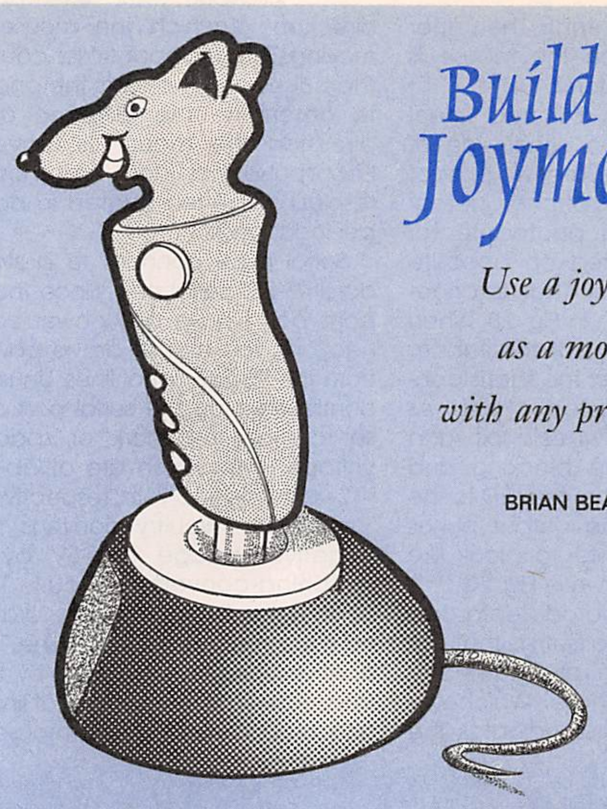
Build the Joymouse

Use a joystick as a mouse with any program.

BRIAN BEARD



A computer mouse and a joystick are both pointing devices. They both have button inputs. However, a joystick is an "absolute" position device. Its control stick can deflect only so far from the center position. A mouse, on the other hand, is a "relative" device—it can be lifted off the table and moved while the cursor on the computer screen stays where it is. Unless you have an Apple Macintosh, you can't use one in place of the other. Wouldn't it be nice if you could use a joystick as a mouse on an IBM or IBM-compatible computer?

The Joymouse described here is just the device to fill that need. The circuit board is small enough to be mounted inside a joystick. With the flick of a switch, the Joymouse can change from joystick to mouse and back.

What's in a Joystick? An analog joystick is a very simple device. The mechanical arrangement of a typical joystick is shown in Fig. 1A. Electrically, it has two 100,000-ohm potentiometers and two normally-open pushbuttons. The shafts of the potentiometers are mechanically connected to the stick by gimbals that isolate the left-to-right (X-axis) movement from the forward-to-backward (Y-axis) movement. The stick is usually spring loaded so that it returns to the center position whenever it is released.

Joysticks for IBM and IBM-compatible computers are designed to work with the specification for the original IBM Game-Control Adapter. That adapter has eight inputs—four digital inputs and four resistive inputs. Two inputs of each type are grouped as a pair of joysticks, called Joystick A and Joystick B. The wiring diagram and connector-pinout description for hooking up two joysticks to an IBM Game Adapter is shown in Fig. 1B.

Joysticks have a male DB15 connector that connects to the female DB15 game-port connector on the PC. If two joysticks are to be connected to the game port, a Y-shaped adapter is used to split the signals for each joystick into individual DB15 connectors that are wired as Joystick A. Joysticks that have additional features such as a throttle control, rudder control, or four buttons use the inputs of both Joystick A and Joystick B at the same time for the additional controls. Obviously, since that type of joystick appears to the game port as two joysticks, only one device can be hooked up at a time.

Examinning the wire diagram, we see that one side of each potentiometer is tied to a 5-volt power source with the wiper being connected to one of the resistive inputs. The resistive-input circuit has a capacitor connected to ground. That capacitor and the potentiometer form an RC-timer circuit. Moving the joystick changes the charging time of the capacitor. The joystick driver-software periodically grounds the resistive input, and then times how long it takes for the voltage on the capacitor to rise to a certain level. That time lapse indicates the position of the joystick handle. The circuit is very similar to an LM555-based "one-shot" timer circuit. In fact, some earlier game-port cards used that exact chip for the input circuit.

Most joysticks have trim tabs for each axis that allow the center-position resistance to be set by rotating the body of the potentiometer. The buttons on the joystick are connected to ground and the button inputs. The button inputs have resistors that apply a 5-volt signal to the input on the game card when the buttons are not pressed. Pressing a button grounds that particular input. The trigger on the joystick is normally wired as button 1. Some joysticks have an auto-fire circuit connected to the trigger. When the trigger is held down, an oscillator circuit pulses the button-1 line faster than a person possibly could.

What's in a Mouse? Mice, because of the way they are connected to computers, need to have some intelligence built into them, instead of having some interface hardware built into the computer. A single-chip microcontroller (MCU) takes the information from the mouse buttons and the x- and y-axis inputs and sends it to the computer.

