Touch Controllers

Reference Guide

MicroTouch®

Document Title: *Touch Controllers Reference Guide*
Document Number: 19-213, Version 2.2

Contents

About This Manual

CHAPTER 1 MicroTouch Touchscreen Controllers

APPENDIX C TouchPen Controllers

APPENDIX D Daughterboard Controllers

Index

About This Manual

This reference manual, which is intended for developers of touch systems, provides information about the MicroTouch touchscreen controllers and available firmware commands. The description of each command includes the command syntax, the default value, how the command works, and the response from the controller.

Developers may use this information when writing touch applications, developing a custom driver or touch configuration, or testing their touch systems. Developers can use firmware commands to initialize the controller, select operating modes, specify data formats, and execute diagnostic functions.

Most touchscreen users do not have to use firmware commands in order to use their touch systems. Many changes that can be made with firmware commands can be made using *TouchWare*, the software delivered with the touchscreen. For example, users can use TouchWare to calibrate the touchscreen, to determine the controller type and firmware version, and to specify the active pen mode.

What You Need to Know

This document assumes you are familiar with firmware commands and how to use them. Executing some commands may alter the performance of your touch product. You should be aware of the results of using these commands before executing them.

MicroTouch Support Services

MicroTouch provides extensive support services through our technical support organization, web site, and bulletin board system (BBS).

MicroTouch Technical Support

Technical Support is available as follows:

- 24 hours a day, Monday through Friday (excluding holidays)
- 9:00 a.m. to 5:00 p.m. Eastern Standard Time, Saturday and Sunday (excluding holidays)

Whenever you contact Technical Support, please provide the following information:

- Part number and serial number from the MicroTouch label on your monitor or touchscreen controller
- Type of MicroTouch touchscreen
- Version number of your MicroTouch TouchWare
- Make and model of your personal computer
- Name and version number of your operating system
- Type of mouse connected to your system
- List of other peripherals connected to your computer
- List of application software in use

You can contact MicroTouch Technical Support by calling the hot line, sending a fax, or sending electronic mail.

- Technical Support Hot Line: 978-659-9200
- Technical Support Fax: 978-659-9400
- Technical Support E-Mail: support@microtouch.com

MicroTouch on the World Wide Web

You can visit the MicroTouch web site at the following address:

http://www.microtouch.com

You can download MicroTouch touchscreen software and drivers, obtain regularly updated technical information on MicroTouch products, and learn more about our company.

MicroTouch Bulletin Board System

MicroTouch also has a Bulletin Board System (BBS) that you can access 24 hours a day, 7 days a week. You can use the BBS to download updates of the latest drivers and obtain regularly updated technical information on MicroTouch products.

You can reach the MicroTouch BBS at the following numbers:

- 978-659-9250
- 978-683-0358

To connect to the BBS, you need standard communication software and a modem that supports 2400, 4800, 9600, 14400, or 28800 baud. Additionally, the communication parameters must be set as follows:

No parity, 8 data bits, and 1 stop bit (N81)

Once you establish a modem connection with the BBS, the system prompts you to log in using your name. You can register with MicroTouch the first time you log in to the BBS. The menu of available options is self-explanatory.

MicroTouch Corporate Headquarters and Worldwide Offices

United States

MicroTouch Systems, Inc.
300 Griffin Brook Park Drive
Methuen, MA 01844
Main Phone: 978-659-9000
Main Fax: 978-659-9100
Web Site: http://www.microtouch.com
E-Mail: touch@microtouch.com
Tech Support Hot Line: 978-659-9200
Tech Support Fax: 978-659-9400
Tech Support E-Mail: support@microtouch.com

Australia

MicroTouch Australia, Pty Ltd.
797 Springvale Road
Mulgrave Victoria 3170 Australia
Phone: +61 (03) 9561 7799
Fax: +61 (03) 9561 7393
Web Site: http://www.microtouch.com.au
E-Mail: touch@microtouch.com.au
Tech Support E-Mail: support@microtouch.com.au

France

MicroTouch Systems SARL
Europarc de Créteil
19, rue Le Corbusier
94042 Créteil Cedex France
Phone: +33 (1) 45 13 90 30
Fax: +33 (1) 45 13 90 34

Germany

MicroTouch Systems GmbH
Schiess-Str. 55
40549 Düsseldorf Germany
Phone: +49 (0) 211-59907-0
Fax: +49 (0) 211-599 06 55

Hong Kong

MicroTouch Systems, Inc.
Unit D, 9/F, Trust Tower
68 Johnston Road
Wanchai, Hong Kong, China
Phone: +852 2333 6138; +852 2334 6320
Fax: +852 2333 6861

Italy

MicroTouch Systems srl
Via Solferino, 12a
20052 Monza (MI) Italy
Phone: +39 (0) 39-230-2230
Fax: +39 (0) 39-230-2370

Japan

MicroTouch Systems K.K.
Bellevue Mizonokuchi Building 3F,
3-2-3, Hisamoto, Takatsu-ku,
Kawasaki-shi, Kanagawa 213 Japan
Phone: +81 (044) 811-1133
Fax: +81 (044) 811-1143

Korea

MicroTouch Systems, Inc.
#402, 4th Floor, Nam-Kyung Building
769-6 Yeoksam-Dong, Kangnam-Gu
Seoul, Korea
Phone: +82 (2) 552-3198
Fax: +82 (2) 552-3210

Taiwan R.O.C.

MicroTouch Systems, Inc.
3F-12, No. 351, Chung Shan Road, Sec. 2
Chung Ho City, Taipei
Taiwan R.O.C.
Phone: +886 (02) 2226-0875
Fax: +886 (02) 2226-4824

United Kingdom

MicroTouch Systems, Ltd.
163 Milton Park
Abingdon
Oxon OX14 4SD
England
Phone: +44 (0) 1235-444400
Fax: +44 (0) 1235-861603
BBS: +44 (0) 1235-861620

CHAPTER 1

MicroTouch Touchscreen Controllers

MicroTouch offers several advanced low-power, surface-mount controllers that are designed for reliability and easy installation. Each controller provides superior performance and delivers unparalleled sensitivity, accuracy, and fast response.

This chapter presents the following information:

- Overview of the MicroTouch touchscreen controllers
- Factory default settings for each controller
- How to initialize the controller
- How to communicate with the controller (send commands and receive responses)

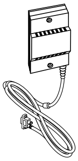
Overview of Touchscreen Controllers

MicroTouch offers controllers for both capacitive touchscreens and resistive touchscreens. Table 1 lists the MicroTouch touchscreen controllers, including name, part number, technology supported, and mounting options.

Table 1. MicroTouch Touchscreen Controllers

Controller	Part Number	Technology	Mounting
Serial/SMT2	14-05 14-09 (no case)	Capacitive	External or internal
Serial/SMT3 (Replaced by Serial/SMT3V)	14-78	Capacitive	External or internal
Serial/SMT3V	14-78 14-88 (no case)	Capacitive	External or internal
Serial/SMT3R	44-83	Resistive	External or internal
Serial/SMT3RV	44-94	Resistive	External or internal
Serial/SMT2 Daughterboard	14-73	Capacitive	On CPU board
Serial/SMT3V Daughterboard	14-89	Capacitive	On CPU board
PC Bus SMT2	14-34	Capacitive	In 16-bit PC expansion slot
PC Bus SMT3V	14-99	Capacitive	In 16-bit PC expansion slot
PC Bus SMT3RV	44-85	Resistive	In 16-bit PC expansion slot
TouchPen 4	64-65	Capacitive Digitizer	Internal
TouchPen 4+	64-68	Capacitive Digitizer	Internal
MousePort	14-82	Capacitive	External or internal
Chip Sets	Several options available	Capacitive Resistive	Integrated into the design of your system board

Serial/SMT Controllers

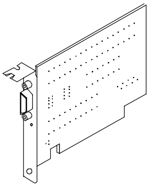


The Serial/SMT controllers are compact (3.5 x 2.25 x 0.3 inches), RS-232 serial controllers. The controller can be internally mounted in your monitor, or enclosed in a molded plastic case (3.75 x 2.5 x 0.9 inches) and mounted to the back or side of your monitor.

Daughterboard Controllers

The Daughterboard controller is a CMOS serial add-on board (3.5 x 2.25 x 0.3 inches) that you mount onto your CPU board. You can easily integrate the Daughterboard controller onto a system board you may be designing.

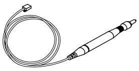
PC Bus Controllers



The PC Bus controller is a half-slot, bus card that you install in your system. It has its own serial communication (COM) port, enabling you to use your existing COM ports for other peripherals.

To use the PC Bus controller, your computer must have an available 16-bit ISA (Industry Standard Architecture) expansion slot. The touchscreen cable connects to the port on the controller.

TouchPen Controllers



The TouchPen controller offers the same features as the Serial/SMT capacitive controller, with the addition of pen support. The controller can accept touch input from both a finger and the touch pen.

This RS-232 serial controller, which measures a trim 1.35 x 4.8 x 0.3 inches, is designed to easily fit inside flat panel displays and CRTs. The TouchPen controller is always mounted internally.

The tethered touch pen attaches to your display. Several cable lengths are available.

MousePort Controller

The MousePort controller has an attached 8-foot, 6-pin mini-din PS/2 connector. You can connect this controller to a PS/2 mouse port, leaving your serial communication and bus slots available for other peripherals.

Chip Set Controllers

Chip sets are available to those developers who want to integrate a MicroTouch touchscreen controller directly into their own circuitry.

Chip sets include an optimized controller circuit that can be used with a MicroTouch capacitive or five-wire resistive touchscreen. Chip sets are designed for maximum flexibility and ease of implementation. For more information on the available chip set controllers, contact MicroTouch.

Identifying Your Touchscreen Controller

To identify your controller type, you can use the following firmware commands:

- Output Identity (OI)
- Unit Type (UT)
- Unit Type Verify (UV)

All controllers support the Output Identity command. The Unit Type and Unit Type Verify commands provide additional information about a controller. For details about these commands, refer to Chapter 2.

Controller Default Settings

Table 2 lists the default settings for each controller. The sections that follow provide more information on each setting.

Table 2. Controller Default Settings

Controller	Communication Parameters	AutoBaud	Data Format and Operating Mode
Serial/SMT2 Serial/SMT2 Daughterboard PC Bus SMT2 PC Bus SMT3V	N, 7, 2, 9600	Enabled (AE)	Format Decimal (FD) Mode Stream (MS)
Serial/SMT3V Serial/SMT3V Daughterboard Serial/SMT3RV PC Bus SMT3RV	N, 7, 2, 9600	Disabled (AD)	Format Decimal (FD) Mode Stream (MS)
Serial/SMT3 Serial/SMT3R	N, 8, 1, 9600	Not available	Format Tablet (FT) Mode Stream (MS)
TouchPen 4 TouchPen 4+	N, 8, 1, 9600	Not available	Format Tablet (FT) Mode Stream (MS) Pen or Finger (PF)
MousePort	Not applicable to mouse port	Not available	Format Tablet (FT) Mode Stream (MS)

Communication Parameters

Table 2 lists the default communication parameters for your controller. The communication parameters include the following variables:

- Parity type (N=none, O=odd, and E=even)
- Number of data bits (7 or 8)
- Number of stop bits (1 or 2)
- Baud rate (19200, 9600, 4800, 2400, 1200)

MicroTouch recommends that you use N, 8, 1 (no parity, 8 data bits, and 1 stop bit) and 9600 baud for most touch applications. All MicroTouch touchscreen drivers communicate with the controller at N, 8, 1, 9600.

For the TouchPen controller, MicroTouch recommends 19200 baud for use with character recognition systems that require high pen data rates. Systems that cannot handle the higher data rates seen from the pen may result in degraded pen performance. To improve performance on these systems, use a slower baud rate.

Automatic Baud Rate Detection

⚠ As listed in Table 2, AutoBaud Enable is the factory default for some touchscreen controllers. However, MicroTouch recommends that you *disable autobaud* for several reasons.

When AutoBaud is enabled, the controller changes its communication rate to that of the next command from the host system. Thereafter, the controller sets its communication rate to the first command received from the host system after powering on the unit.

Although the AutoBaud Enable command sets the communication rate, it does not automatically set the parity, the number of data bits, and the number of stop bits. The controller cannot communicate with the host system unless all communication parameters are the same.

Because this automatic feature is limited to the communication rate, MicroTouch recommends that you issue an AutoBaud Disable command to turn off the AutoBaud feature. After you disable AutoBaud, send a Parameter Set command to change the communication parameters (parity, number of data bits, number of stop bits, and baud rate).

Data Formats

Data format refers to the type of packet the controller uses to send the X, Y touch coordinate to the host system.

As listed in Table 2, the default data format for your controller may be Format Decimal or Format Tablet. However, MicroTouch recommends that you use Format Tablet for the following reasons:

- Format Tablet uses only 5 bytes per point and provides the most rapid response time to a touch.
- Format Tablet is the most efficient and most compact data format (sends approximately 192 packets per second at 9600 baud).
- Format Tablet includes a status byte. The status byte contains information on whether the X, Y coordinate is generated from a touchdown, a touch continuation (when the finger is resting on the screen), or a touch liftoff.
- Format Tablet is supported by all MicroTouch touchscreen controllers.
- Format Tablet is the standard for MicroTouch product development and is the format used by all touchscreen drivers written by MicroTouch.

There are several firmware commands that let you select a different data format, such as Format Binary, Format Decimal, Format Hexadecimal, and Format Zone. However, not all touchscreen controllers support these data formats. For more information on other data formats, refer to Chapter 2.

Operating Modes

The *operating mode* specifies the conditions under which the controller sends the X, Y touch coordinates (input data packet) to the host system.

Mode Stream is the default operating mode for all MicroTouch touchscreen controllers. In Mode Stream, the controller sends a continuous stream of data packets. The controller sends the data as long as the touch device (finger or pen) continues to touch the screen.

Because Mode Stream sends touch data continually, it is the most versatile mode and provides the best response time and overall feel.

MicroTouch recommends that the touchscreen generate an interrupt as each byte in the data stream arrives. Because touchdown and liftoff events are specially coded, your software always knows exactly what the user is doing and can provide instant feedback.

There are several firmware commands that let you select a different operating mode, such as Mode Point, Mode Down/Up, or Mode Polled. However, not all touchscreen controllers support these modes. Each mode specifies a different set of conditions for the transmission of the touch coordinates. The best operating mode for your application depends on the type of touch input required. For more information on these modes, refer to Chapter 2.

Controller Initialization

To achieve optimal touchscreen performance, MicroTouch recommends that you initialize your controller. The initialization commands vary depending on your controller.

Table 3 lists the firmware commands you should send to the controller during your factory configuration.

Additionally, if your controller supports the Output Status command, the controller automatically sends the output status information to the host on power-up. Therefore, the host can detect a power-up condition. Note that the controller sends the status information only if autobaud is disabled. If autobaud is enabled, the controller waits for the host to send a command.

For more information on each command, refer to Chapter 2.

Table 3. Initialization Commands

Controller	Initialization Command Sequence	Output Status
Serial/SMT2 Serial/SMT3V Serial/SMT3RV Serial/SMT2 Daughterboard Serial/SMT3V Daughterboard PC Bus SMT2 PC Bus SMT3V PC Bus SMT3RV	For these controllers, issue the following commands: Reset <SOH>R<CR> AutoBaud Disable <SOH>AD<CR> Parameter Set <SOH>PN812<CR> Format Tablet <SOH>FT<CR> Mode Stream <SOH>MS<CR> Parameter Lock <SOH>PL<CR>	These controllers send output status information on power-up.
Serial/SMT3 Serial/SMT3R TouchPen 4 TouchPen 4+ MousePort	For these controllers, you only need to send a Reset command after power-up.	Not applicable

Communicating with the Controller

This section provides information on sending firmware commands to and receiving commands from the MicroTouch controller.

Commands to the controller are provided on signal **Receive Data** (RXD) as a serial data stream. Responses are data sent from the controller to the host system in response to the commands received by the controller. Controller responses to the host system are provided on signal **Transmit Data** (TXD) as a serial data stream.

Sending Commands to the Controller

When you send a command to the controller, you must use the correct command format. The general format of a command is as follows:

<Header>Command<Terminator>

Note: The following descriptions of header, command, and terminator, use MicroTouch's terminal emulator key sequences. You may need to enter the sequence in a different format, depending on your emulator.

The *header* is the first character in the command string and is the ASCII start-of-header character <SOH>. The ASCII <SOH> character is equivalent to 01 hexadecimal. To start the command sequence, use the key sequence Ctrl A (^A).

The *command*, which always follows the header, consists of ASCII uppercase letters and numbers.

The *terminator* is the last character of each command string and is an ASCII carriage return <CR>. An ASCII <CR> character is equivalent to 0D hexadecimal. To end the command sequence, use Enter or the key sequence Ctrl M (^M).

