Z-100 LifeLine
Since 1989

March 2022

#137

Microsoft 2.1A Serial Mouse with 9-pin female connector

Connecting a Serial Mouse to a Z-100 Computer

by Steven Vagts
Editor and Publisher
"Z-100 LifeLine"

You may be aware that we have a disk or two in the *"Z-100 LifeLine"* Software Library that uses a mouse input for games. A mouse can also be used for applications such as word processors and graphics programs, such as AutoCad. Further, Serial Mice are still available for reasonable prices from Ebay.

Well, recently, a reader reported that he had purchased a copy of AutoCad and needed a serial mouse. So he asked if I had the necessary mouse drivers that he would also need. I responded that I have mouse drivers in the Software Library, including Paul Herman's versions for Microsoft Mice and Logitech Mice and also ZMouse, another popular driver of the time. However, I had never attempted to get a mouse to work on the Z-100.

That got me wondering, what would it take to get a serial mouse to work on my Z-100?

The trouble with serial ports, & RS232 specifically, is it immediately causes confusion about where to connect things. For example, do you connect Transmit Data from one device to Transmit Data on the other or do you cross them over (a 'null' modem cable does this).

So, let's do a review of what a Serial Port is, what a Serial Mouse is, and how to get one to work on the Z-100.

History of RS232 Serial Port

In the early 1960s, a standards committee, today known as the Electronic Industries Association (EIA), developed a common interface standard for data communications equipment. At that time, data communications was thought to mean digital data exchange between a centrally located mainframe computer and a remote computer terminal, or possibly between two terminals without a computer involved. These devices were linked by telephone voice lines, and consequently required a modem at each end for signal translation.

While simple in concept, the many opportunities for data error that occur when transmitting data through an analog channel required a relatively complex design. It was thought that a standard was needed to ensure reliable communication, and to enable the interconnection of equipment produced by different manufacturers.

From these ideas, the RS232 standard was born. It formally defined the connecting signals between a DTE (Data Terminal Equipment) device, such as a computer terminal, and a DCE (Data Communication Equipment) device, such as a modem. The RS-232 standard was widely accepted and became commonly used in computer serial ports. The standard defined the electrical characteristics and timing of signals, the meaning of the different signals, and the physical size and pinout of the connectors.

As originally implemented, the equipment at the far end of the connection was named the DTE device (Data Terminal Equipment, usually a computer or terminal), had a male DB25 connector, and utilized 22 of the 25 available pins for signals or ground. The equipment at the near end of the connection (the telephone line interface) was named the DCE device (Data Circuit-terminating Equipment, usually a modem), had a female DB25 connector, and utilized the same 22 available pins for signals and ground.

The cable linking the DTE and DCE devices was a parallel straight-through cable with no crossovers or self-connects in the connector hoods. If all devices exactly followed this standard, all cables would be identical, and there would be no chance that an incorrectly wired cable could be used.

Over the 60+ years since this standard was developed, the EIA published 3 modifications, the most recent being the EIA232F standard introduced in 1997. Besides changing the name from RS232 to EIA232, some signal lines were renamed and various new ones were defined, including a shield conductor.

The RS-232 serial port became a standard feature of a personal computer, used for connections to modems, printers, data storage, uninterruptible power supplies, mice, and other peripheral devices. The serial port connector also changed from 25 pins, used in our Z-100 computer to the DB9 serial port connector, more properly called the RS232C DE-9 connector, used in some PCs and many other devices. Here are the current signal definitions for the four connectors:

Looking into the **Male DB25 DTE** Connector:	**Female DB25 DCE** Connector:
1 - Shield or Frame Ground	
2 - Transmitted Data (TD) (TXD*) (out)	**2 - Received Data (RD) (in)**
3 - Received Data (RD) (RXD*) (in)	**3 - Transmitted Data (TD) (out)**
4 - Request to Send (RTS) (out)	**4 - Clear to Send (CTS) (in)**
5 - Clear to Send (CTS) (in)	**5 - Request to Send (RTS) (out)**
6 - DCE Ready (in) **Data Set Ready (DSR)***	**6 - DTE Ready (out)** **Data Terminal Ready (DTR)***
7 - Signal Ground	
8 - Received Line Signal Detect (in) Data Carrier Detect (DCD)*	8 - (out)
9 - (reserved for testing) +Voltage*	
10 - (reserved for testing) - Voltage*	
11 - (unassigned)	
12 - Sec Received Line Signal Detect (in) Sec Data Carrier Detect*	12 - (out)
13 - Sec Clear to Send (in)	13 - Sec Request to Send (out)
14 - Sec Transmitted Data (out)	14 - Sec Received Data (in)
15 - Transmitter Signal Timing (in) Transmitter Clock (DCE)*	15 - (DCE Source) (out)
16 - Sec Received Data (in)	16 - Sec Transmitted Data (out)
17 - Receiver Signal Timing (in) Receiver Clock*	17 - (DCE Source) (out)
18 - Local Loopback (out)	18 - (in)
19 - Sec Request to Send (out)	19 - Sec Clear to Send (in)
20 - DTE Ready (out) **Data Terminal Ready (DTR)***	**20 - DCE Ready (in)** **Data Set Ready (DSR)***
21 - Remote Loopback (out) Signal Quality Detector*	21 - (in)
22 - Ring Indicator (RI) (in)	22 - (out)
23 - Data Signal Rate Selector (out)	23 - (in)
24 - Transmitter Signal Timing (out) Transmitter Clock (DTE)*	24 - (DTE Source) (in)
25 - Test Mode (in)	25 - (out)

