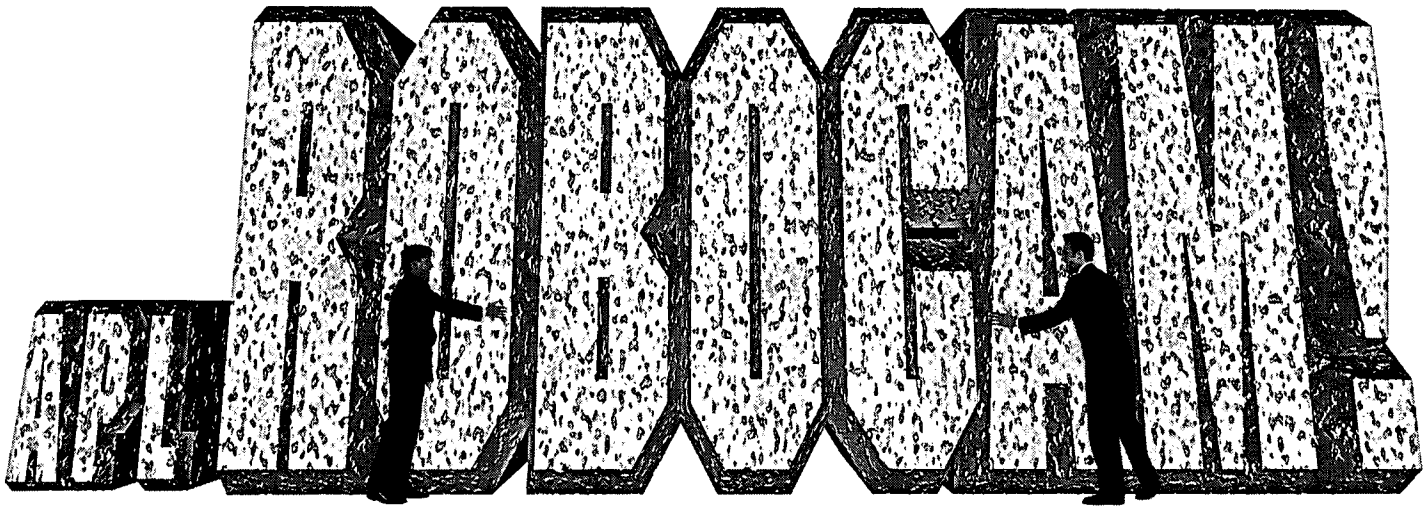
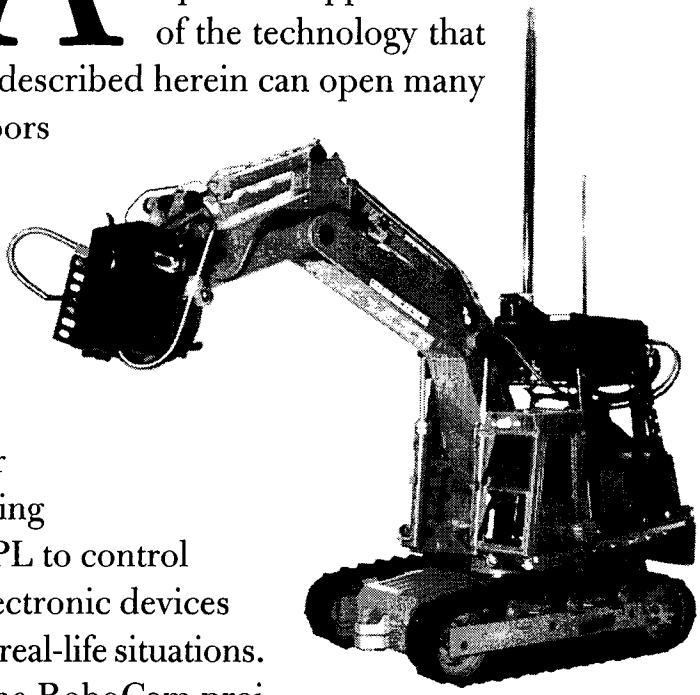
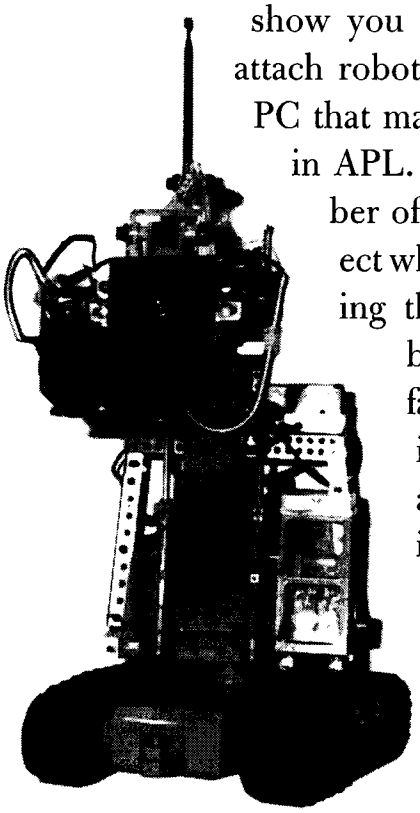


Build your own Voice-Commanded and Video-Equipped APL Robot

—by John Manges
Atlanta, Georgia

ALTHOUGH THIS PROJECT may at first seem like "just a toy," you must realize that the practical applications of the technology that is described herein can open many doors for using APL to control electronic devices in real-life situations. The RoboCam project merges the disciplines of electronics, video, voice-recognition, and APL programming in an easy-to-follow "hands-on" approach.





