

ADVANCE INFORMATION



An introduction to OpenRL

A platform-independent API for hardware acceleration
of intensive image processing tasks

www.openrl.org



V2.0
5 September 2005

An introduction to OpenRL

Platform-independent API for hardware acceleration



What is OpenRL?

OpenRL is a new platform-independent Application Programming Interface (API) which allows developers of high-performance image processing applications (such as video effects plug-ins for non-linear editing), to take advantage of high performance hardware acceleration platforms, such as Aspex's Accelera™ 3000 PCI-X card.

The OpenRL API is a high-level API, allowing programmers to access the power of embedded hardware acceleration platforms without having to learn a new programming model.

Unlike OpenGL™ or DirectX™, OpenRL provides a high-level node-based architecture, optimised for processing bitmap images. With OpenRL, developers can quickly develop complex image processing applications by connecting together different "nodes" or filters.

What does the name "OpenRL" mean?

OpenRL stands for "Open Rendering Library". We decided to call the API "OpenRL", as it shares several conceptual similarities with the widely-used OpenGL API:

- A software interface to graphics processing hardware
- Platform-independent, supporting multiple graphics hardware platforms
- Operating-System independent
- An open industry standard
- Royalty-free API

How is OpenRL different from OpenGL or DirectX?

OpenGL and DirectX were developed and optimised for the display of synthetic objects (such as PC graphics or 3D games) onto a monitor using the GPU on a graphics card.

OpenGL is not well suited for 2D processing of real (bitmap) images, such as film or video editing, which only use a small subset of the OpenGL pipeline, and which usually operate from disk: disk or memory: memory rather than being drawn on a screen.

Non-linear editing systems and effects plug-ins often use parts of OpenGL for rendering previews to the screen, but still use the CPU for final rendering.

OpenRL is intended to allow acceleration of that final rendering step.

To quote nVidia: "OpenGL and DirectX expose too many hardware details...and are burdened with large portions of the API dedicated to interactive functionality that is not used in film."

How is OpenRL different from DirectShow?

Although conceptually similar to DirectShow, which also provides a high-level node-based architecture, OpenRL will be supported across multiple platforms, whereas DirectShow is limited to Windows operating systems.

OpenRL also includes native support for hardware acceleration, including acceleration of multiple graph nodes by the same hardware platform.

Will OpenRL be an open standard?

Yes, the OpenRL API will be a royalty-free, open standard.

Is OpenRL part of the OpenGL/OpenML standards?

Not at the moment. OpenGL and OpenML™ are maintained by the Khronos Group. The OpenRL API is being developed independently by Imagineer Systems and Aspex Semiconductor.

What is the architecture of OpenRL?

OpenRL is a node-based architecture. This means that each library function is represented as a "node", or filter, with input and output "pins". Additional pins are used to configure parameters for the nodes. The programmer constructs the required functionality by connecting together nodes in a "node tree", or filter graph, similar to that used in Apple Shake or Microsoft DirectShow.

An introduction to OpenRL

Platform-independent API for hardware acceleration



Is OpenRL a framework, a library, or an API?

OpenRL is all three! OpenRL consists of three main elements:

1. The **OpenRL API**, which provides C functions to create nodes, connect them together using pins, and to set and read values of pins;
2. **OpenRL node libraries**, with a core library of basic image processing functions, and a framework for development of new library functions. OpenRL libraries consist of functional code, and supporting XML metadata, which describes the capabilities & performance of the libraries, allowing the scheduler to make intelligent decisions about execution of the node tree;
3. The **OpenRL Scheduler**, which controls the execution of the OpenRL node tree on the available resources.

Can I create custom OpenRL nodes?

Yes. There are two ways of creating custom OpenRL nodes or filters:

1. A "macro node" can be created, allowing a complete OpenRL node tree to be grouped into a single OpenRL node. Individual control pins within the macro can be defined as constants, or flagged as being macro-level control pins.
2. A new function can be created with an OpenRL-compliant C interface, and published as an OpenRL node with appropriate XML metadata.

Can I make OpenRL functions look like DirectShow filters?

Yes, an entire OpenRL node tree can be encapsulated in a DirectShow™ wrapper and integrated into a DirectShow filter graph.

Is the OpenRL API written in C, or C++?

For maximum compatibility, the OpenRL API is written in C. However, as the node-based architecture of OpenRL is inherently object-oriented, OpenRL functions can be called directly from C++.

What image types and resolutions are supported?

OpenRL is fully resolution-independent.

OpenRL allows the user to specify a number of image sizes and formats:

- RGB, RGBA, Greyscale, Greyscale + Alpha
- 8, 10, 12 and 16 bits per component

The default OpenRL image types have multiple components packed into an integer number of bytes.

How do I convert images from my non-linear editing format?

Where the target input and output formats do not correspond to any of the OpenRL image formats, special interface nodes can be created to translate the image formats at the boundaries of the node tree.

For example, custom OpenRL nodes can be created to take images from a third-party application such as Adobe AfterEffects, process the images using the OpenRL node tree, and return translated images back to AfterEffects.