A typical mouse mechanism is shown in Fig. 2A. The mechanical portion of the mouse consists of a rubber-coated steel ball and a pair of roller shafts. The two shafts are set at 90° angles to each other so that one rotates when the mouse is moved left and right. The other shaft rotates when the mouse is moved back and forth.

Each shaft has a slotted wheel attached to it. On either side of each slotted wheel are two pairs of LEDs and two pairs of photodetector. The LED and photodetector pairs are positioned on opposite sides of the wheel. That arrangement can be seen in Fig. 2B. When the mouse is moved, the rubber ball presses against the shafts connected to the slotted wheels. As the ball rolls, the wheels spin and the slots alternate blocking and passing the light from the LED to the photodetector. The output pulses from the photodetectors look like the pulse train shown in Fig. 2C.

The two LED-photodetector sets on each axis are offset so that the pulses from each photodetector are 90° out of phase. Whichever output switches first indicates the direction in which the mouse is moving. The microcontroller counts the pulses and monitors the phase to determine the distance and direction the mouse is moved. Each pulse is a "dot," and mouse resolution is often quoted in dots-per-inch (dpi).

Serial mice connect to a standard RS-232 serial port. Since those ports don't have any power connections, the mouse draws power from the RS-232 signal lines. Unused control lines on the serial port are set to either the "mark" or "space" voltage level, which are at about -12 volts and +12 volts, respectively. The mouse circuitry converts the negative voltage to +5V by a switching-converter circuit. The negative voltage is usually drawn from the TD (transmit data) line. The positive voltage is supplied by the RTS (request to send) line. That line is also used to detect if a mouse is attached to a serial port. The software toggles the RTS line from off to on and then looks for a response from the mouse on the RD (receive data) line. If a mouse is connected and working, it sends the letter M in ASCII code. If, after a few milliseconds, no "M" is received, the driver software reports that there is no mouse connected and resets RTS to the off state.

How a Mouse Communicates. Since the mouse is an input device, it doesn't make sense to send messages when there is no information to send. Messages are only sent when the mouse is moved, a button is pressed, or a button is released. The messages sent by the mouse are referred to as *packets*. There are several formats for mouse-data packets, but the Microsoft Serial Mouse format has become the de facto industry standard for

