# RTA-OSEK

Binding Manual: TriCore17x6/Tasking

## Contact Details


| | |
|---|---|
| **ETAS Group**<br>www.etasgroup.com | |
| **ETAS GmbH**<br>70469 Stuttgart, Germany<br>Tel.:+49 711 89661-0<br>Fax:+49 711 89661-300<br>sales.de@etas.com | **ETAS Inc.**<br>Ann Arbor, MI 48103, USA<br>Tel.: +1 888 ETAS INC<br>Fax: +1 734 997 9449<br>sales.us@etas.com |
| **ETAS K.K.**<br>Yokohama 220-6217, Japan<br>Tel.: +81 45 222-0900<br>Fax: +81 45 222-0956<br>sales.jp@etas.com | **ETAS S.A.S.**<br>94588 Rungis Cedex, France<br>Tel.: +33 (1) 56 70 00 50<br>Fax: +33 (1) 56 70 00 51<br>sales.fr@etas.com |
| **ETAS Korea Co. Ltd.**<br>Seoul 137-889, Korea<br>Tel.: +82 2 5747-016<br>Fax: +82 2 5747-120<br>sales.kr@etas.com | **ETAS Ltd.**<br>Burton-upon-Trent<br>Staffordshire DE14 2WQ, UK<br>Tel.: +44 1283 54 65 12<br>Fax: +44 1283 54 87 67<br>sales.uk@etas.com |
| **ETAS (Shanghai) Co., Ltd.**<br>Shanghai 200120, P.R. China<br>Tel.: +86 21 5037 2220<br>Fax: +86 21 5037 2221<br>sales.cn@etas.com | **ETAS Automotive India Pvt. Ltd.**<br>Bangalore 560 068, India<br>Tel.: +91 80 4191 2585<br>Fax: +91 80 4191 2586<br>sales.in@etas.com |

# Copyright Notice



Version: RM00074-002



## Disclaimer

The information in this document is subject to change without notice and does not represent a commitment on any part of LiveDevices. While the information contained herein is assumed to be accurate, LiveDevices assumes no responsibility for any errors or omissions.

In no event shall LiveDevices, its employees, its contractors or the authors of this document be liable for special, direct, indirect, or consequential damage, losses, costs, charges, claims, demands, claim for lost profits, fees or expenses of any nature or kind.

## Trademarks

RTA-OSEK, RTA-TRACE and LiveDevices are trademarks of LiveDevices Ltd.

Windows and MS-DOS are trademarks of Microsoft Corp.

OSEK/VDX is a trademark of Siemens AG.

All other product names are trademarks or registered trademarks of their respective owners.

# Contents

# 1 About this Guide

This guide provides target-specific information for the TriCore17x6/Tasking port of LiveDevices' RTA-OSEK. It supplements the more general information in the *RTA-OSEK User Guide*.

A port is defined as a specific target microcontroller/target toolchain pairing. This guide tells you about integration issues with your target toolchain and issues that you need to be aware of when using RTA-OSEK on your target hardware. Port specific parameters of implementation are also provided, giving the RAM and ROM requirements for each object in the RTA-OSEK Component and execution times for each API call to the RTA-OSEK Component.

## 1.1 Who Should Read this Guide?

The reader should have an understanding of real time embedded programming in an OSEK context. You should read this guide if you want to know low-level technical information to integrate the RTA-OSEK Component into your application.

## 1.2 Conventions

**Important:** Notes that appear like this contain important information that you need to be aware of. Make sure that you read them carefully and that you follow any instructions that you are given.

**Portability:** Notes that appear like this describe things that you will need to know if you want to write code that will work on any processor running the RTA-OSEK Component.

Program code, file names, C types and symbols, and RTA-OSEK API call names all appear in the `courier` typeface. When the name of an object is made available to the programmer the name also appears in the `courier` typeface, so, for example, a task named Task1 appears as a task handle called `Task1`.

# 2 Toolchain Issues

This chapter contains important details about RTA-OSEK and your toolchain. A port of the RTA-OSEK Component is specific to both the target hardware and a specific version of the compiler toolchain. You must make sure that you build your application with the supported toolchain.

If you are interested in using a different version of the same toolchain, please contact LiveDevices to confirm whether or not this is possible.

## 2.1 Compiler

The RTA-OSEK Component was built using the following compiler:

| | |
|---|---|
| Vendor | Tasking |
| Compiler | TriCore VX-toolset C compiler |
| Version | v2.2r3 Build 156.1.13 |

The compulsory compiler options for application code are shown in the following table:

| Option | Description |
|---|---|
| `-C<cpu>` | Selects target CPU |
| `--silicon-bug=cpu-tc013, cpu-tc048, cpu-tc060, cpu-tc065, cpu-tc068, cpu-tc069, cpu-tc070, cpu-tc071, cpu-tc072, cpu-tc074, cpu-tc081, cpu-tc082, cpu-tc083, cpu-tc094, cpu-tc095, cpu-tc096` | Includes workarounds for Tc17X6 CPU silicon bugs |

The C file that RTA-OSEK generates from your OIL configuration file is called `osekdefs.c`. This file defines configuration parameters for the RTA-OSEK Component when running your application.

The compulsory compiler options for `osekdefs.c` are shown in the following table:

| Option | Description |
|---|---|
| `-C<cpu>` | Selects target CPU |
| `--silicon-bug=cpu-tc013, cpu-tc048, cpu-tc060, cpu-tc065, cpu-tc068, cpu-tc069, cpu-tc070, cpu-tc071, cpu-tc072, cpu-tc074, cpu-tc081, cpu-tc082, cpu-tc083, cpu-tc094, cpu-tc095, cpu-tc096` | Includes workarounds for Tc17X6 CPU silicon bugs |
| `--integer-enumeration` | Forces C enums to be integers |

The prohibited compiler options for `osekdefs.c` are shown in the following table:

| Option | Description |
|---|---|
| `-g` | Enable debugging information |

To support the use of multiple CPU configurations the environment variable CPU_TYPE should be set up to match the desired CPU target (e.g. Tc1796).

The startup code supplied with the Tasking toolset is fully compatible with RTA-OSEK and does not require modification.

The compiler is used with flags to include workarounds for known TriCore silicon problems.

**Important:** If tracing is enabled then the compiler will generate the following warnings when compiling the automatically generated file `osekdefs.c`:

Warning W517 “ambiguous 'else' part - suggest braces” - The code to which this warning refers compiles in the correct way.

Warning W549 “condition is constant” - The use of a constant condition is intentional and allows the compiler to optimize away some unneeded code.

Warning W560 “possible truncation at implicit conversion to type "xxxx"” - This warning arises because RTA-TRACE uses different sized data items depending on the tracing mode (simple or advanced) and on the target processor. The code to which the warning refers compiles in the correct way.

The above warnings are benign and build scripts generated by RTA-OSEK automatically suppress these warnings.

## 2.2 Assembler

The RTA-OSEK Component was built using the following assembler:

| | |
|---|---|
| Vendor | Tasking |
| Assembler | TriCore VX-toolset assembler |
| Version | v2.2r3 Build 134.1.4 |

The compulsory assembler options for application code are shown in the following table:

| Option | Description |
|---|---|
| `-C<cpu>` | Selects target CPU |
| `--silicon-bug=cpu-tc013, cpu-tc048, cpu-tc060, cpu-tc065, cpu-tc068, cpu-tc069, cpu-tc070, cpu-tc071, cpu-tc072, cpu-tc074, cpu-tc081, cpu-tc082, cpu-tc083, cpu-tc094, cpu-tc095, cpu-tc096` | Includes workarounds for Tc17X6 CPU silicon bugs |

The assembly file that RTA-OSEK generates from your OIL configuration file is called `osgen.asm`. This file defines configuration parameters for the RTA-OSEK Component when running your application.

The compulsory assembler options for `osgen.asm` are shown in the following table:

| Option | Description |
|---|---|
| `-C<cpu>` | Selects target CPU |
| `--silicon-bug=cpu-tc013, cpu-tc048, cpu-tc060, cpu-tc065, cpu-tc068, cpu-tc069, cpu-tc070, cpu-tc071, cpu-tc072, cpu-tc074, cpu-tc081, cpu-tc082, cpu-tc083, cpu-tc094, cpu-tc095, cpu-tc096` | Includes workarounds for Tc17X6 CPU silicon bugs |

The prohibited assembler options for `osgen.asm` are shown in the following table:

| Option | Description |
|---|---|
| `-g` | Enable debugging information |

**Important:** The assembler will generate warning W292 "suspicious instruction concerning CPU functional defect TC113_CPU14" when assembling the automatically generated file `osgen.asm`. Functional defect TC113_CPU14 is not present in any production TriCore v1.3 processor cores and so the warning may be ignored (the defect was present in some prototype v1.3 processor cores). Build scripts generated by RTA-OSEK automatically suppress this warning.

**Important:** The assembler will generate warning W809 "suspicious instruction concerning CPU functional defect CPU_TC.048" when assembling the automatically generated file `osgen.asm`. The defect only affects a load from memory into an address register. In the code in question the load is of an immediate value not the contents of memory, so the warning may safely be ignored. Build scripts generated by RTA-OSEK automatically suppress this warning.

## 2.3 Linker/Locator

In addition to the sections used by application code, the following RTA-OSEK sections must be located:

| Sections | ROM/RAM | Description |
|---|---|---|
| `.rodata.os*.os_pid` | ROM | RTA-OSEK read-only data |
| `.rodata.os*.os_pird` | ROM | RTA-OSEK initialization data |
| `.rodata.os*.os_pnird` | ROM | RTA-OSEK near initialization data |
| `.text.os*.os_text` | RAM | RTA-OSEK library code |
| `.bss.os*.os_pir` | RAM | RTA-OSEK initialized data (Initialized by StartOS() API call) |
| `.data.os*.os_pir2` | RAM | RTA-OSEK initialized data (Must be initialized during C-startup) |
| `.bss.os*.os_pur` | RAM | RTA-OSEK uninitialized data (Initialized by StartOS() API call) |
| `.bss.os*.os_pur2` | RAM | RTA-OSEK uninitialized data (Must be initialized during C-startup) |
| `.sbss.os*.os_pnir` | RAM | RTA-OSEK A0 initialized data (Initialized by StartOS() API call) |
| `.sbss.os*.os_pnir2` | RAM | RTA-OSEK A0 uninitialized data (Must be initialized during C-startup) |

| Sections | ROM/RAM | Description |
|---|---|---|
| .sdata.os*.os_pnir2 | RAM | RTA-OSEK A0 initialized data (Must be initialized during C-startup) |
| .sbss.os*.os_cntr | RAM | RTA-OSEK A0 uninitialized data (Must be initialized during C-startup) |
| .sdata.os*.os_cntr | RAM | RTA-OSEK A0 initialized data (Must be initialized during C-startup) |
| .bss.os*.os_trace_ram | RAM | RTA-TRACE data buffer |

In some cases, sections produced by the linker must be located according to special constraints. The following table indicates which sections must be located with which particular constraints:

| Sections | Constraints |
|---|---|
| *.os_pnir | Must be placed in the compiler A0 addressed RAM |
| *.os_pnir2 | Must be placed in the compiler A0 addressed RAM |
| *.os_cntr | Must be placed in the compiler A0 addressed RAM |

It is assumed that for all memory sections initialized by the `StartOS()` API call that the compiler C-startup routines will first initialize these sections to zero. When the `StartOS()` API call is invoked these sections will then be initialized for use by RTA-OSEK. Hence the `os_pir` and `os_pnir` sections are described as containing RTA-OSEK initialized data but are also bss/sbss sections. Where a bss/sbss section is described as being initialized by C-startup then we assume that the section is initialized to all zero.

The RTA-OSEK Component requires the user stack to be quad word aligned. In order to achieve this, `__TC112_COR16__` should be defined when using the linker script files supplied with the Tasking toolchain.

**Important:** The RTA-OSEK Component makes use of relative 24-bit signed addressing mode. This means the library must be contained within a 16Mbyte memory block. 32-bit addressing is used externally providing no restrictions on placement of user code and data.

For improved performance, the RTA-OSEK Component uses a small amount of RAM data that should be located in a section that is addressable with a sign-extended 16-bit offset from address register A0. Please refer to the compiler documentation on the use of this section.

### 2.3.1 Direct Addressing

By default 32-bit addressing is used between application code and the RTA-OSEK Component so that the entire memory space can be supported. The RTA-OSEK component can also use direct addressing to reduce the code size and improve performance. Direct addressing requires that all code

objects must be located either within a relative signed 24-bit displacement address range of the current address, or an absolute disp24 address range (See the Infineon v1.3 Instruction set manual for further details). These address ranges cover either a ±16 Mbytes region from the current address, or the first 2 Mbytes of every 256 Mbytes. Direct addressing cannot be used between code objects located in the internal FLASH memory and the external memory regions or in the CSRAM of v1.3 CPUs. To make use of direct addressing application code should be compiled with the macro `OS_DIRECT_CALLS` defined.

