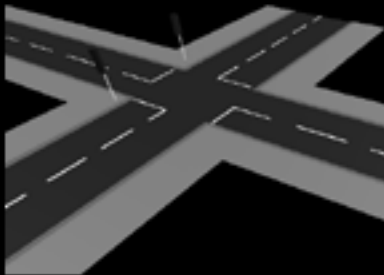
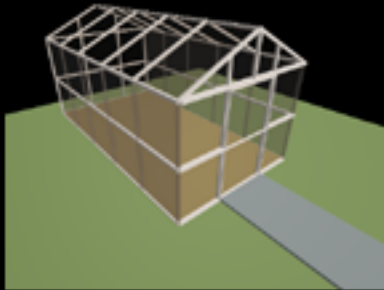


VR Control



User Guide and Language Reference

VR Control combines active components with a program editor. This enables control programs to be written that work directly with the on-screen active components.

A mesh is added as a background to bring the active model to life and give it the appearance of a functional model.

The language used in VR Control is called VIRCO.

This guide contains a commands summary and error message list.

To get started we recommend reading the on-line tutorials which will guide you as to the capabilities of VR Control.

For up to date information on any changes to VR Control load the **Read Me** file.

The **Read Me** file contains important information on installing and running VR Control and the active models.

You will need to be familiar with Windows and programming terminology before using the program.

Install Instructions

Installing VR Control for Windows 98/Me/NT4/XP

From the Control Panel double-click on the Add/Remove Programs icon. In the Add/Remove Programs Properties window, next click onto the Install/Uninstall tab card and then the Install icon. In the Command Line box type D:\Setup and follow the on-screen instructions. VR Control uses DirectX technology, to run VR Control you need DirectX 8.1 or higher installed on each computer running VR Control.

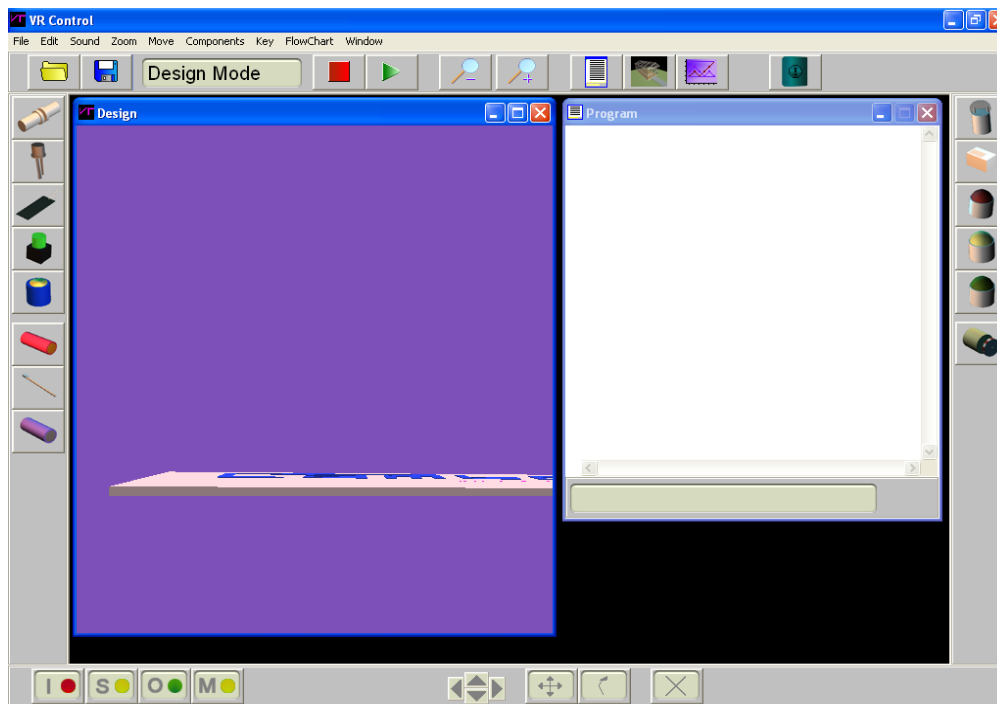
Running VR Control

Click on the VR Control icon in the Start menu.

This opens the Main, Design and Program windows.

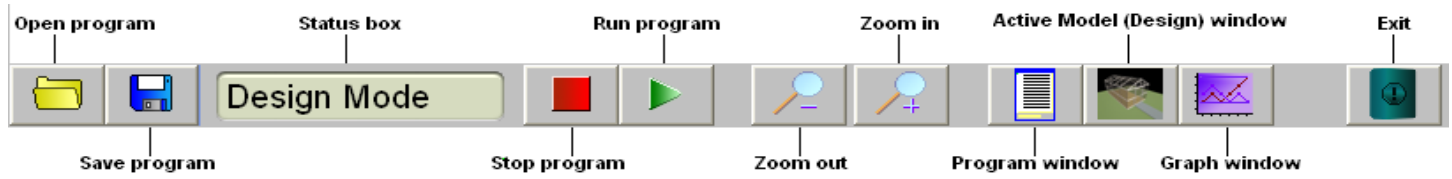
Main Window

The Design and Program windows are contained within the Main window.