The object of this article is to show you how to build and attach robotic devices to your PC that may be programmed in APL. There are a number of steps to this project which include selecting the robotic device, building an interface to the PC, adding video feedback, and programming in APL.

My initial criteria was to build a device programmable in APL, voice-operated, capable of sending live video back to the control console, and not cost much to build with easily-obtainable parts.

Rather than building your own robotic device from scratch, it is far easier and cheaper to modify an existing product. Your first step is to purchase a radio-controlled toy car or truck from

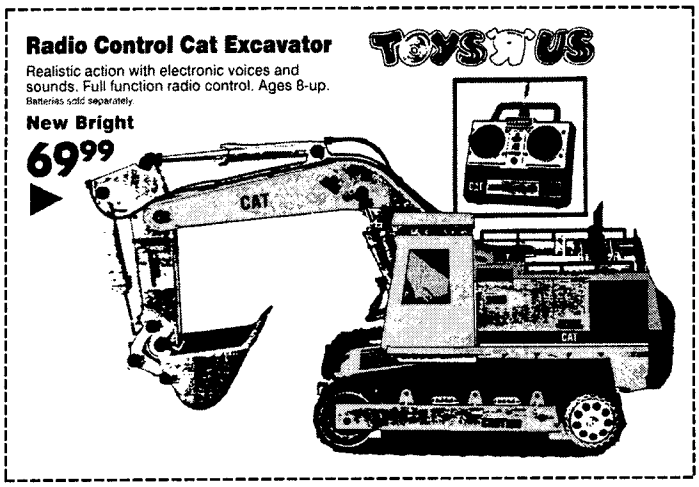


Figure 1

a local toy store. I went to a nearby Toys-Я-Us and purchased a New Bright CAT Excavator, which seemed heavy duty enough to carry extra electronics (Figure 1). I was particularly drawn to the arm and shovel, and felt it would be easy to modify for other purposes. The CAT cost me $49 on sale about a year ago. It is currently retail listed at $69. You needn't get the same radio-controlled toy, as any of them can be interfaced to the computer. However, depending on the amount of extra electronics and batteries you plan to attach to the device, look for units that can handle added weight.

The CAT Excavator itself requires a 9.6-volt nicad battery pack, and the remote controller requires a single 9-volt battery. The transmitter produces about 100-milliwatt output at 27 MHz.

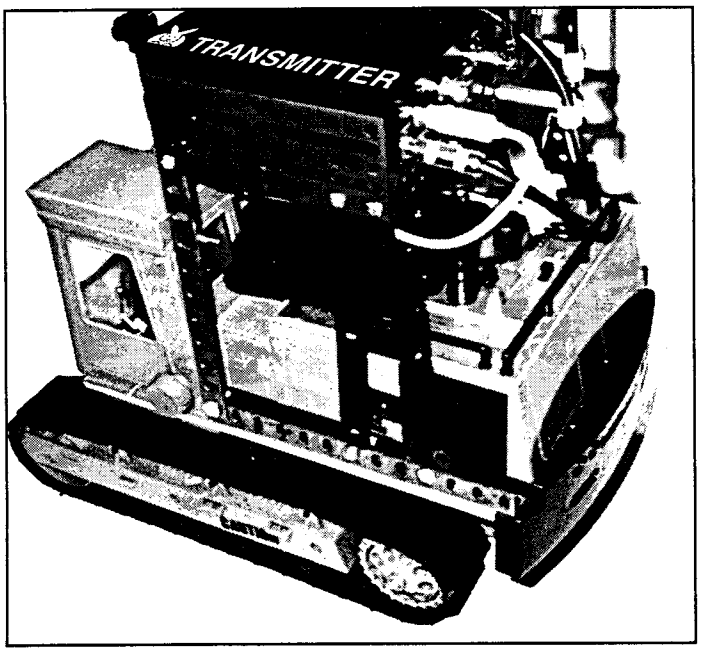


Figure 2

This gives the robot an operating range of about 100 yards. Higher priced and better quality transmitters and boosters can expand the operating range up to a quarter mile.

On my CAT Excavator, I built a platform out of Erector Set girders onto the top of the base unit (Figure 2). This gave me an area to mount the video transmitter and extra batteries. I had to position the upper rails so that they would not interfere with the operation of the arm. I then removed the scoop from the end of the arm. The pins that hold most of the moving parts of the arm are simply pushed together, and removal and installation is a breeze. To the end of the arm, I then created a floating cage, made of coat-hanger wires and more Erector Set pieces (Figure 3). This floating cage then holds a small black-and-white video

