Touch System Diagnostics (CTDIAG) User's Guide

Carroll

TOUCH

Touch System Diagnostics (CTDIAG) User's Guide

Carroll TOUCH

an AMP company

April 1998

Part #: 2950-0012-3.30N

Copyright

Trademarks

Disclaimer

Carroll Touch has a policy of continually improving products as new technology becomes available. Carroll Touch reserves the right to make changes and improvements to the specifications of this equipment at any time without notice.

Carroll Touch has made every attempt to ensure that the information in this document is accurate and complete. Carroll Touch assumes no liability for any damages that result from the use of this manual or the equipment it documents. Carroll Touch reserves the right to make changes to this document at any time without notice.

Table of Contents

List of Figures

List of Tables

Welcome

As computers become a part of daily life, a technology that makes them easier to use has become a necessity. Carroll Touch provides the solution through the power of touch.

Because pointing or touching is a natural means of indicating choice, touch systems are ideal for selection-based applications where easy-to-follow menus guide a user through a series of steps or choices. Touch is well accepted by the casual user because its simple, natural interface hides the complexity of computer systems.

Thank you for your purchase of a Carroll Touch product and welcome to Carroll Touch.

Purpose

This *Touch System Diagnostics (CTDIAG) User's Guide* contains all the information necessary to integrate, test, and troubleshoot Carroll Touch touch systems. While CTDIAG can be used with guided acoustic wave touch systems, you may find another diagnostics program, GWDIAG, is more useful for that touch technology; see the *Guided Wave Diagnostics User's Guide* for additional information.

A companion document, the *Touch System Programmer's Guide*, contains detailed programming information on the use and functions of the Smart-Frame Protocols and the Touch Application Program Interface (TAPI) driver.

Audience

This guide is designed for the advanced user, programmer, or software engineer who is installing and integrating Carroll Touch touch systems with host computers or who needs to troubleshoot or debug touch systems.

Organization

Chapter 1, “Introduction,” gives an overview of CTDIAG, its use, and its interface.

Chapter 2, “Starting CTDIAG,” contains the information you need to install the Touch System Diagnostics software.

Chapter 3, “Configuring a Touch System,” describes how to set up your touch system, either automatically or manually, before using it.

Chapter 4, “Setting Communication Parameters,” explains how to set the communication parameters.

Chapter 5, “Testing Touch System Hardware and Software,” describes how to use the scan reporting mode to check hardware functionality and how to use the coordinate (x, y) reporting mode to check software functionality.

Chapter 6, “Touch System Information,” explains how to display a summary of current touch system status.

Chapter 7, “Debug,” explains how to check the operation of Smart-Frame Protocol commands and reports.

Chapter 8, “TAPI Debug,” discusses how to check the operation of the TAPI driver.

Chapter 9, “Command Line Options,” describes how to set the CTDIAG configuration, communication, and autodetection parameters using the command line, rather than menus.

Appendix A, “Error Messages,” lists and explains CTDIAG error messages.

Appendix B, “The CTDIAG.CFG File,” explains the contents and use of the CTDIAG configuration file.

Conventions

For clarity, this guide uses certain conventions to visually distinguish different types of information. The conventions are:

- **Bold** is used to emphasize a word or phrase, including definitions of important concepts.
- SMALL CAPITAL LETTERS (such as SPACE or ENTER) indicate a key on the keyboard.
- `Courier` font indicates file names, directory names, messages displayed by the computer, parameters in command lines, and information to be typed by the user.
- *Italics* indicate a command, sequence, function name, or mode (such as *Debug Mode*).
- Reports (such as the Touch State Report) and menus (such as the Configuration Menu) use initial capital letters.
- “Select” an option means to use the arrow keys to highlight that option and press ENTER, or to type the mnemonic for the option.
- *`Courier italic`* font indicates a variable in a command line for which you must substitute a value.
- Hexadecimal numbers are identified with capital H; for example, 1BH is the hexadecimal value 1B.
- Information of particular importance or actions that may have undesirable results if performed improperly are included under the headings ***Note*** and ***Caution***.

1 Introduction

The Touch System Diagnostics software (CTDIAG) is a rich and multi-featured software package that will verify that a touch system has been installed correctly. It is also a tool that will help you diagnose and correct any problems you encounter in touch system hardware or software.

This chapter gives an overview of CTDIAG, its use, and its interface. It discusses the following topics:

- Purpose.
- Features.
- Quick Test.
- Debugging.
- Using the Menus.

Purpose

CTDIAG is a menu-driven diagnostic program that you can use to verify touch system installation, troubleshoot touch system operations, and debug. CTDIAG works with any Carroll Touch touch system that uses a hardware-based controller (HBC), a software-based controller (SBC), or an RS-232 controller. CTDIAG also works with any other Carroll Touch touch system that connects to a host computer through an RS-232 serial interface and that uses the Carroll Touch Smart-Frame Protocol.

Even if you do not need to use CTDIAG for error detection, you will find it very useful to work through the program's features and options to enhance your understanding of the touch system.

Features

CTDIAG offers a wide range of hardware and software configuration options, including support for:

- Both scanning infrared and guided acoustic wave touch technologies.
- Software-based, hardware-based, and serial (RS-232) controllers.
- Carroll Touch proprietary Smart-Frame Protocol (SFP).
- Direct, polling, and interrupt-based interface methods.

Additional software features include:

- Touch reporting methods optimized for testing hardware and software.
- Touch modes that determine when touch coordinates are reported.
- Autodetection of touch system hardware and software.
- Reporting of system status.
- Command line input as an alternative to menu input.
- Debug options for SFP and the Touch Application Program Interface (TAPI).

Quick Test

This section describes a quick method to verify touch system installation and troubleshoot touch systems using an HBC or serial (RS-232) controller. Before starting, you must be in the DOS environment.

- If you are in Windows™ 3.x, exit Windows completely to the DOS prompt. Do not use the MS-DOS® Prompt icon in Windows.

- If you are in Windows 95®, restart your computer in MS-DOS mode.
- If you installed Windows 95 in its own directory and kept the previous version of MS-DOS and Windows 3.x, restart your computer in the previous version of MS-DOS.
- If you are running under Windows NT™, reboot your computer using a DOS bootable disk. Or, if you have DOS installed in another partition, reboot your computer using that partition.

Take these steps:

1. Change directories to `c:\ctouch\ctdos\ctdiag` (or other directory that you specified during installation). (See Chapter 2 for details.)
2. Type `ctdiag` to start CTDIAG.
   If the Test Menu appears, your touch system is configured correctly and all communication parameters are correct. Skip steps 3 - 5 and go to step 6.
   If the Test Configuration Menu appears, you need to configure your touch system and double check the communication parameters. Continue to step 3.
3. From the Test Configuration Menu, choose HBC Controller or Serial (RS-232 Controller), as appropriate, then press ENTER to go to the Test Menu. (See Chapter 3 for details.)
   If you are using an HBC, skip step 4 and go to step 5.
   If you are using an RS-232, continue to step 4.
4. Check that the correct COM port is displayed in the Status line at the bottom right of the screen. (See Chapter 4 for details.)
   If the correct COM port is displayed, go to step 5.
   If the correct COM port is not displayed:
   - From the Test Menu, choose Set Communications Parameters.
   - From the RS-232 Communication Parameters Menu, choose the correct Comm Port with the F1, F2, F3, or F4 function key, or customize a Comm Port using F5 and F6.
   - Press ENTER to return to the Test Menu. Continue to step 5.
5. From the Test Menu, choose System Info. Record any errors that are reported at the top of the display, then press the ESC key to return to the Test Menu. (See Chapter 6 for details.)
6. From the Test Menu, choose Touch Coordinates. If the touch system is working properly, the Touch Coordinates screen appears. As you touch the screen and slide your finger across it, a small cursor tracks your finger movements and the x, y coordinates are displayed at the top of the screen. (See Chapter 5 for details.)
7. Exit CTDIAG and use your touch screen.

If errors are reported on the System Information display, if the System Information display or the Touch Coordinates screen does not appear, or if you see `Touch System Init Failed` at the bottom of the screen, recheck all cable connections, reboot the computer and touch system, then retest. If your system does not work upon retest, you may need to modify the other configuration options or communication parameters. Review the information in Chapters 3 and 4 for details.

Debugging

CTDIAG also offers you the option to debug touch system problems by examining the communication between the computer and the touch screen controller, with or without the TAPI driver. At this level of communication, you are examining in detail the commands in the Smart-Frame Protocol and the functions of the TAPI driver.

Debug Mode allows you to send and receive Smart-Frame Protocol reports and commands to determine if the information is being transmitted and received properly. (See Chapter 7 for details.)

TAPI Debug Mode allows you to send and receive Smart-Frame Protocol reports and commands through the TAPI driver to determine if the TAPI driver is functioning correctly. (See Chapter 8 for details.)

Using the Menus

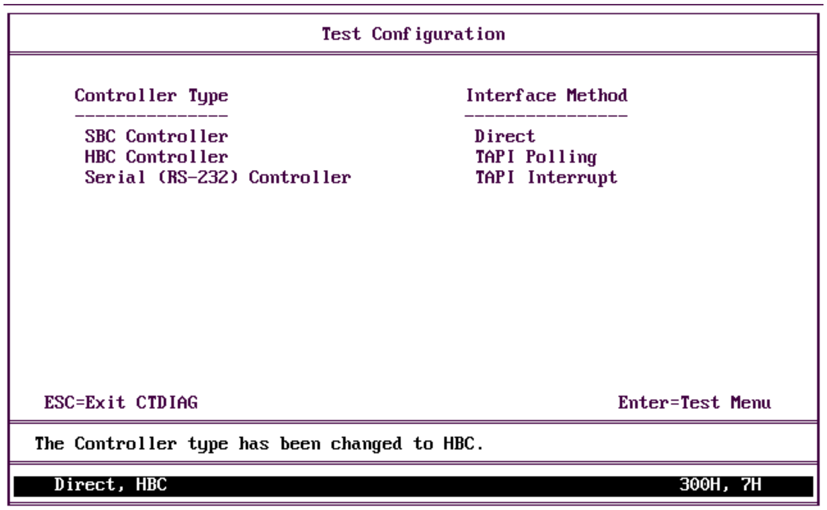
Some CTDIAG menus offer a mnemonic method to make a menu selection. If the option on a CTDIAG menu contains a highlighted letter, or **mnemonic**, you can select that option by simply typing that letter without pressing ENTER. For example, some CTDIAG menus accept the letter E as the exit command.

On some CTDIAG menus, you can use the up and down arrow keys on the keyboard to highlight a menu option. Once highlighted, select that option by pressing ENTER.

Pressing ESC on any menu returns you to the previous menu.

At the bottom of every CTDIAG menu are two lines, as shown in the example menu in Figure 1-1.

- The Message line displays prompts, error messages, or brief help descriptions of the selected menu option. In Figure 1-1, for example, the Message line confirms that the type of controller was changed to HBC.



```
                         Test Configuration

    Controller Type                        Interface Method
    ---------------                        ----------------
     SBC Controller                         Direct
     HBC Controller                         TAPI Polling
     Serial (RS-232) Controller             TAPI Interrupt






 ESC=Exit CTDIAG                                       Enter=Test Menu

 The Controller type has been changed to HBC.

    Direct, HBC                                              300H, 7H
```

Figure 1-1. Example CTDIAG Menu

- The Status line summarizes the configuration and communication parameters for the touch system currently in use. In Figure 1-1, the left portion of the Status line reports a configuration of the direct interface method and a hardware-based controller. The right portion indicates the I/O address and hardware interrupt request number.

If the example were for a serial (RS-232) controller, the right portion of the Status line would list these communications parameters: comm port, I/O address, hardware interrupt request number (IRQ), baud rate, parity, data bits (fixed at 8), and stop bit. These parameters are defined and described in Chapter 4, “Setting Communication Parameters.”

If a TAPI driver is loaded, the right portion of the Status line indicates the software interrupt where the driver is installed.

2 Starting CTDIAG

This chapter contains the information you need to install the Touch System Diagnostics software. It discusses the following topics:

- Installing the Controller Hardware.
- System Requirements.
- CTDIAG Files.
- Installing CTDIAG.
- Starting CTDIAG.
- Autodetection.

Installing the Controller Hardware

Before using or even installing CTDIAG, you should first install the touch system hardware. Each type of Carroll Touch controller comes with its own installation instructions. Once hardware installation is complete, double-check:

- the SBC or HBC to see that the modular digital interface (MDI) cable of the touch frame is correctly connected to the touch controller.
- the serial (RS-232) controller to see that the MDI cable of the touch frame, power supply, and communication cables are all correctly connected to the RS-232 controller.

Caution

Follow the installation instructions carefully; otherwise, you may damage the controller or touch frame!

System Requirements

CTDIAG can be used to check the touch system interface with any IBM PC/AT/XT or compatible (clone) computer that has an 8-, 16-, or 32-bit bus and that uses the MS-DOS or PC-DOS (version 2.1 or above) operating system.

CTDIAG Files

All files associated with CTDIAG are included on the Carroll Touch Driver Installation disk. The various CTDIAG files are listed in Table 2-1.

