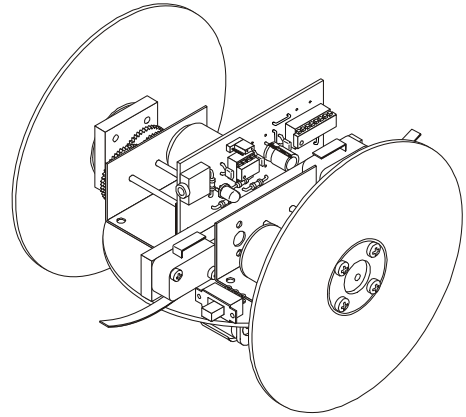


DIZZY

DESCRIPTION

DIZZY is a two-wheeled user programmable vehicle. When it hits an object, *DIZZY* backs away, spins around quickly (hence its name) and continues in another direction. *DIZZY* also changes direction when it doesn't hit anything for some time.

DIZZY has a lever arm microswitch at each end that detects an obstruction. It is driven by two electric motors, each with its own gearbox, and is controlled by a PICAXE-08M microprocessor. The prototype *DIZZY* was built using recycled CDs, but the concept has scope for individual variation.



INVESTIGATION

This project provides a number of different aspects of the *DIZZY* for investigation. Some ideas are listed below and in the "Further Development" section.

- The vehicle you design may be evaluated for vehicle balance, turning circle, component layout and space efficiency. This could be compared to our prototype for efficiency and performance.
- Evaluate the suitability of various materials. For example: CD's, Aluminium, PVC and Perspex.
- Investigate different wheel base configurations. For two, three and four wheeled vehicle options, what differences would need to be made to the mechanics, electronics and program?
- Investigate adding more "intelligence" to the program.
- Investigate other devices that could use two bi-directional motors and one switch input.

1. COMPONENTS REQUIRED

1.1 COMPONENTS SUPPLIED. The following components are supplied in the kit:

- | | |
|---|--|
| 1 x Printed Circuit Board (P.C.B.) | 1 x Jumper |
| 1 x PICAXE-08M (microprocessor) | 2 x Microswitch - long lever |
| 1 x L293D (motor driver) | 1 x Sliding switch (small) |
| 1 x Socket for I.C. (16 pin) | 1 x Battery holder- 4AA |
| 1 x Socket for I.C. (8 pin) | 22 x Self-tapping screw - 2.6mm x 4mm |
| 5 x Capacitor - 0.1uF (monolythic or equivalent) | 4 x Screw - M3 x 20mm |
| 1 x Capacitor - 100uF (electrolytic) | 4 x Nut - M3 |
| 1 x Resistor - 220 Ohms (Red-Red-Brown-Gold) | 1 x LED (Red) |
| 2 x Resistor - 10k Ohms (Brown-Black-Orange-Gold) | |
| 1 x Resistor - 22k Ohms (Red-Red-Orange-Gold) | 2 x Sets of Multi-ratio Gearbox |
| 1 x Stereo socket - 3.5mm | components (refer separate sheet for the |
| 1 x Header strip - 3 pins | component list) |

1.2 ADDITIONAL REQUIREMENTS

- 1.2.1 The following items are available from us, but need to be separately ordered: 2.3 and 2.6 diameter drill bits; and 1.5 volt AA batteries (4 required).
- 1.2.2 A PICAXE serial interface cable. This can be purchased from a number of suppliers or may be constructed as per instructions in this document.
- 1.2.3 A PC with 9-pin serial (RS-232) interface. (The PC may require an USB to RS-232 adapter.)
- 1.2.4 The PICAXE programming editor software requires a PC running Windows 95 or later with approximately 20MB free space. Any PC Windows operating system works in textual 'BASIC' mode - a Pentium 4 processor or later is recommended for graphical flowcharting.



SCORPIO TECHNOLOGY VICTORIA PTY. LTD.

