Chapter 9

Input Devices

This chapter discusses *input devices*—the devices used to communicate with the computer. The most common input device is of course the keyboard, and this chapter discusses keyboards in depth. Also discussed are mice because they are now a standard requirement to operate a modern PC with a GUI (graphical user interface) such as Windows or OS/2. Finally, this chapter also discusses the game or joystick interface, used to input signals from joysticks, paddles, or other game devices.

Keyboards

One of the most basic system components is your keyboard. The keyboard is the primary input device and is used for entering commands and data into the system. This section examines the keyboards available for IBM and compatible systems. This section examines the different types of keyboards, how the keyboard functions, the keyboard-to-system interface, and finally keyboard troubleshooting and repair.

Types of Keyboards

Since the introduction of the original IBM PC, IBM has created three different keyboard designs for PC systems, which have become standards in the industry shared by the compatible manufacturers as well. The three primary keyboard types are as follows:

- 83-key PC and XT keyboard
- 84-key AT keyboard
- 101-key enhanced keyboard

This section discusses each keyboard type and shows its layout and physical appearance. Because most systems use the 101-key enhanced keyboard design, this version is emphasized.

83-Key PC and XT Keyboard. When the PC was first introduced, it had something that few other personal computers had at the time: an external detachable keyboard. Most other small personal computers had the keyboard built in, like the Apple II. Although the external design was a good move on IBM's part, its keyboard design was not without criticism. One of the most criticized components of the original 83-key keyboard is the awkward layout (see fig. 9.1). The Shift keys are small and in the wrong place on the left side. The Enter key also is too small. These oversights were especially irritating at the time because IBM had produced the Selectric typewriter, perceived as a standard for good keyboard layout.

Fig. 9.1
PC and XT 83-key keyboard layout.

The PC/XT keyboard has a built-in processor that communicates to the motherboard via a special serial data link. The communication is one-way, which means that the motherboard cannot send commands or data back to the keyboard. For this reason, IBM 83-key keyboards have no LED indicator lights. Because the status of the Caps Lock, Num Lock, and Scroll Lock are maintained by the motherboard, there is no way to make sure that any LED indicator lights remain in sync with the actual status of the function. Many aftermarket (non-IBM) PC keyboards added the lights, and the keyboard attempted to keep track of the three functions independently of the motherboard. This worked in most situations, but it was entirely possible to see the LEDs become out of sync with the actual function status. Rebooting corrected this temporary problem, but it was annoying nonetheless. By eliminating the lights, IBM keyboards did not have this potential problem.

The original 83-key PC/XT keyboard is no longer used and is not electrically compatible with AT motherboards, although some aftermarket units may be compatible by moving an XT/AT switch usually found on the bottom of the keyboard.

84-Key AT Keyboard. When the AT was introduced in 1984, it included a new keyboard—the 84-key unit (see fig. 9.2). This keyboard corrected many problems of the original PC and XT keyboards. The position and arrangement of the numeric keypad was modified. The Enter key was made much larger, like that of a Selectric typewriter. The Shift key positions and sizes were corrected. IBM also finally added LED indicators for the status of the Caps Lock, Scroll Lock, and Num Lock toggles.

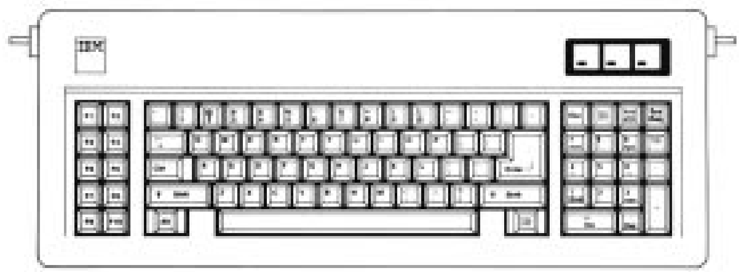

Fig. 9.2
AT 84-key keyboard layout.

These keyboards use a slightly modified interface protocol that is bidirectional. This means that the processor built into the keyboard can talk to another processor (called the keyboard controller chip) built into the motherboard. The keyboard controller on the motherboard also can send commands and data to the keyboard, which allows functions such as changing the keyboard typematic (or repeat) rate as well as the delay before repeating begins. The keyboard controller on the motherboard also scans code translations, which allows a much easier integration of foreign language keyboards into the system. Finally, the bidirectional interface can be used to control the LED indicators on the keyboard, thus ensuring that the status of the function and indicators are always in sync.

The 84-key unit that came with the original AT system is no longer used, although it does have the proper electrical design to be compatible with the newer systems. It lacks some of the keys found in the newer keyboards and does not have as nice a numeric keypad section, but many prefer the more Selectric style layout of the alpha-numeric keys. Also some prefer the ten function keys arranged on the left-hand side as opposed to the enhanced arrangement in which 12 function keys are lined up along the top.

Enhanced 101-Key (or 102-Key). In 1986, IBM introduced the "corporate" enhanced 101-key keyboard for the newer XT and AT models (see fig. 9.3). I use the word "corporate" because this unit first appeared in IBM's RT PC, which is an RISC (Reduced Instruction Set Computer) system designed for scientific and engineering applications; keyboards with this design are now supplied with virtually every type of system and terminal IBM sells. This universal keyboard has a further improved layout over that of the 84-key unit, with perhaps the exception of the Enter key, which reverted to a smaller size. The 101-key enhanced keyboard was designed to conform to international regulations and specifications for keyboards. In fact, other companies such as DEC and TI had already been using designs similar to the IBM 101-key unit even earlier than IBM. The IBM 101-key units originally came in versions with and without the status indicator LEDs, depending on whether the unit was sold with an XT or AT system. Now there are many other variations to choose from, including some with integrated pointing devices.

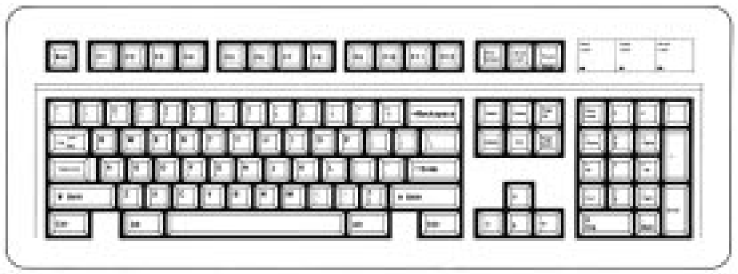

Fig. 9.3
101-key enhanced keyboard layout.

The enhanced keyboard is the current standard and is available in several different variations, but all are basically the same electrically and can be interchanged. IBM and its keyboard and printer subsidiary Lexmark have produced a number of versions including some newer ones with built-in pointing devices. Most of the enhanced keyboards attach to the system via the standard 5-pin DIN connector, but many newer ones come with cables for the 6-pin mini-DIN connector found on many newer systems including the IBM PS/2s and most Slimline compatibles. Although the connectors may be physically different, the keyboards are not, and you can either interchange the cables or use a cable adapter to plug one type into the other.

Because older PC/XT-style systems lack the bidirectional keyboard interface required to drive the LED indicators found on most enhanced keyboards, IBM even made models without these lights (Caps Lock, Num Lock, and Scroll Lock). These keyboards are the same as the others, but they are missing the small add-on circuit board used to drive the lights. If you use an older XT system with an enhanced keyboard that has the LEDs, they simply remain dark. Some compatible keyboard manufacturers have added circuitry internal to the keyboard that turns the LEDs on and off whenever you press the appropriate keys. In such a case, however, it is possible for the LEDs to get "out of sync" with the actual state of the toggles, which are actually maintained within the computer and not the keyboard.

The 101-key keyboard layout can be divided into the following four sections:

- Typing area
- Numeric keypad
- Cursor and screen controls
- Function keys

The 101-key arrangement is similar to the Selectric keyboard layout with the exception of the Enter key. The Tab, Caps Lock, Shift, and Backspace keys have a larger striking area and are located in the familiar Selectric locations. Ctrl and Alt keys are on each side of the space bar. The typing area and numeric keypad have home-row identifiers for touch typing.

The cursor and screen-control keys have been separated from the numeric keypad, which is reserved for numeric input. (As with other PC keyboards, you can use the numeric keypad for cursor and screen control when the keyboard is not in Num Lock mode.) A division-sign key and an additional Enter key have been added to the numeric keypad.

The cursor-control keys are arranged in the inverted T format. The Insert, Delete, Home, End, Page Up, and Page Down keys, located above the dedicated cursor-control keys, are separate from the numeric keypad. The function keys, spaced in groups of four, are located across the top of the keyboard. The keyboard has two additional function keys (F11 and F12). The Esc key is isolated in the upper left corner of the keyboard. Dedicated Print Screen/Sys Req, Scroll Lock, and Pause/Break keys are provided for commonly used functions.

Foreign language versions of the enhanced keyboard include 102-keys and a slightly different layout from the 101-key United States versions.

One of the enhanced keyboard's many useful features is removable keycaps. With clear keycaps and paper inserts, you can customize the keyboard. Keyboard templates are also available from IBM to provide specific operator instructions.

The enhanced keyboard probably will be on any desktop system IBM introduces for quite some time. It is currently the most popular design and does not show any signs of being replaced in the future. Because most compatible systems use this same type of keyboard, it is relatively easy to move from one system to another without relearning the layout.

Compatibility. The 83-key PC/XT is different from all the others and normally only plugs into a PC/XT system that does not use the motherboard-based 8042-type keyboard controller chip. This is definitely true for IBM's keyboards, but also is true for many compatible units as well. Some compatibles may be switchable to work with an AT-type motherboard via an XT/AT switch.

The 84-key unit from IBM works only on AT-type motherboards and does not work at all with PC/XT systems. Again, some aftermarket designs may have an XT/AT switch to allow for compatibility with PC/XT-type systems.

The enhanced keyboards from IBM are universal and auto-switching, meaning that they work in virtually any system from the XT to the PS/2 or any IBM-compatible by simply plugging them in. Some may require that a switch be moved on the keyboard to make it compatible with PC/XT systems that do not have the 8042-type keyboard controller on the motherboard. In some cases, you also may need to switch to a different cable with the proper system end connector, or use an adapter.

Although the enhanced keyboard is electrically compatible with any AT-type motherboard and even most PC/XT-type motherboards, many older systems will have software problems using these keyboards. IBM changed the ROM on the systems to support the new keyboard properly, and the compatible vendors followed suit. Very old (1986 or earlier) machines may require a ROM upgrade to properly use some of the features on the 101-key enhanced keyboards, such as the F11 and F12 keys. If the individual system ROM BIOS is not capable of operating the 101-key keyboard correctly, then the 101-key keyboard may not work at all (as with all three ROM versions of the IBM PC); the additional keys (F11 and F12 function keys) may not work; or you may have problems with keyboard operation in general. In some cases, these compatibility problems cause improper characters to appear when keys are typed causing the system to beep, and general keyboard operation is a problem. These problems can often be solved by a ROM upgrade to a newer version with proper support for the enhanced keyboard.

If you have an IBM system, you can tell whether your system has complete ROM BIOS support for the 101-key unit: when you plug in the keyboard and turn on the system unit, the Num Lock light automatically comes on and the numeric keypad portion of the keyboard is enabled. This method of detection isn't 100 percent accurate, but if the light goes on, your BIOS generally supports the keyboard. A notable exception is the IBM AT BIOS dated 06/10/85; it turns on the Num Lock light, but still does not properly support the enhanced keyboard. All IBM BIOS versions dated 11/15/85 or newer do have proper support for the enhanced keyboards.

