Motor Bee

Motor Control for the PC

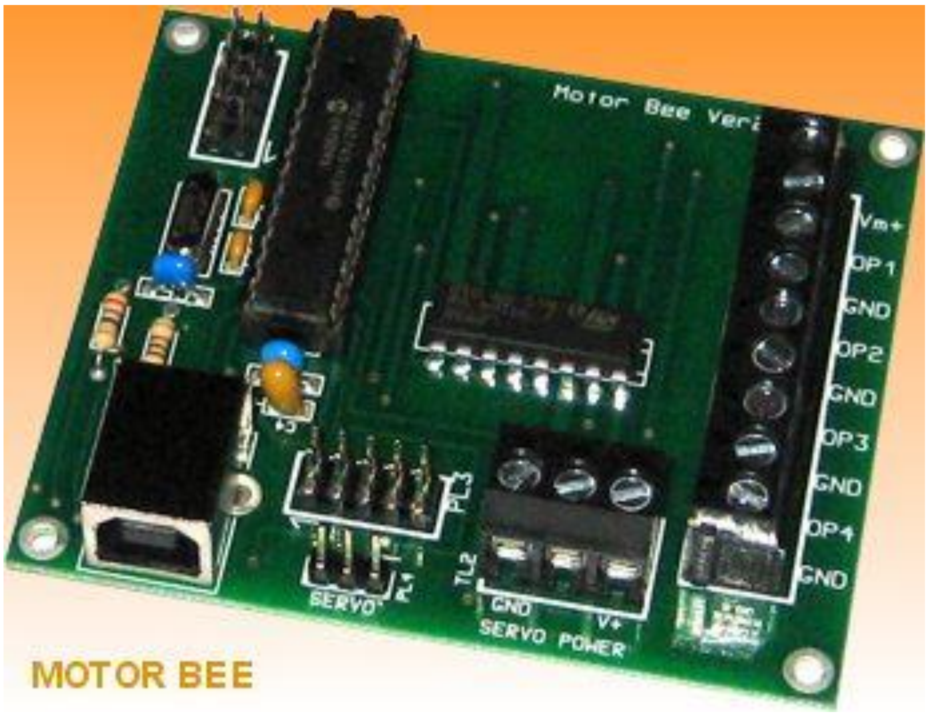
Installation and Users Manual V2.5
(Including Motor-Way Software)

Available exclusively from
PC Control Ltd.
www.pc-control.co.uk

Contents

1. Introduction

The Motor-BEE is a versatile USB adaptor, which allows the PC user to explore the world of real time control and automation. It is a tool, which is attractive to both the novice and experienced user.



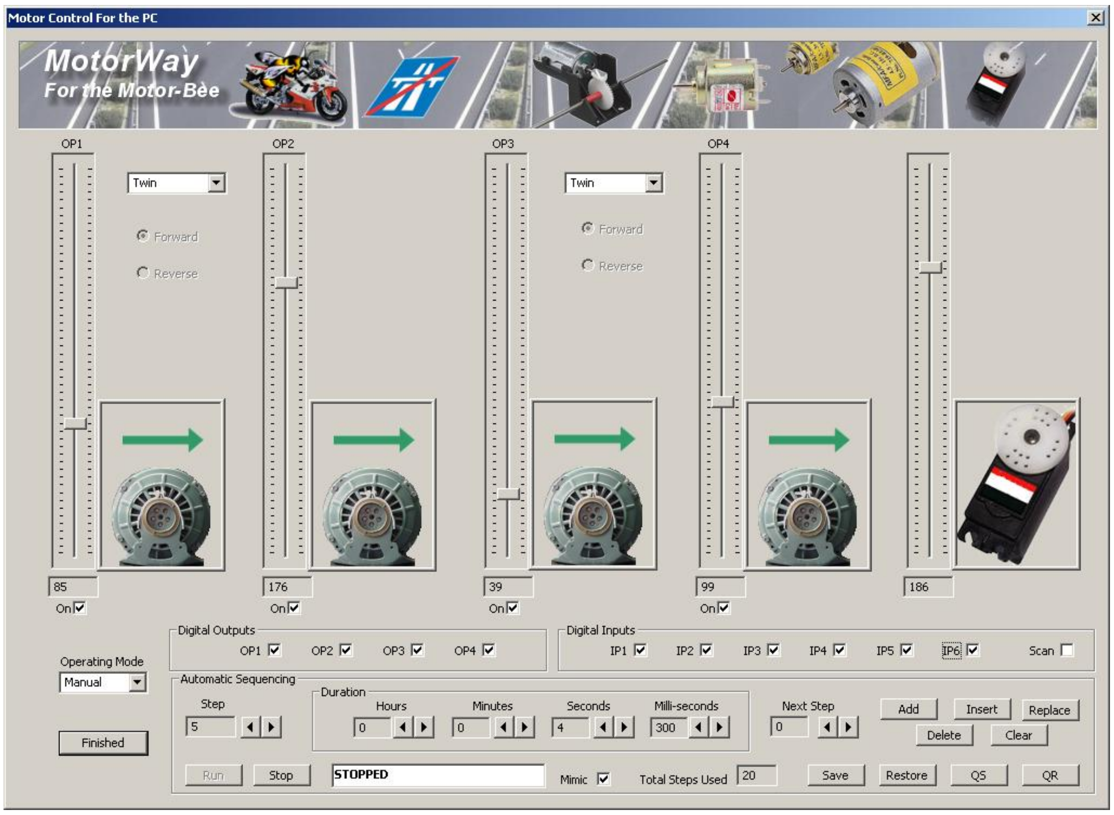
For the absolute beginner it can be used straight from the box as a flexible controller for a wide range of motion projects. The beginner can take advantage of the ease of connectivity of USB, making connection to the PC simple. Screw terminals mean that external devices to be controlled can be attached without any need for soldering. The included “Motor-Way” application software allows the beginner to “drive” and evaluate motors and servos quickly and easily using the manual speed /direction/position controls. Once a manual system is proven to work correctly it is a simple step to automate this using the sequence programming facilities of “Motor-Way” to quickly create control sequences without any need for prior programming knowledge or PLC techniques.

For the intermediate/advanced user a DLL is provided to allow the programmer to construct their own software applications to take advantage of the Motor-BEE hardware without having to know the details of USB communication protocols etc..

