The Scorpion

SOFTWARE OVERVIEW

Harprit S. Sandhu
2505 South Neil Street
Champaign, Illinois 61820

When designing a product, it is necessary to make a number of decisions that are transparent to the final user. Since the Rhino Scorpion is intended as a sophisticated and expandable floor mobile robot, certain criteria had to be kept in mind at all times. Of these, price and performance are the obvious prime considerations, but there are many others of interest to the robot enthusiast.

Since a robot interacts with its environment, we emphasized sensory and environmental controls; the input and output for the robot. In order to have good I/O control, we must select a good I/O chip. One of the best-known sophisticated I/O chips is the 6522. Two 6522s provide 32 I/O lines and four counter/timers. The Scorpion provides expansion room for a third if needed. Now, what can be done with these 32 lines?

First let's consider the floor over which the robot runs. Floors are usually flat, but they can have patterns. If we could detect changes in brightness, we could find and follow edges. However, to really explore the floor, the robot needs two detectors. At this point we have used up a total of three of the 32 I/O lines. One is used to turn on lights to illuminate the floor, and two are used to monitor the resultant brightness.

To create an interesting device, the Scorpion should be capable of making noises. One I/O line is used to control a speaker. For starters, we provide the ability to control the tone and its duration, using two of the timers on the 6522s. We have now used a total of four I/O lines and two timers.

As the robot runs about, it will undoubtedly run into obstructions. To make intelligent decisions, we must determine which part of the robot encountered an obstruction. This will require a minimum of four switches to identify left, right, front, and back collisions. To provide even finer resolution, we can use eight switches (two on each side) to detect left-front, right-rear, etc. This scheme uses up another 8 control lines.

Of course, the robot must move. This motion requires a minimum of two motors to control both speed and direction. We decided to use stepper motors which are each controlled by two control lines. In addition, wheel speed is controlled by using two more timers on the 6522.

We need additional sensory information in addition to the bumper sensors. Vision would be an ideal addition. Unfortunately, conventional vision systems are too expensive. What if we could build an optical scanner that scanned the environment, collected the information, and displayed the information on the video screen? A display of 40 characters by 25 lines would produce 1000 pixels of varying brightness. This should be sufficient for recognizing many simple objects. We could experiment with pattern recognition, perform calculations to enhance contrast, and find edges.

To gather this information, a cadmium/sulfide (CdS) photosensor is mounted at the focal point of a solar cigarette lighter and connected to an oscillator. As the assembly

