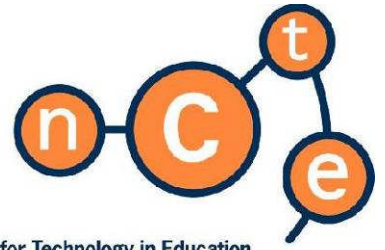




TECHNOLOGY
SUBJECTS
SUPPORT
SERVICE



National Centre for Technology in Education
Ionad Náisiúnta don Teicneolaíocht san Oideachas

Incorporating Computer Control into Student Project Work using a PLC Microcontroller

Incorporating Computer Control into Student Project Work using a PIC Microcontroller

Published by:

The National Centre for Technology in Education
And
T4 – Technology Subjects Support Service

National Centre for Technology in Education
Dublin City University
Glasnevin
Dublin 9
Tel: +353 1 700 8200
Email: info@ncte.ie
Web: www.ncte.ie / www.scoilnet.ie

T4 – Technology Subjects Support Service
Galway Education Centre
Cluain Mhuire
Wellpark
Galway
Tel: +353 91 745 650
Email: admin@t4.ie
Web: www.t4.ie

Copyright © National Centre for Technology in Education 2010.

Permission granted to reproduce for educational use providing the source is acknowledged.
Copying for any other purposes prohibited without the prior written permission of the publisher.

Please note

- Screenshots used in this manual may appear different to those on computer screens used by participants; variations in versions of the software and differing operating systems may be in use.

GENIE Design Studio is the electronic simulation software on which this module is based. This does not imply any endorsement by the NCTE of a product or company. The reader should be aware that typically there are many products and companies providing similar services in areas related to ICT. Participants should be as informed as possible before making decisions on purchases of ICT products or services.

Incorporating Computer Control into Student Project Work using a PIC Microcontroller

Duration

12 hours

Aim

The aim of the **Incorporating Computer Control into Student Project Work using a PIC Microcontroller** course is to support the integration of ICT into the learning and teaching of basic electronic principles and the application of these principles in the design and production of circuits suitable for student project work.

The course involves hands-on practical work for teachers and focuses on using GENIE Design Studio to enhance the teaching and learning of Electronics across the technology subjects.

Objectives

Participants will be able to:

- Understand how the evolution of electronic engineering has affected control technology since the 1940s.
- Appreciate the need to improve student project work through the use of control technology.
- Build a generic PIC project board suitable for use in student projects.
- Program the PIC board, using the free *GENIE Design Studio* software, to control a range of inputs and outputs relevant to a range of student projects.
- Carry out basic troubleshooting and fault finding where necessary.

The course material CD contains folders that are designed to be dealt with in order.

Prior to commencing the course, it is assumed that participants have basic skills in ICT and Electronics.

Table of Contents

1 GENIE Design Studio Interface	4
2 Connecting the PIC Board to the PC.....	5
3 Testing the PIC board	5
4 Controlling Single Outputs.....	10
5 Controlling Multiple Outputs.....	12
6 Simulating a Program	13
7 Creating a Loop & Debugging Live	14
8 Playing Sounds	15
9 Playing Tunes	15
10 Responding to Digital Signals	17
11 Responding to Analogue Signals.....	20
12 Using Subroutines	22
13 Compare	23
14 Increment and Expression Commands.....	24
15 Parallel Processing	27
16 GENIE Design Studio Help	28
17 Further Exercises.....	29
18 Useful Links	30

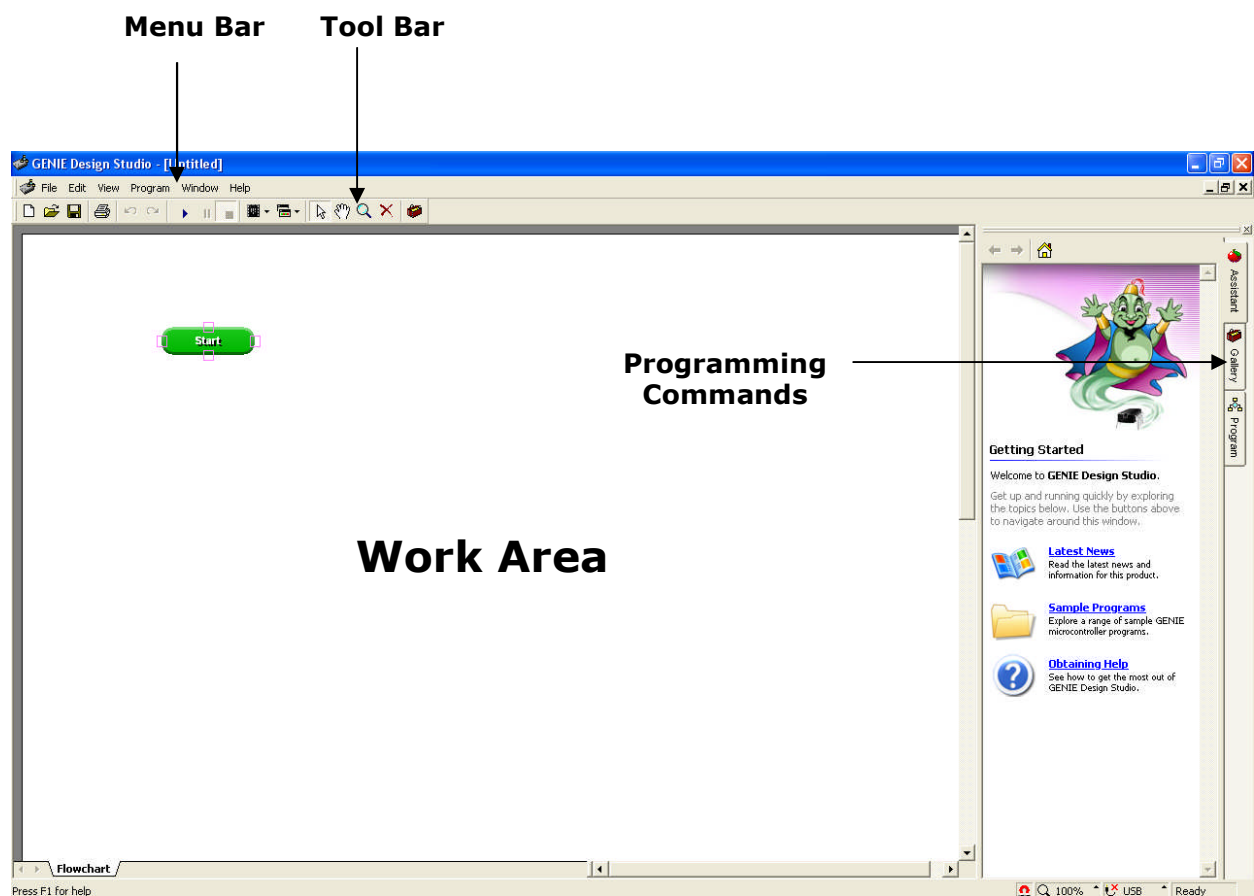
GENIE Design Studio Training

To open GENIE Design Studio: Click the GENIE Design Studio desktop or use the start menu.



icon on the

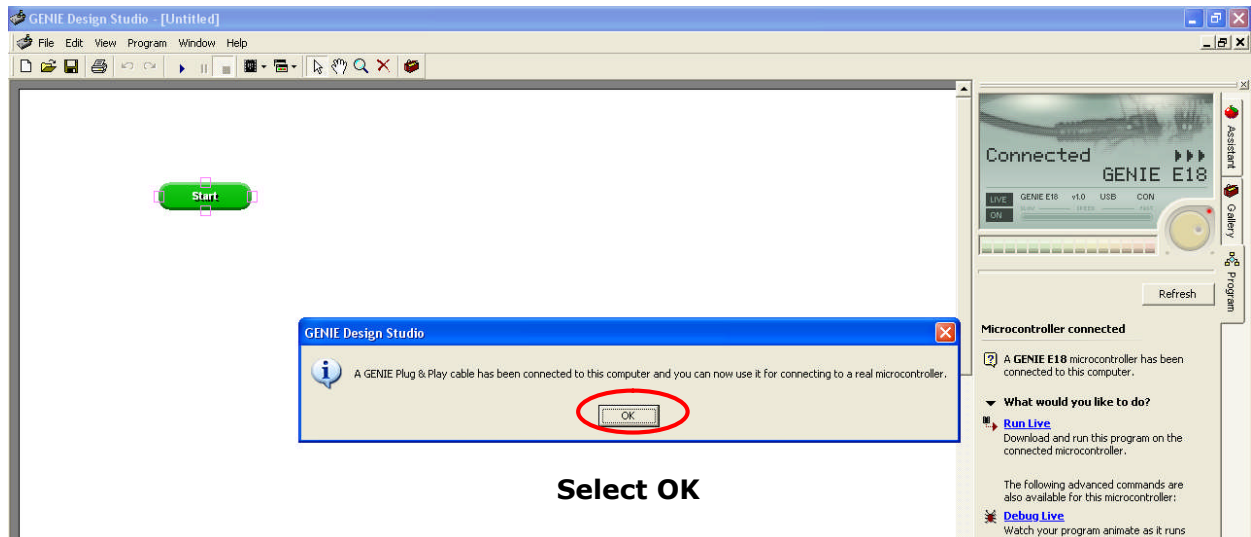
1 GENIE Design Studio Interface:



2 Connecting the PIC Board to the PC

Connect your PIC board to a suitable power supply – either via the 9v PSU or using 4 or 6 x 1.5v batteries. It is not recommended to use a 9v PP9 battery due to their tendency to run down quickly and inability to sustain a steady voltage.

Now connect the GENIE USB Plug & Play Cable. The following should appear:

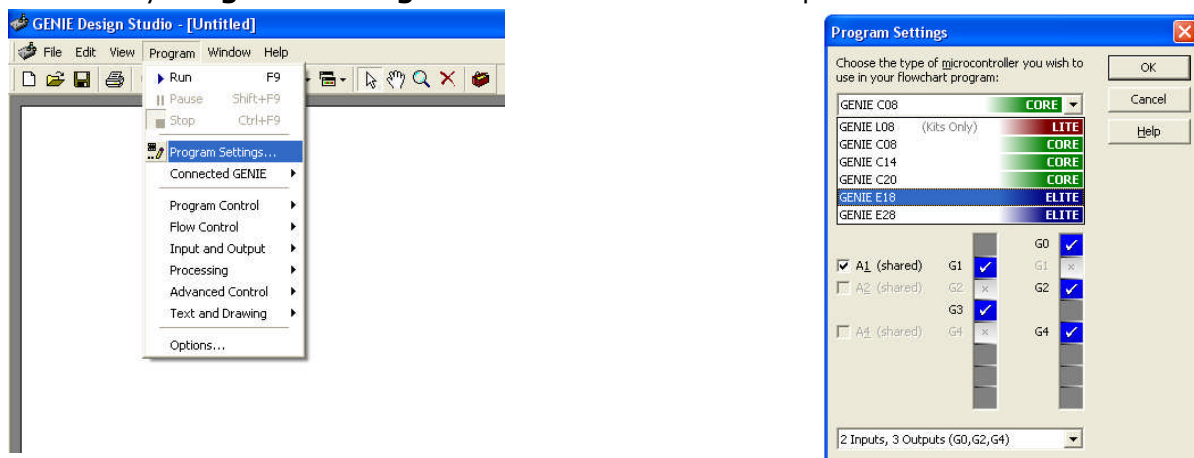


NOTE: You must install the driver for this cable beforehand. Instructions are available in the Software folder on your CD or in the Help and Support Guides in the main Help menu.

