

THE ECOS RTOS LETS YOU QUICKLY SET UP A NO-COST
EMBEDDED-SOFTWARE-DEVELOPMENT ENVIRONMENT.

A free RTOS for any embedded device

THE OPEN-SOURCE eCos (Embedded Configurable Operating System) is ideal for deeply embedded systems in which memory resources and real-time execution constraints are top design issues. The system supports a range of development platforms—from standard PCs to PDAs—using many of the most popular embedded processors. Comprehensive device drivers, supported platforms, and third-party software offerings make eCos ideal for any embedded device.

The royalty-free eCos RTOS, a good choice for embedded applications, supports Gnu open-source development tools, and, because eCos is open-source you can download and test eCos for free (Reference 1). A new release, Version 2.0, is due out in the coming months. You can find the latest information about new ports for eCos and other related information on the eCos Web site, <http://sources.redhat.com/ecos>.

eCos works in real-time applications and provides such features as pre-emptable tasks with multiple priority levels, low latency-interrupt handling, multiple scheduling policies, and multiple synchronization methods. The core functions also include exception handling, timers, counters, device drivers, memory management, and ISO C and math libraries. It provides a complete development and debugging environment that includes Gnu-based compilers, assemblers, linkers, debuggers, and simulators. Configuration and building tools are available to run on either a Linux or a Windows host environment.

The system supports a range of 16-, 32-, and 64-bit processor architectures, including ARM, Fujitsu FRV, Hitachi H8/300, Intel x86, Matsushita AM3x, MIPS, NEC V8xx, PowerPC, Samsung Calm-RISC16/32, SPARC, SPARClike, and SuperH. Figure 1 shows the overall system architecture of some of the components of the modular, highly configurable eCos.

common interface to the processor architectures and board platforms that eCos supports. It allows the higher software layers to have a common interface to the hardware, regardless of the hardware implementation.

The HAL comprises architecture, variant, and platform submodules. The architecture contains the processor-specific implementation, such as code for CPU start-up, interrupt handling, and other functions specific to the instruction set of the processor architecture. A variant is a specific processor within the processor family. For example, the MPC860 and the MPC850 are two variants within the PowerPC processor architecture. The variant may contain code for specific on-chip peripherals, such as an MMU or I/O channel. The platform contains the implementation for a piece of hardware that includes the processor architecture and variant. The platform submodule includes code for board start-up and chip-selection configuration.

Although many board ports are available as baselines for your own project, you will eventually need to port eCos to your own hardware. Various sources

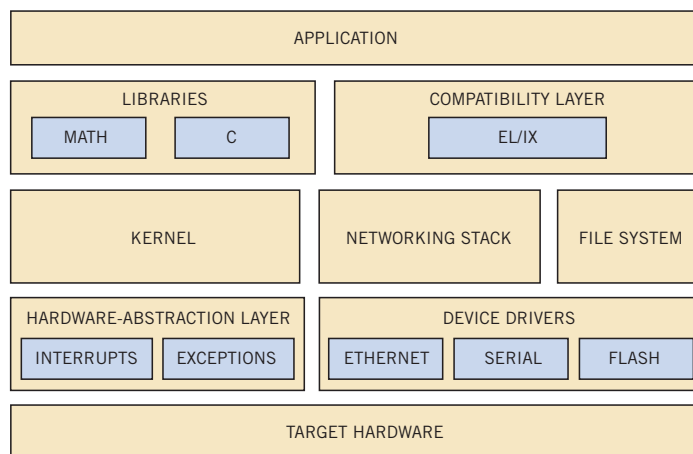


Figure 1

The overall system architecture of some of the components of eCos is modular.

A COMMON INTERFACE

The HAL (hardware-abstraction layer), which resides at the level closest to the hardware, provides a

of information can help you with this process (Reference 2).

AT THE KERNEL

The heart of any RTOS is the kernel. The eCos kernel provides selectable scheduling policies along with mechanisms for thread synchronization, exception handling, interrupt handling, counters, and clocks.

The scheduler operates in bit-map and multilevel-queue modes. The bit-map scheduler represents each thread with a bit in a bit map. Each thread must have a unique priority. The multilevel-queue scheduler allows threads to exist at the same priority level, and threads of the same priority are time-sliced, giving each thread an opportunity to run.

eCos uses a split interrupt-handling scheme, which splits interrupt processing into the ISR (interrupt-service routine) and the DSR (deferred-service routine). The ISR contains the bare minimum processing for the interrupt, such as acknowledging the interrupt. The DSR handles other interrupt processing, such as signaling a semaphore for a thread to begin processing data that was just received. This scheme maintains low interrupt latency. The kernel provides various mechanisms for thread synchronization, including mutexes, semaphores, condition variables, flags, and message boxes.

eCos offers a variety of device drivers that include serial, Ethernet, flash ROM, USB, and PCMCIA. An I/O API performs basic functions, such as sending and receiving data and configuring the device itself.

LINUX-COMPATIBILITY SUPPORT

EL/IX is an API based on Linux (Reference 3). It enables portability across Linux, embedded Linux, and RTOSs. This compatibility preserves the investment in software development and developer knowledge. The EL/IX application-level standard does not attempt to provide any device-driver compatibility.

