

# CYBER PET

BY KEIRON MCGEEVER

## BUILDING AND PROGRAMMING A SIX LEGGED WALKING ROBOT

Last year 'Mac' described a way of making a walking two legged robot (well it makes us laugh) and showed how difficult it is to make a stable robot that can walk. This time he shows us how to make a much more reliable walking machine.

In the article "An Intelligent Robot" (issue 5 of News and Views) we featured how to build a PIC controlled two wheeled buggy that was programmed to be light seeking. The challenge is to go deeper into the programming this time. Well the same programme and control on the walking robot produces some much more interesting behaviour, and modifying the programme a little gives the robot some animal-like behaviour that may have you talking to it like a pet!

### WHAT YOU NEED

Apart from the components for the robot (see Free Kits) you will need:

Description	Cat No.
» 1 x PIC High Power Project Board,	CHI 031
» 1 x L293D motor driver chip,	L293D
» 1 x PIC16F627 chip,	CHI 012
» 2 x large pear cams,	TG1 017
» 2 x LDR's, ORP 12	EL1 002
» 2 x micro-switches,	ES1 004
» 2 x battery snaps,	EC1 004
» 2 x 6 volt battery holders,	EC1 003
» 1 x double pole, single throw switch (it can be a DPDT)	ES1 002
» 2 x resistors (5.6k)	ER2 5K6
» 2 x Economy Gearboxes inline motor mount Ratio 1:120	EW2 ECK4

(All available from the Teaching Resources Catalogue from Middlesex University [www.mutr.co.uk](http://www.mutr.co.uk)).

### TO CNC OR NOT CNC?

To build this robot it is useful to have a CNC router or milling machine to make the parts, or know someone who does. You can access the drawings as a download file from the TEP website at the bottom of this page.

The body of the robot is made from 2mm thick acrylic sheet and can be cut out by hand, but it is much more accurate to use a CNC router or miller. It is all profile milling and therefore manufacturing time is fairly short, depending on the machine you are using. The original designs were done on Pro/Desktop and converted to Techsoft 2D files and the first prototype (a single motor design) was cut out on the TEP milling machine. Later versions (the two motor designs) were produced on the Laser Cutter at Middlesex University with thanks to Professor John Cave.

### FREE KITS

If you have not got access to a CNC machine we have a few 'laser cut' kits (Only body parts available - you will still need the electronic components) available on a first come first served basis by contacting Mark Watson at:

**Free Robot Kit**  
**Technology Enhancement Programme**  
**International Manufacturing Centre**  
**University of Warwick**  
**Coventry**  
**CV4 7AL.**

### ASSEMBLY

The advantages of CNC manufacture is that you end up with a kit of very accurate parts that only need bolting together, apart from a little line bending of the main body part to make the mounting lugs for the motors and the legs (see Fig 1).

» Start by line bending the main body, making sure that you bend the mounting lugs for the legs and the motors in the right directions (Fig 1).

» The motors are mounted onto the lugs using either two 6BA nuts and bolts or two self-tapping screws

(2.5mm to 3mm dia.). The mounting lugs are designed to accommodate half the locating spigot on the motor and have the terminals on the inside of the robot (Pic 2).



[Fig 1]



[Pic 2]

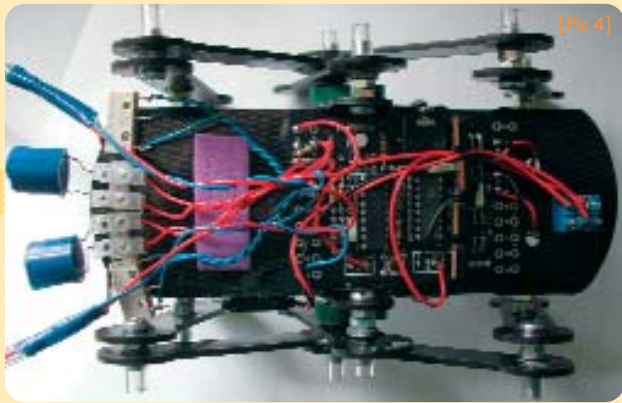
» Next fit the micro-switches and the connector blocks (for the LDR's) to the front of the robot with suitable self-tapping screws and the on/off switch to the back of the robot with the toggle on the motor side of the body. (Pic 1)



[Pic 1]

» Fit the motor driver chip onto the PIC board and attach the Black wire of the battery snap to the right hand power connector labelled PWR G on the board (Pic 4).

» On the right hand side of the board attach wires to output pins 4, 5, 6, and 7 labelled motor A and motor B.



[Pic 4]

On the left hand side of the board inputs 0 and 1 are the analogue inputs and have three connecting pads each and inputs 2, 3, 4, and 6 are the digital inputs with two connecting pads each.

» Attach wires to inputs 2 and 3 and then to the micro-switches using the common and normally open pins on the switches.