2 Hardware Installation

Simply connect the Motor-BEE to any available USB port (*This will require a standard USB cable*). Windows operating system will automatically detect and install the appropriate device drivers. The Motor-BEE is regarded by Windows as a standard HID (Human Interface Device) which makes it very easy to install as the drivers for these are already part of windows.

3. Motor-Way Application Software



OP1: 85, On ✓, Twin, [✓ Forward] Reverse
OP2: 176, On ✓
OP3: 39, On ✓, Twin, [✓ Forward] Reverse
OP4: 99, On ✓
186

Digital Outputs: OP1 ✓ OP2 ✓ OP3 ✓ OP4 ✓
Digital Inputs: IP1 ✓ IP2 ✓ IP3 ✓ IP4 ✓ IP5 ✓ IP6 ✓ Scan ☐

Operating Mode: Manual

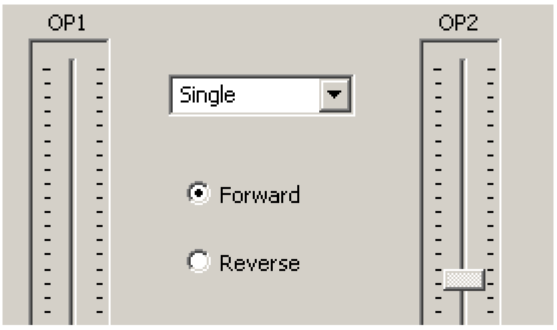
Step: 5
Hours: 0 Minutes: 0 Seconds: 4 Milli-seconds: 300
Next Step: 0
STOPPED
Mimic ✓
Total Steps Used: 20

3.1 Overview

When the Motor-Way application is started, click on the "Run" menu option to initiate the control dialog box (shown below). The first time MotorWay is run the customary disclaimer agreement will ask for your agreement. Click "I Agree" to never see it again and to start MotorWay.

Motor-Way allows the user to directly control the speed and direction of attached motors, the absolute position of a servo and the state of 4 general purpose digital outputs. It also provides a status monitor of 6 digital inputs. Two distinct modes of operation are available, namely Manual and Automatic. In manual mode each of the attached motors and/or servo can be directly controlled by using the available sliders (for speed and position) and tick boxes for on/off control. In automatic mode these configurations of speed and position can be combined into a series of "steps" which have a duration and sequence order. This allows a series of pre-programmed motions to take place performing the required automation.

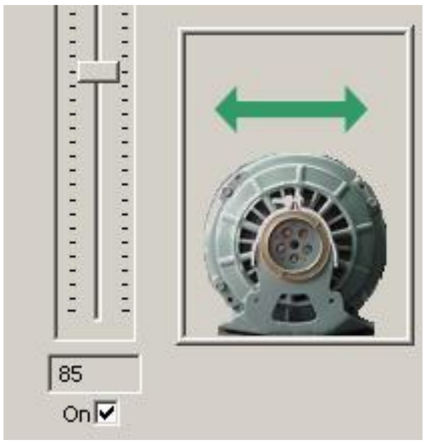
3.2 Manual Operation



Once your external motors/servo are attached to MotorBee you can check correct operation by driving each of them manually. Manual mode is selected using the drop down menu box on the screen. You should then select the configuration of your attached motors using the

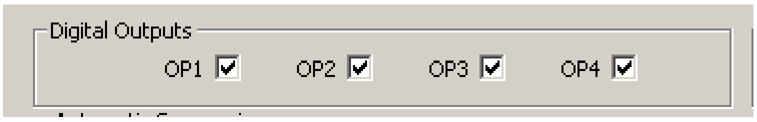
Twin/Single options on each pair of outputs. I.e. outputs 1 and 2 on the MotorBee corrspond to the two leftmost vertical slider controls on the screen. If you have one motor attached to these outputs, to be used for bi-directional control then choose the "Single" option from the drop down menu located between the two slider controls. This will confirm the choice of a single motor with forward and reverse control by the displayed motor image for that pair of outputs. If, on the other hand, you have a separate motor attached to each of outputs 1 and 2 and therefore only require forward direction control on each, then select the "Twin" option for the same drop down menu. This will also confirm selection with suitable motor images. These choices should also be made in a similar manner for outputs 3 and 4 according to your configuration of attached motors. It should be noted that even if you forget to select "single" and connect a single motor between a pair of ouputs, then your system will still work correctly as long as you only have one output of the pair on at any one time. This is , in fact, all that the choice of "single" ensures happens.

There is no need to make any configuration choices for the servo control, as there is only one servo output of the standard type.



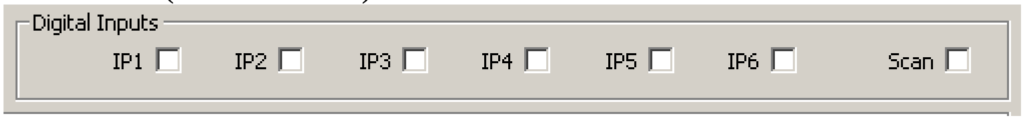
Once these configuration choices have been made you can directly "drive" your motors/servo. Where you have "Twin" selected, the motors attached can be turned on/off using the tick box at the foot of the corresponding slider control. The slider control provides speed control of that motor. Using the mouse pointer you can alter this setting, very simply, by "left click and drag" the slider up or down. It is graduated in a range of 1-250. This corresponds to a speed of 0 to 100% of the motors maximum speed; (for those who are more curious, it actually corresponds to the mark to space ratio of the applied pulse width modulation signal which is the technique used to control the speed of the motors. i.e. a value of 50 gives a mark to space ratio of 50 high to 200 low). The current position of the slider (and therefore the corresponding speed) is also shown numerically at the foot of the slider.

Where a single motor is connected to a pair of outputs and the "single" option is selected for that pair, you will have the additional option of direction control. Simply click the forward or reverse selector to change direction. Note that when direction is changed the corresponding slider for that direction now controls the speed. This allows individual control of speed in both directions. The speed control not appropriate to that direction is "greyed out". (i.e. non-functional), to make this clear.