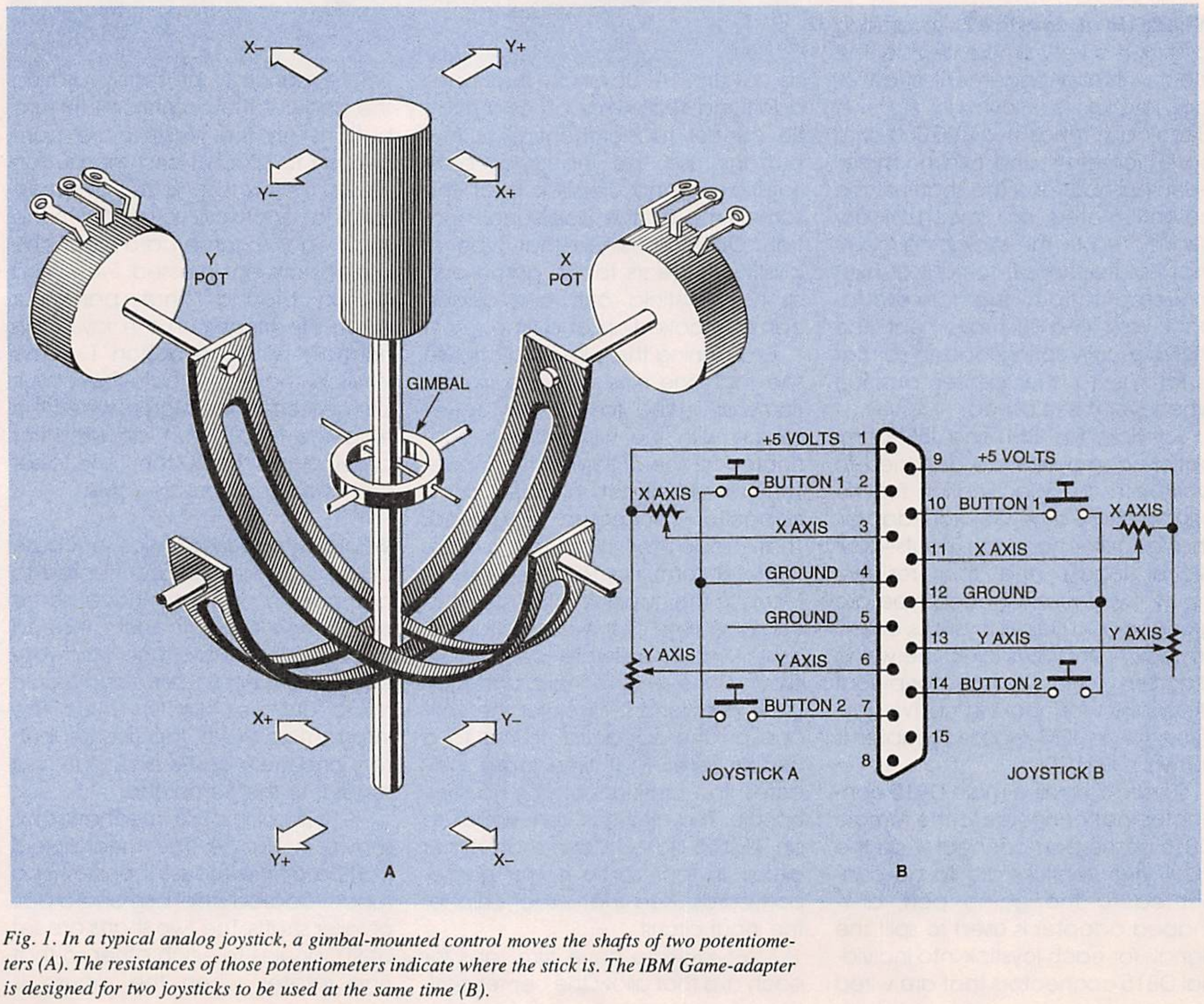


Fig. 1. In a typical analog joystick, a gimbal-mounted control moves the shafts of two potentiometers (A). The resistances of those potentiometers indicate where the stick is. The IBM Game-adapter is designed for two joysticks to be used at the same time (B).

TABLE 1
MICROSOFT MOUSE PACKET FORMAT

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

LB is the state of the left button, 1 = pressed, 0 = released
RB is the state of the right button, 1 = pressed, 0 = released
X0–X7 is movement of the mouse in the X direction
Y0–Y7 is movement of the mouse in the Y direction

serial mice. The Microsoft format is shown in Table 1.

The Microsoft mouse format only allows for two buttons—if a three-button mouse is working in a Microsoft-compatible mode, the middle button is ignored. All moves are sent in an 8-bit two's-complement binary-numbers format. That arrangement requires seven bits plus a sign bit. Since the packet sends seven-bit bytes with the highest bit unused, the movement values are split between two bytes. The data for the X-axis movement is stored in the second byte, and the Y movement information is stored in the third byte. The first byte combines the extra bits from both the X and Y movement data with the state-of-the-mouse buttons. The mouse driver software in the computer separates out the X and Y movement data from the different bytes in the packet.

The packets are sent at 1200 baud with one stop bit and no parity. Although the Microsoft format only requires 7-bit data, most mice actually send 8-bit bytes with the most significant bit set to one. Apparently, the mouse-driver software ignores D7.

How the Joymouse Works. The Joymouse circuit, shown in Fig. 3, converts the resistive and button inputs from an analog joystick into a serial data stream that emulates a

