User’s Manual

TRAKKER Antares® Application Simulator

Intermec

A UNOVA Company

Intermec Technologies Corporation
6001 36th Avenue West
P.O. Box 4280
Everett, WA 98203-9280

U.S. service and technical support: 1-800-755-5505
U.S. media supplies ordering information: 1-800-227-9947

Canadian service and technical support: 1-800-688-7043
Canadian media supplies ordering information: 1-800-268-6936

Outside U.S. and Canada: Contact your local Intermec service supplier.

Manual Change Record

This page records the changes to this manual. The manual was originally released as version 001.

Version	Date	Description of Change
002	5/97	Added information on the keypad_type and serial_receive_mode INI parameters. Also explained how to configure the Simulator to support international characters.
003	4/98	Added information on the Optical Input and Screen Type INI parameters.
004	1/99	No changes to Simulator manual for release -004.
005	7/99	Added 14 new INI parameters to Chapter 3.
		Documented the new Simulator Editor in Chapter 4.
		Added new error and status messages to Chapter 5.
006	1/00	Minor changes were made to Simulator manual for release -006.

Contents

3 INI File Parameter Descriptions

Before You Begin

This section introduces you to standard warranty provisions, cautions and notes, document formatting conventions, and sources of additional product information.

Warranty Information

To receive a copy of the standard warranty provision for this product, contact your local Intermec support services organization. In the U.S. call 1.800.755.5505, and in Canada call 1.800.688.7043. If you live outside of the U.S. or Canada, you can find your local Intermec support services organization on the Intermec Web site at www.intermec.com.

Cautions and Notes

The cautions and notes in this manual use the following format.

CAUTION

Caution
A caution alerts you to an operating procedure, practice, condition, or statement that must be strictly observed to prevent equipment damage or destruction, or corruption or loss of data.

Conseil
Une précaution vous alerte d’une procédure de fonctionnement, d’une méthode, d’un état ou d’un rapport qui doit être strictement respecté pour empêcher l’endommagement ou la destruction de l’équipement, ou l’altération ou la perte de données.

Note: Notes are statements that either provide extra information about a topic or contain special instructions for handling a particular condition or set of circumstances.

About This Manual

This manual describes how to use the Trakker Antares® Application Simulator, which helps you create, test, and debug applications for Trakker Antares terminals with the Programmable Option.

Use this manual in conjunction with Part I, *Trakker Antares PSK Reference Manual*, which describes the PSK library functions that the Application Simulator captures and simulates.

Communications Ports May Not Be Available

Both this manual and the Simulator software refer to communications ports that may not be available on your Trakker Antares terminals. The ports are COM1, COM2, and an RF network port, NET. To learn which ports are available on your terminal, check your user's manual.

Intended Audience

This manual is intended for experienced PC programmers who already understand return values, know how to program in C, and know how to use the Microsoft Visual C/C++ and Microsoft CodeView for DOS debugger. They have already read Part I, *Trakker Antares PSK Reference Manual,* so they understand how to create programs for Trakker Antares terminals.

How This Manual Is Organized

This manual is organized as follows:

Chapter	What You'll Find
1	*Getting Started* Introduces the Application Simulator and explains how to install the software. Also explains how to start and exit the Simulator TSR.
2	*Working With the Application Simulator* Describes how the Simulator works, explains which features the Simulator can make your PC mimic, and gives hints for debugging an application.
3	*INI File Parameter Descriptions* Lists and describes the parameters in the initialization (INI) file.
4	*Customizing INI Files With the Editor* Explains how to use the Editor to customize the INI file.
5	*Troubleshooting* Lists and describes the status and error messages you may see.
A	*Adding the Simulator to the Microsoft Visual C/C++ Tools Menu* Describes how to manually add Simulator commands to the Microsoft Visual C/C++ Tools Menu, in case the commands were not added automatically during the PSK and Simulator installation.

Terminology

You should be aware of how these terms are used in this manual:

Term	Meaning
DCS 300	The term refers to the new data collection server that replaces the Model 200 Controller. Unless otherwise noted, you can use either the DCS 300 or the Model 200 Controller.
Trakker Antares terminals or terminals	These terms refer to Intermec's Trakker Antares family of terminals, such as the T242X hand-held terminals and the T248X stationary terminals.
PSK function, library function, or function	These terms refer to the library functions described in Part I, *Trakker Antares PSK Reference Manual*.

Conventions for Input From a Keyboard or Keypad

You should be aware of these formatting conventions for representing input from a keyboard or keypad.

Convention	Meaning
Bold	Keys that you press on a PC keyboard are shown in **bold**. For example, "press **Enter**" means you press the key labeled "Enter" on the keyboard. The first letter of a key name is always capitalized.
$\boxed{A}$	Keys that you press on the Trakker Antares keypad are shown by icons. For a complete list of the keypad keys, see your terminal user's manual.
$\boxed{=f}$ $\boxed{g}$	When a series of keys are shown with no connectors between them, you must press and release each key in the order shown. For example, to use viewport pagedown on a terminal, you press and release the $\boxed{=f}$ key and then press and release the $\boxed{g}$ key.
Ctrl-Alt-Del	When a series of keys are shown with a dash between them, you must press and hold the keys in the order shown and then release them all. For example, to boot a PC, you press and hold **Ctrl**, press and hold **Alt**, press and hold **Del**, and then release the keys.

Conventions for Commands

You should be aware of these formatting conventions for entering commands.

Convention	Meaning
`Courier`	Commands are shown in Courier exactly as you must type them. For example: `dir`
Italics	Variables are shown in italics. You must enter a real value for the variable. For example, replace *filename* with the name of the INI file in this command: `IMT24SIM` *`filename`*`.ini`.

Other Intermec Manuals

You may need additional information when working with the Application Simulator. Please visit our Web site at www.intermec.com to download many of our current manuals in PDF format. To order printed versions of the Intermec manuals, contact your local Intermec representative or distributor.

Also, you should see the online README file provided with the software. This README file contains important information that was not available when this manual was printed, such as operating guidelines.

1

Getting Started

1 Getting Started

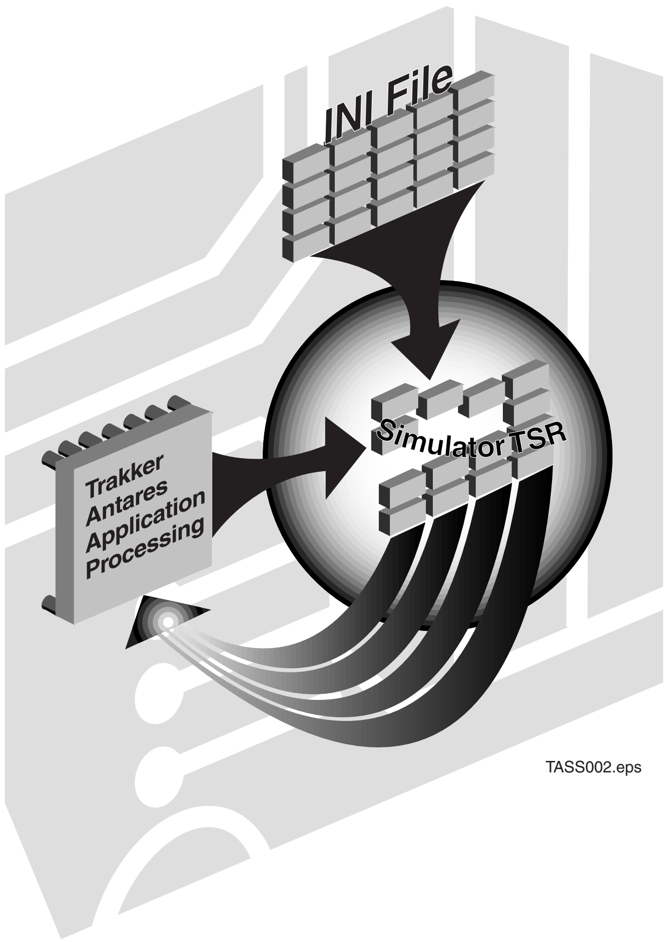
This chapter introduces the Trakker Antares Application Simulator, describes its installation, explains how to use the Simulator with other products as part of your development process, and ends with instructions for starting and exiting the TSR.

Introduction to the Application Simulator

The Trakker Antares Application Simulator helps you create, debug, test, and run on your PC any applications you create for Trakker Antares terminals. The Simulator lets you use Microsoft Visual C/C++ and Microsoft Codeview for DOS to develop Trakker Antares applications.

Without the Simulator, you cannot run Trakker Antares applications on a PC because the applications contain PSK functions that are not PC-compatible. With the Simulator, however, you can run these applications on a PC. The Simulator captures the PSK functions and terminal-specific interrupts before they disrupt the PC.

This illustration shows how the Simulator operates.



While a Trakker Antares application is processing on a PC, it issues a terminal-specific system interrupt.