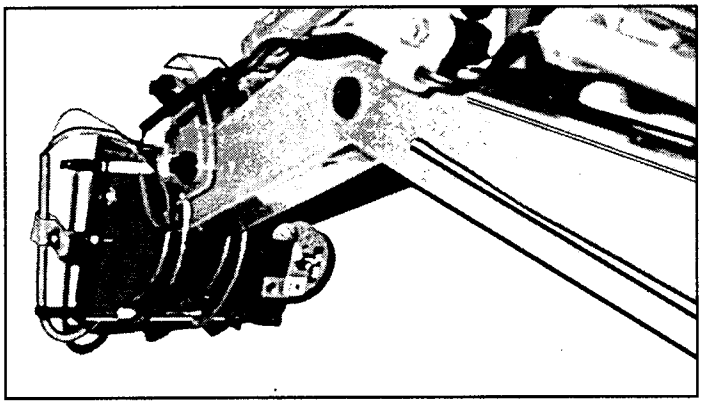


Figure 3

camera. The camera runs about ten hours on a standard 9-volt battery, which is stored next to the camera on the floating cage. The camera is a "PC-3" miniature video camera. It cost $189.95 from Supercircuits, 13015 Debarr Drive, Austin, Texas, 512-335-9777. The case for the PC-3 costs an extra $19.95, but it is worth it. The PC-3 is extremely light (1.32 ounces) and very small (1⅞" x 2⅞" x 1"), making it perfect for robotic applications. The picture quality is quite good.

The cage allows the camera to swivel with the arm from a direct looking down position to an angle looking up. The video send wires from the camera were put into a flexible spring and trailed across the top of the arm to the rear platform.

I then purchased a VCR rabbit for $45 from a local building supply store. Both the transmitter and receiver work off transformers that deliver 18 volts. I attached the transmitter to the robot's rear platform and plugged the camera feed into it (as shown here in Figure 4).

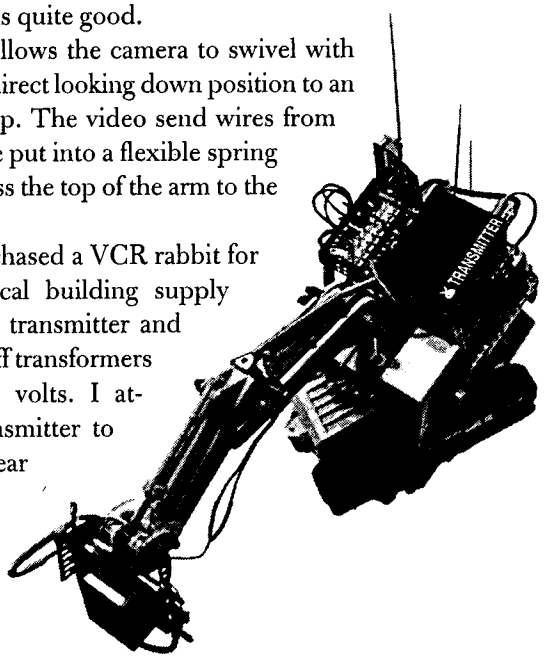


Figure 4

For the 18 volts required to run it, I purchased two 9.6-volt nicad packs, the same type that the robot uses, and hooked them in series. Two standard 9-volt batteries in series would only last a few minutes.

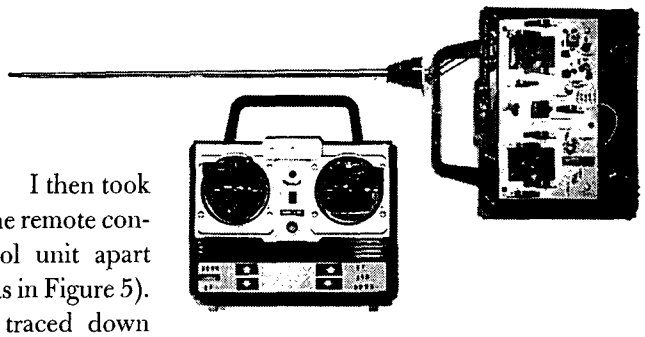


Figure 5

I then took the remote control unit apart (as in Figure 5). I traced down the contacts on the unit and determined which ones caused the actions to occur. My initial attempt was to use switching transistors to switch the contacts from the computer's parallel port, but found that the manufacturer had reverse-biased the control transistors, meaning that for one direction the electricity flowed plus to minus, and for the opposite direction, the electricity flowed minus to plus. When hooking switching transistors together with common emitters (ground), the reverse-biasing caused all sorts of problems. So I changed my logic to use infra-red light emitting and collecting diodes. What I essentially did was create opto-isolators. I stuck each pair into a tube of heat shrink, facing each other, and then shrank them (Figure 6). I mounted these onto a PC board, along with dropping resistors and attached RS-232

