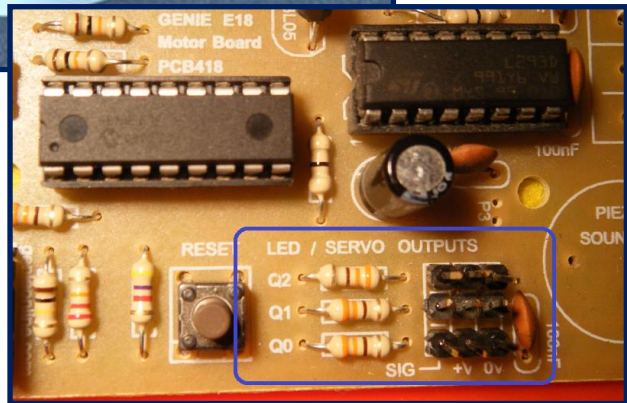
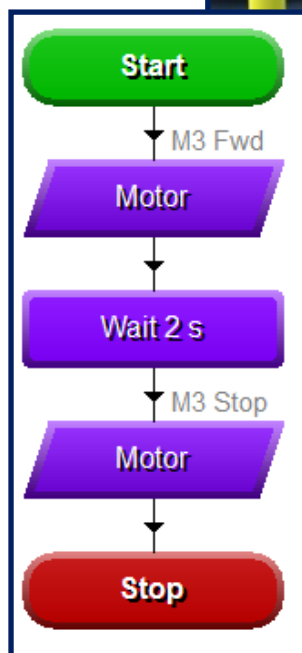


Programming PIC Microchips



Programming the PIC microcontroller using Genie Programming Editor

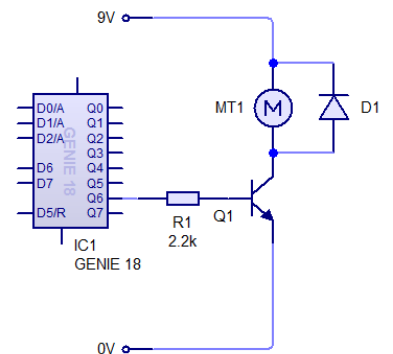


DC motor control:

DC motors are the most common way of providing movement to projects. They can be connected to gears, wheels and mechanism, and are a perfect way of adding mobility. DC motors are generally connected to a power supply by two wires and the direction of rotation is determined by the order of the connections i.e. by reversing the connections (motor supply voltage) the direction of a DC motor can be changed. DC motors, however, do not offer accurate, controlled motion.



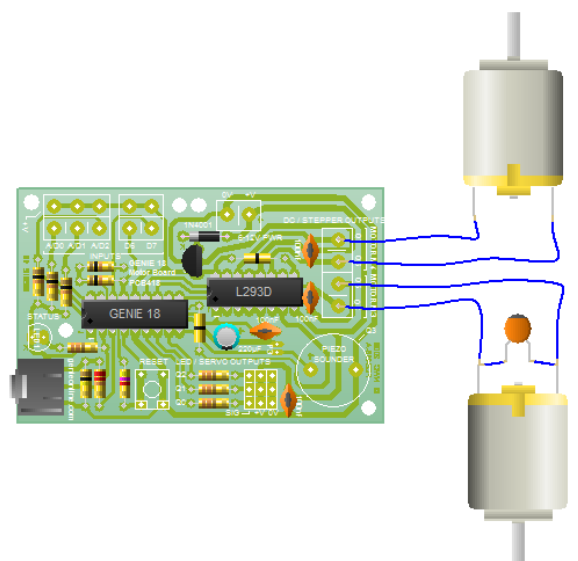
DC motors draw a large amount of current and as a result cannot be wired straight from a control system such as a PIC. DC motors require a driver circuit in order to amplify the output signal and to provide a higher current.



A transistor acting as a current amplifier

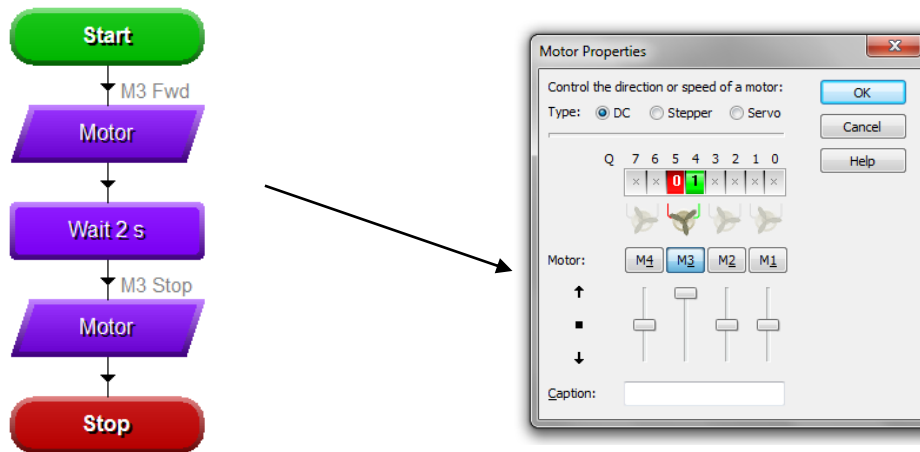
The Genie E18 Motor Control Board can control two DC motors simultaneously. The board includes an L293D chip which allows forward and reverse control of both motors. The motors are connected to the board as show below.

