

■ Procedures

To *procname* ... End
 Return
 Global
 Local
 AutoDestruct
 ParamCount *Optional parameters*

■ Brackets

() *Parameter list; expression grouping*
 [] *Multiple statement*

■ Control flow

If Else
 While
 Do ... While
 Forever
 Repeat
 Every *mS timed loop*
 Index *Loop count*
 Index0 *From 0*
 Break
 Select Case

■ Math Operators

x := *y* *Assignment*
 + - * /
 << >> *Shift*
 ^ *Power*
 Div *Integer division*
 Mod
 Abs
 Sqrt
 Exp
 Log
 Sin
 Cos
 Tan
 Asin
 Acos
 Atan

■ Logical and comparative

AndAlso
 OrElse
 IsFalse
 = <> < > <= >=

■ Bitwise operators

And
 Or
 Eor
 Inv *Invert bits*

■ Values

True
 False
 Nil
 \$1234ABCD *Hex constant*
 %10010111011100101 *Binary constant*
 123.456 *Float constant*
 123.456E10 *Float constant*
 "Hello World" *String constant*
 'A' *Character constant*
 \\ \" \h *Escape sequences*

■ Timing

Wait *mS wait*
 Await
 Every *mS timed loop*

■ Multitasking

Start *code*
 Stop *task*
 Stop All

■ System

List [*name, All, Task, Word*]
 Help
 Protect (n) *Program flash*

■ Printing

Print
 Print To
 Printf
 Home
 CLS
 CR
 BS
 Beep
 Chr *Print ASCII character*
 Font
 GotoXY
 Htab
 Vtab
 Left
 Right
 Centre
 ~ *Print as hex*
 ~~ *Print as binary*
 : *Format specifier*

■ Objects

Make
 New
obj.msg *Send a message*

■ Exception handling

Try
 Catch
 Exit

■ Data structures

Array
 Buffer
 Class
 String
 TextBlock

■ Comments

; *comment* *Line comment*

■ Classes/Methods

Class ... End
 TO *method* ... End
 This
 Global
 Assignment
 AutoDestruct
 Is

■ Data types

Any
 Int 8 *Unsigned*
 Int 16 *Unsigned*
 Int *32 bits*
 Float *IEEE single precision*
 String
 Buffer
 Class
 Array
 As Int
 As Float
 TypeOf

■ Pre-processor

#Define
 #Define (param)
 #UnDef
 #Redefine
 #If
 #Else
 #Elif
 #EndIf

■ Advanced

@ *Form a pointer*
 ! *De-ref pointer*
 [! (ptr) (params)] *Call a msg ptr*
 @.MsgName *Form a msg ref*
 obj.!(msgRef) (params) *Call a msg ref*
 This. [class]Message *Class-assertion*
 This. [^]Message *Class-assertion*
 This. []Message *De-assertion*

■ Built in object types

This is a list of object types built into Venom. You can also write your own types using the **Class** keyword.

AlphaLCD
Analogue
Array
Buffer
CANBus
CRCGenerator
DateTime
Digital
Encrypter
Ethernet
FileSystem
FTPClient
FTPServer
GraphicsLCD
HashGenerator
HTTPTServer
I2Cbus
IProt
Keypad
NumberReader
OnBoardLED
OneWire
OperatingSystem
PIDController
POP3Mailbox
PPPProt
ProtAnalyser
PulseCounter
PulseWidthIn
PulseWidthOut
RandomNumberGen
RealTimeClock
SafeData
Semaphore
SerialPort
Shaft
SMSLink
SMTPSender
SPI
Stopwatch
String
TCPProt
Timer
TouchScreen

UDProt
XMODEMLink

■ System commands

These names are interpreted as system commands. They may be used as method names, but are not recommended for other variable names.

Debug	Many useful utilities
PrintF	The PrintF command
Protect	Copy the program to flash
Reset	Reset the VM2
Run	Run the program

■ Operator precedence

As Int As Float IsFalse • (dot)
! ? Abs int TypeOf Sin Cos, etc
* / ^ Div Mod >> <<
+ -
> < >= <= = <>
AndAlso OrElse And Or Eor

■ Multi-tasking

Tasks are swapped on a Round-Robin scheme. The task swap period is nominally **1mS** per task.

Most applications use between one and four tasks. If you use more tasks than this then you may want to review your design.

In general a program is easier to write and understand if all secondary tasks are started near the start of the program, and never halt.

Signalling between tasks typically uses global variables (including global objects). Resources used by more than one task may be locked. Some locking is handled automatically by the system, mostly in the Print command.

■ Development

VenomIDE is the development environment for Venom. Download it for free from:

www.microrobotics.co.uk

■ Documentation

The *Venom Tutorial* (200 pages) and *Venom2 Help File* (500 pages) are available from within VenomIDE or at

www.microrobotics.co.uk

■ Getting started

- Set the *Prog Mode* switch to **ON** and power up the VM2 Application Board
- Type **Y** at the *Clear RAM* prompt on the VenomIDE terminal screen, to reset the VM2's program memory.

■ Your first program

- Open a new Venom file using VenomIDE's **File>New** menu
- Type in a new program, such as:

```
To main
  Print "Hello World", CR
End
```
- By convention the main part of a program is called **main**
- Download your program using **F7**
- Run your program using **F10**

■ Help

Right-clicking on any Venom keyword in your program links to the Help Page for that keyword. If there is more than one 'hit' then you are taken to the index entries for that keyword.

■ Debugging

Most debugging is done using Print statements in your code. You can also hit Ctrl-T when a program is running to dump the state of each task to the terminal.

At the command line you can run sections of your code or print out the value of any global variable.

■ Release

■ Run at power-up

To have your application run at every startup, turn off the *Prog Mode* switch.

■ Protect your application in Flash

To protect your application from accidental erasure in the field, copy it from RAM to flash using **Protect (1)**.

■ Production programming

To program a batch of VM2 during production see *Production Programming* in the *Venom2 Help File*.