The Simulator terminate and stay resident (TSR) program captures the interrupt, uses values from the initialization (INI) file to assemble a response, and sends the response to the application.

The application accepts the response and continues processing.

For a detailed technical description, see "How the Simulator Works" in Chapter 2.

Installing the Simulator

The Application Simulator is installed automatically when you install the Trakker Antares Programmer's Software Kit (PSK). For instructions, see Chapter 1 in Part I, *Trakker Antares PSK Reference Manual*.

The installation creates and fills these directories on your PC:

- C:\INTERMEC\IMT24\SIM
- C:\INTERMEC\IMT24\LIB
- C:\INTERMEC\IMT24\INCLUDE
- C:\INTERMEC\IMT24\SAMPLES

The installation creates a batch file in the WINDOWS or WIN95 directory called IMT24SIM.BAT that lets you use the IMT24SIM.EXE command from any directory. The installation also adds the IMT24SIM environment variable to AUTOEXEC.BAT. The environment variable points to the locations of the Simulator software.

The installation creates a Trakker Antares Development Tools group on the Windows desktop. The group contains icons for the FileCopy utility, the Editor, the Editor's online help system, and the README file.

Read the README file before you use the Simulator. This file may contain information that was not available when the manual was printed. For help using the Editor, see Chapter 4, "Customizing INI Files With the Editor."

Note: If Microsoft Visual C/C++ is already installed when you install the Application Simulator, the installation adds two commands to the Tools menu: Codeview for Trakker and Simulation for Trakker. You can choose Codeview for Trakker to load the Simulator TSR into memory and start Microsoft Codeview for DOS, or you can choose Simulation for Trakker to load the Simulator TSR into memory and run the application you are currently editing.

Using the Simulator With Other Products

The Application Simulator was designed to be used with the Microsoft Visual C/C++ Professional Edition (version 1.0 or 1.5x) application development software. You can use Microsoft Visual C/C++ and Microsoft Codeview for DOS to create and debug applications for your Trakker Antares terminals.

You can install an Intermec wedge on your PC to simulate bar code input. For help configuring a wedge, see “Bar Code Input” in Chapter 2.

The Simulator TSR should not interfere with the normal operation of other software on your PC. Therefore, you can load the Simulator TSR into memory and leave it running if you have sufficient RAM available.

Using the Simulator in the Development Process

The Application Simulator can be an integral part of your application development process. For example, if you installed Microsoft Visual C++ before you installed the Simulator, you can follow these steps:

1. Start Microsoft Visual C/C++.
2. Open your Trakker Antares project or create a new project.
3. Select Simulation for Trakker from the Tools menu to load the Simulator into memory.
4. Run and debug the application.
5. Exit the debugger. The Simulator is unloaded from memory and you are returned to Microsoft Visual C/C++.

Note: If you are using Windows 95, you may receive a DOS informational message when you exit the debugger and terminate the Simulator. The message requests you to press Ctrl-C to exit the DOS session.

Starting the Simulator

The Application Simulator is a TSR program, which is a program that is loaded into DOS memory and runs in the background. This table lists the methods for starting the Simulator.

Operating System	Method of Starting the Simulator	How Long the Simulator Is Valid
Windows 95, 98, or NT	Start the TSR from a DOS window while Windows is running.	The TSR is valid only in that DOS window.
	Choose Simulation for Trakker from the Tools menu of Microsoft Visual C/C++.	The TSR is valid until you exit the DOS session that is created.
	Choose Codeview for Trakker from the Tools menu of Microsoft Visual C/C++.	The TSR is valid until you exit Codeview.
Windows 3.1	Start the TSR from DOS before you start Windows.	The TSR is valid until you shut down your PC, or you exit Windows and unload the TSR.
	Start the TSR from your AUTOEXEC.BAT file.	The TSR is valid until you shut down your PC, or you exit Windows and unload the TSR.
	Start the TSR from a DOS window while Windows is running.	The TSR is valid only in that DOS window.
	Choose Simulation for Trakker from the Tools menu of Microsoft Visual C/C++.	The TSR is valid until you exit the DOS session that is created.
	Choose Codeview for Trakker from the Tools menu of Microsoft Visual C/C++.	The TSR is valid until you exit Codeview.
	Start the TSR from a DOS session spawned from Codeview.	The TSR is valid until you exit Codeview.

If you are using Windows 95, follow these guidelines:

- Do not start Codeview, spawn a DOS session, and start the TSR because the TSR will remain valid only until you exit the spawned DOS session. It will not be valid when you return to Codeview.
- Do not start the TSR from the AUTOEXEC.BAT file because it may crash your PC.

Here are the commands for starting the Simulator. You can use these commands from the DOS prompt.

To start the Simulator using default INI file

- At the DOS prompt, type this command:

  ```
  imt24sim
  ```

  When the software finishes loading, this message appears:

  ```
  Simulator has been loaded with:  IMT24SIM.INI
  ```

  You can now use your application development software to run and debug Trakker Antares applications. IMT24SIM.INI is the default INI file.

To start the Simulator using a custom INI file

- At the DOS prompt, type this command:

  ```
  imt24sim filename.ini
  ```

  Where *filename.ini* is the name of your customized INI file.

  When the software finishes loading, this message appears:

  ```
  Simulator has been loaded with:  filename.ini
  ```

  You can now use your application development software to run and debug Trakker Antares applications.

Note: If you are using Windows 3.1, you can also include these commands in your AUTOEXEC.BAT file so the TSR is loaded automatically when you boot the PC.

Exiting the Simulator

To unload the Application Simulator from memory, type this command at the DOS prompt:

```
imt24sim /d
```

This message appears:

```
Simulator has been unloaded.
```

2

Working With the Application Simulator

This chapter describes how the Simulator works and how it simulates various terminal features and PSK functions.

How the Simulator Works

The Application Simulator consists of three parts:

Simulator The Simulator terminate and stay resident (TSR) program runs in the background on your PC. The Simulator captures terminal-specific system interrupts and makes your PC mimic a Trakker Antares terminal.

INI file The initialization (INI) file specifies how the Simulator simulates terminal features such as communications. IMT24SIM.INI is the default INI file. To learn about the parameters in the INI file, see Chapter 3.

Simulator Editor The Simulator Editor is a Windows-based tool that you use to set the parameters that are stored in the IMT24SIM.INI file. For help using the Editor, see Chapter 4.

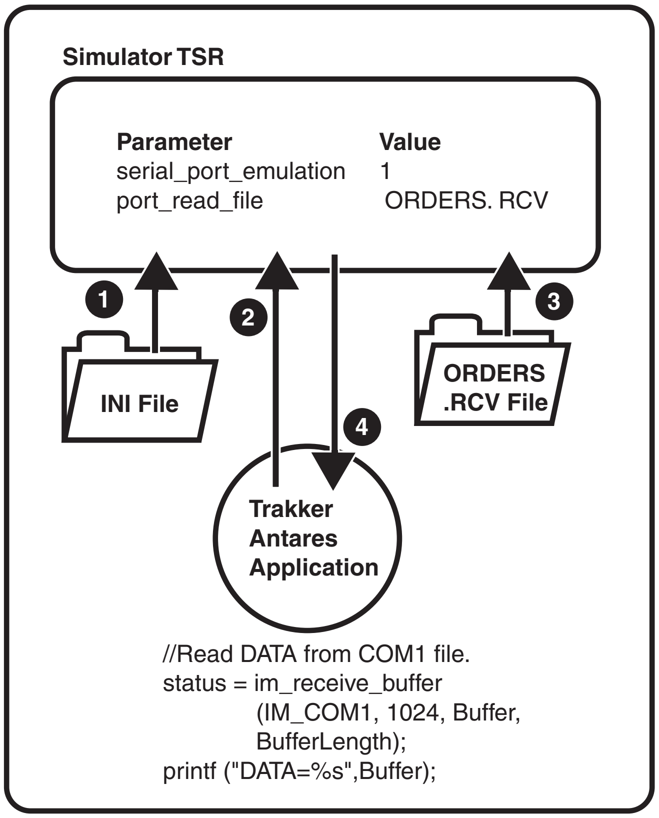
The Simulator uses the parameters in the INI file to simulate some terminal features and PSK functions, as described in “Simulated Features and Functions“ later in this chapter. For example, you can use INI parameters to test how the Trakker Antares application handles incoming serial communications.

Example: How to Simulate Incoming Serial Communications

1. In the INI file, set the serial_port_emulation parameter to 1 to indicate that you are simulating serial communications through an ASCII file.
2. In the INI file, set the port_read_file parameter to ORDERS.RCV to identify the ASCII file that contains the data to be input as if received on the terminal’s COM1 port.
3. Create the ORDERS.RCV file and fill it with data. The data will be input to the application as if received on the terminal’s COM1.