Table 2-1. CTDIAG Files

File Name	**Contents**
`CTDIAG.CFG`	CTDIAG configuration file, which contains the autodetection search list.
`CTDIAG.EXE`	Touch System Diagnostics program.
`READ.ME`	Additional information.

Installing CTDIAG

If you are installing the Carroll Touch driver for Windows 3.1 or MS-DOS, CTDIAG is automatically installed when you install the driver. If you are installing the Carroll Touch driver for Windows95 or Windows NT, you will need to run `INSTALL.EXE` to install CTDIAG. See the READ ME file on your installation disk for details.

CTDIAG files are automatically placed in the `c:\ctouch\ctdos\ctdiag` directory, unless you specify a directory other than the default (`c:\ctouch)` during installation.

If you receive a separate floppy disk containing CTDIAG, you will need to either copy or extract files to your hard disk, depending upon the floppy's contents.

Copying Files

If your floppy disk contains the files listed in Table 2-1, take the following steps:

1. At the DOS prompt, create the `c:\ctouch\ctdos\ctdiag` directory.
2. Use the DOS *copy* command to copy all files from the floppy disk to that directory.

Extracting Files

If your floppy disk contains a file entitled `CTDIAG`*`xx`*`.EXE` (for example, `CTDIAG33.EXE` for version 3.3 of CTDIAG`)`, you have CTDIAG in a self-extracting, zip file format. Take the following steps:

1. At the DOS prompt, create the `c:\ctouch\ctdos\ctdiag` directory.
2. Use the DOS *copy* command to copy `CTDIAG`*`xx`*`.EXE` from the floppy disk to that directory.
3. Change to the newly-created directory.
4. Type `CTDIAG`*`xx`*`.EXE` to execute the zip file. A series of messages appears as the file is decompressed. All CTDIAG files are placed in the directory.
5. Delete the `CTDIAG`*`xx`*`.EXE` file, since it is no longer needed.

If you download CTDIAG from the Carroll Touch web site, it is in this self-extracting format.

Starting CTDIAG

To start CTDIAG, take the following steps:

1. In DOS, change to the directory containing CTDIAG (ordinarily, this is `c:\ctouch\ctdos\ctdiag`).

   ### *Note*

   You must run CTDIAG directly from the DOS environment. If you are running Windows 3.x, exit to the DOS prompt. In Windows 95, restart the computer in MS-DOS mode. (Do not use the MS-DOS Prompt icon!) Under Windows NT, you must reboot into a partition containing DOS or reboot using a DOS bootable disk.

2. Type `ctdiag` and press ENTER. CTDIAG then starts autodetection.

   ### *Note*

   A number of options and parameters may be specified on the CTDIAG command line, rather than using the menu system described in Chapters 3 through 8; see Chapter 9 for details.

Autodetection

Autodetection tries to automatically identify the type of touch system on your computer system to save you the effort of doing so.

If autodetection is successful, CTDIAG displays a summary of the current configuration and the message `Autodetection complete.` The Test Menu (Chapter 4) then appears so you can move on to touch system testing (Chapter 5), information (Chapter 6), and/or debugging (Chapters 7 and 8).

If autodetection is not successful, the message `Autodetection failed` appears, followed by the Test Configuration Menu; the Status line displays the default configuration and communication parameters. Set the configuration parameters (Chapter 3) before moving to the Test Menu (Chapter 4). If needed, you can also set the communication parameters by using the Set Communication Parameters option on the Test Menu (Chapter 4). Then, you can move on to touch system testing (Chapter 5), information (Chapter 6), and/or debugging (Chapters 7 and 8).

For details on autodetection, see Appendix B, “The CTDIAG.CFG File.”

3 *Configuring a Touch System*

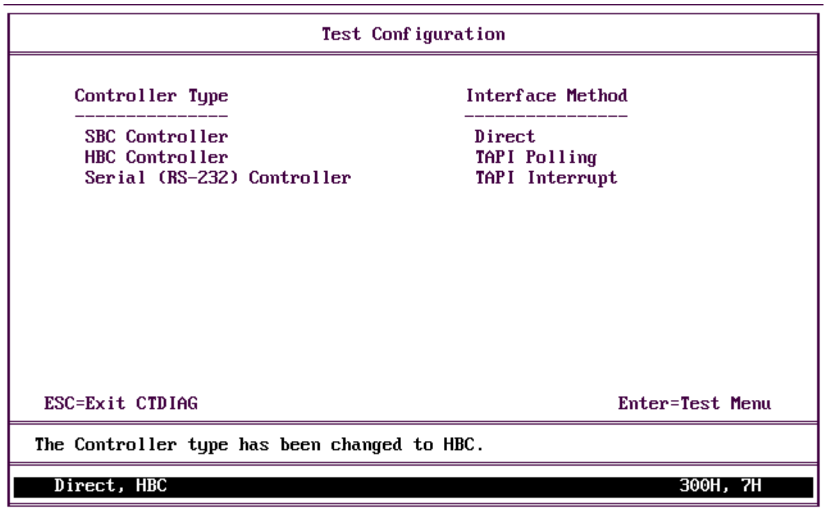
The CTDIAG autodetection feature attempts to automatically determine the type of touch system installed on your computer. If autodetection fails, CTDIAG offers a series of menus that lets you define the configuration of the touch system.

This chapter discusses the following topics:

- Controller Type.
- Interface Method.

You may also use the CTDIAG command line to set the configuration options and communication parameters, as explained in Chapter 9, “Command Line Options.”

If autodetection does not successfully identify the touch system on your computer, the Test Configuration Menu, shown in Figure 3-1, appears. You will need to configure your touch system, which simply means defining what combination of hardware and software is being used



```
                         Test Configuration

   Controller Type                       Interface Method
   ---------------                       ----------------
    SBC Controller                        Direct
    HBC Controller                        TAPI Polling
    Serial (RS-232) Controller            TAPI Interrupt

 ESC=Exit CTDIAG                                   Enter=Test Menu

The Controller type has been changed to HBC.

  Direct, HBC                                           300H, 7H
```

Figure 3-1. Test Configuration Menu

Controller Type

CTDIAG supports the following types of touch controllers:

- Software-based controller (SBC).
- Hardware-based controller (HBC).
- Serial (RS-232) controller.

Note

The SBC requires the TAPI polling or TAPI interrupt interface method; it does not function under the direct interface method.

Use the mnemonic or arrow key to select the type of controller to use; note that it appears in the status line at the bottom of the screen. The default controller is serial (RS-232).

If you need to change the communication parameters associated with the SBC or HBC (interrupt number, I/O address) or serial controller (comm port, baud rate, parity, stop bits), see Chapter 4, “Setting Communication Parameters.”

Interface Method

You may choose to interact with the touch system software using one of three interface methods: direct, TAPI polling, or TAPI interrupt.

Direct is the usual method of interacting with the touch system for serial and hardware-based controllers. A touch on the screen interrupts the infrared beams or guided waves; the point of interruption is then interpreted and the corresponding software action is taken.

Note

The SBC cannot use the direct interface method; it must operate using either TAPI polling or TAPI interrupt.

The Touch Application Program Interface (TAPI) is a set of software functions that lets an application communicate with the touch system, bypassing any direct interface to the hardware.

Caution

Use of the TAPI driver, as well as the TAPI polling and TAPI interrupt interface methods, requires experience with programming issues. See the *Touch System Programmer's Guide* for additional information on TAPI and on TAPI drivers.

Before using either the TAPI polling or TAPI interrupt interface method, you must first install the appropriate TAPI driver for your controller. For information on loading the TAPI drivers, refer to Chapter 8, “TAPI Debug.” In polling mode, CTDIAG uses the TAPI *CheckForReports* function. In interrupt mode, CTDIAG uses the TAPI User Event Handler.

Use the mnemonic or arrow key to select the interface method to use; note that it appears in the status line at the bottom of the screen. The default method is direct.

If you need to change the location of the TAPI software interrupt, see Chapter 4, “Setting Communication Parameters.”

4

Setting Communication Parameters

On occasion, you may need to change the value of one or more communication parameters when the CTDIAG defaults do not match the hardware defaults. This may occur when autodetection fails, or when you change the hardware defaults. All communication parameters may be set through a Communications Parameters menu.

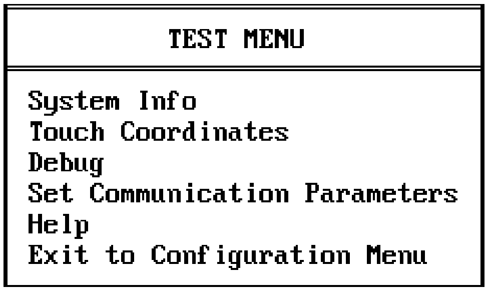
This chapter discusses the following topics:

- RS-232 Communication Parameters.
- HBC Communication Parameters.
- TAPI Communication Parameters.

There are two additional ways to set communication parameters. You may use entries in the CTDIAG command line, as explained in Chapter 9, “Command Line Options.” You may also display the RS-232 Communications Parameters menu through the Debug option, described in Chapter 7, “Debug.”

The parameters that regulate communication between the controller and the computer depend upon the type of controller used in your touch system. To set any of these communication parameters, take the following steps:

1. From the Test Configuration Menu, press ENTER to display the Test Menu, shown in Figure 4-1.



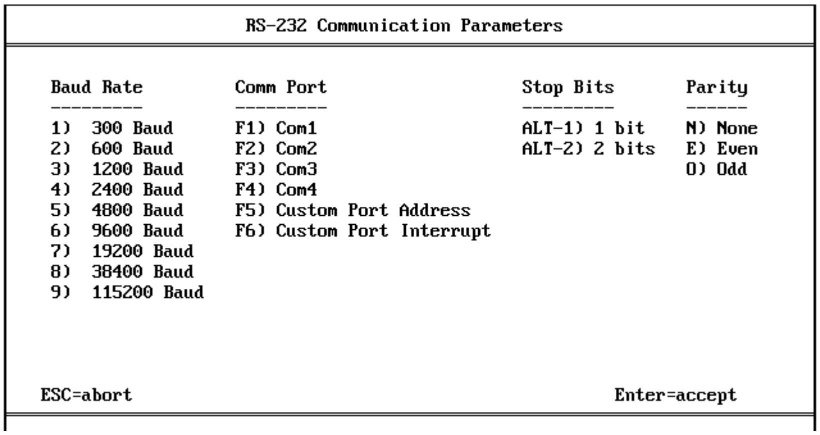
```
TEST MENU

System Info
Touch Coordinates
Debug
Set Communication Parameters
Help
Exit to Configuration Menu
```

Figure 4-1. Test Menu

2. Select Set Communication Parameters from the Test Menu. The RS-232, HBC, or TAPI Communication Parameters Menu appears, depending upon the controller type defined in the Test Configuration Menu.

RS-232 Communication Parameters

The RS-232 Communication Parameters Menu (Figure 4-2) lets you define another value for baud rate, comm port, stop bits, and parity. Type the number, function key, key combination, or letter listed to the left of the parameter. For example, to change the baud rate to 9600, type 6; to change the comm port to 3, press the F3 function key.



```
                       RS-232 Communication Parameters

  Baud Rate        Comm Port                     Stop Bits      Parity
  ---------        ---------                     ---------      ------
  1)  300 Baud     F1) Com1                      ALT-1) 1 bit   N) None
  2)  600 Baud     F2) Com2                      ALT-2) 2 bits  E) Even
  3)  1200 Baud    F3) Com3                                     O) Odd
  4)  2400 Baud    F4) Com4
  5)  4800 Baud    F5) Custom Port Address
  6)  9600 Baud    F6) Custom Port Interrupt
  7)  19200 Baud
  8)  38400 Baud
  9)  115200 Baud



 ESC=abort                                               Enter=accept
```

Figure 4-2. RS-232 Communication Parameters Menu

Baud Rate

Baud rate is the speed of data transfer between the RS-232 controller and the computer.

The default CTDIAG baud rate is autobaud for infrared systems and 9600 for guided wave systems.

The controller baud rate must match the CTDIAG baud rate. See the RS-232 controller installation instructions for details on configuring the jumper switch blocks. If the controller is configured to use autobaud, CTDIAG can communicate with the touch system at all baud rates. If the controller is set to a specific baud rate rather than autobaud, CTDIAG and the controller must be set to the same baud rate.

Comm Port

A **comm port** is the interface that governs and synchronizes the flow of data between the serial touch system and the computer. The default port is COM1.

When you select a comm port, the port address and port interrupt are automatically changed to the standard for that port.

- COM1: 3F8H and 4H
- COM2: 2F8H and 3H
- COM3: 3E8H and 4H
- COM4: 2E8H and 3H

If the RS-232 port address differs from the standard COM1 - COM4 address, you must define both a custom port address and custom port interrupt. Port addresses may range from 200H to 3F8H, in modulo 8 increments; allowable port interrupts are 2, 3, 4, 5, and 7. When you enter a non-standard port address or port interrupt, the comm port reference is deselected.

Stop Bits

A **stop bit** is a bit inserted by the serial touch system into the data stream to inform the computer that the transmission of a byte of data is complete.