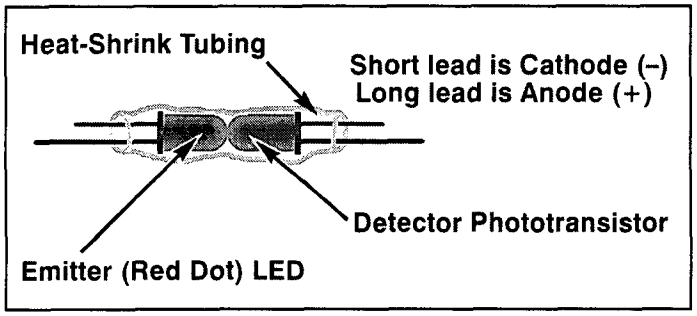


Figure 6

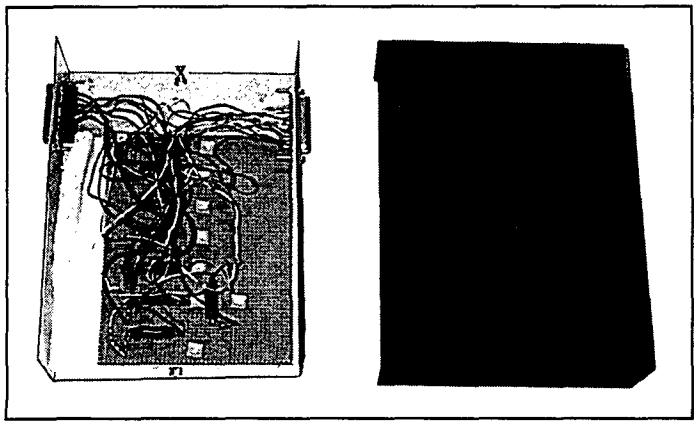


Figure 7

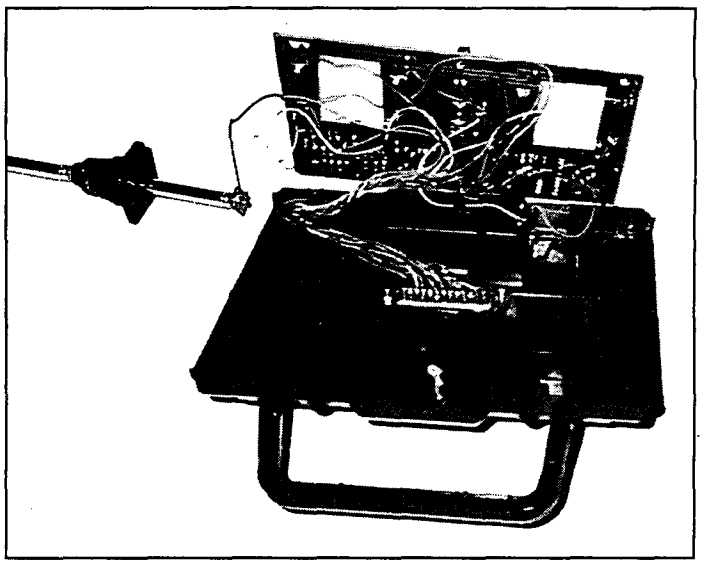


Figure 8

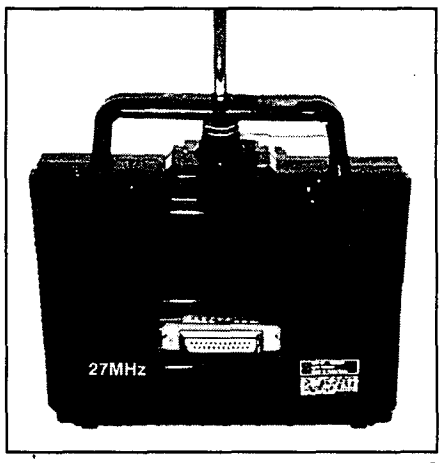


Figure 9

connectors to the ends. The entire assembly was then put into a project box, and the RS-232 connectors were mounted to the outsides (Figure 7). I then installed another RS-232 connector into the back plate of the remote control, and attached wires from the connectors to the switch contacts (Figures 8-9). Then using two printer cables, I attached the remote control to the project box, and the project box to the parallel port of my computer (Figure 10).

In order for the computer to recognize that the device hooked to the parallel port is active, a +5-volt signal must be sent into the port at the Printer Ready connection. Rather than have to deal with a separate power supply or batteries to deliver the printer ready signal, I used the signal from my existing printer. I accomplished this by taking a printer "*A/B*" switch and adding a SPST switch to it. My *A* position is for the printer, my *B* position is for the robot. The extra switch,when engaged, bridges the "printer ready" signal (Pin #13) from *A* to *B*.

This completes the necessary connections from the parallel port through the robotic device. The next step is the voice interface and APL programming.

For the Voice interface, I used the Covox Voice Blaster software for Sound Blaster and compatible boards. I set up a group called RoboCam that contains the actions for the device and some other control features. I created a chart of commands that in turn call an APL function named *AUTOPOST* (Figure 11).