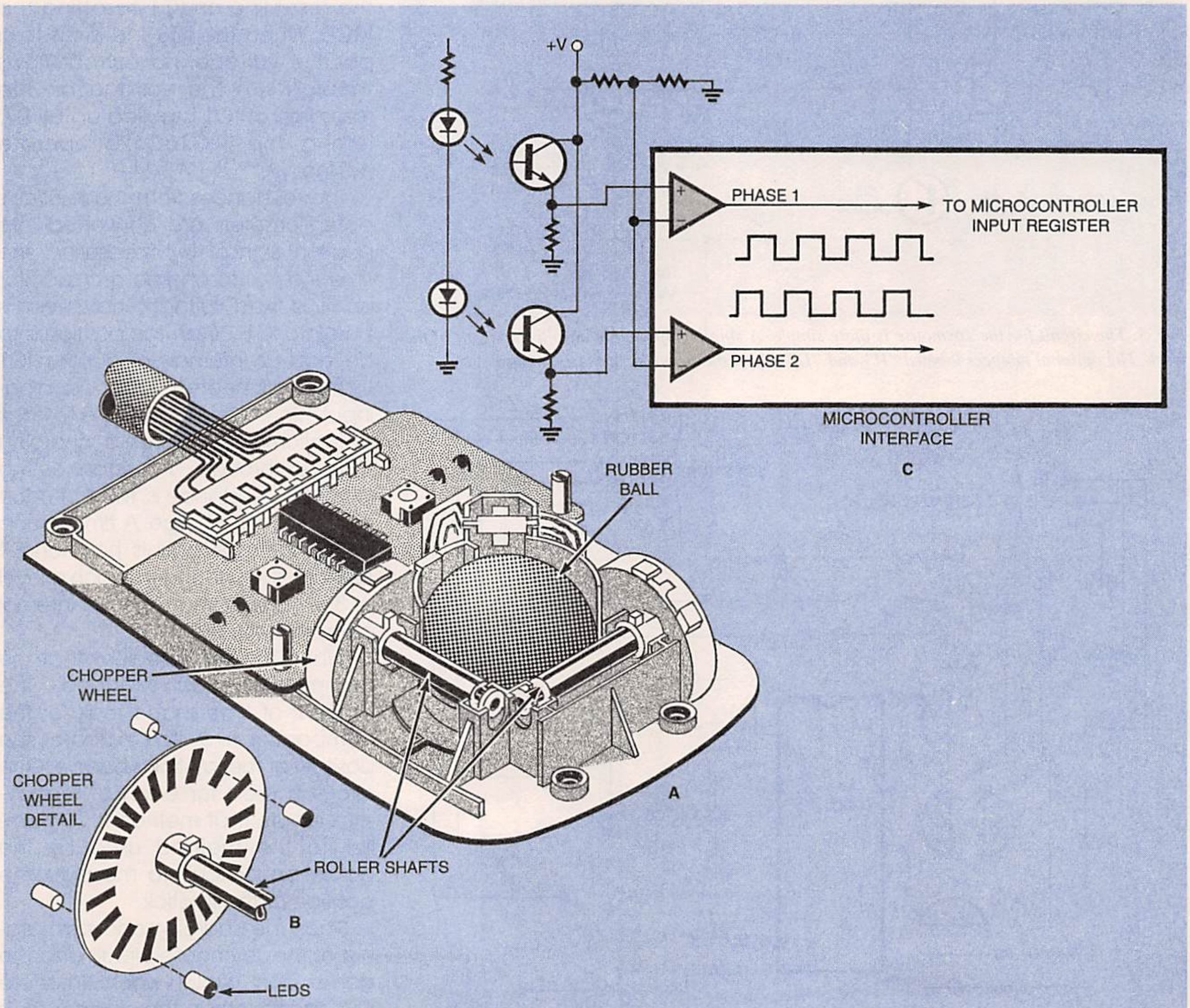


Fig. 2. In a typical mouse, a rubber ball spins the shafts of two slotted wheels (A). The slotted wheels block the light between a pair of LEDs and a pair of photodetectors (B). The photodetectors are set so that a pair of out-of-phase signals will indicate which direction the shaft is spinning (C).

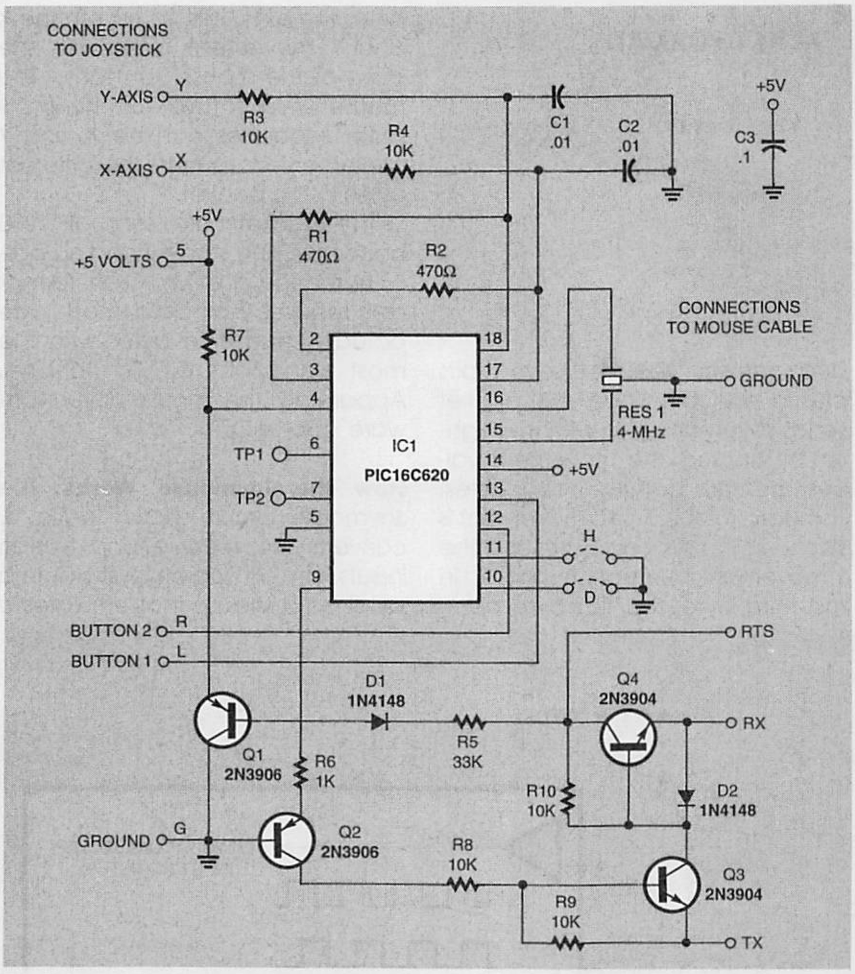


Fig. 3. The circuit for the Joymouse is quite simple. A single-chip microcontroller does all of the work. The optional jumpers labeled "H" and "D" adjust the response of the Joymouse.