3 Testing the PIC Board

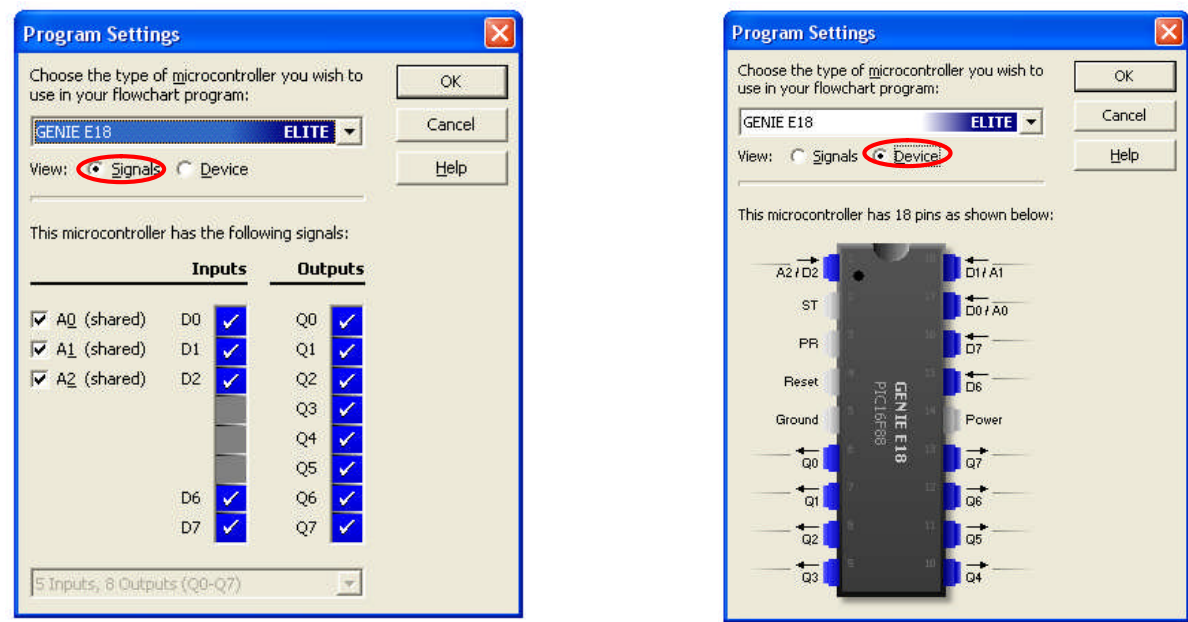
Before commencing programming, test that the outputs are all working properly.

Firstly tell the software what PIC chip you are using. Select **Program** from the menu bar followed by **Program Settings**. Select the **GENIE E18** chip.



NOTE: Please consult the file **Setting GENIE E18 chip as default.avi** located in the folder **4. GENIE Design Studio Training** for instructions on how to set the E18 chip as the default chip selection. Otherwise, it will have to be selected each time a new program is started.

The **Program Settings** window provides information about the inputs and outputs available on the selected chip:



Number of inputs and outputs available

Chip configuration

The following conventions are followed for identification purposes:

- Outputs are denoted by the letter 'Q'
- Analogue inputs e.g. LDR or thermistor are denoted by the letter 'A'
- Digital inputs e.g. switches are denoted by the letter 'D'

GENIE E18

ELITE

Fully-featured 18-pin GENIE microcontroller.

Pins

The GENIE E18 microcontroller has 18 legs (known as pins) and these are used as follows:

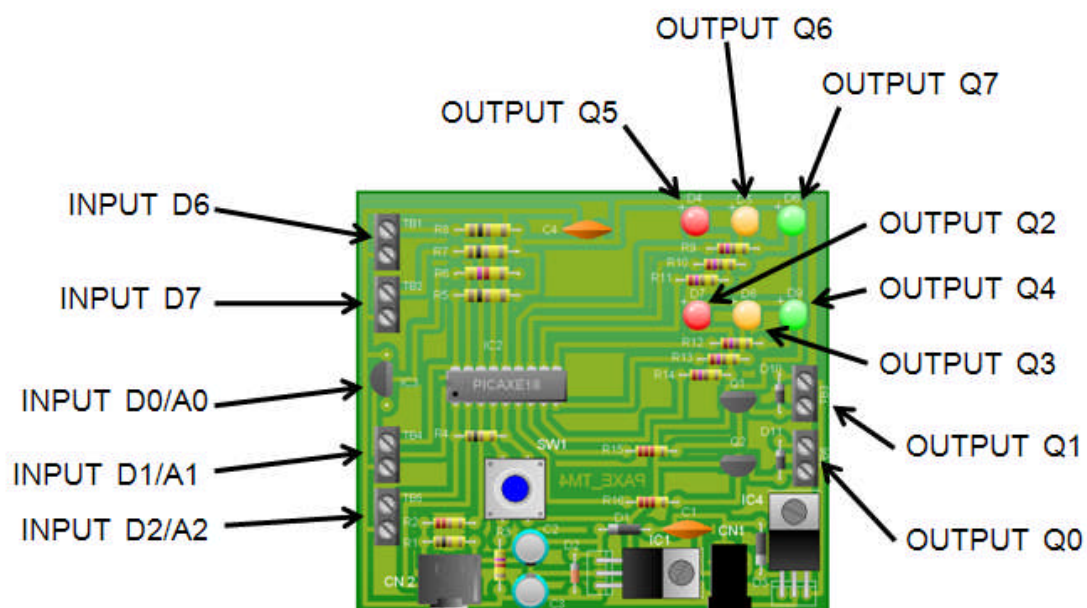
Pin	Description
1	Analogue input A2 or digital input D2
2	Status output (ST)
3	Programming input (PR)
4	Reset (when pin goes low)
5	Ground (zero volt) supply voltage
6	Digital output Q0
7	Digital output Q1
8	Digital output Q2
9	Digital output Q3
10	Digital output Q4
11	Digital output Q5
12	Digital output Q6
13	Digital output Q7
14	Power supply voltage (2-5.5V only)
15	Digital input D6
16	Digital input D7
17	Analogue input A0 or digital input D0
18	Analogue input A1 or digital input D1

For the purposes of this course we will connect the following inputs and outputs to the PIC board:

INPUTS	CONNECTED DEVICE
D0/A0	NTC Thermistor 20K
D1/A1	NORPS12 LDR
D2/A2	NONE
D6	Micro switch
D7	Push Switch PTM

OUTPUTS	CONNECTED DEVICE
Q0	Speaker/Sounder
Q1	Motor
Q2	Red LED
Q3	Yellow LED
Q4	Green LED
Q5	Red LED
Q6	Yellow LED
Q7	Green LED

The specified devices should be connected to the PIC board as shown during manufacture:



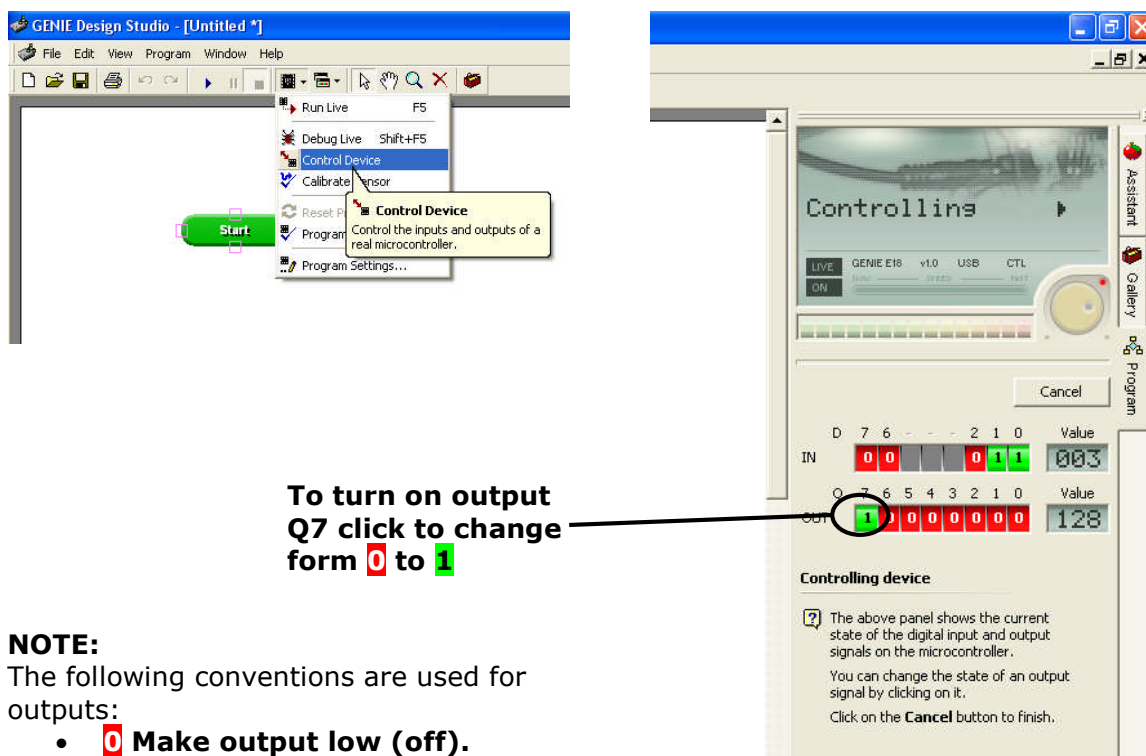
NOTE:

- Outputs Q2 to Q7 are connected to a 330Ω resistor in order to reduce the voltage to approx 2.2V, suitable for use with most common LEDs.
- On the student project board at the position for outputs Q0 and Q1 there are terminal blocks so that outputs such as bulbs and motors can be attached. These outputs require a larger current than is available directly from the PIC. To overcome this, a transistor is used with its base attached to the PIC. The small current from the PIC is used to switch on the transistor allowing a large current to enter the collector. This makes it possible to use outputs like bulbs, motors and relays etc.

- c) Some DC motors can cause the PIC to malfunction due to the generation of electromagnetic interference (sometimes called 'noise') by the motor commutators. To counteract this, fit a 220nF suppression capacitor directly across the motor connections or use a solar type motor.

There are 2 simple methods of testing that outputs are working properly before programming:

1. Open the **Microcontroller** menu from the tool bar and select **Control Device**. In the **Controlling** menu, click on each output from Q7 to Q0 in turn changing it from a **0** to a **1**. Each connected output on the PIC board should activate in turn.



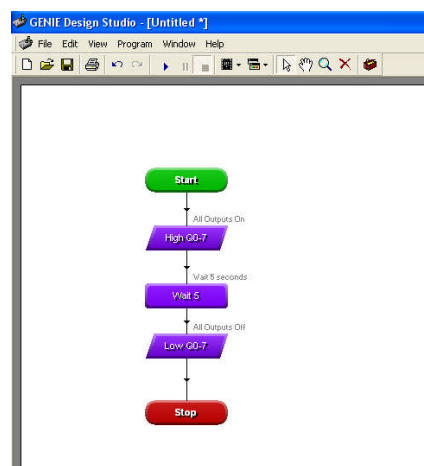
NOTE:

The following conventions are used for outputs:

- **0** Make output low (off).
- **1** Make output high (on).
- **T** Toggle this output (where high is set to low and low is set to high).
- **X** Leave output alone.