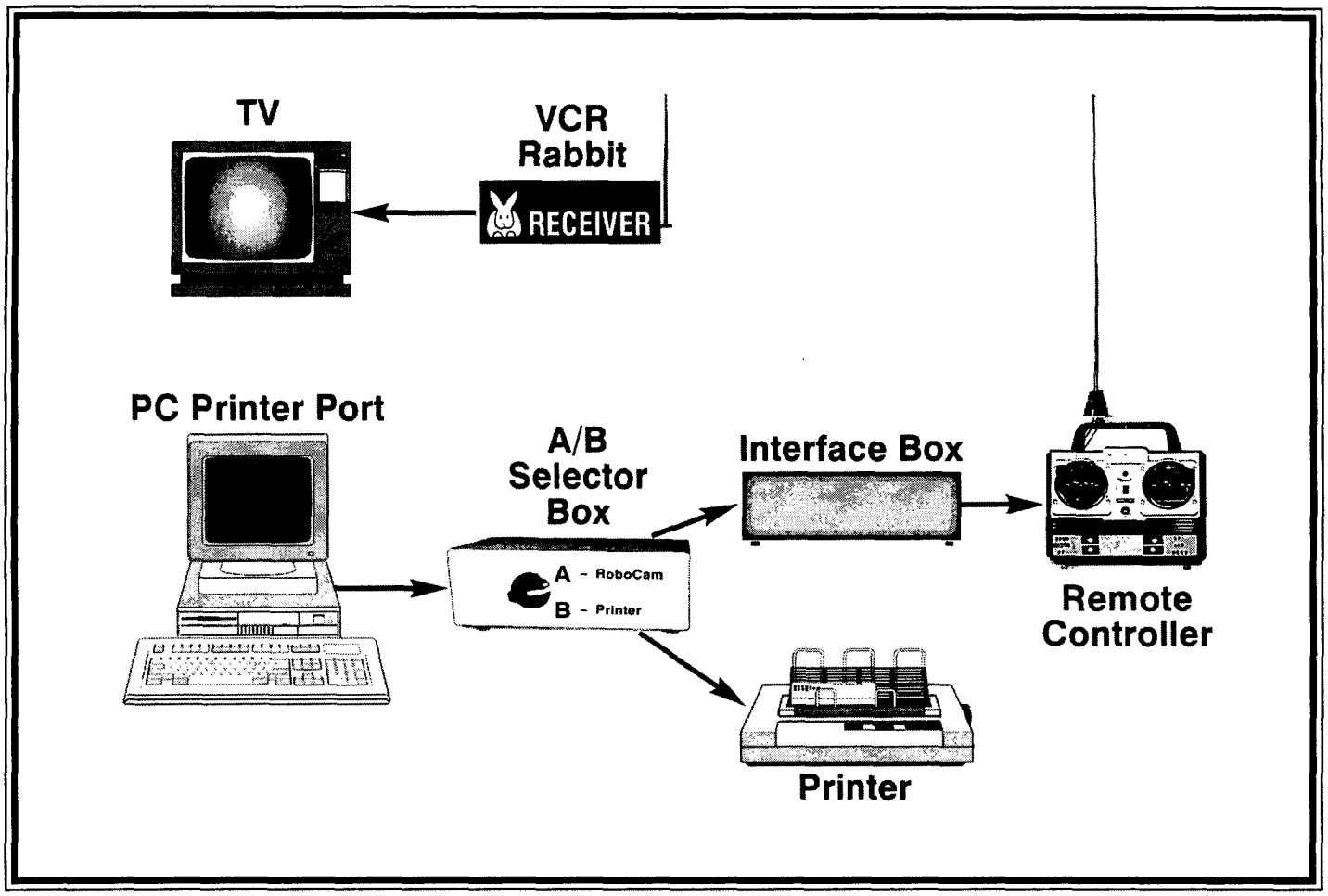


Figure 10

```
    ∇ AUTOPOST RP;⎕IO;X
[1]   ⍝ By: John Manges
[2]   ⍝ Des: Posts ⎕AV[RP] to Parallel Port using Processor 80
[3]    ⎕IO←1
[4]    X←80 ⎕SVO 'CTLP'
[5]    X←⎕SVO 'CTLP'
[6]    CTLP←(⎕AF 255), ⎕AV[RP]      ⍝ Post values to port
[7]    →(~RP∈2 3 33 129)/0          ⍝ Check for limited actions
[8]    X←⎕DL 1                      ⍝ Delay for 1 second
[9]    CTLP←(⎕AF 255),⎕AV[1]        ⍝ Full stop
    ∇
```

Figure 11

All control of the robot is done through the parallel port. The parallel port is intended primarily for connecting the computer to printers and plotters, but with APL and the LED/Phototransistor interface, it can be used to control the robot. The port has a decimal address of 888 (or 0378H hex), and contains 25 pins at the connector. You can verify the address of your parallel port by using the utility DEBUG. At the DEBUG prompt type "D 40:0" and press Enter. This step displays the hexadecimal values of the active I/O port addresses, first serial and then parallel. To exit DEBUG, type "Q" and then press Enter. The functions of the 25 pins at the connector are shown in the diagram in Figure 12. The eight data output lines trigger the LED's in the interface, which in turn activate the phototransistors that turn on each circuit in the remote controller. All eight switches can be set by a single posting to the parallel port address. Each data bit is latched, meaning it stays on until it has been explicitly turned off. Some commands, such as Arm Up, require that the actions be limited in duration and be turned off when finished. The *AUTOPOST* function simply posts the appropriate value into the address to turn on the corresponding actions (Figure 13).

