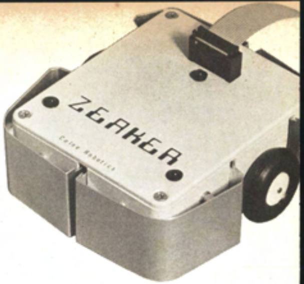
ZEAKER MICRO-ROBOT

PART 2 **DAVID BUCKLEY**



IN the Control Station the holder for the four C-cell Nicads is bolted to the bottom of the box (actually the lid, but the box is used upside down), offset towards one side to allow room for the power/charge lamp. The Nicad supply on/off switch and indicator l.e.d. are fitted to the front of the box. The 3·5mm jack socket for the ZX81 power supply and 2½ foot link to the ZX81 are fitted to the rear.

The three 16 way ribbon cables (two to the computer interface board and the umbilical to Zeaker) leave the box through cut outs in what is now the lid, these should be of a size such that the ribbon cable is just clamped when the box is closed.

CONTROL STATION CIRCUIT DESCRIPTION

The circuit (Fig. 1) can be divided into two distinct parts: the computer READ port and the computer WRITE port. In the prototype both these ports are memory mapped at address 35000 decimal.

Dealing first with the READ port. Data lines D0 to D7 are normally held high by resistor pack IC3. Lines D6 and D7 terminate on pads inside Zeaker and are not used hence D6 and D7 are always high. The remaining lines D0 to D5 terminate at the insulated pillars set into the sides of Zeaker and indicate the state of the tactile sensor switches, a low data line implying that Zeaker is touching something (see table of sensor codes). All the fenders are connected to 0V and on impact with an obstacle a fender will move in and make contact with one or more pillars, hence shorting the respective data lines to 0V.

Turning now to the WRITE port. D0, D1 control the port drive motor; D1, D2 control the starboard drive motor; D4 the pen; D5 the lights; and D6 and D7 the horn.

One end of the port drive motor goes to the centre tag of the battery and the other to the junction of TR3 and TR4. Turning on TR3 will cause the motor to run forwards and turning on TR4 will cause it to run in reverse.

D0 high is port motor forwards and D1 high is port motor reverse. IC1a and IC1b are wired as a set reset latch which ignores the condition D0 high and D1 high, so preventing destruction of TR3 and TR4.

When D0 goes high the output of IC1a goes low, so turning on TR2 which turns on TR3, when D0 is low TR2 is held off by R2 and R3. TR1 is normally held on by R1 and R4 and so shorts TR4 base to earth but when D1 goes high the output of IC1b goes low and turns off TR1, TR4 is now able to turn on by base current through R5.

The operation of the starboard motor is similar except that forward and reverse are switched around so TR8 on gives starboard motor forwards. This evens out battery drain when going forwards or backwards.

Data line D4 high turns on TR9 which turns on TR10, switching on the solenoid which lowers the pen. The l.e.d. D3 provides indication that the solenoid is activated. Diode D4 is to short out the inductive high reverse voltage when current through the solenoid is halted by TR10 turning off.

Data line D5 high turns on TR11 which turns on TR12, lighting D1 and D2, the red and green l.e.d.s.

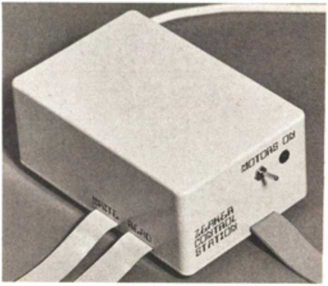
Data lines D6 and D7 control the horn, via the reset lines of IC2 which is wired as two astables. The frequency components R20, R21, C2 and R22, R23, C5 were chosen by experiment so that the tones sounded right when both are on together. C3 and C4 are the usual blocking capacitors.

CONTROL STATION PCB

The p.c.b. for the control station is shown in Fig. 2 with the component layout shown in Fig. 3. The resistors by the 556 are mounted on end but the others are mounted flat. There are a number of wire links to be soldered in place and these are best done with insulated wire. None of the transistors need heatsinks and are all mounted vertically. The three 16 way flying leads can be soldered directly into the p.c.b. but it is easier to use headers on the leads and plug them into the p.c.b.