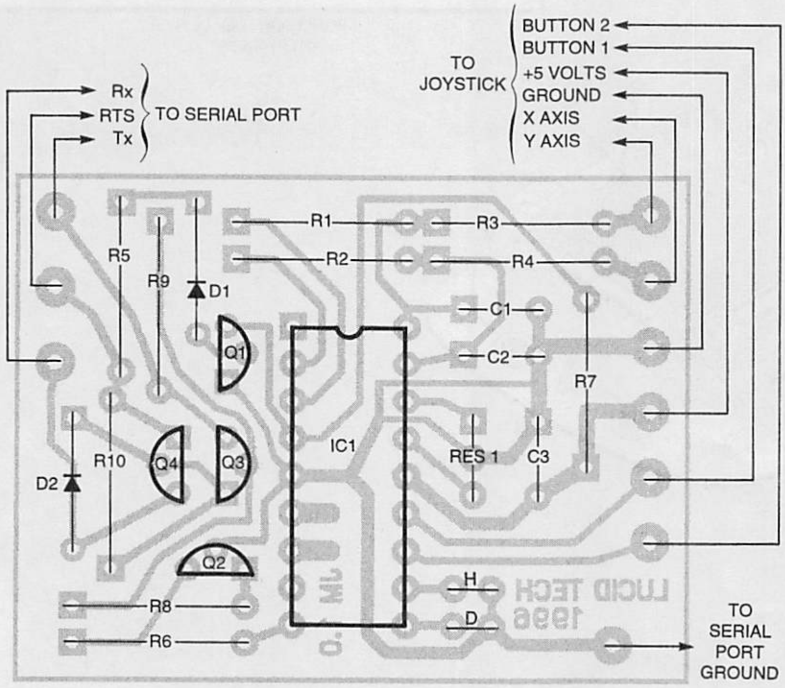


Fig. 4. If you are using either a PC board from a kit or the foil pattern shown in this article, use this parts-placement diagram to locate the components.

Microsoft serial mouse. Because the Joymouse is connected to both the serial port and the game port, the game port's 5-volt supply is used to power the Joymouse circuitry.

The heart of the Joymouse is IC1, a PIC16C620 microcontroller. That chip combines a unique set of features that make the single-chip design possible. Those features include 2 analog comparators, 13 digital input/output lines, a programmable voltage reference, a power-up reset timer, a "watchdog" reset timer, 512 bytes of EPROM, 80 bytes of RAM, and a high-performance RISC CPU.

The reset pin on IC1 is controlled by the RTS signal on the serial port. When the RTS line has a negative voltage, Q1 turns on. That grounds the reset line on IC1, resetting the MCU. When the RTS line is set to a positive voltage, no current flows through Q1. The voltage on the reset line of IC1 is pulled up by R7, letting the PIC16C620 operate normally.

The resistances from the joystick's potentiometers are converted into position signals by measuring the time it takes to charge a capacitor up to a fixed voltage. That circuit is built into IC1. When the position is to be read, an internal capacitor in IC1 that is connected to an analog-input pin is discharged. The joystick potentiometer supplies a charging voltage through a resistance. The lower the resistance is, the faster the capacitor will charge. A timing loop in the software counts how long it takes for the voltage to reach a level of 3.59 volts. At that level, an internal comparator switches.

When the software sees that the comparator has switched, the amount of time that it took for the comparator to switch indicates the position of the joystick's handle. One circuit is used for each of the joystick inputs. That method is very similar to the method used by the game port circuit to measure the position of the joystick.

Since the charging and discharging of the Joymouse circuit and the game port circuit would interfere with each other, the Joymouse is designed to disconnect the joystick from the game port when it is being used as a mouse. Those signals are