Parity

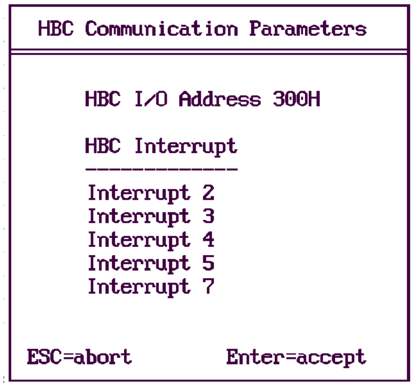
Parity is a parameter used for error-checking to ensure that the data that was transmitted by the serial touch system is identical to that received by the computer, and vice versa.

On infrared systems, the default CTDIAG parity is autoparity; on guided wave systems, it is even.

The controller parity must match the CTDIAG parity. See the RS-232 controller installation instructions for details on configuring the jumper switch blocks. If the controller is configured to use autoparity, CTDIAG can communicate with the touch system at all parity settings. If the controller is set to a specific parity rather than autoparity, CTDIAG and the controller must be set to the same parity.

HBC Communication Parameters

The HBC Communication Parameters Menu (Figure 4-3) lets you define another value for the HBC I/O address or HBC interrupt. Type the mnemonic (underlined letter) for the new value you need.



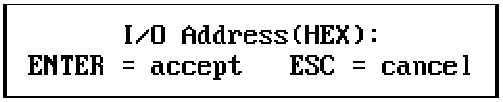
```
 HBC Communication Parameters

     HBC I/O Address 300H

     HBC Interrupt
     -------------
     Interrupt 2
     Interrupt 3
     Interrupt 4
     Interrupt 5
     Interrupt 7

ESC=abort        Enter=accept
```

Figure 4-3. HBC Communication Parameters Menu

HBC I/O Address

The **I/O address** defines the hardware base address location of an HBC's hardware registers on the controller card.

When you select HBC I/O Address from the HBC Communication Parameters Menu, the I/O Address Menu (Figure 4-4) appears.



```
      I/O Address(HEX):
ENTER = accept   ESC = cancel
```

Figure 4-4. I/O Address Menu

Type the value for the I/O address; if the value you typed is outside the allowable range of 200H to 3F0H, an error message appears and the I/O Address Menu reappears.

The I/O address at which the HBC attempts to communicate with the touch system may be set by configuring jumper switch blocks on the controller. See the HBC controller installation instructions for details. The controller I/O address must match the CTDIAG I/O address.

If you configure the HBC to use an I/O address of 300H (the default setting), there is no need to change this parameter in CTDIAG because the CTDIAG default is also 300H. If the HBC is configured to use an I/O address other than the default, CTDIAG and the controller must be set to the same I/O address.

HBC Interrupt

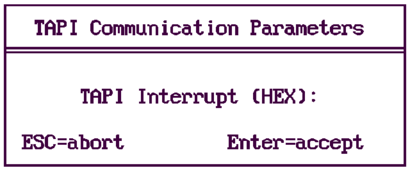
The **hardware interrupt** request line is a dedicated hardware line between the touch system and the computer. It defines where to search for an HBC.

The interrupt number at which the HBC attempts to communicate with CTDIAG may be set by configuring jumper switch blocks on the controller. See the HBC controller installation instructions for details. The controller interrupt must match the CTDIAG interrupt.

If you configure the HBC to use an interrupt number of 7 (the default setting), there is no need to change this parameter in CTDIAG because the CTDIAG default is also 7. If the HBC is configured to use an interrupt number other than the default, CTDIAG and the controller must be set to the same interrupt number.

TAPI Communication Parameters

The TAPI Communication Parameters Menu (Figure 4-5) lets you define the software interrupt for the TAPI driver. Type the hexadecimal value of the new address, then press ENTER.



TAPI Communication Parameters

TAPI Interrupt (HEX):

ESC=abort Enter=accept

Figure 4-5. TAPI Communication Parameters Menu

The default interrupt of 55H was selected because it does not conflict with any known major MS-DOS software title, but any unused software interrupt may be used.

Caution

Selecting a software interrupt is not a simple matter; it requires a basic understanding of programming and DOS functions.

5

Testing Touch System Hardware and Software

One of the most useful functions of CTDIAG is touch coordinate testing, which tests the touch system hardware and software. This chapter discusses the following topics:

- Testing Hardware with Scan Reporting.
- Testing Software with Coordinate Reporting.
- Relationship between Scan and Coordinate Reporting.

Touch coordinate testing includes:

- **Scan reporting**, which reports the results of a touch in physical beams. It is especially useful for checking hardware operations and is available for scanning infrared systems only.
- **Coordinate reporting**, which reports the results of a touch in x, y coordinates. It is especially useful for checking software operations.

Testing Hardware with Scan Reporting

When a stylus interrupts an infrared beam, the software detects that interruption and reports a touch. In scan reporting, the touch is defined by physical beams that are interrupted in each axis. This type of reporting is most useful for checking the functionality of the hardware, since the report may identify beams that are not working.

Note

Scan reporting is not available for guided wave touch screens, nor is it available for infrared touch screens using an SBC.

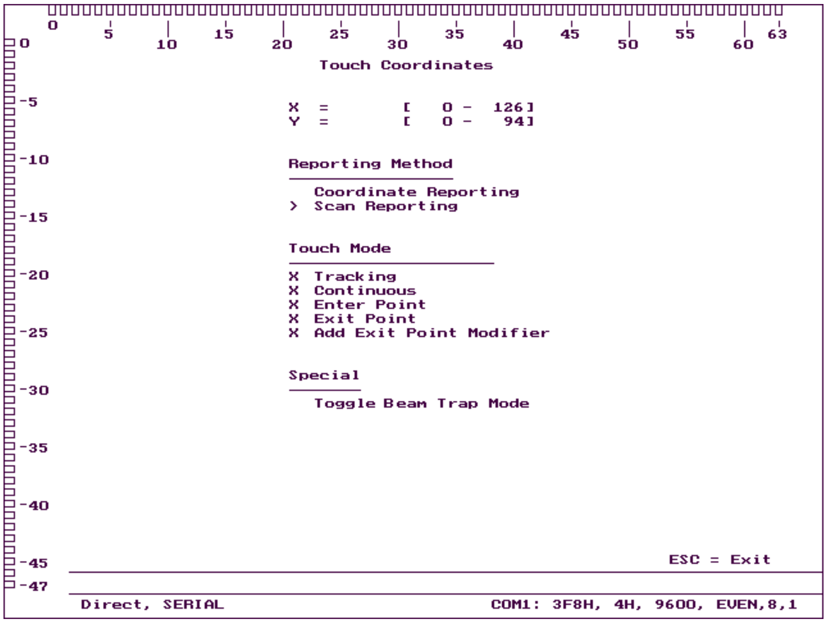
Using Scan Reporting

To start scan reporting, take the following steps:

1. From the Test Configuration Menu (Figure 3-1), press ENTER to display the Test Menu.
2. From the Test Menu (Figure 4-1), select the Touch Coordinates option.
3. When the Touch Coordinates screen appears (Figure 5-1), type S to enter scan reporting.

   The screen contains a grid of small boxes, representing the physical beams of the infrared touch system, along the left and top of the screen.

In this example screen, the horizontal (x-axis) grid at the top of the screen is numbered from 0 to 63 and the vertical (y-axis) grid at the left of the screen is numbered from 0 to 47. Thus, this grid represents a infrared touch frame with 64 physical beams along the x-axis and 48 physical beams along the y-axis, since beams are numbered starting with 0 rather than 1.



```
Touch Coordinates

X  =        [   0 -  126]
Y  =        [   0 -   94]

Reporting Method

   Coordinate Reporting
>  Scan Reporting

Touch Mode

X  Tracking
X  Continuous
X  Enter Point
X  Exit Point
X  Add Exit Point Modifier

Special

   Toggle Beam Trap Mode

                                    ESC = Exit

Direct, SERIAL                      COM1: 3F8H, 4H, 9600, EVEN,8,1
```

Figure 5-1. Touch Coordinates Screen (Scan Reporting)

To verify that the physical beams are working correctly, simply place a stylus on the touch screen. This interrupts the beams and highlights the corresponding grid boxes, indicating which beams have been interrupted. For example, if you place a stylus in the middle of the screen, the grid boxes in the middle of the x-axis (around 30) and the y-axis (around 23) will highlight. At times more than one grid box per axis is highlighted, since a finger is generally wide enough to break more than one beam on each axis.

As you move the stylus randomly around the touch screen, the grid boxes highlight as the corresponding beams are interrupted, and return to normal once the stylus moves out of the beam. If you place multiple styli on the touch screen, all beam interruptions are indicated.

Touch Mode options are not available under scan reporting.

The *Beam Trap Mode* is an additional graphic indicator. If the *Beam Trap Mode* is on, the grid box of an interrupted beam remains highlighted, rather than turning off, even after a stylus no longer interrupts the beam. This creates a “permanent” record of all beam interruptions, until you toggle the beam trap off. Toggle the beam trap on or off by typing B.

To check the entire grid, move a stylus along the perimeter of the touch screen, making sure that each grid box highlights when the corresponding beam is interrupted. If *Beam Trap Mode* is on, all grid boxes should be highlighted at the end of the process. If a few grid boxes remain not highlighted, run the stylus through those beams again to see if they simply were not interrupted on the first pass.

As an additional check, place the stylus in each corner of the screen; the grid boxes for the maximum or minimum physical beam values should highlight. In Figure 5-1, for example, those values are:

• Upper left corner	0, 0
• Upper right corner	63, 0
• Lower left corner	0, 47
• Lower right corner	63, 47

Troubleshooting

If an error message appears in the Message line, look up the suggested actions in Appendix A, “Error Messages.” Other problems that may arise during scan reporting, as well as their solutions, include:

<u>Some beams read as interrupted.</u>

- Check that all beams are clear of obstructions and that the bezel is clean.
- Cycle the power to the touch system and try again. If the error persists, contact Carroll Touch Technical Support.

<u>There are gaps where you cannot cause beams to be interrupted.</u>

- Enable the *Beam Trap Mode* of the Touch Coordinate Test screen, and slowly move your finger along the x- and y-axes in order to block every beam. If the corresponding grid boxes do not highlight, contact Carroll Touch Technical Support.

<u>All beams read as though they were being interrupted.</u>

If you are using an HBC:

- Check that the controller card is installed properly.
- Check that the frame is connected to the controller.
- Check that the I/O address and interrupt number settings on the controller match those shown on the Status line.
- Select another I/O address and interrupt number setting to avoid possible conflict with other computer hardware. The addresses of other hardware may be found in the IBM AT I/O Port Map in the HBC installation instructions.

If you are using a serial (RS-232) controller or a Smart-Frame:

- Check that power is supplied to the RS-232 controller or Smart-Frame.
- Check that the communication cable is properly connected to the correct comm port.
- Ensure that the controller or Smart-Frame jumpers (if any) are set to either autobaud or to a fixed baud rate and parity that matches those shown on the Status line.
- Ensure that the stop bits parameter on the Status line is set to 1.
- Check that the port parameter on the Status line is set to the comm port to which the touch system is connected.

Testing Software with Coordinate Reporting

When a finger or stylus interrupts an infrared beam or guided wave, the software detects that interruption and reports a touch. In coordinate reporting, the touch is reported in x, y coordinates. This type of reporting is most useful for checking the functionality of the software.

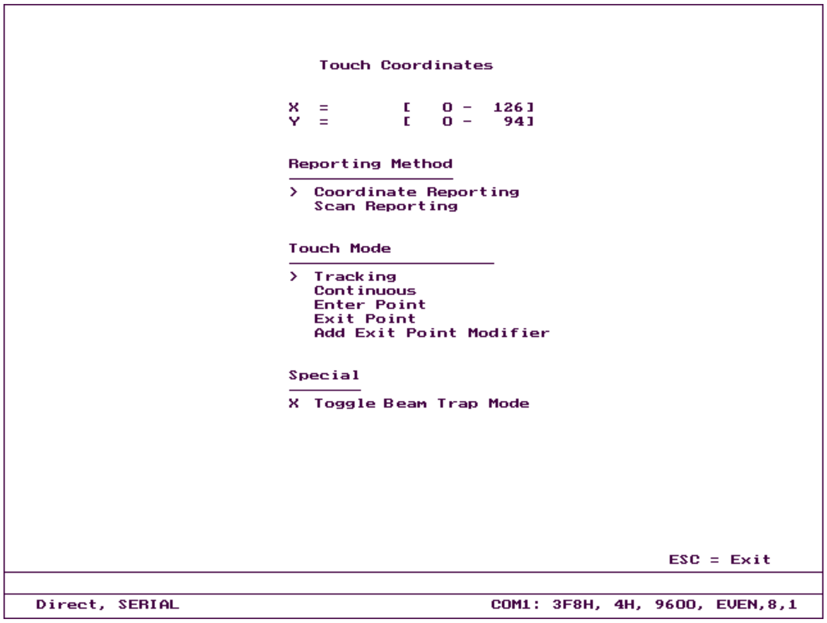
Using Coordinate Reporting

To start coordinate reporting, take the following steps:

1. From the Test Configuration Menu (Figure 3-1), press ENTER to display the Test Menu.
2. From the Test Menu (Figure 4-1), select the Touch Coordinates option.
3. When the Touch Coordinates screen appears (Figure 5-2), type R to enter coordinate reporting.