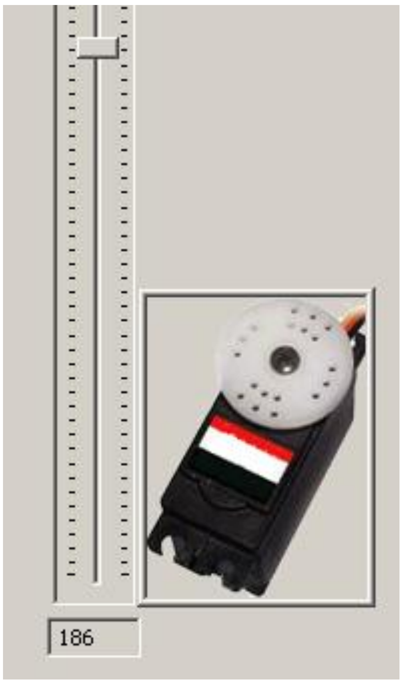


The 4 digital outputs on the MotorBee

may be “toggled” on and off by clicking on the tick boxes in the digital outputs section. While these are being written to the MotorBee the 6 digital inputs are also read and updated in the “Digital Inputs” section. For testing you can also choose to update the digital inputs continuously by ticking the “scan” tick box. It is not recommended to keep scanning inputs while you are running a control sequence in automatic mode(discussed later).



Although the digital inputs are not directly used by Motor-Way software they can play an important role in any control system created by your own software (eg you may use an input to determine when some motion has reached a limit of travel or have external switch controls to start and stop sequences etc…). For this reason they are included on the main MotorWay screen to help your evaluation of this aspect of the MotorBee hardware.

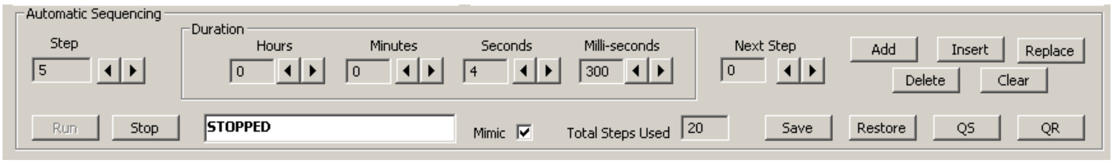


The servo control system is designed to control the vast majority of standard servos available on the domestic, hobbyist and light industrial market. It provides a standard control signal output which , when connected to servo, causes the servo to move to that absolute position. The slider control on the right side of the screen performs this control. As users familiar with servos will know, the control signal used to specify the absolute position required is a pulse whose width varies between 1 and 2 milliseconds in duration. This pulse is repeated every 20ms. A pulse of width 1.5ms corresponds to a servo position of exactly midway in its range of travel. This also corresponds to a slider position half way up in Motor-Way. Again the range of the slider is 1 to 255 so midway is 128. As the slider is moved away from this position the servo will move to its new position in response. Although the vast majority of servos adopt the 1.5ms standard for midway, the figures used for either extremity of travel is less well defined. In some servos the extreme left position is specified by 1.25ms and the extreme right by 1.75ms. In others these are represented by 1.0ms and 2.0ms respectively. It is for this reason that no judgement has been made in Motor-Way as to the absolute position of the attached servo, or its range of travel, when moving the slider. All that can be said is that the control signal will vary from 1.0ms to 2.0ms over the full range of slider movement. As a general rule of thumb a typical servo will turn through 180 degrees over it’s full range of travel and since our slider has a range of 1 to 255 with perhaps the middle 80% of travel within the servo range, you can assume a rough correspondence of 1 degree of servo movement for each slider change of 1. (128 corresponding to the 90 degrees position)

When you choose your servo for this application you should avoid driving it beyond it’s recommended range of travel. This can sometimes cause the servo to overheat and may cause damage. In practice, though, they are usually quite tolerant of short durations beyond their specified limits.

3.3 Automatic Operation

In manual mode above we have seen how each of the controls can be used to turn motors on and off and vary their speed and direction and also how to position a servo. This combination of on/off speeds and positions we call a configuration. To make automation more useful we need to be able to “program” a set of these configurations by specifying their duration and the order in which they occur. This is the function of automatic mode.



This mode provides the user with a means to edit, save, restore and run a sequence of “configurations” that will be used to directly control the devices attached to the Motor-Bee adaptor board. This is known as Directed Sequencer Control. The sequence is made up of up to 10000 individual configurations or “steps”. The number of the current step being edited is always shown in the “Step” box. Each step has all of the following elements

- A duration (from 0.1sec up to 99hours)
- A speed for each of the 4 motors
- A direction for each motor (when only 2 motors used)
- On/Off for each motor
- Position of the servo (to approx 1 degree resolution)
- A pattern of outputs (on or off for each of the 4 digital outputs)
- The number of the next step to be executed

To specify a “step” simply set the motor speeds, directions etc using the same controls already described in “Manual” mode. Then specify a duration and “Next Step” in the automatic mode area of the screen.

The duration is specified in Hours, Minutes, Seconds and milliseconds. The overall resolution of the time interval is 0.1 seconds. This provides a realistic level of control for devices in the “real world” of automation and robotics.

Choosing a value of 0 for the “Next Step” causes straight sequential execution of steps. Choosing any other value will cause the sequence execution order to “Jump” to that step. The most common use of this facility is to specify the next step of the last in the sequence to be the first. This will produce a constantly looping control sequence.

Once you are happy with the configuration of the step it may be “added” to the current sequence by pressing the “ADD” button. In doing this you will notice that the “Total Steps” box will increase by 1 and the current step box will indicate the number of the last step in the current sequence. You can continue adding steps until your sequence is ready (up to 10,000 steps). When you are ready, you can then “Run” the sequence by pressing the “RUN” button. This will cause each step to be executed in turn starting with step 1. If the “Mimic” box is ticked (discussed later) you will see all of the controls change to reflect the setting stored in that step while they are being applied to the external devices. You can “STOP” the sequence at any time by pressing the “STOP” button. You should note that stopping a sequence will leave all external devices running according to the setting applied during the step in which the stop was pressed. However, if, necessary, you can then always go back to manual mode and turn off any motors left running etc… Pressing the “RUN” button again will re-start execution at step1 of the programmed sequence. Note that the last step in a sequence will always cause the sequence to stop even if the "next" box is set to another location (eg to loop back). To avoid this add a dummy step after the one in which you wish to loop back.