   For help formatting the data, see Step 1 in the “To simulate data input to the application through NET” procedure in “Communications Input and Output” later in this chapter.
4. Start the Simulator and run your Trakker Antares application. When the application issues the im_receive_buffer PSK function, the Simulator reads from ORDERS.RCV to simulate incoming serial communications.
5. Test how the application responds to the incoming communication.

This illustration shows how the Simulator TSR simulates incoming data:



```
//Read DATA from COM1 file.
status = im_receive_buffer
            (IM_COM1, 1024, Buffer,
            BufferLength);
printf ("DATA=%s",Buffer);
```

TASS003.eps

❶ When you start the Simulator, it reads the parameters from the INI file, parses the parameter names, and saves the values into variables in memory.

❷ The Trakker Antares application executes on the PC. To read incoming data, the application executes the im_receive_buffer function, which issues a terminal-specific interrupt.

❸ The Simulator TSR captures the interrupt. Because serial_port_emulation is enabled, the Simulator TSR reads data from the port_read_file until it reaches a CR/LF. The Simulator TSR uses this data to assemble the response to the im_receive_buffer function.

❹ The Simulator passes the response to the application, which accepts the information as the status and return values of the im_receive_buffer function. The application continues executing.

Understanding the Limitations of the Simulator

Read these notes to understand the limitations of the Application Simulator:

- Start the Simulator before you run any Trakker Antares applications on your PC. Otherwise, you receive an error message and the application does not run.
- The Simulator does not help you test the application's user interface or performance. You can test those characteristics far better with a Trakker Antares terminal. Always test your application by running it on a terminal after you have finished debugging the logic.
- Intermec provides you with the LLIBCA.LIB library to link to when you are creating applications for terminals that are not running in DOS mode. If you link to the Microsoft LLIBCA.LIB library instead, you may find that an application containing an erroneous input combination will fail on the Trakker Antares terminal, but will not be detected by the Simulator.

Simulated Features and Functions

The Simulator automatically simulates many features of the Trakker Antares terminal's performance as well as many PSK functions. Other features and functions are simulated according to values you set for parameters in an initialization (INI) file.

To learn how each feature and function is simulated, read these sections:

- Features That Are Automatically Simulated
- Features That Are Configured With an INI File
- Features That Are Not Simulated
- How PSK Functions Are Simulated

Features That Are Automatically Simulated

The Simulator can reproduce these terminal features:

- Text Display
- Function Left and Function Right Keys
- Viewport

Text Display

If the application calls the im_set_input_mode function to select the programmer input mode, the Simulator limits character echoing to the lines and columns based on the display size you specified by calling the im_set_display_mode function.

Some display characteristics are represented on the PC in color, as shown in the next table.

Display Characteristics	Background Color	Foreground Color	Blinking*
Normal	Black	Light gray	
Reverse video	Light gray	Black	
Underline	Black	Magenta	
Underline and reverse video	Magenta	Black	
Normal and blinking	Black	Light gray	Yes
Reverse video and blinking	Light gray	Black	Yes
Underline and blinking	Black	Magenta	Yes
Underline, reverse video, and blinking	Magenta	Black	Yes
Bold	Black	White	
Bold and reverse video	Light gray	Dark gray	
Bold and underline	Black	Magenta	
Bold, underline, and reverse video	Magenta	White	
Bold and blinking	Black	White	Yes
Bold, reverse video, and blinking	Light gray	Dark gray	Yes
Bold, underline, and blinking	Black	Magenta	Yes
Bold, underline, reverse video, and blinking	Magenta	White	Yes

* Text will blink only in the DOS environment.

Function Left and Function Right Keys

The Simulator simulates the terminal's Function Left ($\boxed{\_\mathbf{f}}$) and Function Right ($\boxed{=\mathbf{f}}$) keys as follows:

To Simulate This Key	Press These Keys
$\boxed{\_\mathbf{f}}$ (Function Left)	Alt
$\boxed{=\mathbf{f}}$ (Function Right)	Alt Shift

For example, to use viewport page down on a terminal, you press $\boxed{\_\mathbf{f}}$ $\boxed{3}$. To use viewport page down on a PC using the Simulator, you sequentially press **Alt 3** (you must press the 3 on the number pad).

Viewport

The Simulator simulates the viewport for the terminal.

Features That Are Configured With an INI File

If you set the corresponding parameters in the INI file, the Simulator can reproduce these features:

- Bar Code Input
- Communications Input and Output
- File Transfer
- International Characters
- Terminal Emulation Keypad or Programmable Keypad

Bar Code Input

You can simulate bar code input using either of these two methods:

- By pressing a special key sequence and typing “bar code” data.
- By configuring an Intermec wedge and scanning actual bar code labels.

To simulate bar code input with a keyboard

1. Press the key sequence specified by the sim_wand_key in the INI file. (The default key sequence is **Ctrl–G**.)

Note: You specify the symbology of the simulated bar code label with the label_symbology parameter in the INI file.

2. Type the data you want entered into the application as if it were bar code input from a scanner or wand.
3. Press **Enter** to terminate the simulated bar code input.

To simulate bar code input with an Intermec wedge

1. Attach an Intermec wedge to your PC.
2. Set the first preamble characters to match the sim_wand_key value. For help, see the next procedure, “To determine how to set the preamble for the Intermec wedge.”
3. Use the wedge to scan bar code labels. The bar code data is entered into the Trakker Antares application.
4. Scan Enter or have a Return in the postamble to terminate the bar code input.

To determine how to set the preamble for the Intermec wedge

- See your wedge or scanner documentation for help setting the preamble. Also, consider this example:

  Your sim_wand_key is **Ctrl–G**, so you must set the preamble to **Ctrl–G**. Because the wedge is in Set Preamble mode, you cannot scan the BEL character, even though it represents **Ctrl–G** in the full ASCII chart. Instead, you must consult the PC/Workstation Keyboard Mapping table (in your wedge or scanner documentation) to learn which characters to scan for **Ctrl** and **G**. According to the table, you must scan the SO character for **Ctrl** and the lowercase g character for **G**. (If you were to scan SO and uppercase G, the preamble would be set to **Ctrl–Shift–G**.)

Communications Input and Output

The Simulator can simulate input and output data through the terminal's COM1, COM2, and network port (NET). The next two procedures illustrate how to simulate I/O through NET.

To simulate data input to the application through NET

1. Type sample data in an ASCII file. You need to know which PSK function you will use in the Trakker Antares application to read data from the file.

   For example, both im_receive_buffer and im_receive_input read a line of data from the file each time they are called. A line of data is either:

   - A data string that ends in a <CR><LF>
   - A data string that is 1024 bytes long (without a <CR><LF>)

   With each subsequent call, both functions continue reading data where they left off in the file until they reach an EOF. If the functions are called again after reaching the EOF, they respond differently:

   im_receive_buffer This function starts reading data at the top of the file.

   im_receive_input This function does **not** start reading data at the beginning of the file. This practice allows keyboard, scanner, or wand input.

2. Specify the path and filename of the ASCII file in the network_read_file INI parameter.
3. Set the network_emulation INI parameter to 1 to enable the Simulator to conduct its network communications through ASCII data files.
4. Load the Simulator and run the application.
5. Verify that the application read the data correctly from the ASCII file.

To simulate data output from the application through NET

1. Specify the path and filename of the output file in the network_write_file INI parameter.
2. Set the network_emulation INI parameter to 1 to enable the Simulator to conduct its network communications through ASCII data files.
3. Load the Simulator and run the application.
4. The application creates the ASCII file and writes data to it when the im_transmit_buffer or im_transmit_file functions execute.
5. Verify that the application wrote the data correctly to the ASCII file.
6. Test the application on a Trakker Antares terminal to make sure that the application is handling the input and output communications protocols correctly.

File Transfer

The Simulator can simulate the return values for im_receive_file and im_transmit_file functions, which let you transfer files between the terminal and a DCS 300. However, the Simulator does not transfer or simulate transferring the file specified in the im_receive_file and im_transmit_file functions.

International Characters

You can configure the Simulator to support Western European characters (such as é, ü, and õ).

To configure the Simulator to support international characters

1. Set the keypad_type INI parameter to 1 for the programmable keypad.
2. Set your PC's display to Code Page 850 by adding these commands to your AUTOEXEC.BAT and CONFIG.SYS files:

   - AUTOEXEC.BAT
     ```
     nlsfunc
     mode con cp prep=((850)c:\windows\command\ega.cpi)
     chcp 850
     ```
   - CONFIG.SYS
     ```
     country=001, ,c:\windows\command\country.sys
     device=c:\windows\command\display.sys con=(ega,850,1)
     ```
3. Make sure you run the Simulator in a true DOS window, not a DOS screen. The international characters will not be displayed in a DOS screen.

   If you start the Simulator by selecting Simulator for Trakker from the Tools menu of Microsoft Visual C/C++, you are running the Simulator in a DOS screen and international characters will not be displayed. Windows 95 users can press **Alt–Enter** to change the DOS screen to a true DOS window.

