Stepper motors in students' projects – when and how to use them

Les Porter and Daniel Stanton

Brunel University Department of Design

Abstract

Developments in simplifying the programming of PIC Microcontrollers have made the devices more accessible to students and to school technology departments. Following our paper on interfacing with the PIC, our research[1] has shown that although many Key Stage 3 and 4 students are now beginning to use PICs in their design and make projects, for many students driving LEDs to simulate output devices is still as far as they wish to get. Our research also shows that recently in many schools much progress appears to have been made and DC motors and other inductive devices are being interfaced with the PIC and both Key Stages 3 and 4. We have been asked to take the initial paper a little bit forward and produce some guidelines for driving stepper motors.

This paper provides a short background on the functionality of stepper motors and goes on to describe some possible ways of driving stepper motors from a PIC.

Introduction

Following our article 'Interfacing Students' Projects with PIC2 Microcontroller' in *The Journal of Design and Technology Education* (Volume 5 No 1) we have had a number of requests to expand on the article and give some hints about 'Stepper Motor Driving'.

Stepper motors (in brief)

Most school projects that require a motor use a simple DC (direct current) motor. For example, to drive a 'buggy' you would use two DC motors, one to drive each wheel. The two motors could be controlled independently to give the directional movement that the buggy requires. This could be both running forward, both running backwards, or one driving forward while the other is driving backwards allowing the vehicle to spin turn.

Unfortunately it is not possible to control exactly how far each motor turns. Two DC motors which are manufactured by the same manufacturer, have the same part number and look identical will perform in slightly different ways. With a DC motor this problem could be overcome by electronically counting the exact number of turns of the shaft and feeding the information back to the microcontroller (the PIC), but that is not as straightforward as it may first seem.

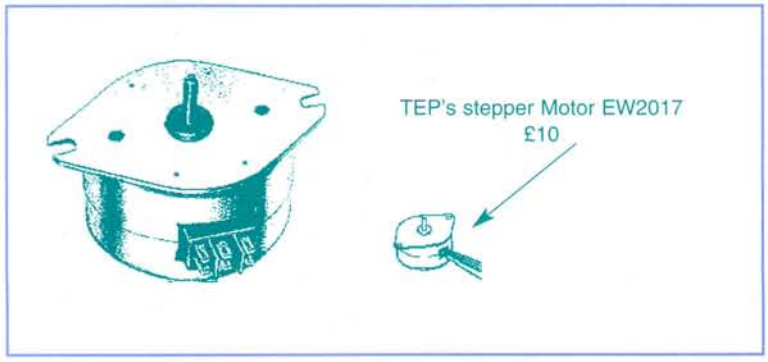

Figure 1: Unipolar stepper motor.

Stepper motors were developed to overcome this problem. If you have ever thought how your printer or your photocopy machine can deliver sheet after sheet of paper very accurately, then the reason for this is that its drive is produced by a stepper motor and not a simple DC motor.

School projects are becoming more and more sophisticated (especially at A' and AS' Level) and we are often getting calls asking for help in the field of developing projects using stepper motors, so we thought we would write this paper to hopefully explain some of the concepts.

Stepper motors are designed to ease the problems of accurate rotational movement control. They are available in two types, either unipolar or bipolar. In both types the motor moves in a series of steps each turning the rotor. Stepper motors are available with different numbers of steps per revolution combinations. The most common stepper motor for school use has a 7.5° step angle rotation of the spindle (48 steps per revolution). The number of steps and speed of the motor is determined by the frequency of the input single applied.

For the purpose of school projects we would suggest that unipolar stepper motors are easier to work with and in this paper only describe their use.

Stepper motors are not so easy to drive as are normal DC motors; they cannot be simply 'switched on', but instead must be turned on by sending a series of pulses to the four coils that make up the motor.

The motor has four coils (the stator) arranged around a permanent magnet rotor. When these coils are switched on and off in the correct combination, the rotor moves in increments or steps.

The rotor is constructed from a number of permanent magnets that obviously have fixed North and South poles (see Figure 2).

If two of the opposite stator electromagnets are turned on, then these draw the rotor to that point. If that pair of electromagnets are then turned off and the opposite pair are turned on then the rotor is pulled around by one step. The rotor is then held rigidly at the new position until the next pair of stator coils is turned on.