To view the steps already programmed into the sequence, use the arrow buttons adjacent to the "STEP" box. Once a step is displayed it can be edited very simply by making the desired alterations to the slider controls etc… followed by pressing the "REPLACE" button.

To insert a step into the programmed sequence simply press the "INSERT" button. This will cause all steps equal to, or higher numbered than the current one to be incremented by 1 and the step number displayed to be replaced with the one just configured. Similarly pressing the "DELETE" button removes the current step from the sequence and renumbers all higher steps by –1.

Once you have created a sequence of programmed steps as described above, you can save your sequence using the "SAVE" button. This will allow you to store the file according to your name choice and destination directory. This can subsequently be restored by pressing the "RESTORE" button and choosing the appropriate saved file.

As a convenience, there is a "Quick Save" and "Quick Restore" function denoted by the "QS" and "QR" buttons respectively. These will simply save (or restore) the current sequence to a file called "c:\quicksave.mtb". When using "QuickSave" any previous contents of the file will be overwritten without prompting.

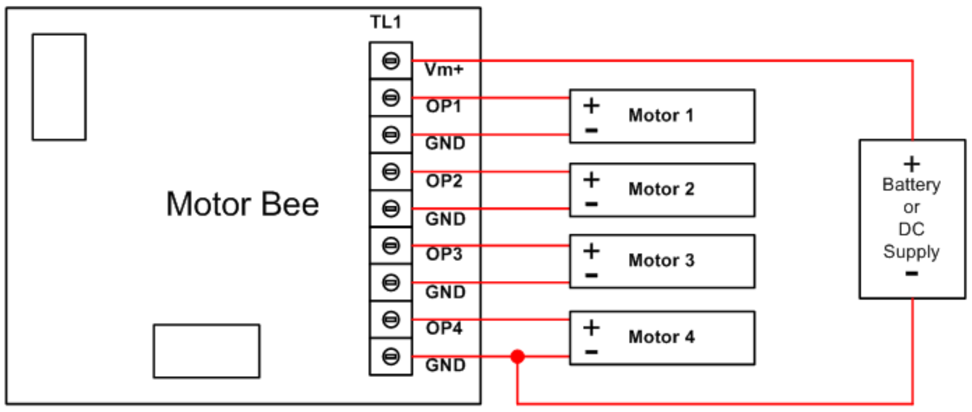
Mimic Control. When the "Mimic" box is ticked, the slider controls will animate according to the current step being executed. This is useful for testing and evaluating your control system. However, you should consider turning off this facility if you are using a very fast changing sequence (eg 100ms steps) or your PC is relatively slow. The overhead in updating the display may affect the accuracy of timing, especially on some older PC's. With the mimic turned off the sequence will still operate when "RUN" is pressed but you will not see the changes on the screen, only on your attached motors etc…

4. Connecting Devices to Motor-Bee

There are three separate connecting areas on the MotorBee. The screw terminals TL1 and the two 10 way headers PL2 and PL3. The screw terminals are used for connecting DC motors, PL2 is used for connecting the digital inputs and PL3 for connecting a servo and the digital outputs. It is not necessary for all outputs to be connected. If you simply want to drive a single motor then that is all you need to connect.

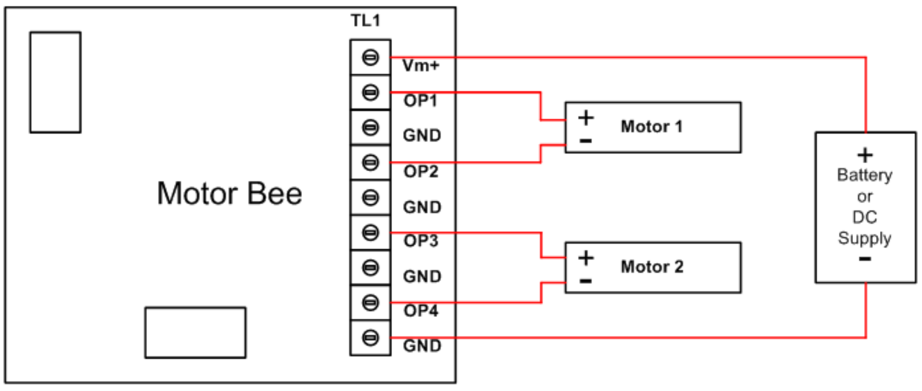
4.1 Connecting Motors

MotorBee can drive up to 4 DC motors with independent on/off and speed control for each. The motor outputs on screw terminals TL1 are labelled OP1 – OP4. Connecting a DC motor requires its positive terminal connecting to an output (eg OP1) and its negative terminal connecting to a ground (GND). To allow dc motors of different voltage ratings to be used, a separate motor supply terminal is provided (Vm+). If, for example, you are using 6v DC motors you would connect the positive terminal of the 6v supply to terminal TL1 (Vm+) and the negative supply to any of the GND terminals on TL1.
WARNING: Never connect an external motor supply of less than 5v. This will cause reverse biasing of the main L293D driver chip on the MotorBee which can damage one or more components on the board. If using very small motors labelled as requiring less than 5v (eg 3v) it is still recommended to use a minimum 5v supply and use the power regulation of the MotorBee to restrict the power delivered to the small motor.



Connecting Four Independent Motors

If you would like to drive a DC motor in both directions then it should be connected between two output terminals. For example connect the positive motor terminal to OP1 and the negative motor terminal to OP2. The supply pin Vm+ and a GND should still be connected to an external supply as described above.

