Micro-Robotics Ltd
www.microrobotics.co.uk
Tel +44 (0) 1223 523100
Registered Office: 135 Ditton Walk, Cambridge CB5 8QB
Registered in England No. 03983610 VAT No. GB 393 2883 17

Migrating to Venom2

The release of VM2 – a completely new *VM* hardware platform – made some degree of change to the Venom language unavoidable. So, as any code ported from VM1 to VM2 would have to be edited anyway, it was a good opportunity for us to make other useful changes to the new language.

This document details the differences between *Venom2* and *Venom-SC*. We will only give a brief description of the changes here – you are referred to the *Venom2 Help File* for the full description.

Part 1: Language Changes

Here is the list of changes to the *language* (as opposed to the library of *objects*) made when we created Venom2, usually because Venom-SC was deficient in the area of the change.

User-defined Classes

You can now define your own classes of objects allowing far more sophistication in Venom applications. For more details see the *Venom Tutorial* and *Venom2 Help File*.

Optional Parameters

Parameter lists to Venom procedures may now declare some of all of the parameters as optional by putting square brackets around the optional part of the parameter list. The number of parameters supplied in a particular call is given by the new keyword ParamCount, and parameters may be accessed as an array using the Parameter(n) Venom function.

```
TO procedure(a, b, [c, d])
   Repeat ParamCount
   [
      Print Parameter(Index0), CR
   ]
END
```

PrintF printing added

Although **`PrintF`** is actually a message name we will deal with it in this section as it is important enough to treat as part of the language. Briefly, the **`PrintF`** message can be sent to any object that accepts text (e.g a SerialPort or a String object), or used 'by itself' it can be used to send print to the default output stream – the main serial port usually.

It takes parameters that are very similar to the C languages's *printf()* function, but also has some very useful extensions, such as printing binary values and printing objects.

(The old **`:"C format string"`** formatting has been removed)

True change to 1

The value of the Venom constant **True** has been changed from -1 to 1.

NOT operator changed to INV

The `NOT` keyword has been removed from the language and replaced by `INV`. `INV` does exactly the same as `NOT` did - it inverts each bit in the operand to yield the bitwise inverse of the result.

True logical operators added

Three new operators have been added to improve *logical expressions*: `AndAlso`, `OrElse` and `IsFalse`.

`AndAlso` evaluates to `TRUE` if both of it's operands are non-zero, and furthermore, it is 'lazy', or 'short circuiting' – that is it won't go on to evaluate the right hand operand if the left hand operand is zero (`FALSE`).

`OrElse` evaluates to `TRUE` if either of its operands are non-zero. It is also 'lazy' – it won't go on to evaluate the right hand operand if the left hand operand is non-zero.

`IsFalse` evaluates to `TRUE` if the operand was `FALSE` (or zero), and it evaluates to `FALSE` if the operand was non-zero.

Shift operators added

The new operators `<<` and `>>` perform left and right binary shift operations. They have the same operator precedence as `*`, `/`, `^` etc.

Array operation changed

The *size* parameter of constant arrays is now optional – if you leave it out the array will be sized according to the number of initialisers you supply.

Non-volatile arrays removed

Non-volatile arrays have been removed, as they were too confusing in their syntax and operation. Use non-volatile files (in SRAM or Flash) for similar functionality. The Reset, Valid and Checksum messages have been removed as they were only used in non-volatile arrays.

CR internal representation changed

The internal representation of `CR` inside Venom2 is the single character with ASCII value $0A or 10 in decimal. This will be converted to the carriage return/line feed pair [$0D, $0A] by some objects when you print `CR` to them.

DECLARE removed

Declare has been removed as it's function has been largely superseded by the 'code analysis' tool.

CATCH, REGION, THROW, EXIT changed

There has been a major re-write of the error and exception handling system. A new keyword **`TRY`** has been introduced, and the keywords **`THROW`** and **`REGION`** have been removed. **`CATCH`** and **`EXIT`** have changed in their function.
The new system is much easier to understand and use, and is similar to that used in C++ and C#.
Please read the Tutorial and Help file for how to use the new system. If you have used any of the keywords listed above then your code will need some modification.

Startup code changed

The default startup procedure in Venom2 has slightly different code in it compared to Venom-SC. **`LIST startup`** will list the default startup code.

Macros improved

`#DEFINE` macros have been much improved over Venom-SC:

- Macros can take parameters.
- Macros use their own separate name space.
- Macros can appear inside a procedure definition.
- Macros can be undefined (with **`#UNDEF`**)

- Macros can be redefined without a warning using **`#REDEFINE`**

Conditional Compilation supported

`#IF`, **`#ENDIF`**, **`#ELSE`**, **`#ELIF`** are recognised byVenom2, allowing you to select which bits of your application program are seen by the compiler.

Mode change no longer restarts controller

This isn't really a change to the language, but is a change of behaviour of the operating system: when changing the state of the program mode switch the controller doesn't now restart in the new mode – you have to press the Reset button to restart the controller in the selected mode.

Part 2: Object Changes

Object name changes

We have changed the name of some objects to make them more descriptive or less cumbersome.

- `SerialIO` has changed to `SPI`
- `AsynchronousSerial` has changed to `SerialPort`

SerialPort

There are now many more serial ports supported by the hardware, and they can run at much higher speeds.

