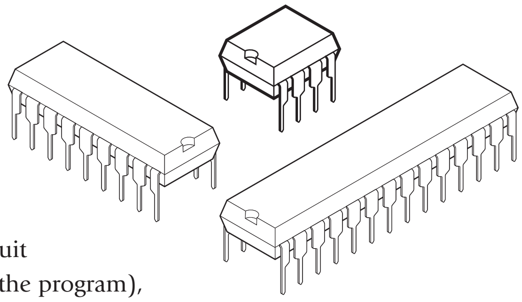
ELECTRONIC PETS

What is a microcontroller?

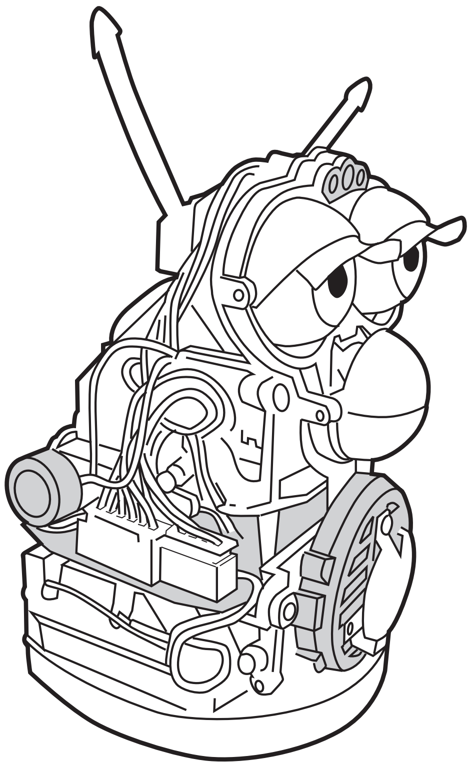


A microcontroller is often described as a 'computer-on-a-chip'. It can be used as an 'electronic brain' to control a product, toy or machine.

The microcontroller is an integrated circuit ("chip") that contains memory (to store the program), a processor (to process and carry out the program) and input/output pins (to connect switches, sensors and output devices like motors).

Microcontrollers are purchased 'blank' and then programmed with a specific control program. This program is written on a computer and then 'downloaded' into the microcontroller chip. Once programmed the microcontroller is built into a product to make the product more intelligent and easier to use.

Example use of a microcontroller.



In most toy shops there are now lots of intelligent 'toys' available. These toys can move, make noises and respond to being touched or being placed in a dark place.

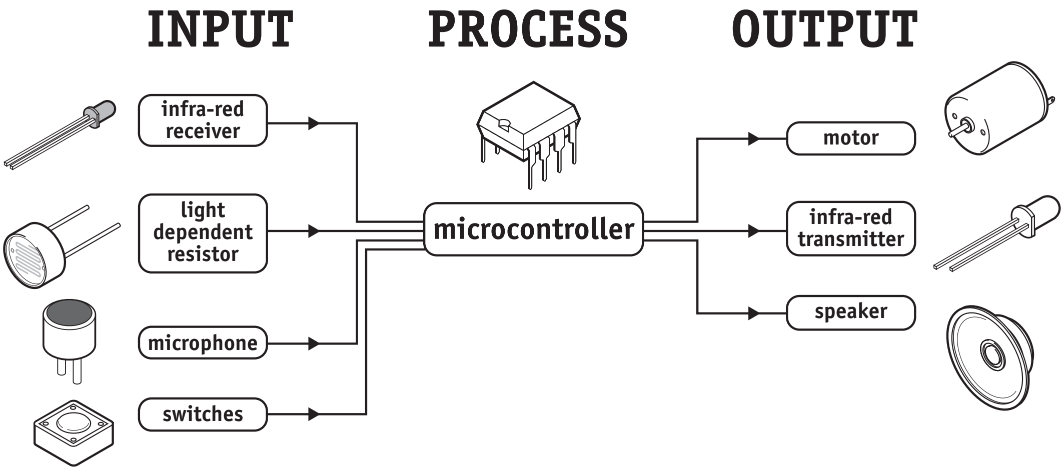
An example of one such toy is the 'Furby' made by Tiger Electronics. The Furby has a microcontroller as it's electronic 'brain' and reacts (to being touched or being placed in the dark) by moving or making sounds.

The Furby reacts to the outside world by sensors and switches. It has a push switch on it's front and back, a microswitch in it's mouth and a light sensor (LDR) between it's eyes. There is also a microphone on it's side so that it can detect sounds.

The Furby's movement is provided by an electric motor. It also has a speaker to generate sounds and an infra-red LED so that it can send signals to other Furby's that might be nearby.

The microcontroller is the 'brain' of the creature. Microcontrollers are powerful electronic components that have a memory and can be programmed to switch things on and off in a special sequence. The microcontroller in the Furby, for instance, has been programmed to switch off the motor and speaker when the light sensor has detected it to be dark (i.e. the Furby goes to sleep).

Block Diagrams



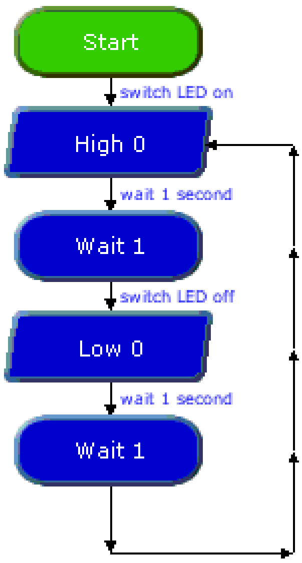
The electronic system that controls the Furby toy can be drawn as a 'block diagram'. The light sensor, microphone and switches all provide information to the microcontroller, and so these are known as 'inputs'. The microcontroller then 'decides' how to behave and may then operate the outputs e.g. make the motor move or generate a sound with the speaker. If another Furby is nearby they can 'communicate' by infra-red signals transmitted, and received, by the microcontroller.

What is the PICAXE System?

The microcontrollers used in devices such as the Furby can be difficult to program, as they generally use a complicated programming language called 'assembler code', which can be quite difficult to learn.

The PICAXE system makes the microcontrollers much easier to program. The control sequence can be drawn (and simulated) on the computer as a flowchart, or written in a simpler programming language called BASIC. This makes it much easier to use the microcontroller as the complicated 'assembler code' does not need to be learnt.

A Logicator flowchart is shown here. It flashes a light on and off every second.



Building your own Electronic Pet

Design Brief

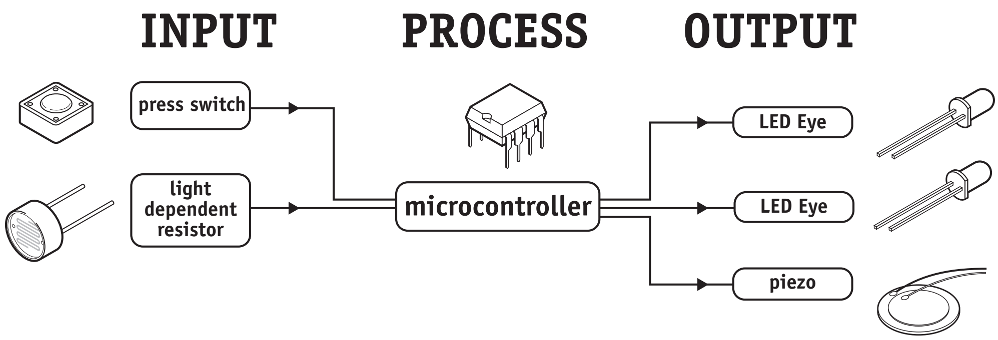
Design and make an electronic 'cyberpet' toy. The cyberpet must be programmed with it's own 'personality' so that it reacts in a unique way.

Design Specification Points

1) The design will use a PICAXE-08 microcontroller as it's brain.
2) The design will include LED eyes, a piezo sounder to generate noises and could also optionally use a motor to generate movement.
3) The design will be able to react to touch and changes in light level.
4) The cyberpet can be designed as a flat '2-dimensional' panel or as a full '3-dimensional' creature.

Block Diagram

The block diagram for your cyberpet may look like this:



Designing Your Cyberpet

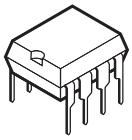
Your cyberpet can be any shape or size you choose. You may like to design the 'face' of the pet using a computer graphics program or by drawing by hand. You could scan in a picture of a living animal, or design a completely new 'robot' animal.

Things to think about:
The electronic components will need to be mounted inside (or underneath) your cyberpet. The LEDs and LDR will need to poke through holes (normally 5mm wide, although LEDs are also available in other sizes). You should also think carefully about where the batteries are going to be stored and where the wires are going to be connected.

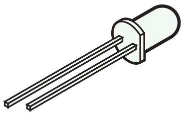
ELECTRONIC COMPONENTS

The main electronic components you may need for your cyberpet are shown here. The next few pages describe each of these components in more detail, and also provide some programming ideas that may be useful when you are later programming your cyberpet with it's own personality:

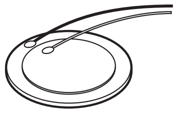
PICAXE-08 (or 08M) microcontroller



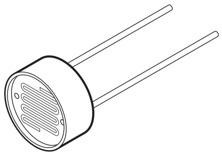
light emitting diode (LED)



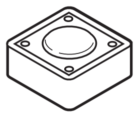
piezo sounder



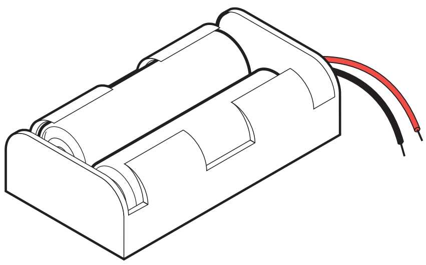
light dependent resistor (LDR)



switch

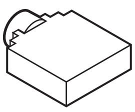


batteries

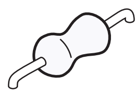


and you will also need

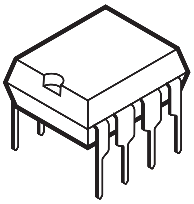
picaxe download socket



resistors



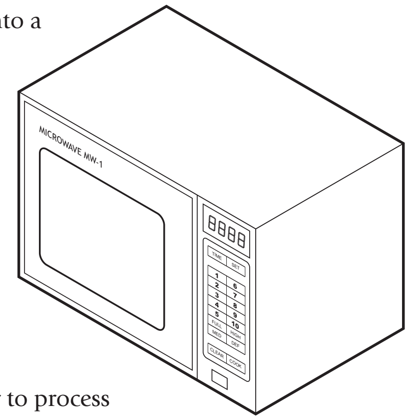
Section 2 - Electronic Components
Microcontrollers



What is a microcontroller?