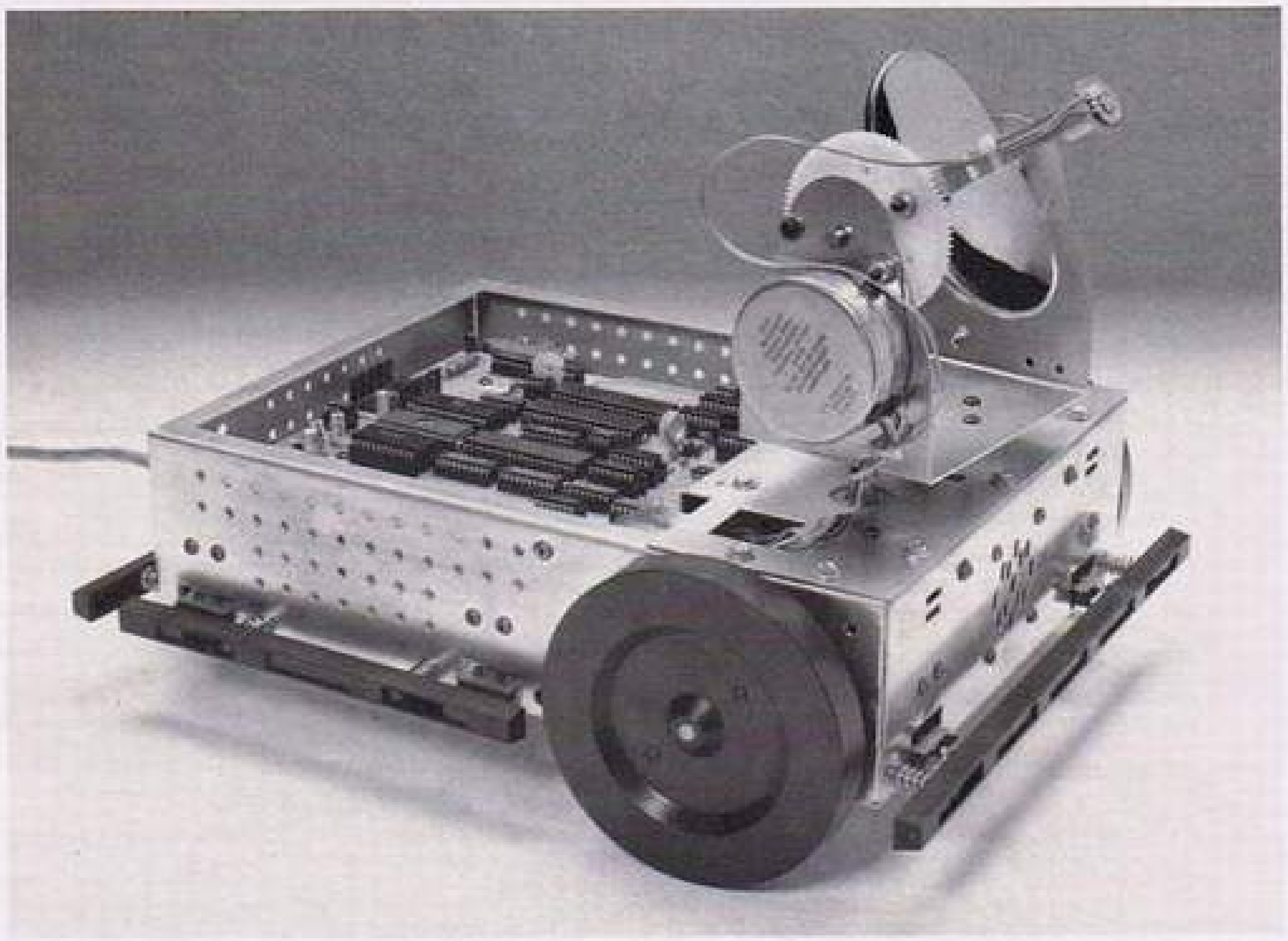

Photo 1. A fully-assembled Scorpion with the cover removed to expose the computer control card. The 0.5 inch on-center holes along the sides and rear of the Scorpion can be used to attach Meccano and Erector set components to the system. The two microswitches behind each bumper and the two-axis optical scanner help form a rugged and reliable robot suitable for experimentation in artificial intelligence and robotic programming. The robot sells in kit form for $660.

About the Author:
Harprit S. Sandhu is the President of Rhino Robots, PO Box 4010, 3402 North Mattis, Champaign, Illinois 61820

INSTRUCTION	MEANING OR PURPOSE
/7M + ## – ##	Start both drive motors
/4L – ##	Start left drive motor
/4R + ##	Start right motor
/2HR	Reset horizontal scanner
/5HM – ##	Move horizontal scanner
/4HS##	Do a horizontal scan
/2VR	Reset vertical scanner
/5VM – ##	Move vertical scanner
/4VS##	Do a vertical scan
/2E#	Control the eyes
/2G#	Control ground tracker lamps
/5Sffdd	Control speaker frequency and duration
/7Waaaa##hhhh....	Write programmable memory locations
/7Raaaa##	Read memory locations
/1R	Reset the system
/3Yhh	Set I/O master byte
/3Zhh	Ignore/Obey the I/O master byte
/8C + ######	Move left motor a specific amount
/8D + ######	Move right motor a specific amount

INSTRUCTIONS THAT HAVE ANSWERS

QUESTION	MEANING,USE	ANSWER
/1B	Bumper status	Bhh
/1S	Scanner data	SHhhhhhh..
/1E	Eye status	E#
/1G	Ground lamp status	G#
/1T	Tracker status	T###
/1M	Motor move status	M – #### + ####
/7Raaaa##	Read memory	Rhhhhhh..
/1Y	Read I/O byte	Yhh
/1X	Read busy byte	Xhh
/1C	Left motor steps to go	C####
/1D	Right motor steps to go	D####
/1P	All parameters	PBhhE#G#T###M – #### + ####
	parameters continued	YhhXhhC####D####