To make the diagram clear (Figure 2) only two pairs of coils are shown, not the usual four pairs.

Driving the stepper motor

Until recently the ubiquitous IC, SA1027, was mainly used for driving stepper motors in school projects. This has now become an obsolete IC so we have had to look for other methods of creating the drive. The PIC microcontroller has become the obvious choice for doing this.

The stepper motor can be made to run continuously by a four-line program, which can be put into an endless software loop which will just repeat over and over again at a predetermined speed. See Figures 4a and 4b as an example of this for running both forward and backward.

For most of your students, that is not what they would want to do with a stepper motor. They could produce a 'forward/backward' sequence much more simply and cheaply by using a DC motor. In our earlier paper we made a few comments about PIC programming but in the main discussed interfacing the microcontroller with students' projects, not the programming. In that paper we talked about our research programme and we briefly discussed simplified PIC programming systems.[3] These simplified systems give the PIC user the ability to program the microcontroller without having to learn the intricate commands of the programming language, and then when having written the program using an assembler that the computer understands and finally 'blowing' it to the PIC. If you would like to learn about programming a PIC from first principles John Morton's new book, *PIC – Your Personal Introductory Course*[4] will provide you with all you want to know. Our research shows that for many students all they want is to get the PIC to control the systems that they have designed so the simplified programming systems prove splendid for those types of task.

Simplified programming

For the purpose of driving the stepper motor, we look at two approaches in this paper. We use a straightforward program to illustrate such programming. We look at PIC Logicator and PICAXE to provide the following project outcomes:

- press button to start
- stepper motor moves forward
- some limit switch is sensed
- stepper motor drives backwards until...
- home position is found.

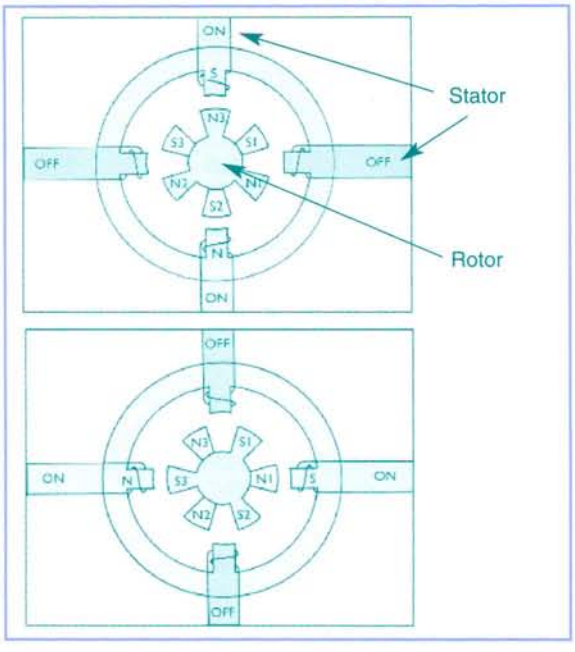

Figure 2: Stepper motor showing stator and rotor arrangement.

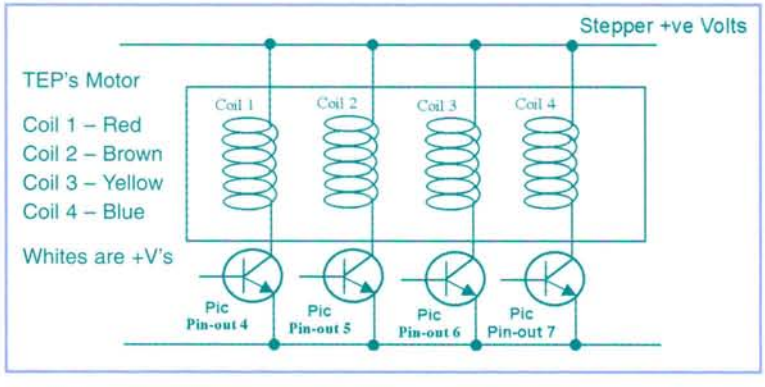

Figure 3: Stepper motor stator coil arrangement.

Figure 4a: Stepper motor coils – running forward.

