Process Control With Personal Computers
(Part II)

Sensors, stepping motors and interfaces for computerized control systems

By Dr. H. Edward Roberts

Last month, Part I introduced the basics of process control with IBM PC/XT/AT or compatible computers and an internal and external controller bus system to allow such standard computers to perform these "real world" tasks. The system used consisted of Datablocks' LINK interface board and an A-II external bus structure in a shielded case, the latter allowing a host of encased control modules to be plugged in. This creates some 2,000 additional I/O channels without interfering with the normal computer system's operation.

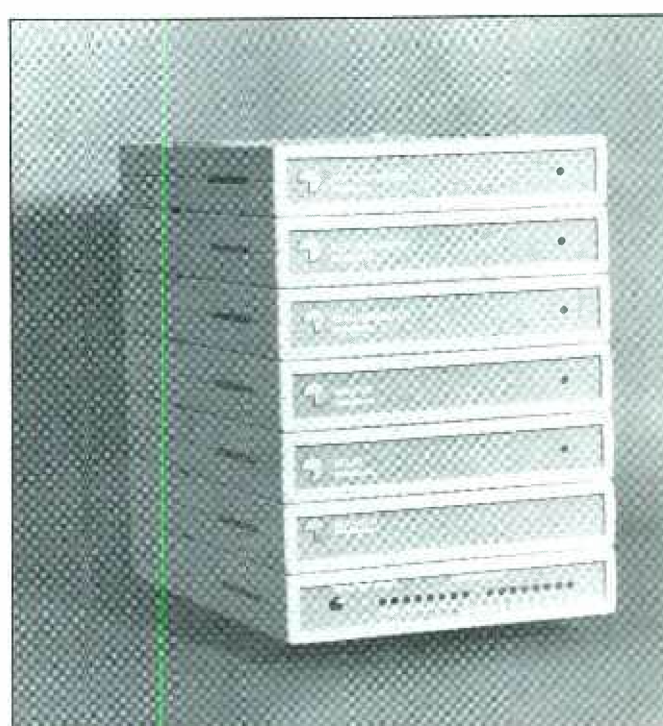
Block diagrams of the interface boards and the parts they contain are shown in Figs. 4 and 5, with construction plans, parts required and a kit source noted. Now we'll discuss external devices, such as sensors and stepping motors, that are needed to make it all happen, as well as software control and a few simple control projects.

Transducers

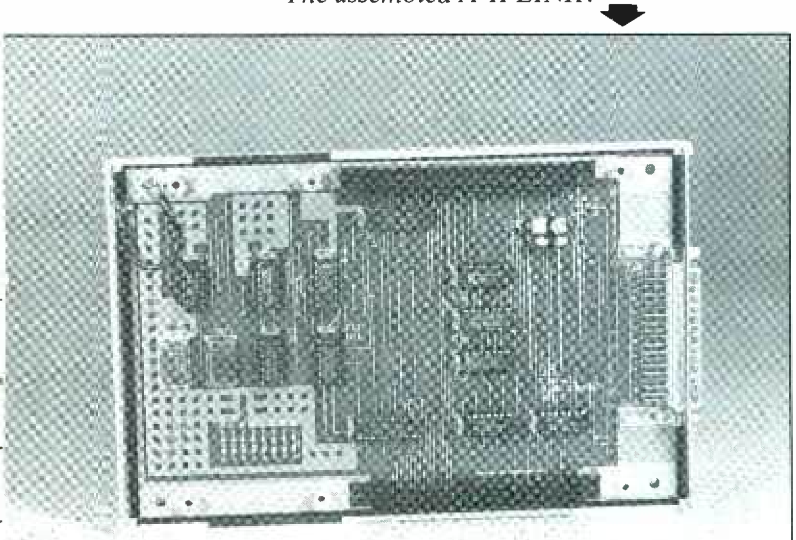
Transducers or sensors are needed to "hear," "see" and "feel" the outside world. Basically, they're devices that convert a physical change to an electrical value, say, temperature to voltage. This allows a computer to deal with various physical quantities or energy to be measured, monitored and/or controlled. The sensors generate analog electrical signals which must then be converted to digital signals through an analog-to-digital converter (ADC) in order for a computer to handle them. A digital-to-analog converter (DAC) is employed to send signals from the computer to the physical device to be controlled.

Transducers come in many types

DataBlocks A-II external control modules plug together as function blocks.



The assembled A-II LINK.



