tyco
Electronics

Elo Entuitive Touchmonitor User Guide 1229L

For 12.1" LCD Touchmonitors with Optional Magnetic Stripe Reader, Finger Print Reader and Rear Facing Customer Display

Revision A

Elo Entuitive Touchmonitor User Guide

12.1" LCD Touchmonitor with Optional Mag Stripe Reader, Finger Print Reader and Rear Facing Customer Display

ET1229L Series

entuitive
Touchmonitors

Revision A

P/N 008577

Elo TouchSystems, Inc.

1-800-ELOTOUCH
www.elotouch.com

tyco
Electronics

elo
TOUCHSYSTEMS

Disclaimer

The information in this document is subject to change without notice. Elo TouchSystems makes no representations or warranties with respect to the contents hereof, and specifically disclaims any implied warranties of merchantability or fitness for a particular purpose. Elo TouchSystems reserves the right to revise this publication and to make changes from time to time in the content hereof without obligation of Elo TouchSystems to notify any person of such revisions or changes.

Trademark Acknowledgments

IntelliTouch, SecureTouch, AccuTouch, Entuitive, and MonitorMouse are trademarks of Elo TouchSystems, Inc.

Other product names mentioned herein may be trademarks or registered trademarks of their respective companies. Elo TouchSystems claims no interest in trademarks other than its own.

Table of Contents

CHAPTER

1

INTRODUCTION

Product Description

The 1229L is a retail terminal designed to present information to the operator and the customer. The 1229L is available in serial and USB versions. The 1229L functionally consists of a 12.1" LCD main display with a touchscreen, an optional vacuum fluorescent display (VFD) Customer Display, an optional fingerprint reader (USB version only), an optional credit card reader, and a 6 port USB (USB version only) Hub. The main display element is a 12.1 inch diagonal SVGA resolution (800x600) LCD display. The main display will consist of an LCD Display and touchscreen. Three types of touchscreens can be selected in the 1229L as options. They are AccuTouch, Intellitouch, and IR. The Customer Display is a twenty character two line vacuum fluorescent display (VFD). The VFD can display 40 characters in a 20 character 2 row format. Each character is made by various fonts using a 5x7 pixel format matrix.

The fingerprint reader translates illuminated images of fingerprints into digital code for further software processing, e.g. enrollment (fingerprint registration) and verification (authentication of registered users). The fingerprint reader uses the SEIR method and CMOS image sensor to capture high contrast, high resolution fingerprint images. A series of algorithms extracts minutiae data from the image, mapping the distinguishing characteristics of fingerprint ridge ends, splits, dots, and arches. To identify or verify a fingerprint, a proprietary matching algorithm compares the extracted minutiae points from the input fingerprint on the optical module to a previously stored sample. The entire matching process takes roughly one second. There is a fingerprint reader available in the USB version of the 1229L and none in the serial version of the 1229L.

The credit card reader reads all three stripes on a standard credit card or drivers license. The credit card is read by sliding the credit card, stripe side toward the display through the credit card reader forward or backward. There is a serial credit card reader and a USB credit card reader.

The Hub provides 4 internal USB ports to be used by the credit card reader, the fingerprint reader, the touchscreen, and the customer display. The hub also supplies two USB ports to the outside of the back of the 1229L for external use. The hub is only used by the USB version of the 1229L. The 1229L is powered by 12 VDC from a universal type power supply brick.

Detailed LCD Display Performance Requirements

2.1" TFT LCD Display Panel	
Display Format:	1800 x 600
Display area 12.1"	246.0mm (H) x 184.5mm (V)
Pixel Pitch 12.1"	0.3075mm (H) x 0.3075mm (V)
Contrast Ratio	300 typical
Brightness	300 nits typical with no touchscreen; 246 nits with AccuTouch; 276 nits with IntelliTouch; 276 nits with IR
Accutouch Transmission	82% typical
IntelliTouch Transmission	92% typical
IR Touchscreen Transmission	92% typical
Response Time	Tr = 30 msec/Tf = 20 msec typical
Display Color	16 million colors with dithering
Horizontal Viewing Angle	+/- 60 degrees typical at CR=10
Vertical Viewing Angle	-50/+40 degrees typical at CR=10

Video Interface Connector	The video interface connector is a High density 15 pin type HD-15.
Indicator Lamps	The LCD display shall provide an indicator lamps to give the status of the Power Management System
Audio Jack	Provides for audio input signal to the stereo speakers mounted in the 1229L.

Customer Display

The Customer Display is a twenty character two line vacuum fluorescent display (VFD). It consists of a VFD and VFD controller. There is a serial version controller and a USB controller. The actual VFD is common to the serial and USB versions.

Serial Version

Optical Parameters	
Characters per row	20
Number of rows	2
Character configuration	5x7 dot matrix
Character Height	9.5 mm
Character width	6.2 mm
Character configuration	ASCII
Character color	Blue green
MTBF	300,000 hours

Fingerprint Reader

There is a fingerprint reader in the USB version and none in the serial version.

General Description-FDU01B is a PC peripheral FRD (Fingerprint Recognition Device) for USB (Universal Serial Bus) connections.

Specifications	
Sensor	SecuGen FOR
Image Capture Speed	600ms / frame
Image Transfer Speed	500Byte / ms
Pixel Resolution	356 x 292
USB Signaling Type	Full Speed Type

Theory of Operation

The USB host initiates communication with the FDU01 using operation commands (Sensor LED On, Fingerprint Capture Start and Stop). Fingerprint data are then captured by the CMOS sensor at a total image size of 356 x 292 with 8-bit gray level. The image frame transfer speed is 500 bytes/ms. It takes about 600 milliseconds to send one frame of image data over USB protocols. FDU01 uses the SecuGen FOR (Fingerprint Optic Reader).

Sensor Specifications

Sensor	CMOS Image Sensor
Resolution	500dpi
Verifying Time	< 1sec
Image Capture Error Rate	< 0.1%
Life Time	Typically 40,000Hrs

Credit Card Reader

There is a serial credit card reader and a USB credit card reader.
The USB version is available in HID and Keyboard emulation versions. The reader reads all three stripes on a standard credit card or drivers license. A green LED provides the operator with continuous status of the reader operations.

Reference Standards-Conform to applicable standards	International Standards Organization, American National Standards Institute, California Drivers License, American Association of Motor Vehicle Administrators
Message Format	ACCII
Card Speed	3 to 50 IPS
MTBF	Electronics 125,000 hrs; Head 1,000,000 passes

Six Port USB Hub

The Hub provides 4 internal USB ports to be used by the credit card reader, the fingerprint reader, the touchscreen, and the customer display. The hub also supplies two USB ports to the outside of the back of the 1229L for external use. The hub is only used by the USB version of the 1229L. The hub meets the following requirements:

Specification

Full compliance with USB specification 1.0, 1.1 and HID Class Definition Rev 1.0.

Hub shall be self powered

Hub shall provide 2 external and 4 internal downstream ports with individual port over current detection, protection and recovery. Supports both Open Host Controller Interface (OHCI) and Universal Host Controller Interface (UHCI).

Supports Suspend and Resume operation.

Bus fault detection and recovery.

External 12 VDC Power Supply

The 1229L shall be powered by 12 VDC from a universal type power supply brick. The power supply shall provide the following capability:

Input voltage 100 to 240 vac

Input frequency 47 to 63 hz

Output voltage 12 vdc

Output line and load regulation +/- 2%

CHAPTER

2

INSTALLATION AND SETUP

This chapter discusses how to install your LCD touchmonitor and how to install Elo TouchSystems driver software.

Unpacking Your Touchmonitor

Check that the following items are present and in good condition:

Touchmonitor

Brick power supply

VGA cable

Serial cable
(one for each option)

OR

USB cable

CD and Quick Install Guide

Power cable US/Canada

European power cable

Product Overview

Main Unit

Rear View

Side View

Base Bottom View

Kensington™ Lock

The Kensington™ lock is a security device that prevents theft. To find out more about this security device, go to http://www.kensington.com.

Serial Interface Connection

Your touchmonitor comes with one of the following touchscreen connector cables: *Serial* (RS-232) cable ***or*** *USB* cable. (For Windows 98, 2000, Me and XP systems only.)

To set up the display, please refer to the following figures and procedures:

Remove the Back Cover

The cables are connected at the back of the monitor.

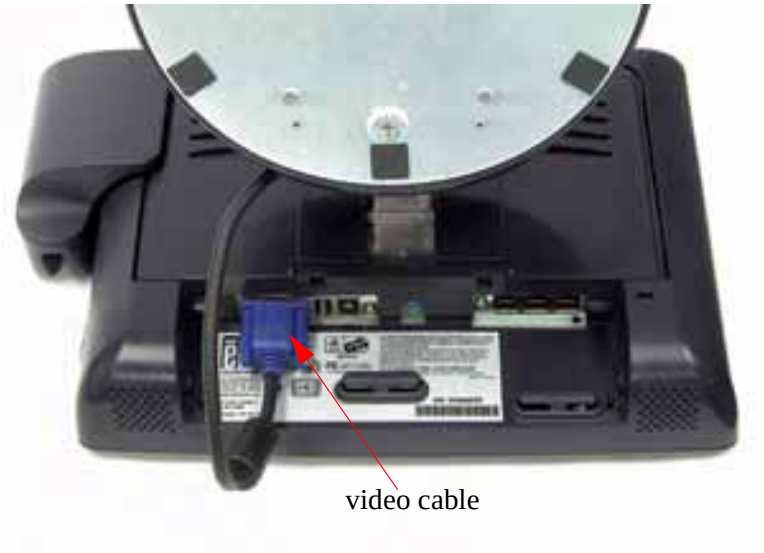
- To remove the cover, grasp the lip of the cover and pull towards you until it snaps off.

CAUTION Before connecting the cables to your touchmonitor and PC, be sure that the computer and touchmonitor are turned off.

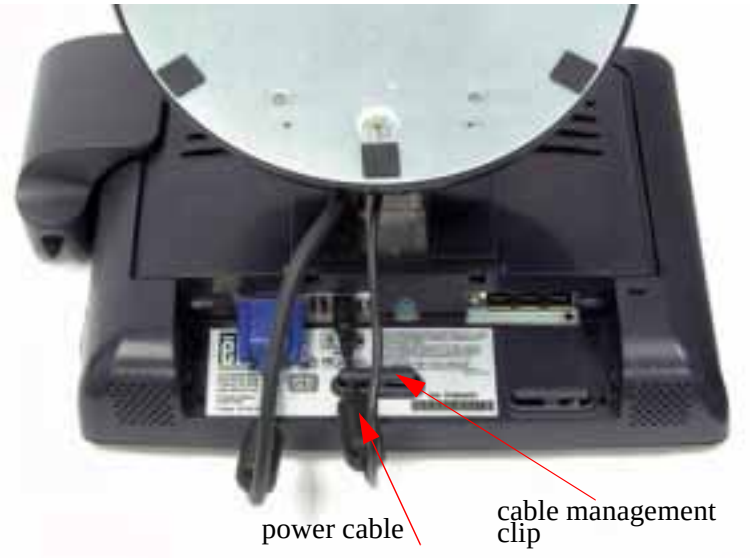
NOTE: Before connecting the cables to the touchmonitor, route all the cables through the hole in the stand as shown in the picture above.

Serial Connection Option

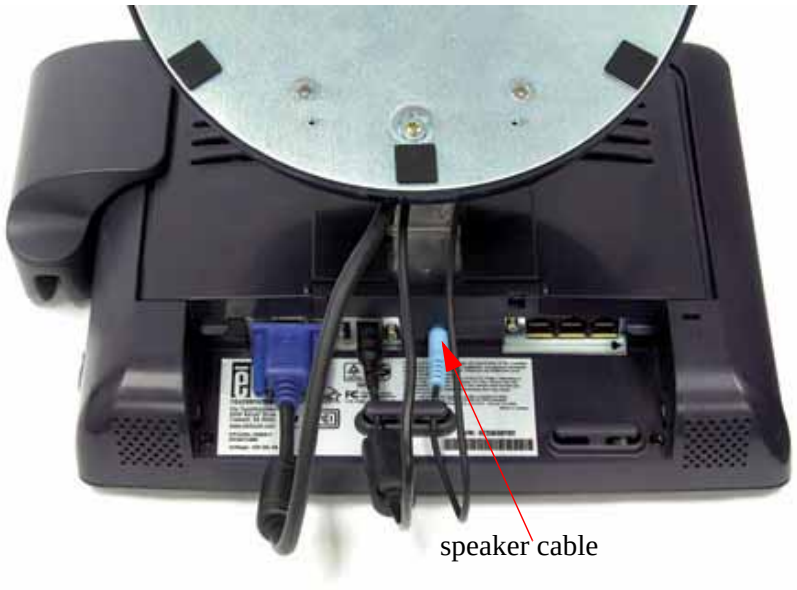
The following illustrations guide you step by step in connecting your touchmonitor using a serial cable connection.



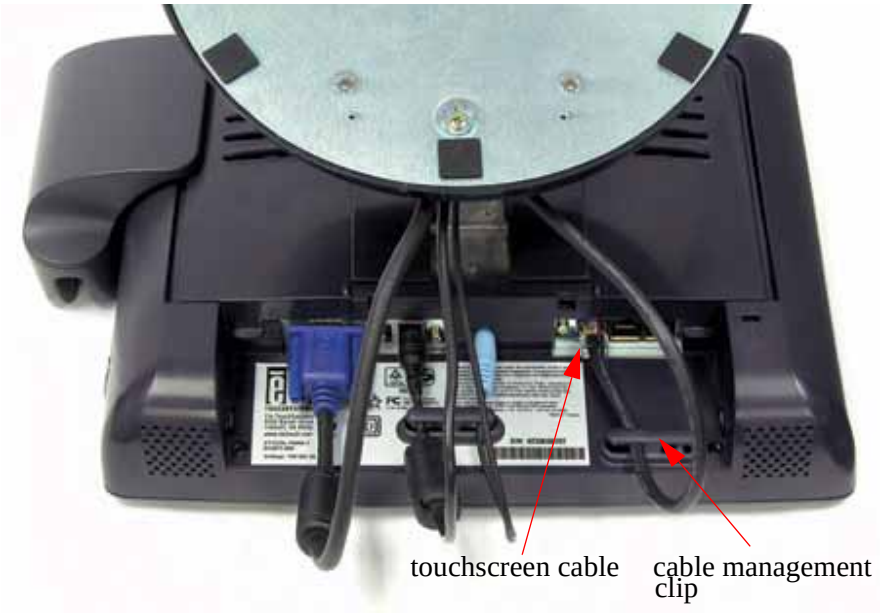
- Connect one end of the *video cable* to the rear side of computer and the other to the LCD monitor. Tighten by turning the two thumb screws clockwise to ensure proper grounding.



- Connect one end of the *brick power supply* to the monitor and the other end to the connector of the power cable.
- Connect the *power cable* to the power port in the monitor.
- After connecting the power cable, secure the cable under the cable management clip.

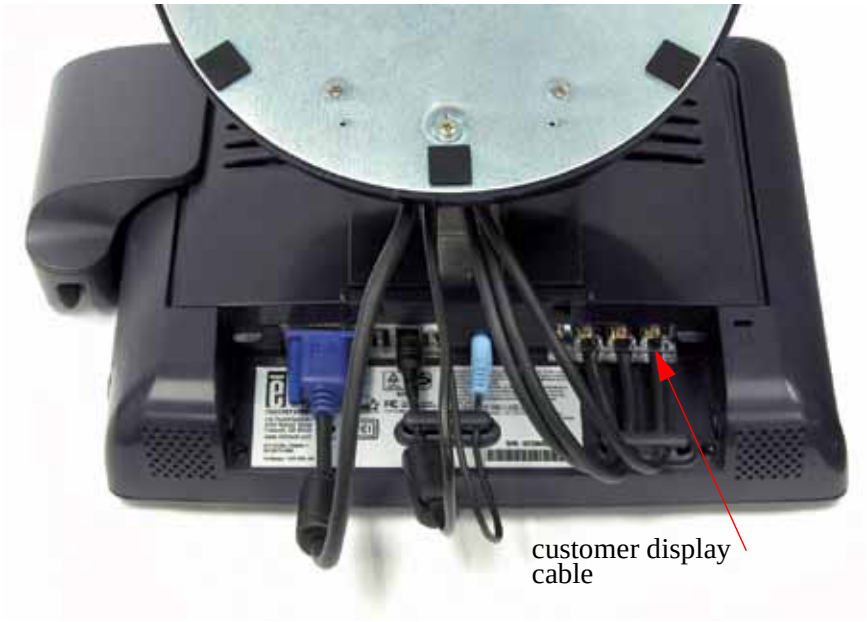


- Connect one end of the *speaker cable* to the speaker port in the computer and the other end to the port in the monitor
- After connecting the speaker cable, secure it under the cable management clip.



- Connect one end of the *serial touchscreen cable* to the rear side of computer and the other to the LCD monitor. Secure the cable under the cable management clip

- Connect one end of the *MSR cable* to the computer and the other end to the monitor.

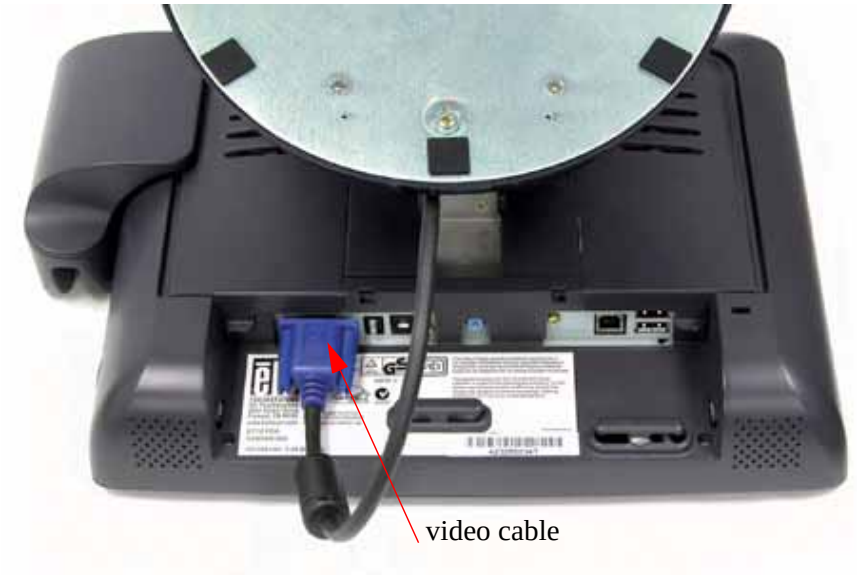


Connect one end of the *customer display cable* to the computer and the other end to the monitor. Secure the cable under the cable management clip.

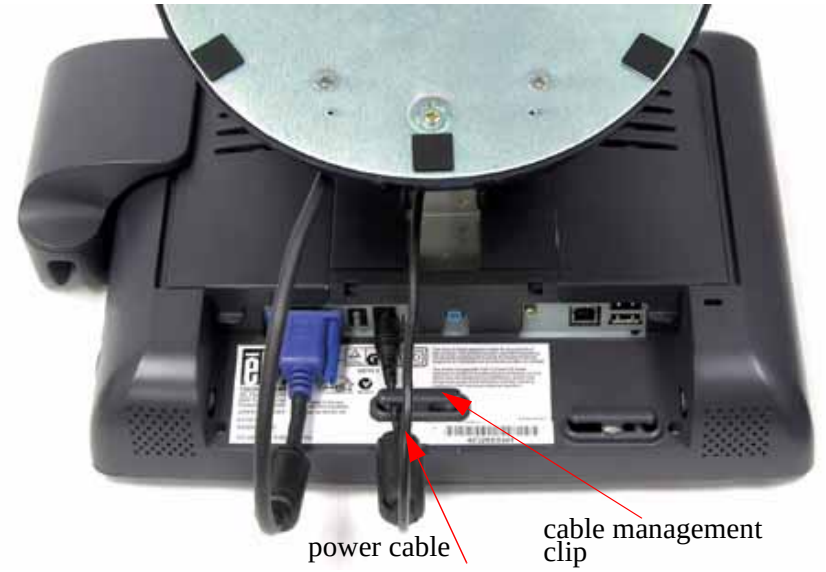
- Press the *power button* on front panel to turn the monitor power on.

USB Interface Connection

The following illustrations guide you step by step in connecting your touchmonitor using a USB cable connection.

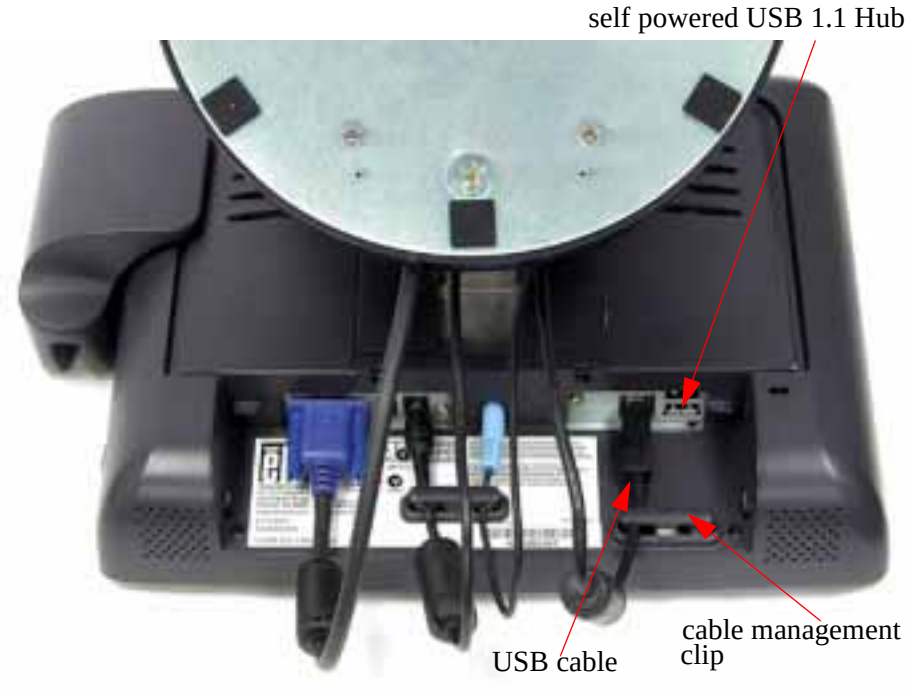


- Connect one end of the *video cable* to the rear side of computer and the other to the LCD monitor. Tighten by turning the two thumb screws clockwise to ensure proper grounding.



- Connect one end of the *brick power supply* to the monitor and the other end to the connector of the power cable.
- Connect the power cable to the power port in the monitor.
- After connecting the power cable, secure it under the cable management clip.

- Connect one end of the *speaker cable* to the speaker port in the computer and the other end to the port in the monitor. After connecting the speaker cable, secure it under the cable management clip,



- Connect one end of the *USB cable* to the rear side of the computer and the other to the LCD monitor. Secure the cable under the cable management clip.
- The USB cable is for optional touch, MSR, CD and Finger Print Reader. Only one USB cable is needed because the device contains a self powered USB 1.1 Hub. Two self powered ports are available for running other USB devices. For touch only, no USB Hub is present.

Replace the Back Cover

- When you have attached all the cables to the monitor, gently bring all the cables toward the stand so they fit under the cover lip.
- Snap the back cover in place over the connections.

Optimizing the LCD Display

To ensure the LCD display works well with your computer, configure the display mode of your graphic card to make it less than or equal to 800 x 600 resolution, and make sure the timing of the display mode is compatible with the LCD display. Refer to Appendix A for more information about resolution. Compatible video modes for your touchmonitor are listed in Appendix C.

Installing the Peripheral Device Drivers

Finger Print Reader

NOTE: This driver is for MS Windows 9x through XP.

1. On the TouchTools CD, browse to *Touch Monitor Peripherals\Finger Print Readers\driver\EasyInstall\FDP02.*
2. Double-click *setup.exe*

Follow the Install Shield Wizard procedure to complete the installation.

For a detailed software development kit, browse to *Touch Monitor Peripherals\Finger Print Readers* and open the following files:

- FDxSDKforWindows1 .20.zip
- SecuBSPSDK for Windows2 .10.zip

You will enter one of the following serial numbers depending on your operating system:

- FDx SDK for Windows: 31-100s101-3586383
- FDx SDK for Windows CE: 32-100s101-9713291
- SecuBSP SDK for Windows: 41-100s101-7685871
- SecuBSP SDK for Windows CE: 42-100s101-1155462
- SecuBSP SDK: 51-100s101-5963137

Once the driver setup is complete, the demo program can be run from *Touch Monitor Peripherals\Finger Print Readers\FPR Demo\BSPDemo.exe*

Magnetic Stripe Reader

No drivers are needed.

Testing the Serial Version:

1. Insure the MSR serial is connected. (Make a note of the serial port, COM1, COM2, etc. it is connected to.)
2. Launch *HyperTerminal*
3. Go to *file>new connection* and type "*MSR*" then select *OK*.
4. In the *Connect to Menu* select *Com port* used in step one, then select *OK*.
5. In the *Connection Property* box set the following values:
   - Bits per second to 9600
   - Data Bits to 8
   - Parity to None
   - Stop bits to 1
   - Flow control to Hardware

The device will immediately display the serial number. Slide any credit card through the MSR to view the data.

Testing the USB MSR Keyboard Emulation

1. Plug in the device.
2. Open MS Word.
3. Slide the card through the MSR to view the data.

Testing the USB-HID Class MSR

1. On the CD, browse to *Touch Monitor Peripherals\Magnetic Stripe Card Readers\Demo.*
2. Open the *Readme.txt* and follow instructions to test the unit.

Rear Facing Customer Display

Serial Customer Display

The serial customer displays do not need drivers. To get an image on the display:

1. Plug the DB-9 connector into serial port COM1 of computer.
2. Plug the RJ-45 cable into display.
3. In windows click on *Start > Run*
4. Enter *cmd > OK*
5. Type *MODE COM1 96,N,8,1 > Enter*
6. Type *TYPE CON > COM1 > Enter*
7. Type *ELO > Enter*

The display will show ELO.

USB Customer Display

Plug in the USB cable attached to the Customer Display unit. The New Hardware Wizard dialog box will appear.

1. Choose *Next* and select *"Search for the best driver for your device (Recommended)"* then choose *Next*.
2. When a list of search locations is displayed, place a checkmark on the drive containing the driver package:

Touch Monitor Peripherals\Rear Facing Customer Displays\Drivers\xxx\LCLD9.sys, where xxx is Win98 for a Windows 98 based system or 2000 for a Windows XP/2000 based system.

3. Insert the disk into your drive. (If the driver files have been copied to your hard drive or have been distributed on CD, place a checkmark on "Specify a location" and browse to select the directory containing the driver files.)
4. Choose *Next*. Once the Customer Display driver has been detected choose *Next* again.
5. Wait while driver files are copied to your computer.
6. Insert your Windows CD if prompted and choose *Finish*

To test the drivers:

1. In windows click on *Start* > *Run*
2. Enter *"cmd"* > *OK*
3. Type "*ECHO ELO*>\.*LCLD9*\" > *Enter*

The display will show ELO.

Installing the Touch Driver Software

Elo TouchSystems provides driver software that allows your touchmonitor to work with your computer. Drivers are located on the enclosed CD-ROM for the following operating systems:

- Windows XP
- Windows 2000
- Windows Me
- Windows 98
- Windows 95
- Windows NT 4.0
- CE 2.x, 3.0, 4x
- Windows XP Embedded
- Windows 3.x
- MS DOS
- OS/2

Additional drivers and driver information for other operating systems (including Macintosh and Linux) are available on the Elo TouchSystems web site at www.elotouch.com.

Your Elo USB touchmonitor is plug-and-play compliant. Information on the video capabilities of your touchmonitor is sent to your video display adapter when Windows starts. If Windows detects your touchmonitor, follow the instructions on the screen to install a generic plug-and-play monitor.

Refer to the appropriate following section for driver installation instructions.

Installing the Serial Touch Driver

Installing the Serial Touch Driver for Windows XP, Windows 2000, Me, 95/98 and NT 4.0

NOTE: For Windows 2000 and NT 4.0 you must have administrator access rights to install the driver.

1. Insert the Elo CD-ROM in your computer's CD-ROM drive.
2. If the AutoStart feature for your CD-ROM drive is active, the system automatically detects the CD and starts the setup program.
3. Follow the directions on the screen to complete the driver setup for your version of Windows.
4. If the AutoStart feature is not active:
5. Click *Start* > *Run*.
6. Click the *Browse* button to locate the EloCd.exe program on the CD-ROM.
7. Click *Open*, then *OK* to run EloCd.exe.
8. Follow the directions on the screen to complete the driver setup for your version of Windows.

Installing the Serial Touch Driver for MS-DOS and Windows 3.1

You must have a DOS mouse driver (MOUSE.COM) installed for your mouse if you wish to continue using your mouse along with your touchmonitor in DOS.

To install Windows 3.***x*** and MS-DOS from Windows 95/98, follow the directions below:

1. Insert the Elo CD-ROM in your computer's CD-ROM drive.
2. From DOS, type d:\EloDos_W31 to change to the correct directory on the CD-ROM (your CD-ROM drive may be mapped to a different drive letter).
3. Type install and press *Enter* to start the installation.
4. Align the touchscreen.

You must have already completed Steps 1 and 2 before proceeding. Refer to Chapter 2 of the Elo DOS and Windows Driver Guide as necessary for additional installation information.

To run the INSTALL program:

1. Type INSTALL at the DOS prompt in the directory containing the driver install files.
2. INSTALL asks you to select the software to install. Then choose d:\EloDos_W31 from the displayed list.
3. INSTALL also asks you for the paths to use during installation, or you may use its defaults. INSTALL creates directories as necessary, and warns you if they exist.

If you are updating your software, you may wish to specify the paths containing the earlier versions, and overwrite the obsolete files. All executable programs are upward compatible. For a list of differences from each previous version of the drivers, be sure to select "Differences from Previous Versions" during the installation process.

INSTALL updates your AUTOEXEC.BAT file with the drivers you select. INSTALL makes a copy of your original AUTOEXEC.BAT file, called AUTOEXEC.OLD. If you already have Elo driver commands in your AUTOEXEC.BAT file, they will be commented out.

When INSTALL is finished, it leaves a file called GO.BAT in the subdirectory you specified. GO loads the touchscreen driver, runs the calibration program ELOCALIB, and gives you some final instructions.

If you are using Windows 3.1, you will also calibrate the touchscreen within Windows 3.1 with the Touchscreen Control Panel.

Installing the USB Touch Driver

Installing the USB Touch Driver for Windows XP, Windows 2000, Me and 98

1. Insert the Elo CD-ROM in your computer's CD-ROM drive.

   If Windows XP, Windows 2000,Windows 98, or Windows Me starts the Add New Hardware Wizard:

2. Choose *Next*. Select "Search for the best driver for your device (Recommended)" and choose *Next*.
3. When a list of search locations is displayed, place a checkmark on "Specify a location" and use *Browse* to select the \EloUSB directory on the Elo CD-ROM.
4. Choose *Next*. Once the Elo TouchSystems USB touchscreen driver has been detected, choose *Next* again.
5. You will see several files being copied. Insert your Windows 98 CD if prompted. Choose *Finish*.

If Windows XP, Windows 2000,Windows 98, or Windows Me does not start the Add New Hardware Wizard:

NOTE: For Windows XP and Windows 2000 you must have administrator access rights to install the driver.

1. Insert the Elo CD-ROM in your computer's CD-ROM drive.

   If the AutoStart feature for your CD-ROM drive is active, the system automatically detects the CD and starts the setup program.

2. Follow the directions on the screen to complete the driver setup for your version of Windows.

If the AutoStart feature is not active:

1. Click *Start* > *Run*.
2. Click the *Browse* button to locate the EloCd.exe program on the CD-ROM.
3. Click *Open*, then *OK* to run EloCd.exe.
4. Follow the directions on the screen to complete the driver setup for your version of Windows.

CHAPTER

3

OPERATION

About Touchmonitor Adjustments

Your touchmonitor will unlikely require adjustment. Variations in video output and application may require adjustments to your touchmonitor to optimize the quality of the display.

For best performance, your touchmonitor should be operating in native resolution, that is 800x600 at 60-75 Hz. Use the Display control panel in Windows to choose 800x600 resolution.

Operating in other resolutions will degrade video performance. For further information, please refer to Appendix A.

All adjustments you make to the controls are automatically memorized. This feature saves you from having to reset your choices every time you unplug or power your touchmonitor off and on. If there is a power failure your touchmonitor settings will not default to the factory specifications.

To restore factory set up, choose it from the OSD. See page 3-35.

	Control	*Function*
1	*Power Switch*	Turns the display system power *on* or *off*.
2	*Select*	Displays the OSD menus on the screen and used to select (“Up” and “Down” direction) the OSD control options on the screen.
3	◀	(1) Adjusts the decreasing value of the selected OSD control option.
4	▶	(1) Adjusts the increasing value of the selected OSD control option. (2) Enables/disables the mute and volume option.
5	*Menu*	Menu display and menu exit.

Controls and Adjustment

OSD Lock/Unlock

You are able to lock and unlock the OSD feature. The monitor is shipped in the unlocked position.

To lock the OSD:

1. Press the Menu button and ▶ button simultaneously for 2 seconds. A window will appear displaying “OSD Unlock”. Continue to hold the buttons down for another 2 seconds and the window toggles to “OSD Lock”.

Power Lock/Unlock

You are able to lock/unlock the Power feature. The monitor is shipped in the unlocked position.

To lock the power:

1. Press the Menu button and the ◀ simultaneously for 2 seconds. A window will appear displaying “Power Unlock”. Continue holding the buttons down for another 2 seconds and the window toggles to “Power Lock”.

OSD Menu Functions

To display the OSD Menu press the *Menu* button.

1. Press the ▶ button to select the different OSD control option.
2. When the function you want to change is displayed, press the *Select* button.

To adjust the Value of the function:

1. Pressing the ▶ button increases the value of the selected OSD control option.
2. Pressing the ◀ button decreases the value of the selected OSD control option.

After adjusting the values, the monitor will automatically save the changes.

NOTE: The OSD screen will disappear if no input activities are detected for 10 seconds.

OSD Control Options

Brightness

- Background Luminance of the LCD panel is adjusted.

Contrast

- Adjusts the contrast or the values of color gain (RED, GREEN or BLUE).

Saturation, Hue, Flesh Tones

- Adjusts the color intensity and tint so faces appear natural.

Phase

- Adjusts the phase of the dot clock.

Auto Adjust

- Clock system auto adjustment (under 5 seconds).

OSD Left/Right

- The OSD screen is moved vertically right and left.

OSD Up/Down

- The OSD screen is moved vertically up and down.

Clock

- Adjusts the ratio of dividing frequency of the dot clock.

Color Temperature

- Sets R, G, B gain.

Current Input

- The frequency of the horizontal/vertical synchronizing signal under the input is indicated.

OSD Position

- Allows the OSD indication position to be selected.

Language

- Languages used for OSD menu display: English, French, German, Spanish and Japanese.

Recall Defaults

- Recalls the factory OSD default settings.

OSD Timeout

- Adjusts the amount of time in which the OSD will disappear.

Power-Save (No Input)

- The LCD panel background is cut when there is no signal input (AC line power consumption of 5w or less).

Power LED Display & Power Saving

General Power Saving Mode

When the power switch are switch *on*, this LED lights in *green*.

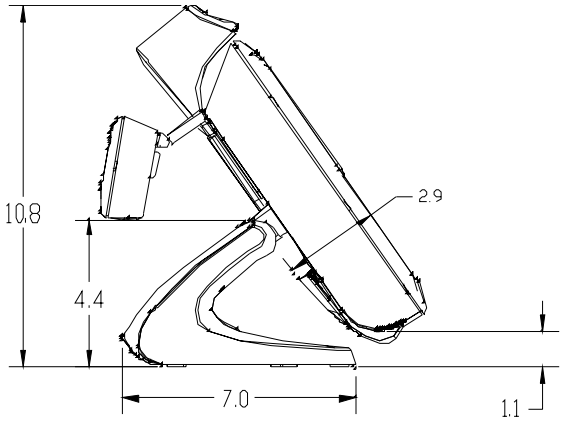
The LED indicates the different power status with altered LED colors when monitor operates in different modes (see following table).

Mode	*Power Consumption*	*H-Sync.*	*V-Sync.*	*Indicator*
On	18W max.	Pulses	Pulses	Green
Standby	8W max.	No	Pulses	Orange
Suspend	8W max.	Pulses	No	Orange
Off	8W max.	No	No	Yellow

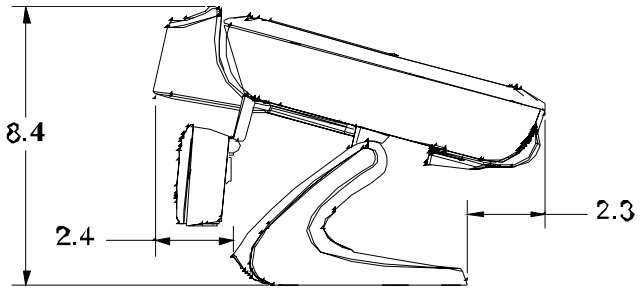
We recommend switching the monitor off when it is not in use for a long period of time.

Display Angle

For viewing clarity, you can tilt the LCD forward (up to -5 degrees) or backward (up to 60 degrees.)



Tilt 35° from vertical



Tilt 75° from vertical

CAUTION In order to protect the LCD, be sure to hold the base when adjusting the LCD, and take care **not** to touch the screen.

CHAPTER

4

TROUBLESHOOTING

If you are experiencing trouble with your touchmonitor, refer to the following table. If the problem persists, please contact your local dealer or our service center. Elo Technical Support numbers are listed on the last page of this manual.

Solutions to Common Problems

Problem	*Suggestion(s)*
The monitor does not respond after you turn on the system.	Check that the monitor's Power Switch is on.
	Turn off the power and check the monitor's power cord and signal cable for proper connection.
Characters on the screen are dim	Refer to the ***Controls and Adjustments*** section to adjust the brightness.
The screen is blank	During operation, the monitor screen may automatically turn off as a result of the Power Saving feature. Press any key to see if the screen reappears.
	Refer to the ***Controls and Adjustments*** section to adjust the brightness.
OSD or power buttons don't work	Check to see that they are not locked out. See page 3-33.
"Out of Range" display	check to see of the resolution of your computer is higher than that of the LCD display.
	Reconfigure the resolution of your computer to make it less than or equal to 1024x768. 800x600 is optimal. See Appendix A for more information on resolution.
Touch doesn't work	Make sure cable is securely attached at both ends.

APPENDIX

A

NATIVE RESOLUTION

The native resolution of a monitor is the resolution level at which the LCD panel is designed to perform best. For the Elo LCD touchmonitor, the native resolution is 800 x 600 for the 12.1 inch size. In almost all cases, screen images look best when viewed at their native resolution. You can lower the resolution setting of a monitor but not increase it.

Input Video	*12.1" LCD*
640x480 (VGA)	Transforms input format to 800x600
800x600 (SVGA)	Displays in Native Resolution

The native resolution of an LCD is the actual number of pixels horizontally in the LCD by the number of pixels vertically in the LCD. LCD resolution is usually represented by the following symbols:

VGA	*640x480*
SVGA	*800x600*
XGA	*1024x768*

As an example, a SVGA resolution LCD panel has 800 pixels horizontally by 600 pixels vertically. Input video is also represented by the same terms. XGA input video has a format of 1024 pixels horizontally by 768 pixels vertically. When the input pixels contained in the video input format match the native resolution of the panel, there is a one to one correspondence of mapping of input video pixels to LCD pixels. As an example, the pixel in column 45 and row 26 of the input video is in column 45 and row 26 of the LCD. For the case when the input video is at a lower or higher resolution than the native resolution of the LCD, the direct correspondence between the video pixels and the LCD pixels is lost. The LCD controller can compute the correspondence between video pixels and LCD pixels using algorithms contained on its controller. The accuracy of the algorithms determines the fidelity of conversion of video pixels to LCD pixels. Poor fidelity conversion can result in artifacts in the LCD displayed image such as varying width characters.

APPENDIX

B

TOUCHMONITOR SAFETY

This manual contains information that is important for the proper setup and maintenance of your touchmonitor. Before setting up and powering on your new touchmonitor, read through this manual, especially Chapter 2 (Installation), and Chapter 3 (Operation).

1. To reduce the risk of electric shock, follow all safety notices and never open the touchmonitor case.
2. Turn off the product before cleaning
3. Your new touchmonitor is equipped with a 3-wire, grounding power cord. The power cord plug will only fit into a grounded outlet. Do not attempt to fit the plug into an outlet that has not been configured for this purpose. Do not use a damaged power cord. Use only the power cord that comes with your Elo TouchSystems Touchmonitor. Use of an unauthorized power cord may invalidate your warranty.
4. The slots located on the sides and top of the touchmonitor case are for ventilation. Do not block or insert anything inside the ventilation slots.
5. It is important that your touchmonitor remains dry. Do not pour liquid into or onto your touchmonitor. If your touchmonitor becomes wet do not attempt to repair it yourself.

Care and Handling of Your Touchmonitor

The following tips will help keep your Elo Entuitive touchmonitor functioning at the optimal level.