Red Hat has initiated the EL/IX API as an open-source project. Other companies involved with EL/IX include Intel, MIPS,

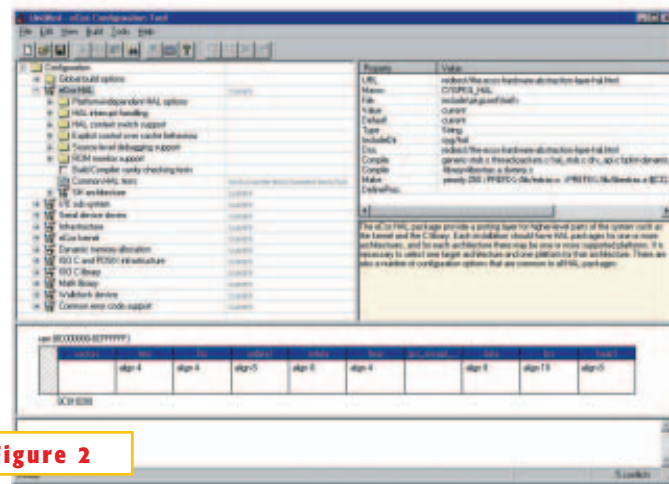


Figure 2

The configuration tool displays a graphical representation of the software packages available in the eCos system.

Toshiba, and Pacific Softworks. Developers can contribute to the EL/IX project by registering online at <http://sources.redhat.com/elix/form.html>.

The base function of EL/IX is the Posix 1003.1-1996 Specification (ISO/IEC 9945-1, Reference 4). Therefore, operating systems that conform to the Posix standard should be largely compatible with the EL/IX specification. The EL/IX specification includes a subset of the ISO C standard as well as some extensions from Linux/Gnu, BSD, and SYSV that you can apply to embedded applications.

EL/IX allows you to easily and quickly port applications among EL/IX-compliant operating systems and allows you to write, test, debug, and analyze embedded applications using a desktop system running Linux. You can then quickly port the application to the actual hardware platform. Third parties can also cover both the embedded Linux and the eCos markets by developing middleware products that conform to the EL/IX specification.

THIRD-PARTY SUPPORT

Along with the core functions, other eCos features extend the system's capabilities, allowing you to use it for a range of embedded applications. Examples of these functions include networking, including TCP/IP stack and SNMP support; μ tron 3.0 compatibility, SMP support; RedBoot ROM monitor, RAM/ROM file system, Journalling Flash File System Version 2 (JFFS2), PCI library, and USB slave support.

Because eCos is open-source, third-party developers contributed many of these features. Other third-party contributions include the GoAhead Web server, Microwindows Port, eSOAP (Embedded SOAP), Kaffe Java Virtual Machine, Bluetooth Stack, and Wireless Application Protocol Stack. You can search the eCos discussion list at <http://sources.redhat.com/ml/ecos-discuss> and the eCos Web site for the latest contributions and ports (Reference 5).

HOST DEVELOPMENT TOOLS

You can run the host development tools, including the eCos building tools and the Gnu embedded-development tool chain, under Windows (using Cygwin, www.cygwin.com) or Linux. One of eCos' key features, as its name implies, is its configuration system. The configuration system gives you control over the functions and features that the eCos image will include at runtime. For example, if you need networking support, you can layer the TCP/IP networking package on top of the Ethernet driver to provide that function. If the application needs no networking support, you can eliminate these packages and conserve system resources. The eCos configuration system is ideal for software components that third parties develop to enhance its features.

You can build eCos either from the command line or by using the graphical configuration tool (Figure 2). The configuration tool displays a graphical representation of the software packages available in the eCos system. Using the configuration tool, you can tailor components for your application, such as configuring the serial ports it uses for debugging communications or eliminating unused packages. This feature allows you to build an application-specific image of the eCos RTOS and, therefore, prevents you from wasting resources with unwanted software modules.

Along with the ability to configure and build the eCos image, the configuration tool also allows you to run tests on the target hardware platforms. You can adapt

these tests, which the eCos source repository includes, to support your production-testing needs.

Details of setting up the configuration and Gnu development tools to build the eCos RTOS for particular hardware targets are available online at <http://sources.redhat.com/ecos/getstart.html>.

ECOS SUPPORT

Additional support is available from a range of sources, including <http://sources.redhat.com/ecos/docs-latest>. Another source of information is the eCos discussion mailing list. An announcement list is available online at <http://sources.redhat.com/ml/ecos-announce>. Detailed development questions about eCos reside online at <http://sources.redhat.com/ml/ecos-discuss>. You can find other mailing lists, containing different information, at <http://sources.redhat.com/ecos/intouch.html>.

The discussion list is a way to obtain answers from developers using eCos and going through the same problems that

you are encountering. Many of these developers are glad to lend support to fellow developers. You may subscribe to either of the mailing lists to receive posts directly. To limit e-mail traffic, you may want to subscribe to a digest version of the mailing-list posts. Additional information about the mailing lists is online at <http://sources.redhat.com/ecos/intouch.html>.

To stay on top of the latest changes to the eCos source code, you need to access the anonymous CVS (Concurrent Versions System) system. Use the source-code repository to make sure that you have the latest version of the eCos source code. You can access the repository using a number of free CVS clients. Additional information about anonymous CVS access appears online at <http://sources.redhat.com/ecos/anoncv.html>.

Along with the vast number of third-party contributions, eCos provides the core functions necessary to meet the specifications of any embedded device. The open-source nature of eCos, and the

Gnu development tools enable you to quickly set up a no-cost embedded-software-development environment. □

REFERENCES

1. <http://sources.redhat.com/ecos/getstart.html>.
2. Massa, Anthony J, "eCos Porting Guide," *Embedded Systems Programming* magazine, January 2002, pg 34.
3. <http://sources.redhat.com/elix>.
4. <http://standards.ieee.org/regauth/posix>.
5. <http://sources.redhat.com/ecos/contrib.html>.

AUTHOR'S BIOGRAPHY

Anthony J Massa works as a senior software engineer. He has been designing and developing embedded software since 1994. He holds a BS in electrical engineering. This article is adapted from his upcoming book Embedded Development with eCos, which Prentice Hall will publish this fall. You can reach him at amassa@san.rr.com.