The LDR's are a little more difficult to attach as they use the 5.6k resistors to set the sensitivity of the inputs.

» The 5.6k resistor is attached to the outer pad of input 0 and the centre pad and then two wires, to attach the LDR to, are fixed to the centre pad and the inner pad (note, the resistor and a wire both go into the centre pad).

» Do the same for input 1. The wires are then attached to the LDR with the double connector block which is fixed to the base.

» Now fix the PIC board to the chassis with double sided sticky pads and connect the wires for motor A to the left hand motor and the wires for motor B to the right hand motor via the holes in the rear of the body.

» Fit the battery box to the underside of the body, just in front of the motors; connect the Red wire of the battery snap to the DPST switch and the switch to the PIC board power connector labelled PWR V+.

» At this point it is a good idea to test your circuit using the Test Programme Procedure.

» Now start to assembly the rest of the robot by taking each of the large pear cams and screwing a 20mm x 4mm dia machine screw through the hole in the small end of the cam (In fact we are using the cams as cranks).

» Fit the cams to the motor drive axles, to do this you will have to drill out the hole in the large end of the cam to 5.5mm dia. You may also have to put some spacer washers (3) on the motor axle first to prevent the cam fouling the body of the robot (take care not to overstress the main body of the robot whilst doing this). This makes a crank with which to drive the legs.

» Fit a 20mm x 4mm dia bolt to the middle leg lug with two locked nuts (see Fig 2).



[Fig 2]

» Do the same for the front and rear leg lugs.

» Identify the middle legs (they have slots in the top) and fit the slots over the top lugs and the middle hole to the crank (you will need to put three spacer washers at the top and one in the middle).

» Fit another washer over the top of the slider and fix it in place with either a locking nut or a short length of 6mm dia polythene tubing.

» Identify the front and rear legs and fit these to the fixing lugs in the same way not forgetting to include the spacer washers.

» The top of the front and rear legs are connected to the crank in the middle with the connecting rods (the ones without holes in the middle). At the front

you will require one spacing nut and a washer and at the back two spacing nuts and a washer. Hold all the legs in place with either a locking nut or length of tubing.

Use the Test Programme Procedure again to check that everything is working properly and grease up all the joints. Do not make any of the moving joints too tight as the robot will work better with some slack in the system.

» Finally bend the actuating arms on the micro-switches to 45 degrees about halfway along their length and push a 100mm length of 6mm polythene tubing onto them that you have bent to curve down in front of the robot but not quite reach the floor. These will act as feelers and stop the robot bumping into obstacles. A further enhancement is to fit LED's into the ends of the tube, but beware of battery drain (see Pic 1)

## TEST PROGRAMME PROCEDURE

You are now ready to test your robot. Using the "Chip Factory", power up and select programme New and chip 18L (the code for the 16F627) and input the following test programme:

00	out	080	both motors go forward
01	If 3 on	out 016	left motor forward right motor stopped
02	If 2 on	out 064	right motor forward left motor stopped
03	goto	00	return to start
04			end programme

Burn this programme onto your PIC and fit it to your robot. When the robot is powered up check that both motors are running in the same direction and forwards. If one of the motors is running in reverse and the other forwards change the polarity on the reversing motor by swapping the wires round.

Now press the left hand micro-switch and the right hand motor should stop and the left hand motor go forward. The opposite should happen for the right hand micro-switch. Adjust the positions of the switches if this does not happen.

## BASIC LIGHT SEEKING PROGRAMME

00	out	080	Both motors on forward
01	if a = b	goto 04	If both Analogue light levels equal goto 4
02	if a > b	goto 07	If analogue light higher on lhs goto 7
03	if b > a	goto 10	If analogue light higher on rhs goto 10
04	if 2 on	goto 10	If lhs microswitch bumped goto 10
05	if 3 on	goto 07	If rhs microswitch bumped goto 7
06	goto	00	Goto 00
07	out	016	Motor B forward 'left turn'
08	wait	010	Turn for 1 second
09	goto	00	Goto 00
10	out	064	Motor A forward 'right turn'
11	wait	010	Turn for 1 second
12	goto	00	Goto 00
13			

This programme will make your robot autonomous, but only test it by placing it on the floor, otherwise damage could occur.

In the above programme, a and b are the analogue inputs connected to the LDR's and the chip is being asked to compare the light level falling on each LDR. If you have positioned the LDR's so that they look to the left and the right, then the robot will only go in a straight line if the light levels are the same.

The micro-switches act as bumpers and if the robot bumps into an obstacle it will turn away from it. This is a very simple programme but it will make the robot perform some very complicated behaviour.