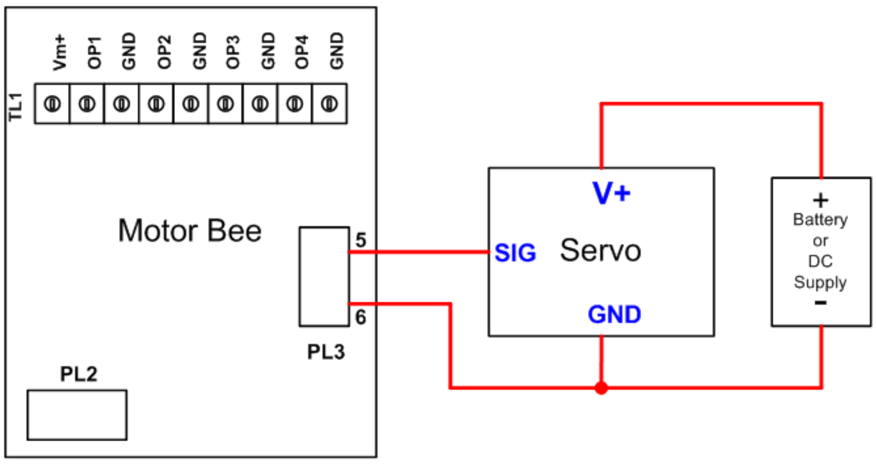


Connecting Two Reversible Motors

Care should be taken in the choice of DC motors to be attached. It is especially important to check the power requirements. i.e. voltage and current. The most important specifications to remember is to limit the motor supply voltage below 24v maximum, limit the continuous current per motor to 600mA (1.2A peak) and the power dissipation to 4 watts. The full specifications for the motor driver IC (L293D), used on the MotorBee, can be found in its data sheet provided on the installation CD. If in any doubt check the temperature of the motor driver chip in the centre of the MotorBee. If it is too hot to comfortably touch then you are probably supplying too much power.

4.2 Connecting a Servo

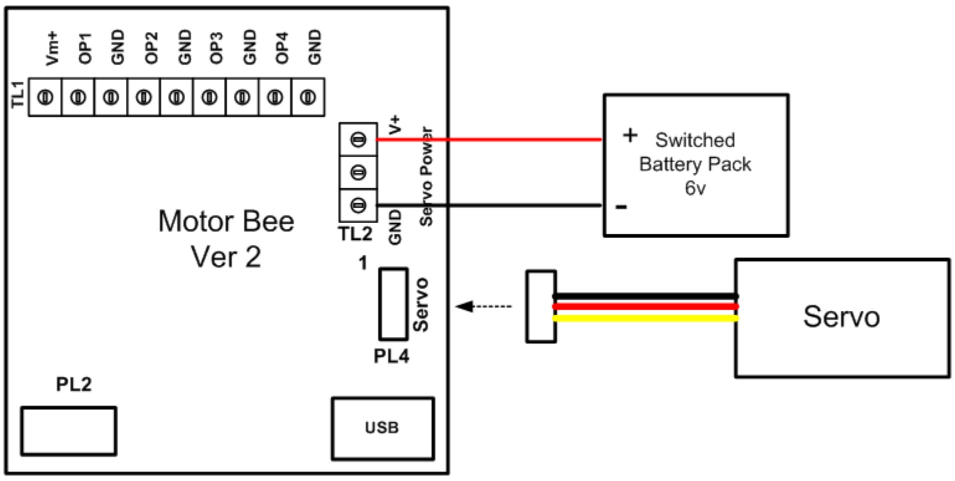
MotorBee can drive a standard servo from the PL3 connector. The power to the servo is supplied externally to the MotorBee so there is no power limitation considerations in this case. The control signal specifying the required position of the servo is supplied on pin 6 of PL3 with the corresponding GND on any of the even numbered pins. A 10 ribbon cable socket or equivalent should be used to make the connection. According to your choice of servo it will require an external voltage supply. This is not connected through MotorBee but the GND on PL3 and the GND on the servo supply should be connected together. This GND may be called “0v” or perhaps even “V-“ or even “Negative Terminal” on the servo power supply.



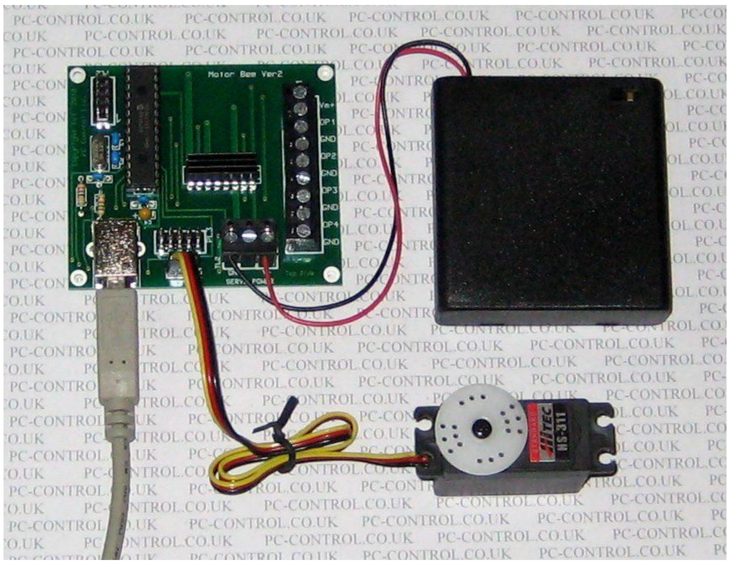
Connecting a Servo

Since servo positioning is sensitive to electrical noise spikes, care should be taken when connecting and running motors and servo at the same time. A common sign of electrical interference is a slight "twitching" of the servo after it has moved to it's specified position. If this happens you should make sure your GND connections are secure and also fit a suitable capacitor across the terminals of any motors connected. On some of the low power motors we have found a 47uF capacitor solves any such problems. (always ensure the voltage rating of the capacitor is suitable for the application).

Note: There are two versions of the MotorBee board. They function identically but look slightly different. The difference is in the servo connection options. On the standard Motor bee a servo is connected to PL3 as shown above. In the MotorBee Ver2 the above connection to PL3 can still be made but there are two other connectors to provide an alternative connection. They are TL2 (three way screw terminals) and PL4 (three way header). When these are present they offer a convenient way to connect a servo (with standard connector only) and its supply separately as shown below. See document "Servo Pack Details" on the installation CD for more details on how to connect a servo using this option.