A.B.N. 34 056 661 422

17 Inverell Ave., Mt. Waverley, Vic. 3149

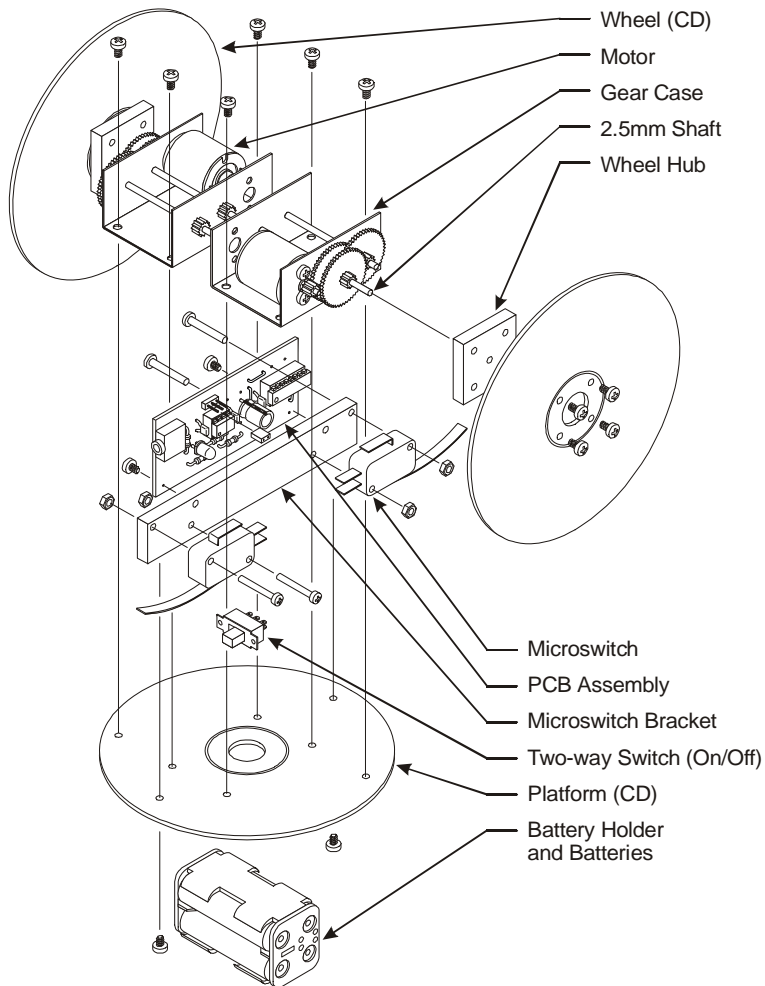
Tel: (03) 9802 9913 Fax: (03) 9887 8158

www.scorpiontechnology.com.au

1.2.5 The PICAXE programming editor software can be downloaded for free from www.rev-ed.co.uk or from www.picaxe.co.uk.

1.2.6 Material for platform and wheels, wheel hubs and microswitch bracket; electric hook-up wire in assorted colours. We used 4.5 and 6mm PVC sheet. (For plastic sheet refer to the Telstra Yellow Pages under the heading "Plastics Fabricators.")

2. MECHANICAL DESIGN CONSIDERATIONS



DESIGN CONCEPT

2.1 PLANNING

2.1.1 *DIZZY* consists of a platform, on which components are mounted, and a wheel on each side. For our prototype, we used CD's for the platform and wheels, but other materials may be used.

2.1.2 Before starting construction, carefully plan and lay out all the components (platform, wheels, gear cases, battery holder, microswitches and P.C.B.) on a sheet of paper or a suitable computer program. It is best to look at the vehicle as a complete unit, and not just as separate parts.

2.1.3 *DIZZY* works properly if it is balanced so the platform is approximately horizontal when *DIZZY* is at rest. Position the centre of gravity as low as possible. Note that the batteries and motors are the heaviest components.

2.2 PLATFORM

2.2.1 Locate both gearboxes with their driving shafts in line with the platform's centre, and a suitable distance apart, to provide room for the microswitches and P.C.B. to be mounted.

2.2.2 Determine a suitable position for the P.C.B. assembly and the on/off sliding switch. We used screws to attach the P.C.B. to the microswitch bracket (6mm thick PVC), which was then attached to the platform. Hot-melt glue was used to attach the switch to the platform.

2.2.3 Locate the battery holder underneath the platform so that the platform is approximately horizontal when the batteries are inserted. Investigate alternate battery holder attachment methods. Possibilities include: hot-melt glue, screws, double-sided tape and hook-and-loop tape ("Velcro" or equivalent).

2.3 WHEELS

2.3.1 A suitable operating speed is provided by using two Multi-ratio gearboxes and CD's for wheels. (refer to the "Assembling the Multi-ratio Gearbox" teaching unit supplied)

2.3.2 Determine the wheels to be used, and how to attach them to the gearbox output shaft. For ours, we attached 4.5mm thick PVC with a central 2.3mm hole to each (CD) wheel.

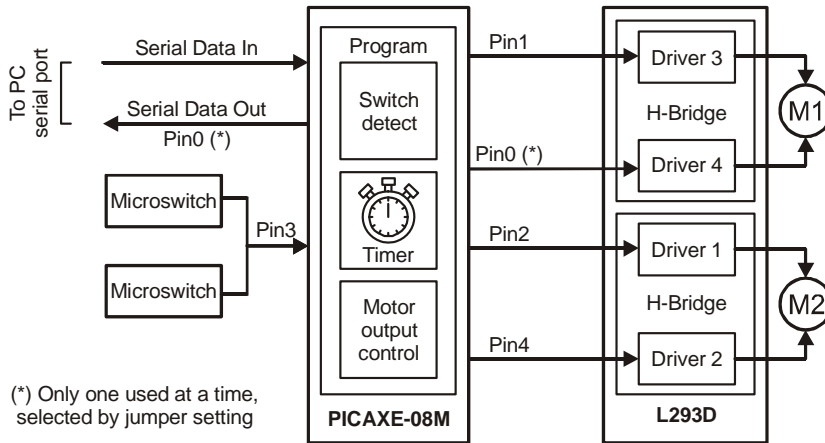
2.3.3 Other wheels may be used if desired, but you need to consider if a different gear ratio will be required (See the Scorpio Technology website or catalogue for options).

3. HOW THE CIRCUIT WORKS (THEORY)

3.1 CIRCUIT AND PROGRAM OVERVIEW

NOTE: PICAXE documentation refers to "Input Pins" and "Output Pins", which are not the same as the physical pins on a device. To avoid confusion, in this document "leg" means a physical pin of an integrated circuit and "pin" means a logical input or output.

3.1.1 A program must be downloaded into the PICAXE-08M memory, to control the wheels rotation in response to input from switches and an internal timer. The program can be customised as desired.