	In words				Digitally			
	Coil 1	Coil 2	Coil 3	Coil 4	Coil 1	Coil 2	Coil 3	Coil 4
Step 1	on	off	off	on	1	0	0	1
Step 2	on	off	on	off	1	0	1	0
Step 3	off	on	on	off	0	1	1	0
Step 4	off	on	off	on	0	1	0	1

Figure 4b: Stepper motor coils – running backwards.

	In words				Digitally			
	Coil 1	Coil 2	Coil 3	Coil 4	Coil 1	Coil 2	Coil 3	Coil 4
Step 1	off	on	on	off	0	1	1	0
Step 2	on	off	on	off	1	0	1	0
Step 3	on	off	off	on	1	0	0	1
Step 4	off	on	off	on	0	1	0	1

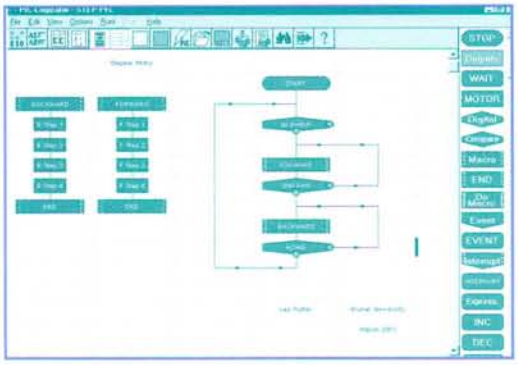

Figure 5. PIC Logicator flow chart for driving a stepper motor.

In PIC Logicator (see Figure 5) we use a flow chart with its main tree having six elements which include two macros. Each of the macros has six elements.

You will notice that in each of the macros (Figure 6a and Figure 6b) each of the step lines are those that we looked at in Figures 4a and 4b.

This is just the start of the program to give students the feel of stepper motor driving. Between each of the step commands in the macros short time delays (wait commands) are required to control the speed of steps of the motor as running like this the microcontroller's internal clock would be too fast to drive the motor.

We have not tried to give examples of digital switches as these could be any device, such as a microswitch or photo-diode, that will give a digital signal to the PIC. We have given some advice on inputs in our earlier paper.

Using PICAXE

PICAXE provides a number of ways in which we could use the microcontroller. This device is based on the 28 pin PIC16F872 micro-controller and has its ports predesignated to provide eight digital outputs, eight digital inputs, four analogue inputs and two serial interface pins. It has the ability to be written to in BASIC (similar to Basic Stamp programming) or can be programmed via flowcharts with Crocodile Clips software.[5]

The power of PICAXE is its simplicity. No programmer, eraser or complicated electronic system. The PIC is programmed via a three-wire connection to the computer's serial port.

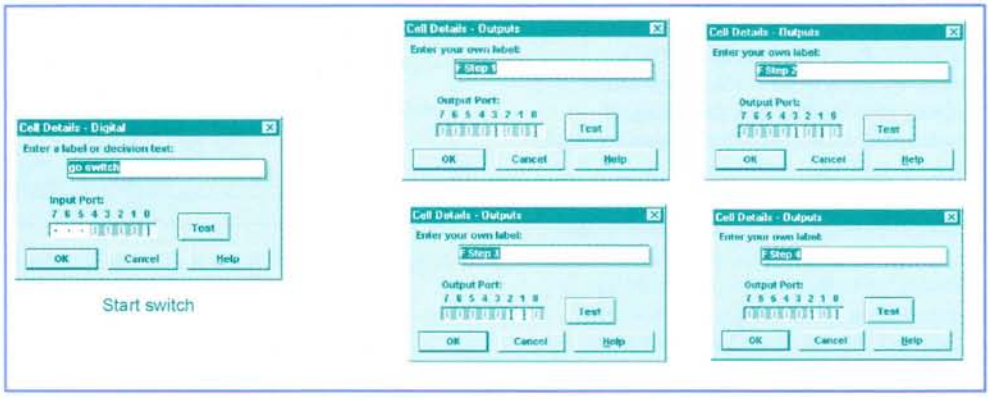

Figure 6a: Forward macro.

