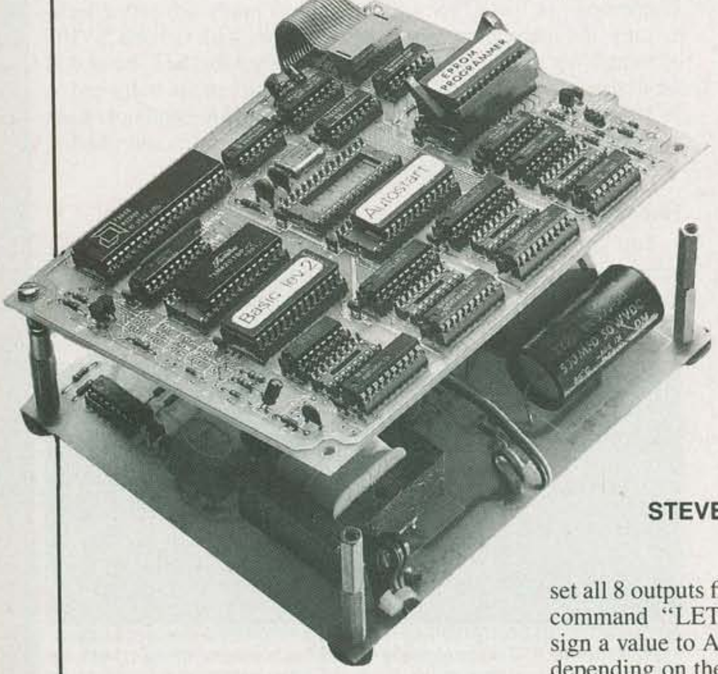
BUILD THIS

HOME CONTROL COMPUTER

In this concluding part of our look at a control computer you can build, we'll finish our discussion of the I/O systems and show you a sample application for the unit.



STEVEN E. SARNS

Part 3 AS WE PROMISED LAST time, we'll start this month with a look at the byte-wide I/O port.

The 8-bit parallel I/O port is the easiest to design. All that is required for inputs is to enable 8 buffers onto the data bus for each port. That job is done by IC2 and IC3, which are 74LS541 buffers. For outputs, all we have to do is to latch the status of the data bus, for which we use IC6 and IC7, 74LS377 D-type flip-flops.

While the byte-wide port is best suited to data transfer, it is possible to use it for bit-wise control as well—but it's not always easy to isolate the single bit that's of interest. In the case of input testing, it involves a mathematical process called *bit masking*. If the output port is used for single-bit control, a record of the state of the port must be maintained in the program so that the proper output word can be formed, that will toggle the desired bit but leave all of the other bits unchanged.

The same simple software commands used for the bit addressable ports are used to control the 8-bit parallel ports. For example, "OUT(16,0): OUT(16,255)" will set all 8 outputs first to zero then to 1. The command "LET A=INP(16)" will assign a value to A in the range of 0 to 255 depending on the status of the 8 inputs.

Analog inputs

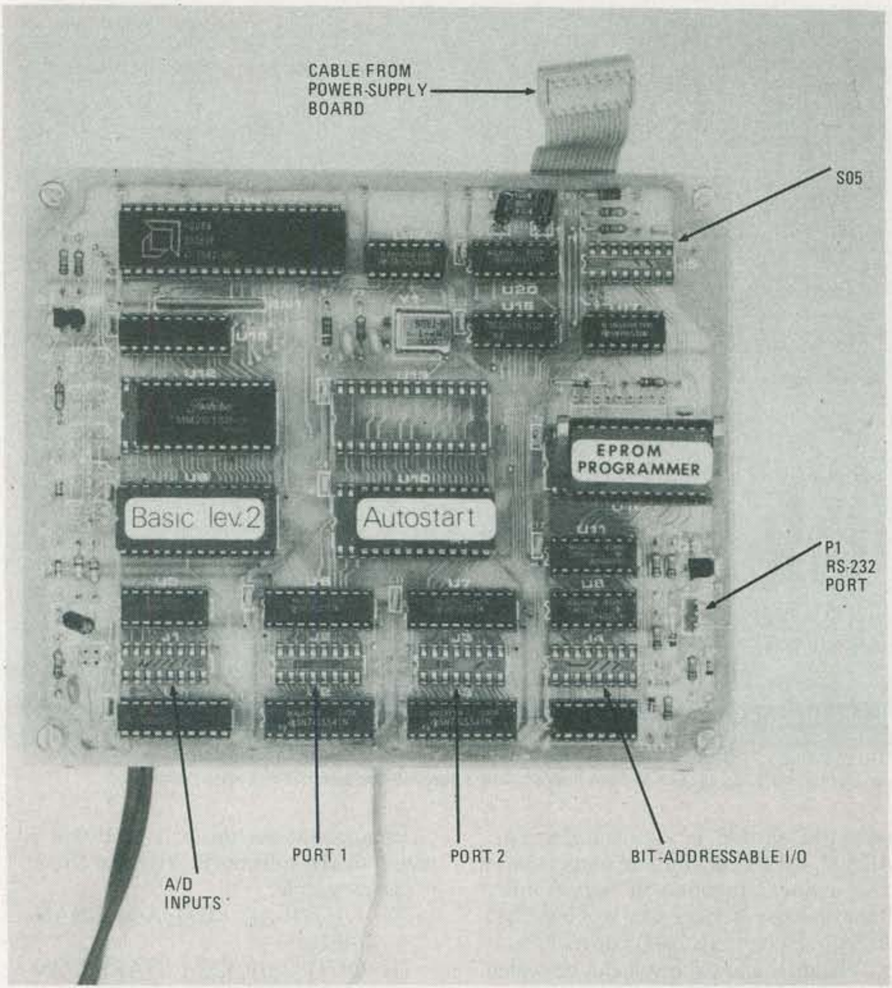
Most natural processes appear to us as analog or continuous-time functions. The temperature, pressure, humidity, sound levels, and light levels surrounding you are analog quantities. But the measuring process can convert the analog function into a discrete digital function. For example, we know that the temperature does not change in steps. But if your thermometer is only accurate to one-half degree, then the temperature will appear to change in 1/2-degree steps. If you used a thermometer with higher resolution, you could extend the number of digits used to express the temperature. But no matter how accurate the thermometer, you cannot express the temperature exactly.