- To avoid risk of electric shock, do not disassemble the brick supply or display unit cabinet. The unit is not user serviceable. Remember to unplug the display unit from the power outlet before cleaning.
- Do not use alcohol (methyl, ethyl or isopropyl) or any strong dissolvent. Do not use thinner or benzene, abrasive cleaners or compressed air.
- To clean the display unit cabinet, use a cloth lightly dampened with a mild detergent.
- Avoid getting liquids inside your touchmonitor. If liquid does get inside, have a qualified service technician check it before you power it on again.
- Do not wipe the screen with a cloth or sponge that could scratch the surface.
- To clean the touchscreen, use window or glass cleaner. Put the cleaner on the rag and wipe the touchscreen. ***Never*** apply the cleaner directly on the touchscreen

APPENDIX

C

TECHNICAL SPECIFICATIONS

Display Modes

Your Elo Entuitive touchmonitor is compatible with the following standard video modes:

Item	*Resolution*	*Type*	*H. Scan(KHz)*	*V. Scan(Hz)*	*Pol.*
1	640X350	VGA	31.469	70.087	+ / -
2	640X350	VESA85	37.861	85.080	+ / -
3	720X400	VGA	31.469	70.087	- / +
4	720X400	VESA85	37.927	85.039	- / +
5	640X480	VGA	31.469	59.940	- / -
6	640X480	VESA72	37.861	72.809	- / -
7	640X480	VESA75	37.500	75.000	- / -
8	640X480	VESA85	43.269	85.008	- / -
9	800X600	SVGA	35.156	56.250	+ / +
10	800X600	SVGA	37.879	60.317	+ / +
11	800X600	VESA72	48.077	72.188	+ / +
12	800X600	VESA75	46.875	75.000	+ / +
13	800X600	SVGA	53.674	85.061	+ / +

Touchmonitor Specifications

Model		*ET1229L*
LCD Display		12.1” TFT Active Matrix Panel
Display Size		246(H) x 184.5(V) mm
Pixel Pitch		0.3075(H) x 0.3075(V) mm
Display Mode		VGA 640 x 350 (70 / 85Hz)
		VGA 720 x 400 (70 / 85Hz)
		VGA 640 x 480 (60 / 72 / 75 / 85Hz)
		SVGA 800 x 600 (56 / 60 / 72 / 75Hz)
Native		SVGA 800 x 600
Max. Resolution		SVGA 1024 x 768
Contrast Ratio		300 : 1 (typical)
Brightness		300 Cd/m^2 with AT 246 cd/m^2, IT 276 cd/m^2, IR 276 cd/m^2
Response Time		Tr= 20 msec, Tf= 30 mesc; 30 ms (min.) / 100 ms (max.)
Display Color		262K
Viewing Angle		(L/R)= $-60^o/+60^o$ (typical), (U/D) $+50^o/-40^o$ (typical); 60^o or 120^o total (typical), $+50^o/-40^o$ or 90^o total (typical)
Input Signal	Video	R.G.B. Analog 0.7V $_{\text{peak to peak}}$
	Sync	TTL Positive or Negative
Signal Connector		15 Pin Mini D-Sub
Front Control		Power on / off with LED, Menu / Select (up, down), Adjustment (+, -)
OSD		Brightness, Contrast RGB Contrast, Saturation, Hue, Auto Setup, H/V-Position, Frequency, Phase, Track, Text/Graphics, Expansion, Mode Inf., Rom/Ram Ver., Recall
Plug & Play		DDC1 / 2B
Touch Panel (optional)		AccuTouch, IntelliTouch and CarrollTouch
Power Adapter		Input AC 100-240V, 50-60Hz, Output DC 12V/2.5A (max.)
Operating Conditions	Temp	0^oC ~ 40^oC (41^oF ~ 95^oF)
	Humidity	20% ~ 80% (No Condensation)
	Altitude	To 12,000 Feet
Dimensions (HxWxD)		310 x 312 x 146mm
Weight (Net)		19.84lbs., monitor weight 11.90 lbs.
Certifications		UL, C-UL, FCC-A, CE, TUV-GS, VCCI, MPRII, C-TICK

AccuTouch Touchscreen Specifications

Mechanical	
Construction	Top: Polyester with outside hard-surface coating with clear or antiglare finish.
	Inside: Transparent conductive coating.
	Bottom: Glass substrate with uniform resistive coating. Top and bottom layers separated by Elo-patented separator dots.
Positional Accuracy	Standard deviation of error is less than 0.080 in. (2.03 mm). This equates to less than ±1%.
Touchpoint Density	More than 100,000 touchpoints/in² (15,500 touchpoints/cm²).
Touch Activation Force	Typically less than 4 ounces (113 grams).
Surface Durability	Meets Taber Abrasion Test (ASTM D1044), CS-10F wheel, 500 g. Meets pencil hardness 3H.
Expected Life Performance	AccuTouch technology has been operationally tested to greater than 35 million touches in one location without failure, using a stylus similar to a finger.
Optical	
Light Transmission (per ASTM D1003)	Typically 85% at 550-nm wavelength (visible light spectrum).
Visual Resolution	All measurements made using USAF 1951 Resolution Chart, under 30 X magnification, with test unit located approximately 1.5 in. (38 mm) from surface of resolution chart.
	Antiglare surface: 6:1 minimum.
Haze (per ASTM D1003)	Antiglare surface: Less than 15%.
Gloss (per ASTM D2457)	Antiglare surface: 90 ± 20 gloss units tested on a hard-coated front surface.

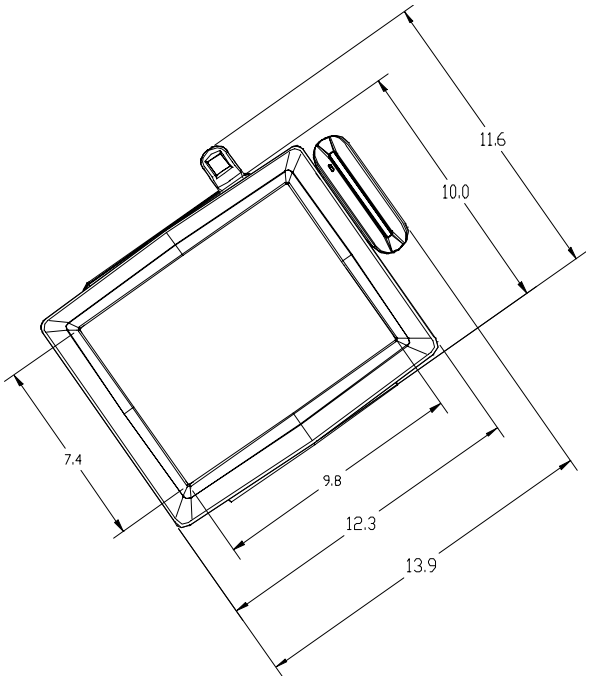
IntelliTouch Touchscreen Specifications

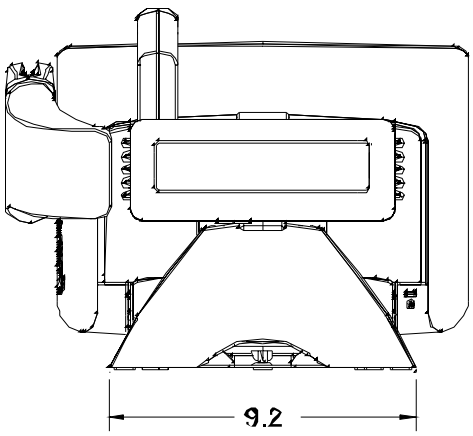
Mechanical	
Positional Accuracy	Standard deviation of error is less than 0.080 in. (2.03 mm). Equates to less than ±1%.
Touchpoint Density	More than 100,000 touchpoints/in^2 (15,500 touchpoints/cm^2).
Touch Activation Force	Typically less than 3 ounces (85 grams).
Surface Durability	Surface durability is that of glass, Mohs' hardness rating of 7.
Expected Life Performance	No known wear-out mechanism, as there are no layers, coatings, or moving parts. IntelliTouch technology has been operationally tested to more than 50 million touches in one location without failure, using a stylus similar to a finger.
Sealing	Unit is sealed to protect against splashed liquids, dirt, and dust.
Optical	
Light Transmission (per ASTM D1003)	90%
Visual Resolution	All measurements made using USAF 1951 Resolution Chart, under 30X magnification, with test unit located approximately 1.5 in (38 mm) from surface of resolution chart. Clear surface: Excellent, with no noticeable degradation. Antiglare surface: 6:1 minimum.
Gloss (per ASTM D2457 using a 60-degree gloss meter)	Antiglare surface: Curved: 60 ± 20 gloss units or 75 ± 15 gloss units.
Environmental	
Chemical Resistance	The active area of the touchscreen is resistant to all chemicals that do not affect glass, such as: Acetone Toluene Methyl ethyl ketone Isopropyl alcohol Methyl alcohol Ethyl acetate Ammonia-based glass cleaners Gasoline Kerosene Vinegar
Electrostatic Protection (per EN 61 000-4-2, 1995)	Meets Level 4 (15 kV air/8 kV contact discharges).

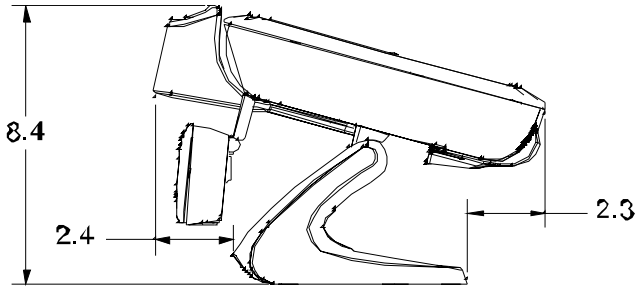
Infrared Touchscreen Specifications

Mechanical	
Input Method	Input Method Finger or gloved hand activation
Electrical	
Positional Accuracy	Typical centroid accuracy: 2 mm with 1 mm STD error
Resolution	Touchpoint density is based on controller resolution of 4096 x 4096
Touch Activation Force	No minimum touch activation force is required
Controller	Board: Serial (RS232) or USB 1.1
Optical	
Light Transmission	Glass overlay: 90% per ASTM D1003-92
Environmental	
Chemical Resistance	Glass overlays: The touch active area of the touchscreen is resistant to chemicals that do not affect glass, such as: acetone, toluene, methyl ethyl ketone, isopropyl alcohol, methyl alcohol, ethyl acetate, ammonia-based glass cleaners, gasoline, kerosene, vinegar. Polycarbonate bezel: around perimeter of display has some sensitivity to hydrocarbons.
Durability	
Surface Durability	Glass filter option: Surface durability is that of glass, Mohs' hardness rating of 7.

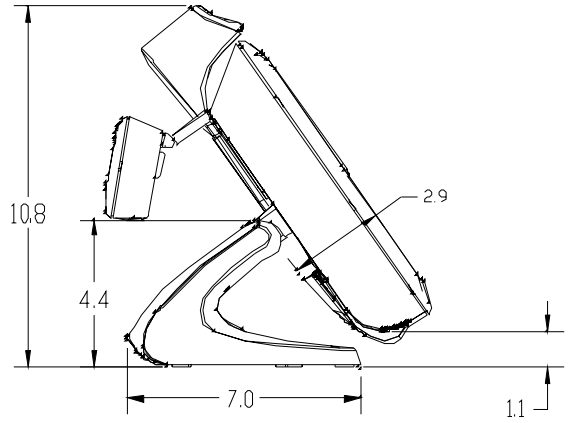
12.1" LCD Touchmonitor (ET1229L-XXWA-1) Dimensions



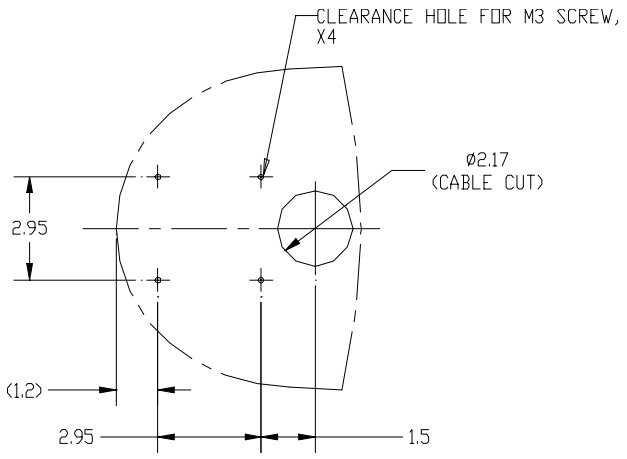




Tilt 75° from vertical



Tilt 35° from vertical



BASE MOUNTING PATTERN

REGULATORY INFORMATION

I. Electrical Safety Information:

A) Compliance is required with respect to the voltage, frequency, and current requirements indicated on the manufacturer's label. Connection to a different power source than those specified herein will likely result in improper operation, damage to the equipment or pose a fire hazard if the limitations are not followed.

B) There are no operator serviceable parts inside this equipment. There are hazardous voltages generated by this equipment which constitute a safety hazard. Service should be provided only by a qualified service technician.

C) This equipment is provided with a detachable power cord which has an integral safety ground wire intended for connection to a grounded safety outlet.

1) Do not substitute the cord with other than the provided approved type. Under no circumstances use an adapter plug to connect to a 2-wire outlet as this will defeat the continuity of the grounding wire.

2) The equipment requires the use of the ground wire as a part of the safety certification, modification or misuse can provide a shock hazard that can result in serious injury or death.

3) Contact a qualified electrician or the manufacturer if there are questions about the installation prior to connecting the equipment to mains power.

II. Emissions and Immunity Information

A) Notice to Users in the United States: This equipment has been tested and found to comply with the limits for a Class B digital device, pursuant to Part 15 of FCC Rules. These limits are designed to provide reasonable protection against harmful interference in a residential installation. This equipment generates, uses, and can radiate radio frequency energy, and if not installed and used in accordance with the instructions, may cause harmful interference to radio communications.

B) Notice to Users in Canada: This equipment complies with the Class B limits for radio noise emissions from digital apparatus as established by the Radio Interference Regulations of Industrie Canada.

C) Notice to Users in the European Union: Use only the provided power cords and interconnecting cabling provided with the equipment. Substitution of provided cords and cabling may compromise electrical safety or CE Mark Certification for emissions or immunity as required by the following standards:

This Information Technology Equipment (ITE) is required to have a CE Mark on the manufacturer's label which means that the equipment has been tested to the following Directives and Standards:

This equipment has been tested to the requirements for the CE Mark as required by EMC Directive 89/336/EEC indicated in European Standard EN 55 022 Class B and the Low Voltage Directive 73/23/EEC as indicated in European Standard EN 60 950.

D) General Information to all Users: This equipment generates, uses and can radiate radio frequency energy. If not installed and used according to this manual the equipment may cause interference with radio and television communications. There is, however, no guarantee that interference will not occur in any particular installation due to site-specific factors.

1) In order to meet emission and immunity requirements, the user must observe the following:

a) Use only the provided I/O cables to connect this digital device with any computer.

b) To ensure compliance, use only the provided manufacturer's approved line cord.

c) The user is cautioned that changes or modifications to the equipment not expressly approved by the party responsible for compliance could void the user's authority to operate the equipment.

2) If this equipment appears to cause interference with radio or television reception, or any other device:

a) Verify as an emission source by turning the equipment off and on.

b) If you determine that this equipment is causing the interference, try to correct the interference by using one or more of the following measures:

i) Move the digital device away from the affected receiver.

ii) Reposition (turn) the digital device with respect to the affected receiver.

iii) Reorient the affected receiver's antenna.

iv) Plug the digital device into a different AC outlet so the digital device and the receiver are on different branch circuits.

v) Disconnect and remove any I/O cables that the digital device does not use. (Unterminated I/O cables are a potential source of high RF emission levels.)

vi) Plug the digital device into only a grounded outlet receptacle. Do not use AC adapter plugs. (Removing or cutting the line cord ground may increase RF emission levels and may also present a lethal shock hazard to the user.)

If you need additional help, consult your dealer, manufacturer, or an experienced radio or television technician.






MPRII

This Class B digital apparatus complies with Canadian ICES-003.
Cet appareil numérique de la classe B est conforme à la norme NMB-003 du Canada.

WARRANTY

Except as otherwise stated herein or in an order acknowledgment delivered to Buyer, Seller warrants to Buyer that the Product shall be free of defects in materials and workmanship. With the exception of the negotiated warranty periods; the warranty for the touchmonitor and components of the product is 2 years.

Seller makes no warranty regarding the model life of components. Seller's suppliers may at any time and from time to time make changes in the components delivered as Products or components.

Buyer shall notify Seller in writing promptly (and in no case later than thirty (30) days after discovery) of the failure of any Product to conform to the warranty set forth above; shall describe in commercially reasonable detail in such notice the symptoms associated with such failure; and shall provide to Seller the opportunity to inspect such Products as installed, if possible. The notice must be received by Seller during the Warranty Period for such product, unless otherwise directed in writing by the Seller. Within thirty (30) days after submitting such notice, Buyer shall package the allegedly defective Product in its original shipping carton(s) or a functional equivalent and shall ship to Seller at Buyer's expense and risk.

Within a reasonable time after receipt of the allegedly defective Product and verification by Seller that the Product fails to meet the warranty set forth above, Seller shall correct such failure by, at Seller's options, either (i) modifying or repairing the Product or (ii) replacing the Product. Such modification, repair, or replacement and the return shipment of the Product with minimum insurance to Buyer shall be at Seller's expense. Buyer shall bear the risk of loss or damage in transit, and may insure the Product. Buyer shall reimburse Seller for transportation cost incurred for Product returned but not found by Seller to be defective. Modification or repair, of Products may, at Seller's option, take place either at Seller's facilities or at Buyer's premises. If Seller is unable to modify, repair, or replace a Product to conform to the warranty set forth above, then Seller shall, at Seller's option, either refund to Buyer or credit to Buyer's account the purchase price of the Product less depreciation calculated on a straight-line basis over Seller's stated Warranty Period.

THESE REMEDIES SHALL BE THE BUYER'S EXCLUSIVE REMEDIES FOR BREACH OF WARRANTY. EXCEPT FOR THE EXPRESS WARRANTY SET FORTH ABOVE, SELLER GRANTS NO OTHER WARRANTIES, EXPRESS OR IMPLIED BY STATUTE OR OTHERWISE, REGARDING THE PRODUCTS, THEIR FITNESS FOR ANY PURPOSE, THEIR QUALITY, THEIR MERCHANTABILITY, THEIR NONINFRINGEMENT, OR OTHERWISE. NO EMPLOYEE OF SELLER OR ANY OTHER PARTY IS AUTHORIZED TO MAKE ANY WARRANTY FOR THE GOODS OTHER THAN THE WARRANTY SET FORTH HEREIN. SELLER'S LIABILITY UNDER THE WARRANTY SHALL BE LIMITED TO A REFUND OF THE PURCHASE PRICE OF THE PRODUCT. IN NO EVENT SHALL SELLER BE LIABLE FOR THE COST OF PROCUREMENT OR INSTALLATION OF SUBSTITUTE GOODS BY BUYER OR FOR ANY SPECIAL, CONSEQUENTIAL, INDIRECT, OR INCIDENTAL DAMAGES.

Buyer assumes the risk and agrees to indemnify Seller against and hold Seller harmless from all liability relating to (i) assessing the suitability for Buyer's intended use of the Products and of any system design or drawing and (ii) determining the compliance of Buyer's use of the Products with applicable laws, regulations, codes, and standards. Buyer retains and accepts full responsibility for all warranty and other claims relating to or arising from Buyer's products, which include or incorporate Products or components manufactured or supplied by Seller. Buyer is solely responsible for any and all representations and warranties regarding the Products made or authorized by Buyer. Buyer will indemnify Seller and hold Seller harmless from any liability, claims, loss, cost, or expenses (including reasonable attorney's fees) attributable to Buyer's products or representations or warranties concerning same.

INDEX

USB (UNIVERSAL SERIAL BUS) KEYBOARD EMULATION SWIPE READER

TECHNICAL REFERENCE MANUAL

Manual Part Number 99875206 Rev 6

JUNE 2003

MAGTEK

REGISTERED TO ISO 9001:2000

20725 South Annalee Avenue
Carson, CA 90746
Phone: (310) 631-8602
FAX: (310) 631-3956
Technical Support: (651) 415-6800
www.magtek.com

REVISIONS

Rev Number	Date	Notes
1	28 Aug 01	Initial Release
2	11 Sep 01	Sec 1: Removed Hardware Configurations P/Ns 210- 40108, 40110, 40111, 40112.
3	05 Oct 01	Removed HID from Title Page
4	12 Dec 02	Section 4, Command Number: Corrected GET and SET PROPERTY descriptions
5	28 Jan 03	Changed copyright symbol so pdf copies would print on all printers
6	03 Jun 03	Front Matter: added ISO line to logo, changed Tech Support phone number, added new warranty statement.

Limited Warranty

MagTek, Inc. warrants that the Product described in this document is free of defects in materials and workmanship for a period of one year from the date of purchase where the date of purchase is defined as the date of shipment from MagTek. During this warranty period, MagTek shall, at their option, repair or replace without charge for either parts or labor, any failure, malfunction, defect or nonconformity which prevents the product from performing in accordance with MagTek's published technical specifications and manuals.

This warranty does not apply to wear of the magnetic read head. This warranty shall not apply if the product is modified, tampered with, or subject to abnormal working conditions. This warranty does not apply when the malfunction results from the use of the Product in conjunction with ancillary or peripheral equipment where it is determined by MagTek that there is no fault in the Product itself.

Notification by the Customer to MagTek of any condition described above should be directed to the Customer's MagTek Sales Representative or to MagTek's Help Desk at (651) 415-6800. If the Product is to be returned from the Customer to MagTek, a returned material authorization (RMA) will be issued by MagTek. The Customer shall be responsible for shipping charges to MagTek, (20801 S. Annalee Ave., Carson, CA 90746). MagTek shall be responsible for shipping charges back to the Customer.

Repair or replacement as provided under this warranty is the exclusive remedy. This warranty is in lieu of all other warranties, express or implied.

FCC WARNING STATEMENT

This equipment has been tested and found to comply with the limits for Class B digital device, pursuant to Part 15 of FCC Rules. These limits are designed to provide reasonable protection against harmful interference when the equipment is operated in a residential environment. This equipment generates, uses, and can radiate radio frequency energy and, if not installed and used in accordance with the instruction manual, may cause harmful interference to radio communications. However, there is no guarantee that interference will not occur in a particular installation.

FCC COMPLIANCE STATEMENT

This device complies with Part 15 of the FCC Rules. Operation of this device is subject to the following two conditions: (1) This device may not cause harmful interference; and (2) this device must accept any interference received, including interference that may cause undesired operation.

CANADIAN DOC STATEMENT

This digital apparatus does not exceed the Class B limits for radio noise for digital apparatus set out in the Radio Interference Regulations of the Canadian Department of Communications.

Le présent appareil numérique n'émet pas de bruits radioélectriques dépassant les limites applicables aux appareils numériques de las classe B prescrites dans le Réglement sur le brouillage radioélectrique édicté par les ministère des Communications du Canada.

CE STANDARDS

Testing for compliance to CE requirements was performed by an independent laboratory. The unit under test was found compliant to Class B.

UL/CSA

This product is recognized per Underwriter Laboratories and Canadian Underwriter Laboratories 1950.

TABLE OF CONTENTS

FIGURES

TABLES

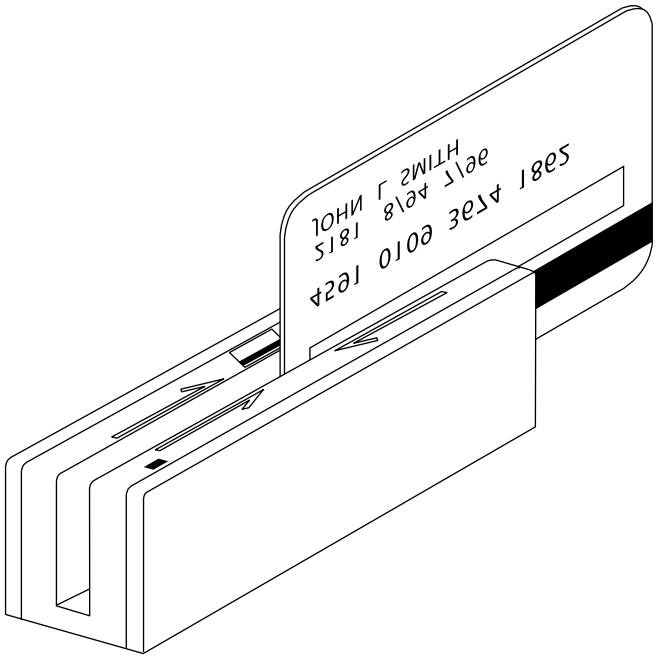


Figure 1-1. USB Swipe Reader

SECTION 1. FEATURES AND SPECIFICATIONS

The USB (Universal Serial Bus), HID Keyboard Emulation, Swipe Reader is a compact magnetic stripe card reader, which conforms to ISO standards. The Reader is compatible with the PC series of personal computers or any device with a USB interface. A card is read by sliding it, stripe down and facing the LED side, through the slot either forward or backward.

A LED (Light Emitting Diode) indicator on the Reader panel provides the operator with continuous status of the Reader operations.

The reader emulates a USB Human Interface Device (HID) keyboard. This allows host applications designed to acquire card data from keyboard input to seamlessly acquire the card data from the USB swipe reader.

Caution

If another keyboard is connected to the same host as this device and a key is pressed on the other keyboard while this device is transmitting, then the data transmitted by this device may get corrupted.

Because of potential "data interleave" issues associated with the USB Keyboard interface, MagTek recommends that the USB Keyboard Emulation MSR product should only be used by customers who have previously used MagTek's Keyboard Wedge MSR, or who are interfacing with an existing PC software application which gathers card data from the keyboard port. If previous applications were based upon RS-232 serial interface MSR's, or if this is a brand new development effort, it is strongly recommended that you use the MagTek's "standard version" of the USB MSR (Non-Keyboard Emulation Version). Please refer to Technical Manual 99875191 for further information regarding the "standard version" USB MSR.

FEATURES

Major features of the Swipe Reader are as follows:

- Powered through the USB – no external power supply required
- Hardware Compatible with PC or any computer or terminal with a USB interface
- Bidirectional card reading
- Reads encoded data that meets ANSI/ISO/CDL/AAMVA standards and others such as ISO track 1 format on track 2 or 3.
- Reads up to three tracks of card data
- LED for status
- Compatible with USB specification Revision 1.1
- Compatible with HID specification Version 1.1
- Can use standard Windows drivers for communications. No third part device driver is required.

- Many programmable configuration options
- Non-volatile flash EEPROM memory for configuration storage
- Built-in 6 foot USB cable

HARDWARE CONFIGURATIONS

The hardware configurations are as follows:

Part Number	Tracks	Color
21040107	TK 1,2,3	Pearl White
21040109	TK 1,2	Pearl White

ACCESSORIES

The accessories are as follows:

Part Number	Description
21042806	USB MSR Demo Program with Source Code (Diskette)
99510026	USB MSR Demo Program with Source Code (WEB)

REFERENCE DOCUMENTS

Axelson, Jan. *USB Complete, Everything You Need to Develop Custom USB Peripherals*, 1999. Lakeview Research, 2209 Winnebago St., Madison WI 53704, 396pp., *http://www.lvr.com*.

USB Human Interface Device (HID) Class Specification Version 1.1.

USB (Universal Serial Bus) Specification, Version 1.1, Copyright 1998 by Compaq Computer Corporation, Intel Corporation, Microsoft Corporation, NEC Corporation.

USB Implementers Forum, Inc*., www.usb.org*.

SPECIFICATIONS

Table 1-2 lists the specifications for the USB Swipe Reader. Figure 1-2 shows the dimensions for the standard product. Other sizes are available by special order.

Table 1-2. Specifications

Reference Standards	ISO 7810 and ISO 7811/CDL/ AAMVA*
Power Input	5V From USB port
Recording Method	Two-frequency coherent phase (F2F)
Message Format	ASCII
Card Speed	3 to 50 IPS
MTBF	Electronics: 125,000 hours. Head: 1,000,000 passes
ELECTRICAL	
Current Normal Mode Suspend Mode	 30mA 300uA
MECHANICAL (STANDARD PRODUCT)	
Weight	4.5 oz. (127.57 g)
Cable length	6ft.
Connector	USB Type A plug
ENVIRONMENTAL	
Temperature	
Operating	32°F to 131°F (0°C to 55°C)
Storage	-22°F to 158°F (-30°C to 70°C)
Humidity	
Operating	10% to 90% noncondensing
Storage	Up to 100% noncondensing
Altitude	
Operating	0-10,000 ft. (0-3048 m.)
Storage	0-50,000 ft. (0-15240 m.)

* ISO (International Standards Organization), CDL (California Drivers License), and AAMVA (American Association of Motor Vehicle Administrators).

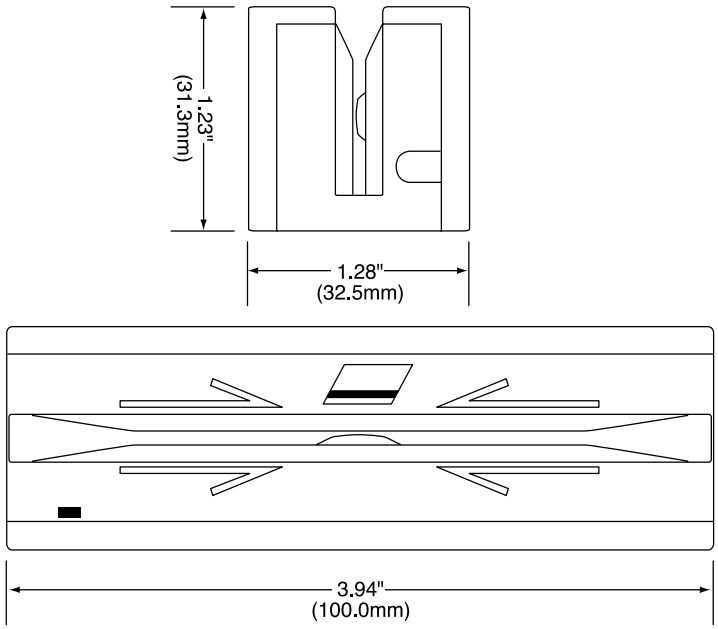


Figure 1-2. Dimensions

SECTION 2. INSTALLATION

This section describes the cable connection, the Windows Plug and Play Setup, and the physical mounting of the unit.

USB CONNECTION

Connect the USB cable to a USB port on the host. The Reader, LED Indicator, and pin numbers for the 4-pin connector are shown in Figure 2-1.

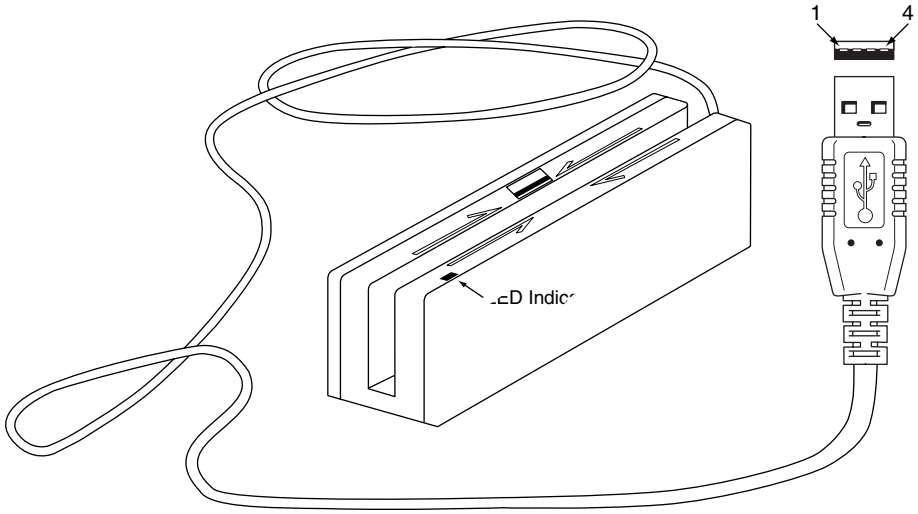


Figure 2-1. Reader Cable and Connector

Pin numbers and signal descriptions for the cable shown in the illustration are listed in Table 1-1.

Table 2-1. 4-Pin Connector

Pin Number	Signal	Cable Color
1	V_{CC}	Red
2	- Data	White
3	+Data	Green
4	Ground	Black

WINDOWS PLUG AND PLAY SETUP

On hosts with the Windows operating system, the first time the device is plugged into a specific USB port, Windows will pop up a dialog box, which will guide you through the process of installing a device driver for the device. After this process is completed once, Windows will no longer request this process as long as the device is plugged into the same USB port. The device driver that Windows will install for this device is the driver used for HID keyboard devices and it is part of the Windows operating system. When the dialog box pops up, follow the instructions given to you in the dialog box. Sometimes Windows will find all the files it needs on its own without giving you any prompts. Other times Windows will need to know the location of the files it needs. If Windows prompts you for the file locations, insert the CD that was used to install Windows on your PC and point Windows to the root directory of the CD. Windows should find all the files it needs there.

MOUNTING

The Reader may be mounted with screws or fastening tape as described below.

Caution

The Reader should be mounted such that the bottom (mounting side) is not exposed to the user. This is because the mounting side of the reader may be susceptible to electrostatic discharge.

1. The Reader can be mounted on a surface in three ways:

   - By two screws through the surface attached to the bottom of the unit and running the cable on the top of the surface;

   - By two screws through the surface attached to the bottom of the unit and by drilling a hole in the surface for the cable and running the cable through the hole;

   - By attaching the unit to the surface with fastening tape and running the cable on the top of the surface.

   Note

   The two mounting inserts are 3 mm diameter; 0.5 mm pitch; 6.4 mm deep. The length of the screws used depends on the mounting surface thickness and the thickness of washers (if used).

   The mounting dimensions are shown in Figure 2-2. Determine the method of mounting required.

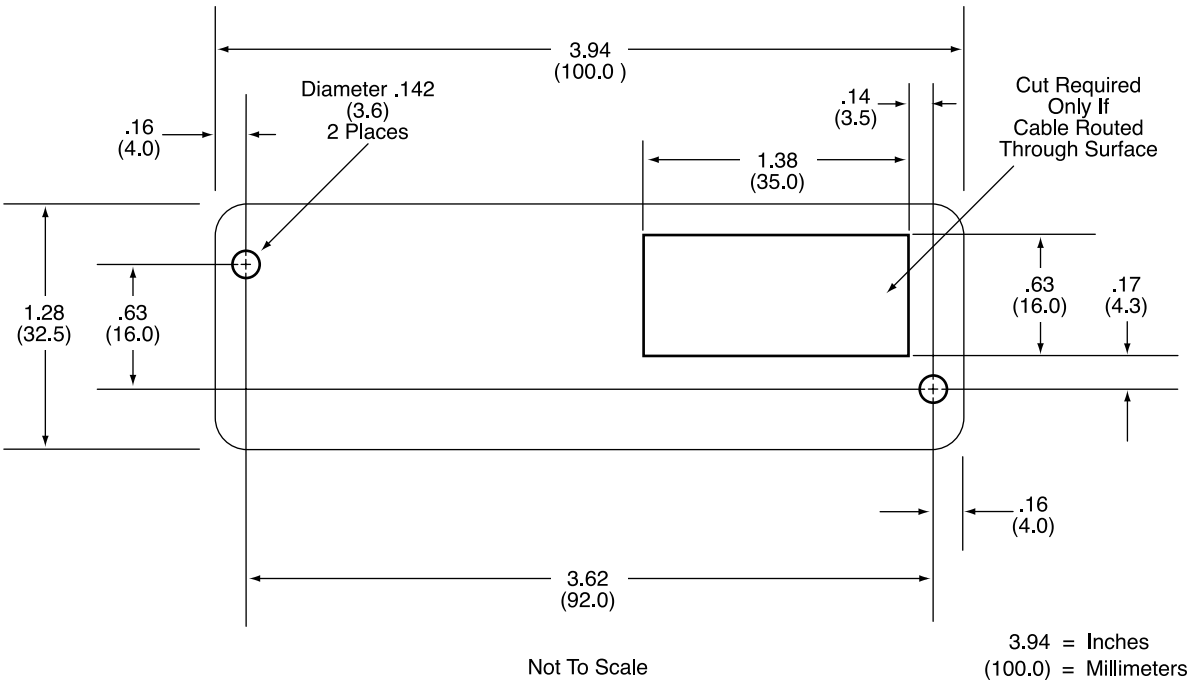


Figure 2-2. Mounting Hole Dimensions For Surface

2. Ensure the Reader is positioned on a flat, accessible surface with at least 4 inches clearance on either end for room to swipe a card. Orient the Reader so the side with the LED is facing the direction of intended use.

   If fastening tape is to be used, clean the area that the Reader will be mounted on with isopropyl alcohol. Remove the adhesive protective cover on the fastening tape, and position the Reader and push down firmly.

3. Mount the Reader.

SECTION 3. OPERATION

This section describes the LED Indicator and Card Read.

LED INDICATOR

The LED indicator will be either off, red, or green. When the device is not powered, the LED will be off. When the device is first plugged in, the LED will be red. As soon as the device is plugged in, the host will try to enumerate the device. Once the device is enumerated the LED will turn green indicating that the device is ready for use. When a card is being swiped, the LED will turn off temporarily until the swipe is completed. If there are no errors decoding the card data then the LED will turn green. If there are any errors decoding the card data, the LED will turn red for approximately two seconds to indicate that an error occurred and then turn green. Anytime the host puts the device into suspend mode, the LED will turn off. Once the host takes the device out of suspend mode, the LED will return to the state it was in prior to entering suspend mode.

CARD READ

A card may be swiped through the Reader slot when the LED is green. The magnetic stripe must face toward the front (the side with the LED) and may be swiped in either direction. If there is data encoded on the card, the device will attempt to decode the data and then send the results to the host as if the data was being typed on a keyboard. After the results are sent to the host, the device will be ready to read the next card.

SECTION 4. USB COMMUNICATIONS

This device conforms to the USB specification revision 1.1. This device also conforms with the Human Interface Device (HID) class specification version 1.1. The device communicates to the host as a HID keyboard device. The latest versions of the Windows operating systems, Windows 98, Me, and 2000, all come with a standard Windows USB HID keyboard driver.

This is a full speed USB device. This device is powered from the USB bus. Its vendor ID is 0x0801 and its product ID is 0x0001. The device will go into suspend mode when directed to do so by the host. The device will wakeup from suspend mode when directed to do so by the host. The device does not support remote wakeup.

HOST APPLICATIONS

This device can be used with existing applications that acquire card data via keyboard input. Also, applications that communicate to this device can be easily developed. These applications can be easily developed using compilers such as Microsoft's Visual Basic or Visual C++. To demonstrate this device's card reading capabilities any application that accepts keyboard input such as Window's Notepad can be used.

CARD DATA

The card data is converted to ASCII and transmitted to the host as if it had been typed on a keyboard. Any data with ASCII values 0 – 31 or 127 will be transmitted as their equivalent control code combination. For example a carriage return value 13 (0D hex) will be sent as (^M) where ^ represents the Ctrl key on the keyboard.

Caution

If another keyboard is connected to the same host as this device and a key is pressed on the other keyboard while this device is transmitting, then the data transmitted by this device may get corrupted.

Because of potential "data interleave" issues associated with the USB Keyboard interface, MagTek recommends that the USB Keyboard Emulation MSR product should only be used by customers who have previously used MagTek's Keyboard Wedge MSR, or who are interfacing with an existing PC software application which gathers card data from the keyboard port. If previous applications were based upon RS-232 serial interface MSR's, or if this is a brand new development effort, it is strongly recommended that you use the MagTek's "standard version" of the USB MSR (Non-Keyboard Emulation Version). Please refer to Technical Manual 99875191 for further information regarding the "standard version" USB MSR.

The device's programmable configuration options affect the format of the card data.

The card data format for the default configuration is as follows:

[Tk1 SS][Tk1 Data][ES][Tk2 SS][Tk2 Data][ES][Tk3 SS][Tk3 Data][ES][CR]

where:

Tk1 SS	=	% (7-bit start sentinel)
Tk2 SS	=	; (ISO/ABA 5-bit start sentinel)
		@ (7-bit start sentinel)
Tk3 SS	=	+ (ISO/ABA start sentinel)
		! (CA drivers licence start sentinel)
		# (AAMVA start sentinel)
		& (7-bit start sentinel)
ES	=	? (end sentinel)
CR	=	(carriage return) (0D hex)

All data will be sent in upper case regardless of the state of the caps lock key on the keyboard. If no data is detected on a track then nothing will be transmitted for that track. If an error is detected on a track the ASCII character E will be sent in place of the track data to indicate an error.

The card data format for all programmable configuration options is as follows:

[P11][P13][Tk1 SS][Tk1 Data][ES][LRC][P14][P5][P13][Tk2 SS][Tk2 Data][ES][LRC][P14][P5][P13][Tk3 SS][Tk3 Data][ES][LRC][P14][P5][P12]

where:

ES	=	? (end sentinel)
LRC	=	Longitudinal redundancy check character
P5	=	Terminating character
P11	=	Pre card character
P12	=	Post card character
P13	=	Pre track character
P14	=	Post track character
Tk1 SS	=	% (7-bit start sentinel)
Tk2 SS	=	; (ISO/ABA 5-bit start sentinel)
		P6 (7-bit start sentinel)
Tk3 SS	=	P8 (ISO/ABA start sentinel)
		P7 (CA drivers licence start sentinel)
		P9 (AAMVA start sentinel)
		P10 (7-bit start sentinel)

All fields with the format P# are programmable configuration property numbers. They are described in detail later in this document.

PROGRAMMABLE CONFIGURATION OPTIONS

This device has a number of programmable configuration properties. These properties are stored in non-volatile EEPROM memory. These properties can be configured at the factory or by the end user using a program supplied by MagTek. Programming these parameters requires low level communications with the device. During normal device operation, the device acts like a USB HID keyboard so the host operating system takes care of all low level communications with

the device so that the application developer is not burdened with these low level details. Details on how to communicate with the device to change programmable configuration properties follows in the next few sections. These details are included as a reference only. Most users will not need to know these details because the device will be configured at the factory or by a program supplied by MagTek. Most users may want to skip over the next few sections on low level communications and continue with the details of the configuration properties.

LOW LEVEL COMMUNICATIONS

It is strongly recommended that application software developers become familiar with the HID specification the USB specification before attempting to communicate directly with this device. This document assumes that the reader is familiar with these specifications. These specifications can be downloaded free from www.usb.org.

HID USAGES

HID devices send data in reports. Elements of data in a report are identified by unique identifiers called usages. The structure of the device's reports and the device's capabilities are reported to the host in a report descriptor. The host usually gets the report descriptor only once, right after the device is plugged in. The report descriptor usages identify the devices capabilities and report structures. For example, a device could be identified as a keyboard by analyzing the device's report descriptor. Usages are four byte integers. The most significant two bytes are called the usage page and the least significant two bytes are called usage IDs. Usages that are related can share a common usage page. Usages can be standardized or they can be vendor defined. Standardized usages such as usages for mice and keyboards can be found in the HID Usage Tables document and can be downloaded free at www.usb.org. Vendor defined usages must have a usage page in the range 0xff00 – 0xffff. All usages for this device use the standard HID keyboard usages or vendor defined magnetic stripe reader usage page 0xff00. The vendor defined usage IDs for this device are defined in the following table. The usage types are also listed. These usage types are defined in the HID Usage Tables document.

