8088 I/O Ports

Q *I recently started a project with an 8088 microprocessor as described in the "Drawing Board" column in your October 1995 issue. When I looked in the November issue, I saw that the Drawing Board column had been dropped and the rest of the project never appeared. I have a basic understanding of how to design the circuit but am not completely sure how to demultiplex the address and data lines. I would definitely like to know how to implement I/O ports and memory-mapped I/O. Any help you could give me would be greatly appreciated.—*
M. A., Shrewsbury, MA

A We regret very much that the promised columns weren't completed; some things are beyond a magazine's or an editor's control. Some related projects by the same author are in *The 8088 Project Book*, by Robert Grossblatt, published by TAB/McGraw-Hill, which you can order through any bookstore. Meanwhile, here's some information to get you started.

First, if you don't actually need 8088 software compatibility, you might consider switching to a CPU with simpler circuit requirements. The 8088 requires several external chips just to generate its clock signals and handle its input and output. By contrast, microcontrollers such as the 87C750 or COP8 put the whole computer on a single chip, including RAM, EPROM, and input-output ports. Their assembly languages are related but not identical to that of the 8088. For high-end systems, consider the Intel 80386EX, which is almost a whole PC on a chip.

If you already have an 8088 (almost) running, here are some circuits you can use. We'll start by telling you how to add ports to the bus of a PC; then we'll explain what to do if you're using a simpler 8088 circuit with different control signals.

The 8088 bus has address lines, data lines, and control lines. On the PC, the most important control lines are /IOW, /IOR, /MEMW, and /MEMR, for "I/O port write," "I/O port read," "memory write," and "memory read," respectively.