Table 1. Instruction set for Scorpion operating system. All questions are answered and terminated with hexadecimal 0D0D0A. In the instruction set definitions, the following conventions are used: # = single decimal digit; hh = two-digit hexadecimal value between 00 and FF; aaaa = a hexadecimal digit between 0000 and FFFF.

scans the room, the oscillator varies from 13 Hz up to 1300 Hz. The Scorpion control board dedicates eight bits to reading a digital counter and another bit to resetting the counter. Two more motors are used to control the horizontal and vertical scans.

We have now used up a total of 29 bits. Adding two phototransistor eyes uses up two of the last three bits. The last available control bit is used to control the motor power so that we can shut them off completely. This saves a lot of power when we are using batteries and also allows us to turn off the four main motors and turn on the two optional motors without needing more power.

Software Control. One of the problems encountered when designing a small system is the temptation to make the system do everything imaginable. Unfortunately, in a small system, this usually means nothing is developed very well. The ideal is to design a spartan but clever system that the user can modify. Flexibility is the foremost target.

Since part of the Scorpion's operating system is in EPROM, we moved all system constants to programmable memory so they can be modified. This allows you to extend and change the operating system without having to create new EPROMs.

The operating system looks at the programmable memory four times: after system housekeeping, after processing all the system-defined instructions, before the interrupt routine, and at the end of the interrupt routine. You can add your own operating system and extend the system with your own instructions. You can also alter the interrupt routines.

A useful experimental system must be able to talk back to a host computer. Only then can you close the control loop and use powerful machine intelligence procedures. Don't ever overlook the necessity of feedback in any system.

Other Hardware Information. All ideas developed for and on the Scorpion are directly applicable to full-sized mobile robots, machine intelligence investigations, and pattern recognition studies and schemes. The Scorpion provides the hardware necessary to try out your ideas without requiring large amounts of money.

The two-axis optical scanner can discriminate between lights of varying brightness, and provides the ability to perform celestial navigation by assuming that a light bulb at the ceiling is the north star. A seven-bit counter allows the discrimination of up to 127 brightnesses. Adjustable sensitivity is provided. This scanner allows you to collect optical data. You can manipulate the data mathematically to increase contrast and discern edges, images, etc. Optical recognition strategies can be worked out. The scanner resolution is 1.5 degrees per step.

Control Instructions. Instructions are sent to the Scorpion by transmitting data to the port to which the robot is connected. The instruction format is as follows:

- The first byte is always a slash, "/." Whenever the Scorpion sees a "/," it resets the buffer and prepares to receive information.
- The next byte usually tells the Scorpion how many bytes of data to expect. Up to nine bytes of data transmitted in ASCII format can be accepted. A decimal "1" is transmitted as a hexadecimal 31.
- The data itself follows the number. The first byte after the "length of data" byte is an alphabetic character. This usually identifies the type of instruction being sent. The data is always in hexadecimal ASCII format.

Questions. The second set of instructions are *questions* that the Scorpion answers. For example, we can ask the Scorpion for the status of the two ground tracking phototransistors whenever we want. The question format is the same format used for instructions.

The *answers* given by the Scorpion can vary in length from 1 to 300 bytes. The answer format organization follows:

- The first byte identifies the type of information sent. It can be an alphabetic character like "E." In our case "E" represents the eyes.
- The data follows in hexadecimal ASCII format.
- All answer transmissions end in hexadecimal '0D0D0A', representing two carriage returns and a line feed. This can be changed to whatever you want. □

Next Month. *Next month's article, the second in this three part series about the new Scorpion robot, describes the first half of the instruction set in detail.*

Reader Feedback

To rate this article, circle the appropriate number on the Reader Service card.

55	65	75
Excellent	Good	Fair