In IBM systems that support the enhanced keyboard, if it is detected on power up, Num Lock is enabled, and the light goes on. If one of the older 84-key AT-type keyboards is detected, then the Num Lock function is not enabled because these keyboards do not have arrow keys separate from the numeric keypad. When the enhanced keyboards first appeared in 1986, many users (myself included) were irritated upon finding that the numeric keypad was automatically enabled every time the system boots. Most compatibles began integrating a function into the system setup that allowed specification of the Num Lock status on boot. Unfortunately, IBM still does not have this feature in many PS/2 systems.

Some thought that the automatic enabling of Num Lock was a function of the enhanced keyboard because none of the earlier keyboards seemed to operate this way. Remember that this function isn't really a keyboard function—instead it's a function of the motherboard ROM BIOS, which identifies an enhanced 101-key unit and turns on the Num Lock as a "favor." In systems that cannot disable the automatic numeric keypad enable feature, you can use the DOS 6.0 or higher version NUMLOCK= parameter in CONFIG.SYS to turn Num Lock on or off as desired. If you are running an earlier version of DOS than 6.0, then you can simply use one of the many public domain programs available for turning off the Num Lock function. Placing the program command to disable Num Lock in the AUTOEXEC.BAT file turns off the numeric keypad each time the system reboots.

In an informal test, I plugged the new keyboard into an earlier XT. The keyboard seemed to work well. None of the keys that didn't exist previously, such as F11 and F12, were operable, but the new arrow keys and the numeric keypad did work. The enhanced keyboard seems to work on XT or AT systems, but does not function on the original PC systems because of BIOS and electrical interface problems. Many compatible versions of the 101-key enhanced keyboards have a manual XT/AT switch on the bottom that may allow the keyboard to work in an original PC system.

Keyboard Technology

The technology that makes up a typical PC keyboard is very interesting. This section focuses on all aspects of keyboard technology and design including the keyswitches, the interface between the keyboard and the system, scan codes, and the keyboard connectors.

Keyswitch Design. Several types of switches are used in keyboards today. Most keyboards use one of several variations on a mechanical keyswitch. A mechanical keyswitch relies on a mechanical momentary contact-type switch to make electrical contact in a circuit. Some high-end keyboard designs use a totally different nonmechanical design that relies on capacitive switches. This section discusses these switches and the highlights of each design.

The most common type of keyswitch is the mechanical type, available in the following variations:

- Pure mechanical
- Foam element
- Rubber dome
- Membrane

The *pure mechanical* type is just that, a simple mechanical switch that features metal contacts in a momentary contact arrangement. Often a tactile feedback mechanism—consisting of a clip and spring arrangement to give a “clickety” feel to the keyboard and offer some resistance to pressing the key—is built-in. A company called Lite-On manufactures this type of keyboard using switches from Alps Electric (see Appendix B, the “Vendor List”). Mechanical switches are very durable, usually have self-cleaning contacts, and normally are rated for 20 million keystrokes, which is second only to the capacitive switch. They also offer excellent tactile feedback.

Foam element mechanical switches were a very popular design in some older keyboards. Most of the older compatible keyboards, including those made by Keytronics and many others, use this technology. These switches are characterized by a foam element with an electrical contact on the bottom that is mounted on the bottom of a plunger attached to the key itself (see fig. 9.4).

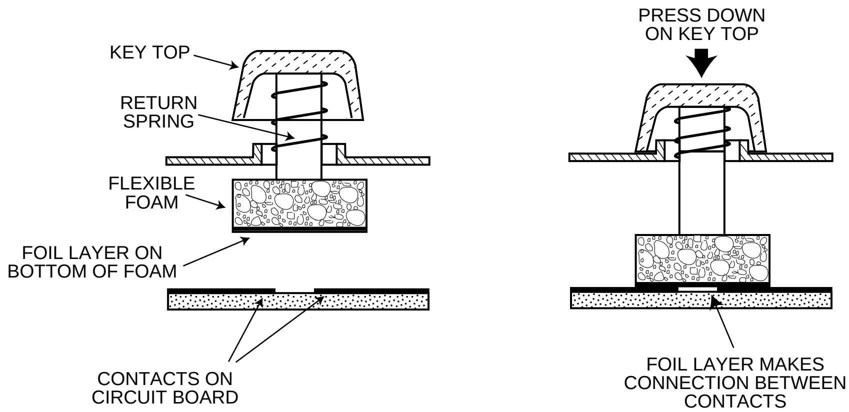

Fig. 9.4

Typical foam element mechanical keyswitch.

When the switch is pressed down, a foil conductor on the bottom of the foam element closes a circuit on the printed circuit board below. A return spring pushes the key back up when the pressure is released. The foam dampens the contact, helping to prevent bounce, but unfortunately gives these keyboards a “mushy” feel. The big problem with this type of keyswitch design is that there is often little in the way of tactile feedback, and systems with these keyboards often resort to tricks like clicking the PC’s speaker to signify that contact has been made. Compaq used keyboards of this type (made by Keytronics) in many of its older systems, but perhaps the most popular user today is Packard Bell. Preference in keyboard feel is somewhat subjective, and personally I do not favor the foam element switch design.

Another problem with this type of design is that it is more subject to corrosion on the foil conductor as well as the circuit board traces below. When this happens, the key strikes may become intermittent, which can be frustrating. Fortunately, these keyboards are among the easiest to clean. By disassembling this type of keyboard completely, you usually can remove the circuit board portion, without removing each foam pad separately, and expose the bottoms of all the pads. Then you can easily wipe the corrosion and dirt off the bottom of the foam pads as well as the circuit board, thus restoring the keyboard to like-new condition. Unfortunately, over time, the corrosion problem will occur again. Using some Stabilant 22a from D.W. Electrochemicals (see Appendix B) is highly recommended to improve the switch contact action as well as to prevent future corrosion. Because of problems like this, the foam element design is not used much anymore and has been superseded in popularity by the rubber dome design.

Rubber dome switches are mechanical switches that are very similar to the foam element type, but are also improved in many ways. Instead of a spring, these switches use a rubber dome that has a carbon button contact on the underside. As you depress a key, the key plunger presses down on the rubber dome, causing it to resist and then collapse all at once, much like the top of an oil can. As the rubber dome collapses, the user feels the tactile feedback, and the carbon button makes contact between the circuit board traces below. When the key is released, the rubber dome re-forms and pushes the key back upward.

The rubber eliminates the need for a spring and provides a reasonable amount of tactile feedback without any special clips or other parts. A carbon button is used because it is resistant to corrosion and also has a self-cleaning action on the metal contacts below. The rubber domes are formed into a sheet that completely protects the contacts below from dirt, dust, and even minor spills. This type of design is the simplest, using the fewest parts. These things make this type of keyswitch very reliable and contribute to making rubber dome-type keyboards the most popular in service today.

If rubber dome keyboards have a drawback at all, it is that tactile feedback is not as good as many would like. Although it is reasonable for most, some prefer more tactile feedback than rubber dome keyboards normally provide.

The *membrane keyboard* is a variation on the rubber dome type in which the keys themselves are no longer separate, but are formed together in a sheet that sits on the rubber dome sheet. This severely limits key travel, and membrane keyboards are not considered usable for normal touch typing because of this. They are ideal in extremely harsh environments. Because the sheets can be bonded together and sealed from the elements, membrane keyboards can be used in situations in which no other type could survive. Many industrial applications use rubber dome keyboards especially for terminals that do not require extensive data entry, but are used to operate equipment such as cash registers.

Capacitive switches are the only nonmechanical type of switch in use today (see fig. 9.5). These are the Cadillac of keyswitches; they are much more expensive than the more common mechanical rubber dome, but also are more resistant to dirt and corrosion and offer the highest quality tactile feedback of any type of switch.

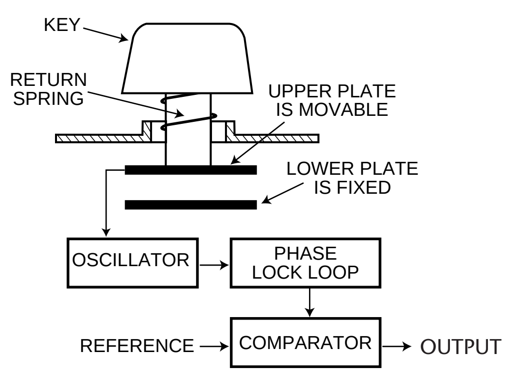

Fig. 9.5
A capacitive keyswitch.

A capacitive switch does not work by making contact between conductors. Instead, two plates usually made of plastic are connected in a switch matrix designed to detect changes in capacitance of the circuit.

When the key is depressed, the plunger moves the top plate relative to the fixed bottom plate. Usually a mechanism provides for a distinct over-center tactile feedback with a resounding “click.” As the top plate moves, the capacitance between the two plates changes and is detected by the comparator circuitry in the keyboard.

Because this type of switch does not rely on metal contacts, it is almost immune to corrosion and dirt. These switches are very resistant to key bounce problems resulting in multiple characters appearing from a single strike, and are also the most durable in the industry—rated for 25 million or more keystrokes, as opposed to 10 to 20 million for other designs. The tactile feedback is unsurpassed because a relatively loud click and strong over-center feel normally are provided. The only drawback to this design is the cost. Capacitive switch keyboards are among the most expensive designs, but the quality of the feel and durability are worth it.

Traditionally, the only vendors of capacitive keyswitch keyboards are IBM and its keyboard division Lexmark, which is why these keyboards have always seemed to stand out from the rest.

The Keyboard Interface. A keyboard consists of a set of switches mounted in a grid or array called the *key matrix*. When a switch is pressed, a processor in the keyboard itself identifies which key is pressed by identifying which grid location in the matrix shows continuity. The keyboard processor also interprets how long the key is pressed and can even handle multiple keystrokes at the same time. A 16-byte hardware buffer in the keyboard can handle rapid or multiple keystrokes, passing each one in succession to the system.

When you press a key, in most cases the contact actually bounces slightly, meaning that there are several rapid on-off cycles just as the switch makes contact. This is called *bounce*, and the processor in the keyboard is designed to filter this or *debounce* the keystroke. The keyboard processor must distinguish bounce from a double key strike actually intended by the keyboard operator. This is fairly easy because the bouncing is much more rapid than a person could simulate by striking a key quickly several times.

The keyboard in an IBM-compatible system is actually a computer itself. It communicates to the main system through a special serial data link that transmits and receives data in 11-bit packets of information consisting of eight data bits in addition to framing and control bits. Although it is indeed a serial link (the data flows on one wire), it is not compatible with the standard RS-232 serial port commonly used to connect modems. The original PC/XT setup was a one-way connection; however, the AT design is bidirectional, which means that the keyboard can receive as well as send data. In this manner, the AT keyboard could be reprogrammed in several different ways.

The processor in the original PC keyboard was an Intel 8048 microcontroller chip, but newer keyboards often use an 8049 version that has built-in ROM, or other microcontroller chips compatible with the 8048 or 8049. For example, IBM has always used a custom version of the Motorola 6805 processor in its enhanced keyboards, which is compatible with the Intel chips. The keyboard's built-in processor reads the key matrix, debounces the keypress signals, converts the keypress to the appropriate scan code, and transmits the code to the motherboard. The processors built into the keyboard contain their own RAM, possibly some ROM, and a built-in serial interface.

In the original PC/XT design, the keyboard serial interface is connected to an 8255 Programmable Peripheral Interface (PPI) chip on the motherboard of the PC/XT. This chip is then connected to the interrupt controller IRQ1 line, which is used to signal that keyboard data is available. The data itself is sent from the 8255 to the processor via I/O port address 60h. The IRQ1 signal causes the main system processor to run a subroutine (INT 9h) that interprets the keyboard scan code data and decides what to do.