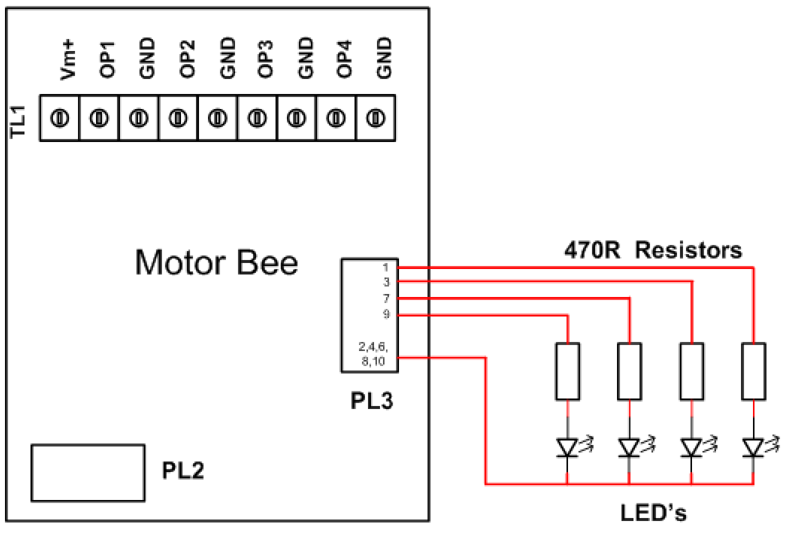


Connecting the Servo



4.3 Connecting Digital Outputs

PL3 is also used for connecting the 4 digital outputs available on MotorBee. Pins 1,3,7 and 9 are used for outputs 1 – 4 respectively.

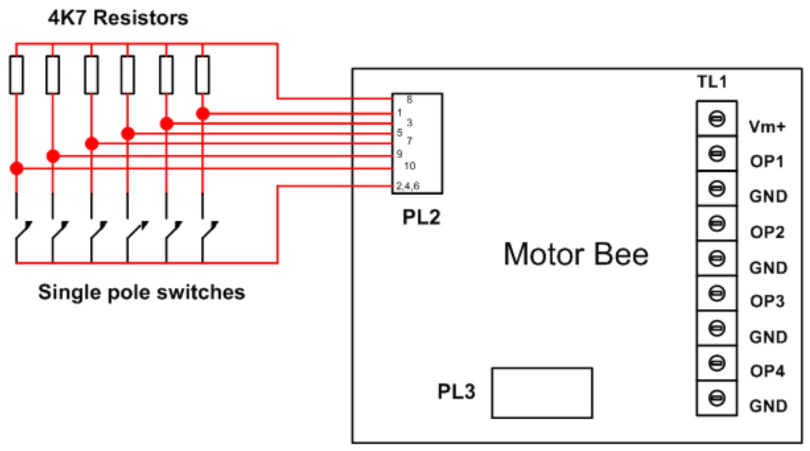


Connecting Digital Outputs

The outputs are standard digital logic providing 5 volts when on and 0 volts when off. They are not designed for driving heavy current devices and have absolute limits of 25mA sink or source. One most common use for these outputs is driving indicator LED's. If this is your intention always use the high efficiency ones which give maximum brightness for minimum current. (NB: a resistor is also required in the LED circuit). They are also commonly used to drive relays via suitable interface devices such as the ubiquitous ULN2003 transistor array.

4.4 Connecting Digital Inputs

MotorBee has 6 digital inputs. These are connected to PL2 via a standard 10 way header requiring a 10 way ribbon cable socket or equivalent for connection. The voltage input range for these inputs is 0 – 5v DC. As a convenience to help minimise your external circuitry a 5v DC supply is provided on this header. This is intended to make it simple to connect external switches. It should be noted that this 5v supply is derived from the USB connection and must only be used for very light loads i.e. less than 50mA maximum.



Connecting Switches to the Digital Inputs

Typically these inputs are used to add limit switches to automated devices so that the motor being controlled activates the limit switch when, for example, the moving device has reached the end of it's range of travel. This would allow the control software to turn off (or perhaps reverse) the motor at the correct point. Some applications have used these for light sensor inputs for similar reasons.

5 Connector Pinouts

Motor Connections: Pinout of the Screw Terminals(TL1)

Pin	Signal description
1	Vm+ Positive Motor Supply
2	OP1
3	GND
4	OP2
5	GND
6	OP3
7	GND
8	OP4
9	GND

Digital Inputs: Pinout of 10 way header, PL2

Pin	Signal description
1	Input 6
2	GND
3	Input 5
4	GND
5	Input 4
6	GND
7	Input 3
8	+5v (27R current limit) 50mA max
9	Input 2
10	Input 1

Servo and Digital Outputs: Pinout of 10 way header, PL3

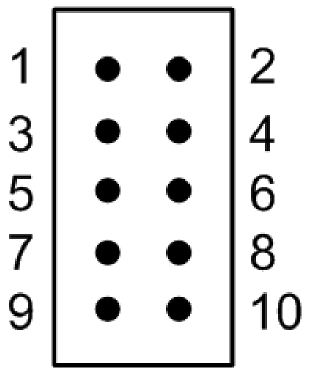
Pin	Signal description
1	Output 1
2	GND
3	Output 2
4	GND
5	Servo Signal
6	GND
7	Output 3
8	GND
9	Output 4
10	GND

Alternative Servo Connector (Motorbee Ver2 only):
Pinout of 3 way header, PL4

Pin	Signal description
1	GND
2	V+ (positive terminal of servo supply)
3	Servo Signal

Alternative Servo Power Connector (Motorbee Ver2 only):
Pinout of 3 way screw terminals, TL2

Pin	Signal description
1	V+ (servo positive supply)
2	Not Connected
3	GND (servo supply ground (0v)



PL2 - PL3
Pin Arrangement

6. Writing your own software for Motor-Bee

To use Motor-Bee straight from the box does not require any programming other than entering the step details into Motor-Way. However, if you prefer to design your own software then the following information will be of use.

Provided with Motor-Bee is a DLL (dynamic link library) called “mtb.dll”. This encapsulates the functions used by Motor-Way in communicating with Motor-Bee across the USB interface into a few simple functions easily understood and used in custom software. Although the DLL was written in ‘C’ it can be used (called) by programs written in a number of popular languages including BASIC (visual BASIC etc..).

6.1 Programming Motor-Bee in Visual Basic

Although Motor-Bee comes with it's own software (Motor-Way) to allow the beginner to start using it in home automation projects very quickly, it also comes with a DLL interface to allow the intermediate and advanced user to write their own programs for it. The DLL provides a general purpose interface that greatly simplifies the task of writing programs for a USB device. It can be tricky manipulating the USB comms into sending and receiving messages to and from a device which can easily be plugged and unplugged at any time. The DLL eliminates all of these headaches by simplifying the task into three library functions.