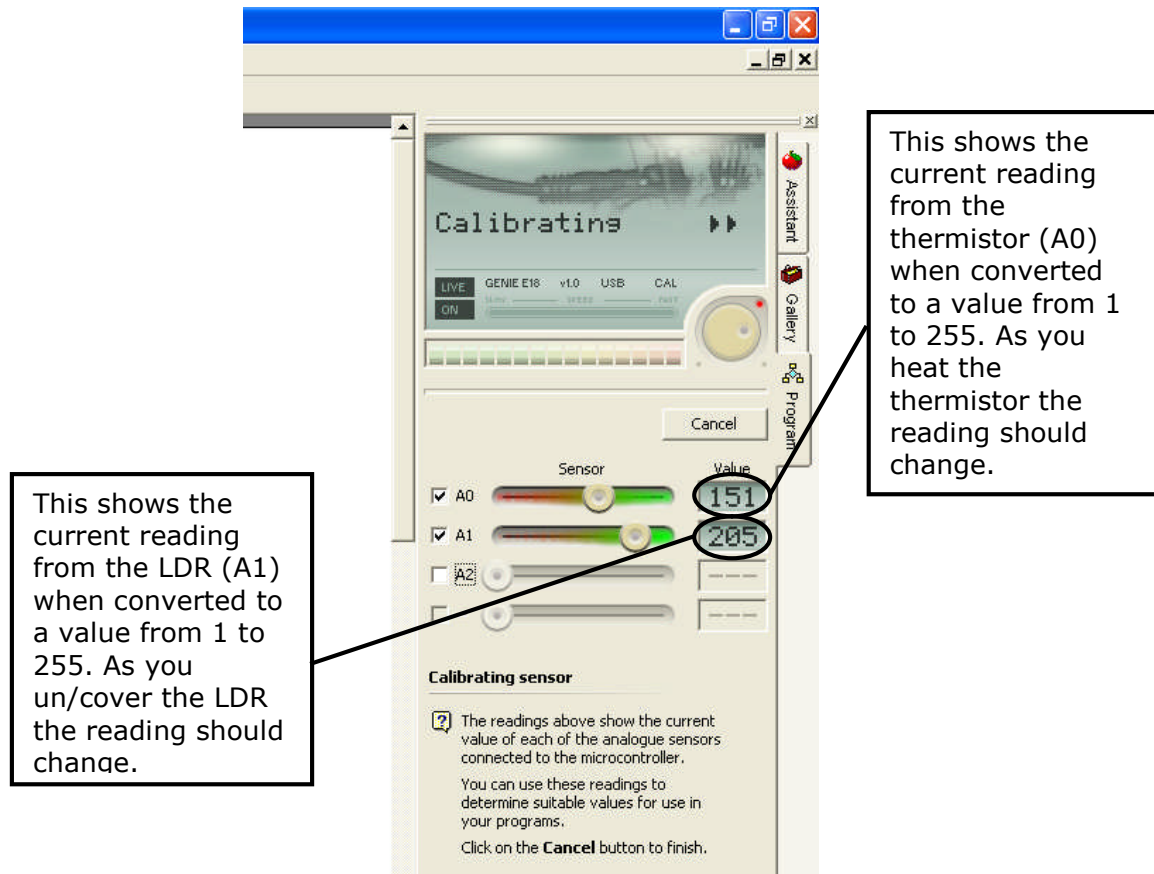
Using the Control Device command is very useful as it helps to check if PCB tracks and soldered joints are secure and working properly.

2. Download a simple program to activate all the outputs such as the one shown below. This program activates all the outputs for 5 seconds. Writing programs will be dealt with in the next section.



We can also check that our analogue sensors are operating:

In the **Microcontroller** menu, select **Calibrate Sensor**. Ensure that the thermistor is connected in A0/D0 input and the LDR is connected in A1/D1 input.

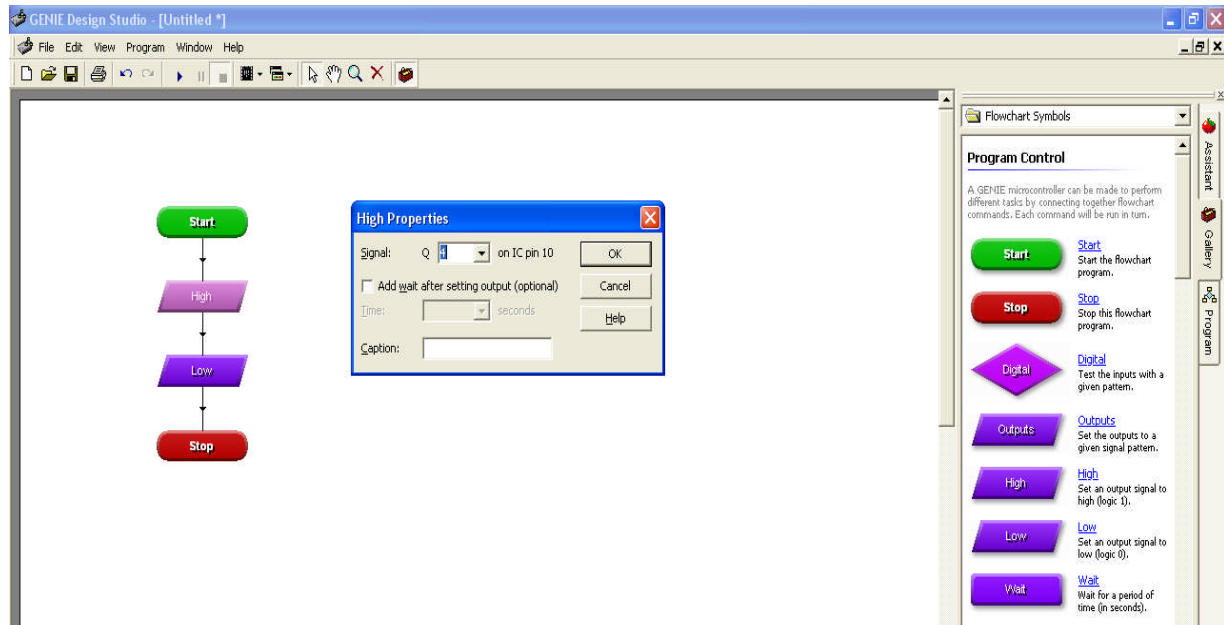


NOTE:

1. For the purposes of this training, we are not using input A2/D2.
2. It is not necessary to always connect the LDR into A1/D1. It could be connected into either of the other two combined analogue/digital inputs i.e. A0/D0 or A2/D2. This applies to all analogue inputs.
3. This method of calibrating analogue sensors is very useful because the readings shown here by the software can be used to determine suitable values for use in your programs later.

4 Controlling Single Outputs

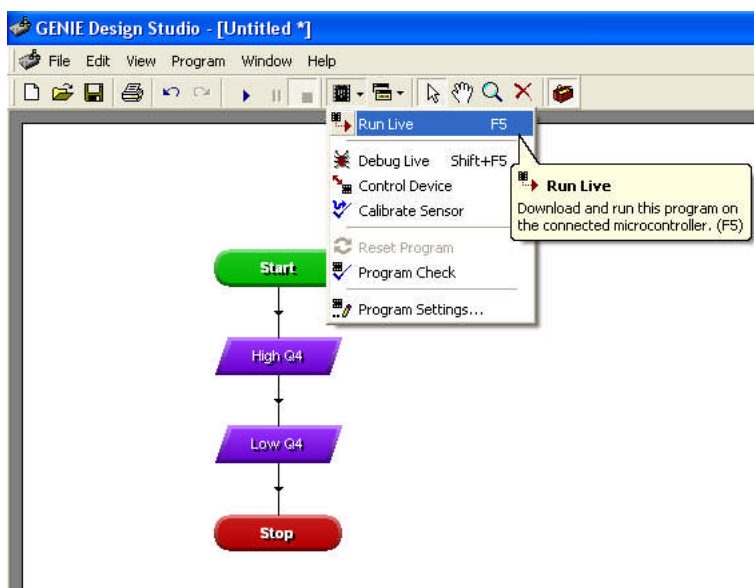
Construct the following flowchart by left clicking and dragging the programming commands from the **Gallery**. Commands will automatically link together when you drop them in place.



Click on the **High** command to open the **High Properties** window. Select **Q4** and then **OK**. This will turn on (referred to as high) output Q4.

Click on the **Low** command to open the **Low Properties** window. Select **Q4** and then **OK**. This will turn off (referred to as low) output Q4.

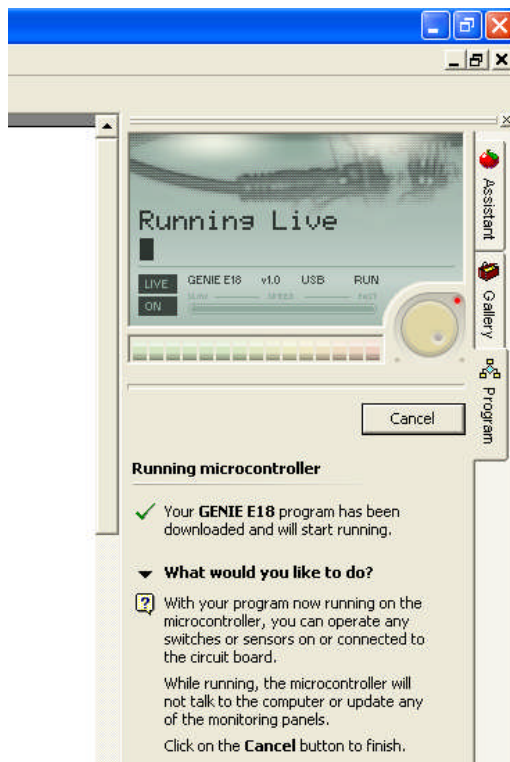
In order to download the program to the GENIE chip select **Run Live** from **Microcontroller** on the tool bar as shown below:



The **Run Live** command allows you to download and run your program at full speed just as if the PCB were not connected to the computer at all. While in this mode you will not be able to animate nor monitor the state of inputs, outputs and variables etc.

NOTE: Ensure you have selected the correct GENIE E18 chip as shown in section 1.3

The software will now show that the program is running on the board as follows:



QUESTION:

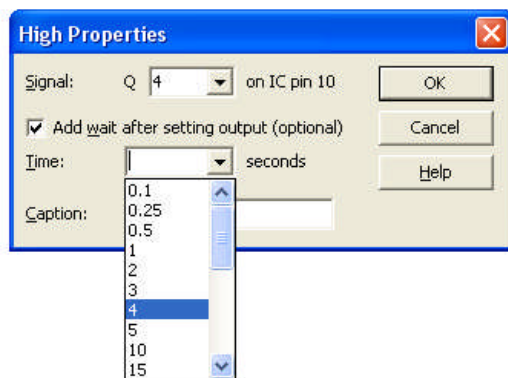
Why does nothing appear to be happening?

ANSWER:

We instructed the software to turn on output Q4 and then turn it off again.

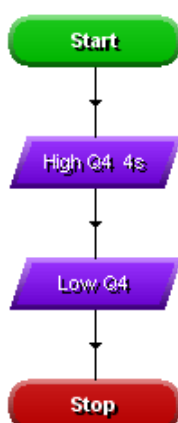
All this happened so quickly that we did not have time to see it!

This problem can be rectified by adding a wait after setting the output as shown below:



Open the **High Properties** window again, tick the box to add wait after setting output and then select 4 seconds. Click **OK**.

NOTE: Many commands such as High and Low allow you to automatically include a Wait. It can also be included as a separate command as shown in the next section.



The 4s delay is indicated beside the output.

Now download the edited program using the **Run Live** command.

QUESTION:

What happens?

ANSWER:

The LED connected to Q4 comes on for 4 seconds.

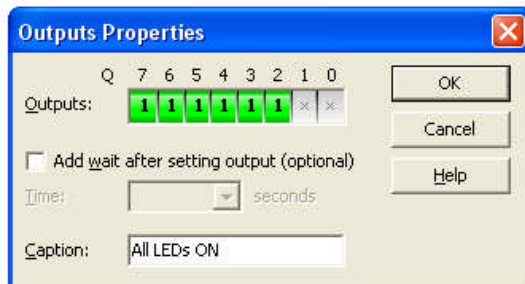
5 Controlling Multiple Outputs

Set up the commands shown below as a new program:

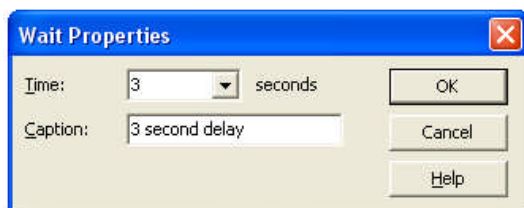
The Outputs command lets us control all of the 8 possible outputs simultaneously:

Outputs

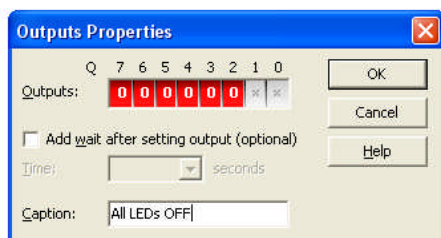
Turn on the 6 LEDs Q2 to Q7 by activating the command and editing the Outputs Properties window as shown below:



This is a separate **Wait** command. it can be used anywhere in a flowchart in order to delay it or to set a time to keep the outputs on for as shown:



Set the **Outputs** command again to turn off the LEDs Q2 to Q7 as shown:



NOTE: Captions are very useful as a clear indicator of what is happening and students should be encouraged to use them.

