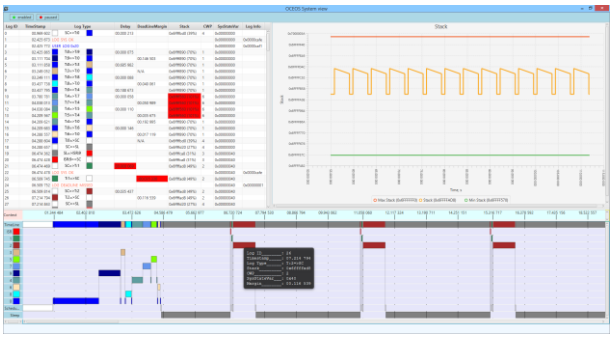
OCEOS

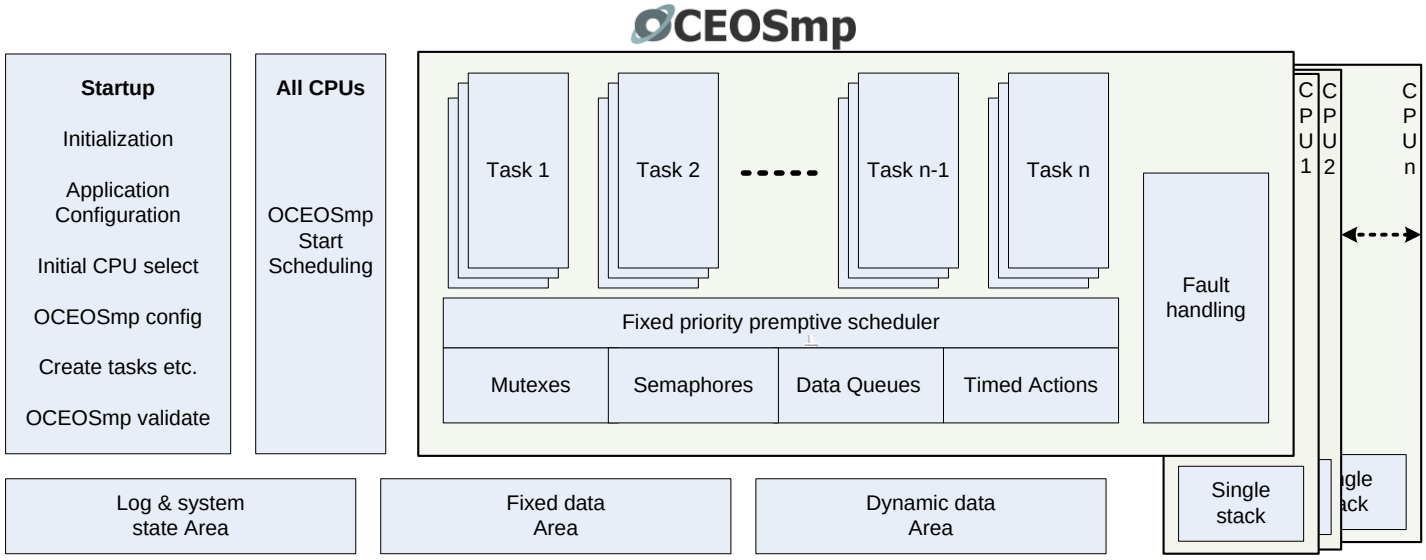
Real-Time Operating System
ECSS Category B, ISO 26262

