Introduction to Aemulor Pro

Aemulor Pro is the professional edition of Aemulor, allowing many more 26-bit applications to be used on Castle's IYONIX pc. In particular, programs such as filing systems, audio applications and games that Aemulor is unable to run properly.

The basic version of Aemulor, which is sufficient to run most desktop applications, was described in an earlier Foundation RISC User article, so here we will concentrate upon the new features of Aemulor Pro and how it differs from Aemulor.

Sound & Filing Systems

Aemulor Pro extends the 26-bit emulation to include support for audio output and filing system code. Specifically, it allows the use of 26-bit voice generators, channel handlers and schedulers which covers almost all forms of sound production.

Emulating 26-bit voice generators poses some interesting technical challenges because Aemulor itself must be capable of running in IRQ mode with IRQs enabled, ie. it cannot use normal procedure calls (via the BL instruction). This constraint was anticipated when Aemulor was first developed, thus avoiding the need to extensively re-work the code, but it was still one of the greatest technical hurdles in Aemulor Pro's development.

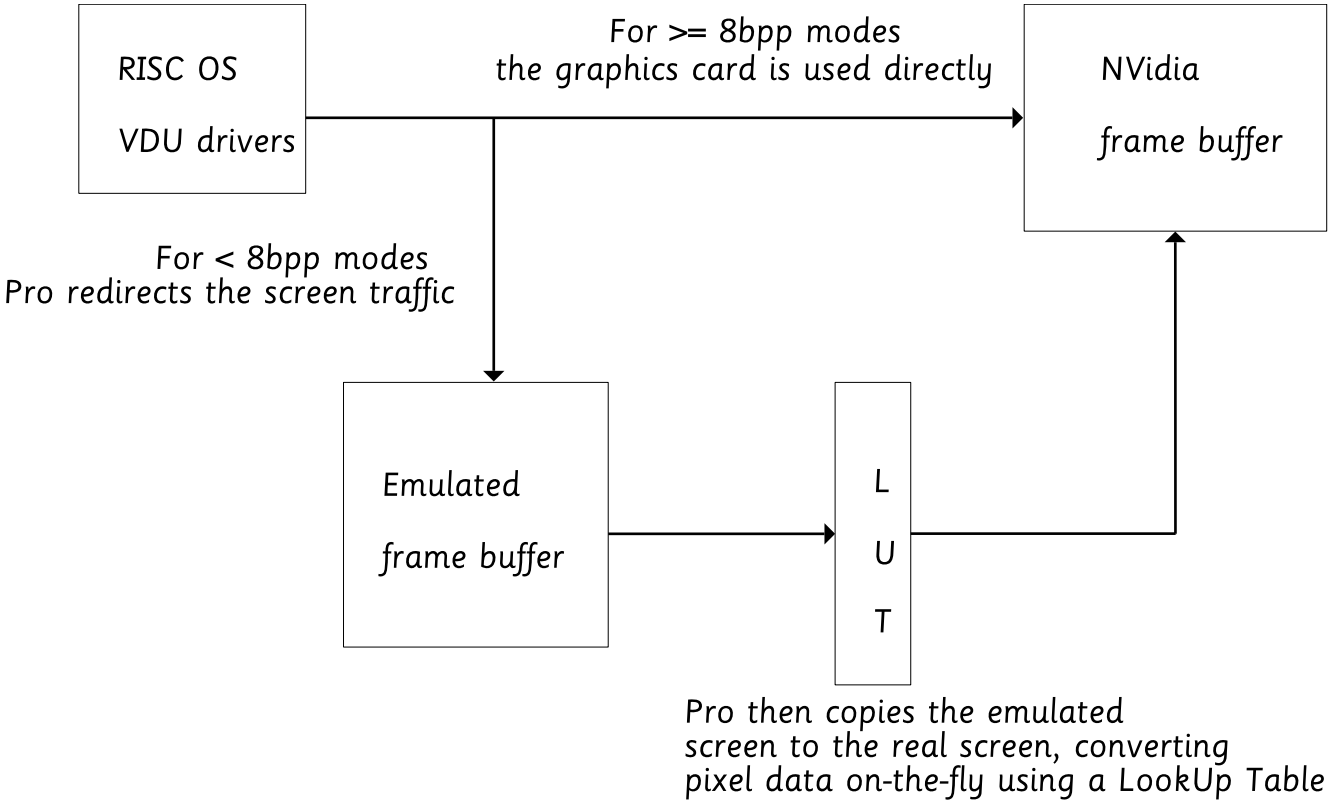
Supporting filing systems, however, is no more difficult than emulating vector handlers, for example, which the basic version of Aemulor already does. The necessary code wasn't ready when Aemulor was first released because very few applications actually require the use of 26-bit filing systems. It's worth nothing that the 26-bit filing systems are visible to the 32-bit world as well, allowing their use through the Filer in the normal manner. 32-bit applications can also use the low-colour screen modes which Aemulor Pro provides.