This chapter lists each command as a string of ASCII characters consisting of a header, the command, and a terminator as follows:

<SOH>*Command*<CR>

Receiving Responses from the Controller

After executing a command, the controller returns a response or acknowledgment to the host system. Each controller response consists of a header, the command response, and a terminator in the following format:

<Header>Command Response<Terminator>

Note: The following descriptions of header, response, and terminator, use MicroTouch's terminal emulator key sequences. The format of controller responses varies depending on the terminal emulation mode you are using.

The *header* is the first character in the response string and is the ASCII start-of-header character <SOH>. An ASCII <SOH> character is equivalent to 01 hexadecimal. The value returned will be the ASCII key sequence Ctrl A (^A).

The *response*, which always follows the header, is usually a range of ASCII characters depending on the type of command received. Responses can be in many forms.

For example, one standard response is <SOH>**0**<CR> (ASCII character 'zero' or 30 hexadecimal). This response indicates a successful command completion. The controller received a valid command and executed the command properly.

Another standard response is <SOH>**1**<CR> (ASCII character 'one' or 31 hexadecimal). This response indicates the command failed. The controller received an invalid command and did not execute the command. Here are some possible reasons for the failure:

- The command was not formatted correctly.
- The system parameters were not set up to allow command execution.
- The controller does not support the command.

Note: There are exceptions to the meaning of these responses. For details on each command response, refer to Chapter 2.

The *terminator* is the last character of each response string and is an ASCII carriage return <CR>. An ASCII <CR> represents 0D hexadecimal. The value returned in the response will be the ASCII key sequence Ctrl M (^M).

In this chapter, responses are shown as a string of ASCII characters consisting of a header, the response, and a terminator as follows:

<SOH>*Response*<CR>

CHAPTER 2

Firmware Commands

Developers can use firmware commands to control the operation of the touchscreen controller. The firmware commands, which are usually issued by a driver or utility program on the host system, control the operation of the touchscreen controller.

This chapter

- Lists the available firmware commands
- Details which commands are available on the different MicroTouch controllers
- Describes how to use each firmware command

For each firmware command, this chapter includes information about command syntax, a description of each command, and the response you will receive from the controller when you use the command.

Summary of Firmware Commands

To optimize the performance of the touchscreen controllers and simplify the development of custom drivers, MicroTouch recommends you use the commands listed in Table 4.

Additionally, MicroTouch recommends that developers writing drivers or applications that communicate directly with our controllers use the commands listed in Table 4 regardless of their controller type. Using these commands ensures compatibility with all existing and future MicroTouch controllers.

Table 5 lists the MicroTouch touchscreen controllers and the firmware commands supported by each controller.

Table 4. Firmware Commands Recommended for Development

Command Name	ASCII Code	Description
Calibrate Extended	CX	Initiates an interactive, two-point calibration.
Calibrate Raw	CR	Collects the raw X and Y coordinates prior to normal scaling, linearization, and filtering process.
Finger Only	FO	Sets the TouchPen controller to accept only finger input.
Format Raw	FR	Returns the signal level (amount of touch) of each of the four touchscreen corners in digital format.
Format Tablet	FT	Outputs the X, Y touch coordinate data in a five-byte packet.
Get Parameter	GP	Returns all power-up and run time parameters used by the controller.
Mode Stream	MS	Sends a continuous stream of X, Y coordinate data when you touch the screen.
Null Command	Z	Queries the controller and waits for a response.
Output Identity	OI	Identifies the controller type and the firmware version.
Pen Only	PO	Sets a TouchPen controller to accept only pen input.
Pen or Finger	PF	Sets a TouchPen controller to accept both pen and finger input.
Reset	R	Initializes the hardware and the firmware, causes the controller to stop sending data, and recalculates the environmental conditions.
Restore Defaults	RD	Returns the controller to the factory default operating parameters.
Set Parameter	SP	Sets all power-up and run time parameters used by the controller.
Unit Type Unit Type Verify	UT UV	Identifies the type of touchscreen controller connected to your system.

Table 5. MicroTouch Touch Controller Firmware Commands

Command Name	ASCII Code	SMT2[1], PC Bus SMT2	SMT3V[1], SMT3RV, PC Bus SMT3RV PC Bus SMT3V	SMT3, SMT3R, MousePort	TouchPen 4 TouchPen 4+
Default Settings		N72, 9600 AE, FD, MS	N72, 9600 (AD/AE)[2], FD, MS	N81, 9600 FT, MS	N81, 9600 FT, MS, PF
AutoBaud Disable	**AD**	✓	Note 2		
AutoBaud Enable	**AE**	✓	Note 2		
Calibrate Extended	**CX**	✓	✓	✓	✓
Calibrate Interactive	**CI**	✓	✓		
Calibrate New	**CN**	✓	✓		
Calibrate Raw	**CR**	✓	✓	✓	✓
Filter Number	**FN***nn*	✓	✓		
Finger Only	**FO**		Note 3	Note 3	✓
Format Binary [Stream]	**FB[S]**	✓	✓		
Format Decimal	**FD**	✓	✓		
Format Hexadecimal	**FH**	✓	✓		
Format Raw	**FR**	✓	✓	✓	✓
Format Tablet	**FT**	✓	✓	✓	✓
Format Zone	**FZ**	✓	✓		
Frequency Adjust	**<Ctrl C>F***nn*	✓	Note 4		
Get Parameter Block	**GP***n*	✓	✓	✓	✓
Mode Down/Up	**MDU**	✓	✓		
Mode Inactive	**MI**	✓	✓		
Mode Point	**MP**	✓	✓		
Mode Polled	**MQ**	✓	✓		
Mode Status	**MT**	✓	✓		
Mode Stream	**MS**	✓	✓	✓	✓
Null Command	**Z**	✓	✓	✓	✓
Output Identity	**OI**	✓	✓	✓	✓
Output Status	**OS**	✓	✓		
Parameter Lock	**PL**	✓	✓		
Parameter Set	**P***pds*[*b*]	✓	✓		
Pen Only	**PO**				✓
Pen or Finger	**PF**				✓
Reset	**R**	✓	✓	✓	✓
Restore Defaults	**RD**	✓	✓	✓	✓
Sensitivity Set	**SE***n*	✓	✓		
Set Parameter Block	**SP***n*	✓	✓	✓	✓
Unit Type	**UT**			✓	✓
Unit Type Verify	**UV**		✓		

1. SMT2 includes the SMT2 Daughterboard; SMT3V includes the SMT3V Daughterboard.
2. All firmware versions support the AD command. The AE command is available only in version 5.4 and later. The PC Bus SMT3V controller uses AE as the default; the other controllers use AD.
3. Returns a 0 response, which indicates the controller received a valid command. However, the specified controllers perform no function when receiving the command. Provided for compatibility only.
4. Available only in firmware version 5.5 and later.

AutoBaud Disable

> **Note:** *MicroTouch provides this command for compatibility with older touchscreen controllers. To ensure compatibility with all existing and future MicroTouch controllers, do not use this command when developing your touch drivers and applications.*

Syntax: <SOH>**AD**<CR>

Description: Turns off the automatic baud rate detection feature.

When you disable AutoBaud, the controller maintains the communication rate currently set in non-volatile RAM (NOVRAM). The controller continues to use this communication rate until you change it with either the Parameter Set command or the AutoBaud Enable command.

AutoBaud Enable is the factory default for some touchscreen controllers. However, because this automatic feature is limited to the communication rate, MicroTouch recommends that you issue an AutoBaud Disable command to turn off the AutoBaud feature. After you disable AutoBaud, send a Parameter Set command to change the communication parameters (parity, number of data bits, number of stop bits, and baud rate).

Response: <SOH>**0**<CR> Positive response.

AutoBaud Enable

> **Note:** *MicroTouch provides this command for compatibility with older touchscreen controllers. To ensure compatibility with all existing and future MicroTouch controllers, do not use this command when developing your touch drivers and applications.*

Syntax: <SOH>**AE**<CR>

Description: Turns on the automatic baud rate detection feature.

When you enable AutoBaud, the controller changes its communication rate to that of the next command from the host system. Thereafter, the controller sets its communication rate to the first command received from the host system after powering on the unit.

Although the AutoBaud Enable command sets the communication rate, it does not automatically set the parity, the number of data bits, and the number of stop bits. The controller cannot communicate with the host system unless all communication parameters are the same.

AutoBaud Enable is the factory default for some touchscreen controllers. However, because this automatic feature is limited to the communication rate, MicroTouch recommends that you issue an AutoBaud Disable command to turn off the AutoBaud feature. After you disable AutoBaud, send a Parameter Set command to change the communication parameters (parity, number of data bits, number of stop bits, and baud rate).

Response: <SOH>**0**<CR> Positive response.

Calibrate Extended

Syntax: <SOH>**CX**<CR>

Description: Initiates an interactive, two-point calibration.

During the calibration process, you define the active area of the touchscreen by mapping locations to an absolute X, Y coordinate system. You touch two *target areas* on the screen. Touching the target areas sends the X, Y coordinates for those touch points to the controller. The controller calculates all other touch points based on these two points.

The Calibrate Extended command functions exactly like the Calibrate New command with one exception. For Calibrate Extended, the calibration targets (points) are set inward from the corner of the video image. This enhancement makes the calibration process easier and more accurate.

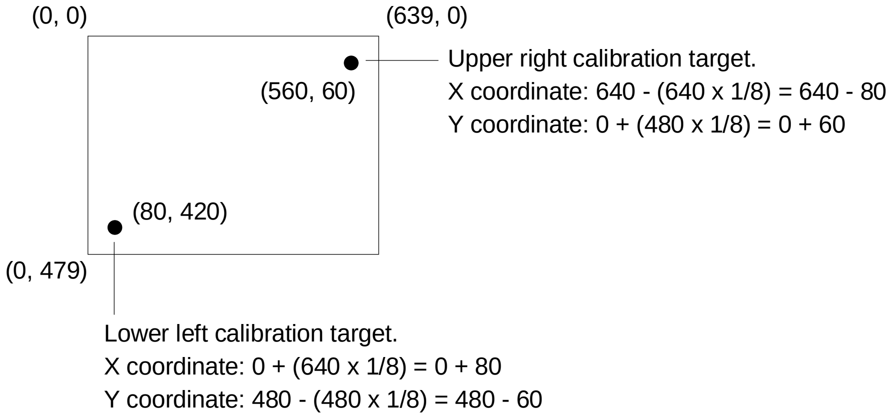
Determining Target Areas

The default calibration targets (points) are located 12.5% (1/8) inward from the corner of the video image.

For example, suppose the display resolution of your monitor is 640 x 480. The Calibrate Extended command calculates the amount to move inward as follows:

- Amount to move inward in the X direction: 640 x 1/8 = 80
- Amount to move inward in the Y direction: 480 x 1/8 = 60

The Calibrate Extended command then positions the first calibration target inward from the lower left corner (0,479) and the second calibration target inward from the upper right corner (639,0). The following illustration shows how the calibration targets are calculated.


(0, 0)
(639, 0)
(560, 60)
Upper right calibration target.
X coordinate: 640 - (640 x 1/8) = 640 - 80
Y coordinate: 0 + (480 x 1/8) = 0 + 60
(80, 420)
(0, 479)
Lower left calibration target.
X coordinate: 0 + (640 x 1/8) = 0 + 80
Y coordinate: 480 - (480 x 1/8) = 480 - 60

For some controllers, you can adjust the default calibration points using the Set Parameter Block command. For more information, contact MicroTouch.

Guidelines for Calibrate Extended

Here are several guidelines for using the Calibrate Extended command:

- The controller uses the data immediately before liftoff to register a calibration touch. Therefore, users can touch off the target, move their finger to the target, hold for one second, and then lift off their finger. Instructing users to touch this way results in a more accurate calibration.

- If you are using both a pen and your finger as touch devices, you must calibrate the screen twice: once with your finger and once with the pen. The system stores both sets of calibration points.

- The controller stores the data in non-volatile memory (NOVRAM). Therefore, you do not have to calibrate the screen each time you power on the system. You should, however, recalibrate the touchscreen any time the video display changes or gets repositioned.

- You can cancel a calibration at any time by issuing a Reset command.

Calibrate Extended Procedure

▶ To use the CX command:

1. Enter the Calibrate Extended (CX) command.

   The controller sends an acknowledgment of <SOH>0<CR>.

2. Touch the screen at a lower left target, which is located 12.5% (1/8) in from the corner of the video image.

   The controller returns an acknowledgment of <SOH>1<CR>. This is a positive response. If you receive a negative response, try touching the screen again.

3. Touch the screen at an upper right target, which is located 12.5% (1/8) in from the corner of the video image.

   The controller returns an acknowledgment of <SOH>1<CR>. If you receive a negative response, try touching the screen again.

Touching the two valid calibration points results in a successful calibration. If either calibration point is invalid, the calibration fails. Table 6 lists how the controller responds if the Calibrate Extended failed.

Table 6. Response to Failed Calibration Extended

Controller	Response to Failed Calibration Extended
Serial/SMT2 Serial/SMT3V Serial/SMT3RV Serial/SMT2 Daughterboard Serial/SMT3V Daughterboard PC Bus SMT2 PC Bus SMT3V PC Bus SMT3RV	Returns to the factory default calibration.
Serial/SMT3 Serial/SMT3R TouchPen 4 TouchPen 4+ MousePort	Returns to the previous calibration values (that is, these controllers discard the touch coordinates and do not change the calibration points from their previous values).

Response:	<SOH>**1**<CR>	Positive response. Indicates that the controller received a valid touch coordinate (point) when the target was touched. Two valid touch points indicate a successful calibration.
	<SOH>**0**<CR>	Negative response. Indicates that the touch point is out of range of the expected target area. If you receive a negative response, try touching the target area again.
	<SOH>**2**<CR> or (No Response)	Indicates that the user did not touch the target long enough to provide an accurate calibration point. Only the following controllers return a 2 for this condition: Serial/SMT3, Serial/SMT3R, TouchPen 4, TouchPen 4+, and MousePort. All other controllers return no response.

Calibrate Interactive

Note: *MicroTouch provides this command for compatibility with older touchscreen controllers. To ensure compatibility with all existing and future MicroTouch controllers, do not use this command when developing your touch drivers and applications.*

Syntax: <SOH>**CI**<CR>

Description: Initiates an interactive, two-point edge calibration. The Calibrate Interactive command functions like the Calibrate New command except for the following conditions:

- Calibrate Interactive does not check if a touched point is within the limit. Calibrate Interactive performs no error checking.
- Calibrate Interactive swaps the points if the location of the lower left point and upper right point are reversed.

MicroTouch recommends that you use Calibrate Extended to calibrate the touchscreen.

Calibrate New

> **Note:** *MicroTouch provides this command for compatibility with older touchscreen controllers. To ensure compatibility with all existing and future MicroTouch controllers, do not use this command when developing your touch drivers and applications.*

Syntax: <SOH>**CN**<CR>

Description: Initiates an interactive, two-point *edge* calibration.

During the calibration process, you define the active area of the touchscreen by mapping locations to an absolute X, Y coordinate system. You touch two *target areas* on the screen. Touching the target areas sends the X, Y coordinates for those touch points to the controller. The controller calculates all other touch points based on these two points.

The Calibrate New command is similar to the Calibrate Extended command, except that the calibration points are located at the corners (edge) of the video image.

Guidelines for Calibrate New

Here are several guidelines for using the Calibrate New command:

- The controller uses the data immediately before liftoff to register a calibration touch. Therefore, users can touch off the target, move their finger to the target, hold for one second, and then lift off their finger. Instructing users to touch this way results in a more accurate calibration.

- The controller stores the data in non-volatile memory (NOVRAM). Therefore, you do not have to calibrate the screen each time you power on the system. You should, however, recalibrate the touchscreen any time the video display changes or gets repositioned.

- You can cancel a calibration at any time by issuing a Reset command.

Calibrate New Procedure

► To use the CN command:

1. Enter the Calibrate New (CN) command.

   The controller sends an acknowledgment of <SOH>0<CR>.

2. Touch the screen at a lower left target, which is located at the corner of the video image.

   The controller returns an acknowledgment. Refer to Table 7 for the acknowledgment values for the format you are using. If the acknowledgment is positive, continue to the next step. If the acknowledgment is negative, repeat this step.

3. Touch the screen at an upper right target, which is located at the corner of the video image.

   The controller returns an acknowledgment. Refer to Table 7 for the acknowledgment values for the format you are using. If the acknowledgment is positive, calibration is complete. If the acknowledgment is negative, repeat this step.

Touching the two valid calibration points results in a successful calibration. If either calibration point is invalid, the calibration fails. If the calibration fails, the controller returns to the factory default calibration (that is, the controller ignores the user-defined coordinates and sets the calibrated area to the factory default values).