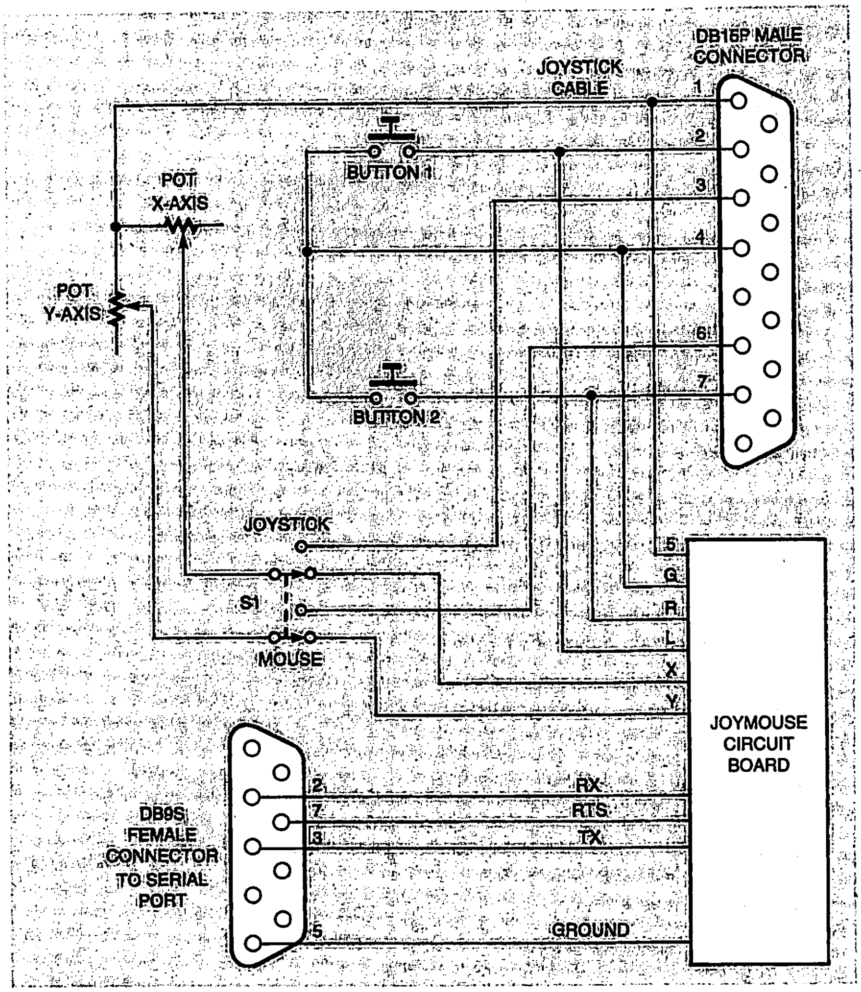


Fig. 5. This wiring diagram shows how to connect the Joymouse if you want to install the board directly into the base of a joystick. The four-wire serial cable should be the same length as the joystick's game-port cable.

PARTS LIST FOR THE JOYMOUSE

SEMICONDUCTORS

IC1—PIC16C620P
Q1, Q2—2N3906, PNP transistor
Q3, Q4—2N3904, NPN transistor
D1, D2—1N4148 silicon diode

RESISTORS

(All resistors are ¼-watt, 5% units.)
R1, R2—470-ohm
R3, R4, R7-R10—10,000-ohm
R5—33,000-ohm
R6—1,000-ohm

CAPACITORS

C1, C2—0.01-µF, Mylar
C3—0.1-µF, ceramic-disk

ADDITIONAL PARTS AND MATERIALS

RES1—4-MHz ceramic resonator, ECS ZTT-4.00MG (Digi-Key X902-ND or similar)
S1—double-pole, double-throw switch
PC board, 18-pin IC socket, 4-conductor cable, female DB9 connector, plastic DB9 connector cover, DB15 connectors (see text), jumpers, etc.

Note: The following is available from Lucid Technologies, 12907 Crookston Lane, Unit 22, Rockville, MD 20851, Web: http://www.cs.com/lucid: Complete kit of all parts, PC board, complete printed schematics, and technical manual on disk (specify 5¼-inch or 3½-inch), $25.00; partial kit consisting of programmed IC1, complete printed schematics, and technical manual on disk (specify 5¼-inch or 3½-inch), $13.00. Please add $3.00 for shipping and handling. Maryland residents add appropriate sales tax.

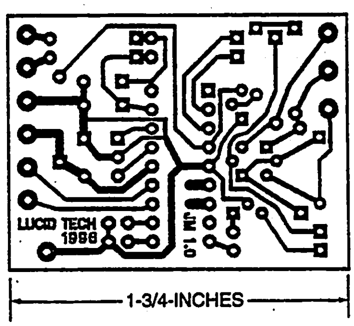
switched by a double-pole double-throw switch. The joystick pushbuttons, on the other hand, are connected to resistors on the game-port card, so there is no problem connecting the Joymouse directly to those inputs.

The RS-232 cable that connects the Joymouse to the PC's serial port has four lines. The RTS and TD lines are used for the positive and negative voltages that are needed by the Joymouse. A reference voltage level is provided by SG (signal ground), and RD is the serial data that is sent by the Joymouse to the PC.

The logic for the serial data transmitter is done by the Joymouse software. The serial data stream from IC1 uses an external circuit to shift the voltage levels to the RS-232 voltage standards. When the serial data level is high, Q2 turns on. That, in turn, switches Q3 on. The negative voltage on the collector of Q3 keeps Q4 turned off. As a result, the voltage level on RD is pulled to the negative RS-232 voltage through D2. When the serial data level is low, Q2 and Q3 are turned off. That leaves Q4 turned on, pulling RD to the positive RS-232 voltage.