In an AT-type keyboard design, the keyboard serial interface is connected to a special keyboard controller on the motherboard. This is an Intel 8042 Universal Peripheral Interface (UPI) slave microcontroller chip in the original AT design. This microcontroller is essentially another processor that has its own 2K of ROM and 128 bytes of RAM. An 8742 version that uses EPROM (Erasable Programmable Read Only Memory) can be erased and reprogrammed. Often when you get a motherboard ROM upgrade from a motherboard manufacturer, a new keyboard controller chip is included because it has somewhat dependent and updated ROM code in it as well. Some systems may use the 8041 or 8741 chips, which differ only in the amount of built-in ROM or RAM, whereas other systems now have the keyboard controller built into the main system chipset.

In an AT system, the (8048-type) microcontroller in the keyboard sends data to the (8042-type) motherboard keyboard controller on the motherboard. The motherboard-based controller also can send data back to the keyboard as well. When the keyboard controller on the motherboard receives data from the keyboard, it signals the motherboard with an IRQ1 and sends the data to the main motherboard processor via I/O port address 60h just as in the PC/XT. Acting as an agent between the keyboard and the main system processor, the 8042-type keyboard controller can translate scan codes and perform several other functions as well. Data also can be sent to the 8042 keyboard controller via port 60h, which is then passed on to the keyboard. Additionally, when the system needs to send commands to or read the status of the keyboard controller on the motherboard, it does so by reading or writing through I/O port 64h. These commands are also usually followed by data sent back and forth via port 60h.

The 8042 keyboard controller also is used by the system to control the A20 memory address line, which controls access to system memory over one megabyte. This aspect of the keyboard controller is discussed in Chapter 7, "Memory."

Typematic Functions. If a key on the keyboard is held down, it becomes *typematic*, which means that the keyboard repeatedly sends the keypress code to the motherboard. In the AT-style keyboards, the typematic rate is adjustable by sending the keyboard processor the appropriate commands. This is not possible for the earlier PC/XT keyboard types because the keyboard interface is not bidirectional.

AT-style keyboards have a programmable typematic repeat rate and delay parameter. The DOS MODE command in versions 4.0 and later enables you to set the keyboard typematic (repeat) rate as well as the delay before typematic action begins. The default value for the RATE parameter (r) is 20 for IBM AT-compatible systems, and 21 for IBM PS/2 systems. The default value for the DELAY parameter is 2. To use the MODE command to set the keyboard typematic rate and delay, use the command as follows:

MODE CON[:] [RATE=r DELAY=d]

The acceptable values for the rate "r" and the resultant typematic rate in characters per second (cps) are shown in table 9.1.

Table 9.1 DOS 4.0+ MODE Command Keyboard Typematic Rate Parameters

RATE No.	Rate ± 20%
32	30.0cps
31	26.7cps
30	24.0cps
29	21.8cps
28	20.0cps
27	18.5cps
26	17.1cps
25	16.0cps
24	15.0cps
23	13.3cps
22	12.0cps
21	10.9cps
20	10.0cps
19	9.2cps
18	8.6cps
17	8.0cps
16	7.5cps
15	6.7cps
14	6.0cps
13	5.5cps
12	5.0cps
11	4.6cps
10	4.3cps
9	4.0cps
8	3.7cps
7	3.3cps
6	3.0cps
5	2.7cps

RATE No.	Rate ± 20%
4	2.5cps
3	2.3cps
2	2.1cps
1	2.0cps

Table 9.2 shows the values for DELAY and the resultant delay time in seconds.

Table 9.2 DOS MODE Command Keyboard Typematic Delay Parameters

DELAY No.	Delay Time
1	.25sec
2	.50sec
3	.75sec
4	1.00sec

For example, I always place the following command in my AUTOEXEC.BAT file:

MODE CON: RATE=32 DELAY=1

This command sets the typematic rate to the maximum speed possible, or 30 characters per second. It also trims the delay to the minimum of 1/4 second before repeating begins. This command "turbocharges" the keyboard and makes operations requiring repeated keystrokes work much faster, such as moving within a file using arrow keys. The quick typematic action and short delay can sometimes be disconcerting to ham-fisted keyboard operators. In that case, slow typists may want to leave their keyboard speed at the default until they become more proficient.

If you have an older system or keyboard, you may receive the following message:

```
Function not supported on this computer
```

This indicates that your system, keyboard, or both do not support the bidirectional interface or commands required to change the typematic rate and delay. Upgrading the BIOS, keyboard, or both may enable this function, but it is probably not cost-effective to do this on a very old system.

Keyboard Key Numbers and Scan Codes. When you press a key on the keyboard, the processor built into the keyboard (8048- or 6805-type) reads the keyswitch location in the keyboard matrix. The processor then sends a serial packet of data to the motherboard that contains the scan code for the key that was pressed. In AT-type motherboards that also use an 8042-type keyboard controller, the 8042 chip then translates the actual keyboard scan code into one of up to three different sets of system scan codes, which are then sent to the main processor. It can be useful in some cases to know what these scan codes are, especially when troubleshooting keyboard problems, or when otherwise reading the keyboard or system scan codes directly in software.

When a keyswitch on the keyboard sticks or otherwise fails, the scan code of the failed keyswitch is usually reported by diagnostics software, including the POST (Power On Self-Test) as well as conventional disk-based diagnostics. This means that you have to identify the particular key by its scan code. Tables 9.3 through 9.7 list all the scan codes for every key on the 83-, 84-, and 101-key keyboards. By looking up the reported scan code on these charts, you can determine which keyswitch is defective or needs to be cleaned.

Note that the 101-key enhanced keyboards are capable of three different scan code sets, with Set 1 as the default. Some systems use one of the other scan code sets during the POST, including some of the PS/2 machines. For example, my P75 uses Scan Code Set 2 during the POST, but then switches to Set 1 during normal operation. This is rare, but useful to know if you are having difficulty interpreting the Scan Code number.

IBM also assigns each key a unique key number to distinguish it from the others. This is important when trying to identify keys on foreign keyboards, which may use different symbols or characters from the United States models. In the case of the enhanced keyboard, most foreign models are missing one of the keys (key #29) found on the United States version and have two other additional keys (keys #42 and #45) as well. This accounts for the 102-key total rather than the 101 keys found on the United States version.

Figure 9.6 shows the keyboard numbering and character locations for the original 83-key PC keyboard, and table 9.3 shows the scan codes for each key relative to the key number and character.

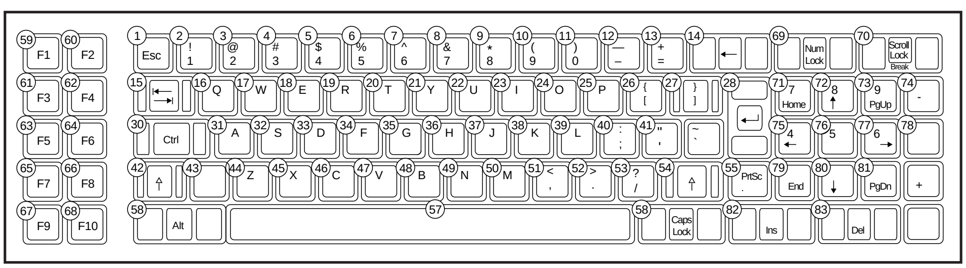

Fig. 9.6
83-key PC keyboard key number and character locations.

Table 9.3 83-Key (PC/XT) Keyboard Key Numbers and Scan Codes

Key Number	Scan Code	Key
1	01	Escape
2	02	1
3	03	2
4	04	3
5	05	4
6	06	5

Key Number	Scan Code	Key
7	07	6
8	08	7
9	09	8
10	0A	9
11	0B	0
12	0C	-
13	0D	=
14	0E	Backspace
15	0F	Tab
16	10	q
17	11	w
18	12	e
19	13	r
20	14	t
21	15	y
22	16	u
23	17	i
24	18	o
25	19	p
26	1A	[
27	1B	]
28	1C	Enter
29	1D	Ctrl
30	1E	a
31	1F	s
32	20	d
33	21	f
34	22	g
35	23	h
36	24	j
37	25	k
38	26	l
39	27	;
40	28	‘
41	29	`
42	2A	Left Shift
43	2B	\
44	2C	z
45	2D	x

(continues)

Table 9.3 Continued

Key Number	Scan Code	Key
46	2E	c
47	2F	v
48	30	b
49	31	n
50	32	m
51	33	,
52	34	.
53	35	/
54	36	Right Shift
55	37	*
56	38	Alt
57	39	Space bar
58	3A	Caps Lock
59	3B	F1
60	3C	F2
61	3D	F3
62	3E	F4
63	3F	F5
64	40	F6
65	41	F7
66	42	F8
67	43	F9
68	44	F10
69	45	Num Lock
70	46	Scroll Lock
71	47	Keypad 7 (Home)
72	48	Keypad 8 (Up arrow)
73	49	Keypad 9 (PgUp)
74	4A	Keypad -
75	4B	Keypad 4 (Left arrow)
76	4C	Keypad 5
77	4D	Keypad 6 (Right arrow)
78	4E	Keypad +
79	4F	Keypad 1 (End)
80	50	Keypad 2 (Down arrow)
81	51	Keypad 3 (PgDn)
82	52	Keypad 0 (Ins)
83	53	Keypad . (Del)

Figure 9.7 shows the keyboard numbering and character locations for the original 84-key AT keyboard. Table 9.4 shows the scan codes for each key relative to the key number and character.

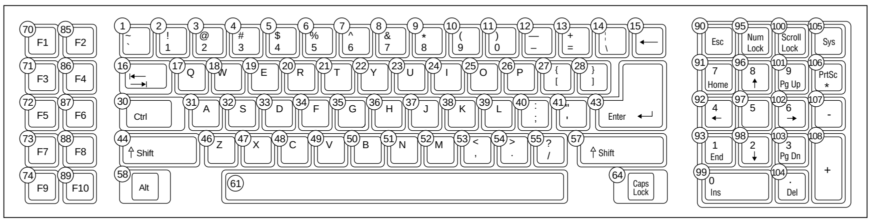

Fig. 9.7
84-key AT keyboard key number and character locations.

Table 9.4 84-Key AT Keyboard Key Numbers and Scan Codes

Key Number	Scan Code	Key
1	29	`
2	02	1
3	03	2
4	04	3
5	05	4
6	06	5
7	07	6
8	08	7
9	09	8
10	0A	9
11	0B	0
12	0C	-
13	0D	=
14	2B	\
15	0E	Backspace
16	0F	Tab
17	10	q
18	11	w
19	12	e
20	13	r
21	14	t
22	15	y
23	16	u
24	17	i

(continues)

Table 9.4 Continued

Key Number	Scan Code	Key
25	18	o
26	19	p
27	1A	[
28	1B	]
30	1D	Ctrl
31	1E	a
32	1F	s
33	20	d
34	21	f
35	22	g
36	23	h
37	24	j
38	25	k
39	26	l
40	27	;
41	28	‘
43	1C	Enter
44	2A	Left Shift
46	2C	z
47	2D	x
48	2E	c
49	2F	v
50	30	b
51	31	n
52	32	m
53	33	,
54	34	.
55	35	/
57	36	Right Shift
58	38	Alt
61	39	Space bar
64	3A	Caps Lock
65	3C	F2
66	3E	F4
67	40	F6
68	42	F8
69	44	F10
70	3B	F1
71	3D	F3

Key Number	Scan Code	Key
72	3F	F5
73	41	F7
74	43	F9
90	01	Escape
91	47	Keypad 7 (Home)
92	4B	Keypad 4 (Left arrow)
93	4F	Keypad 1 (End)
95	45	Num Lock
96	48	Keypad 8 (Up arrow)
97	4C	Keypad 5
98	50	Keypad 2 (Down arrow)
99	52	Keypad 0 (Ins)
100	46	Scroll Lock
101	49	Keypad 9 (PgUp)
102	4D	Keypad 6 (Right arrow)
103	51	Keypad 3 (PgDn)
104	53	Keypad . (Del)
105	54	SysRq
106	37	Keypad *
107	4A	Keypad -
108	4E	Keypad +

Figure 9.8 shows the keyboard numbering and character locations for the 101-key enhanced keyboard. Tables 9.5 through 9.7 show each of the three scan code sets showing the scan code for each key relative to the key number and character. Note that Scan Code Set 1 is the default; the other two are rarely used. Figure 9.9 shows the layout of a typical foreign language 102-key version of the enhanced keyboard, in this case a U.K. English version.