Note: The Calibrate New command does not send an error code if it does not collect enough readings for a given point. Instead, the command simply discards the point and waits for you to touch the same point.

Response: A positive response indicates that the controller received a valid touch coordinate (point) when the target was touched. A negative response indicates that the touch point is out of range of the expected target area. If you receive a negative response, try touching the target area again.

The value of the positive response for the upper right touch point varies depending on whether you are using Format Binary, Format Tablet, Format Decimal, or Format Hexadecimal.

In general, the positive response is <SOH>1<CR> and the negative response is <SOH>0<CR>. There are, however, two exceptions to this rule. If you are using Format Decimal or Format Hexadecimal, the positive response is <SOH>0<CR> and the negative response is <SOH>1<CR>.

Table 7 lists the positive and negative responses returned for each calibration point (lower left and upper right) based on the data format being used.

Table 7. Response Values for the Calibrate New Command

Response	**Format Binary or Format Tablet**	**Format Hexadecimal or Format Decimal**
Lower Left Positive	<SOH>1<CR>	<SOH>1<CR>
Lower Left Negative	<SOH>0<CR>	<SOH>0<CR>
Upper Right Positive	<SOH>1<CR>	<SOH>0<CR>
Upper Right Negative	<SOH>0<CR>	<SOH>1<CR>

Calibrate Raw

Syntax: <SOH>**CR**<CR>

Description: Allows the collection of raw (signed) X and Y coordinates prior to the normal scaling, linearization, and filtering processes. The controller sends the coordinates whenever a touch is detected and continues to send a stream of data as long as a finger or pen remains in contact with the touchscreen.

The Calibrate Raw data is a 5-byte packet that includes 1 status byte and 4 bytes of binary X, Y coordinate data. Each X, Y coordinate includes 10 binary bits and 1 sign bit. The 10 bits represent coordinates within a range of -1024 to +1023.

To use the Calibrate Raw command, the controller and host system must be in an 8-bit data communication mode. The Calibrate Raw command returns a negative response if the controller is in 7-bit format. Also, the TouchPen controller must be in Pen Only mode or Finger Only mode before executing the Calibrate Raw command.

To end Calibrate Raw mode, issue a Reset command.

MicroTouch uses the Calibrate Raw command during manufacturing and testing, and recommends you use this command for diagnostics when you want raw data. Use the Calibrate Extended command for standard interactive, two-point calibration.

Response: <SOH>**0**<CR> Positive response.

After the controller is in Calibrate Raw mode, touching the screen causes the controller to return a response in the following format:

SXxYy

where:

S = Status byte; first byte of data. Refer to Table 8.

Xx = X (horizontal) coordinate data; second and third bytes of data.

Yy = Y (vertical) coordinate data; fourth and fifth bytes of data.

	MSB*			Bits				LSB*
Data Sequence	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
S - Byte 1	1	S6	S5	S4	S3	S2	S1	S0
X - Byte 2	0	X3	X2	X1	X0	Reserved		
x - Byte 3	0	Xs**	X9	X8	X7	X6	X5	X4
Y - Byte 4	0	Y3	Y2	Y1	Y0	Reserved		
y - Byte 5	0	Ys**	Y9	Y8	Y7	Y6	Y5	Y4

***** MSB = Most Significant Bit, LSB = Least Significant Bit
****** s = sign bit

Table 8 describes the meaning of the bits in the status byte (Byte 1).

Table 8. Calibrate Raw Status Bits

Bit	Description	Values
S0	Switch 1 status	For the TouchPen only. 1 = Switch is on (pressed). 0 = Switch is off.
S1	Switch 2 status	
S2 – S4	Reserved	—
S5	Pen or Finger	For the TouchPen only. 1 = Screen touched with a pen. 0 = Screen touched with a finger.
S6	Proximity (touch state)	1 = Touchscreen is being touched (a touchdown or a continued touch). 0 = Touchscreen is not being touched (a touch liftoff or inactive). When the proximity bit changes from 1 to 0 (touch liftoff), the controller outputs one final set of X, Y coordinate data with the bit equal to 0 and the X, Y coordinate data equal to the last touch point.
S7	Packet synchronization	Always 1.

Filter Number

> **Note:** *MicroTouch provides this command for compatibility with older touchscreen controllers. To ensure compatibility with all existing and future MicroTouch controllers, do not use this command when developing your touch drivers and applications.*

Syntax: <SOH>**FN***nn*<CR>

where:

nn = Two ASCII characters ranging from 10 to 99.

Description: Sets the number of X, Y values that the controller uses to generate an accurate coordinate after the touchscreen receives an initial touch. You want to be sure a user's finger makes full contact with the touchscreen before the controller sends any data coordinates to the host system.

For example, FN20 causes the controller to use the first 20 samples in determining a touch location.

Note: Very quick *touches* are always captured and sent to the host system regardless of the filter number. For example, if a quick touch generates only 5 samples, the controller still sends a coordinate to the host system.

You need to change the filter value when system noise or a high sensitivity setting causes erroneous input when the screen is first touched. Specifically, you may need to make adjustments in the following situations:

- For button-selection applications
  - The host system sometimes receives the wrong button selection.
  - The host system does not properly recognize a button if the button is touched on an outer edge.

- For cursor applications
  - The cursor first moves to a point near the finger and then moves to the true touchdown point.
  - The cursor follows the finger appropriately while drawing, but produces inaccurate cursor movement when the screen is initially touched.

Response: <SOH>**0**<CR> Positive response.

Finger Only

Syntax: <SOH>**FO**<CR>

Description: Sets the mode of operation of a TouchPen controller to accept only finger input. The TouchPen controller ignores pen input.

There are three modes available:

- Finger Only mode detects finger contact only and processes finger coordinate data.
- Pen Only mode detects pen contact only and processes pen coordinate data.
- Pen or Finger mode detects pen and finger contact, giving priority to pen contact when both are detected. Pen or Finger mode is the default mode for TouchPen controllers.

The pen mode changes back to the default setting at power-up, or if you issue a Restore Defaults command. You can use the Set Parameter Block command to change the default setting.

Choose the appropriate mode for your application. For example, applications requiring only signature input use Pen Only mode. Gaming applications typically use Finger Only mode, and point-of-sales applications that require signature verification may use Pen or Finger mode.

Additionally, changing the pen mode setting can optimize the performance of the touchscreen. In Pen or Finger mode, the TouchPen controller checks for input from both a pen and a finger. The controller always gives priority to the pen. If you are not currently using the pen for your touch application, use Finger Only mode for optimum performance.

Response: <SOH>**0**<CR> Positive response.

Format Binary

> **Note:** *MicroTouch provides this command for compatibility with older touchscreen controllers. To ensure compatibility with all existing and future MicroTouch controllers, do not use this command when developing your touch drivers and applications.*

Syntax: <SOH>**FB**<CR>

Description: Outputs the X, Y touch coordinate data as a 5-byte packet in a binary format. The packet includes the following 5 bytes:

- 1 header byte (status byte because Format Binary issues a Mode Status command)
- 2 bytes of binary X coordinate data
- 2 bytes of binary Y coordinate data

When called, Format Binary automatically issues a Mode Polled command and a Mode Status command.

In Mode Polled, the controller sends an X, Y coordinate only when requested by the host system and only when a user touches the screen. The host system must send an XON character (Ctrl Q or ^Q) before the controller can send the 5-byte data packet.

The controller sends the buffered data upon request from the host. Data is sent as a string of decimal ASCII characters, ranging from the ‘space’ character (20 hexadecimal) to the question mark (?) character (3F hexadecimal). The protocol establishes the X and Y coordinate output as 0 to 1023.

The Mode Status command forces the first byte (that is, the header byte) to become a status byte. The status byte defines whether the X, Y coordinates are generated from a touchdown, a touch continuation (when the finger is resting on the screen), or a touch liftoff.

Response: <SOH>**0**<CR> Positive response.

After the controller is in Format Binary mode, the host must issue a Ctrl Q (^Q) before the controller sends the touch coordinate.

For each *touchdown event*, Format Binary always sends the following hexadecimal packet followed by the 5-byte data packet:

0x17, 0x20, 0x20, 0x20, 0x20

The 5-byte data packet has the following format:

<*Status*>*Xx*,*Yy*

where:

<*Status*> = Defines how the X, Y coordinates are generated, where:

^Y (Hex 19) is a touchdown (first position of finger on the screen).

^\ (Hex 1C) is a continued touch (position of finger remains on the screen).

^R (Hex 18) is a touch liftoff (last position of finger on the screen).

Xx = X (horizontal) coordinate data. Total of 2 bytes.

Yy = Y (vertical) coordinate data. Total of 2 bytes.

	MSB*			**Bits**				**LSB***
Data Sequence	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Header **-** Byte 1	S7	S6	S5	S4	S3	S2	S1	S0
X **-** Byte 2	0	0	1	X9	X8	X7	X6	X5
x **-** Byte 3	0	0	1	X4	X3	X2	X1	X0
Y **-** Byte 4	0	0	1	Y9	Y8	Y7	Y6	Y5
y **-** Byte 5	0	0	1	Y4	Y3	Y2	Y1	Y0

***** MSB = Most Significant Bit, LSB = Least Significant Bit

Format Binary Stream

Note: *MicroTouch provides this command for compatibility with older touchscreen controllers. To ensure compatibility with all existing and future MicroTouch controllers, do not use this command when developing your touch drivers and applications.*

Syntax: <SOH>**FBS**<CR>

Description: Functions exactly like the Format Binary command, except the Format Binary Stream command initializes the operating mode to Mode Stream instead of Mode Polled.

In Mode Stream, the controller sends a continuous stream of X, Y coordinate data when you touch the screen. The controller continues to send data as long as you touch the screen. The controller sends the data even if the touch is stationary and unchanging.

Format Decimal

Note: *MicroTouch provides this command for compatibility with older touchscreen controllers. To ensure compatibility with all existing and future MicroTouch controllers, do not use this command when developing your touch drivers and applications.*

Syntax: <SOH>**FD**<CR>

Description: Outputs the X, Y touch coordinate data as a 9-byte packet in a decimal format (using the hexadecimal representation of decimal values). The packet includes the following 9 bytes:

- 1 header byte
- 3 bytes of X coordinate data
- An ASCII comma
- 3 bytes of Y coordinate data
- A terminator byte

Data is sent as a string of decimal ASCII characters (0 to 9). The output range for both the X and Y data is 000 to 999.

When activated, Format Decimal resets the Mode Status to report the standard <SOH> header. If the last format command was Format Binary, then Format Decimal sets the output mode to Mode Stream.

Format Decimal, which sends approximately 106 packets per second at 9600 baud, is not as efficient as Format Tablet. Format Decimal does not contain touchdown and liftoff information unless you also use the Mode Status command.

Format Hexadecimal is the same as Format Decimal except the controller returns the X, Y coordinates in hexadecimal instead of decimal.

Format Tablet, which sends approximately 192 packets per second at 9600 baud, is the most efficient packet. It also contains touchdown and liftoff information. Format Tablet is the standard for MicroTouch product development.

Response: <SOH>**0**<CR>

After the controller is in Format Decimal mode, touching the screen causes the controller to return a response in the following format:

<HDR>*Xxx***,***Yyy*<CR>

where:

<HDR> = Start-of-header (Hex 01). If you send a Mode Status command after a Format Decimal command, this first byte becomes a status byte. The status byte defines whether the X, Y coordinates are generated from a touchdown, a touch continuation (when the finger is resting on the screen), or a touch liftoff. For more details, refer to the Mode Status command later in this chapter.

Xxx = X (horizontal) coordinate data. Total of 3 bytes.

, = ASCII comma that separates the X and Y coordinate data.

Yyy = Y (vertical) coordinate data. Total of 3 bytes.

<CR> = Terminator (Hex 0D).

Format Hexadecimal

> **Note:** *MicroTouch provides this command for compatibility with older touchscreen controllers. To ensure compatibility with all existing and future MicroTouch controllers, do not use this command when developing your touch drivers and applications.*

Syntax: <SOH>**FH**<CR>

Description: Outputs the X, Y touch coordinate data as a 9-byte packet in a hexadecimal format. The packet includes the following 9 bytes:

- 1 header byte
- 3 bytes of X coordinate data
- An ASCII comma
- 3 bytes of Y coordinate data
- A terminator byte

Data is sent as a string of hexadecimal ASCII characters (0 to 9, A to F). The output range for both the X and Y data is 000 to 3FF.

When activated, Format Hexadecimal resets the Mode Status to report the standard <SOH> header. If the last format command was Format Binary, then Format Hexadecimal sets the output mode to Mode Stream.

Format Hexadecimal, which sends approximately 106 packets per second at 9600 baud, is not as efficient as Format Tablet. Format Hexadecimal does not contain touchdown and liftoff information unless you also use the Mode Status command.

Format Decimal is the same as Format Hexadecimal except the controller returns the X, Y coordinates in decimal instead of hexadecimal.

Format Tablet, which sends approximately 192 packets per second at 9600 baud, is the most efficient packet. It also contains touchdown and liftoff information. Format Tablet is the standard for MicroTouch product development.

Response: <SOH>**0**<CR>

After the controller is in Format Hexadecimal mode, touching the screen causes the controller to return a response in following format:

<HDR>*Xxx***,***Yyy*<CR>

where:

<HDR> = Start-of-header (Hex 01). If you send a Mode Status command after a Format Hexadecimal command, this first byte becomes a status byte. The status byte defines whether the X, Y coordinates are generated from a touchdown, a touch continuation (when the finger is resting on the screen), or a touch liftoff. For more details, refer to the Mode Status command later in this chapter.

Xxx = X (horizontal) coordinate data. Total of 3 bytes.

, = ASCII comma that separates the X and Y coordinate data.

Yyy = Y (vertical) coordinate data. Total of 3 bytes.

<CR> = Terminator (Hex 0D).

Format Raw

Syntax: <SOH>**FR**<CR>

Description: Returns the signal level (amount of touch) of each of the four touchscreen corners in digital format. The returned values are not corrected for offset and stray values. However, you can obtain the offset and stray values using the Get Parameter Block command. For more information, refer to the description of the Get Parameter Block command later in this chapter.

The Format Raw data is a 7-byte packet that includes 1 status byte and 6 bytes of binary corner data. The data format for the packet is fixed in order to provide the most efficient transfer of data. The first byte of each packet always has its high bit (Bit 7) set to provide synchronization with the host system. Each corner value is 10 bits, which are delivered in 2 bytes, and has a range of 0 to 1023.

To use the Format Raw command, the controller and host system must be in an 8-bit data communication mode. The Format Raw command returns a negative response if the controller is in 7-bit format. Also, TouchPen controllers must be in Pen Only mode or Finger Only mode before executing the Format Raw command.

To terminate Format Raw, issue a Reset command. The controller may return several bytes of data between the time you issue a Reset command and the controller receives it. You can either scan the data stream for the Reset acknowledgment, or you can ignore the response to the first Reset command and then issue a second Reset after approximately 10 seconds has passed.

Use the Format Raw command for diagnostics. Use Format Tablet for standard touchscreen operation.

Response: <SOH>**0**<CR> Positive response.

After the controller is in Format Raw mode, the controller returns a response in the following format:

<7-byte-packet><7-byte-packet>...<7-byte-packet>...

Byte	**Bits 0 – 7**	
1	b0 – b3:	Drive level (amount of signal sent from controller)
	b4 – b6:	Reserved
	b7:	Synchronization bit (Always 1)
2	b0 – b2:	3 most significant bits of upper left (UL) corner
	b3:	Always 0
	b4 – b6:	3 most significant bits of lower left (LL) corner
	b7:	Always 0
3	b0 – b2:	3 most significant bits of lower right (LR) corner
	b3:	Always 0
	b4 – b6:	3 most significant bits of upper right (UR) corner
	b7:	Always 0
4	b0 – b6:	7 least significant bits of lower left (LL) corner
	b7:	Always 0
5	b0 – b6:	7 least significant bits of upper left (UL) corner
	b7:	Always 0
6	b0 – b6:	7 least significant bits of upper right (UR) corner
	b7:	Always 0
7	b0 – b6:	7 least significant bits of lower right (LR) corner
	b7:	Always 0

Format Tablet

Syntax: <SOH>**FT**<CR>

Description: Outputs the X, Y touch coordinate data in a 5-byte packet. The packet includes 1 status byte and 4 bytes of binary X, Y coordinate data. The protocol also establishes the X and Y coordinate output as 14 binary bits providing a range of 0 to 16,383.

The low order bits (X3 – X0 and Y3 – Y0) are not significant in a 1024 by 1024 touchscreen because data can fluctuate with each touch, and therefore may not be completely accurate.