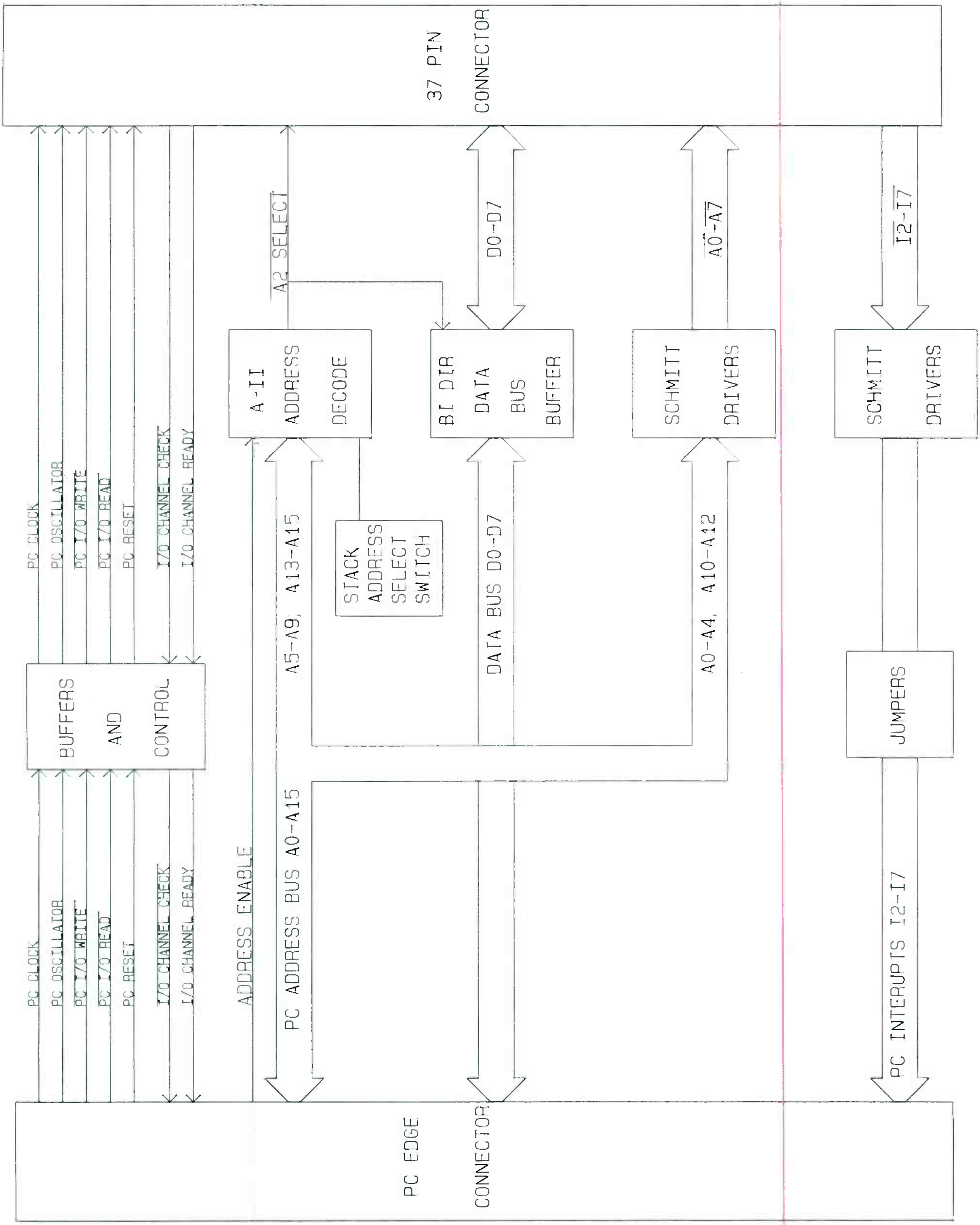


Fig. 4. Block diagram of PC-LINK, PC side.