CONTROL STATION CHECKOUT

The easiest way is to plug the read and write cables into the interface board, plug the ZX81 power supply into the back of the control station and plug the flying power lead



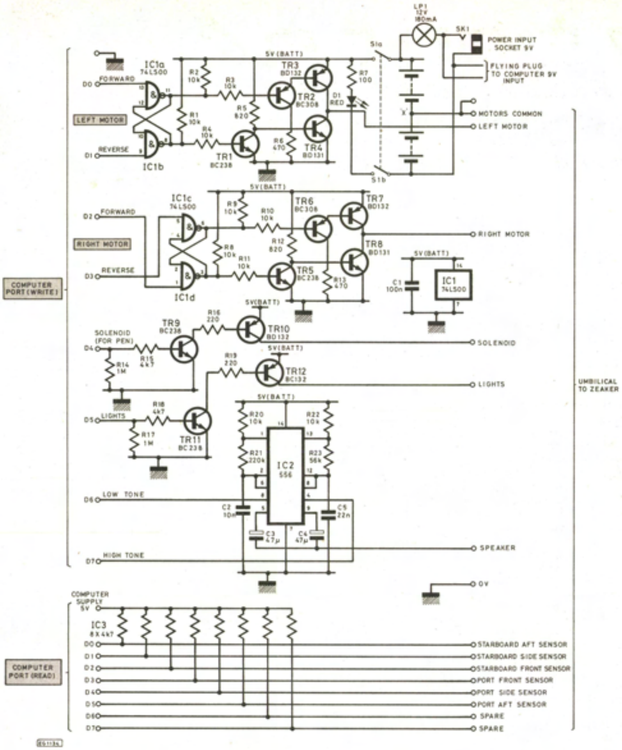


Fig. 1. Circuit diagram of the Control Station

into the ZX81 9V socket and check that the computer still works. The computer is only connected to the control station and Zeaker by the 0V line and the data lines from the buffer chips on the interface board so this shouldn't be a problem.

Plug the umbilical into Zeaker and POKE the interface board port with zero; this turns off all the outputs. Now switch on the 5V supply and nothing should happen. POKE the port with the various control codes (Table 1) and check that Zeaker functions. When any of the BD power transistors are turned on there should only be about 0·5 volt between collector and emitter and about 0·1 volt between collector and emitter of their driving transistors.

PROGRAMMING ZEAKER

To program Zeaker all that is needed is a computer with an 8-bit output latch and an unlatched input port with 6 or 8

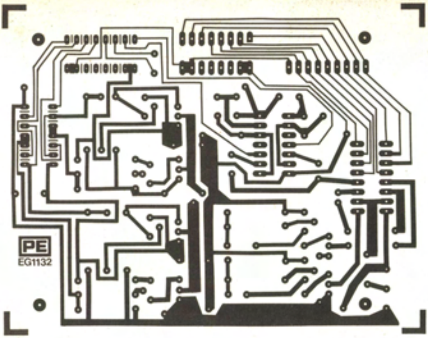


Fig. 2. P.c.b. design

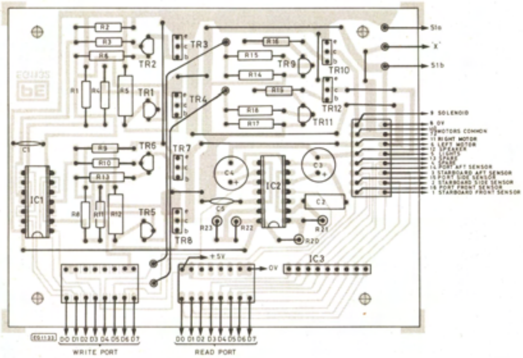


Fig. 3. Component layout

bits (6 for the unexpanded Zeaker). The particular way that these are available will depend upon the microcomputer and interface used. The signal allocation for the output and input connector leads are shown in Fig. 5.

Writing (POKEing for a memory mapped port) a word to the port sets the respective bits of the output latch. Reading (PEEKing) the port will return the status of the bump sensors.

A list of the control codes and their effect is given in Table 1. Any combination of control codes can be written to the port and Zeaker will be controlled by their combined effect e.g. writing a 1 will set the port motor to forwards and writing a 4 will set the starboard motor to forwards, hence writing 5 (=4+1) will set both motors on forwards. Writing 37 (=5+32) will set both motors on forwards and also switch on the lights.

COMPONENTS . . .

CONTROL STATION

Resistors

R1,R2,R3,R4,R8,R9,R11, R19,R20,R22	10k (10 off)
R5,R12 ½W 10% carbon	820 (2 off)
R6,R13 ½W 5%	470 (2 off)
R7	100
R14,R17 ½W 2%	1M (2 off)
R15,R18 ½W 5%	4k7 (2 off)
R16,R19 ½W 2%	220 (2 off)
R21 ½W 5%	220k
R23 ½W 5%	56k

All resistors ¼W 5% carbon except where otherwise stated

Capacitors

C1	100n ceramic disc
C2	10n ceramic disc
C3,C4	47µ 16V elect (2 off)
C5	22n ceramic disc

Semiconductors

D1	red l.e.d.
TR1,TR5,TR9,TR11	BC238 (4 off)
TR2,TR6	BC308 (2 off)
TR3,TR7,TR10,TR12	BD132 (4 off)
TR4,TR8	BD131 (2 off)
IC1	74LS00
IC2	556
IC3	pack of 8 commoned resistors RS type 140 271

Miscellaneous

ABS plastic box 150 × 100 × 60mm
Battery holder for 4 C-cells
Nicad C-cells (4 off)
MES lampholder
MES 12V 280mA bulb
Double pole on/off switch
3·5mm jack plug and socket
16 pin d.i.l. IDC header (5 off)
16 pin d.i.l. sockets (3 off)
14 pin d.i.l. sockets (2 off)
3 metres of 16 way grey ribbon cable
Molex 16 pin JD connector to mate with the 5332 series connector on Zeaker

Constructor's Note

The toe cover cap is available from most hardware stores and the wheels and motor gearboxes assemblies are obtainable from hobby shops.

A complete kit of parts for the vehicle (including machined, cut and ready-bent items) and control station with a manual and software is available from Colne Robotics Ltd., Beaufort Road, off Richmond Road, Twickenham TW1 2PH (01-892 8197/8241). Price £59·95 inc. VAT.

Colne Robotics are also able to supply the separate parts, please write or phone for details.

The two longer programs mentioned in the text are available from Colne either on cassette or as a print-out. Please state the type of computer being used with Zeaker when ordering programs.

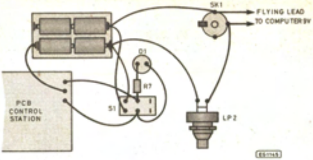
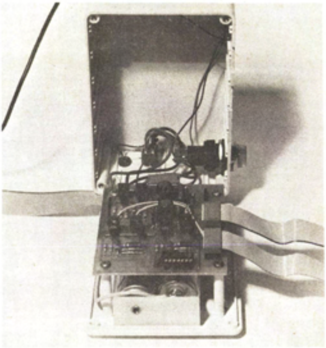
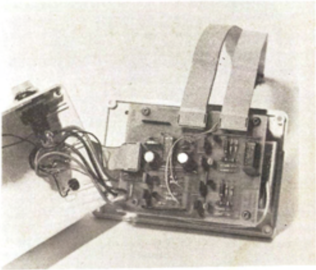


Fig. 4. Wiring diagram





Interior views of the Control Station showing the battery housing and the p.c.b. mounting details

FROM COMPUTER WRITE PORT

Pin 1

D7	D6	D5	D4	D3	D2	D1	D0
0V	0V					5V	5V

TO COMPUTER READ PORT

Pin 1

D7	D6	D5	D4	D3	D2	D1	D0
0V	0V					5V	5V

UMBILICAL CORD

0V	Motor Common 2·5V	Motor Starboard	Lights	Spare	Sensor Starboard Rear	Sensor Starboard Side	Sensor Starboard Front
Solenoid	Motor Common 2·5V	Motor	Speakers	Spare	Sensor Port Rear	Sensor Port Side	Sensor Port Front

Fig. 5. PSU DIL headers Signal Allocations

KEYBOARD TEACH PROGRAM
1K ZX81

```
                                       COMMENTS
 (1 REM TEACH/DOIT)
   5 FAST
  10 LET A=35000        ____________ 35000 IS A PORT
                                       ADDRESS
  20 POKE A,O           ____________ SWITCH OFF VEHICLE
  30 DIM M$ (10,2)
  40 LET K1=.6          ____________ STRAIGHT RUN
                                       CONSTANT
  50 LET K2=.61         ____________ TURN CONSTANT
  60 FOR S=1 TO 10
  65 CLS
  70 PRINT "TEACH", "STEP";S,
       "MOVE AND DIST/ANGLE/
       TIME"
  80 INPUT C$
  90 INPUT D
 100 IF C$="F" THEN LET
       M$(S,1)=CHR$ 5
 110 IF C$="B" THEN LET
       M$(S,1)=CHR$ 10
 120 IF C$="L" THEN LET
       M$(S,1)=CHR$ 6
 130 IF C$="R" THEN LET
       M$(S,1)=CHR$ 9
 140 IF C$="S" THEN LET
       M$(S,1)=CHR$ 0
 150 LET M$(S,2)=CHR$ D
 160 IF C$="F" OR C$="B"
       THEN LET M$(S,2)=CHR$
       (DxK1)
 170 IF C$="R" OR C$="L"
       THEN LET M$(S,2)=CHR$
       (DxK2)
 210 NEXT S
 220 PRINT "TO DOIT PRESS D"
 230 PAUSE 50000                     WAIT UNTIL ANY KEY
                                       PRESSED
 240 FOR S=1 TO 10
 250 POKE A,CODE M$(S,1)
 260 PAUSE CODE M$(S,2)
 270 NEXT S
 280 POKE A,O
 290 GOTO 230
```

When a port is read the resulting number will depend on which if any of the six sensor switches are closed. If none are closed i.e. if Zeaker is not touching anything then all 8 bits will be high and a read will return 255. If say the starboard front sensor is pressed then from Table 3, D3 will be low and hence a read will return 247. If both front sensors are pressed in then both D2 and D3 will be low and hence a read will return 243.

RANDOM MOVE PROGRAM
1K ZX81

```
                                       COMMENTS
   1 REM RANDOM MOVE
  10 LET A=35000        ____________ 35000 IS PORT
                                       ADDRESS
  20 POKE A,O           ____________ SWITCH OFF VEHICLE
  30 PRINT "PRESS S TO STOP"
  40 IF INKEY$="S" THEN
       GOTO 200
  50 PAUSE 5
  60 POKE A,32+5        ____________ SWITCH ON LIGHTS
                                       AND FORWARD
  70 PAUSE 5
  80 POKE A,5
  90 LET B=PEEK A
 100 IF B=255 THEN GOTO 40
 110 POKE A,64+10       ____________ LOW HORN AND
                                       BACKWARDS
 120 PAUSE 30
 130 IF B>250 THEN GOTO 170
 140 POKE A,128+6       ____________ HIGH HORN AND
                                       LEFT
 150 PAUSE 50×RND       ____________ RANDOM TURN TIME
 160 GOTO 40
 170 POKE A,128+9       ____________ HIGH HORN AND
                                       RIGHT
 180 PAUSE 50*RND       ____________ RANDOM TURN TIME
 190 GOTO 40
 200 POKE A,O           ____________ SWITCH OFF VEHICLE
```

CONTROL CODES

1 Port forward
2 Port back
4 Starboard forward
8 Starboard back
16 Solenoid on
32 Lights on
64 Horn 1
128 Horn 2
192 Horn 1 plus Horn 2

Table 1

SENSOR CODES

D0 Starboard sensor Aft closed
D1 Starboard sensor Side closed
D2 Starboard sensor Front closed
D3 Port sensor Front closed
D4 Port sensor Side closed
D5 Port sensor Aft closed
D6 Spare
D7 Spare

Table 2

It is reasonably easy to determine which sensors are closed by subtracting the return value from 255 and transforming the result into binary. For the previous example of both front sensors closed this returns 243. Now $255-243=12=8+4=2^3+2^2$ hence data lines 3 and 2 are at 0V which from Table 2 means that both front sensors are closed.

Although this may seem a complicated procedure to go through each time, remember a computer controls Zeaker and it will do all the tiresome calculations.

The initial software consists of two short programs, one which allows you to build a simple pattern and repeat it and the other lets Zeaker find its own way around obstacles, and two longer programs, which are available from Colne, one of which allows several patterns to be built up and joined together and the other allowing Zeaker to memorise its environment and to avoid obstacles sensibly.

For the initial ZX81 version of Zeaker the two short programs will each fit into 1K of memory and so can be run on an unexpanded ZX81. Writing a 1K program to control Zeaker from BASIC does not allow the full range of Zeaker's capabilities to be used. However, the two ZX81 1K programs here should give an idea of the ease with which Zeaker may be controlled from BASIC.

PROGRAM NOTES

Encoding the move code and time of move into the character array M$ saves 80 bytes over using the numerical array. Against this must be set the 18 bytes for the CODE, CHR $ and $ used in the listing, resulting in 62 bytes saved. The program just fits in 1K and can be edited and run. Report code 4 (out of memory) comes up most of the time but should be ignored. The program can store up to 10 moves and prompts for the move.

F=Forward	and	Distance millimetres
B=Backward		Distance millimetres
R=Rotate Right		Angle degrees
L=Rotate Left		Angle degrees
S=Stop		Time in 50th second

To escape from the program press break.

The constants K1 and K2 should be fine tuned to the particular vehicle. The maximum value of any entry in M$ is 255 and hence entering numbers greater than this in response to the prompt will cause the program to halt with an error code.

FURTHER DEVELOPMENTS

Although all 8 data read lines are connected to Zeaker only 6 of them are actually used, the remaining two terminate at pads by the Molex connector on the p.c.b. in the lid of Zeaker.

It is intended that these spare lines should be used to interface to add-on circuitry which will enable Zeaker to follow a white line, induction loop cable or simply seek or avoid light or heat.

Also instead of sending just the horn tones down the umbilical cord to the speaker it is possible to connect the output of a complex sound generator or a computer speech board to a pad by the umbilical cord connection on the p.s.u. board and Zeaker will be able to chuff along like a steam train or emit some more appropriate sound. It could also give a running commentary on its progress e.g. "Forward", "Right", "Left", "Ouch"! "Hit something at Left Front" etc.

Whilst Zeaker is relatively simple it is capable of quite complex interactions with its environment and in many respects it is only limited by the ingenuity of the controlling software. ★