To use Format Tablet, the controller and host system must be in an 8-bit data communication mode. The Format Tablet command returns a negative response if the controller is in 7-bit format.

Format Tablet is the most efficient packet (sends approximately 192 packets per second at 9600 baud). It also contains touchdown and liftoff information. Format Tablet is the standard for MicroTouch product development.

For comparison, Format Hexadecimal and Format Decimal (which send approximately 106 packets per second at 9600 baud) are not as efficient as Format Tablet. These data formats do not contain touchdown and liftoff information unless you also use the Mode Status command.

Response: <SOH>**0**<CR> Positive response.

After the controller is in Format Tablet mode, touching the screen causes the controller to return a response in the following format:

SXxYy

S = Status byte; first byte of data. Refer to Table 9.

Xx = X (horizontal) coordinate data; second and third bytes of data.

Yy = Y (vertical) coordinate data; fourth and fifth bytes of data.

	MSB*			**Bits**				**LSB***
Data Sequence	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
S - Byte 1	1	S6	S5	S4	S3	S2	S1	S0
X - Byte 2	0	X6	X5	X4	X3	X2	X1	X0
x - Byte 3	0	X13	X12	X11	X10	X9	X8	X7
Y - Byte 4	0	Y6	Y5	Y4	Y3	Y2	Y1	Y0
y - Byte 5	0	Y13	Y12	Y11	Y10	Y9	Y8	Y7

***** MSB = Most Significant Bit, LSB = Least Significant Bit

Table 9 defines the status bits (Byte 1) for the Format Tablet data.

Table 9. Format Tablet Status Bits

Bit	**Description**	**Values**
S0	Switch 1 status	For the TouchPen only. 1 = Switch is on (pressed). 0 = Switch is off.
S1	Switch 2 status	
S2 – S4	Reserved	—
S5	Pen or Finger	For the TouchPen only. 1 = Screen touched with a pen. 0 = Screen touched with a finger.
S6	Proximity (touch state)	1 = Touchscreen is being touched (a touchdown or a continued touch). 0 = Touchscreen is not being touched (a touch liftoff or inactive). When the proximity bit changes from 1 to 0 (touch liftoff), the controller outputs one final set of X, Y coordinate data with the bit equal to 0 and the X, Y coordinate data equal to the last touch point.
S7	Packet synchronization	Always 1.

Format Zone

Note: *MicroTouch provides this command for compatibility with older touchscreen controllers. To ensure compatibility with all existing and future MicroTouch controllers, do not use this command when developing your touch drivers and applications.*

Syntax: <SOH>**FZ**<CR>

Description: Outputs the X, Y touch coordinate data as a 5-byte packet and indicates whether the touch occurred inside or outside the calibration area.

When called, Format Zone automatically issues a Mode Stream command and then sends the 5-byte packet. Data is sent as a string of decimal ASCII characters (" " to "?"). The protocol establishes the X and Y coordinate output as 0 to 1023.

Response: <SOH>**0**<CR> Positive response.

After the controller is in Format Zone mode, touching the screen causes the controller to return a response in the following format:

<Zone>Xx,Yy

where:

<Zone> = Indicates whether the touch point is within the calibration range (zone). The format is as follows:

Event	Within Zone	Outside Zone
Touchdown	ASCII D	ASCII L
Continued Touch	ASCII B	ASCII J
Touch Liftoff	ASCII A	ASCII I

Xx = X (horizontal) coordinate data. Total of 2 bytes.

Yy = Y (vertical) coordinate data. Total of 2 bytes.

	MSB*			**Bits**				**LSB***
Data Sequence	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Zone - Byte 1	S7	S6	S5	S4	S3	S2	S1	S0
X - Byte 2	0	0	1	X9	X8	X7	X6	X5
x - Byte 3	0	0	1	X4	X3	X2	X1	X0
Y - Byte 4	0	0	1	Y9	Y8	Y7	Y6	Y5
y - Byte 5	0	0	1	Y4	Y3	Y2	Y1	Y0

***** MSB = Most Significant Bit, LSB = Least Significant Bit

If you send a Format Zone command and then send a Mode Status command, the Format Zone command no longer reports whether the the touch point is within the calibration range (zone). Instead, Format Zone sends a touch status byte preceding the X, Y coordinate data. For more information on the status byte, refer to the Mode Status command later in this chapter.

Frequency Adjust

Note: *MicroTouch provides this command for compatibility with older touchscreen controllers. To ensure compatibility with all existing and future MicroTouch controllers, do not use this command when developing your touch drivers and applications.*

Syntax: <SOH><Ctrl C>**F***nn*<CR>

where:

nn = Two ASCII characters that define the frequency setting. Refer to Table 10 for values.

Description: Lets you change the operating frequency of the sensor drive signal (DRVOUT) that is fed from the controller to the sensor and drives the sensor output. You may change the signal frequency to help eliminate erratic or jittery cursor movement due to electrical interference.

The frequency of the sensor drive signal should not be the same as any frequency, or second or third harmonics of any frequency, in close proximity to the sensor. (The CRT scanning signals may cause interference to the controller if their frequency or harmonics are close to those of the DRVOUT signal.)

Although the syntax for the Change Frequency command is the same for the SMT2 and SMT3 series of controllers, the frequency selected by the command is different for each type of controller. The factory default setting is also different for each type of controller. Table 10 lists the available frequency settings (in Hz).

Caution: If your touch application uses the Change Frequency command with a specified value, be sure to check the frequency if you replace a SMT2 controller with a SMT3 controller.

Table 10. Frequency Settings (in Hz)

nn	Serial/SMT2 Frequency	Serial/SMT3 Frequency
15	25000.00 (default)	33500.00
16	22727.27	34600.00
17	22727.27	35400.00
18	20833.33	35900.00
19	19230.77	36900.00
20	19230.77	38400.00
21	17857.14	38700.00 (default)
22	16666.67	39500.00
23	16666.67	41900.00
24	15625.00	42500.00
25	14705.88	No change
26	14705.88	No change
27	13888.89	No change
28	13157.89	No change
29	13157.89	No change
31	11904.76	No change
32	11904.76	No change
33	11363.64	No change
34	10869.57	No change
35	10869.57	No change
36	10416.67	No change
37	10000.00	No change
39	9615.38	No change
40	9259.26	No change
41	9259.26	No change
42	8928.57	No change
43	8620.69	No change
44	8620.69	No change
45	8333.33	No change

Response: <SOH>**0**<CR> Positive response.

Get Parameter Block

Description: Allows access to all power-up and run time parameters used by the controller.

The Get Parameter Block (GP) command works in conjunction with the Set Parameter Block (SP) command. You use this pair of commands for configuration and diagnostic purposes.

You use the Get Parameter Block command to retrieve the parameters. You then use the Set Parameter Block command to modify the data and write the data back to the controller. The blocks include calibration and initialization data, linearization data, and run time variables.

Command syntax and controller block details are not provided in this manual because the block data is subject to change with each firmware release and because block changes using the SP command may cause system problems. For information about command syntax and block descriptions, contact MicroTouch.

Mode Down/Up

> **Note:** *MicroTouch provides this command for compatibility with older touchscreen controllers. To ensure compatibility with all existing and future MicroTouch controllers, do not use this command when developing your touch drivers and applications.*

Syntax: <SOH>**MDU**<CR>

Description: Send one X, Y coordinate for each touchdown and each liftoff on the touchscreen.

In Mode Down/Up, the controller sends the first X, Y coordinate when the screen is touched, sends no data while the touch is held, and sends a second X, Y coordinate at touch liftoff. The format of the coordinate data depends on the last format command received by the controller.

Response: <SOH>**0**<CR> Positive response.

Mode Inactive

> **Note:** *MicroTouch provides this command for compatibility with older touchscreen controllers. To ensure compatibility with all existing and future MicroTouch controllers, do not use this command when developing your touch drivers and applications.*

Syntax: <SOH>**MI**<CR>

Description: Sends no X, Y coordinates when the screen is touched.

The Mode Inactive command shuts down all controller data transmission and causes the controller to stop reporting all touch points.

Response: <SOH>**0**<CR> Positive response.

Mode Point

Note: *MicroTouch provides this command for compatibility with older touchscreen controllers. To ensure compatibility with all existing and future MicroTouch controllers, do not use this command when developing your touch drivers and applications.*

Syntax: <SOH>**MP**<CR>

Description: Sends a single X, Y coordinate for each touchdown.

In Mode Point, the controller sends one X, Y coordinate when the screen is touched. The controller sends no further coordinates while the touch is held or at touch liftoff. The format of the coordinate data depends on the last format command received by the controller.

Response: <SOH>**0**<CR> Positive response.

Mode Polled

Note: *MicroTouch provides this command for compatibility with older touchscreen controllers. To ensure compatibility with all existing and future MicroTouch controllers, do not use this command when developing your touch drivers and applications.*

Syntax: <SOH>**MQ**<CR>

Description: Sends an X, Y coordinate only when requested by the host system and only when a user touches the screen. The format of the coordinate data depends on the last format command received by the controller.

To request touch coordinate data, the host system sends an XON character (Ctrl Q or ^Q) to the controller. The controller returns one X, Y coordinate to the host system only if a user touched the screen since the last ^Q.

Mode Polled causes the controller to report a touch coordinate as follows:

- If the controller receives a ^Q before a touchdown is detected, the controller sends a single packet immediately upon detecting a touch. The controller waits for the next ^Q to send the touchdown or liftoff coordinates depending on whether the finger is touching at the time the next ^Q is received. The controller reports one packet at a time for each ^Q received while the finger is touching the screen.

- If the controller detects a touchdown before a ^Q is received, the controller remembers the touchdown coordinate and waits for a ^Q before sending out the touchdown packet.

- If the controller detects both a touchdown and a liftoff before a ^Q is received, the controller remembers both points and waits for a ^Q. Upon receiving a ^Q, the controller reports the touchdown packet and waits for the next ^Q before sending the liftoff packet.

- If the host system sends two or more ^Q signals before a touchdown occurs, the controller does not send multiple packets when a touchdown is detected.

- The controller stores only one set of touchdown and liftoff points. If the screen is touched more than once before the controller receives a ^Q, the controller reports the most recent touchdown point only. If a touchdown is reported, it will report the last liftoff point regardless of how many times liftoff has happened before the second ^Q is received.

Note that the controller does not send any data in response to the ^Q request if a user does not touch the screen. The controller holds the ^Q and informs the host system the next time a user touches the screen. Therefore, the controller will send the X, Y coordinate data to the host at any time after a ^Q if there was no touch at the time the host sent the ^Q.

Response:	<SOH>**0**<CR>	Positive response.

Mode Status

> **Note:** *MicroTouch provides this command for compatibility with older touchscreen controllers. To ensure compatibility with all existing and future MicroTouch controllers, do not use this command when developing your touch drivers and applications.*

Syntax: <SOH>**MT**<CR>

Description: Sends a touch status byte preceding the X, Y coordinate data sent in response to a Format Decimal or Format Hexadecimal command. The format of the data depends on the last format command received by the controller.

By default, Format Decimal and Format Hexadecimal send a 9-byte data packet. The first byte is the header byte. If you use the Mode Status command, the first byte becomes a status byte. This status byte defines whether the X, Y coordinates are generated from a touchdown, a touch continuation (when the finger is resting on the screen), or a touch liftoff.

MicroTouch recommends that you use the Mode Status command in conjunction with Format Decimal and Format Hexadecimal so that the touch data includes status information in the packet header byte. By default, Format Decimal and Format Hexadecimal do not contain the status information in the header byte.

Note: You should always send the format command (Decimal and Hexadecimal) first, and then send the Mode Status command.

You do not need to send a Mode Status command if you are using Format Tablet, Format Binary, or Format Binary Stream.

- The Format Tablet protocol automatically includes status information in the first byte of data.
- Format Binary and Format Binary Stream automatically issue a Mode Status command.

Response: <SOH>**0**<CR> Positive response.

If the controller is in Format Decimal or Format Hexadecimal mode and you receive a positive response to Mode Status, touching the screen causes the controller to return the following response:

<*Status*>*Xxx*,*Yyy*<CR>

where:

<*Status*> = Defines how the X, Y coordinates are generated, where:

^Y (Hex 19) is a touchdown (first position of finger on the screen).

^\ (Hex 1C) is a continued touch (position of finger remains on the screen).

^R (Hex 18) is a touch liftoff (last position of finger on the screen).

Xxx = X (horizontal) coordinate data. Total of 3 bytes.

, = ASCII comma that separates the X and Y coordinate data.

Yyy = Y (vertical) coordinate data. Total of 3 bytes.

<CR> = Terminator (Hex 0D).

Mode Stream

Syntax: <SOH>**MS**<CR>

Description: Sends a continuous stream of X, Y coordinate data when you touch the screen. The controller continues to send data as long as you touch the screen. The controller sends the data even if the touch is stationary and unchanging.

The format of the coordinate data depends on the last format command received by the controller.

If you are using a TouchPen controller, the controller must be in the appropriate mode (Pen or Finger mode, Pen Only mode, or Finger Only mode) for the pen or finger to be detected.

Note: Format Raw automatically uses Mode Stream to send X, Y coordinate data.

Response: <SOH>**0**<CR> Positive response.

Null Command

Syntax: <SOH>**Z**<CR>

Description: Queries the controller and waits for a response.

Use Z to determine that you are communicating with the controller or to make sure that a utility is communicating with the controller. Using this command does not affect the controller’s current operating parameters.

Response: <SOH>**0**<CR> Positive response.

Output Identity

Syntax: <SOH>**OI**<CR>

Description: Returns a 6-character identifier, which describes the controller type and the firmware version number.

Response: <SOH>*CcXxxx*<CR>

where:

Cc = Two ASCII characters that describe the type of controller.

A3	=	Serial/SMT2 Serial/SMT2 Daughterboard Serial/SMT3V Serial/SMT3V Daughterboard Serial/SMT3RV PC Bus SMT3V PC Bus SMT3RV
A4	=	PC Bus SMT2
P5	=	TouchPen 4 TouchPen 4+
Q1	=	Serial/SMT3 Serial/SMT3R MousePort

Xxxx = Four ASCII characters that indicate the firmware version number in decimal format. The first two characters represent the version number; the last two characters represent the revision level. For example, 0100 means Version 1, Revision 0 (that is 1.0) or 0510 means Version 5, Revision 1 (5.1).

Output Status

Note: *MicroTouch provides this command for compatibility with older touchscreen controllers. To ensure compatibility with all existing and future MicroTouch controllers, do not use this command when developing your touch drivers and applications.*

Syntax: <SOH>**OS**<CR>

Description: Causes the controller to report the status of its hardware.

On power-up, the controller automatically sends the output status information to the host. Therefore, the host can detect a power-up condition. Note that the controller sends the status information only if autobaud is disabled. If autobaud is enabled, the controller waits for the host to send a command.

Response: <SOH>*ab*<CR>

where *ab* is a 2-character bit-mapped status response with the following bit breakdown:

a0 = RAM error.

a1 = ROM error.

a2 = Analog-to-digital error.

a3 = NOVRAM error

a4 = ASIC error.

a5 = Power on flag, if SMT2[2]. Set to 1 at power on and reset after the Output Status command.

Reserved, always 0 if SMT3V[2].

a6 = Reserved, always 1.

a7 = Reserved, always 0.

b0 = Cable NOVRAM error[1].

b1 = Hard NOVRAM error.

b2 = Reserved.

b3 = Reserved.

b4 = Reserved.

b5 = Software reset flag. Set to 0 after power on; set to 1 after Reset command.

b6 = Reserved, always 1.

b7 = Reserved, always 0.

[1] The cable NOVRAM error will flash the LED until the controller receives the first valid command from the host.

[2] SMT2 includes the Serial/SMT2, Serial/SMT2 Daughterboard, and PC Bus SMT2; SMT3V includes the Serial/SMT3V, Serial/SMT3RV, Serial/SMT3V Daughterboard, PC Bus SMT3V, and PC Bus SMT3RV controllers.

Parameter Lock

> **Note:** *MicroTouch provides this command for compatibility with older touchscreen controllers. To ensure compatibility with all existing and future MicroTouch controllers, do not use this command when developing your touch drivers and applications.*

Syntax: <SOH>**PL**<CR>

Description: Writes and stores the data format and operating mode of the controller into non-volatile memory (NOVRAM).

Any time you make changes to the data format or the operating mode, you should issue a Parameter Lock command to store the new settings to the NOVRAM. Therefore, the settings are not lost when the unit is powered down.

Response: <SOH>**0**<CR> Positive response.

Parameter Set

Note: *MicroTouch provides this command for compatibility with older touchscreen controllers. To ensure compatibility with all existing and future MicroTouch controllers, do not use this command when developing your touch drivers and applications.*

Syntax: <SOH>**P***pds*[*b*]<CR>

where:

p = Parity type.