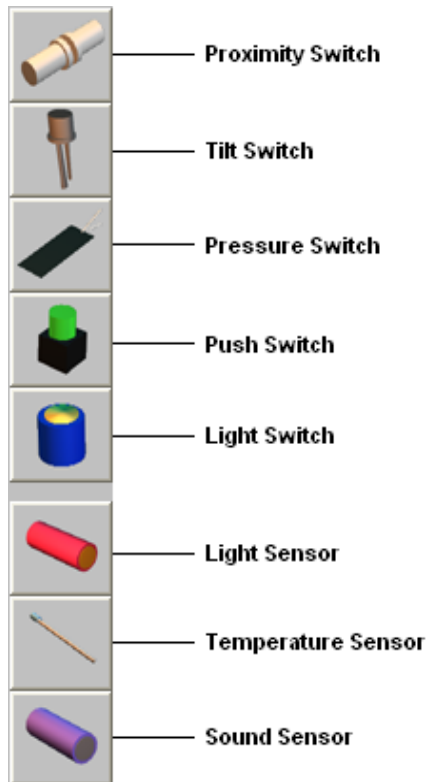
Control Bar

The main program icons are on this bar.



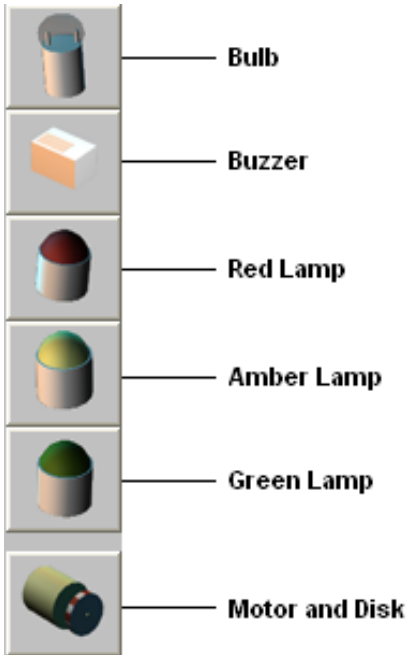
Inputs and Sensors Bar

Input components and Sensor icons are on this bar.



Outputs Bar

Output component icons are on this bar.

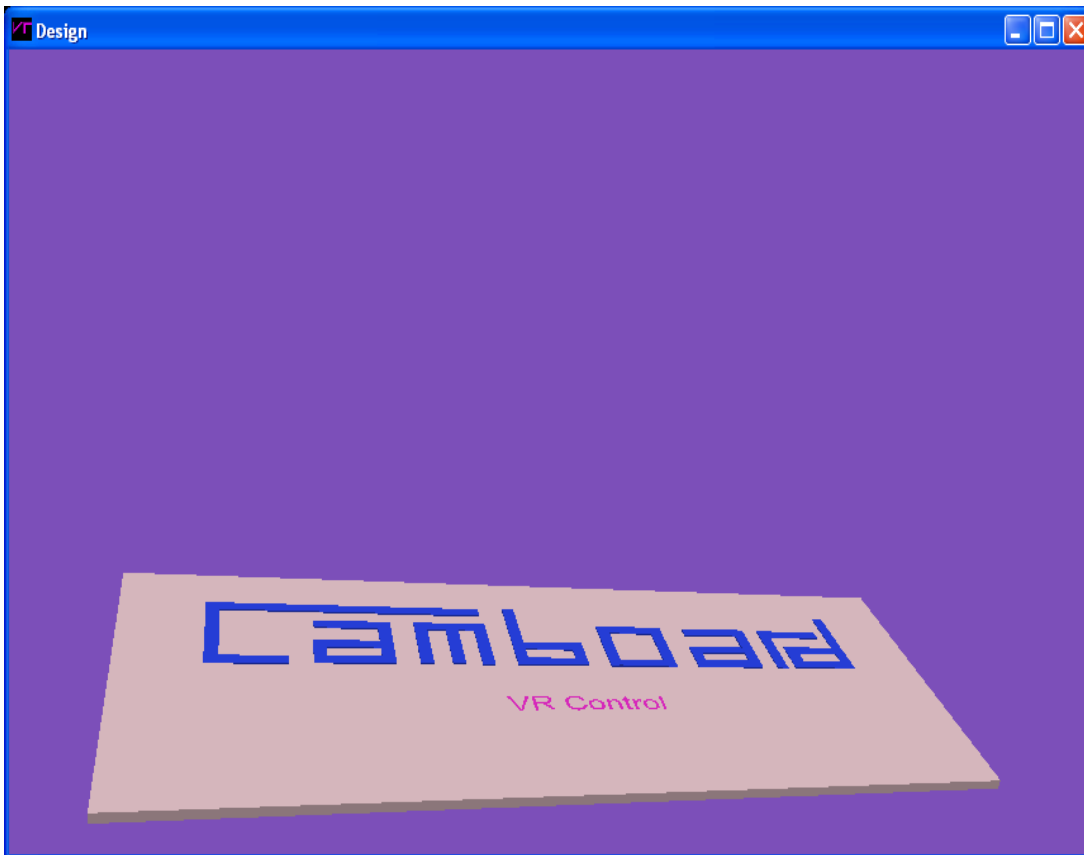


Status Bar

Icons for the Inputs, Sensors, Outputs and Motors status windows.