Note that, for the example infrared system, the X = value at the top of the screen displays a range of possible values for x coordinates from 0 to 126. Similarly, Y = displays a range of y coordinates from 0 to 94.

Because the *Beam Trap Mode* operates with physical beams, not x, y coordinates, it is not an available option within coordinate reporting.



```
                    Touch Coordinates

               X  =       [   0 -  126]
               Y  =       [   0 -   94]

               Reporting Method
               ________________
               >  Coordinate Reporting
                  Scan Reporting

               Touch Mode
               ____________________
               >  Tracking
                  Continuous
                  Enter Point
                  Exit Point
                  Add Exit Point Modifier

               Special
               _______
               X  Toggle Beam Trap Mode



                                                    ESC = Exit

  Direct, SERIAL                      COM1: 3F8H, 4H, 9600, EVEN,8,1
```

Figure 5-2. Touch Coordinate Test Screen (Coordinate Reporting)

The Touch Mode defines what is meant by a touch. For example, does a touch occur when the stylus enters the active area of the touch screen or when it leaves the active area? Select the various touch modes, as explained in the next few paragraphs, to test that x, y coordinates are reported properly.

Tracking Mode

In *Tracking Mode*, touches are reported as long as a stylus is moving in the active area of the touch screen; reporting stops when the stylus is stationary, and resumes with stylus movement. To use *Tracking Mode*, type T.

When a stylus touches the touch screen, a small tracking box appears beneath the touch point and the x, y coordinates of the touch point are displayed in X = and Y = fields at the top of the screen. As a stylus moves around the touch screen, the tracking box follows it and the coordinates in the X = and Y = fields are updated to reflect the position of the new touch point.

Note that there is a small, stationary bar across the upper third of the tracking box. This indicates that coordinate reporting occurs whenever the stylus moves, but ceases when the stylus remains stationary.

To check the entire range of touch coordinates, move a stylus along the perimeter of the touch screen, making sure that the x coordinate increases as you move right and decreases as you move left and that the y coordinate increases as you move down and decreases as you move up. If a stylus is in the corner of the screen, the coordinates should reflect the maximum or minimum x, y coordinates. In Figure 5-2, these values are:

• Upper left corner	0, 0
• Upper right corner	126, 0
• Lower left corner	0, 94
• Lower right corner	126, 94

Continuous Mode

In *Continuous Mode*, touches are reported as long as a stylus is in the active area of the touch screen, whether moving or stationary. To use *Continuous Mode*, type C.

As in *Tracking Mode*, the tracking box follows your finger around the screen and the coordinates are updated to reflect the position of the new touch point. However, the bar within the tracking box is continuously scrolling through the box to indicate that coordinates are constantly reported, whether the stylus is moving or stationary.

Enter Point Mode

In *Enter Point Mode*, a touch is reported when a stylus enters the active area of the touch screen. The tracking box jumps to the initial touch point, but does not follow the stylus around the screen; the stylus must exit and reenter the screen to generate another touch. To use *Enter Point Mode*, type E.

Exit Point Mode

In *Exit Point Mode*, a touch is reported only when a stylus leaves the active area of the touch screen. The tracking box jumps to the point where the stylus is lifted off the screen; the stylus must reenter and exit the screen to generate another touch. A touch is not reported when a stylus enters or moves about the active area of the touch screen. To use *Exit Point Mode*, type X.

Add Exit Point Modifier

If you add an Exit Point Modifier, touch coordinates are reported whenever a stylus leaves the active area of the touch screen, in addition to whatever coordinates are reported under the Touch Mode that is active. For example, in *Enter Point Mode* with the modifier added, touch coordinates are reported whenever a stylus enters the screen (*Enter Point Mode*) and leaves the screen (Exit Point Modifier).

If used in conjunction with *Exit Point Mode*, adding the modifier results in double reporting of each exit point.

To add an Exit Point Modifier, type A.

Troubleshooting

If an error message appears in the Message line, look up the suggested actions in Appendix A, “Error Messages.” Other problems that may arise during coordinate reporting, as well as their solutions, include:

Touch Coordinates screen appears, but does not respond to touches.

If you are using an HBC or RS-232 controller, use scan reporting to check the infrared touch screen.

If you are using an SBC or HBC:

- Check that the controller card is installed properly.
- Check that the touch frame is connected to the controller.
- Check that the I/O address and interrupt number settings on the controller match those shown on the Status line.
- Select another I/O address and interrupt number setting to avoid possible conflict with other hardware in the PC. The addresses of other hardware may be found in the IBM AT I/O Port Map of the SBC or HBC installation instructions.

If you are using an RS-232 controller or a Smart-Frame:

- Check that power is supplied to the RS-232 controller or Smart-Frame.
- Check that the communication cable is properly connected to the correct comm port.
- Ensure that the controller or Smart-Frame jumpers (if any) are set to either autobaud or to a fixed baud rate and parity that matches those shown on the Status line.
- Ensure that the stop bits parameter on the Status line is set to 1.
- Check that the port parameter on the Status line is set to the comm port to which the touch system is connected.

Touch Coordinates screen appears, but non-contiguous messages appear unexpectedly.

- Make sure you are only interrupting the infrared beam with one finger - use a pen as a stylus to make sure. If messages persist, the frame may have failed beams. If you are using an HBC or RS-232 controller, use scan reporting to check further.

Touch Coordinates screen appears, but there are dead spots on the screen.

- Make sure you are using your finger or a stylus that is at least 5/16" diameter. The infrared frame may have failed beams. Use scan reporting to check further.

Relationship between Scan and Coordinate Reporting

As mentioned earlier, the touch software interprets the physical beam data to report touch information. To achieve finer resolution than the physical infrared beam grid provides, the touch software interpolates a virtual beam between each pair of physical beams. The physical beams are assigned even numbers (0, 2, 4, and so on) and the virtual beams are assigned odd numbers (1, 3, 5, and so on).

The relationship between physical beams (scan reporting) and x, y coordinates (coordinate reporting) is determined by the equations:

Maximum x coordinate = 2 * (number of x physical beams - 1)
Maximum y coordinate = 2 * (number of y physical beams - 1)

For example, in the infrared screen shown in Figure 5-1, the x axis has a range of 0 to 63, representing 64 physical beams (scan reporting). Using the equation (2 * 64 - 1) yields a result of 127, which is the number of x coordinates in the same example infrared screen in Figure 5-2 (coordinate reporting).

Detailed information on the use of physical beams, virtual beams, logical beams, logical coordinates, and beam averaging is found in the *Touch System Programmer's Guide*.

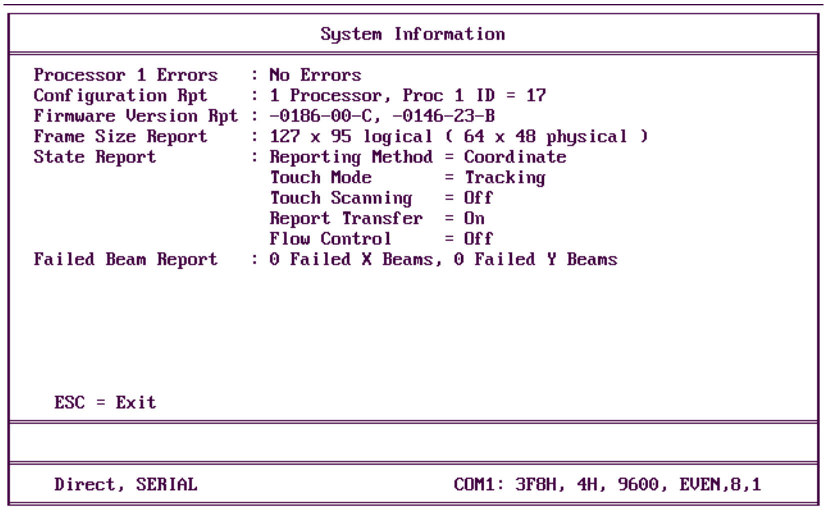
6

Touch System Information

A summary report of the touch system is available through CTDIAG.

To view system information for touch systems, take the following steps:

1. From the Test Configuration Menu (Figure 3-1), press ENTER to display the Test Menu.
2. Select System Info from the Test Menu (Figure 4-1). The System Information screen, shown in Figure 6-1, appears.



```
                              System Information

 Processor 1 Errors   : No Errors
 Configuration Rpt    : 1 Processor, Proc 1 ID = 17
 Firmware Version Rpt : -0186-00-C, -0146-23-B
 Frame Size Report    : 127 x 95 logical ( 64 x 48 physical )
 State Report         : Reporting Method = Coordinate
                        Touch Mode       = Tracking
                        Touch Scanning   = Off
                        Report Transfer  = On
                        Flow Control     = Off
 Failed Beam Report   : 0 Failed X Beams, 0 Failed Y Beams




   ESC = Exit


   Direct, SERIAL                      COM1: 3F8H, 4H, 9600, EVEN,8,1
```

Figure 6-1. System Information Screen

This screen reports the results of various commands in the Smart-Frame Protocol.

- Processor Errors identifies the current error conditions. Information comes from the *Get_Error_Report (32H)* command.
- Configuration Report reports the number of processors, and their identifiers, in the touch system. Information comes from the *Get_Configuration_Report (33H)* command.
- Firmware Version Report is a list of the firmware versions for the various touch system components. Each version is a 10-byte ASCII-encoded string. Information comes from the *Get_Firmware_Version_Report (34H)* command.
- Frame Size Report describes the frame size of the infrared touch system in both logical (x, y) coordinates and physical beams. It is based upon the *Get_Frame_Size_Report (37H)* command. This report is not valid for guided acoustic wave touch systems.
- State Report lists the current settings for regular coordinate reporting, based upon the *Get_State_Report (47H)* command. Definitions of most of these parameters are contained in Chapter 5. The flags include:

  - Reporting Method: Coordinate reporting or scan reporting is active.
  - Touch Mode: *Tracking Mode*, *Continuous Mode*, *Exit Point Mode*, or *Enter Point Mode* is active, with or without an Exit Point Modifier.
  - Touch Scanning: Touch system is not detecting (off) or is detecting (on) touch.
  - Report Transfer: Smart-Frame Protocol reports are not sent (off) or are sent (on) to the host.
  - Flow Control: Hardware handshaking is not used (off) or is used (on) to regulate serial communications.
- Failed Beam Report identifies failed infrared beams and analyzes, if possible, the cause. Information comes from the *Get_Failed_Beam_Report (36H)* command. This report is not valid for guided acoustic wave touch systems.

For detailed information on these commands, refer to the *Touch System Programmer's Guide*.

7 Debug

The Debug option lets you send and receive SFP commands and reports and directly observe the communication between the host and the touch system. This provides a simple way to check operations of the commands.

This chapter discusses the following topics:

- Debug Overview.
- Debug Screen.
- Debug Commands and Debug Help.
- Trace Buffer.
- Debugging under the Smart-Frame Protocol.

Caution

Use of Debug requires an understanding of the complexities of the Smart-Frame Protocol. While this chapter attempts to present this information in as simple a manner as possible, you need some understanding of programming concepts to understand and to use Debug. Refer to the *Touch System Programmer's Guide* for more information.

Debug Overview

Debug allows you to observe the SFP commands sent to the touch system and the reports received from the touch system. For example, an SFP command might be a request to the touch system for the current touch coordinates and the associated report would be the actual touch coordinates.

You can also use another debugger, based upon the Touch Application Program Interface (TAPI), to monitor host/touch system communication, as described in Chapter 8. Either approach produces identical results, since the commands and reports are identical and only the method of communication (not using or using TAPI) is different.

Note

For SBC troubleshooting, you must use the TAPI-based debugger. The SBC hardware alone cannot communicate with SFP; it must communicate through TAPI.

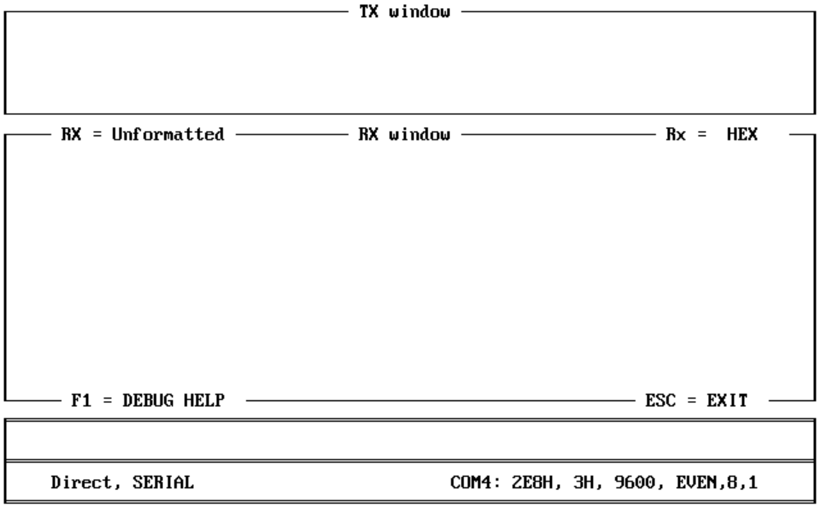
Debug Screen

To start a debug session, take the following steps:

1. From the Test Configuration Menu (Figure 3-1), press ENTER to display the Test Menu.
2. Select Debug from the Test Menu (Figure 4-1). The Debug screen, shown in Figure 7-1, has two windows.