Terminal Emulation Keypad or Programmable Keypad

You can simulate either the terminal emulation keypad or the programmable keypad by setting the keypad_type parameter in the INI file. You should choose the keypad that will be used on the terminals that will run the application being tested with the Simulator.

Setting the keypad_type parameter is equivalent to selecting Configuration Menu, Terminal Menu, and Keypad from the TRAKKER Antares 2400 Menu System and then setting the keypad type for the terminal.

Features That Are Not Simulated

The Simulator cannot reproduce these features:

Feature	Description
Contrast level	The Simulator does not simulate the contrast set for the terminal.
Speed and performance	The Simulator does not simulate the speed or performance of a Trakker Antares terminal. Your Trakker Antares applications run as fast as your PC can execute them.
Special key sequences	The terminal’s keypad contains fewer keys than a standard PC-AT keyboard, but you can produce all 102 PC-AT keys with the terminal by pressing a variety of key combinations. The special key sequences are listed in your Trakker Antares terminal user’s manual. When using the Simulator to run Trakker Antares applications on a PC, you do not use special key sequences because your PC keyboard contains all 102 PC-AT keys. For example, to type a comma (**,**) on a terminal, you press the [f] [v] keys. During a simulation, you simply press the comma key on the PC keyboard or you can press the **Alt V** keys sequentially. **Note:** As described in “Function Left and Function Right Keys” earlier in this chapter, you can press Alt on your PC keyboard to simulate the [f] key.
Double-byte functions	The Simulator does not simulate double-byte symbologies nor does it simulate double-byte characters.

How PSK Functions Are Simulated

Most PSK functions are automatically simulated by the Simulator. However, the Simulator can simulate some PSK functions only with the preset values in the INI file.

This table lists the functions that require the INI file configuration:

PSK Function	INI File Parameter
im_get_label_symbology	label_symbology
im_receive_buffer	network_read_file port*n*_read_file serial_receive_mode
im_receive_field	keypad_type label_postamble label_postamble_string label_preamble label_preamble_string label_time_stamp network_read_file port*n*_read_file
im_receive_file	receive_file_return
im_receive_input	keypad_type label_postamble label_postamble_string label_preamble label_preamble_string label_symbology label_time_stamp network_read_file port*n*_read_file
im_transmit_buffer	network_write_file port*n*_write_file
im_transmit_file	transmit_file_return

For a complete list of PSK functions, see Part I, *Trakker Antares PSK Reference Manual*.

3

INI File Parameter Descriptions

INI File Parameter Descriptions

This chapter explains why you should customize your INI file, describes the parameters in the INI file, and explains which PSK library functions will receive the INI parameters as return values and out parameters.

Why and How You Customize INI Files

The INI parameters control how the Application Simulator simulates a Trakker Antares terminal executing an application. You can customize the parameters so the Simulator mimics the conditions against which you want to test your Trakker Antares applications.

Note: Customizing the INI file is optional. You do not have to customize the INI file if you are satisfied with the default values in IMT24SIM.INI.

You can customize INI parameters using the Simulator Editor or any ASCII text editor:

- The Editor provides a graphic environment for changing your INI files. Instead of typing the settings, you select them from menus, and dialog boxes. For help using the Editor, see Chapter 4.
- If you create and edit INI files with an ASCII text editor, use a copy of the default INI file to make sure you conform to the formatting conventions.

Parameter Descriptions

This section contains descriptions of the parameters in the INI file, in alphabetical order. The first description, Sample, illustrates the type of information presented for each parameter and is not a valid parameter.

Note: Before setting a parameter that refers to the communications ports COM1, COM2, or NET, check your terminal user's manual to make sure those ports are available on your terminal.

Sample

Purpose	The purpose of the parameter.
Default	The default value for the parameter.
Values	The values you can set for the parameter.
Function	The PSK library function that receives the parameter as a return value or out parameter. An out parameter specifies a value that is returned by the function. Some parameters do not have a PSK function associated with them.
Notes	Optional information about the parameter.

add_carriage_return

Purpose	Enables or disables adding a carriage return character to the end of each buffer written to the port*n*_write_file using im_transmit_buffer.
Default	1 - enable
Values	0 - disable 1 - enable
Function	im_transmit_buffer
Notes	Each time you use im_transmit_buffer to write a string into the port*n*_write_file, the Simulator adds a carriage return character after the string. This helps delineate each string. If you disable this parameter, each string is appended directly to the previous string.

battery_status_return

Purpose	Specifies the value to be returned when the application checks the power remaining in the main battery.
Default	100
Values	A number from 0 to 100 (in increments of ten)
Function	im_battery_status
Notes	You can use this parameter to test how the Trakker Antares application responds when the main battery power needs to be charged.

font_scan_row_select

Purpose	Specifies the starting row number (0 to 15) of the glyph font to be displayed.
Default	0
Values	0 to 15
Function	im_overlay_setup

keypad_type

Purpose	Specifies the Trakker Antares terminal keypad that the Simulator will mimic.
Default	0 - terminal emulation
Values	0 - terminal emulation 1 - programmable
Function	im_receive_field im_receive_input
Notes	Setting the keypad_type parameter is the equivalent of selecting the Configuration Menu, Terminal Menu, and Keypad from the TRAKKER Antares 2400 Menu System and then selecting the keypad type for the terminal. The Simulator can support Western European characters (such as é, ü, and õ) when you choose the programmable keypad. For help configuring the Simulator to support these characters, see “International Characters” in Chapter 2.

label_preamble

Purpose	Lets you add a user-defined string to the front of the label data. The user-defined string is specified in the label_preamble_string parameter.
Default	0 - disabled
Values	0 - disabled 1 - enabled
Function	im_receive_field im_receive_input

label_preamble_string

Purpose	Specifies the user-defined string to be added to the front of the label data if the label_preamble parameter is enabled.
Default	blank
Values	Any ASCII text
Function	im_receive_field im_receive_input

label_postamble

Purpose	Lets you append a user-defined string to the end of the label data. The user-defined string is specified in the label_postamble_string parameter.
Default	0 - disabled
Values	0 - disabled 1 - enabled
Function	im_receive_field im_receive_input

label_postamble_string

Purpose	Specifies the user-defined string that can be appended to the end of the label data if the label_postamble parameter is enabled.
Default	blank
Values	Any ASCII text
Function	im_receive_field im_receive_input

label_time_stamp

Purpose	Specifies whether the current date and time will be appended to the label data.
Default	0 - disabled
Values	0 - disabled 1 - enabled
Function	im_receive_field im_receive_input

label_symbology

Purpose	Specifies the symbology of the last simulated scanned label.
Default	1 – Code 39
Values	0 - Unknown 1 - Code 39 2 - Code 93 3 - Code 49 4 - Interleaved 2 of 5 (I 2 of 5) 5 - Codabar 6 - UPC/EAN 7 - Code 128 8 - Code 16K 9 - Plessey 10 - Code 11 11 - MSI
Function	im_get_label_symbology im_receive_input
Notes	The value of this INI parameter is returned to the application in the symbology parameter of the function calls that use this INI parameter. When testing a Trakker Antares application, you can simulate the act of scanning a label by pressing the key sequence specified in the sim_wand_key parameter. When testing a T2090 application, you can press **Ctrl-G** to simulate scanning a label.

label_symbologyid

Purpose	Specifies the AIM symbology ID of the data entered while simulating bar code input.
Default	None
Values	Enter up to four ASCII characters, such as]A1.
Function	im_get_label_symbologyid
Notes	The value of this INI parameter is returned to the application in the symbologyID parameter of the function calls that use this INI parameter.

network_emulation

Purpose	Specifies whether the network operations are emulated through ASCII data files, which are named in the network_read_file and network_write_file parameters.
Default	1 - enabled
Values	0 - disabled 1 - enabled
Function	Not applicable

network_read_file

Purpose	Names the ASCII text file that contains data to be read by the application as if it were received on the network port (NET).
Default	netread.rcv
Values	Any filename
Function	im_receive_buffer im_receive_input

Notes Test the application on a terminal to ensure that the application is handling the input communications protocols correctly.