Magnetic Stripe Reader usage page 0xff00:

Usage ID (Hex)	Usage Name	Usage Type	Report Type
20	Command message	Data	Feature

REPORT DESCRIPTOR

The HID report descriptor is structured as follows:

Item	Value(Hex)
Usage Page (Generic Desktop)	05 01
Usage (Keyboard)	09 06
Collection (Application)	A1 01
Usage Page (Key Codes)	05 07
Usage Minimum (224)	19 E0
Usage Maximum (231)	29 E7
Logical Minimum (0)	15 00
Logical Maximum (1)	25 01
Report Size (1)	75 01
Report Count (8)	95 08
Input (Data, Variable, Absolute)	81 02
Report Count (1)	95 01
Report Size (8)	75 08
Input (Constant)	81 03
Report Count (5)	95 05
Report Size (1)	75 01
Usage Page (LEDs)	05 08
Usage Minimum (1)	19 01
Usage Maximum (5)	29 05
Output (Data, Variable, Absolute)	91 02
Report Count (1)	95 01
Report Size (3)	75 03
Output (Constant)	91 03
Report Count (6)	95 06
Report Size (8)	75 08
Logical Minimum (0)	15 00
Logical Maximum (101)	25 66
Usage Page (Key Codes)	05 07
Usage Minimum (0)	19 00
Usage Maximum (101)	29 66
Input (Data, Array)	81 00
Logical Maximum (255)	26 FF 00
Usage Page (vendor defined (MSR))	06 00 FF
Usage (command data)	09 20
Report Count	95 18
Feature (Data, Variable, Absolute, Buffered Bytes)	B2 02 01
End Collection	C0

COMMANDS

Command requests and responses are sent to and received from the device using feature reports. Command requests are sent to the device using the HID class specific request Set_Report. The response to a command is retrieved from the device using the HID class specific request Get_Report. These requests are sent over the default control pipe. When a command request is sent, the device will Nak the Status stage of the Set_Report request until the command is completed. This insures that as soon as the Set_Report request is completed, the Get_Report request can be sent to get the command response. The usage ID for the command message was shown previously in the Usage Table.

The following table shows how the feature report is structured for command requests:

Offset	Field Name
0	Command Number
1	Data Length
2 – 23	Data

The following table shows how the feature report is structured for command responses.

Offset	Field Name
0	Result Code
1	Data Length
2 – 23	Data

COMMAND NUMBER

This one byte field contains the value of the requested command number. The following table lists all the existing commands.

Value	Command Number	Description
0	GET_PROPERTY	Gets a property from the device
1	SET_PROPERTY	Sets a property in the device

DATA LENGTH

This one byte field contains the length of the valid data contained in the Data field.

DATA

This multi-byte field contains command data if any. Note that the length of this field is fixed at 22 bytes. Valid data should be placed in the field starting at offset 2. Any remaining data after the valid data should be set to zero. This entire field must always be set even if there is no valid data. The HID specification requires that Reports be fixed in length. Command data may vary in length. Therefore, the Report should be filled with zeros after the valid data.

RESULT CODE

This one byte field contains the value of the result code. There are two types of result codes: generic result codes and command specific result codes. Generic result codes always have the most significant bit set to zero. Generic result codes have the same meaning for all commands and can be used by any command. Command specific result codes always have the most significant bit set to one. Command specific result codes are defined by the command that uses them. The same code can have different meanings for different commands. Command specific result codes are defined in the documentation for the command that uses them. Generic result codes are defined in the following table.

Value	Result Code	Description
0	SUCCESS	The command completed successfully.
1	FAILURE	The command failed.
2	BAD_PARAMETER	The command failed due to a bad parameter or command syntax error.

GET AND SET PROPERTY COMMANDS

The Get Property command gets a property from the device. The Get Property command number is 0.

The Set Property command sets a property in the device. The Set Property command number is 1.

The Get and Set Property command data fields for the requests and responses are structured as follows:

Get Property Request Data:

Data Offset	Value
0	Property ID

Get Property Response Data:

Data Offset	Value
0 – n	Property Value

Set Property Request Data:

Data Offset	Value
0	Property ID
1 – n	Property Value

Set Property Response Data:
None

The result codes for the Get and Set Property commands can be any of the codes list in the generic result code table.

Property ID is a one byte field that contains a value that identifies the property. The following table lists all the current property ID values:

Value	Property ID	Description
0	SOFTWARE_ID	The device's software identifier
1	SERIAL_NUM	The device's serial number
2	POLLING_INTERVAL	The interrupt pipe's polling interval
3	TRACK_ID_ENABLE	Track enable / ID enable
4	TRACK_DATA_SEND_FLAGS	Track data send flags
5	TERMINATION_CHAR	Terminating char / per track or card flag
6	SS_TK2_7BITS	Start sentinel char for track 2 – 7 bit data
7	SS_TK3_CADL	Start sentinel char for track 3 – CA drivers license
8	SS_TK3_ISO_ABA	Start sentinel char for track 3 – ISO/ABA
9	SS_TK3_AAMVA	Start sentinel char for track 3 - AAMVA
10	SS_TK3_7BITS	Start sentinel char for track 3 – 7 bit data
11	PRE_CARD_CHAR	Pre card char
12	POST_CARD_CHAR	Post card char
13	PRE_TK_CHAR	Pre track char
14	POST_TK_CHAR	Post track char

The Property Value is a multiple byte field that contains the value of the property. The number of bytes in this field depends on the type of property and the length of the property. The following table lists all of the property types and describes them.

Property Type	Description
Byte	This is a one byte value. The valid values depend on the property.
String	This is a multiple byte ASCII string. Its length can be zero to a maximum length that depends on the property. The value and length of the string does not include a terminating NUL character.

SOFTWARE_ID PROPERTY

Property ID: 0
Property Type: String
Length: Fixed at 11 bytes
Get Property: Yes
Set Property: No
Description: This is an 11 byte read only property that identifies the software part number and version for the device. The first 8 bytes represent the part number and the last 3 bytes represent the version. For example this string might be "21042802A05". Examples follow:

Example Get SOFTWARE_ID property Request (Hex):

Cmd Num	Data Len	Prp ID
00	01	00

Example Get SOFTWARE_ID property Response (Hex):

Result Code	Data Len	Prp Value
00	01	32 31 30 34 32 38 30 32 41 30 35

SERIAL_NUM PROPERTY

Property ID: 1
Property Type: String
Length: 0 – 15 bytes
Get Property: Yes
Set Property: Yes
Default Value: The default value is no string with a length of zero.
Description: The value is an ASCII string that represents the device's serial number. This string can be 0 – 15 bytes long. This property is stored in non-volatile EEPROM memory so it will not change when the unit is power cycled. The value of this property, if any, will be sent to the host when the host requests the USB string descriptor. When this property is changed, the unit must be power cycled to have these changes take effect for the USB descriptor. If a value other than the default value is desired, it can be set by the factory upon request. Examples follow.

Example Set SERIAL_NUM property Request (Hex):

Cmd Num	Data Len	Prp ID	Prp Value
01	04	01	31 32 33

Example Set SERIAL_NUM property Response (Hex):

Result Code	Data Len	Data
00	00	

Example Get SERIAL_NUM property Request (Hex):

Cmd Num	Data Len	Prp ID
00	01	01

Example Get SERIAL_NUM property Response (Hex):

Result Code	Data Len	Prp Value
00	03	31 32 33

POLLING_INTERVAL PROPERTY

Property ID: 2
Property Type: Byte
Length: 1 byte
Get Property: Yes
Set Property: Yes
Default Value: 1
Description: The value is a byte that represents the devices polling interval for the Interrupt In Endpoint. The value can be set in the range of 1 – 255 and has units of milliseconds. The polling interval tells the host how often to poll the device for card data packets. For example, if the polling interval is set to 10, the host will poll the device for card data packets every 10ms. This property can be used to speed up or slow down the time it takes to send card data to the host. The trade-off is that speeding up the card data transfer rate increases the USB bus bandwidth used by the device, and slowing down the card data transfer rate decreases the USB bus bandwidth used by the device. This property is stored in non-volatile EEPROM

memory so it will not change when the unit is power cycled. The value of this property, if any, will be sent to the host when the host requests the device's USB endpoint descriptor. When this property is changed, the unit must be power cycled to have these changes take effect for the USB descriptor. If a value other than the default value is desired, it can be set by the factory upon request. Examples follow:

Example Set POLLING_INTERVAL property Request (Hex):

Cmd Num	Data Len	Prp ID	Prp Value
01	02	02	0A

Example Set POLLING_INTERVAL property Response (Hex):

Result Code	Data Len	Data
00	00	

Example Get POLLING_INTERVAL property Request (Hex):

Cmd Num	Data Len	Prp ID
00	01	02

Example Get POLLING_INTERVAL property Response (Hex):

Result Code	Data Len	Prp Value
00	01	0A

TRACK_ID_ENABLE PROPERTY

Property ID: 3
Property Type: Byte
Length: 1 byte
Get Property: Yes
Set Property: Yes
Default Value: 95 (hex)
Description: This property is defined as follows:

id	0	T_3	T_3	T_2	T_2	T_1	T_1

Id 0 – Decodes standard ISO/ABA cards only
1 – Decodes AAMVA, CA DL/ID and 7-bit cards also

$T_{\#}$ 00 – Track Disabled
01 – Track Enabled
10 – Track Enabled/Required (Error if blank)

This property is stored in non-volatile EEPROM memory so it will not change when the unit is power cycled. When this property is changed, the unit must be power cycled to have these changes take effect. If a value other than the default value is desired, it can be set by the factory upon request.

TRACK_DATA_SEND_FLAGS PROPERTY

Property ID: 4
Property Type: Byte
Length: 1 byte
Get Property: Yes
Set Property: Yes
Default Value: 63 (hex)
Description: This property is defined as follows:

0	SS	ES	LRC	0	LC	Er	Er

SS 0 – Don't send Start Sentinel for each track
1 – Send Start Sentinel for each track

ES 0 – Don't send End Sentinel for each track
1 – Send End Sentinel for each track

LRC 0 – Don't send LRC for each track
1 – Send LRC for each track

Note that the LRC is the unmodified LRC from the track data. To verify the LRC the track data needs to converted back from ASCII to card data format and the start sentinels that were modified to indicate the card encode type need to be converted back to their original values.

LC 0 – Send card data as upper case
1 – Send card data as lower case

Note that the state of the Caps Lock key on the host keyboard has no affect on what case the card data is transmitted in.

Er 00 – Don't send any card data if error
01 – Don't send track data if error
11 – Send 'E' for each track error

This property is stored in non-volatile EEPROM memory so it will not change when the unit is power cycled. When this property is changed, the unit must be power cycled to have these changes take effect. If a value other than the default value is desired, it can be set by the factory upon request.

TERMINATION_CHAR PROPERTY

Property ID: 5
Property Type: Byte
Length: 1 byte
Get Property: Yes

Set Property: Yes
Default Value: 0D (hex) (carriage return)
Description: This property is defined as follows:

mod	c	c	c	c	c	c	c

mod 0 – Send c after card data
1 – Send c after each track

c 1-127 – 7 bit ASCII char code
0 – send nothing

This property is stored in non-volatile EEPROM memory so it will not change when the unit is power cycled. When this property is changed, the unit must be power cycled to have these changes take effect. If a value other than the default value is desired, it can be set by the factory upon request.

SS_TK2_7BITS PROPERTY

Property ID: 6
Property Type: Byte
Length: 1 byte
Get Property: Yes
Set Property: Yes
Default Value: 40 (hex) '@'
Description: This character is sent as the track 2 start sentinel for cards that have track 2 encoded in 7 bits per character format. If the value is 0 no character is sent. If the value is in the range 1 – 127 then the equivalent ASCII character will be sent.

This property is stored in non-volatile EEPROM memory so it will not change when the unit is power cycled. When this property is changed, the unit must be power cycled to have these changes take effect. If a value other than the default value is desired, it can be set by the factory upon request.

SS_TK3_CADL PROPERTY

Property ID: 7
Property Type: Byte
Length: 1 byte
Get Property: Yes
Set Property: Yes
Default Value: 21 (hex) '!'
Description: This character is sent as the track 3 start sentinel for cards that have track 3 encoded in California drivers license format. If the value is 0 no character is sent. If the value is in the range 1 – 127 then the equivalent ASCII character will be sent.

This property is stored in non-volatile EEPROM memory so it will not change when the unit is power cycled. When this property is changed, the unit must be power cycled to have these changes take effect. If a value other than the default value is desired, it can be set by the factory upon request.

SS_TK3_ISO_ABA PROPERTY

Property ID: 8
Property Type: Byte
Length: 1 byte
Get Property: Yes
Set Property: Yes
Default Value: 2B (hex) '+'
Description: This character is sent as the track 3 start sentinel for cards that have track 3 encoded in ISO/ABA format. If the value is 0 no character is sent. If the value is in the range 1 – 127 then the equivalent ASCII character will be sent.

This property is stored in non-volatile EEPROM memory so it will not change when the unit is power cycled. When this property is changed, the unit must be power cycled to have these changes take effect. If a value other than the default value is desired, it can be set by the factory upon request.

SS_TK3_AAMVA PROPERTY

Property ID: 9
Property Type: Byte
Length: 1 byte
Get Property: Yes
Set Property: Yes
Default Value: 23 (hex) '#'
Description: This character is sent as the track 3 start sentinel for cards that have track 3 encoded in AAMVA format. If the value is 0 no character is sent. If the value is in the range 1 – 127 then the equivalent ASCII character will be sent.

This property is stored in non-volatile EEPROM memory so it will not change when the unit is power cycled. When this property is changed, the unit must be power cycled to have these changes take effect. If a value other than the default value is desired, it can be set by the factory upon request.

SS_TK3_7BITS PROPERTY

Property ID: 10 (0A hex)
Property Type: Byte
Length: 1 byte
Get Property: Yes
Set Property: Yes
Default Value: 26 (hex) '&'

Description: This character is sent as the track 3 start sentinel for cards that have track 3 encoded in 7 bits per character format. If the value is 0 no character is sent. If the value is in the range 1 – 127 then the equivalent ASCII character will be sent.

This property is stored in non-volatile EEPROM memory so it will not change when the unit is power cycled. When this property is changed, the unit must be power cycled to have these changes take effect. If a value other than the default value is desired, it can be set by the factory upon request.

PRE_CARD_CHAR PROPERTY

Property ID: 11 (0B hex)
Property Type: Byte
Length: 1 byte
Get Property: Yes
Set Property: Yes
Default Value: 0
Description: This character is sent prior to all other card data. If the value is 0 no character is sent. If the value is in the range 1 – 127 then the equivalent ASCII character will be sent.

This property is stored in non-volatile EEPROM memory so it will not change when the unit is power cycled. When this property is changed, the unit must be power cycled to have these changes take effect. If a value other than the default value is desired, it can be set by the factory upon request.

POST_CARD_CHAR PROPERTY

Property ID: 12 (0C hex)
Property Type: Byte
Length: 1 byte
Get Property: Yes
Set Property: Yes
Default Value: 0
Description: This character is sent after all other card data. If the value is 0 no character is sent. If the value is in the range 1 – 127 then the equivalent ASCII character will be sent.

This property is stored in non-volatile EEPROM memory so it will not change when the unit is power cycled. When this property is changed, the unit must be power cycled to have these changes take effect. If a value other than the default value is desired, it can be set by the factory upon request.

PRE_TK_CHAR PROPERTY

Property ID: 13 (0D hex)
Property Type: Byte
Length: 1 byte
Get Property: Yes
Set Property: Yes
Default Value: 0
Description: This character is sent prior to the data for each track. If the value is 0 no character is sent. If the value is in the range 1 – 127 then the equivalent ASCII character will be sent.

This property is stored in non-volatile EEPROM memory so it will not change when the unit is power cycled. When this property is changed, the unit must be power cycled to have these changes take effect. If a value other than the default value is desired, it can be set by the factory upon request.

POST_TK_CHAR PROPERTY

Property ID: 14 (0E hex)
Property Type: Byte
Length: 1 byte
Get Property: Yes
Set Property: Yes
Default Value: 0
Description: This character is sent after the data for each track. If the value is 0 no character is sent. If the value is in the range 1 – 127 then the equivalent ASCII character be sent.

This property is stored in non-volatile EEPROM memory so it will not change when the unit is power cycled. When this property is changed, the unit must be power cycled to have these changes take effect. If a value other than the default value is desired, it can be set by the factory upon request.

SECTION 5. DEMO PROGRAM

The purpose of this demo program is not to demonstrate card reading with this HID keyboard emulation device. Use a text editor application such as Windows Notepad to demonstrate card reading for this HID keyboard emulation device. Any application that allows user input from a keyboard should be sufficient to demonstrate card reading for this device.

The primary purpose of the demo program, when used with this HID keyboard emulation device, is to allow users to change the device's programmable configuration properties. This is accomplished by sending commands to the device with the demo program. The demo program also comes with source code that can be used as a guide for application developers who what to change the device's programmable configuration properties in an application. However, it is unlikely that application developers will want to change these properties in an application since these properties only need to be set once and can be set at the factory. This program is written in Visual Basic.

The demo program does not support this HID keyboard emulation device on the Windows 2000 platform. It only works on Windows 98 and Me.

When the demo program is run, a button for reading cards is displayed along with a button for sending commands. The card reading option is not supported for this HID keyboard emulation device. Use a text editor application such as Windows Notepad to demonstrate card reading for this HID keyboard emulation device.

The part numbers for the demo program can be found in this document in Section 1 under Accessories.

INSTALLATION

To install the demo program, run the setup.exe file and follow the instructions given on the screen.

OPERATION

To operate the demo program perform the following steps:

- Plug the device into a USB port on the host
- If this is the first time the device has been plugged into the host, then follow the instructions on the screen for installing the Windows HID device driver. This is explained in more detail in the installation section of this document.
- Run the demo program.
- To read cards and view the card data do not use the demo program. Use a text editor program such as Windows Notepad.
- To send commands to the device, click on the send commands button.
- Enter a command in the Message edit box. All data entered should be in hexadecimal bytes with a space between each byte. Enter the command number followed by the command data if there is any. The application will automatically calculate and send the command data

length for you. For example, to send the GET_PROPERTY command for property SOFTWARE_ID enter 00 00.

- Press Enter or click on Send message to send the command and receive the result.
- The command request and the command result will be displayed in the Communications Dialog edit box.
- The Clear Dialog button clears the Communication Dialog edit box.

SOURCE CODE

Source code is included with the demo program. It can be used as a guide for application development. It is described in detail, with comments, to assist developers. The book *USB Complete* by Jan Axelson is also a good guide for application developers, especially the chapter on Human Interface Device Host Applications (see "Reference Documents" in Section 1).

PORT POWERED
SWIPE READER
TECHNICAL REFERENCE MANUAL

Manual Part Number 99875094 Rev 10

JULY 2001

MAGTEK

20725 South Annalee Avenue
Carson, CA 90746
Phone: (310) 631-8602
FAX: (310) 631-3956
Technical Support: (888) 624-8350
www.magtek.com

REVISIONS

Rev Number	Date	Notes
1	11 Dec 97	Initial Release
2	18 Dec 97	Sec 1 Changed spec Dimensions; Sec 1, 3 Changed illustrations for clarity.
3	11 May 98	Sections 1, 2, and 3 revised to reflect latest firmware revisions. Sec 4 deleted.
4	15 Jun 98	Two part numbers added.
5	1 Mar 99	Sec 1, Added 3 part numbers, changed specs, Changed Dimensions Figure 1-3, removed Figure 1-4, Mounting Dimensions, added Mag-Tek Windows Drivers; Added note to Table 1-1. Section 2, added mounting instructions and Figure 2-1, Mounting Dimensions. Section 3, added 3 track symbols to Table 3-1 and 3 sign-on configurations to 3-2.
6	14 Jun 99	Title change, Removed MT-211 and RS-232; Sec 1, Table 1-1, added Pin List for Cable 21040077, added RS-232 Communication; Sec 2, added Demo Program from Net; Sec 3, Clarified Fig 3-1, Described firmware P/Ns and revisions.
7	1 Dec 99	Section 1: Added P/N 21040084, Updated table for 9- and 25-pin connectors; Section 3: Added P/N 21040084 to Sign-on table.
8	21 Sep 00	Editorial changes throughout. Sec 1: Configuration list expanded and moved to Sec 3; Specification weight changed from 5.9 oz to 5.8oz, Converted symbols to Metric System [SI]. Sec 3: Added 5 new part numbers with firmware, tracks, and configurations.
9	09 Mar 01	Front Matter: Corrected Agency Approvals to include Class B for FCC and Class B for CE. Changed RMA Warranty address to 20801 S. Annalee. Section 3: Removed “Track 3 – 7 bit” line from Table 3-1. Added 094 and 096 configurations in Table 3-2.
10	25 Jul 01	Front Matter: Agency Approvals: Corrected Class B for CE and Corrected UL and CUL . Copyright 2001 added.

Limited Warranty

Mag-Tek, Inc. (hereinafter "Mag-Tek") warrants this Mag-Tek product IN ITS ENTIRETY, to be in good working order for a period of one year from the date of purchase from Mag-Tek. Should this product fail to be in good working order at any time during this warranty period, Mag-Tek will, at its option, repair or replace this product at no additional charge except as set forth below. Repair parts and replacement products will be furnished on an exchange basis and will be either reconditioned or new. All replaced parts and products become the property of Mag-Tek. This limited warranty does not include service to repair damage to the product resulting from accident, disaster, misuse, abuse, or non-Mag-Tek modification of the product.

Limited Warranty service may be obtained by delivering the product during the warranty period to Mag-Tek (20801 S. Annalee Ave., Carson, CA 90746). If this product is delivered by mail, you agree to insure the product or assume the risk of loss or damage in transit, to prepay shipping charges to the warranty service location and to use the original shipping container or equivalent.

ALL EXPRESS AND IMPLIED WARRANTIES FOR THIS PRODUCT, INCLUDING THE WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE, ARE LIMITED IN DURATION TO A PERIOD OF ONE YEAR FROM THE DATE OF PURCHASE, AND NO WARRANTIES, WHETHER EXPRESS OR IMPLIED, WILL APPLY AFTER THIS PERIOD, EXCEPT AS PROVIDED IN THE PRECEDING SENTENCE. EACH PURCHASER UNDERSTANDS THAT THE MAG-TEK PRODUCT IS OFFERED AS IS.

IF THIS PRODUCT IS NOT IN GOOD WORKING ORDER AS WARRANTED ABOVE, YOUR SOLE REMEDY SHALL BE REPAIR OR REPLACEMENT AS PROVIDED ABOVE. IN NO EVENT WILL MAG-TEK BE LIABLE TO YOU FOR ANY DAMAGES, INCLUDING ANY LOST PROFITS, LOST SAVINGS OR OTHER INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OF OR INABILITY TO USE SUCH PRODUCT, EVEN IF MAG-TEK HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES, OR FOR ANY CLAIM BY ANY OTHER PARTY.

THIS WARRANTY GIVES YOU SPECIFIC LEGAL RIGHTS, AND YOU MAY ALSO HAVE OTHER RIGHTS WHICH VARY FROM STATE TO STATE.

FCC WARNING STATEMENT

This equipment has been tested and found to comply with the limits for Class B digital device, pursuant to Part 15 of FCC Rules. These limits are designed to provide reasonable protection against harmful interference when the equipment is operated in a residential environment. This equipment generates, uses, and can radiate radio frequency energy and, if not installed and used in accordance with the instruction manual, may cause harmful interference to radio communications. However, there is no guarantee that interference will not occur in a particular installation.

FCC COMPLIANCE STATEMENT

This device complies with Part 15 of the FCC Rules. Operation of this device is subject to the following two conditions: (1) This device may not cause harmful interference; and (2) this device must accept any interference received, including interference that may cause undesired operation.

CANADIAN DOC STATEMENT

This digital apparatus does not exceed the Class B limits for radio noise for digital apparatus set out in the Radio Interference Regulations of the Canadian Department of Communications.

Le présent appareil numérique n'émet pas de bruits radioélectriques dépassant les limites applicables aux appareils numériques de las classe B prescrites dans le Réglement sur le brouillage radioélectrique édicté par les ministère des Communications du Canada.

CE STANDARDS

Testing for compliance to CE and FCC requirements was performed by an independent laboratory. The unit under test was found compliant to Class B.

UL/CSA

This product is recognized per Underwriter Laboratories and Canadian Underwriter Laboratories 1950.

TABLE OF CONTENTS

FIGURES

TABLES

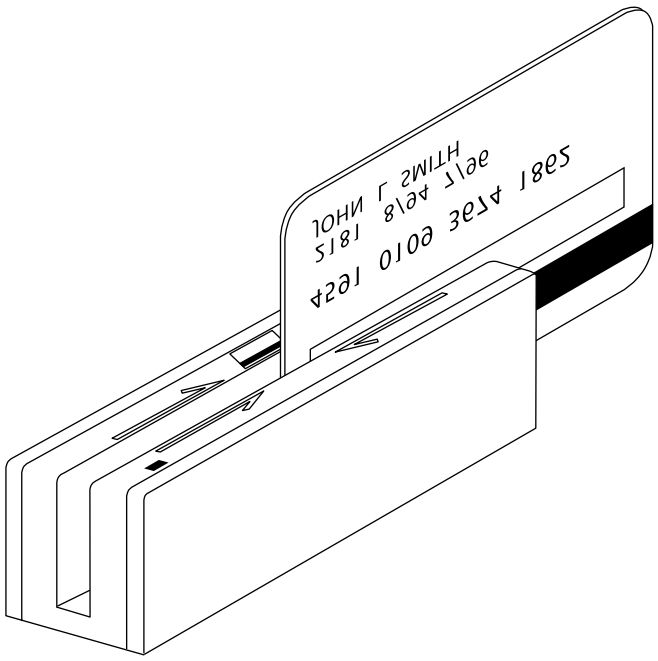


Figure 1-1. Port-Powered Swipe Reader

SECTION 1. FEATURES AND SPECIFICATIONS

The Port Powered Swipe Reader is a compact magnetic stripe card reader which conforms to ISO/ANSI standards. The Reader is compatible with the PC series of personal computers or any device with a serial RS-232 interface. A card is read by sliding it, stripe down and facing the LED side, through the slot either forward or backward.

A green LED (Light Emitting Diode) indicator on the Reader panel provides the operator with continuous status of the Reader operations.

When power is applied, the Reader transmits a sign-on ID message. About 150 milliseconds after DTR is applied, the Reader sends the part number of the firmware in the following form: 21088819A01 <CR>. The first 8 characters indicate the firmware number; the letter is the revision, which is followed by a revision sublevel of 01 to 99. The <CR> indicates carriage return (0x0D). The sign-on messages for part numbers are listed in Section 3. Timing is also shown in Section 3.

Since the input voltage is supplied by a relatively low source of power, the Reader depends on its input capacitor to maintain proper charge during all operations. In order to reduce the drain on this internal power source during data transmission, the output data is transmitted in 5 to 6 millisecond bursts with a 10-millisecond gap between bursts to allow the capacitor to recharge. The PC software should be able to tolerate this 10-millisecond space between characters. The Timing is shown is Section 3, Figure 3-1. Configurations, including part numbers, firmware, tracks, and unit configuration, are listed in Section 3, Table 3-2.

MAG-TEK DEVICE DRIVERS FOR WINDOWS

The Mag-Tek Device Drivers for Windows, Part Number 30037385, may be used with the Port Powered Swipe Reader. When this program is used, refer to *Mag-Tek Device Driver for Windows, Programming Reference Manual*, Part Number 99875125.

FEATURES

Major features of the Swipe Reader are as follows:

- Powered through the RS-232 serial port – no external power supply required
- Hardware Compatible with PC or any computer or terminal with an RS-232 interface
- Software Compatible with Procomm, or any RS-232 communications program
- Bidirectional card reading
- Reads encoded data that meets ANSI/ISO/CDL/AAMVA standards
- Green LED for status

CONFIGURATION

The Reader, LED Indicator, pin numbers for the 9-pin connector, and the Adapter are shown in Figure 1-2.

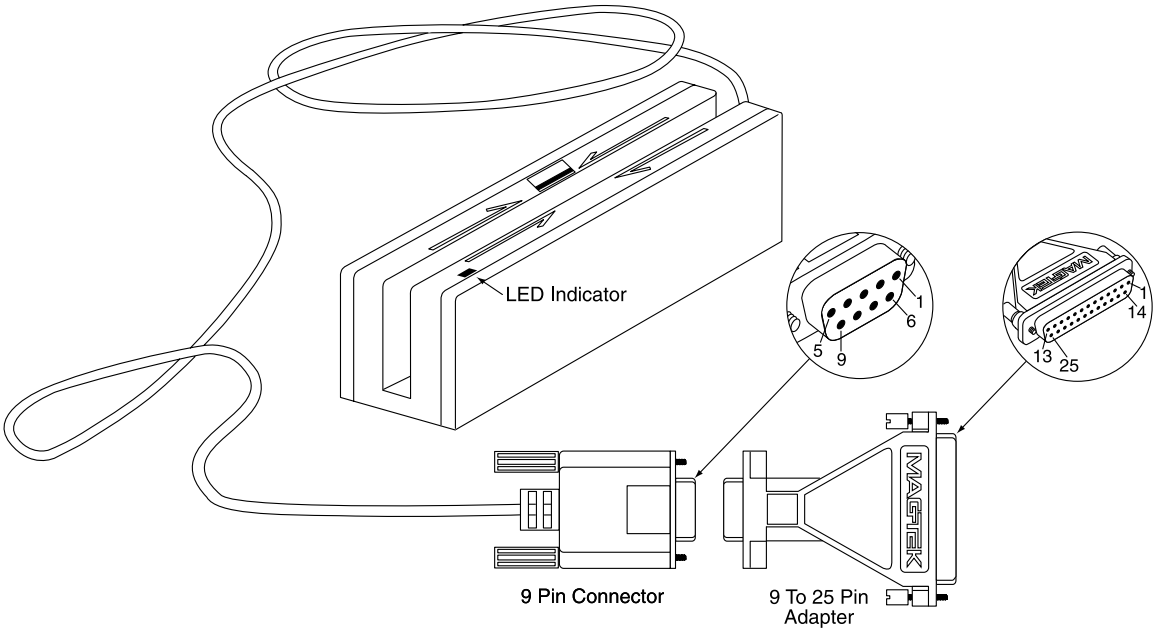


Figure 1-2. Reader Cable and Optional Adapter

Pin numbers and signal descriptions for the 9-pin (DE9) cable and 25-pin (DB25) adapter shown in the illustration are listed in Table 1-1. Also listed is the pin list OEM version, P/N 21040077.

Table 1-1. OEM and 9-Pin Connectors and 25-Pin Adapter

Connector for OEM Version 21040077 J2 on the PCB	25-pin Adapter	DE9-pin Connector	Signal
	-	1	NC*
1	3	2	RXD (to PC)
2	2	3	TXD** (from PC)
3	20	4	DTR (from PC)
4	7	5	GND
	-	6-9	NC*

* No Connection

** Pin must be connected to TXD (or DTR if TXD not available).

SPECIFICATIONS

Table 1-2 lists the specifications for the Port Powered Swipe Reader. Figure 1-3 shows the dimensions for the standard product. Other sizes are available by special order.

Table 1-2. Specifications

OPERATING	
Reference Standards	ISO/ANSI/ CDL/ AAMVA*
Power Input	From RS-232 interface
Recording Method	Two-frequency coherent phase (F2F)
Message Format	ASCII
Card Speed	3 to 50 IPS
MTBF	Electronics: 125,000 hours. Head: 1,000,000 passes
ELECTRICAL	
DTR Voltage	5 to 15 VDC
Current Quiescent Transmitting Peak at Power On	 1 to 2 mA typical (continuous) 8 to 9 mA typical (5 ms duration) 12 mA
RS-232 Communication	9600 bps, no parity, 8 data bits, 1 stop bit
MECHANICAL (STANDARD PRODUCT)	
Dimensions	Length: 3.94” (100.0 mm), Width: 1.28” (32.5 mm) Height: 1.23” (31.3 mm)
Weight	Reader 5.8 oz. (165 gr.)
Cable length	5 Ft. (1.5 m)
Connector	9 pin D female (May require a 25-pin adapter)
ENVIRONMENTAL	
Temperature	
Operating	32°F to 131°F (0°C to 55°C)
Storage	-22°F to 158°F (-30°C to 70°C)
Humidity	
Operating	10% to 90% noncondensing
Storage	Up to 100% noncondensing
Altitude	
Operating	0-10,000 ft. (0-3048 m.)
Storage	0-50,000 ft. (0-15240 m.)

* ISO (International Standards Organization), ANSI (American National Standards Institute), CDL (California Drivers License), and AAMVA (American Association of Motor Vehicle Administrators).

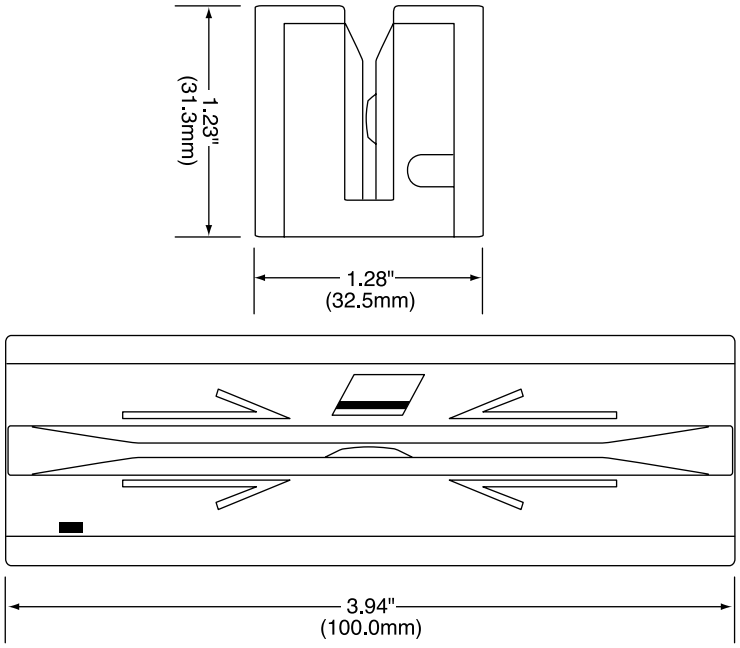


Figure 1-3. Dimensions

SECTION 2. INSTALLATION

The hardware installation consists of plugging the cable into the PC and optional adapter, if required, Com Port setup, and testing the Reader.

REQUIREMENTS

- Port Powered Swipe Reader
- Optional 9- to 25-pin Adapter, P/N 78200018
- PC with Com Port
- Procomm, Hyper Terminal, Mag-Tek Windows Drivers, or other RS-232 communications program

MOUNTING

1. The Reader can be mounted on a surface in three ways:

   - By two screws through the surface attached to the bottom of the unit and running the cable on the top of the surface;

   - By two screws through the surface attached to the bottom of the unit and by drilling a hole in the surface for the cable and running the cable through the hole;

   - By attaching the unit to the surface with fastening tape and running the cable on the top of the surface.

   Note

   The two mounting inserts are 3 mm diameter; 0.5 mm pitch; 6.4 mm deep. The length of the screws used depends on the mounting surface thickness and the thickness of washers (if used).

   The mounting dimensions are shown in Figure 2-1. Determine the method of mounting required.

2. Ensure the Reader is positioned on a flat, accessible surface with at least 4 inches clearance on either end for room to swipe a card. Orient the Reader so the side with the LED is facing the direction of intended use.

   If fastening tape is to be used, clean the area that the Reader will be mounted on with isopropyl alcohol. Remove the adhesive protective cover on the fastening tape, and position the Reader and push down firmly.

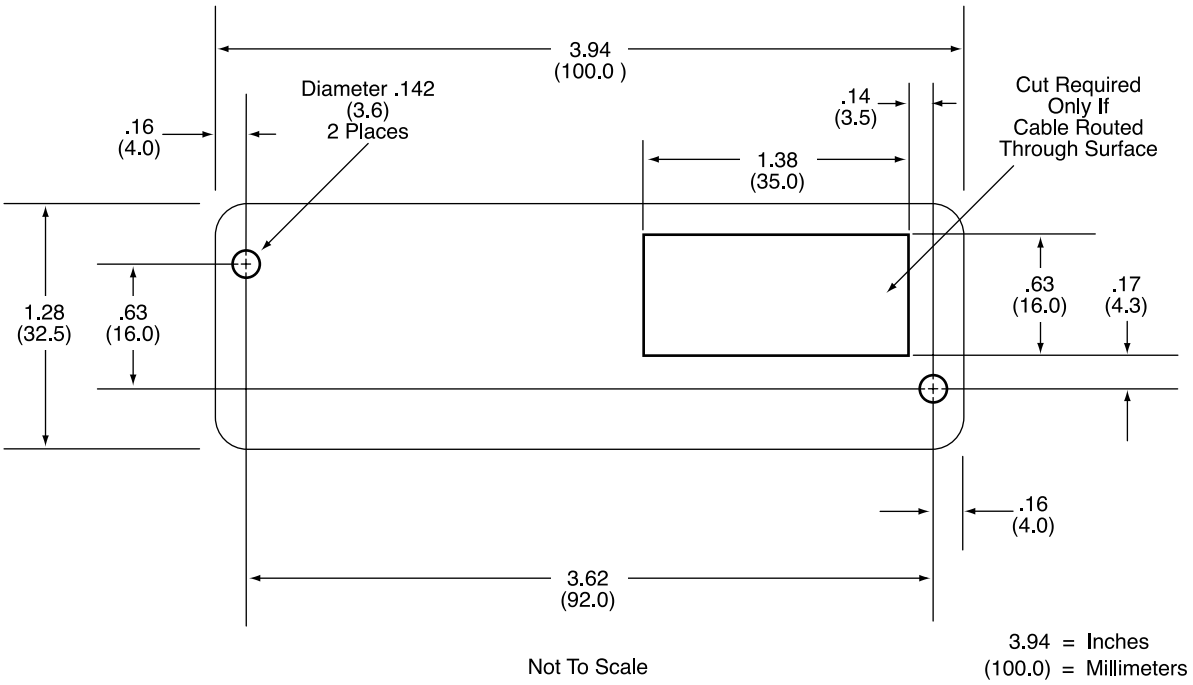


Figure 2-1. Mounting Hole Dimensions For Surface

3. Mount the Reader.

INSTALLATION AND TEST

To install the Swipe Reader, perform the following steps:

1. Connect the Swipe Reader cable connector into a 9-pin serial Com Port on the PC. If a 25-pin Adapter is required, plug the 9-pin connector on the Reader into the Adapter, and the adapter into the PC.

2. Open a communications program such as the Mag-Tek Encoder/Reader Demonstration Program, which may be obtained from the Internet at www.magtek.com. Navigate to the Demo Programs and select Reader & Encoder Demos (Win 95/98/NT).

3. On the program, select the Com Port the Reader is connected to.

4. If the Com Port selected is correct, the green LED on the Reader will light; if the *wrong* Com Port is selected, the LED will not light.

5. Select the baud rate of 9600.

6. Select 8 data bits, no parity, 1 stop bit.

7. With the LED on, swipe a card. The data on the screen will show Track 1 beginning with "%" and ending with "?". Track 2 begins with ";" and ends with "?". Track 3 begins with "+" (normal) or "!" (CDL) and ends with "?". The following is an example:

   %B123^Smith/Joann^9812101000?;1122223333334444444444?<0x0D>

   If a track cannot be read, an **E** will appear in place of the track data; for example, if Track 2 is bad and Tracks 1 and 3 are good, the display will be similar to the following:

   %1111111111111111111?;E?+3333333333333333333?<0x0D>

   If Tracks 1 and 3 are bad and Track 2 is good, the display will be similar to the following:

   %E?;2222222222222222222?+E?<0x0D>

8. If the data on the screen is not numeric or alphanumeric similar to the above, check the communications rate. If the alphanumeric characters are similar to the above, the unit is ready for operation.

SECTION 3. OPERATION

Included in this section are Indicator, Card Read, Reader to Host Message Format, and a timing diagram of sign-on ID.

LED INDICATOR

A green LED indicator on the panel gives the operator the status of the Reader. If the cabling is correct and the correct Com Port is selected, the indicator will be on. If the indictor does not come on, check the cabling and the Com Port. The LED is turned off during a card swipe and while the unit is transmitting.

CARD READ

A card may be swiped through the Reader slot when the green LED is lit. The magnetic stripe must face toward the front (the side with the LED) and may be swiped in either direction.

READER TO HOST MESSAGE FORMAT

Track data is sent in the following order: SS, Card Data, ES.

The format in which data is transmitted (in track order) after a card is read successfully is as follows:

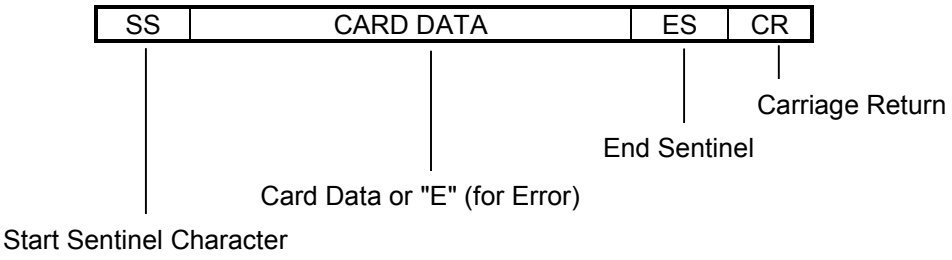


Table 3-1 lists Start Sentinel and End Sentinel symbols.

Table 3-1. SS and ES Track Symbols

Start Sentinel	End Sentinel	Description
%	?	Track 1
;	?	Track 2
+	?	Track 3 - ISO
#	?	Track 3 - AAMVA
!	?	Track 3 - CDL

TIMING FOR ID SIGN ON

Timing for the ID Sign-on and transmission bursts (5 ms with 10 ms between bursts) are shown in Figure 3-1.

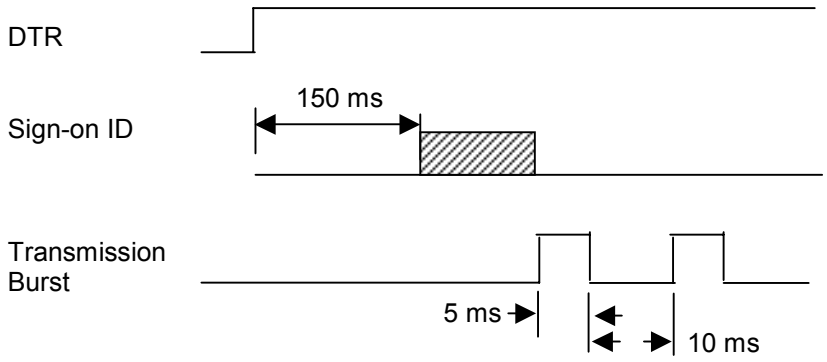


Figure 3-1. Timing For ID Sign-on and Transmission Bursts.