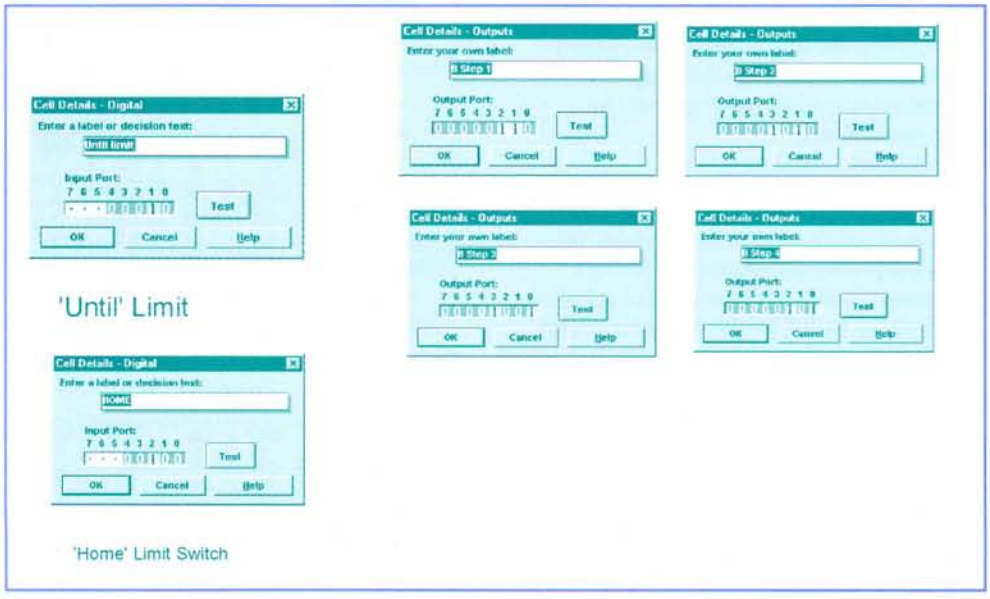

Figure 6b: Backwards macro.

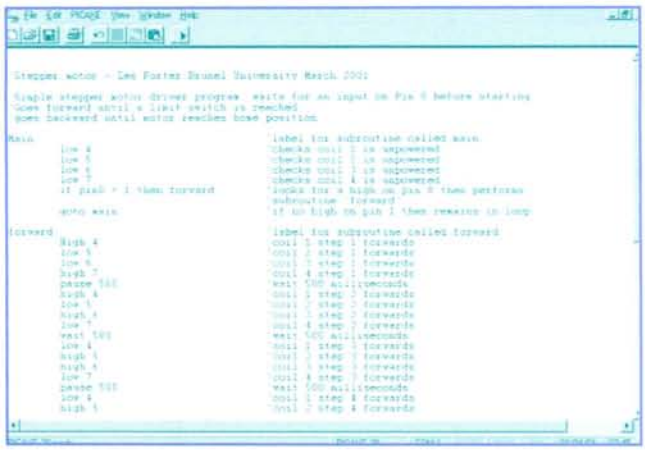

Figure 7: Screen grab of PICAXE stepper program.

As you will see, using PICAXE for our simple stepper motor driver is perhaps not the most economical way of using such a powerful PIC, but perhaps our simplistic approach will encourage students to think deeper about possible uses of the IC.

In the example that follows, we assume that the stepper motor coils are connected to PICAXE 'outputs 4, 5, 6 and 7' and that the switch input is connected to 'input 0'. The limit switch is connected to 'input 1' and the home position is connected to 'input 2'. The length of the pause routines can be altered according to the required speed of the stepper motor.

Sample program (see Figure 7 for screen grab of the program).

The program listing for this program is presented as Appendix 1.

Using the L293D to drive a stepper motor

The PIC provides low current outputs and can source (give out) a maximum of around 50mA. This means that without 'interfacing' LEDs are the only feasible output devices that can be used. LEDs are fine to simulate an output function, but most students will want to drive real devices. The stepper motor is one such device and the diagram below (Figure 8) shows the way the L293D can be used to drive a stepper motor.

The pin configuration numbers in Figure 8 refer to the PICAXE 28 pin PIC but by visiting the Planet Microchip web site[6] it is possible to find the pin designations for all the family.