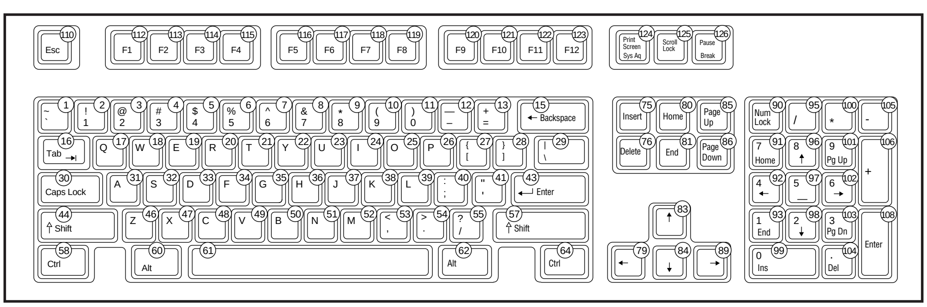

Fig. 9.8

101-key enhanced keyboard key number and character locations (U.S. version).

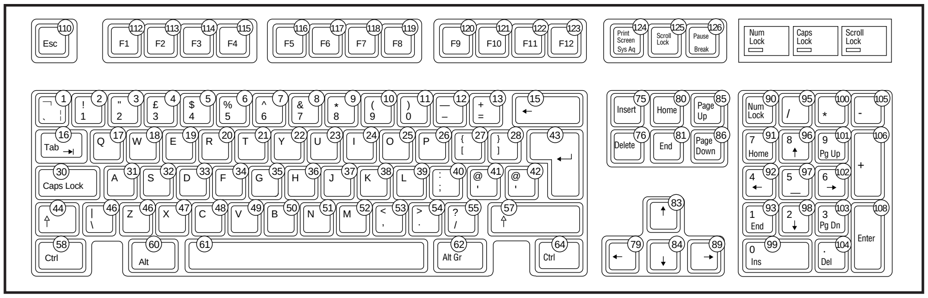

Fig. 9.9
102-key enhanced keyboard key number and character locations (U.K. English version).

Table 9.5 101/102-Key (Enhanced) Keyboard Key Numbers and Scan Codes (Set 1)

Key Number	Scan Code	Key
1	29	`
2	02	1
3	03	2
4	04	3
5	05	4
6	06	5
7	07	6
8	08	7
9	09	8
10	0A	9
11	0B	0
12	0C	-
13	0D	=
15	0E	Backspace
16	0F	Tab
17	10	q
18	11	w
19	12	e
20	13	r
21	14	t
22	15	y
23	16	u
24	17	i
25	18	o
26	19	p
27	1A	[

Key Number	Scan Code	Key
28	1B	]
29	2B	\ (101-key *only*)
30	3A	Caps Lock
31	1E	a
32	1F	s
33	20	d
34	21	f
35	22	g
36	23	h
37	24	j
38	25	k
39	26	l
40	27	;
41	28	'
42	2B	# (102-key *only*)
43	1C	Enter
44	2A	Left Shift
45	56	\ (102-key *only*)
46	2C	z
47	2D	x
48	2E	c
49	2F	v
50	30	b
51	31	n
52	32	m
53	33	,
54	34	.
55	35	/
57	36	Right Shift
58	1D	Left Ctrl
60	38	Left Alt
61	39	Space bar
62	E0,38	Right Alt
64	E0,1D	Right Ctrl
75	E0,52	Insert
76	E0,53	Delete
79	E0,4B	Left arrow
80	E0,47	Home
81	E0,4F	End

(continues)

Table 9.5 Continued

Key Number	Scan Code	Key
83	E0,48	Up arrow
84	E0,50	Down arrow
85	E0,49	Page Up
86	E0,51	Page Down
89	E0,4D	Right arrow
90	45	Num Lock
91	47	Keypad 7 (Home)
92	4B	Keypad 4 (Left arrow)
93	4F	Keypad 1 (End)
95	E0,35	Keypad /
96	48	Keypad 8 (Up arrow)
97	4C	Keypad 5
98	50	Keypad 2 (Down arrow)
99	52	Keypad 0 (Ins)
100	37	Keypad *
101	49	Keypad 9 (PgUp)
102	4D	Keypad 6 (Left arrow)
103	51	Keypad 3 (PgDn)
104	53	Keypad . (Del)
105	4A	Keypad -
106	4E	Keypad +
108	E0,1C	Keypad Enter
110	01	Escape
112	3B	F1
113	3C	F2
114	3D	F3
115	3E	F4
116	3F	F5
117	40	F6
118	41	F7
119	42	F8
120	43	F9
121	44	F10
122	57	F11
123	58	F12
124	E0,2A,E0,37	Print Screen
125	46	Scroll Lock
126	E1,1D,45,E1,9D,C5	Pause

Table 9.6 101/102-Key (Enhanced) Keyboard Key Numbers and Scan Codes (Set 2)

Key Number	Scan Code	Key
1	0E	`
2	16	1
3	1E	2
4	26	3
5	25	4
6	2E	5
7	36	6
8	3D	7
9	3E	8
10	46	9
11	45	0
12	4E	-
13	55	=
15	66	Backspace
16	0D	Tab
17	15	q
18	1D	w
19	24	e
20	2D	r
21	2C	t
22	35	y
23	3C	u
24	43	i
25	44	o
26	4D	p
27	54	[
28	5B	]
29	5D	\ (101-key *only*)
30	58	Caps Lock
31	1C	a
32	1B	s
33	23	d
34	2B	f
35	34	g
36	33	h
37	3B	j
38	42	k

(continues)

Table 9.6 Continued

Key Number	Scan Code	Key
39	4B	l
40	4C	;
41	52	'
42	5D	# (102-key *only*)
43	5A	Enter
44	12	Left Shift
45	61	\ (102-key *only*)
46	1A	z
47	22	x
48	21	c
49	2A	v
50	32	b
51	31	n
52	3A	m
53	41	,
54	49	.
55	4A	/
57	59	Right Shift
58	14	Left Ctrl
60	11	Left Alt
61	29	Space bar
62	E0,11	Right Alt
64	E0,14	Right Ctrl
75	E0,70	Insert
76	E0,71	Delete
79	E0,6B	Left arrow
80	E0,6C	Home
81	E0,69	End
83	E0,75	Up arrow
84	E0,72	Down arrow
85	E0,7D	Page Up
86	E0,7A	Page Down
89	E0,74	Right arrow
90	77	Num Lock
91	6C	Keypad 7 (Home)
92	6B	Keypad 4 (Left arrow)
93	69	Keypad 1 (End)
95	E0,4A	Keypad /

Key Number	Scan Code	Key
96	75	Keypad 8 (Up arrow)
97	73	Keypad 5
98	72	Keypad 2 (Down arrow)
99	70	Keypad 0 (Ins)
100	7C	Keypad *
101	7D	Keypad 9 (PgUp)
102	74	Keypad 6 (Left arrow)
103	7A	Keypad 3 (PgDn)
104	71	Keypad . (Del)
105	7B	Keypad -
106	E0,5A	Keypad +
108	E0,5A	Keypad Enter
110	76	Escape
112	05	F1
113	06	F2
114	04	F3
115	0C	F4
116	03	F5
117	0B	F6
118	83	F7
119	0A	F8
120	01	F9
121	09	F10
122	78	F11
123	07	F12
124	E0,12,E0,7C	Print Screen
125	7E	Scroll Lock
126	E1,14,77,E1,F0,14,F0,77	Pause

Table 9.7 101/102-Key (Enhanced) Keyboard Key Numbers and Scan Codes (Set 3)

Key Number	Scan Code	Key
1	0E	`
2	16	1
3	1E	2
4	26	3
5	25	4

(continues)

Table 9.7 Continued

Key Number	Scan Code	Key
6	2E	5
7	36	6
8	3D	7
9	3E	8
10	46	9
11	45	0
12	4E	-
13	55	=
15	66	Backspace
16	0D	Tab
17	15	q
18	1D	w
19	24	e
20	2D	r
21	2C	t
22	35	y
23	3C	u
24	43	i
25	44	o
26	4D	p
27	54	[
28	5B	]
29	5C	\ (101-key *only*)
30	14	Caps Lock
31	1C	a
32	1B	s
33	23	d
34	2B	f
35	34	g
36	33	h
37	3B	j
38	42	k
39	4B	l
40	4C	;
41	52	‘
42	53	# (102-key *only*)
43	5A	Enter
44	12	Left Shift

Key Number	Scan Code	Key
45	13	\ (102-key *only*)
46	1A	z
47	22	x
48	21	c
49	2A	v
50	32	b
51	31	n
52	3A	m
53	41	,
54	49	.
55	4A	/
57	59	Right Shift
58	11	Left Ctrl
60	19	Left Alt
61	29	Space bar
62	39	Right Alt
64	58	Right Ctrl
75	67	Insert
76	64	Delete
79	61	Left arrow
80	6E	Home
81	65	End
83	63	Up arrow
84	60	Down arrow
85	6F	Page Up
86	6D	Page Down
89	6A	Right arrow
90	76	Num Lock
91	6C	Keypad 7 (Home)
92	6B	Keypad 4 (Left arrow)
93	69	Keypad 1 (End)
95	77	Keypad /
96	75	Keypad 8 (Up arrow)
97	73	Keypad 5
98	72	Keypad 2 (Down arrow)
99	70	Keypad 0 (Ins)
100	7E	Keypad *
101	7D	Keypad 9 (PgUp)
102	74	Keypad 6 (Left arrow)

(continues)

Table 9.7 Continued

Key Number	Scan Code	Key
103	7A	Keypad 3 (PgDn)
104	71	Keypad . (Del)
105	84	Keypad -
106	7C	Keypad +
108	79	Keypad Enter
110	08	Escape
112	07	F1
113	0F	F2
114	17	F3
115	1F	F4
116	27	F5
117	2F	F6
118	37	F7
119	3F	F8
120	47	F9
121	4F	F10
122	56	F11
123	5E	F12
124	57	Print Screen
125	5F	Scroll Lock
126	62	Pause

These key number figures and scan code tables can be very useful when troubleshooting stuck or failed keys on a keyboard. Diagnostics can report the defective keyswitch by the scan code, which varies from keyboard to keyboard as to which character it represents as well as the location.

Keyboard/Mouse Interface Connectors. Keyboards have a cable available with one of two primary types of connectors at the system end. Most aftermarket keyboards have a cable connected inside the keyboard case on the keyboard end and require that you open up the keyboard case to disconnect or test. Actual IBM enhanced keyboards use a unique cable assembly that plugs into both the keyboard and the system unit. This makes cable interchange or replacement an easy plug-in affair. A special connector called an SDL (Shielded Data Link) is used on the keyboard end, and the appropriate DIN (Deutsche Industrie Norm) connector at the PC end. Any IBM keyboard or cable can be ordered separately as a spare part. The newer enhanced keyboards come with an externally detachable keyboard cable that plugs into the keyboard port with a special connector, much like a telephone connector. The other end of the cable is one of the following two types:

- Earlier systems use a 5-pin DIN connector.
- PS/2 and most Low Profile compatible systems use a 6-pin mini-DIN connector.