The firmware controls the operation of Sign-on ID and Transmission bursts in the following format:

210888xxLnn <CR>

Where:

the first 8 digits are the firmware part number (xx represents the Swipe Reader series),

L is the alpha revision,

nn is the number sub-revision.

<CR> is 0x0D.

Table 3-2 lists the available part number, firmware, and configuration.

Table 3-2. Sign-on ID for Configurations

Part Number	Firmware	Track Configuration	Configuration*
21040071	21088811	1,2	Pearl White
21040073	21088812	2,3	Pearl White
21040074	21088817	1,2,3	Pearl White
21040075	21088814	2	Pearl White
21040077	21088817	1,2,3	Black, No Cover, No Cable
21040079	21088811	1,2	Black
21040080	21088814	2	Black
21040081	21088811	1,2	Black (150 mm)
21040082	21088817	1,2,3	Black
21040084	21088811	1,2	Pearl White (with STX and ETX)
21040086	21088817	1,2,3	Pearl White, No Cover, 12" Cable, 6-pin
21040088	21088824	1,2,3	Pearl White, 4800/7O, 10' cable
21040089	21088811	1,2	Pearl White, 10' cable
21040091	21088811	1,2	Black, No Cover, 5.9" Cable, 4-pin
21040092	21088817	1,2,3	Pearl White, 5 m cable
21040094	21068811	1,2	Pearl White, No Cover
21040096	21088811	1,2	Black, 4” Cable, 4-pin

*All cables are 6' DE9 unless otherwise specified.

MAGTEK DEVICE DRIVERS FOR WINDOWS
PROGRAMMING REFERENCE MANUAL

Manual Part Number: 99875125 Rev 6

NOVEMBER 2001

MAGTEK

20725 South Annalee Avenue
Carson, CA 90746
Phone: (310) 631-8602
FAX: (310) 631-3956
Technical Support: (888) 624-8350
www.MagTek.com

REVISIONS

Rev Number	Date	Notes
1	20 Nov 98	Initial Release
2	16 Feb 99	Sec 1: Editorial comments for clarification; Sec 2: Added c_wr_secure and trks 1, 2, and 3; Sec 3: Editorial comments for clarification; Appendix A: Added MT-85 and clarified tables; Appendix D: Added c_wr_secure and tks 1, 2, and 3 and MT-85 Encoder sheet.
3	27 Apr 99	Global: Changed names of Mt-211 and MT-215 to port powered readers; Sec 3: Added card insertion note to event; Sec 4: Added this section, Data Parsing. Appendix A: Changed file names. Appendix D. Changed names.
4	21 Oct 99	Sec 1: added: part numbers of media, special commands, MICR material; Sec 2: changed properties table; Sec 3: added errors 45 and 60 to write command; Sec 4: added descriptions to language format; updated default formats; Sec 5: replaced Visual Basic example; Appendix A; Completely revised; Appendix D: added applied_fmt to all forms.
5	14 Dec 99	Appendix A: Added statement about "Long File Names" under "Adding MagTek Device Drivers" General Notes number 4; added statement to "Completing the Installation" about sharing a single port; Edited "Removing the Drivers"; added "Configuration Examples of NT Drivers." Appendix D: Under IntelliPIN PINPad and MSR, added statement under Remarks about IntelliPIN driver; under MiniWedge MSR added statement about ASCII and Character Conversion.
6	30 Nov 01	Editorial changes throughout and added Software Version MTD 1.10, which includes Windows ME/2000/XP.

Limited Warranty

Mag-Tek, Inc. (hereinafter "Mag-Tek") warrants this Mag-Tek product IN ITS ENTIRETY, to be in good working order for a period of 90 days from the date of purchase from Mag-Tek. Should this product fail to be in good working order at any time during this warranty period, Mag-Tek will, at its option, repair or replace this product at no additional charge except as set forth below. Repair parts and replacement products will be furnished on an exchange basis and will be either reconditioned or new. All replaced parts and products become the property of Mag-Tek. This limited warranty does not include service to repair damage to the product resulting from accident, disaster, misuse, abuse, or non-Mag-Tek modification of the product.

Limited Warranty service may be obtained by delivering the product during the warranty period to Mag-Tek (20725 S. Annalee Ave., Carson, CA 90746). If this product is delivered by mail, you agree to insure the product or assume the risk of loss or damage in transit, to prepay shipping charges to the warranty service location and to use the original shipping container or equivalent.

ALL EXPRESS AND IMPLIED WARRANTIES FOR THIS PRODUCT, INCLUDING THE WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE, ARE LIMITED IN DURATION TO A PERIOD OF 90 DAYS FROM THE DATE OF PURCHASE, AND NO WARRANTIES, WHETHER EXPRESS OR IMPLIED, WILL APPLY AFTER THIS PERIOD, EXCEPT AS PROVIDED IN THE PRECEDING SENTENCE. EACH PURCHASER UNDERSTANDS THAT THE MAG-TEK PRODUCT IS OFFERED AS IS.

IF THIS PRODUCT IS NOT IN GOOD WORKING ORDER AS WARRANTED ABOVE, YOUR SOLE REMEDY SHALL BE REPAIR OR REPLACEMENT AS PROVIDED ABOVE. IN NO EVENT WILL MAG-TEK BE LIABLE TO YOU FOR ANY DAMAGES, INCLUDING ANY LOST PROFITS, LOST SAVINGS OR OTHER INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OF OR INABILITY TO USE SUCH PRODUCT, EVEN IF MAG-TEK HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES, OR FOR ANY CLAIM BY ANY OTHER PARTY.

THIS WARRANTY GIVES YOU SPECIFIC LEGAL RIGHTS, AND YOU MAY ALSO HAVE OTHER RIGHTS WHICH VARY FROM STATE TO STATE.

TABLE OF CONTENTS

FIGURES

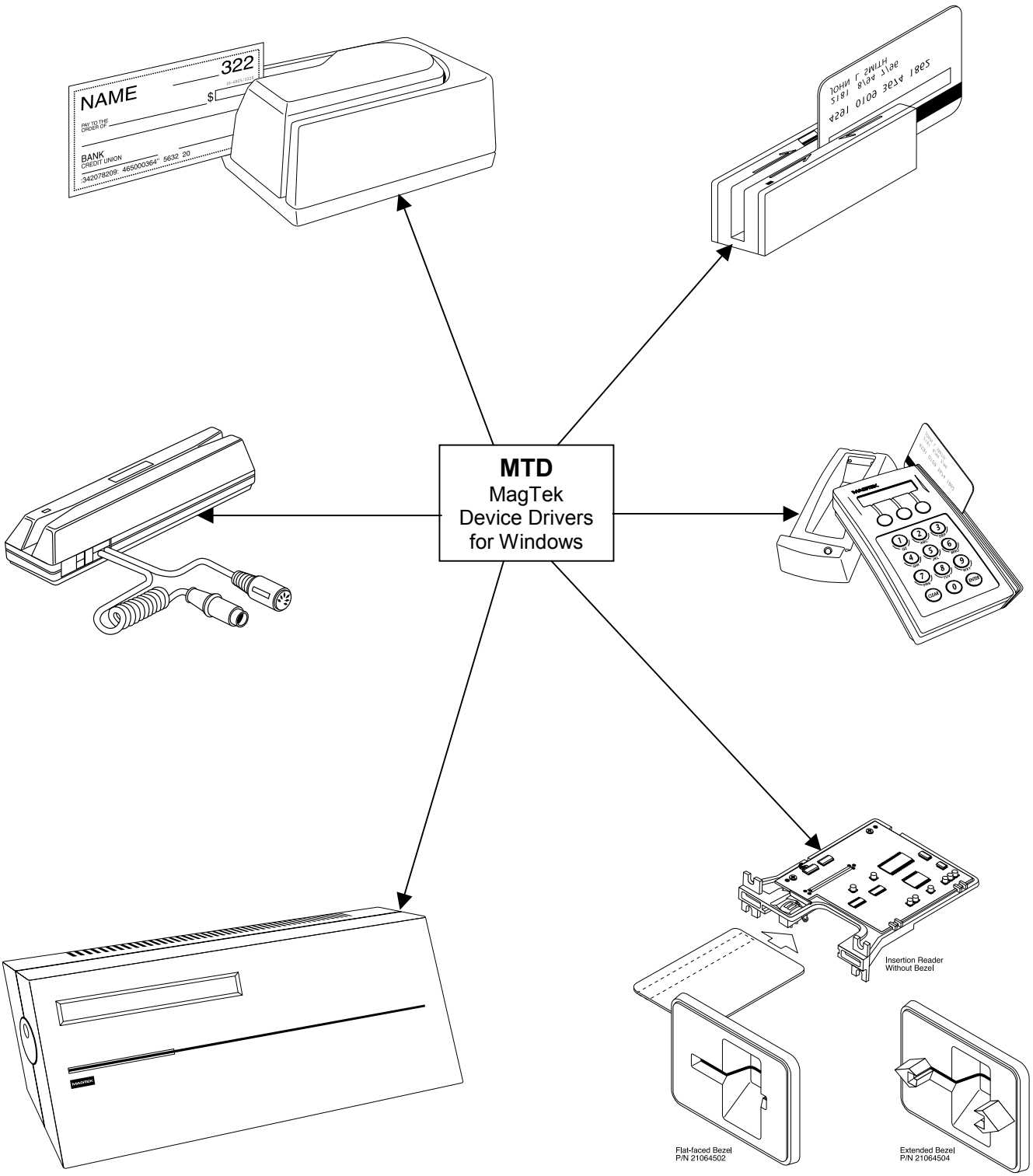


Figure 1-1. MagTek Devices and Device Drivers for Windows

SECTION 1. OVERVIEW

The MagTek Device (MTD) Drivers for Windows is a collection of individual drivers that support a number of MagTek products. These drivers provide a uniform application interface for controlling a wide range of MagTek devices. The drivers, combined with a device control language, solve many of the difficulties application developers face when attempting to control hardware devices. The difficulties mount when faced with the task of developing an application that supports an entire product line of devices.

Part Numbers for the MTD are as follows:

Part Number	Windows Version	Medium
30037385	All	CD
99510030	9X/ME	Internet* (MTD110-9x-ME)
99510031	NT	(MTD110-NT)
99510032	2000/XP	(MTD110-2K-XP)

*www.magtek.com

PROBLEMS WITH CONTROLLING DEVICES

The major problems with developing an application that supports an entire product line of devices are as follows:

- ***Each MagTek device has a unique set of commands***. The commands usually perform similar functions on a particular class of devices but either differ in syntax or have small variations in their functionality. An application would have to implement a custom mechanism to control each device it supported–much like DOS applications had to do to support various printers.

- ***Most MagTek devices communicate via data streams, not packets***. This means that an application receives data from the device one character at a time; it only receives partial command responses. It would be the application's responsibility to collect the incoming data and parse it into individual responses.

- ***Responses from MagTek devices are inherently asynchronous***. When an application sends a command that requires a response, the response from the device arrives (or worse, begins to arrive) long after the command is sent. The application would have to either poll the device until all of the response is collected or implement a callback mechanism to collect and receive it.

- ***Most MagTek devices maintain a communication protocol of some kind***. In addition to this, the protocols differ between devices. For example, some devices frame responses with STX and ETX control characters and others simply use a CR or require a checksum in the frame. To deal with this, an application would have to recognize and implement all of the various protocols for the devices it supports.

- ***MagTek devices are attached to the host in different ways***. MagTek devices may be attached to a serial port, parallel port, to another device or even to the keyboard port. All these ports differ greatly in nature and would all have to be accessed by the application. Additionally, meaningful communication with a device attached to the keyboard port would be tricky at best. This is because the operating system does not provide a means to send data to the keyboard port nor any mechanism to discriminate between the device data and manual keystrokes.

BENEFITS OF A CONTROL LANGUAGE AND DRIVER

A device control language is defined to support most of the functionality of all MagTek devices. As noted previously, most devices of a particular class have similar functionality. The control language defines a common set of commands that perform these functions in the same way for all MagTek devices, thus eliminating device-specific coding for most applications. If the need arises to perform an operation on a device not covered by the common command set, a "raw" send and receive command can be used to communicate directly with the device, effectively eliminating any limitations on the amount of control you have over the device.

The control language is based on a simple property/command model. This model is familiar to most developers who deal with properties and methods in development environments such as Visual Basic or Delphi. You set up the device by getting and setting properties and operate it by invoking commands.

The command set presents a synchronous interface to the application even though the device operates asynchronously, greatly simplifying the effort in retrieving responses from a device. The pattern is simple: send a command to the device and invoke a read command, which will not complete until after the entire response is received from the device.

The control language is implemented by a driver, which completes the solution for the application developer. The driver adds the following benefits:

- ***Gives easy access to the device.*** All MagTek devices are presented uniformly as a virtual serial port, regardless of how they are actually attached to the host.

- ***Hides the communication protocol.*** Adding and stripping frames, performing checksums, detecting and correcting communication errors, etc, are handled completely by the driver. The application sees only the data that it is interested in and can be assured that it is free from transport errors.

- ***Converts the incoming data stream into complete responses.*** The application receives data from the device in easy to use packets. The entire response to a command is received in a single operation.

- ***Makes it easier to upgrade to a new device***. The driver shields you from differences in the new device's commands or interface. When upgrading the device, an application can

usually remain unchanged, even though the new device may be very different from the old one.

The features of a driver that implement a device control language completely shield an application developer from the complexities of device-specific functionality.

LANGUAGE OVERVIEW

The device control language is text based and designed to utilize the read and write file I/O facilities of the underlying operating system. All commands, their responses and properties consist of text strings that are written to or read from the driver using basic file I/O. The control language is based on a property/command model that is similar to the notions of properties and methods as accepted in environments such as Visual Basic or Delphi.

Properties

All properties are accessed in a uniform way: by using a **`get`** (**`/get`** *prop*) or **`set`** (**`/set`** *prop*) command. Properties are either read/write or read only. A **`set`** command with a read only property will fail. All properties are identified by a string name and use strings for their arguments. Properties defined by the control language fall into the following three groups:

- ***Capability properties*** – These properties contain information about the capabilities of a particular device and are generally read only. They allow an application to query a device's capabilities to determine if the device is suitable for a particular task. Included in this category are **`c_cardwpin`**, **`c_check`**, **`c_pin`**, and **`c_magnetic`** (e. g., **`/get c_check`**).

- ***Configuration properties*** – These properties configure a device for different modes of operation or may alter the way some commands behave. Because of this, they are usually readable and writable. They give an application the ability to set up a device for a particular task that requires a specific, non-default mode of operation. Included in this category are **`capitalize`**, **`dev_version`**, and **`port_name`** (e.g., **`/set capitalize 1`**).

- ***Device-specific properties*** –These properties cover configuration requirements that are not common among MagTek devices, even if the devices belong to the same class. An application can determine if a particular set of device-specific properties is available by first querying the device's capabilities or version. Refer to Appendix D, Device Driver Summaries, for a particular driver to see how these properties are affected with an individual device.

Properties can be "action" properties. That is, the driver may execute an action on the device when a property is set. For example, an application can enable or disable magnetic stripe tracks by setting the **`trk_enable`** property. The driver responds by sending one or more commands to the device to enable or disable the desired tracks.

COMMANDS

Like properties, commands are identified by a string name and have string arguments. All commands are terminated by line feed <LF> or a carriage return. To invoke a command, an application simply writes it to the driver in the same manner as writing to a file or serial port. If the command has a response defined for it, the application reads it from the driver using the same I/O handle as in the write.

Four types of commands are defined by the device control language:

- ***Non-interactive*** – These commands manipulate the device without requiring any interaction with the user. The property commands **`get`** and **`set`**, **`reset`** and **`ver`** are examples of this type.

- ***Interactive*** – These commands interact with the user. They do not necessarily require the user to do anything but may only prompt the user to do something. **`display`** is an example of such a command. Others, such as **`read`** or **`write`**, however, require user interaction to complete. For example, the user must either swipe a card or cancel the operation in order to complete a read command.

- ***Device-specific*** – These commands give access to device-specific features. For example, the **`load_key`** command is available for MagTek devices that use keys to encrypt data before sending it to the host.

- ***Raw*** – These are effectively escape commands. They allow the application to bypass the driver to perform device-specific operations that are not included in the driver syntax and not supported elsewhere. With these commands, an application has no limitations on the amount of control it has over a device. The raw commands can be formatted exactly as specified in the device documentation. The command bracketing will be inserted by the driver if required (e.g., <stx> and <etx> will be inserted for certain devices). Three commands are defined for this type: **`rawsend`** and **`rawrecv`**, used to send and receive data directly to the device, and **`rawxact`**, a transactional version that is a combination of the first two.

A small set of interactive and non-interactive commands is all that is required for an application to perform the most common tasks with these devices. Device-specific or raw commands should rarely ever be needed.

TYPICAL OPERATION

This section describes a typical pattern that an application developer may use to operate a device. Although it is the most typical pattern, it is by no means the only viable one. Refer to Section 5, Example Applications, to see how to use the drivers in various applications.

Open a device

Access to the device is obtained by opening the **com*xx***: port that the device was installed as. This is not the hardware port that the device may be attached to, but a *virtual* **com*xx* :** port presented by the driver (e.g., COM5 or higher). A handle is returned by the *open* function and is required for all subsequent interactions with the driver. When opened, the driver initializes itself and, where required, the device.

Some drivers support automatic settings. In this mode, the driver first attempts to communicate with the device at the previous setting or at the default setting if it is the first time. (The setting for the initial attempt is grayed out in the manual settings fields.) If the driver does not receive a response, it will adjust the settings and try again. This sequence continues until the device responds or until all possible settings have been attempted.

If the driver is set for the automatic mode, it may take considerably longer for the device driver to detect an error. In particular, if the device is not connected to the specified port or if its power is off, the device driver may take several seconds attempting all possible settings before it returns an error. The application program should be tolerant of this delay.

Query the device's capabilities

The application now queries the device to determine if it can perform the required task. The capability properties (**`c_xxx`**) are provided for this purpose. For example, if an application requires the ability to read checks, it can **`get`** the **`c_check`** property to determine if the device can read checks.

Prepare the device for work

The device is prepared for operation by setting one or more of the configuration properties. Its mode of operation and other features are set up by these properties. Setting the **`capitalize`** property to **`1`** to cause all data written to or read from a card's magnetic strip to be capitalized is an example of this type of initialization. In some cases, modifying a property may cause the driver to execute functions on the device.

Use the device

The device is now fully initialized ready for operation. Because most tasks with the device require interaction with the user, the application operates the device using primarily the interactive commands. A typical scenario is when, in response to some event, the user is prompted to swipe a card by using the **`display`** command, followed by a **`read`** command to

instruct the device to return the card data when swiped. All the facilities of the driver are utilized during this stage of operation.

Close the device

When the application is finished with the device, it simply closes the port using the handle obtained when it opened it. The driver shuts down the device if required.

Note

In some cases with Windows 95, the Driver may not be closed properly. This will leave the port open and will prevent further communications with that device until the computer is rebooted.

METHODS OF ACCESSING THE DEVICE

This section describes how to use control language commands in a Visual Basic development environment using the MSComm (Microsoft Communication) component.

Obtaining access to the device

If the MSComm (Microsoft Communication) ActiveX component is used to access the device, set the **`CommPort`** property to the com port *number* of the device. Then, set the **`PortOpen`** property to **`True`** to open it. The following example shows how:

```
'set error handling
 On Error Resume Next

'open the port
 Comm.CommPort = 5
 Comm.PortOpen = True
 If Err.Number <> 0 Then
    <<process error>>
 End If
on error goto 0
```

Note

After issuing an Open command, the computer may spend several seconds attempting to communicate with the device. During this time the computer will appear to be hung up.

If file I/O access is desired, you have the option of using either the device's friendly name, such as **`\\.\micr+`** (where **`\\.\`** specifies to Windows that this is a device and not a file) or its port name, **`COM<5..15>`**. The friendly name is more intuitive and easier to remember than a port number; however, the serial method gives the programmer better control of the device. The port number can be found in the operating system's device UI. For example, open Control Panel/System/Device Manager/MagTek and select a specific driver. Under Properties, select the

Settings tab. This gives both the Friendly Name and the port name (**COM<5-15>**). It also identifies the physical port that will be used to communicate with the device.

Open the device using either of the previous names. Use whatever facility is provided by your development environment for opening files. For Visual Basic, do the following:

```
'set error handling
 On Error Resume Next

'open the port for binary access
 Open "\\.\micr+" For Binary Access Read Write As #1
 If Err.Number <> 0 Then
 <<process error>>
 End If
 on error goto 0
```

Note

The friendly name of the device, as found in the operating system's device UI (Device Manager in Windows 95, for example), must be prefixed with " **\\.** *" in order to open the device. If the previous example did not have the prefix, it would create a file named* ***micr+*** *in the current directory–clearly not the desired result.*

Interacting with the device

An application interacts with the device by sending commands to the device and reading its responses. Commands are sent by writing to the opened port and responses from the device or property requests are retrieved by reading from the port.

To interact with the device using the MSComm component, invoke a command by assigning it to MSComm's **Output** property. The response is received by MSComm's **OnComm** event handler as a **comEvReceive** event or by directly polling the port. The entire response to a command or property request is received as a single event.

```
'submit echo command
 Comm.Output = "/echo Hello" + Chr$(10)

 Private Sub Comm_OnComm()
   'return if not a receive event
    If Comm.CommEvent = comEvReceive Then
       'process received data
       a$ = Comm.Input     'get echo data
    Else
       <<process non-read event>>
    End If
 End Sub
```

If using file I/O access, interaction with the device is indistinguishable from writing to or reading from a file.

```
'set up error handling
On Error Resume Next

'submit echo command
Put #1, , "/echo Hello" + Chr$(10)

'declare an input buffer
a$ = String(2000, Chr$(0))

'read echo response from device
Get #1, , a$
If Err.Number <> 0 Then
   <<process error>>
End If
```

Note

File I/O interaction with the device is synchronous; the read operation will block until a response is received from the device or is returned by the driver (as in a property request). This means that a read command cannot be canceled because the computer will not accept any new commands while one is pending. The only exception to this is when the development environment provides access to the Win32 API, giving the application the ability to use overlapped file I/O.

Releasing access to the device

Releasing access to the device is very simple. If using MSComm, close the device by setting its **PortOpen** property to **False**:

```
  'close the port
mscomm1.PortOpen = FALSE
```

If opened as a file, close it as in the following:

```
'close the port
Close #1
```

ERRORS AND ERROR PROCESSING

A command's execution status is returned to an application in the command's response, if it has one. The status value is a two digit numeric field located at positions 23 and 24 of the response (refer to Appendix C. Status Codes for a description of all error conditions) .

Errors are processed differently for property manipulation. If an error occurs while getting a property, the response will be returned with an empty property value. No status is returned when setting a property because the **set** command has no response defined for it.

If a command returns a non-zero status, indicating an error, an application can typically respond in the following manner:

1. It can prompt the user to repeat the action and re-submit the command. This is typical if the status does not indicate a failure, per se, but that the device may not be ready yet or first needs some other interaction by the user.

2. It can reset the device and prompt the user to repeat the action. Typically, this action is necessary if the device's state or configuration has been corrupted, but is otherwise functioning correctly.

3. Finally, the application can refuse to continue operation of the device. An application should do this only if the returned status indicates that the device is malfunctioning.

HANDLING SPECIAL COMMANDS

Some devices such as the IntelliPIN PLUS support a set of commands that are not standard and/or do not follow the usual protocol. The ***Generic Driver*** can be used to support these commands. It does not know how to communicate with any device and does not support any protocol. The *Generic Driver* allows the application to send any string to a device. When the *Generic Driver* is used, the application must form the command, insert packet characters, and compute a check character where required.

In particular, the IntelliPIN PLUS supports a set of commands that require <SI> and <SO> as command brackets instead of the usual <stx> and <etx> characters. These special commands cannot be used with the IntelliPIN PLUS drivers. If the <SI>/<SO> commands are required in an application, the *Generic Driver* can be used to formulate the commands and recognize the responses.

For example, in order to select one of the Multi-Master keys in the IntelliPIN PLUS, the format of the command is:

```
<SI>08[address]<SO>{LRC}
```

The IntelliPIN PLUS driver cannot generate this command since all commands supported by the driver begin with <stx> and end with <etx>. To solve the problem, open the *Generic Driver* and send the following command to select master key number 3:

```
/rawsend \x0F083\x0E\x35
```

where <SI> is 0x0F and <SO> is 0x0E.

The *Generic Driver* can be used whenever a deviation from the standard protocol is required or when no protocol exists at all. However, the *Generic Driver* does not support any properties like all of the other drivers. It is only available to support those cases that cannot be handled with the standard drivers.

FILE PROPERTIES

When updating the MagTek Device Drivers, discussing performance characteristics, or reporting errors, it will be important to identify the part number and version of the associated file(s). In order to determine which version is installed, use Windows Explorer and go to the \Windows\System directory. Right click on the associated "VXD" driver file (see Appendix A.

Installation And Setup) and select *Properties*. Click on the *Version* tab. Note the *File Version, Part Number,* and *Description*.

INSTALLATION

The drivers are installed by means of the Windows “Add New Hardware” facility in Windows 95/98/ME and the "INF" installation feature in Windows NT/2000/XP. Refer to "Appendix A. Installation And Setup" for a full description of the installation procedure.

MICR Format Numbers

In order to retrieve the built-in check properties (chk_***), the driver automatically configures the MICR units to format number 6500. However, there are some cases, especially outside the United States, where the check information is not consistent with format number 6500. In these cases, the installer has the option of modifying the format number string in the OEMSETUP.INF file.

The format number can be changed to another value (e.g., 7700 to allow use of a flex format) by editing the field following the format number entry (%CheckFormatCodeName%) in the OEMSETUP.INF file. This must be changed in three places depending on which drivers are to be used (MICR+, MiniMICR RS232, and MiniMICR Wedge). By defining a flex format that would duplicate the 6500 output format, the driver will still be able to parse the check data and present the individual properties (e.g., chk_account, chk_amount, chk_number, and chk_transit). If a suitable format cannot be developed to present the individual properties, the driver will still be able to present the check data (chk_data) as received from the MICR reader. If the existing format number in the MICR device is suitable, set the %CheckFormatCodeName% entry to null (i.e., “”), so it will not be modified by the Driver.

Refer to the appropriate MICR Technical Reference Manual for more information about the use of format numbers and available MICR fields.

SECTION 2. PROPERTIES

This section lists the properties that are used in the MagTek Drivers. Properties can be interrogated by issuing a **`get`** command and modified with a **`set`** command. Refer to Section 3. Commands for complete description and examples of all commands.

The **`c_xxx`** properties are set by the driver and reflect the device's **capabilities**. However, the **`c_xxx`** properties **do not** indicate the configuration of the device. For example, a device may be capable of reading all three magnetic tracks but be configured to only read two tracks or a MICR reader, while often configured with a magnetic stripe reader, may not have an MSR installed. Unless otherwise noted, **`1`** means the capability is available, **`0`** or **null** (i.e., the value is not present) means that the capability is not available.

In this table, the *Access* information indicates whether the property can be modified (Read/Write –R/W) or merely accessed (Read Only–R).

Property	Access	Description
`account_no`	R/W	Cardholder account number, including check digit. It is set by the application to be used in PIN encryption commands (IntelliPIN).
`amount`	R/W	Transaction amount in cents, without punctuation (IntelliPIN).
`applied_fmt`	R	Indicates which format template was used to parse the magnetics data. If no template or rule is applied, this property returns a null.
`c_card_stat`	R	**`1`** indicates that the driver supports retrieval of card sensor status (e.g., PPINSERT)
`c_cardwpin`	R	**`1`** if the device supports reading of a card and a PIN in response to a single command (e.g., IntelliPIN).
`c_check`	R	**`1`** if the device can read checks (e.g., MICR devices).
`c_events`	R	**`1`** indicates that the driver supports unsolicited event notification (e.g., PPINSERT).
`c_keypress`	R	**`1`** if the device supports retrieval of a key press (e.g., IntelliPIN).
`c_keystring`	R	**`1`** if the device supports retrieval of a sequence of key presses (e.g., IntelliPIN).
`c_magnetic`	R	**`1`** if the device can read magnetic cards.
`c_mechanics`	R	This value indicates how the card reader's mechanism operates: **`0`** – manually operated device or no card reader **`1`** – device is mechanized and supports "eject" **`2`** – device is mechanized and supports "eject" and "confiscate"
`c_pin`	R	**`1`** if the device supports reading of PINs (e.g., IntelliPIN).
`c_smart`	R	**`1`** if the device supports smart cards.
`c_tracks`	R	A three-character string, representing the tracks supported by the device. The left-most position indicates track 1. Thus **`110`** indicates that the device can access tracks 1 and 2 but not track 3. See **`trk_enable`** to determine which tracks are enabled.

Property	Access	Description
c_write	R	**1** if the device can encode a magnetic card in either LoCo or HiCo; **2** if the device can encode a magnetic card in only the setting indicated in **wr_coer**
c_wr_secure	R	**0** if the device does not support secure mode; **1** if the device can switch between secure and non-secure mode (see **wr_secure**); **2** if the device only operates in the secure mode.
capitalize	R/W	Set this to **0** to prevent the driver from capitalizing the data for the **read** and **write** commands. The default value for this property is **1** (enable capitalization).
card_stat	R	Current card sensor status: **0** = not blocked, **1** = blocked (PPINSERT).
chk_account	R	Check account number from check (MICR).
chk_amount	R	Check amount from check (MICR).
chk_bankid	R	Bank ID number from the transit field (MICR).
chk_data	R	Output data string as received from MICR reader (MICR).
chk_format	R/W	Indicates the format of the check data. Set to **6500** by default. If this property is modified by the application, the **chk_xx** properties (except **chk_data** and **chk_status**) will be set to null. (MICR)
chk_mod10	R	Mod10 check digit from the transit field (MICR).
chk_number	R	Check number (MICR).
chk_routing	R	Routing number from the transit field (MICR).
chk_status	R	2-digit status code from the check just read (MICR).
chk_transit	R	Transit number from check (MICR).
cmd_pending	R	Command pending–indicates which command, if any, is pending. If none is pending, the second argument will be null: **/get cmd_pending***<LF>*
dblpinentry	R/W	Set to **1** to enable double PIN entry such as when requesting a new PIN; set to **0** when verifying a customer's PIN (IntelliPIN).
dev_status	R	Device status. **0** means device is connected and operational. Any other value indicates a device-specific error. If the device fails to respond, a null value is reported: **/get dev_status***<LF>*
dev_version	R	Device version string. This value is read directly from the device, if the device supports a version string. *<CR>* characters in the string read from the device will be replaced with **/**. This property will be useful in reporting operational problems to MagTek.
enable_cmc7	R/W	Set this property to **1** to enable CMC-7 characters decoding, **0** to disable it. This is used for international checks; see MICR manual for more information. (MICR)

Property	Access	Description
enc_key	R/W	Encryption key to use for the next encryption process (IntelliPIN): **M** for Master key **S** for Session key **0-3** for lower working keys **A-J** for upper working keys
enc_key_sn	R/W	Serial number of encryption key. Used to specify key serial number for activating/deactivating PIN encryption in MSK mode and to return the key serial number in DUKPT mode. The key serial number is specified in clear text (IntelliPIN).
enc_mode	R/W	Current encryption mode – **msk** or **dukpt** (IntelliPIN).
entry_echo	R/W	Specifies how to display the characters when entered from the keypad on the LCD screen (IntelliPIN): • empty value to display as entered • (minus) to suppress display • $ to display as amount The value of this property affects the operation of the **read key_string** command. By default this property is empty.
entry_len	R/W	Maximum number of characters (1-32) to be collected with the **read key_string** command. An empty value (default) for this property converts to a length of 1. (IntelliPIN)
entry_tout	R/W	Entry timeout: number of seconds (15-255) to wait for keypad input. (IntelliPIN)
events_on	R/W	Set to **1** to enable unsolicited event notifications. The default is **0**. (PPINSERT)
invalcmdrsp	R/W	Invalid command response: set to **1** to enable responses to invalid commands (useful during program development). This is set to **0** (disabled) by default.
key_parity	R/W	Set to **1** to enable parity check on encryption keys. (IntelliPIN)
lasterr	R	Status from the last command sent to the driver. A successfully executed command will reset this value to **0**. This property is useful for checking the operation of the **set** commands. After each **set,** the response to **get lasterr** should be **0**.
max_pin_len	R/W	Maximum PIN length (IntelliPIN): • 1 – 16 for **ibm** format (IBM 3624) • 4 – 12 for **ansi** format (ANSI 9.8)
msg1 - msg4	R/W	Messages to show on LCD screen with various commands. **msg1** – used by the **read** and **display** commands **msg2** – used by the **display** and **read pin** commands **msg3** – used by the **read pin** command **msg4** – used by the **key_press** and **key_string** operations To specify leading spaces, use **\x20**. See the **display** command for more information. (IntelliPIN)

Property	Access	Description
offline_enc	R/W	Set to **1** to enable encode capability in standalone mode with keyboard; **0** prevents standalone encoding (MT-95).
oper_tout	R/W	Operational timeout in seconds (15-255). (IntelliPIN)
pin_blk_fmt	R/W	PIN block format (IntelliPIN): **ansi** (ANSI 9.8) or **ibm** (IBM 3624)
pinfilldig	R/W	PIN fill digit (0..9, A..F) when **pin_blk_fmt** is **ibm** (IntelliPIN)
port_name	R	Indicates the virtual port number (e.g., COM6) derived from the friendly port name.
pwroffdelay	R/W	Power off time delay in minutes (5-255). (IntelliPIN)
s_down_tout	R/W	Shutdown timeout in hours (1-31). Set to **0** to disable. (IntelliPIN)
track1ss **track2ss** **track2ss**	R R R	Indicates Start Sentinel on Track 1 as received from the device. Indicates Start Sentinel on Track 2 as received from the device. Indicates Start Sentinel on Track 3 as received from the device.
trivpinchk	R/W	Set to **1** for trivial PIN checki.e., don't allow 1234. (IntelliPIN)
trk_enable	R/W	Enable reading and writing of individual tracks. The value of this property is a string of three characters, with **0** representing disabled tracks and **1** representing enabled tracks, e.g., **110** enables tracks 1 and 2 and disables track 3.
trk1data	R	Data from track 1 excluding start sentinel and end sentinel.
trk2data	R	Data from track 2 excluding start sentinel and end sentinel.
trk3data	R	Data from track 3 excluding start sentinel and end sentinel.
visa_mac1 **visa_mac2** **visa_mac3**	R	Message authentication codes returned by device after PIN is collected (DUKPT mode only). (IntelliPIN)
wr_coer	R/W	Encode Coercivity Mode (MT-95). Specifies the energy level used to encode the magnetic stripe: **0** = automatic selection **1** = LoCo only mode **2** = HiCo only mode
wr_secure	R/W	**0** indicates the card can be removed between a read and write operation. Set this to **1** to turn on secure online encode mode (MT-95).
xact_type	R/W	Transaction type – **d** = debit, **c** = credit (IntelliPIN).

SECTION 3. COMMANDS

This section describes all of the commands that can be used with the MagTek Windows Device Drivers. Some commands require parameters to indicate to the driver exactly what function is to be performed. While there are a few device-specific commands, most commands can be used with any device.

DATA FORMAT

All commands sent to the driver and all responses received are strings of printable ASCII characters delimited by *<LF>*. The driver will also accept *<CR>* as a delimiter. All command and response strings begin with the character **/**. If a command has arguments, they should be separated with one or more white spaces. The driver accepts space *<SP>* and *<TAB>* as white space characters.

Note

A command delimiter sent immediately after the previous command delimiter is interpreted as an empty command and is ignored by the driver.

RESPONSES

All responses to the transaction commands are formatted with fixed fields, to allow them to be parsed either by scanning for white spaces or by using constant offsets into the response string. In the descriptions of the commands found later in this section, the arguments sent with the responses are shown in their respective locations but may not indicate the exact number of spaces. The actual responses are sent in a fixed-field format, as shown in the following table:

Field	Offset	Size	Comment
command name	0 (0-11)	12	This field identifies the command that produced this response, e.g., **/get** is followed by 8 spaces to fill the 12 locations.
arg1	12 (12-23)	12	Fixed-size argument – value depends on the command sent. A property name is left justified in the field and begins in location 12. Status information is right justified in the field (with a trailing space) so the SS value will always be located at positions 21 and 22.
arg2	24 (24-??)	var	Variable size argument – used for responses with variable-size data, like **/get** *prop* or **read** *status data.*

Examples:

```
00000000001111111112222222222
01234567890123456789012345678 9
/read            -00082
/get        trk_enable  110
```

NOTATION CONVENTIONS

The following conventions are used in the tables that follow.

`Fixed Size (Bold)`	Used to represent literals (symbols, exactly as sent or received from driver)
Italic	Used to represent placeholders (variable fields)
[]	Expression parts in brackets are optional. The brackets are never a part of the syntax
<LF>	ASCII control character. The only ASCII control characters used are *<LF>* (0x0A) and *<CR>* (0x0D).
(a|b)	Means that the expression can be either a or b, e.g., X(1|2) means either X1 or X2. The parentheses and the | are never part of the syntax.

COMMAND DESCRIPTIONS

The following list of commands includes function, syntax, errors, remarks, and examples as applicable.

cancel

Function Cancel a command.

Syntax **`/cancel`** [*cmd*]

The optional *cmd* can be any of the transaction commands such as:

`/cancel rawrecv`
`/cancel rawxact`
`/cancel read`
`/cancel write`

If *cmd* is omitted, any pending commands will be canceled.

Errors If the specified command is not active, the command is ignored and there is no response.

Remarks The command being canceled will send a response immediately.

Example If a **`read`** command has been issued but the operation is to be aborted:

Command **`/cancel read`***<LF>*

Response **`/read    -00082`***<LF>*

display

Function Show a single message or two alternating messages on the device's display.

Syntax **`/display`** *[x]*

The optional argument *x* indicates the message to be displayed.

Errors *none*

Remarks If the optional argument *x* is provided, this command displays it as a single message. If *x* is **@**, the driver sends a command to the device to display the idle message **00** ("Welcome"). If *x* is omitted, the command uses the values of the **msg1** and **msg2** properties for the message texts. If **msg2** is empty, this command displays the text in **msg1**; otherwise, it displays the texts in **msg1** and **msg2** as alternating messages. The message texts are displayed unmodified, except for any '\' characters, which are used as escape characters:

\r is converted to 0x0D (shown as <*CR*> in this document)

\n is converted to 0x0A (shown as <*LF*> in this document), e.g., to be used as line separator for LCD screens that can display multiple lines

\\ is converted to \

\x*hh* is converted to a character with ASCII value *hh* (always two hex digits). Not all ASCII values can be displayed.

Leading and trailing spaces are removed from the message texts in the *x* argument and the **msg1** and **msg2** properties. **\x20** may be used for adding leading spaces.

To center the message "Thank You" on the IntelliPIN LCD:

Command **`/display \x20\x20\x20Thank  You`**

Response *none*

echo

Function Echo data–driver test command.

Syntax **`/echo`** *string*

string is limited to 11 characters (the width of the 'arg1' field in the response format) without any embedded spaces.

Errors *none*

Remarks The driver responds by echoing the command back. If the command specifies a string that is longer than 11 characters or if a space appears, the response will be truncated. There is no translation for escape (\x00) commands. This command cannot be cancelled with **`/cancel`**.

Example If you wish to ensure that the driver is properly installed, request it to echo a string:

Command **`/echo Testing`**<*LF*>

Response **`/echo  Testing`**<*LF*>

event

Function Response to an unsolicited event notification.

Syntax *none*

Errors *none*

Remarks This response can occur when an unsolicited event, such as card inserted, occurs.

The format of the response is: **`/event`** *n data*

n is a numeric event code:

`1` – medium has been inserted into the reader

`2` – medium has been removed from the reader

data specifies the type of medium that was inserted/removed:

`M` – magnetic

Events are sent to the application only if the **`c_events`** property is **`1`** (driver supports events) and the **`events_on`** property is set to **`1`** by the application.

If a card has already been inserted when the driver is opened, there will not be any notification when **`events_on`** is enabled. Consequently, it is recommended that **`/get card_stat`** be issued immediately after opening the driver to see if a card is blocking the sensor.

Example If you wish to be notified when a card has been inserted into the PPINSERT:

Command **`/set events_on 1`**<*LF*>

Response **`/event 1 M`**<*LF*>

When a card is inserted into the slot.

get

Function Get a property.

Syntax **`/get`** *prop*

prop is one of the valid properties shown in Section 2 or any of those from data parsing.

Errors **`/get abc`**<*LF*>

Since **abc** does not exist.

Remarks The driver sends a response in the format: **`/get`** *prop val*.

If the requested property does not exist, the *val* field will be empty, i.e., <*LF*> follows the *prop* field. If the command was cancelled, both the *prop* and *val* fields will be empty. In some cases, this command will interrogate the device to determine the property setting. Some properties cannot be interrogated if a command (such as read) is pending. The value will be null in this case.

Example If you wish to find out which tracks are enabled, request the **`trk_enable`** property:

Command **`/get trk_enable`**<*LF*>

Response **`/get trk_enable 110`**<*LF*>

Indicating track 1 & 2 are enabled, track 3 is disabled.

load_key

Function Load an encryption key into the device.

Syntax **`/load_key`** *n key*

n can be one of the following values:

`M` – master key (*key* is in clear text)

`S` – session key (*key* is encrypted under Master Key)

`0` ... **`3`** – lower working keys (*key* is encrypted under Session Key)

`A` ... **`J`** – upper working keys (*key* is encrypted under Session Key)

key is the 16- or 32-character value of the key to be loaded.

Errors **`/load_key 30`***<LF>*

If the *n* field is invalid, *key* is the wrong length, or the device sends an error (e.g., there is a key parity error).

`/load_key 45`*<LF>*

If the required key is not loaded.