Some stepper motors draw quite a large current. The L293D will allow motors that draw up to about half amp to be driven, but by doubling up the chip it is possible to drive about 1 amp. To double the chips up connect two IL293Ds in parallel i.e. out pin 1 on the first L293D goes to in pin 1 on the second L293D, out pin 2 on the first L293D goes to pin 2 on the second L293D and so on.

The L293D will become quite hot with continuous use. If possible leave a large pad on your PCBs connecting all the 0 Volt pins of the chip. Alternatively a heatsink bonded to the top of the chip will keep it cool.

If we take a second look at Figure 3 you will notice that there are transistors used to drive each of the four coils of the stepper motor. As we discussed in our earlier paper[7] the PIC can only source (give out) about 50mA which would not be enough current to drive a

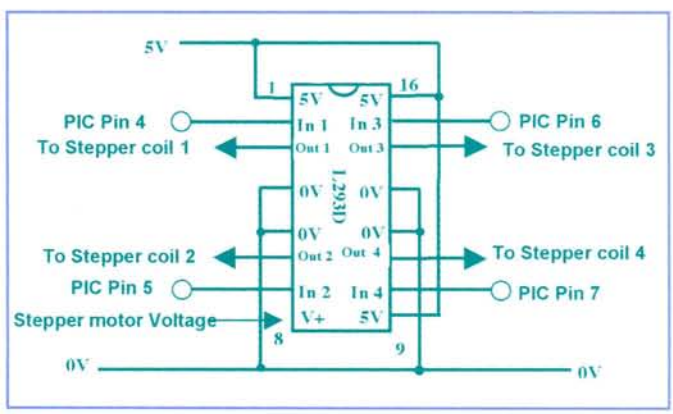

Figure 8: Using the L293D IC to drive a stepper motor.

stepper motor. The transistors in this diagram are amplifying the current and that is really what the L293D IC (above) is doing. We could do exactly the same using a Darlington pair transistor arrangement as an input on each of the coil lines of the stepper motor.

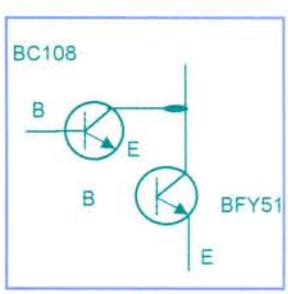

Figure 9: A typical Darlington Pair arrangement.

In order to make PIC outputs drive efficiently, we need both high gain and quite a high current and to do this we could use two transistors linked together, one with high gain and one with a high power rating (Figure 9 for example).

Instead of using two separate transistors, we would suggest using a transistor called BCX38B which looks like a single transistor, but is in fact two transistors set up as a Darlington Pair in a single package which gives a high gain and a high current capability. This discrete device has a collector-current rating of 800mA and will easily drive output devices such as small DC motors, but for some stepper motors it may be a little under-powered (or we could use a really small transistor such as the ZTX651 which would do the same job).

As an alternative to individual coil drivers (or the L293D for that matter) we could use an integrated IC to do the task. The ULN2803A

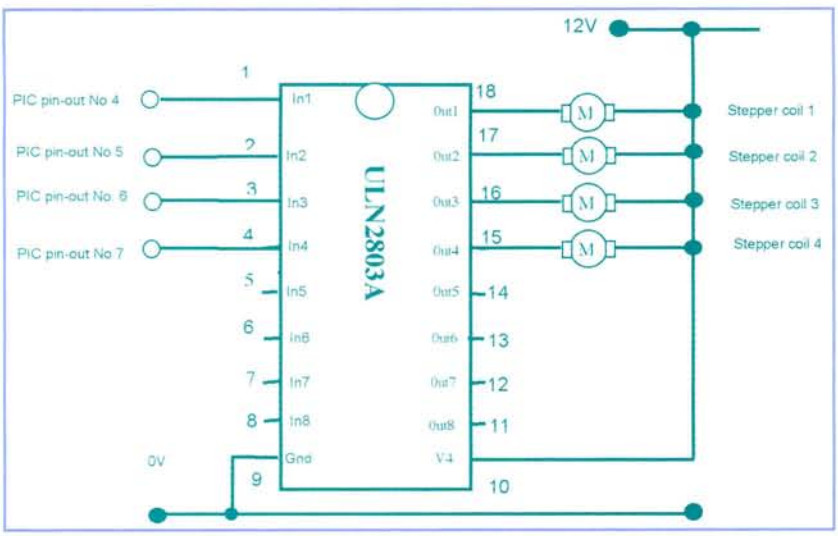

Figure 10: ULN2803A used to control stepper motor (again diagram or PIC pin-outs refer to PICAXE pin configuration).