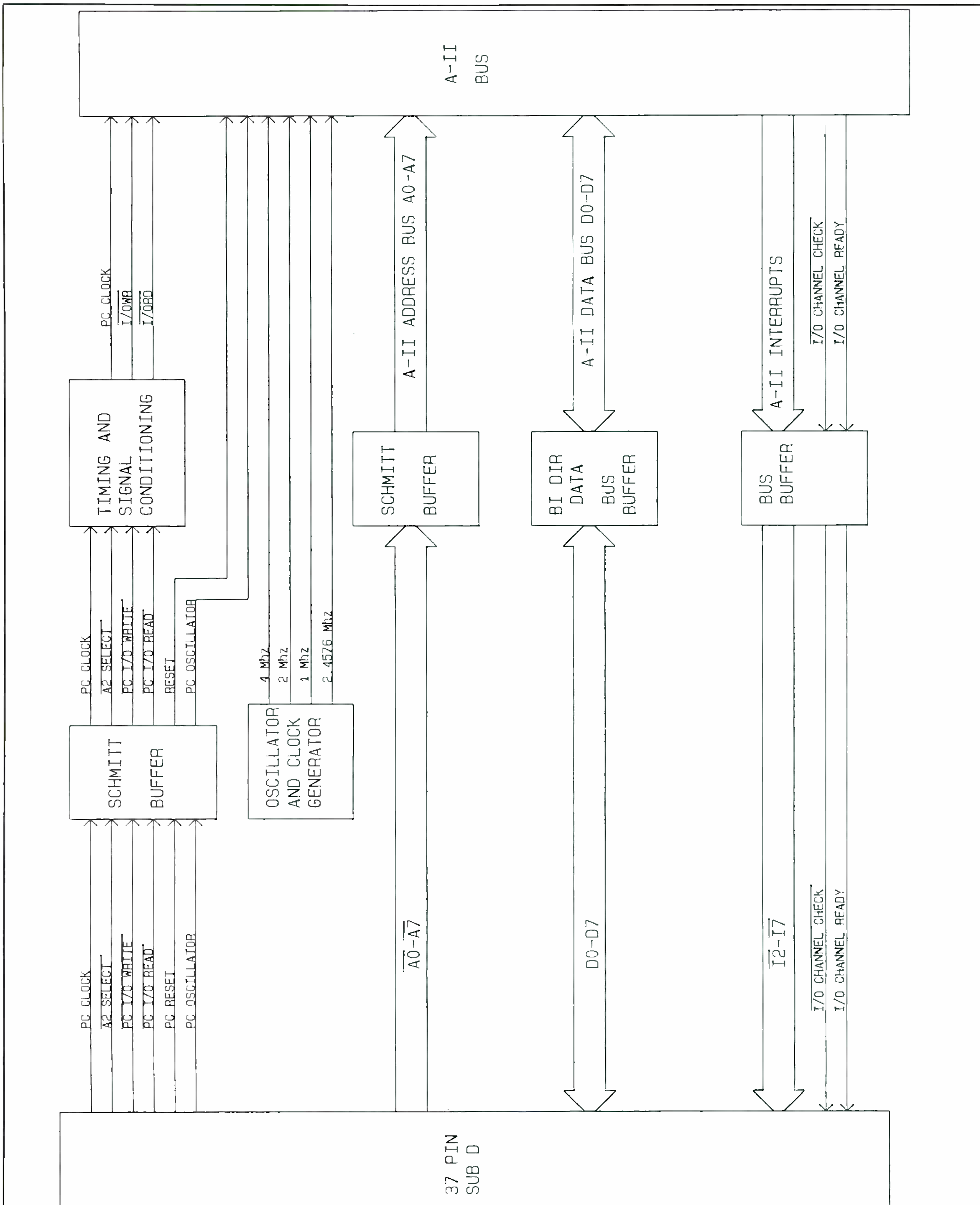


Fig. 5. Block diagram of PC-LINK, A-II side.

PARTS LIST

(PC Side)

Integrated Circuits

IC1—74HC03 open-drain NAND gate (Motorola)

IC2—74HC688 8-bit equality detector (Motorola)

IC3—74HC245 octal transceiver (Motorola)

IC4 thru IC7—74HC14 inverting Schmitt trigger (Motorola)

IC8—74HC00 quad 2-input NAND gate (Motorola)

Capacitors

C1 thru C8—0.47-μF ceramic disc

Resistors (¼-watt, 5% tolerance)

R1 thru R14—10,000 ohms

Miscellaneous

SW1—4-position spst DIP switch

Printed-circuit board and mounting bracket with screw (DataBlocks); six 14-pin and two 20-pin IC sockets; 37-pin D-type right-angle connector; machine hardware; solder; etc.

(A-II Side)

Integrated Circuits

IC2,IC2—74HC245 octal transceiver (Motorola)

IC3,IC11—74HC73 dual J-K flip-flop with clear (Motorola)

IC4,IC10—74HC74 dual D flip-flop with preset and clear (Motorola)

IC5,IC6,IC7—74HC14 hex inverting Schmitt trigger (Motorola)

IC8—74HC00 quad 2-input NAND gate (Motorola)

IC9—74HC04 hex inverter (Motorola)

IC12—74HC10 triple 3-input NAND gate (Motorola)

Capacitors

C1 thru C12—0.47-μF ceramic disc

C13,C16—47-μF, 16-volt electrolytic

C14,C15—47-μF, 10-volt electrolytic

Resistors (¼-watt, 5% tolerance)

R1 thru R8—10,000 ohms

Oscillators

OSC1—8.0 MHz (NEC)

OSC2—2.4576 MHz (NEC)

Miscellaneous

Printed-circuit board (DataBlocks); ten 14-pin and two 20-pin IC sockets; 37-pin D-type male right-angle connector (Amp); 37-pin shielded cable with 37-pin D-type connector on both ends (DataBlocks); two male 60-pin Molex connectors; two 60-pin female Molex connectors; machine hardware; solder; etc.

Note: The following items are available from DataBlocks, Inc., P.O. Box 449, Alamo, GA 30411 (tel.: 800-652-1336): IBM (or compatible) half of LINK—double-sided, plated and silkscreened pc board, $29; complete kit of parts, including pc board, ICs, etc., $65. Complete LINK, IBM and A-II halves—pc boards only, $49; complete kit of parts, including pc boards, connectors, cabling, ICs, etc., but excluding case, $118; complete assembled and tested LINK, in case, $187. shielded A-II case, $12. All schematics and foil patterns (included with foregoing kit) are available separately for $8. Add $5 P&H. Georgia residents, please add state sales tax.

and forms, depending on the physical properties they must sense. Consequently, there are transducers specifically designed to sense temperature, pressure, light, humidity, flow, chemical action, etc. There are a variety of different types within each category, of course, giving users a great deal of flexibility. For example, to measure or control temperature, one could use a thermocouple-junction device that produces voltages that are proportional to temperatures. These are low-cost, rugged devices, but don't feature tight accuracy or quick response time. Other choices for this purpose include thermistors, which react to temperature changes with a shift in its resistance, but their operating range is rather narrow; resistance-temperature detectors, which are more refined (and costlier) devices than thermistors, feature fast response time and precision measurement capabilities when used with bridge circuitry; and semiconductor temperature sensors feature great linearity.

Once transducers have been selected, it might be necessary to calibrate them. Typically, conversion tables to correct for temperature dependencies, nonlinearities, etc., are supplied by the transducer's manufacturer. Once the transducers have been selected, calibration curves prepared and the microcomputer interfaced to the transducer(s), channels for data collection and device control are established. Next comes the software, which must gather and prepare information as it's collected for output, whether for controlling an external device or for printed reports. For the former, this might mean transmitting the necessary information in the appropriate format to, perhaps, increase or decrease flow, raise or lower pressure or temperature, and so on. For data logging, this might mean transferring data to a hard-copy device such as a printer, or to a disk for information storage.

Sensor Systems

Let's look at some of the more common types of transducers to gain a better understanding of how to design a sensor system. Consider a pressure transducer, which is used to convert a fluid pressure, say, air or liquid, into a proportional voltage that may then be input to a computer device such as an A/D converter. An integral component of most pressure transducers is the strain gauge, which is itself a transducer.

Strain gauges are, in the simplest terms, merely devices that alter their electrical characteristics when subjected to stress that slightly deforms them. Strain is formally defined as the ratio of mean deformation per unit length.