### 2.3.2 Use of SPRAM

The RTA-OSEK Component supports placement of the vector table RTA-OSEK code and RTA-OSEK read only data in SPRAM.

#### Interrupt vector table placement in RAM

By default the Interrupt vectors are located in ROM. To support RAM location the file `osgen.asm` should be compiled with the command line option `-DOS_VEC_RAM=0`. This define causes the vector table entries to be compiled with the INIT section directive modifier to place it in RAM. Note any Category 1 ISR handlers declared in other C files should include the directive `#pragma section vector_init` to ensure that all interrupt vector entries are placeable in RAM. Please refer to the compiler documentation on the use of this directive.

#### RTA-OSEK code and constant data placement in RAM

The code and constant data sections of the RTA-OSEK Component are supplied by the run-time libraries and the files `osgen.asm` and `osekdefs.c`. By default these sections are placed in ROM but this can be overridden to place them into RAM.

To place ROM data in the file `osgen.asm` into RAM the command line option for assembling the file should include the command line option `-DOS_DATA_SPRAM`. The inclusion of this define adds the INIT modifier to the section directive so the `os_pid` section will be placed in RAM.

The C code file `osekdefs.c` contains fragments of OS of code and data. To make the code placeable in RAM the command line should include the command `-DOS_CODE_SPRAM`. The presence of this define includes the directive `#pragma section code_init` in the code. To make the ROM data placeable into RAM the command line should include the command `-DOS_DATA_SPRAM` to include the directive `#pragma section const_init` into the code. Please refer to the compiler documentation on the use of these directives.

To support the placement of the RTA-OSEK Component API calls within RAM multiple versions of the libraries are provided. The default versions of the

libraries (i.e. all named `rtk_(s/t/e).a` library) configure the code and constant data sections to be placed in ROM. An alternative set of libraries are provided that have been compiled from the same code base but with the `#pragma section code_init` and `#pragma section const_init` directives. These libraries all include `_ram` in the name (i.e. `rtk_s_ram.a` for the standard build library). The `os_text` and sections and `os_pid` sections of these libraries can then be located in RAM.

### Placing RTA-OSEK application components in SPRAM

By default the linker will place the initializable code and data section in RAM. Modifications are required to the linker command file to correctly place the sections in SPRAM. To place the vector table in SPRAM the linker command file must include a define of INTTAB with an address in SPRAM (i.e. `#define INTTAB 0xD400B000` for the Tc1796, or `0xC0005000` for the Tc1797). To place A0 addressed variables in on-chip RAM the define A0_START_ADDRESS should be used (i.e. `#define A0_START_ADDRESS 0xD0003700` for the Tc1796 SPRAM, or `0xD0000000` for the Tc1797 LDRAM). To place the library `osgen.asm` and `osekdefs.c` code and data sections into SPRAM the linker command file should contain a section such as the following:

```
// place the OS in SPRAM
section_layout spe:tc:linear
{
  group os_code (ordered, run_addr=mem:spe:csram)
  {
        select "*.os_text";
        select "*.os_pid";
  }

  group os_data (ordered, run_addr=mem:spe:dsram)
  {
        select "*.os_pnir";
        select "*.os_pnir2";
        select "*.os_pur";
        select "*.os_pur2";
        select "*.os_pir";
        select "*.os_pir2";
  }
}
```

## 2.4 Debugger

ORTI is the OSEK Run-Time Interface that is supported by RTA-OSEK. Support is provided for the debuggers in the following table. Further information about ORTI for RTA-OSEK can be found in the *RTA-OSEK ORTI Guide*.

| ORTI Compatible Debuggers |
| --- |
| Tasking Crossview Pro 2.2r3 Build 062 |
| Lauterbach Trace32 Build 1194 or later |

# 3 Target Hardware Issues

## 3.1 Interrupts

This section explains the implementation of RTA-OSEK's interrupt model for TriCore17x6/Tasking. You can find out more about configuring interrupts for RTA-OSEK in the *RTA-OSEK User Guide*.

### 3.1.1 Interrupt Levels

In RTA-OSEK interrupts are allocated an Interrupt Priority Level (IPL). This is a processor independent abstraction of the interrupt priorities that are available on the target hardware. You can find out more about IPLs in the *RTA-OSEK User Guide*. The hardware interrupt controller is explained in the *Infineon TriCore User's Manual - System Units*.

The following table shows how RTA-OSEK IPLs relate to interrupt priorities on the target hardware:

| IPL Value | CCPN Field Of ICR Register | Description |
|---|---|---|
| 0 | 0 | User level |
| 1-255 | 1-255 | Category 1 and 2 interrupts |

### 3.1.2 Interrupt Vectors

For the allocation of Category 1 and Category 2 interrupt handlers to interrupt vectors on your target hardware, the following restrictions apply:

| Vector | Legality |
|---|---|
| 1-255 | Category 1<br>Note all Category 1 interrupt vectors must be configured to be higher than the highest Category 2 interrupt vector. |
| 1-255 | Category 2 |

The valid base addresses for the vector table are:

| Base Address | Notes |
|---|---|
| BIV | Set by this register. See TriCore Architecture manual for a full explanation. |

### 3.1.3 Category 1 Handlers

Category 1 interrupt service routines (ISRs) must correctly handle the interrupt context themselves, without support from the operating system. The Tasking C compiler can generate appropriate interrupt handling code for a C function

decorated with the `__interrupt(<vector>) __enable_` function qualifier. You can find out more in your compiler documentation.

### 3.1.4 Category 2 Handlers

Category 2 ISRs are provided with a C function context by the RTA-OSEK Component, since the RTA-OSEK Component handles the interrupt context itself. The handlers are written using the OSEK OS standard `ISR()` macro, shown in Code Example 3:1.

```
#include "MyISR.h"
ISR(MyISR) {
  /* Handler routine */
}
```

**Code Example 3:1 - Category 2 ISR Interrupt Handler**

You must not insert a return from interrupt instruction in such a function. The return is handled automatically by the RTA-OSEK Component.

### 3.1.5 Vector Table Issues

RTA-OSEK relies on the compiler-generated vector table and does not produce its own table. For this reason, the Vector Generation option in the RTA-OSEK GUI has no effect.

The following table shows the syntax for labels attached to RTA-OSEK Category 2 interrupt handlers.

| Vector Location | Label |
|---|---|
| `BIV + 0x0020` | `os_wrapper_001` |
| `BIV + 0x0040` | `os_wrapper_002` |
| ... | ... |
| `BIV + 0x1fe0` | `os_wrapper_0ff` |

### 3.1.6 IO Privilege Mode

The RTA-OSEK Component operates with the TriCore CPU in supervisor mode at all times and this must not be altered.

### 3.1.7 Interrupt Priorities

When an interrupt becomes pending, it is handled as soon as the configured vector number is strictly greater than the current hardware priority value in ICR.CCPN.

RTA-OSEK supports a straightforward, pre-emptive interrupt model, where each ISR runs at the same priority as the vector number. This matches the default TriCore interrupt behavior. To achieve this, configure the Priority for each ISR to be the same as the Vector for that ISR.

For example, an application with two Category 2 ISRs, A and B and two Category 1 ISRs, C and D, could be configured as follows:

| ISR | Category | Vector | Priority |
|---|---|---|---|
| D | 1 | 4 | 4 |
| C | 1 | 3 | 3 |
| B | 2 | 2 | 2 |
| A | 2 | 1 | 1 |

Note that the range of interrupt vectors used does not need to be contiguous – i.e. there can be gaps in the range of vectors used.

The Category 1 ISRs would be written as

```
void __interrupt(4) __enable_ D(void)
{
  /* handler code for D */
}
```

and

```
void __interrupt(3) __enable_ C(void)
{
  /* handler code for C */
}
```

## Serialized Category 2 ISRs

In addition to the straightforward, pre-emptive interrupt model, RTA-OSEK also supports serialized Category 2 ISRs. A contiguous group of Category 2 ISRs, with lowest Vector `u` and highest Vector `v`, can be serialized by raising all their Priorities to `v`.

The RTA-OSEK Component starts Category 2 ISRs at their specified Priority, achieving the appropriate serialization at run-time.

For correct schedulability analysis in the presence of serialized, or "shared priority" Category 2 ISRs, interrupt arbitration information must be entered for each shared priority. The arbitration order must be set so that the higher vector numbers come earlier in the arbitration order.

For example, an application with two Category 2 ISRs, A and B and two Category 1 ISRs, C and D, could be configured as follows:

| ISR | Category | Vector | Priority |
|---|---|---|---|
| D | 1 | 4 | 4 |
| C | 1 | 3 | 3 |
| B | 2 | 2 | 2 |
| A | 2 | 1 | 2 |

| | |
|---|---|
| Arbitration | Level 2 ordering: B, A. |

Serialization causes higher vector ISRs to be delayed until the completion of lower vector ISRs of the same priority but it gives the following advantages:

- If a resource is used only by a group of serialized ISRs, they can get and release that resource at zero overhead, using the RTA-OSEK static interface to resources.
- Inhibiting pre-emption between ISRs reduces the worst-case stack and CSA requirements for an application.

## Serialized Category 1 ISRs

RTA-OSEK also supports serialization of Category 1 ISRs. The highest vector Category 1 ISRs can be serialized by raising all their Priorities to `255`.

Furthermore, such serialized Category 1 ISRs must be written using either the `__interrupt(<vector>)` or `__interrupt_fast(<vector>)` function qualifiers *without* the `__enable_` modifier and they are not permitted to use the `__enable()` intrinsic. This ensures that they execute with interrupts disabled, giving the appropriate serialization at run-time.

For correct schedulability analysis in the presence of serialized, or "shared priority" Category 1 ISRs, interrupt arbitration information must be entered for priority `255`. The arbitration order must be set so that the higher vector numbers come earlier in the arbitration order.

For example, an application with two Category 2 ISRs, A and B and two Category 1 ISRs, C and D, could be configured as follows:

| ISR | Category | Vector | Priority |
|---|---|---|---|
| D | 1 | 4 | 255 |
| C | 1 | 3 | 255 |
| B | 2 | 2 | 2 |
| A | 2 | 1 | 2 |
| Arbitration | Level 255 ordering: D, C. | | |
| Arbitration | Level 2 ordering: B, A. | | |

The Category 1 ISRs would be written as

```
void __interrupt(4) D(void)
{
  /* handler code for D */
}
```

and

```
void __interrupt(3) C(void)
{
  /* handler code for C */
}
```

Serialization causes higher vector ISRs to be delayed until the completion of lower vector ISRs of the same priority but it gives the following advantage:

- Inhibiting pre-emption between ISRs reduces the worst-case stack and CSA requirements for an application.

### 3.1.8 Default Interrupt

The default interrupt handler does not require the `__interrupt(<vector>)` or `__interrupt_fast(<vector>)` qualifier and must be declared as a standard C function. The RTA-OSEK Component handles the interrupt context itself in this case.

## 3.2 Register Settings

The RTA-OSEK Component requires the following registers to be initialized before calling `StartOS()`.

| Register | Required Value | Notes |
|---|---|---|
| BIV | Interrupt vector table base address | |
| FCX | Free context list.<br>This should be initialized to point to a suitably large linked list of context areas (performed by Tasking C startup code) | |
| PSW.IO | Supervisor (2) | Hardware default is 2 |

The RTA-OSEK Component uses the following hardware registers. They should not be altered by user code.

| Registers Used | Notes |
|---|---|
| PCXI | Previous context information register |
| FCX | Free CSA list head pointer |
| PSW | Processor Status Word |
| ICR | Interrupt Control Register |

## 3.3 Stack Usage

### 3.3.1 Number of Stacks

A single stack is used. The first argument to `StackFaultHook` is always 0.

`osStackOffsetType` is a scalar, representing the number of bytes on the stack, with C type `unsigned long`.

### 3.3.2 Stack Usage within API Calls

The maximum stack usage within RTA-OSEK API calls, excluding calls to hooks and callbacks, is as follows:

#### Standard

API max usage (bytes): 16

#### Timing

API max usage (bytes): 16

#### Extended

API max usage (bytes): 24

To determine the correct stack usage for tasks that use other library code, you may need to contact the library vendor to find out more about call stack usage.

### 3.3.3 Stack Mode

The RTA-OSEK Component operates with the PSW.IS bit set to one. This ensures that only one stack is used throughout an application.

### 3.3.4 Context Save Areas (CSAs)

RTA-OSEK does not currently support calculation of CSA use.

The worst case CSA usage for Task-to-Task switching in standard build is 6 CSAs. In timing and extended builds the figure is 7 CSAs.

For example, consider an application in standard build that has 10 tasks, the code for each of which requires up to 4 CSAs. The combined CSA requirement for those tasks is at most (4 + 6) * 10 = 100 CSAs. This maximum would occur when all but the highest priority task had been started and then pre-empted by the next, higher priority task and when the highest priority task was currently running.

### 3.3.5 Call Depth Counter

The RTA-OSEK Component can operate with the TriCore call depth counter either enabled or disabled. The RTA-OSEK Component does not alter the call depth counter register settings.