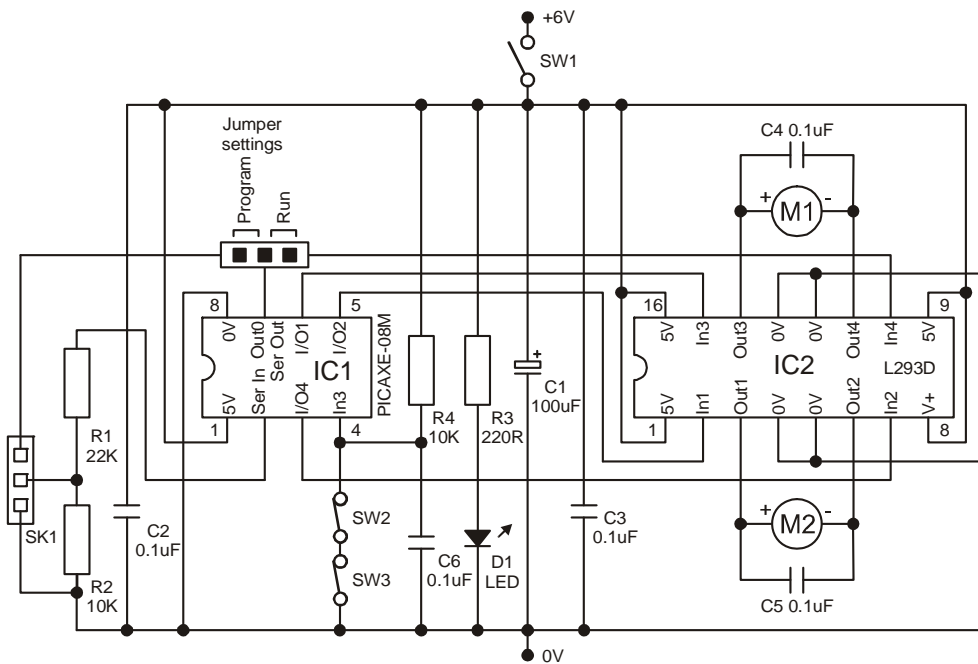


(*) Only one used at a time, selected by jumper setting

BLOCK DIAGRAM

3.1.2 Pin0 on the PICAXE is shared by the download circuit and one of the motor outputs. To download a program, move the jumper to the "program" position. This connects Pin0 to the stereo socket. Using the download cable, connect the stereo socket to a serial port on the PC and download the program.

3.1.3 When the program has downloaded, move the jumper to the "run" position. The PICAXE-08M then executes the commands in the program.



CIRCUIT DIAGRAM

3.1.4 When our program starts running, the motors are set to run in one ("forward") direction. The main program loops continually check the status of both microswitches on pin3 as well as an internal timer. When either switch is pressed or the timer has expired, the PICAXE-08M outputs a timed sequence of high/low voltages on pin0, pin1, pin2 and pin4.

3.1.5 Responding to the high/low voltage sequence on its inputs, the L293D opens/closes the high-current electronic switches in its H-bridge circuits.

Each H-bridge circuit (consisting of two "driver" blocks) controls the direction of one motor. Consequently, the motors turn backwards, forwards and stop as commanded by the PICAXE.

3.2 ABOUT PICAXE MICROCONTROLLERS

3.2.1 The PICAXE* is a type of chip called a microcontroller, which is another name for a single chip computer. The PICAXE has similar features to a PC: CPU (central processing unit), RAM (random access memory), ROM (read only memory), I/O (input/output) lines, timers and A/D (analogue /digital) converters.

* PICAXE is a tradename for this item from Revolution Education Ltd.

3.2.2 The Flash memory (EEPROM - Electrically Erasable Programmable Read Only Memory) in a PICAXE allows it to be reprogrammed many times (typically 100,000). This means that you can develop a program and constantly check the effects of changes.

3.2.3 A PICAXE program is created using an easy to learn version of the BASIC programming language (our preferred method) or using flowcharting software.

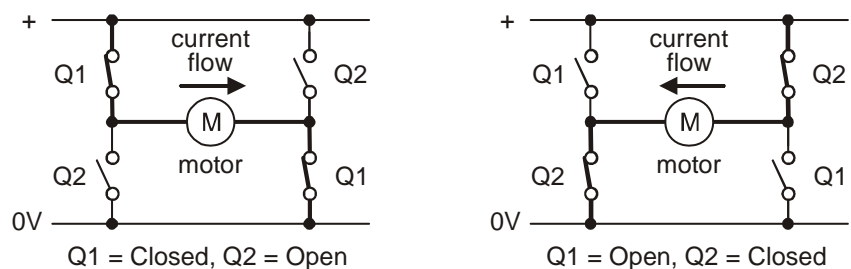
NOTE: The PICAXE contains a 'bootstrap' program that enables the program to be downloaded to the PICAXE using the serial cable. Do not substitute other blank / standard PIC programmers.