Figure 9.10 and table 9.8 show the physical layout and pinouts of all the respective keyboard connector plugs and sockets.

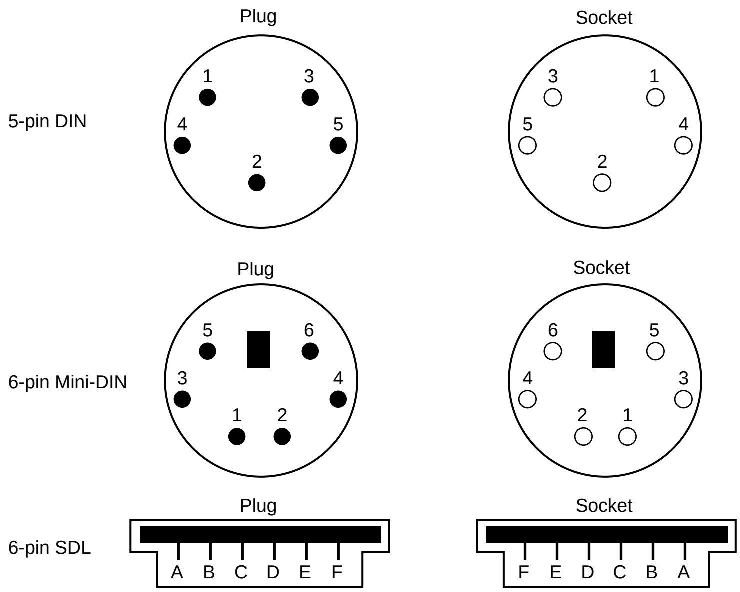

Fig. 9.10
Keyboard and mouse connectors.

Table 9.8 Keyboard Connector Signals

Signal Name	5-Pin DIN	6-Pin Mini-DIN	6-Pin SDL
Keyboard Data	2	1	B
Ground	4	3	C
+5v	5	4	E
Keyboard Clock	1	5	D
Not Connected	—	2	A
Not Connected	—	6	F
Not Connected	3	—	—

DIN = German Industrial Norm (Deutsche Industrie Norm), a committee that sets German dimensional standards
SDL = Shielded Data Link, a type of shielded connector created by AMP and used by IBM and others for keyboard cables

Motherboard mouse connectors use the 6-pin mini-DIN connector and have the same pinout and signal descriptions as the keyboard connector; however, the data packets are incompatible. This means that you easily can plug a motherboard mouse (PS/2 style) into a mini-DIN keyboard connector, or plug the mini-DIN type keyboard connector into a motherboard mouse port; however, neither would work properly in this situation.

Keyboard Troubleshooting and Repair

Keyboard errors are usually caused by two simple problems. Other more difficult intermittent problems can arise, but these are also much more rare. The most common problems are as follows:

- Defective cables
- Stuck keys

Defective cables are easy to spot if the failure is not intermittent. If the keyboard stops working altogether or every keystroke results in an error or incorrect character, then likely the cable is the culprit. Troubleshooting is simple, especially if you have a spare cable on hand. Simply replace the suspected cable with one from a known working keyboard, and verify to see if the problem still exists. If it does, the problem must be elsewhere. You also can test the cable for continuity with it removed from the keyboard by using a DMM (Digital Multi-Meter). DMMs that have a built-in audible continuity tester make this procedure much easier to perform. Wiggle the ends of the cable as you check each wire to make sure that there are no intermittent connections. If you discover a problem with the continuity in one of the wires, replace the cable or the entire keyboard if that is cheaper. Because replacement keyboards are so inexpensive, sometimes it can be cheaper to replace the entire unit than to get a new cable!

Often you first discover a problem with a keyboard because the system has an error during the POST. Most systems use error codes in a 3xx numeric format to distinguish the keyboard. If you have any such errors during the POST, write them down. Some BIOSs do not use cryptic numeric error codes and simply state something like the following:

```
Keyboard stuck key failure
```

This message normally would be displayed by a system with a Phoenix BIOS if a key were stuck. Unfortunately, the message does not identify which key it is!

If your 3xx (keyboard) error is preceded by a two-digit hexadecimal number, this number is the scan code of a failing or stuck keyswitch. Look up the scan code in the tables provided in this section to determine which keyswitch is the culprit. These charts tell you to which key the scan code refers. By removing the keycap of the offending key and cleaning the switch, you often can solve the problem.

For a simple test of the motherboard keyboard connector, you can check voltages on some of the pins. Use figure 9.10 in the preceding section as a guide, and measure the voltages on various pins of the keyboard connector. To prevent possible damage to the system or keyboard, first turn off the power before disconnecting the keyboard.

Then unplug the keyboard and turn the power back on. Make measurements between the ground pin and the other pins according to table 9.9. If the voltages read are within these specifications, the motherboard keyboard circuitry is probably OK.

Table 9.9 Keyboard Connector Specifications

DIN Connector Pin	Mini-DIN Connector Pin	Signal	Voltage
1	5	Keyboard Clock	+2.0v to +5.5v
2	1	Keyboard Data	+4.8v to +5.5v
3	—	Reserved	—
4	3	Ground	—
5	4	+5v Power	+2.0v to +5.5v

If your measurements don't match these voltages, the motherboard might be defective. Otherwise, the keyboard cable or keyboard might be defective. If you suspect the cable is the problem, the easiest thing to do is replace the keyboard cable with a known good one. If the system still does not work normally, then you may have to replace either the entire keyboard or the motherboard.

In some systems, such as many IBM PS/2s, the motherboard keyboard and mouse connectors are protected by a fuse that can be replaced. Look for any type of fuse on the motherboard in the vicinity of the keyboard or mouse connectors. Other systems may have a socketed keyboard controller chip (8042-type). In that case, it may be possible to repair the motherboard keyboard circuit by simply replacing this chip. Because these chips have ROM code in them, it is best to get the replacement from the motherboard or BIOS manufacturer.

A list of standard POST and diagnostics keyboard error codes is displayed as follows:

Error Code	Description
3xx	Keyboard errors
301	Keyboard reset or stuck-key failure (XX 301, XX = scan code in hex)
302	System unit keylock switch is locked
302	User indicated keyboard test error
303	Keyboard or system-board error; keyboard controller failure
304	Keyboard or system-board error; keyboard clock high
305	Keyboard +5v error; PS/2 keyboard fuse (on motherboard) blown
341	Keyboard error
342	Keyboard cable error
343	Keyboard LED card or cable failure
365	Keyboard LED card or cable failure
366	Keyboard interface cable failure
367	Keyboard LED card or cable failure

Disassembly Procedures and Cautions. Repairing and cleaning a keyboard often requires you to take it apart. When performing this task, you must know when to stop! Some keyboards literally come apart into hundreds of little pieces that are almost impossible to reassemble if you go too far. An IBM keyboard generally has these four major parts:

- Cable
- Case
- Keypad assembly
- Keycaps

You easily can break down a keyboard to these major components and replace any of them, but don't disassemble the keypad assembly or you'll be showered with hundreds of tiny springs, clips, and keycaps. Finding all these parts (several hundred of them) and piecing the unit back together is not a fun way to spend time. You also may not be able to reassemble the keyboard properly. Figure 9.13 shows a typical keyboard with the case opened.

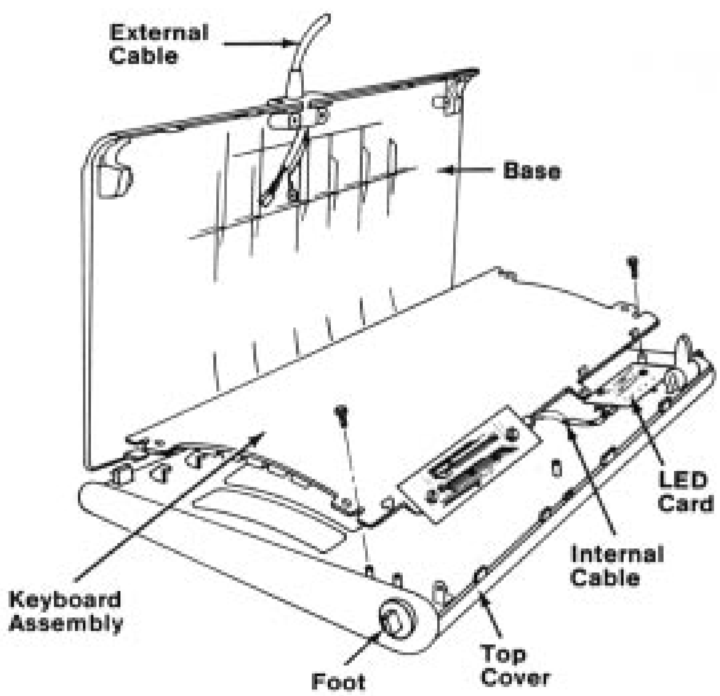

Fig. 9.11
Typical keyboard components.

Another problem is that you cannot purchase the smaller parts separately, such as contact clips and springs. The only way to obtain these parts is from another keyboard. If you ever have a keyboard that's beyond repair, keep it around for these parts. They might come in handy some day.

Most repair operations are limited to changing the cable or cleaning some component of the keyboard, from the cable contact ends to the key contact points. The keyboard cable takes quite a bit of abuse, and therefore can fail easily. The ends are stretched, tugged, pulled, and generally handled roughly. The cable uses strain reliefs, but you still might have problems with the connectors making proper contact at each end or even with wires that have broken inside the cable. You might want to carry a spare cable for every type of keyboard you have.

All keyboard cables plug into the keyboard and PC with connectors, and you can change the cables easily without having to splice wires or solder connections. With the earlier 83-key PC and 84-key AT keyboards, you must open the case to access the connector to which the cable attaches. On the newer 101-key enhanced keyboards from IBM and Lexmark, the cable plugs into the keyboard from the outside of the case, using a modular jack and plug similar to a telephone jack. This design also makes the IBM/Lexmark keyboards universally usable on nearly any system (except the original PC) by easily switching the cable.

The only difference, for example, between the enhanced keyboards for an IBM AT and an IBM PS/2 system is the attached cable. PS/2 systems use a tan cable with a smaller plug on the computer side. The AT cable is black and has the larger DIN-type plug on the computer side. You can interchange the enhanced keyboards as long as you use the correct cable for the system.

The only feasible ways to repair a keyboard are to replace the cable and clean the individual keyswitch assemblies, the entire keypad, or the cable contact ends. The individual spring and keyswitch assemblies are not available as a separate part, and disassembling the unit to that level is not advisable because of the difficulty in reassembling it. Other than cleaning a keyboard, the only thing you can do is replace the entire keypad assembly (virtually the entire keyboard) or the cable.

Cleaning a Keyboard. One of the best ways to maintain a keyboard in top condition is periodic cleaning. As preventive maintenance, you should vacuum the keyboard weekly or at least monthly. You also can use canned compressed air available at electronics supply houses to blow the dust and dirt out instead of using a vacuum. Before you dust a keyboard with the compressed air, turn the keyboard upside down so that the particles of dirt and dust collected inside can fall out.

On all keyboards, each keycap is removable, which can be handy if a key sticks or acts erratically. For example, a common problem is a key that doesn't work every time you press it. This problem usually results from dirt collecting under the key. An excellent tool for removing keycaps on almost any keyboard is the U-shaped chip-puller tool. You simply slip the hooked ends of the tool under the keycap, squeeze the ends together to grip the underside of the keycap, and lift up. IBM sells a tool specifically for removing keycaps from its keyboards, but the chip puller works even better. After removing the cap, spray some compressed air into the space under the cap to dislodge the dirt. Then replace the cap and check the action of the key.