Software. Any device with a microcontroller inside is really defined by its software, and the Joymouse is no different. Most of the program consists of subroutines—small blocks of instructions that are called from the main program loop. Some of the subroutines in the Joymouse program include measuring the joystick resistances, creating the data packet for the serial port, sending that data, and setting up various timing delays.

In a real mouse, if there are no pulses coming from the photodetectors, the mouse is not moving.



Here is the foil pattern for the Joymouse. A single-sided board is easy to etch and assemble.

However, the Joymouse always sees a resistance for the two joystick potentiometers. How does it know

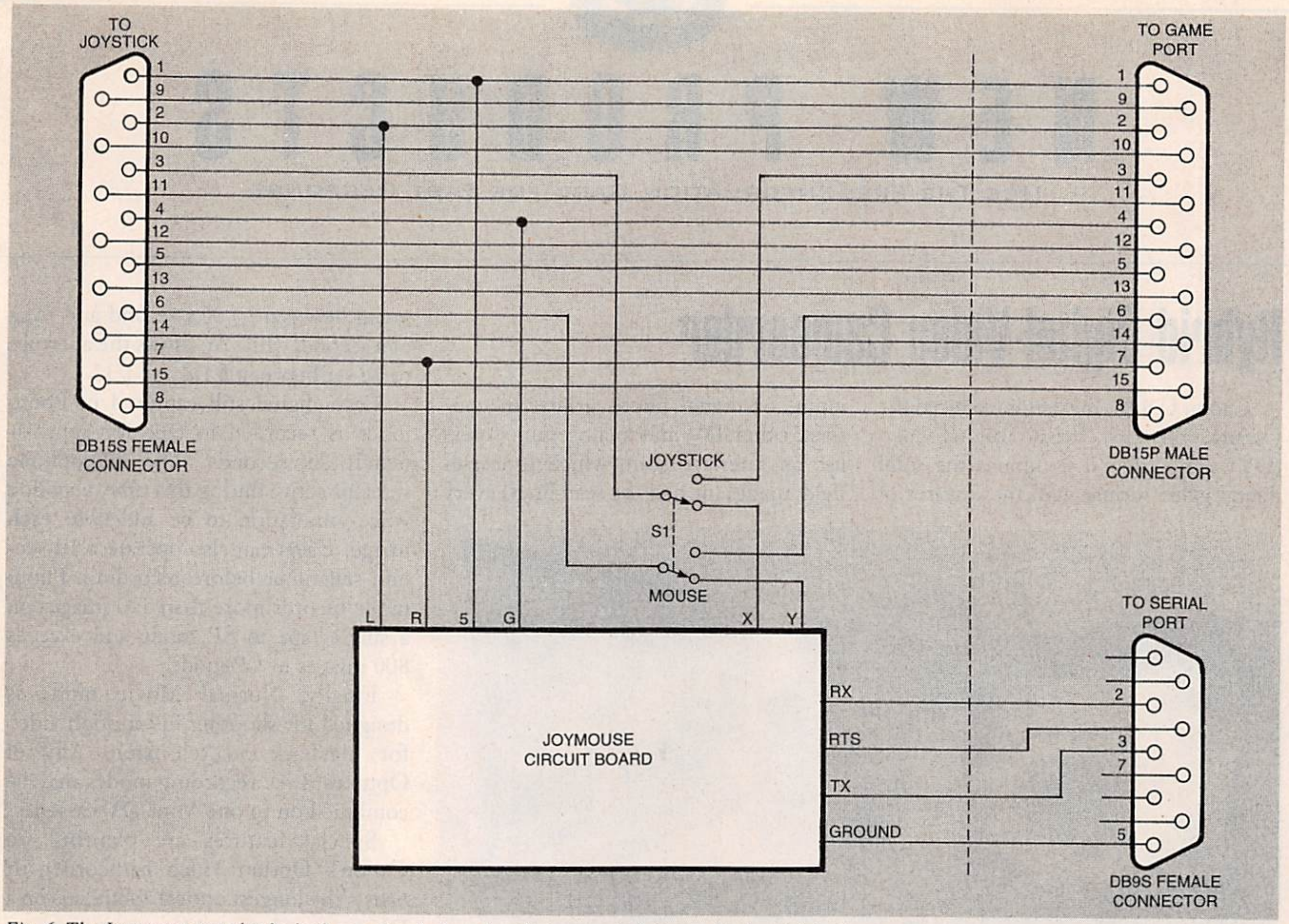


Fig. 6. The Joymouse can also be built as a stand-alone box that connects between the joystick and the computer. Switch S1 selects either joystick or mouse operation.

exactly what the center resistances are for each axis? The answer is that when RTS goes positive and the Joymouse MCU starts running its program, it assumes that the joystick is centered. The first joystick readings are used for the center position values. Since the resistances at the center might not be exactly 50,000 ohms due to component tolerances and trim tab settings, the Joymouse will keep checking the joystick readings until both values are between 38,000 ohms and 62,000 ohms.