# 4 Parameters of Implementation

This chapter provides detailed information on the functionality, performance and memory demands of the RTA-OSEK Component.

The RTA-OSEK Component is highly scalable. As a result, different figures will be obtained when your application uses different sets of features. These feature-sets give six classes of RTA-OSEK, depending on whether your application uses events, shared task priorities and/or multiple (queued) task activations. You should identify which class your application belongs to and then use the figures from the appropriate column in the table.

The following hardware was used to take the measurements in this chapter:

| **Processor** | **TC1796** |
|---|---|
| Clock speed (MHz) | 80 |
| Code memory | Internal scratchpad RAM |
| Read-only data memory | Internal scratchpad RAM |
| Read-write data memory | Internal scratchpad RAM |

## 4.1 Functionality

The OSEK Operating System Specification specifies four conformance classes. These attributes apply to *systems* built with OSEK OS objects. The following table specifies the number of OSEK OS and COM objects supported per conformance class.

| **Configuration** | **Application Uses** | | | | | |
|---|---|---|---|---|---|---|
| **Events** | **No** | | | **Yes** | | |
| **Shared Task Priorities** | **No** | | **Yes** | **No** | | **Yes** |
| **Multiple Task Activations** | **No** | **Yes** | | **No** | **Yes** | |
| Maximum number of tasks | 32 | 32 | 32 | 32 | 32 | 32 |
| Maximum number of not suspended tasks | 32 | 32 | 32 | 32 | 32 | 32 |
| Maximum number of priorities | 32 | 32 | 32 | 32 | 32 | 32 |
| Number of tasks per priority (for BCC2 and ECC2) | n/a | 32 | 32 | n/a | 32 | 32 |
| Upper limit for number of basic task activations per task priority | 1 | 255 | 255 | 1 | 255 | 255 |
| Maximum number of events per task | 0 | 0 | 0 | 32 | 32 | 32 |
| Limits for the number of alarm objects (per system / per task) | not limited by RTA-OSEK | | | | | |
| Limits for the number of standard resources (per system) | 255 | 255 | 255 | 255 | 255 | 255 |
| Limits for the number of internal resources (per system) | not limited by RTA-OSEK | | | | | |
| Limits for the number of nested resources (per system / per task) | 255 | 255 | 255 | 255 | 255 | 255 |

| Configuration | Application Uses | | | | | |
|---|---|---|---|---|---|---|
| Events | No | | | Yes | | |
| Shared Task Priorities | No | | Yes | No | | Yes |
| Multiple Task Activations | No | Yes | | No | Yes | |
| Limits for the number of application modes | 4294967295 | | | | | |

## 4.2 Hardware Resources

### 4.2.1 ROM and RAM Overheads

The following tables give the ROM and RAM overheads for the RTA-OSEK Component (in bytes). The OSEK COM overheads are quoted separately. If you do not use messages, your application will not include this overhead for the parts of OSEK COM required to implement messaging.

#### Standard

| Configuration | | Application Uses | | | | | |
|---|---|---|---|---|---|---|---|
| Events | | No | | | Yes | | |
| Shared Task Priorities | | No | | Yes | No | | Yes |
| Multiple Task Activations | | No | Yes | | No | Yes | |
| OS overhead | RAM | 46 | 46 | 46 | 46 | 46 | 46 |
| | ROM | 168 | 168 | 174 | 282 | 282 | 288 |
| COM overhead | RAM | 8 | 8 | 8 | 8 | 8 | 8 |
| | ROM | 16 | 16 | 16 | 16 | 16 | 16 |

#### Timing

| Configuration | | Application Uses | | | | | |
|---|---|---|---|---|---|---|---|
| Events | | No | | | Yes | | |
| Shared Task Priorities | | No | | Yes | No | | Yes |
| Multiple Task Activations | | No | Yes | | No | Yes | |
| OS overhead | RAM | 66 | 66 | 66 | 66 | 66 | 66 |
| | ROM | 240 | 240 | 246 | 354 | 354 | 360 |
| COM overhead | RAM | 8 | 8 | 8 | 8 | 8 | 8 |
| | ROM | 16 | 16 | 16 | 16 | 16 | 16 |

## Extended

| Configuration | | Application Uses | | | | | |
|---|---|---|---|---|---|---|---|
| Events | | No | | | Yes | | |
| Shared Task Priorities | | No | | Yes | No | | Yes |
| Multiple Task Activations | | No | Yes | | No | Yes | |
| OS overhead | RAM | 84 | 84 | 84 | 84 | 84 | 84 |
| | ROM | 294 | 294 | 300 | 410 | 410 | 416 |
| COM overhead | RAM | 8 | 8 | 8 | 8 | 8 | 8 |
| | ROM | 16 | 16 | 16 | 16 | 16 | 16 |

### 4.2.2 ROM and RAM for OSEK OS Objects

In addition to the base OS overhead, detailed in Section 4.2.1, each OSEK OS object requires ROM and/or RAM. RTA-OSEK provides additional sub-task types for each task type in OSEK (basic and extended), determined by the offline configuration tools. They are as follows:

| OSEK Class | Termination | Arithmetic |
|---|---|---|
| BCC1 | Lightweight | Integer or Floating-Point |
| BCC1 | Heavyweight | Integer or Floating-Point |
| BCC2 | Light or Heavy | Integer or Floating-Point |
| ECC1 | Heavyweight | Integer |
| ECC1 | Heavyweight | Floating-Point |
| ECC2 | Heavyweight | Integer |
| ECC2 | Heavyweight | Floating-Point |

The following tables give the ROM and/or RAM requirements (in bytes) for each OS object in the RTA-OSEK Component. (Note that the OSEK COM class was set to CCCA for systems without events, CCCB for systems with events. A default message of size 10 bytes was used for both CCCA and CCCB. The CCCB message size includes queued messages.)

## Standard

| Configuration | | Application Uses | | | | | |
|---|---|---|---|---|---|---|---|
| Events | | No | | | Yes | | |
| Shared Task Priorities | | No | | Yes | No | | Yes |
| Multiple Task Activations | | No | Yes | | No | Yes | |
| BCC1 Lightweight task | RAM | 0 | 0 | 0 | 0 | 0 | 0 |
| | ROM | 36 | 36 | 36 | 36 | 36 | 36 |
| BCC1 Heavyweight task | RAM | 4 | 4 | 4 | 4 | 4 | 4 |
| | ROM | 40 | 40 | 40 | 40 | 40 | 40 |
| BCC2 task | RAM | n/a | 8 | 10 | n/a | 8 | 10 |
| | ROM | n/a | 48 | 56 | n/a | 48 | 56 |
| ECC1, Integer task | RAM | n/a | n/a | n/a | 28 | 28 | 28 |
| | ROM | n/a | n/a | n/a | 60 | 60 | 60 |
| ECC1, floating-point task | RAM | n/a | n/a | n/a | 30 | 30 | 30 |
| | ROM | n/a | n/a | n/a | 60 | 60 | 60 |
| ECC2, Integer task | RAM | n/a | n/a | n/a | n/a | n/a | 30 |
| | ROM | n/a | n/a | n/a | n/a | n/a | 68 |
| ECC2, floating-point task | RAM | n/a | n/a | n/a | n/a | n/a | 32 |
| | ROM | n/a | n/a | n/a | n/a | n/a | 68 |
| Category 2 ISR | RAM | 0 | 0 | 0 | 0 | 0 | 0 |
| | ROM | 68 | 68 | 68 | 68 | 68 | 68 |
| Category 2 ISR, floating-point | RAM | 1 | 1 | 1 | 1 | 1 | 1 |
| | ROM | 108 | 108 | 108 | 108 | 108 | 108 |
| Resource | RAM | 0 | 0 | 0 | 0 | 0 | 0 |
| | ROM | 20 | 20 | 20 | 20 | 20 | 20 |
| Internal resource | RAM | 0 | 0 | 0 | 0 | 0 | 0 |
| | ROM | 0 | 0 | 0 | 0 | 0 | 0 |
| Linked resource | RAM | 0 | 0 | 0 | 0 | 0 | 0 |
| | ROM | 20 | 20 | 20 | 20 | 20 | 20 |
| Alarm | RAM | 12 | 12 | 12 | 12 | 12 | 12 |
| | ROM | 60 | 60 | 60 | 60 | 60 | 60 |
| Counter | RAM | 4 | 4 | 4 | 4 | 4 | 4 |
| | ROM | 122 | 122 | 122 | 122 | 122 | 122 |
| Message | RAM | 11 | 11 | 11 | 31 | 31 | 31 |
| | ROM | 20 | 20 | 20 | 56 | 56 | 56 |
| Flag | RAM | 4 | 4 | 4 | 4 | 4 | 4 |
| | ROM | 1 | 1 | 1 | 1 | 1 | 1 |
| Message resource | RAM | 0 | 0 | 0 | 0 | 0 | 0 |
| | ROM | 20 | 20 | 20 | 20 | 20 | 20 |

| Configuration | | Application Uses | | | | | |
|---|---|---|---|---|---|---|---|
| Events | | No | | | Yes | | |
| Shared Task Priorities | | No | | Yes | No | | Yes |
| Multiple Task Activations | | No | Yes | | No | Yes | |
| Event | RAM | 0 | 0 | 0 | 0 | 0 | 0 |
| | ROM | 4 | 4 | 4 | 4 | 4 | 4 |
| Priority level | RAM | 0 | 0 | 6 | 0 | 6 | 6 |
| | ROM | 0 | 0 | 12 | 0 | 12 | 12 |
| ScheduleTable | RAM | 16 | 16 | 16 | 16 | 16 | 16 |
| | ROM | 66 | 66 | 66 | 66 | 66 | 66 |
| ScheduleTable Expiry | RAM | 0 | 0 | 0 | 0 | 0 | 0 |
| | ROM | 12 | 12 | 12 | 12 | 12 | 12 |
| Arrivalpoint (readonly) | RAM | 0 | 0 | 0 | 0 | 0 | 0 |
| | ROM | 12 | 12 | 12 | 12 | 12 | 12 |
| Arrivalpoint (writable) | RAM | 12 | 12 | 12 | 12 | 12 | 12 |
| | ROM | 12 | 12 | 12 | 12 | 12 | 12 |
| Schedule | RAM | 16 | 16 | 16 | 16 | 16 | 16 |
| | ROM | 36 | 36 | 36 | 36 | 36 | 36 |
| Taskset (readonly) | RAM | 0 | 0 | 0 | 0 | 0 | 0 |
| | ROM | 4 | 4 | 4 | 4 | 4 | 4 |
| Taskset (writable) | RAM | 4 | 4 | 4 | 4 | 4 | 4 |
| | ROM | 4 | 4 | 4 | 4 | 4 | 4 |

## Timing

| Configuration | | Application Uses | | | | | |
|---|---|---|---|---|---|---|---|
| Events | | No | | | Yes | | |
| Shared Task Priorities | | No | | Yes | No | | Yes |
| Multiple Task Activations | | No | Yes | | No | Yes | |
| BCC1 Lightweight task | RAM | 12 | 12 | 12 | 12 | 12 | 12 |
| | ROM | 48 | 48 | 48 | 48 | 48 | 48 |
| BCC1 Heavyweight task | RAM | 16 | 16 | 16 | 16 | 16 | 16 |
| | ROM | 52 | 52 | 52 | 52 | 52 | 52 |
| BCC2 task | RAM | n/a | 20 | 22 | n/a | 20 | 22 |
| | ROM | n/a | 60 | 68 | n/a | 60 | 68 |
| ECC1, Integer task | RAM | n/a | n/a | n/a | 40 | 40 | 40 |
| | ROM | n/a | n/a | n/a | 72 | 72 | 72 |
| ECC1, floating-point task | RAM | n/a | n/a | n/a | 42 | 42 | 42 |
| | ROM | n/a | n/a | n/a | 72 | 72 | 72 |
| ECC2, Integer task | RAM | n/a | n/a | n/a | n/a | n/a | 42 |