InitMotoBee( ), SetMotors ( ) and Digital_IO( )

InitMotoBee() is called somewhere near the start of your program and takes care of all of the USB comms initialisation and prepares the Motor-Bee for receiving messages.
SetMotors() can be called at any time during your program to configure the speed and direction of the motors and the position of the servo.
Digital_IO() can also be called at any time to both set the digital outputs and read the digital inputs.

InitMotoBee Function

```
InitMotoBee()
```

This function has no parameters and returns a boolean value of TRUE if the call was successful , FALSE otherwise. It should be called prior to the use of any other function listed below. Its main role is to initialise scan the USB devices attached to the computer, identify the MotorBee and set up a comms channel to it. This is all transparent to the callers program which simply makes the call somewhere near the start of the program.

SetMotors Function

```
SetMotors(   ByVal op1 As Integer,
             ByVal speed1 As Integer,
             ByVal op2 As Integer,
             ByVal speed2 As Integer,
             ByVal op3 As Integer,
             ByVal speed3 As Integer,
             ByVal op4As Integer,
             ByVal speed4 As Integer,
             ByVal servo As Integer      )
```

The SetMotors function has 9 parameters which are all integers passed by value.

- op1 – op4 correspond to the on/off state of the 4 motors. ‘1’ is on, ‘0’ is off.
- speed1 – speed4 correspond to the speed of each motor. This has a range of 0 to 254 and corresponds to 0 to 100% of maximum speed at the applied voltage.
- When using a motor in bi-directional mode it will be connected to two of the outputs (eg OP1 and OP2). To operate the motor in the forward direction OP1 should be "On" and set to the desired speed. OP2 also needs to be set "Off" (its speed does not matter). To operate the motor in reverse direction OP1 should be "Off" (speed does not matter) and OP2 "On" with the desired speed.
- servo corresponds to the position of the servo device. This has a range of 0 to 255 with 128 corresponding to the “neutral” or midway position.

Digital_IO Function

```
Digital_IO(   ByRef inputs As Integer,
              ByVal outputs As Integer)
              As Boolean
```

The Digital_IO function has two parameters. The inputs parameter is passed by reference so that the results of reading the digital inputs on the MotorBee board can be returned in this variable. For example, if the 6 digital inputs were in the following state

1 –on, 2-off, 3-off, 4-on, 5 off, 6 off

the value retuned in “inputs” would be 9. i.e. 00001001 in binary. From this you can see that input 1 corresponds to bit0, input 2 is bit 1, etc…

The output parameter is passed by value and corresponds to the on/off state of the 4 digital outputs. Specifically bit 0 corresponds to digital output 1, bit 1 to output2 etc….. As an example the following call would set outputs 1 and 2 on , 3 and 4 off.

Digital_IO (inputs, 3)   since 3 in binary is 00000011

The only other thing that a VB program must do is to declare the functions that it is going to use within the DLL and the name of the DLL itself. This must be done at the start of your program or at least before any references to the three functions are made. The following is a program excerpt showing how this is done...

```
Declare Function InitMotoBee Lib "mtb.dll" () As Boolean

Declare Function SetMotors Lib "mtb.dll" (ByVal on1 As Integer,
                                          ByVal speed1 As Integer,
                                          ByVal on2 As Integer,
                                          ByVal speed2 As Integer,
                                          ByVal on3 As Integer,
                                          ByVal speed3 As Integer,
                                          ByVal on4 As Integer,
                                          ByVal speed4 As Integer,
                                          ByVal servo As Integer)
                                          As Boolean

Declare Function Digital_IO Lib "mtb.dll" (ByRef inputs As Integer,
                                           ByVal outputs As Integer)
                                           As Boolean
```

The first declaration states that the function InitMotoBee has no parameters, is found in mtb.dll and returns a boolean value. The second states that SetMotors has 9 integer parameters passed by value rather than reference, is found in mtb.dll and also returns a boolean value. The third state that Digital_IO has two integer parameters; one passed by reference, one by value, is found in mtb.dll and, again, returns a boolean value. It should be noted that the

Lib "mtb.dll"

lets the program know where to find the mtb.dll file. When written like this it assumes, since there is no path information, that the mtb.dll file can be found in the windows system directory (c:\windows\system32) If you like you can copy the file mtb.dll on the installation disk to the system32 directory and the above statement will work perfectly. Alternatively you can copy the file to some other location and give that location in the declaration as in the example below...

```
Declare Function InitMotoBee Lib "c:\mylibrary\mtb.dll" () As Boolean
```

To speed up your development of software for the Motor-Bee a complete working example is included on the installation CD. It is called VBmotor and creates a very simple form based program that has individual buttons for various functions such as initialising the Motor-Bee and setting configurations for motor speeds etc…. The main (and only) screen is shown below. This has been written using Microsoft Visual Studio .net and the files on the CD contain the full workspace (solution) details, to allow you to immediately open and start editing or running this application.

Even if you don't have Visual Studio, the source code is virtually self explanatory with the most relevant sections being in "Form1.vb" which can even be opened in a simple text editor such as notepad.

6.2 Programming Motor-Bee in Visual C++

Ignoring some of the formalities in the construction of a Visual C++ program for the windows environment the techniques in using “mtb.dll” consists of four main tasks….

Loading the DLL into memory

Before any functions within the DLL can be used it is necessary to instruct windows to load it into memory. This is done by calling the LoadLibrary() function. i.e.

```
……
HINSTANCE BeeHandle;     // declaration of variable to hold the handle to the dll
….
BeeHandle = LoadLibrary(“mtb.dll”);          // load the dll into memory and return handle
```

The declaration of the variable BeeHandle used to store the handle to a DLL , uses a built in type definition which is called HINSTANCE in this particular ‘C’ compiler, but you should use the appropriate one defined in your own compiler for this purpose.

The LoadLibrary() function return a handle to the DLL if the load is successful otherwise NULL. Ideally your own program should check for a NULL returned and give an error message. Make sure the function parameter is the full pathlist to where you copied the bee.dll file from the installation CD.