Screen mode emulation

The IYONIX's NVidia graphics card, in common with many other modern graphics cards, does not support the low-colour screens used by old games and applications. Therefore, 26-bit programs which can only work in 2-,4- or 16-colour screen modes cannot be used on the IYONIX pc even with the help of Aemulor. This is a particular problem for Sibelius which demands a 4-colour screen mode and which is very important to a number of RISC OS users.

To solve this problem, Aemulor Pro steps in between the RISC OS kernel and the NVidia driver, to provide the screen modes that the graphics hardware cannot provide itself. The result is that these low-colour screen modes are thus made available to those 26-bit programs which need them, but also to the 32-bit world. These modes can even be used in the desktop, though this is not recommended because Aemulor Pro has to convert the screen image in software which carries a fairly severe performance overhead for large screen modes. Old 26-bit programs generally use lower-resolution screen modes, however, so this isn't a problem.

The emulated screen modes are implemented by telling RISC OS to use a different area of memory for the screen image, so that the full graphics capabilities of the RISC OS kernel are still available, including scrolling and bankswitching. This ensures much better compatibility than would be achieved by intercepting all graphics operations and reimplementing them ourselves.



Upon entering a low-colour screen mode, Aemulor Pro tells the NVidia graphics driver to switch into an 8bpp screen mode, and then converts the emulated screen image to 8pp for display. This conversion is done up to 15 times per second to ensure a smooth update, which is particularly important for games. A future version of Aemulor Pro may use a different approach to screen updating but the idea is currently unproven so I won't describe it here.

Hardware emulation

Some old games bypass the OS and directly manipulate the hardware they expect to be present, particularly the VIDC and IOC on Archimedes-era machines and, to a lesser extent, their later counterparts the VIDC20 and IOMD on the RiscPC. Because these devices are not present on the IYONIX pc, they must be emulated by Aemulor Pro for those games to work. In fact a number of those games will not work on the RiscPC because it lacks a VIDC or other Archimedes-era hardware.

With this in mind, Aemulor Pro intercepts accesses to the address ranges at which such hardware is present on the Archimedes and RiscPC machines and emulates its functionality in software. This hardware emulation provides the following:

* VIDC/VIDC20 palette registers
* IOC/IOMD IRQ/FIQ registers
* IOC/IOMD Timers

These are the hardware features most often manipulated directly by games instead of calling the APIs provided by the OS. Aemulor Pro does not currently emulate the remainder of the Archimedes/RiscPC hardware such as floppy disc access, or direct audio output using VIDC because such operations are invariably performed via the host OS.

For each 26-bit application used under Aemulor Pro there is a configuration option to choose whether 'Archimedes' or 'RiscPC' hardware is emulated. In fact, wherever possible, Aemulor Pro will make both available to the application to make configuration easier for the user - eg. VIDC and VIDC20 use different addressing ranges so Aemulor Pro makes both available to the application. The configuration option just

tells Aemulor Pro which machine it should emulate for that application when a decision *must* be made.

One hardware feature currently not emulated by Aemulor Pro is the screen configuration registers of the VIDC and VIDC20 devices. These are manipulated by a small number of games to achieve some scrolling effects, so it's anticipated that they too will be emulated in a future release of Aemulor Pro.

The hardware emulation should also make it easier to get 26-bit podule drivers working under Aemulor Pro, though in fact we've had very few enquiries about this possibility because most podule hardware has now been superseded.

Just In Time Compilation

Whilst Aemulor's emulation is already a lot faster than was ever anticipated, extra speed is usually beneficial and one of the promised features of Aemulor Pro is faster emulation over the basic version of Aemulor. Whilst the current release of Aemulor Pro is actually slightly faster than Aemulor, greater speed improvements should be attained in a future release by the inclusion of the JIT compiler. The compiler is currently disabled, pending further development and reliability improvements.