In order for the computer to be able to "understand" analog quantities, they have to be converted into digital quantities. That's the purpose of the analog-to-digital (A/D) converter.

The first parameter used to describe our A/D converter is the number of bits the result will be—the resolution of the measurement. The smallest increment or change of the result is always one *least-significant-bit* (LSB). An 8-bit conversion resolves to one part in 256. The next parameter is the *span* of the conversion. That is the physical range that is represented by the zero and full-scale values of the A/D output. Thus, if our 8-bit converter had a span from 0° to 255°, each bit of output would represent one degree. If the span were reduced to 125°, each bit would represent 1/2°. The number of bits of the conversion determines the resolution of the converter. The span of the conversion determines the resolution of the physical quantity. Increasing the number of bits or decreasing the span will increase the resolution of the result.

The accuracy of the A/D conversion takes into account not only repeatability, but the absolute accuracy of the output compared to the input. That is really a measure of the accuracy of the standard used during the conversion process. If a reference diode defines full-scale as 2.55 volts, and if that reference should drift to 2.56 volts, then 2.56 volts will be required at the A/D input to produce a result at the output of 255. That potential source of inaccuracy can be eliminated by using a technique called ratiometric measurement. Ratiometric measurements are made using the same source for both the reference and the transducer. Thus any variations will be present in both and will cancel out.

The last parameter we should consider is the number of channels—the number of inputs that can be connected to the A/D converter. Obviously with 2 A/D converters, we can measure 2 different points.



COMPUTER BOARD. Note how the 24-pin devices are plugged into the 28-pin sockets *leaving pins 1 and 2 empty.*

However, by equipping a single A/D converter with an analog input multiplexer we can also measure several points (but not simultaneously).

There are two common methods of performing the conversion. The first is called integration. The input voltage causes a proportional input current to charge a capacitor to a specific voltage level. The time required to charge the capacitor is inversely proportional to the input voltage. The second method is called successive approximation (SAR). In that method, the analog input is compared to the output of a digital to analog (D/A) converter. The D/A converter is adjusted until its output matches the unknown analog input. The input to the D/A converter that produces the match becomes the output of the A/D converter.

The number of tries before a match is made is never more than the number of bits of resolution. Thus an 8-bit A/D will find a match in no more than 8 tries.

Integrating A/D's are very accurate but slow. Twelve- and 14-bit conversions take tens of milliseconds. Successive approximation (SAR) A/D's are fast, but are not as accurate. Eight-bit conversions may take only a few microseconds.

The A/D converter selected for our computer is IC1, the ADC0805 from National Semiconductor. That 8-bit SAR converter can span input voltages as low as .25 volts. Provision has been made for ratiometric or absolute conversion. The converter has true differential inputs and zero-offset adjustment. An analog multiplexer is included in the circuit that accommodates up to 8 input channels.

The ADC0805 is designed to be used with microprocessors, so all we have to do is to connect it to the data and control buses.