A microcontroller is often described as a 'computer-on-a-chip'. It is an integrated circuit that contains memory, processing units, and input/output circuitry in a single unit.

Microcontrollers are purchased 'blank' and then programmed with a specific control program. Once programmed the microcontroller is built into a product to make the product more intelligent and easier to use.



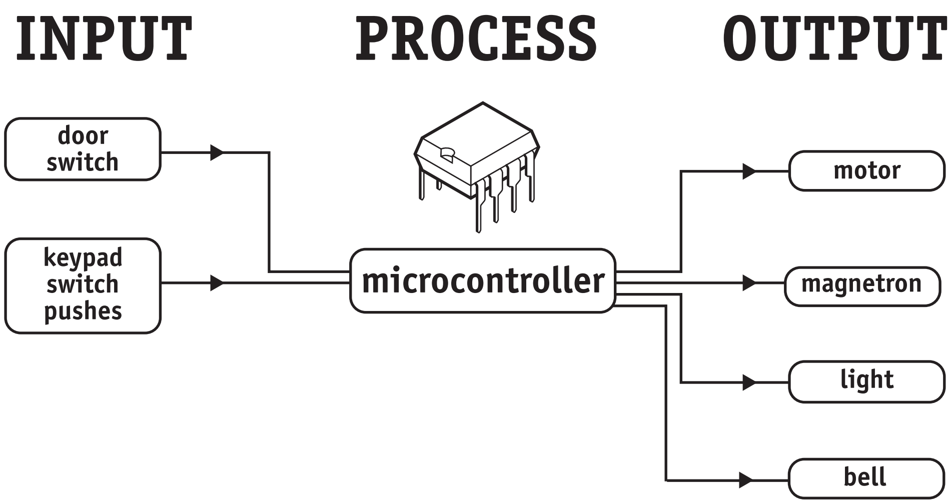
Where are microcontrollers used?

Applications that use microcontrollers include household appliances, alarm systems, medical equipment, vehicle subsystems, and electronic instrumentation. Some modern cars contain over thirty microcontrollers - used in a range of subsystems from engine management to remote locking!

As an example, a microwave oven may use a single microcontroller to process information from the keypad, display user information on the seven segment display, and control the output devices (turntable motor, light, bell and magnetron).

How are microcontrollers used?

Microcontrollers are used as the 'brain' in electronic circuits. These electronic circuits are often drawn visually as a 'block diagram'. For instance a simplified block diagram for the microwave above could be drawn like this:



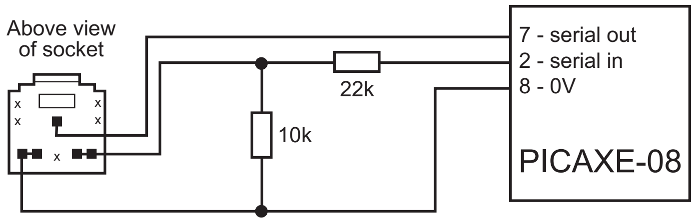
The program for the microcontroller is developed (and tested) on the computer and then downloaded into the microcontroller. Once the program is in the microcontroller it starts to 'run' and carries out the instructions.

How are programs written?

Programs are drawn as flowcharts or typed as 'BASIC' listings. This is is explained in the programming section (section 3) later in this booklet.

How is the program transferred to the microcontroller?

The PICAXE-08 microcontroller is programmed by connecting a cable from the serial port at the back of the computer to a socket on the printed circuit board (PCB) beside the microcontroller. This socket (which looks like a headphone socket as found on an MP3 player) connects to two legs of the microcontroller and to 0V from the battery. This allows the computer and the microcontroller to 'talk' to allow a new program to be downloaded into the microcontroller's memory.



The socket and interfacing circuit is included on every PCB designed to be used with the PICAXE-08 microcontroller. This enables the PICAXE microcontroller to be re-programmed without removing the chip from the PCB - simply connect the cable whenever you want to download a new program!

The circuit diagrams of PICAXE circuits often do not include the components above to make it easier to understand the input/output connections. However the two resistors and the socket are always built onto every PICAXE project board!

Output 0

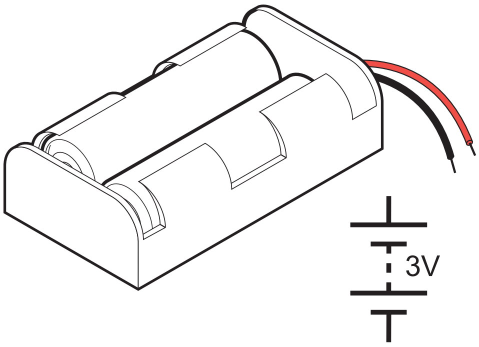
With the PICAXE-08 system leg 7 has two functions - when a program is being run the leg is known as output 0 and can control outputs like LEDs and motors.
When a program is being downloaded the same leg acts as the 'serial out' pin, 'talking' to the computer. Therefore if you also have an output such as an LED connected to the leg, you will find that the LED will flicker on and off as the program download takes place.

Note:
Most desktop computers have one or two serial ports, normally labelled COM1 and COM2 which can be used with the AXE026 serial cable. Laptop computers will use USB ports with the AXE027 USB cable, for which the COM port number will vary (e.g. COM5 or COM6). The Logicator software used to create the programs must be configured for the correct COM port – select **PIC>PIC Type** to select the correct serial port for your machine.

Batteries

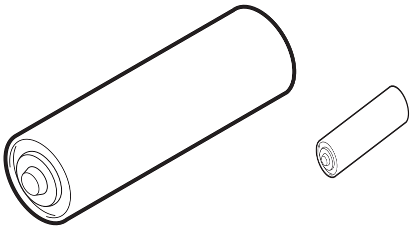
What is a battery?

A battery is a self-contained source of electronic energy. It is a portable power supply. Batteries contain chemicals that store energy. When connected into a circuit this chemical energy is converted to electrical energy that can then power the circuit.



Which battery size should I use?

Batteries come in all sorts of types and sizes. Most battery packs are made up of a number of 'cells', and each cell provides about 1.5V. Therefore 4 cells will generate a 6V battery and 3 cells a 4.5V battery.



As a general rule, the larger the battery the longer it will last (as it contains more chemicals and so will be able to convert more energy). A higher voltage battery does not last longer than a lower voltage battery Therefore a 4.5V battery pack made up of 3 AA cells will last much longer than a 9V PP3 battery, as it contains a larger total amount of chemical energy as it is physically larger. Therefore items that require more power to work (e.g. a CD walkman which contains a motor and laser to read the CD's) will always use AA cells rather than PP3 batteries.

Microcontrollers generally require 3 to 5V to work, and so it is better to use a battery pack made up of three AA size cells (4.5V). Never use a 9V PP3 battery as the 9V supply will damage the microcontroller.

Which battery type should I use?

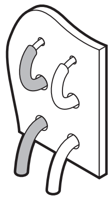
Different batteries are made of different chemicals. Zinc-carbon batteries are the cheapest, and are quite suitable for many microcontroller circuits. Alkaline batteries are more expensive, but will last much longer when driving devices like motors that require larger currents. Lithium batteries are much more expensive but have a long life, and so are commonly used in computer circuits to provide a clock backup.

Rechargeable batteries can be recharged when they 'run-down'. They are generally made up of nickel and cadmium (Ni-cad) or nickel metal hydroxide (NiMH) chemicals.

Safety!

Never 'short circuit' any battery. Alkaline and rechargeable batteries can provide a very large current, and can get so hot that they will actually melt the battery box if you short circuit them! Always make sure you connect the battery around the correct way (red positive (V+) and black negative (0V or ground)). The microcontroller chip will get hot and be damaged if the battery is connected the wrong way around.

Using battery snaps.



Battery packs are often connected to electronic printed circuit boards by battery snaps. Always ensure you get the red and black wires the correct way around. It is also useful to thread the battery snap through holes on the board before soldering it in place - this provides a much stronger joint that is less likely to snap off.

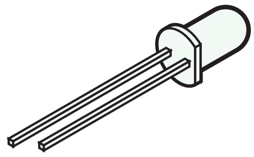
Never accidentally connect a 9V PP3 battery to the battery snap - this will damage the microcontroller, which only works between 3 and 5V. 4.5V is recommended.

Soldering to battery boxes.

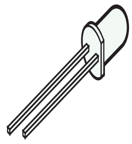
Some small battery boxes require wires to be soldered to metal contacts on the battery box. In this case you must be very careful not to overheat the metal contact. If the contacts gets very hot they will melt the plastic and fall off. A good way of stopping this happening is to ask a friend to hold the metal contact with a pair of small pliers. The pliers will act as a 'heat-sink' and help stop the plastic melting.

Light Emitting Diode (LED)

What is an LED?



A Light Emitting Diode (LED) is an electronic component that gives out light when current passes through it. An LED is a special type of diode. A diode is a component that only allows current to flow in one direction. Therefore when using a diode, it must always be connected the correct way around.



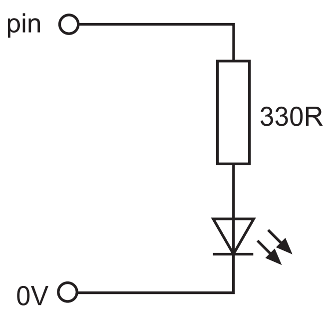
The positive (anode) leg of an LED is longer than the negative (cathode) leg (shown by the bar on the symbol). The negative leg also has a flat edge on the plastic casing of the LED.

What are LEDs used for?

LEDs are mainly used as indicator lights. Red and green LEDs are commonly used on electronic appliances like televisions to show if they are switched on or in 'standby' mode. LEDs are available in many different colours, including red, yellow, green and blue. Special 'ultrabright' LEDs are used in safety warning devices such as the 'flashing lights' used on bicycles. Infra-red LEDs produce infra-red light that cannot be seen by the human eye but can be used in devices such as video remote-controls.

Using LEDs.

LEDs only require a small amount of current to work, which makes them much more efficient than bulbs (this means, for instance, that if powered by batteries the LEDs will light for a much longer time than a bulb would). If too much current is passed through an LED it will be damaged, and so LEDs are normally used together with a 'series' resistor that protects the LED from too much current.



The value of the resistor required depends on the battery voltage used. For a 4.5V battery pack a 330R resistor can be used, and for a 3V battery pack a 120R resistor is appropriate.

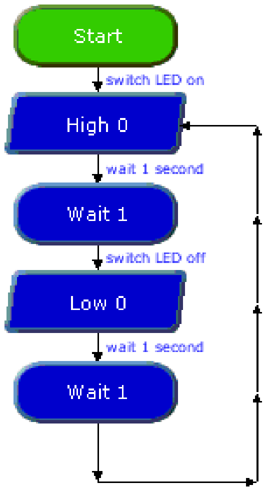
Connecting the LED to a microcontroller.