   The **transmit (Tx) window** displays the SFP commands as they are entered at the keyboard and sent from CTDIAG to the touch system. Commands appear in low-intensity white (gray).

   The **receive (Rx) window** displays the SFP commands that are transmitted to, and the reports that are received from, the touch system.

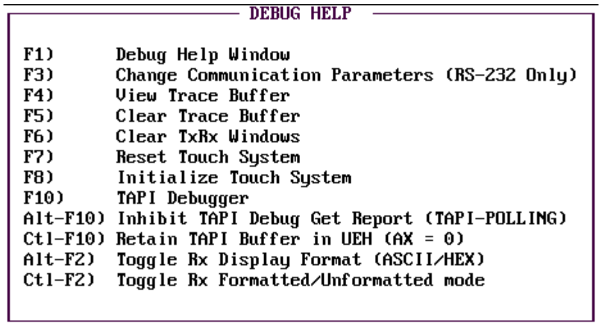


```
Direct, SERIAL                    COM4: 2E8H, 3H, 9600, EVEN,8,1
```

Figure 7-1. Debug Screen

Debug Commands and Debug Help

A number of debug commands are available to help you with troubleshooting. To see a brief summary of these commands, press F1 for debug help (Figure 7-2). You do not have to display the Help window to use these functions; it is simply a reminder of the available commands.



```
DEBUG HELP

F1)      Debug Help Window
F3)      Change Communication Parameters (RS-232 Only)
F4)      View Trace Buffer
F5)      Clear Trace Buffer
F6)      Clear TxRx Windows
F7)      Reset Touch System
F8)      Initialize Touch System
F10)     TAPI Debugger
Alt-F10) Inhibit TAPI Debug Get Report (TAPI-POLLING)
Ctl-F10) Retain TAPI Buffer in UEH (AX = 0)
Alt-F2)  Toggle Rx Display Format (ASCII/HEX)
Ctl-F2)  Toggle Rx Formatted/Unformatted mode
```

Figure 7-2. Debug Help Window

Debug Help Window (F1 key) displays the Debug Help Window.

Change Communication Parameters (RS-232 Only) (F3 key) displays the RS-232 Communication Parameters Menu so you can change communication parameters when testing a serial (RS-232) controller.

The Debug screen reappears once you have made any necessary changes. For information on these parameters, see Chapter 4.

View Trace Buffer (F4 key) displays the Trace Buffer screen, which contains a copy of the communications trace buffer. See the next section for more information.

Clear Trace Buffer (F5 key) erases the communications trace buffer.

Clear TxRx Windows (F6 key) erases the transmit and receive windows.

Reset Touch System (F7 key) sends a 400 ms break or the appropriate reset command, depending on the type of controller. A reset is a subset of a full initialization. For more information, refer to the *Touch System Programmer's Guide*.

Initialize Touch System (F8 key) sets off a sequence of activities, including a reset, power on testing, and error checking. For more information, refer to the *Touch System Programmer's Guide*.

TAPI Debugger (F10 key) is described in detail in Chapter 8.

Inhibit/Enable TAPI Debug Get Report (TAPI-POLLING) (ALT + F10 keys) is a toggle. In the TAPI polling interface method, reports in the application buffer are ordinarily cleared. You can override this to view the reports. Refer to Chapter 8 for details on TAPI debug.

Retain/Do not retain TAPI Buffer in UEH (AX = 0) (CTRL + F10 keys) is a toggle. Reports are ordinarily transferred from the TAPI buffer to the application buffer. If desired, you can retain the TAPI buffer in the TAPI User Event Handler (UEH). Refer to Chapter 8 for details on TAPI debug.

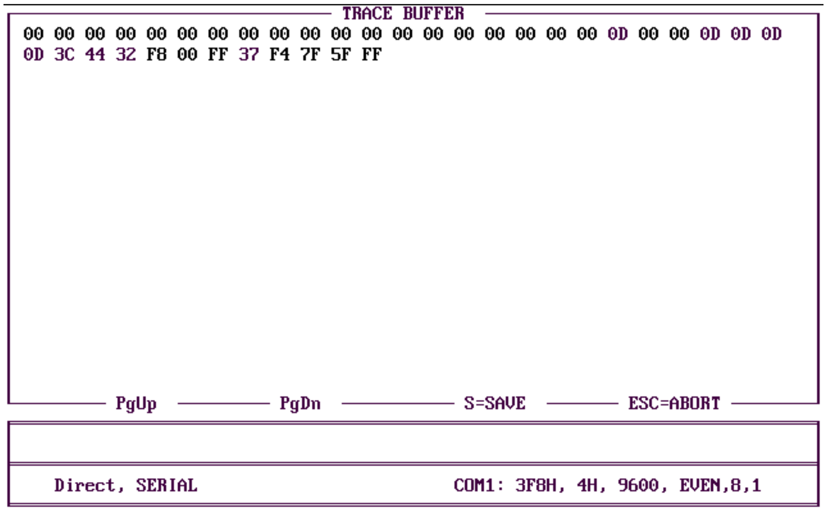
Toggle Rx Display Format (ASCII/HEX) (ALT + F2 keys) switches the display format between ASCII and hexadecimal in the receive window.

Toggle Rx Formatted/Unformatted Mode (CTRL + F2 keys) switches between *Formatted Mode* and *Unformatted Mode* in the receive window. Modes are described in the SFP section later in this chapter.

Trace Buffer

The trace buffer maintains a record of all communication between CTDIAG and the touch system.

To make a copy of the buffer and display it in the Trace Buffer screen (Figure 7-3), press F4. While you are viewing the copy of the trace buffer, CTDIAG continues to record any new communication in the trace buffer. This new communication will not appear in the Trace Buffer screen currently on display, since the screen is merely a static copy of the actual buffer. The new communication will, however, appear in the receive window when you return to the Debug screen and will be in the Trace Buffer the next time you request a copy of it.



```
TRACE BUFFER
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 0D 00 00 0D 0D 0D
0D 3C 44 32 F8 00 FF 37 F4 7F 5F FF
```

```
Direct, SERIAL                    COM1: 3F8H, 4H, 9600, EVEN,8,1
```

Figure 7-3. Trace Buffer Screen

When the buffer is full, it is emptied on a first in, first out (FIFO) basis. To clear the buffer any time you are in Debug, press F5. You may save a copy of the buffer to a file by typing S and specifying a file name.

Debugging under the Smart-Frame Protocol

This section describes debugging under the SFP and gives all examples in hexadecimal.

Commands

To issue a command under the Smart-Frame Protocol, simply type its function number. An SFP command example is 32, the hexadecimal code for the *Get_Error_Report* command.

Reports

An SFP report has the format:

header	report bytes	trailer

The trailer is always `FF`. An example of a report returned from the *Get_Error_Report* command is:

`F8`	`00`	`FF`

`F8` is the header and `00` indicates no errors. An example of a touch coordinate report is:

`FE`	`35 2D`	`FF`

`FE` is the header and `35` and `2D` are the x, y touch coordinates.

Transmit Window

When you type a command number, it is echoed in the Tx window.

Receive Window

Press Ctrl + F2 to toggle the window mode between formatted and unformatted. In *Unformatted Mode*, commands and corresponding reports fill a line, then wrap around to the next line, while *Formatted Mode* puts each command and associated report on a separate line for easier viewing.

Commands appear in gray and reports in white.

The current format for the screen display (ASCII or hexadecimal) is noted in the upper right corner of the receive window; press Alt + F2 to toggle the format between the two.

Debug Example

This example demonstrates how to use Debug to initialize a touch system and to send and receive touch coordinates, using standard SFP commands. This example uses an infrared system with an HBC in the direct interface method.

1. Select Debug from the Test Menu. The Debug screen appears, similar to the one shown in Figure 7-1. Both the transmit and receive windows are empty and set to display data in hexadecimal format.

2. Press F8 to initialize the infrared touch system. (Remember, you can press F1 to see the help screen that reminds you of the commands available in debug.)
3. Press the F4 key to make a copy of the trace buffer and display it in the Trace Buffer screen.

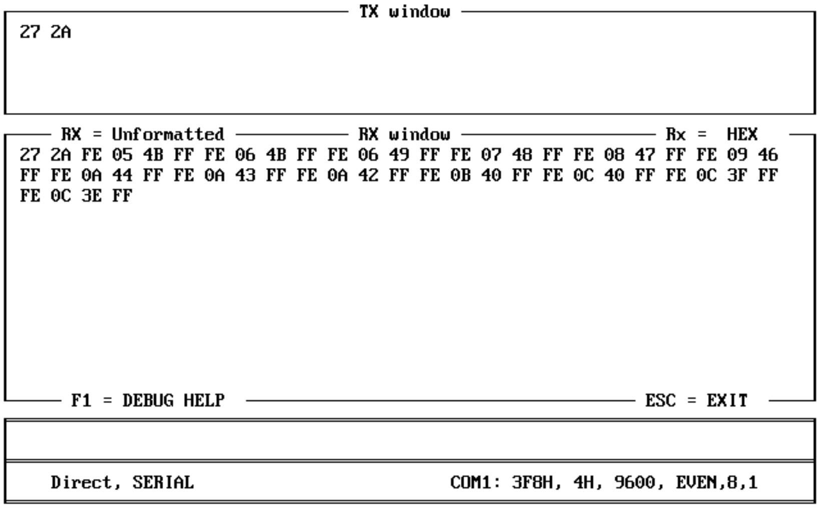
   The Trace Buffer screen appears, as shown in Figure 7-3. On the monitor, commands are gray and reports are white. A string of hexadecimal commands and reports is displayed in the trace buffer. These are the SFP commands issued by CTDIAG for the SFP initialization sequence. Their meanings are:

`3C`	=	*Software_Reset.*
`44`	=	*Report_Transfer_On.*
`32`	=	*Get_Error_Report.*
`F8 00 FF`	=	Error report, indicating no errors.
`37`	=	*Get_Frame_Size_Report.*
`F4 7F 5F FF`	=	Frame size report, indicating 7F (127 in decimal) x-axis logical coordinates and 5F (95 in decimal) y-axis logical coordinates.

   There may be additional data before the string if you had previously selected other options that transmit and receive data from the touch system.
4. Press ESC to return to the Debug screen and type `27` followed by `2A`.

   The command string `27 2A` appears in both the transmit and receive windows, indicating that the *Continuous_Mode* (27H) and the *Touch_Scanning_On* (2AH) commands have been issued and received. On the monitor, commands are gray.
5. Touch the display, slide your finger across it, then withdraw your finger.

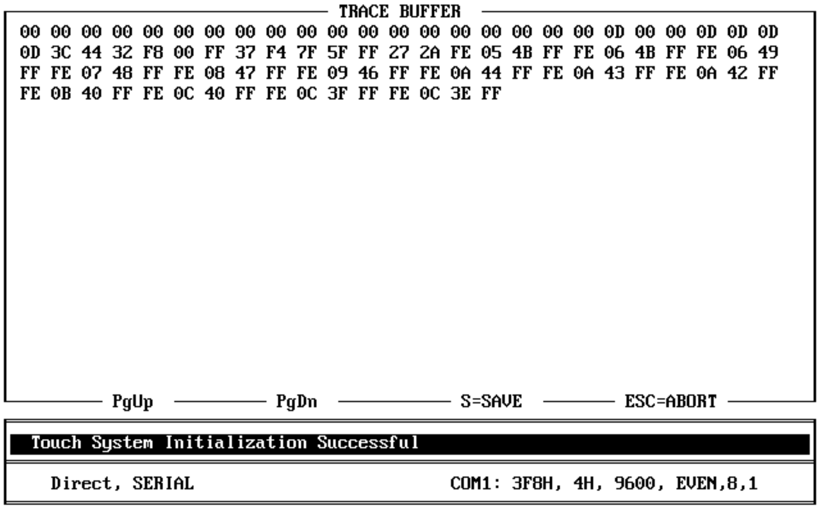
   Reports of the form `FE xx yy FF` begin to appear in the receive window; 13 sets of coordinates are shown in Figure 7-4. These are coordinate reports sent from the touch system reflecting the x, y location of your finger on the display. Because the touch system is in *Continuous Touch Mode*, the reports continue as long as your finger is on the display. On the monitor, reports are white.



```
TX window
27 2A

RX = Unformatted            RX window                 Rx =  HEX
27 2A FE 05 4B FF FE 06 4B FF FE 06 49 FF FE 07 48 FF FE 08 47 FF FE 09 46
FF FE 0A 44 FF FE 0A 43 FF FE 0A 42 FF FE 0B 40 FF FE 0C 40 FF FE 0C 3F FF
FE 0C 3E FF

F1 = DEBUG HELP                                           ESC = EXIT

Direct, SERIAL                        COM1: 3F8H, 4H, 9600, EVEN,8,1
```

Figure 7-4. Debug Screen: SFP Touch Coordinates

6. Press F4 to copy the trace buffer into the Trace Buffer screen, as shown in Figure 7-5. Data in the buffer is displayed in a continuous stream. Note that the 13 sets of touch coordinates now appear in the trace buffer.



```
TRACE BUFFER
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 0D 00 00 0D 0D 0D
0D 3C 44 32 F8 00 FF 37 F4 7F 5F FF 27 2A FE 05 4B FF FE 06 4B FF FE 06 49
FF FE 07 48 FF FE 08 47 FF FE 09 46 FF FE 0A 44 FF FE 0A 43 FF FE 0A 42 FF
FE 0B 40 FF FE 0C 40 FF FE 0C 3F FF FE 0C 3E FF

PgUp          PgDn            S=SAVE         ESC=ABORT

Touch System Initialization Successful
Direct, SERIAL                        COM1: 3F8H, 4H, 9600, EVEN,8,1
```

Figure 7-5. Trace Buffer Screen: SFP Touch Coordinates

7. Touch the display again and move your finger around, noting that no change occurs on the Trace Buffer screen. Press ESC to return to the Debug screen. The presence of your finger in the touch frame while you were viewing the Trace Buffer screen increased the number of coordinate reports displayed in the receive window.