Remarks This command is used to load a key into the device. With all but the master key, the selected key is encrypted under another key so the application must know the encrypted value of the key. The response to this command is: **`/load_key`** *SS*
SS is a two digit status code; 00 – success, 30 – invalid, 45 – rejected, etc.

Example To load the session key encrypted under the master key:

Command **`/load_key S 99E1E835662DEA94`***<LF>*

Response **`/load_key 00`***<LF>*

rawrecv

Function Receive data from the device.

Syntax **/rawrecv**

Errors **/rawrecv** **45**<*LF*>

If a command is already pending.

/rawrecv **82<LF>**

If the command was canceled by the user (e.g., with CLEAR key)

Remarks This command overrides the default processing of the next message that comes from the device and returns it to the application as a **rawrecv** response. Only one message from the device will be processed in this manner, after that the driver switches to normal operation. The response to this command is in the following format: **/rawrecv** *status x*

status is a 2-digit decimal value (refer to Appendix C. Status Codes for a complete description of the status values)

x is the data received from the device with the following characters replaced:

- <*CR*> is replaced by **\r**
- <*LF*> is replaced by **\n**
- \ is replaced by ****
- any other non-printable characters are replaced by **\xhh**, where *hh* is the two digit hex code of the character.

If a **/rawsend** command is sent that will cause the device to send back a response, the application should either submit a **/rawrecv** command before sending the data with **/rawsend**, or (better) use the **/rawxact** command.

Note

In some cases, the framing characters in the response are extracted by the driver and are not presented to the application.

Example To receive card data when the IntelliPIN is operating in the VeriFone mode:

Command **/rawrecv**<*LF*>

Response **/rawrecv** **00** **;12345?**<*LF*>

rawsend

Function Send arbitrary data to the device.

Syntax **/rawsend** *x*

x is an arbitrary string which is transmitted directly to the device. The string *x* is passed as-is to the device, except for '\' which is used as an 'escape' character:

- **\r** is converted to <*CR*>
- **\n** is converted to <*LF*>
- \\ is converted to \
- **\x*hh*** is converted to a character with ASCII value *hh* (always two hex digits), e.g., **\x20** is converted to a space.

Errors *none*

Remarks This command as with the other **raw** commands supports any features that have not been implemented in the standard set of commands. **Note**: the driver inserts appropriate framing characters, e.g., <stx> and <etx>.

Example To change the default message 00 to show "Welcome to Our Bank" on two lines of the IntelliPIN:

Command **/rawsend 5100Welcome to\x1COur Bank**<*LF*>

Response *none*

rawxact

Function Execute a send/receive transaction with the device in raw mode.

Syntax **/rawxact** *x*

x is an arbitrary string which is transmitted directly to the device. The string *x* is passed as-is to the device, except for '\' which is used as an 'escape' character:

- **\r** is converted to <*CR*>
- **\n** is converted to <*LF*>
- \\ is converted to \
- **\x*hh*** is converted to a character with ASCII value *hh* (always two hex digits), e.g., **\x20** is converted to a space.

Errors **/rawxact 45**<*LF*>

If a command is already pending.

/rawxact 82<*LF*>

If the command was canceled by the user (e.g., with CLEAR key)

Remarks This command is a combination of **/rawsend** and **/rawrecv**. It sends the supplied data to the device, overrides the default processing of the next message that comes from the device and returns it to the application as a **/rawxact** response. After the response is returned (or canceled), the driver switches to normal operation. The syntax for this command is identical to the syntax of the **/rawsend** command; the syntax of the response is identical to the **/rawrecv** response.

Example To load a master key of 23AB4589EF6701CD into the IntelliPIN:

Command **/rawxact 9423AB4589EF6701CD**<*LF*>

Response **/rawxact 00 940**<*LF*>

read

Function Read data from the device.

Syntax **/read** *[[x] y]*

The optional argument *x* specifies the data source; if *x* is missing, a card will be read. Refer to the Read Argument table below for a description data sources.

The optional argument *y* is used to specify a message to be displayed on the LCD screen, if supported, before carrying out the command. If *y* is omitted and the device supports a display, the text in the **msg1** property is shown. In order to use *y*, the *x* argument must be present. See the **display** command for the description of the message format for *y*.

Errors **/read -00045**<*LF*>

If a command is already pending or the **enc_key** is not defined for **read pin**.

/read -00082<*LF*>

If the command was canceled by the application (82) or by the user (83) (e.g., with CLEAR key).

Remarks The response to this command has the following format: **/read** *status data*

The *status* field is a 6-character string aligned to the right in the arg1 field. It is formatted as follows: $TX_1X_2X_3SS$

T defines the type of data that was read:

C = a check was read
M = a magnetic card was read
P = a PIN was read
K = a key press or string was read
- = indeterminate: no data was received from the device. Returned on errors not specific to the data type, such as command canceled (SS=**82**).

X_i define a media-specific status. For checks, this is the decimal representation of the check read status, as defined in the MICR specification. For magnetic cards, XXX indicates the read status for each of the three magnetic tracks (see **card** in the Read Arguments table below for a description of the status). For PIN data this status is always **000**; for keypress and string data, *XXX* is the data length in characters.

SS is a two-digit status code. **00** indicates a good read (but some tracks may be bad); any other status code indicates an error. These error codes indicate an error in the communication between the driver and the device or driver's internal errors. Read errors are reported in the X_i fields and do not cause the SS field to be set to a non-zero value. See Appendix C. Status Codes.

The *data* format is described in the **write** command below.

Example To request an amount to be entered by the customer on the IntelliPIN:

Command **/set entry_len 6**<*LF*>
/read key_string Enter the amount<*LF*>
Response **/read K00300 123**<*LF*>

Example To read a card (from any device):

Command **/read card**<*LF*>
Response **/read M10900 ;12345?**<*LF*>
track 1 error, track 2 good, track 3 blank

Read Arguments

The optional argument *x* used in the **read** command specifies the type of data to read and *y* specifies the text to be displayed. The following table describes the recognized *x* arguments for the **read** command:

Read Argument	Description
any	Read any type of data. This option is equivalent to **read** without any arguments.
card	Read magnetic stripe card. Display message (**msg 1**) if defined. When the user swipes a card, the response will be in the following format: **/read M***$X_1X_2X_3$SS data* *X_1* define the track read status for each of the three tracks, as follows: 0 = good track 1 = bad track 9 = no track data. *SS* is a two-digit status code; it is not affected by errors reported in the *X_i* field: 00 – successful read 82 – canceled, etc. *data* is the card data for all successfully read tracks.
card_w_pin	Read magnetic stripe card and collect PIN from cardholder. Display messages if defined. This command is similar to the **read card** command except that after the card is swiped, the device collects and stores the cardholder's PIN. The PIN can be collected later by issuing the **read pin** command. Before issuing this command, the following properties may be set: **msg1, msg2, msg3** – messages to be displayed while waiting for card swipe and PIN entry (a default message will be used if these properties contain empty strings). The response to this command is identical to the **read card** response; if successful, it returns the track data from the magnetic card. If the response status *SS* is **00**, the **read pin** command can be used to collect the PIN.
check	Read check data. When the user reads a check, the response will be in the following format: **/read C***$X_1X_2X_3$SS data* *XXX* is the decimal representation of the check read status, as defined in the MICR specification, e.g., **004** indicates a bad character in the check number field. *SS* is a two-digit status code: 00 – successful read, 82 – canceled, etc. This status is not affected by errors reported in the XXX field. *data* is the check data, which is also available in **chk_data**. The data format depends on the setting of the **chk_format** property.

Read Argument	Description
`chk_or_card`	Read magnetic stripe card or check data. When a card or check is swiped through the device, the driver sends the respective response.
`key_press`	Display a message (`msg4`) on the LCD screen, if available, and wait for a key on the keypad to be pressed. The device will wait for `entry_tout` seconds for the key press (by default 0 for no timeout). The response to this command is: `/read K`*XXXSS K* *XXX* is the number of keys collected. Always 001 on successful read, 000 if failed. *SS* is a two-digit status code: 00 – successful read, 81 – timeout, etc. *K* is the ASCII representation of the pressed key (if *SS* is **00**).
`key_string`	Display a message (`msg4`) on the LCD screen, if available, and collect a string of key presses (digits) from the device. The following properties affect this command: • `entry_tout` – number of seconds to wait for input (by default 0 for no timeout) no timeout) • `entry_echo` – how to display the characters entered from the keypad on the LCD screen: empty value to display as entered, `-` (minus) to suppress display, `$` to display as amount. Empty by default. • `entry_len` – maximum number of characters to be collected. An empty value for this property is interpreted as a length of 1 by the device (default). The response to this command is in the following format: `/read K`*XXXSS data* *XXX* is the data length in characters *SS* is a two digit status code: 00 – successful read 81 – timeout 83 – input aborted, etc. *data* is the string collected from the device.

Read Argument	Description
pin	Collect PIN from cardholder and read PIN data from the device. The following properties may be set before issuing this command: • **account_no** – cardholder account number, including check digit, if required • **amount** – transaction amount in cents, without punctuation, if required • **enc_key** – (MSK mode only) encryption key to use: **M** for master, **S** for session, **0-3** for lower working keys, **A-J** for upper working keys. • **xact_type** – (DUKPT mode only) transaction type: **D** for debit, **C** for credit The response will be: **/read P000***SS pin_block* *SS* is a two-digit status code: 00 – successful read 45 – **enc_key** is not defined 83 – aborted, etc. *pin_block* is the encrypted PIN block as returned by the device. Upon successful read, the following properties will be set: • **Visa_mac1, visa_mac2, visa_mac3** – message authentication codes (DUKPT mode only) • **enc_key_sn** – serial number of encryption key (DUKPT mode only)

reset

Function Reset the device.

Syntax **`/reset`**

Errors *none*

Remarks Clear any pending operations and reset the device to initial state (for mechanized card devices this command will also eject the card). This does not affect any of the properties.

Example To return a device to its initial state:

Command **`/reset`**`<LF>`

Response *none*

set

Function Set a property.

Syntax **`/set`** *prop val*

prop is one of the valid properties (R/W) shown in Section 2. Properties

val represents the value of that property.

Errors *none*

Remarks This command is used to define each of the properties that are required prior to sending a command.

Example To load the key serial number in the IntelliPIN:

Command **`/set enc_key_sn 0123456789012345`**`<LF>`

Response *none*

ver

Function Read driver version.

Syntax **`/ver`**

Errors *none*

Remarks The response to this command is sent in the following format: **`/ver`** *num text*

num is the driver's part number

text is a free format version string. It may contain a tagged-format data enclosed in parentheses, as shown in this example

This **is not** the version of the device.

Example To determine the version of the currently active driver:

Command **`/ver`**`<LF>`

Response **`/ver 30037395 Mag-Tek Device Driver (Version=1.04 Model=IntelliPIN)`**`<LF>`

write

Function Data encode command.

Syntax **/write** *data*

Errors

/write **94**<*LF*>
Encode is not supported on this device.

/write **34**<*LF*>
The *data* field was in the incorrect format.

/write **82**<*LF*>
The **write** command was canceled.

/write **45**<*LF*>
Device in wrong mode (e.g., if **/read** already issued)

/write **60**<*LF*>
Error during write operation (e.g., on MT-95)

Remarks The *data* field is in the following format:

[**%***an-data***?**][**;***n-data***?**|**@***a-data***?**][(**+***n-data***?**|**#***an-data***?**|**!***an-data***?**|**&***an-data*)]

an-data is alphanumeric data (ASCII characters ' ' to '_' (0x20 to 0x7f))

n-data is numeric data (ASCII characters '0' to '?' (0x30 to 0x3f))

The data should not contain the end sentinel character (**?**).

If the application sends data for an alphanumeric track that contains lowercase characters (ASCII values beyond 0x60), they will be capitalized if **capitalize** = **1**. To disable this and send the data as-is to the device, set the **capitalize** property to **0**. The three sub-sections of the data string represent the three tracks on the magnetic card. The data for each track begins with a start sentinel character, which defines both the track number and the data format for the track:

%	identifies track 1 (7-bit alphanumeric)
;	identifies track 2 (5-bit numeric)
@	identifies track 2 (7-bit alphanumeric)
+	identifies track 3 (5-bit numeric)
!	identifies track 3 (CA Driver License)
#	identifies track 3 (alphanumeric, AAMVA)
&	identifies track 3 (7-bit alphanumeric)

Note that any or all of the data may be missing, but the order of the data for the tracks must always be in order (1, 2, 3). A missing track is interpreted as "don't write" for the data encode command – that track will not be overwritten by the encode operation.

The response sent for this command is: **/write** *status*.

status is **00** if the encode succeeded and non-zero if it failed.

See the definitions of the status values in Appendix C. Status Codes.

Example Encode tracks 1 and 2:

Command **/write %B12345^TEST^0000?;12345?**<*LF*>

Response **/write** **00**<*LF*>

SECTION 4. MAGNETIC CARD DATA PARSING

This section describes the flexible data parsing language to be used by the MagTek device drivers to parse specific fields from magnetic card data and expose those fields as properties which may be retrieved by an application using the **/get** command. The data parsing language is flexible in that it can define both standard and custom formats to be parsed by the driver.

GOALS

For most MagTek devices, the MTD drivers completely hide the device-specific commands and peculiarities, thereby allowing applications to use the same command set and logic for all devices.

Up to this point, the above mentioned encapsulation has not been applied to the data returned by the device when a magnetic card is swiped. It has been left to the application to interpret the card data. This can become troublesome because the track formats and or/data contained on each track vary depending on the type of card (e.g., ATM or Drivers License).

The goals for the flexible data parsing are:

- easy to specify formats
- allow parsing of standard formats
- allow extending formats with custom fields
- allow detection of format and applying different parsing
- allow for missing tracks and missing fields by setting the corresponding property to empty
- allow presets to be loaded from the registry
- to expose parsed fields to applications via the **/get** command
- allow MagTek or system integrators to define formats in the driver installation file (OEMSETUP.INF).

ASSUMPTIONS

- The driver validates the format template and rules for syntax, but it cannot validate the format string for correctness in relation to parsing the fields of data. For example, if the format string specifies that a field has a fixed size of 3 and it actually has a fixed size of 4, the driver will not detect this.
- There is no backward parsing (i.e., field identifiers come before the field). For example, if A identifies an account number, it cannot follow the account number (e.g., 12344556A). It must come before the account number (e.g., A12344556).
- Beginning and end sentinels are specified in the format string for magnetic data formats.
- The terminating separator that follows a variable length property field is included in the format string as a literal.
- There are no parsing interdependencies between fields of data and/or format rules.

- Property names specified in format rules are 11 characters or less, consisting of alphabetic characters, digits, and '_'. The property name begins with an alphabetic character.
- Properties used in format strings do not conflict with properties defined by the driver. If there is a duplicate property (e.g., dev_version) specified in the format strings, the driver will return the value of the parsed property rather than the device version string.
- Magnetic stripe formats are comprised of the following types of fields.

Format Code	–	One or two characters specifying the format of the data to follow
Field Separator	–	Used to delimit fields of data
Fixed-Size	–	Data field which is fixed-length
Variable-Size	–	Data field which is variable-length and is terminated by a field separator
Optional	–	The data is either a fixed-size field or a field separator (if the field is not present)

DESCRIPTION

The MTD driver supports up to 8 different card formats. Each format consists of a name, a template, and a set of rules. There may be multiple rules for a single template, but there can only be one template per format name. The *name* identifies the format. The *template* provides a high-level format to which the data is to be compared so as to determine if the rules for the format in question should be applied. The *rules* are specific format strings that specify how to parse the data and the properties into which the parsed data is to be stored.

When the driver applies a format, it will make that knowledge available to an application through a property which can be retrieved with the /**get** command.

The driver may be parameterized with the formats via values in the device's software key in the registry. The following REG_SZ registry values are supported where *x* is a number 1-8.

fmt*x*_name	name for format
fmt*x*_template	format template
fmt*x*_rules	one or more comma-delimited rules

When the driver receives data from the device, it attempts to match the incoming data to one of the templates. If a template matches, the driver attempts to parse the data using one of the rules corresponding to the matched template. It sequentially attempts to apply each rule in the order that it appears in the fmtx_rules property. If the driver cannot apply any of the rules, the driver attempts to match the data to the next template and apply its rules until it either successfully applies a rule or runs out of templates.

If the driver is successful in applying one of the rules, the name of the applied format is available in the property **applied_fmt.**

LANGUAGE FORMAT

Format Name
(fmtx_name)

The format name specifies an identifier by which to identify the format template and/or rules being applied. The maximum length of this property is 11 characters. The names can be repeated on subsequent templates.

Format Template
(fmtx_template)

The format template provides a high-level structure to which the incoming data must conform in order to apply the format's rules. It is formed by concatenating characters and asterisks contained in angle brackets (<>) or parenthesis. The *format template* string cannot exceed 63 characters. The following is an example:

%<*>?;59<*>?(!|#)<*>?

The above template specifies that if track 1 exists; the first two characters following the start sentinel of track 2 are "59"; and the start sentinel character for track 3 is either '!' or '#' then the rules for this template should be applied.

The <*> symbol specifies a don't-care situation. All data up to the character following the <*> in the template string is ignored when evaluating the data against the template. All other characters in the template string must be matched with the data.

Format Rules
(fmtx_rules)

The format rules property specifies one or more rules that describe how the data is to be parsed. It is a comma-separated string of rules where each rule has the following format:

{<rule>}

Because the '{' and '}' characters are used to delimit each rule and specify optional tracks, these characters cannot be specified as literals within the rule.

A format rule describes how the data is to be parsed. Characters that must be matched as literals are placed as is in the string or preceded with a '\' if the character is one of the following: '[', ']', '(', ')', '*', '_', '<', '>', ':', '.', or '\'. Fields that are either to be parsed or ignored are contained within <>. The *format rules* string cannot exceed 1027 characters. The following is an example for retrieving the customer name and account number from track 1:

{%B<acct_no>^<cust_name>^<*>?}

The '%' specifies the start sentinel and 'B' specifies a format ID for the track. These two characters must be matched for the remainder of the rule to be executed. <acct_no> specifies that all data up to the following '^' should be stored in a property named "acct_no". <cust_name> specifies that all data up to '^' should be stored in a property named "cust_name". <*> specifies that the remainder of the track data up to '?' should be ignored.

The following table describes the procedure for specifying fields. Remember that property names can have a maximum of 11 characters.

Note

If there is a property specified more than once in a rule, the last successful match will be saved in the property. The driver will ignore previous matches and the value will not be compared to the previously saved value for consistency.

Field Type	Example	Description
Variable size field	<acct_no>	All data up to the next field separator or end sentinel is stored in a property named "acct_no".
Fixed size field	<exp_date[n]>	Store the next *n* characters in a property named "exp_date".
Variable size field with limit	<cust_name[x..y]>	Store at least *x* characters and at most *y* characters up to the next field separator or end sentinel into property named "cust_name".
Variable size (ignore)	<*>	Ignore all characters up to the next character specified in the format string (usually a field separator) or the end sentinel character (?).
Fixed size (ignore)	<*[n]>	Ignore the next *n* bytes.
Variable size with limit (ignore)	<*[x..y]>	Ignore at least *x* characters and at most *y* characters up to the next literal found.

Field Type	Example	Description
Literal	^	A literal is placed in the string as is and is used to determine if a particular format should be applied and to mark the end of a variable-length field.
Non-ASCII literal	\r, \n, \\, \xhh	Specify an escape character or non-ASCII character. • **\r** is converted to <CR> • **\n** is converted to <LF> • **** is converted to **** • **\xhh** is converted to a character with ASCII value *hh* (always two hex digits).
Optional choice	(x|y|…)	The field specifies a choice where the data can be either a literal or a property field. There may be any number of literals specified but there may not be more than 1 property field, for example (=|<country_code[3]>). If the character is a '=', skip it; otherwise store the next three characters into a property named "country_code".
Optional field	[x]	Specifies an optional sequence that may or may not be present in the data. *x* may be one or more literal fields, property fields, or optional choice fields.
Optional track	{xy}	The data parser will not enforce that the track be present in the data when attempting to match the data to the template or rule. *x* must be a literal field or an optional choice field containing a literal. *y* may be any sequence of fields except for another optional track field.

There can be more than one rule specified for a particular format template. The rules should be placed in a single string enclosed in curly braces (i.e., '{' and '}') and delimited with commas ','. When the driver applies rules for a particular template, it sequentially attempts to apply each rule in the order it is provided in the fmtx_rules string. For example: "{rule 1},{rule 2},{rule 3}" would cause the driver to first try to apply rule 1. If the incoming data did not match rule 1, the driver attempts to apply rule 2 followed by rule 3 if rule 2 fails. If no rules can be applied, the driver attempts to match the incoming data to the next template.

The property name can also contain a modifier at the end preceded by a ':' which specifies the type of data to store in that property. For example <cust_name:A> specifies that customer name should contain alphabetic characters, spaces, and punctuation. The modifier may also be used with ignore-fields (i.e., <*>). If no modifier is provided, any type of characters is assumed. The set of supported modifiers is described in the following table:

Modifier	Description
A	Alphabetic characters (A..Z a..z), space, and punctuation (. , : ') are allowed.
D	Numeric characters (0..9).
N	Alphanumeric characters. This is the union of A and D.
\xhh	**\xhh** is converted to a character with ASCII value *hh* (always two hex digits). Only this character is allowed in the field. This modifier is only valid for "ignore" type fields.
*	Any character is allowed (default if no modifier supplied).

DEFAULT FORMATS

The MTD drivers will be assigned parameters with default formats for parsing magnetic stripe data. The formats will be placed in the INF file for the driver and written to the registry when the driver is installed. Some examples are shown below; more are included with the drivers. In these examples, spaces are inserted between fields for readability; they should not be included in the actual rules.

```
fmt1_name     "ISO59"
fmt1_template"%B<*>^<*>^<*>?;59<*>=<*>?"
fmt1_rules    "{%B<*>^<*[3]><LastName>/<FirstName>\x20<MidName>
              ^<*[7]><DiscData1>?
              ;<PAN[13..19]>=<*[3]><ExpDate[4]><SrvCode[3]><DiscData2>?},
              {%B<*>^<*[3]><LastName>/<FirstName>^<*[7]><DiscData1>?
              ;<PAN[13..19]>=<*[3]><ExpDate[4]><SrvCode[3]><DiscData2>?}"

fmt2_name     "BankCardA"
fmt2_template"%A<*>^<*>^<*>?;<*>=<*>?"
fmt2_rules    "{%A<LastName>/<FirstName>\x20<MidName>^<*>^<*[7]><DiscData1>?
              ;<PAN[13..19]>=<ExpDate[4]><SrvCode[3]><DiscData2>?},
              {%A<LastName>/<FirstName>^<*>^<*[7]><DiscData1>?
              ;<PAN[13..19]>=<ExpDate[4]><SrvCode[3]><DiscData2>?}"

fmt3_name     "BankCard"
fmt3_template"%B<*>^<*>^<*>?;<*>=<*>?"
fmt3_rules    "{%B<*>^<LastName>/<FirstName>\x20<MidName>.<Title>
              ^<*[7]><DiscData1>?
              ;<PAN[13..19]>=<ExpDate[4]><SrvCode[3]><DiscData2>?},
              {%B<*>^<LastName>/<FirstName>.<Title>^<*[7]><DiscData1>?
              ;<PAN[13..19]>=<ExpDate[4]><SrvCode[3]><DiscData2>?}"

fmt4_name     "CADL"
fmt4_template"%(C|S|D|I|R)<*>?;600646<*>?{(#|!)<*>?}"
fmt4_rules    "{%<*[1]><FirstName>\x20<MidName>\x20<LastName>[\x20]<*:\x20[0..57]>
              <Adr[29]><City[13]>?
              ;<*[6]><DLID[9]><*>=<ExpDate>=<DateOfBirth[8]>?
              {(#|!)<*[8]><State[2]><ZIP[9]><Sex[1]><Hair[3]><Eye[3]><Hgt[3]><Wgt[3]>
              <*>?}}"
```

```
fmt5_name     "AAMVA"
fmt5_template"%<*>?;<*>?{(+|%|#|!)<*>?}"
fmt5_rules    "{%<State[2]><City>^<LastName>$<FirstName>$<MidName>^<Adr>^<*>?
              ;<*[6]><DLID>=<ExpDate[4]><DateOfBirth[8]><*>?
              {(+|!|#|%)<*[2]><ZIP[11]><*[16]><Sex[1]><Hgt[3]><Wgt[3]><Hair[3]>
              <Eye[3]><*>?}},
              {%<State[2]><City>^<LastName>$<FirstName>^<Adr>^<*>?
              ;<*[6]><DLID>=<ExpDate[4]><DateOfBirth[8]><*>?
              {(+|!|#|%)<*[2]><ZIP[11]><*[16]><Sex[1]><Hgt[3]><Wgt[3]><Hair[3]>
              <Eye[3]><*>?}}"
```

In the examples for CADL (California Drivers License) and AAMVA (all other drivers licenses), the braces around the rules for track 3 indicate that track 3 is optional.

EXAMPLE

Retrieving properties from a magnetic card

In this example, the rules above have been stored in the registry by the installation script.

The following data is received from the device:

```
%B1234567890074589^SMITH/JOHN Q.MR^99121012547000000000000123?
;1234567890074589=991210112547?
```

Format 1 (ISO59) would not be applied because the first two digits of track 2 are not 59. Format 2 (BankCardA) would not be applied since there is not an 'A' following the start sentinel. However, the data fits the template for format 2 (BankCard).

The following properties and their corresponding values will be exposed:

LastName	→	"SMITH"
FirstName	→	"JOHN"
MidName	→	"Q"
Title	→	"SMITH"
DiscData1	→	"254700000000000123"
PAN	→	"1234567890074589"
ExpDate	→	"9912"
SrvCode	→	"101"
DiscData2	→	"12547"

The application receives the successful read response **/read M00900 <card data>.**

The application issues **/get applied_fmt**.
The driver responds with **/get applied_fmt BankCard.**
The application issues **/get FirstName** to the driver.
The driver responds with **/get FirstName JOHN.**

The application issues **/get LastName** to the driver.
The driver responds with **/get LastName SMITH.**

The application issues **/get PAN** to the driver.
The driver responds with **/get PAN 1234567890074589.**

The application issues **/get ExpDate** to the driver.
The driver responds with **/get ExpDate 9912**.

After all of the required properties have been retrieved, the application can place them in appropriate strings as required by the application.

SECTION 5. EXAMPLE APPLICATIONS

While each application in this section is oriented toward a specific programming language, different devices are addressed in each example. It may be useful for the reader to look at all examples to understand how the MagTek Windows Drivers can operate with various MagTek devices.

PROGRAMMING HINTS

When opening a Keyboard Wedge device, the application must wait for any key press to complete, e.g., **ALT-0**. The application should wait until all keys have been released.

VISUAL BASIC EXAMPLE

This program is a simple example of using the MagTek Windows device drivers in Visual Basic. It opens the device driver and waits for the user to click the read button. At that time, it arms the driver for the read operation and waits for a read to take place. When the check data (in the case of a MICR) is received, it displays the data and waits for the read button to be pressed again.

The user first presses the Start button to open the port. After that, the Read button is pressed to initiate a read. After the check is read, the Read button can be pressed again for another cycle. The Exit button can be pressed at any time to quit the program.

```
Option Explicit

'+-----------------------------+
'|     MTD Driver example      |
'+-----------------------------+
'| written in Visual Basic 5.0 |
'+-----------------------------+
'
' (c) Copyright Mag-Tek, Inc. 1999
' All rights reserved
'
' Mag-Tek Part Numbers:
' Source code - 30037336 REV 101
' PROG 3.5" - 30037335 REV 101

' Purpose:  This program is a simple example of using the
' Mag-Tek Windows device drivers (MTD) in Visual Basic.  It
‘ opens the device driver and waits for the user to click the
‘ read button.  At that time, it arms the driver for the read
' operation and waits for a read to take place.  When the
' check data (in the case of a MICR) is received, it displays
' the data and waits for the read button to be pressed again.

' The user first presses the Start button to open the port.
' After that, the Read button is pressed to initiate a read.
' After the check is read, the Read button can be pressed
' again for another cycle.  The Exit button can be pressed
' at any time to quit the program.
```

```
' The form needs to contain:
'  1) an "MSComm" object named MSComm1

'  2) a button named btnStart, should be set to Enabled
'     and Visible with the caption "Start"

'  3) a button named btnRead, should be set to Disabled
'     and Visible with caption "Read"

'  4) a button named btnExit, should be set to Enabled
'     and Visible with caption "Exit"

'  5) a text box named txtInfo, should be set to Visible, Enabled and
'     MultiLine containing initial text of "Click the Start button to
'     open the port"

' Note: Lines shown ending in an underscore are continuation line, i.e.
'       its one BASIC statement, split over two or more lines.
'       The underscore MUST be preceded by a space, otherwise BASIC
'       will interpret it as part if the statement and generate an
'       error.

' This is the global buffer we'll use to collect the data
Dim RcvdData$

'+---------------+
'| btnExit_Click |
'+---------------+-----------------------------------+
'| Close the com port (if open) and exit the program |
'+---------------------------------------------------+
Private Sub btnExit_Click()
  If MSComm1.PortOpen Then
    MSComm1.PortOpen = False
  End If
  Unload Me
End Sub

'+---------------+
'| btnRead_Click |
'+---------------+------------------+
'| This function does the following: |
'|  1) Disable the read button       |
'|  2) Send the read command         |
'|  3) Wait for the read response    |
'|  4) Display the read data         |
'|  5) Reenable the read button      |
'+-----------------------------------+
Private Sub btnRead_Click()
  ' Disable the read button so we don't get two read
  ' commands pending
  btnRead.Enabled = False

  ' Clear the receive buffer
  RcvdData$ = ""

  ' Send the read command
```

```
  MSComm1.Output = "/read card" & Chr$(10)

  ' If the device has check reading capability, then the
  ' following command would be used to read only the check
  ' data
  ' MSComm1.Output = "/read check" & Chr$(10)

  ' If the device can read only one media type (e.g. a
  ' card reader) then the read command "/read" command can
  ' be is issued by itself.
  ' MSComm1.Output = "/read" & Chr$(10)

  ' If the device is capable of reading more than one
  ' media type and the application is capable of accepting
  ' data from any of the media, then the read command can
  ' be issued by itself or with the "any" parameter.  (They
  ' are equivalent.)
  ' MSComm1.Output = "/read" & Chr$(10)
  '   or
  ' MSComm1.Output = "/read any" & Chr$(10)

  ' Ask the user to do the read
  txtInfo.Text = "Please swipe a card or click on Exit to quit"

  ' Wait until the card is read.
  ' In real life, the program can do other things while
  ' waiting for the data
  Do
    DoEvents
  Loop Until Len(RcvdData$) > 0

  ' Display the received data
  txtInfo.Text = RcvdData$

  ' Reenable the read button
  btnRead.Enabled = True
End Sub

'+----------------+
'| btnStart_Click |
'+----------------+-------------------------------------------------------+
'| This function does the following:                                      |
'|  1) Set up the buttons and display                                     |
'|  2) Open the device under its "friendly name" as a file                |
'|  3) Retrieve its "unfriendly name" (e.g. "COM12")                      |
'|  4) Extract the com port number from the unfriendly name               |
'|  5) Close the device (IMPORTANT: this must be done or you will not     |
'|     be able to open the device again, in any mode, without             |
'|     resetting the computer)                                            |
'|  6) Open the device under its "unfriendly name" as a serial device     |
'+------------------------------------------------------------------------+
Private Sub btnStart_Click()
  ' will hold the fully qualified name of the driver
  Dim NewName$

  ' will be used to get data from the device driver
  Dim buf$
```

```
' will hold the numeric port number
Dim PortNumber As Integer

' prevent the Start button from being pressed again
btnStart.Enabled = False

txtInfo.Text = "Please wait.  Opening the port as File IO"
txtInfo.Refresh

' declare space for an input buffer
buf$ = String(2000, Chr$(0))

' If the virtual serial port number is unknown, it can be
' obtained by opening the driver in "File" mode with
' the "Friendly Name" and asking for the virtual COM port number.
'
' The sequence is:
' 1) Open the driver as a binary file
' 2) Request the "port_name" property
' 3) Close the driver
' 4) Open the serial port using the number obtained above
' 5) Send/receive commands/data
' 6) Close the serial port when done
'
' As of release 1.08.01 of the MTD drivers,
' the default Friendly Names are:
' ------------------------------------------------------------------
' "Mag-Wedge"
' "MT-85"
' "MT-95"
' "Port-powered swipe reader"
' "Port-powered insert reader"
' "MiniWedge"
' "MICR+"
' "Mini MICR RS-232"
' "Mini MICR Wedge"
' "IntelliPIN RS-232"
' "IntelliPIN Wedge"
' "IntelliPIN MICR Aux"
' "Generic Serial (RS-232)"
' "Generic Wedge (Keyboard)"
'
' Prepend "\\.\" to the "friendly" name which
' tells Windows that this is a device name and not a file name
NewName$ = "\\.\" + "MiniWedge"

' Trap the "file not found" error if the
' device is not present or ready
On Error Resume Next

' Try to open the device, this can take anywhere from one
' second to one minute
Open NewName For Binary Access Read Write As #1

' If the driver was unable to open the device, then
' inform the user
```

```
If Err.Number <> 0 Then
   ' Process error using Err.Description
   ' contains error description for the demo,
   ' we'll just display it
   txtInfo.Text = Err.Description

   ' Reset the error handling
   On Error GoTo 0

   ' exit this sub
   Exit Sub
End If

' reset the error handling
On Error GoTo 0

' send the command to get the port number
Put #1, , "/get port_name" + Chr$(10)

' get the response from driver which should contain the
' com port number
Get #1, , buf$

' Expected response:
' (character position in the response string)
'                1111111112222222222
'       12345678901234567890123456789
' e.g. "/get        port_name   COM14"

'+=======================================+
'|| IMPORTANT: CLOSE THE DEVICE DRIVER    ||
'||            BEFORE TRYING TO REOPEN IT ||
'+=======================================+
Close #1

' Make sure we got back a valid response.
' This checks that we have received a "/get" response and that
' "port_name" and "COM" are present and in the right locations.
If Left(buf, 4) = "/get" _
   And InStr(buf, "port_name") = 13 _
   And InStr(buf, "COM") = 25 Then

   ' Just for information, display the com port number
   txtInfo.Text = "Opening Serial IO on port " & Mid(buf, 25, 5)

   ' Get the port number value from character position 28
   ' (and 29 if two digits long) of the response
   PortNumber = Val(Mid(buf, 28, 2))

   '+-----------------------------------+
   '| open the driver as a serial device |
   '+-----------------------------------+

   ' make sure the on_comm function will be
   ' triggered by the device driver by setting
   ' the receive threshold to 1 (one)
   MSComm1.RThreshold = 1
```

```
    ' Set the com port number retrieved from the response
    MSComm1.CommPort = PortNumber

    ' Open the com port and establish communications with the device
    MSComm1.PortOpen = True

    ' enable the read button
    btnRead.Enabled = True

    txtInfo.Text = "Click on the Read button to read a" _
      & "card or Exit to quit."
  Else
    ' If we got here, then the device did not open correctly
    ' as a file IO so some kind of error handling is needed
    txtInfo.Text = "Error: Got back: " & buf
  End If

End Sub

'+------------------+
'| Form_QueryUnload |
'+------------------+--------------------------+
'| When this form is closed make sure the port |
'| is closed                                   |
'+---------------------------------------------+
Private Sub Form_QueryUnload(Cancel As Integer, UnloadMode As Integer)
  If MSComm1.PortOpen Then
    MSComm1.PortOpen = False
  End If
End Sub

'+----------------+
'| MSComm1_OnComm |
'+----------------+------------------------+
'| This event is automatically activated   |
'| whenever the device driver returns data |
'| to the program                          |
'+-----------------------------------------+
Private Sub MSComm1_OnComm()

  ' If this event handler was called because data was
  ' received from the device (via the device driver), then
  ' process that data
  '
  ' In this demo, it is just stored in the "RcvdData" buffer
  If MSComm1.CommEvent = comEvReceive Then
    RcvdData$ = MSComm1.Input
  End If
End Sub
```

C++ EXAMPLE

The following is an example of C++:

```
/* ---------------------------------------------------------------------- */
/*                          TST: Test Application                         */
/*                                                                        */
/*             MTDTEST.C - Test module for Mag-Tek device drivers          */
/* ---------------------------------------------------------------------- */
/* Version 1.00                                          $Revision::     $ */
/* ---------------------------------------------------------------------- */

#include <windows.h>
#include <stdio.h>
#include <conio.h>
#include <string.h>

/* --- Static variables ------------------------------------------------- */

static volatile BOOL    quit = FALSE;
static char             sbuff[128];
static HANDLE           drv_h;
static HANDLE           in_threadh;
static HANDLE           out_threadh;
static OVERLAPPED       ov_r, ov_w;

/* --- Macro definitions ------------------------------------------------ */

#define OPEN_DEVICE(name)                                                  \
   CreateFile(                                                             \
      (name),                       /* LPCTSTR - pointer to name of the file */ \
      GENERIC_READ | GENERIC_WRITE,/* DWORD   - access (read-write) mode    */ \
      0,                            /* DWORD   - share mode                  */ \
      NULL,                         /* LPSECURITY_ATTRIBUTES                 */ \
                                    /*         - pointer to security attribs */ \
      OPEN_EXISTING,                /* DWORD   - how to create               */ \
      0  |                                                                 \
      FILE_FLAG_OVERLAPPED,         /* DWORD   - file attributes             */ \
      NULL                          /* HANDLE  - template handle             */ \
      )

/* --- Internal Function Prototypes ------------------------------------- */

void  input_thread     (void *p);
void  output_thread    (void *p);

/* --- Main ------------------------------------------------------------- */

int main ( int argc, char *argv[])
   {
   HANDLE   ret_h;
   DWORD    ws;
   DWORD    retdw;
   int      stage=1;
```

```
   /** clear overlapped structure */
   memset ( &ov_r, 0, sizeof (ov_r) );
   memset ( &ov_w, 0, sizeof (ov_w) );

   if (argc < 2)
      drv_h = OPEN_DEVICE ("COM5");   /* Must Specify proper COM# as default */
   else
      drv_h = OPEN_DEVICE (argv[1]);
   if (drv_h == INVALID_HANDLE_VALUE)
      {
      ws = GetLastError();
      printf("Can NOT open device : %s. Error : 0x%lx", "", ws);
      return ( stage);
      }

   { DCB dcb;
   GetCommState(drv_h, &dcb);
   dcb.BaudRate = CBR_9600;
   dcb.Parity   = NOPARITY;
   dcb.ByteSize = 8;
   dcb.StopBits = ONESTOPBIT;
   dcb.fParity  = 0;
   dcb.fBinary  = 1;
   dcb.fOutxCtsFlow = 0;
   dcb.fOutxDsrFlow = 0;
   dcb.fDtrControl = DTR_CONTROL_ENABLE;
   SetCommState(drv_h, &dcb);
   }

   #define STAGE(idx, op, msg)                                  \
                                ret_h = op;                     \
                                if (ret_h==NULL)                \
                                   {                            \
                                   printf("%s\n", (msg));       \
                                   break;                       \
                                   }                            \
                                stage = idx;

   do {
      STAGE ( 6, CreateEvent (NULL, TRUE, FALSE, NULL),
                  "Can't Create Overlapped Event(read)" );
      ov_r.hEvent = ret_h;

      STAGE ( 7, CreateEvent (NULL, TRUE, FALSE, NULL),
                  "Can't Create Overlapped Event(write)" );
      ov_w.hEvent = ret_h;

      STAGE ( 8,
               CreateThread(
                  NULL,       // address of thread security attributes
                  0L,         // initial thread stack size, in bytes
                  (LPTHREAD_START_ROUTINE)output_thread,// adr of thread function
                  NULL,       // argument for new thread
                  0L,         // creation flags 0-run immediately
                  &retdw      // address of returned thread identifier
                  ),
               "Can't Create output thread" );
      out_threadh = ret_h;
STAGE ( 9,
               CreateThread(
```

```
                NULL,       // address of thread security attributes
                0L,         // initial thread stack size, in bytes
                (LPTHREAD_START_ROUTINE)input_thread,// addr of thread function
                NULL,       // argument for new thread
                0L,         // creation flags 0-run immediately
                &retdw      // address of returned thread identifier
                ),
             "Can't Create input thread" );
      in_threadh = ret_h;
      Sleep(100);
      printf("\nTest Console started. (press <^Z> to terminate).\n");
      } while (0);

   switch ( stage)
      {
      case 9:
         WaitForSingleObject (in_threadh, INFINITE); printf ("\n");
      case 8:
         quit = TRUE;
         ws = WaitForSingleObject ( out_threadh, 300);
         if (ws != WAIT_OBJECT_0)
         {
            DWORD ret_len;
         }
         SetEvent (ov_r.hEvent); //@@out_ev);
         ws = WaitForSingleObject ( out_threadh, INFINITE);
         CloseHandle ( out_threadh );
         CloseHandle ( in_threadh );
      case 7: CloseHandle ( ov_w.hEvent );
      case 6: CloseHandle ( ov_r.hEvent );
      case 1: CloseHandle ( drv_h );
      }

   return (0);
   }

/* --- Helpers ------------------------------------------------------------- */

#define SINGLE_CHARS

void  input_thread     (void *p)
   {
   int   ch;
   DWORD ws;
   char  str[100];

   ch = 0;
   while(!quit)
      {
      #ifdef SINGLE_CHARS
      ch = getch();
      printf("%c", ch);
      if (ch == 13)
       printf("\n");
      if ( ch == 0 )
             {
                   if (kbhit())
                   {
                         ch = 0x100 + getch();
                   }
```

```
            }
      #else
      gets(str);
      strcat(str, "\n");
      ch = str[0];
      #endif
      switch (ch)
         {
         case 0x1a:        // <Ctrl-Z> - emergency exit
            printf("\n---Exit---\n");//@@
            quit = TRUE;
            break;
         default:
            if (ch < 0x100)
               {
               BOOL           rs;
               DWORD          ret_len;
               #ifdef SINGLE_CHARS
               rs = WriteFile(drv_h, &ch, 1, &ret_len, &ov_w);
               #else
               rs = WriteFile(drv_h, str, strlen(str), &ret_len, &ov_w);
               #endif
               if (!rs)
                  {
                  ws = GetLastError ();
                  if ( ws != ERROR_IO_PENDING)
                     printf("DeviceIOControl (Write) Error : %i (0x%x)\n", ws, ws );
                  }
               rs = GetOverlappedResult (
                     drv_h,            // handle
                     &ov_w,            // address of overlapped structure
                     &ret_len,         // address of actual bytes count
                     TRUE              // wait flag
                     );
               if (!rs)
                  {
                  ws = GetLastError ();
                  printf("Write Error : %i (0x%x)\n", ws, ws );
                  }
               }
            else
               {
               }
            break;
         } /* switch (ch) */

      // give output thread chance to catch 'quit' character from driver
      // @@ there should be a better way to do this
      if (ch == 0x1b)
                  Sleep(200);
      }
   }

#define BUFSZ  128

void  output_thread    (void *vp)


{
   BOOL           rs;
```

```
   DWORD          read_len=0;
   char           wbuff[1];
   char*          p;

   while (!quit)
      {
      rs = ReadFile(drv_h, wbuff, sizeof(wbuff), &read_len, &ov_r);
      if ( !rs)
         {
         rs = GetLastError ();
         if ( rs != ERROR_IO_PENDING)
            {
            printf("DeviceIOControl (Read) Error : %i (0x%x)\n", rs, rs );
            break;
            }
         }
      rs = WaitForSingleObject ( ov_r.hEvent, INFINITE);
      rs = GetOverlappedResult (
            drv_h,         // handle of file, pipe, or communications device
            &ov_r,         // address of overlapped structure
            &read_len,     // address of actual bytes count
            FALSE          // wait flag
            );
      if (quit)
                  break;
      if ( rs )
         {
         p = wbuff;
         while (read_len >0)
            {
            if (*p == 0x1a)
                        {
                              quit = TRUE;
                              printf("\n\nExiting Test...");
                              break;
                        }
            putch (*p);
            ++p;
            --read_len;
            }
         }
      }
   };
// end of file
```

POWER BUILDER EXAMPLE

The following example illustrates how to set up PowerBuilder (from Sybase) to read magnetic data from the IntelliPIN device. Since PowerBuilder does not interface to a serial port very easily, a third-party OCX is required. The first part of this application note shows how to load an ActiveX component. The main program script shows how to interface with the OCX, the MTD Windows Driver, and the MagTek device (in this case the IntelliPIN).