Now download the program using the **Run Live** command. The 6 LEDs in outputs Q2 to Q7 will activate for 3 seconds.

NOTE: Pressing the on board reset switch will re-run the latest downloaded program.

We will now look at how programs can be simulated before downloading.

6 Simulating a Program

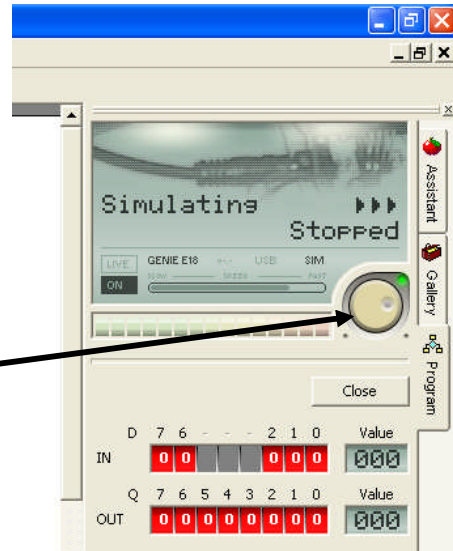
Using the same program, select the **Run** command  on the tool bar.

The Simulating window shown will open:

This panel displays what is happening to each of the inputs and outputs as the program is put through a simulation.

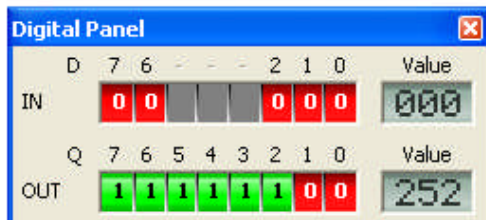
As the simulation progresses, each command in the flowchart highlights as it is processed.

The speed at which the simulation runs at can be adjusted by rotating the **Speed Dial**.

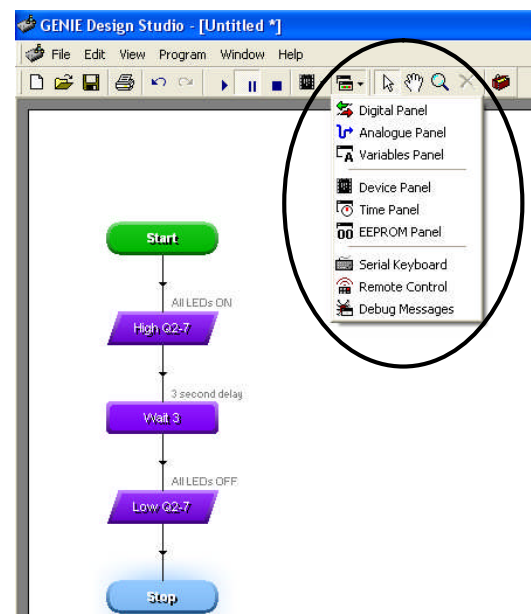
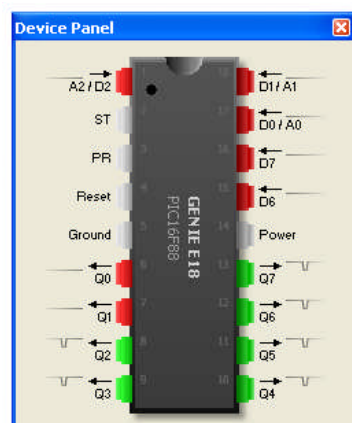


By selecting **Panels** on the tool bar, a menu appears offering various panels that can be opened in order to monitor the operation of a program either during simulation or in debug live mode which we will look at later:

A separate **Digital Panel** similar to that shown already is available. It should display the following during operation:

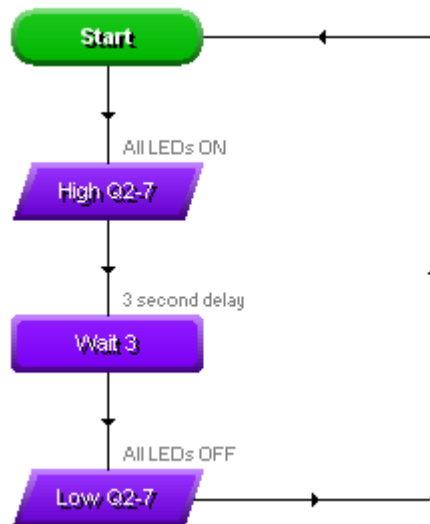


The **Device Panel** will display what is happening on the actual GENIE chip during operation of the program:



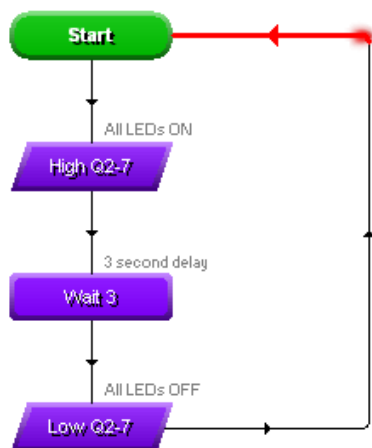
7 Creating a Loop & Debugging Live

Create the new program shown below by removing the stop and adding a loop back to the start:

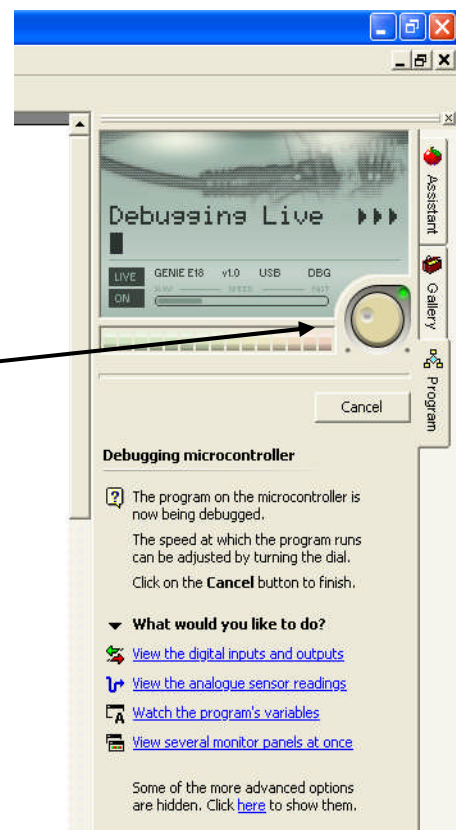


Select **Debug Live** from the **Microcontroller** menu. An information panel will appear giving details of the program that is about to be debugged. Read the information and click on the **Start** button when you are ready to begin.

Once started the software will enter **Debugging Live** mode allowing you to see your program animate. It will be coloured red to indicate it is animating in live mode.



The **Speed Dial** can still be used to control how fast it animates.



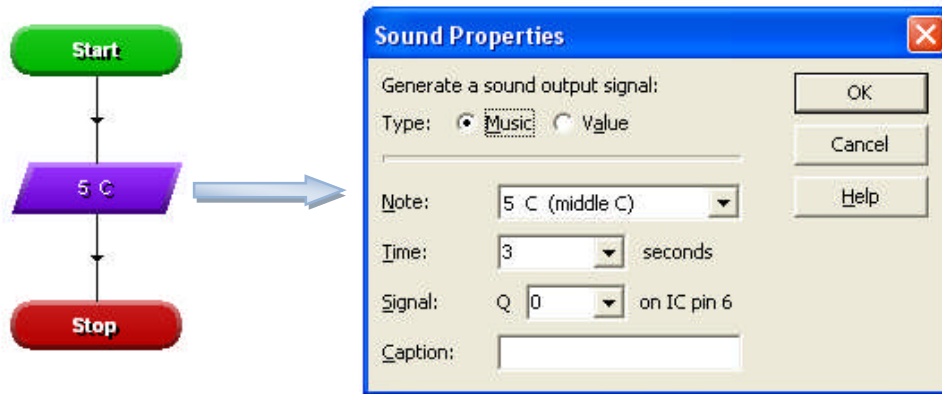
Debugging Live is very useful for watching different aspects of programs in terms of inputs and outputs as they operate, in order to identify a problem or improve performance. The option to monitor several different panels at once is very useful.

NOTE: Debugging Live can be paused allowing you stop the program and check your PCB.

Question: Why will this program not operate as expected if we download using Run Live?

8 Playing Sounds

Create the following program using the settings shown. The **Sound** command is located in the **Input and Output** folder in the main **Gallery**:



We must select the output that has a suitable speaker connected in order to play the middle C note for 3 seconds – in this case Q0.

Download and test the program using either **Debug Live** or **Run Live**. Disconnect the USB and test the program using the reset switch on the PCB.

EXERCISE 1

Create a program to play a simple tune repeatedly, for example:

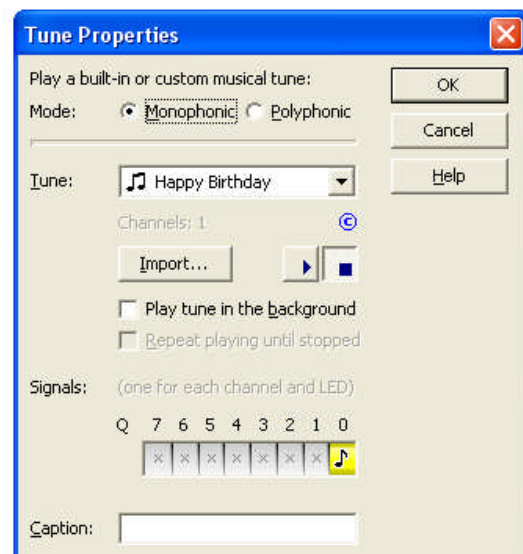
- Batman theme
- Twilight Zone theme

Test the program on the PCB and save as Ex1.


9 Playing Tunes

Create the following program using the settings shown. The **Tune** command is located in the **Input and Output** folder in the main **Gallery**:

Monophonic tunes play one note at a time. **Polyphonic** tunes play two notes at a time. They sound better but require a second (hidden) flowchart and take up more memory space. You can choose in-built tunes or **Import** suitable music files. The **Play** button allows you listen to a preview of the tune.



There are a number of features available for playing tunes:

- Choosing **Play tune in the background** means that the flowchart does not wait for the tune to stop playing before moving on to the next command.
- A background tune will play once before stopping unless the **Repeat playing until stopped** option is enabled.
- You can stop a tune playing in the background anytime by using the **Stop Tune** command  located in the **Input and Output** folder.

- **Signals** determines which of the outputs is connected to your speaker component – Q0 in this case. Polyphonic music requires a separate output for each musical channel.
- You can also select LED outputs to flash in time with the music:



Musical output



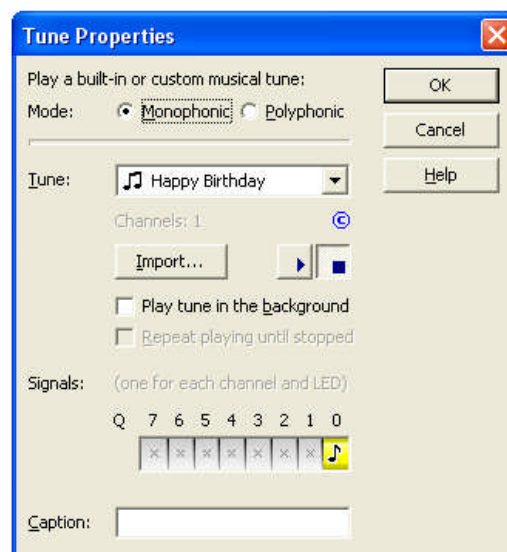
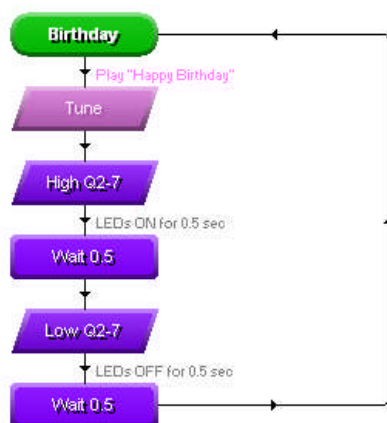
LED output

EXERCISE 2

A student has been trying to create a program that will:

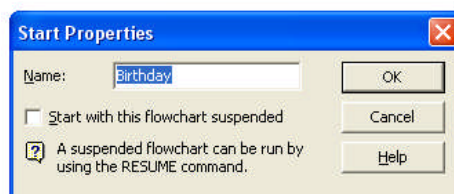
- Play 'Happy Birthday' repeatedly.
- Flash all the LEDs on/off while the tune is playing.