Although the board includes two capacitors connected across each motor, it may be necessary to add extra capacitors. An extra capacitor can be soldered across the motor connection as shown. This helps to protect the board from high levels of voltage spike, created by the motors.



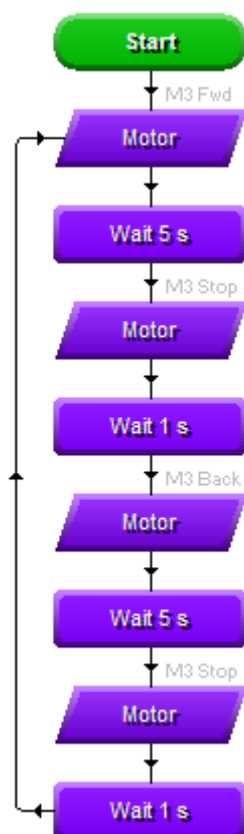
Prog. 1 - Controlling a DC motor.

Program 1 will turn on a motor and allow the motor to run for 2 seconds before turning the motor off.



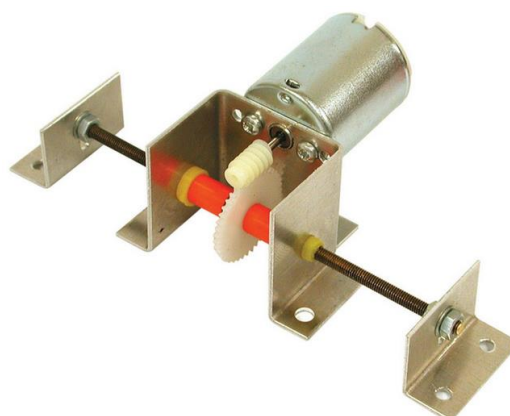
Double click on the motor command to choose how to control the motor, in this case motor M3. Click on the DC option to control a normal DC motor. Each DC motor is controlled by a paired set of outputs. Adjust the slider to change the direction of the motor. The degree to which the slider is moved also controls the speed of the motor. When the slider is at the top or bottom, the motor will turn at its maximum speed in the chosen direction.

Prog. 2 – Forward and Reverse control of a DC Motor



Program 2 provides forward and reverse control of motor M3. The limits travelled by the linear actuator in this case are controlled by including a wait command after each motor command, open loop system.

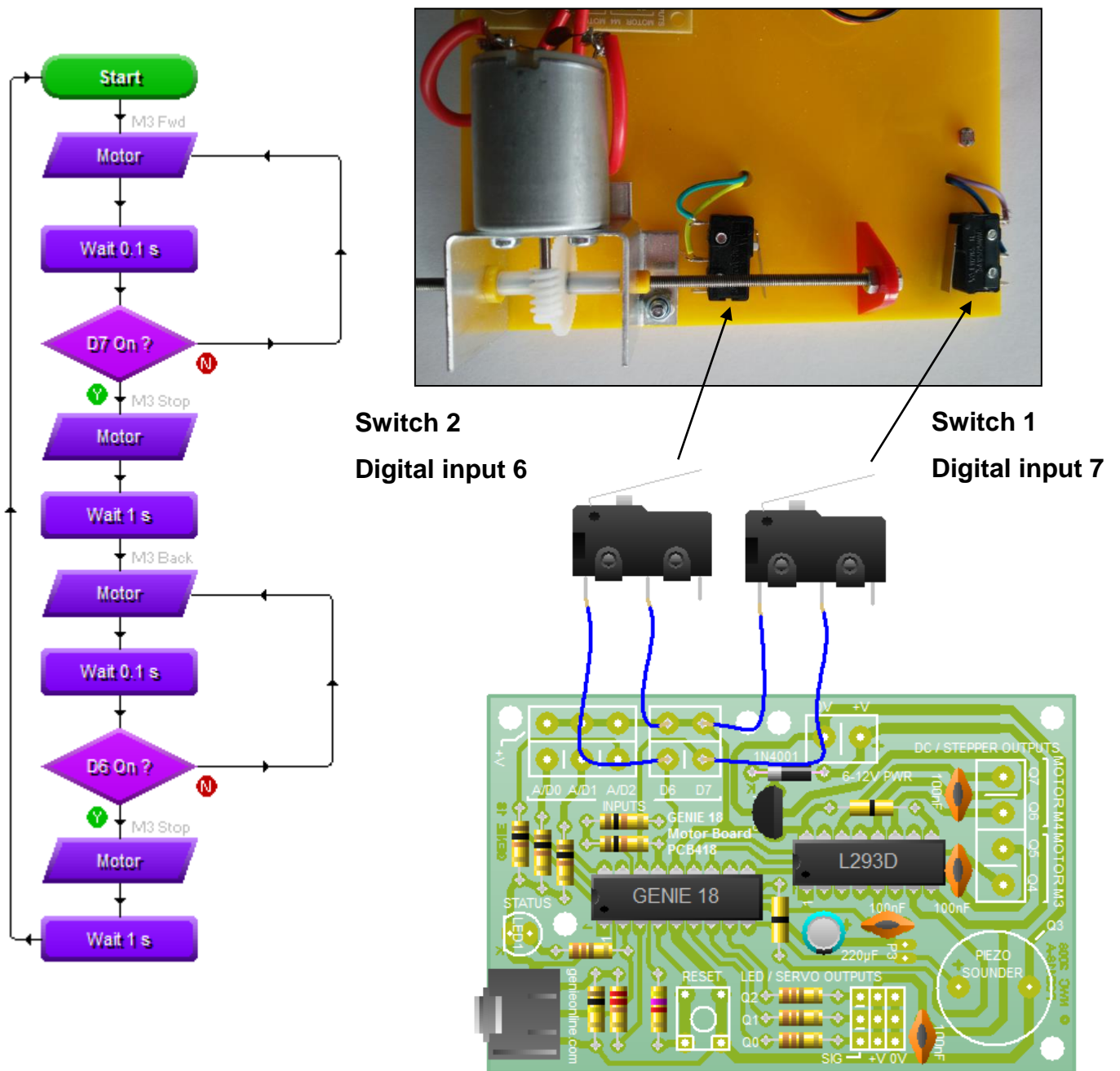
Is this an accurate way of limiting the distance travelled?