The following communication ActiveX components are available for use with PowerBuilder:

Product	Company	Web Site	Phone
IO ActiveX Control	Software Island	members.aol.com/easyio	N/A
Comm Library	EllTech	elltech.com	800-227-8047
COMM-DRV/LIB	WCSC	www.wcscnet.com	800-966-4832

In our example, we have chosen "IO ActiveX Control" from Software Island. Here is a method that can be used to install this component:

1. In a PowerBuilder application, open a new window.
2. From the "Controls" dropdown menu, select "OLE".
3. From the "Create New" tab, select the intended OCX, for example, "IO Control". (It is assumed that the OCX has already been registered by installing it according to the manufacturer's directions.) Then click "OK".
4. Left click anywhere on the open window and drop the component onto it.
5. Right click on the newly installed component and select "Properties". Enter "mtd" into the "Name" text field. Enter "MTD OCX" into the "Display Name" and "Tag" text fields. Click "OK".
6. Right click anywhere on the window outside the new component then select "Properties". Enter "ole_io" into the "Title:" text field. Deselect the "Visible" check-off box so the window will not be shown then click "OK".
7. Right click anywhere on the window outside the new component then select the "Script" option. Insert the following script into the "ole_io" window.

```
////////////////////////////////////////////////////////
//    Window Script to load OCX for Mag-Tek Driver.     //
//    This is the script for the invisible window that  //
//    contains the OLE object.                          //
////////////////////////////////////////////////////////
integer result

result = mtd.object.Open("COM5:", "")
// COM5 is the virtual port name which was automatically
// assigned to IntelliPIN RS-232 Driver upon installation.
// It may be different for your installation.

if result < 1 then
   MessageBox("Open Read",result)
   return 0
END if
```

8. Close the PowerScript Painter window and answer “Yes” to “Save changes…”.
9. Close the Window Painter window and answer “Yes” to “Save changes…”. At the “Save Window” dialog box, enter “ole_io” then click “OK”.
10. Open the PowerScript window for the main application and integrate the following commands into the application. (This demo application prompts the user to read a card. The program will continue to loop until the “Cancel” button is pushed.)

```
////////////////////////////////////////////////////////////////
//    Application to demonstrate use of OLE ActiveX Component  //
//    to interface to Mag-Tek Windows Drivers (MTD).           //
////////////////////////////////////////////////////////////////
string response
integer result

// Open ActiveX frame window to load the ole_io control.
// This may take a few seconds while the port is opened.
Open (ole_io)

// Include any commands required for your application.

// Specify the number of seconds to wait for card to be read
ole_io.mtd.object.SetTimeOut(120)
// Define the message to be shown on the IntelliPIN to read a card.
// The end of line (~n) must be inserted for driver commands.
ole_io.mtd.object.WriteString("/set msg1 Read a Card~n")

NextCard:
// Request the card to be read.
ole_io.mtd.object.WriteString("/read card~n")
// Wait for the card to be swiped.
response = ole_io.mtd.object.ReadString(250)
// See if the card was read.
if response <> "" then
    // Remove "PIN Pad is processing" Display from IntelliPIN
       ole_io.mtd.object.WriteString("/display Thank You~n")
    // Show the card data in a Message window.
          result = MessageBox("Read Card?",response,Exclamation!,OKCancel!)
       else
    // It was a timeout from the OCX.
    // Must cancel the active command if the read was not performed.
       result = ole_io.mtd.object.WriteString("/cancel read~n")
    //ignore the response to cancel
       response = ole_io.mtd.object.ReadString(50)
end if
if result = 1 then
    goto NextCard
end if
```

APPENDIX A. INSTALLATION AND SETUP

The distribution disks contain the MTD Driver files for many of the MagTek products. In addition to the drivers, there are number of files that are required to support the installation and operation of these drivers. The disk contents are listed in the tables below.

Some of the Drivers support multiple configurations of the associated product. For example, the IntelliPIN Driver (IPIN.VXD) provides an interface vehicle for three different interface configurations. When a Driver is installed, be sure to select the proper interface type for your installation.

After installing a driver, you will be given the option of adjusting the **Port Name** (virtual port) and the **Connect to** (physical port) values. The **Port Name** is the COMxx port by which the device will be addressed. The **Connect to** is the port that the device is physically attached to on the PC.

INSTALLING DEVICE DRIVERS (W95/98/ME)

File or Directory Name	Device Friendly Name	DESCRIPTION
OEMSETUP.INF		Installation descriptor file
README.TXT		Describes the disk file contents and provides installation procedures
\W95_DRV	Directory	The following Windows 95/98/ME device drivers are located in this directory:
DMAPLD.VXD		DriverMagic Advanced Part Library
DMVXD.VXD		DriverMagic engine
DMVXDD.VXD		DriverMagic Windows 9x/ME Driver Part Kit
GENERIC.VXD		Generic Driver that allows communication with any device using any command format.
IPIN.VXD	IntelliPIN RS-232 IntelliPIN Wedge IntelliPIN MICR Aux	IntelliPIN Driver (RS-232, keyboard, and MICR+ aux port interfaces)
MAGCDFLT.DLL		Resource DLL for the default locale
MAGCDFLT.HLP		Default Help File
MAG-TEKCL.DLL		Class Installer
MAG-TEKCL.VXD		Class driver for Windows 9x/ME
MAGWEDGE.VXD	Mag-Wedge	Mag-Wedge Driver (keyboard interface)
MICRPLUS.VXD	MICR+	MICR Plus Driver (RS-232 interface)
MINIMICR.VXD	Mini MICR RS-232 Mini MICR Wedge	Mini MICR Driver (RS-232 and keyboard interfaces)
MINIWEDG.VXD	MiniWedge	Mini-Wedge Driver (keyboard interface)
MT85.VXD	MT-85	MT-85 Driver (RS-232 interface)
MT95.VXD	MT-95	MT-95 Driver (RS-232 interface)
MTPPINSR.VXD	Port-powered insert reader	Port Powered Insert Driver (RS-232 interface)
MTPPSWIP.VXD	Port-powered swipe reader	Port Powered Swipe Driver (RS-232 interface)

If problems occur when adding or updating drivers, it may be necessary to remove previous versions. *See the "**Removing the Driver (W95/98/ME)**" section later in this document for instructions on how to uninstall the driver.*

General Notes:

1. The computer and device should be powered off when connecting any devices.
2. Although you do not have to have the device connected to install the driver, it is highly recommended. This allows the device and driver to be tested when the driver is installed.
3. Note which hardware port each device is using on the computer as this information will be used later in the driver installation process.
4. Because of a bug in Win95 installer, if an installation image is put in a location with a long filename, the installer does not find it and says, "the specified location does not contain information about your hardware". To avoid that problem, put the installation image in the directory that does not contain long filenames in its full path.

Adding the First Device Driver (W95/98/ME)

1. Start Windows.
2. Open the Control Panel and double click on the "Add New Hardware" icon.
3. Click the **Next** button to advance to the first input screen. (In Windows 98/ME, you will have to click **Next** one more time.)
4. Select the **No** radio button when asked if you want Windows to search for your new hardware and click the **Next** button.
5. Select **Other devices** from the list then click the **Next** button. (If MagTek is included in the "hardware types" list, go to **Adding Another Device Driver (W95/98/ME)**
6. Click on the **Have Disk** button.
7. Insert the driver program disk into the CD drive and enter **d:** into the dialog or use **Browse** to point to where the installation file (oemsetup.inf) is located; click the **OK** button on the dialog box.
8. Select the device to be installed from the list of models and click the **Next** button.
9. Click the **Finish** button. The computer will take a moment to install the driver. Please be patient. (**Do NOT click the "Finish" button again!**)
10. If the computer displays a message stating "System Settings Change" and requests that you restart the computer, please do so.
11. Continue with **Completing the Installation (W95/98/ME)** below.

Adding Another Device Driver (W95/98/ME)

If at least one driver is already installed, follow these steps:

1. Start Windows.
2. Open the Control Panel and double click on the "Add New Hardware" icon.
3. Click the **Next** button to advance to the first input screen. . (In Windows 98/ME, you will have to click **Next** one more time.)
4. Select the **No** radio button when asked if you want Windows to search for your new hardware and click the **Next** button.

5. Select **MagTek** from the list of Hardware Types, then click the **Next** button.
6. Select the device to be installed from the displayed list box and click the **Next** button.
7. Click the **Finish** button. The computer will take a moment to install the driver. Please be patient. If the installation file cannot be found, click **Browse** and point to where the installation file (oemsetup.inf) is located; click the **OK** button on the dialog box.
8. If the computer displays a message stating "System Settings Change" and requests that you restart the computer, please do so.
9. Continue with **Completing the Installation (W95/98/ME)** below.

Updating an Installed Device Driver (W95/98/ME)

When a newer version of a driver is available, use these steps to update it:

1. Start Windows.
2. Right-click on **My Computer** on the desktop or open the Control Panel and double click on the **System** icon then select **Properties**.
3. Click on the Device Manager tab.
4. Click on the plus sign in front of the MagTek list item to expand it.
5 Double click on the driver to be updated or click once on the driver then click the **Properties** button.
6. Click on the Driver tab.
7. Click on the Update Driver button.
8. Select the **No** radio button when asked if you want Windows to search for your new hardware and click the **Next** button.
9. Select **MagTek** from the list, if shown, then click the **Next** button.
10. Select the device to be installed from the displayed list box and click the **Next** button.
11. Click the **Finish** button. The computer will take a moment to install the driver. Please be patient.
12. If the computer displays a message stating "System Settings Change" and requests that you restart the computer, please do so.
13. Continue with **Completing the Installation (W95/98/ME)** below.

Completing the Installation (W95/98/ME)

The Windows Add New Hardware Wizard will install the selected driver. If this is the first time a MagTek device driver is installed, it will also add the **MagTek** device class in the Device Manager.

When the installation has been completed, the device configuration property sheet will be displayed (the window title is **Installed Device**). Perform the following steps to configure the device:

1. Either accept the default selection for the virtual **Port Name** or select the desired port (COM5-COM15) to be associated with the device from the **Port Name** combo box and modify the device's friendly name if the default is not acceptable.
2. Select the port to which the device is connected (see *General Notes* in **Installing Device Drivers (W95/98/ME)** above) from the **Connect to** combo box.
3. Click on the **Test** button to validate the port settings and verify the device's presence. (The **Test** function only works on virtual COM ports 5 through 10. The test will fail on COM11-15 but the device will still be accessible from any application.)
4. Click **OK** to save the settings.
5. If the computer displays a message stating 'The specified "Connect To" port is used by another device…' make sure the "Connect To" port settings is correct. If it is (multiple devices can share a single port, but only one at a time can be selected), click on the **OK** button to finish the installation. Otherwise, click on the **Cancel** to change the port.

Modifying A Device Driver's Settings (W95/98/ME)

To modify the device driver's settings, perform the following steps:

1. Right-click on **My Computer** on the desktop or open the Control Panel and double click on the **System** icon then select **Properties**. See Figure A-1.
2. Select the **Device Manager** tab.
3. Expand the **MagTek** class by clicking the plus sign. Find the required device under the **MagTek** class then click on **Properties**.
4. Select the **Settings** tab to view the driver configuration. **Port Name** indicates the virtual port number and **Connect to** indicates the physical port.
5. Click the **Advanced** button (see Figure A-2) to view the communication settings. Some devices (e.g., MICR+) support automatic settings, which allow the driver to determine the present setup of the device. If required, click the **Specify settings manually** radio button and modify the communication setup. Click **OK** when done.

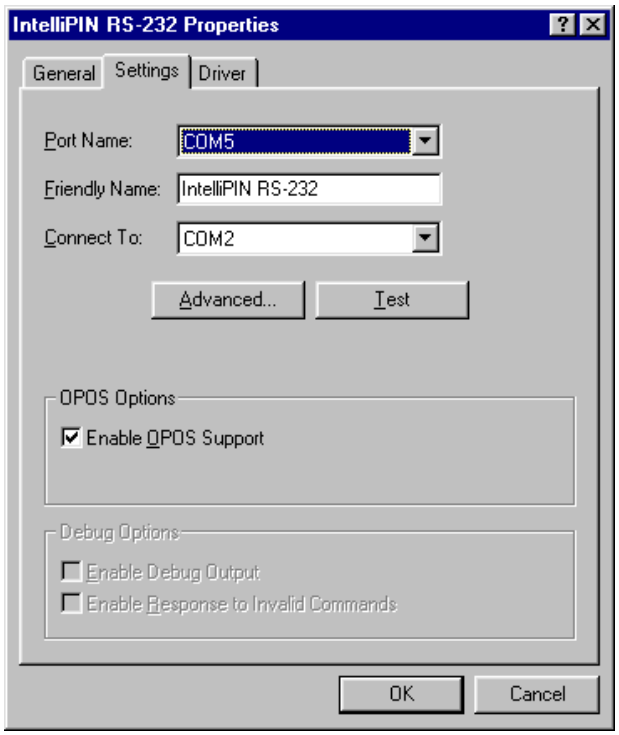


Figure A-1. Properties Settings, Windows 95/98/ME

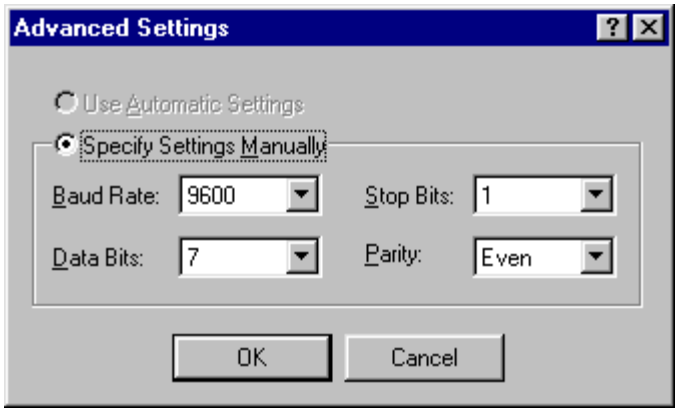


Figure A-2. Advanced Settings, Windows 95/98/ME

Removing the Drivers (W95/98/ME)

Caution

The following assumes familiarity with the Registry Editor. Improper use of the Registry Editor can cause Windows to cease to function. Please follow the instructions carefully.

Complete removal of the drivers requires two steps: (1) remove the drivers from the system using the Device Manager and (2) remove the driver files manually after all devices have been removed by the Device Manager.

To remove the drivers, follow these steps:

1. Stop any applications that are using the drivers. This will insure that all of the ports that are going to be removed are closed.
2. Right-click on **My Computer** on the desktop or open the Control Panel and double click on the **System** icon then select **Properties**.
3. Select the **Device Manager** tab and click on the plus sign at MagTek.
4. Select the device under the **MagTek** group and click on **Remove**. Then click **OK**. After all device drivers have been removed in this manner, go to step 5.
5. Using Explorer or some other file manager, remove the following driver VXDs from **C:\Windows\System**:

   GENERIC.VXD
   IPIN.VXD
   MAGWEDGE.VXD
   MICRPLUS.VXD
   MINIMICR.VXD
   MINIWEDG.VXD
   MT85.VXD
   MT95.VXD
   MTPPINSR.VXD
   MTPPSWIP.VXD

   The driver files may be removed only if no drivers are currently installed that require them. In particular, the class driver (MAG-TEKCL.VXD) must remain if any device type is still installed. The driver files may be removed when all devices of that particular type have been removed.
6. Remove the support files from **C:\Windows\System**. The support files are:

   DMAPLD.VXD
   DMVXD.VXD
   DMVXDD.VXD
   MAG-TEKCL.DLL
   MAG-TEKCL.VXD
   MAGCDFLT.HLP

MAGCDFLT.DLL
MAGCxxx.HLP (locale specific)
MAGCxxx.DLL (locale specific)

7. Find and remove the copy of the **Mag-TekOEMSETUP.INF** file made by Windows. In release 1 of Windows 95, it is located in **C:\Windows\inf**. With the OSR2 release of Windows 95 (Win95B) and Windows 98/ME, the files will be located in **C:\Windows\inf\other**.
8. Run the Registry Editor by clicking on **Start** button then select **Run**. Type **REGEDIT** into the text box and press the Enter key.
9. Delete the following sub-trees from the registry:
   HKEY_LOCAL_MACHINE\Software\Mag-Tek\ClassMap and
   HKEY_LOCAL_MACHINE\System\CurrentControlSet\Services\Class\Mag-Tek.
10. When in Windows 95, remove the following values from the registry:

HKEY_LOCAL_MACHINE\System\CurrentControlSet\Control\InstalledFiles

DMAPLD.VXD
DMVXD.VXD
DMVXDD.VXD
IPIN.VXD
MAGCDFLT.DLS
MAGCDFLT.HLP
MAG-TEKCL.DLS
MAG-TEKCL.VXD
MAGWEDGE.VXD
MICRPLUS.VXD
MINIMICR.VXD
MINIWEDG.VXD
MT85.VXD
MT95.VXD
MTPPINSR.VXD
MTPPSWIP.VXD

11. When in Windows 98/ME, remove the following values from the registry:

HKEY_LOCAL_MACHINE\Software\Microsoft\Windows\CurrentVersion\Setup\SetupX\Inf\OEMName

%windir%\inf\other\MAGTE~1.INF
%windir%\inf\other\MAG-TEKOEMSETUP.INF

HKEY_LOCAL_MACHINE\System\CurrentControlSet\Control\Installed Files\Rename

MAGCDFLT.DLS
MAG-TEKCL.DLS

HKEY_LOCAL_MACHINE\System\CurrentControlSet\Control\SessionManager\Known16DLLs

MAG-TEKCL.DLL

12. Close the Registry Editor by selecting **File / Exit**.

INSTALLING DEVICE DRIVERS (WNT)

File or Directory Name	Friendly Name	DESCRIPTION
OEMSETUP.INF		Installation descriptor file
README.TXT		Describes the disk file contents and provides installation procedures
\WNT_DRV	Directory	The following Windows NT device drivers are located in this directory:
DMAPLD.DLL		DriverMagic Advanced Part Library
DMNTK.DLL		DriverMagic engine
DMNTKD.DLL		DriverMagic Windows 9x/ME Driver Part Kit
GENERIC.SYS		Generic Driver that allows communication with any device using any command format.
IPIN.SYS	IntelliPIN RS-232 IntelliPIN Wedge IntelliPIN MICR Aux	IntelliPIN Driver (RS-232, keyboard, and MICR+ aux port interfaces)
MAG-TEKCL.SYS		Class driver for Windows 95
MAGWEDGE.SYS	Mag-Wedge	Mag-Wedge Driver (keyboard interface)
MICRPLUS.SYS	MICR+	MICR Plus Driver (RS-232 interface)
MINIMICR.SYS	Mini MICR RS-232 Mini MICR Wedge	Mini MICR Driver (RS-232 and keyboard interfaces)
MINIWEDG.SYS	MiniWedge	Mini-Wedge Driver (keyboard interface)
MT85.SYS	MT-85	MT-85 Driver (RS-232 interface)
MT95.SYS	MT-95	MT-95 Driver (RS-232 interface)
MTCFG.EXE		Command-line configuration utility
MTD_KBH.SYS		Keyboard hook Driver
MTPPINSR.SYS	Port-powered insert reader	Port Powered Insert Driver (RS-232 interface)
MTPPSWIP.SYS	Port-powered swipe reader	Port Powered Swipe Driver (RS-232 interface)

In Windows NT, only users with Administrator privileges may install system components. Log on as Administrator (or as a user with full administrative privileges) before attempting to install the MTD driver.

It is important to uninstall the previous version of MTD and re-boot the system before installing this version of the driver. The installation script provided cannot upgrade MTD from versions prior to version 1.09. The old driver can be uninstalled by using the Windows NT Installation

Wizard. Open the Wizard by double clicking on the Add/Remove Programs icon in the Control Panel. On the Install/Uninstall tab. Find and select the entry that reads

MTD preliminary release (uninstall)

or

Mag-Tek Device Drivers (MTD) - uninstall,

then click on the Add/Remove button. Re-boot the system after uninstalling the old version.

Installing the Driver Binaries (WNT)

To install the driver binaries (*.SYS), follow these steps:

1. Insert the installation media and open the drive using Windows Explorer.
2. Select the OEMSETUP.INF file and run the "Install" command from Explorer's "File" menu.
3. Windows NT will not display any messages if the installation completes successfully. If there are any problems, an error message will be shown. If a file cannot be located, use the browse button to find it.
4. Restart the system to load the MTD drivers.

Note

If, during the installation, a strange behavior is observed - failure, or some other unexpected error–a system reboot is necessary before continuing or repeating the failed operation.

Uninstalling the Drivers (WNT)

Close any application that may have the MTD driver open before attempting to uninstall it. Failure to do this will cause the uninstallation to fail–after that the system must be re-booted before a subsequent attempt to uninstall the driver could be performed.

The driver can be uninstalled by using the Windows NT Installation Wizard. Open the Wizard by double-clicking on the Add/Remove Programs icon in the Control Panel. On the Install/Uninstall tab, find and select the entry that reads

Mag-Tek Device Drivers (MTD) - uninstall

then click on the Add/Remove button. Administrative privilege is required to perform this operation. The uninstallation removes all MTD files and adjust the registry as required. The system must be re-booted to remove the keyboard hook driver from memory. Reinstallation will fail if the system is not re-booted after uninstalling the driver.

INSTALLING DEVICE DRIVERS (W2000/XP)

File or Directory Name	Friendly Name	DESCRIPTION
MTD_KBH.INF		Keyboard hook installation descriptor file
MTD_KBH.SYS		Keyboard hook Driver
OEMSETUP.INF		Installation descriptor file
README.TXT		Describes the disk file contents and provides installation procedures
\I386	Directory	Microsoft keyboard drivers
\W2K_DRV	Directory	The following Windows 2000/XP device drivers are located in this directory:
DMNTK.DLL		DriverMagic engine
DMNTKD.DLL		DriverMagic Windows Driver Part Kit
GENERIC.SYS		Generic Driver that allows communication with any device using any command format.
IPIN.SYS	IntelliPIN RS-232 IntelliPIN Wedge IntelliPIN MICR Aux	IntelliPIN Driver (RS-232, keyboard, and MICR+ aux port interfaces)
MAGTEKCL.SYS		Class driver for Windows 95
MAGWEDGE.SYS	Mag-Wedge	Mag-Wedge Driver (keyboard interface)
MICRPLUS.SYS	MICR+	MICR Plus Driver (RS-232 interface)
MINIMICR.SYS	Mini MICR RS-232 Mini MICR Wedge	Mini MICR Driver (RS-232 and keyboard interfaces)
MINIWEDG.SYS	MiniWedge	Mini-Wedge Driver (keyboard interface)
MT85.SYS	MT-85	MT-85 Driver (RS-232 interface)
MT95.SYS	MT-95	MT-95 Driver (RS-232 interface)
MTCFG.EXE		Command-line configuration utility
MTPPINSR.SYS	Port-powered insert reader	Port Powered Insert Driver (RS-232 interface)
MTPPSWIP.SYS	Port-powered swipe reader	Port Powered Swipe Driver (RS-232 interface)

In Windows 2000/XP, only users with Administrator privileges may install system components. Log on as Administrator (or as a user with full administrative privileges) before attempting to install the MTD driver.

It is important to uninstall the previous version of MTD and re-boot the system before installing this version of the driver. The installation script provided cannot upgrade MTD from versions prior to version 1.10. The old driver can be uninstalled by using the Windows 2000/XP Installation Wizard. Open the Wizard by double clicking on the Add/Remove Programs icon in the Control Panel. On the Install/Uninstall tab. Find and select the entry that reads

MTD preliminary release (uninstall)

or

Mag-Tek Device Drivers (MTD) - uninstall,

then click on the Add/Remove button. Re-boot the system after uninstalling the old version.

Installing the Driver Binaries (W2000/XP)

To install the driver binaries, follow these steps:

1. Insert the installation media and open the drive using Windows Explorer.
2. Select the OEMSETUP.INF file and run the “Install” command from Explorer’s “File” menu.
3. Windows 2000/XP will not display any messages if the installation completes successfully.
4. Open the Control Panel and double click on the “System” icon.
5. Click on the "Hardware" tab.
6. Click on the "Device Manager" button.
7. Click on the ‘+’ to expand the "Keyboards" entry in the Device Manager list.
8. Right click on the "PC/AT Enhanced PS/2 Keyboard (101/102-Key)" entry.
9. Click the "**Properties"** item in the dialog box.
10. Click on the "Driver" tab.
11. Click the "Update Driver" button.
12. Click the “**Next**” button to advance to the first input screen.
13. Select the "Display a list of the known drivers..." radio button.
14. Click the “**Next**” button to advance to the next input screen.
15. Click on the "Have Disk" button.
16. Enter installation drive and directory in the "Copy Manufacturer's file from:" text box.
17. Click the “**Next**” button to advance to the next input screen.
18. Click on "**Yes**" to the "Update Driver Warning".

19. Click the "**Next**" button to advance to the next input screen.
20. Click on "**Yes**" to the "Digital Signature Not Found".
21. Click "Finish".
22. Click "Close" on the "System" dialog.
23. Answer "**Yes**" to the "Restart System" prompt.

Note

If, during the installation, a strange behavior is observed - failure, or some other unexpected error–a system reboot is necessary before continuing or repeating the failed operation.

Uninstalling the Drivers (W2000/XP)

Close any application that may have the MTD driver open before attempting to uninstall it. Failure to do this will cause the uninstallation to fail–after that the system must be re-booted before a subsequent attempt to uninstall the driver could be performed.

The driver can be uninstalled by using the Windows Installation Wizard. Open the Wizard by double-clicking on the Add/Remove Programs icon in the Control Panel. On the Install/Uninstall tab, find and select the entry that reads

Mag-Tek Device Drivers (MTD) - uninstall

then click on the Add/Remove button. Administrative privilege is required to perform this operation. The uninstallation removes all MTD files and adjusts the registry as required.

Uninstalling the Keyboard Hook Driver (W2000/XP)

For Windows 2000/XP, the keyboard hook driver must be uninstalled after the driver binaries (cf. above) are uninstalled and before rebooting.

1. Open the Control Panel and double click on the "System" icon.
2. Click on the "Hardware" tab.
3. Click on the "Device Manager" button.
4. Click on the '+' to expand the "Keyboards" entry in the Device Manager list.
5. Right click on the "PC/AT Enhanced PS/2 Keyboard (101/102-Key)" entry.
6. Click the "**Properties"** item in the dialog box.
7. Click on the "Driver" tab.
8. Click the "Update Driver" button.
9. Click the "**Next**" button to advance to the first input screen.
10. Select the "Search for a suitable driver… " radio button.
11. Click the "**Next**" button to advance to the next input screen.

12. Uncheck all “Optional search locations” check boxes.
13. Click the “**Next**” button to advance to the next input screen.
14. Click the “**Next**” button to advance to the next input screen.
15. Answer "**Yes**" to the "Confirm Driver Install". (Note: This uninstallation procedure may hang at step 15. This is a non-disruptive hang-up. User should wait 10 seconds and do a hard re-boot. Windows 2000/XP should recover without a system check or scan disk.)
16. Click "Finish".
17. Click "Close" on the "System" dialog.
18. Answer "**Yes**" to the "Restart System" prompt.

WINDOWS NT/2000/XP CONFIGURATION UTILITY

To add or set up MagTek devices, use the MTCFG.EXE utility. It is installed by the installation procedure described above. The examples below show a typical setup of a keyboard device and a serial device. For a more detailed description of the MTCFG.EXE utility, see "Using the MTCFG Utility (WNT/2000/XP)" below. As with the driver installation, this phase requires the current user to have Administrator privileges.

A device does not have to be physically connected at the time when it is set up. The driver will only access the device when it is opened.

The installation procedure consists of two phases: (a) installing the driver binaries and (b) configuring MagTek devices.

No re-boot is necessary after adding a device with MTCFG.

Adding a Keyboard Device (WNT/2000/XP)

The MiniWedge is used in the following example:

1. Select an unused COM port number for the device. Choose any number between 5 and 255 that is not used by other devices (if in doubt, check the Ports Control Panel–it displays all COM*x* names currently used).
2. Enter the following command at the DOS command prompt (the example assumes COM5 was selected):
   ```
   mtcfg COM5 ”MiniWedge” ”FriendlyName=MiniWedge”
   ```
3. The third argument (“FriendlyName=...”) is optional and may be omitted if no friendly name is needed. Quotes are required around arguments if they include spaces.
4. If the device is added successfully, MTCFG will display the following prompt:
   ```
   Re-starting MTD driver - close all applications using MTD.
            Press <Enter> to restart MTD
   ```
5. Press the Enter key to complete the MTD configuration–if the operation was successful, the configuration utility displays:

```
MTD was successfully re-started. The changes you made are now in effect.
```

Note

Only a single keyboard device can be installed at a time.

Adding a Serial Device (WNT/2000/XP)

Mini MICR is used in the following example:

1. Select an unused COM port number for the device. Choose any number between 5 and 255 that is not used by other devices (if in doubt, check the Ports Control panel–it displays all COMx names currently used). MTCFG will verify that the COM name selected is not used by another MagTek device, but it will not check against non-MagTek devices.
2. Select a serial port to which the device will be attached (any standard serial port may be used; do not use the name of an existing MagTek device here). MTCFG allows multiple MagTek devices to be configured as attached to the same port–in this case these devices cannot be opened simultaneously.
3. Enter the following command at the NT command prompt (the example assumes that the device is physically connected to COM2 and MTD device will appear as COM6):

   ```
   mtcfg COM6 "Mini MICR RS-232" UsePort=COM2 "FriendlyName=Mini MICR RS-232"
   ```

   The value specified for **UsePort** must be all uppercase (e.g., use **COM2**, not **com2** or **Com2**). The fourth argument ("FriendlyName=...") is optional and may be omitted.
4. If the device is added successfully, MTCFG will display the following prompt:

   ```
   Re-starting MTD driver - close all applications using MTD.
                  Press <Enter> to restart MTD
   ```

5. Press the Enter key to complete the MTD configuration–if the operation was successful, the configuration utility displays:

   ```
   MTD was successfully re-started. The changes you made are now in effect.
   ```

Adding an 'IntelliPIN MICR Aux' Device (WNT/2000/XP)

Before installing the IntelliPIN Aux device, make sure that the MICR+ device has been successfully installed and opened. If the device has not been successfully opened, the MICR+ device may not be properly configured to operate with a device attached to its auxiliary port thereby preventing the IntelliPIN Aux device from opening. Additionally, make sure that the communication settings for both devices are identical. When using both the MICR+ and the IntelliPIN drivers, the MICR+ driver must be opened before the IntelliPIN driver and closed after the IntelliPIN driver is closed.

The procedure for adding an "IntelliPIN MICR Aux" device is similar to the procedure for adding a serial device. The only difference is that the "UsePort" parameter must be "AUX port on " followed by the friendly name or port name of the device to which the device is attached. (The example assumes that the device is physically connected to auxiliary port of MICR+ device

and MTD device will appear as COM7. The MICR+ device appears to the system as COM12 and has a friendly name : MICR+.) Two examples are shown:

```
mtcfg COM7 "IntelliPIN MICR Aux" "UsePort=AUX port on MICR+"
                "FriendlyName=IntelliPIN AUX"
                              or
mtcfg COM7 "IntelliPIN MICR Aux" "UsePort=AUX port on COM12"
                "FriendlyName=IntelliPIN AUX"
```

Viewing the List of Configured Devices (WNT/2000/XP)

Execute MTCFG with no command line arguments. If the above examples have been used, the displayed result will be:

```
port Conn. to           model                   friendly name
COM5                    MiniWedge               MiniWedge
COM6 COM1               Mini MICR RS-232        Mini MICR RS-232
COM7 AUX port on MICR+  IntelliPIN MICR Aux     IntelliPIN AUX
```

Using the MTCFG Utility (WNT/2000/XP)

MTCFG.EXE is a command-line utility installed with the MTD drivers. It requires that the driver binaries be correctly installed, as described in the previous sections. MTCFG cannot be used to install the driver binaries; running it from the installation media before the driver has been installed will result in the following error message:

```
Mag-Tek driver is not correctly installed. Please install the driver
before using this program.
```

The same message will be displayed if the installation has been modified manually, e.g., the installation script has been renamed or removed or the driver's Registry data has been removed.

It is recommended that all applications that may have opened a MagTek device be terminated before using MTCFG to change a device's configuration, or to add or remove a device.

Command syntax summary

Command Syntax	Meaning
`mtcfg`	list installed MagTek device drivers
`mtcfg -?`	display a help page
`mtcfg -help`	display a help page
`mtcfg -models`	list available MagTek device models
`mtcfg` *port-name*	list settings for a given device
`mtcfg` *port-name* `–all|more`	verbose list of settings
`mtcfg` *port-name model* [*settings*]	* add and configure a new device
`mtcfg` *port-name* `–delete`	* delete a device
`mtcfg` *port-name settings*	* change settings for a device

* these commands require Administrator privilege. MTCFG will display an error message if the current user is not an Administrator.

Displaying Configuration Information (WNT/2000/XP)

To display the list of configured MagTek devices, use the following syntax:

```
mtcfg
```

To display the settings for a single device, use:

```
mtcfg COMx
```

COM*x* is the name (virtual port) of the device, as set when the device was first configured. This name is shown in the leftmost column in the list of devices. This command displays only the common settings for the device–the ones that are most likely to require modification. To display all device settings, including all data parsing format strings, use the following syntax:

```
mtcfg COMx –all | more
```

The pipe symbol and "more" will present the information one screen at a time.

Adding New Devices (WNT/2000/XP)

To add a new device use the following command syntax:

```
mtcfg port-name model
```

or

```
mtcfg port-name model settings
```

port-name is the name (virtual port) chosen for the new device. It must not be used by another device in the system (MagTek or other). The port name in the form COM*xxx* (valid values are COM5 .. COM255). MTCFG will verify that the name is not used by other MagTek devices that were set up with this utility, but it will not check whether the name is used by any other device in the system.

model is the full name of the device model to be added. The name should be enclosed in quotes if it contains spaces. Use "`mtcfg -models`" to see a list of models. The model names used by MTCFG are the ones specified in the [Models] section of the MTD installation script (OEMSETUP.INF).

settings specifies one or more device settings in the form *name=value*. The syntax for these is identical to the syntax used when modifying the settings of an already installed device. See the next section for a list of common settings. Specifying any settings when adding a device is optional–they can always be specified later (see the next section), but it is recommended to include at least those settings that are required for the device to operate, e.g., `UsePort` for serial devices.

Configuration Examples for Windows NT/2000/XP

These examples are for illustration only. Most of the command line entries will have to be modified to accommodate the actual installation.

Device or driver	Command Line	Comment
Generic RS-232	`MTCFG COM5 "Generic Serial (RS-232)" FriendlyName=MT-80 UsePort=COM1 baud=4800 parity=0 datasize=7`	Be sure to specify the proper communication parameters for the selected device.
Generic KB	`MTCFG COM6 "Generic Wedge (Keyboard)" FriendlyName=MagReader`	"UsePort" is not required for keyboard devices.
IntelliPIN RS-232	`MTCFG COM7 "IntelliPIN MICR Aux" FriendlyName=IntelliPIN "UsePort=AUX port on MICR+"`	The MICR+ driver must be installed before this driver.
IntelliPIN RS-232	`MTCFG COM8 "IntelliPIN RS-232" FriendlyName=PINPad UsePort=COM2`	Communication parameters may be required.
IntelliPIN KB	`MTCFG COM9 "IntelliPIN Wedge" "FriendlyName=IntelliPIN KB"`	Quotes are used for Friendly Name to allow the space.
Mag-Wedge	`MTCFG COM10 "Mag-Wedge" "FriendlyName=Wedge Reader"`	
MICR+	`MTCFG COM11 "MICR+" FriendlyName=MICR+ UsePort=COM1`	Communication parameters may be required.
Mini MICR RS-232	`MTCFG COM12 "Mini MICR RS-232" FriendlyName=MICRS UsePort=COM1`	Communication parameters may be required.
Mini MICR KB	`MTCFG COM13 "Mini MICR Wedge" FriendlyName=MICRW`	
MiniWedge	`MTCFG COM14 "MiniWedge" FriendlyName=MSR`	
MT-85	`MTCFG COM15 "MT-85" "FriendlyName=MSR Encoder" UsePort=COM2`	Communication parameters may be required.
MT-95	`MTCFG COM16 "MT-95" FriendlyName=MT-95 UsePort=COM1 baud=9600 parity=-1 datasize=8`	Communication parameters may not be required.
Port Powered Insert Reader	`MTCFG COM17 "Port-powered insert reader" FriendlyName=PPInsert UsePort=COM1`	No communication parameters are required.
Port Powered Swipe Reader	`MTCFG COM18 "Port-powered swipe reader" FriendlyName=PPSwipe UsePort=COM2`	No communication parameters are required.

Modifying a Device Driver's Settings (WNT/2000/XP)

Use the following syntax to change settings of a device:

```
mtcfg <port-name> <setting1> [<setting2> [<setting3>...]]
```

each of the settings is specified as

name=value

if *value* contains spaces, the whole *name=value* string should be enclosed in quotes (not just the value), e.g., to specify the string "MT-85 on COM1" as the friendly name for COM5 with a baud rate of 9600 bps, use the following syntax:

```
mtcfg COM5 "FriendlyName=MT-85 on COM1" baud=9600
```

Following is a list of common settings that can be changed for a device. Most settings have a default value and may be missing when the list of settings is requested for a device (e.g., by typing "MTCFG COM5").

Name	Use
baud	(optional, used for serial devices only) device's baud rate, specified as an integer (e.g., 9600)
parity	(optional, used for serial devices only) an integer specifying the parity used by the device: use -1, 0, 1, 2, or 3 for None, Even, Odd, Space and Mark parity respectively.
datasize	(optional, used for serial devices only) specifies the device's serial word size in bits: 7 or 8.
stopbits	(optional, used for serial devices only) stop bits to use on transmission: 1 or 2.
UsePort	the serial port to which the device is connected. Must specify a valid standard serial port (or a port that is 100% compatible with a standard serial port).
FriendlyName	(optional) alternative name for the device. If specified, the device may be opened from user mode using this name (the prefix \\.\ must be added to the name. For example if FriendlyName=Port-powered swipe reader, this device can be opened as "\\.\Port-powered swipe reader")
EnableFDP	(optional) Enable Flexible Data Parsing. Set this to 1 to enable data parsing and to 0 to disable data parsing.
PortName	Specifies the port name under which the device is visible to user-mode applications. Modifying this setting also changes the name used to refer to this device when using the MTCFG utility, e.g., if `mtcfg COM5 PortName=COM8` is executed, the device that was COM5, must be referred to as COM8 in any subsequent invocations of MTCFG. This setting is treated specially by MTCFG–it will validate the port name and make sure that it is not used by other MagTek devices before making the change.

Device settings other than the ones listed above should not be modified without carefully reviewing the driver's engineering documentation for the specific device model.

Removing a Device (WNT/2000/XP)

To remove a MagTek device use the following command syntax:

```
mtcfg port-name -delete
```

The device is removed and all non-default settings specified for it are lost.

This operation does not remove any files from the system. To remove all devices and uninstall the MTD driver, follow the instructions in the next section.

MTD PROGRAMMING EXAMPLES

Example programs are included in the following directory:

File or Directory Name	DESCRIPTION
\EXAMPLES\CPP	Visual C++ example application (executable and source).
\EXAMPLES\DELPHI	MSCOMM and file I/O based Delphi sample applications (executables and sources)
\EXAMPLES\VB50	MSCOMM and file I/O based Visual Basic sample applications (executables and sources)
\EXAMPLES\PwrBldr	Power Builder example using the IntelliPIN

APPENDIX B. COMMAND LIST SUMMARY

This is a consolidated list of all available commands for the MagTek Windows Drivers.

Command	Description	Page
/cancel *cmd*	Cancel a command. *cmd* can be any of the transaction commands.	16
/display *[x]*	Display a message or two alternating messages on the LCD screen.	17
/echo *string*	Driver test command.	17
/event *n data*	Response to an unsolicited event notification.	18
/get *prop*	Get a property.	18
/load_key *n key*	Load a key into the device.	19
/rawrecv	Receive data from the device	20
/rawsend *x*	Send arbitrary data to the device.	21
/rawxact *x*	Execute a send/receive transaction with the device in raw mode.	21
/read *[[x] y]*	Read data from the device.	22
/reset	Clear any pending operations and reset the device to initial state.	26
/set *prop val*	Set a property.	26
/ver	Read driver version.	26
/write *data*	Encode magnetic stripe command.	27

APPENDIX C. STATUS CODES

The following table defines the status codes returned in command responses. Note that it is not meant as a complete list of status codes–new codes may be added as necessary.