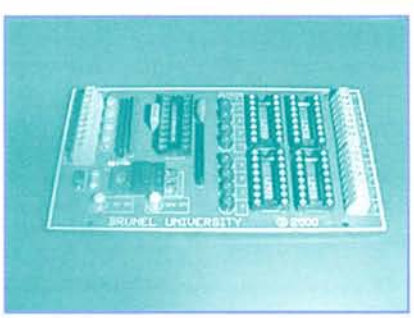

Figure 11: 16F84 interface board.

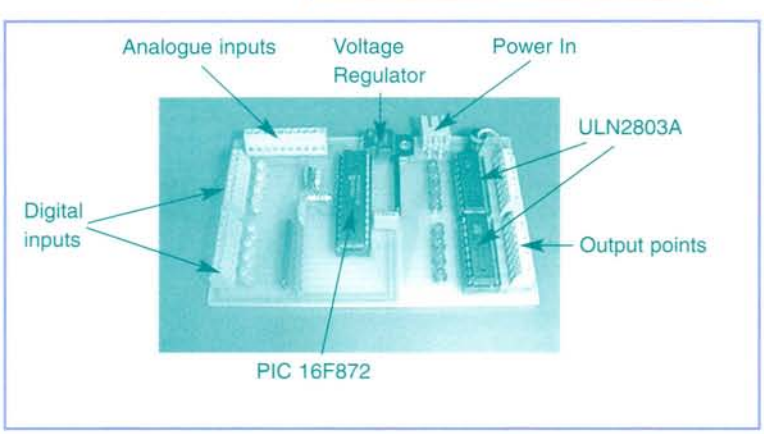

Figure 12: 16F872 interface board.

(Figure 10) is the perfect package for this. This is simply a single chip that contains 8 Darlington transistors similar to BCX38B, the chip also contains back EMF suppression diodes so therefore no external diodes are required. Each Darlington Pair in the IC has a collector-current rating of 550mA which makes it a very useful device for driving inductive components. This really is a versatile IC and has lots of uses, but in Figure10, above, we give you an example of how it could be used to drive a stepper motor.

As part of our research programme we have developed interface boards which could be used for students to interface their PIC programs to stepper motors. Figure 11 shows the interface board that is available for the Logicator PIC 16F84 driving from L293D chips and Figure 12 shows the board available to drive the PICAXE 16F872 chip using the ULN2803A IC.

These boards are available from the Design and Technology Research Bureau at Brunel University[7] and can be supplied either as unpopulated circuit boards or part of fully populated boards according to the needs of students.

Conclusions

It is important to remember that stepper motors are designed for specific applications where cheap/accurate positioning and controllable speed is required. When students are developing projects that require some sort of electromechanical control, think carefully about what they really are trying to achieve and see whether they could use a DC motor first and measure its relative position using micro switches, as this approach tends to be cheaper. If they decide that a stepper motor is the way to go then remember that the whole system will require a datum to operate from (the home position) or the stepper motor will lose its way.

Another, sometimes overlooked, consideration is the torque required to operate the system. Cheaper stepper motors have little holding torque and will often require adding a gearbox to provide the required driving force. This obviously increases the cost of the project and reduces the speed of the motor.

Although the cost may be higher, stepper motors show students a good method of providing positional accuracy and with the widening availability of using microcontroller based systems in schools, allows students to gain a worthwhile appreciation of rotational control.