When you remove the keycaps, be careful not to remove the space bar on the original 83-key PC and 84-key AT-type keyboards. This bar is very difficult to reinstall. The newer 101-key units use a different wire support that can be removed and replaced much more easily.

Spills also can be a problem. If you tip a soft drink or cup of coffee into a keyboard, you don't necessarily have a disaster. You should immediately (or as soon as possible) flush out the keyboard with distilled water. Partially disassemble the keyboard and use the water to wash the components. (See the following section for disassembly instructions.) If the spilled liquid has dried, let the keyboard soak in some of the water for a while. Then, when you're sure that the keyboard is clean, pour another gallon or so of distilled water over it and through the key switches to wash away any residual dirt. After the unit dries completely, it should be perfectly functional. You may be surprised to know that you can drench your keyboard with water, and it will not harm the components. Just make sure that you use distilled water, which is free from residue or mineral content. Also make sure that the keyboard is fully dry before you attempt to use it, or some of the components might short out: water is a conductor of electricity.

Replacement Keyboards. In most cases, it is cheaper or more cost-effective to simply replace a keyboard rather than repair it. This is especially true if the keyboard has an internal malfunction or if one of the keyswitches is defective. Replacement parts for keyboards are almost impossible to procure, and in most cases the installation of any repair part is difficult. In addition, many of the keyboards supplied with lower cost compatible machines leave much to be desired. They often have a mushy feel, with little or no tactile feedback. A poor keyboard can make using a system a very frustrating experience, especially if you are a touch typist. For all these reasons, it is often a good idea to consider replacing an existing keyboard with something better.

Perhaps the highest quality keyboards in the entire computer industry are those made by IBM, or more accurately Lexmark. Several years ago IBM spun off its keyboard and printer divisions as a separate company called Lexmark. Lexmark now manufactures all IBM brand keyboards and printers and sells them not only to IBM, but to compatible vendors or even end users as well. This means that if you are lucky, your compatible system comes with a Lexmark keyboard, but if not, you can purchase one separately on your own.

Table 9.10 shows the part numbers of all IBM-labeled keyboards and cables. These numbers can serve as a reference when you are seeking a replacement IBM keyboard from IBM directly or from third-party companies. Many third-party companies sell IBM label keyboards for much less than IBM, in both new and refurbished form. Remember that you also can purchase these exact same keyboards through Lexmark, although they do not come with an IBM label.

Table 9.10 IBM Keyboard and Cable Part Numbers

Description	Part Number
83-key U.S. PC Keyboard assembly with cable	8529297
Cable assembly for 83-key PC Keyboard	8529168
84-key U.S. AT Keyboard assembly with cable	8286165
Cable assembly for 84-key Keyboard	8286146
101-key U.S. Keyboard without LED panel	1390290
101-key U.S. Keyboard with LED panel	6447033
101-key U.S. Keyboard with LED panel (PS/2 logo)	1392090
6-foot cable for enhanced keyboard (DIN plug)	6447051
6-foot cable for enhanced keyboard (mini-DIN plug)	61X8898
6-foot cable for enhanced keyboard (shielded mini-DIN plug)	27F4984
10-foot cable for enhanced keyboard (mini-DIN plug)	72X8537

Note that the original 83/84-key IBM keyboards are sold with a cable that has the larger, 5-pin DIN connector already attached. IBM enhanced keyboards are always sold (at least by IBM) without a cable. You must order the proper cable as a separate item. Cables are available to connect the keyboards to either the older system units that use the larger DIN connector or to PS/2 systems (and many compatibles) that use the smaller mini-DIN connector.

Recently, IBM has started selling complete keyboard assemblies under a program called "IBM Options." This program is designed to sell these components in the retail channel to end users of both IBM and compatible systems from other vendors. Items under the IBM Options program are sold through normal retail channels such as CompUSA, Elek Tek, Computer Discount Warehouse (CDW), and so on. These items are also priced much cheaper than items purchased as spare parts; they include a full warranty and are sold as complete packages including cables. Table 9.11 lists some of the IBM Options keyboards and part numbers.

Table 9.11 IBM Options Keyboards (Sold Retail)

Description	Part Number
IBM enhanced keyboard (cable w/DIN plug)	92G7454
IBM enhanced keyboard (cable w/mini-DIN plug)	92G7453
IBM enhanced keyboard, built-in Trackball (cable w/DIN plug)	92G7456
IBM enhanced keyboard, built-in Trackball (cable w/mini-DIN plug)	92G7455
IBM enhanced keyboard, integrated Trackpoint II (cable w/mini-DIN plug)	92G7461

The IBM/Lexmark keyboards use capacitive keyswitches, which are the most durable and lowest maintenance. These switches have no electrical contacts and instead rely on changing capacitance to signal a keystroke in the switch matrix. This type of design does not have wear points like a mechanical switch, and has no metal electrical contacts, which makes it virtually immune to the dirt and corrosion problems that plague other designs.

The extremely positive tactile feedback of the IBM/Lexmark design is also a benchmark of comparison for the rest of the industry. Although keyboard feel is somewhat a personal preference issue, I have never used a keyboard that feels better than the IBM/Lexmark designs. I now equip every system I use with a Lexmark keyboard, including the many clone or compatible systems I use. You can purchase these keyboards through Lexmark or a Lexmark distributor for very reasonable prices. You can find IBM-labeled models available from advertisers in the *Processor* or *Computer Hotline* magazines (see Appendix B) selling for as little as $80 or less.

IBM/Lexmark also sells other versions for very reasonable prices. Many different models are available, including some with a built-in trackball or even the revolutionary Trackpoint II pointing device. Trackpoint II refers to a small stick mounted between the G, H, and B keys. This device is an IBM/Lexmark exclusive and was first featured on the IBM Thinkpad laptop systems, although they now sell the keyboards for use on compatibles and also are licensing the technology to others such as Toshiba. Note that this keyboard only comes with the mini-DIN type connectors for both the keyboard and Trackpoint II portions, and only works with a Motherboard (PS/2 type) mouse port.

Other manufacturers of high-quality keyboards are available. A company called Lite-On manufactures keyboards very similar in feel to the IBM/Lexmark units. They have excellent tactile feedback with a positive click sound as well. They are my second choice to a Lexmark unit. Maxi-Switch also makes a high-quality aftermarket keyboard used by a number of compatible manufacturers including Gateway. These also have a good feel and are recommended. Maxi-Switch also can make its keyboards with your own company logo on them, which is ideal for small clone manufacturers looking for name brand recognition.

Reference Material

If you are interested in more details about keyboard design or interfacing, a company called Annabooks (see Appendix B) publishes a book/disk package called *PC Keyboard Design*. This document defines the protocol between the keyboard and computer for both XT and AT types and includes schematics and keyboard controller source code. The kit includes a license to use the source code and costs $249.

Other excellent sources of information are the various technical-reference manuals put out by IBM. Appendix A contains a list of the important IBM reference manuals in which you can find much valuable information. This information is especially valuable to compatible system manufacturers, because they often do not put out the same level of technical information as IBM, and compatible systems are in many ways similar or even identical to one or more IBM systems. After all, that is why they are called "IBM compatible!" Much of my personal knowledge and expertise comes from poring over the various IBM technical-reference manuals.

Mice

The mouse was invented in 1964 by Douglas Englebart, who was at the time working at the Stanford Research Institute (SRI)—a think tank sponsored by Stanford University. The mouse was officially called an "X-Y Position Indicator for a Display System." Xerox later applied the mouse to its revolutionary Alto computer system in 1973. Unfortunately, at the time these systems were experimental and used purely for research. In 1979, several people from Apple, including Steve Jobs, were invited to see the Alto and the software that ran the system. Steve Jobs was blown away by what he saw as the future of computing, which included the use of the mouse as a pointing device and the graphical user interface it operated. Apple promptly incorporated these features into what was to become the Lisa computer and lured away 15 to 20 Xerox scientists to work on the Apple system. Although Xerox released the Star 8010 computer using this technology in 1981, it was expensive, poorly marketed, and perhaps way ahead of its time. Apple then released the Lisa computer in 1983, and it also was not a runaway success largely because of its $10,000 list price; but by then Jobs already had Apple working on the low-cost successor to the Lisa, the Macintosh. The Apple Macintosh was introduced in 1984; although it was not an immediate hit, the Macintosh has been steadily growing in popularity since that time. Many credit the Macintosh with inventing the mouse and GUI, but, as you can see, this technology was actually borrowed from others including SRI and Xerox. Certainly the Macintosh, and now Microsoft Windows and OS/2, have gone on to popularize this interface and bring it to the legion of IBM-compatible systems.

Although the mouse did not catch on quickly in the IBM-compatible marketplace, today the GUIs for PC systems such as Windows and OS/2 virtually demand the use of a mouse. Because of this, it is common for a mouse to be sold with virtually every new system on the market.

Mice come in many shapes and sizes from many different manufacturers. The largest manufacturers of mice are Microsoft and Logitech. Even though mice may come in different varieties, their actual use and care differ very little. The mouse consists of several components as follows:

- A housing that you hold in your hand and move around on your desktop
- A roller ball that signals movement to the system
- Several (usually two) buttons for making selections
- A cable for connecting the mouse to the system
- An interface connector to attach the mouse to the system

The housing is made of plastic and consists of very few moving parts. On top of the housing, where your fingers normally reside, are buttons. There may be any number of buttons, but typically in the PC world there are only two. If additional buttons are on your mouse, specialized software is required for them to operate. On the bottom of the housing is a small rubber ball that rotates as you move the mouse across the tabletop.

The movements of this rubber ball are translated into electrical signals transmitted to the computer across the cable. Some mice use a special optical sensor that detects movement over a grid. These optical mice have fallen into disfavor because they work only if you use a special grid pad underneath them.

The cable can be any length, but is typically between four and six feet long. (If you have a choice on the length of cable to purchase, go for a longer one. This allows easier placement of the mouse in relation to your computer.)

The connector used with your mouse depends on the type of interface you are using. Three basic interfaces are used, with a fourth combination device possible as well.

After the mouse is connected to your computer, it communicates with your system through the use of a device driver, which can be either separately loaded or built into the system software. For example, no separate drivers are needed to use a mouse with Windows or OS/2, but to use the mouse with most DOS-based programs requires a separate driver to be loaded. Regardless of whether it is built-in, the driver translates the electrical signals sent from the mouse into positional information and information that indicates the status of the buttons.

Internally, a mouse is very simple as well. The ball usually rests against two rollers, one for translating the X-axis movement and the other for the Y-axis. These rollers are usually connected to small discs with shutters that alternately block and allow the passage of light. Small optical sensors detect movement of the wheels by watching an internal infrared light blink on and off as the shutter wheel rotates and “chops” the light. These blinks are then translated into movement along the various axes. This type of setup is called an *Opto-mechanical mechanism* and is by far the most popular in use today (see fig. 9.12).