Because the LED only requires a small amount of current to operate, it can be directly connected between the microcontroller output pin and 0V (with the series protection resistor).

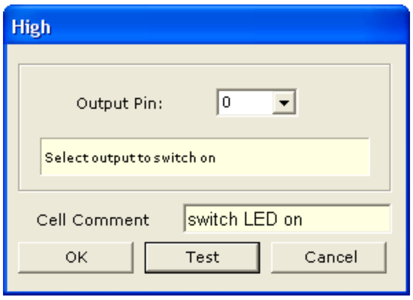
Testing the LED connection.

After connecting the LED it can be tested by a simple flowchart like this.
You will need to use the following flowchart cells:

high
wait
low



When a cell has been dragged onto screen you double click on it to change the output numbers, cell comments etc.



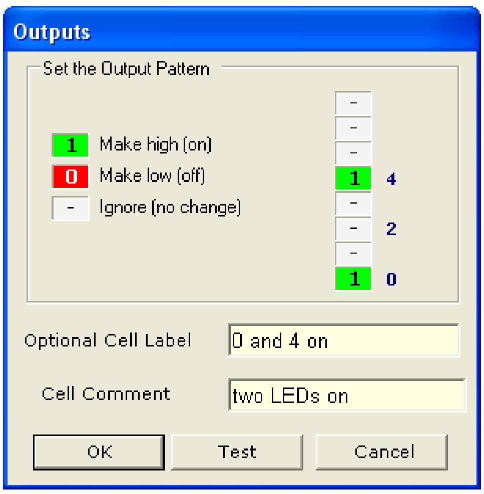
This flowchart would switch the LED on and off every second. If the LED does not work check:

1) the LED is connected the correct way around
2) the correct resistor is used
3) the correct output pin number is being used
4) all the solder joints are good

To switch more than one output on or off at the same time an 'Outputs' command is used instead of high / low commands.

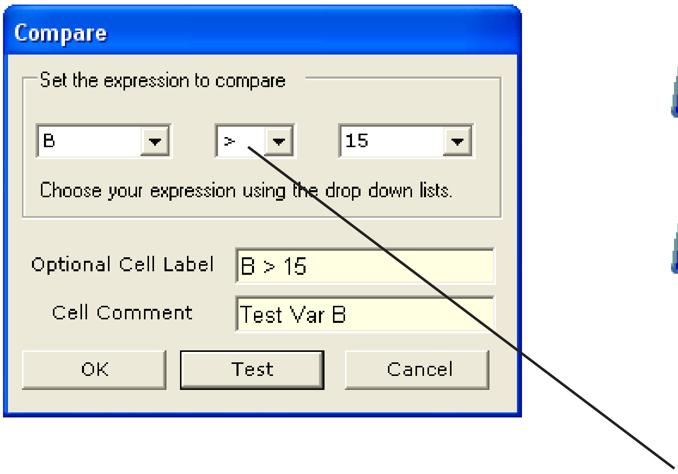
When using an Outputs command each output can be

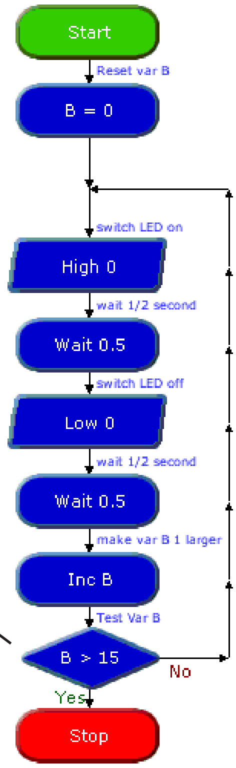
switched on (high)	(set that output number to a green 1)
switched off (low)	(set that output number to a red 0)
ignored (not changed)	(set that output to -)



This more complicated flowchart flashes the LED connected to output pin 0 on and off 15 times by using a variable (B) to count the number of loops that have been carried out. Expression, Compare and INC commands are in the 'Variables' section of the toolbox.

B = 0	is set using an 'Expression' cell
B > 15	is set using a 'Compare' cell
INC B	means 'make value of B one more' (increment)

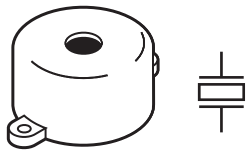




The number of times the code has been repeated is stored in the memory of the PICAXE chip using a 'variable' called B (the PICAXE contains 8 variables labelled A to H). A variable is a 'number storage position' inside the microcontroller than the microcontroller can use to store numbers as the program is carried out.

Buzzers and Piezo-Transducers

What is a piezo transducer?



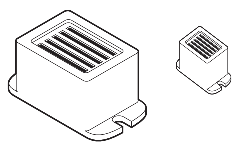
A piezo transducer is a low-cost 'mini-speaker' that can used to make sounds. The sound that the piezo makes can be changed by altering the electronic signals provided by the microcontroller.

Where are piezos used for?

Piezos are used in many different consumer goods to provide 'feedback' to the user. A good example is a vending machine which will 'beep' whenever a keypad switch is pressed to select a drink or snack. The 'beep' provides the user with feedback to tell them their switch push has been successful. Uncased piezos are also often used in musical birthday cards to play a tune when the card is opened.

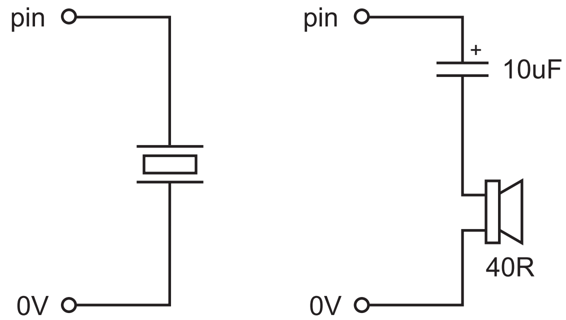
What is the difference between a piezo and a buzzer?

A buzzer contains a small electronic circuit that generates the electronic signal needed to make a noise. Therefore when a buzzer is connected to a battery it will always make the same sound. A piezo does not contain this circuit, and so therefore needs an external signal. This signal can be supplied by the output pin of a microcontroller. A piezo also requires less current to operate and so will last longer in battery powered circuits.

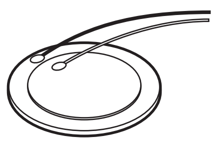


Using piezos.

A piezo is very simple to connect. Simply connect the red wire to the microcontroller output pin and the black wire to 0V (ground).

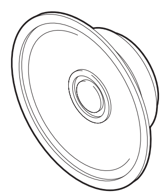


Note that the cheapest piezos do not have a plastic casing to them. In this case it is necessary to mount the piezo on a piece of board (with a sticky pad) to create a noise that can be heard. The board acts as a 'sound-box' to amplify the sound made by the piezo. Make sure the sticky pad is stuck on the correct side of the piezo (the brass side without the wires!).

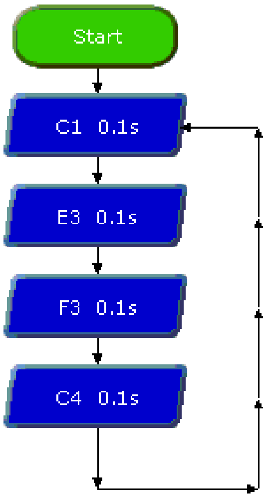


Making More Noise.

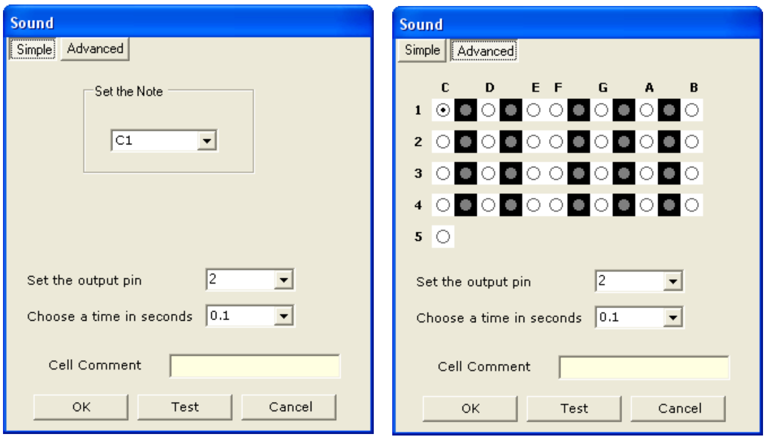
Some times you might want to make a louder noise. In this case it is possible to use a speaker instead of the piezo. When using a speaker it is also necessary to use a capacitor (e.g. 10uF electrolytic capacitor) to prevent damage to the microcontroller.
Remember that, like the piezo, a speaker only works correctly when mounted in a 'sound-box'.



Testing the piezo connection (sound command).

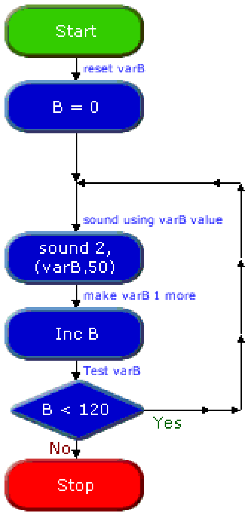


After connecting the piezo it can be tested by a simple flowchart like this, which uses the 'Sound' command.



This flowchart would make the piezo (connected to output pin 2) make 4 different sounds. If the piezo does not work check:

1) the correct output pin number is being used in the flowchart
2) all the solder joints are good

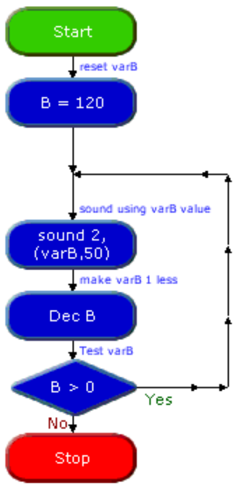


The following Logicator flowchart uses a loop to produce 120 different sounds, using variable B to store the sound value. In this case the 'sound' command is entered manually using the BASIC command cell. The text to enter is

sound 2, (varB, 50)

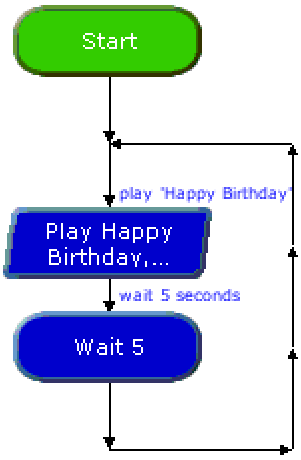
The number stored in variable B is increased by 1 in every loop (1-2-3 etc.) Therefore by using the variable name B in the tone position, the note can be changed on each loop.