Looking into the **Male DB9 DTE** Connector:

- 1 - Received Line Signal Detect
  Data Carrier Detect (DCD)*
- **2 - Received Data (RXD*) (in)**
- **3 - Transmitted Data (TXD*) (out)**
- **4 - DTE Ready (out)**
  Data Terminal Ready (DTR)*
- **5 - Signal Ground**
- **6 - DCE Ready (in)**
  Data Set Ready (DSR)*
- **7 - Request to Send (RTS) (out)**
- **8 - Clear to Send (CTS) (in)**
- 9 - Ring Indicator (RI)
- **Shield - Protective Ground**

Female DB9 DCE Connector:

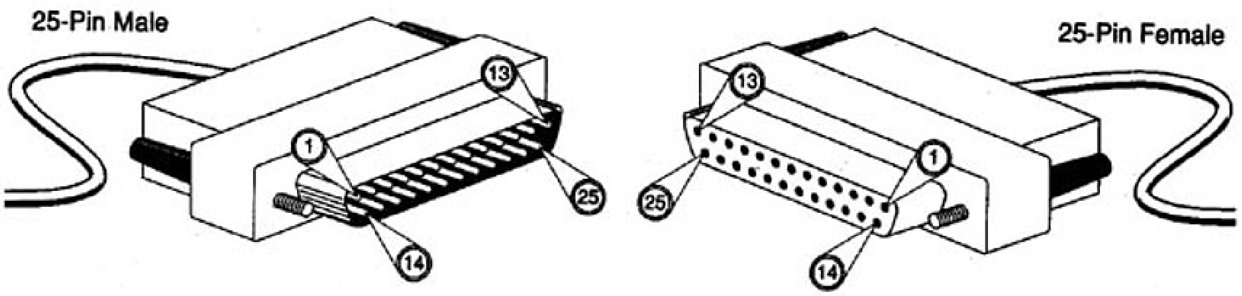
- **2 - Transmitted Data (TXD) (out)**
- **3 - Received Data (RXD) (in)**
- **4 - DCE Ready (in)**
  Data Set Ready (DSR)*
- **6 - DTE Ready (out)**
  Data Terminal Ready (DTR)*
- **7 - Clear to Send (CTS) (in)**
- **8 - Request to Send (RTS) (out)**

Notes:

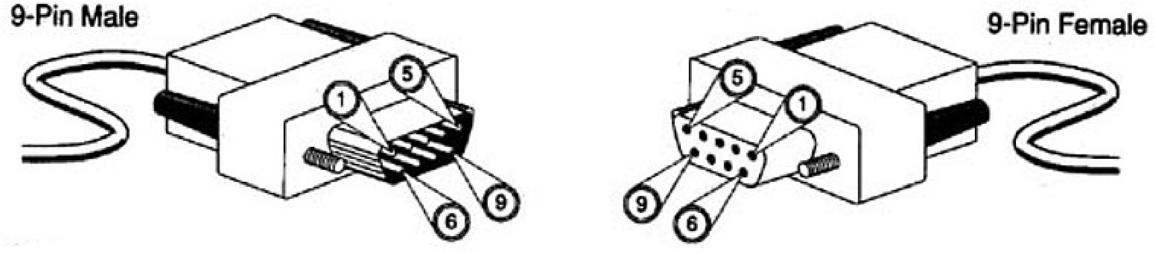
- Different manufacturers changed the names of some signals, shown with an asterisk (*).
- Line Definition Changes for the DB-25 connectors of the **Z-100**:
  Line 21 is for Signal Quality Detector
  Lines 18 and 25 are listed as unassigned (unused?)
- For both connector types, the most commonly used signals are shown in bold.
- Although the signals are still used for the DCE connectors, for change clarity, unchanged signal names are left blank. You can readily see how the signal direction changes from DTE to DCE.

~~~~~~

Here is a look at the two types of connectors that we are discussing:



Data Connector DB-25



Data Connector DB-9
~~~~~~

The RS-232, when compared to later interfaces such as RS-422, RS-485 and Ethernet, had lower transmission speed, short maximum cable length, large voltage swing, large standard connectors, no multipoint capability and limited multidrop capability.