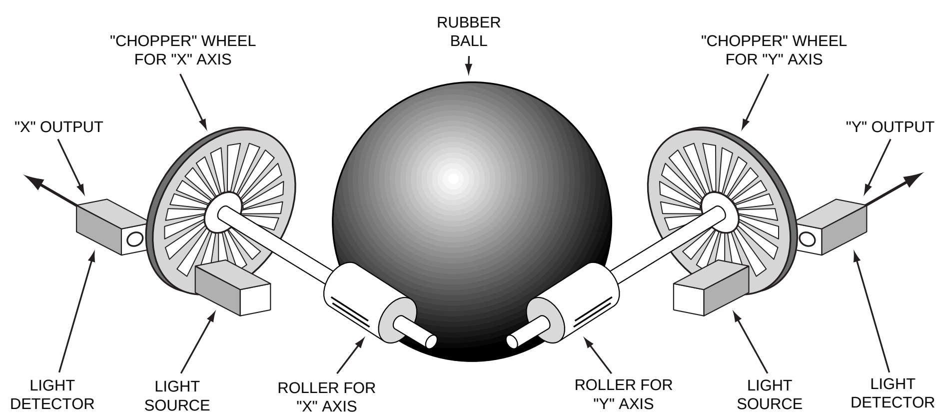

Fig. 9.12
Typical Opto-mechanical mouse mechanism.

The following sections explain the different types of mouse interfaces and how you can care for the mice.

Mouse Interface Types

Mice can be connected to your computer through the following three devices:

- Serial interface
- Dedicated motherboard mouse port
- Bus-card interface

Serial. A popular method of connecting a mouse to most older IBM-compatible computers is through a serial interface. As with other serial devices, the connector on the end on the mouse cable is either a 9-pin or 25-pin male connector. Only a couple of pins in the DB-9 or DB-25 connectors are used for communications between the mouse and the device driver, but the mouse connector typically has all 9 or 25 pins present.

Because most PCs come with two serial ports, a serial mouse can be plugged into either COM1: or COM2:. The device driver, when initializing, searches the ports to determine which one the mouse is connected to.

Because a serial mouse does not connect to the system directly, it does not use system resources by itself. Instead, the resources used are those used by the serial port to which it is connected. For example, if you have a mouse connected to COM2:, then it most likely uses IRQ3 and I/O Port Addresses 2F8h–2FFh.

Motherboard Mouse Port (PS/2). Most newer computers now come with a dedicated mouse port built into the motherboard. This was started by IBM with the PS/2 systems in 1987, so this interface often is referred to as a PS/2 mouse interface. Using this term should not imply that such a mouse can work only with a PS/2, but instead means that the mouse can connect to any system that has a dedicated mouse port on the motherboard.

A motherboard mouse connector usually is exactly the same as the mini-DIN connector used for newer keyboards. In fact, the motherboard mouse port is connected to the 8042-type keyboard controller found on the motherboard. All the PS/2 computers include mini-DIN keyboard and mouse port connectors on the back. Most compatible Slimline computers also have these same connectors for space reasons. Other motherboards have a pin-header type connector for the mouse port because most standard cases do not have a provision for the mini-DIN mouse connector. In that case, an adapter cable is usually supplied with the system that adapts the pin-header connector on the motherboard to the standard mini-DIN type connector used for the Motherboard Mouse.

Connecting a mouse to the built-in mouse port is the best method of connection because you don't lose any interface slots or any serial ports, and the performance is not limited by the serial port circuitry. The standard resource usage for a motherboard (or PS/2) mouse port is IRQ 12 and I/O Port Addresses 60h and 64h. Because the motherboard mouse port uses the 8042-type keyboard controller chip, the port addresses are those of this chip. IRQ 12 is a 16-bit interrupt that is usually free on most systems, and must remain free on any ISA bus systems that have a motherboard mouse port because interrupt sharing is not allowed with the ISA bus.

Serial and Motherboard Mouse Port (PS/2). A hybrid type of mouse can plug into both a serial port and a motherboard mouse port connection. This combo serial-PS/2 mouse is the most popular type because it is more flexible than the other single design types. Circuitry in this mouse automatically detects which type of port it is connected to and configures the mouse automatically to work. These mice usually come with a mini-DIN connector on the end of their cable and also include an adapter between the mini-DIN to 9- or 25-pin serial port connector.

Sometimes people use adapters to try to connect a serial mouse to a motherboard mouse port, or a motherboard mouse to a serial port. This does not work and is not the fault of the adapter. If the mouse does not explicitly state that it is both a serial and PS/2 type mouse, then it does not work on either interface, but instead works only on the single interface for which it was designed. Most of the time, you find the designation for what type of mouse you have printed on the bottom of the mouse.

Bus. A bus mouse is typically used in systems that do not have a motherboard mouse port or any available serial ports. The name *bus mouse* is derived from the fact that the mouse requires a special bus interface board that occupies a slot in your computer and communicates with the device driver across the main motherboard bus. Although the use of a bus mouse is transparent to the user (there is no operational difference between a bus mouse and other types of mice), many people view a bus mouse as less desirable than other types because it occupies a slot that may be needed for other peripherals.

Another drawback to the bus mouse is that it is electrically incompatible with the other types of mice, and because it is not very popular, a bus mouse can be hard to find in a pinch. Also, the bus adapters are typically available for ISA slots only, and because they are always 8-bit cards, you are limited in the choice of non-conflicting hardware interrupts. A bus mouse also can be dangerous because it uses a mini-DIN connector just like the (PS/2) motherboard-type mouse, although they are totally incompatible. I have even heard of people damaging motherboards by plugging a bus mouse into a motherboard mouse connector.

Bus mouse adapter cards usually have a selectable interrupt and I/O port address setting, but the IRQ selection is limited to 8-bit interrupts only. This usually means that you must choose IRQ 5 in most systems that already have two serial ports because all the other 8-bit interrupts will be used. If you also are using another 8-bit only card that needs an interrupt, such as some of the sound cards, then you will not be able to run both devices in the same system without conflicts. All in all, I do not recommend bus mice and think they should be avoided.

One thing to note is that Microsoft sometimes calls a bus mouse an *Inport* mouse, which is its proprietary name for a bus mouse connection.

Mouse Troubleshooting

If you are experiencing problems with your mouse, you need only look in two general places—hardware or software. Because mice are basically very simple devices, looking at the hardware takes very little time. Detecting and correcting software problems can take a bit longer, however.

Hardware Problems. Several type of hardware problems can crop up when you are using a mouse. The most common is a dirty mouse, which is solved by doing some "mouse cleaning." The other relates to interrupt conflicts and is more difficult to solve.

Cleaning Your Mouse. If you notice that the mouse pointer moves across the screen in a jerky fashion, it may be time to clean your mouse. This jerkiness is caused when dirt and dust get trapped around the mouse's ball and roller assembly, thereby restricting its free movement.

From a hardware perspective, the mouse is a very simple device, and cleaning it also is very simple. The first step is to turn the mouse housing over so that you can see the ball on the bottom. Notice that surrounding the ball is an access panel that you can open. There may even be some instructions that indicate how the panel is to be opened. (Some off-brand mice may require you to remove some screws to get at the roller ball.) Remove the panel and you can see more of the roller ball, as well as the socket in which it rests.

If you turn the mouse back over, the rubber roller ball should fall out into your hand. Take a look at the ball. It may be gray or black, but it should have no visible dirt or other contamination. If it does, wash it in soapy water or a mild solvent such as contact cleaner solution or alcohol and dry it off.

Now take a look at the socket in which the roller ball normally rests. You will see two or three small wheels or bars against which the ball normally rolls. If you see dust or dirt on or around these wheels or bars, you need to clean them. The best way is to use a compressed air duster can to blow out any dust or dirt. You also can use some electrical contact cleaner to clean the rollers. Remember, any remaining dirt or dust impedes the movement of the roller ball and means that the mouse won't work as it should.

Put the mouse back together by inserting the roller ball into the socket and then securely attaching the cover panel. The mouse should look just like it did before you removed the panel (except that it may be noticeably cleaner).

Interrupt Conflicts. Interrupts are internal signals used by your computer to indicate when something needs to happen. With a mouse, an interrupt is used whenever the mouse has information to send to the mouse driver. If a conflict occurs and the same interrupt used by the mouse is used by a different device, then the mouse will not work properly, or it may not work at all.

Interrupt conflicts do not normally occur if your system uses a mouse port, but they can occur with the other types of mouse interfaces. If you are using a serial interface, interrupt conflicts typically occur if you add a third and fourth serial port. This is because in ISA bus systems, odd-numbered serial ports (1 and 3) are often improperly configured to use the same interrupts, as are the even-numbered ports (2 and 4). Thus, if your mouse is connected to COM2: and an internal modem uses COM4:, then they both may use the same interrupt, and you cannot use them at the same time. You may be able to use the mouse and modem at the same time by moving the mouse (or the modem) to a different serial port. For instance, if your mouse uses COM1: and the modem still uses COM4:, then you can use them both at once because odd and even ports use different interrupts.

The best way around these interrupt conflicts is to simply make sure that no two devices use the same interrupt. Serial port adapters are available for adding COM3: and COM4: serial ports that do not share the interrupts used by COM1: and COM2:. These boards allow the new COM ports to use other available interrupts, such as IRQ 10, 11, 12, and 15. I never recommend configuring an ISA system with shared interrupts; it is a sure way to run into problems later.

If you are using a bus interface and you suspect an interrupt problem, you can use a program such as Microsoft's MSD (MicroSoft Diagnostics) to help you identify what interrupt the mouse is set to. You get MSD free with Windows 3 or higher as well as MS-DOS 6.0 or higher. If you use OS/2 and/ or IBM DOS, you can still get MSD for free by downloading it from the Microsoft BBS (see Appendix B). Beware that programs such as MSD that attempt to identify IRQ usage are not always 100 percent accurate and usually require that the device driver for the particular device be loaded to work at all. After the IRQ is identified, you may need to change the IRQ setting of the bus mouse adapter or one or more other devices in your system so that everything works together properly.

If your driver refuses to recognize the mouse at all, regardless of what type it is, then try using a different mouse that you know works. Replacing a defective mouse with a known good one may be the only way to identify if the problem is indeed caused by a bad mouse.

I have had problems in which a bad mouse caused the system to lock right as the driver loaded, or even when diagnostics such as MSD attempted to access the mouse. You can easily ferret out this type of problem by loading MSD with the /I option, which causes MSD to bypass its initial hardware detection. Then run each of the tests separately, including the mouse test, to see if the system locks. If the system locks during the mouse test, you have found a problem with either the mouse or the mouse port. Try replacing the mouse to see if that helps. If not, you may need to replace the serial port or bus mouse adapter. If a motherboard-based mouse port goes bad, you either can replace the entire motherboard (usually expensive!) or just disable the motherboard mouse port via jumpers or the system setup program and install a serial or bus mouse instead. This enables you to continue using the system without having to replace the motherboard.

Software Problems. Software problems can be a little trickier than hardware problems. Software problems generally manifest themselves as the mouse "just not working."

In such instances, you need to check the driver and your software applications before assuming that the mouse itself has gone bad.

Driver Software. To function properly, the mouse requires the installation of a device driver. I normally recommend using the drivers built into the Windows or OS/2 operating environments. This means that no additional external driver is necessary. The only reason for loading an external driver (via CONFIG.SYS) is if you want the mouse to work with DOS applications.

If you need the mouse to work under standard DOS, in other words outside of Windows or OS/2, you must load a device driver either through your CONFIG.SYS file or your AUTOEXEC.BAT file. This driver, if loaded in the CONFIG.SYS file, is typically called MOUSE.SYS. The version that loads in the AUTOEXEC.BAT file is called MOUSE.COM. (It is possible that your mouse drivers have different names, depending on who manufactured your mouse.) Again, remember that if you only use a mouse under Windows or OS/2, no external drivers are required because the mouse driver is built-in.

The first step, then, is to make sure that the proper command is in your CONFIG.SYS or AUTOEXEC.BAT file to load the driver. If it is not, add the proper line, according to the information supplied with your mouse. For instance, the proper command to load the mouse driver through the CONFIG.SYS file for a Microsoft mouse is as follows:

```
DEVICE=\DOS\MOUSE.SYS
```

The actual working or syntax of the command may vary, depending on whether you are loading the device into upper memory and where the device driver is located on your disk.