3.3 PICAXE-08M (IC1)

- 3.3.1 When the program runs, it reads the status of both microswitches and controls the direction of both motors. When programmed in this manner, the PICAXE behaves as a simple neural network.
- 3.3.2 PICAXE Leg 1 is connected to the positive terminal (+6V) of the power supply (batteries).
- 3.3.3 PICAXE Leg 2 (serial data in) is used only when transferring a program from a serial port on your PC (COM1: to COM4:) to the PICAXE. The 22k Ohm (R1) and 10k Ohm (R2) resistors must be present for reliable operation. Do not substitute other resistor values.
- 3.3.4 PICAXE Leg 3 (pin4 in the program) is connected to one of the inputs to the motor driver IC (Integrated Circuit) that controls motor M2.
- 3.3.5 PICAXE Leg 4 (pin3 in the program) is connected to the microswitches (SW2 and SW3). Leg 4 is tied to ground by the microswitches, which are wired in series. When either microswitch is opened, positive battery voltage appears on Leg 4 through resistor R4. Capacitor C6 reduces switch bounce.
- 3.3.6 PICAXE Leg 5 (pin2 in the program) is connected to one of the inputs to the motor driver IC, that controls motor M1.
- 3.3.7 PICAXE Leg 6 (pin1 in the program) is connected to another one of the inputs to the motor driver IC, that controls motor M1.
- 3.3.8 PICAXE Leg 7 (pin0 in the program) has two functions. Use the jumper on the header strip to select between the PICAXE being programmed or running the program.
- 3.3.9 PICAXE Leg 8 is connected to the negative terminal (0V) of the power supply (batteries).

3.4 MOTOR DRIVER L293D (IC2)

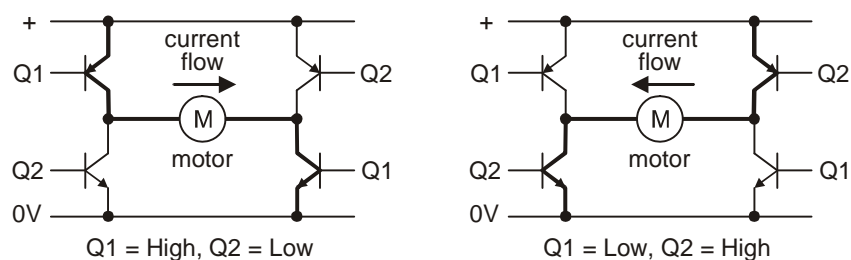
- 3.4.1 The L293D motor driver (IC2) contains two H-bridge circuits that are suitable for driving small DC motors. Each of the "driver" blocks contains transistors that work as electronic switches.



3.4.2 Changing the voltage applied to the input pins changes the direction of current flow through the motor.

- Q1 low, Q2 low - motor stop
- Q1 high, Q2 low - motor forward
- Q1 low, Q2 high - motor reverse
- Q1 high, Q2 high - motor stop

H-BRIDGE CIRCUIT CONCEPT



3.4.3 The motor driver IC is simpler to use and fault-find than using individual transistors and resistors. For more information relating to H-Bridge circuits constructed from transistors, see the "Radio Controlled Vehicle", "Wanderer" and "Seeker" teaching units on the Scorpio Technology

TYPICAL H-BRIDGE CIRCUIT

3.5 CAPACITORS

- 3.5.1 Capacitor C1 (100uF) smooths the battery power from fluctuations caused by motor switching.
- 3.5.2 Capacitors C2 and C3 (0.1uF) smooth the power supply close to the ICs.
- 3.5.3 Capacitors C4 and C5 (0.1uF) reduce the electrical motor noise that reaches the PICAXE.

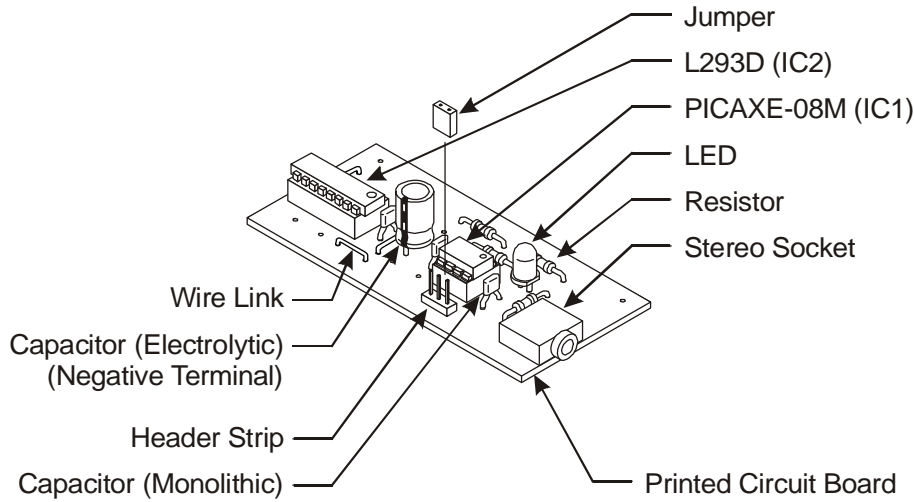
3.6 POWER

- 3.6.1 A red LED (light emitting diode) (D1), in series with a 220Ohm resistor, is used to indicate when power is present on the printed circuit board assembly. The intensity of LED illumination may change during operation.