Active Model (Design) Window



The Design window is where active components are placed.

Designing an active model.

The heart of any active model is the 3D mesh. The 3D mesh is loaded into the Design window from the the Load Mesh menu option. When importing your own meshes its important to consider several factors which will have a bearing of the type of model you can build and simulate with VR Control. When designing your own meshes its important to understand that with some 3D packages features that are not supported by directX. The rule of thumb here is, if the model opens in VR Control and appears the same as your original design, then it should run ok. The best approach is to keep models simple and use as few primitive objects as possible, when designing meshes with a high number of primitive objects its quite possible that some objects may be missing when loaded into VR Control. In this instance design the model so it may be jointed/glued in sections, save each section as an X file. Load each X file using the Load Mesh option and the model will be built. A number of 3D packages are available to design your meshes in, the active models included with VR Control were designed in Caligari TrueSpace software. Keep in mind VR Control doesn't use textures so these will appear as random colour! Once your mesh is designed save it as a DirectX file. These files have the extension of X. Place the file in the main VR Control program directory, Load the mesh into the design window, by using the Load Mesh option. Because of the way VR Control displays meshes there will be differences in the colour and scale of the the mesh. This is due to the default lighting arrangement in VR Control. Light is reflected and absorbed at different angles in VR Control. Once the mesh is loaded into VR Control the task of adding active components can begin.

Connecting active components

Active components are animated which brings life to an active model. Components are selected from the two component bars in the Main window.

Input Components

A number of different input components are available that should cover most needs when designing a model. To connect an active component select one from the left hand panel. Click on the Tilt Switch icon and then click on a model mesh in the design window, clicking on the background will not work!, this connects the Tilt Switch to Input line 1. Clicking on the component will switch it on or off.

Sensors

Three types of sensor are available Temperature, Sound and Light. Click on the Temperature icon and then on a model mesh in the design window. The sensor connects to Sensor line 1.

The temperature value is increased, by clicking over it with the left mouse button and decreased by clicking over it with the right mouse button. The light and sound sensor values are changed in the same way. All sensor readings are expressed in %.

Outputs

A number of Output components are available, click on the Bulb icon and then on a model mesh in the design window to connect the Bulb, the component is connected to Output line 1.

Motor

Click on the Motor icon and then on a model mesh in the design window. The component is connected to Output A.

Inputs, Sensors and Motors line numbers.

VR Control automatically generates line numbers for connected active components. Components are assigned a line in order from 1-6 for Inputs and Outputs. Motors start at A. Sensor numbering is from 1 to 4.

Moving Components

Select the Move option from the Components menu or the Move icon. To move a component click on it with the left mouse button, then holding down the left mouse button drag the component. Moving components in a 3D world can be tricky and aligning the component to precise location requires deselecting the move option and rotating the active model to a different viewing angle. Reselect the component and drag the component. Components can only be moved at design time.

Rotating Components

Select the Rotate option from the Components menu or the Rotate icon.

Disconnecting components

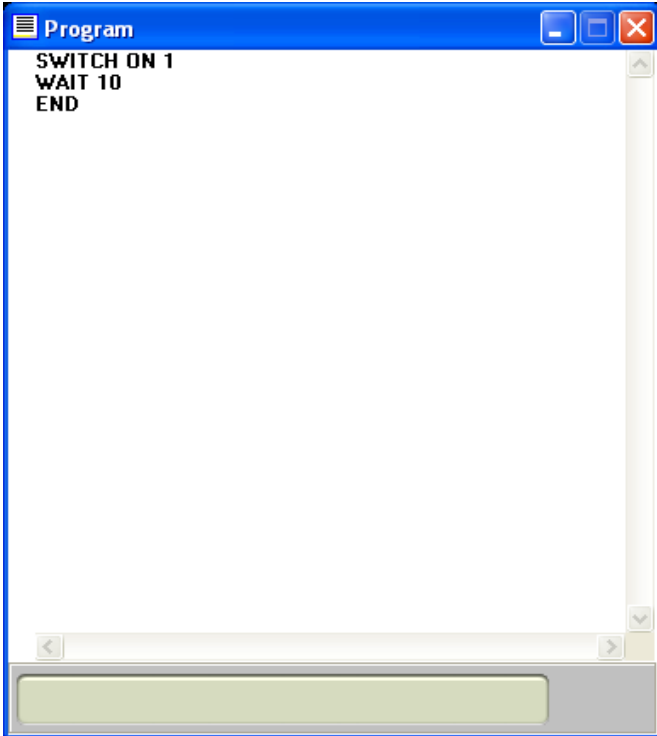
To disconnect and remove a component click on the DELETE component icon. With the right button off your mouse click on the component you wish to delete. A message box opens which asks if you would like to delete the component. Click on the Yes button. This disconnects the component and removes it from the the active model. Components can only be removed at design time.