He has created the following program, but it is still not doing exactly what he wants:



1. Create the flowchart show and observe what it does using Debug Live.
2. Edit the flowchart so that it does what he wants it to do. Save as Ex2.

NOTE: Start has been replaced by 'Birthday' by editing the **Start Properties** window. This should always be done as a reminder of what a flowchart does as shown:



10 Responding to Digital Signals

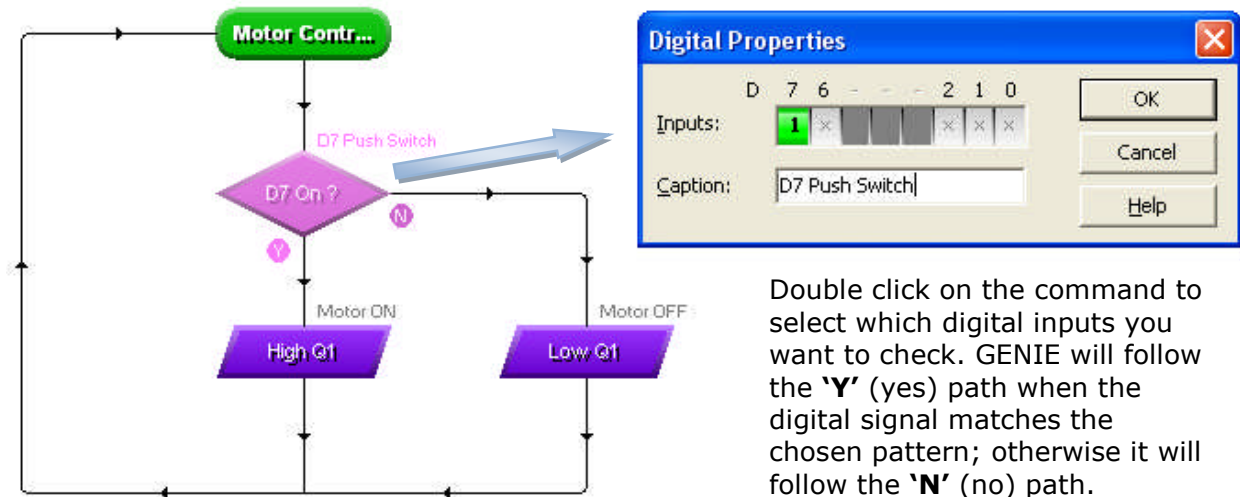
Some types of signal, such as push switches or micro switches, can only be either on or off. These are known as digital signals.

We will be using the **Digital** command, found in the **Flow Control** folder to respond to digital signals.

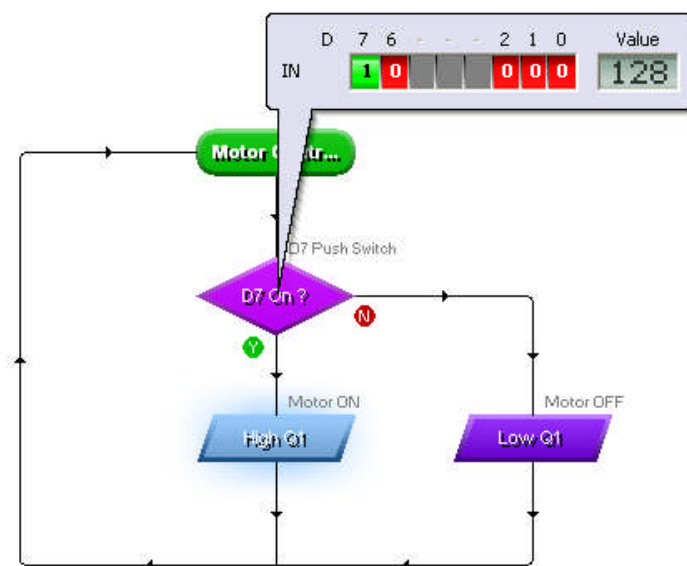


The Digital command allows the program to make a decision based on whether the digital signal is on (high) or off (low). When a digital signal is on, it has the value '1' whereas when it is off, it has the value '0'.

Set up the following flowchart used to control the motor on output Q1 using the push switch on input D7:



If we select **Run** to simulate the program we will initially observe the program following the 'N' path through **Low Q1**. There are two options that allow us simulate switch D7 being pressed:



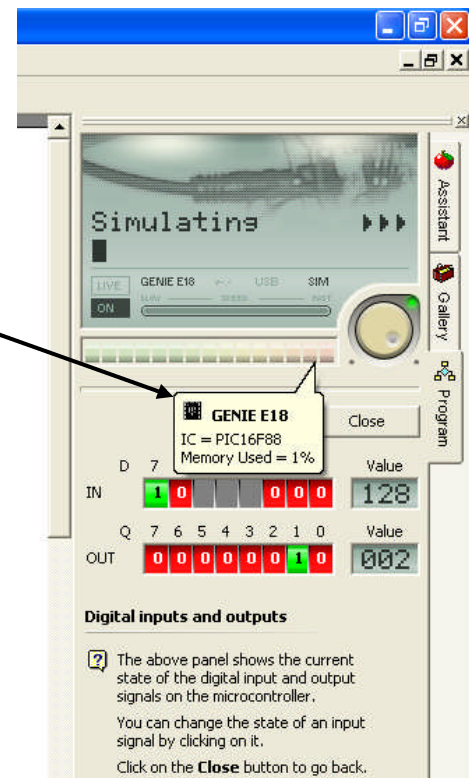
1. Click the **Digital** command to open the window shown. Left click and hold down on D7 to simulate the switch being pressed. The program will now follow the 'Y' path through High Q1 turning on the motor. When D7 is released the program will revert to the 'N' path and the motor Q1 will be off.

2. Select the **View the digital inputs and outputs** option in the **Simulating** window. As before, left click and hold down on D7 to simulate the switch being pressed.

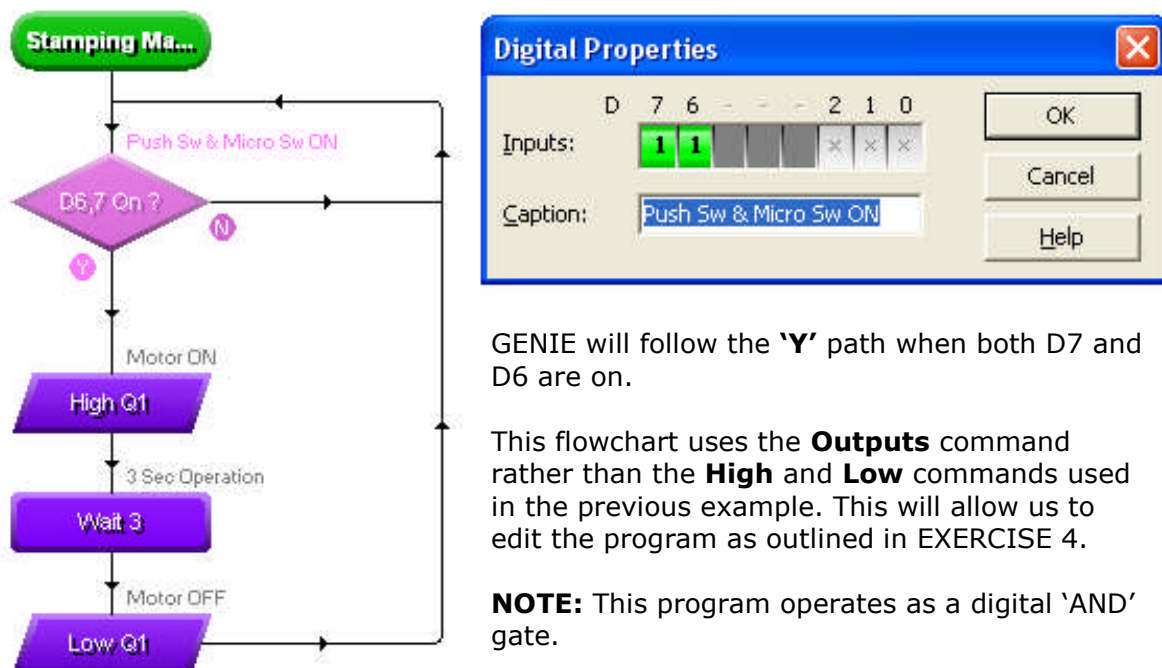
NOTE: The Memory Bar shows how much space has been used on the microcontroller. Hold the mouse over the bar for more details as shown.

EXERCISE 3

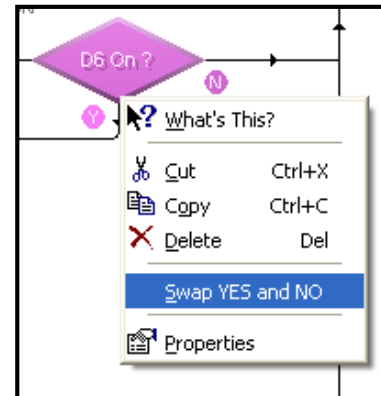
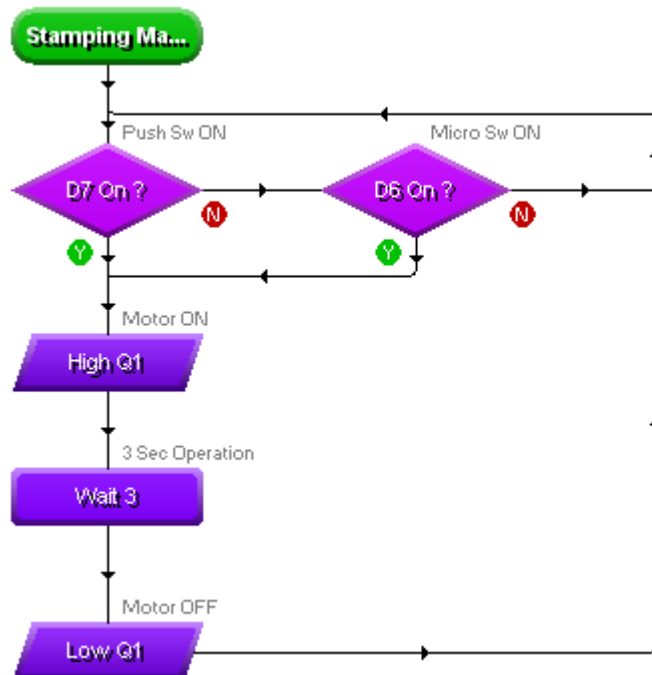
GENIE offers multiple options for observation of the inputs and outputs. Using **Debug Live** mode can you demonstrate how to monitor what is happening on the actual microcontroller device while input D7 is pressed?



Now consider the case of a stamping machine used for making aluminium ID tags. The machine is activated by pressing a push switch causing the motor to run for 3 seconds. However, for safety reasons the push switch will only activate the motor if a protective screen is pulled down far enough that it activates a NO (normally open) micro switch.



Edit the previous flowchart in order to create the program shown below:



NOTE: You can reverse the flow of control in a Digital command by clicking the right mouse button over the command and choosing **Swap YES and NO** as shown above.

EXERCISE 4

What type of digital gate does this program simulate?

Design a flowchart to simulate two switches connected to a digital 'NAND' gate.