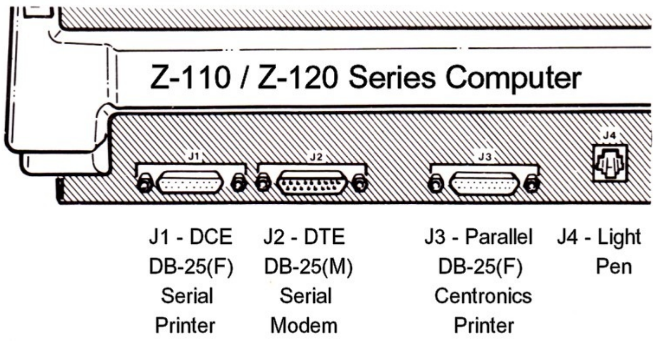
In post-2010 personal computers, USB versions have displaced RS-232 from its peripheral interface roles. Many computers no longer come equipped with RS-232 ports (although some motherboards come equipped with a COM port header that allows the user to install a bracket with a DB9 port) and must use either an external USB-to-RS-232 converter or an internal expansion card with one or more serial ports to connect to RS-232 peripherals.

Nevertheless, thanks to their simplicity and past ubiquity, RS-232 interfaces are still used — particularly in industrial machines, network equipment, and scientific instruments where a short-range, point-to-point, low-speed wired data connection is adequate.

The Serial Ports on the Z-100

The Z-100 has two serial ports, a parallel port, and a light pen connector on the rear of the motherboard. These are:

- **J1** - A female, 25-pin, DB25 serial connector, that provides the necessary EIA-standard RS-232 DCE signals for connection to a serial printer.
- **J2** - A male, 25-pin, DB25 serial connector, that provides the necessary EIA-standard RS-232 DTE signals for use with a telephone modem.
- **J3** - The parallel printer connector, that provides the necessary Centronics-type parallel signals for connection to a parallel printer. Unfortunately, it is **not a fully bi-directional port**.
- **J4** - The light pen connector, that provides the necessary signals for connection to a light pen for on-screen graphics work.



Normally, you must configure your Z-100 to use any input and output devices through the use of the CONFIGUR utility. Such devices include your printer, modem, and other serial or parallel input/output (I/O) devices.

Fortunately, when you are using a mouse or similar device, these setup requirements are taken care of by installing an appropriate software driver, to be addressed shortly.

Microsoft Serial Mouse

The Microsoft serial mouse is the most popular two button serial mouse type. The Microsoft mouse is supported in all major operating systems.

Mouse resolution and tracking rate

Maximum tracking rate for the Microsoft mouse is about 5000 counts per second. The most common range for typical mice is 100 to 400 CPI (count per inch) but can be up to 1000 CPI (cheap ones typically are 100 CPI or 200 CPI models). This means that you can move a 100 CPI mouse up to speed of 50.8 inches per second and 400 CPI mouse maximally at 12.7 inches per second.

The actual accuracy of movement the software sees is determined by the settings of the mouse driver (many mouse drivers have an option to adjust mouse sensitivity).

Microsoft Serial Mouse Pinout

25 pin	9 pin	Wire Name / Comments	
1	Shell	Protective Ground (optional)	
2	3	TD	Serial data from host to mouse (only for power)
3	2	RD	Serial data from mouse to host
4	7	RTS	Positive voltage to mouse
5	8	CTS	
6	6	DSR	
7	5	Signal Ground	
20	4	DTR	Positive voltage to mouse & reset/detection

RTS = Request to Send
CTS = Clear to Send
DSR = Data Set Ready
DTR = Data Terminal Ready

Note: Pins 1 and 9 are not used.

To function correctly:

- Both the RTS and DTR lines must be positive.
- The lines DTR-DSR and RTS-CTS must NOT be shorted.
- Implement the RTS toggle function by setting the RTS line negative & positive again.
- The negative pulse width is at least 100ms.
- After a cold boot, the RTS line is usually set to a negative level. In this case, setting the RTS line to a positive level is also considered an RTS toggle.

Serial data parameters:

1200bps, 7 databits, 1 stop-bit

How the serial mouse works

The typical PC mouse controlling system has the following parts: sensors -> mouse controller -> communication link -> data interface -> driver -> software. Sensors are the movement detectors which sense the mouse movement and button switches which sense the button states.

The mouse controller reads the state of those sensors and knows the current mouse position. When this information changes, the mouse controller sends a packet of data to the computer serial data interface controller. The mouse driver in the computer receives that data packet, decodes the information, and takes the appropriate actions.

RS232 data is usually sent as a packet with 7 or 8 bit words, start, stop, parity bits (may be varied). Sample transmission would include: Start bit (active low, usually between +3v and +15v) followed by data bits, parity bit (depends on protocol used) and finished by stop bit (used to bring logic high, usually between -3v and -15v).

RS232 serial data parameters and packet format

1200bps, 7 databits, 1 stop-bit

The Data Packet is 3 bytes and is sent to the computer every time the mouse state changes (mouse moves or keys are pressed/released).

	D7	D6	D5	D4	D3	D2	D1	D0
1.	X	1	LB	RB	Y7	Y6	X7	X6
2.	X	0	X5	X4	X3	X2	X1	X0
3.	X	0	Y5	Y4	Y3	Y2	Y1	Y0

Note: The bit marked with X is 0 if the mouse is received with 7 databits and 2 stop bits format. It is also possible to use 8 databits and 1 stop bit format for receiving. In this case X gets value 1. The safest thing to get everything working is to use 7 databits and 1 stopbit when receiving mouse information (and if you are making a mouse, then send out 7 databits and 2 stop bits).