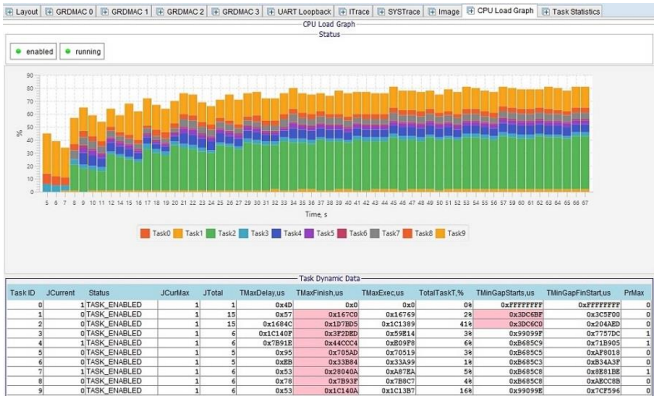
PRODUCT DESCRIPTION

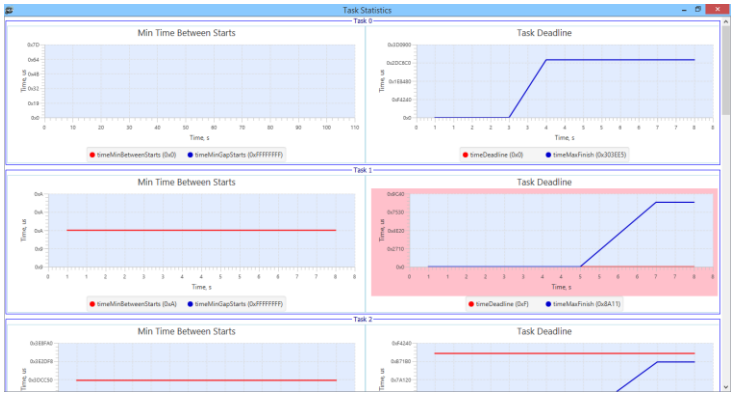
OCEOS is a real-time pre-emptive fixed priority operating system that can be used in applications that require European Cooperation for Space Standardization Category B or ISO 26262 standards. It has a small memory footprint (<10 kBytes), requires only one system stack per CPU rather than a stack for each task, and provides support for precisely timed data outputs independent of task scheduling. OCEOS supports applications running on RISC-V, ARM, & SPARC based hardware. The support of the European Space Agency in developing OCEOS is acknowledged. OCEOS provides the following facilities:



- Fixed priority pre-emptive scheduling
- Based on Stack Resource Policy - unbounded priority inversion and chained blocking cannot occur.
- Single stack per CPU rather than separate stack for each task
- Small code footprint (<10 kBytes for scheduling and mutex)
- Mutex (R/W), Counting Semaphore, and Data Queue support
- High precision timed actions (data output and task start)
- Supports SPARC, ARM, and RISC-V processor architectures
- Deadlocks prevented on single core and alerted on multicore
- DMON debug tool support (execution timeline, CPU usage)
- Support & ISVV services available from OCE
- Gold standard customer support package
- Developed to ECSS Category B and ISO 26262 standards
- Developed in cooperation with European Space Agency (ESA)







OCEOS

single-core

Description and Features

OCEOS was developed for high reliability aerospace applications. Its small size and efficiency make it ideal for use in embedded systems requiring compactness and high reliability.

Real time software is often written as a set of trap/interrupt handlers and tasks managed by a RTOS. The trap/interrupt handlers start due to anomalous conditions or external happenings. They carry out the immediately necessary processing and may ask the RTOS to start a task to complete the processing. The RTOS then schedules the task for execution based on its priority.

In hard real time systems scheduling must ensure that each task completes no later than its deadline, but being early can also be a problem. OCEOS provides a timed output service that allows a data output be set for a precise time independently of scheduling. A task also can be scheduled to start at a precise time, but the actual start time may be later depending on task priority.

OCEOS supports up to 254 tasks with up to 15 current execution instances of each task, allowing one task service multiple units of the same type. Each task has a fixed priority and more than one task may have the same priority. Tasks of the same priority are FIFO scheduled, there is no time slicing between tasks in OCEOS. In OCEOS a pre-emption threshold higher than the task's priority can be set for a task so that once it starts execution it can only be pre-empted by a task with higher priority than this threshold.

Pre-emptions and any traps/interrupts that occur will delay a task's completion and potentially cause it to miss its deadline. Careful analysis is needed to ensure that task deadlines are always met. OCEOS supports this analysis and allows relatively simple determination of worst-case behaviour. Problems such as unbounded priority inversion, chained blocking, and deadlocks cannot occur in OCEOS.

OCEOS provides mutexes to protect critical shared code or data, and inter-task communication using semaphores and queues. A system state variable provides a summary of the current state of the system. Error conditions such as missed deadlines are logged and the system state variable updated. If the system state is not normal actions such as disabling a task or resetting the system may be taken.

OCEOS does not allow dynamic creation of tasks at run time. Virtual memory is not supported. Task priorities are fixed. OCEOS is based on the Stack Resource Policy extension of the Priority Ceiling Protocol [Baker 1991]. OCEOS is provided as a library and is statically linked with an application. Services not needed by an application are omitted by the linker.