Animating 3D models.

To bring models to life animation of selected primitives is possible with VR Control. VR Controls built in primitive selector enables a primitive object to be hidden or shown. For information on how to animate models refer to the "Animating Models Guide". This is a PDF that is in the main VR Control program group.

Program window

The program window is where commands are typed in. When writing a new program make sure the cursor is on the top line.



Running a program

When typing in commands you need to be careful to spell the commands correctly and to make sure that the space between each word or number is correct. A space is just one push on the keyboard space bar. If you encounter error messages look carefully at the spelling of the command words. Type in the following commands put each one on a new line.

```
SWITCH ON 1  
WAIT 10  
END
```

Click on the **RUN** program icon. Before our program will work VIRCO checks to make sure all the command lines are typed in correctly. If there are any mistakes the red error pointer will highlight which line VIRCO found the error at. VIRCO will then try to work out the mistake and produce a message box with advice on the error. If VIRCO does not come across any errors the "Program Running" message will appear in the status box. Our program makes the output line number 1 switch on for 10 seconds.

When VIRCO processes the **END** command, it finishes the program.

Make sure you include the **END** command as the last line of your program.

When programming with any computer language develop your programs from simple short ones and add to them stage by stage.

Test them at every stage to make sure they work. As you use VR Control you will find that some combinations of commands will not work in the way you thought. Modify them with a few different commands to achieve the same result.

Commands window

By clicking on the command or number icons, command lines can be entered into the program window without, in most cases, any typing.

Menu Options

The File menu (Design and Program windows)

New program Select this option to create a new program. VR Control opens a message box prompting you to save your old program and design, before clearing these windows.

Open program... Shows the Open program window for you to load a program and design. Type in the file name or select one from the program list.

Save program Once a program and design have been saved, you can select this option to save any changes to your program and design.

Save program as... To save a program and design, go to the File menu and select the Save program as ... option. In the File Name box replace the untitled.vrc name with the name of your program.

All VIRCO programs have the file extension of vrc click on the OK button. This saves the program and active model.

Load Mesh... Loads a DirectX mesh file into to the Active Model (design) window.

Print Program... Prints a copy of your program.

Configure... An external Deltronics USB control box maybe attached to your computer for use with VR Control.

Back Color... Opens the change background color window, this enables the background color of the Active Model (design) window to be changed.

Exit Closes down VR Control.

Edit Menu (Design and Program windows)

Copy Copies the selected text from the program to the windows clipboard.

Cut Cuts the selected text from the program window.

Paste Pastes text from the clipboard into the program window.

Find Opens the Find window. Type into this box, which command or text you need to find.

Inputs Window

The Red leds light up when the input is on.

Count Window

Each time an input is switched on a count is made.

Outputs Window

Clicking on the leds switches on or off an output line.

**Motors Window**

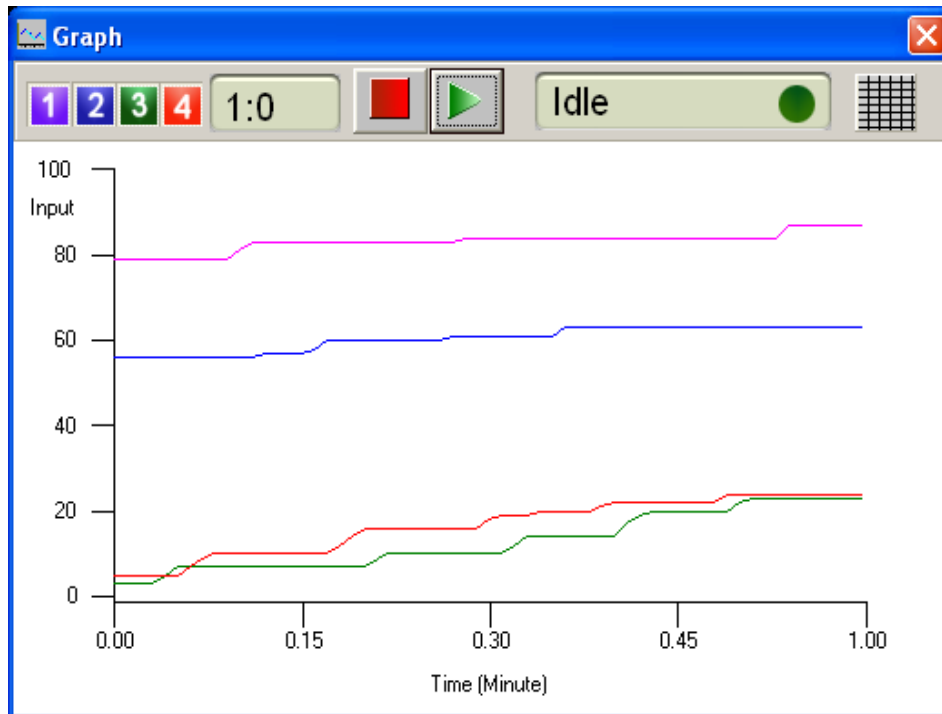
Enables control of each motor.