Value	Mnemonic and Description
00	successful operation
05	port already open
1F	wrong device ID
22	value, buffer, whatever may overflow
30	value not valid in operation context
31	value not valid in module's context
32	value out of range
34	text or formatted data syntax error
35	name invalid in module's context
40	internal error. Unexpected result from a system API
41	driver internal error
45	operation rejected (inappropriate state)
47	operation failed or not successful
60	I/O error (peripheral error)
62	requested item not found
63	duplicated item is not allowed
74	access type not appropriate / not possible
79	wrong device ID
81	a time-out has expired
82	operation cancelled on caller's request
83	operation aborted (on system, user or module's request)
93	feature not implemented
94	partial implementation or feature not supported

APPENDIX D. DEVICE DRIVER SUMMARIES

This section contains summaries of Device Drivers for the for the following models:

- IntelliPIN and IntelliPIN PLUS
- MagWedge Reader
- MiniWedge Reader
- MICR+ Reader
- Mini-MICR Reader
- Port Powered RS-232 Swipe Reader
- Port Powered RS-232 Insertion Reader
- MT-85 Encoder
- MT-95 Encoder

The summary for each model contains a list of the commands properties supported.

INTELLIPIN PINPAD & MSR

File Name IPIN.VXD **Part Number** 30037395

Friendly Name(s) IntelliPIN RS-232, IntelliPIN Wedge & IntelliPIN MICR+ Aux

Remarks The Automatic Settings in the properties sheet are not supported; the communications must be specified manually. When using the IntelliPIN on the MICR+ Aux port, the MICR+ driver must be installed before the IntelliPIN driver; also the IntelliPIN driver must be closed before the MICR+ driver is closed.

Commands Supported					
`/cancel` *cmd*	✔	`/load_key` *n key*	✔	`/reset`	✔
`/display` *[x]*	✔	`/rawrecv`	✔	`/set` *prop val*	✔
`/echo` *string*	✔	`/rawsend` *x*	✔	`/ver`	✔
`/event` *n data*		`/rawxact` *x*	✔	`/write` *data*	
`/get` *prop*	✔	`/read` *[[x] y]*	✔		

Properties Supported								
Property	**Yes**	**Default**	**Property**	**Yes**	**Default**	**Property**	**Yes**	**Default**
`account_no`	✔		`chk_mod10`			`msg3`	✔	
`amount`	✔		`chk_number`			`msg4`	✔	
`applied_fmt`	✔		`chk_routing`			`offline_enc`		
`c_card_stat`			`chk_status`			`oper_tout`	✔	*
`c_cardwpin`	✔	1	`chk_transit`			`pin_blk_fmt`	✔	*
`c_check`		0	`cmd_pending`	✔		`pinfilldig`	✔	*
`c_events`			`dblpinentry`	✔	*	`port_name`	✔	
`c_keypress`	✔	1	`dev_status`	✔		`pwroffdelay`	✔	*
`c_keystring`	✔	1	`dev_version`	✔	*	`s_down_tout`	✔	*
`c_magnetic`	✔	1	`enable_cmc7`			`track1ss`	✔	
`c_mechanics`		0	`enc_key`	✔		`track2ss`	✔	
`c_pin`	✔	1	`enc_key_sn`	✔	*	`track3ss`	✔	
`c_smart`		0	`enc_mode`	✔	*	`trivpinchk`	✔	*
`c_tracks`	✔	111	`entry_echo`	✔		`trk_enable`	✔	*
`c_write`		0	`entry_len`	✔		`trk1data`	✔	
`c_wr_secure`			`entry_tout`	✔		`trk2data`	✔	
`capitalize`	✔	1	`events_on`			`trk3data`	✔	
`card_stat`			`invalcmdrsp`	✔		`visa_mac1`	✔	
`chk_account`			`key_parity`	✔	*	`visa_mac2`	✔	
`chk_amount`			`lasterr`	✔		`visa_mac3`	✔	
`chk_bankid`			`max_pin_len`	✔	*	`wr_coer`		
`chk_data`			`msg1`	✔		`wr_secure`		
`chk_format`			`msg2`	✔		`xact_type`	✔	d

* = Depends on setting in the device.

MAGWEDGE SWIPE READER

File Name MAGWEDGE.VXD **Part Number** 30037348

Friendly Name(s) MagWedge

Remarks The driver cannot determine which tracks are supported on the device, so the `c_tracks` and `trk_enable` properties will always indicate `111`.

Commands Supported					
`/cancel` *cmd*	✔	`/load_key` *n key*		`/reset`	✔
`/display` *[x]*		`/rawrecv`	✔	`/set` *prop val*	✔
`/echo` *string*	✔	`/rawsend` *x*	✔	`/ver`	✔
`/event` *n data*		`/rawxact` *x*	✔	`/write` *data*	
`/get` *prop*	✔	`/read` *[[x] y]*	✔		

Properties Supported								
Property	**Yes**	**Default**	**Property**	**Yes**	**Default**	**Property**	**Yes**	**Default**
`account_no`			`chk_mod10`			`msg3`		
`amount`			`chk_number`			`msg4`		
`applied_fmt`	✔		`chk_routing`			`offline_enc`		
`c_card_stat`			`chk_status`			`oper_tout`		
`c_cardwpin`			`chk_transit`			`pin_blk_fmt`		
`c_check`		0	`cmd_pending`			`pinfilldig`		
`c_events`			`dblpinentry`			`port_name`	✔	
`c_keypress`			`dev_status`	✔		`pwroffdelay`		
`c_keystring`			`dev_version`			`s_down_tout`		
`c_magnetic`	✔	1	`enable_cmc7`			`track1ss`	✔	
`c_mechanics`		0	`enc_key`			`track2ss`	✔	
`c_pin`			`enc_key_sn`			`track3ss`	✔	
`c_smart`		0	`enc_mode`			`trivpinchk`		
`c_tracks`	✔	111	`entry_echo`			`trk_enable`	✔	111
`c_write`		0	`entry_len`			`trk1data`	✔	
`c_wr_secure`			`entry_tout`			`trk2data`	✔	
`capitalize`	✔	1	`events_on`			`trk3data`	✔	
`card_stat`			`invalcmdrsp`	✔	0	`visa_mac1`		
`chk_account`			`key_parity`			`visa_mac2`		
`chk_amount`			`lasterr`	✔		`visa_mac3`		
`chk_bankid`			`max_pin_len`			`wr_coer`		
`chk_data`			`msg1`			`wr_secure`		
`chk_format`			`msg2`			`xact_type`		

* = Depends on setting in the device.

MINIWEDGE MSR

File Name MINIWEDG.VXD **Part Number** 30037340

Friendly Name(s) MiniWedge

Remarks When operating in the Windows Driver mode, the MiniWedge transmits data as ASCII characters instead of scan codes in order to reduce the transmission time. (A full 3-track card can be transmitted in about 0.5 second whereas in the non-driver mode it would take almost 4 seconds.) If this creates problems in certain hardware implementations, the *skip_ascii* and *dev_char_delay* parameters in the registry and/or INF file can be adjusted. The default setting for *dev_char_delay* is "01"; if this seems to be too fast, try setting this to "06". Additionally, the *skip_ascii* value can be set to true ("01") to transmit scan codes.

Commands Supported					
`/cancel` *cmd*	✔	`/load_key` *n key*		`/reset`	✔
`/display` *[x]*		`/rawrecv`	✔	`/set` *prop val*	✔
`/echo` *string*	✔	`/rawsend` *x*	✔	`/ver`	✔
`/event` *n data*		`/rawxact` *x*	✔	`/write` *data*	
`/get` *prop*	✔	`/read` *[[x] y]*	✔		

Properties Supported								
Property	**Yes**	**Default**	**Property**	**Yes**	**Default**	**Property**	**Yes**	**Default**
account_no			chk_mod10			msg3		
amount			chk_number			msg4		
applied_fmt	✔		chk_routing			offline_enc		
c_card_stat			chk_status			oper_tout		
c_cardwpin			chk_transit			pin_blk_fmt		
c_check		0	cmd_pending			pinfilldig		
c_events			dblpinentry			port_name	✔	
c_keypress			dev_status	✔		pwroffdelay		
c_keystring			dev_version	✔		s_down_tout		
c_magnetic	✔	1	enable_cmc7			track1ss	✔	
c_mechanics		0	enc_key			track2ss	✔	
c_pin			enc_key_sn			track3ss	✔	
c_smart		0	enc_mode			trivpinchk		
c_tracks	✔	*	entry_echo			trk_enable	✔	*
c_write		0	entry_len			trk1data	✔	
c_wr_secure			entry_tout			trk2data	✔	
capitalize	✔	1	events_on			trk3data	✔	
card_stat			invalcmdrsp	✔	0	visa_mac1		
chk_account			key_parity			visa_mac2		
chk_amount			lasterr	✔		visa_mac3		
chk_bankid			max_pin_len			wr_coer		
chk_data			msg1			wr_secure		
chk_format			msg2			xact_type		

* = Depends on setting in the device.

MICR+ CHECK READER & MSR

File Name MICRPLUS.VXD **Part Number** 30037349

Friendly Name(s) MICR+

Remarks These devices may or may not have an MSR installed. If not installed, the driver may not properly indicate the **c_tracks** capability.

Commands Supported					
/cancel *cmd*	✔	**/load_key** *n key*		**/reset**	✔
/display *[x]*		**/rawrecv**	✔	**/set** *prop val*	✔
/echo *string*	✔	**/rawsend** *x*	✔	**/ver**	✔
/event *n data*		**/rawxact** *x*	✔	**/write** *data*	
/get *prop*	✔	**/read** *[[x] y]*	✔		

Properties Supported								
Property	**Yes**	**Default**	**Property**	**Yes**	**Default**	**Property**	**Yes**	**Default**
account_no			chk_mod10	✔		msg3		
amount			chk_number	✔		msg4		
applied_fmt	✔		chk_routing	✔		offline_enc		
c_card_stat			chk_status	✔		oper_tout		
c_cardwpin			chk_transit	✔		pin_blk_fmt		
c_check	✔	1	cmd_pending			pinfilldig		
c_events			dblpinentry			port_name	✔	
c_keypress			dev_status	✔		pwroffdelay		
c_keystring			dev_version	✔		s_down_tout		
c_magnetic	✔	1	enable_cmc7	✔	*	track1ss	✔	
c_mechanics		0	enc_key			track2ss	✔	
c_pin			enc_key_sn			track3ss	✔	
c_smart		0	enc_mode			trivpinchk		
c_tracks	✔	*	entry_echo			trk_enable	✔	*
c_write		0	entry_len			trk1data	✔	
c_wr_secure			entry_tout			trk2data	✔	
capitalize	✔	1	events_on			trk3data	✔	
card_stat			invalcmdrsp	✔	0	visa_mac1		
chk_account	✔		key_parity			visa_mac2		
chk_amount	✔		lasterr	✔		visa_mac3		
chk_bankid	✔		max_pin_len			wr_coer		
chk_data	✔		msg1			wr_secure		
chk_format	✔	**	msg2			xact_type		

* = Depends on setting in the device.

** = Depends on setting in INF file (default = 6500). See Section 1, "MICR Format Numbers" for more information.

MINI MICR CHECK READER & MSR

File Name MINIMICR.VXD **Part Number** 30037344

Friendly Name(s) Mini MICR RS-232 & Mini MICR Wedge

Remarks These devices may or may not have an MSR installed. If not installed, the driver may not properly indicate the **c_tracks** capability.

Commands Supported					
/cancel *cmd*	✔	**/load_key** *n key*		**/reset**	✔
/display *[x]*		**/rawrecv**	✔	**/set** *prop val*	✔
/echo *string*	✔	**/rawsend** *x*	✔	**/ver**	✔
/event *n data*		**/rawxact** *x*	✔	**/write** *data*	
/get *prop*	✔	**/read** *[[x] y]*	✔		

Properties Supported								
Property	**Yes**	**Default**	**Property**	**Yes**	**Default**	**Property**	**Yes**	**Default**
account_no			chk_mod10	✔		msg3		
amount			chk_number	✔		msg4		
applied_fmt	✔		chk_routing	✔		offline_enc		
c_card_stat			chk_status	✔		oper_tout		
c_cardwpin			chk_transit	✔		pin_blk_fmt		
c_check	✔	1	cmd_pending			pinfilldig		
c_events			dblpinentry			port_name	✔	
c_keypress			dev_status	✔		pwroffdelay		
c_keystring			dev_version	✔		s_down_tout		
c_magnetic	✔	1	enable_cmc7	✔	*	track1ss	✔	
c_mechanics		0	enc_key			track2ss	✔	
c_pin			enc_key_sn			track3ss	✔	
c_smart		0	enc_mode			trivpinchk		
c_tracks	✔	*	entry_echo			trk_enable	✔	*
c_write		0	entry_len			trk1data	✔	
c_wr_secure			entry_tout			trk2data	✔	
capitalize	✔	1	events_on			trk3data	✔	
card_stat			invalcmdrsp	✔	0	visa_mac1		
chk_account	✔		key_parity			visa_mac2		
chk_amount	✔		lasterr	✔		visa_mac3		
chk_bankid	✔		max_pin_len			wr_coer		
chk_data	✔		msg1			wr_secure		
chk_format	✔	**	msg2			xact_type		

* = Depends on setting in the device.

** = Depends on setting in INF file (default = 6500). See "See Section 1, "MICR Format Numbers" for more information.

PORT-POWERED RS-232 SWIPE READER

File Name MTPPSWIP.VXD **Part Number** 30037346

Friendly Name(s) Port-powered swipe reader

Remarks This driver supports all port-powered swipe readers.

Commands Supported					
`/cancel` *cmd*	✔	`/load_key` *n key*		`/reset`	✔
`/display` *[x]*		`/rawrecv`	✔	`/set` *prop val*	✔
`/echo` *string*	✔	`/rawsend` *x*	✔	`/ver`	✔
`/event` *n data*		`/rawxact` *x*	✔	`/write` *data*	
`/get` *prop*	✔	`/read` *[[x] y]*	✔		

Properties Supported								
Property	**Yes**	**Default**	**Property**	**Yes**	**Default**	**Property**	**Yes**	**Default**
account_no			chk_mod10			msg3		
amount			chk_number			msg4		
applied_fmt	✔		chk_routing			offline_enc		
c_card_stat			chk_status			oper_tout		
c_cardwpin			chk_transit			pin_blk_fmt		
c_check		0	cmd_pending			pinfilldig		
c_events			dblpinentry			port_name	✔	
c_keypress			dev_status	✔		pwroffdelay		
c_keystring			dev_version	✔		s_down_tout		
c_magnetic	✔	1	enable_cmc7			track1ss	✔	
c_mechanics		0	enc_key			track2ss	✔	
c_pin			enc_key_sn			track3ss	✔	
c_smart		0	enc_mode			trivpinchk		
c_tracks	✔	*	entry_echo			trk_enable	✔	*
c_write		0	entry_len			trk1data	✔	
c_wr_secure			entry_tout			trk2data	✔	
capitalize	✔	1	events_on			trk3data	✔	
card_stat			invalcmdrsp	✔	0	visa_mac1		
chk_account			key_parity			visa_mac2		
chk_amount			lasterr	✔		visa_mac3		
chk_bankid			max_pin_len			wr_coer		
chk_data			msg1			wr_secure		
chk_format			msg2			xact_type		

* = Depends on setting in the device.

PORT-POWERED RS-232 INSERTION READER

File Name MTPPINSR.VXD **Part Number** 30037339

Friendly Name(s) Port-powered insert reader

Remarks If **events_on** is enabled, the driver will send **/event 1 M** when the card is inserted. It is suggested that events be disabled (**/set events_on 0**) before the data is read to prevent the removal event from being included at the end of card data. If a card has already been inserted when the driver is opened, there will not be any notification when **events_on** is enabled. Consequently, it is recommended that **/get card_stat** be issued immediately after opening the driver to see if a card is blocking the sensor.

Commands Supported					
/cancel *cmd*	✔	**/load_key** *n key*		**/reset**	✔
/display *[x]*		**/rawrecv**	✔	**/set** *prop val*	✔
/echo *string*	✔	**/rawsend** *x*	✔	**/ver**	✔
/event *n data*	✔	**/rawxact** *x*	✔	**/write** *data*	
/get *prop*	✔	**/read** *[[x] y]*	✔		

Properties Supported								
Property	**Yes**	**Default**	**Property**	**Yes**	**Default**	**Property**	**Yes**	**Default**
account_no			chk_mod10			msg3		
amount			chk_number			msg4		
applied_fmt	✔		chk_routing			offline_enc		
c_card_stat	✔	1	chk_status			oper_tout		
c_cardwpin			chk_transit			pin_blk_fmt		
c_check		0	cmd_pending			pinfilldig		
c_events	✔	1	dblpinentry			port_name	✔	
c_keypress			dev_status	✔		pwroffdelay		
c_keystring			dev_version	✔		s_down_tout		
c_magnetic	✔	1	enable_cmc7			track1ss	✔	
c_mechanics		0	enc_key			track2ss	✔	
c_pin			enc_key_sn			track3ss	✔	
c_smart		0	enc_mode			trivpinchk		
c_tracks	✔	110	entry_echo			trk_enable	✔	110
c_write		0	entry_len			trk1data	✔	
c_wr_secure			entry_tout			trk2data	✔	
capitalize	✔	1	events_on	✔	0	trk3data	✔	
card_stat	✔		invalcmdrsp	✔	0	visa_mac1		
chk_account			key_parity			visa_mac2		
chk_amount			lasterr	✔		visa_mac3		
chk_bankid			max_pin_len			wr_coer		
chk_data			msg1			wr_secure		
chk_format			msg2			xact_type		

* = Depends on setting in the device.

MT-85 LOCO ENCODER

File Name MT85.VXD **Part Number** 30037337

Friendly Name(s) MT-85

Remarks The driver attempts to connect to the device by automatically scanning all connection modes.

Commands Supported					
`/cancel` *cmd*	✔	`/load_key` *n key*		`/reset`	✔
`/display` *[x]*		`/rawrecv`	✔	`/set` *prop val*	✔
`/echo` *string*	✔	`/rawsend` *x*	✔	`/ver`	✔
`/event` *n data*		`/rawxact` *x*	✔	`/write` *data*	✔
`/get` *prop*	✔	`/read` *[[x] y]*	✔		

Properties Supported								
Property	**Yes**	**Default**	**Property**	**Yes**	**Default**	**Property**	**Yes**	**Default**
account_no			chk_mod10			msg3		
amount			chk_number			msg4		
applied_fmt	✔		chk_routing			offline_enc		
c_card_stat			chk_status			oper_tout		
c_cardwpin		0	chk_transit			pin_blk_fmt		
c_check		0	cmd_pending	✔		pinfilldig		
c_events			dblpinentry			port_name	✔	
c_keypress		0	dev_status	✔		pwroffdelay		
c_keystring		0	dev_version	✔		s_down_tout		
c_magnetic	✔	1	enable_cmc7			track1ss	✔	
c_mechanics		0	enc_key			track2ss	✔	
c_pin		0	enc_key_sn			track3ss	✔	
c_smart		0	enc_mode			trivpinchk		
c_tracks	✔	*	entry_echo			trk_enable	✔	*
c_write	✔	2	entry_len			trk1data	✔	
c_wr_secure	✔	0	entry_tout			trk2data	✔	
capitalize	✔	1	events_on			trk3data	✔	
card_stat			invalcmdrsp	✔	0	visa_mac1		
chk_acount			key_parity			visa_mac2		
chk_amount			lasterr	✔		visa_mac3		
chk_bankid			max_pin_len			wr_coer	✔	1
chk_data			msg1			wr_secure		
chk_format			msg2			xact_type		

* = Depends on setting in the device.

MT-95 HICO ENCODER

File Name MT95.VXD **Part Number** 30037347
Friendly Name(s) MT-95
Remarks

Commands Supported					
`/cancel` *cmd*	✔	`/load_key` *n key*		`/reset`	✔
`/display` *[x]*		`/rawrecv`	✔	`/set` *prop val*	✔
`/echo` *string*	✔	`/rawsend` *x*	✔	`/ver`	✔
`/event` *n data*		`/rawxact` *x*	✔	`/write` *data*	✔
`/get` *prop*	✔	`/read` *[[x] y]*	✔		

Properties Supported								
Property	**Yes**	**Default**	**Property**	**Yes**	**Default**	**Property**	**Yes**	**Default**
account_no			chk_mod10			msg3		
amount			chk_number			msg4		
applied_fmt	✔		chk_routing			offline_enc	✔	*
c_card_stat			chk_status			oper_tout		
c_cardwpin			chk_transit			pin_blk_fmt		
c_check		0	cmd_pending	✔		pinfilldig		
c_events			dblpinentry			port_name	✔	
c_keypress			dev_status	✔		pwroffdelay		
c_keystring			dev_version	✔		s_down_tout		
c_magnetic	✔	1	enable_cmc7			track1ss	✔	
c_mechanics		0	enc_key			track2ss	✔	
c_pin			enc_key_sn			track3ss	✔	
c_smart		0	enc_mode			trivpinchk		
c_tracks	✔	111	entry_echo			trk_enable	✔	*
c_write	✔	1	entry_len			trk1data	✔	
c_wr_secure	✔	1	entry_tout			trk2data	✔	
capitalize	✔	1	events_on			trk3data	✔	
card_stat			invalcmdrsp	✔	0	visa_mac1		
chk_account			key_parity			visa_mac2		
chk_amount			lasterr	✔		visa_mac3		
chk_bankid			max_pin_len			wr_coer	✔	*
chk_data			msg1			wr_secure	✔	*
chk_format			msg2			xact_type		

* = Depends on setting in the device.

INDEX

INSTALLATION GUIDE

LD9000 series

Serial Interface Customer Pole Displays

LD9000, LD9200, LD9300, LD9400, LD9500 and LD9900
LD9000X, LD9200X, LD9300X, LD9400X, LD9500X and LD9900X

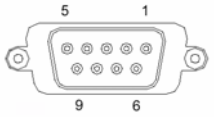
INSTALLATION

Your LD9000 family of pole displays has been pre-assembled to make the installation as simple as possible.

1. Mount the pole display to the metal base plate using the mounting hardware provided.
2. The pole display can be used in a freestanding mode or attached to the counter using the remaining mounting hardware.
3. Connect the round DIN6M connector from the pole display to the round DIN6F connector of the interface cable.
4. Connect the DB9F connector to the computer's serial COM1 or COM2 port.
5. Connect the female phone jack of the power adapter to the male phone jack of the interface cable.
6. Plug the power adapter into a 120VAC outlet.
7. A start up text message (LOGIC CONTROLS POS COMPONENTS) will be present for a short time. When this message disappears the cursor will be displayed at the left-most digit of the top row.

Serial Interface Connector Pinout

1. DCD (tied to pins 4&6)
2. NC
3. RXD from PC
4. DTR (tied to pins 1&6)
5. Ground
6. DSR (tied to pins 1&4)
7. RTS (tied to pin 8)
8. CTS (tied to pin 7)
9. NC



DB9F (to computer)

FUNCTIONAL TEST

The following test sequence will verify that your pole display is working properly. Before you start this procedure, you must install the pole display correctly as outlined under the INSTALLATION section. The functional test should be done under *MSDOS command prompt* by booting up the computer in DOS mode, or shell out to DOS prompt (in Window95/98/ME) or COMMAND prompt (in Windows NT/2000).

NOTE: The actual key entries in the text below are enclosed within quotation marks ("). Do **not** type the quotation marks as part of your entries.

This test procedure assumes the pole display is connected to COM1 of the computer. If COM2 is being used, type COM2 where COM1 is called out.

Close all opened application programs that use the same COM port before shelling out to DOS COMMAND prompt. Enter the following command lines to open the COM port for communication with the pole display:

Type "MODE COM1 96,N,8,1" and press the ENTER key.
Type "TYPE CON>COM1" and press the ENTER key.

LD9000, LD9300, LD9400, LD9500, LD9900

1. Type "ABCDEFGH" and press ENTER key. The display will show "ABCDEFGH" on the first line.
2. To end the test press Ctrl-C (hold down Ctrl and press C).

LD9200

1. Type "!#1ABCDEFGH" and press ENTER key. The display will show "ABCDEFGH" on the first line.
2. To end the test press Ctrl-C (hold down Ctrl and press C).

SOFTWARE COMMANDS

Logic Controls pole displays are controlled by command codes and data from the computer. Commands are transmitted to the pole display as ASCII codes. The command codes listed below are expressed in hexadecimal (base 16) numbers enclosed inside angle brackets < >, in decimal numbers enclosed in parenthesis ( ), and in ASCII characters enclosed in curly brackets { }. Do **not** include the brackets as part of the command. ' ^ ' character denotes 'Ctrl' in the keyboard. Press and hold 'Ctrl', then press the next key.

LOGIC CONTROLS COMMAND SET (LD9000, LD9000X):

1. **Vertical Scroll Mode <12>, (18), {^R}:**
   Data is written into the second row and scrolled to the first row when carriage return is received, leaving the second row empty.
2. **Normal Display Mode <11>, (17), {^Q}:**
   Data can be written into either row. Moves to the left most digit of the other row when line is full.
3. **Brightness Control <04>, (04), {^D}:**
   Brightness of the display can be adjusted with this command followed by data byte <FF>, <60>, <40> or <20>.
4. **Back Space <08>, (08), {^H}:**
   The cursor position moves one digit to the left erasing the previous information.
5. **Horizontal Tab <09>, (09), {^I}:**
   The cursor position shifts one digit to the right without erasing character at original cursor position.
6. **Line Feed <0A>, (10), {^J}:**
   The cursor position moves to the same position in the other row. In vertical scroll mode, if cursor was in second row, the cursor will not move and display will scroll up.
7. **Carriage Return <0D>, (13), {^M}:**
   The cursor moves to the left most digit of the row it is in.
8. **Digit Select <10>, (16), {^P}:**
   Moves the cursor to any position on the display with this command followed by data byte <00> to <27>.
9. **Cursor On <13>, (19), {^S}:**
   Turns on the cursor.
10. **Cursor Off <14>, (20), {^T}:**
    Turns off the cursor.
11. **Reset <1F>, (31), {^_}:**
    All characters are erased and all settings are returned to the power-on reset conditions.
12. **Down Load Font <03><X> <F> <F> <F> <F> <F>:**
    Assign a keyboard key (ASCII code <20> to <7F>) to a different style font. The "X" represents the ASCII code for the selected key. The "F's" represent the 5 segments that will make up the special font.
13. **Message Scroll <05><X><X><X> ... <X><0D>:**
    Scrolls a continuous message of up to 45 characters from right to left on the top line.
14. **Message Scroll Left on Bottom Line <1B><06><X><X><X> ... <X><0D>:**
    Scrolls a continuous message of up to 45 characters from right to left on the bottom line.
15. **Message Scroll Right on Top Line <1B><07><X><X><X> ... <X><0D>:**
    Scrolls a continuous message of up to 45 characters from left to right on the top line.
16. **Message Scroll Right on Bottom Line <1B><0B><X><X><X> ... <X><0D>:**
    Scrolls a continuous message of up to 45 characters from left to right on the bottom line.
17. **Both Side Display <1B><0C>, (27)(12), {^[ ]{^L}:**
    After this command, all messages followed are displayed on both sides of double sided displays.
18. **Front Side Display <1B><0E>, (27)(14), {^[ ]{^N}:**
    After this command, all messages followed are displayed only on front side of double sided displays.
19. **Back Side Display <1B><0F>, (27)(15), {^[ ]{^O}:**
    After this command, all messages followed are displayed only on back side of double sided displays.
20. **Clock Display <1B><1A><h><h><3A><m><m>:**
    Displays real time clock on bottom line in the 12 hour format hh:mm.
21. **Smart Message Scroll Left on Top Line <1B><15><X><X><X> ... <1C> … <1C> … <X><0D>:**
    Scrolls a multi-part message of up to total of 45 characters from right to left on the top line. Message parts are separated by <1C> and each part must be less than 20 characters.
22. **Smart Message Scroll Left on Bottom Line <1B><16><X><X><X> ... <1C> … <1C> … <X><0D>:**
    Scrolls a multi-part message from right to left on the bottom line.
23. **Smart Message Scroll Right on Top Line <1B><13><X><X><X> ... <1C> … <1C> … <X><0D>:**
    Scrolls a multi-part message from left to right on the top line.
24. **Smart Message Scroll Right on Bottom Line <1B><14><X><X><X> ... <1C> … <1C> … <X><0D>:**
    Scrolls a multi-part message from left to right on the bottom line.

GENERAL SPECIFICATIONS

	LD9000	***LD9000X***
OPTICAL:		
Number of rows	2	2 x 2 sides
Number of digits/row	20	20
Dot matrix	5 x 7	5X7
Digit height	0.374in. (9.5mm)	0.374in. (9.5mm)
Digit width	0.244in. (6.2mm)	0.244in. (6.2mm)
Character format	ASCII	ASCII
Brightness (typical)	900 cd/m^2	900 cd/m^2
Display color	Blue-Green	Blue-Green
MTBF (hours)	300,000	300,000
MECHANICAL:		
Weight	2.7 lb.	2.7 lb.
Dimensions (in inches)	(w x h x d)	(w x h x d)
Display head	8.50 x 3.37 x 1.75	8.50 x 3.37 x 2.25
Rectangular base	2.12 x 2.00 x 2.25	2.12 x 2.00 x 2.25
Base plate	4.0 x 0.09 x 8.0	4.0 x 0.09 x 8.0
Overall height (typical)	8 to 25	8 to 25
ELECTRICAL:		
Adapter input power	120VAC, 60Hz; optional 220VAC, 50Hz	
Adapter output power	7.5VAC, 1000mA	
ENVIRONMENTAL:		
Operating temperature	0 to +50 °C	
Storage temperature	-20 to +70 °C	
Relative Humidity	80%, non-condensing	
Vibration (10 to 55 Hz.)	4G's	
Shock	40G's	
CABLES & CONNECTORS:		
Serial		
Display cable	6-pin DIN (male)	
Interface cable	6-pin DIN (female)	
	DB9 (female); optional DB25 (female)	
	Phone jack (male)	
Power adapter	Phone jack (female)	

INSTALLATION GUIDE

LD9000U series

USB Interface Customer Pole Displays

LD9000U, LD9200U, LD9300U, LD9400U, LD9500U, LD9900U, LD9000XU, LD9200XU, LD9300XU, LD9400XU, LD9500XU and LD9900XU

HARDWARE INSTALLATION

Your LD9000U family of pole displays has been pre-assembled to make the installation as simple as possible.

1. Mount the pole display to the metal base plate using the mounting hardware provided.
2. The pole display can be used in a freestanding mode or attached to the counter using the remaining mounting hardware.
3. Connect the female phone jack of the power adapter to the power socket from the pole display.
4. Plug the power adapter into a 120VAC power outlet. Display should show a power on start up message.
5. Plug the USB cable from the pole display into the computer's USB port.

DRIVER INSTALLATION

When hardware is installed correctly, Windows will detect the USB pole display and start device driver installation. Insert the driver diskette into floppy disk drive A and follow instructions on screen to install the driver.

There are two types of drivers available. First one use a specific device name (\\.\LCLD9) to access the pole display directly. Select this driver if you are using OPOS or the POS software was tailored to use this device name. If the POS software is only able to access COM ports for the pole display, use the Virtual COM port device driver.

Installing USB Device Name Drivers:

For Windows 98/ME:

1. When Windows tried to search for a driver, specify a location "A:\Win98" and click [Next].
2. Windows will locate the driver "A:\Win98\LCLD9.inf". Click [Next] to install.
3. If Windows prompts for inserting driver disk again, click [OK] to continue and enter "A:\Win98" in the "Copy files from" dialog box. Then click [OK] to continue.
4. When finished copying the drivers, click [Finish].
5. Remove driver disk and reboot computer.
6. Go to "Device Manager" to check that the pole display has been installed correctly under "Universal Serial Bus Controllers" section.

For Windows 2000/XP:

1. When Windows tried to search for a driver, click on the check box "Specify a location " and click [Next]. Enter "A:\Win2000" for the location and click [Next].
2. If Windows indicates that it also found other drivers that are suitable, click on the check box "Install one of the other drivers".
3. Select driver A:\Win2000\LCLD9.inf" and click [Next].
4. When finished installing, click [Finish].
5. Go to "Device Manager" to check that the pole display has been installed correctly under "Universal Serial Bus Controllers" section.

Installing Virtual COM Port Device Drivers:

For Windows 98/ME

1. When Windows tried to search for a driver, specify a location "A:\W98com" and click [Next].
2. Windows will locate the driver "A:LCLD9usb.inf". Click [Next] to install.
3. If Windows prompts for inserting driver disk again, click [OK] to continue and enter "A:\W98com" in the "Copy files from" dialog box. Then click [OK] to continue.
4. When finished copying the drivers, click [Finish].
5. Remove driver disk and reboot computer.
6. Click on [Start] -> [Settings] -> [Control Panel].
7. Click on [Add/Remove Hardware] and then [Next].
8. Windows will search for Plug & Play devices. Click [Next] to continue.
9. When prompted, click on the check box for "No, the device isn't in the list" and click [Next].
10. When prompted again, click on the check box for "No, I want to select the hardware from a list" and click [Next].
11. In the device list, select "Ports (COM & LPT)" and click [Next].
12. Click on [Have Disk] and select "LCI LCLD9 COM Port". Then click [Next].
13. There is no need to change hardware settings. Just click [Next] to continue installation.
14. When finished installing, click [Finish].
15. Remove driver disk and reboot computer.
16. Go to "Device Manager" to check that the pole display has been installed correctly under "Ports (COM & LPT)" section and note down the Port number.

For Windows 2000/XP:

1. When Windows tried to search for a driver, click on the check box "Specify a location " and click [Next]. Enter "A:\W2kCom" for the location and click [Next].
2. If Windows indicates that it also found other drivers that are suitable, click on the check box "Install one of the other drivers".
3. Select driver A:\W2kCom\LD9COM.inf" and click [Next].
4. When finished installing, click [Finish].
5. Go to "Device Manager" to check that the pole display has been installed correctly under "Ports (COM & LPT)" section and note down the Port number.

SOFTWARE COMMANDS

Logic Controls pole displays are controlled by command codes and data from the computer. Commands are transmitted to the pole display as ASCII codes. The command codes listed below are expressed in hexadecimal (base 16) numbers enclosed inside angle brackets < >, in decimal numbers enclosed in parenthesis ( ), and in ASCII characters enclosed in curly brackets { }. Do **not** include the brackets as part of the command. ' ^ ' character denotes 'Ctrl' in the keyboard. Press and hold 'Ctrl', then press the next key.

LOGIC CONTROLS COMMAND SET (LD9000U, LD9000XU):

1. **Vertical Scroll Mode <12>, (18), {^R}:**
   Data is written into the second row and scrolled to the first row when carriage return is received, leaving the second row empty.
2. **Normal Display Mode <11>, (17), {^Q}:**
   Data can be written into either row. Moves to the left most digit of the other row when line is full.
3. **Brightness Control <04>, (04), {^D}:**
   Brightness of the display can be adjusted with this command followed by data byte <FF>, <60>, <40> or <20>.
4. **Back Space <08>, (08), {^H}:**
   The cursor position moves one digit to the left erasing the previous information.
5. **Horizontal Tab <09>, (09), {^I}:**
   The cursor position shifts one digit to the right without erasing character at original cursor position.
6. **Line Feed <0A>, (10), {^J}:**
   The cursor position moves to the same position in the other row. In vertical scroll mode, if cursor was in second row, the cursor will not move and display will scroll up.
7. **Carriage Return <0D>, (13), {^M}:**
   The cursor moves to the left most digit of the row it is in.
8. **Digit Select <10>, (16), {^P}:**
   Moves the cursor to any position on the display with this command followed by data byte <00> to <27>.
9. **Cursor On <13>, (19), {^S}:**
   Turns on the cursor.
10. **Cursor Off <14>, (20), {^T}:**
    Turns off the cursor.
11. **Reset <1F>, (31), {^_}:**
    All characters are erased and all settings are returned to the power-on reset conditions.
12. **Down Load Font <03><X> <F> <F> <F> <F> <F>:**
    Assign a keyboard key (ASCII code <20> to <7F>) to a different style font. The "X" represents the ASCII code for the selected key. The "F's" represent the 5 segments that will make up the special font.
13. **Message Scroll <05><X><X><X> ... <X><0D>:**
    Scrolls a continuous message of up to 45 characters from right to left on the top line.
14. **Message Scroll Left on Bottom Line <1B><06><X><X><X> ... <X><0D>:**
    Scrolls a continuous message of up to 45 characters from right to left on the bottom line.
15. **Message Scroll Right on Top Line <1B><07><X><X><X> ... <X><0D>:**
    Scrolls a continuous message of up to 45 characters from left to right on the top line.
16. **Message Scroll Right on Bottom Line <1B><0B><X><X><X> ... <X><0D>:**
    Scrolls a continuous message of up to 45 characters from left to right on the bottom line.
17. **Both Side Display <1B><0C>, (27)(12), {^[ ]{^L}:**
    After this command, all messages followed are displayed on both sides of double sided displays.
18. **Front Side Display <1B><0E>, (27)(14), {^[ ]{^N}:**
    After this command, all messages followed are displayed only on front side of double sided displays.
19. **Back Side Display <1B><0F>, (27)(15), {^[ ]{^O}:**
    After this command, all messages followed are displayed only on back side of double sided displays.
20. **Clock Display <1B><1A><h><h><3A><m><m>:**
    Displays real time clock on bottom line in the 12 hour format hh:mm.
21. **Smart Message Scroll Left on Top Line <1B><15><X><X><X> ... <1C> … <1C> … <X><0D>:**
    Scrolls a multi-part message of up to total of 45 characters from right to left on the top line. Message parts are separated by <1C> and each part must be less than 20 characters.
22. **Smart Message Scroll Left on Bottom Line <1B><16><X><X><X> ... <1C> … <1C> … <X><0D>:**
    Scrolls a multi-part message from right to left on the bottom line.
23. **Smart Message Scroll Right on Top Line <1B><13><X><X><X> ... <1C> … <1C> … <X><0D>:**
    Scrolls a multi-part message from left to right on the top line.
24. **Smart Message Scroll Right on Bottom Line <1B><14><X><X><X> ... <1C> … <1C> … <X><0D>:**
    Scrolls a multi-part message from left to right on the bottom line.

INTERFACE TO SOFTWARE USING USB DEVICE NAME

For Non-OPOS application, the application software has to control pole display operation accordingly through the USB device port. To send data to the pole display, the application software has to open and access the device port with device name **\\.\lcld9**. Please see <Sample.c> source code listing for example of using the device name.

The pole display is controlled by command codes and data from the PC. Refer to the Command Set on the left or User Manual for details. Note that as DOS does not support USB, it may not be possible to test the USB pole display with DOS prompt commands. It can only be tested under application software.

GENERAL SPECIFICATIONS

	LD9000U	***LD9000XU***
OPTICAL:		
Number of rows	2	2 x 2 sides
Number of digits/row	20	20
Dot matrix	5 x 7	5X7
Digit height	0.374in. (9.5mm)	0.374in. (9.5mm)
Digit width	0.244in. (6.2mm)	0.244in. (6.2mm)
Character format	ASCII	ASCII
Brightness (typical)	900 cd/m^2	900 cd/m^2
Display color	Blue-Green	Blue-Green
MTBF (hours)	300,000	300,000
MECHANICAL:		
Weight	2.7 lb.	2.7 lb.
Dimensions (in inches)	(w x h x d)	(w x h x d)
Display head	8.50 x 3.37 x 1.75	8.50 x 3.37 x 2.25
Rectangular base	2.12 x 2.00 x 2.25	2.12 x 2.00 x 2.25
Base plate	4.0 x 0.09 x 8.0	4.0 x 0.09 x 8.0
Overall height (typical)	8 to 25	8 to 25
ELECTRICAL:		
Adapter input power	120VAC, 60Hz; optional 220VAC, 50Hz	
Adapter output power	7.5VAC, 1000mA	
ENVIRONMENTAL:		
Operating temperature	0 to +50 °C	
Storage temperature	-20 to +70 °C	
Relative Humidity	80%, non-condensing	
Vibration (10 to 55 Hz.)	4G's	
Shock	40G's	
CABLES & CONNECTORS:		
USB cable	4-pin Type A USB standard connector	
Power adapter	Phone jack (female)	

Models: LD9000 Series

Customer Displays

2 by 20 character display

USER MANUAL

NOTICE

The manufacturer of the POS pole display makes no representations or warranties, either expressed or implied, by or with respect to anything in this manual, and shall not be liable for any implied warranties of fitness for a particular purpose or for any indirect, special or consequential damages. Information in this document is subject to change without notice and does not represent a commitment on the part of the manufacturer.

FCC NOTICE

This equipment generates, uses, and can radiate radio frequency energy and if not installed and used in accordance with this manual, may cause interference to radio communications. It has been tested and found to comply with the limits for a Class A digital device pursuant to Subpart J of Part 15 of FCC Rules, which are designed to provide reasonable protection against interference when operated in a commercial environment. Operation of this equipment in a residential area is likely to cause interference in which case the user at his own expense will be required to take whatever measures may be required to correct the interference.

LOGIC CONTROLS, INC.
355 Denton Avenue
New Hyde Park, NY 11040
TEL: (516) 248-0400
FAX: (516) 248-0443
Email: lci@logiccontrols.com
http://www.logiccontrols.com

TABLE OF CONTENTS

FEATURES

The LD9000 family of pole displays offers a wide range of high quality features and models to choice from. Listed below are the features incorporated into each pole display. Not all features are available in all models. The model identification chart will assist you in selecting the model best suited to your needs.