Save as Ex4.

11 Responding to Analogue Signals

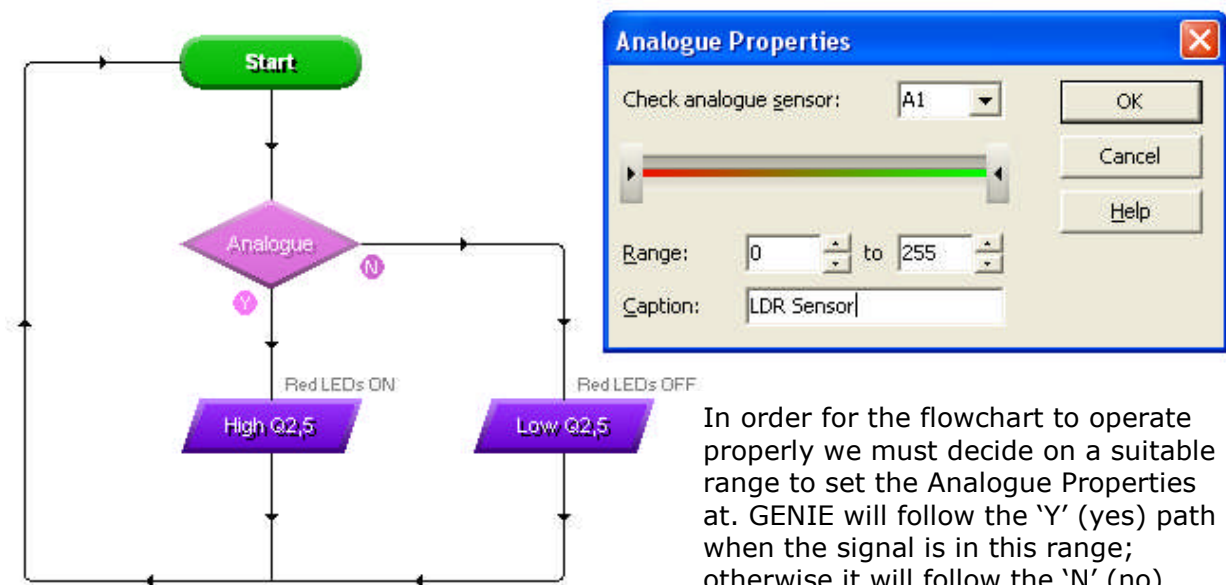
Other types of signal, such as temperature or light, can be at a number of different levels rather than just on or off (as in digital). These types of signal are known as analogue signals.

We will be using the **Analogue** command, found in the **Flow Control** folder to respond to digital signals.

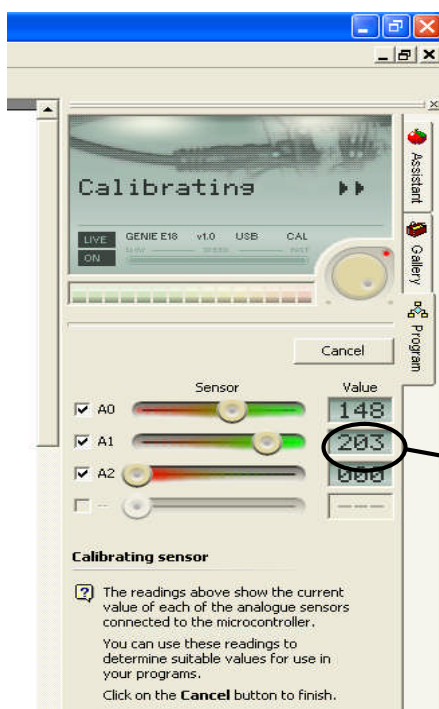


The Analogue command allows you to check whether a signal lies within a given range. With GENIE, analogue levels can vary between 0 (the lowest level) and 255 (the highest).

The following flowchart will be used to turn on the two red LEDs in outputs Q2 and Q5 when the ambient light level falls below a certain level as detected by the LDR input in A1:

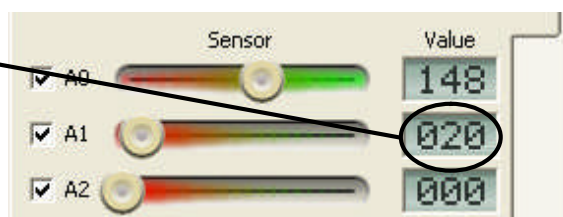


In order for the flowchart to operate properly we must decide on a suitable range to set the Analogue Properties at. GENIE will follow the 'Y' (yes) path when the signal is in this range; otherwise it will follow the 'N' (no) path. Ambient light conditions will vary depending on where you are currently located; near a window, using artificial light etc, so how do we decide on what range to set?

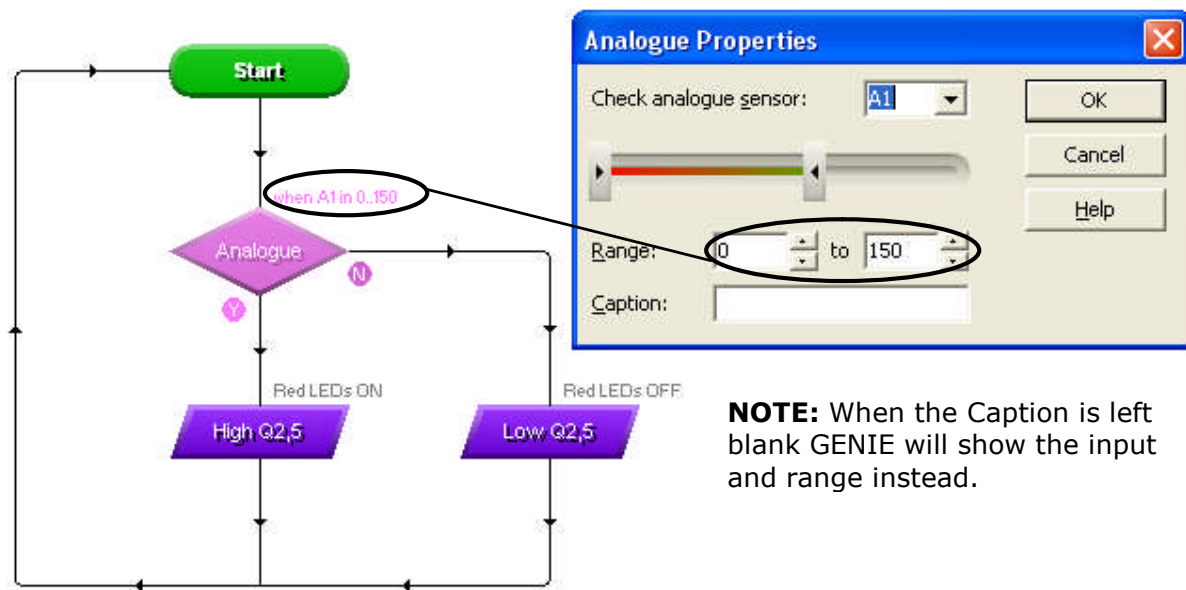


The answer lies in using the **Calibrate Sensor** function described earlier on page 7.

Currently, the LDR is giving a value of 203 in the light but when covered this reduces to 20 as shown:



This means that if the Analogue Properties are set from 0 to 150 GENIE will follow the 'Y' path (turning on Q2 and Q5) when A1 reads anything less than 150 (relatively dark), and follow the 'N' path (turning off Q2 and Q5) when A1 reads anything above 150 (bright):



Complete this flowchart and test using **Debug Live** with the **View several monitor panels at once** option selected. Observe the value of A1 change as you cover it and how Q2 and Q5 activate once it falls below 150 and deactivate when A1 rises above 150.

EXERCISE 5

What are the advantages of being able to calibrate sensors in this way?

Design a flowchart that will turn on a motor (Q1) when the temperature rises to a set level.

Use Calibrate Sensor to identify a suitable value range for the thermistor input A0.

Save as Ex5.

12 Using Subroutines

Subroutines allow you to break a large flowchart into smaller parts, which can then be reused. The flowchart shown uses a subroutine called *LED Warning* to flash the yellow LEDs Q3 and Q6 followed by the green LEDs Q4 and Q7 each time before a motor (Q1) is activated for 1 second.

Subroutines use the following 3 commands from the Flow Control folder:



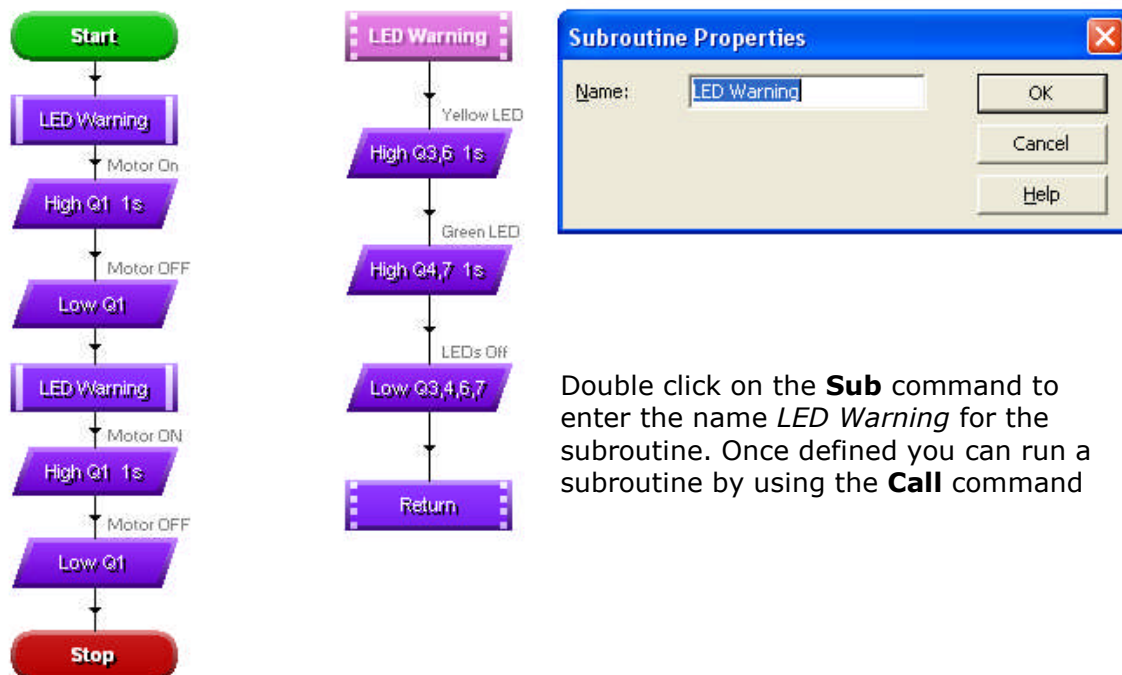
Used to define or name the subroutine.



Return to the main flowchart after a subroutine or an interrupt.

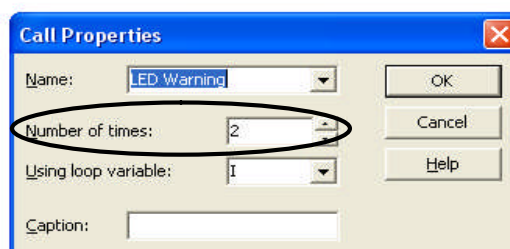


Calls or activates the chosen subroutine in the main flowchart.



The main flowchart calls the '*LED Warning*' subroutine and runs each of its commands until it reaches **Return**, at which point it goes back to the main flowchart.