Analogue Graph Window

To start logging, click on the Start icon, at least one sensor needs to be connected to a mesh. By default the graph displays data as +, click on the line icon to display data as a line graph.

Each connected sensor value is shown in the readings display, to the left of the grid. The graph is set up with a time span of one minute and after this has elapsed the graph will automatically stop logging. The timer shows the current elapsed time. At the end of a logging session data collected can be viewed in the data sheet.

There are a number of facilities for analysing collected data. From the graphs Analyse menu the cross hairs cursor will give a read-out as it passes over data lines of time values.



Graph Window Menus

File Menu

Timespan... Opens the Timespan window. The logging period can be set for between 2 seconds and 60 minutes, each sample rate and time is shown.

Start condition... Logging can be delayed until the start condition is satisfied. To start this function click on the enable box.

Graph name... Enter in the dialog box a name for your graph.

Save graph... Saves a bitmap picture of the graph as a BMP file, the graph can viewed in a paint package.

Print graph... Prints a hardcopy of the graph.

Save data... Data is saved in the CSV file format, which can be imported into a spreadsheet.

Print data Prints a hardcopy of the data.

Edit Menu

Clear Erases data from the graph.

Analyse Menu

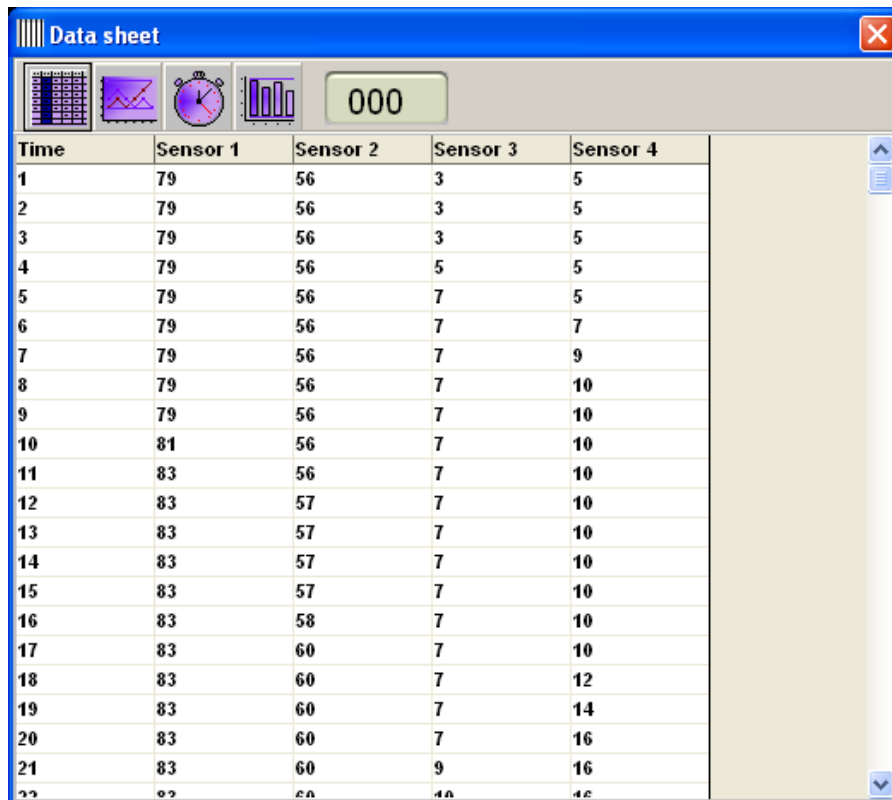
Time To make the cross hairs cursor visible click over the graph, move the centre of the cursor over a graph line, the time will appear in the time status box. To remove the cursor click on the right mouse button.

Data sheet (Analogue) window

While logging takes place each data capture is recorded in the grid. This shows the time in seconds and the value of each sensor. Further analysis of the data is possible by selecting data. Click on the select data icon and hold down the right mouse button while moving the pointer down the column you wish to analyse.

The selected cells background turns from white to blue.

Three options are available to analyse data. Average, change and interval.



Time	Sensor 1	Sensor 2	Sensor 3	Sensor 4
1	79	56	3	5
2	79	56	3	5
3	79	56	3	5
4	79	56	5	5
5	79	56	7	5
6	79	56	7	7
7	79	56	7	9
8	79	56	7	10
9	79	56	7	10
10	81	56	7	10
11	83	56	7	10
12	83	57	7	10
13	83	57	7	10
14	83	57	7	10
15	83	57	7	10
16	83	58	7	10
17	83	60	7	10
18	83	60	7	12
19	83	60	7	14
20	83	60	7	16
21	83	60	9	16

Menu options

Copy data copies all data from the grid onto the Windows clipboard.