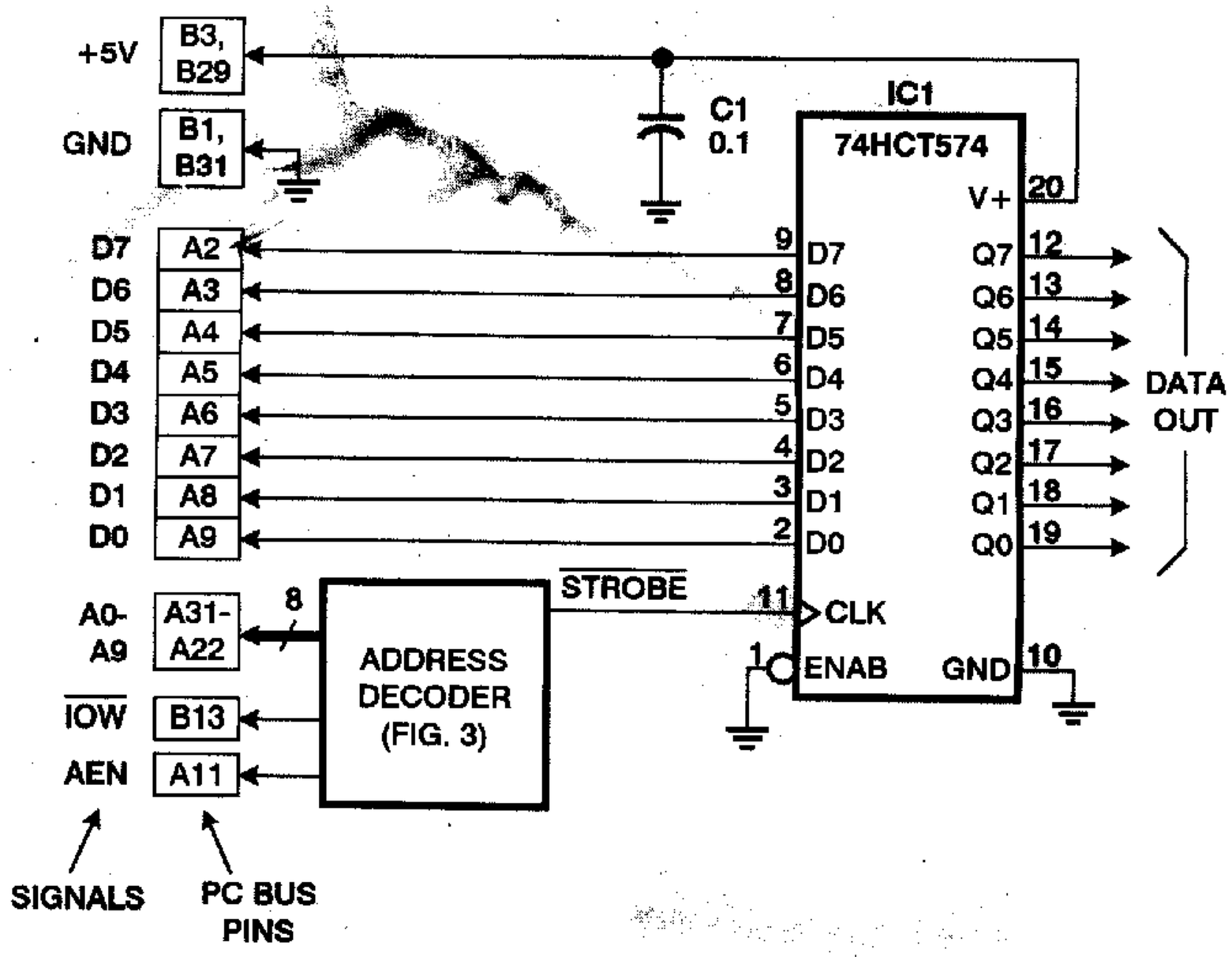

FIG. 1—HERE'S A SIMPLE 8-BIT OUTPUT PORT for a PC bus. Data sent to the appropriate hex address will appear on the output pins.