When a joystick is released, its spring loading returns the stick to the center position. However, due to mechanical slop in the linkages, the resistances at the center are seldom identical from one reading to the next. For that reason, the software doesn't consider the center as just a pair of X and Y resistances, but as a range of values surrounding the center values measured after reset. That region is called the *deadband*. The Joymouse will only send movement data to the PC when one or both resistance values fall outside of the deadband. Moving the stick further from center is equivalent to moving a mouse faster.

The default deadband and ramp should work well with most joysticks, but they can be changed with the jumpers labeled "D" and "H" in Fig. 3. Shorting the "D" jumper will increase the width of the default deadband by 50% to compensate for unusually loose joysticks. Shorting the "H" jumper will decrease the slope of the ramp by 50% making the cursor move slower.

No special software is required on your PC to use the Joymouse. Any Microsoft mouse-compatible driver will work with the Joymouse.

Construction. Building the Joymouse is simple and straightforward. The PC board is a single-sided design that is easy to etch and drill if you want to build your own. Both a complete and partial kit is available from the source given in the Parts List.

Before you assemble the Joystick PC board, IC1 must be programmed with the Joystick software. A pre-programmed chip is available from the source given in the Parts List. If you wish to program your own chip, the software can be downloaded from the Gernsback FTP site (ftp://ftp.gernsback.com). The file name for the Joymouse software is joymouse.hex.

The parts-placement diagram for the Joymouse circuit board is shown in Fig. 4. Be sure to use a low-wattage soldering iron when assembling the board. If an iron with a high-wattage rating is used, the small pads might lift off the board if they are overheated.

It's a good idea to use pins and jumper blocks for the jumpers labeled "D" and "H." That way, the response of the Joymouse can easily be changed if you find that the default response is not to your liking. For now, do not install any jumpers in

(Continued on page 72)

JOYMOUSE

(continued from page 59)

those locations. When all of the components are installed, examine your work for any bad solder joints, solder bridges, or components that have accidentally been installed backwards.

You must decide how you want to mount the completed circuit. Two choices are presented here—a separate box that connects between the joystick and the computer or installing the circuit board inside the base of a joystick.

The connection diagram in Fig. 5 shows how to wire the Joymouse board if it is going to be installed inside a joystick. Since there are many different brands of joysticks compatible with the IBM PC, specific installation instructions are not possible. You must be able to find the correct signal wires that need to be spliced into the Joymouse. The main trigger button (button 1) will be the left mouse button. If there is an autofire circuit connected to the buttons, either remove the autofire circuit or connect the Joymouse button inputs to the input of the autofire circuit. The cable that connects the Joymouse to the computer's serial port should be the same length as the joystick's existing cable.

It is also important to include S1 if you want to be able to switch the Joymouse between joystick and mouse operation. One final item to remember is that if you decide to modify your joystick, you will void any existing warranty from the joystick manufacturer.

If you do not want to modify a joystick, the wiring diagram in Fig. 6 shows how to build the Joymouse into a stand-alone box that would connect between your joystick and the computer's serial and game ports. The female DB15S connector that the joystick will plug into can be mounted directly on the box. The game port and serial port connectors are best mounted on short cables. The rest of the stand-alone construction is similar to the joystick-modification method.

Testing the Joymouse. Turn your computer off, attach the joystick cable to the game port, and connect the mouse cable to the serial port used for a mouse. Adjust the trim tabs on the joystick to their center position and make sure that S1 is set to mouse operation. Turn on the computer and boot to Windows. The joystick should be centered while Windows is starting. The mouse cursor should be controlled by the joystick. Move the cursor around for

(Continued on page 104)

JOYMOUSE

(continued from page 72)

a while to get used to the way the Joymouse works. The cursor will keep moving as long as the stick is displaced from center so you have to start moving the stick back to the center just before you reach your target.

Try adjusting the sensitivity of the mouse driver for cursor speeds that suit you. If you still want a slower cursor, add the "H" jumper on the Joymouse board. If the cursor slowly drifts while you have your hands off the joystick, you might need to increase the size of the deadband by adding the "D" jumper on the Joymouse board.

The two test points on the Joymouse are controlled by the Joymouse software subroutines. Test point TP1 goes high whenever the joystick resistances are being measured, and TP2 goes high whenever a mouse-data packet is being transmitted. In addition to showing that those functions are running, the test points can also be used to synchronize an oscilloscope. That will let you watch the comparator circuits and the RS-232 interface. Ω