The byte marked with 1 is send first, then the others. The bit D6 in the first byte is used for syncronizing the software to mouse packets, if it goes out of sync.

LB is the state of the left button
(1 means pressed down)
RB is the state of the right button
(1 means pressed down)
X7-X0 movement in X direction since
last packet (signed byte)
Y7-Y0 movement in Y direction since
last packet (signed byte)

Mouse identification

When the DTR line is toggled, the Microsoft mouse should send one data byte containing the letter 'M' (ASCII 77).

Logitech Serial Mouse

Logitech uses this same protocol in their mice (e.g., Logitech Pilot mouse and others). The original protocol supported only two buttons, but Logitech added a third button to some of their mouse models. To make this possible, Logitech made one extension to their protocol.

The information of the third button state is sent using one extra byte which is sent after the normal packet when needed. Value 32 (DEC) is sent every time the center button is pressed down. It is also sent with the data packet when the center button is held down and the mouse data packet is sent for other reasons.

When the center button is released, the mouse sends the normal data packet followed by data byte which has value 0 (DEC). As you can see, the extra data byte is sent only when you have used the center button.

Serial data parameters:

1200bps, 8 databits, 1 stop-bit

The data is sent in 5 byte packets in the following format:

	D7	D6	D5	D4	D3	D2	D1	D0
1.	1	0	0	0	0	LB	CB	RB
2.	X7	X6	X5	X4	X3	X2	X1	X0
3.	Y7	Y6	Y5	Y4	Y3	Y4	Y1	Y0
4.	X7'	X6'	X5'	X4'	X3'	X2'	X1'	X0'
5.	Y7'	Y6'	Y5'	Y4'	Y3'	Y4'	Y1'	Y0'

LB is left button state
(0=pressed, 1=released)
CB is center button state
(0=pressed, 1=released)
RB is right button state
(0=pressed, 1=released)
X7-X0 movement in X direction since
last packet in signed byte format
(-128..+127), positive direction right
Y7-Y0 movement in Y direction since
last packet in signed byte format
(-128..+127), positive direction up
X7'-X0' movement in X direction since
sending of X7-X0 packet in signed
byte format (-128..+127), positive
direction right
Y7'-Y0' movement in Y direction since
sending of Y7-Y0 in signed byte format
(-128..+127), positive direction up
The last two bytes in the packet (bytes 4 and 5) contain information about movement data changes which have occurred after data bytes 2 and 3 have been sent.

The DB25 to DB9 Mouse Adapter

So, how can we adapt the standard 9-pin, DB9 (female) Mouse Connector to work on one of our 25-pin, DB25, Z-100 serial ports? Well, first you need to determine which serial port you wish to use. Then you must construct or buy the appropriate 9-pin DB9 to 25-pin DB25 adapter. To construct your own, the pin connections should be as follows:

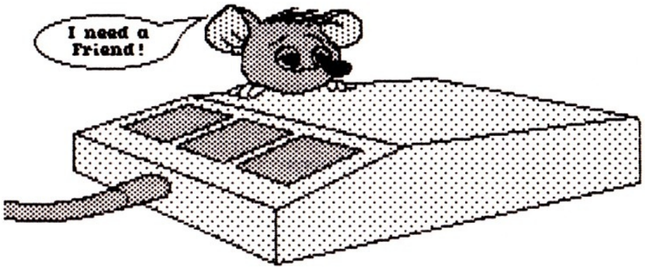
Adapter for Connection to DB25, Female, DCE, Z-100 connector J1:

Male DB25		**Male DB9**
2 - Transmitted Data (TD) (out)	---------->	2 - Received Data (RD) (in)
3 - Received Data (TR) (in)	<----------	3 - Transmitted Data (TD) (out)
4 - Request to Send (RTS) (out)	---------->	8 - Clear to Send (CTS) (in)
5 - Clear to Send (CTS) (in)	<----------	7 - Request to Send (RTS) (out)
6 - DCE Ready (in) Data Set Ready (DSR)*	<----------	4 - DTE Ready (out) Data Terminal Ready (DTR)*
7 - Signal Ground	<---------->	5 - Signal Ground
20 - DTE Ready (out) Data Terminal Ready (DTR)*	---------->	6 - DCE Ready (in) Data Set Ready (DSR)*

Adapter for Connection to DB25, Male, DTE, Z-100 connector J2:

Female DB25		**Male DB9**
2 - Received Data (RD) (in)	<----------	3 - Transmitted Data (TD) (out)
3 - Transmitted Data (TD) (out)	---------->	2 - Received Data (RD) (in)
4 - Clear to Send (CTS) (in)	<----------	7 - Request to Send (RTS) (out)
5 - Request to Send (RTS) (out)	---------->	8 - Clear to Send (CTS) (in)
6 - DTE Ready (out) Data Terminal Ready (DTR)*	---------->	6 - DCE Ready (in) Data Set Ready (DSR)*
7 - Signal Ground	<---------->	5 - Signal Ground
20 - DCE Ready (in) Data Set Ready (DSR)*	<----------	4 - DTE Ready (out) Data Terminal Ready (DTR)*

Mouse Software Driver



MOUSE PACK

By Paul F. Herman

Paul F. Herman developed two general purpose mouse drivers for the Zenith Z-100, Z-150, IBM-PC and compatibles:

Microsoft Serial Mouse
Mouse Systems PC Mouse

Introduction

The Microsoft Serial Mouse Driver (MSMOUSE) was for use with a Microsoft Serial Mouse. The Mouse Systems PC Mouse (PCMOUSE) was for use with a Mouse Systems PC Optical Mouse, Logitech Mouse, and most other popular mice.