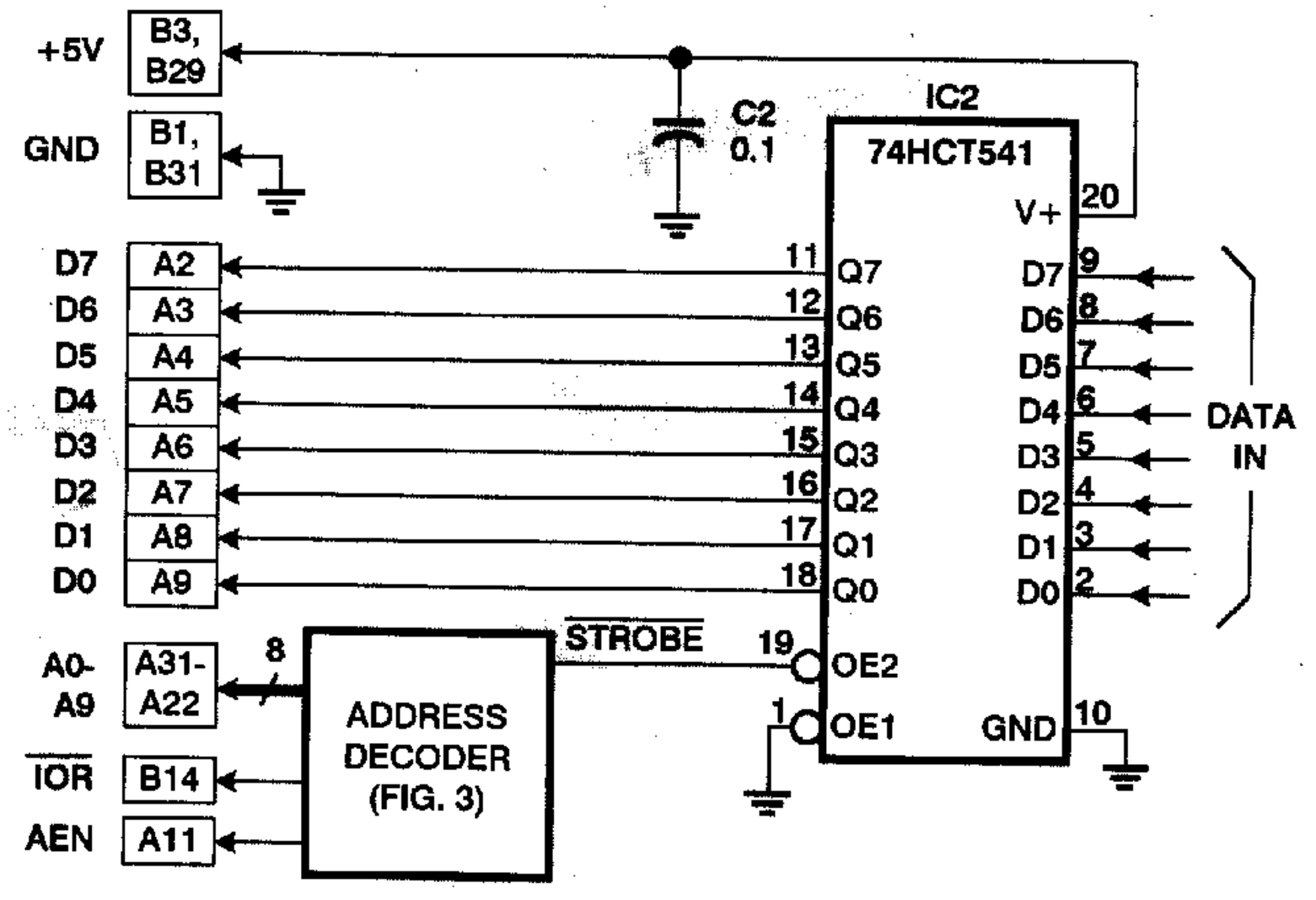

FIG. 2—THE INPUT PORT is nearly identical to the output port, but substitutes a 74HCT541 driver for the 74HCT574 of Fig. 1.

The slash, also written as a minus sign or a bar above the letters, indicates that these lines are "active low," *i.e.*, 0 volts signifies action and +5 volts means do nothing. Another control line, AEN ("address enable"), means "don't use the bus right now" and goes high if a device other than the CPU is using the bus to access memory.

To write to a port, the CPU puts the address on the address lines, the data on the data lines, and 0 volts on /IOW. To read from a port, it puts the address on the address lines, puts 0 volts on /IOR, and reads the signals that the port puts on the data lines. Memory access works the same way except that /MEMW and /MEMR are used instead of /IOR and /IOW.

Figures 1, 2, and 3 show how to add an I/O port to an ISA- or EISA-bus PC. You can build the circuit on a RadioShack 276-1598 printed-circuit board, which has an edge connector that plugs into a slot in a PC; you'll then have a handy board for interfacing the PC to analog-digital converters and other equipment.

The pins on the edge connector are numbered A1-A31 and B1-B31; don't confuse those numbers with the designations of address lines A0-A9 or the data lines D0-D7. The diagrams show bus pin numbers as well as signal designations.

Figure 1 shows how to build an 8-bit output port. We've assigned this one an address of hex 280, *i.e.*, binary 1010000000. The address decoder (which we'll get to in a minute) takes its output low if and only if 1010000000 is on the address bus, AEN=0, and /IOW=0. When that happens, data is latched from the data lines into the 74HCT574 output chip, which holds it until the next time you access the port. You can place data there by executing an OUT instruction to address hex 280 in assembly language or BASIC.

Figure 2 shows an input port. Here the 74HCT541 transceiver transmits data from its inputs to the bus only when the correct address is present, AEN=0, and /IOR=0. The data can be read with an IN instruction from the appropriate address. The input and output ports can have the same address because one is triggered by /IOR and the other by /IOW.

Figure 3 shows how the address is decoded. All the address decoder has to do is recognize a particular pattern of 12 bits (10 address bits plus AEN and /IOW or /IOR). One way to do that is to use a pair of 74HCT688 8-bit magnitude comparators. The 74HCT688 has two sets of 8-bit inputs, called "P" and "Q," and its output goes low when they are equal. So all you have to do is hard-wire the desired bit pattern to the "P" inputs and connect the address bits, AEN, and /IOW or /IOR to the "Q" inputs.

There are plenty of other ways to decode addresses. If you're building several ports at the same time, you might want to run the bottom 3 or 4 address bits to a '138 or '154 demultiplexer so you can decode a block of adjacent addresses with a single circuit. You may also want to decode the higher address lines, A15-A10, to distinguish a larger number of addresses.

Memory-mapped I/O works just like port I/O except that it is triggered by /MEMR or /MEMW rather than /IOR or /IOW. On the 8088, memory-mapped I/O is seldom used because it requires you to decode all 20 address

continued on page 61

Q & A

continued from page 9

lines, and because all available addresses are likely to be occupied by real memory. In fact, that's the main reason for distinguishing ports from memory addresses—it keeps them from conflicting.

If you're working with a bare 8088 rather than a PC, then instead of /IOR, /IOW, /MEMR, /MEMW, and AEN, you have just three control lines, /S0, /S1, and /S2. These are decoded as shown in Fig. 4. For example, instead of looking for AEN=1 and /IOR=0, an input port on this type of system looks for /S0=0, /S1=0, and /S2=1.

For more information about interfacing to 8088s and PC circuitry, see *The Art of Electronics*, by P. Horowitz and W. Hill (Cambridge University Press) and *The Personal Computer from the Inside Out*, by M. Sargent and R. L. Shoemaker (Addison-Wesley). Sargent and Shoemaker's earlier book, *The IBM PC from the Inside Out*, tells even more about the details of the 8088 but is now out of print; you may be able to find it in a library.