Get the addresses of the functions within the DLL

Using the DLL handle returned above you can now obtain pointers to the functions within the DLL. Using the following

```
TypeInitMotoBee          InitMotoBee;
TypeSetMotors            SetMotors;
TypeDigital_IO           Digital_IO

……..
InitMotoBee = (Type InitMotoBee)GetProcAddress( BeeHandle, " InitMotoBee");
SetMotors =(Type Digital_IO)GetProcAddress(BeeHandle, " SetMotors");
Digital_IO =(Type Digital_IO)GetProcAddress(BeeHandle, " Digital_IO ");
……………
```

The TypeInitMotobee, TypeSetMotors and TypeDigital_IO type definitions are contained in the header file “mt.h” which is supplied on the installation CD, and defines the correct type of function pointer to reference the DLL function. mt.h should be included in your own source file eg.

```
………..
#include “mt.h”
………..
```

The call to GetProcAddress() returns a pointer to this function if found within the DLL otherwise NULL. Once the functions pointers have been obtained in this way the internal functions within the DLL are simply accessed like ordinary function calls e.g.

```
………..
InitMotoBee ();
SetMotors(1, 123, 0, 0, 1, 80, 1, 50, 128);
Digital_IO(&inputs, 0x06);
…………
```

Initialising The DLL

Once the addresses of the DLL functions are obtained as above the remainng functions required to use them are very simple. The first step is to initialise the DLL using….

```
int status;
……..
status = InitMotoBee();
```

Your program should check to see if a value of zero has been returned by InitMotoBee (). Any other value indicates an error. E.g. Motor-Bee not connected etc…

Using the Functions

The usage of the SetMotors() and Digital_IO() functions are identical to that described in the “programming with VB section above.

Although this only gives a glimpse of the possibilities of writing your own programs, it should be apparent that the use of the DLL functions greatly simplifies this process. It frees the programmer from the task of getting to know the fine details of programming USB interface communications and lets him concentrate on the main function of controlling motors and servos.

7. Minimum PC System Requirements

Motor-Bee and Motor-Way software do not require a high spec PC for correct operation, but the following system is suggested as a sensible minimum

Processor	500MHz Pentium
Memory	64MB
HDD	10MB free space required
Screen Resolution	1024x768 (256 colours)
Interface	One free USB socket (1.0 or 2.0)
Operating System	Windows XP (or later)

WARNING: The Motor-Bee adaptor should not be connected directly to mains voltages under any circumstances.

MotorBee
Regulatory Compliance and Safety Information

Product Name: MotorBee
Part No. BRD004

IMPORTANT PLEASE RETAIN THIS INFORMATION FOR FUTURE REFERENCE

Warnings
- This product should not be connected to mains voltages under any circumstances.
- This product should be placed on a stable, flat, non-conductive surface in use and should not be contacted by conductive items.
- The connection of non CE compliant devices may affect overall compliance or result in damage to the unit and invalidate the warranty.

Instructions for safe use
- To avoid malfunction or damage to your board please observe the following:
- Do not expose it to water, moisture or place on a conductive surface whilst in operation.
- Do not expose it to heat from any source; the MotorBee is designed for reliable operation at normal ambient room temperatures.
- Take care whilst handling to avoid mechanical or electrical damage to the printed circuit board and connectors.
- Avoid handling the board while it is powered. Only handle by the edges to minimize the risk of electrostatic discharge damage.
- All peripherals used with the board should comply with relevant standards for the country of use and be marked accordingly to ensure that safety and performance requirements are met.

Compliance Information
- The board complies with the relevant provisions of the RoHS Directive for the European Union.

WEEE Directive Statement for the European Union
- In common with all Electronic and Electrical products the board should not be disposed of in household waste. Alternative arrangements may apply in other jurisdictions.

EMC Compliance Statements
European Union (EU) Electromagnetic Compatibility Directive Compliance Statement
This product is in conformity with the protection requirements of EU Council Directive 2004/108/EC on the approximation of the laws of the Member States relating to electromagnetic compatibility.
Warning: This is equivalent to an EN 55022 Class A product. In a domestic environment this product may cause radio interference in which case the user may be required to take adequate measures.

PC Control Ltd. 18 Beech Close, Desborough, Kettering, Northants NN14 2XQ, UK
www.pc-control.co.uk

Terms of Use for all Goods Supplied

Definitions

‘Supplier’ shall mean PC Control Ltd.

‘Buyer’ shall mean the person, company or any other body that purchases or agrees to purchase Goods.

‘Goods’ shall mean all goods and services which the Buyer agrees to buy from the Supplier including replacements for defective Goods, hardware, documentation and software products licensed for use by the Buyer.

Use of the Goods in any way by the Buyer constitutes acceptance of these terms and conditions.

Terms and Conditions

1. The Goods are intended to be part of the buyer’s own design of apparatus and not a finished product in their own right.
2. The Goods supplied are not to be used in any design where there is a risk, however small, either directly or indirectly, of death or personal injury.
3. The Buyer will be responsible for ensuring the fitness for purpose of the Goods for the Buyer’s application.
4. To the extent permitted by law, the Supplier accepts no liability whatsoever or howsoever arising in respect of loss, damage or expense arising from errors in information or advice provided whether or not due to the Supplier’s negligence or that of its employees, agents or sub-contractors save for any loss or damage arising from death or personal injury.
5. To the extent permitted by law, the Supplier shall not be liable to the Buyer by reason of any representation (unless fraudulent), or any implied warranty, condition or other term, or any duty at common law, or under the express terms of any Contract with the Buyer, for any indirect, special or unforeseen loss or damage (whether for loss of profit or otherwise), costs, expenses or other claims for compensation whatsoever (whether caused by the negligence of the Supplier, its employees or agents or otherwise) which arise out of or in connection with the supply of the Goods or their use or resale by the Buyer.
6. The entire liability of the Supplier under or in connection with the Contract with the Buyer shall not exceed the price of the Goods except as expressly provided in these terms and conditions.
7. These terms are an important part of the full terms and conditions of business as published on the website at www.pc-control.co.uk/general-terms.htm which also apply.

If you cannot agree to the terms and conditions of use of the MotorBee then you should return the MotorBee to the supplier within 7 days of receipt to receive a refund. Your use of the board or the associated software in any way whatsoever will be regarded as an acceptance of these terms and conditions.