With the JIT compiler enabled, Aemulor Pro will be able to compile most 26-bit code into equivalent 32-bit code which then executes at the full native speed of the IYONIX pc. In fact, since the JIT compiler tunes the code for better performance on the XScale it will sometimes execute faster than the 26-bit code would if the XScale were capable of running it natively. As a further performance improvement, Aemulor Pro will be able to remember the 32-bit code that it has generated and thus avoid the cost of analysing and recompiling the 26-bit code every time the application is used.

It's worth noting that Aemulor will still be required for running the resultant 32-bit code, however. In general it's not possible to compile the whole 26-bit application into an equivalent standalone 32-bit one (in fact, it would be a rather dubious act on legal grounds!) The reason for this is that, the program must be executed in order to discern what it does - up-front analysis can only achieve so much - and the program is always liable to execute some code that has never previously been executed, eg. in response to an error whilst loading a document. In these cases, there will be no pre-compiled code for handling the error, and Aemulor Pro must be running so that it can fall back to emulating the 26-bit code.

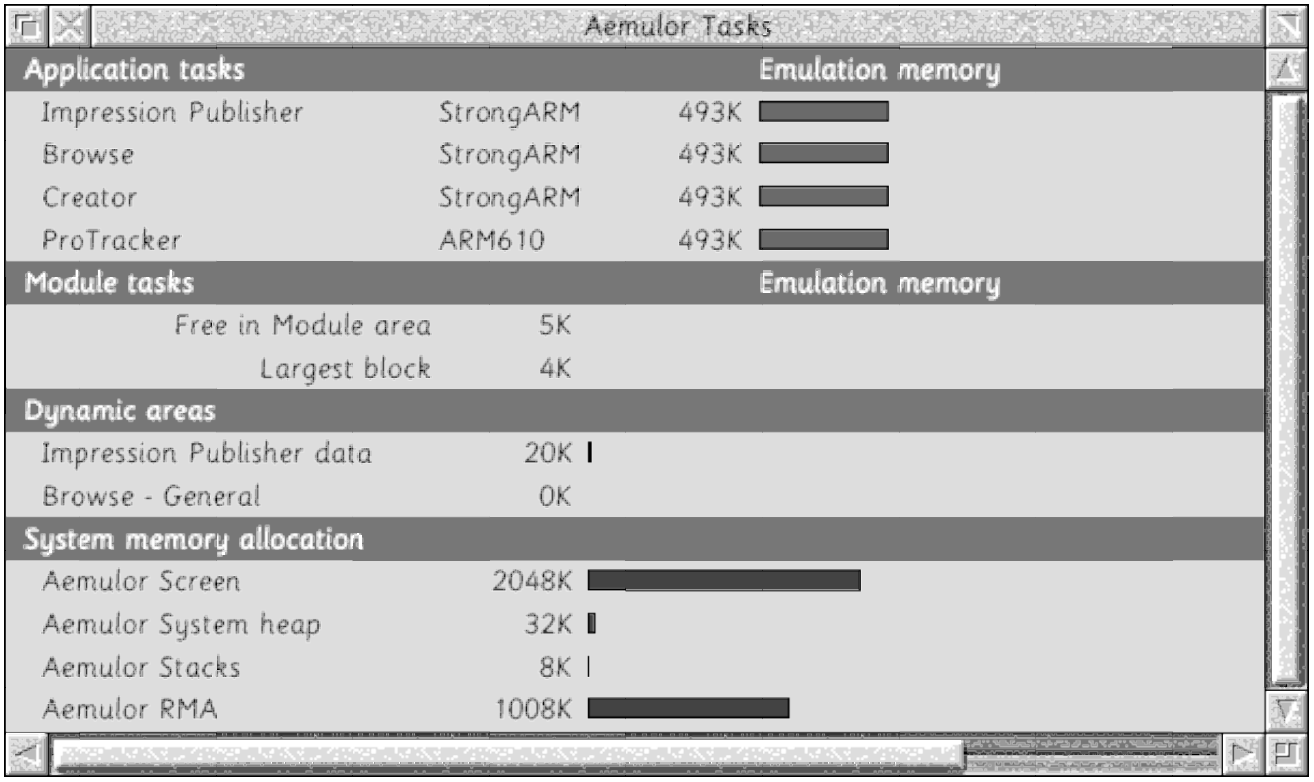
The inclusion of compilation in Aemulor Pro is particularly exciting because it opens up the possibility of many more optimisations such as substituting XScale-tuned equivalent code or increasing the performance of floating point maths by avoiding the repeated decoding of instructions.