The mouse driver is loaded into memory when needed (or by a batch file) and remains resident in memory until you reboot your computer. Once loaded, the mouse driver may be re-configured at any time. The mouse driver takes less than 2000 bytes of memory.

Distribution Disk Contents

The Mouse Pack Distribution Disk contains these files:

MSMOUSE.COM	Mouse driver for Microsoft Mouse.
PCMOUSE.COM	Mouse driver for most other mice.
MOUSE.ASM	Source code (for both mouse drivers)
MMSMOUSE.BAT	Batch file for re-assembling MSMOUSE.COM
MPCMOUSE.BAT	Batch file for re-assembling PCMOUSE.COM
READ.ME	This file has additional information not included in this manual.

Mouse Pack Concepts

The MOUSE PACK Mouse driver is installed by running the appropriate .COM file for your mouse. The driver installation routine does the necessary configuration depending on command line arguments, programs the serial port for mouse operation, and installs an interrupt routine in memory. Subsequently, whenever the mouse is moved or a button is pushed, the interrupt routine processes the mouse input and returns control to the user program.

When the interrupt routine has received a complete block of data from the mouse, it places appropriate key or scan codes into the MS-DOS BIOS keyboard buffer. In other words, if the mouse is moved 5 units to the right, the interrupt routine places 5 right arrow key codes into the keyboard buffer. The currently running user program processes the new key codes the same way it would if the keys were typed on the keyboard.

System Requirements

Hardware:
The MOUSE PACK Mouse drivers support **any combination** of the following mice to the computer listed:

COMPUTER	**MOUSE**
Zenith Z-100	Microsoft Serial Mouse
Zenith Z-150	Mouse Systems PC Mouse
IBM-PC or clone	Logitech Mouse

Operating System:
MS-DOS or PC-DOS versions 2.0 or higher is required for proper operation.

Application Programs:
The MOUSE PACK mouse drivers work by placing key codes into the MS-DOS BIOS keyboard buffer (also known as the type ahead buffer). Essentially, any program which makes use of the BIOS buffer will work with the mouse drivers.

Some programs that have been used with MOUSE PACK are DOODLER, WordStar, Condor, PeachCalc, PeachText, MultiPlan, and other less well known programs. One noteable program which will **NOT** work with MOUSE PACK is Lotus 1-2-3.

Other programs, such as Microsoft WORD have their own mouse support built in.

Getting Started
Where to Put the Mouse Driver:
In order to begin using the MOUSE PACK Mouse Driver, you should copy one of the mouse driver programs onto your working disk. If you will use the Microsoft Mouse, you should use MSMOUSE.COM. If you use the Logitech Mouse, Mouse Systems PC Mouse, or most other mice, use PCMOUSE.COM.

The mouse driver program may be placed in the root directory or in any other directory accessible by the MS-DOS PATH. Putting the Mouse driver in the BIN (or DOS) directory would be a good idea. PATH access is important so that you may load and configure the mouse driver from anywhere. If you only have one application you intend to use with a mouse, the mouse driver program may be copied into the same directory as the application program.

Loading the Mouse Driver:
To load the Mouse driver into memory, simply type the name of the mouse driver program followed by a RETURN. Command line arguments may be used to configure the mouse for an alternate port or other special functions. See the discussion under **Configuring the Mouse Driver.** Once the mouse driver is loaded, it remains in memory ready to use until you reboot the computer.

Using the Mouse Driver:
Once the mouse driver is loaded into memory, the mouse will be active and ready for use. Moving the mouse in any direction will have the same effect as pressing the cursor keys on the keyboard.

Functions provided by pressing the mouse buttons will depend on what codes the buttons have been configured to generate, and the action of your application program. The default setting for each button is to duplicate a Function key; F1 is the left button, F2 is the right button, and if you have a three button mouse, F3 is the center button.

For example, with the default configuration, when the DOS prompt is displayed on the screen, the left button will copy one character from the last command entered (just as pressing the F1 key does). Moving the mouse to the left will erase characters on the command line.

Note: Since the MOUSE PACK driver works by putting characters into the MS-DOS type ahead buffer, it is possible to cause a 'buffer overrun' if the mouse is moved too fast. When the buffer becomes full, any additional characters are disregarded until the application program reads the keyboard and makes more room in the buffer. The result of this buffer overrun condition is that the mouse will not track accurately.