N = No parity
O = Odd parity
E = Even parity

d = Number of data bits (**7** or **8**).

s = Number of stop bits (**1** or **2**).

b = Communication rate.

1	= 19200 baud	**4**	= 2400 baud
2	= 9600 baud	**5**	= 1200 baud
3	= 4800 baud		

Description: Lets you adjust the communication parameters (parity, data bits, and stop bits) of the controller. Optionally, you can change the communication rate by appending an additional character to the command string. Upon execution of the Parameter Set command, the controller automatically stores the new settings, the current operating mode, and the current data format in NOVRAM.

The communication parameters of the host system must match the present settings of the controller when the command is given for it to be accepted and the changes implemented.

The process of changing the parameters takes three steps:

- The host system must first communicate with the controller using a matched set of parameters.

- The Parameter Set command is issued with the new parameters to the controller. The new settings take effect immediately.
- The host system must be changed to the new parameters in order to communicate with the controller again.

Examples:	<Ctrl A>PN813<CR>	Sets the serial line to no parity, eight data bits, one stop bit, and 4800 baud.
	<Ctrl A>PN81<CR>	Sets the parity, data bits, and stop bits; leaves the baud at its previous value.

Caution: The settings are immediately written to NOVRAM, and all future communication must occur at the new values. It is possible to set the parameters to values that prevent future communication with the controller.

For example, using PROCOMM, you enter <Ctrl A>PN815<CR> to set the communication rate to 1200. However, PROCOMM does not support 1200 baud. The controller will now pass data at 1200 baud, but the host system will not be able to read the data. The controller will expect to receive all commands at 1200 baud and will not recognize any attempts to change the communication rate. The controller is effectively locked up. If AutoBaud is enabled, you can correct this situation by a power-down/power-up sequence. If AutoBaud is disabled, you need to use Microcal or another terminal application to set the communication rate at the new values.

The communication rates that can be set with the AutoBaud command are the same as the rates you can set with the Parameter Set command. Therefore, the AutoBaud command no longer finds 7200, 3600, 2000, 1800, 600, 300, 200, 150, 135, and 110 baud. Also, some MicroTouch controllers do not support the AutoBaud command.

Response:	<SOH>**0**<CR>	Positive response.

Pen Only

Syntax: <SOH>**PO**<CR>

Description: Sets the operational mode of a TouchPen controller to accept only pen input. The controller ignores finger data.

There are three pen modes available:

- Pen Only mode detects pen contact only and processes pen coordinate data.
- Pen or Finger mode detects pen and finger contact, giving priority to pen contact when both are detected. Pen or Finger mode is the default mode for TouchPen controllers.
- Finger Only mode detects finger contact only and processes finger coordinate data.

The pen mode changes back to the default setting at power-up, or if you issue a Restore Defaults command. You can use the Set Parameter Block command to change the default setting.

Choose the appropriate mode for your application. For example, applications requiring only signature input use Pen Only mode. Gaming applications usually use Finger Only mode, and point-of-sales applications that require signature verification may use Pen or Finger mode.

Additionally, changing the pen mode setting can optimize the performance of the touchscreen. In Pen or Finger mode, the TouchPen controller checks for input from either a pen or a finger. The controller always gives priority to the pen. If you are not currently using the pen for your touch application, use Finger Only mode for optimum performance.

Response: <SOH>**0**<CR> Positive response.

Pen or Finger

Syntax: <SOH>**PF**<CR>

Description: Sets the operational mode of a TouchPen controller to accept both pen and finger input. Pen or Finger mode is also called *automatic* mode.

In Pen or Finger mode, the controller gives higher priority to the pen.

- If the controller detects both pen and finger touches at the same time, pen contact has higher priority. The controller acknowledges only the pen touches and sends pen coordinate data to the host system.
- If the controller detects only finger contact, it sends the finger coordinate data to the host system.
- If you are using your finger and the pen touches the screen, the pen overrides the finger input. In this case, the controller automatically sends a finger liftoff coordinate when the pen touches the screen.
- If you are using the pen and you lift the pen from the screen, the system does not recognize finger (or hand) touch until after a specified time delay.
- This delay prevents accidental screen touches from being interpreted as input. For example, if you rest your hand on the screen while using the pen, you can lift the pen up and put it back again without your hand touch being acknowledged. You can use the Set Parameter Block command to change the amount of time the system waits before acknowledging finger contact.
- If a finger or hand is on the screen when the pen lifts off, the system ignores the finger or hand until you lift your finger (or hand) off the screen and touch the screen again.

There are three pen modes available:

- Pen or Finger mode detects pen and finger contact, giving priority to pen contact when both are detected. Pen or Finger mode is the default mode for TouchPen controllers.
- Finger Only mode detects finger contact only and processes finger coordinate data.
- Pen Only mode detects pen contact only and processes pen coordinate data.

The pen mode changes back to the default setting at power-up, or if you issue a Restore Defaults command. You can use the Set Parameter Block command to change the default setting.

Choose the appropriate mode for your application. For example, applications requiring only signature input use Pen Only mode. Gaming applications usually use Finger Only mode, and point-of-sales applications that require signature verification may use Pen or Finger mode.

Additionally, changing the pen mode setting can optimize the performance of the touchscreen. In Pen or Finger mode, the TouchPen controller checks for input from either a pen or a finger. The controller always gives priority to the pen. If you are not currently using the pen for your touch application, use Finger Only mode for optimum performance.

Response: <SOH>**0**<CR> Positive response.

Reset

Syntax: <SOH>**R**<CR>

Description: Initializes the hardware and the firmware, causes the controller to stop sending data, and recalculates the environmental conditions (for example, stray and offset values). The Reset command also cancels the Format Raw and Calibrate Raw commands and returns the controller to normal operation.

MicroTouch recommends that the host system issue a Reset command whenever the host system is powered on and is attempting to establish communication with the controller.

Depending on the controller, the amount of time needed to execute a Reset command ranges from 225 milliseconds to 800 milliseconds. Therefore, the application program should wait and be sure it receives the command response before issuing another command to the controller following the reset.

Response: <SOH>**0**<CR> Positive response.

Restore Defaults

Syntax: <SOH>**RD**<CR>

Description: Returns to the factory default operating parameters. The Restore Defaults command copies the MicroTouch factory default parameters from ROM to the non-volatile memory (NOVRAM) and then executes a Reset command.

Table 11 lists the factory defaults for each touchscreen controller. The Restore Defaults command is useful in situations where inadvertent commands to the controller have rendered the touchscreen inoperative.

Table 11. Factory Defaults

Parameter	**SMT2** **PC Bus SMT2** **PC Bus SMT3V**	**SMT3V** **SMT3RV** **PC Bus SMT3RV**	**SMT3** **SMT3R**	**MousePort**	**TouchPen 4** **TouchPen 4+**
Baud Rate	9600	9600	9600	N/A	9600
Serial Settings	N, 7, 2	N, 7, 2	N, 8, 1	N/A	N, 8, 1
AutoBaud	Enabled	Disabled	N/A	N/A	N/A
Data Format	Format Decimal	Format Decimal	Format Tablet	Format Tablet	Format Tablet
Operating Mode	Mode Stream	Mode Stream	Mode Stream	Mode Stream	Mode Stream
Pen Mode	N/A	N/A	N/A	N/A	Pen or Finger mode
Return to Factory Calibration	Yes	Yes	Yes	Yes	Yes

Note the serial communication settings restored by this command. Application programs operating at other communication settings that issue this command must change to the default settings to receive the command response and re-establish communication with the controller.

Note: After you issue a Restore Defaults command, calibrate your touchscreen using the Calibrate Extended command.

The Restore Defaults command requires approximately 75 to 100 milliseconds, plus the execution time of the Reset command (225 to 800 milliseconds). Therefore, the application program should wait and be sure it receives the command response before issuing another command to the controller.

Response:	<SOH>**0**<CR>	Positive response.

Sensitivity Set

> **Note:** *MicroTouch provides this command for compatibility with older touchscreen controllers. To ensure compatibility with all existing and future MicroTouch controllers, do not use this command when developing your touch drivers and applications.*

Syntax: <SOH>**SE***n*<CR>

where:

n = Sensitivity level.

0 = Normal touch (default)
1 = More sensitive (level 1)
2 = Very sensitive (level 2)
3 = Most sensitive (level 3)

Description: Changes the sensitivity of the touchscreen. Use the sensitivity setting to adjust the touch for differences in systems and touchscreen implementations. You can adjust the sensitivity level for your personal preference.

Upon execution of this command, the sensitivity setting (touchdown threshold and liftoff threshold), the current operating modes, and the current data format are stored in the controller's non-volatile memory (NOVRAM).

Response: <SOH>**0**<CR> Positive response.

Set Parameter Block

Description: Sets power-up and run time parameters used by the controller.

The Set Parameter Block (SP) command works in conjunction with the Get Parameter Block (SP) command. You use this pair of commands for configuration and diagnostic purposes.

You use the Get Parameter Block command to retrieve the parameters. You then use the Set Parameter Block command to modify the data and write the data back to the controller. The blocks include calibration and initialization data, linearization data, and run time variables.

Note that a Set Parameter Block 1 command always performs the following operations:

- Receives block 1 data
- Writes data to the NOVRAM
- Resets the controller

Command syntax and controller block details are not provided in this manual because the block data is subject to change with each firmware release and because block changes using the SP command may cause system problems. For information about command syntax and block descriptions, contact MicroTouch.

Unit Type

Syntax: <SOH>**UT**<CR>

Description: Responds with an 8-character identity string. This string identifies the type of controller currently attached to the system, lists the features supported by the controller, and outputs the status of the controller hardware (a self-test code).

Response: Returns an identification code up to 8 ASCII characters in the following format:

<SOH>*TtFfffSs*<CR>

where:

Tt = Two ASCII characters that identify the controller type.

TP = TouchPen series of controllers
QM = Serial/SMT3 series of controllers

Ffff = Four ASCII characters that indicate the features supported by the controller.

R = Indicates a resistive controller
V = Indicates the Serial/SMT3V series of controllers
******** = Indicates no additional features configured

Ss = Two ASCII characters that provide status information about the controller hardware. The two characters represent one byte. Each character is in the range 0 to 9 and A to F.

Table 12 defines the meaning of each bit in the status byte. Each bit can be set to 1 or 0, where:

1 = Error
0 = No error
00 = No diagnostic errors (normal response)

Note: If your controller does not support the Unit Type command, you can use the Unit Type Verify command or the Output Identity and Output Status commands to obtain information about the controller type, the firmware revision, and the hardware status.

Table 12. Bit Definition for the Unit Type Command

Bit	Serial/SMT3 Status	TouchPen Status
0	Reserved.	RAM error. Hardware malfunction.
1	ROM error. Firmware checksum verification error.	Same.
2	PWM error. Unable to establish PWM operating range at power-up. Nonrecoverable error.	Analog-to-digital (A/D) error. The A/D converter malfunctioned.
3	NOVRAM error. The operating parameters in the controller NOVRAM are invalid. Using defaults.	Same.
4	HDW error. The controller hardware failed (unable to initialize or configure gate array). Nonrecoverable error.	ASIC error. The Application Specific Integrated Circuit (ASIC) failed.
5	Reserved.	Reset flag. 1 = A Unit Type command has not been issued since the last reset. 0 = A Unit Type command has been issued since the last reset.
6	Cable NOVRAM error[1]. The linearization data in the cable NOVRAM is invalid.	Reserved.
7	NOVRAM2 error. The linearization data in the controller NOVRAM is invalid.	Same.

[1] The cable NOVRAM error will flash the LED until the controller receives the first valid command from the host.

Unit Type Verify

Syntax: <SOH>**UV**<CR>

Description: Responds with an 8-character identity string. This string identifies the type of controller currently attached to the system, lists the features supported by the controller, and outputs the status of the controller hardware (a self-test code).

Refer to the Unit Type command for a description of the 8-character identity string. The Unit Type command and the Unit Type Verify command return the exact same information.

CHAPTER 3

Controller LED Diagnostics

MicroTouch controllers are highly reliable units; however, there may be occasions when the controller does not perform exactly as you expected. Serial/SMT controllers provide visual feedback via an LED indicator on the controller.

When you power-up the unit, the LED is bright until the controller start-up sequence completes. Following start-up, the LED becomes dim and remains dim as long as you do not touch the touchscreen. When you touch the screen, the LED becomes bright.

A flashing (or blinking) LED indicates the controller’s power-on self-test failed. Refer to Table 13 and Table 14 for a description of each error code.

Some errors are nonrecoverable, meaning that normal touchscreen operation cannot occur. Other errors assume default conditions, and touchscreen operation can proceed. For example, an incorrect NOVRAM checksum is a recoverable error. In this case, operation continues using factory default conditions (as if a Restore Defaults command was issued).

Serial/SMT2 LED Codes

Table 13 decribes the meaning of a blinking status light (LED) for all controllers that use Serial/SMT2 hardware. The SMT2 series includes the following controllers:

- Serial/SMT2 controller
- Serial/SMT2 Daughterboard
- PC Bus SMT2 controller

Table 13. LED Diagnostic Codes for SMT2 Series of Controllers

LED Flashes (per 10 seconds)	**Error Description**
1	RAM error. Hardware malfunction.
2	ROM error. Firmware checksum verification error.
3	Analog-to-digital error. Unable to establish A/D operating range at power-up.
4	NOVRAM error. The NOVRAM operating parameters are invalid. Using defaults.
5	ASIC error. The Application Specific Integrated Circuit (ASIC) failed.

Serial/SMT3 LED Codes

Table 14 decribes the meaning of a blinking status light (LED) for all controllers that use Serial/SMT3 hardware. The SMT3 series includes the following controllers:

- Serial/SMT3, Serial/SMT3V, Serial/SMT3R, and Serial/SMT3RV controllers
- Serial/SMT3V Daughterboard
- PC Bus SMT3V and PC Bus SMT3RV controllers
- MousePort controller

For controllers in the SMT3 series, you can use the Unit Type command or the Unit Type Verify command to obtain the result of the self-test. You can interpret the result using the self-test bit described in Table 14.

Table 14. LED Diagnostic Codes for SMT3 Series of Controllers

LED Flashes (per 10 seconds)	**Self-Test Bit** (Unit Type or Unit Type Verify Commands)	**Error Description**
1	0	Reserved.
2	1	ROM error. Firmware checksum verification error.
3	2	PWM error. Unable to establish PWM operating range at power-up. Nonrecoverable error.
4	3	NOVRAM error. The operating parameters in the controller NOVRAM are invalid. Using defaults.
5	4	HDW error. The controller hardware failed (unable to initialize or configure gate array). Nonrecoverable error.
6	5	Reserved.
7	6	Cable NOVRAM error[1]. The linearization data in the cable NOVRAM is invalid.
8	7	NOVRAM2 error. The linearization data in the controller NOVRAM is invalid.

[1] The cable NOVRAM error will flash the LED until the controller receives the first valid command from the host.

TouchPen Diagnostics

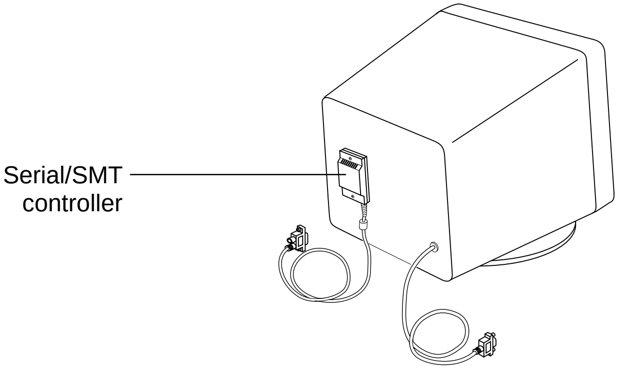
TouchPen controllers do not have an LED. You can, however, use the Unit Type command to obtain status information on the controller hardware.

APPENDIX A

Serial/SMT Controllers

The Serial/SMT controllers are compact (3.5 x 2.25 x 0.3 inches), RS-232 serial controllers. The controller can be internally mounted in your monitor, or enclosed in a molded plastic case (3.75 x 2.5 x 0.9 inches) and mounted to the back or side of your monitor.

This appendix provides controller specifications such as power requirements, environmental requirements, and cable connectors.



Serial/SMT2 Controller Mechanical

Figure 1 shows the overall dimensions of the Serial/SMT2 controller and the locations of the mounting holes and connectors.