Select data Selects data from the grid to be analysed.

Analyse Menu

Average when data is selected the average is displayed in the Average display, this function is also available in the Average frame.

Change the start and finish numbers from the selected data column are subtracted with the result displayed in the Change display, this function is also available in the Change frame.

Interval The start and finish numbers from the selected data time column are subtracted and the result displayed in the Interval display, this function is also available in the Interval frame.

Commands Guide

This is a reference list of VIRCO commands and command lines. Where a group of commands are shown in a sentence i.e. "IF INPUT 2 IS ON THEN" they must be entered in this way. VIRCO will not accept command line variations which are not listed here.

COUNT (n) = 0

Resets the input count variable to 0.

Example: **COUNT 4 = 0** Resets input line 4 count to 0.

END

The **END** command informs VIRCO that this is the end of your program. The **END** command must be the last line of your program.

Example: **END**

END IF

The **END IF** command line informs VIRCO that the **IF** command line has finished.

Example: **END IF**

END PROC

This is used at the end of a procedure.

Example: **END PROC**

END REPEAT

Informs VIRCO this is the end of a **REPEAT** (number) or **REPEAT FOREVER** loop.

Example: **END REPEAT**

END WHILE

Used in a while loop. When VIRCO encounters **END WHILE** either it returns to the start of the **WHILE** loop to test the condition again or carries out the commands below **END WHILE**.

Example: **END WHILE**

GO (procedure name)

To call a procedure use the **GO** command followed by the name of the procedure you wish to call. When the procedure has been carried out VIRCO resumes the program at the line below the GO (name) command line.

Example: **GO lights on**

IF...THEN

Compares a variable to a number if the result is true then the commands are carried out until VIRCO reaches **END IF**. If the result is false the commands are not carried out.

Example: **IF :pl = 6 THEN** carries out the commands if the variable pl is equal to 6 until it reaches **END IF**.

Example: **IF :pl > 32 THEN** carries out the commands if the variable pl is greater than 32 until it reaches **END IF**.

Example: **IF :pl < 8 THEN** carries out the commands if the variable pl is less than 8 until it reaches **END IF**.

IF COUNT (number) > (number) THEN

When you switch on any of the input lines, a count is made in the counts window. This command line enables the count variable to be incorporated into VIRCO programs. If the input line count specified is higher than the comparison number the commands below are carried out, otherwise VIRCO ignores all the command lines until it reaches **END IF**.

Example:

```
IF COUNT 1 > 22 THEN  
SWITCH ON 1  
END IF
```

IF COUNT (number) < (number) THEN

If the input line count specified is lower than the comparison number the commands below are carried out, otherwise VIRCO ignores all the command lines until it reaches **END IF**.

```
IF COUNT 4 < 8 THEN  
SWITCH OFF 6  
END IF
```

IF COUNT (number) = (number) THEN

If the input line count specified equals the comparison number the commands below are carried out, otherwise VIRCO ignores all the command lines until it reaches **END IF**.

```
IF COUNT 6 = 7 THEN  
SWITCH ON ALL  
END IF
```

IF INPUT (number) AND (number) IS OFF THEN

If the two specified input lines are off, carries out the commands until **END IF**.

Example: **IF INPUT 2 AND 5 IS OFF THEN**

IF INPUT (number) AND (number) IS ON THEN

If the two specified input lines are on, carries out the commands until **END IF**.

Example: **IF INPUT 1 AND 6 IS ON THEN**

IF INPUT (number) OR (number) IS OFF THEN

If either of the two specified input lines are off, carries out the commands until **END IF**.

Example: **IF INPUT 3 OR 6 IS OFF THEN**

IF INPUT (number) OR (number) IS ON THEN

If either of the two specified input lines are on, carries out the commands until **END IF**.

Example: **IF INPUT 1 OR 6 IS ON THEN**

IF INPUT (number) IS OFF THEN

Similar to **IF INPUT (number) IS ON THEN** except this line determines if the input is off.

Example: **IF INPUT 4 IS OFF THEN**

IF INPUT (number) IS ON THEN

This command line is similar to the **WAIT UNTIL INPUT (number) IS ON** command line. If the condition is not met, VIRC0 skips over the command lines, until it reaches the **END IF** command. Alternatively if the condition is met VIRC0 carries out the command lines until **END IF**.

Example:

IF INPUT 4 IS ON THEN

SWITCH ON 6

END IF

If input four is on then carry out the commands until **END IF**.

IF OUTPUT (number) IS OFF THEN

This line determines if the output is off.

Example: **IF OUTPUT 3 IS OFF THEN**

IF OUTPUT (number) IS ON THEN

This line determines if the output is on.

Example: **IF OUTPUT 2 IS ON THEN**

IF SENSOR (sensor number) > (number) THEN

Determines if the Sensors reading is greater than the number specified. If so, the commands below are carried out, otherwise VIRCO ignores all the command lines until it reaches **END IF**.