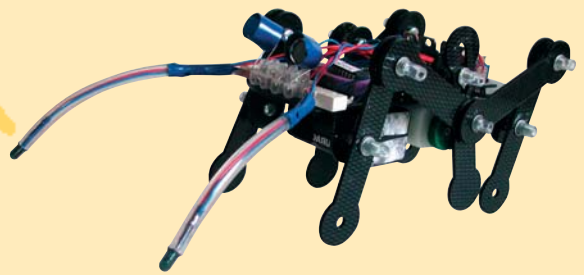


## USING VARIABLES IN PIC PROGRAMMES

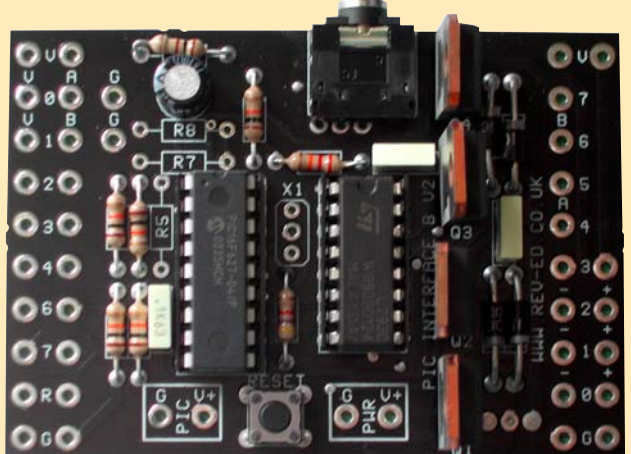
[Serious PIC programming]

```

00  Let x = 000
01  Let y = 000
02                                     Out 000
03  Let y = a plus b
04  Let y = y plus 1
05  If x > y                           goto 08
06  Let x = a plus b
07                                     goto 02
08                                     out 080
09                                     wait 010
10                                     out 000
11  let y = 014
12  let x = 080
13                                     goto 44
14  let x = 080
15                                     out x
16  if 2 on                             goto 22
17  if 3 on                             goto 31
18  if a = b                           goto 14
19  if a > b                           goto 38
20  if b > a                           goto 41
21                                     goto 14
22  let y = 025
23  let x = 160
24                                     goto 44
25                                     wait 020
26  let y = 029
27  let x = 096
28                                     goto 44
29                                     wait 020
30                                     goto 11
31  let y = 034
32  let x = 160
33                                     goto 44
34                                     wait 015
35  let y = 029
36  let x = 144
37                                     goto 44
38  let y = 062
39  let x = 096
40                                     goto 44
41  let y = 062
42  let x = 144
43                                     goto 44
44                                     out 000
45                                     wait 001
46                                     out x
47                                     out 000
48                                     out 000
49                                     out 000
50                                     out 000
51                                     out x
52                                     out 000
53                                     out 000
54                                     out 000
55                                     out x
56                                     out 000
57                                     out 000
58                                     out x
59                                     out 000
60                                     out x
61                                     goto y
62  let y = a plus b
63  if y < 010                          goto 00
64  if y > 180                          goto 00
65                                     wait 001
66                                     goto 15
    
```



I did not write this programme straight off, it evolved slowly and painfully from the original programme. Some of it worked first time and some parts of it had to be re-written several times to sort out the bugs. It uses the variables x and y as this allows the Chip Factory to have subroutines (mini programmes which perform repetitive tasks).



▲ [Fig 3] PIC Controller Board

## WHAT DOES IT DO?

The first two lines of the programme set the variables x and y to zero, this is important as the values of x and y are stored in non-volatile memory in the chip and will remain even when the chip has been switched on and off. When the programme is running the values of x and y are changed many times in a second depending on which routine is running and having an incorrect starting value could cause lots of problems.

## GETTING STARTED

Lines 02 to 07 are a loop they are there to stop the robot from starting as soon as the switch is turned on. Instead the loop measures the ambient light level in the robots environment and waits for a significant change in level before branching off to the main part of the programme. In much the same way as a cat flea on the rug will wait for the shadow of the cat to pass over it before jumping on board.

In this loop a and b are the light sensors attached to the analogue inputs on the chip. On the first time through the loop y is made equal to the outputs from a and b, the light level in the room. The value of y is then increased by 1 (line 04), which ensures that y will be greater than x, which on the first time round should be equal to zero. So at line 05 the loop will not branch but continue to line 06 where x is made equal to the outputs from a and b before looping back to the start of the loop at 02 (line 07). The second time through the loop y should still be greater than x and the loop should still not branch and the robot will still not start. This state will continue until there is a significant and sudden change in the light level falling on the sensors causing the loop to branch to line 08 and the robot to start forward with the out command 080 (see the out commands table).

We think you will agree that's quite enough programming until issue 7. We will also be reviewing the forthcoming PIC programming book from TEP in that issue and no surprises Keiron is one of the authors. If you really cannot get it to work Keiron will be pleased to hear from you and can be contacted at: [kmcgeeve@cercot.demon.co.uk](mailto:kmcgeeve@cercot.demon.co.uk)