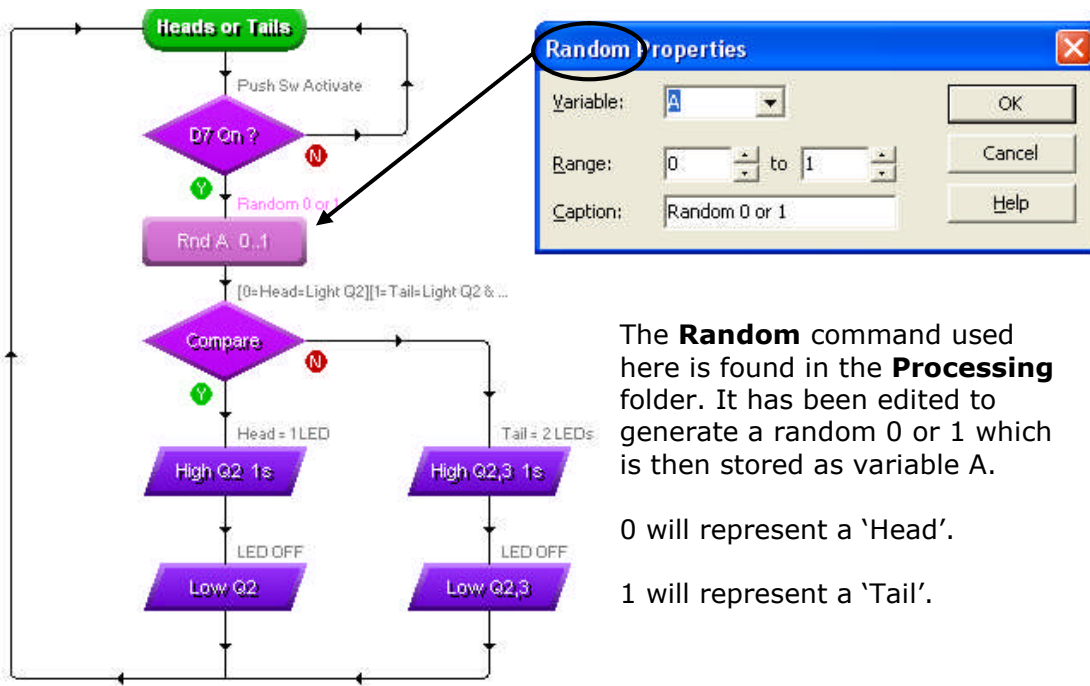
The Call command also gives the option of running the subroutine more than once:



13 Compare

The **Compare** command is found in the **Flow Control** folder. It performs a mathematical comparison between one or more variables or values. There are 10 variables named **A** to **J**. Each variable can hold a single whole number between 0 and 255. The flowchart will follow the 'Y' (yes) path if the comparison is true otherwise the 'N' (no) path will be followed.

Create the flowchart below designed to simulate the tossing of a coin with LED Q2 lighting to simulate a 'Head' and LEDs Q2 & Q3 lighting to simulate a 'Tail'. The simulation is only activated when the push switch on input D7 is pressed:

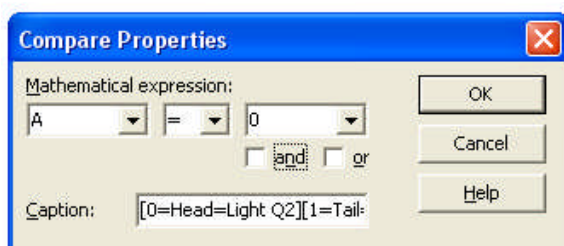


The **Random** command used here is found in the **Processing** folder. It has been edited to generate a random 0 or 1 which is then stored as variable A.

0 will represent a 'Head'.

1 will represent a 'Tail'.

The Compare command allows you to specify which variables or values are to be compared:



In this case, if the value stored in A is a 0 then the flowchart will follow the 'Y' path and light LED Q2 indicating a 'Head'.

If the value stored in A is a 1 then the flowchart will follow the 'N' path and light LEDs Q2 and Q3 indicating a 'Tail'.

Selecting either the **and** or **or** options will allow you to perform a second comparison at the same time. With the **and** option checked the 'Y' path will only be followed when both expressions are true, while selecting the **or** option will mean that the 'Y' path will be followed when either, or both expressions are true.

EXERCISE 6

Create a flow chart to simulate the rolling of a die with one LED lighting to indicate a 1, two LEDs lighting to indicate a 2 and so on.

Pressing the push switch D7 causes a new roll each time.

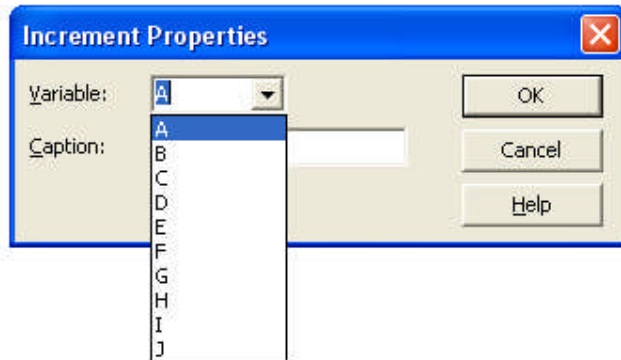
Save as Ex6.

14 Increment and Expression Commands

The **Increment** command



increases the value a variable by 1. It is located in the **Processing** folder.



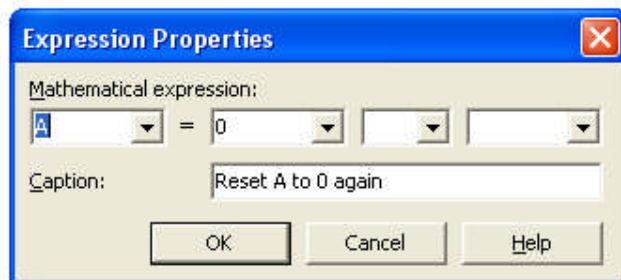
Double clicking on the **Inc** command allows you to specify which variable is to be incremented.

There are 10 variables named A to J and each can hold a single number between 0 and 255.

The Expression command



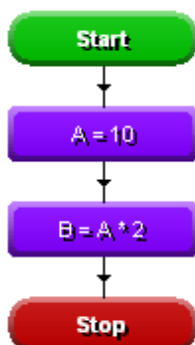
performs a mathematical operation and stores the result in a given variable.



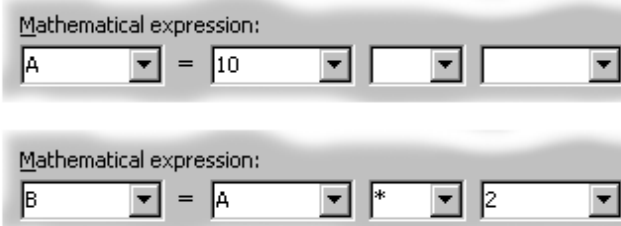
Double clicking on the **Expr** command allows you to decide how the value of a variable is to be changed.

Select the variable (A to J) to be changed from the left hand box and the value to be set in the second box as shown in the example.

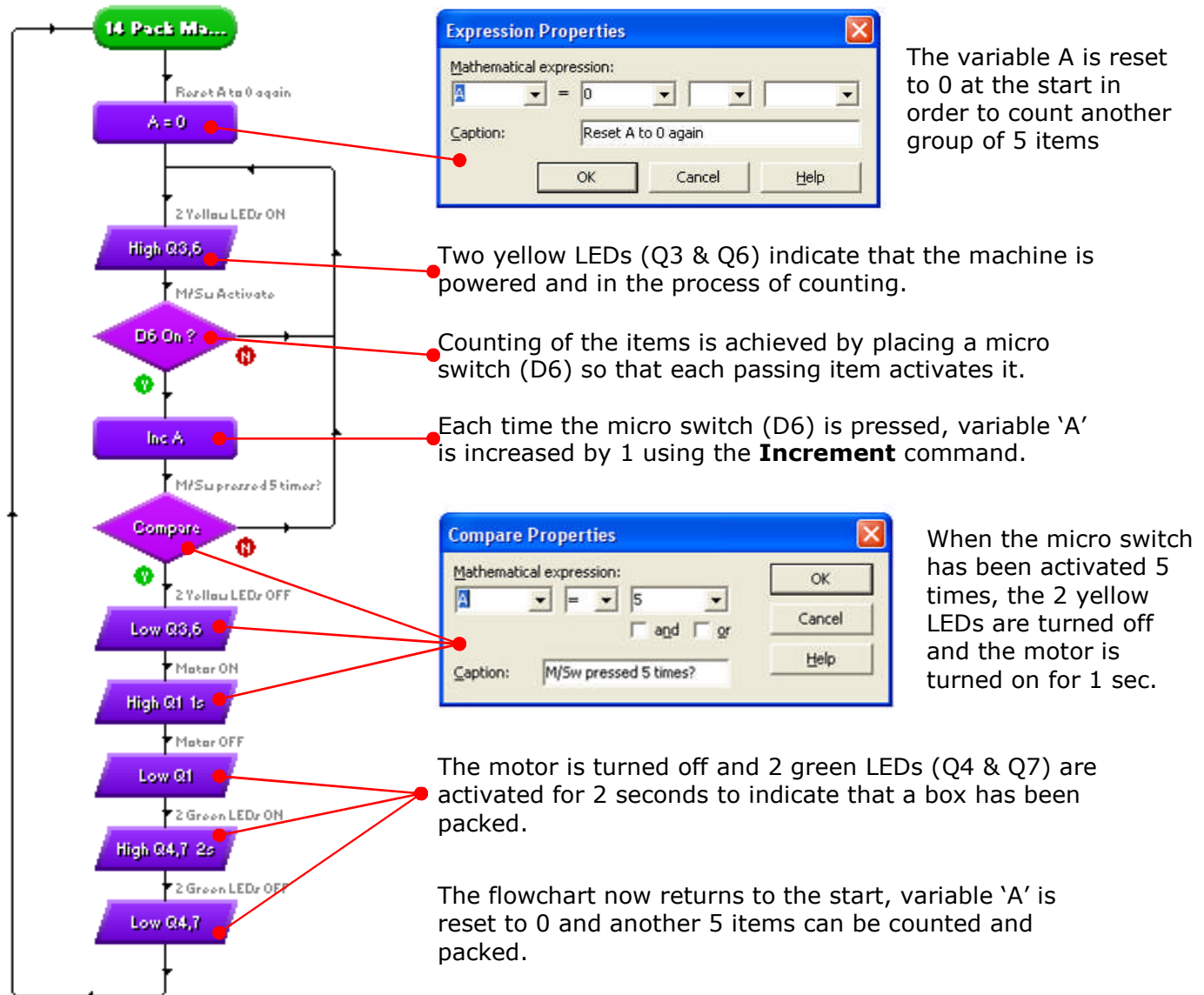
For more complex operations you can also use the third and fourth boxes as shown below:



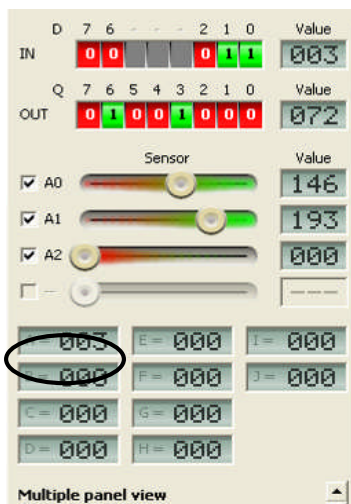
After the flowchart on the left has run, variable A will contain the value 10 and variable B will contain the value 20 (10×2).



Create the following flowchart, designed to simulate the control of a machine designed to pack every 5 items passing on a conveyor belt into a box.



Using **Debug Live**, download and animate the program on your project board.



Use the **Multiple panel view** to observe the inputs, outputs and variable 'A' increase by 1 each time switch D6 is pressed.

