

# RTOS-UH

## Real-time operating system

Ever more and more complex technical systems demand the use of computers for open and closed loop control. **RTOS-UH** supports the use of control computers by a straight concept and high real-time reactivity for fast and flexible software reaction to changes and events in the processes.

**RTOS-UH** enables software to react immediately upon external events (interrupts). All administrative operations of the operating system can be interrupted and the processor can be freed for the handling of external events. Even problems with very high reactivity requirements can be solved reliably by a control computer.

In practice, most often it is of great use to subdivide the complex task of steering a plant or controlling a process into smaller and simpler subtasks. **RTOS-UH** supports the breakdown of complex problem definitions by providing a framework to solve these subtasks individually and let these subtasks interact freely. The number of tasks in the system is only limited by the size of the memory. **RTOS-UH** allocates computing time to the individual task strictly based on their priority and allows for time sharing operation on each priority level. A very high number of priority levels allows for fine grained control of program interaction. The tasking is under full control by the user.

**RTOS-UH** is a complete realtime operating system. With its strictly modular structure, it is scalable from a small ROM-based runtime kernel up to complete workstation level with file system, network, and user interaction. Beginning by the realtime kernel the system is broadened by adding individual components. At start up, the kernel scans for additional modules and extends its functionality according to the components found.

There is no need for special configuration tools to tailor the systems scope of services to the needs. A complete development system can be build using the same run-time kernel and delivering the same runtime behavior as a dedicated target system. User interfaces – commonly only available at workstations – can be added to targets to increase their observability and servicability.

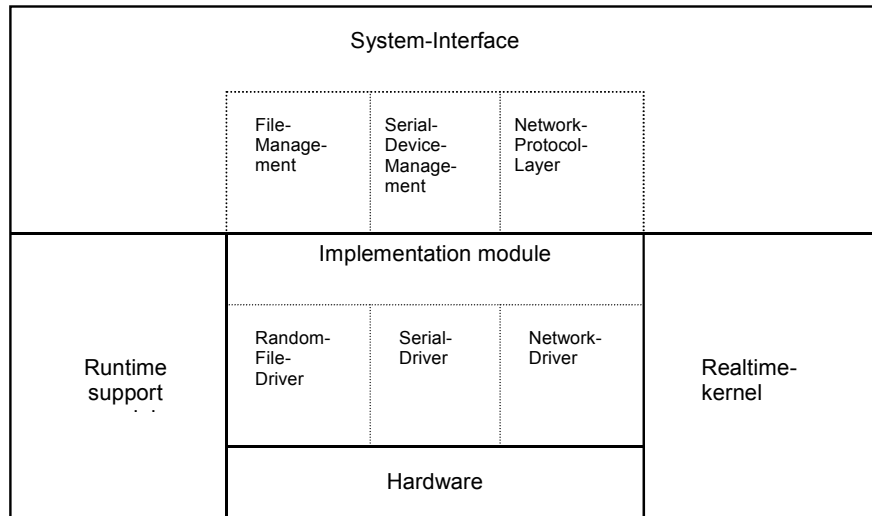
**Why  
RTOS-UH**

**Realtime**

**Multitasking**

**Programming-  
system**

---



The modular structure of the system and the automatic self-configuration of independent individual building blocks allows the system configuration even by an application designer.

## Operating system

As operating system, **RTOS-UH** manages the resources of a computer. An I/O concept based on messages and message queues, an automatic memory management and an integrated error handling serve for comfortable application programming without a large system overhead. A realtime-oriented command interface supports program development and program inspection also under realtime conditions.

## Realtime kernel

**RTOS-UH** is runnable on all processors of the PowerPC and the MC68xxx families. The assembler-coded realtime kernel provides all fundamental system services like:

- Dispatcher
- Process manager (scheduler)
- I/O management based on messages and message queues
- Memory management
- Error handling

The realtime kernel is identical in all **RTOS UH** implementations. All specific system modules as well as the resident user programs are integrated during start-up of the operating system.