The PSK function you use to read data from the RCV file affects how you format the data in the file:

im_receive_buffer Reads a buffer of data each time it is called. The RCV file should contain one or more data strings. Each data string is terminated by a CR/LF character, which indicates the end of the buffer. If there is no CR/LF, the function reads up to 1024 bytes of data. With each subsequent call, im_receive_buffer continues reading data where it left off in the file until it reaches an EOF. If the function is called again after reaching EOF, it starts reading data from the beginning of the file.

im_receive_input Reads a line at a time, similar to im_receive_buffer. However, because im_receive_input accepts input from multiple sources, when the function reaches the EOF, it does not start reading data at the beginning of the file again. This practice allows keyboard, scanner, and wand input.

network_receive_file_return

Purpose	Specifies the value to be returned to your application when you use an im_receive_file function to request that a file be sent to the terminal from the DCS 300.
Default	00H - success
Values	00H - success
	56H - invalid_param
	74H - net_open_error
	75H - net_close_error
	81H - request_failure
	80H - file_open_error
	85H - file_close_error
	83H - receive_failure
	84H - file_write_error
	55H - general_error
	11AH - timed_out_error
	6BH - net_config_error
	77H - net_write_error
	86H - controller_deny
Function	im_receive_file
Notes	The Simulator does not receive or simulate receiving the file specified in the im_receive_file function. If your application opens the file after receiving it, you must make sure that a copy of the file is in your current directory when you run the application through the Simulator. Otherwise, your open command will fail.
	You must test the application on a terminal to make sure that the application handles the communications protocols correctly and that the file is received on the terminal without error.

network_transmit_file_return

Purpose	Specifies the value to be returned to your application when you use an im_transmit_file function to send a file from the terminal to the DCS 300.
Default	00H - success
Values	00H - success 56H - invalid_param 74H - net_open_error 75H - net_close_error 81H - request_failure 80H - file_open_error 85H - file_close_error 83H - receive_failure 84H - file_write_error 55H - general_error 11AH - timed_out_error 6BH - net_config_error 77H - net_write_error 86H - controller_deny
Function	im_transmit_file
Notes	The Simulator does not transmit or simulate transmitting the file specified in the im_transmit_file function. If your application opens the file before transmitting it, you must make sure that a copy of the file is in your current directory when you run the application through the Simulator. Otherwise, your open command will fail. You must test the application on a terminal to make sure that the application handles the communications protocols correctly and that the file is transmitted to the DCS 300 without error

network_write_file

Purpose	Names the ASCII text file that will receive the data that the application writes to the network port.
Default	netwrite.trx
Values	Any filename
Function	im_transmit_buffer im_transmit_file
Notes	Test the application on a terminal to make sure that the application is handling the output communications protocols correctly.

number_scan_line_select

Purpose	Specifies the number of scan rows (1 to 16) of the glyph font to be displayed.
Default	1
Values	1 to 16
Function	im_overlay_setup

optical_input_n

Purpose	Specifies the optical sensor input status while simulating optical sensor input. *n* in this parameter designates the optical sensor channel (1, 2, 3 or 4).
Default	0 - OFF
Values	0 - OFF 1 - ON
Function	im_get_sensor_all im_get_sensor_input im_receive_field im_receive_input
Notes	The value of this INI parameter is returned to the application in the optical sensor status parameter of the function calls that use this INI parameter. When testing an application, you can simulate the act of optical sensor input by pressing the key sequence specified in the sim_optical_key parameter. The optical sensor channel and optical state value parameter specifies the value of the simulated optical sensor input.

portn_read_file

Purpose Names the ASCII text file that contains data to be read by the application as if it were received on a COM port. The *n* in port*n*_read_file is the COM port number (1, 2 or 3).

Default comport*n*.rcv, where *n* is the COM port number (1, 2 or 3)

Values Any filename

Function im_receive_buffer
im_receive_input

Notes Do **not** set this parameter to com*n*.rcv. Your PC will expect data from its own COM*n* port. Test the application on a terminal to make sure that the application is handling the input communications protocols correctly. The PSK function you use to read data from the RCV file affects how you format the data in the file:

im_receive_buffer Reads a buffer of data each time it is called. The RCV file should contain one or more data strings. Each data string is terminated by a CR/LF character, which indicates the end of the buffer. If there is no CR/LF, the function reads up to 1024 bytes of data. With each subsequent call, im_receive_buffer continues reading data where it left off in the file until it reaches an EOF. If the function is called again after reaching EOF, it starts reading data from the beginning of the file.

im_receive_input Reads a line at a time, similar to im_receive_buffer. However, because im_receive_input accepts input from multiple sources, when the function reaches the EOF, it does not start reading data at the beginning of the file again. This practice allows keyboard, scanner, and wand input.

portn_write_file

Purpose Names the ASCII text file that will receive data the application writes to a COM port. The *n* in port*n*_write_file is the COM port number (1, 2 or 3).

Default comport*n*.trx, where *n* is the COM port number (1, 2 or 3)

Values Any filename

Function im_transmit_buffer
im_transmit_file

Notes Do **not** set this parameter to com*n*.trx. Your PC will try to send data to its own COM*n* port. You must test the application on a terminal to make sure that the application is handling the output communications protocols correctly.

scan_port_read_file

Purpose	Identifies the ASCII file that contains data to be read by the application as if it were received on the scanner port.
Default	scanport.rcv
Values	Any filename
Function	im_receive_buffer im_receive_input
Notes	Do **not** set this parameter to com*n*.rcv. Your PC will expect data from its COM*n* port.

Test the application on a terminal to make sure the application is handling the communications protocols correctly.

The PSK function you use to read data from the RCV file affects how you format the data in the file:

im_receive_buffer Reads a buffer of data each time it is called. The RCV file should contain one or more data strings. Each data string is terminated by a CR/LF character, which indicates the end of the buffer. If there is no CR/LF, the function reads up to 1024 bytes of data. With each subsequent call, im_receive_buffer continues reading data where it left off in the file until it reaches an EOF. If the function is called again after reaching EOF, it starts reading data from the beginning of the file.

im_receive_input Reads a line at a time, similar to im_receive_buffer. However, because im_receive_input accepts input from multiple sources, when the function reaches the EOF, it does not start reading data at the beginning of the file again. This practice allows keyboard, scanner, and wand input.

scan_port_write_file

Purpose	Identifies the file that will receive data the application writes to the scanner port.
Default	scanport.trx
Values	Any filename
Function	im_receive_buffer im_receive_input
Notes	Do not set this parameter to com*n*.trx. Your PC will try to send data to its COM*n* port.

Test the application on a terminal to make sure the application is handling the communications protocols correctly.

screen_type_select

Purpose	Specifies the Trakker Antares terminal display type that the Simulator will mimic.
Default	0 - 20x16 (T242X)
Values	0 - 20x16 (T242X) 1 - 80x25 (T2455 LCD display) 2 - 40x25 (T2481/T2486) 3 - 40x4 (T2480/T2485) 4 - 80x25 (T2455 EL (electroluminescent) display) 5 - 16x2 (2460/2461) 6 - 20x16 (2410/2415)
Function	im_get_display_type
Notes	Setting this parameter is equivalent to selecting Configuration Menu, Terminal Menu, and Display from the TRAKKER Antares 2400 Menu System and then setting the display type for the terminal.

serial_port_emulation

Purpose	Specifies whether the serial port operations are simulated.
Default	1 - enabled
Values	0 - disabled 1 - enabled
Function	Not applicable
Notes	For Trakker Antares applications, serial port operations are simulated through ASCII data files, which are named in the port*n*_read_file and port*n*_write_file (where *n* is 1, 2 or 3 to identify the port number).

serial_receive_mode

Purpose Specifies whether the Simulator receives serial data one line at a time (Line mode) or one character at a time (Character mode).

Default 0 - line_mode

Values 0 - line_mode
1 - character_mode

Function im_receive_buffer

Notes In Line mode, the Simulator reads a line of characters till it reaches <CR><LF> and returns from im_receive_buffer with that string. In Character mode, the Simulator reads and returns only one character at a time.

sim_optical_key

Purpose Specifies the key sequence that causes the application to accept keyboard input as if it were optical sensor input.

Default Ctrl-O

Values A key combination that includes one or more control keys (Ctrl, Shift, or Alt) and a character key (A to Z). Sample key combinations include: **Ctrl-B**, **Shift-C**, and **Alt-L**.

Function im_get_sensor_all
im_get_sensor_input
im_receive_input
im_receive_field

Notes To simulate optical sensor input:

1. Press a key sequence, for example, **Ctrl-O**.

   Although you can press the keys simultaneously or sequentially, you may decide to press them sequentially to avoid conflict with Microsoft Windows key code capture.
2. Press 1, 2, 3, or 4 to designate the optical sensor channel.
3. Press one of the status keys, 0 (OFF) or 1 (ON), to end the input.

   If the input sequence is not entered in the right order, it will be reset to the beginning.

sim_wand_key

Purpose	Specifies the key sequence that causes the application to accept subsequent keyboard input as if it were wand input.
Default	**Ctrl–G**
Values	A key combination that includes one or more control keys (Ctrl, Shift, Alt) and a character key (A to Z).
Function	Not applicable

Notes The user presses the key sequence either simultaneously or sequentially, types data that the application accepts as input from a wand, and presses **Enter** to indicate the end of the input.