8. Press the F4 key again to copy the trace buffer into the Trace Buffer screen. There is now more data on the screen, reflecting the coordinate reports sent during the touch in step 7.
9. Press the ESC key to return to the Debug screen, then ESC again to return to the Test Menu.

8 TAPI Debug

TAPI Debug permits communication with a Touch Application Program Interface (TAPI) driver via TAPI function calls. This allows you to experiment with TAPI functions and manually verify that the TAPI driver is loaded and functioning correctly.

This chapter discusses the following topics:

- Loading the TAPI Driver.
- Debug and TAPI Drivers.
- TAPI Debug Example.

Caution

Use of TAPI Debug requires an understanding of the complexities of the Smart-Frame Protocol and the Touch Application Program Interface. While this chapter attempts to present this information in as simple a manner as possible, you need some understanding of programming concepts to use TAPI Debug effectively. Refer to the *Touch System Programmer's Guide* for more information.

Loading the TAPI Driver

TAPI driver is a generic term that refers to the SBC driver, the HBC driver, or the serial (RS-232) driver. Each driver is a terminate-and-stay-resident (TSR) program that provides a controller-independent set of function calls (the TAPI function calls) to CTDIAG. The driver communicates to a specific controller via PC bus I/O ports or an RS-232 port, depending on the type of controller.

The appropriate TAPI driver must be installed before using TAPI Debug. To load the driver with default values, change to the `c:\ctouch\ctdos` directory and type:

```
ltapi.bat
```

This batch program will load the appropriate driver for your controller, based upon the controller type you selected during installation.

Refer to the *Touch System Programmer's Guide* for more information on installing and using TAPI drivers. If you attempt to start TAPI Debug without installing a TAPI driver, a message such as this is displayed:

```
TAPI Driver Not Found at S/W Interrupt 55H
```

Debug and TAPI Drivers

Sending commands and reports through the TAPI driver produces the same results as direct communication between CTDIAG and the touch system (described in Chapter 7). This is because the commands and reports are identical; only the method of communication is different.

TAPI Debug Example

This example describes how to use TAPI Debug to obtain RS-232 communication parameters with *GetCommunicationParameters* (TAPI function 4). As documented in the *Touch System Programmer's Guide*, the only call needed for *GetCommunicationParameters* is an `AX` value of 4. The expected returns for an RS-232 are `BH` (driver type), `BL` (comm port), `CH` (parity), and `CL` (baud rate).

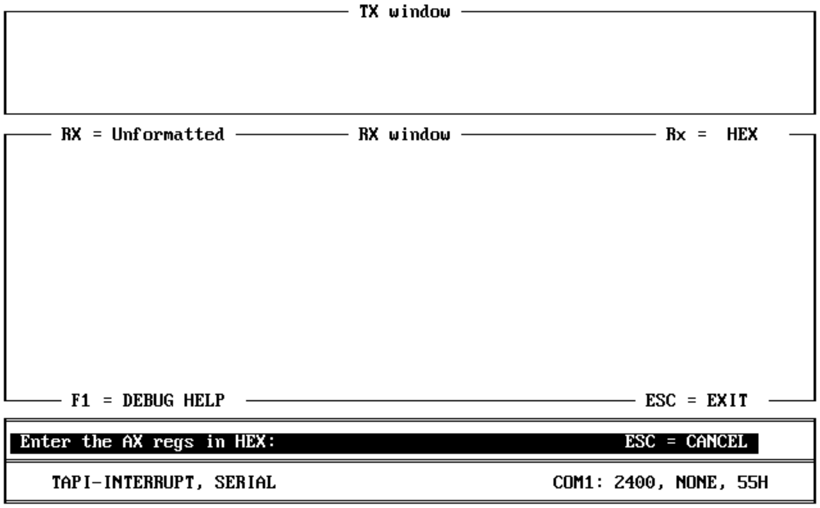
To use TAPI Debug, take the following steps. You may exit TAPI Debug at any time by pressing ESC.

1. Install the RS-232 driver by moving to the `c:\ctouch\ctdos` directory in DOS and typing:

```
ltapi.bat
```

If the driver does not install correctly, check all communication parameters, then refer to the *Touch System Programmer’s Guide* for additional information.

2. Start CTDIAG and use TAPI interrupt as the interface method. You can specify TAPI interrupt as a command line option (`ctdiag i:T;` see Chapter 9 for details) or in the `CTDIAG.CFG` configuration file (see Appendix B for details). The Test menu appears. Otherwise, select the TAPI interrupt interface method from the Test Configuration Menu, then advance to the Test menu.
3. Select Debug from the Test Menu and press F10 to start TAPI Debug. As shown in Figure 8-1, CTDIAG prompts for the hexadecimal AX register value in the Message line.



```
Enter the AX regs in HEX:
TAPI-INTERRUPT, SERIAL                    COM1: 2400, NONE, 55H
```

Figure 8-1. SFP TAPI Debug Screen

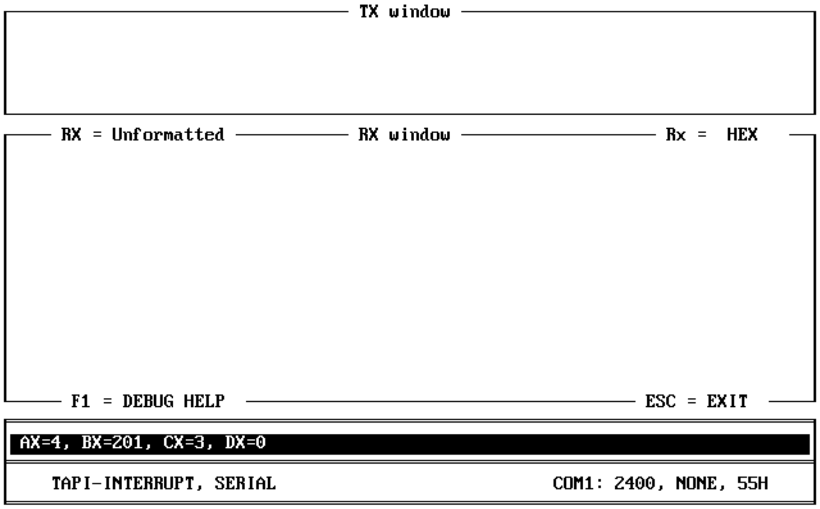
4. Type 4 to select TAPI function 4, *GetCommunicationParameters*.

   CTDIAG loads the value 4 into the AX register, then prompts you to:

   ```
   Enter the BX regs in HEX:
   ```

5. Because AX is the only call needed for this function, press Enter to accept the default value of 0H for the BX register. When prompted, press Enter to accept the default value of 0H for the CX and DX registers, as well.

CTDIAG loads the value 0H into the BX, CX, and DX registers, makes the function call to the TAPI driver via the TAPI software interrupt (usually 55H) and displays the returns in the registers, as shown in Figure 8-2.



```
AX=4, BX=201, CX=3, DX=0
TAPI-INTERRUPT, SERIAL                    COM1: 2400, NONE, 55H
```

Figure 8-2. SFP TAPI Debug Results

An AX value of 4 is the correct function call for *GetCommunicationParameters.* BX is split into a high and a low value: BH represents the driver type and its value of 2 is a code for an RS-232 controller; BL represents the comm port, with a value of 1 in this example. CH and CL are the parity and baud rate which, in this case is 0 (no parity) and 3 (baud rate of 2400). Remember that these are the parameters used by TAPI to communicate with the controller - not the parameters used by CTDIAG.

The TAPI driver is installed and operating correctly, since it has returned the proper communication parameters for your RS-232 in response to the *GetCommunicationParameters* command.

6. Press ESC to exit TAPI Debug Mode and return to the Test Menu.

9 Command Line Options

You may set configuration, communication, and autodetection parameters using the CTDIAG command line, rather than using the menus described in earlier chapters. Terminology has been explained in detail in Chapters 2, 3 and 4.

This chapter discusses the following topics:

- CTDIAG Command Line.
- Examples.

CTDIAG Command Line

The CTDIAG command line has the following syntax:

`ctdiag` *`parameters`*

If you use any command line parameters, CTDIAG skips autodetection. The available *`parameters`* follow.

Note

The command line is case-insensitive. Parameters may be either uppercase or lowercase, and may be arranged in any order.

`N` = Specifies no autodetection.

`A`*`nnn`* = Specifies the I/O address. *`nnn`* may be a value from 200H to 3F0H. The default is 300H.

`I`*`n`* = Specifies the hardware interrupt (IRQ). *`n`* may be 2, 3, 4, 5 or 7 (the default).

`S`*`nn`* = Specifies the TAPI software interrupt. *`nn`* may be any unused software interrupt. The default is 55H.

`C`*`n`* = Specifies the number of the communications port. Use for serial (RS-232) controller only. *`n`* may be:
1 = COM1 (default).
2 = COM2.
3 = COM3.
4 = COM4.

`B`*`nnnnnn`* = Specifies the baud rate. *`nnnnnn`* may be 300, 600, 1200, 2400, 4800, 9600 (default), 19200, 38400, or 115200. Use for serial (RS-232) controller only.

`P`*`x`* = Specifies the parity. Use for serial (RS-232) controller only. *`x`* may be:
E = Even parity (default).
O = Odd parity.
N = None.

`I:`*`x`* = Specifies the interface method. *`x`* may be:
D = Direct (default).
P = TAPI polling.
T = TAPI interrupt.

`C:x` = Specifies the controller type. *x* may be:
C = Software-based controller.
H = Hardware-based controller.
R = Serial (RS-232) controller (default).

`Xx` = Specifies the origin of the x-axis of the touch screen. *x* may be:
L = Left side (default).
R = Right side.

`Yx` = Specifies the origin of the y-axis of the touch screen. *x* may be:
T = Top (default).
B = Bottom.

> ***Note***
>
> Command line parameters are evaluated from left to right. For example, if you type:
>
> ```
> ctdiag n a2e8 i5 c1
> ```
>
> COM1 is used instead of address 2E8H and hardware interrupt 5.

To display help for the command line parameters, type the following at the DOS prompt:

```
ctdiag ?
```

Examples

Autodetection

To start CTDIAG with autodetection, type:

```
ctdiag
```

The `CTDIAG.CFG` file determines the search order for the touch system. Since `CTDIAG.CFG` contains many commonly used combinations of hardware and software parameters, this command can usually determine the configuration of your touch system.

No Autodetection

To start CTDIAG without autodetection, type:

```
ctdiag N
```

Because no other parameters are specified on the command line, default values are used. The command line equivalent of those defaults is:

```
ctdiag I:D C:R C1 B96 PE XL YT
```

Serial (RS-232) Controller

To start CTDIAG with a serial (RS-232) controller using COM2 and a baud rate of 19200, type:

```
ctdiag C:R C2 B19200
```

CTDIAG uses the defaults of even parity and direct interface method.

Hardware-Based Controller (HBC)

To start CTDIAG with an HBC using I/O address 280H and hardware interrupt 5, type:

```
ctdiag C:H A280 I5
```

CTDIAG uses the default interface method, which is direct.

To start CTDIAG with an HBC using hardware interrupt 5, type:

```
ctdiag C:H I5
```

CTDIAG uses the default of 300H for the I/O address.

TAPI Polling

To start CTDIAG with the TAPI polling interface method, type:

```
ctdiag I:P
```

The default software interrupt is 55H. RS-232 is the default controller type; if the controller type is HBC, it is necessary to specify it by using the `C:H` entry.

Non-Standard Installation

To start CTDIAG with a frame that has been mounted with the touch origin on the right bottom corner of the screen, type:

```
ctdiag C:H A280 I5 XR YB
```

A

Error Messages

Most of the touch system problems that can be detected by CTDIAG are identified through scan reporting and coordinate reporting, as explained in Chapter 5. However, CTDIAG also produces a number of error messages, which are explained in this appendix.