## Implementation

The implementation module consists of hardware-specific initialization code and device drivers. By its modular structure, **RTOS-UH** offers

- Driver for serial and parallel interfaces
- Mass storage driver for Floppys and non removable disks
- Network driver for the deployment in LANs and WANs

All components can also be used individually and can be configured in accordance to specific requirements.

---

## Runtime support

**RTOS-UH** provides all application programming interfaces in a runtime module. These functions are reentrant, so the operating system and applications can use them at the same time. The commonly used integration of the respective drivers and arithmetic routines into the user programs is void, resulting in reduced memory requirements. Loadable and executable programs require only little space in RAM or ROM.

Systems not headed for specific applications call for an efficient command interface. With commands to show and modify the state of individual tasks, **RTOS-UH** delivers insight and control over the multitasking. At any time a detailed inspection of the systems current working state is possible.

The realtime oriented commands are borrowed from the conforming PEARL-statements:

## Command interface

### RTOS-UH realtime commands

```
WHEN event [ACTIVATE ] taskname [PRIO number];
```

```
WHEN event C[ONTINUE] task;
```

```
AT time [ACTIVATE ] taskname [PRIO number];
```

```
AT time C[ONTINUE] taskname;
```

```
AFTER duration [ACTIVATE ] taskname [PRIO number];
```

```
AFTER duration C[ONTINUE] taskname;
```

```
AT time ALL duration [ACTIVATE ] taskname [PRIO number];
```

```
AT time ALL duration UNTIL time [ACTIVATE ] task [PRIO number];
```

```
AT time ALL duration DURING duration [ACTIVATE ] task [PRIO number];
```

```
AFTER duration ALL duration UNTIL time [ACTIVATE ] task [PRIO number];
```

The testing of applications, in particular difficult when running under realtime constraints on a multitasking system, is substantially simplified by an integrated trace facility and lots of low-level debug tools.

The system programming interface provides a uniform software interface to applications und service tools. Build upon fundamental services delivered by the implementation modules, the system interface integrates higher level administrative tasks and provides a hardware independent layer of abstraction. The system interface is identical for all versions of **RTOS-UH**, so applications can be easily ported between different processors using **RTOS-UH**.

## System interface

---

## Speed

Speed and dependability of timing are the most relevant aspects of a realtime operating system for control. Benchmark data for **RTOS-UH** show these values:

Processor Clock frequency Board	PowerPC 7455 1 GHz MVME 5500	MC68060 50 MHz MVME 177
Context switch (a to b, both tasks runnable)	<b>0.8 <math>\mu</math>s</b>	<b>6.5 <math>\mu</math>s</b>
Task wake up (by interrupt)	<b>2.3 <math>\mu</math>s</b>	<b>11.5 <math>\mu</math>s</b>
Longest interrupt lock-out (by design)	<b>109 instructions</b>	<b>89 instructions</b>

## Storage requirement

**RTOS-UH** is famous for its low memory requirements. The following table gives an impression of memory requirements:

	PowerPC		MC68xxx	
	RAM	ROM	RAM	ROM
runtime kernel	24 kByte	52 kByte	5 kByte	23 kByte
with user interface, file systems and network	450 kByte	800 kByte	430 kByte	320 kByte
each task	300 Byte		220 Byte	

## Availability

IEP supports the deployment of **RTOS-UH**

- for the MC68xxx-family of MOTOROLA processors:
  - Microcontroller 683xx
  - stand alone systems based on 68000, 68010, 68020, 68030, 68040, 68060
- for the PowerPC-family
  - Microcontroller MPC5xx, MPC8xx, MPC8xxx
  - Systems based on MPC603, MPC604, MPC750, ...

For all kinds of processors, ready-to-run boards based on standard bus systems are available (VMEbus, CompactPCI).

**RTOS-UH** is also available for multiprocessor systems.

## Development-environment

Apart from its first class realtime behavior, **RTOS-UH** offers complete support for software development.

All programming tools as well as compilers for PEARL and CREST-C are available either as generic tools or for cross development using the Microsoft Windows operating system.