JAMES A. GUPTON JR.

Computer Control for the UNICORN-1 ROBOT

Part 10—If your robot is going to use computer control, here are some thoughts on the subject that will help you in setting up your system.

WITH THE UNICORN-1 ROBOT OPERATING under radio control, what now? Why, *computer* control, of course! This part will deal with that subject, although, because of its complexity, only in general terms.

For those of you already involved with computers—micro or otherwise—much of what will be discussed here may seem elementary. For those who have not yet been exposed to that fascinating area of electronics we will try to keep things as simple as possible.

What will be covered here will be the *concepts* involved in having the actions of a robot determined by an electronic device rather than by a human operator. That's where much of the challenge of computer control comes in.

A human can exercise his judgment—without necessarily having to think about it—and change the robot's actions to meet the circumstances. The computer also has to exercise judgment, but before it can do that it must be taught—or programmed—*how* to make judgments; that involves a great deal of highly detailed programming.

For those of you who are unfamiliar with computers, it is not enough just to connect a computer to the robot and say, "Go ahead . . . do your stuff." Every action must be pre-planned, and, more important, every consequence of every action must be considered and the appropriate reaction prepared.

That is one reason why we will not present specific programs for robot control but will, instead, talk about the way those programs will have to function.

Methods of computer-control

To put it broadly, there are three ways that a computer can be used to control the robot's actions.

The first, and simplest, would substitute a computer, *located outside the robot*, for the command consoles described earlier in this series. That computer would be linked to the robot either by cable or by radio.

The program for that system would be fairly simple and would allow the operator to type in a command, to which the robot would respond. For example, entering "GO" or "G" would cause the robot to move forward; "TURN LEFT" or "L" would cause it to turn to the left, and so on.

That elementary program could be modified to operate with a speech-recognition device—several of which are available for a couple of hundred dollars—to allow the robot to respond to the spoken word. The vocabulary would be limited (but adequate) but the commands would have to be given to the external computer, not to the robot directly.

The second system would be a program, or series of programs, that would command the robot to perform a predefined sequence of actions.

For example, the robot might be instructed to move forward for ten seconds, stop, raise its right arm in a salute, beep its horn, and then turn around and return to its starting position.

Such programs could become very elaborate, but have a major drawback. Unless the robot is equipped to respond to its environment (and, so far, it isn't) any unknown factor that enters the picture could have serious consequences.

Using the program above as an example, suppose that, unknown to you, the robot is facing a brick wall, five feet in front of it. Shortly after the robot begins to carry out the instructions given to it by the computer, it will run smack into that wall! Not only will that interfere with the rest of the program, but it can also cause damage to the robot and, possibly, the wall. Or maybe, instead of a brick wall, there's a person or a piece of furniture in the way. The overall damage—and its consequences—could be considerably more serious.

In any case where the robot is operating without human intervention, provision must be made for the program to be overridden!

Any program of that nature must contain some means for the human supervisor to stop or alter the robot's actions at any time. That is one reason that the "drop-dead" circuit was included on the latch board (Part 9)—one command would activate that circuit and cause the robot to stop in its tracks, should any unforseen circumstance arise.

The third method of computer control, and the most fascinating, involves the robot having its own, on-board, computer. The precautions given for the second method also hold here. We'll talk about that method in more detail shortly.

Interfacing

Whichever method is chosen, the robot must be equipped to respond to (and, perhaps, "talk back" to) the computer. Fortunately, the circuits already being used by the robot are designed with that in mind.

There are two formats that computers can use to output data or to receive it: *parallel* and *serial*. The parallel format is always used by the computer internally.

The unit of information that the computer uses for communication is called a *byte*. A byte is made up of eight bits (*bi*nary dig*its*)—each one either at a logic-"high" or logic-"low" state—and the computer operates on all eight bits at once. Frequently, when a computer is used to operate a printer, the parallel format is used and eight lines are used to connect to the printer—one for each bit of the byte.

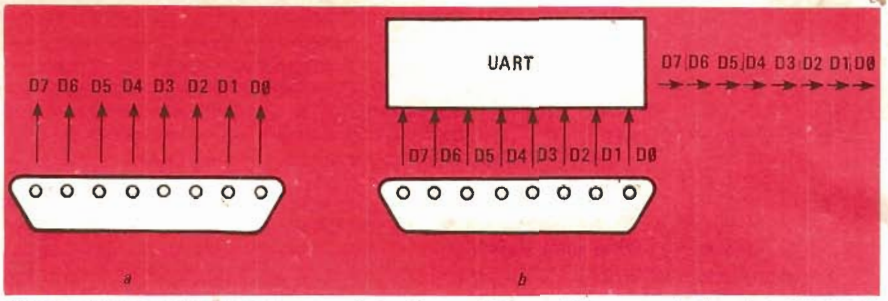


FIG. 83—ALL THE BITS of a byte are sent simultaneously in parallel communications (*a*). A UART (*b*) converts parallel data into serial data for transmission over a single line.