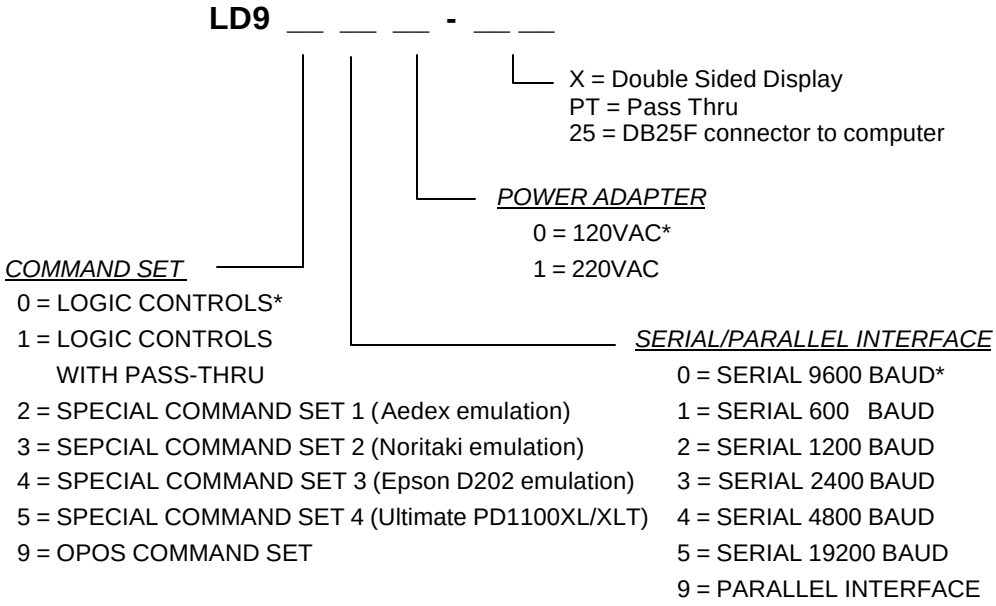
Features - All Models

- Bright blue-green fluorescent display
- Large 9mm character height
- Double sided display available
- Supports high speed serial protocol: up to 19,200 baud rate, 1 stop bit
- Automatic message scrolling
- Two line display with 20 characters per line
- Matched optical lens for better viewing contrast
- Ergonomic design
- Direct RS232C or parallel interface
- Long life and trouble free operation
- Five adjustable viewing angles
- Simple installation
- Available with 120V or 220V Power Adapters

Features - Model dependent

- Emulation of other popular command sets
- Double sided display
- User definable character
- Parallel pass-thru
- True RS232C pass-thru
- Real time clock
- One time message scrolling
- Ability to disable attention code
- Ability to change attention code

MODEL IDENTIFICATION



* Default Values
** Call for description of Command Sets

CARTON CONTENTS

1. Pole display, pre-assembled.
2. Interface cable, comes with a DB9F connector (to computer) and a DIN6F connector (to pole display) as standard equipment. Optional DB25F connector (to computer) is available.

   Note: This cable is only supplied with standard serial pole displays with no pass-thru function. Other interface options will have different cables.
3. Transformer adapter 120VAC to 6.0VAC (optional 220VAC).
4. Metal base plate with mounting hardware.
5. Quick Installation Guide.

INSTALLATION

Your PD9000 family of pole displays has been pre-assembled to make the installation as simple as possible.

Serial Interface Non-pass-thru Models Installation

1. Mount the pole display to the metal base plate using the mounting hardware provided.
2. The pole display can be used in a freestanding mode or attached to the counter using the remaining mounting hardware.
3. Connect the round DIN6M connector from the pole display to the round DIN6F connector of the interface cable.
4. Connect the DB9F connector to the computer's serial COM1 or COM2 port.
5. Connect the female phone jack of the power adapter to the male phone jack of the interface cable.
6. Plug the power adapter into a 120VAC outlet.
7. A start up text message (LOGIC CONTROLS POS COMPONENTS) will be present for a short time. When this message disappears the cursor will be displayed at the left-most digit of the top row.

Parallel Interface Non-pass-thru Models Installation

1. Mount the pole display to the metal base plate using the mounting hardware provided.
2. The pole display can be used in a freestanding mode or attached to the counter using the remaining mounting hardware.
3. Connect the DB25M connector to the computer's parallel printer port (LPT1).
4. Connect the female phone jack of the power adapter to the male phone jack of the pole display cable.
5. Plug the power adapter into a 120VAC outlet.
6. A start up text message (LOGIC CONTROLS POS COMPONENTS) will be present for a short time. When this message disappears the cursor will be displayed at the left-most digit of the top row.

Serial Interface Pass-thru Models Installation

1 Mount the pole display to the metal base plate using the mounting hardware provided.
2 The pole display can be used in a freestanding mode or attached to the counter using the remaining mounting hardware.
3 Connect the DB25M connector to the peripheral device or a serial pass-thru terminator (optional accessory). Turn on power of the peripheral device.
4 Connect the DB9F connector to the computer's serial COM1 or COM2 port.
5 Connect the female phone jack of the power adapter to the male phone jack of the pole display cable.
6 Plug the power adapter into a 120VAC outlet.
7 A start up text message (LOGIC CONTROLS POS COMPONENTS) will be present for a short time. When this message disappears the cursor will be displayed at the left-most digit of the top row.

Parallel Interface Pass-thru Models Installation

1 Mount the pole display to the metal base plate using the mounting hardware provided.
2 The pole display can be used in a freestanding mode or attached to the counter using the remaining mounting hardware.
3 Connect the DB25F connector to the peripheral device or parallel pass-thru terminator (optional accessory). Turn on power of the peripheral device.
4 Connect the DB25M connector to the computer's parallel printer port (LPT1).
5 Connect the female phone jack of the power adapter to the male phone jack at the DB25M/DB25F connector.
6 Plug the power adapter into a 120VAC outlet.
7 A start up text message (LOGIC CONTROLS POS COMPONENTS) will be present for a short time. When this message disappears the cursor will be displayed at the left-most digit of the top row.

FUNCTIONAL TEST

The following test sequence will verify that your pole display is working properly. Before you start this procedure, you must install the pole display correctly as outlined under the INSTALLATION section. The functional test should be done under *MSDOS command prompt* by booting up the computer in DOS mode, or shell out to DOS prompt (in window95/98/ME) or COMMAND prompt (in windows NT/2000).

For double sided displays, the messages will be shown on both sides at the same time.

NOTE: The actual key entries in the text below are enclosed within quotation marks ("). Do **not** type the quotation marks as part of your entries.

Serial Interface Pole Displays Functional Test

This test procedure assumes the pole display is connected to COM1 of the computer. If COM2 is being used, type COM2 where COM1 is called out.

Close all opened application programs that use the same COM port before going into DOS command prompt. Enter the following command lines to open the COM port for communication with the pole display:

Type "MODE COM1 96,N,8,1" and press the ENTER key.
Type "TYPE CON>COM1" and press the ENTER key.

LD9000, LD9300, LD9400, LD9500 and LD9900

1. Type "ABCDEFGH" and press ENTER key. The display will show "ABCDEFGH" on the first line.
2. To end the test press Ctrl-C (hold down Ctrl and press C).

LD9200

1. Type "!#1ABCDEFGH" and press ENTER key. The display will show "ABCDEFGH" on the first line.
2. To end the test press Ctrl-C (hold down Ctrl and press C).

LD9100-PT, LD9300-PT, LD9400-PT and LD9500-PT

1. Type “ABCDEFGH” and press ENTER key. The display will show “ABCDEFGH” on the first line.
2. Type “**^A**PASSTHRU” (**^A** is entered as Ctrl-A) and press ENTER key. The data will be passed through to the peripheral (e.g. a printer). These characters are not shown on the display.
3. Type “!#**^B**NUMBER12345” (**^B** is entered as Ctrl-B), then press ENTER key. The display will show “NUMBER12345” on the first line.
4. To end the test press Ctrl-C (hold down Ctrl and press C).

LD9200-PT

1. Type “!#1ABCDEFGH” and press ENTER key. The display will show “ABCDEFGH” on the first line.
2. Type “PASSTHRU” and press ENTER key. The data will be passed through to the peripheral (e.g. a printer). These characters are not shown on the display.
3. Type “!#1NUMBER12345”, then press ENTER key. The display will show “NUMBER12345” on the first line.
4. To end the test press Ctrl-C (hold down Ctrl and press C).

Parallel Interface Pole Displays Functional Test

This procedure assumes the pole display is connected to the LPT1 port of the computer. If parallel port LPT2 is being used, type LPT2 where LPT1 is called out.

LD9090, LD9390, LD9490, LD9590 and LD9990

1. Type “ECHO ABCDEFGH>LPT1” and press ENTER key. The display will show “ABCDEFGH” on the first line.

LD9290

1. Type “ECHO !#1ABCEDFGH>LPT1” and press ENTER key. The display will show “ABCDEFGH” on the first line.

LD9190-PT, LD9390-PT, LD9490-PT and LD9590-PT

1. Type “ECHO ABCDEFGH>LPT1” and press ENTER key. The display will show “ABCDEFGH” on the first line.
2. Type “ECHO **^A**PASSTHRU>LPT1” (**^A** is entered as Ctrl-A) and press ENTER key. The data will be passed through to the peripheral (e.g. a printer). These characters are not shown on the display.
3. Type “ECHO !#**^B**NUMBER12345>LPT1” (**^B** is entered as Ctrl-B), then press ENTER key. The display will show “NUMBER12345” on the first line.

LD9290-PT

1. Type “ECHO !#1ABCEDFGH>LPT1” and press ENTER key. The display will show “ABCDEFGH” on the first line.
2. Type “ECHO PASSTHRU>LPT1” and press ENTER key. The data will be passed through to the peripheral (e.g. a printer). These characters are not shown on the display.
3. Type “ECHO !#1NUMBER12345>LPT1”, then press ENTER key. The display will show “NUMBER12345” on the first line.

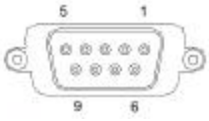
For further testing of OPOS displays (LD9900 and LD9990), follow the instructions in the readme.doc file after software installation.

INTERFACE CONNECTION

Serial Interface Connector Configuration

The pin out configuration for the standard serial pole display is a DB9F connector. It plugs directly into the serial port of the computer.

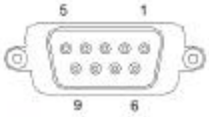
1. DCD (tied to pins 4&6)
2. NC
3. TXD from PC
4. DTR (tied to pins 1&6)
5. Ground
6. DSR (tied to pins 1&4)
7. RTS (tied to pin 8)
8. CTS (tied to pin 7)
9. NC



DB9F (to computer)

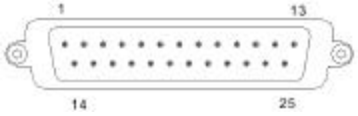
For pass-through models, the display cable comes with two connectors. The DB9F is to be connected to the computer while the DB25M is for connection to the peripheral.

1. DCD (tied to pins 4&6)
2 TXD to PC
3 RXD from PC
3. DTR (tied to pins 1&6)
4. Ground
5. DSR (tied to pins 1&4)
6. RTS (tied to pin 8)
7. CTS (tied to pin 7)
8. NC



DB9F (to computer)

1. NC
2. TXD to peripheral
3. RXD from peripheral
4. RTS
5. CTS (tied to pin 20)
6. NC
7. Ground
8. NC
9-19 NC
20 DTR (tied to pin 5)
21-25 NC

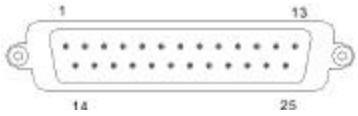


DB25M (to peripheral)

Parallel Interface Connector Configuration

All standard parallel pole displays will have a DB25 male connector. It is connected to the printer port of the computer.

1.	–Strobe
2.	Data 0
3.	Data 1
4.	Data 2
5.	Data 3
6.	Data 4
7.	Data 5
8.	Data 6
9.	Data 7
10.	–Ack
11.	Busy
12-17	NC
18-25	Ground



DB25M (to computer)

For pass-through models, the display cable comes with a dual DB25M/F connector. The DB25M is to be connected to the computer while the DB25F is for connection to the peripheral.

1.	–Strobe
2.	Data 0
3.	Data 1
4.	Data 2
5.	Data 3
6.	Data 4
7.	Data 5
8.	Data 6
9.	Data 7
10.	–Ack
11.	Busy
12.	Paper End
13.	Select
14.	-Auto Feed
15.	-Error
16.	-Initialize Printer
17.	-Select In
18-25.	Ground

DB25M (to computer)

DB25F (to peripheral)

SOFTWARE COMMANDS

Logic Controls pole displays are controlled by command codes and data from the computer. The model of pole display that you have will determine which command set works with your pole. Refer to the model identification chart for further information.

Commands are transmitted to the pole display as ASCII codes. The command codes listed below are expressed in hexadecimal (base 16) numbers enclosed inside angle brackets < >, in decimal numbers enclosed in parenthesis ( ), and in ASCII characters enclosed in curly brackets { }. Do not include the brackets as part of the command. ' ^ ' character denotes 'Ctrl' in the keyboard. Press and hold 'Ctrl', then press the next key.

LOGIC CONTROLS COMMAND SET (LD9000, LD9000-X):

1. **Vertical Scroll Mode <12>, (18), {^R}:**
   Data is written into the second row and transferred to the first row when carriage return is received, leaving the second row empty.

2. **Normal Display Mode <11>, (17), {^Q}:**
   Data can be written into either row. Moves to the left most digit of the other row when line is full.

3. **Brightness Control <04>, (04), {^D}:**
   The brightness of the display can be adjusted using this command followed by a data byte <FF>, <60>, <40> or <20>.

4. **Back Space <08>, (08), {^H}:**
   The cursor position moves one digit to the left erasing the previous information.

5. **Horizontal Tab <09>, (09), {^I}:**
   The cursor position shifts one digit to the right without erasing character at original cursor position.

6. **Line Feed <0A>, (10), {^J}:**
   The cursor position moves to the same position in the other row. In vertical scroll mode, if cursor was in second row, the cursor will not move and display will scroll up.

7. **Carriage Return <0D>, (13), {^M}:**
   The cursor moves to the left most digit of the row it is in.

8. **Digit Select <10>, (16), {^P}:**
   Moves the cursor to any position on the display with this command followed by a data byte of <00> to <27>, or in decimal (00) to (39).

9. **Cursor On <13>, (19), {^S}:**
   Turns on the cursor.

10. **Cursor Off <14>, (20), {^T}:**
    Turns off the cursor.

11. **Reset <1F>, (31), {^_}:**
    All characters are erased and all settings are returned to the power-on reset conditions.

12. **Down Load Font <03><X> <F> <F> <F> <F> <F>:**
    Assign a keyboard key (ASCII code <20> to <7F>) to a different style font. The “X” represents the ASCII code for the selected key. The “F’s” represent the 5 segments that will make up the special font.

13. **Message Scroll Left on Top Line**
    <05><X><X><X> ... <X><0D>:
    Scrolls a continuous message of up to 45 characters from right to left on the top line.

14. **Message Scroll Left on Bottom Line**
    <1B><06><X><X><X> ... <X><0D>:
    Scrolls a continuous message of up to 45 characters from right to left on the bottom line.

15. **Message Scroll Right on Top Line**
    <1B><07><X><X><X> ... <X><0D>:
    Scrolls a continuous message of up to 45 characters from left to right on the top line.

16. **Message Scroll Right on Bottom Line**
    <1B><0B><X><X><X> ... <X><0D>:
    Scrolls a continuous message of up to 45 characters from left to right on the bottom line.

17. **Both Side Display <1B><0C>, (27)(12), {^[ ]{^L}:**
    After this command, all messages followed are displayed on both sides of double sided displays.

18. **Front Side Display <1B><0E>, (27)(14), {^[ ]{^N}:**
    After this command, all messages followed are displayed only on front side of double sided displays.

19. **Back Side Display <1B><0F>, (27)(15), {^[ ]{^O}:**
    After this command, all messages followed are displayed only on back side of double sided displays.

20. **Clock Display <1B><1A><h><h><3A><m><m>:**
    Displays real time clock on bottom line in the 12 hour format hh:mm.

21. **Smart Message Scroll Left on Top Line**
    <1B><15><X><X><X> ... <1C> … <1C> … <X><0D>:
    Scrolls a multi-part message of up to total of 45 characters from right to left on the top line. Message parts are separated by <1C> and each part must be less than 20 characters.

22. **Smart Message Scroll Left on Bottom Line**
    <1B><16><X><X><X> ... <1C> … <1C> … <X><0D>:
    Scrolls a multi-part message of up to total of 45 characters from right to left on the bottom line. Message parts are separated by <1C> and each part must be less than 20 characters.

23. **Smart Message Scroll Right on Top Line**
    <1B><13><X><X><X> ... <1C> … <1C> … <X><0D>:
    Scrolls a multi-part message of up to total of 45 characters from left to right on the top line. Message parts are separated by <1C> and each part must be less than 20 characters.

24. **Smart Message Scroll Right on Bottom Line**
    <1B><14><X><X><X> ... <1C> … <1C> … <X><0D>:
    Scrolls a multi-part message of up to total of 45 characters from left to right on the bottom line. Message parts are separated by <1C> and each part must be less than 20 characters.

PASS-THRU COMMAND SET (LD9100-PT, LD9100X-PT):

All software commands of the non-pass-thru single sided model are available with following additional commands for pass-thru and double-sided display control. When power is turned on or after a reset command has been initiated, all text is displayed on the pole display.

Extended Pass-thru Command Set is available as option.

Standard Pass-thru Command Set

1. **Data to Peripheral <01>, (01), <^A>:**
   All data following this command will be sent to the peripheral until a “Data to Display” command is received.

2. **Data to Display <21><23><02>, (33)(35)(02), {!}{#}{^B}:**
   All data following this command will be sent to the pole display until a “Data to Peripheral” command is received.

Extended Pass-thru Command Set

1. **Data to Peripheral <01><1C><1D><1E>, (01)(28)(29)(30):**
   All data following this command will be sent to the peripheral until a “Data to Display” command is received.

2. **Data to Display <17><18><19><02>, (23)(24)(25)(02):**
   All data following this command will be sent to the pole display until a “Data to Peripheral” command is received.

HARDWARE CONFIGURATION

Serial pole displays were factory configured for serial RS232C interface using the following protocol:

- 9600 Baud Rate
- 8 Data Bits
- 1 Stop Bit
- No Parity

Other optional baud rates are available with factory settings. Refer to the model identification chart for further information.

DISPLAY CHARACTER CODES

				D7	0	0	0	0	0	0	0	0
				D6	0	0	0	0	1	1	1	1
				D5	0	0	1	1	0	0	1	1
				D4	0	1	0	1	0	1	0	1
D3	**D2**	**D1**	**D0**		***0***	***1***	***2***	***3***	***4***	***5***	***6***	***7***
0	0	0	0	***0***		DP	SP	0	@	P	‘	p
0	0	0	1	***1***		DC 1	!	1	A	Q	a	q
0	0	1	0	***2***		DC 2	"	2	B	R	b	r
0	0	1	1	***3***	DL	DC 3	#	3	C	S	c	s
0	1	0	0	***4***	DI M	DC 4	$	4	D	T	d	t
0	1	0	1	***5***	MS		%	5	E	U	e	u
0	1	1	0	***6***			&	6	F	V	f	v
0	1	1	1	***7***			’	7	G	W	g	w
1	0	0	0	***8***	BS		(	8	H	X	h	x
1	0	0	1	***9***	HT		)	9	I	Y	i	y
1	0	1	0	***A***	LF		*	:	J	Z	j	z
1	0	1	1	***B***			+	;	K	[	k	{
1	1	0	0	***C***	CR		,	<	L	\	l	|
1	1	0	1	***D***			-	=	M	]	m	}
1	1	1	0	***E***			.	>	N	^	n	~
1	1	1	1	***F***		RS T	/	?	O	_	o	

GENERAL SPECIFICATIONS

	LD9000	***LD900X***
OPTICAL:		
Number of rows	2	2 x 2 sides
Number of digits/row	20	20
Dot matrix	5 x 7	5 x 7
Digit height)	0.374in. (9.5mm)	0.374in. (9.5mm)
Digit width	0.244in. (6.2mm)	0.244in. (6.2mm)
Character config.	ASCII	ASCII
Brightness (typical)	900 cd/m^2	900 cd/m^2
Display color	Blue-Green	Blue-Green
MTBF (hours)	300,000	300,000
MECHANICAL:		
Weight	2.7 lb.	2.7 lb.
Dimensions (in inches)	(w x h x d)	(w x h x d)
Display head	8.50 x 3.37 x 1.75	8.50 x 3.37 x 1.75
Rectangular base	2.12 x 2.0 x 2.25	2.12 x 2.0 x 2.25
Base plate	4.0 x 0.09 x 8.0	4.0 x 0.09 x 8.0
Overall height (typical)	24	24
ELECTRICAL:		
Adapter input power	120VAC, 60Hz; optional 220VAC, 50Hz	
Adapter output power	7.5VAC, 1000mA	
ENVIRONMENTAL:		
Operating temperature	0 to +50 ^{o}C	
Storage temperature	-20 to +70 ^{o}C	
Relative Humidity	80%, non-condensing	
Vibration (10 to 55 Hz.)	4G's	
Shock	40G's	
CABLES & CONNECTORS:		
Serial		
Display cable	6-pin DIN (male)	
Interface cable	6-pin Din (female) DB9 (female); optional DB25 (female) Phone jack (male)	
Power adapter	Phone jack (female)	
Parallel		
Display cable	DB25 (female)	

FDU02

PC Peripheral Type,
USB Fingerprint Recognition Device

General Description

FDU02 is a Fingerprint Capture Device that supports the USB (Universal Serial Bus) interface and can do hot attachment, Plug & Play and alternate functions. The FDU02 can also support the PS/2 mouse interface and without a specific mouse driver. But for fingerprint sensor functions, the user must install a fingerprint sensor specific driver. The FDU02 uses bulk mode to transfer fingerprint image data and three bulk modes to control the USB interface board. Bulk mode allows multiple devices to be connected to a PC at one time.

Specifications

Supply Voltage:
- Min. 4.75V, Typ. 5.0V, Max. 5.25V

Current Consumption
- SecuGen Hamster III: 60mA (Max)
- SecuGen OptiMouse: 110mA (Max)
- Suspend current: 450uA (Max)

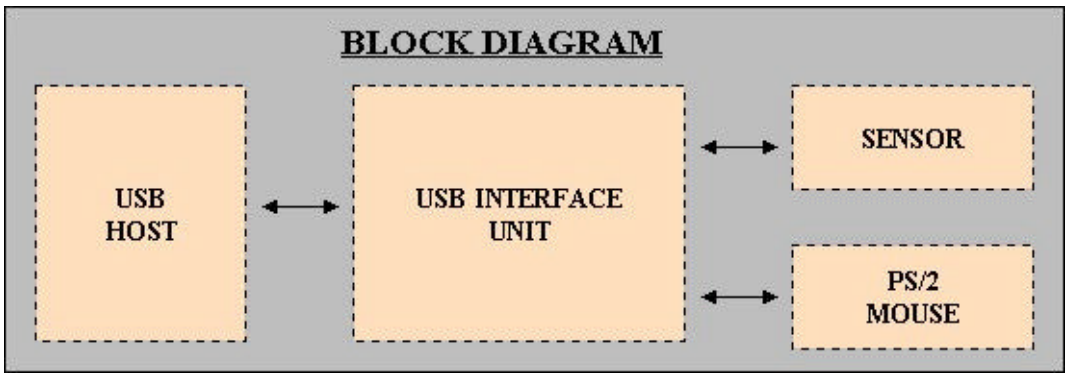
Operating Temperature: 0? ~60? (MAX)
Image Capture Speed: 400ms / frame
Pixel Resolution: 260 x 300
USB Signal Type:
- USB Spec 1.1, High Speed

Feature

- USB 1.1 compliant
- Supports alternate function
- Supports PS/2 mice Interface
- Supports remote wakeup
- No additional power supply needed (Bus-powered device)
- Immune to electrostatic discharge

Application

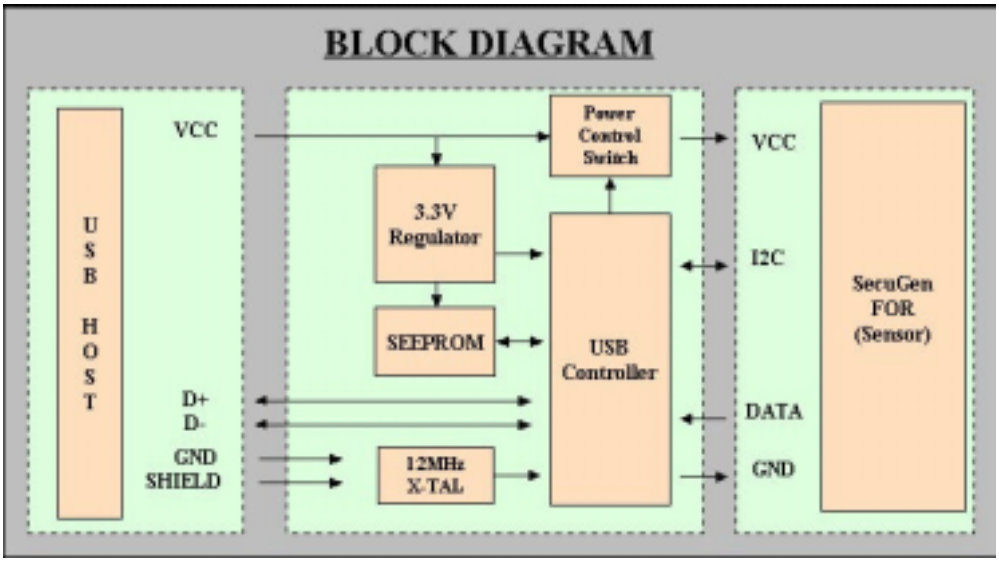
- ✓ Fingerprint Recognition Hamster
- ✓ Fingerprint Recognition Mouse
- ✓ Fingerprint Recognition Keyboard
- ✓ Fingerprint Recognition Smartcard
- ✓ Client/Server Solution



Theory of Operation

The USB host initiates the communication with the FDU02 using operation commands. Fingerprint data are then captured by the CMOS sensor at a total image size of 400 x 300 with 8-bit gray level. The image frame transfer speed is around 800 bytes/ms. It takes about 400 milliseconds to send one complete frame of image data over USB protocols. FDU02 uses the SecuGen Fingerprint Optic Reader. For developers, the SecuGen Software Development Kit (SDK) Collection provides the tools necessary to integrate this module into their own software applications using SecuGen's powerful Fingerprint Recognition Algorithm.

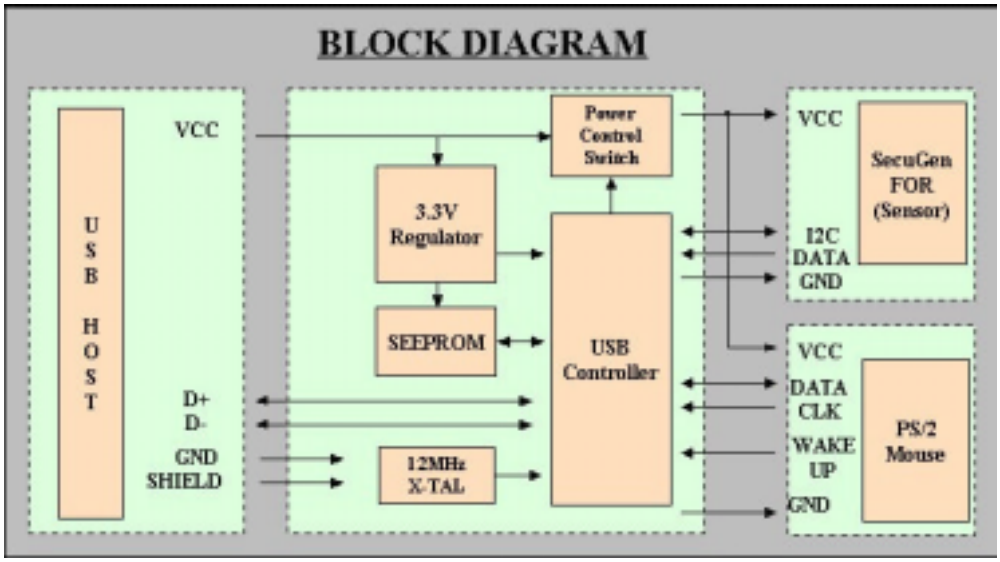
♦ **Supports only fingerprint optic recognition function**



FDU02

PC Peripheral Type,
USB Fingerprint Recognition Device

♦ **Supports both fingerprint optic recognition and mouse function**



Connector Pin Description

Name	No	States	Description
GND	1	Power	Sensor Ground
CLK	2	Input	Sensor Clock
SDATA0	3	Output	Sensor Data0
SDATA1	4	Output	Sensor Data1
SDATA2	5	Output	Sensor Data2
SDATA3	6	Output	Sensor Data3
DA	7	Input	Sensor DA
CL	8	Input	Sensor CL
VCC	9	Power	Sensor VCC (DC +5V)

Table 1 JP1 Pin Description

Name	No	States	Description
VCC	1	Power	VCC (DC +5V)
USB_A	2	I/O/Z	USBD -
USB_B	3	I/O/Z	USBD +
GND	4	Power	Ground
SHIELD	5	Power	Shield

Table 2 J1 Pin Description

Name	No	States	Description
GND	1	Power	Ground
PS/2 CLK	2	I/O	PS/2 CLOCK
PS/2 DATA	3	I/O	PS/2 DATA
VCC	4	Power	VCC (DC +5V)
WAKEUP	5	Input	Mouse wake-up
N.C	6	N.C	Reserved
OUT_VCC	7	Power	Controlled VCC (DC +5V)

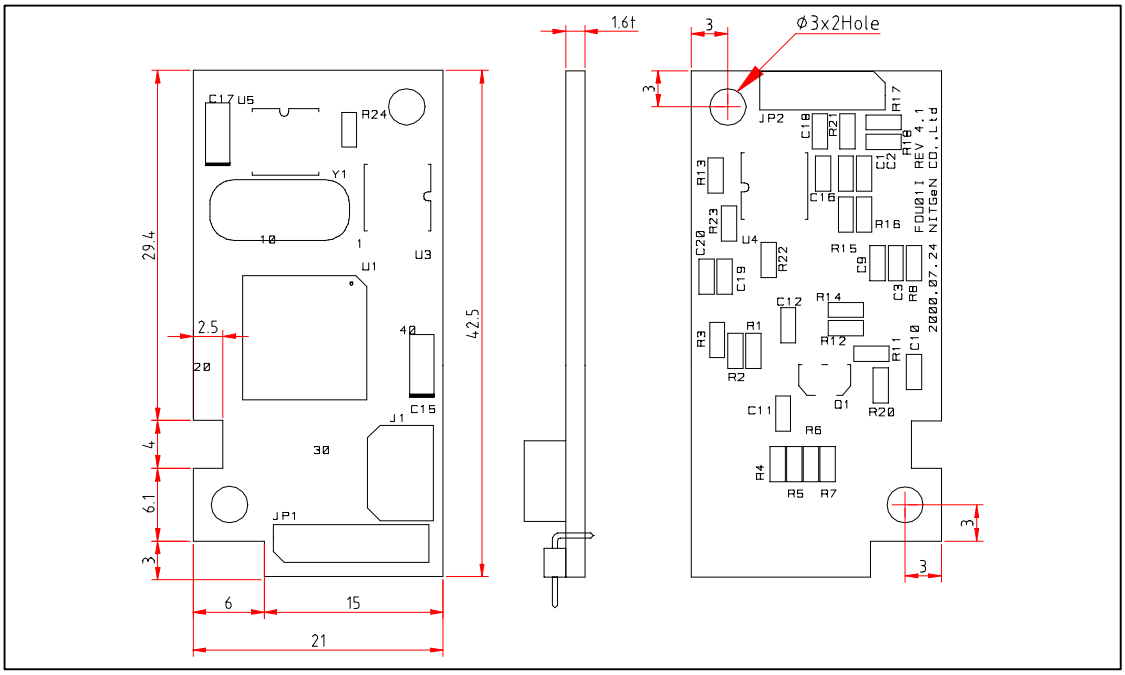
Table 3 JP2 Pin Description

FDU02

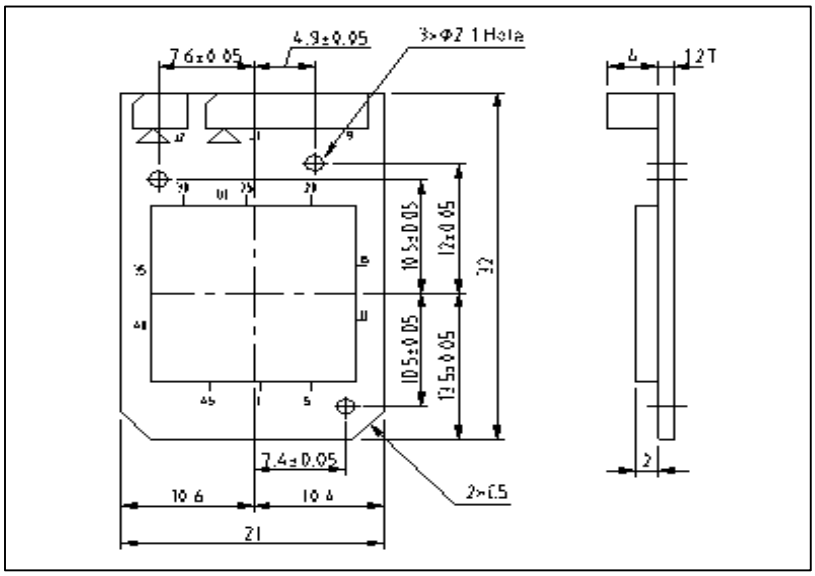
PC Peripheral Type,
USB Fingerprint Recognition Device

Outline Dimension

♦ **FDU02AI**



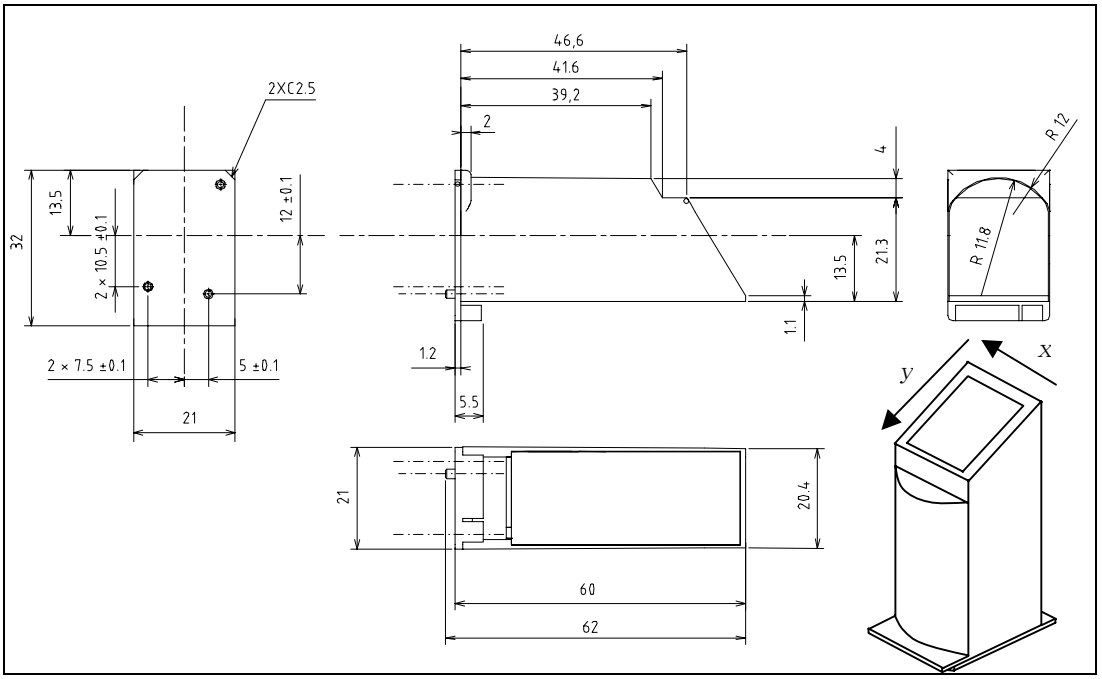
OPP02M PCB Dimensions



FDU02

PC Peripheral Type,
USB Fingerprint Recognition Device

Mechanical Dimensions(OPP02M)



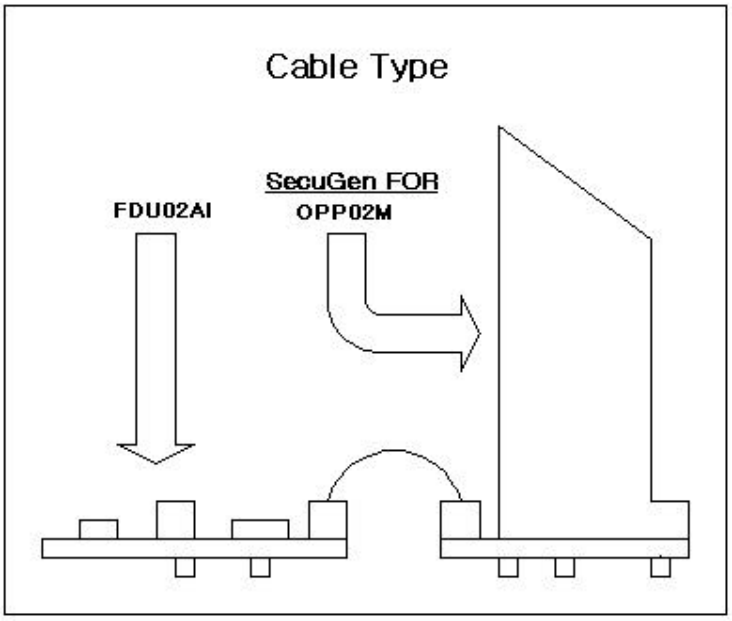
Optical Specification

Hardness of prism surface	750Hk
Resolution	500 ± 10 PPI
Effective Sensing Area	12.7mm (x) x 14.9mm (y)
Effective Pixel Array	400 x 300 (260 x 300 After Correction on X-axis)
Image Aspect Ratio	0.85 (x / y)
Magnification along x-direction	1.00000
Magnification along y-direction	0.641
Non-linear Distortion of Image	<3%
Ambient Light Condition	>5000 Lux

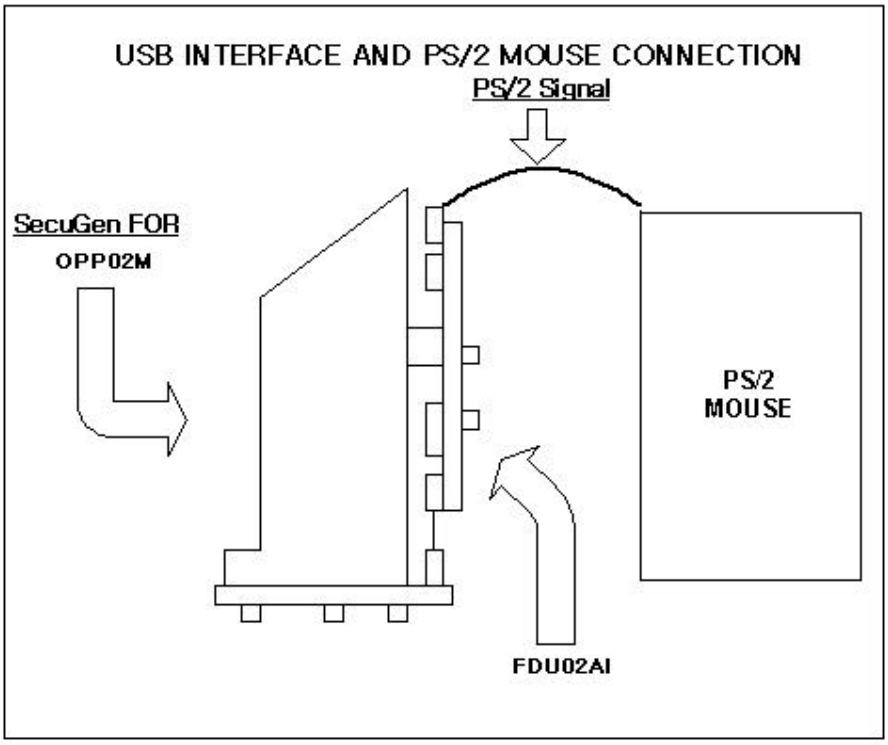
Mounting

♦ **Cable connection (Between USB interface and Sensor)**

FDU02AI USB interface board (FDU02AI) is cable-connected to the Fingerprint Sensor by a 9-pin cable connection (use the Molex 9 Pin Connector P/N 53048-0910).



♦ **Mouse Connection (Between USB interface and PS/2 mouse)**



Application

♦ SecuGen Hamster III (HFDU02A)

SecuGen Hamster III is the name of SecuGen's fingerprint recognition PC peripheral device that uses the FDU02. Using this device is simple and convenient and is used for substituting or enhancing the use of passwords, which are susceptible to theft or are difficult to remember.

This device can support Plug & Play, alternate function and hot attachment. Compared to a computer mouse, this device is small and has an ergonomic, compact design. It can be used for processing fingerprint image data and connect to both bus-powered hubs and self-powered hubs due to its low power consumption.

[Figure 1] SecuGen FDU02A USB

Check out Elo's Web site!

www.elotouch.com

Get the latest...

- **Product information**
- **Specifications**
- **News on upcoming events**
- **Press releases**
- **Software drivers**

Getting in Touch with Elo

To find out more about Elo's extensive range of touch solutions, visit our Web site at www.elotouch.com or simply call the office nearest you:

USA & Headquarters
Elo TouchSystems, Inc.
1000 Kaiser Drive
Fremont, CA 94555-3613

(800) ELO-TOUCH (800-356-8682)
Tel 1-510-739-5016
Fax 1-510-790-0621
elosinfo@elotouch.com

Germany
Elo TouchSystems GmbH & Co. KG
Hardgraben 8
D-85521 Ottobrunn
Germany

Tel +49 (89) 60822-0
Fax +49 (89) 60822-150
elosales@elotouch.com

Belgium
Elo TouchSystems
Diestsesteenweg 692
B-3010 Kessel-Lo
Belgium

Tel +32 (16) 35-2100
Fax +32 (16) 35-2101
elosales@elotouch.com

Japan
Touch Panel Systems K.K.
Sun Homada Bldg. 2F
1-19-20 Shin-Yokohama
Kohoku-ku, Yokohama
Kanagawa 222-0033
Japan

Tel +81 (45) 478-2161
Fax +81 (45) 478-2180
www.tps.co.jp

tyco Electronics

elo TOUCHSYSTEMS