3.6.2 Sliding switch SW1 is used to control power to the circuit.

4. ASSEMBLING *DIZZY*

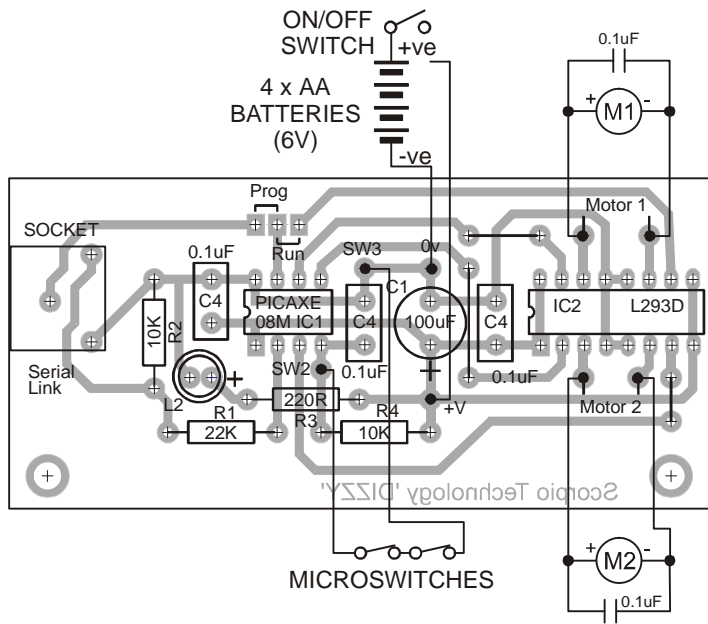
4.1 PRINTED CIRCUIT BOARD ASSEMBLY



PCB COMPONENT TERMINOLOGY

4.1.1. Solder the resistors (4) to the PCB. Trim the resistor leads and solder three of these to the PCB as wire links. The wire links are shown as straight lines on the PCB overlay.

4.1.2 Solder the remaining components to the PCB in this order: IC sockets, three-pin header strip, stereo socket, LED, capacitors and electrolytic capacitor. Trim the component leads as required. **CAUTION:** Take care in the correct orientation of the IC sockets, LED and electrolytic capacitor.



PRINTED CIRCUIT BOARD OVERLAY AND WIRING

Unsoldering and replacing damaged or wrongly positioned components will waste time. During soldering, do not overheat the printed circuit board and components.

WARNING: The electrolytic capacitor will be damaged and may cause injury if it is installed in the wrong direction and power is applied.

4.1.3 Do not insert the IC's into their sockets yet. These will be inserted during testing.

NOTE: For wiring, it is useful if wires of different colours are used. This can assist in tracing wires during fault finding. When soldering wires, strip a short piece of insulation from the end of the wire, twist the strands and "tin" them with solder.

4.2 GEARBOX AND MOTOR ASSEMBLY

Refer to the sheet – Assembling the Multi-ratio Gearbox. For *DIZZY* we suggest using the Triple-reduction version of this gearbox.

After assembly, both shafts should be cut to the required lengths - use a file to remove burrs.

4.2.1 Solder a 0.1uF capacitor across the terminals of each motor.

4.2.2 Solder a suitable length of wire to each of the four motor terminals.

4.2.3 Press the 10T pinion onto the motor shaft. Hint: Place the gear on the bench, insert the motor shaft into the worm gear's hole and gently tap the end of the shaft (where it exits the motor) with a small hammer. Stop when the worm gear is 3mm from the motor's body.

WARNING: Don't just push the motor down by hand as this can push the motor armature out of its bearings and jam the motor.

- 4.2.4 Secure the motor to the gearbox case using two self-tapping screws. Pass the motor wires through the larger hole in the gearbox case.

4.3 MECHANICAL ASSEMBLY

NOTE: When drilling holes: a 2.6mm drill bit provides clearance holes for bolts and screws; and a 2.3mm drill for self-tapping screws and holes that press fit onto the 2.5mm steel rod.

- 4.3.1 Position the gearbox assemblies on the platform. Drill six holes for self-tapping screws in the platform to match the position of the holes in each gearbox case.
- 4.3.2 Assemble a driving wheel to each gearbox output shaft. If a wheel or gear slips on its shaft, then score the shaft using a hammer and cold-chisel.
- 4.3.3 Use six self-tapping screws to secure both gearbox assemblies to the platform.
- 4.3.4 Form the lever arms of the microswitches so that they form a gentle curve and that each **microswitch clicks when the platform is approximately 45 degrees from horizontal.**
- 4.3.5 Make a bracket for the microswitches and P.C.B. assembly. Use screws and nuts to attach the microswitches to the bracket. Attach the P.C.B. assembly using two self-tapping screws.
- 4.3.6 Attach the bracket to the platform using two self-tapping screws.
- 4.3.7 Solder a length of wire between the "Normally Closed" terminals (usually marked with "NC" or "2") on both microswitches. Solder a length of wire between the "Common" terminal (usually marked with a "C" or a "1") on one microswitch to "SW2" on the printed circuit board. Solder a length of wire between the "Common" terminal on the other microswitch to "SW3" on the printed circuit board.
- 4.3.8 Attach the battery holder below the platform. Pass the wires through a hole in the platform.

4.4 WIRING

NOTE: Refer to the wiring diagram for more details.

- 4.4.1 Solder the motor wires to the "Motor 1" and "Motor 2" connections on the PCB.
- 4.4.2 Solder wires from the battery holder to the sliding switch. Note that the red wire is positive and the black wire is negative. Solder a length of wire between the switch and the negative power connection (0V) and another wire to the positive power connection (V+).
- 4.4.3 Attach the sliding switch (on/off switch) to a suitable position on the platform.