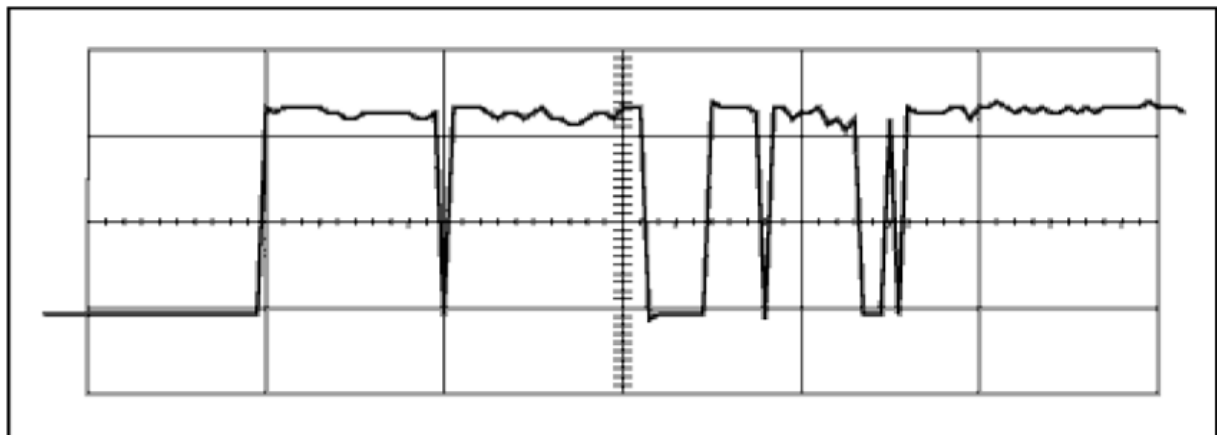
A=003 indicates that micro switch D6 has been pressed 3 times.

If you are happy that the program appears to be running OK on your board, download it using the Run Live option. Does it now work properly?

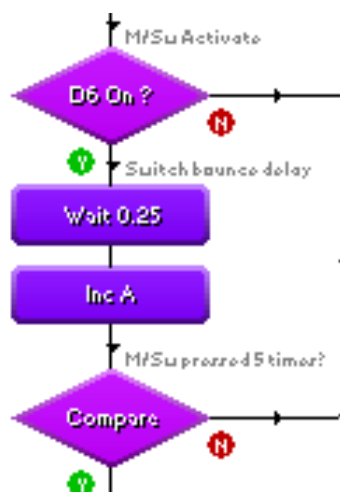
You will probably find that the program does not operate properly. More than likely when you press the micro switch D6 once the program jumps straight to activating the motor followed by the 2 green LEDs and then returns to the start. It does not appear to wait until the switch has been pressed 5 times?

Switches can do some really odd things. Most engineers learn this after connecting a switch or a relay to a digital system. Switches don't make and break cleanly on the time scales of digital systems. Instead, a typical switch makes multiple transitions during the tens of milliseconds required to open or close, due to effects that include age, operating inertia, mechanical design, and the microscopic condition of the switch-contact surfaces. Commonly called "switch bounce," this behaviour is an inescapable fact of life.

After connecting a standard switch to a digital counting circuit, you can observe several counts on opening and several counts on closing. This erratic action can wreak havoc on data, because the exact number of counts does not necessarily repeat in the long term. Switch bounce is not consistent from unit to unit, lot to lot, or even over the life of an individual switch. Some switch types don't appear to bounce when new, but all mechanical switches bounce sometimes. Nothing can ensure that another switch of the same type will act the same way, or that a particular switch will remain bounce-free as it ages.



This oscilloscope display shows the rising-edge switch bounce for a small pushbutton switch showing an approximate 5ms bounce interval that includes 10 transitions between the off and on states. Like a bouncing ball, the switch-action frequency increases toward the right.



In order to overcome this problem in our program a short time delay (usually 0.25 to 0.5 seconds) can be introduced into the flowchart after the switch is activated as shown.

This allows the 'bounce' to settle before the switch input is used to control anything; in this case, before incrementing variable 'A' by 1.

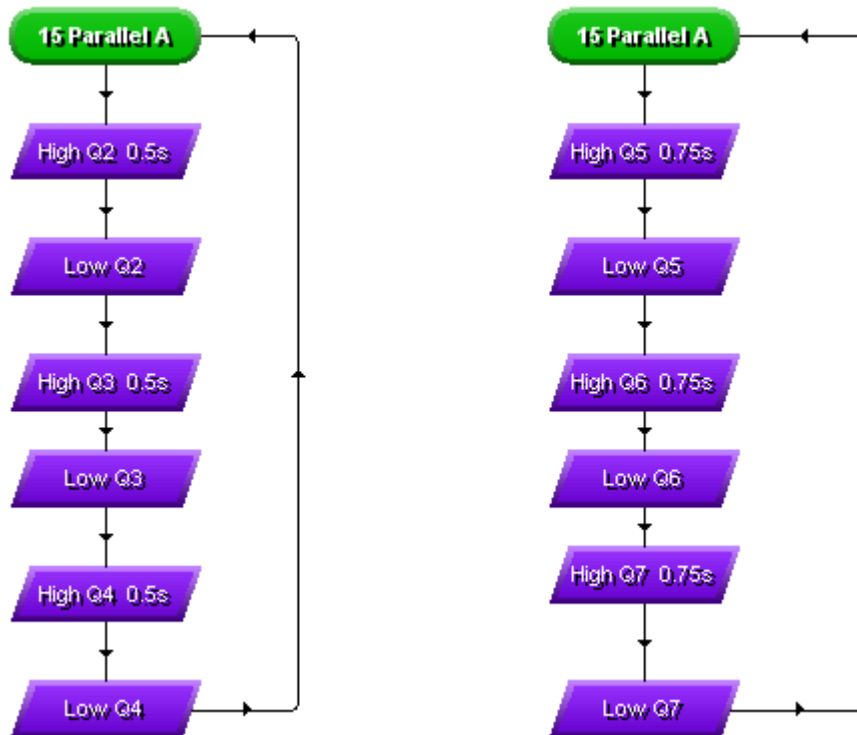
Edit your flowchart as shown and retest on your board using the Run Live option.

15 Parallel Processing

Adding another start to a flowchart program will cause the PIC microcontroller to run both flowcharts at the same time. This is known as **Parallel Processing**.

The GENIE E18 microcontroller will allow 4 separate flowcharts to be run simultaneously.

Create the flowchart shown below and download using the **Debug Live** option:



In the above example, two separate things happen at the same time:

1. LED outputs Q2, Q3 and Q4 are made to go high and low in sequence for 0.5 seconds each.
2. LED outputs Q5, Q6 and Q7 are also made to go high and low in sequence but for 0.75 seconds each.

This means that both groups of LEDs are flashing at different rates. This is known as being out of phase.

For programs with more than one flowchart you can choose for a flowchart to start suspended (not running) by selecting the **Start with this flowchart suspended** option found in the **Start Properties** window.

A suspended flowchart can be restarted by using the **Resume** command (in which case a flowchart name must have been entered).

The **Resume** command

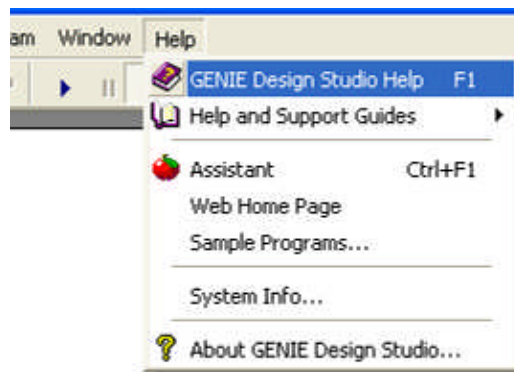


can be found in the Advanced Control folder.

16 GENIE Design Studio Help

These notes are only an introduction to GENIE Design Studio. It contains many more commands that have not been mentioned at all.

Explanations of these other commands can be found in the **Help** section of the software by clicking on GENIE Design Studio Help as shown below:



Then select **Flowchart commands** as shown:



Programming GENIE microcontrollers

GENIE **microcontrollers** allow you to add intelligence and control to your design projects.

The following topics lead you through the process of creating and downloading programs to a real GENIE microcontroller.



Introducing GENIE

- [Overview](#)
- [GENIE microcontrollers](#)
- [Understanding signals](#)
- [Program settings](#)

Flowchart programming

- [Introducing flowcharts](#)
- [Drawing a flowchart](#)
- [Linking commands together](#)
- [Simulating a flowchart](#)
- [Monitoring panels](#)
- [Flowchart commands](#)

Using GENIE

- [Connecting to the computer](#)
- [Downloading and running](#)
- [Debugging a program live](#)
- [Controlling a device](#)
- [Calibrating a sensor](#)

Project Boards and Kits

- [PCB108 > L08/C08 Activity Kit](#)
- [PCB208 > C08 Project Board](#)
- [PCB308 > C08 Jukebox Kit](#)
- [PCB214 > C14 Project Board](#)
- [PCB220 > C20 Project Board](#)
- [PCB118 > E18 Activity Kit](#)
- [PCB218 > E18 Project Board](#)
- [PCB228 > E28 Project Board](#)

Technical information

- [Device features](#)
- [Customizing GENIE options](#)

LITE [GENIE L08](#)

CORE [GENIE C08](#)
[GENIE C14](#)
[GENIE C20](#)

ELITE [GENIE E18](#)
[GENIE E28](#)

This will open a window displaying all current GENIE commands. By clicking on any command the user will be given a simple explanation of what the command does along with a sample flowchart showing how the command may be used.

17 Further Exercises

Exercise 7	A set of pedestrian traffic lights are located near a school. They stay continuously green for the traffic and red for the pedestrians unless a button is pressed by a pedestrian. When the button is pressed, amber light comes on for the traffic. The pedestrian light turns green when the traffic light turns red. The pedestrian has 10 seconds to cross before the lights turn back to the original setting.
Exercise 8	A system is required to control three separate lamps based on light levels. It automatically switches them on one by one as darkness falls, and switches them off in the same way as conditions grow lighter.
Exercise 9	Expand Exercise 8 in order to create a light meter. In bright sunlight the 6 LED outputs will be lit. As the light level falls, the LEDs will switch off one by one.
Exercise 10	You have been requested to design an alarm system which will sound a two note alarm when a burglar steps on a pressure pad (simulated by a micro switch). The two note alarm will continue to play until a push switch is pressed. When it is pressed the alarm turns off until reactivated by the pressure pad.
Exercise 11	Using the flowchart from Exercise 8, design a flowchart for an alarm system suitable for protecting a safe with both a pressure pad located on the floor in front of the safe and a method of activating an alarm if the safe door is opened. Include a reset push switch as before.
Exercise 12 Parallel Processing	<p>Intelligent houses will help conserve energy in the future. Design a system that will simulate the following automatically:</p> <ul style="list-style-type: none"> • Turn off the lights when they are not needed • Turn off the heating when the house is warm enough • Turn on ventilation fans if the house gets too warm <p>Use the following outputs as simulations:</p> <ul style="list-style-type: none"> ▪ 2 yellow LEDs Q3 and Q6 to simulate house lights ▪ 1 red LED Q2 to simulate the heating system ▪ Motor Q1 to simulate the ventilation fans
Exercise 13 Parallel Processing	<p>Students are required to design a project based on a vehicle theme in order to demonstrate some of the capabilities of their PIC project board. The vehicle should demonstrate the following:</p> <ol style="list-style-type: none"> 1. Headlights (Q3&Q6) activate automatically when darkness falls. 2. Air conditioning fan (Q1) activates automatically when temperature rises. 3. Horn plays a simple tune when activated by push switch (D7). 4. Brake lights (Q2 & Q5) flash in a set sequence when activated by the pedal/micro switch (D6). <p>The vehicle structure should be designed and manufactured in order to allow the observer see as much of the PIC project as possible and to allow them interact easily with the required inputs.</p>

18 Useful Links

The following website links will be useful for further study in the area of PIC microcontrollers:

- <http://www.genieonline.com/>
- <http://www.rev-ed.co.uk/>
- <http://www.picaxeforum.co.uk/>
- <http://www.logicator.co.uk/>
- <http://www.technologystudent.com/pics/picdex1.htm>
- <http://www.ajbox.co.uk/>
- <http://www.new-wave-concepts.com/>
- http://www.youtube.com/watch?v=u4Ia_YRUCy0
- http://www.t4.ie/Professional_Development/RD8_Technology/Robotics/A%20Guideline%20to%20using%20%20Pic%20Logicator.pdf
- http://www.economatics-education.co.uk/secondary/education/90,94,0/1536/PIC-Logicator_Version_2.htm
- http://www.bbc.co.uk/science/robots/techlab/sub_selector.shtml