The following program does the same task but backwards (counting down instead of up).



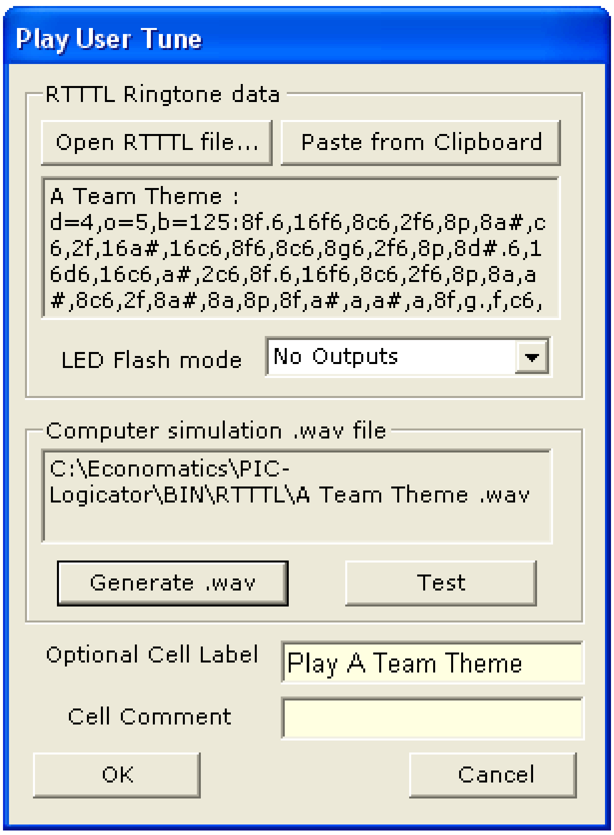
Testing the piezo connection (play tune commands).

If using a PICAXE-08M chip you can also play musical tunes on the piezo.

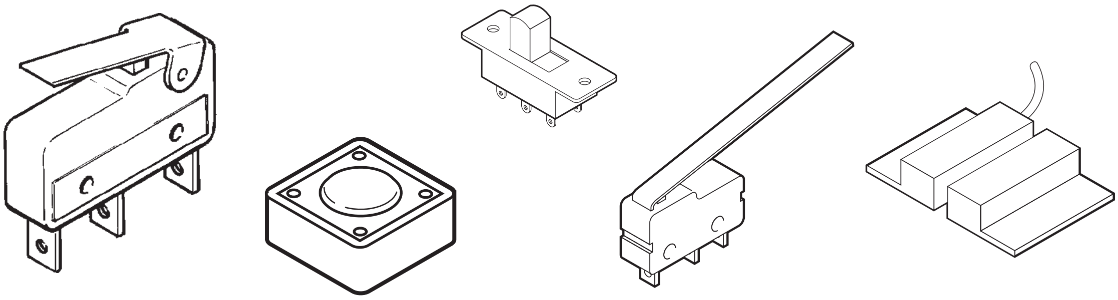


The ‘Play Tune’ command allows 4 pre-programmed tunes such as Happy Birthday to be played. You can also choose to flash outputs 0 and 4 at the same time if you would like.

The ‘Play User Tune’ command allows mobile phone ringtones in Nokia RTTTL format to be imported and played. This enables you to choose your own tune.

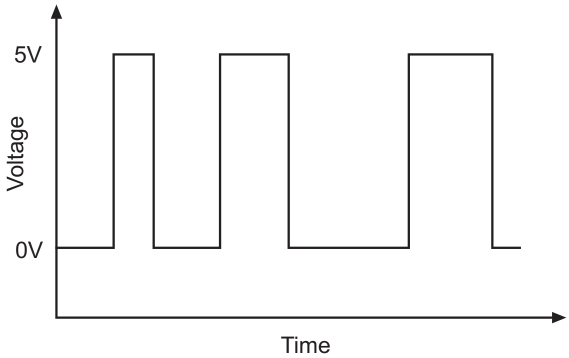


Digital Sensors (Switches)



What are switches?

A digital sensor is a simple ‘switch’ type sensor that can only be ‘on’ or ‘off’. If a graph is drawn of the on-off signals as the switch is pushed it will look like this:



Switches are electronic components that detect movement. There are a large number of different types of switches e.g:

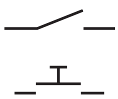
push switches that detect a momentary 'push'
micro-switches with long levers that detect small movements
tilt-switches that detect jolting
reed-switches that detect a magnet being moved

What are switches used for?

Push switches are commonly used on device like keypads. Micro-switches are used in burglar alarms to detect if the cover is removed from the alarm box. Reed switches are used to detect doors and windows being opened and tilt switches are often used to detect movement in devices such as toys, hair-dryers and tool-box alarms.

Switch Symbols.

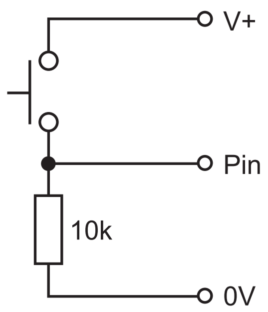
The symbols for a slide switch and a push switch are shown here.



Using switches

A switch is used with a resistor as shown in the diagram. The value of the resistor is not that important, but a 10k resistor is often used. When the switch is 'open' the 10k resistor connects the microcontroller input pin down to 0V, which gives a low 'off' (logic level 0) 0V signal to the microcontroller input pin.

When the switch is activated, the input pin is connected to the positive battery supply (V+). This provides a high 'on' (logic level 1) signal to the microcontroller.

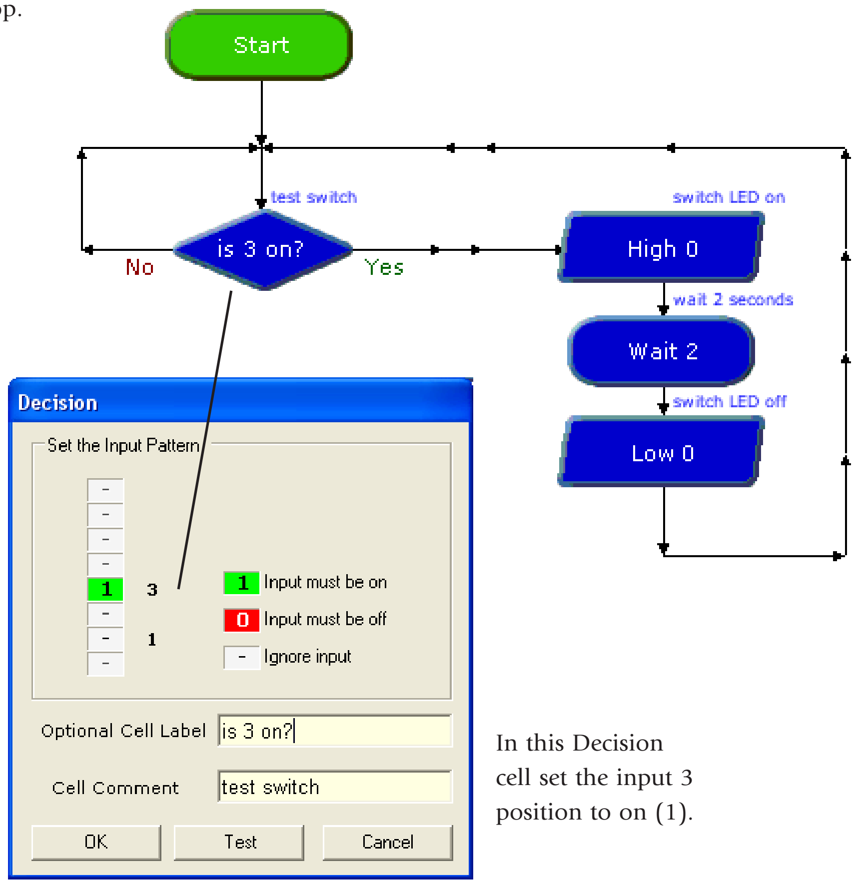


Testing the switch

After connecting the switch it can be tested by a simple flowchart which uses a 'Decision' command. This flowchart will switch an output on and off according to if the switch is pushed or not. Before you test the switch, make sure that you have tested output 0. you can do this by using the Logicator for PIC micros flowchart on page 10.

In this program the 'no route' makes up a continuous loop. If the input is off the flowchart just loops around time and time again.

If the switch is then pushed the program jumps to the second section of the flowchart. The flowchart then flashes output 0 on for two seconds before returning to the main loop.

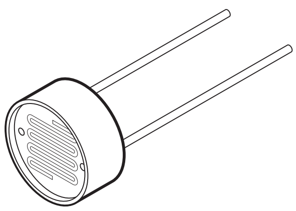


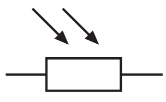
In this Decision cell set the input 3 position to on (1).

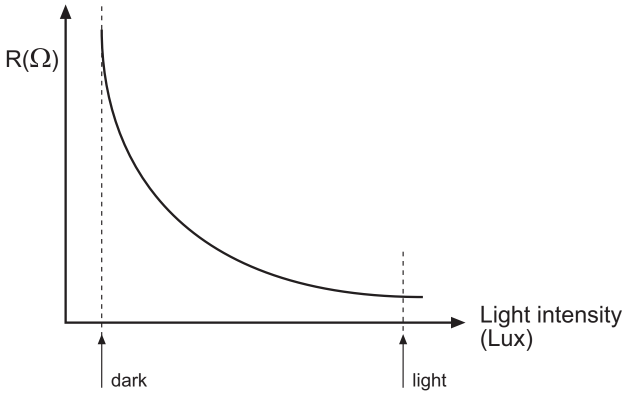
Light Dependent Resistor (LDR)

What is an LDR?

A Light Dependent Resistor (LDR) is special type of resistor that reacts to changes in light level. The resistance of the LDR changes as different amounts of light fall on the top 'window' of the device. This allows electronic circuits to measure changes in light level.



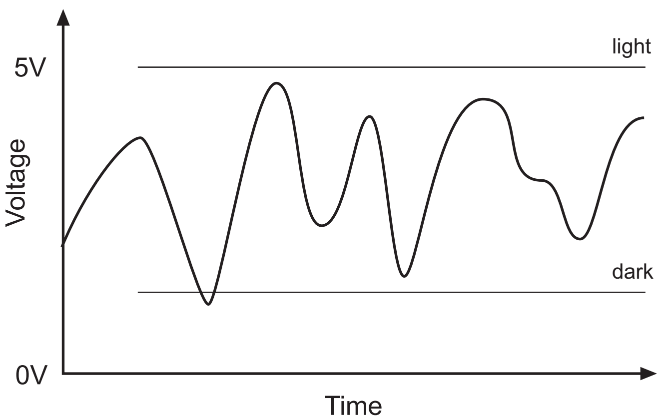




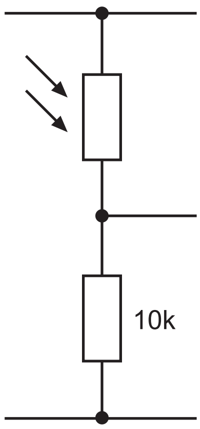
What are LDRs used for?