5. ELECTRICAL TESTING

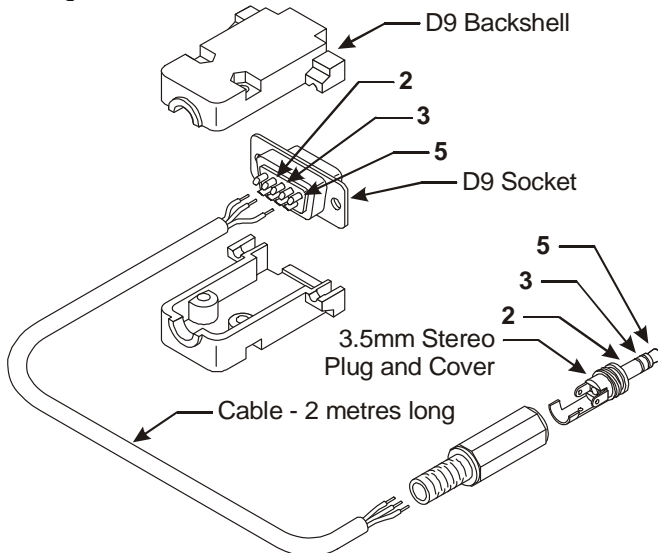
- 5.1 Inspect the soldering for short circuits, poor solder joints and poor "wetting" of the components leads or pads.
- 5.2 Insert four 1.5Volt AA batteries into the battery holder. Move the sliding switch to the "on" position. Check that the LED illuminates (this shows that power is available).
- 5.3 If the LED does not illuminate:
- Check that the batteries are properly inserted in the battery holder.
 - Check that the battery voltage is above 5.5 volts. (If low, replace the batteries.)
 - Check that battery voltage is present between legs 1 and 8 on IC1. Leg 1 should be positive.
 - Check the wiring against the wiring diagram.
 - Check the values of the resistors against the circuit diagram.
 - Check that the LED is the right way around.
 - Check that the LED is working by using a 220 Ohm resistor and 6Volt (battery) power.
- 5.4 Move the switch to "off". Check the ICs orientation - the end with leg 1 is identified with a notch /dimple at one end. Line up the legs of each IC with its IC socket holes and press down firmly. Don't use the letters/numbers on the IC to identify leg numbers.
NOTE: It may be necessary to bend the IC legs to line them up with the socket holes.
CAUTION: Integrated circuits will be damaged if they are installed in the wrong direction or if power supply (battery) connections are reversed.
- 5.5 Position the jumper on the header strip so that it is closest to the electrolytic capacitor. Move the switch to "on". Check that the LED illuminates (this checks that power is available).

NOTE: as the PICAXE has not yet been programmed, the motors will not turn and will not respond when the microswitches are pressed. (At this stage nothing will appear to be working.)

6. PROGRAMMING THE PICAXE

6.1 PICAXE SERIAL CABLE

If required, construct a PICAXE serial cable, as shown.



PICAXE SERIAL CABLE CONSTRUCTION

6.2 INSTALL PICAXE SOFTWARE FOR THE PROGRAMMING EDITOR

6.2.1 Start up and log into your PC. (On some PCs you need to log in as the 'Administrator' to install software. See your systems administrator if you don't have administrative rights to)

6.2.2 Download and run the installation file from the software page at www.rev-ed.co.uk or www.picaxe.co.uk. Follow the instructions to install PICAXE's programming editor software.

6.2.3 Insert the PICAXE serial cable into a 9 pin serial port (COM1), or into a USB to serial adapter. Take note of the "COM" port number that you are using. (The serial ports on some PCs are labelled A and B rather than numbers – usually A is COM1 and B is COM2).

6.3 START PICAXE PROGRAMMING EDITOR SOFTWARE

6.3.1 Click Start>Programs>Revolution Education>Programming Editor to start the software.

6.3.2 If the Options screen does not automatically appear, click the View>Options menu. On the 'Mode' tab select the PICAXE microcontroller you're using (PICAXE-08M). On the 'Serial Port' tab select the appropriate serial COM port (usually COM1) then click OK.

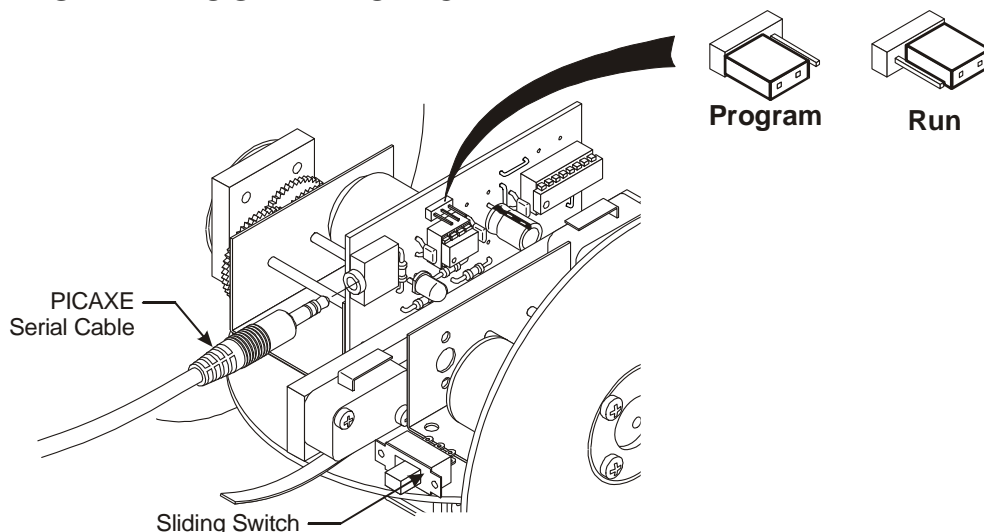
6.3.3 The PICAXE programming editor software is ready to use.

6.4 PROGRAM THE PICAXE

6.4.1 Copy our program into the PICAXE programming editor software.
- the program & flowchart are at the end of this unit.