| Configuration | | Application Uses | | | | | |
|---|---|---|---|---|---|---|---|
| Events | | No | | | Yes | | |
| Shared Task Priorities | | No | | Yes | No | | Yes |
| Multiple Task Activations | | No | Yes | | No | Yes | |
| | ROM | n/a | n/a | n/a | n/a | n/a | 80 |
| ECC2, floating-point task | RAM | n/a | n/a | n/a | n/a | n/a | 44 |
| | ROM | n/a | n/a | n/a | n/a | n/a | 80 |
| Category 2 ISR | RAM | 12 | 12 | 12 | 12 | 12 | 12 |
| | ROM | 130 | 130 | 130 | 130 | 130 | 130 |
| Category 2 ISR, floating-point | RAM | 14 | 14 | 14 | 14 | 14 | 14 |
| | ROM | 158 | 158 | 158 | 158 | 158 | 158 |
| Resource | RAM | 0 | 0 | 0 | 0 | 0 | 0 |
| | ROM | 20 | 20 | 20 | 20 | 20 | 20 |
| Internal resource | RAM | 0 | 0 | 0 | 0 | 0 | 0 |
| | ROM | 0 | 0 | 0 | 0 | 0 | 0 |
| Linked resource | RAM | 0 | 0 | 0 | 0 | 0 | 0 |
| | ROM | 20 | 20 | 20 | 20 | 20 | 20 |
| Alarm | RAM | 12 | 12 | 12 | 12 | 12 | 12 |
| | ROM | 60 | 60 | 60 | 60 | 60 | 60 |
| Counter | RAM | 4 | 4 | 4 | 4 | 4 | 4 |
| | ROM | 122 | 122 | 122 | 122 | 122 | 122 |
| Message | RAM | 11 | 11 | 11 | 31 | 31 | 31 |
| | ROM | 20 | 20 | 20 | 56 | 56 | 56 |
| Flag | RAM | 4 | 4 | 4 | 4 | 4 | 4 |
| | ROM | 1 | 1 | 1 | 1 | 1 | 1 |
| Message resource | RAM | 0 | 0 | 0 | 0 | 0 | 0 |
| | ROM | 20 | 20 | 20 | 20 | 20 | 20 |
| Event | RAM | 0 | 0 | 0 | 0 | 0 | 0 |
| | ROM | 4 | 4 | 4 | 4 | 4 | 4 |
| Priority level | RAM | 0 | 0 | 6 | 0 | 6 | 6 |
| | ROM | 0 | 0 | 12 | 0 | 12 | 12 |
| ScheduleTable | RAM | 16 | 16 | 16 | 16 | 16 | 16 |
| | ROM | 66 | 66 | 66 | 66 | 66 | 66 |
| ScheduleTable Expiry | RAM | 0 | 0 | 0 | 0 | 0 | 0 |
| | ROM | 12 | 12 | 12 | 12 | 12 | 12 |
| Arrivalpoint (readonly) | RAM | 0 | 0 | 0 | 0 | 0 | 0 |
| | ROM | 12 | 12 | 12 | 12 | 12 | 12 |
| Arrivalpoint (writable) | RAM | 12 | 12 | 12 | 12 | 12 | 12 |
| | ROM | 12 | 12 | 12 | 12 | 12 | 12 |
| Schedule | RAM | 16 | 16 | 16 | 16 | 16 | 16 |

| Configuration | | Application Uses | | | | | |
|---|---|---|---|---|---|---|---|
| Events | | No | | | Yes | | |
| Shared Task Priorities | | No | | Yes | No | | Yes |
| Multiple Task Activations | | No | Yes | | No | Yes | |
| | ROM | 36 | 36 | 36 | 36 | 36 | 36 |
| Taskset (readonly) | RAM | 0 | 0 | 0 | 0 | 0 | 0 |
| | ROM | 4 | 4 | 4 | 4 | 4 | 4 |
| Taskset (writable) | RAM | 4 | 4 | 4 | 4 | 4 | 4 |
| | ROM | 4 | 4 | 4 | 4 | 4 | 4 |

## Extended

| Configuration | | Application Uses | | | | | |
|---|---|---|---|---|---|---|---|
| Events | | No | | | Yes | | |
| Shared Task Priorities | | No | | Yes | No | | Yes |
| Multiple Task Activations | | No | Yes | | No | Yes | |
| BCC1 Lightweight task | RAM | 16 | 16 | 16 | 16 | 16 | 16 |
| | ROM | 60 | 60 | 60 | 60 | 60 | 60 |
| BCC1 Heavyweight task | RAM | 20 | 20 | 20 | 20 | 20 | 20 |
| | ROM | 60 | 60 | 60 | 60 | 60 | 60 |
| BCC2 task | RAM | n/a | 24 | 26 | n/a | 24 | 26 |
| | ROM | n/a | 68 | 76 | n/a | 68 | 76 |
| ECC1, Integer task | RAM | n/a | n/a | n/a | 44 | 44 | 44 |
| | ROM | n/a | n/a | n/a | 80 | 80 | 80 |
| ECC1, floating-point task | RAM | n/a | n/a | n/a | 46 | 46 | 46 |
| | ROM | n/a | n/a | n/a | 80 | 80 | 80 |
| ECC2, Integer task | RAM | n/a | n/a | n/a | n/a | n/a | 46 |
| | ROM | n/a | n/a | n/a | n/a | n/a | 88 |
| ECC2, floating-point task | RAM | n/a | n/a | n/a | n/a | n/a | 48 |
| | ROM | n/a | n/a | n/a | n/a | n/a | 88 |
| Category 2 ISR | RAM | 16 | 16 | 16 | 16 | 16 | 16 |
| | ROM | 142 | 142 | 142 | 142 | 142 | 142 |
| Category 2 ISR, floating-point | RAM | 18 | 18 | 18 | 18 | 18 | 18 |
| | ROM | 170 | 170 | 170 | 170 | 170 | 170 |
| Resource | RAM | 8 | 8 | 8 | 8 | 8 | 8 |
| | ROM | 28 | 28 | 28 | 28 | 28 | 28 |
| Internal resource | RAM | 0 | 0 | 0 | 0 | 0 | 0 |
| | ROM | 0 | 0 | 0 | 0 | 0 | 0 |
| Linked resource | RAM | 8 | 8 | 8 | 8 | 8 | 8 |
| | ROM | 28 | 28 | 28 | 28 | 28 | 28 |

| Configuration | | Application Uses | | | | | |
|---|---|---|---|---|---|---|---|
| Events | | No | | | Yes | | |
| Shared Task Priorities | | No | | Yes | No | | Yes |
| Multiple Task Activations | | No | Yes | | No | Yes | |
| Alarm | RAM | 12 | 12 | 12 | 12 | 12 | 12 |
| | ROM | 64 | 64 | 64 | 64 | 64 | 64 |
| Counter | RAM | 4 | 4 | 4 | 4 | 4 | 4 |
| | ROM | 126 | 126 | 126 | 126 | 126 | 126 |
| Message | RAM | 11 | 11 | 11 | 31 | 31 | 31 |
| | ROM | 24 | 24 | 24 | 60 | 60 | 60 |
| Flag | RAM | 4 | 4 | 4 | 4 | 4 | 4 |
| | ROM | 1 | 1 | 1 | 1 | 1 | 1 |
| Message resource | RAM | 8 | 8 | 8 | 8 | 8 | 8 |
| | ROM | 28 | 28 | 28 | 28 | 28 | 28 |
| Event | RAM | 0 | 0 | 0 | 0 | 0 | 0 |
| | ROM | 4 | 4 | 4 | 4 | 4 | 4 |
| Priority level | RAM | 0 | 0 | 6 | 0 | 6 | 6 |
| | ROM | 0 | 0 | 12 | 0 | 12 | 12 |
| ScheduleTable | RAM | 16 | 16 | 16 | 16 | 16 | 16 |
| | ROM | 66 | 66 | 66 | 66 | 66 | 66 |
| ScheduleTable Expiry | RAM | 0 | 0 | 0 | 0 | 0 | 0 |
| | ROM | 12 | 12 | 12 | 12 | 12 | 12 |
| Arrivalpoint (readonly) | RAM | 0 | 0 | 0 | 0 | 0 | 0 |
| | ROM | 20 | 20 | 20 | 20 | 20 | 20 |
| Arrivalpoint (writable) | RAM | 20 | 20 | 20 | 20 | 20 | 20 |
| | ROM | 20 | 20 | 20 | 20 | 20 | 20 |
| Schedule | RAM | 20 | 20 | 20 | 20 | 20 | 20 |
| | ROM | 44 | 44 | 44 | 44 | 44 | 44 |
| Taskset (readonly) | RAM | 0 | 0 | 0 | 0 | 0 | 0 |
| | ROM | 4 | 4 | 4 | 4 | 4 | 4 |
| Taskset (writable) | RAM | 4 | 4 | 4 | 4 | 4 | 4 |
| | ROM | 4 | 4 | 4 | 4 | 4 | 4 |

### 4.2.3 Size of Linkable Modules

The RTA-OSEK Component is demand linked. This means that each API call is placed into a separately linkable module. The following sections list the module sizes (in bytes) for each API call in the 3 RTA-OSEK build types (standard, timing, and extended).

In some cases there are multiple variants of particular API calls. This is because the offline configuration of RTA-OSEK can determine when

optimized versions of the API calls can be used. The smallest and fastest call will be selected. In these cases, modules sizes are given for each variant under the particular configuration of the RTA-OSEK Component for which the call is valid.

The call variants are as follows:

| Variant | Description |
|---|---|
| 1i | Idle task is only ECC task. |
| CCCA | OSEK COM class. |
| CCCB | OSEK COM class. |
| CLEx | Resource tests in Extended OS Status. |
| fp | ECC task uses floating-point. |
| H | Used for heavyweight termination only. |
| Hook | Pre- and Post- Task hooks are used. |
| KL | API is called from OS level. |
| KL1i | API is called from OS level, idle task is only ECC task. |
| KL2 | Activated taskset has one BCC2 task. |
| LExt | Used for lightweight termination in Extended Status. |
| ServiceID | `ErrorHook` uses `GetServiceID`, but does not use `GetServiceParameters`. |
| Parameters | `ErrorHook` uses `GetServiceID` and `GetServiceParameters`. |
| NoHook | Pre- and/or Post- Task hooks are not used. |
| NS | No context switch is possible. |
| NS1i | No context switch is possible, idle task is only ECC task. |
| NS2 | Activated taskset has one BCC2 task. |
| NSH | Chain from heavyweight task, not to higher priority. |
| NSL | Chain from lightweight task, not to higher priority. |
| Shared | Resource is used by tasks and ISRs. |
| SW | A context switch is made if required. |
| SW2 | Activated taskset has one BCC2 task. |
| SWH | Chain from heavyweight task to possibly higher priority. |
| SWL | Chain from lightweight task to possibly higher priority. |

| Variant | Description |
|---|---|
| Task | Resource is used only by tasks. |

## Standard

| Configuration | | | Application Uses | | | | | |
|---|---|---|---|---|---|---|---|---|
| Events | | | No | | | Yes | | |
| Shared Task Priorities | | | No | | Yes | No | | Yes |
| Multiple Task Activations | | | No | Yes | | No | Yes | |
| Service name | Variant | Notes | | | | | | |
| ActivateTask | SW | 1 | 142 | 198 | 238 | 148 | 204 | 272 |
| | NS | | 126 | 182 | 222 | 132 | 188 | 256 |
| | KL | 2 | 68 | 124 | 164 | 74 | 130 | 202 |
| TerminateTask | LExt | 3 | n/a | n/a | n/a | n/a | n/a | n/a |
| | H | 5 | 14 | 14 | 14 | 14 | 14 | 14 |
| ChainTask | SWL | 1, 8 | 138 | 192 | 228 | 144 | 198 | 268 |
| | SWH | 1, 9 | 168 | 224 | 260 | 174 | 230 | 300 |
| | NSL | 8 | 138 | 192 | 228 | 144 | 198 | 268 |
| | NSH | 9 | 156 | 212 | 248 | 162 | 218 | 288 |
| Schedule | | | 94 | 94 | 122 | 94 | 94 | 122 |
| GetTaskID | | | 30 | 30 | 30 | 30 | 30 | 30 |
| GetTaskState | | | 80 | 80 | 80 | 92 | 92 | 92 |
| EnableAllInterrupts | | | 30 | 30 | 30 | 30 | 30 | 30 |
| DisableAllInterrupts | | | 38 | 38 | 38 | 38 | 38 | 38 |
| ResumeAllInterrupts | | | 42 | 42 | 42 | 42 | 42 | 42 |
| SuspendAllInterrupts | | | 54 | 54 | 54 | 54 | 54 | 54 |
| ResumeOSInterrupts | | | 42 | 42 | 42 | 42 | 42 | 42 |
| SuspendOSInterrupts | | | 58 | 58 | 58 | 58 | 58 | 58 |
| GetResource | Task | 7 | 22 | 22 | 28 | 22 | 22 | 28 |
| | Combined | 6 | 72 | 72 | 72 | 72 | 72 | 72 |
| | CLEx | 3 | n/a | n/a | n/a | n/a | n/a | n/a |
| ReleaseResource | Task | 7 | 76 | 76 | 76 | 76 | 76 | 76 |
| | Combined | 6 | 176 | 176 | 176 | 176 | 176 | 176 |
| | CLEx | 3 | n/a | n/a | n/a | n/a | n/a | n/a |
| SetEvent | SW | 1 | n/a | n/a | n/a | 124 | 124 | 200 |
| | NS | | n/a | n/a | n/a | 106 | 106 | 182 |
| | NS1i | 10 | n/a | n/a | n/a | 66 | n/a | n/a |
| | KL | 2 | n/a | n/a | n/a | 62 | 62 | 140 |
| | KL1i | 2, 10 | n/a | n/a | n/a | 24 | n/a | n/a |
| ClearEvent | | | n/a | n/a | n/a | 72 | 72 | 72 |