Prog. 3 – Using limits switches to provide feedback (closed loop system)

To achieve better accuracy of the distance travelled, limit switches are used to provide feedback. Digital commands are included to test for digital inputs provided by the two limit switches.

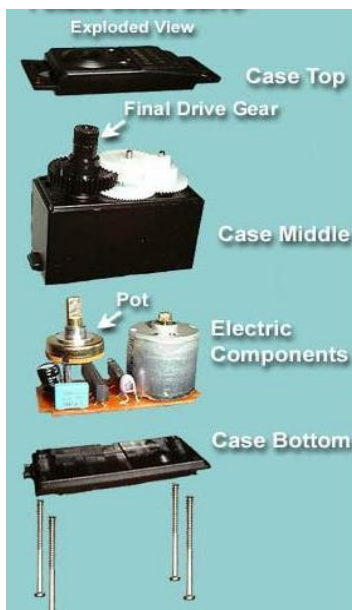
Additional wait commands of 0.1s are included to reduce the possibility of electrical interference.



Servo Motors for use with Genie Software

Introduction to the Servo Motor:

The servo motor offers the smoothest and greatest control, better than the stepper and DC motors. They can be told to rotate to a specific point, making them ideal for applications that require precise movement. The rotation of a servo is limited; most rotate from 90° to 180° though some can complete a full rotation. They cannot rotate continually due to their structure so are unsuitable for driving wheels, but their torque and control make them suitable for powering robotic arms and such.



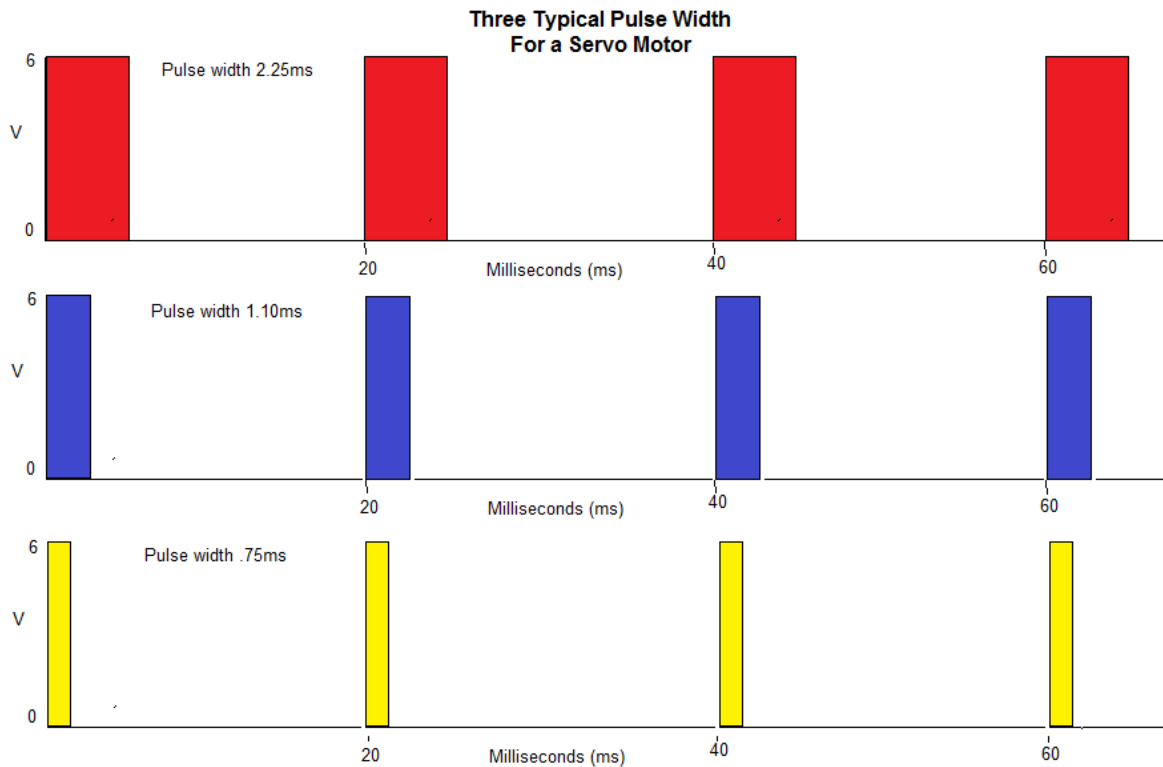
Servo motors are made up of a number of components that are housed neatly in a self-contained unit. They contain a motor, gearbox and driver controller, allowing them to be controlled directly from a microcontroller.

Servos have three wires connected to them. Two wires generally red and black (though this can vary in some motors) provide 6v & 0v. The third cable must feed directly from the PIC output pin and cannot be received via a transistor chip; this is to ensure that it is a frequency that the servo receives, not a straight current.