I also added a command to the voice macro to say the action it is now performing. This helps to identify any problems in the cabling or connections when testing, and gives a more personal

Pin #	PC Type	Description
1	Output-	Strobe
2	Output+	Data Bit 0
3	Output+	Data Bit 1
4	Output+	Data Bit 2
5	Output+	Data Bit 3
6	Output+	Data Bit 4
7	Output+	Data Bit 5
8	Output+	Data Bit 6
9	Output+	Data Bit 7
10	Input-	Acknowledge
11	Input+	Busy
12	Input+	Printer Error/Paper End
13	Input+	Printer Ready/On line
14	Output-	Line Feed after Carriage Return
15	Input-	Printer Error
16	Output-	Initialize Printer
17	Output-	Select/Deselect Printer
18	Ground-	Data Bit 0 Ground
19	Ground-	Data Bit 1 Ground
20	Ground-	Data Bit 2 Ground
21	Ground-	Data Bit 3 Ground
22	Ground-	Data Bit 4 Ground
23	Ground-	Data Bit 5 Ground
24	Ground-	Data Bit 6 Ground
25	Ground-	Data Bit 7 Ground

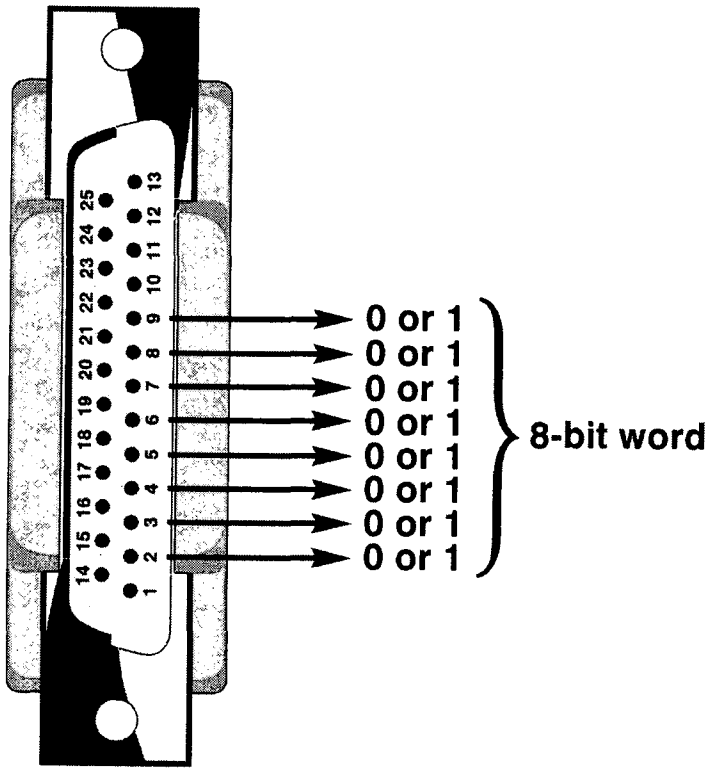


Figure 12

```
Arm Down ..... AUTOPOST 129
Arm Up ..... AUTOPOST 33
Camera Down ..... AUTOPOST 3
Camera Up ..... AUTOPOST 2
Full Ahead ..... AUTOPOST 73
Left Forward ..... AUTOPOST 9
Left Reverse ..... AUTOPOST 5
Reverse ..... AUTOPOST 21
Right Forward ..... AUTOPOST 65
Right Reverse ..... AUTOPOST 17
RoboCam ..... )LOAD ROBOCAM
Spin Left ..... AUTOPOST 69
Spin Right ..... AUTOPOST 25
Stop ..... AUTOPOST 1
```

Figure 13

touch to the robot. A testing routine I have included is called "*DANCE*". *DANCE* will perform each of the thirteen commands and then stop. Another testing routine is *SPIRAL*, which attempts to spiral the computer into smaller and smaller circles and then go into a spin. Please note that on occasion, too sharp of a turn can pull the tractor tread off the base wheels of the unit.

To increase functionality of the robot, you can add a wireless microphone for the voice input, and then link the video output into the cable system of your house. This approach will let you go to any TV on your cable system and be able to see and direct the camera from any part of your house.

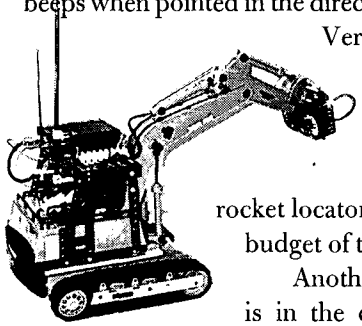
You can also attach a wireless microphone directly to the arm of the robot, and by cannibalizing an old baby monitor, you can send the Sound Blaster PC output back to the robot. This gives the appearance that the robot is listening and talking to you without any other computer support. Remember though, that for every additional radio controlled device added, you lose distance in the range in which it may be operated.

Building a RoboCam or similar device can be a rewarding experience. Please note that RoboCam is simply an experimental device, and is not intended for commercial use. If you build a similar device, it must be labeled "Experimental," in accordance with FCC regulations. Contact your local Amateur Radio club or Model Aircraft clubs for more information on FCC regulation of radio-controlled devices.

In summary, I have shown you the basics in constructing an interface between APL and a robotic device. The APL programming is very simple, and the interface requires only a few hours of tinkering and soldering to build it. Consider this project as a "first-step" upon which you can add more sophisticated ap-

proaches as your expertise permits. The best part is that the entire RoboCam device costs under $500 to build. If you eliminate the video portion, you can cut the cost in half.

Further development of RoboCam could include digitization and image-processing of the video images that in turn cause a programmed action to occur, such as centering upon a target, or identifying a battery-charging station. Another approach could be stationary infra-red emitters in a room that allow the robotic device to triangulate to determine position within the room. I recently saw a radio transmitter and locator at Toys-Я-Us that is used for model rockets. The transmitter goes into the nose cone, and emits a signal using battery power. A hand-held locator beeps when pointed in the direction of the nose-cone transmitter.