| Configuration | | | Application Uses | | | | | |
|---|---|---|---|---|---|---|---|---|
| Events | | | No | | | Yes | | |
| Shared Task Priorities | | | No | | Yes | No | | Yes |
| Multiple Task Activations | | | No | Yes | | No | Yes | |
| GetEvent | | | n/a | n/a | n/a | 12 | 12 | 12 |
| WaitEvent | <default> | | n/a | n/a | n/a | 236 | 236 | 466 |
| | fp | 11 | n/a | n/a | n/a | 266 | 266 | 536 |
| | 1i | 10 | n/a | n/a | n/a | 18 | n/a | n/a |
| GetAlarmBase | | | 64 | 64 | 64 | 64 | 64 | 64 |
| GetAlarm | | | 92 | 92 | 92 | 92 | 92 | 92 |
| SetRelAlarm | | | 436 | 436 | 436 | 436 | 436 | 436 |
| SetAbsAlarm | | | 474 | 474 | 474 | 474 | 474 | 474 |
| CancelAlarm | | | 84 | 84 | 84 | 84 | 84 | 84 |
| InitCounter | | | 66 | 66 | 66 | 66 | 66 | 66 |
| GetCounterValue | | | 80 | 80 | 80 | 80 | 80 | 80 |
| GetScheduleTableStatus | | 34 | 60 | 80 | 80 | 60 | 80 | 80 |
| NextScheduleTable | | 34 | 92 | 182 | 182 | 92 | 182 | 182 |
| StartScheduleTable | | 34 | 124 | 166 | 166 | 124 | 166 | 166 |
| StopScheduleTable | | 34 | 98 | 118 | 118 | 98 | 118 | 118 |
| ScheduleTable expiry point | ActivateTask | | 20 | 20 | 20 | 20 | 20 | 20 |
| ScheduleTable expiry point | SetEvent | | n/a | n/a | n/a | 22 | 22 | 22 |
| ScheduleTable expiry point | Callback | | 6 | 6 | 6 | 6 | 6 | 6 |
| ScheduleTable expiry point | Tick counter | | 20 | 20 | 20 | 20 | 20 | 20 |
| ScheduleTable expiry point | Final | | 142 | 142 | 142 | 142 | 142 | 142 |
| GetISRID | | 4 | n/a | n/a | n/a | n/a | n/a | n/a |
| Process container | Yielding | 32 | 62 | 62 | 62 | 62 | 62 | 62 |
| Process container | Non-Yielding | 33 | 40 | 40 | 40 | 40 | 40 | 40 |
| osek_tick_alarm | <default> | | 72 | 72 | 72 | 72 | 72 | 72 |
| | KL | 2 | 32 | 32 | 32 | 32 | 32 | 32 |
| osek_incr_counter | | | 92 | 92 | 92 | 92 | 92 | 92 |
| GetActiveApplicationMode | | 30 | n/a | n/a | n/a | n/a | n/a | n/a |
| StartOS | | | 268 | 268 | 268 | 268 | 268 | 268 |
| ShutdownOS | NoHook | 12 | 44 | 44 | 44 | 44 | 44 | 44 |
| | Hook | 13 | 58 | 58 | 58 | 58 | 58 | 58 |
| InitCOM | | | 6 | 6 | 6 | 6 | 6 | 6 |
| CloseCOM | | | 6 | 6 | 6 | 6 | 6 | 6 |
| StartCOM | | | 28 | 28 | 28 | 28 | 28 | 28 |
| StopCOM | | | 18 | 18 | 18 | 18 | 18 | 18 |
| ReadFlag | | 30 | n/a | n/a | n/a | n/a | n/a | n/a |
| ResetFlag | | 30 | n/a | n/a | n/a | n/a | n/a | n/a |
| ReceiveMessage | CCCA | 14 | 66 | 66 | 66 | 166 | 166 | 166 |

| Configuration | | | Application Uses | | | | | |
|---|---|---|---|---|---|---|---|---|
| Events | | | No | | | Yes | | |
| Shared Task Priorities | | | No | | Yes | No | | Yes |
| Multiple Task Activations | | | No | Yes | | No | Yes | |
| | CCCB | 15 | 166 | 166 | 166 | 166 | 166 | 166 |
| GetMessageResource | | | 34 | 34 | 34 | 34 | 34 | 34 |
| ReleaseMessageResource | | | 34 | 34 | 34 | 34 | 34 | 34 |
| GetMessageStatus | | | 42 | 42 | 42 | 42 | 42 | 42 |
| SendMessage | SW CCCA | 1, 14 | 90 | 90 | 90 | 214 | 214 | 214 |
| | SW CCCB | 1, 15 | 200 | 200 | 200 | 214 | 214 | 214 |
| | NS CCCA | 14 | 90 | 90 | 90 | 214 | 214 | 214 |
| | NS CCCB | 15 | 200 | 200 | 200 | 214 | 214 | 214 |
| | KL CCCA | 2, 14 | 48 | 48 | 48 | 172 | 172 | 172 |
| | KL CCCB | 2, 15 | 158 | 158 | 158 | 172 | 172 | 172 |
| main_dispatch | NoHook | 12 | 134 | 134 | 172 | 134 | 134 | 172 |
| | Hook | 13 | 170 | 170 | 208 | 170 | 170 | 208 |
| sub_dispatch | B1LF | 19 | 20 | 20 | 20 | 20 | 20 | 20 |
| | B1HI | 20 | 98 | 98 | 98 | 98 | 98 | 98 |
| | B1HF | 21 | 104 | 104 | 104 | 104 | 104 | 104 |
| | B2LI | 22 | n/a | 86 | 110 | n/a | 86 | 110 |
| | B2LF | 23 | n/a | 94 | 118 | n/a | 94 | 118 |
| | B2HI | 24 | n/a | 290 | 362 | n/a | 290 | 362 |
| | B2HF | 25 | n/a | 296 | 370 | n/a | 296 | 370 |
| | E1HI | 26 | n/a | n/a | n/a | 424 | 424 | 492 |
| | E1HF | 27 | n/a | n/a | n/a | 432 | 432 | 498 |
| | E2HI | 28 | n/a | n/a | n/a | n/a | n/a | 492 |
| | E2HF | 29 | n/a | n/a | n/a | n/a | n/a | 498 |
| ErrorHook support | | 16 | 38 | 38 | 38 | 38 | 38 | 38 |
| | ServiceID | 17 | 46 | 46 | 46 | 46 | 46 | 46 |
| | Parameters | 18 | 66 | 66 | 66 | 66 | 66 | 66 |
| validity_checks | | 3 | n/a | n/a | n/a | n/a | n/a | n/a |
| Timing_dispatch | | 4 | n/a | n/a | n/a | n/a | n/a | n/a |
| Timing_termination | | 4 | n/a | n/a | n/a | n/a | n/a | n/a |
| ActivateTaskset | SW | 1 | 138 | 280 | 316 | 160 | 322 | 392 |
| | NS | | 122 | 264 | 300 | 144 | 306 | 376 |
| | KL | 2 | 64 | 204 | 240 | 86 | 246 | 318 |
| ChainTaskset | SWL | 1, 8 | 152 | 316 | 354 | 160 | 344 | 418 |
| | SWH | 1, 9 | 186 | 362 | 406 | 194 | 392 | 466 |
| | NSL | 8 | 152 | 316 | 354 | 160 | 344 | 418 |
| | NSH | 9 | 172 | 350 | 394 | 180 | 380 | 454 |
| GetTasksetRef | | | 8 | 8 | 8 | 8 | 8 | 8 |

| Configuration | | | Application Uses | | | | | |
|---|---|---|---|---|---|---|---|---|
| Events | | | No | | | Yes | | |
| Shared Task Priorities | | | No | | Yes | No | | Yes |
| Multiple Task Activations | | | No | Yes | | No | Yes | |
| MergeTaskset | | | 54 | 54 | 54 | 54 | 54 | 54 |
| AssignTaskset | | | 8 | 8 | 8 | 8 | 8 | 8 |
| RemoveTaskset | | | 64 | 64 | 64 | 64 | 64 | 64 |
| TestSubTaskset | | | 56 | 56 | 56 | 56 | 56 | 56 |
| TestEquivalentTaskset | | | 54 | 54 | 54 | 54 | 54 | 54 |
| TickSchedule | SW | 1 | 190 | 162 | 162 | 162 | 162 | 162 |
| | NS | | 168 | 138 | 138 | 138 | 138 | 138 |
| | KL | 2 | 124 | 94 | 94 | 94 | 94 | 94 |
| AdvanceSchedule | SW | 1 | 182 | 148 | 148 | 148 | 148 | 148 |
| | NS | | 160 | 128 | 128 | 128 | 128 | 128 |
| | KL | 2 | 116 | 84 | 84 | 84 | 84 | 84 |
| StartSchedule | | | 80 | 80 | 80 | 80 | 80 | 80 |
| StopSchedule | | | 66 | 66 | 66 | 66 | 66 | 66 |
| GetScheduleStatus | | | 94 | 94 | 94 | 94 | 94 | 94 |
| GetScheduleValue | | | 78 | 78 | 78 | 78 | 78 | 78 |
| GetScheduleNext | | | 12 | 12 | 12 | 12 | 12 | 12 |
| SetScheduleNext | | | 8 | 8 | 8 | 8 | 8 | 8 |
| GetArrivalpointDelay | | | 8 | 8 | 8 | 8 | 8 | 8 |
| SetArrivalpointDelay | | | 8 | 8 | 8 | 8 | 8 | 8 |
| GetArrivalpointTasksetRef | | | 6 | 6 | 6 | 6 | 6 | 6 |
| GetArrivalpointNext | | | 8 | 8 | 8 | 8 | 8 | 8 |
| SetArrivalpointNext | | | 8 | 8 | 8 | 8 | 8 | 8 |
| TestArrivalpointWritable | | | 36 | 36 | 36 | 36 | 36 | 36 |
| GetExecutionTime | | | 6 | 6 | 6 | 6 | 6 | 6 |
| GetLargestExecutionTime | | | 8 | 8 | 8 | 8 | 8 | 8 |
| ResetLargestExecutionTime | | | 6 | 6 | 6 | 6 | 6 | 6 |
| GetStackOffset | | | 18 | 18 | 18 | 18 | 18 | 18 |

## Timing

| Configuration | | | Application Uses | | | | | |
|---|---|---|---|---|---|---|---|---|
| Events | | | No | | | Yes | | |
| Shared Task Priorities | | | No | | Yes | No | | Yes |
| Multiple Task Activations | | | No | Yes | | No | Yes | |
| Service name | Variant | Notes | | | | | | |
| ActivateTask | SW | 1 | 142 | 198 | 238 | 148 | 204 | 272 |
| | NS | | 126 | 182 | 222 | 132 | 188 | 256 |
| | KL | 2 | 68 | 124 | 164 | 74 | 130 | 202 |
| TerminateTask | LExt | 3 | n/a | n/a | n/a | n/a | n/a | n/a |
| | H | 5 | 14 | 14 | 14 | 14 | 14 | 14 |
| ChainTask | SWL | 1, 8 | 138 | 192 | 228 | 144 | 198 | 268 |
| | SWH | 1, 9 | 168 | 224 | 260 | 174 | 230 | 300 |
| | NSL | 8 | 138 | 192 | 228 | 144 | 198 | 268 |
| | NSH | 9 | 156 | 212 | 248 | 162 | 218 | 288 |
| Schedule | | | 116 | 116 | 144 | 116 | 116 | 144 |
| GetTaskID | | | 30 | 30 | 30 | 30 | 30 | 30 |
| GetTaskState | | | 80 | 80 | 80 | 92 | 92 | 92 |
| EnableAllInterrupts | | | 30 | 30 | 30 | 30 | 30 | 30 |
| DisableAllInterrupts | | | 38 | 38 | 38 | 38 | 38 | 38 |
| ResumeAllInterrupts | | | 42 | 42 | 42 | 42 | 42 | 42 |
| SuspendAllInterrupts | | | 54 | 54 | 54 | 54 | 54 | 54 |
| ResumeOSInterrupts | | | 42 | 42 | 42 | 42 | 42 | 42 |
| SuspendOSInterrupts | | | 58 | 58 | 58 | 58 | 58 | 58 |
| GetResource | Task | 7 | 22 | 22 | 28 | 22 | 22 | 28 |
| | Combined | 6 | 72 | 72 | 72 | 72 | 72 | 72 |
| | CLEx | 3 | n/a | n/a | n/a | n/a | n/a | n/a |
| ReleaseResource | Task | 7 | 100 | 100 | 100 | 100 | 100 | 100 |
| | Combined | 6 | 222 | 222 | 222 | 222 | 222 | 222 |
| | CLEx | 3 | n/a | n/a | n/a | n/a | n/a | n/a |
| SetEvent | SW | 1 | n/a | n/a | n/a | 124 | 124 | 200 |
| | NS | | n/a | n/a | n/a | 106 | 106 | 182 |
| | NS1i | 10 | n/a | n/a | n/a | 66 | n/a | n/a |
| | KL | 2 | n/a | n/a | n/a | 62 | 62 | 140 |
| | KL1i | 2, 10 | n/a | n/a | n/a | 24 | n/a | n/a |
| ClearEvent | | | n/a | n/a | n/a | 72 | 72 | 72 |
| GetEvent | | | n/a | n/a | n/a | 12 | 12 | 12 |
| WaitEvent | <default> | | n/a | n/a | n/a | 350 | 350 | 582 |
| | fp | 11 | n/a | n/a | n/a | 372 | 372 | 632 |