The potentiometer is connected to the motor through a gear train. A signal is given to the motor to rotate to a given position, as the motor turns it also moves the potentiometer causing its resistance to change. This resistance is monitored by the control circuit ensuring that the servo is in the correct position. If the servo is accidentally knocked out of place the control circuit will know and correct its positioning. This is an example of closed loop control.

Servos are positioned using a technique called Pulse Width Modulation. Servos generally require a pulse width of 0.75ms to 2.25ms every 20ms. This pulse width is

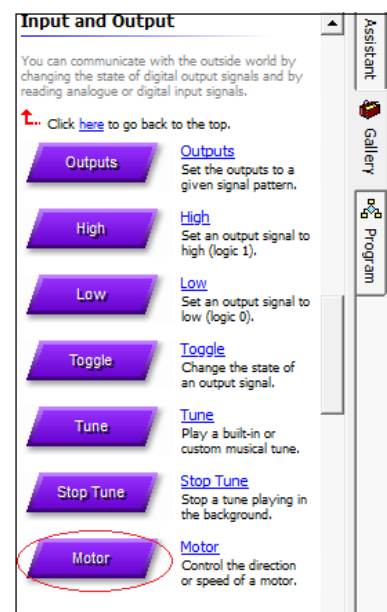
constantly repeated every 20ms, if the pulse is lost the servo can lose its position. Pulse widths of 0.75ms to 2.25ms take the servo from 0° to 150°, any pulse widths between these can be selected to achieve the required position. It is the pulse width that controls the position of the servo, not the number of times it is repeated each second.



Genie uses a value range between 75- 225 in its servo motor command to send the corresponding pulse width between 0.75ms and 2.25ms. A variety of servo sizes are available that offer a range of torques and motion.

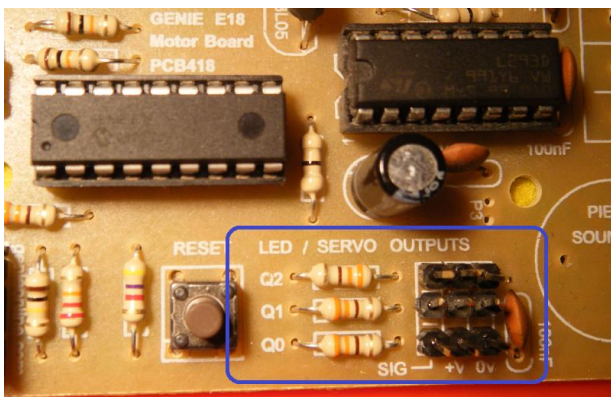
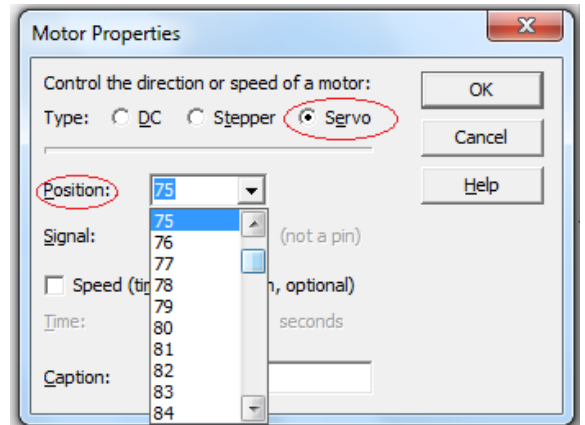
Programming the Servo Motor

In previous software programs a servo motor had its own designated Servo command block that could be dropped into the flowchart. Genie uses a generic *Motor* Command Block that is selected for the programming of DC, Servo and Stepper Motors. This can be found in the Gallery under the heading *Inputs and Outputs*.

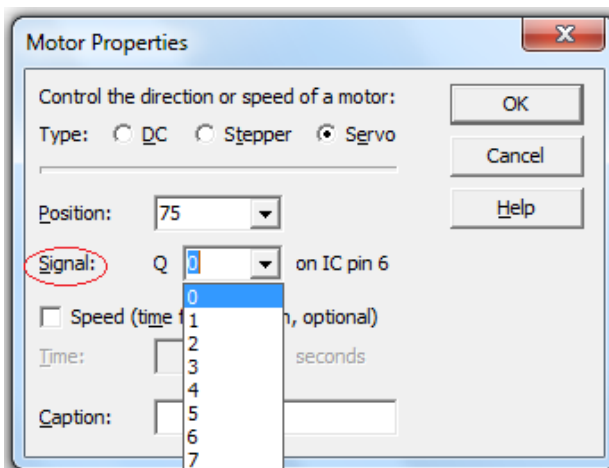


Double click on the Motor Command block and select *Servo* under types of motor.

To control a servo a number of parameters must be outlined. The motor needs to be told its position from 0° to 150° (or the max range of a specific motor). This is achieved by selecting a position value between 75 & 225 as discussed previously.

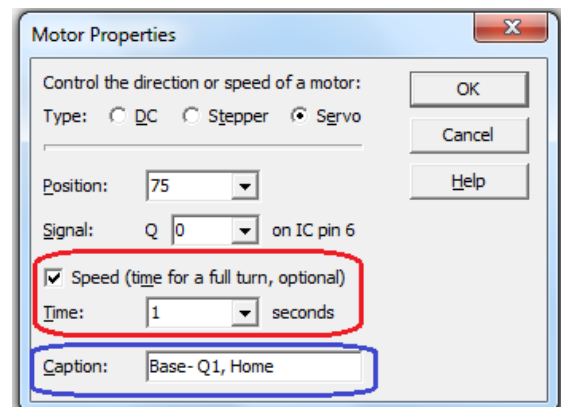


The servo motor has three wires, 0v, V+ and a signal wire that delivers the pulse directly from the Genie chip to the servo. On the Genie E18 Motor Board there are three designated servo outputs incorporated into the board- Q0, Q1, Q2.