A conversion is initiated by writing (anything) to the A/D converter. The result will be available by the time BASIC interprets the next program statement. The channel-select controls of the input multiplexer (IC5 pins 9, 10, and 11) are connected to three outputs of the 8-bit port at SO2. That doesn't mean that you have to use the port, only the A/D converter—it can be used for data transfer. The only requirement is that the multiplexer's control inputs be stable when the A/D converter is making a measurement.

Using the A/D converter is a simple 1-2-3 procedure. First, select the channel (using the 8-bit output port). Second, write to the A/D to start the conversion. Third, read the result. In BASIC, we could write:

```
10 OUT(16,0)
20 OUT(32,0)
30 PRINT INP(32)
```

Line 10 selects channel 0, line 20 writes a zero to the A/D to start conversion, line 30 prints the result.

Digital-to-analog conversion

Most of the same parameters that we noted for A/D conversion also apply in D/A conversion. Although the control computer doesn't have D/A capabilities, they can be added with very little effort. As shown in Fig. 13-a, a simple resistor-capacitor network is the only hardware required. Remember the description of the SAR A/D converter? We can implement a D/A converter in a similar manner with the components we have on the board and some software.

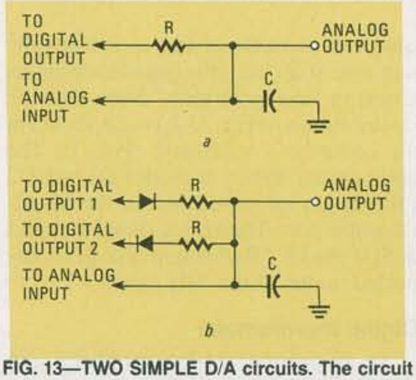


FIG. 13—TWO SIMPLE D/A circuits. The circuit shown in *b* is also shown in Fig. 1, the schematic of the computer. The values of the components depend on your particular application.

One of the bit-addressable outputs and one input of the A/D converter must be dedicated to each D/A channel. The digital output is used to charge a capacitor through a resistor, and the A/D input measures the voltage on the capacitor. That measured voltage is compared to the desired value (which is stored in the program). If the voltage is too low, the output is turned on. If the voltage is too high, the output is turned off. The selection of the R-C time constant, and the frequency that the output is corrected by the program, determine the accuracy of that simple approach.

The major limitation of that approach is that the ripple and response time of the circuit are directly related. If minimum ripple is required, a long response time is the result. That can be avoided with a slightly more sophisticated circuit, such as the one shown in Fig. 13-b.

That circuit uses one bit-addressable output to charge the capacitor and another to discharge it. When the voltage is correct, the capacitor is neither being charged nor discharged. Thus, the only contribution to ripple is the discharge current imposed by the load. The response

A BASIC interpreter for the control computer is available (see Ordering Information). BASIC I can operate on 26 variables, and includes commands such as:

LIST
RUN
IF
GOTO
PRINT
STOP
XMIT
POKE
OUT
NEW
LET
FOR/NEXT
GOSUB/RTN
INPUT
REM
PEEK
INP

The BASIC II operating system includes all that is contained in BASIC I. In addition, it can operate on 52 variables and 1 array, and contains about twice as many commands. Those extra commands include FILL, STAT, DUMP, EPROM (which programs an EPROM with the current program) MOD, ONERROR, and many others.

time can be made as fast as it takes the software to decide if the capacitor requires charging or discharging. Note that the circuit shown in Fig. 13-b is also shown in the computer's schematic (Fig. 1). The open circles shown at diodes D1 and D2 (or R2) are for connection to the bit-addressable port. The analog signal is output at SO1 pin 15. (That output also gets connected to one of the A/D inputs.)

Digital thermometer

As an example of the simplicity and flexibility that smart control can provide, we'll look at how we could use the control computer to measure the temperature. The first step is to select the transducer. There are several options. We could use National Semiconductor's LM335, which has a linear output of 10 mV/°C. Another possible choice is Analog Devices' AD590, which has a current output that is also a linear function of temperature. Both of those devices can be connected directly to one of the A/D converter's input channels and the result can be scaled to read degrees. But suppose you just reach into the junk box and come out with an unmarked, unknown thermistor. You can connect it to the input of the A/D converter with a pull up resistor to +5VDC. Using IF statements, your program can find the correct temperature:

```
100 OUT(16,0):REM CHANNEL 0 CONNECTED TO THERMISTOR
110 OUT(32,0):REM START CONVERSION
120 LET A=INP(32):REM ASSIGN RESULT TO A
200 IF A=0 PRINT "45"
201 IF A=1 PRINT "44"
202 IF A=2 PRINT "42"
.
.
.
455 IF A=255 PRINT "5.8"
500 PRINT "DEGREES"
510 GOTO 100
```

Although that program will make any professional programmer shudder, it works, it's simple, and it doesn't require hours of debugging. One of the features that makes this computer so powerful is that it can be up-and-running in a minimum of development time.