LDRs are used in automatic street lamps to switch them on at night and off during the day. They are also used within many alarm and toys to measure light levels.

The LDR is a type of **analogue sensor**. An analogue sensor measures a continuous signal such as light, temperature or position (rather than a digital on-off signal like a switch). The analogue sensor provides a varying voltage signal. This voltage signal can be represented by a number in the range 0 and 255 (e.g. very dark = 0, bright light = 225.)



Using LDRs.



A LDR can be used in two ways. The simplest way to use an LDR is as a simple on-off ("digital") switch - when the light level is above a certain value (called the 'threshold value') the LDR will provide an on signal, when the light level is below a certain value the LDR will provide an off signal.

In this case the LDR is used in a potential divider with a standard resistor. The value of the standard resistor sets the 'threshold value'. For most LDRs a suitable value is 10k.. If desired the fixed resistor can be replaced by a variable resistor so that the threshold value can be 'tuned' to different light values.

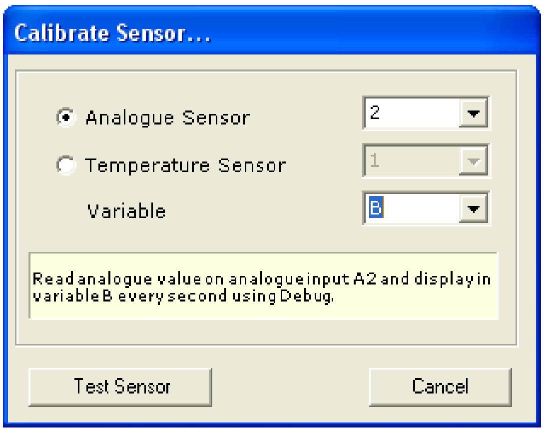
A more versatile way of using the LDR is to measure a number of different light values, so that decisions can be made at varying light levels rather than just one fixed threshold value. A varying value is known as an 'analogue' value, rather than a digital 'on-off' value. To measure analogue values the microcontroller must contain an 'analogue to digital converter (ADC)' and the programming software must support use of this ADC. Most microcontrollers only contain ADC on certain input pins, and so the input pin connection must be carefully selected. With the 8 pin microcontroller only pin1 can be used.

The electronic circuit for using the ADC is a potential divider identical to the circuit above. The analogue 'measurement' is carried out within the microcontroller itself.

Calibrating the LDR (used as analogue input)

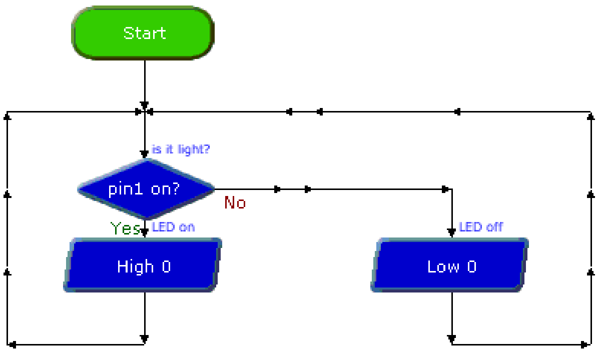
When using analogue sensors it is often necessary to calculate the 'threshold' value necessary for the flowchart (ie the values 50 and 100 in the analogue flowchart overleaf). The debug command provides an easy way to see the 'real-time' value of a sensor, so that the threshold value can be calculated by experimentation.

The PIC > Analogue Calibration Wizard menu allows you to download a debug program straight into the PICAXE chip. Once running the light level value will be displayed on screen - updated every second. As the Light falling on the LDR sensor is altered, the variable value will show the current sensor reading.



Testing the LDR (using as a digital on/off switch)

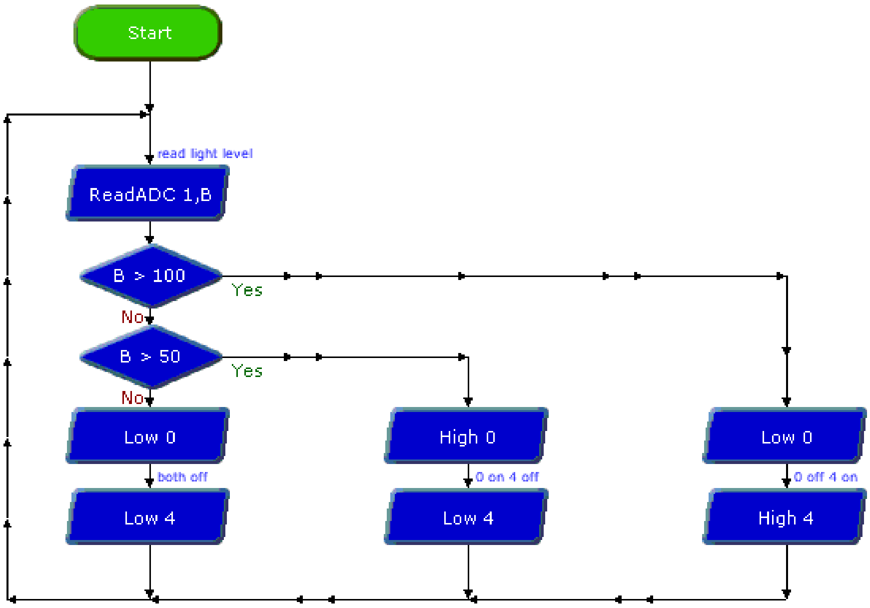
After connecting the LDR it can be tested as a digital switch by a simple flowchart using a decision command like this:



This flowchart will switch output 0 on and off according to the light level.

Testing the LDR (using as an analogue light sensor)

After connecting the LDR it can be tested as an analogue sensor by a flowchart using a readadc cell and a compare cell like this:



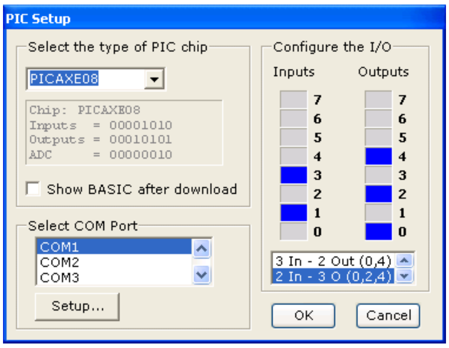
The 'readadc' command is used to read the analogue value (a number between 0 and 255) into variable B. Once the number is in variable B it can be tested to see if it is greater than 100 or greater than 50. If it is greater than 100 output 4 is switched on, if it is between 50 and 100 output 0 is switched on, and if it is less than 50 both outputs are switched off.

SECTION 3
PROGRAMMING - DRAWING FLOWCHARTS

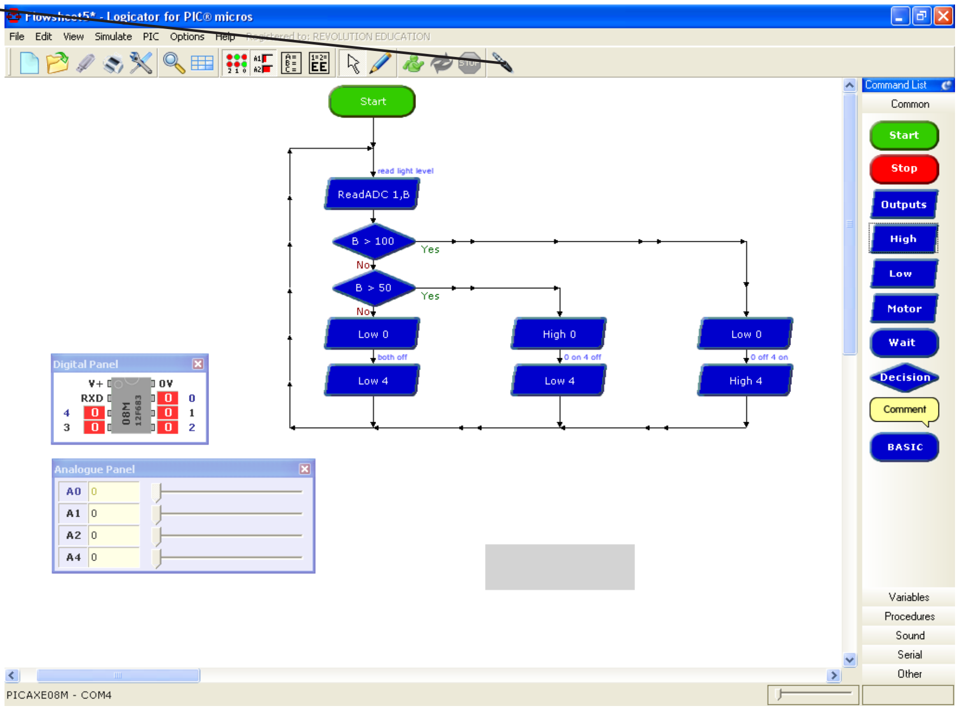
Flowcharts are a useful tool that allow programs to be drawn graphically to make them easier to understand. Logicator allows flowcharts to be drawn on screen. These flowcharts can then be downloaded or converted to PICAXE BASIC listings. The flowcharts can also be printed or exported as graphics files for inclusion within project portfolios.

Detailed instructions for drawing/downloading a flowchart:

1. Connect the PICAXE cable to the computer serial or USB port. Note which port it is connected to.
2. Start Logicator for PIC micros.
3. Select “PIC > Select PIC type”.
4. Click on the chip type tab and select PICAXE-08/08M. Select the correct i/o pin arrangement for this project, which is 2 in (in 1,3) and 3 out (out 0,2,4).
5. Click on the ‘COM Port’ tab and select the serial port that the PICAXE cable is connected to. Click ‘OK’ (for AXE027 USB cables the ‘Setup’ button can be used to discover the correct COM port).
6. Start a new flowchart by clicking File>New.
7. Draw the flowchart by dragging the correct symbols onto the screen, and then using the mouse to draw arrows between the symbols. You can also use the “Alt” key and the arrow keys to draw the lines.
8. Once the flowchart is complete it can be downloaded into the PICAXE by clicking the ‘jack plug’ (Program) button. The flowchart can also be converted into a BASIC program by selecting PIC >Convert Flowsheet to BASIC.
9. To print or save the flowchart, use the File menu options. To export the flowchart as a graphic file, use the File>Export menu.