The correct output can be selected by clicking on the down arrow adjacent to the *Signal* window. There are 8 outputs available for selection but only three are suitable for use with a servo on the Genie Motor Board.

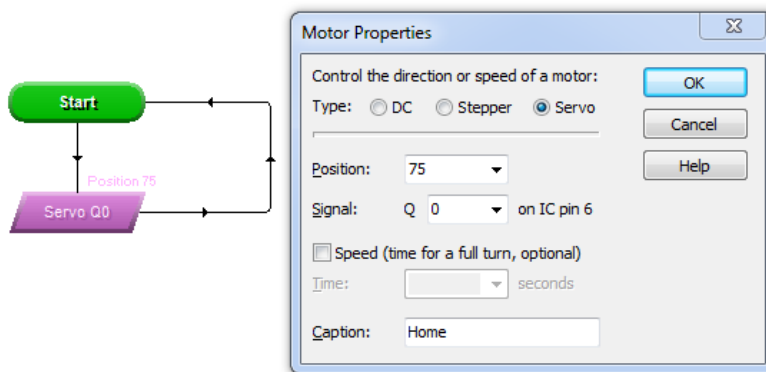
There are two optional parameters left in the Motor Properties window, a Speed and Caption option. The Speed option is extremely useful if the rotation speed of the servo needs to be slowed to give a



more prolonged action. It helps to prevent a sudden jerk movement from different positions. In this example the servo is given 1sec to move into position.

Final a caption can be added to the Command Block for clarity.

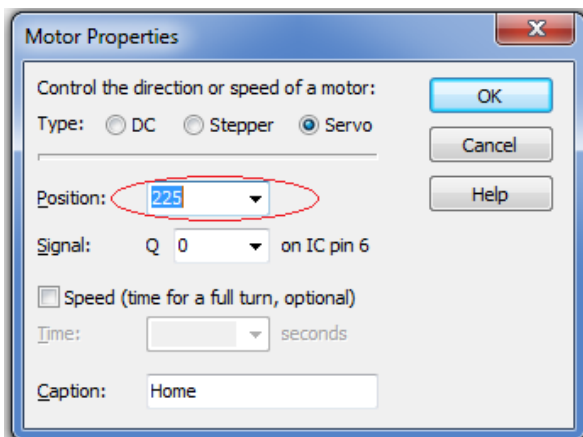
Program 1: Home Position (Barrier Closed)



It is important to be aware of the servo motors actual 0° (position 75) for practical movement of a designated project. In this instance the 0° position is the car park barrier in the closed position.

This program will take the servo to 0° (Home) and keep it there indefinitely. This will allow the user time to manually adjust the barrier arm, having it parallel with the ground (barrier closed).

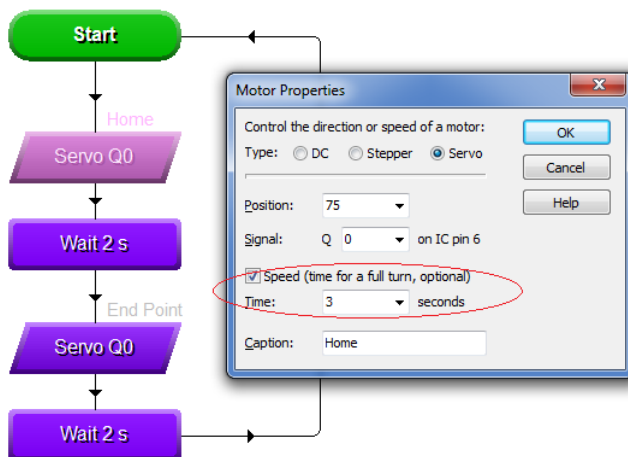
Program 2: End Position



As discussed above the servo can move through 150°. This end position is achieved by selecting Position 225 (75 + 150 = 225).

Change the Motor Block to 225 and download to take servo to that position. How can opening the barrier (vertical) be achieved?

Program 3: Range of Motion with Speed Control



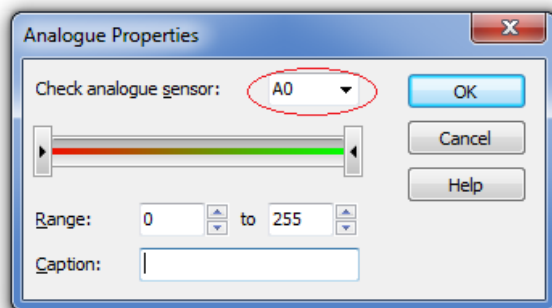
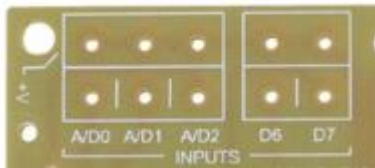
This program loops the servo between its Home 0° and End 225° positions, pausing for 2 second between each movement.

It is possible to slow the servo’s movement between each position by ticking the *Speed* box. *Note- the time selected is always for a full*

rotation (150°) of the motor and not the time to go a desired movement of < 150°.