Table A-1. Error Messages

Message:	An attempt to initialize the touch system using the command line parameters failed. Check the command line parameters for correct values. Check to insure that the touch system is installed properly. Press ‘ENTER’ to proceed to the configuration menu or press ‘ESC’ to exit.
Meaning:	The specified command line parameters failed to initialize the touch system.
Action:	Take the corrective action specified and either proceed to the Test Configuration Menu or exit.
Message:	An attempt to initialize the touch system using the CTDIAG touch system default values failed. If your touch system is set up for different values, use the command line options to define your touch system. Check to insure that the touch system is installed properly. Press ‘ENTER’ to proceed to the configuration menu or press ‘ESC’ to exit.
Meaning:	The CTDIAG default values failed to initialize the touch system.
Action:	Take the corrective action specified and either proceed to the Test Configuration Menu or exit.

Table A-1. Error Messages (Continued)

Message:	`Autodetection failed. No Touch System listed in CTDIAG.CFG was found.`
	`Check the CTDIAG.CFG file to insure your touch system is specified in an entry.`
	`Check to insure that the touch system is installed properly.`
	`Press ‘ENTER’ to proceed to the configuration menu or press ‘ESC’ to exit.`

Meaning:	Autodetection has examined all entries in the `.CFG` file and has not detected any of the configurations.
Action:	Take the corrective action specified and either proceed to the Test Configuration Menu or exit.
Message:	`CMDERR Indicated in Report Packet - Report Type`
Meaning:	A report indicates an error occurred. This indication appears as a non-zero in the third byte of the report. The report type is the value contained in the fourth byte of the report and appears in the error message as follows: ProtocolVersion Report, SwitchToClassicSfp Report, TouchState Report, TouchMode Report, CoordinateRanges Report, Waveform Report
Action:	Select Debug from the Test Menu and examine the Trace history to determine more about the condition.
Message:	`COM`*`x`*` does not exist`
Meaning:	The selected comm port cannot be detected.
Action:	Type a command line with the proper serial port identified by the `C` switch.
Message:	`Controller initialized ok but frame not found`
Meaning:	The controller is an HBC and no frame is detected.
Action:	Autodetection proceeds to the next `.CFG` file configuration. If the desired configuration is the HBC configuration, terminate and ensure the frame is properly attached to the HBC.

Table A-1. Error Messages (Continued)

Message:	`Driver is not installed`
Meaning:	The appropriate TAPI driver for the controller must be installed in order to execute.
Action:	Install the appropriate TAPI driver (HBC, SBC, RS-232).
Message:	`Frame fail to respond`
Meaning:	CTDIAG cannot communicate with the HBC or RS-232 controller or Smart-Frame.
Action:	• Check that power is supplied to the RS-232 controller or Smart-Frame. • Check that the communication cable is properly connected to the correct comm port. • Check that the controller or Smart-Frame jumpers (if any) are set to either autobaud or to a fixed baud rate and parity that matches those shown on the Status line. • Check the Status line to see that the stop bit is set to 1. • Check the Status line to see if the COM parameter is set to the comm port to which the touch system is connected.
Message:	`Hardware missing, poor connection, incorrect address or interrupt`
Meaning:	CTDIAG cannot communicate with the SBC in the Touch Coordinates screen.
Action:	• Check that the controller card is installed properly in the PC. • Check that the frame is connected to the controller. • Check to ensure that the I/O address and interrupt number settings on the controller match those on the Status line. • Select another I/O address and interrupt number setting to avoid possible conflict with other hardware in the PC. The addresses of other hardware may be found in the IBM AT I/O Port Map of your controller installation instructions.

Table A-1. Error Messages (Continued)

Message:	HBC TAPI Driver Not Found at S/W Interrupt *xx*H
Meaning:	CTDIAG cannot find the HBC TAPI driver at the indicated software interrupt.
Action:	Install the proper TAPI driver at the desired software interrupt. Otherwise, select the proper controller interface method and/or software interrupt using the CTDIAG menus.
Message:	Invalid Comm Port
Meaning:	CTDIAG could not detect the specified comm port in the Comm Port menu.
Action:	Select a valid comm port.
Message:	Invalid command line parameters... (followed by a list of valid command line parameters)
Meaning:	An incorrect command line parameter was encountered when loading CTDIAG.
Action:	Enter the correct command line parameters as indicated by the error printout.
Message:	Invalid Report Packet Received - Report Type
Meaning:	The expected report was not received.
Action:	Select Debug from the Test Menu and examine the Trace history to determine more about the condition.
Message:	Invalid Report Received - Configuration Report
Meaning:	This error occurs during a request for System Information. A report was not received after issuing a *Get_Configuration_Report* command.
Action:	Select Debug from the Test Menu and examine the Trace history to determine more about the condition.

Table A-1. Error Messages (Continued)

Message:	Invalid Report Received - Error Report
Meaning:	This error occurs during a request for System Information. A report was not received after issuing a *Get_Error_Report* command.
Action:	Select Debug from the Test Menu and examine the Trace history to determine more about the condition.
Message:	Invalid Report Received - Failed Beam Report
Meaning:	This error occurs during a request for System Information. A report was not received after issuing a *Get_FailedBeam_Report* command.
Action:	Select Debug from the Test Menu and examine the Trace history to determine more about the condition.
Message:	Invalid Report Received - Firmware Version Report
Meaning:	This error occurs during a request for System Information. A report was not received after issuing a *Get_FirmwareVersion_Report* command.
Action:	Select Debug from the Test Menu and examine the Trace history to determine more about the condition.
Message:	Invalid Report Received - GetFrameSize Report
Meaning:	This error occurs during a request for System Information. A report was not received after issuing a *Get_FrameSize_Report* command.
Action:	Select Debug from the Test Menu and examine the Trace history to determine more about the condition.
Message:	Invalid Report Received - State Report
Meaning:	This error occurs during a request for System Information. A report was not received after issuing a *Get_State_Report* command.
Action:	Select Debug from the Test Menu and examine the Trace history to determine more about the condition.

Table A-1. Error Messages (Continued)

Message:	`IO address is not on an 8 byte boundary***`
Meaning:	The selected RS-232 I/O address is not on an 8-byte boundary.
Action:	Change the I/O address to a correct address on an 8-byte boundary. The last digit must equal 0 or 8, such as 2F0H or 2F8H.
Message:	`IO address is not on an 16 byte boundary***`
Meaning:	The selected HBC I/O address is not on a 16 byte boundary.
Action:	Change the I/O address to a correct address on a 16 byte boundary. The last digit must equal 0, such as 2F0H.
Message:	`IO address is not within range***`
Meaning:	The selected I/O address is not within the allowable address range.
Action:	Select an I/O address within the acceptable range of the controller or comm port.
Message:	`No CTDIAG.CFG found...`
Meaning:	CTDIAG attempts to execute autodetection and no `.CFG` file can be found.
Action:	If no `.CFG` file is found, CTDIAG uses the autodetection command line parameters or, if none, the CTDIAG default parameters. If desired, terminate CTDIAG, copy a `.CFG` file into the current directory, and restart.
Message:	`No serial ports found`
Meaning:	CTDIAG found no serial ports installed in the PC when the Serial (RS-232) option was selected from the Test Configuration Menu.
Action:	You must have a serial port in the PC to use CTDIAG to test an RS-232 controller or Smart-Frame system.

Table A-1. Error Messages (Continued)

Message:	No VALID Comm Port Address found
Meaning:	CTDIAG could not find an entry in the comm port table maintained in RAM by the ROM BIOS that matched either the default COM1 I/O address of 3F8H or the default COM2 I/O address of 2F8H.
Action:	This error is usually associated with the use of multi-port comm boards. Make sure that the comm board is configured so that at least one comm port is at one of the default I/O addresses.
Message:	Out of Range
Meaning:	The I/O address value given in the I/O Address menu was outside of the range of valid I/O addresses.
Action:	Select a value that lies within the valid range (200H through 3F0H) of I/O addresses.
Message:	Report Packet Expected but not Received - Report Type
Meaning:	A command was sent but no report packet was received.
Action:	Select Debug from the Test Menu and examine the Trace history to determine more about the condition.
Message:	Requires a VGA system
Meaning:	CTDIAG requires a VGA video system to run some tests.
Action:	Install a VGA video adapter.
Message:	RS232 TAPI Driver Not Found at S/W Interrupt *xx*H
Meaning:	CTDIAG cannot find the RS-232 TAPI driver at the indicated software interrupt.
Action:	Install the proper TAPI driver at the desired software interrupt. Otherwise, select the proper controller interface method and/or software interrupt using the CTDIAG menus.

Table A-1. Error Messages (Continued)

Message:	`The SBC controller type cannot be used with the Direct interface method`
Meaning:	A test is attempted with the SBC and the direct interface method selected.
Action:	If the SBC controller is selected, the TAPI interface method must be selected.
Message:	`Touch did not de-activate. Touch is still enabled.`
Meaning:	An attempt was made to disable the touch using the *SetTouchMode* command.
Action:	Select Debug from the Test Menu and examine the Trace history to determine more about the condition.
Message:	`Touch System Initialization Failed`
Meaning:	CTDIAG cannot successfully complete initialization.
Action:	Select Debug from the Test Menu and examine the Trace history to determine more about the condition.
Message:	`Touch System Init Failed - Corrupted Report Received`
Meaning:	An expected report was corrupted during initialization.
Action:	Select Debug from the Test Menu and examine the Trace history to determine more about the condition.
Message:	`Touch System Init Failed - Expected Report Not Received`
Meaning:	An expected report was not received during initialization.
Action:	Select Debug from the Test Menu and examine the Trace history to determine more about the condition.
Message:	`Touch System Init Failed - Unable To Open Com Port`
Meaning:	The touch system cannot communicate with the comm port correctly. This error usually occurs on the first attempt to initialize the touch system.
Action:	Check for the correct comm port and I/O address specified on the command line.

Table A-1. Error Messages (Continued)

Message:	`Touch System Init Failed - Unable To Send Command`
Meaning:	The system is unable to send one of the commands in the initialization process.
Action:	Select Debug from the Test Menu and examine the Trace history to determine more about the condition.
Message:	`Unable to open file`
Meaning:	CTDIAG is unable to open a DOS file in which to place a copy of the Trace buffer.
Action:	Check that there is space available on the disk and that the disk is not write-protected.
Message:	`Unable to Send Command - Coordinate Ranges Report`
Meaning:	An attempt to send the *GetCoordinateRange* command is not completed.
Action:	Select Debug from the Test Menu and examine the Trace history to determine more about the condition.
Message:	`Unable to Send Command - SetTouchModes Command`
Meaning:	An attempt to send the *SetTouchMode* command is not completed.
Action:	Select Debug from the Test Menu and examine the Trace history to determine more about the condition.

B

The CTDIAG.CFG File

The autodetection process uses `CTDIAG.CFG`, a file that contains a list of commonly used touch system configurations. Details on the `CTDIAG.CFG` file are given in this appendix, in the following topics:

- Autodetection and the CTDIAG.CFG File.
- CTDIAG.CFG File Entry Definitions.

Autodetection and the CTDIAG.CFG File

To determine the type of touch system in use, CTDIAG attempts to communicate with the touch system using a list of commonly used touch system configurations, contained in the `CTDIAG.CFG` file. This process is known as **autodetection**. By default, `CTDIAG.CFG` is in the `c:\ctouch\ctdos\ctdiag` directory.

Each touch system configuration in the file has a header of `[SearchListEntry]`. Table B-1 contains an example of the first few entries from a typical `CTDIAG.CFG` file.

Table B-1. Example of CTDIAG.CFG File

```
[SearchListEntry]
Interface Method = TAPI Interrupt
TAPI Interrupt = 55H

[SearchListEntry]
Interface Method = Direct
Controller Type = Serial (RS232)
Comm Port = Com1
Baud Rate = 9600
Parity = Even

[SearchListEntry]
Interface Method = Direct
Controller Type = HBC
I/O Address = 300H
Hardware Interrupt = 7

[SearchListEntry]
Interface Method = Direct
Controller Type = Serial (RS232)
Comm Port = Com2
Baud Rate = 9600
Parity = Even
```

When CTDIAG starts, it refers to the first entry in `CTDIAG.CFG`, then tries to communicate with the touch system using the parameters in that entry. Using the first entry in Table B-1, for example, autodetection would attempt to communicate with the controller using the TAPI interrupt method and the appropriate TAPI driver (via interrupt 55H) for your controller.

If communication is successful, CTDIAG displays a line summarizing the current configuration, then the message `Autodetection complete.` The Test Menu then appears.

If communication is not successful, CTDIAG refers to the next `[SearchListEntry]` and attempts the search again. As the search progresses, messages appear indicating the configurations that were not found. If CTDIAG reaches the end of the `CTDIAG.CFG` file without finding a match, the message `Autodetection failed` appears, followed by the Test Configuration Menu.

Once autodetection has been run successfully, a description of the successful configuration is placed at the beginning of the `CTDIAG.CFG` file, under the header `[LastFound]`. A sample `[LastFound]` entry is shown in Table B-2.

Table B-2. Example of Last Found Entry in CTDIAG.CFG File

```
[Last Found]
Interface Method = Direct
Controller Type = Serial(RS232)
Comm Port = Com1
Baud Rate = 2400
Parity = Even
```

Because `[LastFound]` is the first entry in the file, CTDIAG will look for this configuration first the next time you invoke CTDIAG, speeding the autodetection process.