Configuring the Mouse Driver - Summary
The Mouse drivers may be user configured for different serial ports and specifications. Configuration is accomplished by entering command arguments when invoking the mouse driver. All command line arguments should be separated by SPACES and may appear in any order.

~~~~~~

Valid Command Line Arguments are:

Spn Serial Port Selection ... n = 1 or 2

For the IBM-PC	**For the Z-100**
SP1 = COMM1	SP1 = Serial Port A (J1)
SP2 = COMM2	SP2 = Serial Port B (J2)

XSn Horizontal Step Sensitivity . . n = 1 to 100
YSn Vertical Step Sensitivity . . . . n = 1 to 100
The step sensitivity indicates how many 'units' the mouse must move before generating a cursor code.

XDn Horizontal Damping . . . . n = 1 to 100
YDn Vertical Damping . . . . n = 1 to 100
~~~~~~

Damping makes it easier to draw a straight line with the mouse. Use small numbers to draw straight lines and large numbers to do freehand sketches for graphics applications.

LBn Left Button Code . . . . n = 0 to 255
MBn Middle Button Code (PCMOUSE.COM only)
RBn Right Button Code
See Appendix for a list of Button Codes

?? Two question marks anywhere on the command line will cause the driver to list all configuration info.

Notes:

- The 'n' numeric portion of the command should be entered using decimal numbers, and should IMMEDIATELY follow the two letter command argument. DO NOT put a space between the two letters and the numeric argument.
- Any of the command line arguments may be omitted.
  - If the mouse driver is already installed, any unspecified arguments will be left the same.
  - If the mouse driver is being installed for the first time, default values will be used for all unspecified arguments. The list below shows the default values which are used:

		Z-100	**IBM-PC or Clone**
Serial Port		SP2 (Serial B) (J2)	SP1 (COMM1)
Step Sensitivity		XS2	XS2
		YS2	YS2
Damping Factor		XD10	XD10
		YD10	YD10
Button Codes			
	Both drivers	LB151 (F1)	LB59 (F1)
	PCMOUSE.COM	MB152 (F2)	MB60 (F2)
	MSMOUSE.COM	RB152 (F2)	RB60 (F2)
	PCMOUSE.COM	RB153 (F3)	RB61 (F3)

Port Selection
The default mouse port will depend on which computer you are using:

- On the Z-100, the default port is Serial Port 'B' (J2). Serial Port 'B' is used by plugging the mouse into J2 on the Z-100.
- On a PC-compatible computer, the default port is COMM1. The mouse should be plugged into the COMM1 connector.

To accommodate different hardware, the mouse driver may also be configured for Serial Port 'A' on the Z-100, or COMM2 on a PC compatible. To specify the serial port, use the SP command on the command line when invoking the mouse driver.

For examples, the command:

MSMOUSE SP1 will configure the Z-100 for Serial Port A.
MSMOUSE SP2 will configure the IBM-PC for COMM2.

Step Sensitivity
The mouse step sensitivity tells the mouse how many 'mouse units' to move before placing a keycode in the keyboard buffer. With the PC Mouse, these units are viewable as lines on the mouse pad. With other mice, the 'mouse units' are arbitrary, but constant, amounts.

A small step sensitivity causes large screen cursor movement with little mouse motion. Large step sensitivity results in just the opposite; the mouse must be moved quite a bit to get much motion on the screen.

The default step sensitivity for both horizontal and vertical movement is 2. To configure the mouse driver for step sensitivity, use the **XS** and **YS** commands on the command line when invoking the mouse driver.

Damping Factor
The damping factor makes it possible to keep the cursor going in a straight line. If the damping factor is set to 100, the mouse will trace precisely as it is moved, making it very difficult to draw a straight line. When the damping factor is set to 1, it is almost impossible to move at an angle.

For example, if the X damping is 5, you may move the mouse one vertical unit for every 5 horizontal units, and still draw a straight horizontal line.

The default damping factor for both horizontal and vertical movement is 10. To configure the mouse driver for other damping, use the **XD** and **YD** commands on the command line when invoking the mouse driver.

Example: MSMOUSE XD1 YD5

Button Codes
The mouse driver may be configured to use many different button codes. You may find it useful to configure the mouse for button codes which are useful for a particular application program. The default button codes depend on which computer and mouse you are using:

Button Codes	**Z-100**	**IBM-PC/Clone**
Both drivers	LB151 (F1)	LB59 (F1)
PCMOUSE.COM	MB152 (F2)	MB60 (F2)
MSMOUSE.COM	RB152 (F2)	RB60 (F2)
PCMOUSE.COM	RB153 (F3)	RB61 (F3)

To configure the mouse for other button codes, use the **LBn**, **MBn**, and **RBn** commands on the command line when invoking the mouse driver. See the Appendix following this discussion for a complete list of available button codes for your computer.

Batch File Configuration
If you intend to use the mouse regularly, you should load the mouse driver as a normal part of booting up. This can be accomplished by including the mouse driver name (MSMOUSE or PCMOUSE) in your AUTOEXEC.BAT file. The AUTOEXEC.BAT file must reside in your root directory. Be sure the mouse driver program is in the root directory or in a directory accessible by your PATH.