Very useful for locating a rocket that has drifted out of sight, but even more useful for usage as a "homing" signal for a roving robot. The price for the model rocket locator was about $25—well within the budget of the home experimenter.

Another use for the RoboCam interface is in the controlling of TTL-compatible devices or chips. Radio Shack recently announced a sale on its sound digitizing chip, which is TTL compatible (programmable by 8-bit words from the interface). The chip saves up to 20 seconds of sound, and can be cascaded for longer messages. The message remains even when the EEPROM chip is powered off. It would be quite easy to write an APL function to control the operations of the chip. Just think, a mock "soundboard" for your PC for under $20. If your robot is wireless-microphone-equipped back to a voice-recognition processor, you could add command-encoded sound chips into destinations triggered by the robot's proximity. Or, say you wanted to change the voice commands to be more like musical tones, so the robot could be commanded only by prerecorded sound chips, and would ignore regular voices.

Whatever the application, it is hoped you will use this information to experiment on your own, and to help promote the use of APL via projects like RoboCam. The examples shown are for APL2/PC running in DOS on a 486/33, but will work with a little modification in other dialects of APL. I am currently working on a much larger version of RoboCam which will try to overcome some of the limitations that this prototype encountered. ■

John Manges, a.k.a. "doc," is President of the South East APL Users' Group. He can be reached at "SEAPLdoc@AOL.COM".

Parts List for RoboCam

Qty	Approx. Cost	Description
1	$50	New Bright CAT Excavator
8	16	Infrared emitter/detector (RS Cat. No. 276-142)
8	2	270 ohm resistors (RS Cat. No. 271-1314)
2	6	25-pin RS-232 male plugs
1	3	25-pin RS-232 female plug
1	4	3' RS-232 printer cable (PC to interface)
1	12	18' RS-232 printer cable (interface to controller)
1	5	Printer A/B switch box
1	2	SPST mini-toggle switch
1	4	6" x 8" project box
1	1	Package of black heatshrink tubing
1	1	PCB board, multipurpose
3	16	9.6-volt NiCad battery packs
	1	#22 single or stranded gauge wire
	–	Erector set girders or equivalent
1	210	PC-3 Video camera and case
1	45	VCR Rabbit transmitter and receiver
2	1	RCA phone jacks, male
	39	Covox Voice Blaster
	$418	Total approximate cost to build RoboCam

Command Number	Hex Bit Positions 1 2 3 4 5 6 7 8	Decimal Number	Command Action	Limit Duration	Voice Command Mnemonic
1	0 0 0 0 0 0 0 1	1	Camera Up	Y	*"CAM UP"*
2	0 0 0 0 0 0 1 0	2	Camera Down	Y	*"CAM DOWN"*
3	0 0 0 1 0 1 0 0	20	Full Reverse	N	*"REVERSE"*
4	0 1 0 0 0 1 0 0	68	Spin Left	N	*"LEFT SPIN"*
5	0 0 0 0 0 1 0 0	4	Left Reverse	N	*"BACK LEFT"*
6	0 0 0 1 1 0 0 0	24	Spin Right	N	*"RIGHT SPIN"*
7	0 1 0 0 1 0 0 0	72	Full Ahead	N	*"AHEAD"*
8	0 0 0 0 1 0 0 0	8	Left Forward	N	*"GO LEFT"*
9	0 0 0 1 0 0 0 0	16	Right Reverse	N	*"BACK RIGHT"*
10	0 0 1 0 0 0 0 0	32	Arm Up	Y	*"LIFT ARM"*
11	0 1 0 0 0 0 0 0	64	Right Forward	N	*"GO RIGHT"*
12	1 0 0 0 0 0 0 0	128	Arm Down	Y	*"DROP ARM"*
13	0 0 0 0 0 0 0 0	0	Full Stop	N	*"HALT"*