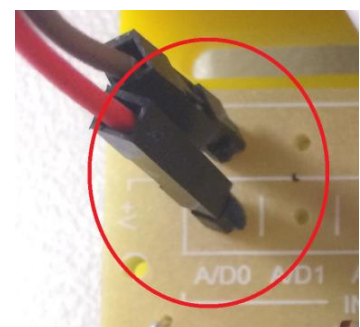
Program 4: Barrier Control using an Analogue sensor (LDR)

Temperature and light are an example of analogue conditions. Genie converts the varying input levels to a scale from 0-255. The Genie 18 Motor board has three analogue inputs A0, A1, A2 allowing for up to three different sensors. It is also worth noting that these can be used as digital inputs (D0, D1, D2) if required. However D6/D7 cannot be used as analogue inputs.



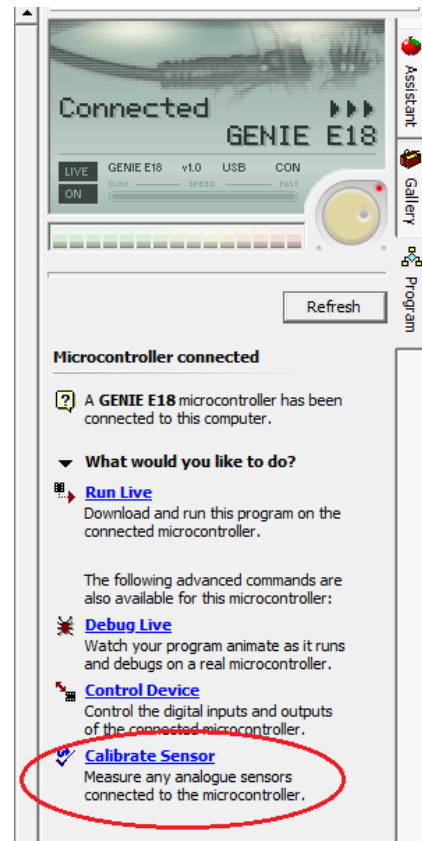
The Analogue block can be found in the Gallery under Flow Control.

The LDR for the barrier is connected to A0; this sensor must be selected from the drop down menu in the Analogue block.

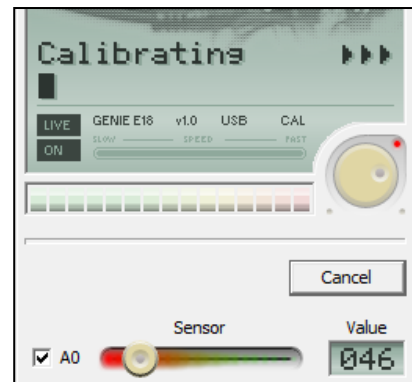


This program should open the barrier when the LDR is covered (signalling a car wishing to enter) and close again once the LDR is uncovered (car has driven into car park). To decide on a suitable light level it is necessary to view the actual live readings of the LDR in its current setting.

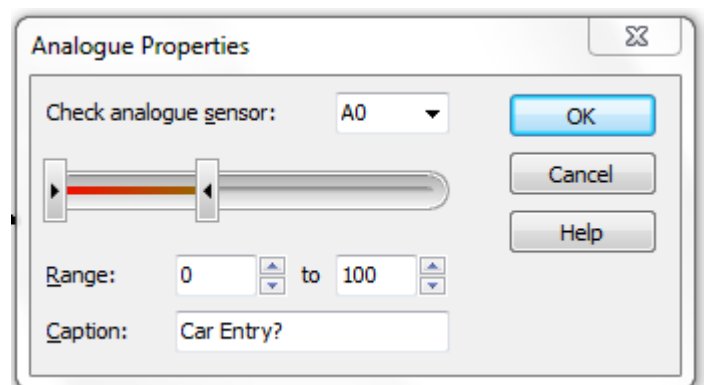
Ensure that the device is turned on and connected via the Genie cable. Select *Calibrate Sensor* under the *Program* Tab on the right-hand side of the screen. This will give a live readout of the three analogue sensors, A0, A1, A2.