Whilst it should be stated that JIT compilation will not work for all applications, it is expected to work for the vast majority of StrongARM-aware programs, and the other emulation engines will always be available for those applications which cannot be used with the compilation.

Task display

One of the features most often requested for inclusion in Aemulor was a list of the 26-bit applications being emulated so that it's easy to determine whether a program is being emulated by Aemulor or running natively. Because Aemulor was written to operate transparently wherever possible, it's not always obvious what's being emulated, at least not until you try to use a 26-bit application without Aemulor loaded. Often that will crash the entire machine!

So Aemulor Pro includes a Task display window which is modelled on the familiar interface of the Task Manager, with each emulated application being shown alongside a graphical display of the memory being used for its emulation.



With the current release of Aemulor Pro this is a fixed amount, but with the introduction of JIT compilation there will be a tradeoff between the amount of memory used and the emulation speed of an application. Instructing the compiler to allocate more memory for your favourite programs will normally allow them to run faster.

Future development

To conclude, I'll discuss some possible directions in which Aemulor Pro may be developed in the future, but please be aware that these are still highly tentative.

Improved handling of low-resolution screen modes

This is one improvement that's due for inclusion in the next release of Aemulor Pro because it affects a number of users. Low-resolution screen modes, ie. those which are often used by older games, can cause problems for today's much higher-resolution, higher frame-rate monitors. Where it is possible to produce a usable picture, it can often appear heavily pixellated oweing to the low resolution, and the frame rate will often be too high, causing the game, and especially the music, to play too quickly.

To solve these problems, Aemulor Pro will in future be capable of rescaling the screen image in software so that the monitor can be operated in a suitable higher-resolution, higher frame-rate mode, whilst the game sees the resolution and frame rate that it expects. The user interface will be extended to allow the user to specify which native screen mode should be used for each emulated mode.

Control of the emulation speed

A further problem encountered by users playing games under Aemulor Pro is that the game can often play too quickly oweing to the much greater speed of the IYONIX pc. Some limited control of the emulation speed is available because Aemulor Pro has three emulation engines with very different speeds - the ARM3, ARM610 and StrongARM engines. However, this is a fairly crude adjustment and some games will not work with all emulation engines, so it's likely that a future version of Aemulor Pro will include finer control of the speed, even allowing you to slow down a game just for playing those tricky bits!

Protecting the 32-bit OS and applications

Some 26-bit programs, particularly older games, attempt to do nasty things to the system, often in an attempt to ensure that there's enough memory available. Since the IYONIX pc has a much larger amount of memory these tricks are now redundant and only cause harm. Since Aemulor sits between the 26-bit program and the operating system, it can protect the OS and 32-bit applications from at least some of these changes.

Similarly, a number of 26-bit games don't tidy up after themselves with the result that when you quit the game, especially if you stop it abruptly using Alt-Break, RISC OS itself crashes and must be reset. With Aemulor acting as a protective layer between the game and the OS it should ultimately be possible to Alt-Break any 26-bit program with impunity because Aemulor can do the necessary tidying up (removing handlers, deregistering voices etc) on its behalf.

Running pre-RiscPC software on the RiscPC

Since Aemulor Pro is now capable of running a number of software titles that haven't worked since the introduction of the RiscPC and StrongARM CPU, it has been suggested that Aemulor could be modified to run these programs on a RiscPC. Technically this should be relatively easy because the ARM3 and ARM610 engines present in Aemulor Pro do not require features specific to the XScale CPU. It remains to be seen how much work this would involve, what sort of performance it would give running on a StrongARM RiscPC, and just how many programs would benefit from it.

Windowed emulation of single-tasking applications

A couple of users have mooted the idea of Aemulor Pro being able to run singletasking, full-screen programs such as games, within a window on the desktop - akin to using a TaskWindow but with full support for all graphical output, audio output, keyboard and mouse input etc. From a technical standpoint this is a large amount of work, but the end result is highly desirable and it would be a very welcome addition to Aemulor Pro.

Other possible applications of the StrongARM engine

With the arrival of hardware emulation in Aemulor Pro, the emulator has moved a step further towards being a full machine-emulation rather than a compatibility layer between applications and the OS. A natural progression of this would be to develop Aemulor Pro into a virtual machine, such as a RiscPC or even an IYONIX pc, which could be used to 'sandbox' applications - that is, to run them in a protected environment so that they can't crash the host OS.

This would be particularly useful to software developers because a virtual machine can always provide diagnostic information, even in the event of a software failure that takes the emulated OS with it.