Example:

```
IF SENSOR 1 > 45 THEN  
SWITCH ON ALL  
END IF
```

IF SENSOR (sensor number) < (number) THEN

If the Sensors reading is less than the number specified, the commands below are carried out, otherwise VIRCO ignores all the command lines until it reaches **END IF**.

Example:

```
IF SENSOR 2 < 37 THEN  
SWITCH OFF ALL  
END IF
```

IF SENSOR (sensor number) = (number) THEN

When the Sensors reading is the same as the number specified, the commands below are carried out, otherwise VIRCO ignores all the command lines until it reaches **END IF**.

Example:

```
IF SENSOR 4 = 30 THEN  
SWITCH ON 1,2  
END IF
```

IF TIME > (number) THEN

The computers system time is available with this command line, when you specify a time it must be in this format style:-

Example: 23:12:34

Example: 02:01:09

If the system time number is greater than the one specified, the commands below are carried out, otherwise VIRCO ignores all the command lines until it reaches **END IF**.

Example:

```
IF TIME > 11:45:55 THEN  
SWITCH ON 2  
END IF
```

Note: the system time is based on a twenty four hour clock!

IF TIME < (number) THEN

If the system time number is less than the time number specified, the commands below are carried out, otherwise VIRCO ignores all the command lines until it reaches **END IF**.

Example:

```
IF TIME < 09:55:08 THEN  
SWITCH ON 4  
END IF
```

IF TIME = (number) THEN

If the system time number equals the time number specified, the commands below are carried out, otherwise VIRCO ignores all the command lines until it reaches **END IF**.

Example:

```
IF TIME = 14:43:37 THEN  
SWITCH ON 5  
END IF
```

MOTOR (letter) GO FORWARD

Rotates the motor shaft forwards.

Example: **MOTOR A GO FORWARD**

MOTOR (letter) GO BACKWARD

Turns the motor shaft backwards.

Example: **MOTOR C GO BACKWARD**

MOTOR (letter) STOP

Switches motor off.

Example: **MOTOR A STOP**

POWER (letter),(level)

This command sets the power level of the four motor outputs A,B,C and D. The power level is between 0 and 31.

Example: **POWER A,10**

PROC (name)

A procedure is a list of commands which is called by the main program.

The purpose is to group together a series of commands which you would have to keep typing in over and over again.

Each procedure must use a name which does not conflict with a command word. The name of the procedure is entered after the **PROC** command word.

Examples: **PROC one PROC lights PROC leave**

This procedure switches on and off output lines

PROC lights

SWITCH ON 1 TO 4

WAIT 5

SWITCH OFF 1 TO 4

WAIT 5

END PROC

The **END PROC** command tells VIRCO that this is the end of the procedure.

The procedure(s) must be below the main program but before the **END** command or they will not work properly.

Note: You may call a second procedure from one procedure, but VIRCO does not support calling a further procedure from the second procedure.

REM (text)

The **REM** command is useful for putting in remarks about the program.

Type in the **REM** command followed by any sentence or remarks about the program you are writing.

Example: **REM this is the start of the procedure.**

REPEAT (number)

Repeats the commands on the lines below until it reaches the **END REPEAT** command, when it begins to execute the commands again. It repeats the command line sequence by the number after **REPEAT**.

REPEAT 4 command line will repeat the commands below 4 times.

Example:

```
REPEAT 4  
SWITCH ON 1  
WAIT 1  
SWITCH OFF 1  
WAIT 1  
END REPEAT  
END
```

This example switches on and off output line 1, 4 times.

When you use the **REPEAT** command you must inform VIRCO by using the **END REPEAT** command that you have finished the repeat sequence.

REPEAT FOREVER

Repeats the commands on the lines below until it reaches **END REPEAT** when it begins to execute the commands again. It repeats the commands forever!

Example: **REPEAT FOREVER**

RUB OUT

This command rubs out all the messages in the write panel.

Example: **RUB OUT**

SOUND ON

Makes a sound through the computers internal speaker.

SUB END REPEAT

Signals the end of a **SUB REPEAT (number)/ SUB REPEAT FOREVER** loop.

Example: **SUB END REPEAT**

SUB REPEAT (number)

To repeat a further series of commands within a **REPEAT (number)** loop, use **SUB REPEAT (number)** and the **SUB END REPEAT** command lines.

Example:

```
REPEAT 4  
WAIT 1  
SWITCH ON 1  
WAIT 1  
SWITCH OFF 1  
SUB REPEAT 3  
SWITCH ON 4  
WAIT 1  
SWITCH OFF 4  
WAIT 1  
SUB END REPEAT  
END REPEAT  
END
```

If you go to a procedure from within a **REPEAT (number)** loop and require a further nested repeat loop, use the **SUB REPEAT (number)** and the **SUB END REPEAT** command lines.