| Configuration | | | Application Uses | | | | | |
|---|---|---|---|---|---|---|---|---|
| Events | | | No | | | Yes | | |
| Shared Task Priorities | | | No | | Yes | No | | Yes |
| Multiple Task Activations | | | No | Yes | | No | Yes | |
| | 1i | 10 | n/a | n/a | n/a | 122 | n/a | n/a |
| GetAlarmBase | | | 64 | 64 | 64 | 64 | 64 | 64 |
| GetAlarm | | | 92 | 92 | 92 | 92 | 92 | 92 |
| SetRelAlarm | | | 436 | 436 | 436 | 436 | 436 | 436 |
| SetAbsAlarm | | | 474 | 474 | 474 | 474 | 474 | 474 |
| CancelAlarm | | | 84 | 84 | 84 | 84 | 84 | 84 |
| InitCounter | | | 66 | 66 | 66 | 66 | 66 | 66 |
| GetCounterValue | | | 80 | 80 | 80 | 80 | 80 | 80 |
| GetScheduleTableStatus | | 34 | 60 | 80 | 80 | 60 | 80 | 80 |
| NextScheduleTable | | 34 | 92 | 182 | 182 | 92 | 182 | 182 |
| StartScheduleTable | | 34 | 124 | 166 | 166 | 124 | 166 | 166 |
| StopScheduleTable | | 34 | 98 | 118 | 118 | 98 | 118 | 118 |
| ScheduleTable expiry point | ActivateTask | | 20 | 20 | 20 | 20 | 20 | 20 |
| ScheduleTable expiry point | SetEvent | | n/a | n/a | n/a | 22 | 22 | 22 |
| ScheduleTable expiry point | Callback | | 6 | 6 | 6 | 6 | 6 | 6 |
| ScheduleTable expiry point | Tick counter | | 20 | 20 | 20 | 20 | 20 | 20 |
| ScheduleTable expiry point | Final | | 142 | 142 | 142 | 142 | 142 | 142 |
| GetISRID | | 4 | 36 | 36 | 36 | 36 | 36 | 36 |
| Process container | Yielding | 32 | 62 | 62 | 62 | 62 | 62 | 62 |
| Process container | Non-Yielding | 33 | 40 | 40 | 40 | 40 | 40 | 40 |
| osek_tick_alarm | <default> | | 72 | 72 | 72 | 72 | 72 | 72 |
| | KL | 2 | 32 | 32 | 32 | 32 | 32 | 32 |
| osek_incr_counter | | | 92 | 92 | 92 | 92 | 92 | 92 |
| GetActiveApplicationMode | | 30 | n/a | n/a | n/a | n/a | n/a | n/a |
| StartOS | | | 338 | 338 | 338 | 338 | 338 | 338 |
| ShutdownOS | NoHook | 12 | 44 | 44 | 44 | 44 | 44 | 44 |
| | Hook | 13 | 58 | 58 | 58 | 58 | 58 | 58 |
| InitCOM | | | 6 | 6 | 6 | 6 | 6 | 6 |
| CloseCOM | | | 6 | 6 | 6 | 6 | 6 | 6 |
| StartCOM | | | 28 | 28 | 28 | 28 | 28 | 28 |
| StopCOM | | | 18 | 18 | 18 | 18 | 18 | 18 |
| ReadFlag | | 30 | n/a | n/a | n/a | n/a | n/a | n/a |
| ResetFlag | | 30 | n/a | n/a | n/a | n/a | n/a | n/a |
| ReceiveMessage | CCCA | 14 | 66 | 66 | 66 | 166 | 166 | 166 |
| | CCCB | 15 | 166 | 166 | 166 | 166 | 166 | 166 |
| GetMessageResource | | | 34 | 34 | 34 | 34 | 34 | 34 |
| ReleaseMessageResource | | | 34 | 34 | 34 | 34 | 34 | 34 |

| Configuration | | | Application Uses | | | | | |
|---|---|---|---|---|---|---|---|---|
| Events | | | No | | | Yes | | |
| Shared Task Priorities | | | No | | Yes | No | | Yes |
| Multiple Task Activations | | | No | Yes | | No | Yes | |
| GetMessageStatus | | | 42 | 42 | 42 | 42 | 42 | 42 |
| SendMessage | SW CCCA | 1, 14 | 90 | 90 | 90 | 214 | 214 | 214 |
| | SW CCCB | 1, 15 | 200 | 200 | 200 | 214 | 214 | 214 |
| | NS CCCA | 14 | 90 | 90 | 90 | 214 | 214 | 214 |
| | NS CCCB | 15 | 200 | 200 | 200 | 214 | 214 | 214 |
| | KL CCCA | 2, 14 | 48 | 48 | 48 | 172 | 172 | 172 |
| | KL CCCB | 2, 15 | 158 | 158 | 158 | 172 | 172 | 172 |
| main_dispatch | NoHook | 12 | 188 | 188 | 228 | 188 | 188 | 228 |
| | Hook | 13 | 230 | 230 | 270 | 230 | 230 | 270 |
| sub_dispatch | B1LF | 19 | 12 | 12 | 12 | 12 | 12 | 12 |
| | B1HI | 20 | 90 | 90 | 90 | 90 | 90 | 90 |
| | B1HF | 21 | 100 | 100 | 100 | 100 | 100 | 100 |
| | B2LI | 22 | n/a | 52 | 86 | n/a | 52 | 86 |
| | B2LF | 23 | n/a | 60 | 94 | n/a | 60 | 94 |
| | B2HI | 24 | n/a | 226 | 326 | n/a | 226 | 326 |
| | B2HF | 25 | n/a | 234 | 334 | n/a | 234 | 334 |
| | E1HI | 26 | n/a | n/a | n/a | 422 | 422 | 490 |
| | E1HF | 27 | n/a | n/a | n/a | 430 | 430 | 496 |
| | E2HI | 28 | n/a | n/a | n/a | n/a | n/a | 490 |
| | E2HF | 29 | n/a | n/a | n/a | n/a | n/a | 496 |
| ErrorHook support | | 16 | 38 | 38 | 38 | 38 | 38 | 38 |
| | ServiceID | 17 | 46 | 46 | 46 | 46 | 46 | 46 |
| | Parameters | 18 | 66 | 66 | 66 | 66 | 66 | 66 |
| validity_checks | | 3 | n/a | n/a | n/a | n/a | n/a | n/a |
| Timing_dispatch | | 4 | 84 | 84 | 84 | 84 | 84 | 84 |
| Timing_termination | | 4 | 82 | 82 | 82 | 82 | 82 | 82 |
| ActivateTaskset | SW | 1 | 138 | 280 | 316 | 160 | 322 | 392 |
| | NS | | 122 | 264 | 300 | 144 | 306 | 376 |
| | KL | 2 | 64 | 204 | 240 | 86 | 246 | 318 |
| ChainTaskset | SWL | 1, 8 | 152 | 316 | 354 | 160 | 344 | 418 |
| | SWH | 1, 9 | 186 | 362 | 406 | 194 | 392 | 466 |
| | NSL | 8 | 152 | 316 | 354 | 160 | 344 | 418 |
| | NSH | 9 | 172 | 350 | 394 | 180 | 380 | 454 |
| GetTasksetRef | | | 8 | 8 | 8 | 8 | 8 | 8 |
| MergeTaskset | | | 54 | 54 | 54 | 54 | 54 | 54 |
| AssignTaskset | | | 8 | 8 | 8 | 8 | 8 | 8 |
| RemoveTaskset | | | 64 | 64 | 64 | 64 | 64 | 64 |

| Configuration<br>Events<br>Shared Task Priorities<br>Multiple Task Activations | | | Application Uses | | | | | |
|---|---|---|---|---|---|---|---|---|
| | | | No | | | Yes | | |
| | | | No | | Yes | No | | Yes |
| | | | No | Yes | | No | Yes | |
| TestSubTaskset | | | 56 | 56 | 56 | 56 | 56 | 56 |
| TestEquivalentTaskset | | | 54 | 54 | 54 | 54 | 54 | 54 |
| TickSchedule | SW | 1 | 190 | 162 | 162 | 162 | 162 | 162 |
| | NS | | 168 | 138 | 138 | 138 | 138 | 138 |
| | KL | 2 | 124 | 94 | 94 | 94 | 94 | 94 |
| AdvanceSchedule | SW | 1 | 182 | 148 | 148 | 148 | 148 | 148 |
| | NS | | 160 | 128 | 128 | 128 | 128 | 128 |
| | KL | 2 | 116 | 84 | 84 | 84 | 84 | 84 |
| StartSchedule | | | 80 | 80 | 80 | 80 | 80 | 80 |
| StopSchedule | | | 66 | 66 | 66 | 66 | 66 | 66 |
| GetScheduleStatus | | | 94 | 94 | 94 | 94 | 94 | 94 |
| GetScheduleValue | | | 78 | 78 | 78 | 78 | 78 | 78 |
| GetScheduleNext | | | 12 | 12 | 12 | 12 | 12 | 12 |
| SetScheduleNext | | | 8 | 8 | 8 | 8 | 8 | 8 |
| GetArrivalpointDelay | | | 8 | 8 | 8 | 8 | 8 | 8 |
| SetArrivalpointDelay | | | 8 | 8 | 8 | 8 | 8 | 8 |
| GetArrivalpointTasksetRef | | | 6 | 6 | 6 | 6 | 6 | 6 |
| GetArrivalpointNext | | | 8 | 8 | 8 | 8 | 8 | 8 |
| SetArrivalpointNext | | | 8 | 8 | 8 | 8 | 8 | 8 |
| TestArrivalpointWritable | | | 36 | 36 | 36 | 36 | 36 | 36 |
| GetExecutionTime | | | 118 | 118 | 118 | 118 | 118 | 118 |
| GetLargestExecutionTime | | | 14 | 14 | 14 | 14 | 14 | 14 |
| ResetLargestExecutionTime | | | 12 | 12 | 12 | 12 | 12 | 12 |
| GetStackOffset | | | 18 | 18 | 18 | 18 | 18 | 18 |

## Extended

| Configuration<br>Events<br>Shared Task Priorities<br>Multiple Task Activations | | | Application Uses | | | | | |
|---|---|---|---|---|---|---|---|---|
| | | | No | | | Yes | | |
| | | | No | | Yes | No | | Yes |
| | | | No | Yes | | No | Yes | |
| Service name | Variant | Notes | | | | | | |
| ActivateTask | SW | 1 | 238 | 300 | 338 | 252 | 306 | 368 |
| | NS | | 274 | 336 | 374 | 288 | 342 | 404 |
| | KL | 2 | 130 | 192 | 230 | 144 | 198 | 260 |
| TerminateTask | LExt | 3 | 94 | 94 | 94 | 94 | 94 | 94 |

| Configuration | | | Application Uses | | | | | |
|---|---|---|---|---|---|---|---|---|
| Events | | | No | | | Yes | | |
| Shared Task Priorities | | | No | | Yes | No | | Yes |
| Multiple Task Activations | | | No | Yes | | No | Yes | |
| | H | 5 | 118 | 118 | 118 | 118 | 118 | 118 |
| ChainTask | SWL | 1, 8 | 262 | 328 | 366 | 278 | 334 | 400 |
| | SWH | 1, 9 | 298 | 368 | 404 | 314 | 374 | 434 |
| | NSL | 8 | 300 | 366 | 404 | 316 | 372 | 438 |
| | NSH | 9 | 330 | 400 | 436 | 346 | 406 | 466 |
| Schedule | | | 246 | 246 | 274 | 246 | 246 | 274 |
| GetTaskID | | | 46 | 46 | 46 | 46 | 46 | 46 |
| GetTaskState | | | 224 | 224 | 224 | 228 | 228 | 228 |
| EnableAllInterrupts | | | 46 | 46 | 46 | 46 | 46 | 46 |
| DisableAllInterrupts | | | 54 | 54 | 54 | 54 | 54 | 54 |
| ResumeAllInterrupts | | | 84 | 84 | 84 | 84 | 84 | 84 |
| SuspendAllInterrupts | | | 70 | 70 | 70 | 70 | 70 | 70 |
| ResumeOSInterrupts | | | 84 | 84 | 84 | 84 | 84 | 84 |
| SuspendOSInterrupts | | | 74 | 74 | 74 | 74 | 74 | 74 |
| GetResource | Task | 7 | 348 | 348 | 326 | 348 | 348 | 326 |
| | Combined | 6 | 330 | 330 | 330 | 330 | 330 | 330 |
| | CLEx | 3 | 298 | 298 | 298 | 298 | 298 | 298 |
| ReleaseResource | Task | 7 | 330 | 330 | 330 | 330 | 330 | 330 |
| | Combined | 6 | 450 | 450 | 450 | 450 | 450 | 450 |
| | CLEx | 3 | 292 | 292 | 292 | 292 | 292 | 292 |
| SetEvent | SW | 1 | n/a | n/a | n/a | 294 | 294 | 370 |
| | NS | | n/a | n/a | n/a | 328 | 328 | 404 |
| | NS1i | 10 | n/a | n/a | n/a | 228 | n/a | n/a |
| | KL | 2 | n/a | n/a | n/a | 186 | 186 | 264 |
| | KL1i | 2, 10 | n/a | n/a | n/a | 136 | n/a | n/a |
| ClearEvent | | | n/a | n/a | n/a | 138 | 138 | 138 |
| GetEvent | | | n/a | n/a | n/a | 138 | 138 | 138 |
| WaitEvent | <default> | | n/a | n/a | n/a | 452 | 452 | 674 |
| | fp | 11 | n/a | n/a | n/a | 480 | 480 | 726 |
| | 1i | 10 | n/a | n/a | n/a | 230 | n/a | n/a |
| GetAlarmBase | | | 194 | 194 | 194 | 194 | 194 | 194 |
| GetAlarm | | | 170 | 170 | 170 | 170 | 170 | 170 |
| SetRelAlarm | | | 554 | 554 | 554 | 554 | 554 | 554 |
| SetAbsAlarm | | | 588 | 588 | 588 | 588 | 588 | 588 |
| CancelAlarm | | | 160 | 160 | 160 | 160 | 160 | 160 |
| InitCounter | | | 230 | 230 | 230 | 230 | 230 | 230 |
| GetCounterValue | | | 212 | 212 | 212 | 212 | 212 | 212 |