You may also find it convenient to change the mouse configuration before using different application programs by using a different batch file for each application. An example batch file listing is given here which configures the mouse and then runs the DOODLER Graphics Package.

```
PCMOUSE XS2 YS2 XD100 YD100
DOODLER
PCMOUSE XS2 YS2 XD10 YD10
```

This batch file loads (or reconfigures) the PCMOUSE driver with parameters appropriate for the DOODLER Graphics Package. Then the DOODLER program is loaded and run. After exiting DOODLER, the batch file continues to reconfigure the mouse driver to the default parameters.

Suggestions for Configuration
The default configuration for the mouse drivers are for general use and do not suit all programs ideally. We give here some suggestions for optimal configurations for different types of programs.

DOODLER Graphics Package:
The step sensitivity should reflect the aspect ratio of the screen. With the Z-100, the Y/X aspect ratio is about two to one. Therefore, YSn should be twice as large as XSn. For example, XS2 YS4. For very intricate work, try using XS25 YS50. For drawing freehand figures with curved lines, use XD100 YD100. For drawing straight lines and geometric designs, use XD1 YD1. A good compromise is the default of XD10 YD10. For DOODLER, the F1, F2, F3 button codes are ideal. For other drawing programs, set the button codes to commonly used command keys.

My PAINT100 Program:
The default step rate of 2 was far too sensitive, moving the pointer all over the place, so I increased the sensitivity to 10. And, I wanted the mouse buttons to reflect the Function keys given at the bottom of the work screen; that is, F2 (152) sets and clears the Reverse Video function, F3 (153) sets and clears the Graphics function, and F4 (154) clears the Insert Character, Reverse Video, Graphics and Color functions. Therefore, I wanted the mouse buttons to do the same. If you wish the same configuration, depending upon the mouse type, use either of the commands:

```
MSMOUSE XS10 YS10 XD10 YD10 LB152 RB153
PCMOUSE XS10 YS10 XD10 YD10 LB152 MB153 RB154
```

Text Editors:
The step sensitivity should be set for personal preferences. The default values should be satisfactory for most use. The damping should be set at XD1 YD1 to make it easy to stay on the same line. Button codes will depend on which editor you are using. Most editors allow macro keys to be programmed. This makes it easy to coordinate with the mouse by configuring button codes for the macro keys.

Spread Sheet Program:
Use the same suggestions given for Text Editors.

Can I use a PS/2 Mouse with a PS/2 to 9-pin DB9 connector?

The quick answer is **NO**! The only obvious difference is the connector; PS-2 mice use a round 6-pin plug, and serial mice use a 9-pin DB9 connector. However, what is not obvious is that PS-2 mice connect electronically to the keyboard controller chip inside your computer, whereas serial mice are designed to talk to the RS-232 controller in the computer. Both controllers are actually serial, but the PS-2 is a special case designed to run at a fixed speed (determined by the computer) and is for input only.

However, I have found a couple of adapters on line in the form of circuit boards (about $20) that **may** work at converting the PS/2-type signals to those necessary for serial ports. Whether these actually work, I leave to the more adventurous of you.

One more word of caution. From what I found on the internet, the PS-2 to DB9 serial adapter plugs all have different internal circuitry, and you will not know what it is until you put a meter on it. I have one, and its wiring was not the same as any of the suggested circuits that I found on the internet. So I cannot recommend just trying an adapter.

APPENDIX A - Z-100 Button Codes

CODE	KEY	CODE	KEY	CODE	KEY	CODE	KEY	CODE	KEY
0	CTL @								
1	CTL A	52	4	103	g	154	F4	205	SH ENTER
2	CTL B	53	5	104	h	155	F5	206	
3	CTL C	54	6	105	i	156	F6	207	
4	CTL D	55	7	106	j	157	F7	208	
5	CTL E	56	8	107	k	158	F8	209	
6	CTL F	57	9	108	l	159	F9	210	
7	CTL G	58	:	109	m	160	F10	211	
8	BACK SPACE	59	;	110	n	161	F11	212	
9	TAB	60	<	111	o	162	F12	213	SH HELP
10	LINE FEED	61	=	112	p	163	ICHR	214	SH F0
11	CTL K	62	>	113	q	164	INSLINE	215	SH F1
12	CTL L	63	?	114	r	165	UP	216	SH F2
13	RETURN	64	@	115	s	166	DOWN	217	SH F3
14	CTL N	65	A	116	t	167	RIGHT	218	SH F4
15	CTL O	66	B	117	u	168	LEFT	219	SH F5
16	CTL P	67	C	118	v	169	HOME	220	SH F6
17	CTL Q	68	D	119	w	170	BREAK	221	SH F7
18	CTL R	69	E	120	x	171		222	SH F8
19	CTL S	70	F	121	y	172		223	SH F9
20	CTL T	71	G	122	z	173	- (KPD)	224	SH F10
21	CTL U	72	H	123	{	174	. (KPD)	225	SH F11
22	CTL V	73	I	124	|	175		226	SH F12
23	CTL W	74	J	125	}	176	0 (KPD)	227	DCHR
24	CTL X	75	K	126	~	177	1 (KPD)	228	DEL LINE
25	CTL Y	76	L	127	DEL	178	2 (KPD)	229	SH UP
26	CTL Z	77	M	128		179	3 (KPD)	230	SH DOWN
27	ESC	78	N	129		180	4 (KPD)	231	SH RIGHT
28	CTL /	79	O	130		181	5 (KPD)	232	SH LEFT
29	CTL]	80	P	131		182	6 (KPD)	233	SH HOME
30	CTL ^	81	Q	132		183	7 (KPD)	234	SH BREAK
31	CTL _	82	R	133		184	8 (KPD)	235	
32	SPACE	83	S	134		185	9 (KPD)	236	
33	!	84	T	135		186		237	SH -(KPD)
34	"	85	U	136		187		238	SH .(KPD)
35	#	86	V	137		188		239	
36	$	87	W	138		189		240	SH 0(KPD)
37	%	88	X	139		190		241	SH 1(KPD)
38	&	89	Y	140		191		242	SH 2(KPD)
39	'	90	Z	141	ENTER	192		243	SH 3(KPD)
40	(	91	[	142		193		244	SH 4(KPD)
41	)	92	\	143		194		245	SH 5(KPD)
42	*	93	]	144		195		246	SH 6(KPD)
43	+	94	^	145		196		247	SH 7(KPD)
44	,	95	_	146		197		248	SH 8(KPD)
45	-	96	`	147		198		249	SH 9(KPD)
46	.	97	a	148		199		250	
47	/	98	b	149	HELP	200		251	
48	0	99	c	150	F0	201		252	
49	1	100	d	151	F1	202		253	
50	2	101	e	152	F2	203		254	
51	3	102	f	153	F3	204		255	