You can set the sim_wand_key parameter to one or more control keys (Ctrl, Shift, Alt) and a character key (A to Z). For example:

- **Alt–A**
- **Shift–Alt–B**
- **Ctrl–Shift–C**
- **Ctrl–Shift–Alt–D**

video_scan_row_select

Purpose	Specifies the starting row number (0 to 15) of the 16 by 16 area of the display panel.
Default	0
Values	0 to 15
Function	im_overlay_setup

xmodem_receive_file_return

Purpose	Specifies the value to be returned to your application when you use an im_xm_receive_file function to request that a file be sent to the terminal from the host using XModem protocol.
Default	00H - success
Values	00H - success 55H - general_error 56H - invalid_param 74H - connection_open_error 75H - connection_close_error 80H - file_open_error 84H - file_write_error 85H - file_close_error 11AH - timed_out_error 140H - malloc_error 150H - file_transfer_user_abort 151H - file_transfer_lost_connection 152H - file_transfer_receiver_cancelled 226H - invalid_com_port_number
Function	im_xm_receive_file
Notes	The Simulator does not receive or simulate receiving the file specified in the im_xm_receive_file function. If your application opens the file after receiving it, you must make sure that a copy of the file is in your current directory when you run the application with the Simulator. Otherwise, your open command will fail. You must test the application on a terminal to make sure that the application handles the communications protocols correctly and that the file is received on the terminal without error.

xmodem_transmit_file_return

Purpose	Specifies the value to be returned to your application when you use an im_xm_transmit_file function to send a file from the terminal to the host using XModem protocol.
Default	00H - success
Values	00H - success 55H - general_error 56H - invalid_param 74H - connection_open_error 75H - connection_close_error 80H - file_open_error 84H - file_write_error 85H - file_close_error 11AH - timed_out_error 140H - malloc_error 150H - file_transfer_user_abort 151H - file_transfer_lost_connection 153H - file_transfer_transmitter_cancelled 226H - invalid_com_port_number
Function	im_xm_transmit_file
Notes	The Simulator does not transmit or simulate transmitting the file specified in the im_xm_transmit_file function. If your application opens the file before transmitting it, you must make sure that a copy of the file is in your current directory when you run the application with the Simulator. Otherwise, your open command will fail. You must test the application on a terminal to make sure that the application handles the communications protocols correctly and that the file is transmitted to the DCS 300 without error.

xmodem1k_receive_file_return

Purpose	Specifies the value to be returned to your application when you use an im_xm1k_receive_file function to request that a file be sent to the terminal from the host using XModem-1K protocol.
Default	00H - success
Values	00H - success 55H - general_error 56H - invalid_param 74H - connection_open_error 75H - connection_close_error 80H - file_open_error 84H - file_write_error 85H - file_close_error 11AH - timed_out_error 140H - malloc_error 150H - file_transfer_user_abort 151H - file_transfer_lost_connection 152H - file_transfer_receiver_cancelled 226H - invalid_com_port_number
Function	im_xm1k_receive_file
Notes	The Simulator does not receive or simulate receiving the file specified in the im_xm1k_receive_file function. If your application opens the file after receiving it, you must make sure that a copy of the file is in your current directory when you run the application with the Simulator. Otherwise, your open command will fail. You must test the application on a terminal to make sure that the application handles the communications protocols correctly and that the file is received on the terminal without error.

xmodem1k_transmit_file_return

Purpose	Specifies the value to be returned to your application when you use an im_xm1k_transmit_file function to send a file from the terminal to the host using XModem-1K protocol.
Default	00H - success
Values	00H - success
	55H - general_error
	56H - invalid_param
	74H - connection_open_error
	75H - connection_close_error
	80H - file_open_error
	84H - file_write_error
	85H - file_close_error
	11AH - timed_out_error
	140H - malloc_error
	150H - file_transfer_user_abort
	151H - file_transfer_lost_connection
	153H - file_transfer_transmitter_cancelled
	226H - invalid_com_port_number
Function	im_xm1k_transmit_file
Notes	The Simulator does not transmit or simulate transmitting the file specified in the im_xm1k_transmit_file function. If your application opens the file before transmitting it, you must make sure that a copy of the file is in your current directory when you run the application with the Simulator. Otherwise, your open command will fail.
	You must test the application on a terminal to make sure that the application handles the communications protocols correctly and that the file is transmitted to the DCS 300 without error.

ymodem_receive_file_return

Purpose	Specifies the value to be returned to your application when you use an im_ym_receive_file function to request that a file be sent to the terminal from the host using YModem protocol.
Default	00H - success
Values	00H - success
	55H - general_error
	56H - invalid_param
	74H - connection_open_error
	75H - connection_close_error
	80H - file_open_error
	84H - file_write_error
	85H - file_close_error
	11AH - timed_out_error
	140H - malloc_error
	150H - file_transfer_user_abort
	151H - file_transfer_lost_connection
	152H - file_transfer_receiver_cancelled
	226H - invalid_com_port_number
Function	im_ym_receive_file
Notes	The Simulator does not receive or simulate receiving the file specified in the im_ym_receive_file function. If your application opens the file after receiving it, you must make sure that a copy of the file is in your current directory when you run the application with the Simulator. Otherwise, your open command will fail.
	You must test the application on a terminal to make sure that the application handles the communications protocols correctly and that the file is received on the terminal without error.

ymodem_transmit_file_return

Purpose	Specifies the value to be returned to your application when you use an im_ym_transmit_file function to send a file from the terminal to the host using YModem protocol.
Default	00H - success
Values	00H - success 55H - general_error 56H - invalid_param 74H - connection_open_error 75H - connection_close_error 80H - file_open_error 84H - file_write_error 85H - file_close_error 11AH - timed_out_error 140H - malloc_error 150H - file_transfer_user_abort 151H - file_transfer_lost_connection 153H - file_transfer_transmitter_cancelled 226H - invalid_com_port_number
Function	im_ym_transmit_file
Notes	The Simulator does not transmit or simulate transmitting the file specified in the im_ym_transmit_file function. If your application opens the file before transmitting it, you must make sure that a copy of the file is in your current directory when you run the application with the Simulator. Otherwise, your open command will fail. You must test the application on a terminal to make sure that the application handles the communications protocols correctly and that the file is transmitted to the DCS 300 without error.

4

Customizing INI Files With the Editor

This chapter introduces the Simulator Editor and describes how to use it to customize the initialization (INI) file for the Trakker Antares Application Simulator.

Introduction to the Simulator Editor

The Simulator Editor is a Windows-based tool for viewing and setting the parameters stored in the IMT24SIM.INI file. The INI file contains parameters that specify how certain features are simulated. For descriptions of the parameters, see Chapter 3, "INI File Parameter Descriptions."

Note: Because IMT24SIM.INI is a text file, you can use any text editor to change the values of the parameters. You do not need to use the Simulator Editor.

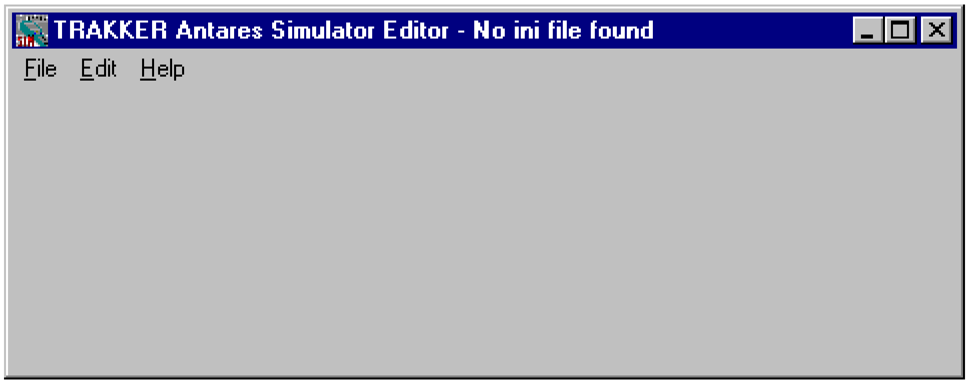
The Editor can be used with Intermec's two application simulators:

- Trakker Antares Application Simulator, which simulates applications for the Trakker Antares family of terminals, including the T2425 hand-held terminal and the T248X stationary terminal.
- Trakker T2090 Application Simulator, which simulates applications for the T2090 computer.

Starting the Simulator Editor

Start the Simulator Editor from the Trakker Antares Development Tools group on the Windows desktop or on the Start menu.

The Editor window appears:



No INI File Found

When you start the Editor, it attempts to open the default INI file, IMT24SIM.INI, in the working directory. If the Editor does not find the IMT24SIM.INI file in the working directory, the message, No ini file found, is displayed in the title bar of the window.

You can open IMT24SIM.INI or another INI file by choosing Open from the File menu. Or you can exit the Editor, copy IMT24SIM.INI to the working directory, and restart the Editor.

Introduction to the User Interface

This utility conforms to standard Microsoft Windows conventions, so it should be easy to familiarize yourself with the user interface.

Menu Bar

The main menu bar contains the File, Edit, and Help commands:

File Lets you open, create, and save INI files. Also lets you restore the parameters to their defaults, print INI files, and exit the Editor.