Example:

```
REPEAT 6  
GO lights  
END REPEAT  
PROC lights  
SUB REPEAT 3  
SWITCH ON ALL  
WAIT 2  
SWITCH OFF ALL  
WAIT 2  
SUB END REPEAT  
END PROC  
END
```

SUB REPEAT FOREVER

Repeats the commands on the lines below until it reaches **SUB END REPEAT** when it begins to execute the commands again. This command line is used within a standard **REPEAT (number)** or **REPEAT FOREVER** loop.

Example: **SUB REPEAT FOREVER**

STOP

Exits the program you are running.

Example: **STOP**

SWITCH OFF (number),(optional number)

The **SWITCH OFF** command turns off an output line.

To switch an output line off we need a number after **SWITCH OFF**. Like the **SWITCH ON** command this number can be between 1 to 6.

Example: **SWITCH OFF 6** This command switches off output line 6.

Example: **SWITCH OFF 1** This command switches off output line 1.

Example: **SWITCH OFF 3** This command switches off output line 3.

To switch off multiple output lines place a comma after each number (except the last number).

Example: **SWITCH OFF 2,4** Example: **SWITCH OFF 2,4,6**

SWITCH ON (number),(optional number)

The **SWITCH ON** command turns on an output line.

To switch an output line on we need a number after the **SWITCH ON** command. This number can be between 1 to 6.

Example: **SWITCH ON 4** This command switches on output line 4.

Example: **SWITCH ON 2** This command switches on output line 2.

Example: **SWITCH ON 6** This command switches on output line 6.

Example: **SWITCH ON 3** This command switches on output line 3.

To switch on multiple output lines place a comma after each number (except the last number).

Example: **SWITCH ON 1,3** Example: **SWITCH ON 1,5,6**

SWITCH OFF (number) **AND** (number)

This command line switches off two output lines at once. **SWITCH OFF 1 AND 4** will switch off lines 1 and 4.

Example: **SWITCH OFF 3 AND 5** switches off output lines 3 and 5.

Example: **SWITCH OFF 2 AND 4** switches off output lines 2 and 4.

SWITCH ON (number) **AND** (number)

With this command line we can switch on two lines at once. **SWITCH ON 1 AND 4** will switch on lines 1 and 4.

Example: **SWITCH ON 3 AND 6** switches on output lines 3 and 6.

Example: **SWITCH ON 2 AND 4** switches on output lines 2 and 4.

SWITCH OFF (first number) **TO** (second number)

This command line switches off all the output lines between and including the two numbers.

Example: **SWITCH OFF 2 TO 5** switches off output lines 2 to 5.

Example: **SWITCH OFF 4 TO 6** switches off output lines 4 to 6. Note that for this command line to work the first of the two numbers must be lower than the end number.

Example: **SWITCH OFF 6 TO 2** will not work.

Example: **SWITCH OFF 2 TO 6** will work.

SWITCH ON (first number) **TO** (second number)

This command line switches on all the output lines between and including the two numbers.

Example: **SWITCH ON 3 TO 6** switches on output lines 3 to 6.

Example: **SWITCH ON 1 TO 5** switches on output lines 1 to 5. Note that for this command line to work the first of the two numbers must be lower than the end number.

Example: **SWITCH ON 6 TO 1** will not work.

Example: **SWITCH ON 1 TO 6** will work.

SWITCH OFF ALL

Switches off all output lines.

SWITCH ON ALL

Switches on all output lines.

WAIT (number)

The **WAIT** command will not execute any further commands until the time; expressed in seconds, has elapsed.

The number after **WAIT** is how long you want the program to wait for. **WAIT 10** will hold for 10 seconds.

WAIT 3 will hold for 3 seconds.

Example: **WAIT 7** holds for 7 seconds.

The maximum number **WAIT** will hold for is 60.

WAIT UNTIL INPUT (number) IS OFF

Using this command line lets you halt the program, until one of the eight input lines is off.

Example: **WAIT UNTIL INPUT 3 IS OFF**

When VIRCO comes across this line in your program it waits, until input 3 is off, before continuing with the program.

WAIT UNTIL INPUT (number) IS ON

Using this command line lets you halt the program, until one of the eight input lines is on.

Example: **WAIT UNTIL INPUT 5 IS ON**

When VIRCO comes across this line in your program it waits, until input 5 is on, before continuing with the program.

WAIT UNTIL SENSOR (sensor number) > (number)

This command line lets you halt the program, until the sensor reading is higher than the number you have specified.

Example: **WAIT UNTIL SENSOR 1 > 23**

When VIRCO comes across this line in a program it waits; until the reading for sensor 1 is higher than the number after the greater than symbol(>), before continuing with the program.

WAIT UNTIL SENSOR (sensor number) < (number)

VIRCO waits until the sensor reading is lower than the number you have specified.

Example: **WAIT UNTIL SENSOR 2 < 46**

WAIT UNTIL SENSOR (sensor number) = (number)

VIRCO carries out the commands below this line when the sensor reading equals the number you have specified.

Example: **WAIT UNTIL SENSOR 4 = 68**