APPENDIX B - IBM-PC (or Clone) Button Codes

CODE	KEY	CODE	KEY	CODE	KEY	CODE	KEY	CODE	KEY
0	CTL BREAK	30	ALT A	60	F2	90	SH F7	120	ALT 1
1		31	ALT S	61	F3	91	SH F8	121	ALT 2
2		32	ALT D	62	F4	92	SH F9	122	ALT 3
3		33	ALT F	63	F5	93	SH F10	123	ALT 4
4		34	ALT G	64	F6	94	CTL F1	124	ALT 5
5		35	ALT H	65	F7	95	CTL F2	125	ALT 6
6		36	ALT J	66	F8	96	CTL F3	126	ALT 7
7		37	ALT K	67	F9	97	CTL F4	127	ALT 8
8		38	ALT L	68	F10	98	CTL F5	128	ALT 9
9		39		69		99	CTL F6	129	ALT 0
10		40		70		100	CTL F7	130	ALT -
11		41		71	HOME	101	CTL F8	131	ALT =
12		42		72	UP	102	CTL F9	132	CTL PGUP
13		43		73	PGUP	103	CTL F10		
14		44	ALT Z	74		104	ALT F1		
15	SHIFT TAB	45	ALT X	75	LEFT	105	ALT F2		
16	ALT Q	46	ALT C	76		106	ALT F3		
17	ALT W	47	ALT V	77	RIGHT	107	ALT F4		
18	ALT E	48		78		108	ALT F5		
19	ALT R	49		79	END	109	ALT F6		
20	ALT T	50		80	DOWN	110	ALT F7		
21	ALT Y	51		81	PGDN	111	ALT F8		
22	ALT U	52		82	INS	112	ALT F9		
23	ALT I	53		83	DEL	113	ALT F10		
24	ALT O	54		84	SH F1	114	CTL *(KPD)		
25	ALT P	55		85	SH F2	115	CTL LEFT		
26		56		86	SH F3	116	CTL RIGHT		
27		57		87	SH F4	117	CTL END		
28		58		88	SH F5	118	CTL PGDN		
29		59	F1	89	SH F6	119	CTL HOME		

APPENDIX C - Copyrights & Trademarks

The MOUSE PACK Mouse Driver Package, documentation, software and source code were all Copyright (C)1985 by Paul F. Herman. These were subsequently transferred to *"Z-100 LifeLine"*, Steven W. Vagts, Editor and Publisher, in 1994, when Steve assumed publishing duties of the *"Z-100 LifeLine"* with issue #31.

Please direct any questions or comments to Steven W. Vagts, Editor and Publisher of the *"Z-100 LifeLine"* at:

Steven W. Vagts
211 Sean Way
Hendersonville, NC 28792
Phone: (828) 685-8924
Email: **z100lifeline@swvagts.com**

The above Mouse Software Driver discussion is mainly from **Paul F. Herman's "MOUSE PACK" Mouse Driver Package**, designed for the Zenith Z-100 series computer and the IBM-PC computer and numerous clones. While the LifeLine recognizes the Trademarks of all of our commonly mentioned products, the discussion mentioned the trademark owners of several unique products that we should recognize:

Doodler Graphics Package - Paul F. Herman
PC Mouse - Mouse Systems Corp.
Multiplan - Microsoft
WordStar - Micropro International Corp.
PeachText & PeachCalc - Peachtree Software
Condor - Condor Computer Corp.
Lotus 1-2-3 - Lotus Development Corp.

I hope that you found this article interesting, informative and helpful. Enjoy.

Cheers,

Steve

Steven W. Vagts
Z-100 LifeLine