Edit Lets you select a category of parameters (such as Communications and Labels), causing the corresponding tab section to become active in the Editor window.

Help For details, see the "Online Help" section later in this chapter.

Tabs

The parameters are divided into categories, and each category has its own tab in the Editor window. To view the parameters in each category, click the corresponding tab or choose the category name from the Edit menu.

You set the value for each parameter by

- selecting buttons.
- checking boxes.
- typing into the fields.
- choosing items from drop-down lists.

You can set reset a parameter to its default value by clicking the Default button next to the parameter. You can reset all parameters to their defaults by choosing Restore defaults from the File menu.

Online Help

You can access the online help from the Help menu on the Editor's menu bar. The online help contains all the information in this chapter and describes how to use the Editor with both application simulators.

The online help also contains the parameter descriptions from Chapter 3. If you click on any parameter's field, radio button, or checkbox, you can press **F1** to display the parameter's description.

Creating a New INI File

You can create and customize new INI files with the Editor. Each new file is a duplicate of the default INI file with all the parameters set to their default values.

To create a new INI file

1. From the File menu, choose New.
2. The Select File Type dialog box appears. Select the T24xx file type and click the OK button. The Editor creates an untitled file with its parameters set to their defaults.
3. Customize the parameters if necessary.
4. From the File menu, choose Save.
5. Specify a file name with the INI extension. If necessary, select the path for the new file. Click the Save button.
6. Click the OK button when the Editor prompts you to save all the values in the file.

Opening an Existing INI File

You can open an existing INI file to view, edit, or print the file.

To open an existing INI file

- From the bottom of the File menu, choose the INI file from the list of five most recently opened files.

Or,

1. From the File menu, choose Open. The Open dialog box appears.
2. Select the file name. If necessary, you can browse for the file.
3. Click the Open button. The file opens, you return to the main menu, and the filename is displayed in the title bar. You can begin customizing the file.

Setting INI Parameters

The INI parameters are divided into functional categories, and each category is included on a tab in the Editor window. For descriptions of the INI parameters, see Chapter 3. For help customizing the parameters, see the next procedure.

To set a parameter

1. Click the tab to choose the type of parameter you want to edit. The tab you select moves forward in the Editor window.
2. To set the value for each parameter, select the button, type in the field, or choose an item from the drop-down list.

   For help deciding which value to set for each parameter, see Chapter 3.

Saving Changes

You can save the changes to an INI file

- into the current INI file and continue working.
- into the current INI file when you exit the Editor.
- into a new INI file.
- into an existing INI file.

Note: The current file is the file that is currently open. The current file name is displayed in the title bar of the Editor window.

To save the changes into the current INI file and continue working

1. From the File menu, choose Save. The Editor saves the file and displays this message:

   `Saved all values in` *`file`*`.`
2. Click the OK button.

To save the changes into the current INI file when you exit the Editor

1. From the File menu, choose Exit. If you made changes to the INI file that you have not saved yet, the Editor displays this message:

   ```
   The filename file has been changed.
   Do you want to save the changes?
   ```

2. Click the Yes button. The changes are saved into the current INI file, the Editor shuts down, and you return to the Windows desktop.

To save the changes into a new INI file

1. From the File menu, choose Save As. The Save As dialog box appears.
2. In the File name field, type the new filename. If necessary, select a directory from the Directories list box.
3. Click the Save button. The Editor creates the new file, saves the changes, and displays this message:

   ```
   Saved all values in file.
   ```

4. Click the OK button.

To save the changes into an existing INI file

1. From the File menu, choose Save As. The Save As dialog box appears.
2. Select the name of the file where you want to save the changes. You may need to browse for the correct directory.
3. Click the Save button. The Editor displays this message:

   ```
   This file already exists. Replace existing file?
   ```

4. Click the Yes button. The Editor saves the changes into the file and displays this message:

   ```
   Saved all values in file.
   ```

5. Click the OK button.

Discarding Changes

You can discard changes when you exit the Editor.

To discard the changes

1. From the File menu, choose Close. If you made changes to the INI file that you have not saved yet, the Editor displays a message similar to this one:

   ```
   The filename file has been changed.
   Do you want to save the changes?
   ```

2. Click the No button. The file is closed and the changes are not saved.

Or,

1. From the File menu, choose Exit. If you made changes to the INI file that you have not saved yet, the Editor displays a message similar to this one:

   ```
   The filename file has been changed.
   Do you want to save the changes?
   ```

2. Click the No button. The changes are not saved, the Editor shuts down, and you return to your Windows desktop.

Restoring the Default Values

You can reset all the parameters in the current INI file to their default settings at any time while the Editor is running.

To restore one parameter to its default

- Click the Default button next to the parameter.

To restore all parameters to their defaults

1. From the File menu, choose Restore Defaults. The Editor displays this message:

   ```
   All parameters will be reset to their default values. Do you
   want to proceed?
   ```

2. Click the Yes button. The Editor resets the parameters to their defaults and displays the message:

   ```
   All defaults have been restored.
   ```

3. Click the OK button.

Printing INI Files

You can print INI files from the Editor or with any ASCII text editor. Printing INI files is a good way to keep track of contents of the INI files, especially if you are using multiple files.

Note: If you have not set up the printer for the Editor, choose Print Setup from the File menu and select the printer options as you would for any Windows application.

To print an INI file

1. From the File menu, choose Print.
2. Click the OK button at the Print dialog box.

Exiting the Simulator Editor

When you exit the Editor, you shut down the Editor and close the current INI file. If you changed the current file and have not saved those changes yet, the Editor prompts you to save or discard the changes.

To exit the Editor

- From the File menu, choose Exit.

  If you saved all changes to the current INI file, the Editor simply shuts down and you return to your Windows desktop.

  If you made changes to the INI file that you have not saved yet, the Editor prompts you to save or discard the changes.

  ```
  The filename file has been changed.
  Do you want to save the changes?
  ```

  Click the Yes button if you want to save the changes. Click the No button if you do not want to save the changes.

5
Troubleshooting

Troubleshooting 5

This chapter describes problems you may encounter while using the Simulator and error messages you may see while using the Editor.

Problems Operating the Simulator

This section describes problems you may see when using the Application Simulator.

Linking to the Wrong Library

Both Intermec and Microsoft provide you with the LLIBCA.LIB library. If you are creating .BIN applications, you must link to the Intermec LLIBCA.LIB library. If you link to the Microsoft LLIBCA.LIB library, an application containing an erroneous input combination will fail on the Trakker Antares terminal, but will not be detected by the Simulator.

If you are creating DOS .EXE applications for terminals running ROM-DOS, you must link to the LLIBCE.LIB library.

Problems Simulating Bar Code Input With a Wedge

If you are having difficulty using an Intermec wedge to provide bar code input while you run a Trakker Antares application, you may have set the wedge preamble incorrectly.

For help setting the preamble to match the value of the sim_wand_key parameter, see “Bar Code Input” in Chapter 2.

Problems Displaying International Characters

If Western European characters (such as é, ü, and õ) are not displayed when you run a Trakker Antares application, make sure you are complying with these guidelines:

- You set the keypad_type INI parameter to 1 for the programmable keypad.
- You set your PC’s display to Code Page 850.
- You are running the Simulator in a true DOS window, not a DOS screen.

For help, see the commands listed in “International Characters” in Chapter 2.

Only the Communications and Labels Tabs Appear

The Simulator Editor displays only the Communications and Labels tabs when you edit an INI file for the T2090 Application Simulator instead of the Trakker Antares Application Simulator.

To correct the problem, open an existing Trakker Antares INI file. Or you can select New from the File menu and choose T24xx from the Select Type dialog box. The rest of the tabs should appear.

Error and Status Messages

This table describes the error and status messages you may see when using or installing the Simulator TSR or Editor. Follow the instructions in the Suggested Action column to recover from the error.

Message	Description	Suggested Action
All parameters will be reset to their default values. Do you want to proceed?	You have chosen Restore Defaults from the File menu. All parameters in the INI file will be reset to their default values.	Click the Yes button to restore the defaults, or click the No button to cancel the request.
All defaults have been restored.	This message confirms that all the parameters in the INI file have been retored to their default values.	Click the OK button.
Cannot change to directory.	You specified a directory name that is invalid or does not exist.	Make sure you specified the directory name correctly. Make sure the directory exists. Then try again.
Cannot open file *name*.	The Simulator TSR could not find either the INI file supplied on the command line or the default IMT24SIM.INI file. The TSR was loaded with the default INI values.	Make sure you specified the filename correctly, and make sure the file exists. Make sure the IMT24SIM.INI file is in the directory specified by the IMT24SIM environment variable.
Cannot open TSR communication file.	You selected the Update Simulator command from the File menu, but the Editor could not create the EDITTSR.TMP file required for loading new parameter values. The Simulator will not be updated with the new parameter values. Your PC may not have enough disk space available to create the EDITTSR.TMP file.	Check how much disk space is free and delete unnecessary files.
File name must be imt209sm.ini.	The Simulator Editor thinks you are editing an INI file for the T2090 Application Simulator.	Open a new INI file and make sure you choose T24xx at the Select Type dialog box to indicate that you are using the Trakker Antares Application Simulator.
Filename file not found. Please verify the correct file name was given.	You specified a file that the application cannot find.	Make sure you entered the file name correctly. Make sure the file exists.