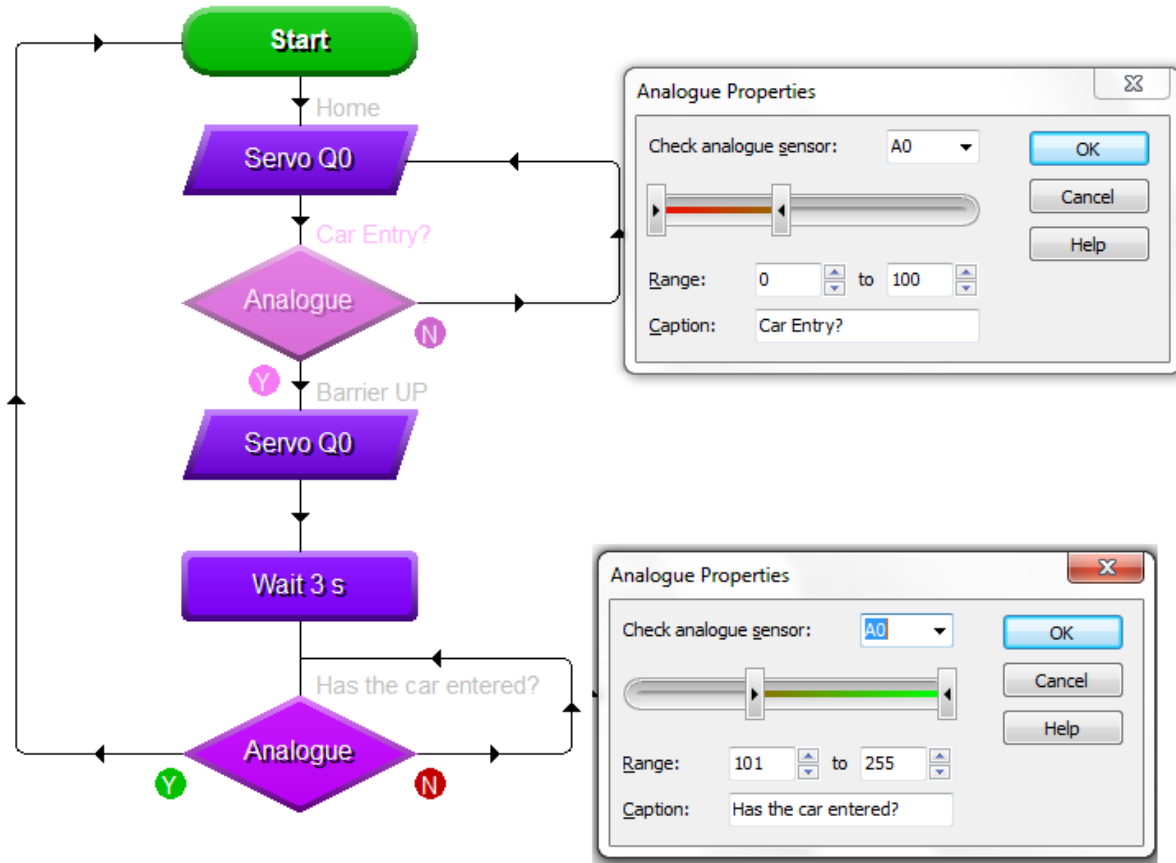
The sample levels here show an uncovered (bright) reading of 206 and a covered (dark) reading of 46.



The Analogue block can now be set to a suitable range. A reading of between 0 and 100 will signal that the LDR is covered and this will trigger the opening of the barrier.

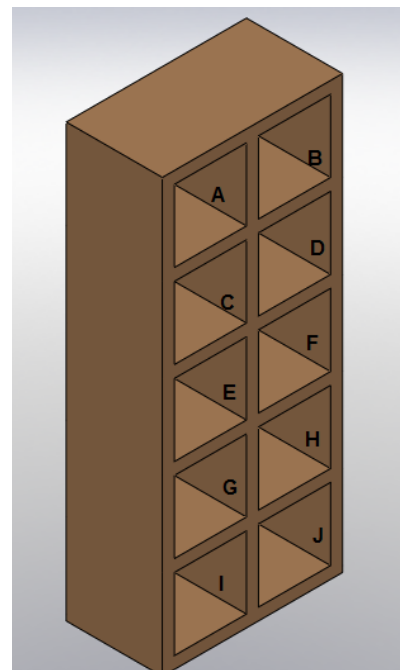


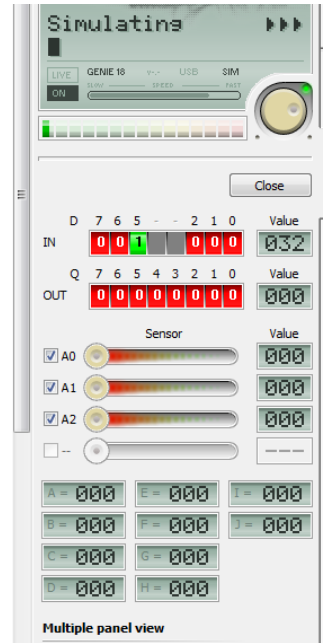
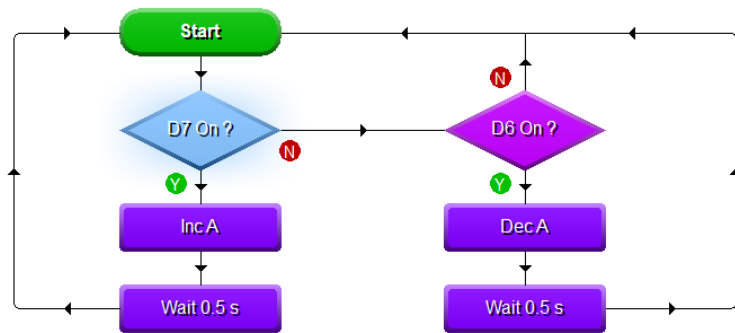
Complete the flowchart as shown below.