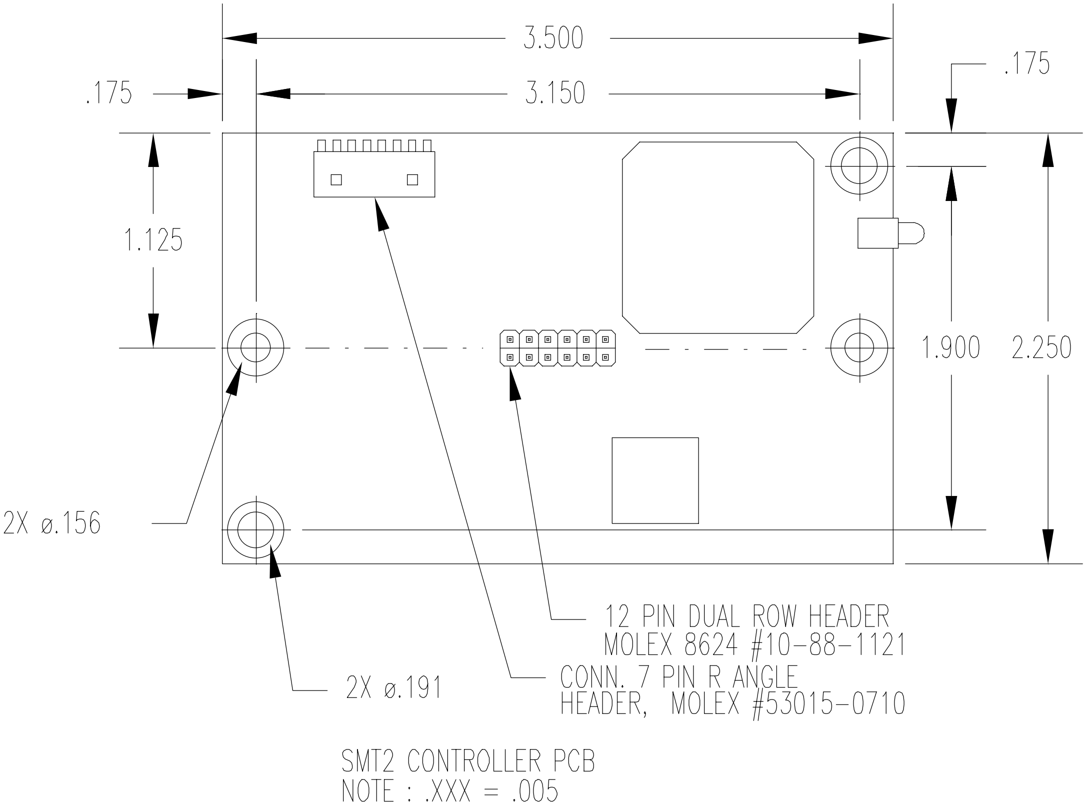


Figure 1. Serial/SMT2 Touchscreen Controller

Serial/SMT3 Controller Mechanical

Figure 2 shows the overall dimensions of the Serial/SMT3 controller and the locations of the mounting holes and connectors.

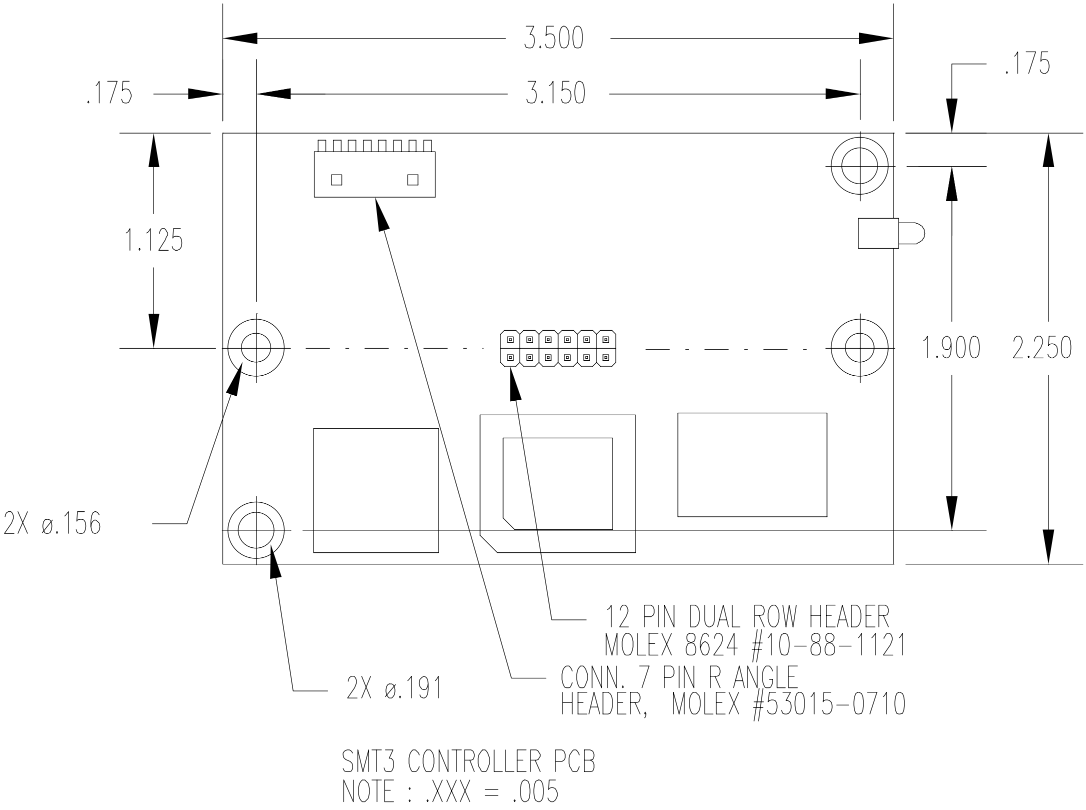


Figure 2. Serial/SMT3 Touchscreen Controller

Technical Specifications

Power:	*Serial/SMT2 Series of Controllers* +5V Input: +5 VDC (70 mA typical, 85 mA maximum), ±5% regulation, 100 mV maximum ripple and noise. +12V Input: 8 – 15 VDC (80 mA typical, 100 mA maximum), 400 mV maximum ripple. *Serial/SMT3 Series of Controllers* +5V Input: +5 VDC (47 mA typical, 60 mA maximum), ±5% regulation, 100 mV maximum ripple and noise. +12V Input: 8 – 15 VDC (47 mA typical, 60 mA maximum), 400 mV maximum ripple.
Operating Temperature:	0 to 55 degrees C.
Relative Humidity:	0 to 95% noncondensing.
Circuit Board Size:	3.5 x 2.25 inches, 0.3-inch clearance height.
Enclosure Size:	3.75 x 2.5 x 0.9 inches, molded plastic enclosure.

Status Light (LED) Diagnostics

The LED status light on the Serial/SMT controller provides information on power-up, screen touches, and hardware problems. For more information on the status light, refer to Chapter 3.

Female Connector on the Touchscreen Cable

The touchscreen (sensor) cable has a 12-pin (2 x 6) dual row female connector that plugs into the controller. Table 15 describes the pins on this connector.

Table 15. Touchscreen Cable Connector (2 x 6) for Serial/SMT Controllers

Pin	Wire Color	Description
1	———	Factory test point, no connect
2	———	Serial/SMT2: Factory test point, no connect
	———	Serial/SMT3: Factory test point, no connect
	Brown	Serial/SMT3R: Drives the sense layer of the resistive touchscreen
3	———	Factory test point, no connect
4	———	+5 VDC
5	Gray	Power supply ground
6	Green	Chassis (earth) ground
7	Orange	+12V input
8	Brown	Serial/SMT2: Connects to the sensor shield, which is driven with an AC voltage
	Brown	Serial/SMT3: Connects to the sensor shield, which must be grounded
	Cable shield/drain wire	Serial/SMT3R: Must be grounded
9	White	Upper right (UR) corner
10	Red	Lower right (LR) corner
11	Black	Upper left (UL) corner
12	Blue	Lower left (LL) corner

Communication Connector

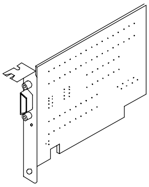
All Serial/SMT controllers have an attached RS-232 communication cable with a 9-pin D female connector. Table 16 describes the pins for this cable, which connects to a serial communication (COM) port on the PC. A 9-pin to 25-pin adapter is available.

Table 16. COM Connector (9-pin D, RS-232) for Serial/SMT Controllers

9-pin D	7-pin Molex	Wire Color	Description
1	No connection	———	Data Carrier Detect (DCD). Connected to DTR and DSR.
2	2	Brown	Transmit Data (TXD). Pin 2 is the controller’s output to the host.
3	3	Red	Receive Data (RXD). Pin 3 is the controller’s receive from the host.
4	No connection	———	Data Terminal Ready (DTR). Connected to DSR and DCD.
5	5	Blue	Power supply ground.
6	No connection	———	Data Set Ready (DSR). Connected to DTR and DCD.
7	1	Black	Request To Send (RTS). Connected to CTS.
8	4	Green	Clear To Send (CTS). Connected to RTS.
Sleeve	6	White	DC power jack (+5 VDC).
Pin	7	———	Cable shield connected to ground. DC power jack (ground).
Shell	7	———	Chassis (earth) ground.

APPENDIX B

PC Bus Controllers



The PC Bus controller is a half-slot, bus card that you install in your system. It has its own serial communication (COM) port, enabling you to use your existing COM ports for other peripherals.

To use the PC Bus controller, your computer must have an available 16-bit ISA (Industry Standard Architecture) expansion slot. The touchscreen cable connects to the port on the controller.

PC Bus Controller Mechanical

Figure 3 shows the overall dimensions of the PC Bus controller.

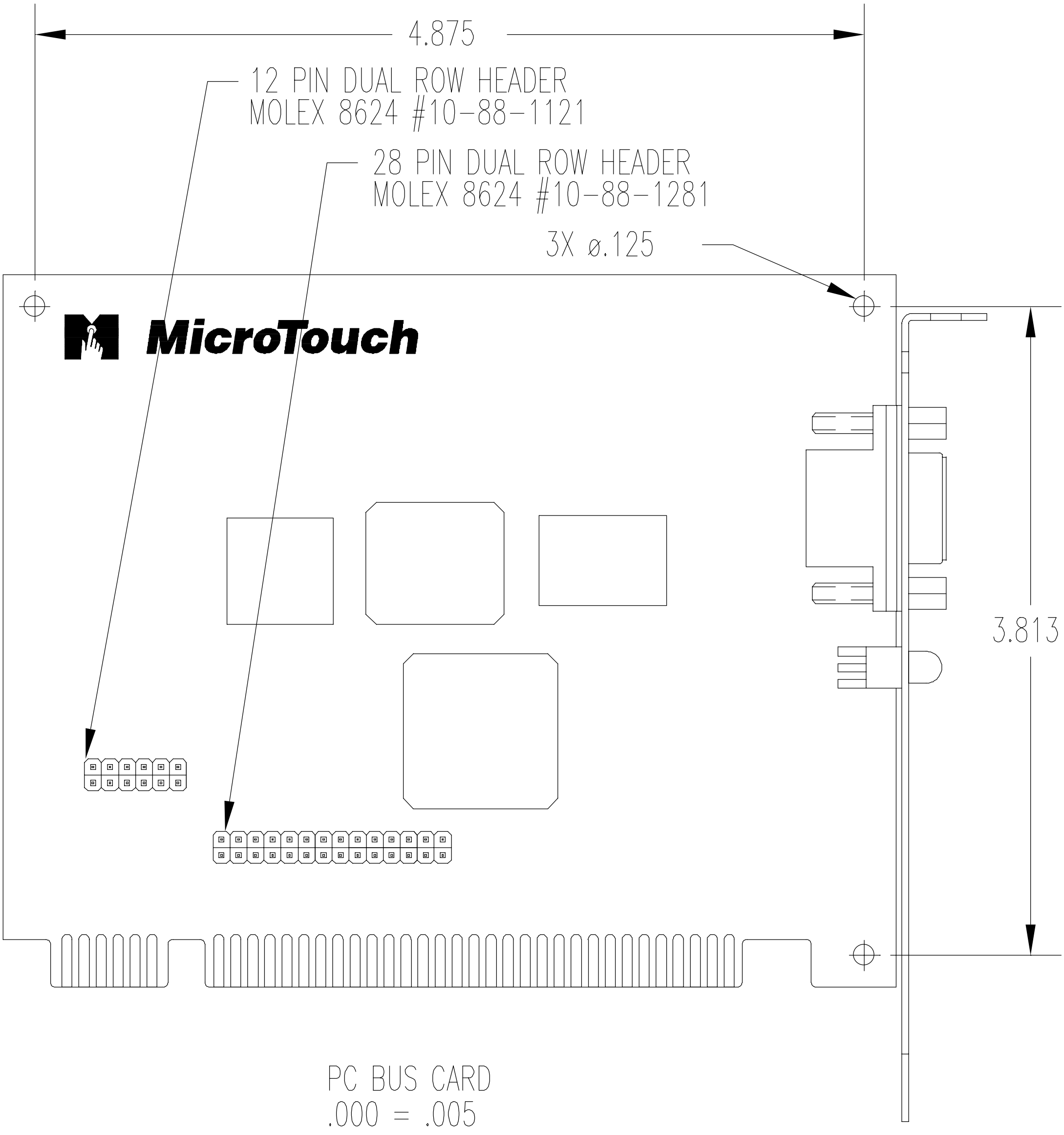


Figure 3. PC Bus Touchscreen Controller

Technical Specifications

Power:	+5 VDC (200 mA typical, 300 mA maximum), ±5% regulation. +12 VDC (70 mA typical, 100 mA maximum), ±10% regulation. -12 VDC (50 mA, typical, 70 mA maximum), ±10% regulation. 100 mV maximum ripple and noise.
Operating Temperature:	0 to 55 degrees C.
Relative Humidity:	0 to 95% noncondensing.
Circuit Board Size:	PC expansion bus half card.
Cable:	Shielded cable with a 9-pin D connector to attach the touchscreen to the PC Bus controller. Several lengths available. Do not substitute.

Status Light (LED) Diagnostics

The LED status light on the PC Bus controller provides information on power-up, screen touches, and hardware problems. For more information on the status light, refer to Chapter 3.

Connectors and Cabling

Table 17 describes the pins on the PC Bus cable.

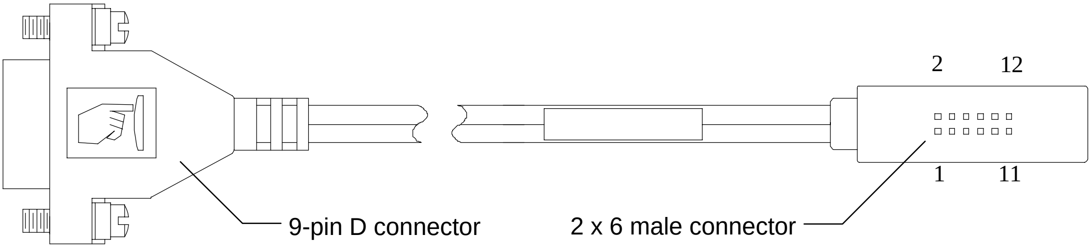


Table 17. Pin Definitions for the PC Bus Controller Cable

9-pin D Connector	2 x 6 Male Connector	Description
———	1 2 3	Test points
3	4	+5 VDC
Shell	5	Power supply ground
Shell	6	Chassis (earth) ground
No connection	7	+12V input
1	8	Shield (driven by AC voltage)
7	9	Upper right (UR) corner
6	10	Lower right (LR) corner
9	11	Upper left (UL) corner
8	12	Lower left (LL) corner

Jumpers on the PC Bus Controller

The PC Bus controller communicates with the computer through an asynchronous serial port on the controller. Every serial device in your PC must use a unique serial communication (COM) port and a unique interrupt request (IRQ).

The PC Bus controller uses the following default settings:

- Communication Port: COM3
- Interrupt Request: IRQ4

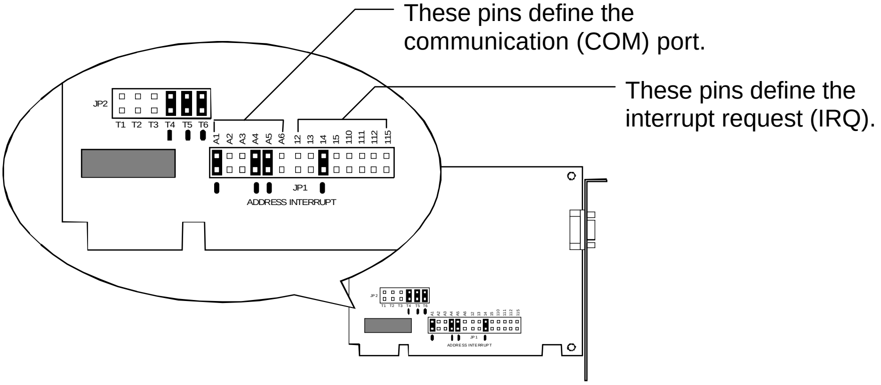
By default, most PC configurations use IRQ4 for COM1 or COM3. If your mouse is already using COM1/IRQ4, you need to change the default IRQ. To use different settings, change the jumpers *before* you install the PC Bus controller into your computer.

