

ENEAA OSE® 5.5



High-Availability RTOS for Complex, Distributed Systems

Enea OSE® is a compact, robust, high-performance real-time operating system optimized for distributed, fault-tolerant multi-processor systems requiring true deterministic real-time behavior and “five nines” or higher availability. Utilizing a modular, scalable, microkernel architecture, Enea OSE shortens development time, enhances reliability and reduces life time maintenance costs for a wide range of systems, from mobile phones and automobiles, to medical instruments and telecom infrastructure.

Enea OSE® employs a distributed software architecture that makes it easy for programmers to conceptualize, partition, and develop complex applications, whether they are deployed on a single CPU or distributed across multiple CPUs, blades and shelves. This communications driven architecture, utilizing asynchronous, direct message passing, encourages a consistent programming model that leads to development of compact, modular code, which in turn speeds development time and makes long term maintenance much more cost effective. In fact, Enea OSE’s message-passing architecture provides some unique advantages for both multicore and distributed systems development:

- Inherently modular, distributed architecture
- Simple model that is intuitive and easy to maintain for developers
- Consistency in application design and long term maintenance
- Highly scalable

Enea OSE was designed for applications demanding the utmost in reliability and high availability. Enea OSE’s memory protection facilities enhance security and reliability by preventing errant or malicious applications from crashing the kernel and other applications. Enea OSE’s built-in task (or, in Enea OSE terminology, “process”) monitoring and advanced error detection and handling

not only simplify application development and debugging, but make distributed systems easier to test, upgrade, and certify. Support for dynamic run time loading of applications in the field ensures application hot swapping and in-field upgrading without planned downtime. This combination of capabilities helps ensure that OSE meets “five nines” high-availability requirements for telecommunication applications.

But scalability, maintainability, reliability, and high-availability don’t mean a compromise in developing, debugging, and analyzing application code. Enea OSE is fully supported by the Enea® Optima tools suite, featuring an Eclipse-based Integrated Development Environment, system-level browsing, analysis, profiling and system simulation capabilities that greatly simplify the debugging and optimization of complex distributed applications.

Modular Layered Microkernel Architecture

Enea OSE’s modular, layered architecture simplifies configuration while enhancing, scalability, and reliability. The Enea OSE microkernel provides the foundation for the Enea OSE RTOS, providing basic services such as preemptive priority-based task management, direct, asynchronous message passing for intertask (inter- process) communication and synchronization, and error

PRODUCT FEATURES

- Designed for fault tolerant, distributed systems
- Modular, layered microkernel architecture
- Event-driven, deterministic real-time response
- Simple, intuitive, asynchronous direct message-passing model
- Scalable hybrid multicore solution – exploiting the advantages from both SMP and AMP models
- Memory protected
- Advanced error handling and remediation
- Built-in task (process) monitoring and failure detection
- Dynamic, run-time program loading
- Power management with low-power sleep mode
- Demand paging support for optimizing RAM usage
- Comprehensive networking/security support
- Multiple file system choices including a crash-safe, journaling file system
- Distributed system-level simulation
- Eclipse-based integrated development environment and tools suite

handling. The microkernel also provides memory management facilities that enable Enea OSE to take advantage of MMU hardware for memory protection.

ENEAA OSE® 5.5



Built atop the Enea OSE microkernel is the core basic layer, which offers file system and networking services, device driver management, debug, and extensive C/C++ runtime support. The core extension layer offers optional services such as file managers, networking stacks, and dynamic run-time loaders for upgrading and hot swapping application software. The core extension layer also offers support for the message-based Enea® LINX interprocess communications software, which enables programmers to establish reliable communications between processes running on different processors and/or cores.

The minimal standard configuration for the Enea OSE operating system is 350 kbytes, which includes the full kernel services layer and core basic services layer. However, users can also realize smaller configurations by excluding portions of the core basic services layer.