The piezoelectric effect has long been employed in strain-measuring devices by utilizing crystals as strain gauges. Capacitive and inductance effects have also been used. However, each of these display some weaknesses. The most common strain gauges are those that provide a change in resistance proportional to the applied strain. Semiconductor strain gauges have also been used which, while physically small, are

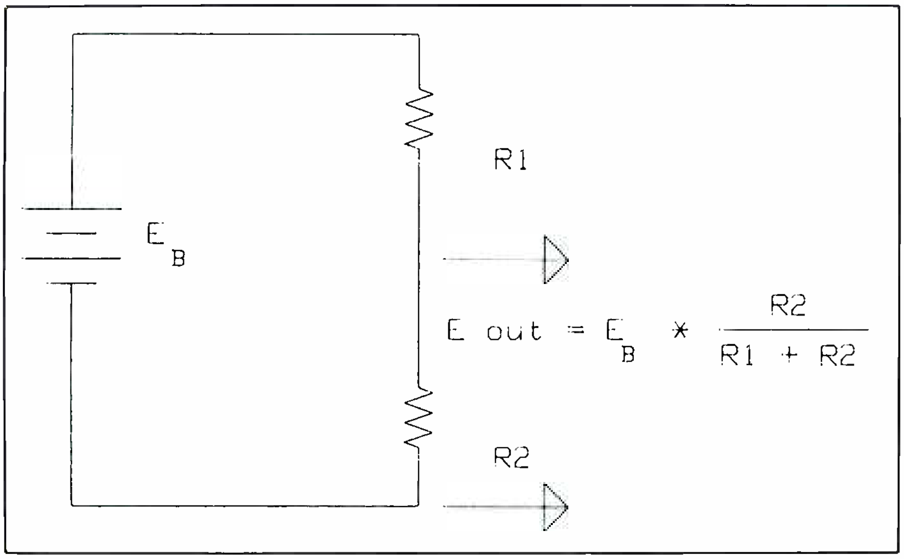


Fig. 6. A typical voltage divider.

capable of providing relatively large output voltages (for example, 5 volts full-scale). They also exhibit a great sensitivity to strains applied and are inherently nonlinear. Semiconductor devices also exhibit an undesirable strong temperature dependence that must be corrected in order for the readings to be accurate.

The bonded resistance strain gauge is the most widely used tool for strain measurement. It consists of a grid of fine wire or metallic foil bonded to a thin insulating material called the carrier matrix. The resistance gauge is glued to the object in which strain is to be measured, and the strain is determined by measuring the resulting change in resistance in the gauge when the object is loaded. These devices can be used to produce a pressure transducer by bonding the resistance strain gauge to a flexible diaphragm which is in contact with the gas or liquid whose corresponding pressure is to be measured. Such devices are usually provided with a compensation network. The latter provides control over sensitivity, zero balance, thermal effects on sensitivity and the thermal effect on the zero setting of the transducer.

Historically, the Wheatstone bridge, due to its simplicity and sensitivity, has proven to be the most common device for measuring resistive changes in strain gauges. The microcomputer, in combination with bridge-circuit technology, allows use of simpler circuits, increased measurement accuracy and collection of larger amounts of data by utilizing multichannel systems. This is accomplished, in part, by utilizing the microcomputer to balance the bridge circuit, compensate for nonlinearities, and handle switching and storing in multichannel applications.

Another popularly used transducer is the photoconductive device. To examine how it works, let's first review the voltage divider. In Fig. 6, a simple voltage divider, we see two resistors in series. The voltage across *R2* is equal to the input voltage times R2/(R1 + R2). If we increase *R2*, the voltage across it will increase and, likewise, if we decrease *R2*, the voltage across it will decrease.

Now let's replace *R2* with a photoresistor, a device whose resistance is a function of the light striking its surface. As light increases, resistance decreases. Thus, output voltage will be inversely related to the brightness of the light shining on the photoresistor. Photoresistors are quite easy to use, as you can see. The value of *R1* is selected to be approximately equal to the value of the photoresistor at the light level to be used. In general, one should select photoresistors with a relatively low "on" resistance to minimize noise pickup. Photoresistors are relatively slow devices. Therefore, they will not respond to high-speed changes in light intensity as will photodiodes and phototransistors. However, their simplicity and sensitivity make them a good choice for a wide variety of light-sensing applications.

There are a variety of sensors available for temperature control. Among the simplest and most versatile in this category is the thermistor, a resistor whose resistance varies as a function of temperature. Actually, this is true of all resistors, but thermistors are especially sensitive to temperature changes. Thermistors can be used in voltage-divider circuits like the photoresistor circuit cited. The output in this case varies as a function of temperature. You select thermistors based on the temperatures at which you expect them to normally operate.

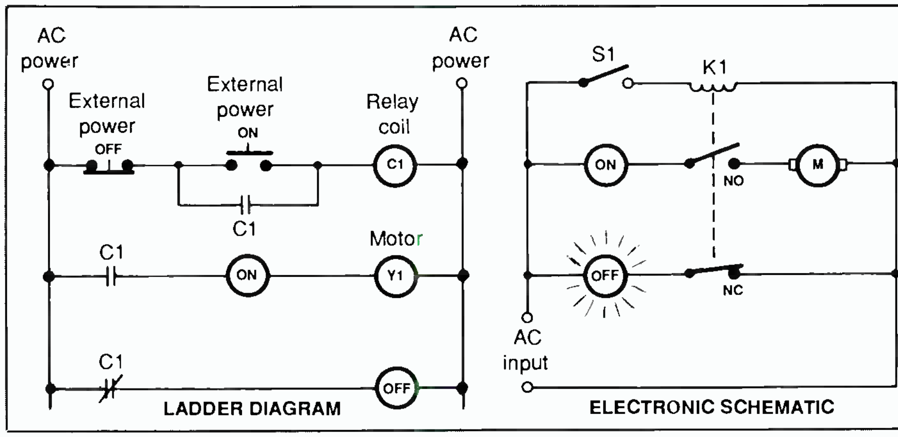
The foregoing transducer examples are representative of sensors used in process control systems. In actual practice, virtually any sensor that's capable of producing an electrical output can be used as an input device for a control system.

