

POSIX Conformance

LynuxWorks responds to the need for interoperability between systems



More than ever, organizations in all industries need ways to ensure that their software systems can function across multiple platforms. In response to this requirement for interoperability, the POSIX® standard was born.

What is POSIX?

POSIX—the Portable Operating System Interface—is a family of standards designed to ensure source-code portability of application programs across hardware and operating systems. It is increasingly mandated for commercial applications and government contracts.

LynxOS onboard

Providing optimal interoperability between safety-critical systems, the POSIX-conformant LynxOS® family of real-time operating systems is the natural choice for mission-critical defense systems development, and boasts more than 20 years of experience supporting military and government customers.



Because POSIX conformance assures code portability between systems, it is increasingly mandated for commercial applications and government contracts. For instance, the United States Joint Technical Architecture-Army (JTA-A) standards set specifies that conformance to POSIX is critical to support software interoperability.

In addition, to ensure future system interoperability and to support software reuse, the U.S. Navy Open Architecture (OA) guidelines can require that software systems be strictly conformant to POSIX profile 54. The Navy OA initiative employs commercial-off the-shelf (COTS) POSIX-conformant products as a means for providing portable software that can be used across Naval surface, subsurface and air platforms. POSIX-conformant LynxOS has passed Open Architecture Computing Environment (OACE) standards and is Navy OACE Category 3-compliant.

The POSIX standards provide for communication between an application and the underlying operating system. POSIX was developed by the Institute of Electrical and Electronics Engineers (IEEE) and is recognized by the International Organization for Standardization (ISO) and American National Standards Institute (ANSI).

Who needs POSIX?

POSIX conformance directly benefits professionals who:

- Develop applications where portability and compatibility is an objective

- Buy hardware and software systems
- Manage companies that are deciding on future corporate computing directions
- Implement operating systems

When an application is based on a POSIX-conformant operating system, the future costs of adding new features or porting of the source code are reduced. Code written for one POSIX operating system will generally port easily to another, including most UNIX® and Linux® systems.

In this way, POSIX makes developers' jobs easier, and organizations get to preserve their software investment into the future when upgrades become necessary.

POSIX also allows for systems to be designed with modules that run different POSIX-based operating systems. For example, LynuxWorks' BlueCat® Linux is often used with the POSIX-conformant LynxOS real-time operating system.

What makes a POSIX operating system?

A savvy engineer just needs to ask a few quick questions about an operating system in order to determine if it might fully support the POSIX standards.

1. Does each process in the operating system reside in a different name space and have its own symbols table?
2. Does the operating system support the *fork()* call?
3. Can the operating system distinguish between threads and processes?
4. Does the operating system support signals?

POSIX real-time and threads extensions

LynxOS real-time operating system serves as foundation software for millions of mission-critical applications worldwide.

LynxOS brings POSIX.1 conformance to these applications, and it also supplies them with all the services specified by POSIX 1.b (real-time extensions) and POSIX 1.c (threads extensions).

The POSIX real-time and thread extensions are later additions to the original POSIX.1 standard, and they

have extensive applicability for real-time and embedded systems.

The real-time extensions include priority scheduling, real-time signals, clocks and timers, semaphores, message passing, shared memory, asynch and synch I/O and memory locking. The threads extensions include specifications for thread creation, control, and clean-up; thread scheduling; thread synchronization and signal handling.

POSIX processes

In POSIX, each executing instance of a program is called a process, and is

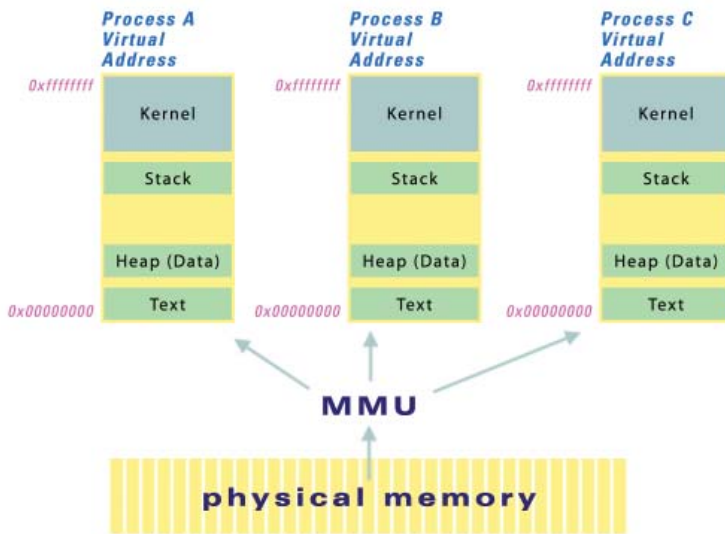
created when a parent process invokes the *fork()* call, or when a process is spawned. Each process by definition has its own protected address space. Processes are kept separate through the use of memory protection and name spaces.

Every process owns one or more threads, each of which may be instructed to execute program code. The program code may contain system calls that create additional threads. Each thread is a flow of control within the parent process. The Memory Management Unit (MMU) is used to physically isolate processes from each other so that they cannot inadvertently infringe on each other. Due to this enforced separation, communications between processes can only take place by using kernel services.

POSIX threads

An important POSIX concept is the distinction between threads and processes. Not all operating systems are able to make this important distinction. Those which do not are likely to categorize all threads and processes together as "tasks."

POSIX threads are the schedulable entities that run within each process. Each process will be the parent of one



POSIX and embedded systems

The POSIX family of standards includes many individual specifications and extensions for operating system services. POSIX.1, .1b and .1c are especially relevant in real-time and embedded systems. The POSIX.13 profiles 51-54 allow further specialization of these standards to fit the needs of simple or complex system computers.

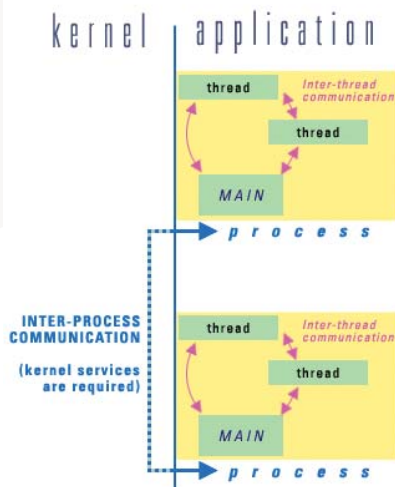
POSIX.1: Operating System Core Services	Basic operating system interfaces; includes support for process creation and control, signals, floating point exceptions, segmentation violations, illegal instructions, bus errors, timers, file and directory operations, pipes, C library (standard C), I/O port interface and control.
POSIX.1b: Real-Time Extensions	Functions needed for real-time systems; includes support for priority scheduling, real-time signals, clocks and timers, semaphores, message passing, shared memory, asynch and synch I/O, and memory locking.
POSIX.1c: Thread Extensions	Functions to support multiple threads within a process; includes support for thread creation, control, and clean-up, thread scheduling, thread synchronization, and signal handling.

POSIX conformance is worth more than POSIX compliance

main thread, but it may be the parent of several more threads as well.

The various threads running within a process all share the virtual address space of the parent process and do not have a parent-child relationship with each other. Unlike POSIX processes, threads can share data and communicate with each other by using globals.

POSIX system calls will refer to a specific process ID or thread ID—these IDs are not interchangeable.



POSIX signals

Signals are integral to any UNIX or Linux application and are supported by the POSIX.1 standard.

POSIX.1 signals are used for many synchronous and asynchronous notifications, such as terminating a child process or informing a process that it has issued a memory violation. Signals can be caught, ignored, blocked, unblocked, handled, and more.

Signals in POSIX are quite powerful. Each thread can block incoming signals on a per-signal basis and define signal handlers for each signal that it might receive.

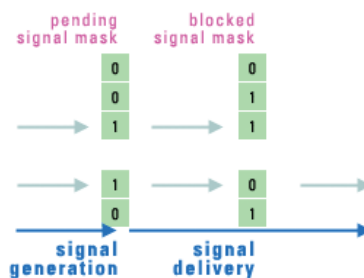
All POSIX operating systems do not implement POSIX standards to the same degree.

POSIX *conformance* is what embedded developers are usually looking for. POSIX conformance means that the POSIX.1 standard is supported in its entirety. In the case of the LynxOS real-time operating system (RTOS), the specifications of the POSIX.1b and POSIX.1c subsets are also supported.

POSIX *compliance* is a less powerful label, and could merely mean that a product provides partial POSIX support. "POSIX compliance" just means that documentation is available to show which POSIX features are supported and which ones are not.

POSIX is structured as a set of optional features, so operating-system vendors don't need to support all possible POSIX features in order to claim "compliance."

The real-time signals that are defined by POSIX 1.b are even more useful, in that they can carry data, be queued (thus guaranteeing delivery to a specific thread), and be prioritized.



POSIX.13 profiles

The POSIX.13 family of four related real-time profiles (51 through 54) covers applications from the very small-scale to full-featured.

POSIX.13 profile 51 defines a very simple real-time device, with a single

process and multiple threads, but no file system.

Profile 52 defines the addition of a file system, profile 53 adds multiprocessing capabilities; and profile 54 adds both.

Computers following the larger POSIX.13 profiles, such as 54, might operate at the top of a distributed system, with simpler computers at the bottom running smaller POSIX.13 profiles.

The Linux operating system is considered to be of profile 54, so Blue-Cat Linux is frequently used with the LynxOS real-time operating system, both during development and prototyping phases, and as part of larger systems.

LynxOS : The only certified POSIX-conformant RTOS

The LynxOS real-time operating system (RTOS) from LynuxWorks has the most inherently secure, reliable design of virtually any commercial-off-the-shelf (COTS) embedded operating system today, and is the most open hard RTOS available.

LynxOS has supported a full POSIX process model since its introduction in 1988.

Best of all, POSIX is the natural interface for LynxOS, so POSIX calls are not an optional add-on library.

LynxOS is the only RTOS certified by the IEEE as POSIX-conformant. In addition to the POSIX.1 specification,

LynxOS supports all of the routines in POSIX.1b and POSIX.1c. subsets (real-time and threads extensions).

Full POSIX conformance and deterministic performance makes LynxOS ideal for real-time systems intended to:

- Execute complex series of tasks within set periods of time
- Support multiple applications with multiple interrupting devices
- Take full advantage of today's powerful high-end microprocessor and advanced networking architectures

Because LynxOS is designed from the ground up for conformance to open system interfaces, developers can

leverage existing Linux, UNIX, and POSIX programming talent for embedded real-time projects.

LynxOS not only conforms to POSIX standards, but it is also the only RTOS that provides true Linux ABI-compatibility. Linux ABI-compatibility means that applications written for Linux can easily run on LynxOS with no loss of functionality and little time and effort involved.

With the LynuxWorks solution, developers maximize the value of their code investment and improve their time to market.

POSIX conformance at work

The Improved Data Modem (IDM) from Innovative Concepts, Inc. (ICI) can interconnect the U.S. Army and the U.S. Air Force's major networks for both maneuver and fire support, and also provide critical linkage to the military's legacy systems.

Plans to add a subset of FCBC2 (Force XXI Battle Command, Brigade and Below) software to the IDM were threatening to require physical changes to the IDM in order to accommodate the FCBC2's Solaris™ operating system.

Instead, it was decided to port the FCBC2 software to a different operating system altogether.

The first porting attempt was to a non-POSIX operating system—VxWorks®.

Since VxWorks was already being used in the IDM, porting the FCBC2 software to it seemed a logical approach—at first.

Unfortunately, the lack of POSIX conformance proved to be a significant disadvantage, and it ultimately contributed to the failure of the attempted VxWorks port, despite the three years that had been spent on that effort.

Finally, the earlier project team cut its losses and turned to the LynxOS real-time operating system (RTOS), which is POSIX-conformant, therefore providing the UNIX compatibility that was important for the software porting.



In fact, in just six short months, the port to LynxOS succeeded as expected. "We knew it would not be difficult to port the FCBC2 code to LynxOS," said Bob Woodward, director, tactical communication systems at ICI. "And it wasn't."



LynuxWorks, Inc.
855 Embedded Way
San José, CA 95138-1018
408.979.3900
408.979.3920 fax
www.lynuxworks.com

LynuxWorks Europe
Craven House
121 Kingsway, Holborn
London WC2B 6PA
United Kingdom
+44 208 906 9506
+44 208 906 2338 fax

©2008 LynuxWorks, Inc. LynuxWorks and the LynuxWorks logo are trademarks, and LynxOS and BlueCat are registered trademarks of LynuxWorks, Inc. Linux is a registered trademark of Linus Torvalds. All other trademarks are the trademarks and registered trademarks of their respective owners.

All rights reserved. Printed in the USA.