Handling the PC Bus Controller

The PC Bus controller is a printed circuit board. Static electricity can damage the controller. Before handling the controller, discharge static electricity from your body by touching bare, grounded metal. While handling the controller, do not walk across carpeting and do not touch materials (plastic, vinyl, Styrofoam) that create static electricity.

Locating the Jumpers

Take a moment to locate the jumpers that define the communication settings for the PC Bus controller.



These pins define the communication (COM) port.

These pins define the interrupt request (IRQ).

Setting the Communication Port

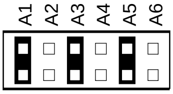
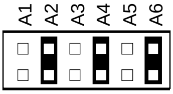
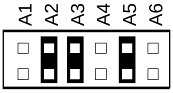
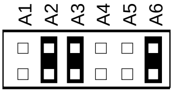
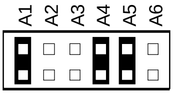
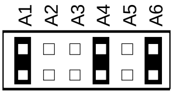
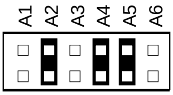
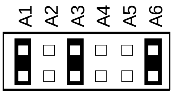
The pins labeled A1 – A6 on JP1 define the serial communication (COM) port for the PC Bus controller.

- Valid ports are COM1 through COM8. Refer to Table 18.
- The default is COM3.

In most PC configurations, the mouse uses COM1. However, if you are using a touchscreen and a mouse, both devices cannot use the same COM port. You must be sure there are no device conflicts.

Note: The PC Bus controller does support COM8. However, most PCs reserve COM8 for a diskette drive. Additionally, the Microcal Diagnostic utility only searches for the touchscreen on COM1 through COM7. Therefore, if you choose COM8, you cannot use Microcal to test the operation of the touchscreen.

Table 18. Setting the Communication Port (JP1)

COM Port (I/O Address)	**Jumper Settings**	**COM Port (I/O Address)**	**Jumper Settings**
COM1 (3F8 – 3FF)		COM5 (2E0 – 2E7)	
COM2 (2F8 – 2FF)		COM6 (2F0 – 2F7)	
COM3* (3E8 – 3EF)		COM7 (3E0 – 3E7)	
COM4 (2E8 – 2EF)		COM8 (3F0 – 3F7)	

* Default

Setting the Interrupt Request

The pins labeled I2 – I15 on JP1 define the interrupt request (IRQ) for the PC Bus touchscreen controller.

- Valid IRQs are 2, 3, 4, 5, 10, 11, 12, and 15. Refer to Table 19.
- The default is IRQ4.

You can use any IRQ for the PC Bus controller as long as another device in your system configuration is not using the same IRQ. The PC Bus controller cannot share an IRQ with another device.

Predefined IRQs

As outlined in Table 19, some IRQs have predefined uses. For example, most PC configurations use IRQs as follows:

- Use IRQ2 for the second Programmable Interrupt Controller (PIC)
- Use IRQ3 for either COM2 or COM4
- Use IRQ4 for either COM1 or COM3
- Use IRQ5 for the second parallel port (LPT2)

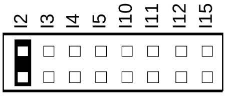
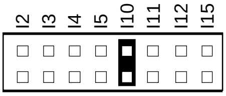
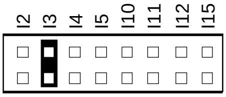
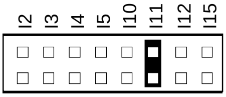
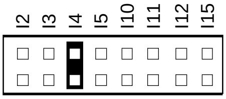
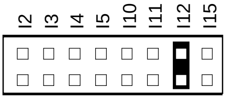
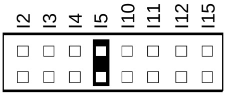
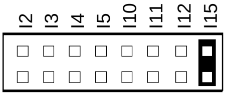
Additionally, some PC configurations may be using IRQ10 – IRQ15 for a modem or a primary/secondary IDE controller (for example, a hard disk controller). You must know the resources that your system devices use.

Preventing Device Conflicts

By default, the PC Bus controller uses COM3/IRQ4. If your system is already using COM1/IRQ4 for an existing device, be sure to change the IRQ that the PC Bus controller will use. The PC Bus controller must use a unique IRQ and cannot share an IRQ with another device.

For example, a mouse typically uses COM1/IRQ4. If you are using a mouse with the touchscreen, the mouse and the controller cannot both use IRQ4. If both devices use the same IRQ, a hardware conflict will result. The mouse or the touchscreen will not work.

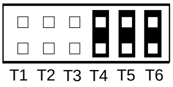
Table 19. Setting the Interrupt Request (JP1)

Interrupt	Jumper Settings	Interrupt	Jumper Settings
IRQ2 (9) (PIC)		IRQ10	
IRQ3 (COM2, COM4)		IRQ11	
IRQ4* (COM1, COM3)		IRQ12	
IRQ5 (LPT2)		IRQ15	

* Default

Setting the JP2 Jumper for Proper Operation

Look at the T1 – T6 pins on the JP2 jumper:



MicroTouch configures each controller at the factory with a jumper on the T4 pin, the T5 pin, and the T6 pin.

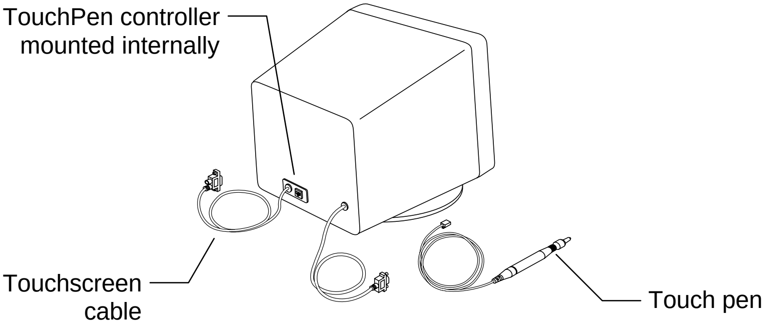
- A jumper must be on the T4 pin and the T5 pin for the PC Bus controller to work properly.
- The jumper on the T6 pin is a spare jumper and does not affect controller operation.

Warning: Placing a jumper on the T1 pin, the T2 pin, or the T3 pin will prevent the PC Bus touchscreen controller from working.

APPENDIX C

TouchPen Controllers

The TouchPen controller offers the same features as the Serial/SMT capacitive controller, with the addition of pen support. The controller can accept touch input from both a finger and the touch pen.



This RS-232 serial controller, which measures a trim 1.35 x 4.8 x 0.3 inches, is designed to easily fit inside flat panel displays and CRTs. The TouchPen controller is always mounted internally.

The tethered touch pen attaches to your display. Several cable lengths are available.

This appendix provides controller specifications such as power requirements, environmental requirements, and cable connectors.

TouchPen Controller Mechanical

Figure 4 shows the overall dimensions of the TouchPen controller and the locations of the mounting holes and connectors.

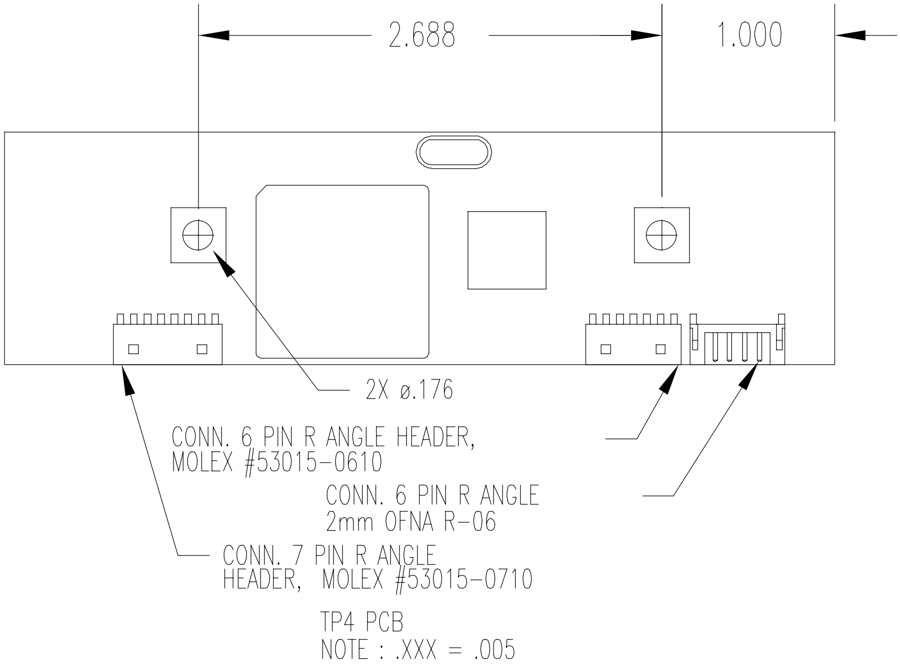


Figure 4. TouchPen Touchscreen Controller

Technical Specifications

Power:	+12V Input: 12 – 16 VDC (100 mA typical, 120 mA maximum), ±5% regulation, 100 mV maximum ripple and noise.
Operating Temperature:	0 to 55 degrees C.
Relative Humidity:	0 to 95% noncondensing.
Circuit Board Size:	1.35 x 4.8 inches, 0.3-inch clearance height.

TouchPen Diagnostics

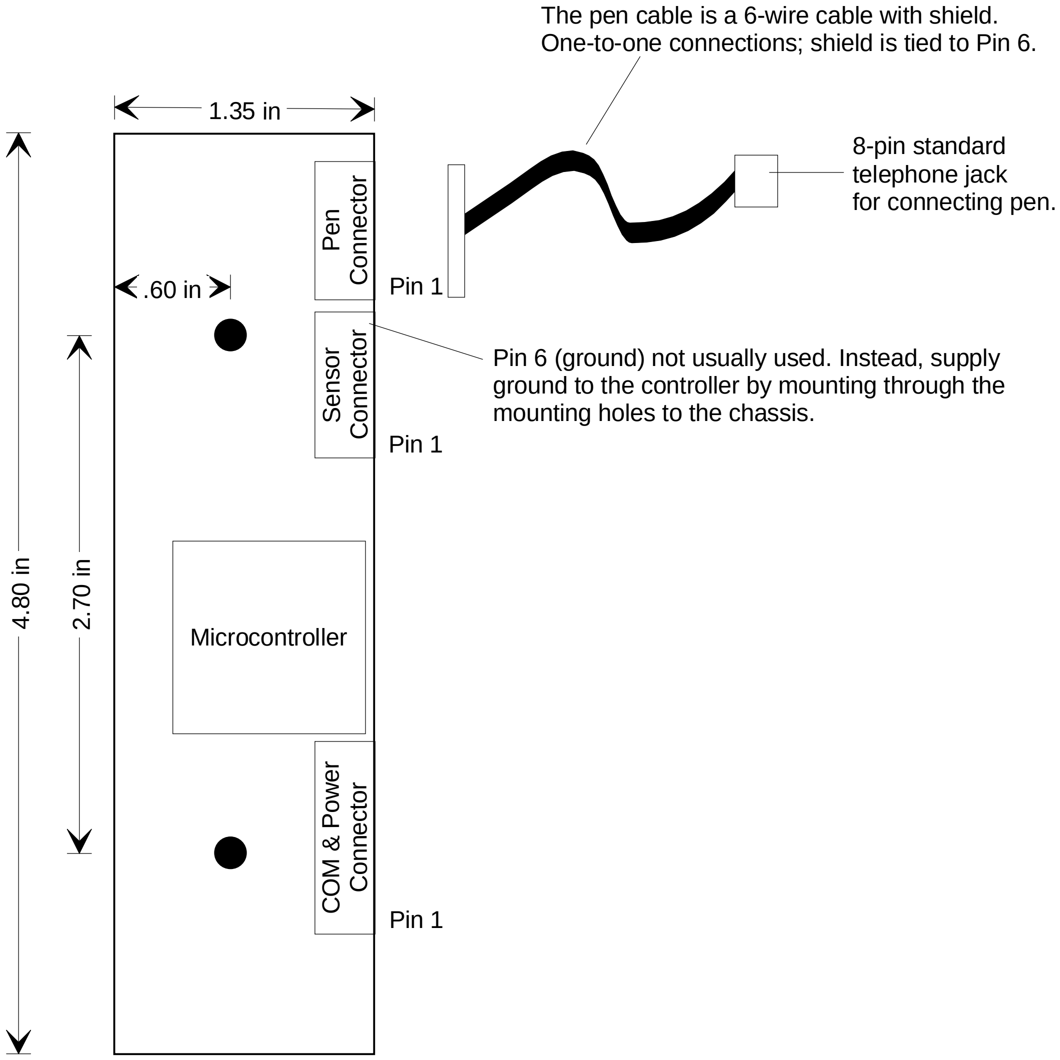
TouchPen controllers do not have an LED. You can, however, use the Unit Type command to obtain status information on the controller hardware.

Connectors and Cabling

Figure 5 shows the layout of the TouchPen controller and connectors. Table 20 describes the pins on each connector.

Table 20. Connectors and I/O Signals for the TouchPen Controller

Connector	Description	Pin Definitions
Pen	OFNA R-06 (6-pin, 2 mm) Connects to OFNA PH-06 housing with 3222PS-2 pins	Pin 1: Ground (supplied to pen) Pin 2: Pen Tip signal Pin 3: Tip switch (data bit S0) Pin 4: Side switch (data bit S1) Pin 5: Switch 3 (data bit S2) Pin 6: Shield drive (supplied to pen)
Sensor	Molex 53015 series (mates with Molex 51004 series)	Pin 1: Upper left (UL) corner Pin 2: Upper right (UR) corner Pin 3: Shield drive (supplied to sensor) Pin 4: Lower right (LR) corner Pin 5: Lower left (LL) corner Pin 6: Ground (supplied to controller)
COM and Power	Molex 53015 series (with 7 pins)	Pin 1: +12 to +16 VDC power input Pin 2: Ground (supplied to controller) Pin 3: RXD (data input to controller) Pin 4: TXD (data output from controller) The MicroTouch standard TouchPen controller does not use Pin 5 – Pin 7. The pin definitions are as follows: Pin 5: -12 to -16 VDC power input Pin 6: +5 VDC power input Pin 7: HSYNC input


The pen cable is a 6-wire cable with shield.
One-to-one connections; shield is tied to Pin 6.
1.35 in
8-pin standard
telephone jack
for connecting pen.
Pen Connector
Pin 1
.60 in
Sensor Connector
Pin 6 (ground) not usually used. Instead, supply ground to the controller by mounting through the mounting holes to the chassis.
Pin 1
4.80 in
2.70 in
Microcontroller
COM & Power Connector
Pin 1

Notes:
1. Diameter of mounting holes is 0.175 inches.
2. Connectors are Molex 53015 series and OFNA R-06.

Figure 5. Layout of the TouchPen Controller and Connectors

APPENDIX D

Daughterboard Controllers

MicroTouch has two models of the Daughterboard controller:

- SMT2 Daughterboard (Part Number: 14-73)
- SMT3V Daughterboard (Part Number: 14-89)

This chapter describes how to integrate a MicroTouch Daughterboard controller onto a system board. It provides the following information:

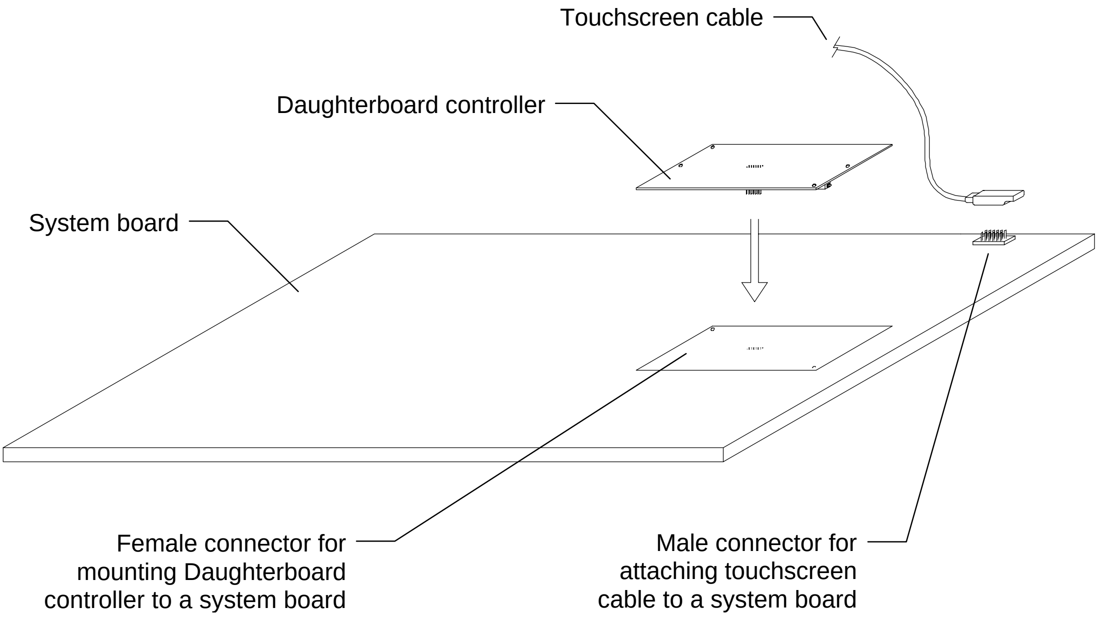
- Overview of the Daughterboard controller
- Mechanical drawing showing dimensions of the Daughterboard controller
- Connectors for attaching the touchscreen cable and mounting the Daughterboard controller onto a system board
- Connector pin outs for the touchscreen cable and Daughterboard controller
- Layout guidelines for the system board

Overview of the Daughterboard Controller

The Daughterboard controller is a low power, miniaturized board that is easily integrated onto a system board. The daughterboard design eliminates the more complicated layout and design issues of a chip set and interfaces directly with the touchscreen through CMOS asynchronous serial communication.