Simple Intuitive Programming Model

In Enea OSE, the mechanisms for interacting with processes and even whole programs (i.e., creating, locating, communicating with and supervising) are identical, whether the tasks are running on the same node, processor, or core. This consistency and transparency simplifies development by enabling programmers to view applications distributed across multiple nodes as a single logical system image, regardless of the network topology.

Enea OSE's transparency also reduces application complexity by making it

easier for programmers to assign specialized roles for individual processors or cores. For example, in a widely distributed communications system, it may be desirable for each node in a cluster to support a different function or role. Because processes can transparently access services located on any node in the cluster, programmers do not have to replicate services (a file system or TCP/IP stack for example) across multiple nodes. This transparency enhances scalability and promotes software reuse, enabling designers to add new nodes with specialized capabilities without having to replicate those capabilities and alter application code residing on other nodes.

Interprocess Communications

Enea OSE supports a variety of traditional mechanisms for interprocess communications and synchronization. But it is direct message passing that makes Enea OSE a true distributed operating system. Direct message passing is the simplest, safest, most flexible method for conducting communications between two processes. It also provides transparency, enabling processes to communicate in a seamless fashion regardless of where they physically reside (i.e., the same or different CPU or memory space).

To boost performance, Enea OSE enables processes to communicate directly with each other on a peer-to-peer basis, without having to synchronize through intermediate mechanisms such as mailboxes and semaphores. Enea OSE also enhances performance and simplifies programming by selecting

the most efficient method for message delivery. For example, when passing messages between processes in the same memory space, the kernel passes message pointers rather than copying data contents between processes.

Enea OSE handles all of the house-keeping associated with message passing (such as queue management). This simplifies application programming, improves reliability and enables applications to be scaled with few if any changes to the application code.

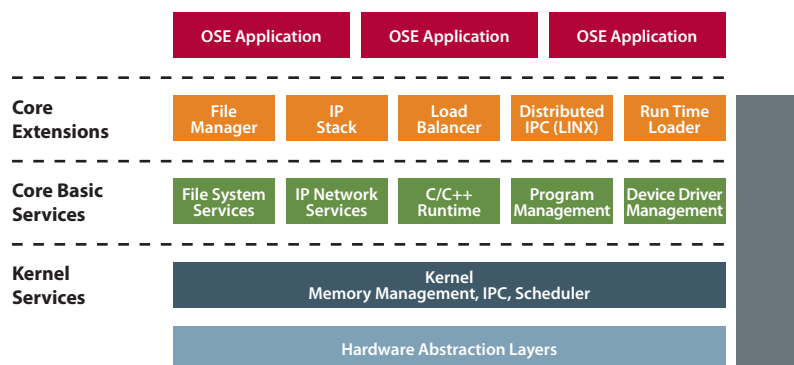
Deterministic, Fully Preemptive Real-Time Microkernel

The Enea OSE microkernel delivers high performance with bounded response times, making it ideal for hard real-time applications. The Enea OSE real-time kernel is fully preemptive, and can service interrupts at any time, even during execution of a system call. All time-critical parts of the kernel are highly optimized, and all kernel execution times are deterministic, independent of the size of the application, memory consumption, or the number of processes.

Memory Protection

Enea OSE can utilize the target processor's MMU to provide protected memory spaces (flash or RAM) for the Enea OSE kernel (supervisory mode) and application processes (user mode). These protected spaces enhance reliability and security by preventing malicious or errant processes in one protected space from corrupting the kernel or application processes in other protected spaces, potentially crashing the entire system. They also:

- Simplify debug by making it easier to isolate faults and flag error conditions such as illegal memory and wild pointer accesses.
- Simplify the design of fault-tolerant systems by providing isolation between recovery units.
- Facilitate safe run-time upgrades by enabling programs from external sources to be loaded into protected memory areas.



Modular Layered Microkernel architecture.





Process Supervision

Enea OSE enhances reliability and robustness by providing built-in supervision for designated processes, programs and nodes. Applications can request supervision for both local and remote processes. When supervision is requested, Enea OSE automatically notifies the application when the supervised process is no longer available (for instance, when a remote system crashes, or a link failure occurs) or becomes available again. This automatic supervision and notification increases availability by enabling systems to identify problems before they become fatal client requests, and by facilitating dynamic component replacement.

Unified Error Detection and Error Handling

Enea OSE's centralized error handling simplifies application development and enables developers to produce clean, readable, compact code that runs faster, consumes less power, is more reliable, and is easier to debug. Unlike conventional RTOSes, Enea OSE does not simply return an error code to the user application when an error event occurs. It invokes an error handler directly from the kernel. This approach eliminates the need for developers to write code that constantly checks error codes interspersed with the application code, resulting in cleaner, easier to read code that is faster to develop and easier to maintain.

In addition, Enea OSE utilizes an error escalation sequencing algorithm as part of the core kernel design. This escalation sequence attempts to resolve errors sequentially from the process, memory block, and system levels. This escalation sequence provides an opportunity to manage errors at various stages of criticality before assuming that the system must be reset or other "last ditch" error management facility is activated.

Multicore support

The increasing demand for higher data rates, drives the need for performance. Moore's law on the exponential increase of number of transistors per integrated circuit is still valid, but due to the also exponential increase of power

consumption, the computer industry has been forced to add more processor cores to the computer chips rather than increasing the CPU frequency.

The advent of Multicore devices introduce a technology shift which affect both the Operating Systems and the Applications. To fully utilize the multiple cores and the processing power that becomes available is both an opportunity and a challenge:

- the Operating System must deliver application support with regard to performance, debug support and scalability
- the Applications must be designed for parallelism – however, most applications are currently designed to be serial
- Amdahls law is used for calculating the performance gain when a program is executed in parallel

The most important features of multicore are:

- **Increased performance:**
The additional performance gain is through added core(s)
- **Decreased power consumption:**
Reduces both the heat dissipation and the power cost
- **Decreased footprint:**
Processor or functionality consolidation to reduce the Bill of Material (BOM)

Based on the Enea OSE micro kernel, OSE Multicore Edition has introduced a new innovative multicore kernel. The kernel is a hybrid solution, providing a homogeneous single OS instance (Symmetric Multiprocessing – SMP on application level), and multiple kernel scheduler instances, one per core (Assymetric Multiprocessing - AMP on kernel level).

The principle of Enea OSE Multicore Edition is based on programs (Load Modules in Enea OSE), meaning that:

- there will be true parallelism when different programs are run on different execution units, e.g. cores
- all processes within a program will be pseudo parallel like in a uncore system – The parallelism of the application can at any time be optimized by the application designer

- program instantiation can be used to further increase the performance – there can be several instances of the same program in the system

Load balancing is an important feature in a multicore system, as the load balancer distributes the workload evenly on the available cores. Enea OSE provides signal interfaces in the Program Manager to configure and collect statistical load measurements, but the actual load balancing is controlled by the application designer and not by the Operating System. Additional profiling features for load measurements exist in the run-mode monitor (RMM).

The key features and competitive advantage of Enea OSE Multicore Edition are:

- **Full backward compatibility for legacy applications:**
Enea OSE legacy code written for a single processor will still execute as expected – Existing customers can directly utilize the multicore features
- **Bare Metal Performance with SMP easy-of-use:**
To fully exploit the advantages from both SMP and AMP models

Power Management

To maximize power efficiency, particularly for mobile devices such as mobile phones, the Enea OSE kernel provides a power-save handler. When the Enea OSE kernel detects that no process is scheduled to run for a certain minimum amount of time, it can invoke this handler, which puts the CPU into a low-power sleep mode. If nothing happens during the sleep period, the processor is awakened just in time to run the next scheduled process. If a sporadic interrupt occurs while the processor is in sleep mode, the kernel awakens the processor immediately.

ENEAS OSE® 5.5



Demand Paging

Enea OSE also provides the programmer with the option of tighter control of RAM usage in applications where RAM must be used very efficiently, which is especially valuable for memory-constrained devices such as mobile phones. Enea OSE's demand paging facility allows programmers to copy read-only pages of code and data as required from cheaper NAND flash into RAM only as required to execute. At any given time, the amount of RAM being used is much less than would be required to store all of the necessary code and data to execute the entire application. This allows system designers to minimize the amount, and thus cost, of expensive RAM and replace it with much cheaper NAND flash.

Dynamic Run-Time Program Loading

Enea OSE lowers development and maintenance costs by enabling developers and field service personnel to modify and upgrade live systems. In Enea OSE, programs can be independently loaded, unloaded, started and stopped during run time without disturbing the execution of other programs. Enea OSE's memory protection facilities shield existing programs from failures that may arise due to loading and unloading of new programs. Users can decide whether to enable memory protection (if the processor is equipped with an MMU) for each program, define memory access permissions, and specify whether the program should be transient or persistent (retained throughout system restarts).

Distributed Systems Support

Enea LINX extends Enea OSE's message passing services and process supervision to multiple CPUs and operating systems, providing transparent IPC services that enable processes located on multiple CPUs and operating systems to communicate in a seamless fashion. Enea LINX is the only IPC that can support any distributed system topology, from a single processor on a single blade, to large networks with complex cluster topologies deployed on hundreds of

processors in a multi-rack system.

Enea LINX provides substantially lower latency and higher throughput than competitive message passing IPC solutions such as TCP and TIPC. This high level of performance enables Enea LINX to be used system wide for both local and remote IPC, thereby simplifying overall system design, increasing reliability and enhancing scalability.

- Works with multiple operating systems (Enea OSE, Enea OSE^{ck}, Linux, etc)
- Ideal for control and data plane communications systems
- Ideal for digital baseband modem communications between DSPs and CPUs
- Supports reliable and unreliable media
- Scales from DSPs to 64-bit CPUs; any network topology
- System-wide IPC solution. Local and remote IPC.
- Low latency, high throughput
- Small footprint (code and data)
- Open source BSD/GPL license (Linux version)

Seamless Desktop Connectivity – Enea OSE Gateway

Enea OSE Gateway extends Enea LINX-like connectivity to desktop operating systems like Windows and Solaris that lack a native Enea LINX port, where real-time response is not critical. Enea OSE Gateway provides a straightforward, consistent OS-level connection based on message passing that facilitates cross development and enables programmers to link Enea OSE processes running on the embedded system with monitoring and control processes running on the management workstation.

File System Support

Enea OSE's journaling based crash-safe file system is optimized for embedded applications. The robustness of the journaling based file system is further increased through the support of meta data checksumming. Hence, it is possible to detect errors caused by low-level software. The modular file system framework for Enea OSE, supports a variety of file handling method-

ologies, and is small enough (less than 200 kbytes) for hand-held devices, yet powerful enough for large-scale distributed systems equipped with a wide range of storage devices. The application programming interface supports both OSE messages and standard UNIX/C function calls (POSIX 1003.1 or ANSI C), making it easy to interface OSE file systems with both UNIX and Enea OSE applications.

Enea OSE's modular file system architecture provides file managers for a broad range of media (i.e., local disks, flash, and RAM banks), and can be readily adapted to new media and file formats. Enea OSE file managers can be loaded and unloaded at runtime, allowing dynamic mounting and unmounting of new file systems. Enea OSE file systems can also be distributed across multiple CPUs. The Enea OSE RTOS and applications can access and administer remote file systems just as they do local file systems.

Networking

Enea OSE provides a plug-in framework for networking protocols. Enea OSE currently supports several RFC-compliant protocol stacks: The available protocol stacks range from an ultracompact, dual-mode (IPv4 and IPv6) host stack, and full-featured, high-performance, dual-mode router stacks optimized for telecom networking systems. The router stacks are extremely scalable, thus allowing features, protocols and/or modules to be omitted at compile time to reduce memory footprint. One of the router stacks also has support for SCTP, i.e. Stream Control Transmission Protocol.

Enea OSE is available with a wide range of optional networking modules, including:

- DNS client (IPv4 and IPv6 lookups)
- PPP support
- NAT and Pack Filter support
- SNTP and NTP for clock synchronization
- IPSec and OpenCrypto support

ENEAA OSE® 5.5



- SSH for secure shell and file transfers
- RADIUS and SSL support
- MIBv2 support and SNMP agent
- Several utilities e.g. Telnet and FTP supporting both IPv4 and IPv6

Comprehensive Development Tools Suite

Enea OSE is available with comprehensive cross-development support, including an Eclipse-based Integrated Development Environment, system-level debug and analysis tools, and a simulation environment that supports a mixture of hard (RTOS) and soft (Linux, Windows, Solaris, etc.) targets.

Enea OSE provides built-in support for kernel-aware, source-code debugging, enabling programmers to examine information such as process type, status, signaling queues, and more. Enea OSE also supports higher-level application debugging, or system level debugging, which enables programmers to visualize events based on message passing and errors within the system.

GNU Compilers and Source-Level Debugger

Enea OSE is available with C/C++ GCC compiler support for all supported architectures, including the standard GCC compiler distribution with source code. Enea provides extensions to the GCC linker that support Enea OSE's dynamic program loading capability. Enea also provides source code for the GDB debugger and extensions to the GDB debugger for Enea OSE-aware source code debugging.

System Debug Tools and Integrated Development Environment

Enea Optima, available on Windows, Linux and Solaris, is a suite of powerful system debug and profiling tools for the Eclipse environment and Enea OSE. Enea Optima releases features a complete integrated development environment (IDE), including the Eclipse platform and Eclipse C/C++ development tools. Enea Optima tools allow developers, integrators and testers to browse and manage embedded distributed systems, to visualize and optimize CPU and memory usage and to capture log information from operating system and application. Common to all tools are the low intrusion run mode and post mortem mode mechanism that allow Enea Optima to be used throughout life the product.

Enea OSE Soft Kernel Environment

The Enea OSE Soft Kernel Environment provides a distributed simulation environment that allows Enea OSE real-time kernel processes to run on a Windows, Linux, or Solaris host. This capability speeds development by enabling programmers to develop and test their applications in the host environment before moving them to the target hardware.

The Enea OSE Soft Kernel Environment is the first simulation environment that can work together with a running real-time system. This enables designers to build a hybrid soft/hard Enea OSE target environment in which designers can run a portion of their application on

the host and migrate it to hard targets as hardware becomes available.

The Enea OSE Soft Kernel Environment is an API simulator that supports all Enea OSE system calls and most standard Enea OSE run-time components. It also supports virtual interrupts, key partner functions (such as networking and media processing), and includes a "soft" board support package. This system-level functionality enables applications to perform the same way on the host system as they do on the target.

Supported Platforms

- ARM consortium family: ARM 9, ARM11, ARM Cortex
- MIPS 64 consortium family (n32 ABI): Netlogic XLR and XLS processors. Cavium Networks Octeon and Octeon Plus.
- Freescale PowerPC family: MPC 5xx, MPC 5xxx, Host Processors MPC 7xxx, PowerQUICC I MPC 8xx, PowerQUICC II MPC 82xx and MPC 83xx, PowerQUICC III MPC 85xx, Host Processors MPC 8xxx. QorIQ: P10xx, P20xx and P40xx.
- IBM PowerPC family (from AMCC): 405EP, 405EXr, 405GP, 440GP, 440GRx, 440GX, IBM PowerPC family (from IBM): 750, 750CX, 750FX, 750GP, 750GX
- INTEL ARM 5 family - XScale,
- Intel Network Processor: IXP465, IXP2350
- Texas Instruments: OMAP (all chipsets, for OMAP DSP)