Feature	Details
Task scheduling	Fixed Priority Preemptive
Scheduling policy	Stack resource policy, unbounded priority inversion and deadlocks cannot occur
Tasks	Up to 255 tasks. Max pending start requests per task 15
Mutexes	Up to 63 mutexes each with fixed priority ceiling
Counting semaphores	Up to 63 counting semaphores, each maximum permit can be set up to 4095
Data queues	Up to 63 data queues, each max entries can be set up to 255
Timed outputs	Max 255, independent of system time
Build environment	RISC-V Microchip, ARM DS, Keil, Segger, SPARC BCC, gcc , others (ask)
Debug	OCE DMON debug tool with OCEOS extensions for analysis
Processor architecture	RISC-V, ARM, SPARC
Standards	ECSS Category B and ISO 26262 ready
Customer Support	Telephone, email, and on-site support packages

OCEOSmp

multicore

A 'white box' real-time operating system

Two main advantages arise from using a multi-core CPU, additional computational power and redundancy. OCEOSmp is designed to support both of these advantages. It makes symmetric use of the CPU cores and allows any core be removed from use if found to be faulty.

In OCEOSmp a core can be restricted to use only by tasks above a certain priority, and a task configuration can specify that the task should only run on a specific CPU core. Usually however any task is allowed to run on any core, with greatly increased throughput when many tasks are ready to run.

Hardware redundancy requires being able to remove faulty cores from use at run time and may involve bringing cores that have been kept in reserve into use. OCEOSmp supports both of these. Selected cores can be exempted from use by OCEOSmp in the initial configuration, allowing them be used by some other aspect of the application.

OCEOSmp treats all CPU cores on an equal basis after the initial start-up sequence. Context switch and other OCEOSmp code is re-entrant and can be run on any core, with critical parts protected from use by more than one core at a time. Multi-processing is symmetric once start-up is complete.

The start-up sequence depends on the CPU family involved. It may involve just one core which then starts other cores, or may involve a number of cores starting simultaneously. In the latter case the application specifies the core to be used in initialising and starting OCEOSmp. Apart from the start-up core other cores will be in a wait state until start-up is complete.

An application that uses OCEOSmp is structured as a number of relatively independent tasks each with a fixed priority from 1 to 254 (highest to lowest priority) that indicates its importance relative to other tasks. Each task also has a fixed pre-emption threshold that is at least as high priority as the task's priority, only tasks with higher priority than its threshold can pre-empt a task.

Each task is configured to allow up to a fixed number of concurrent execution instances in the range 1 to 15. Space is allocated for information on each of these 'jobs' when OCEOSmp is started.

New start requests for a task can occur before a previous job has finished execution, and if a task is configured to allow multiple concurrent executions the new job may then be put into execution on a different core rather than waiting for the previous job to complete.

OCEOSmp provides a system log and an optional context switch log. The location and sizes of these are defined in the application configuration. They are stored in the 'log area', which also holds the indexes used.

A 32-bit word containing a number of system status flags is automatically reset when OCEOSmp is started. The flags are automatically updated when OCEOSmp detects an anomaly such as task not starting due to an inadequate number of jobs. An application defined mask specifies which flags when set should lead to a user defined problem handling function being called.

Information such as the maximum execution time for a task, the number of times pre-empted, the maximum items on a data queue, the upper and lower bounds of a core's system stack and many other items is updated as OCEOSmp schedules tasks and is stored at predefined addresses.

This information is readily available to the application and allows it to check if design assumptions are coming under threat, and if so to anticipate a problem and take appropriate action, perhaps temporarily disabling a task.

OCEOSmp has been designed to minimise context switch times. Interrupts disabled times are distributed across the different CPU cores and kept to a minimum. Interrupt nesting is supported, with OCEOSmp directives acted upon only when interrupt nesting has unwound.

OCEOSmp has been designed specifically to provide a 'white box' operating system whose behaviour can be readily checked and understood. It is deterministic and compact, and allows timing behaviour be guaranteed. It is symmetric in its use of CPU cores. It is robust and provides good support for fault anticipation, detection, isolation and recovery. It allows CPU cores be excluded from use at run time and allows tasks be disable if required. It is supported by OCE's DMON debug monitor, which provides a number of advanced monitoring and debugging features including context switch logging.

For further details email or call:

Sales Department, O.C.E. Technology Ltd.,
NovaUCD, Belfield Innovation Park,
Belfield, Dublin,
D04V2P1, Ireland.

Phone: +353 1 716 3530
Email: sales@ocetechnology.com

Distributor: