# Off your trolley!

*Your own robot to build yourself for under £15?* ***John Billingsley*** *shows you how*

Fancy building the *Popular Computing Weekly* buggy? Here we show you how to very simply construct a robot for either the Spectrum, Commodore 64, ZX81, or BBC machines – all for around £13.

Don't be put off if you think it looks too complicated – all the parts are quite easy to get hold of or if you prefer we can supply you with a kit of parts.

It's a small step from adding extra software to your micro to adding a genuine mobile mechanism, but it will completely change your attitude to the possibilities of your machine.

The secret is to limit your first experiment to the simplest of interfaces. Afterwards you will be inspired to go on to build micromice, or even robots with sensors – but start off gently with a simple trolley.

Rather than getting involved in complicated reversible motor drives, the *Popular* buggy just switches the left and right motors of the trolley on or off. This makes it possible to use an interface consisting of just a single chip for the Commodore, and BBC computers, and just two chips for a Spectrum or ZX81. The trolley is still fairly steerable, although it cannot spin on the spot.

You will be able to teach the trolley a course around the room – or as far as the linking cable will stretch.

When you press the *F* key of your computer, the trolley will roll forwards in a straight line. Release it and press *L* instead, and the trolley will pivot on the left wheel to turn left. Press *R* and the trolley will pivot to the right, press *Space* and it will stop.

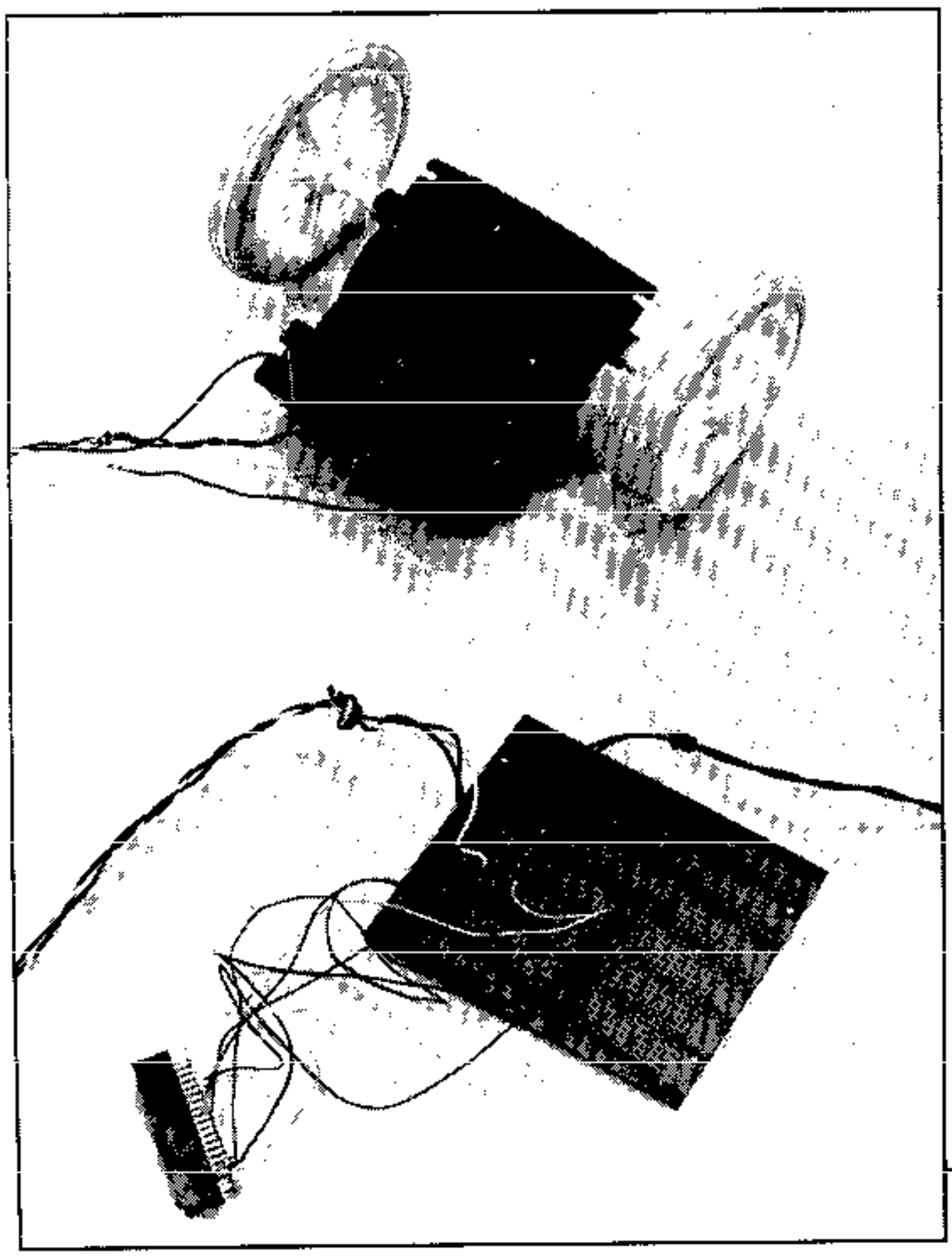
The full program will record in memory the times for which you held down the keys. Lift the trolley back to the start and press *G*, and the trolley will set off on the remembered course. There is no feedback or sensing, so the course may be a bit wobbly – but it's a start towards greater things.

## The trolley

To minimise the work involved in building the trolley, I settled on a gadget sold by Greenweld of Southampton. This is made as the innards of a controllable tank, and has two motors with suitable gearboxes and a cunning magnetic clutch arrangement to help it run straight. They also sell push-on wheels.

To make the trolley, take the motor/gearbox, push on the two wheels. And, that's it. You can't get much simpler than that!

Of course, that leaves the tail of the trolley scraping along on the floor, and you will probably need to add a tail-wheel or skid. That won't change the way it works, however, so I will leave that part entirely up to you.



One thing that you might have to attend to is the cunning magnetic clutch – it can get a bit too cunning for the steering to work. The idea is that a small magnet is mounted on the shaft of each motor, and being close together the magnets will try to lock into line. If one motor is a bit more energetic than the other, there would normally be a tendency for the vehicle to go round in large circles. Instead the magnets fall in line to drive both wheels at exactly the same speed. If the magnets are too close together, the wheels will lock together. To solve this place a thin screwdriver between them through the slot which is conveniently provided, and with two pennies squeeze the ends of the motor shafts towards each other. In this way you will press the magnets and their gears more firmly on to the motor shafts, and the gap will be increased.

Check the clutch by applying a 1.5 volt battery to the wires of just one of the motors. That motor should start alone, without carrying the other motor along too.

There is still one more task in preparing the trolley, and that is to attach the cable. Connect the blue lead of the left motor to the red lead of the right – that is, with the smooth side of the tolley downwards, and the motors towards you. Connect these to the wire of the connecting cable which we will call 'common'. Connect the remaining two major wires each to its own wire of the cable. A few inches of sticky tape will serve both to keep the connections separately insulated, and to anchor the end of the cable to the trolley. If you are feeling wealthy, you can use three wires-worth of ribbon cable for the lead. Instead I recommend that you plait together three lengths of instrument wire, one red, one black and one white. Use the red wire as the 'common', black for the left motor wire and white for the right.

Now test-drive the trolley under manual control. You will need two ordinary 1.5 volt batteries, wired up in series, giving 3 volts. Connect the red (common) wire to the positive terminal of the pair of batteries, and touch the black and white in turn and both together on to the negative end. With both connected the trolley should run straight. You will probably find that to get the trolley to turn, you must first let it stop before driving just one motor.

Now, we have to let the computer take over the task of switching the motors.

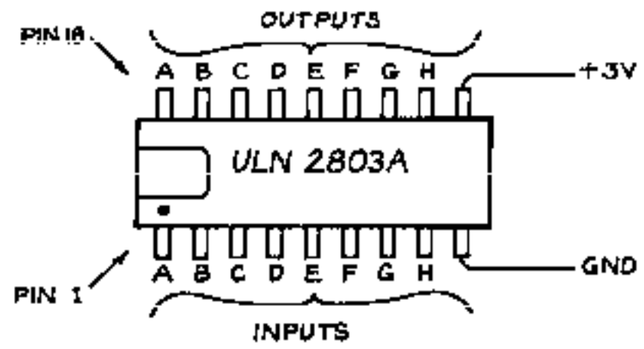
## The power chip

The circuit board required only uses one component (two in the case of the Spectrum version) so it is very straightforward to construct.

The ULN2803 chip which is used contains eight Darlington transistors. We really only need two of these, but the chip is a really convenient way to obtain them. It also gives a lot of scope for future projects.

Connect each of the motor leads to an output of the chip, on pins 18 and 17, while the common motor lead is connected to the +3 volt point of the battery. This point is also linked to pin 10 of the chip. The negative end of the battery is connected to 'system ground', which in this case means pin 9 of the chip which is also connected to the computer's ground.

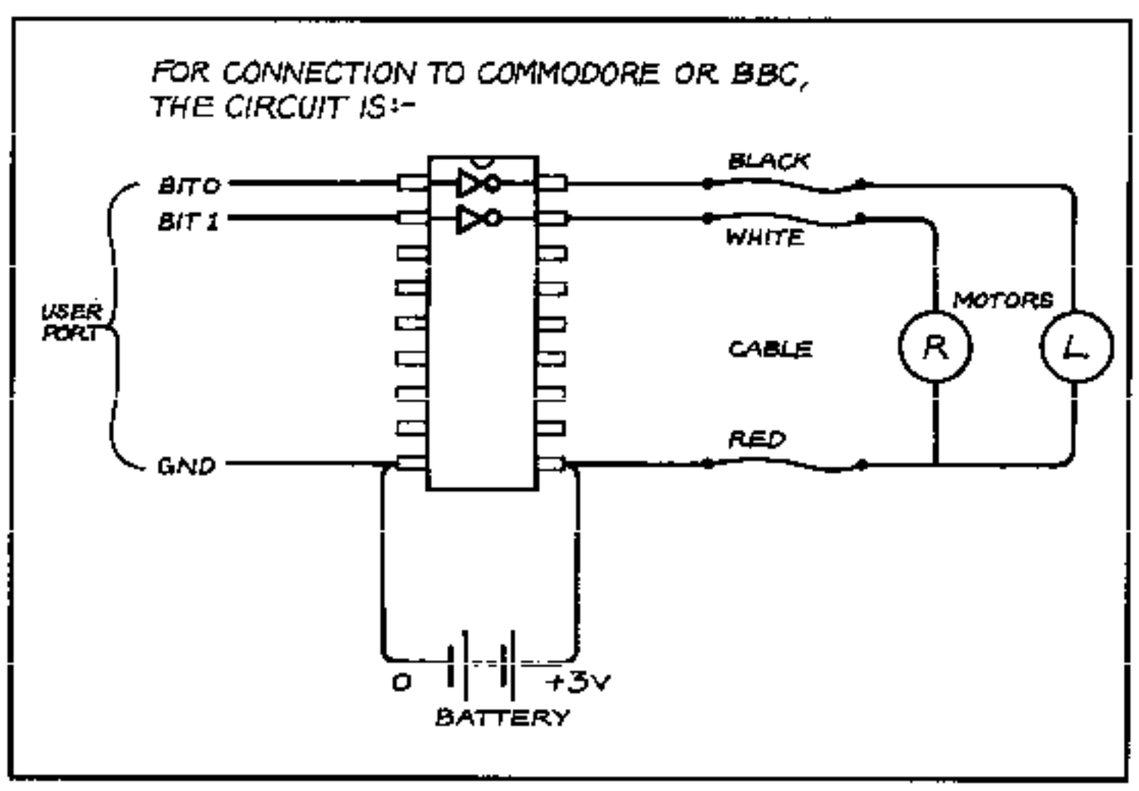
Each of the transistors of the chip is capable of switching 0.5 amps, enough for one of our motors, a relay, or for a




TOP VIEW OF OUTPUT CHIP




future stepping motor. The chip performs the necessary power conversion to step up the feeble current of the computer's 'user port' (if it has one) to the current necessary to drive the motors. We need only connect user port bits 0 and 1 to pins 1 and 2 of the chip to complete the system.



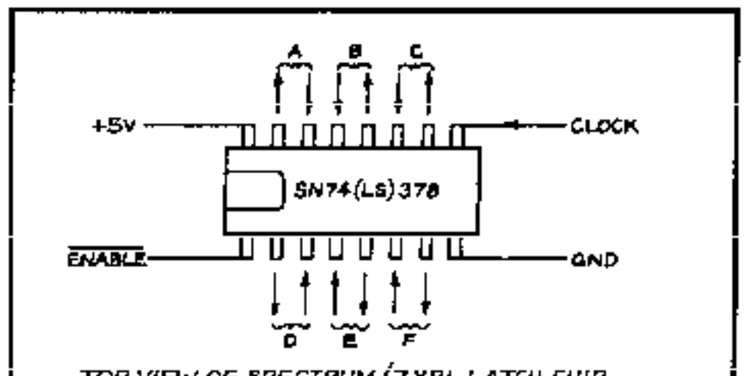

The user port contains eight output lines, each set by one bit of an output byte. The desired value must be *Poked* into a suitable address, or set on the BBC by *?&FE60* = the pins of the connector, and will remain there until another value is output.

## Output from a Sinclair

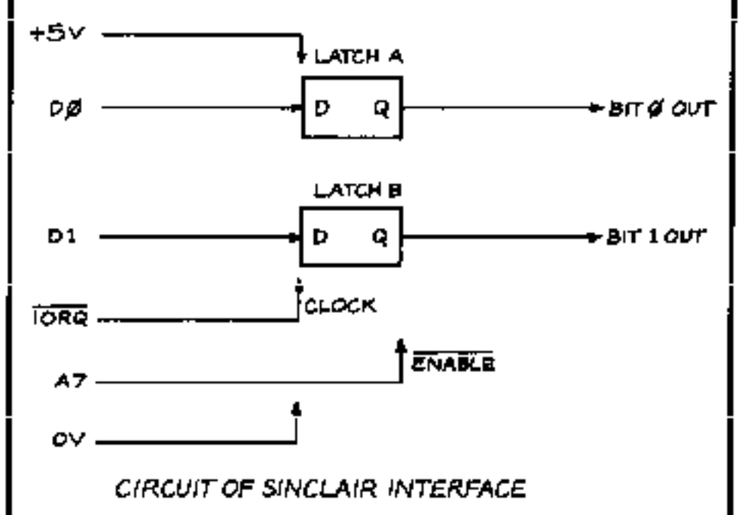
If your computer is a ZX81 or a Spectrum, you will by now be worried that it has no 'user port'. Instead, there is an expansion connector which brings out all the sensitive inner signals, including the address and data bus. As each machine-code instruction is taken from the memory, its value fleetingly appears for a microsecond or two on the data bus. If an *Out* command is executed, the value will also briefly flash on to the bus, but will be gone in an instant. Somehow we have to catch the value in flight, and nail it down so that the motors can respond to it.

The Z80 chip can command a value to be sent to any of 256 addresses, selected by the eight lower address bits. The output is signalled by the input/output request line and *Write* line being pulled low, when the address appears on the address lines and the data bits are presented on Lines D0 to D7. The 'proper' way of adding an output interface would involve decoding the device address, *And*'ing it with I/0 request and the *Write* strobe, and using the resulting pulse to latch the eight bits of the data bus. It's easier to cheat a little.

The extra chip needed for the 74LS378 chip has six latches, a clock line and an enable line. If the enable line is pulled low, than a pulse on the clock line will latch the data inputs so that a steady value will appear at the output until the next pulse. Therefore, as many bits of the data bus as we need are connected (in this case just D0 and D1) to the chip's data inputs. By tying the enable pin of the chip to the I/0 request line of the Spectrum and the chip's clock line to the Spectrum's *Write* line, the chip will only latch signals which are meant as outputs.




TOP VIEW OF SPECTRUM/ZX81 LATCH CHIP



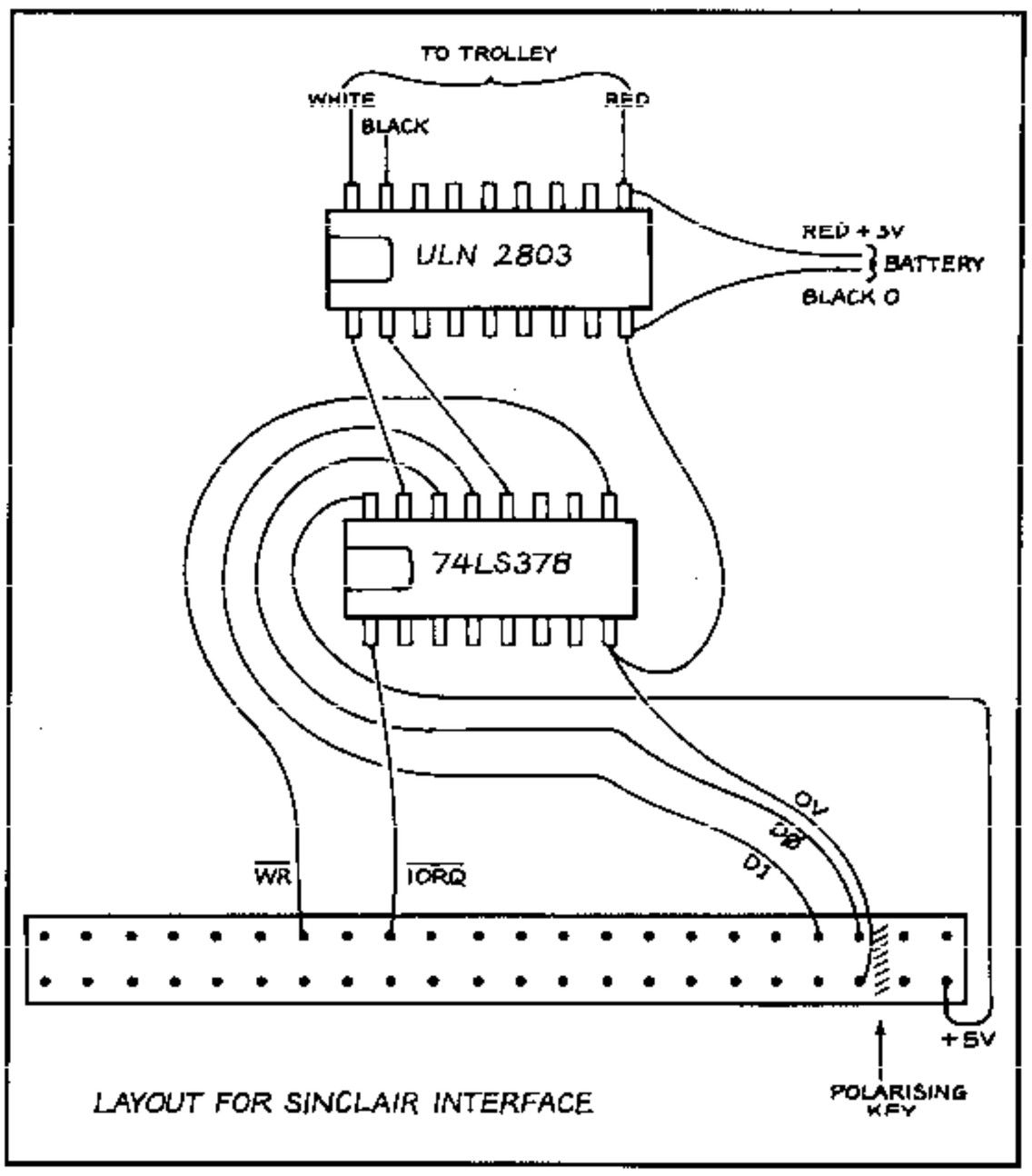

CIRCUIT OF SINCLAIR INTERFACE

The snag is that it will catch *every* output, whether directed to the trolley, to the screen to the printer or the sound chip. The program must therefore avoid writing data to any device but the trolley.

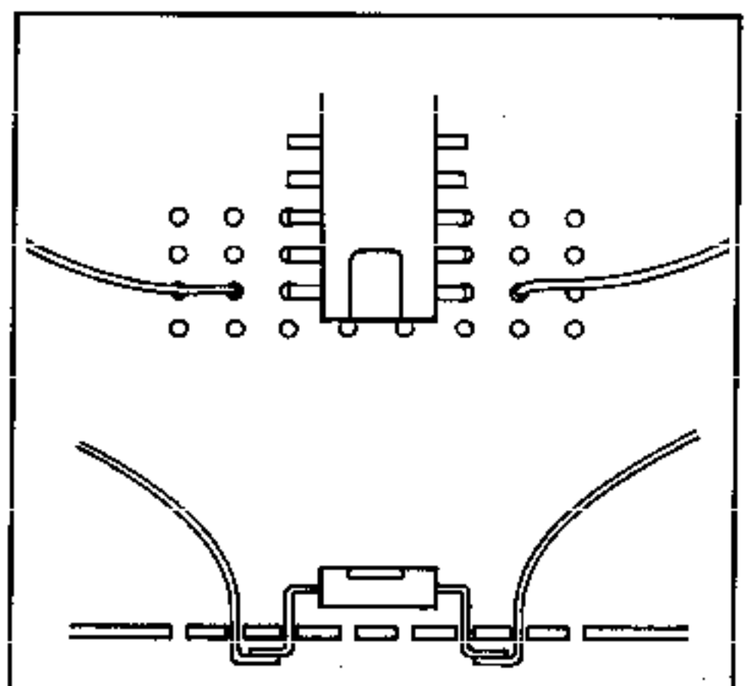
In this simple way, you can obtain six output logic signals with just one chip (eight lines with a 74LS377). To obtain two output bits it is connected by just six wires to the Spectrum or ZX81 expansion port edge connector.

## Practical construction

For a circuit this simple, using an unclad circuit board will be easier. The pins of the chip fit through the holes, and are bent outwards on the underside. To make a connection, a wire is pushed




LAYOUT FOR SINCLAIR INTERFACE



MAKING CONNECTIONS TO THE CHIP

through the adjacent hole and bent over on top of the pin where it is soldered in place. Take care when soldering. Clean the iron before every joint – a quick wipe on a wet tissue will do the trick – and melt fresh solder on to the joint as you solder it. 'Tin' each wire and component before making the joint by melting fresh solder against the iron in contact with that component alone. The result should be a bright film of tin which wets each surface to be connected. Now bend the leads so that they press together, and a swift touch with the iron will result in a reliable joint. Mount the Darlington chip on to the board as shown in the diagram. The three leads to the trolley can now be connected Red to pin 10, White to pin 17 and Black to pin 18. The leads from the battery holder are connected Red (+3 volts) to pin 10, joining the motor lead, and Black (battery negative) to pin 9. Remember that the pins are numbered anticlockwise.

For Commodore and BBC users there are only three more connections to make – to the user port of your machine. On Commodore Pet, C64 and Vic the edge connector is 12 pins wide. Pin A, at the bottom left, is the ground connection and is to be connected to pin 9 of the chip. Pin C, third from the bottom left, is data bit 0 of the user port and you must connect this to pin 1 of the chip. Pin D is data bit 1 and must be connected to chip pin 2.

The BBC version of the kit includes a user port connector with a short length of ribbon cable already attached. The ribbon emerges from the bottom edge when plugged into the computer, and if left straight without twists will emerge from the front beneath the keyboard. In this position, wire number 1 will be on the right, and wire number 20 on the extreme left of the cable. Connect wire 19, which is a ground connection, to pin 9 of the chip. Connect wire 4 to pin 1 of the chip and wire 5 to pin 2.

Spectrum and ZX81 owners must add a second chip before they can try out their system. Mount the 74LS378 a little way from the first chip as shown, and use thin instrument wire to make the links between the chips. Connect pin 13 of the 378 chip to pin 1 of the ULN, connect pin 10 of the 378 to pin 2 of the ULN. These are the outputs of the latches, driving the inputs of the Darlington chip. Connect pin 8 of the 378 to pin 9 of the ULN, linking the grounds.

The expansion port connector for the Spectrum is five pins longer than the one for the ZX81, but the connections which matter here are in the same position in each case relative to the polarising slot. This slot is in position 3 on the ZX81, or position 5 on the Spectrum. Use a 23-pin length of 0.1 inch connector with the polarising pin in position 3, measured from the right when looking into the back of the computer. Now the pin numbers will be given for the ZX81 and this connector, with the Spectrum shown in brackets. Although missing, pin 3 is still counted.

Connect pin 4 lower (Spectrum) to pin 7 of the 378, to link the system ground. Connect pin 1 (3) lower to the 378 pin 16 to provide +5 volts for the chip. Connect pin 4 (6) upper to 378 pin 11. Connect pin 5 (7) upper to 378 pin 12. These are the data bus bits. Connect pin 15 (17) upper to 38 pin 1, computer I/O request to chip enable. Connect pin 17 (19) upper to 378 pin 9, WRITE strobe to the chip's clock.

## Software for controlling the trolley

There are two programs. The second is the all-singing, all-dancing version which will remember a route. The first, however, is a simple test program.

For the BBC and Commodore computers you first need to 'configure' the output. The bits of the interface chip can be used for outputs or inputs in any combination, and a 'data direction register' controls which is which. This register appears as an address in memory, and if you store 255 in it all the bits will be outputs. According to your machine, type the following as a direct command:

BBC: ?&FE62 = 255:PO = &FE60
Pet: POKE 59459,255:PO = 59471
C64: POKE 56579,255:PO = 56577
Vic 20: POKE 37138,255:PO = 37136

The port address has also been specified, so that the next instructions can be common for all Commodore machines.

Test also that the motors will step by using the following direct command:

BBC: ?PO = 0
CBM: POKE PO,0

The left motor should run with:

BBC: ?PO = 1
CBM: POKE PO,1

The right motor should run with:

BBC: ?PO = 2
CBM: POKE PO,2

Both motors should run with:

BBC: ?PO = 3
CBM: POKE PO,3

Something a little more elaborate is needed for the Sinclair, since both motors will be set running any time a character is written to the screen.

```
10 LET a$ = INKEY$
20 IF a$ = "r" THEN OUT 255,1
30 IF a$ = "l" THEN OUT 255,2
40 IF a$ = "f" THEN OUT 255,3
50 IF a$ = " " THEN OUT 255,0
60 GO TO 10
```

Now when you run the program you should be able to control the trolley by pressing *f,l,r* or *Space*. (Note that there is a space between the quotes in Line 50).

## The full program

Now you are ready for the full program.

**Commodore:**

```
 10 POKE 59459,255:PO=59471: REM PET   ***
 10 POKE 56579,255:PO=56577: REM C64M4 ***
 20 DIM B(100),T(100)
 30 N=0:TI$="000000"
 40 POKE PO,0
100 GET A$
110 IF A$="" THEN 100
120 IF A$=" " THEN K=0
130 IF A$="R" THEN K=1
140 IF A$="L" THEN K=2
150 IF A$="F" THEN K=3
160 IF A$="G" THEN 300
170 N=N+1:IF N>100 THEN 300
180 B(N)=K
190 T(N)=TI
200 POKE PO,0
220 FOR J=1 TO 200:NEXT
230 POKE PO,K
240 GOTO 100
310 POKE PO,0
320 FOR M=1 TO N
330 IF TI<T(M) THEN 330
340 POKE PO,0
350 FOR J=1 TO 200:NEXT
370 NEXT M
```

**BBC:**

```
 10 DIM B(100)
 20 DIM T(100)
 30 N=0:TIME=0
 40 J=0:K=0
 50 PORT=&FEA0:?&FEA2=3
100 A$=INKEY$(0)
130 IF A$="" THEN 100
140 IF A$=" " THEN K=0
150 IF A$="L" THEN K=2
160 IF A$="R" THEN K=1
170 IF A$="F" THEN K=3
180 IF A$="G" THEN 300
190 N=N+1
200 B(N)=K
210 T(N)=TIME
220 ?PORT=0
230 J=INKEY(25)
240 ?PORT=K
260 GOTO 100
300 TIME=0
310 FOR M=1 TO N
320 REPEAT UNTIL TIME > T(M)
330 ?PORT=0
340 K=INKEY(25)
350 ?PORT=B(M)
```

**Sinclair:**

```
 10 DIM b(100)
 20 DIM t(100)
 30 LET i=0: LET n=0: LET j=0: LET k=0
100 LET a$=INKEY$
110 LET i=i+1
120 IF i<j THEN  GO TO 100
130 IF a$="" THEN  GO TO 100
140 IF a$=" " THEN  LET k=0
150 IF a$="l" THEN  LET k=2
160 IF a$="r" THEN  LET k=1
170 IF a$="f" THEN  LET k=3
180 IF a$="g" THEN  GO TO 300
190 LET n=n+1
200 LET b(n)=k
210 LET t(n)=i
220  OUT 255,0
230 FOR j=1 TO 100: NEXT j
240  OUT 255,k
250 LET j=1
260 GO TO 100
300 LET i=0
310 FOR m=1 TO n
320 FOR j=1 TO t(m): NEXT j
330 OUT 255,0
340 FOR j=1 TO 100: NEXT j
350 OUT 255,b(m)
360 NEXT m
370 OUT 255,0
```

1 2-cell battery connector
3 metres each of 3 colours of single core cable (for connection to computer)
2 Wheels
3 metres of thin single core cable for wiring
1 small unclad connector board
1 ULN 2803 Darlington Chip
1 0.156 12-way connector (for the Commodore 64)

or

1 20-way connector with ribbon cable attached (for the BBC)

or

1 23-way 0.1 edge-connector with polarising plug in the third position (for the Spectrum)

The Spectrum version also requires
1 74LS 378 chip.

If you do not wish to use the Greenweld gearbox and motors unit (available for £5.95) you will need 2 gearboxes (50:1 ratio or similar) and two small dc motors.

Commodore:

```
10 POKE 59459,255: PO=59471: REM PET
10 POKE 56579,255: PO=56577: REM CBM64
10 POKE 37138,255: PO=37136: REM VIC20
20 DIM B(100), T(100)
30 N=0: TI$="000000"
40 POKE PO, 0
100 GET A$
110 IF A$="" THEN 100
120 IF A$=" " THEN K=0
130 IF A$="R" THEN K=1
140 IF A$="L" THEN K=2
150 IF A$="F" THEN K=3
160 IF A$="G" THEN 300
170 N=N+1: IF N>100 THEN 300
180 B(N)=K
190 T(N)=TI
200 POKE PO,0
220 FOR J=1 TO 200: NEXT
230 POKE PO,K
240 GOTO 100
300 TI$="000000"
310 POKE PO,0
320 FOR M=1 TO N
330 IF TI<T(M) THEN 330
340 POKE PO,0
350 FOR J=1 TO 200: NEXT
360 POKE PO,B(M)
370 NEXT M
```

BBC:

```
10 DIM B(100)
20 DIM T(100)
30 N=0: TIME=0
40 J=0: K=0
50 PORT=&FE60: ?&FE62=3
100 A$=INKEY$(0)
130 IF A$="" THEN 100
140 IF A$=" " THEN K=0
150 IF A$="L" THEN K=2
160 IF A$="R" THEN K=1
170 IF A$="F" THEN K=3
180 IF A$="G" THEN 300
190 N=N+1
200 B(N)=K •
210 T(N)=TIME
220 ?PORT=0
230 J=INKEY(25)
240 ?PORT=K
260 GOTO 100
300 TIME=0
310 FOR M=1 TO N
320 REPEAT UNTIL TIME > T(M)
330 ?PORT=0
340 K=INKEY(25)
350 ?PORT=B(M)
360 NEXT M
```

Sinclair:

```
10 DIM b(100)
20 DIM t(100)
30 LET i=0: LET n=0: LET j=0: LET k=0
100 LET a$=INKEY$
110 LET i=i+1
120 IF i<j THEN GO TO 100
130 IF a$="" THEN GO TO 100
140 IF a$=" " THEN LET k=0
150 IF a$="l" THEN LET k=2
160 IF a$="r" THEN LET k=1
170 IF a$="f" THEN LET k=3
180 IF a$="g" THEN GO TO 300
190 LET n=n+1
200 LET b(n)=k
210 LET t(n)=i
220 OUT 255,0
230 FOR j=1 TO 100: NEXT j
240 OUT 255,k
250 LET j=1
```

```
260 GO TO 100
300 LET i=0
310 FOR m=1 TO n
320 FOR j=1 TO t(m): NEXT j
330 OUT 255,0
340 FOR j=1 TO 100: NEXT j
350 OUT 255,b(m)
360 NEXT m
370 OUT 255,0
```