One of the biggest problems with the separate mouse driver is getting it loaded into an Upper Memory Block (UMB). The older drivers—9.0 and earlier—require a very large block of 40K to 56K UMB to load into, and upon loading they shrink down to less than 20K. Even though they only take 20K or less after loading, you still need a very large area to get them "in the door."

The best tip I can give you with respect to these separate drivers is to use the newest 9.01 or higher drivers from Microsoft. This new driver works with any type of Microsoft-compatible mouse, which basically means just about anything. These new drivers are included free with IBM DOS 6.3 and higher, but surprisingly not with MS-DOS. Microsoft requires that you pay for an upgrade to the newer versions of the mouse driver and does not include the more recent mouse driver with MS-DOS! If you use version 9.01 or later, it will require less memory than previous versions and will automatically load itself into high memory as well. One of the best features is that it first loads itself into low memory; shrinks down to less than 20K; and then moves into a UMB automatically! Not only that, but the driver seeks out the smallest UMB that can hold it, rather than try only the largest as would happen if you use the DEVICEHIGH, LOADHIGH, or LH commands to load the driver. Previous versions of the driver could not fit into an upper memory block unless that block was at least 40K to 56K or larger in size, and would certainly not do it automatically. The enhanced self-loading capability of the mouse driver 9.01 and higher can save much memory space and is very much worth having. I hope that this type of self-loading and self-optimizing technique is used in other device drivers; it will make memory management much easier than it currently is.

After placing the proper driver load command in your CONFIG.SYS or AUTOEXEC.BAT file, reboot the system with the mouse connected and observe that the driver loads properly. If the proper command is in place and the driver is not loading, watch your video screen as your system boots. At some point, you should see a message from the mouse driver indicating that it is loaded. If, instead, you see a message indicating that the loading was not done, you must determine why. For example, the driver may not be able to load because not enough memory is available. After you determine why it is not loading, you need to rectify the situation and make sure that the driver loads. Again the new 9.01 or higher driver versions help greatly with memory problems.

It also is possible that some software requires a certain mouse device driver. For example, Microsoft is now up to mouse driver version numbers above 9.0. If you are using an older mouse driver and your application software requires a newer mouse driver version, the mouse will not work properly (or at all). In such cases, contact your vendor directly and request a mouse driver update. Often you can get these through the vendor's BBS or on CompuServe; however, Microsoft wants you to pay for its new drivers and does not make them available on its BBS. One shortcut around Microsoft is to use IBM DOS 6.3 or higher, which includes the new Microsoft driver for free.

Application Software. If your mouse does not work with a specific piece of application software, check the setup information or configuration section of the program. Make sure that you indicated to the program (if necessary) that you are using a mouse. If it still doesn't work and the mouse works with other software you are using, contact the technical support department of the application software company.

Trackpoint II

In April 1992, I attended the spring Comdex computer show in Chicago, and at one of the IBM booths was an enthusiastic gentleman with what looked like some homemade keyboards. These keyboards had a small rubber tipped "stick" that protruded from between the G, H, and B keys. I was invited to play with one of the keyboards, which was connected to a demonstration system. What I found was amazing! By pressing on the stick with my thumb or index finger, I could move the mouse pointer on the screen. The stick itself did not move and is not a joystick. Instead it had a silicone rubber cap that contained pressure transducers that measured the amount of force my finger or thumb was applying and the direction of the force and moved the mouse pointer accordingly. The harder I pressed, the faster the pointer moved. I could move the pointer in any direction smoothly, by slightly changing the direction of push. The silicone rubber gripped my thumb even though I had been sweating from dashing about the show. After playing around with it for just a few minutes, the movements became automatic—almost as if I could just "think" about where I wanted the pointer to go, and it would go there. After reflecting on this for a minute, it really hit me. This had to be the most revolutionary pointing device since the mouse itself!

This device occupies no space on a desk; does not have to be adjusted for left-handed or right-handed use; has no moving parts to fail or become dirty; and most important, does not require you to move your hands from the home row to use! This is an absolute boon for anybody who touch types.

The gentleman at the booth turned out to be Ted Selker, one of the actual inventors of the device. He and Joseph Rutledge together created this amazing integrated pointing device at the IBM T.J. Watson Research Center. When I asked him when such keyboards would become available, he could not answer. At the time, there were no plans for production, and he was only trying to test user reaction to the device. However, just over six months later on October 20, 1992, IBM announced the Thinkpad 700, which included the Trackpoint II integrated pointing device.

In final production form, the Trackpoint II consisted of a small, red silicone, rubber knob nestled between the G, H, and B keys on the keyboard. Two buttons are placed below the space bar to emulate the LH and RH mouse buttons for making selections. These buttons also can be easily reached without taking your hand off the keyboard. Research done by the inventors found that the act of removing your hand from the keyboard, reaching for a mouse, and replacing the hand on the keyboard takes approximately 1.35 seconds. Almost all this time can be saved each time the track point is used to either move the pointer or make a selection (click or double-click). The combination of the buttons and the positioning knob also allow drag and drop functions to be performed easily as well.

By the way, the reason the device was called Trackpoint II is that IBM had previously been selling a convertible mouse/trackball device called the Trackpoint. No relationship exists between the Trackpoint, which has since been discontinued, and the Trackpoint II integrated device.

Another feature of the Trackpoint II is that a mouse also can be connected to the system to allow for dual pointer use. In this case, a single mouse pointer would still be on the screen; however, both the Trackpoint II and the simultaneously connected mouse could move the pointer. This allows not only the use of both devices by a single person, but in fact two people can use both the Trackpoint II and the mouse simultaneously to move the pointer on the screen! The first pointing device that moves takes precedence and retains control over the mouse pointer on the screen until after completing a movement action. The second pointing device is automatically locked out until the primary device is stationary. This allows both devices to be used, but prevents each one from interfering with the other.

The Trackpoint II is obviously an ideal pointing device for a laptop system where lugging around an external mouse or trackball can be a pain. The trackballs and mini-trackballs built into some laptop keyboards are also very difficult to use and usually require removing your hands from the home row. Mouse and trackball devices are notorious for becoming “sticky” as the ball picks up dirt that affects the internal roller motion. This is especially aggravated with the smaller mini-trackball devices. But the benefits of the Trackpoint II are not limited to laptop systems. IBM/Lexmark now manufactures and sells desktop enhanced keyboards with the Trackpoint II device built-in. These new keyboards are also optional when purchasing a new PS/2 system. One drawback is that the Trackpoint II device in these keyboards initially worked only with systems that used a PS/2- or motherboard-type mouse connector. However, now the Trackpoint II enhanced keyboard is available in two versions. One interfaces via a motherboard mouse port (mini-DIN connector) and has a mini-DIN type keyboard connector as well. The other has a standard DIN keyboard connector and a serial connector for the pointing function. I have listed the part numbers for the IBM enhanced keyboard with the Trackpoint II in the section, “Replacement Keyboards,” earlier in this chapter. You also can purchase these keyboards direct from Lexmark.

The Trackpoint II probably stands as the most important and revolutionary new pointing device since the original invention of the mouse in 1964 by Douglas Englebart at the Stanford Research Institute (SRI). As IBM licenses this technology to other manufacturers, you probably will see this device show up in many different systems. It is already available built into keyboards, which can upgrade many existing systems, and companies such as Toshiba are already using this IBM-developed technology in their own systems.

Game Adapter (Joystick) Interface

The game control or joystick adapter is a special input device that allows up to four paddles or two joysticks to be attached to a PC system. This function can be found on a dedicated ISA or MCA bus adapter card, or can be combined with other functions in a multifunction card. The game connector on the card is a female 15-pin D-Shell type socket (see fig. 9.13).

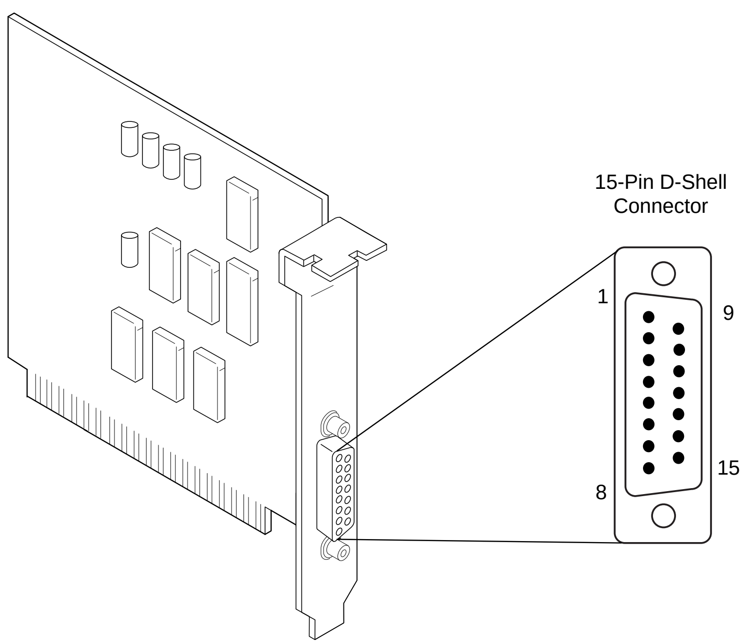

Fig. 9.13
Typical game adapter and 15-pin connector.

The game adapter can recognize up to four switches (called buttons) and four resistive inputs. Each paddle normally has one button and one knob controlling a variable resistor, whereas a joystick normally has two buttons and a central stick that controls two variable resistors. In a joystick, the variable resistors are both tied to the central stick; one indicates the relative vertical position (or x-coordinate), and the other indicates the relative horizontal position (or y-coordinate) of the stick.

Resistor inputs are variable from 0 to 100K ohms. The adapter converts the resistive value to a digital pulse with a duration proportional to the resistive load. Software can time these pulses to determine the relative resistance value. The game adapter does not use much in the way of system resources. The card does not use an IRQ, DMA channel, or memory, and requires only a single I/O address (port) 201h. The adapter is controlled by reading and writing data to and from port 201h.

Table 9.12 shows the interface connector pinout specification for a PC-compatible game adapter.

Table 9.12 PC-Compatible Game Adapter Connector

Pin	Signal	Function	I/O
1	+5v	Paddle 1, Joystick A	Out
2	Button 4	Paddle 1 button, Joystick A button #1	In
3	Position 0	Paddle 1 position, Joystick A x-coordinate	In
4	Ground	—	—
5	Ground	—	—
6	Position 1	Paddle 2 position, Joystick A y-coordinate	In
7	Button 5	Paddle 2 button, Joystick A button #2	In
8	+5v	Paddle 2	Out
9	+5v	Paddle 3 and Joystick B	Out
10	Button 6	Paddle 3 button, Joystick B button #1	In
11	Position 2	Paddle 3 position, Joystick B x-coordinate	In
12	Ground	—	—
13	Position 3	Paddle 4 position, Joystick B y-coordinate	In
14	Button 7	Paddle 4 button, Joystick B button #2	In
15	+5v	Paddle 4	Out

Because this adapter actually reads resistance and easily can be manipulated with standard programming languages, the game adapter serves as a poor man's data acquisition board or real-time interface card. With it, you can hook up to four sensors and four switches and easily read the data in the PC.

Game adapters are available for ISA and MCA bus systems from a number of vendors. Consult the vendor list for some companies that may offer these types of adapters. Generally, the best place to look is one of the larger mail order system and peripheral vendors.

Summary

This chapter has examined all manner of standard input devices. Keyboards and pointing devices (usually mice) were covered in detail, as was a less known input device, the game interface. The next chapter covers video display hardware.