Appendix 1

```
'Simple stepper motor driver program: waits
for an input on Pin 0 before starting.
'Goes forward until a limit switch is reached
'goes backward until motor reaches home
position

Main: 'label for subroutine called main
      low 4  'checks coil 1 is unpowered
      low 5  'checks coil 2 is unpowered
      low 6  'checks coil 3 is unpowered
      low 7  'checks coil 4 is unpowered
      if pin 0 = 1 then fwd 'looks for a
      high on pin 0 then performs subroutine
      'forward'
      goto main     'if no high on pin 0 then
      remains in loop
fwd:   'label for subroutine called
      [fwd]forward
      High 4 'coil 1 step 1 forwards
      low 5  'coil 2 step 1 forwards
      low 6  'coil 3 step 1 forwards
      high 7 'coil 4 step 1 forwards
      pause 500 'waits 500 milliseconds
      high 4 'coil 1 step 2 forwards
      low 5  'coil 2 step 2 forwards
      high 6 'coil 3 step 2 forwards
      low 7  'coil 4 step 2 forwards
      pause 500 'waits 500 milliseconds
      low 4  'coil 1 step 3 forwards
      high 5 'coil 2 step 3 forwards
      high 6 'coil 3 step 3 forwards
      low 7  'coil 4 step 3 forwards
      pause 500 'waits 500 milliseconds
      low 4  'coil 1 step 4 forwards
      high 5 'coil 2 step 4 forwards
      low 6  'coil 3 step 4 forwards
      high 7 'coil 4 step 4 forwards
      pause 500 'waits 500 milliseconds
      if pin 1 = 1 then bkwd 'looks for limit
      switch, if no stays in subroutine
      'forward'
      goto fwd 'stays in loop 'forward' unless
      limit switch is found
bkwd: 'label for subroutine called [bkwd]
      backward
      low 4  'coil 1 step 1 backwards
      high 5 'coil 2 step 1 backwards
      high 6 'coil 3 step 1 backwards
      low 7  'coil 4 step 1 backwards
      pause 500     'waist 500 milliseconds
      high 4 'coil 1 step 2 backwards
      low 5  'coil 2 step 2 backwards
      high 6 'coil 3 step 2 backwards
      low 7  'coil 4 step 2 backwards
      pause 500     'waits 500 milliseconds
      high 4 'coil 1 step 3 backwards
      low 5  'coil 2 step 3 backwards
      low 6  'coil 3 step 3 backwards
      high 7 'coil 4 step 3 backwards
      pause 500     'waits 500 milliseconds
      low 4  'coil 1 step 4 backwards
      high 5 'coil 2 step 4 backwards
      low 6  'coil 3 step 4 backwards
      high 7 'coil 4 step 4 backwards
      pause 500     'waits 500 milliseconds
      if pin 2 = 1 then main 'looks for
      home limit switch, if no stay in
      subroutine 'backward'
      goto bkwd 'stays in loop 'backward'
      unless limit switch is found
```

Notes

1 A national research project investigating teaching and learning within the systems and control strand of the design and technology National Curriculum.

2 PIC= Peripheral Interface Controller. Developed by Microchip Technology inc.: Contact techsupport@microchip.com

3 TEP's (Technology Enhancement Programme) Chip Shop (Starter Kit £99 + optional software £32)

Economatics – PIC Logicator (£125 including software)

Icon Soft Electronics – Nottingham Trent University (£98 single user license)

(Since our earlier paper add Revolution Education's PICAXE Microcontroller Programming System. Business Education Centre, Innova Park, Mollison Avenue, Enfield, Middlesex. EN3 7XU Tel: 020 8350 1315.

Kids Chip from Data Harvest (http://www.data-harvest.co.uk/) is also new to the scene and worth a look at. This innovative system offers an exciting, open-ended design opportunity using the 'flowol software. In addition to it use as a systems and control activity, the Kids Chip PIC system could be used as part of a textiles, resistant material, graphics or electronics product design.

4 Morton, J. (2000) PIC – *Your Personal Introductory Course*: Newnes ISBN 0 7506 3932 6

5 Crocodile Clips = Crocodile Clips Ltd11 Randolph Place Edinburgh Eh3 7TA Tel: +44 (0)131 226 1511 Fax: +44 (0)131 226 1522 Contacts Andrew Hall email: Andrew.Hall@crocodile-clips.com http://www.crocodile-clips.com

6 Planet Microchip web site – http://www.microchip.com/10/index.htm – for information on all the pin configurations for the PIC family of microcontrollers.

7 The Design and Technology Bureau, Brunel University, Runnymede campus, Egham Surrey. TW20 0JZ. Http://www.brunel.ac.uk/depts/des/bureau or e-mail Les.Porter@brunel.ac.uk