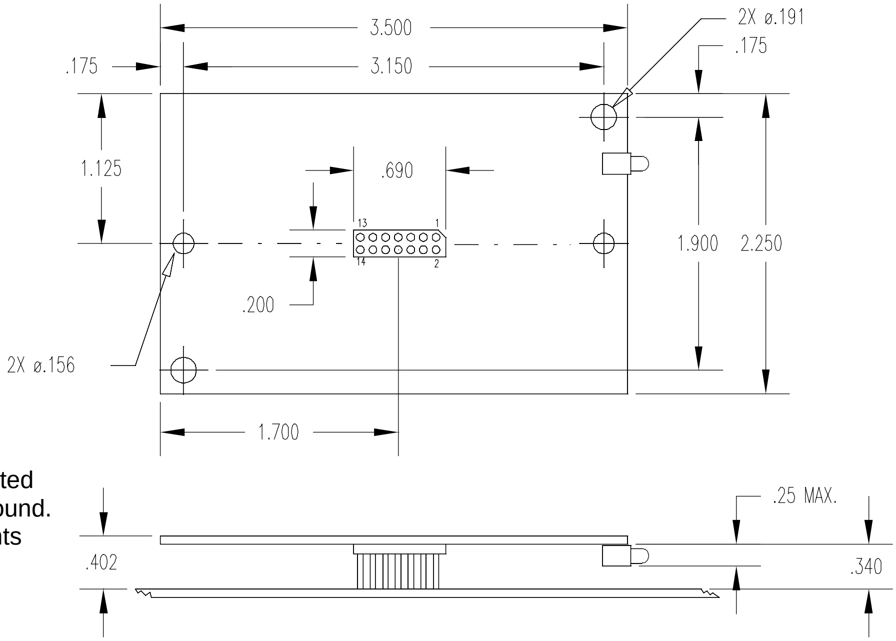
The controller supports CMOS communication rather than RS-232 communication. It is fully compatible with all MicroTouch software drivers and previous controllers. It has a 14-pin dual row male connector on the component side of the printed circuit board (PCB). This connector attaches the Daughterboard controller directly to the system board.

The following illustration shows a touch system configuration with the Daughterboard controller mounted to the system board.


Touchscreen cable
Daughterboard controller
System board
Female connector for mounting Daughterboard controller to a system board
Male connector for attaching touchscreen cable to a system board

Daughterboard Controller Mechanical

Figure 6 shows the overall dimensions of the Daughterboard controller, the locations of the mounting holes and connectors, and the maximum component height.



Notes:
1. All holes are plated thru and tied to ground.
2. All measurements are in inches.
3. .xxx = ±.005

Figure 6. Daughterboard Controller

Technical Specifications

Power:	Power is supplied by the host system. Requires 70 mA typical, 85 mA maximum at +5 VDC, ±5% regulation, 100 mV maximum ripple and noise.
Operating Temperature	0 to 55 degrees C.
Relative Humidity:	0 to 95% noncondensing.
Circuit Board Size:	3.5 x 2.25 inches, 0.3-inch clearance height.

Status Light (LED) Diagnostics

The LED status light on the Daughterboard controller provides information on power-up, screen touches, and hardware problems. For more information on the status light, refer to Chapter 3.

Serial Interface

The serial interface for the touchscreen connects the microcontroller and the host system. The communication interface uses a universal asynchronous communication protocol and the communication levels are CMOS compatible.

The default communication parameters for the Daughterboard controller are N, 7, 2 (no parity, seven data bits, and two stop bits). The standard transmission rate for controller-to-host communication is 9600 baud, with no handshaking. You can change the default communication rate, parity type, number of data bits, and number of stop bits for the Daughterboard controller.

Connectors and Cabling

To integrate the Daughterboard controller onto a system board, you must complete the following steps:

- Design a 12-pin male connector onto the system board to connect the touchscreen cable. The Molex part number for this connector is 8624-10-88-1121. (You can use an industry-standard equivalent part.)
- Design a 14-pin female connector onto the system board to connect the Daughterboard controller. The Molex part number for this connector is 70182-15-45-0907. (You can use an industry-standard equivalent part.)
- Supply the power, the CMOS serial communication, and the touchscreen signals to the Daughterboard controller via the 14-pin dual row male connector.

Female Connector on the Touchscreen Cable

The touchscreen cable has five wires that terminate into a molded 12-pin dual row female connector. It also contains a 1K bit non-volatile memory chip that stores screen linearization coefficients.

Table 21 shows the pin out for the female connector on the touchscreen cable. The Molex part number for this connector is 70182-15-45-0906.

Table 21. Pin Out for the Female Connector (12-pin Dual Row) on the Touchscreen Cable

Pin #	Signal	Definition	Supplied By	Description
1 2 3	CLK CE DIO	NOVRAM clock NOVRAM chip enable NOVRAM data	Daughterboard controller	Signals used by the controller to store and retrieve linearization coefficients from the non-volatile RAM (NOVRAM) located on the touchscreen cable.
4	VCC	+5 VDC	Host system	The +5 volt supply input provided by the host system. The controller requires 70 mA (typical), 85 mA (maximum), ±5% regulation, and 100 mV maximum ripple and noise.
5	GND	Signal ground	Host system	Provides the return path for the supply current.
6	CGND	Chassis ground	Host system	Provides a low impedance path to chassis ground.
7	—	Reserved	—	—
8	DRV	Shield drive	Daughterboard controller	Provides a digitized sine-wave output to drive the shield on the touchscreen.
9 10 11 12	UR LR UL LL	Upper right touch wire Lower right touch wire Upper left touch wire Lower left touch wire	Daughterboard controller	Bi-directional AC signals that detect the current used to determine the touch position.

Male Connector on the Daughterboard Controller

The Daughterboard controller has a 14-pin dual row male connector soldered to the component side of the PCB. This connector provides all sensor signals as well as TXD, RXD, and reset connections to the controller.

Table 22 shows the pin out for the male connector on the Daughterboard controller. The Molex part number for this connector is 8624-10-88-1141.

Table 22. Pin Out for the Male Connector (14-pin Dual Row) on the Daughterboard

Pin #	Signal	Definition	Supplied By	Description
1	CLK	NOVRAM clock	Daughterboard controller	Signals used by the controller to store and retrieve linearization coefficients from the non-volatile RAM (NOVRAM) located on the touchscreen cable.
2	CE	NOVRAM chip enable		
3	DIO	NOVRAM data		
4	VCC	+5 VDC	Host system	The +5 volt supply input provided by the host system. The controller requires 70 mA (typical), 85 mA (maximum), ±5% regulation, and 100 mV maximum ripple and noise.
5, 6	GND	Signal ground	Host system	Provides the return path for the supply current.
7	RST	Reset (active low)	Host system	An active low input that can reset the controller.
8	DRV	Shield drive	Daughterboard controller	Provides a digitized sine-wave output to drive the shield on the touchscreen.
9	UR	Upper right touch wire	Daughterboard controller	Bi-directional AC signals that detect the current used to determine the touch position.
10	LR	Lower right touch wire		
11	UL	Upper left touch wire		
12	LL	Lower left touch wire		
13	RXD	Data from host to the Daughterboard (receive)	—	Provide serial communication between the Daughterboard controller and the host system.
14	TXD	Data to host from the Daughterboard (transmit)		

Supplying Power to the Daughterboard Controller

The host system provides the power to the Daughterboard controller. The controller requires 70 mA (typical), 85 mA (maximum supply current), at +5 volts DC, ±5% regulation, and 100 mV maximum ripple and noise.

Electrical Specifications for Transmit and Receive

Table 23 lists the D.C. characteristics for the universal asynchronous receiver transmitter (UART). Specifically, these electrical specifications are for Pin 13, RXD (data from host) and Pin 14, TXD (data to host) on the male connector on the Daughterboard controller.

Table 23. D.C. Characteristics

Symbol	Description	Min	Max	Units	Test Conditions
V_{OL}	Output Low Voltage		0.3	V	I_{OL} = 200 µA
			0.45	V	I_{OL} = 3.2 mA
			1.5	V	I_{OL} = 7 mA
V_{OH}	Output High Voltage	V_{CC} – 0.3		V	I_{OH} = –200 µA
		V_{CC} – 0.7		V	I_{OH} = –3.2 mA
		V_{CC} – 1.5		V	I_{OH} = –7 mA

Layout of the System Board

Figure 7 shows some typical techniques for two layer boards. The same techniques apply to four layer boards except that the various planes may be put on inner layers.

To design the Daughterboard controller onto the system board and to ensure optimal operation, follow these guidelines:

A. Route the shield signal (Pin 8) and the corner drive lines (Pin 9-UR, Pin 10-LR, Pin 11-UL, Pin 12-LL) on the component side. Keep the shield connection (Pin 8) and the four corner runs (Pins 9 – 12) away from all other runs on the board.

B. Design *locking posts* onto the system board to hold the board in place. If you use non-insulating locking posts, make sure you connect the posts to signal ground.

C. Keep the ground and power etches (or planes) as short and as low impedance as possible.

D. Keep the runs from the touchscreen connector to the Daughterboard connector as short as possible. The connections from Pins 1 – 12 on the touchscreen connector to Pins 1 – 12 on the Daughterboard connector are 1-to-1. Keep the four corner runs and the shield connection (Pins 8 – 12) away from all other runs on the board.

E. Connect the diodes as shown in the Inset A-A diagram. Connect the VCC (+5 V) and the **chassis ground** to the diodes. Place the diodes as close as possible to the touchscreen connector on the system board.

F. Make sure you keep the other runs from the touchscreen connector away from the four corner and shield runs. Be sure the CMOS connections for receive and transmit (Pin 13 and Pin 14) are routed away from the four corner runs and the shield signal.

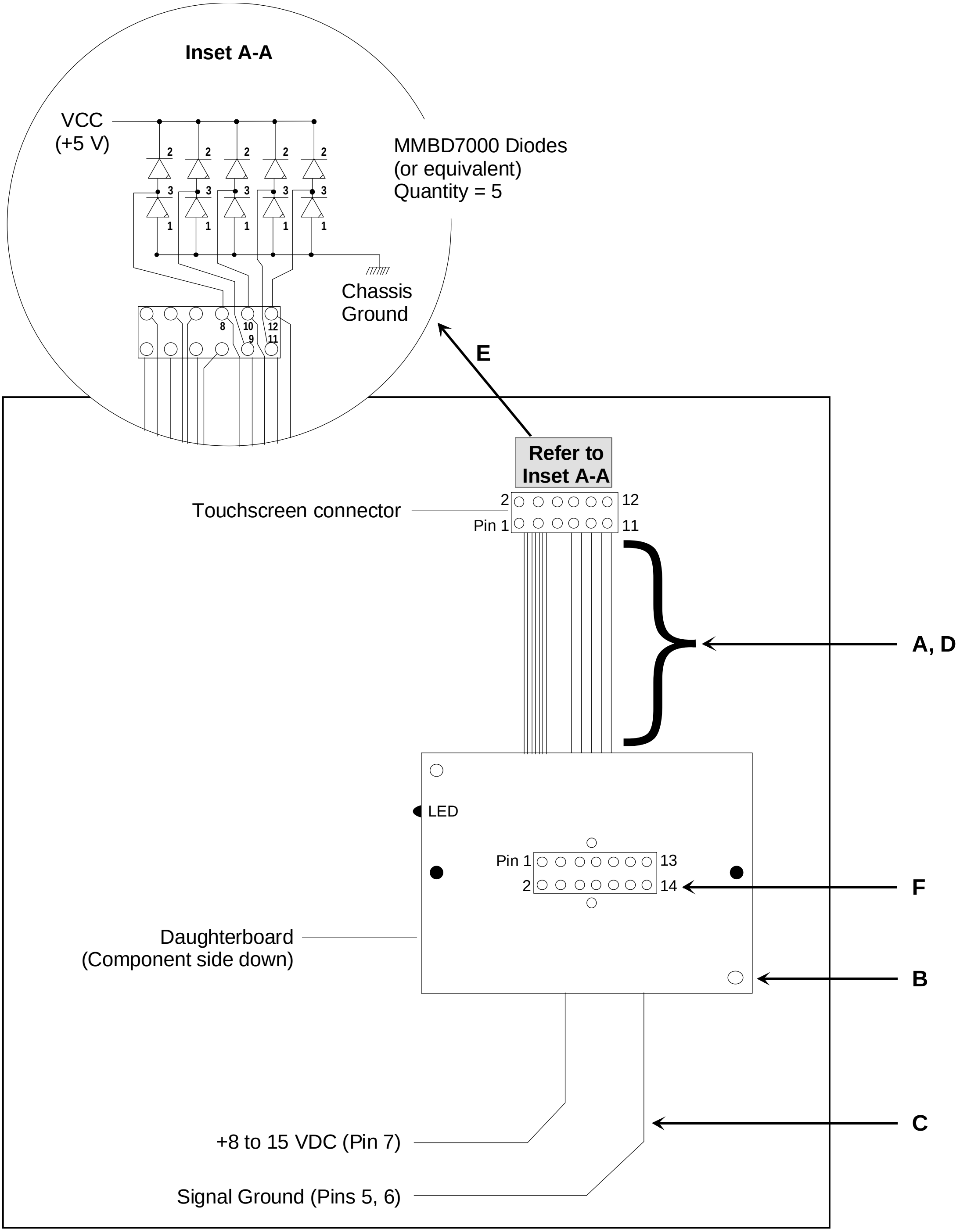


Figure 7. Layout of the Daughterboard Controller on the System Board

Electrostatic Discharge (ESD) Considerations

You should test your completed board layout for general design as well as ESD in accordance with the IEC 801-2 specification. In general, ESD characteristics are different for each design.

Disclaimer: This chapter describes a means by which you can make the connections. It does not guarantee that your board design will pass ESD testing. MicroTouch is not liable for design problems.

The suggested ESD protection diodes are as follows:

Type:	MMBD7000 dual diodes
Size:	SOT-23 case
Quantity:	5

Ordering Information

Table 24 lists the connectors you need to integrate the MicroTouch Daughterboard controller onto a system board. To order these connectors, contact Molex Incorporated at the following address:

Molex Incorporated
2222 Wellington Court
Lisle, Illinois 60532
(312) 969-4550

Table 24. Connectors Required on the System Board

Connector Description	Molex Part Number
Male connector for mounting the touchscreen cable	8624-10-88-1121
Female connector for mounting the Daughterboard controller	70182-15-45-0907

Index

MicroTouch®

MicroTouch Systems, Inc.

Corporate Headquarters
UNITED STATES 978-659-9000, Fax 978-659-9100
World Wide Web: http://www.microtouch.com E-Mail: touch@microtouch.com
FACTURA KIOSKS, a division of MicroTouch Systems, Inc., 716-424-4300, Fax 716-424-4335

MicroTouch Worldwide Offices
AUSTRALIA +61 (03) 9561 7799 • FRANCE +33 (1) 45 13 90 30 • GERMANY +49 (0) 211-59907-0
HONG KONG +852 2333 6138 • ITALY +39 (0) 39-230-2230 • JAPAN +81 (044) 811-1133
KOREA +82 (2) 552-3198 • TAIWAN +886 (02) 2226-0875 • UNITED KINGDOM +44 (0) 1235-444400

Document Number: 19-213