Jack Plug button (Program PIC button)

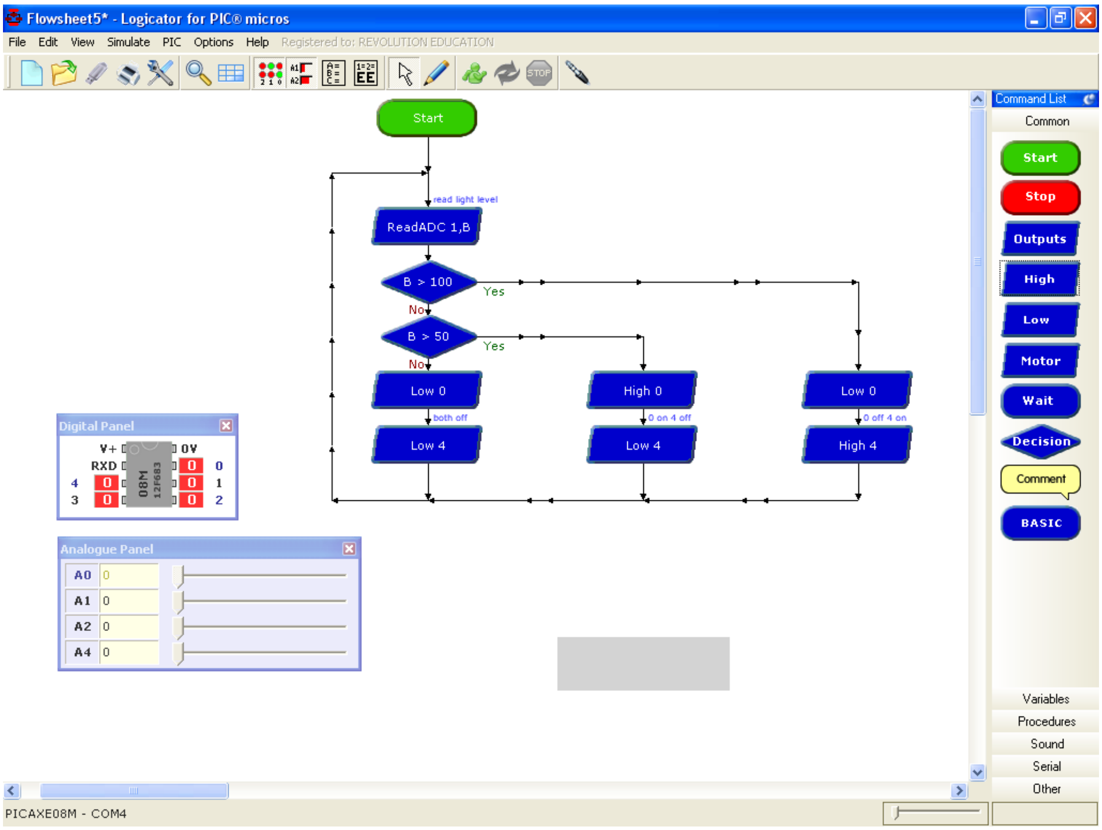


Simulation Methods

There are two simulation methods for the Cyberpet project - either on-screen simulation of the flowchart, or via a Soft System virtual PCB.

Method 1 - On Screen Simulation

To simulate the flowchart, click the green running man (run) button. The flowchart will then start to run on-screen.



As the flowchart runs, each cell is circled in blue as it is carried out. The 'Inputs/ Outputs' and 'Variables' windows also appear when a simulation is being carried out. To adjust the input values click the on-screen input LED or slide the analogue input slider.

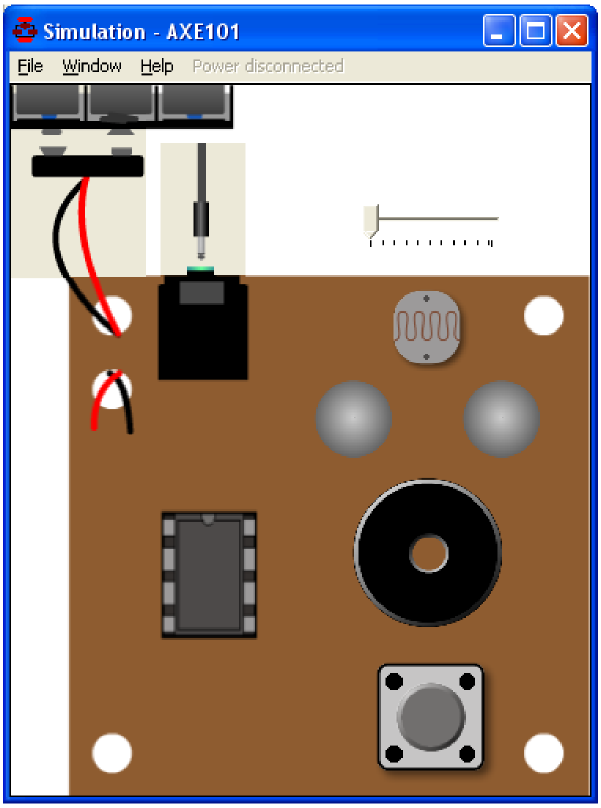
The time delay between shapes can be adjusted via the simulation speed slider in the status bar at the bottom of the screen.

Note that certain commands have no on-screen simulation equivalent feature. In this case the command is simply ignored as the flowchart runs.

Method 2- Virtual PCB Simulation

The Logicator software includes a simulation of the Cyberpet PCB. To open the simulation select this menu:

Simulate > Open Simulation > PICAXE Kits > AXE101 Cyberpet



The cyberpet simulation will then open on screen. Just like a real life PCB you must connect the battery pack and download cable (by clicking on them) and then click the 'Program' jack plug icon above the flowchart. The PICAXE program will then be downloaded into the 'virtual PCB' and then run as on a real chip.

Click on the switch to activate it, or move the anlogue slider to change the light level on the LDR.

Note that the on-screen flowchart simulation and the 'virtual PCB' simulation are completely separate - you do not see the flowchart cells highlighting as the virtual PCB simulation takes place.

Downloading Flowcharts

Flowcharts can be directly downloaded to the microcontroller. Alternately the flowchart can be converted into a BASIC program which is then downloaded.

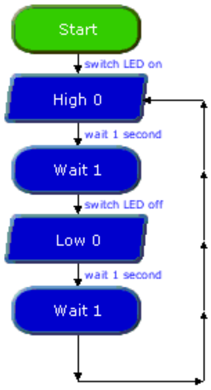
To convert a flowchart select PIC > Convert flowsheet to BASIC. The BASIC program for downloading will then be created.

Shapes that are not connected to the 'start' or 'procedure' shapes in the flowchart are ignored when the conversion takes place. The conversion will stop if an unconnected shape is found. Therefore always use a 'stop' shape or line to complete the flowchart before simulation or conversion.

Note that it is possible to quickly convert and then download a flowchart by pressing Alt and the shortcut key <F5>.

Using descriptive labels

Most command cells have a 'cell comment' box. This comment is placed in small text above the flowchart cell. This is to help you remember the purpose of each cell, so always add a useful text comment.



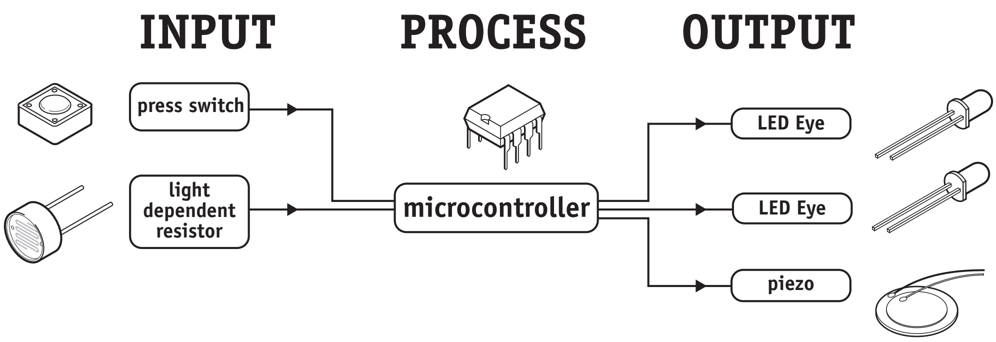
Many of the command cells (eg. Inputs, Outputs and Variable)s can all be named by double clicking on them and entering descriptive text. This text appears within the actual command itself, and is also used to remind you what a specific cell is doing. eg. rather than a cell called "Outputs", you might call it "switch on LED 1".

Saving and Printing Flowcharts

Flowcharts can be saved, printed and exported as graphic files (for adding to word processor documents) via the File menu.

Section 4 - The CyberPet PCB

The CyberPet project uses a PICAXE-08 microcontroller with two LEDs as the pets 'eyes' and a piezo sounder as a 'voice' for the pet.
The project also uses a switch so that the pet can respond to 'touch', and a Light Dependent Resistor (LDR) so that the pet can tell whether it is light or dark.



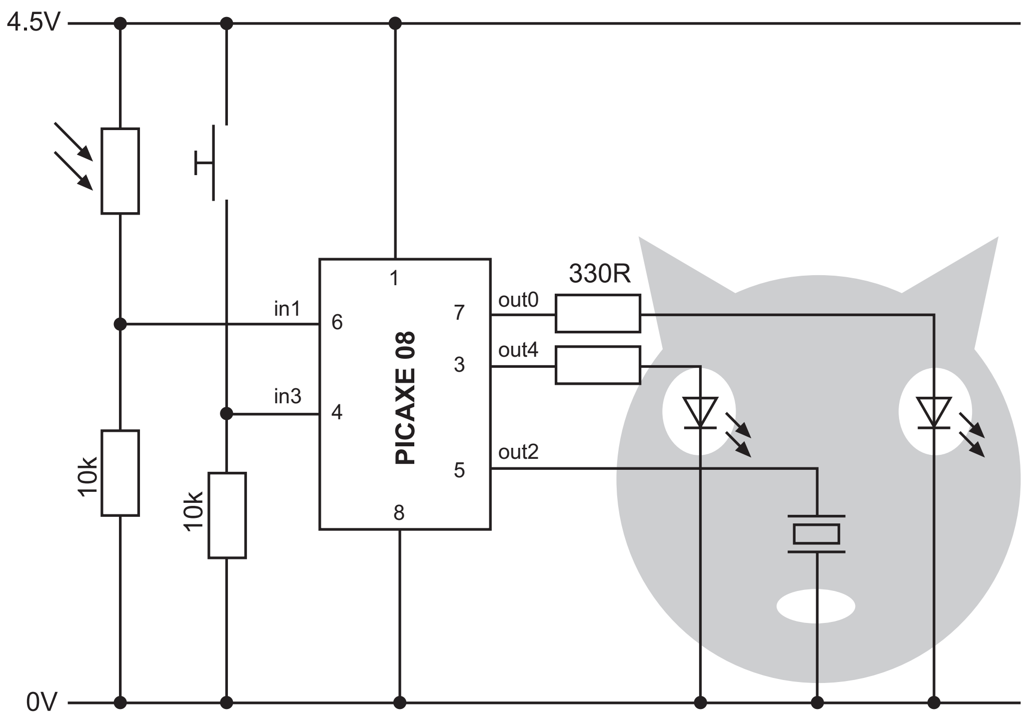
The electronic block diagram is shown below.

output -	pin0 and pin4 are connected to the LEDs
output -	pin2 is connected to the piezo sounder
input -	pin1 is connected to the LDR
input -	pin3 is connected to the push switch

Remember not to confuse the chip 'leg' number with the input/output pin number!