| Configuration | | | Application Uses | | | | | |
|---|---|---|---|---|---|---|---|---|
| Events | | | No | | | Yes | | |
| Shared Task Priorities | | | No | | Yes | No | | Yes |
| Multiple Task Activations | | | No | Yes | | No | Yes | |
| GetScheduleTableStatus | | 34 | 80 | 100 | 100 | 80 | 100 | 100 |
| NextScheduleTable | | 34 | 112 | 206 | 206 | 112 | 206 | 206 |
| StartScheduleTable | | 34 | 148 | 188 | 188 | 148 | 188 | 188 |
| StopScheduleTable | | 34 | 118 | 138 | 138 | 118 | 138 | 138 |
| ScheduleTable expiry point | ActivateTask | | 20 | 20 | 20 | 20 | 20 | 20 |
| ScheduleTable expiry point | SetEvent | | n/a | n/a | n/a | 22 | 22 | 22 |
| ScheduleTable expiry point | Callback | | 6 | 6 | 6 | 6 | 6 | 6 |
| ScheduleTable expiry point | Tick counter | | 20 | 20 | 20 | 20 | 20 | 20 |
| ScheduleTable expiry point | Final | | 160 | 160 | 160 | 160 | 160 | 160 |
| GetISRID | | 4 | 52 | 52 | 52 | 52 | 52 | 52 |
| Process container | Yielding | 32 | 62 | 62 | 62 | 62 | 62 | 62 |
| Process container | Non-Yielding | 33 | 40 | 40 | 40 | 40 | 40 | 40 |
| osek_tick_alarm | <default> | | 116 | 116 | 116 | 116 | 116 | 116 |
| | KL | 2 | 32 | 32 | 32 | 32 | 32 | 32 |
| osek_incr_counter | | | 92 | 92 | 92 | 92 | 92 | 92 |
| GetActiveApplicationMode | | 30 | n/a | n/a | n/a | n/a | n/a | n/a |
| StartOS | | | 354 | 354 | 354 | 354 | 354 | 354 |
| ShutdownOS | NoHook | 12 | 54 | 54 | 54 | 54 | 54 | 54 |
| | Hook | 13 | 68 | 68 | 68 | 68 | 68 | 68 |
| InitCOM | | | 6 | 6 | 6 | 6 | 6 | 6 |
| CloseCOM | | | 6 | 6 | 6 | 6 | 6 | 6 |
| StartCOM | | | 46 | 46 | 46 | 46 | 46 | 46 |
| StopCOM | | | 46 | 46 | 46 | 46 | 46 | 46 |
| ReadFlag | | | 32 | 32 | 32 | 32 | 32 | 32 |
| ResetFlag | | | 36 | 36 | 36 | 36 | 36 | 36 |
| ReceiveMessage | CCCA | 14 | 166 | 166 | 166 | 266 | 266 | 266 |
| | CCCB | 15 | 266 | 266 | 266 | 266 | 266 | 266 |
| GetMessageResource | | | 76 | 76 | 76 | 76 | 76 | 76 |
| ReleaseMessageResource | | | 76 | 76 | 76 | 76 | 76 | 76 |
| GetMessageStatus | | | 82 | 82 | 82 | 82 | 82 | 82 |
| SendMessage | SW CCCA | 1, 14 | 214 | 214 | 214 | 332 | 332 | 332 |
| | SW CCCB | 1, 15 | 318 | 318 | 318 | 332 | 332 | 332 |
| | NS CCCA | 14 | 214 | 214 | 214 | 332 | 332 | 332 |
| | NS CCCB | 15 | 318 | 318 | 318 | 332 | 332 | 332 |
| | KL CCCA | 2, 14 | 122 | 122 | 122 | 240 | 240 | 240 |
| | KL CCCB | 2, 15 | 226 | 226 | 226 | 240 | 240 | 240 |
| main_dispatch | NoHook | 12 | 188 | 188 | 228 | 188 | 188 | 228 |

| Configuration | | | Application Uses | | | | | |
|---|---|---|---|---|---|---|---|---|
| Events | | | No | | | Yes | | |
| Shared Task Priorities | | | No | | Yes | No | | Yes |
| Multiple Task Activations | | | No | Yes | | No | Yes | |
| | Hook | 13 | 230 | 230 | 270 | 230 | 230 | 270 |
| sub_dispatch | B1LF | 19 | 12 | 12 | 12 | 12 | 12 | 12 |
| | B1HI | 20 | 90 | 90 | 90 | 90 | 90 | 90 |
| | B1HF | 21 | 100 | 100 | 100 | 100 | 100 | 100 |
| | B2LI | 22 | n/a | 50 | 84 | n/a | 50 | 84 |
| | B2LF | 23 | n/a | 58 | 92 | n/a | 58 | 92 |
| | B2HI | 24 | n/a | 224 | 322 | n/a | 224 | 322 |
| | B2HF | 25 | n/a | 232 | 330 | n/a | 232 | 330 |
| | E1HI | 26 | n/a | n/a | n/a | 422 | 422 | 490 |
| | E1HF | 27 | n/a | n/a | n/a | 430 | 430 | 496 |
| | E2HI | 28 | n/a | n/a | n/a | n/a | n/a | 490 |
| | E2HF | 29 | n/a | n/a | n/a | n/a | n/a | 496 |
| ErrorHook support | | 16 | 126 | 126 | 126 | 126 | 126 | 126 |
| | ServiceID | 17 | 134 | 134 | 134 | 134 | 134 | 134 |
| | Parameters | 18 | 154 | 154 | 154 | 154 | 154 | 154 |
| validity_checks | | 3 | 22 | 22 | 22 | 22 | 22 | 22 |
| Timing_dispatch | | 4 | 84 | 84 | 84 | 84 | 84 | 84 |
| Timing_termination | | 4 | 82 | 82 | 82 | 82 | 82 | 82 |
| ActivateTaskset | SW | 1 | 320 | 404 | 448 | 370 | 442 | 492 |
| | NS | | 362 | 434 | 478 | 404 | 472 | 526 |
| | KL | 2 | 214 | 300 | 344 | 262 | 336 | 396 |
| ChainTaskset | SWL | 1, 8 | 364 | 448 | 486 | 402 | 472 | 528 |
| | SWH | 1, 9 | 406 | 514 | 546 | 444 | 538 | 586 |
| | NSL | 8 | 422 | 496 | 534 | 448 | 518 | 580 |
| | NSH | 9 | 452 | 548 | 580 | 478 | 570 | 626 |
| GetTasksetRef | | | 108 | 108 | 108 | 108 | 108 | 108 |
| MergeTaskset | | | 318 | 318 | 318 | 318 | 318 | 318 |
| AssignTaskset | | | 204 | 204 | 204 | 204 | 204 | 204 |
| RemoveTaskset | | | 328 | 328 | 328 | 328 | 328 | 328 |
| TestSubTaskset | | | 324 | 324 | 324 | 324 | 324 | 324 |
| TestEquivalentTaskset | | | 322 | 322 | 322 | 322 | 322 | 322 |
| TickSchedule | SW | 1 | 326 | 288 | 288 | 288 | 288 | 288 |
| | NS | | 358 | 344 | 344 | 344 | 344 | 344 |
| | KL | 2 | 236 | 204 | 204 | 204 | 204 | 204 |
| AdvanceSchedule | SW | 1 | 328 | 296 | 296 | 296 | 296 | 296 |
| | NS | | 360 | 352 | 352 | 352 | 352 | 352 |
| | KL | 2 | 244 | 216 | 216 | 216 | 216 | 216 |

| Configuration | | | Application Uses | | | | | |
|---|---|---|---|---|---|---|---|---|
| Events | | | No | | | Yes | | |
| Shared Task Priorities | | | No | | Yes | No | | Yes |
| Multiple Task Activations | | | No | Yes | | No | Yes | |
| StartSchedule | | | 248 | 248 | 248 | 248 | 248 | 248 |
| StopSchedule | | | 210 | 210 | 210 | 210 | 210 | 210 |
| GetScheduleStatus | | | 240 | 240 | 240 | 240 | 240 | 240 |
| GetScheduleValue | | | 192 | 192 | 192 | 192 | 192 | 192 |
| GetScheduleNext | | | 90 | 90 | 90 | 90 | 90 | 90 |
| SetScheduleNext | | | 178 | 178 | 178 | 178 | 178 | 178 |
| GetArrivalpointDelay | | | 128 | 128 | 128 | 128 | 128 | 128 |
| SetArrivalpointDelay | | | 144 | 144 | 144 | 144 | 144 | 144 |
| GetArrivalpointTasksetRef | | | 126 | 126 | 126 | 126 | 126 | 126 |
| GetArrivalpointNext | | | 128 | 128 | 128 | 128 | 128 | 128 |
| SetArrivalpointNext | | | 216 | 216 | 216 | 216 | 216 | 216 |
| TestArrivalpointWritable | | | 144 | 144 | 144 | 144 | 144 | 144 |
| GetExecutionTime | | | 172 | 172 | 172 | 172 | 172 | 172 |
| GetLargestExecutionTime | | | 94 | 94 | 94 | 94 | 94 | 94 |
| ResetLargestExecutionTime | | | 88 | 88 | 88 | 88 | 88 | 88 |
| GetStackOffset | | | 18 | 18 | 18 | 18 | 18 | 18 |

## Notes

| Number | Note |
|---|---|
| 1 | Linked only if upward activations are allowed |
| 2 | Linked only if API is called within ISR |
| 3 | Present only in Extended OS status |
| 4 | Present only in Timing or Extended OS status |
| 5 | Linked only if there are heavyweight tasks in the system |
| 6 | Linked only if Resource is used by both tasks and ISRs |
| 7 | Linked only if Resource is used only by tasks |
| 8 | Linked only if Chaining task is Lightweight |
| 9 | Linked only if Chaining task is Heavyweight |
| 10 | Linked only if Idle task is the only extended task in the system |
| 11 | Linked only if calling Extended task uses floating-point |
| 12 | Linked only if neither Pre- nor Post-TaskHook is used |
| 13 | Linked only if Pre- or Post-TaskHook is used |
| 14 | Linked only if there are no flags, message queues, or message resources in the system, and COM status is not requested. |
| 15 | Linked only if there are any flags, message queues, or message resources in the system, or COM status is requested. |

| Number | Note |
|---|---|
| 16 | Linked only if `USEGETSERVICEID = FALSE` and `USEPARAMETERACCESS = FALSE` |
| 17 | Linked only if `USEGETSERVICEID = TRUE` and `USEPARAMETERACCESS = FALSE` |
| 18 | Linked only if `USEGETSERVICEID = TRUE` and `USEPARAMETERACCESS = TRUE` |
| 19 | Linked only for basic, single-activation, lightweight, floating-point tasks |
| 20 | Linked only for basic, single-activation, heavyweight, integer tasks |
| 21 | Linked only for basic, single-activation, heavyweight, floating-point tasks |
| 22 | Linked only for basic, multiple-activation, lightweight, integer tasks |
| 23 | Linked only for basic, multiple-activation, lightweight, floating-point tasks |
| 24 | Linked only for basic, multiple-activation, heavyweight, integer tasks |
| 25 | Linked only for basic, multiple-activation, heavyweight, floating-point tasks |
| 26 | Linked only for extended, unique priority, integer tasks |
| 27 | Linked only for extended, unique priority, floating-point tasks |
| 28 | Linked only for extended, shared priority, integer tasks |
| 29 | Linked only for extended, shared priority, floating-point tasks |
| 30 | Implemented as a macro, so no code is linked |
| 31 | Not required on some targets |
| 32 | Container for 2 process functions, not highest priority |
| 33 | Container for 2 process functions, highest or APPMODE or ISR |
| 34 | code varies with number of schedule tables; example uses 2 schedule tables |

### 4.2.4 Reserved Hardware Resources

Timer units, interrupts, traps and other hardware resources are not reserved by RTA-OSEK.