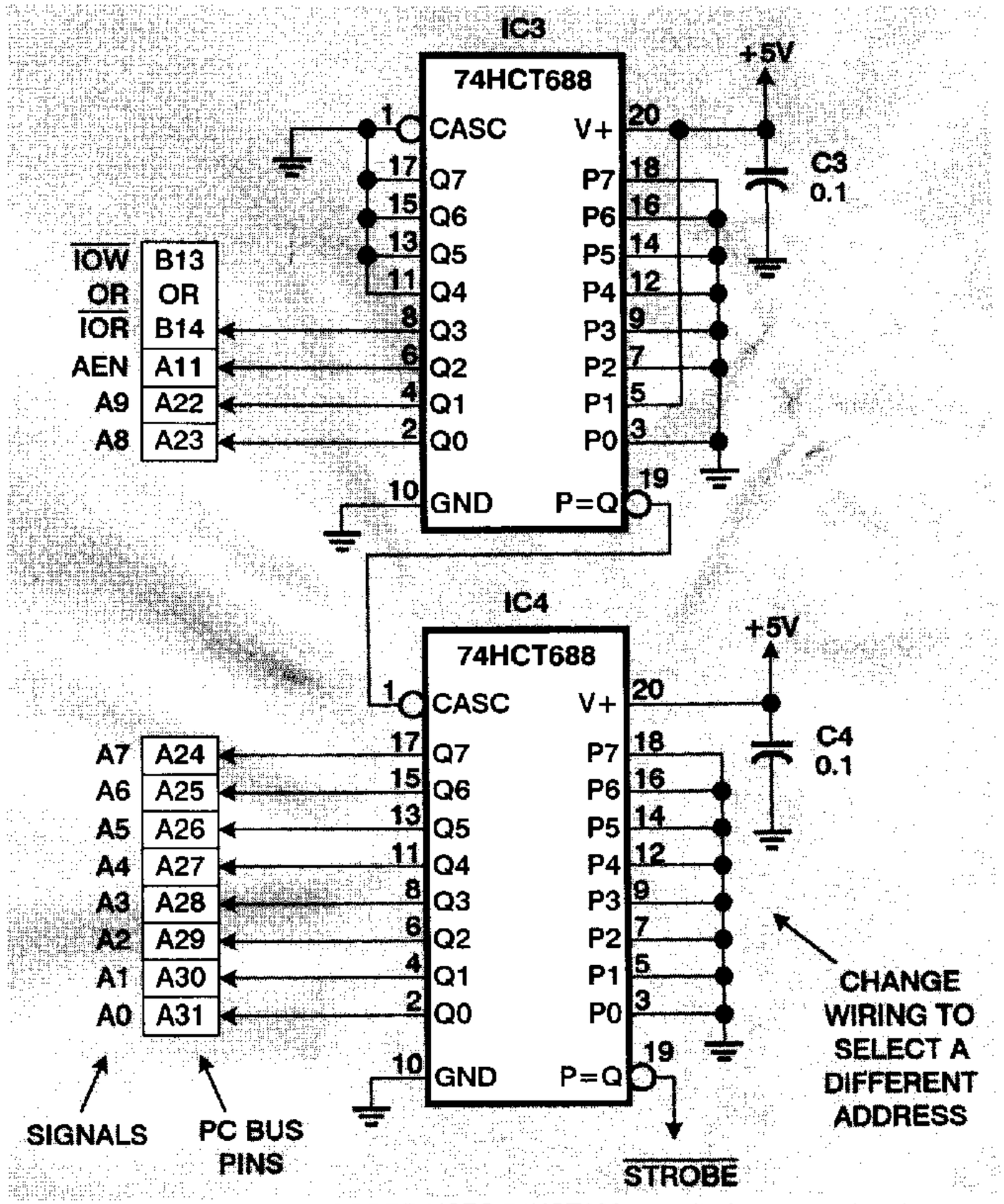

FIG. 3—THIS ADDRESS DECODER USES 74HCT688 magnitude comparators to recognize address 280(hex) when wired as shown.

CPU pin			Operation
S0	S1	S2	
0	0	0	Interrupt acknowledge
0	0	1	I/O port read (IOR)
0	1	0	I/O port write (IOW)
0	1	1	Halt
1	0	0	Memory read (MEMR), instruction
1	0	1	Memory read (MEMR), data
1	1	0	Memory write (MEMW)
1	1	1	Idle