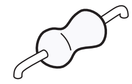
Circuit Diagram

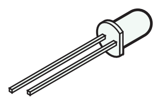
The circuit diagram for the CyberPet project is shown below:



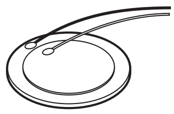
BUILDING THE CYBERPET PCB

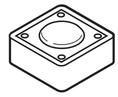
What you will need:



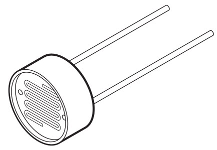


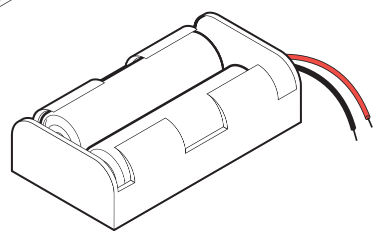
R1 and R2	10k resistor	(brown black orange gold)
R3	22k resistor	(red red orange gold)
R4 and R5	330R resistor	(orange orange brown gold)
R6	10k resistor	(brown black orange gold)





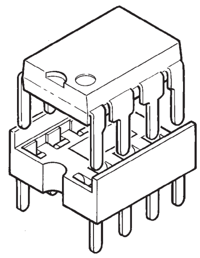
LED1 and 2	5mm red LED
PZ	piezo sounder
LDR	miniature light dependent resistor
SW1	miniature 6mm switch
IC1	8 pin IC socket
IC1	PICAXE-08 or 08M microcontroller
CT1	PICAXE download 3.5mm socket
BT1	battery clip
BT1	4.5V (3xAA) battery box
PCB	printed circuit board



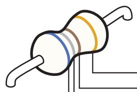


wire (if connecting LEDs and LDR by wires)

Tools:
soldering iron and solder
side cutters



Resistor colour codes



Black	0	0	Black x1	Silver ±10%
Brown	1	1	Brown x10	Gold ±5%
Red	2	2	Red x100	
Orange	3	3	Orange x1000	
Yellow	4	4	Yellow x10,000	
Green	5	5	Green x100,000	
Blue	6	6	Blue x1,000,000	
Violet	7	7		
Grey	8	8		
White	9	9		

Example shown:
blue, grey, brown, gold
= 680R ±5%

Soldering the PCB.

The printed circuit board (PCB) is specially manufactured with a 'solder resist' layer to make it simpler to solder. This is the green 'lacquer' layer that covers the tracks so that the solder does not stick to these tracks. However for successful assembly the PCB must be carefully assembled and soldered.

When soldering always make sure the solder iron tip is hot and clean. To test if it is hot enough try to melt a piece of solder on the tip. The solder should melt almost instantly. Then clean off the melted solder by wiping the tip on a damp sponge.

Remember that solder will only 'stick' to hot surfaces. Therefore never melt the solder on the soldering iron tip and then try to 'drop' it onto the joint – this won't work as the joint will be cold and so the solder won't stick.

To successfully solder you must hold the soldering iron in one hand and the solder in the other. Therefore make sure the board is held on the table so it won't move (e.g. use a bulldog clip or get someone else to hold it for you).

Steps to soldering:

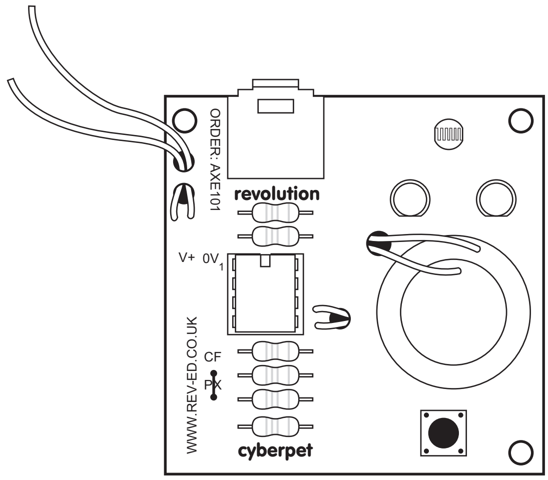
1) Clean the soldering iron tip on the damp sponge
2) Press the soldering iron tip against the pad on the PCB AND the leg of the component. Count to 3 to give the joint time to warm up.
3) Keep the soldering iron in position and touch the solder against the joint. Allow enough solder to melt to cover the joint.
4) Take the solder away first, then the soldering iron
5) Allow the solder to cool for about 5 seconds before trying to move the board.

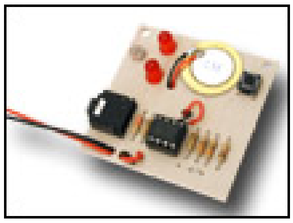
After each joint is made make sure it does not accidentally 'bridge' across to other joints. However be aware that some solder joints (e.g. on the two sides of the PICAXE download socket) have two wires very close together that are already connected by a track (line) on the PCB. In this case it does not matter if the solder joins together.

Tips!

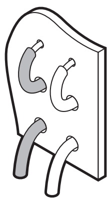
1) Always start with the smallest components like the resistors. Then move onto larger components like the IC socket and then finish with the tall components like capacitors and transistors. Do not try to put all the components in position at once, only do two or three at a time.
2) Always make sure that the components lie flat on the board before they are soldered. When using components with long legs like resistors and LEDs, bend the legs so that the component is held firmly in position before soldering.
3) Make sure the PICAXE stereo download socket 'snaps' into position flat on the board before it is soldered.
4) Make sure that the components that only work one way around (LEDs, diodes, transistors and capacitors) are correctly aligned before soldering (see the marks on the PCB).
5) Piezo sounder wires are very thin. Make sure you do not overheat them or they may melt.
6) Always thread the battery snap wires down and up through the two thread holes before soldering. This helps make a much stronger joint which is less likely to snap off.

How you build your Cyberpet PCB will depend on the shape and size of your design. You may solder all the components directly onto the board, or you may connect some of the components (e.g. the LEDs, the LDR and the switch) by longer pieces of wire so they can be fitted inside your Cyberpet. These instructions presume you are soldering all the components directly on the board. The instructions are identical if you are using longer wires to join some components, although you must be even more careful you get the wires around the correct way on the LEDs.





Do not forget the wire link!



1) Place the three 10k (brown black orange gold) resistors in position. Bend the legs to hold the resistors in position and then solder.
2) Place the 22k (red red orange gold) and two 330 (orange orange brown gold) resistors in position. Bend the legs to hold the resistors in position and then solder.
3) Using an off cut resistor leg, make a wire loop over the letters PX marked beside the 330R resistors. Solder in position. (Ignore the hole above the holes marked CF).
4) Push the PICAXE stereo download socket onto the PCB and make sure it clicks into position (so that it lies flat on the board). Solder the five metal square contacts (the five round plastic support post holes do not have to be soldered). Do not worry if the solder joins on the two metal contacts either side of the socket as they are supposed to be joined anyway.
5) Push the IC socket into position. Make sure the notch at one end points up towards the socket. Fold the legs over to hold the socket in position and then solder.
6) Solder the LDR and two LEDs into position. Make sure the flat on one side of the LED aligns with the flat marked on the PCB.
7) Solder the switch in position (note that it only fits one way around). If using wires solder one wire into either one of the two bottom holes and the other wire into either one of the two top holes.
8) Thread the battery clip down through the large hole by the letters AXE. Thread it back up through the large hole by the letters 101. Then solder the black wire in the hole marked 0V and the red wire in the hole marked V+.
9) Use 1/2 a sticky pad to stick the piezo sounder (brass side) to the top of the PCB. Thread the wires down through the hole below LED1 and back up through the hole marked PZ. Solder the red wire into the bottom hole and the black wire into the top hole. It does not matter if the red wire solder joint joins pin5 of the IC socket as they are supposed to join anyway. However the black wire should NOT join pin 6 of the IC socket.
10) Carefully check the board to make sure there are no missed joints or accidental solder bridges.
11) Insert the microcontroller into the socket, ensuring pin1 faces the stereo socket.

Testing your circuit.

Step 1 – Check the solder joints.

Check that all the joints are connected to both the pad and the wire, and that the wire is held firmly so that it does not 'wobble' when pulled. Also check that the solder does not accidentally bridge between two pads. This is most likely to happen on the LEDs, the LDR and on the piezo. On the stereo socket the two square pads close together on each side can be joined as they are already joined by a track on the board. However they must not be joined to the central round hole.

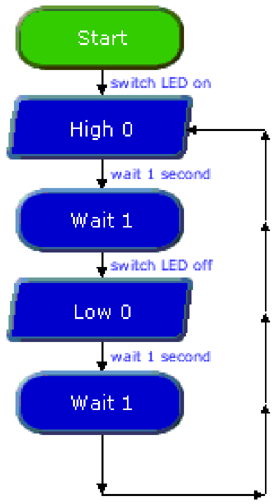
Step 2 - Check the components.

1) Check that the black battery clip wire is in the hole marked '0V' and the red battery clip wire is in the hole marked 'V+'
2) Check that the PICAXE-08 chip is in the socket correctly, with the dent (showing pin1) closest to the stereo socket.
3) Check that the flat edge of the LED is connected to the correct hole on the PCB.
4) Make sure you have not forgotten the wire link over the holes marked PX at the bottom left of the board.
5) Make sure the brass side of the piezo is stuck down with a sticky pad.
6) Check that the socket is correctly soldered, including the middle square pad which is often forgotten by mistake.

Step 3 - Connect the battery.

Check the 3 AA batteries are in the battery box correctly. Connect the battery box to the battery snap and put your finger on the PICAXE chip. If it starts to get hot remove the battery box immediately as there is a problem – most likely that the chip or the battery snap wires are around the wrong way.

Step 4 – Download a program to test LED 0.



Connect the cable to the back of the computer and to the PICAXE socket on the PCB. Make sure the cable is pushed fully into the socket on the PCB.

Make sure the software is in the PICAXE-08 mode and the correct COM port is selected (see section 3 of this booklet for more information).