On the other hand, sometimes it is convenient—or even necessary—to transmit computer data using only a single line (by telephone, for example). In that case, the serial format is used. The computer takes each byte and sends it out bit-by-bit, one after the other, indicating the beginning and end of each byte. At the other end, the eight bits are received in the order in which they were sent; when they have all arrived, they are used in parallel. Both of those systems are illustrated in Fig. 83. The device that performs the parallel-to-serial and the serial-to-parallel conversions is known as a UART (*U*niversal *A*synchronous *R*eceiver/*T*ransmitter). UART's would be used if commands were transmitted to the robot by radio.

If you connect your computer to the robot by means of a cable from the computer's parallel port, it would be a good idea to use twice as many lines as necessary (16) and ground every other one. That will help keep electrical noise from getting mixed in with the data.

(For more information on how computers operate see "Your Own Computer" in the October 1980 issue of **Radio-Electronics** and the article on assembly language computers on page 45 of this issue.)

The decoder-, latch-, and relay-driver-boards in the Unicorn-1 use parallel data. Using the same technique as used with the 7402 IC's on the decoder board, any two bits of an eight-bit word (byte) can be NOR'd or NAND'ed to produce a single control bit for the relay-driver board. If you're knowledgeable, more complex and versatile encoding/decoding schemes can be used.

Which computer?

There are two classes of computers that must be considered: those for external use and those that can be mounted on-board the robot.

Almost any computer that has at least one parallel port can be used for the first purpose and it is not our intention to single out one manufacturer's over the other. If you are contemplating buying a computer, refer to the articles mentioned above.

The important thing is that the computer be equipped with a parallel port and that it be flexible enough to meet your needs—present and anticipated. For example, if you are considering using voice control, make certain that there is a speech-recognition board available for your computer.

It should be noted that some computers—such as the Radio Shack *TRS-80* and the Commodore *PET*—do not have parallel ports as such, but that their expansion connectors—frequently used to connect to printers—are actually just that. The thing to look for is eight *data lines*, usually designated "DØ" through "D7." If you have those, you have your parallel port.

You will also want a cassette and/or disk interface to allow you to save programs that you have written for the robot.

One thing you should avoid are inexpensive computers that are actually glorified video games. They generally will not have the facilities you need and it will prove difficult (or impossible) to add them.

The other possibility is a single-board computer that can be mounted in the robot. In addition to a parallel port and cassette interface, that computer must also have a hexadecimal ("hex") keypad for programming, and some kind of LED display, if it is not going to be used together with an external computer. An example of how such a computer would be interfaced to the robot is shown in Fig. 84.

A good computer for the purpose is the *KIM-1*. Unfortunately, that computer was recently discontinued; but you may still be able to find one here and there. Other possibilities include the *SYM-1* (a sort of super *KIM*), the *ELF-II* or the *Explorer/85* (keypad version). Again, refer to the article on page 45. Both the ELF-*II* and the Explorer/85 are manufactured by Netronics, 333 Litchfield Road, New Milford, CT 06776. The SYM-1 is produced by Synertek Systems Corporation, P.O. Box 552, Santa Clara, CA 95052.

Bear in mind that some of those computers may require a power supply other than 5- or 12-volts DC. In that case a power inverter (see Fig. 85) can be used to turn the robot's 12-volt supply into 117 VAC, which the *computer's* power sup-

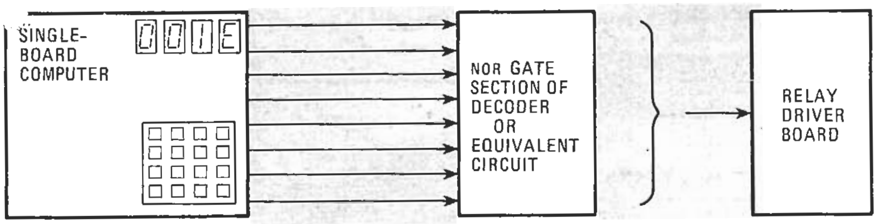


FIG. 84—SINGLE-BOARD COMPUTER can be connected to NOR gate section of latch board or to an equivalent circuit designed to give a single output from a two-bit input. That is only one of many possible schemes.