6.4.2 Save the program. You can keep the original file - for changes use different file names.

6.4 TRANSFER PROGRAM TO PICAXE



CONNECTING THE PICAXE SERIAL CABLE

- 6.4.1 Move the sliding (power) switch to the "off" position.
- 6.4.2 Position the jumper on the header strip in the "program" position. (Refer to the diagram)
- 6.4.4 Connect the PICAXE serial cable between your PC and the stereo socket on the PCB.
- 6.4.5 Move the sliding switch to the "on" position.
- 6.4.6 Run the PICAXE Programming Editor software and transfer our program to the PCB. When the program has finished downloading, one (or both) of the wheels will begin to turn.
- 6.4.7 If the Programming Editor software gives an error message (stating that the program cannot be transferred to the PICAXE):
 - Carry out the checks listed in Section 5 Electrical testing
 - Check that the cable is fully inserted into the stereo socket.
 - Check that the cable is inserted into the correct COM port (usually COM1) on your PC.
 - Check the PICAXE Programming Editor software is set to the correct COM port (as above).
- 6.4.8 Move the sliding switch to the "off" position.
- 6.4.9 Disconnect the PICAXE serial cable from the stereo socket on the PCB.
- 6.4.10 Position the jumper in the "run" position. (Refer to the diagram above.)

7 FUNCTIONAL TEST

NOTE: This test can only be performed after the PICAXE is programmed. The L293D becomes warm if used continuously.

- 7.1 Move the sliding switch to the "on" position. Both wheels should rotate in the same direction. To change the direction of a wheel, swap the wires to the corresponding motor.
- 7.2 Press one of the microswitches. The wheels should rotate backwards, turn in opposite directions and continue in the backwards direction. Press the other microswitch. The wheels should rotate forwards, turn in opposite directions and continue in the forward direction.
- 7.3 Do not press the microswitches for a period of time (about 10 seconds). After this time, the wheels will go through a change of direction sequence.
- 7.4 Move the switch to "off". Press and hold one microswitch, and move the switch to "on". The wheels should move backwards, forwards and then stop. Repeat for the other microswitch - this mode stops the wheels turning before downloading a program. Move the switch to "off".
- 7.5 Switch it "on", put *DIZZY* on the floor and watch it bump into things and change its course.

8 FURTHER DEVELOPMENT

We encourage you to change the program after you've made sure the program works. You can use the following ideas to customise how your *DIZZY* behaves.

NOTE: For programming language details, from the PICAXE "Programming Editor" help menu, open "PICAXE Manual 2 - BASIC Commands".

- 8.1 A flowchart is a graphical representation of the program. When the program encounters an "end" statement, the program will stop until the power is cycled (turned off, then turned on). Work out what condition needs to be filled for the program to end?
- 8.2 A number of parameters are set-up when the program starts running. One at a time, change the values of "wlng" (wait long), "wstp" (wait stop), "wbrf" (wait brief) and "await" (wait timer) in the following program lines.

```
symbol wlng = 2000 'wait long = 2 seconds
symbol wstp = 500 'wait stop = 0.5 seconds
symbol wbrf = 100 'wait brief = 0.1 seconds
let await = 10000 '10 second timer if switch not pressed
```

As a starting point, double and then halve each parameter. (For example, change the value of "wlng" from 2000 to 4000). What effect does each of these parameters have?

- 8.3 When moving forwards, the following pair of lookup tables, after label "skip_forward:" controls what happens after *DIZZY* bumps into an object.

```
lookup aloop, (mstp, mrev, mstp, mlft, mstp), apins
lookup aloop, (wstp, wlng, wstp, wlng, wstp), atime
```

When moving backwards, the following pair of lookup tables, after label "skip_reverse:" controls what happens when *DIZZY* bumps into an object.

```
lookup aloop, (mstp,mfwd,mstp,mleft,mstp), apins
lookup aloop, (wstp,wlng,wstp,wlng,wstp), atime
```

Each pair's top line of lookup tables specifies motor directions and the bottom line specifies the time interval. Available parameters are listed in the "symbols" section of the program. If adding/deleting the number of data entries, in the following program line change the value after "to" by the same amount.

```
for aloop = 0 to 4
```