Draw a flowchart like this and then download it to the PICAXE chip. Use high / low / wait commands.

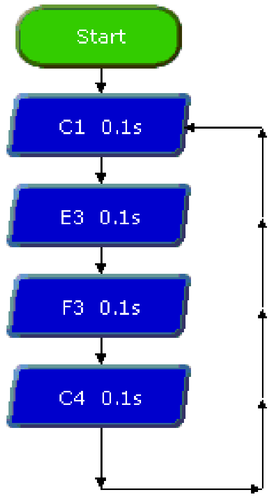
The LED should flicker as the program downloads. After the download is complete the LED should flash on and off every second. If the LED does not flash check that it is around the correct way and that the 330R resistors are in the correct positions on the PCB.

If the program does not download check that the 22k, 10k, socket and IC socket are all soldered correctly. Use a multimeter to make sure you are getting 4.5V across the top legs (1 and 8) of the microcontroller. Check that the cable is pushed firmly into the socket and that the correct serial port is selected within the software.

Step 5 – Test the other LED

Repeat the flowchart in step 4, but use high 4 and low 4 instead of 0. This will test the other LED.

Step 6 – Test the piezo

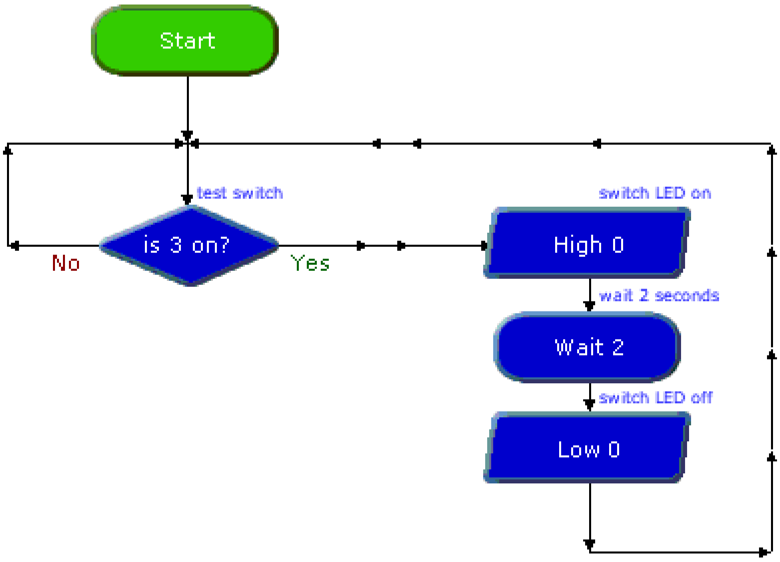


Draw and download the following flowchart using the sound command.

The piezo should make 4 different noises. If it does not make sure the wires are correctly soldered, that it is stuck on the brass side with a sticky pad (it will not work if 'hanging loose') and that the wire link over the letters PX is on the board.

Step 7 – Test the switch

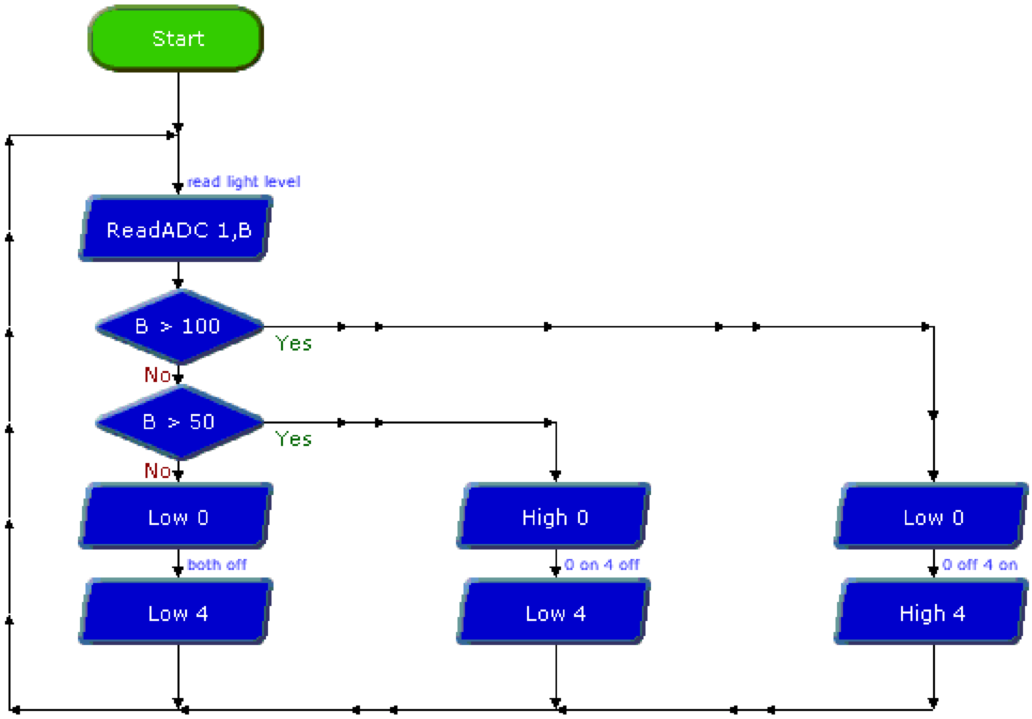
Draw and download the following flowchart, using decision, high, low and wait.



The LED on output 0 should light whenever the switch is pushed. If it does not check that the switch and 10k resistors are correctly soldered.

Step 8 – Test the LDR

Draw and download the following flowchart.



NOTE You may need to change the threshold values if it is dark in the room. e.g. try 60 and 30 instead of 100 and 50

The two LEDs should light in different patterns as you raise and lower your hand over the LDR (so that different amounts of light fall on the LDR). If they do not check that the LDR and 1k/10K resistor are correctly soldered.

If all these tests pass, you can be congratulated as you have correctly built and assembled your Cyberpet! It is now time to develop and test your own program to give it a personality!

Section 5 - Program Ideas.

Now that you have assembled and tested your Cyberpet, it is time to give it a 'personality' by developing your own program. This program can make the pet react in different ways to the light and push switch sensors.

Included on the next page is one more complex example program. It is designed to give you a starting point for your program. You may choose to modify it or to start a completely new program if you prefer.

Be creative!

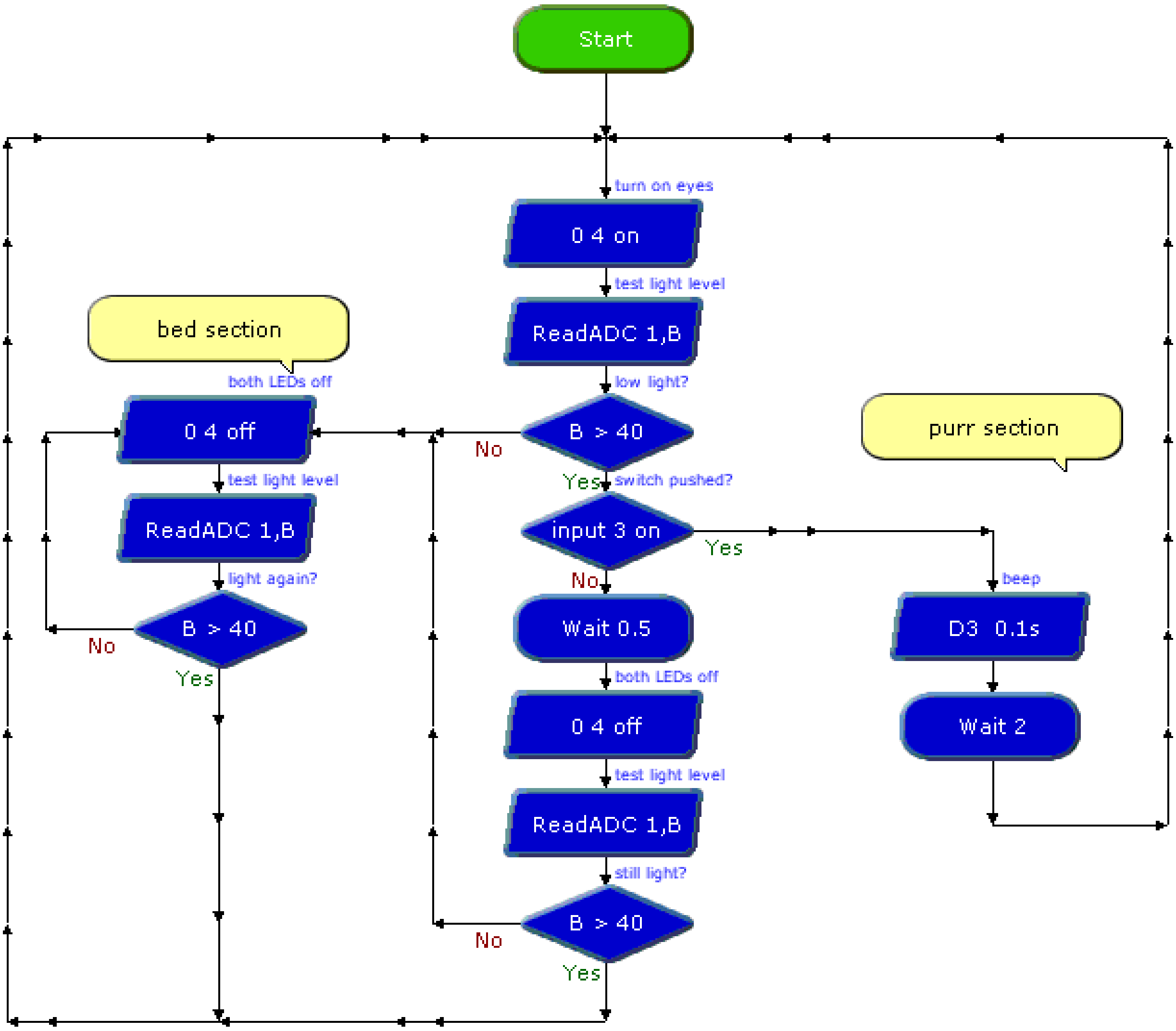
Your cyberpet is your creation so give it an exciting personality!

Flowchart 1 Explanation

This flowchart has a main loop which flashes the LED eyes on and off, and also checks the light sensor and the push switch. When the push switch is pressed a sound is generated on the piezo sounder. Note that you have to hold the switch down until the 'sound' is heard - you cannot just briefly touch the switch for it to work.

If the LDR light sensor is covered the pet will 'go to sleep' until the light level rises again.

Example Flowchart



ACKNOWLEDGEMENT

This project development was funded by the UK Offshore Oil and Gas Industry.
www.oilandgas.org.uk/education/