Counting using Variables

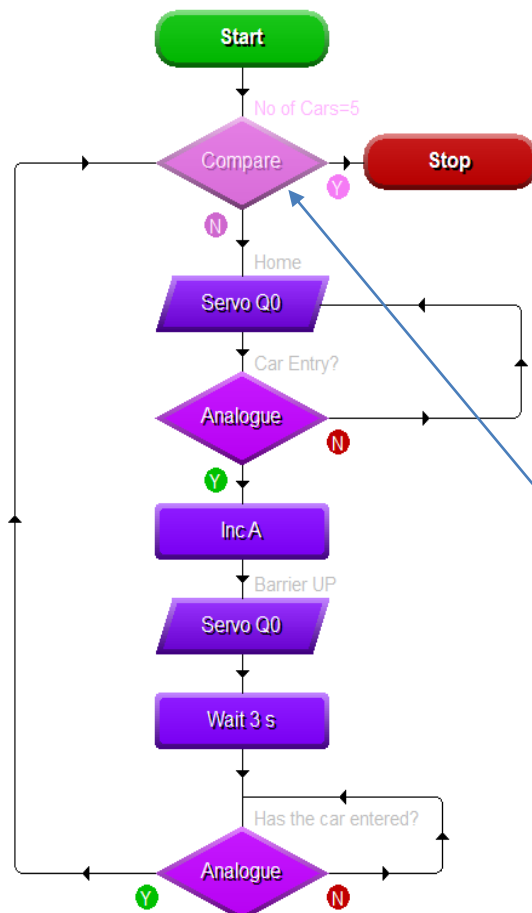
Genie software can store numbers in certain locations, similar to a letter square in an office. There are 10 variables A-J available to store information. During a program a variable can be given a value (Expression), have this value added to (*Inc*) or taken from (*Dec*). Physically this is like adding marbles into the letter square, adding more or taking them away. When required the number of marbles can be counted and the final number used.



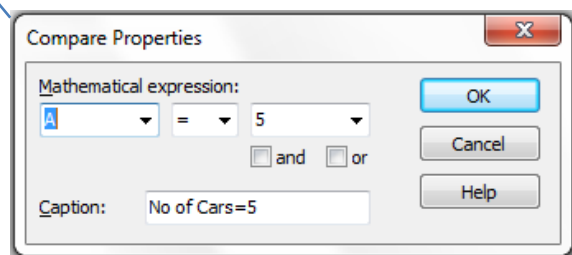


By creating the simple flowchart above and simulating it on Genie Programming Editor the adding and subtracting of these variables can be observed.

Program 5: Car park- Counting up to 5 cars

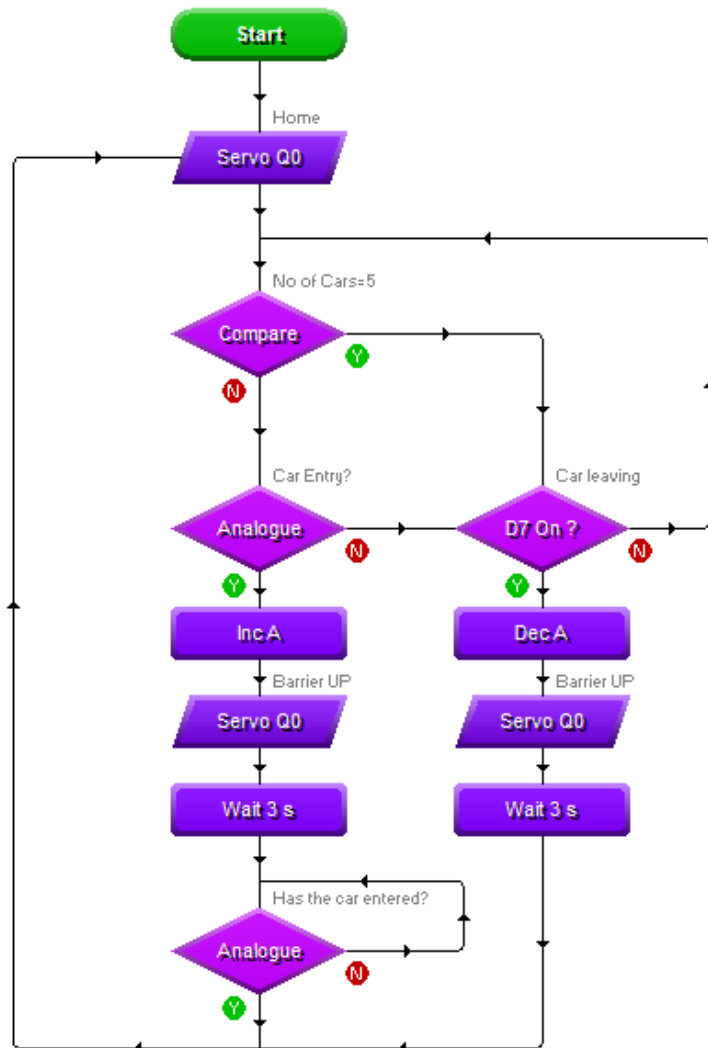


Edit the previous LDR flowchart to add in these extra blocks. The *Inc* block adds in a car each time the barrier rises and the *Compare* block is used to check the number of cars that have entered. The variable A is used to store the information. Any number of cars can enter up to 255 (the largest number that Genie can count to).



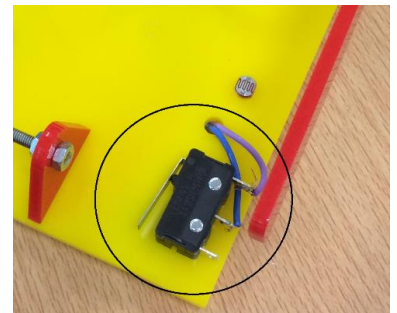
Once the car park reaches 5 cars the program will end.

Program 6: Car park- Counting up and down



Modern car parks monitor the daily flow of traffic in and out. Once the car park reaches its limit, only vehicles are permitted to leave, until there is space available to allow more in.

For a car to leave the flowchart opposite, limit switch



D7 must be pressed. This directs the flow through the *Dec* block and decreases the value of variable A by 1.

If the number of cars reaches 5, the compare block diverts the flow away from the analogue car entry block,

only allowing the option for a car to leave. Once a car leaves and the compare block sees that the number is below 5 it will allow for a car to enter again. This program will run continuously.

