */
/*****
int DelaySet(int din)
{
    PDLclrDEL(din, 0, 0, 0, 0);
}

/*****
/* Call this procedure to preload coefs to DSPs      */
/*****
int PreloadCoef(int din)
{
}

```

## Saving 'C' Text Files

*To save a routing schedule in 'C' text file format*

- ▶ Choose Generate 'C' Source from the Processing menu.

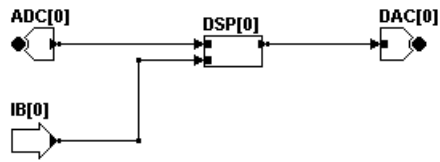
## Pascal Files

See the [XBDRV Software Reference Guide](#) for information concerning software calls.

You may wish to use an AutoRoute routing schedule with custom-designed applications written in Pascal. AutoRoute allows you to generate Pascal code containing the line by line instructions needed to program the routing schedule.

Pascal text files are saved with the default extension, *.pas*.

A simple routing schedule and the Pascal code it generates are presented below.



Code to be included in a future version.

## Saving Pascal Text Files

*To save a routing schedule in Pascal text file format*

- Choose Generate Pascal Source from the Processing menu.