Message	Description	Suggested Action
Incorrect DOS version (need 3.0 or greater).	You attempted to load the Simulator TSR, but your version of DOS is not correct.	Upgrade DOS to 3.0 or greater, or run the Simulator TSR on another PC with the correct version of DOS.
Initialization file does not exist.	You tried to load the TSR, and DOS could not locate the INI file.	Make sure the INI file exists and is mentioned in your AUTOEXEC.BAT file as follows: the PATH statement should include the directory, or the IMT24SIM environment variable should point to the directory.
Initialization file invalid.	You tried to load the TSR with an invalid INI file.	Verify the file's type and contents. Recreate the file with the Editor.
Insufficient memory.	You tried to load the Simulator TSR without sufficient conventional memory. The Simulator requires 50K of conventional memory.	Free some memory and reissue the command to load the Simulator.
No INI File Found	The Simulator Editor did not find the default INI file for the Trakker Antares Application Simulator in the working directory.	Open an INI file by choosing Open from the File menu. Or you can choose from the list of recently opened files at the bottom of the File menu.
Parameter *name* is invalid.	The parameter is not valid for the Trakker Antares Application Simulator.	Delete the parameter from the INI file and restart the application.
Save all values in *filename*.	This message confirms that all the parameters will be saved in the specified file.	Click the OK button.
Simulator has been loaded with *filename*.	You chose the Load Simulator command from the File menu, and the Editor loaded the changes in the specified INI file.	No action required.
Simulator is already loaded with *filename*.INI.	You attempted to load the Simulator TSR when it was already resident in memory. Only one copy of the Simulator TSR can be resident in memory at a time.	No action required.
The Simulator is not loaded. Exit the editor, shut down Windows, and start the TSR.	You chose the Load Simulator command from the File menu, but the Simulator TSR is not currently running.	Save the changes to the INI file, exit the Editor, exit Windows, and start the TSR with a command that also loads the INI file.
The *filename* file has been changed. Do you want to save the changes?	You have made changes that have not been saved yet.	Click the Yes button to save the changes, or click the No button to discard the changes.

Message	Description	Suggested Action
TSR memory corruption - reboot is required!	You attempted to remove the Simulator TSR from memory and DOS memory became corrupted.	Reboot your PC.
Unable to copy or decompress file: *filename*	During the installation, SETUP.EXE was unable to copy or decompress a file.	Choose OK. SETUP.EXE may terminate immediately or continue.
	Even if the installation appears to complete successfully, the software may not be fully installed. You must take corrective action and run SETUP.EXE again.	When SETUP.EXE completes, delete IMT24SIM.DEF from the INTERMEC\IMT24\SIM directory. Run SETUP.EXE again.
	This problem usually occurs if you reinstall the Simulator and the IMT24SIM.DEF file is read only. Because SETUP.EXE cannot overwrite the IMT24SIM.DEF file, the installation fails.	If you still encounter problems, contact your Intermec representative.
Windows is Active! Shut down Windows before unloading the Simulator.	You loaded the Simulator TSR from DOS before you started Windows. Later you attempted to unload the Simulator TSR from Windows.	Exit Windows and unload the TSR from DOS.

A

Adding the Simulator to the Microsoft Visual C/C++ Tools Menu

This appendix shows how to add two Application Simulator commands to the Tools menu in Microsoft Visual C/C++.

Adding the Simulator to the Tools Menu

If Microsoft Visual C/C++ was installed before you installed the PSK and Application Simulator on your PC, two commands were automatically added to the Tools menu of Microsoft Visual C/C++:

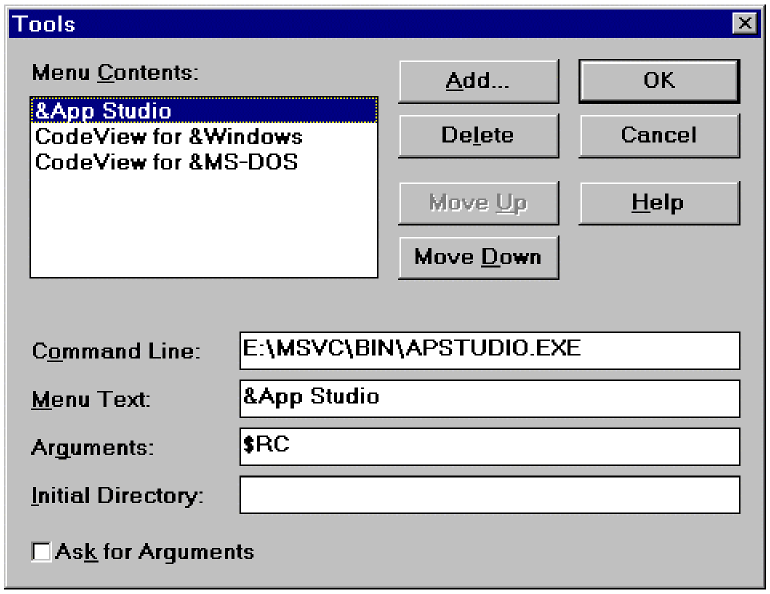
Simulation for Trakker Loads the Simulator TSR into memory.

Codeview for Trakker Loads the Simulator TSR into memory and starts Microsoft Codeview for DOS.

If these commands were not added to the Tools menu, you can add them manually at any time by following the instructions in this appendix.

To add the Simulator to the Tools menu

1. Start Microsoft Visual C/C++, and select Tools from the Options menu. The Tools dialog box appears.


Menu Contents:
&App Studio
CodeView for &Windows
CodeView for &MS-DOS

Command Line: E:\MSVC\BIN\APSTUDIO.EXE
Menu Text: &App Studio
Arguments: $RC
Initial Directory:
☐ Ask for Arguments

2. Choose Add to access the Add Tool dialog box.
3. Select the IMCV.BAT file and choose OK to return to the Tools dialog box. The default location for this file is C:\INTERMEC\IMT24\SIM.

4. Replace the contents of the Menu Text field with:

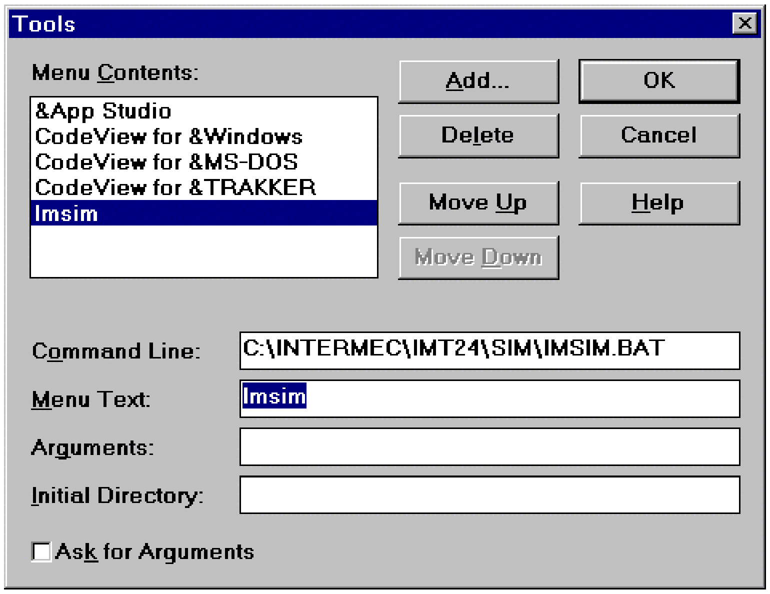
   `CodeView for &Trakker`

   Also, enter this information to the Arguments field:

   `$Target`

5. Choose Add to add a new item to the Tools menu. The Add Tool dialog box appears.

6. Select the IMSIM.BAT file and choose OK to return to the Tools dialog box. The default location for this file is C:\INTERMEC\IMT24\SIM.



Menu Contents:

&App Studio
CodeView for &Windows
CodeView for &MS-DOS
CodeView for &TRAKKER
Imsim

Command Line: C:\INTERMEC\IMT24\SIM\IMSIM.BAT
Menu Text: Imsim
Arguments:
Initial Directory:
☐ Ask for Arguments

7. Replace the contents of the Menu Text field with:

   `&Simulation for Trakker`

   Also, enter this information to the Arguments field:

   `$Target`

8. Choose OK.

9. From the main menu, choose Tools. The two new options appear on the menu.

I

Index