Stepping Motors & Resolvers

Stepping motors are ideally suited for digital control systems because they move a predictable amount for each input pulse.

A stepping motor rotates a predetermined amount each time it "steps." Typical rotations are 1.8, 7.5 and 15 degrees per pulse. Since this is totally predictable, the "feedback is implied." Thus, we can predict the motor's location by keeping track of the number of pulses we have sent. A stepping motor operates

Relay-Ladder System



The relay language is the oldest and most widespread of process control "languages." Actually, it's both a language and a schematic logic representation much like the flow diagrams we use routinely for software design. It was developed when the electromechanical relay was king in the industrial control field. So it's no surprise that the relay contact is the key symbolic element of the language.

The system is often referred to as relay ladder programming or ladder diagrams because the drawing format resembles that of an ordinary ladder. It consists of vertical lines or rails, one at the left and one at the right, that represent input and output power-line legs. Connected between are horizontal rungs of devices, such as relays, switches, motors, etc. Industrial programmable controllers often use keyboards with relay logic symbols on keys.

Input representations such as an input module or a relay contact switch are traditionally placed at the left side of a rung, while outputs follow to the right of them. What seem to be the two small vertical lines that represent a capacitor in an electronic schematic are actually normally open input switches (say, a relay's contacts) in a relay ladder diagram. The same drawing with a diagonal line through it indicates that the contacts are normally closed. Outputs, in turn, are represented by circles.

Further, letter or number labels are used to identify these symbols. For example, an "X" (X1, X2, etc.) might be used to indicate control switches, while a "C" (C1, C2...) designation would identify a relay, which might be in the form of the capacitor-like symbol for contacts or a circle for the energizing coil. Other output devices are frequently designated as "Y" (Y1, Y2...). Note, however, that commercial programmable controllers often use a number system. This might be a four-digit system, with specific I/O number assignments and blocks set aside for particular devices. For example, numbers 0001 to 0050 may be reserved for control relays only. Assignment numbers cannot be repeated for another device except for multiple switch contacts that work with a single relay coil. Accordingly, a dual-contact relay might be shown as two separate switches (capacitor-like symbols) with the same C1 label, while their common coil would be the only other device to carry a C1 label. Assignment numbers would be different, though. (Some manufacturers maintain separate symbol or label numbers, however.)

A variety of sequences and logic operations can be formed by appropriate designs. For example, a rung could have two power control switches (X1 and X2) and a motor (Y1) in series between the two power rails, which would represent an AND logic arrangement. If the switches were in parallel, it would constitute an OR logic setup. In the former, both switches would have to be closed in order for the motor to be powered, while in the latter setup either switch could be closed to run the motor.

In the illustration shown here, you'll observe how the same electronic setup would be drawn for a ladder diagram and an electronic diagram equivalent. Here, if power is off, the "off" lamp indicator, which is in series with the relay's normally closed contact, is lighted since the rung is directly across the ac power line. When power is switched on, the coil is energized and closes the normally open contacts, causing the "on" lamp to light and the motor in series with it to be powered. In turn, the normally closed contact automatically opens and the "off" lamp extinguishes.

This is a simple example, but it illustrates the principle of drawing ladder logic diagrams as contrasted to electronic schematics.

in an anticipated manner by pulling the rotor from one electromagnet to another. This is accomplished externally by applying voltages in sequence from one winding to another. Figure 7, a Stepping Motor Truth Table, depicts the four windings of a typical stepping motor and indicates how the sequential application of voltage to the windings controls shaft movement. If we keep track of which windings were last used, we can determine which winding needs to be activated in order to rotate the motor in either direction.

Unfortunately, there are a couple of "flies in the ointment." First, keeping track of which winding needs to be activated and controlling the appropriate driver is relatively complicated, at least if you don't have a computer. Of course, if you have a computer in the system, this problem is easily taken care of. That

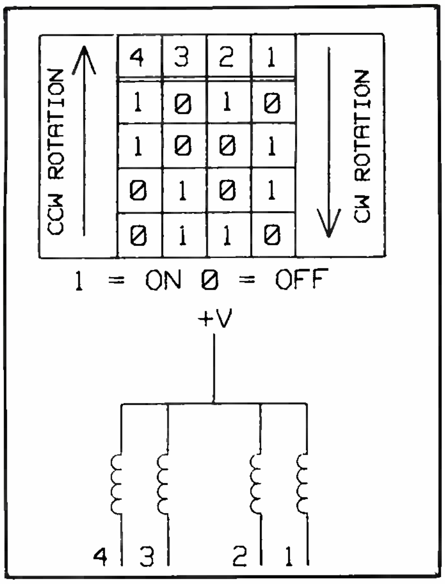


Fig. 7. Stepping motor truth table.

is, a stepping motor either makes a complete step or it doesn't move at all when it is pulsed. If the motor is overloaded, it may stall. Since the feedback is only implied, the controller will not be able to recognize the problem. Therefore, it's important to make sure the system either does not overload the stepper or include some type of override feedback system to warn of a stepper failure.

Another problem with steppers is that they are relatively expensive, especially in large sizes. In addition, high-torque steppers are not available. The maximum practical size stepper is limited to torques of a few foot-pounds. Of course, this torque can be amplified with gears, lead screws, etc., at the expense of speed. Stepping motors are relatively slow devices, operating typically between 30 and 300 rpm.

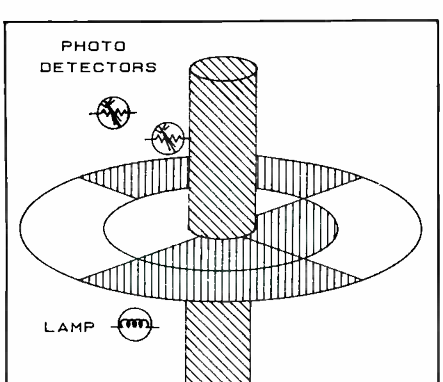


Fig. 8. A two-bit resolver.

How about using plain dc and ac motors for rotary motion, you might ask? Unfortunately, it will not be easy to tell what the motor position is at any given point in time. If a typical dc motor is turned on and then off, the number of rotations or the exact angular position of the shaft is not known. This can be solved by using resolvers, however.

A resolver is simply a device that measures the location of a shaft and reports this to a controller by feeding information back to it. A two-bit resolver in Fig. 8 illustrates an example of a low-precision resolver. This particular resolver divides one complete revolution into four parts and provides a feedback signal describing the angular location of the shaft to an accuracy of two bits. A three-bit resolver would resolve the location into eight parts; a four-bit into 16 parts; etc.

This device works by detecting the presence or absence of light on the photodetectors. For instance, in our example, light striking both detectors would place the rotation in the first 90 degrees. Light on the inner detector and not on the outer detector would place the resolver in the second 90-degree position. You can see how this technique can be expanded to whatever degree of accuracy is required.

Such a technique has the additional advantage of the resolver being completely independent from the rotor. Therefore, a motor failure or other problem would be easy for the computer to detect. The disadvantage is that this technique is more expensive in low-torque applications than stepping motors. But in high-power applications, a resolver coupled with a motor becomes very attractive.

So far, we've been discussing rotary techniques. It's often necessary to convert this to linear motion to make operations practical. To do this, we can drive a screw (called a lead screw) with a nut attached. A sophisticated, low-backlash. low-friction lead screw is called a ball screw. It uses a ball bearing as one half of the thread.

Earlier it was pointed out that the standard rotation for a stepping motor is 1.8 degrees per pulse. Seems like an odd choice, doesn't it? Standard machine tools, such as lathes and milling machines use lead screws to move their mechanisms. These lead screws are designed so that one complete rotation of the lead screw produces 0.2 inch of linear movement. A stepping motor that rotates 1.8 degrees per pulse, therefore, requires 200 pulses for it to make one complete revolution. Therefore, if we connect the stepping motor to a lead screw, each pulse would produce exactly 0.001 inch of linear motion.

By using a stepping motor to produce linear motion, we can then directly convert the number of pulses to linear movement. This makes for simple programming of linear movement. In a similar manner, we can generate linear motion using a conventional motor/resolver combination. Other types of linear-motion devices include simple solenoids, hydraulic and pneumatic cylinders, etc. Resolvers are also required for these linear-motion mechanisms. Resolving techniques similar to the rotary technique described above are possible. Also, a simple linear resistor connected to an A/D converter is a reasonable method to measure linear motion in many applications.

Special Interfaces

Among the typical interfaces needed to move data from one state to another and from one location to another, as previously mentioned, the

IEEE Standard Digital Interface for Programmable Instrumentation is particularly important in control work. It's an accepted standard that allows for the exchange of digital information between system components and instrumentation. Originally developed by Hewlett-Packard and sometimes referred to as the GPIB or HP-IB interface, it is most often called the IEEE-488 interface.

The interface has become so popular that many companies specializing in electronic instrumentation now make it available on their instruments. Additionally, several sources provide personal computer add-on cards to establish the interface. As a result, system integrators now have a compatible means of communication between instrumentation manufactured by different companies and their own systems.

Devices interconnected by this interface fall into three groups: listeners, talkers and controllers. Listeners are devices that have been configured to receive messages. In contrast, talkers are devices that are configured to send messages to other devices, while controller devices configure other devices, including themselves, to be talkers or listeners. Additionally, the controller causes another device to perform a specific action, such as send or receive data. The devices send and receive two types of messages: interface messages and device-dependent messages. Interface messages are those that are sent to cause the interface to react in a certain way. For example, such a message might cause a device to be a listener only, a talker only or both listener and talker.

An interface message may also command the interface on a device to send data or prepare a device's interface to receive data. Device-dependent messages are those that are carried by the interface, but are not processed or used by it. Such a message contains information for the device itself, such as a set of control commands to cause a tape recorder to rewind or the data to be recorded on the recorder.

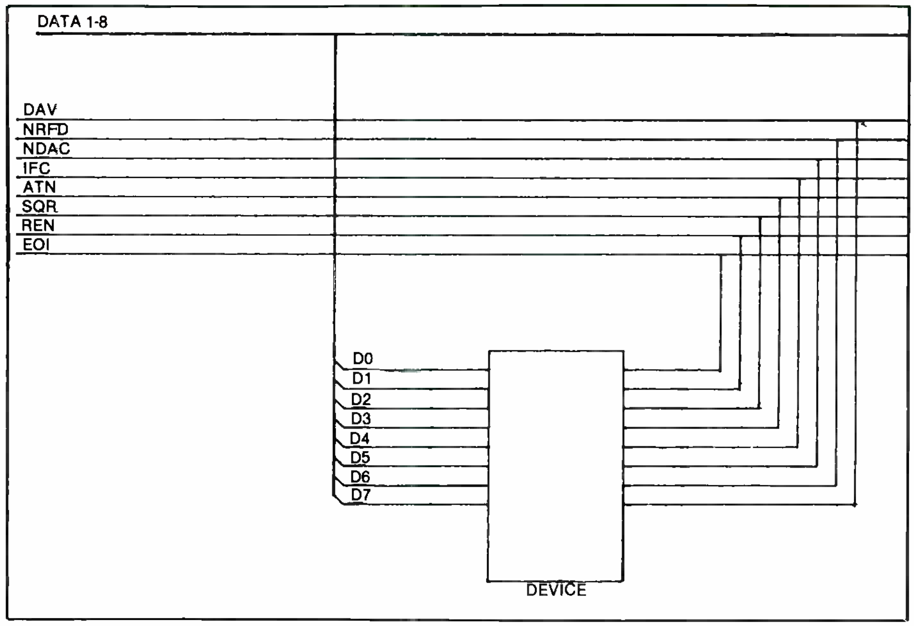


Fig. 9. The IEEE-488 interface bus configuration.

The message exchange across the interface is accomplished through a byte serial, bit parallel transfer between devices. This means that there are eight lines that simultaneously transfer the eight bits in a byte, with multiple bytes transferred sequentially. In addition to the data lines, the interface also contains eight control and handshake lines. Three of these—DAV, NRFD and NDAC—control the data byte transfers by performing a "handshake" between the sending and receiving devices. The other five lines—ATN, IFC, SRQ, REN and EOI—are used to perform system management and control functions and sending control commands, requesting service and clearing devices. Figure 9 illustrates the IEEE-488 interface bus configuration.

Communication over the system is managed by the controller. When a device needs to be "serviced," it informs the controller. The controller thus knows whether the device requesting service needs to send information, that is, becomes a talker, or whether it needs to become a listener and receive information. If the device needs to talk, the controller also knows which device or devices need to listen to the communication. The controller configures the appropriate device to talk and the appropriate device or devices to listen. Once this is done, the controller relinquishes control of the bus and the talker transmits its message to the listeners. At the end of the message transmission, the controller can reconfigure the devices on the bus to meet the talk/listen requirements of the next message.

One of two methods is generally used to interface the system to inform the controller that a device is ready for service: a serial poll or a parallel poll. In the serial poll method, the device that needs servicing asserts the SQR line. The controller senses this and informs the interface system that a serial poll is about to begin. Following that, the controller sequentially addresses each device to determine which one requested service. The one requesting service responds to the poll by setting data bit 7 true.

In the other polling method, a par-

START CODE	HOUSE CODE	FUNCTION OR NUMBER CODE	START CODE	HOUSE CODE	FUNCTION OR NUMBER CODE

Fig. 10. The X-10 Powerhouse Serial Data Stream.

allel poll, the controller assigns each device a dedicated data line that the device uses to respond to a parallel poll. When the controller issues a parallel poll command, each device needing service will respond by asserting as true the data line assigned to it. This allows the controller to get status from all devices at one time.

The standard IEEE-488 interface accommodates up to 15 devices on the system at one time. If these devices are equipped with open-collector drivers, they can communicate at 250,000 bytes per second if the total length of the interconnecting cables doesn't exceed 20 meters. If the devices are equipped with tri-state drivers, the communication rate can increase to 500,000 bytes per second. With shorter cable lengths and specially designed interface timing, the system can achieve a maximum of 1,000,000 bytes per second.

We'll be using an IEEE-488 interface module in a later construction project. Additionally, we'll employ a popular, modestly priced and widely available stand-alone remote-control system, the X-10 Powerhouse system. This system is designed to control up to 16 on/off functions at remote locations. Primarily used to control lighting and appliances in the home, it consists of a centrally located controller and remote stations that are plugged into power outlets or wired into light switches. Each remote can be addressed by the central controller and responds to commands such as off, on, bright and dim. Communication between the controller and remotes is over existing power wiring using a patented data protocol.

The power-line interface is a serial transmission system that transmits binary data from the central controller to the remote stations. The serial data stream, shown in Fig. 10, an X-10 Serial Data Stream, consists of a 4-bit start code, a 4-bit house code and a 5-bit code that's either a number code or a function code. The start code is always 1110. The 16 unique addresses allowed by the 4-bit house code enables adjacent homes to run the X-10 system without interfering with each other. This also permits more than one controller to be operated in the same home if more functions need to be automated than the number and function keys can accommodate.

The number and function code is a unique 5-bit code for each of the 16 number keys and the six function keys on the controller. When a number key or a function key is pressed, the appropriate binary code is transmitted to the remotes over the power line. The codes are transmitted in the order of a start code, house code and number or function code. They are represented on the power line as a sequence of short bursts from a 120-kHz oscillator occurring at a rate of one bit per cycle.

The central controller modulates a 120-kHz signal on the power line to transmit a "1." The burst is 1 millisecond long and is transmitted coincident with the zero crossing of the 60-Hz power signal. The next bit is transmitted at the beginning of the next cycle. If it is another "1," a 120-kHz signal that lasts for 1 millisecond is again transmitted. If it is a "0," no signal is transmitted. The complement of each of the bits is transmitted on the alternate half-cycle of the power waveform. That is, if a "1" is transmitted at the begin-

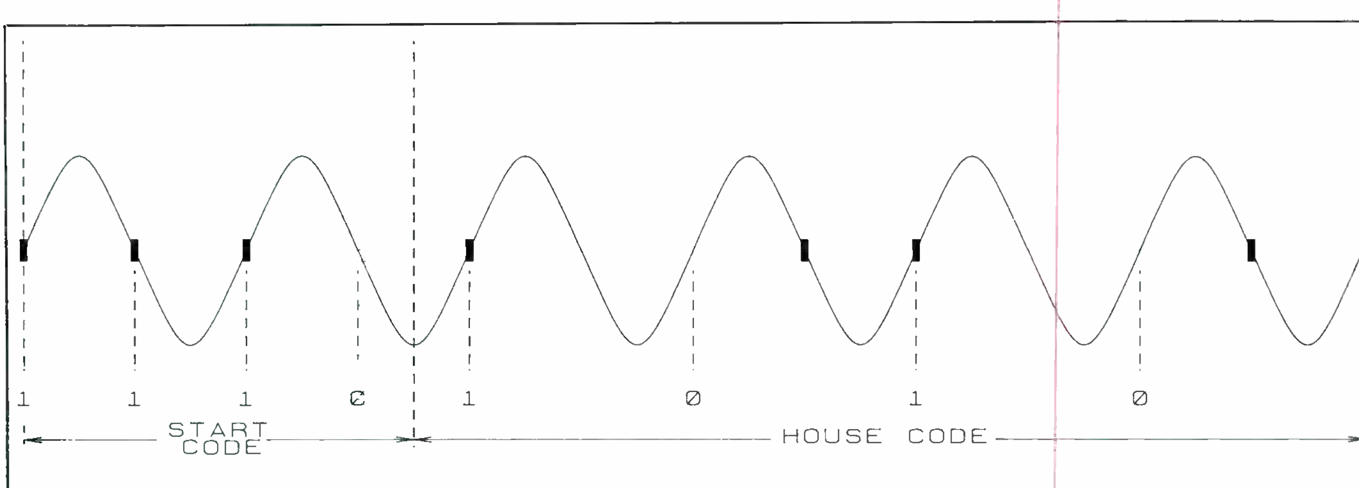


Fig. 11. Transmitted bit representation for X-10.

Control Device Sources

DataBlocks, Inc. (579 Snowhill Road, Glenwood, GA 30428) stocks a complete line of control sensors and actuators mentioned in this article. These items are also available from the following companies:

Stepping Motors-Linear Steppers
Hurst Manufacturing Div.
Emerson Electric Co.
Box 326
Princeton, IN 47670
(812) 385-2564

Oriental Motor U.S.A., Corp.
2701 Toledo Street, Suite 702
Torrance, CA 90503-9971
(213) 515-2264

Solenoids-Stepping Solenoids
A.B. Andrews & Co., Inc.
P.O. Box 12167
Research Triangle Park, NC 27709
(919) 544-1762

Newark Electronics
6950 Peachtree Industrial Blvd.
Norcross, GA 30071
(404) 448-1300

Hydraulic/Pneumatic
Clippard Instrument Laboratory Inc.
7390 Colerain Road
Cincinnati, OH 45235
(513) 521-4261

Parker Hannifin Corp.
Atlanta Fluidpower Sales
2264 Northwest Pkwy., Suite 6
Marietta, GA 30067
(404) 956-0881

Thermistors
Digi-Key Corp.
P.O. Box 677
Thief River Falls, MN 56701
(1-800-344-4539)

Omega Engineering Inc.
One Omega Drive
Box 4047
Stamford, CT 06907
(203) 359-1660

Thermometrics
808 U.S. Highway 1
Edison, NJ 08817
(201) 287-2870

Photoresistors
Mouser Electronics
2401 Hwy 287 North
Mansfield, TX 76063
(817) 483-4422

ning of a cycle, a "0" is transmitted at the zero crossing that occurs one half-cycle later. Similarly, if a "0" is transmitted at the beginning of a cycle, its complement, a "1," is transmitted one half-cycle later.

This Transmitted Bit Representation is depicted in Fig. 11. Here, a start code, 1110, and a house code, 1010, are shown as they might look on an oscilloscope screen as they are being transmitted. Note that the complement of the bits is not transmitted on the start code. This is the only sequence for which the complement buts are not transmitted on the alternate half-cycle.

Each remote module can be configured to respond to the transmitted code by setting a rotary switch on the module to the desired house number and number code. This allows each of the 16 remote modules to be "addressed" from the central controller.

Once addressed, the remote will respond to a function command from the controller. For example, if key number 10 is pressed, the remote module with rotary switch set to 10, say, an outlet module, will be conditioned to receive a function command. If the function key "on" is pressed, module 10 will turn on. The device, say, a lamp that's plugged into module 10 will then be turned on. This can be extended to any of the modules in the system. Some controllers can be programmed to cause commands to be transmitted based upon the time-of-day. Thus, a simple home security system can be devised just by making the house look "lived in" simply by programming various lights in different parts of the home to turn on and off at appropriate times of the day.

One of the most useful modules from the standpoint of the computer user is the Powerhouse X-10 Universal module. This module, available later this year, provides all the hardware needed to interface the X-10 system to a personal computer. With this module, the personal computer user can provide control for up to 256 X-10 modules by taking advantage of all 16 house codes.

Next month, we'll conclude with software control typically used in computerized systems. We'll also design and build two process control systems. One will be for controlling the environment and security in a home, and the other for automatic testing and data logging while using the IEEE-488 interface. ME