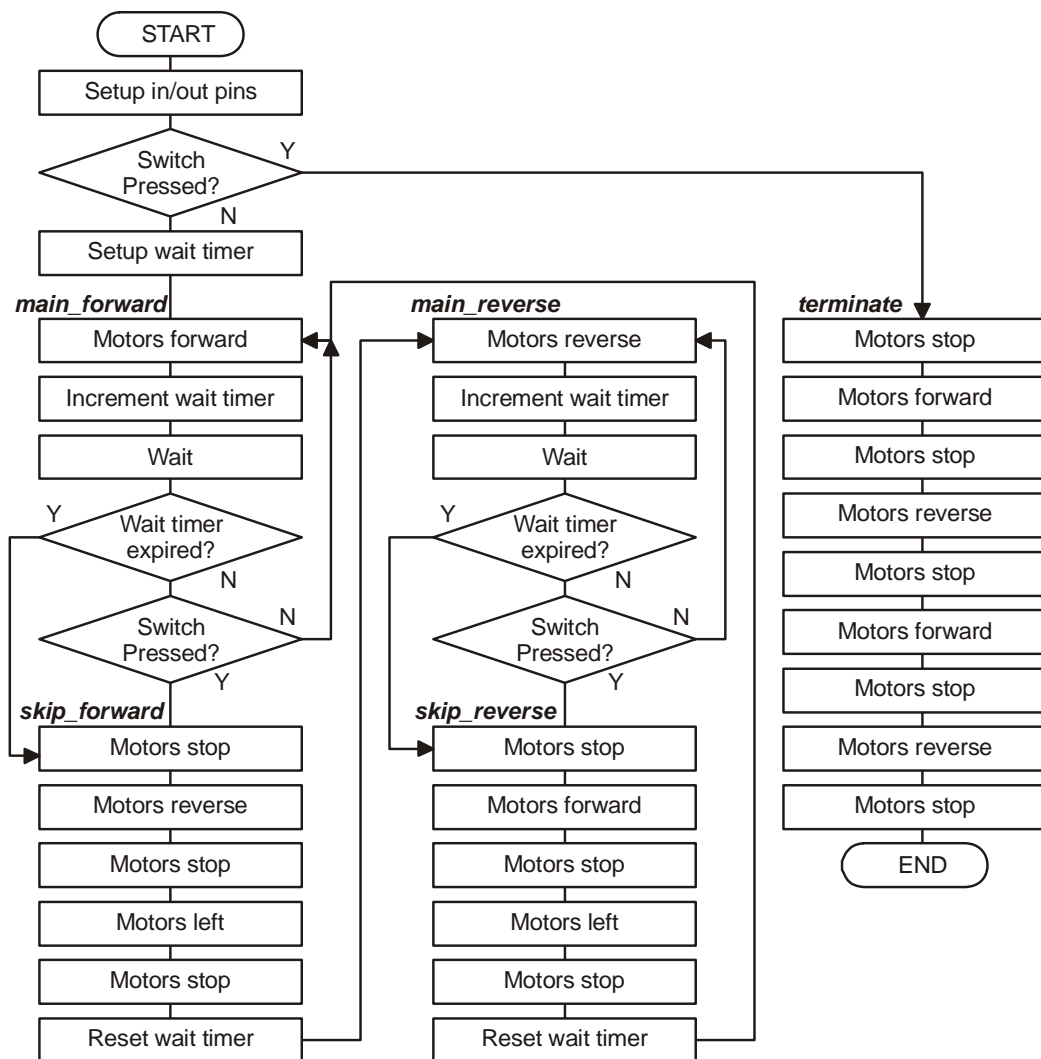
The loop variable "aloop" controls which values will be loaded into "apins" and "atime".

When "aloop" = 0, the first data entry is loaded; when "aloop" = 1, the second data entry is loaded; when "aloop" = 2, the third data entry is loaded; and so on.

Investigate creating new sequences of actions by changing data in the lookup tables. Why should there be a pause before changing the wheel direction?

Congratulations, you have successfully built and customised your own *DIZZY*!!!

HAVE FUN!



PROGRAM FLOW CHART

THE PROGRAM

```
'DIZZY
'(C)2007 PVA Tecwrite - Peter Aleksejevs

'PIN SETTINGS
'pin0 = leg 7 = output = Left motor connection
'pin1 = leg 6 = output = Left motor connection
'pin2 = leg 5 = output = Right motor connection
'pin3 = leg 4 = input = Front and rear switch (1 = pressed)
'pin4 = leg 3 = output = Right motor connection

'Symbols
symbol mfwd = %00010010 'motor forward
symbol mlft = %00010001 'motor left
symbol mrev = %00000101 'motor reverse
symbol mrgt = %00000110 'motor right
symbol mstp = %00000000 'motor stop
symbol wlng = 2000 'wait long = 2 seconds
symbol wstp = 500 'wait stop = 0.5 seconds
symbol wbrf = 100 'wait brief = 0.1 seconds
symbol aloop = b0 'loop counter variable
symbol apins = b1 'which pins are to be output
symbol atime = w1 'time interval for pins to be output(in milliseconds)
symbol acntr = w2 'change direction if switch not pressed for a while
symbol await = w3 'wait timer

'Initial setup
  let dirs = %00010111
  if pin3 = 1 then terminate 'Switch pressed when power turned on.
  let await = 10000 '10 second timer if switch not pressed
  let await = await / wbrf 'Convert into loop cycles

main_forward:
  let pins = mfwd
  let acntr = acntr + 1
  pause wbrf
  if acntr >= await then skip_forward
  if pin3 = 0 then main_forward 'check for switch

skip_forward:
  for aloop = 0 to 4
    lookup aloop, (mstp,mrev,mstp,mlft,mstp), apins
    lookup aloop, (wstp,wlng,wstp,wlng,wstp), atime
    let pins = apins
    pause atime
  next aloop
  let pins = mrev
  pause wlng
  let acntr = 0

main_reverse:
  let pins = mrev
  let acntr = acntr + 1
  pause wbrf
  if acntr >= await then skip_reverse
  if pin3 = 0 then main_reverse 'check for switch

skip_reverse:
  for aloop = 0 to 4
    lookup aloop, (mstp,mfwd,mstp,mlft,mstp), apins
    lookup aloop, (wstp,wlng,wstp,wlng,wstp), atime
    let pins = apins
    pause atime
  next aloop
  let pins = mfwd
  pause wlng
  let acntr = 0
  goto main_forward

terminate:
  for aloop = 0 to 8
    lookup aloop, (mstp,mfwd,mstp,mrev,mstp,mfwd,mstp,mrev,mstp), apins
    let pins = apins
    pause wbrf
  next
end
```

PROGRAM LISTING