Now, suppose we wanted the output of our digital thermometer to go to a DVM instead. The simple D/A plus the pull-up resistor for the thermistor are all that is required. The software to drive that circuit is just as simple:

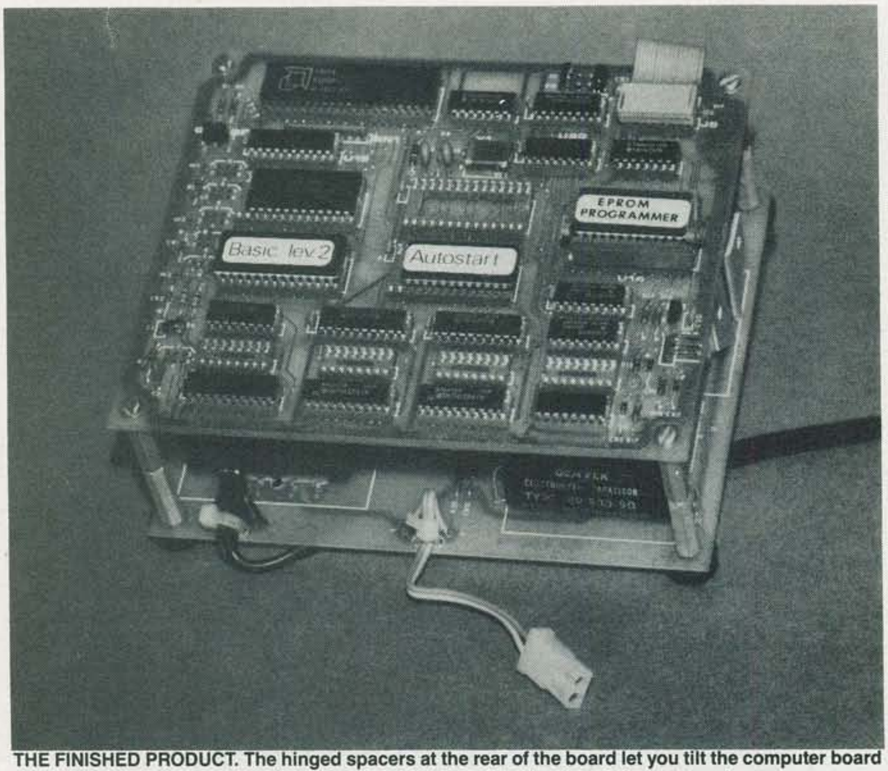
```
700 OUT(16,1): REM A/D CHANNEL 1
710 OUT(32,0): REM START CONVERSION
720 LET B=INP(32): REM D/A OUTPUT VOLTAGE IS STORED IN B
740 REM VARIABLE C CONTAINS DESIRED OUTPUT
750 REM RC IS CONNECTED TO BIT ADDRESSABLE OUTPUT 0
770 IF B<C OUT(0,1): REM CHARGE UP CAPACITOR
780 IF B>C OUT(0,0): REM DISCHARGE CAPACITOR
```

If you wanted to compute the average temperature since the unit was turned on:

```
800 LET D=D+1: REM THE NUMBER OF SAMPLES
810 LET E=E+A : REM THE ACCUMULATED TEMPERATURES
820 LET F=E/D: REM THE AVERAGE TEMPERATURE
```

That illustrates the advantages of smart control over a traditional thermistor/linearizer/meter approach. Although more complex initially, the control computer is unlimited in its adaptability.

Of course, now that you have the temperature information inside the computer, you can use this data to control any of the outputs. Those can control your home heating system etc. **R-E**



THE FINISHED PRODUCT. The hinged spacers at the rear of the board let you tilt the computer board up, giving you easy access to both boards, while keeping the size of the system down.

ORDERING INFORMATION

The following are available from Vesta Technology, Inc., 2849 W. 35th Ave., Denver, CO 80211: KIT 1: Kit of all parts needed to control 7 LS-TTL outputs, monitor 7 inputs, program EPROM's, RS-232 serial port, and 2K RAM (does not include operating system—see below), $99.95; Operating systems contained in ROM: BASIC I operating system, $12.95; BASIC II operating system, $29.95; Forth operating system, $79.95; Assembled, tested, and burned-in control computer with BASIC II operating system, $279; RS-232 cable, $24.95; 2716 EPROM, $6.95; Power-supply/BSR-link kit, including all components, $59.95; Assembled, tested, and burned in power supply, $109. Add $6 for shipping, handling and insurance.