APL functions for RoboCam

```
    ∇ DANCE
[1]  ⍝ By: John Manges
[2]  ⍝ Des: Performs a test of all options with sound
[3]  ⍝ Ver: This version is for non-Voice-Blaster usage
[4]  ⍝      and uses the SPUTTER MONITOR as provided in
[5]  ⍝      The diskette accompanying the book,
[6]  ⍝      SOUND BLASTER: The Official Book, 1993
[7]  ⍝      McGraw-Hill, ISBN 0-07-881907-5
[8]  ⍝ Etc: Note that the sound files are .VOC format
[9]   HOST 'SPUT C:\SBDISK\BMASTER\CAMUP.VOC'
[10]  AUTOPOST 2                          ⍝ CAMERA UP
[11]  HOST 'SPUT C:\SBDISK\BMASTER\CAMDN.VOC'
[12]  AUTOPOST 3                          ⍝ CAMERA DOWN
[13]  HOST 'SPUT C:\SBDISK\BMASTER\REVERSE.VOC'
[14]  AUTOPOST 21                         ⍝ FULL REVERSE
[15]  HOST 'SPUT C:\SBDISK\BMASTER\SPINLF.VOC'
[16]  AUTOPOST 69                         ⍝ SPIN LEFT
[17]  HOST 'SPUT C:\SBDISK\BMASTER\SPINRV.VOC'
[18]  AUTOPOST 5                          ⍝ LEFT REVERSE
[19]  HOST 'SPUT C:\SBDISK\BMASTER\SPINRT.VOC'
[20]  AUTOPOST 25                         ⍝ SPIN RIGHT
[21]  HOST 'SPUT C:\SBDISK\BMASTER\FULLAHD.VOC'
[22]  AUTOPOST 73                         ⍝ FULL AHEAD
[23]  HOST 'SPUT C:\SBDISK\BMASTER\LEFTFW.VOC'
[24]  AUTOPOST 9                          ⍝ LEFT FORWARD
[25]  HOST 'SPUT C:\SBDISK\BMASTER\RIGHTRV.VOC'
[26]  AUTOPOST 17                         ⍝ RIGHT REVERSE
[27]  HOST 'SPUT C:\SBDISK\BMASTER\ARMUP.VOC'
[28]  AUTOPOST 33                         ⍝ ARM UP
[29]  HOST 'SPUT C:\SBDISK\BMASTER\RIGHTFW.VOC'
[30]  AUTOPOST 65                         ⍝ RIGHT FORWARD
[31]  HOST 'SPUT C:\SBDISK\BMASTER\ARMDN.VOC'
[32]  AUTOPOST 129                        ⍝ ARM DOWN
[33]  HOST 'SPUT C:\SBDISK\BMASTER\STOP.VOC'
[34]  AUTOPOST 1                          ⍝ FULL STOP
[35] ⍝ WHEN FINISHED AND STOPPED, SAY 'WORKING'
[36]  HOST 'SPUT C:\SBDISK\BMASTER\WORKING.VOC'
    ∇
```

```
    ∇ DANCE2;X
[1]  ⍝ By: John Manges
[2]  ⍝ Des: Performs a test of all options
[3]  ⍝ Ver: This version is for Voice Blaster usage
[4]   AUTOPOST 2                          ⍝ CAMERA UP
[5]   AUTOPOST 3                          ⍝ CAMERA DOWN
[6]   AUTOPOST 21                         ⍝ FULL REVERSE
[7]   X←⎕DL 3
[8]   AUTOPOST 69                         ⍝ SPIN LEFT
[9]   X←⎕DL 2
[10]  AUTOPOST 5                          ⍝ LEFT REVERSE
[11]  X←⎕DL 1
[12]  AUTOPOST 25                         ⍝ SPIN RIGHT
[13]  X←⎕DL 2
[14]  AUTOPOST 73                         ⍝ FULL AHEAD
[15]  X←⎕DL 4
[16]  AUTOPOST 9                          ⍝ LEFT FORWARD
[17]  X←⎕DL 1
[18]  AUTOPOST 17                         ⍝ RIGHT REVERSE
[19]  X←⎕DL 1
[20]  AUTOPOST 33                         ⍝ ARM UP
[21]  AUTOPOST 65                         ⍝ RIGHT FORWARD
[22]  X←⎕DL 1
[23]  AUTOPOST 129                        ⍝ ARM DOWN
    ∇
```

```
    ∇ SPIRAL;A;I;CTLP;X
[1]  ⍝ By: John Manges
[2]  ⍝ Des: Performs a test of all steering controls
[3]  ⍝     : by travelling in a Right Spiral formation
[4]  ⍝     : in ever smaller circles until ending in a
[5]  ⍝     : spin to the right, then to the left.
[6]  ⍝ Etc: You may need to adjust ⎕DL to suit your CPU
[7]   X←80 ⎕SVO 'CTLP'
[8]   X←⎕SVO 'CTLP'
[9]  ⍝ 69=SPIN LEFT   25=SPIN RIGHT
[10] ⍝ 73=FULL AHEAD   9=LEFT FORWARD
[11]  A←10 4⍴73
[12]  A[⍳3;4]←9
[13]  A[3+⍳3;3 4]←3 2⍴9 0
[14]  A[6+⍳3;2 3 4]←3 3⍴9 0 0
[15]  A[10;]←25 9 26 9
[16]  A←,A
[17]  A←(A≠0)/A
[18]  I←0
[19] L1:→((I←I+1)>⍴A)/0
[20]  CTLP←(⎕AF 255),⎕AV[A[I]]
[21]  X←⎕DL 1.3
[22]  →L1
    ∇
```

```
    ∇ GETSTATUS;A;B;C
[1]  ⍝ By: John Manges
[2]  ⍝ Des: Returns Parallel Port Status
[3]  ⍝     : used in the RoboCam interface
[4]   →(0=⎕NC 'CTLP')/L1
[5]   CTLP←0 2
[6]   B←CTLP
[7]   C←(8⍴2)⊤B
[8]   A←8 12⍴' '
[9]   A[1;]←'Not Busy    '
[10]  A[2;]←'Acknowledge '
[11]  A[3;]←'Out of Paper'
[12]  A[4;]←'Selected    '
[13]  A[5;]←'I/O Error   '
[14]  A[6;]←'Not Used    '
[15]  A[7;]←'Not Used    '
[16]  A[8;]←'Time Out    '
[17]  ⎕←(' 0 '⍕8 1⍴C),A
[18]  →0
[19] L1:STOP ⎕←'CTLP VARIABLE NOT SHARED. RUN SHARE80'
    ∇
```

Extra Credit

Project Corner:

Cut on the dotted line, to make your own pogs.