If the default `CTDIAG.CFG` file that comes with CTDIAG does not contain the configuration for your particular touch system - perhaps you use COM3 or COM4, for example - you may wish to edit `CTDIAG.CFG` to add that configuration.

CTDIAG.CFG File Entry Definitions

You may edit the `CTDIAG.CFG` file directly to add an entry for your particular touch system configuration.

Entry Types

Two types of entries are found in the `CTDIAG.CFG` file:

- `[LastFound]` - specifies the last controller detected during autodetection. This entry is always the first entry in the `CTDIAG.CFG` file.
- `[SearchListEntry]` - specifies the most probable list of controller configurations that may be used. A default `CTDIAG.CFG` is included with CTDIAG.

Definition of Items

To define either a`[LastFound]` or `[SearchListEntry]` entry type, use the following definition items. The default values are bolded.

- Interface Method: **Direct**, TAPI polling, TAPI interrupt.
- Controller type: HBC, **serial** (**RS-232**).
- Comm port: **COM1**, COM2, COM3, COM4.
- Baud rate: 300, 600, 1200, 2400, 4800, **9600**, 19200, 38400, 115200.
- Parity: None, **even**, odd.
- I/O address: HBC address (200 - 3F0) (**300H**).
- Hardware interrupt: HBC interrupt (2, 3, 4, 5, **7**).
- Software interrupt: TAPI software interrupt (50 - 70) (**55**).

The definition items you use to define an entry type will vary according to the type of controller or interface method.

Serial (RS-232) Entry

The definition items used to specify a serial (RS-232) entry are:

- Interface method.
- Controller type.
- Comm port.
- Baud rate.
- Parity.

Any item not defined will use the default. If I/O address or hardware interrupt are specified, they are ignored. To specify a non-standard comm port address and/or interrupt, use the command line parameters (Chapter 9) or the RS-232 Communication Parameters Menu (Chapter 4).

HBC Entry

The definition items used to specify an HBC entry are:

- Interface method.
- Controller type.
- I/O address.
- Hardware interrupt.

Any item not defined will use the default.

TAPI Entry

The definition items used to specify a serial (RS-232) entry are:

- Interface method.
- Software interrupt.

No other items are valid.

Comments

Comments may be used within the `CTDIAG.CFG` file. If the comment designator (`;`) is placed as the first character of a line, no translation of that line is attempted. No comment lines should be placed in the `[Last Found]` entry. All comment lines are preserved upon update of the `CTDIAG.CFG` file.

Glossary

Add Exit Point Modifier

A modifier that can be added to any of the four touch reporting types (*Continuous Mode*, *Enter Point Mode*, *Exit Point Mode*, *Tracking Mode*) under the SFP and that reports the coordinates at which the stylus exits the screen.

axis (x-axis, y-axis, z-axis)

A dimension that makes up the touch coordinate system. The x-axis is the horizontal axis and the y-axis is the vertical axis. The z-axis, available on guided wave systems only, measures the pressure placed on the screen.

baud rate

The speed of data transfer between a peripheral and the computer. This is a communication parameter used by the serial (RS-232) controller.

beam

An infrared light beam emitted by an infrared light-emitting diode (LED) and received by a phototransistor, which are set opposite each other in the touch frame. *Also called* a physical beam or opto-pair.

Beam Trap Mode

An operating mode within scan reporting. When a beam is interrupted, the corresponding grid boxes surrounding the touch active area are highlighted. The grid boxes remain highlighted, rather than turning off once the interruption is removed.

bezel

The plastic protective housing of the touch frame or touch screen, which either replaces or is fastened over the existing bezel of the monitor.

broken beam

A beam in which the infrared light level received by the infrared sensitive phototransistor falls below a threshold value set by the touch system firmware. In normal operation, this is due to a stylus obstructing the beam path from the LED to the phototransistor. Broken beams may also result from a defective LED, phototransistor, or other touch system hardware, and from other obstructions of the beam path. *See* beam.

comm port

The serial (RS-232) port on the back of the computer to which a touch frame or touch screen is connected.

communication parameters

The variables that control the transfer of information between the controller and the computer. For serial (RS-232) controllers, the variables are comm port, parity, baud rate and stop bits. For hardware- and software-based controllers, the variables are I/O address and interrupt number.

Continuous Mode
A touch reporting type under the SFP that reports touch coordinates at intervals from the time a stylus enters the screen until it exits, even if the stylus is unmoving. Add Exit Point can be added as a modifier.

controller
The interface between the touch system and the computer. The controller may be software-based or hardware-based, or may use the computer's serial (RS-232) port. *See* hardware-based controller, serial (RS-232) controller, and software-based controller.

coordinate origin corner
The 0, 0 origin of the video coordinate system. This is usually in the upper left corner of the display.

coordinate reporting
One of two methods that determines the form used to send data from the touch system to the host. The touch system reports x, y coordinate values that identify the touch location. The x and y values are reported as logical coordinates. *See also* scan reporting.

coordinates
The two-dimensional mathematical representation of a point. For example, 29, 38 represents a value of 29 on the x-axis and a value of 38 on the y-axis.

`CTDIAG.CFG` file
The CTDIAG configuration file used during autodetection.

Enter Point Mode
A touch reporting type under the SFP that reports only the coordinates at which a stylus enters the touch screen. Add Exit Point reporting can be added as a modifier.

Exit Point Mode
A touch reporting type under the SFP that reports only the coordinates at which the stylus exits the touch screen. Add Exit Point reporting can be added as a modifier; if you do this, however, the touch system reports the coordinates at which the stylus exits the screen twice, first as a normal Coordinate Report and then as an Add Exit Point Coordinate Report.

Formatted Mode
A mode used in the receive window of the Debug screen. Each command and associated report appear on a separate line for easier viewing.

frame size
A general term for either logical or physical frame size. *See* logical frame size, physical frame size.

guided acoustic wave touch technology
A technology used in touch screens, based upon transmitting acoustic waves through a glass overlay placed over the display surface.

hardware-based controller (HBC)

A touch-system-independent, digital controller containing a microprocessor. The HBC is a half-card installed in the computer, drawing its power through the PC bus and communicating through the bus using an I/O address and hardware interrupt.

hardware interrupt

A dedicated hardware line between the touch system and the computer, defining where to search for an SBC or HBC.

HBC

See hardware-based controller (HBC).

HBC driver

A software program that interfaces with the hardware-based controller via a selectable I/O address and optional hardware interrupt and with the application via the touch application program interface (TAPI).

host (host system)

The computer system to which a touch system is added.

I/O address

A parameter that defines the hardware base address location of an SBC's or HBC's hardware registers on the controller card.

infrared touch technology

A technology used in touch systems, based upon superimposing a grid of invisible infrared beams in front of the display surface.

interrupt number

A parameter that defines where to search for an SBC or HBC controller.

logical (virtual) beam

A member of a set of beams that includes both the physical beams and the interpolated virtual beams, which are imaginary beams that occupy the spaces between physical beams. In each axis, the number of logical beams is twice the number of physical beams minus one.

logical coordinates

A coordinate system consisting of an x- and y-axis, each made up of logical beams. The origin of the coordinate system is the upper left corner of the display/touch frame.

logical frame size

The logical frame size is double the physical frame size minus one, since the space between the opto-pair is considered a virtual beam. *See* beam, physical frame size.

MDI

See modular digital interface (MDI).

modular digital interface (MDI)

The touch-system-to-controller interface created by confining all of the analog functions to the touch system. The MDI makes a standard touch system controller-independent and reduces the touch system cabling requirement to a simple 8-pin standard phone cable up to six feet in length. This extends the allowable distance between the touch system and controller and improves noise immunity.

option

A selection, test, or function available on a menu.

opto-pair

An infrared light-emitting diode (LED) matched with a phototransistor and set opposite one another in an infrared touch frame (opto-matrix frame) and pulsed sequentially so as to send and receive a single beam of infrared light. *See* beam.

parity

A parameter used for error-checking to ensure that the data that was transmitted by the serial peripheral is identical to that received by the computer, and vice versa. This is a communication parameter used by the RS-232 controller.

physical beam

An infrared light beam passing between an LED/phototransistor pair. *See* beam.

physical coordinates

A coordinate system consisting of an x- and y-axis, each made up of physical beams. The origin of the coordinate system is the upper left corner of the display/touch screen.

physical frame size

The actual number of opto-pairs (beams) that make up each axis of the infrared frame. *See* beam, logical frame size.

READ.ME

A file on the installation disk that contains additional or late-breaking information that could not be placed in this guide.

RS-232 controller

See serial (RS-232) controller.

SBC

See software-based controller (SBC).

SBC driver

A software program that interfaces with the software-based controller via a selectable I/O address and hardware interrupt and with the application via TAPI.

scan reporting

One of two methods that determines the form used to send data from the touch system to the host. The touch system reports a list of physical beams that are interrupted in each axis. *See also* coordinate reporting.

scanning infrared technology

See infrared touch technology.

serial (RS-232) controller

A touch-system-independent, digital controller containing a microprocessor. This controller requires an external +12V power supply and communicates using a standard RS-232 serial port.

SFP

See Smart-Frame Protocol (SFP).

Smart-Frame

A Carroll Touch touch frame with built-in serial (RS-232) controller.

Smart-Frame Protocol (SFP)

A Carroll Touch proprietary set of commands and reports used to communicate with Carroll Touch touch systems. This protocol is used by virtually all Carroll Touch infrared touch systems.

software-based controller (SBC)
A touch-system-independent, digital controller that has no microprocessor, but, instead, shares processor time with the host microprocessor. The SBC is a half-card installed in the computer, drawing its power through the PC bus and communicating through the bus using an I/O address and hardware interrupt.

stop bit
The number of bits, usually 1, inserted by the serial (RS-232) controller into the data stream to inform the host computer that the transmission of a byte of data is complete. This is a communication parameter used by the RS-232 controller.

sync error
An error that occurs when the beam number provided by the SBC hardware at the time of an SBC hardware interrupt is not one beam greater than the beam number provided by the previous SBC hardware interrupt.

TAPI
See Touch Application Program Interface (TAPI).

TAPI driver
A software program that interfaces with the touch system controller and with TAPI.

TAPI interrupt
The software interrupt used to communicate with the TAPI driver.

touch active area
The area inside the touch frame or touch screen that is sensitive to touch.

Touch Application Program Interface (TAPI)
A Carroll Touch proprietary series of software function calls within the individual controller driver that interfaces a touch application to a touch system using the Smart-Frame Protocol. TAPI enables a properly-written application to interface with any Carroll Touch controller driver that communicates serially or through the PC bus and that recognizes the Smart-Frame Protocol.

touch frame
A rectangular assembly of circuit boards. Two adjacent circuit boards contain banks of infrared light-emitting diodes (LEDs), while the opposite two contain banks of complementary phototransistor-receivers. The LEDs and phototransistors create a grid of invisible infrared light. The opto-electronics are concealed behind an IR transparent bezel, which shields the opto-electronics from the operating environment, while allowing the IR beams to pass through.

touch mode
A touch mode that determines when touch coordinates are reported. *Enter Point Mode* reports only the coordinates at which a finger or stylus enters the touch screen. *Exit Point Mode* reports only the coordinates at which the stylus exits the touch screen. *Tracking Mode* reports the coordinates of the entry of the touch screen and all movement within the screen. *Continuous Mode* generates reports at intervals from the time the screen is entered until it is exited, even though the finger or stylus is unmoving. Exit Point reporting can be added as a modifier to the four basic reporting types.

Touch Reporting Mode
A method that determines the form used to send data from the touch system to the host. In scan reporting, the touch system reports a list of physical beams that are interrupted in each axis. In coordinate reporting, the touch system reports x, y coordinate values that identify the touch location.

touch screen
A glass overlay containing a transducer and reflectors, placed over the display surface. The transducer is mounted on the edge of the glass and emits an acoustic wave, which travels through the glass overlay as well as over the surface and is reflected back across the overlay. Since the speed of the wave is known and the size of the glass overlay is fixed, the first reflector will send the first signal back first, then the second, and so on.

touch system
The collection of all the components that are necessary to detect a touch and report it to the host. This collection usually consists of the touch frame or screen, protective bezel, controller, and software.

trace buffer
A portion of memory used by CTDIAG to record all communication between the touch system and controller. The communication trace buffer can be viewed through Debug.

Tracking Mode
A touch reporting type under the SFP that reports the coordinates of the initial touch on the touch screen, as well as all movement within the screen. Add Exit Point can be added as a modifier.

Unformatted Mode
A mode used in the receive window of the Debug screen. Under the SFP, commands and corresponding reports fill a line, then wrap around to the next line.

virtual beam
Imaginary beams that occupy the spaces between physical beams.

Index

CONTACTING CARROLL TOUCH:

Carroll Touch
2800 Oakmont Drive
Round Rock, Texas 78664
http://www.carrolltouch.com

(512) 244-3500	Switchboard
(800) 386-8241	Toll Free
(512) 388-5509	Technical Support
(512) 244-7040	Fax