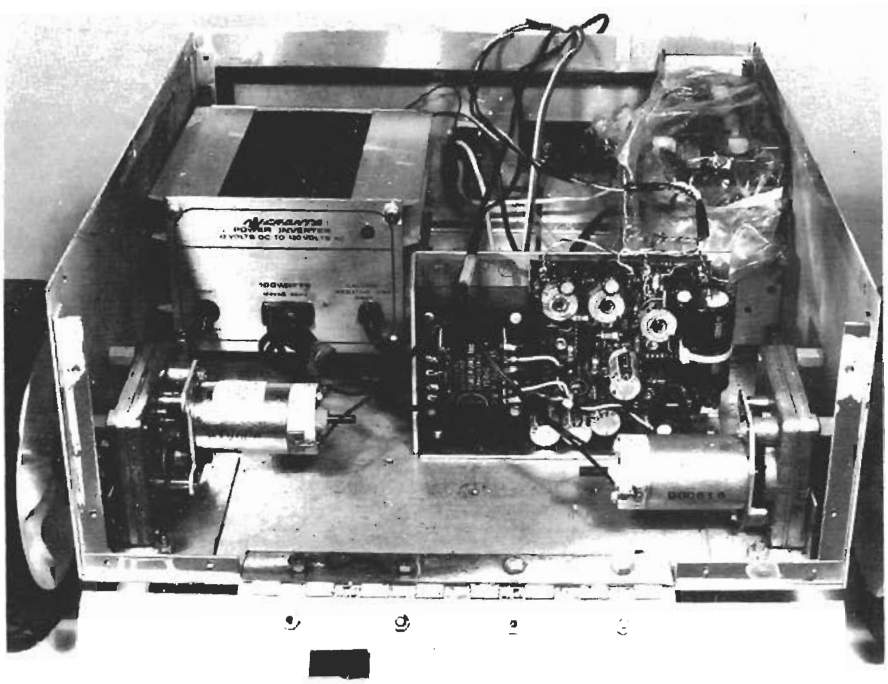


FIG. 85—AN INVERTER (left rear) may be needed if the on-board computer has power requirements other than +5 or +12 volts DC.

ply can then convert readily to its own requirements.

Finally, if you already own a computer but intend to install another in the robot, it would be a good idea to make sure that both computers use the same-type, or compatible, microprocessors. The *KIM-1* and *SYM-1* use the 6502, which is also found in the *Apple II*, OSI *Challenger(s)* and the *PET*, and the Z-80 in the *TRS-80* is compatible with the *Explorer/85's* 8085.

The 1802, used in the *ELF II* and in RCA's *VIP*, is not normally found in larger computers, but that does not mean that an 1802-based single-board computer should not be used in the robot.

The fact that both of your computers use the same microprocessor means that both of them speak the same language, at the microporcessor level. That, in turn, means that you can use your larger computer to develop and debug (troubleshoot) programs to run on the robot's computer and to *download* (transfer from the larger to the smaller computer) those programs, either directly or, if the cassette interfaces are of the same type, from tape.

The programming itself will also be easier, since—assuming that your programs are in machine language and not in BASIC—you will be able to use an assembler, making your work go more quickly and also making it easier to follow the flow of the program.

Programming

As you may have gathered by now, it would be impossible to present computer programs for robot-control, there being so many variables involved.

If you are working with an external computer, you will probably want to work in BASIC or another high-level language, using the OUT command, or its equivalent, to transfer data to the robot.

As mentioned above, the on-board computer will almost certainly have to be programmed in machine language. It's more difficult to work with than BASIC, but it *does* have advantages. Programs take up much less memory space, and also run more efficiently. You may even want to write your "big-computer" programs entirely in machine language through the use of an assembler.

This section has of necessity, been sketchy: after all, even books on the subject have not been able to cover the matter completely.

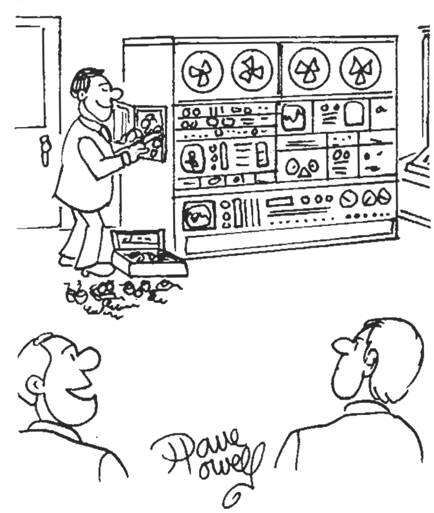
If you are going to use a computer with your robot, we recommend that you do as much supplementary reading as you can. Personal-computer magazines such as Byte magazine and Interface Age have had special issues dealing with robots, and the subject comes up frequently there and in other computer publications. Another good source of information that is often overlooked is your local library.

Todd Loofbourrow's book, *How to Build a Computer-Controlled Robot* (Hayden Publishing Company) contains a number of robot-control programs written for the *KIM-1* (or *SYM-1*) as well as a number of more generalized flowcharts. Much of the information presented there may be adaptable to your robot.

A very good—although rather technical—article on "An Interactive Programming Language for Control of Robots" by Li Chen Wang appeared in the September 1977 issue of Dr. Dobb's Journal of Computer Calisthenics & Orthodontia. It involves a robotic simulation on a computer's video display and its principles could be adapted to control a "flesh-and-blood" robot. (That issue, #18, Volume II, No. 8, is available in limited quantities from: Dr. Dobb's Journal, 1263 El Camino Real, Box E, Menlo Park, CA 94025 for $2.50, postpaid, second class.) It's worth looking into for readers already familiar with computer programming.

In the next part of the Unicorn-1 series we will take a look at sensors. We will discuss sensors in general, and show you some specific examples that can allow your robot—and the computer that controls it—to respond to the world around it.

We would like to hear about how you're doing with your version of Unicorn-1. Write (and send photographs) to: ROBOT UPDATE, **Radio-Electronics,** 200 Park Avenue South, New York, NY 10003. **R-E**



"Charlie's OK at fixing computers. He seldom does any damage that an electronic technician can't repair."