The OpenRL core image processing library from Aspex will include several colour space conversion functions to assist with format translation.

What platforms will OpenRL be supported on?

Although the OpenRL API is platform-independent, the library functions and scheduler are platform-specific. Initial supported platforms will be Linux (RedHat / Mandrake) and Windows XP, with Max OS-X support planned for late 2005/early 2006.

What hardware acceleration platforms can I use with OpenRL?

The initial release of OpenRL will include a core image processing library ported to Aspex's Accelera 3000 platform.

What is the Accelera 3000 platform?

The Accelera 3000 is a full-length PCI-X plug-in card with four Aspex Linedancer™ Extreme Processors and 1GB of SDRAM. With 16,000 parallel processing elements, Accelera offers significant performance increases over single-processor or dual-processor machines.

Unlike fixed-function hardware accelerators, the Accelera board is completely software programmable, allowing multiple processes in the workflow to take advantage of the high performance of Aspex's Linedancer processors.

Linedancers are programmed in 'C', with a small number of extensions to support parallel processing.

An introduction to OpenRL

Platform-independent API for hardware acceleration



What does the OpenRL scheduler do?

The initial release of the Aspex OpenRL scheduler will execute the OpenRL node tree on the hardware resources specified by the programmer/user, i.e. the Accelera 3000 platform, or the host CPU.

For accelerated nodes, it automatically takes care of image transfers from the host machine to the Accelera 3000 card and back again when needed. Where several accelerated nodes are executed sequentially, accelerated code is executed in sequence on the Accelera card to minimise data transfer bandwidth requirements.

Future releases of the Aspex OpenRL scheduler will optimise system performance by intelligently allocating tasks to the appropriate resources based on the projected execution time and image transfer time. For example the scheduler may decide not to accelerate a simple function if the data transfer overhead would negate the benefit of the acceleration.

What functions are implemented in Aspex's core OpenRL library?

Aspex's OpenRL core image processing library consists of the following functions:

Convolution Filters	Generic 1D FIR filter Generic 2D convolution filter (3x3, 5x5, 7x7, nxn, nxm)
Sharpen & Blur	Gaussian Blur, Sharpen, Unsharp
Edge detection	Sobel, Roberts, Prewitt, Frei-chen, Laplacian, Laplacian of Gaussian
Pixel processing	Over compositing (Alpha blend), Add, Sub, Mult, Div, AND, OR, XOR, Invert
Log-Lin conversion	Conversion to and from SMPTE-268M 10-bit logarithmic film colourspace and 16 bit linear colourspace
Colour space conversion	Lookup-table colour correction, RGB ↔ YUV conversion, RGB ↔ HSV conversion
Image transforms	Fast Fourier Transform, Discrete Cosine Transform, Discrete Hadamard Transform

What OpenRL functions will be available from Imagineer Systems?

Imagineer will be licensing several high-value OpenRL-compliant functions, accelerated on the Aspex Accelera 3000 platform. These functions are ideal for developers of high-value video effects systems:

Pixellate	Proprietary feature obfuscation node for identity concealment, etc.
Correlation	An image correlation function, suitable for motion tracking applications
Motion Blur	A proprietary motion blur operator with variable blur per pixel, computed from a homography or flow image. Camera distortion may also be specified.
Retimer	A proprietary motion-based retimer. This can also be used with interlaced video for frame-rate conversion
Temporal resizer	A proprietary up/down resolution converter, using information from other frames in an image sequence to achieve ultra-high quality results
Interlace / Deinterlace	Proprietary motion-based interlacing & de-interlacing
Chroma Keyer	A proprietary chroma keyer
Warping	Projective Warp – translation, rotation, scaling, shear & perspective.

Will there be non-accelerated versions of these functions?

Yes, we will also be developing reference implementations of each of the nodes which will execute on the PC processor.

An introduction to OpenRL

Platform-independent API for hardware acceleration



When will OpenRL be available?

The initial release of OpenRL with the core libraries as described is planned for October 2005.

Can I contribute to the development of the OpenRL standard?

Yes, we would welcome contributions and feedback to the OpenRL development effort. As OpenRL is an open standard, contributors must be prepared for all contributions to become public domain.

Aspex Semiconductor Ltd, Rapid House, 40 Oxford Road, High Wycombe, Buckinghamshire, HP11 2EE. United Kingdom.

Tel: +44 (0) 1494 558121 Fax: +44 (0) 1494 558016 www.aspex-semi.com enquiries@aspex-semi.com

Imagineer Systems Ltd, The Surrey Technology Centre, 40 Occam Road, The Surrey Research Park, Guildford GU2 7YG, UK

Tel +44 (0)1483 685585 Fax +44 (0)1483 685586 www.imagineersystems.com

Aspex, Linedancer, Accelera, and the Aspex logo are trademarks of Aspex Semiconductor Limited.

All other names, products, and services mentioned are trademarks or registered trademarks of their respective holders.