The Baud message is now called Speed.

New Semaphore object

The new Semaphore object implements both the classic semaphore function and also has a Venom Lock, so it may be used in two ways to control access to shared resources in a multitasking application.

Digital syntax

The MAKE syntax for 'onboard' Digital I/Os on the VM2 has been changed. You can now define many attributes for each I/O pin.

Digital objects on PCF8574 ICs have not changed in the way they work.

Operating system changes

There is a new Speed message that is applied to the Operating system object. It controls the clock speed of the processor core. It allows you to set the clock speed in MHz – ranging from 16 to 72 in steps of 8MHz.

There is now no UserSwitch message – you should use Digital objects to read the state of the equivalent inputs.

Removed: NotAsserted, AwaitAsserted and AwaitNotAsserted messages

The message names NotAsserted, AwaitAsserted and AwaitNotAsserted have been removed for any object that had them as they were not often used and were confusing. The increased speed of the VM2 makes their original use redundant.

Replace

```
obj.NotAsserted
```

with

```
obj.Asserted IsFalse
```

And similarly for the other messages.

LED changes

The Flash message now takes bit patterns allowing you to construct your own LED flash sequences.

There is now no Value message for the LED.

String object improvement

String objects have been internally improved by adding an internal write pointer.

Timer

Timer no longer has the Asserted or AwaitAsserted messages.

AlphaLCD

When printing to the AlphaLCD object, there is no word wrapping by default. This means that text will just run off the end of the line and not be seen. You can turn word wrapping on.

NumberReader improvement

NumberReader now takes its *number base* as the first parameter, e.g.: 10, 16 or 2 for decimal, hexadecimal or binary. Other number bases are also possible.

The Conversion message is now called Mapping.

I^{2}C Bus – big change to syntax

The I^{2}C Bus has a totally different programmers' interface to before because it is now implemented in hardware, rather than 'bit banging' software.

GraphicsLCD

Changes to GraphicsLCD object

- Colour displays are supported
- Windows have been removed; their function is largely replaced by TextBoxes.
- There is a new Button object – useful as a basis for implementing Menu-based Graphical User Interfaces.
- Boxes are now specified by the coordinates of a corner and x/y size values.
- Pen message now takes an index to access one of several different 'pens'.
- Sprite message is now Bitmap; Bitmap data format has changed.
- Box borders: LSB is outermost; New 3D borders added
- No Marker, Bar, Display or Graticule messages.
- Default font set is more consistent.

- You can use the upper half of the 8-bit ASCII range for extended characters in fonts, e.g. characters with accents.
- FontData format has changed, 'anti-aliased' fonts are available.
- New fonts are easily created from TrueType fonts.
- There is much more room to store fonts, and they are more easily stored as files in the flash filing system. There is a utility to convert fonts to Venom formats.

DateTime

The Nudge message is now called Adjust.

Keypad consolidation

The Nudge message has been removed entirely.

The InputBuffer function has been incorporated into the keypad object, rather than being in a sub-object.

The Keypad object is not now recommended for use with the Touchscreen object to make Graphical User Interfaces.

RealTimeClock improvement

The clock now has an Adjust message that allows you to correct slow or fast running with a precision of around 1ppm, or 30 seconds per year.

CANBus

The CANBus object is much improved – mainly because the hardware to implement the bus is now built into the VM2. This makes it much faster and easier to use.

There are also a few changes to the object interface, and it's possible we will make more changes as more people come to use it and request improvements.

Make: the parameters to MAKE have changed.

Baud: This message is now called Speed, and takes an actual bit rate as its parameter.

Window, On: the details of these messages have changed to reflect the underlying hardware.

Debug: this message accesses some important debug features in the hardware: the silent and loop back modes.

SafeData

Fully implemented.

Note that object interface has changed for the Get and Put messages to make it easier to use. An Address message has been added.

File systems

There are four filing systems available on the VM2:

- Flash filing System – 7M bytes in the VM2's flash memory
- RAM Filing System: around 1M byte on the VM2's SRAM.
- Memory card Filing System: with suitable hardware on the application board SD and MMC cards may be used – card capacities up to 2GB currently.
- USB Host filing system: with suitable hardware on the application board you can plug in any USB drive.

The Flash Filing System (FFS) has a special feature where it is possible to store files in such a way that they may be accessed by direct memory addressing at very high speed. This is used for high speed access to bitmap images, fonts, audio files, etc.

It's also possible for the VM2 to emulate a USB memory stick – so that the Flash Filing System becomes visible to your PC as a flash drive if you connect it using USB.

The Flash Filing System may be used for update the firmware in a VM2, either during production or in the field. Both the application and operating system firmware are updatable this way.

PulseCounter, PulseWidthOut, PulseWidthIn, Shaft

Fully implemented.

Note that syntax has changed in all cases, and many new features have been added.

SPI (used to be SerialIO)

Mostly implemented. The MAKE parameters have changed and the Put message works in a different way.

Not Implemented Yet

The following objects have not been implemented as yet – but most will be. Please contact us if you need a particular object to be put to the front of the list.

FrequencyIn

Not done yet. This may be implemented on demand.