## 4.3 Performance

### 4.3.1 Execution Times for RTA-OSEK API Calls

The following tables give the execution time (in CPU cycles) for each API call. (Note that: (1) the OSEK COM class was set to CCCA for systems without events and to CCCB for systems with events; (2) `ShutdownOS()` enters an infinite loop; the execution time for `ShutdownOS()` reported below is the time up to the point at which `ShutdownOS()` calls `ShutdownHook()`).

## Standard

| Configuration | | Application Uses | | | | | |
|---|---|---|---|---|---|---|---|
| Events | | No | | | Yes | | |
| Shared Task Priorities | | No | | Yes | No | | Yes |
| Multiple Task Activations | | No | Yes | | No | Yes | |
| Service | Variant | | | | | | |
| ActivateTask | SW | 60 | 79 | 91 | 61 | 73 | 97 |
| | NS | 53 | 73 | 84 | 54 | 65 | 88 |
| | KL | 40 | 59 | 71 | 41 | 54 | 74 |
| TerminateTask | LExt | 0 | 0 | 0 | 0 | 0 | 0 |
| | H | 158 | 159 | 158 | 159 | 157 | 159 |
| ChainTask | SWL | 178 | 202 | 235 | 203 | 213 | 261 |
| | SWH | 273 | 297 | 331 | 299 | 309 | 353 |
| | NSL | 179 | 202 | 235 | 203 | 213 | 259 |
| | NSH | 270 | 294 | 328 | 296 | 306 | 350 |
| Schedule | SW | 49 | 48 | 58 | 50 | 49 | 58 |
| GetTaskID | | 32 | 32 | 32 | 50 | 50 | 50 |
| GetTaskState | | 47 | 47 | 47 | 70 | 70 | 70 |
| EnableAllInterrupts | | 33 | 33 | 33 | 33 | 33 | 33 |
| DisableAllInterrupts | | 30 | 30 | 30 | 30 | 30 | 30 |
| ResumeAllInterrupts | | 39 | 39 | 39 | 39 | 39 | 39 |
| SuspendAllInterrupts | | 31 | 31 | 31 | 31 | 32 | 32 |
| ResumeOSInterrupts | | 38 | 38 | 38 | 38 | 38 | 38 |
| SuspendOSInterrupts | | 31 | 31 | 31 | 31 | 32 | 32 |
| GetResource | Task | 33 | 33 | 33 | 33 | 33 | 33 |
| | Combined | 41 | 40 | 40 | 40 | 40 | 40 |
| | CLEx | n/a | n/a | n/a | n/a | n/a | n/a |
| ReleaseResource | Task | 42 | 43 | 42 | 42 | 44 | 43 |
| | Combined | 58 | 58 | 58 | 58 | 59 | 58 |
| | CLEx | n/a | n/a | n/a | n/a | n/a | n/a |
| SetEvent | SW | n/a | n/a | n/a | 53 | 54 | 55 |
| | NS | n/a | n/a | n/a | 48 | 47 | 48 |
| | KL | n/a | n/a | n/a | 38 | 38 | 38 |
| ClearEvent | | n/a | n/a | n/a | 38 | 38 | 38 |
| GetEvent | | n/a | n/a | n/a | 25 | 25 | 25 |
| WaitEvent | <default> | n/a | n/a | n/a | 324 | 324 | 347 |
| | fp | n/a | n/a | n/a | 329 | 329 | 351 |
| GetAlarmBase | | 44 | 42 | 42 | 41 | 43 | 41 |
| GetAlarm | | 48 | 48 | 48 | 48 | 49 | 48 |

| Configuration | | Application Uses | | | | | |
|---|---|---|---|---|---|---|---|
| Events | | No | | | Yes | | |
| Shared Task Priorities | | No | | Yes | No | | Yes |
| Multiple Task Activations | | No | Yes | | No | Yes | |
| SetRelAlarm | | 60 | 60 | 60 | 61 | 61 | 60 |
| SetAbsAlarm | | 61 | 61 | 60 | 61 | 60 | 59 |
| CancelAlarm | | 41 | 41 | 42 | 41 | 41 | 41 |
| InitCounter | | 40 | 40 | 40 | 41 | 40 | 40 |
| GetCounterValue | | 46 | 46 | 46 | 46 | 46 | 46 |
| osek_tick_alarm | <default> | 47 | 48 | 46 | 47 | 47 | 48 |
| | KL | 34 | 34 | 34 | 35 | 35 | 35 |
| osek_incr_counter | | 16 | 16 | 16 | 16 | 16 | 16 |
| GetActiveApplicationMode | | 14 | 14 | 14 | 15 | 15 | 15 |
| StartOS | | 1598 | 1584 | 1597 | 1597 | 1584 | 1585 |
| ShutdownOS | NoHook | n/a | n/a | n/a | n/a | n/a | n/a |
| | Hook | 25 | 25 | 25 | 25 | 25 | 25 |
| InitCOM | | 18 | 18 | 18 | 19 | 19 | 19 |
| CloseCOM | | 18 | 18 | 18 | 18 | 18 | 18 |
| StartCOM | | 37 | 37 | 37 | 103 | 103 | 106 |
| StopCOM | | 22 | 23 | 22 | 22 | 22 | 22 |
| ReadFlag | | n/a | n/a | n/a | 22 | 19 | 19 |
| ResetFlag | | n/a | n/a | n/a | 17 | 17 | 17 |
| ReceiveMessage | | 35 | 35 | 35 | 123 | 122 | 123 |
| GetMessageResource | | n/a | n/a | n/a | 52 | 52 | 52 |
| ReleaseMessageResource | | n/a | n/a | n/a | 65 | 65 | 66 |
| GetMessageStatus | | n/a | n/a | n/a | 35 | 35 | 35 |
| SendMessage | SW | 99 | 118 | 130 | 182 | 193 | 217 |
| | NS | 92 | 113 | 124 | 174 | 187 | 208 |
| | KL | 65 | 84 | 96 | 146 | 160 | 179 |
| ActivateTaskset | SW | 54 | 289 | 269 | 55 | 290 | 275 |
| | NS | 46 | 279 | 261 | 47 | 281 | 298 |
| | KL | 34 | 269 | 313 | 33 | 269 | 256 |
| | SW2 | 54 | 289 | 270 | 55 | 290 | 275 |
| | NS2 | 46 | 279 | 261 | 47 | 281 | 298 |
| | KL2 | 34 | 269 | 313 | 33 | 271 | 256 |
| ChainTaskset | SWL | 172 | 439 | 473 | 197 | 428 | 468 |
| | SWH | 267 | 507 | 541 | 293 | 524 | 536 |
| | NSL | 175 | 410 | 442 | 196 | 427 | 437 |
| | NSH | 264 | 505 | 538 | 289 | 490 | 563 |
| GetTasksetRef | | 43 | 43 | 43 | 43 | 43 | 43 |

| Configuration | | Application Uses | | | | | |
|---|---|---|---|---|---|---|---|
| Events | | No | | | Yes | | |
| Shared Task Priorities | | No | | Yes | No | | Yes |
| Multiple Task Activations | | No | Yes | | No | Yes | |
| MergeTaskset | | 57 | 57 | 57 | 57 | 57 | 58 |
| AssignTaskset | | 40 | 40 | 40 | 40 | 40 | 40 |
| RemoveTaskset | | 55 | 55 | 55 | 55 | 55 | 55 |
| TestSubTaskset | | 57 | 57 | 57 | 57 | 57 | 57 |
| TestEquivalentTaskset | | 54 | 55 | 54 | 54 | 55 | 54 |
| TickSchedule | SW | 72 | 324 | 367 | 92 | 330 | 318 |
| | NS | 65 | 315 | 359 | 80 | 321 | 309 |
| | KL | 51 | 301 | 346 | 68 | 307 | 296 |
| | SW2 | 72 | 324 | 367 | 92 | 326 | 313 |
| | NS2 | 65 | 315 | 359 | 80 | 317 | 304 |
| | KL2 | 51 | 301 | 346 | 68 | 303 | 291 |
| AdvanceSchedule | SW | 72 | 321 | 366 | 89 | 329 | 317 |
| | NS | 62 | 312 | 358 | 80 | 319 | 308 |
| | KL | 51 | 301 | 345 | 67 | 307 | 295 |
| | SW2 | 71 | 321 | 366 | 89 | 324 | 312 |
| | NS2 | 62 | 312 | 357 | 80 | 315 | 303 |
| | KL2 | 51 | 301 | 345 | 67 | 303 | 290 |
| StartSchedule | | 68 | 68 | 69 | 69 | 68 | 68 |
| StopSchedule | | 64 | 64 | 64 | 63 | 63 | 63 |
| GetScheduleStatus | | 69 | 70 | 69 | 70 | 69 | 69 |
| GetScheduleValue | | 64 | 64 | 64 | 64 | 65 | 64 |
| GetScheduleNext | | 44 | 44 | 44 | 44 | 44 | 44 |
| SetScheduleNext | | 41 | 41 | 41 | 42 | 41 | 41 |
| GetArrivalpointDelay | | 42 | 42 | 42 | 43 | 43 | 43 |
| SetArrivalpointDelay | | 39 | 39 | 39 | 39 | 39 | 39 |
| GetArrivalpointTasksetRef | | 40 | 40 | 40 | 40 | 40 | 40 |
| GetArrivalpointNext | | 40 | 40 | 40 | 40 | 40 | 40 |
| SetArrivalpointNext | | 39 | 39 | 39 | 39 | 39 | 39 |
| TestArrivalpointWritable | | 50 | 50 | 50 | 50 | 50 | 50 |
| GetExecutionTime | | 19 | 19 | 19 | 19 | 19 | 19 |
| GetLargestExecutionTime | | 24 | 24 | 24 | 24 | 24 | 24 |
| ResetLargestExecutionTime | | 18 | 18 | 18 | 18 | 18 | 18 |
| GetStackOffset | | 25 | 25 | 25 | 25 | 25 | 25 |

## Timing

| Configuration | | Application Uses | | | | | |
|---|---|---|---|---|---|---|---|
| Events | | No | | | Yes | | |
| Shared Task Priorities | | No | | Yes | No | | Yes |
| Multiple Task Activations | | No | Yes | | No | Yes | |
| Service | Variant | | | | | | |
| ActivateTask | SW | 60 | 79 | 90 | 61 | 74 | 97 |
| | NS | 53 | 72 | 84 | 54 | 66 | 89 |
| | KL | 41 | 59 | 70 | 41 | 53 | 74 |
| TerminateTask | LExt | 0 | 0 | 0 | 0 | 0 | 0 |
| | H | 282 | 283 | 282 | 287 | 283 | 287 |
| ChainTask | SWL | 285 | 312 | 347 | 312 | 325 | 374 |
| | SWH | 390 | 415 | 451 | 417 | 430 | 478 |
| | NSL | 285 | 312 | 350 | 312 | 326 | 374 |
| | NSH | 387 | 413 | 448 | 414 | 428 | 476 |
| Schedule | SW | 50 | 50 | 58 | 49 | 48 | 58 |
| GetTaskID | | 32 | 32 | 32 | 50 | 50 | 50 |
| GetTaskState | | 47 | 47 | 47 | 69 | 69 | 69 |
| EnableAllInterrupts | | 33 | 33 | 33 | 33 | 33 | 33 |
| DisableAllInterrupts | | 30 | 30 | 31 | 30 | 30 | 30 |
| ResumeAllInterrupts | | 39 | 39 | 39 | 39 | 39 | 39 |
| SuspendAllInterrupts | | 31 | 31 | 31 | 32 | 31 | 31 |
| ResumeOSInterrupts | | 38 | 38 | 38 | 38 | 38 | 38 |
| SuspendOSInterrupts | | 31 | 31 | 31 | 32 | 31 | 31 |
| GetResource | Task | 33 | 33 | 33 | 33 | 33 | 33 |
| | Combined | 40 | 40 | 40 | 41 | 40 | 40 |
| | CLEx | n/a | n/a | n/a | n/a | n/a | n/a |
| ReleaseResource | Task | 40 | 39 | 39 | 39 | 39 | 40 |
| | Combined | 58 | 58 | 58 | 59 | 59 | 58 |
| | CLEx | n/a | n/a | n/a | n/a | n/a | n/a |
| SetEvent | SW | n/a | n/a | n/a | 53 | 52 | 55 |
| | NS | n/a | n/a | n/a | 48 | 47 | 47 |
| | KL | n/a | n/a | n/a | 38 | 39 | 38 |
| ClearEvent | | n/a | n/a | n/a | 38 | 38 | 38 |
| GetEvent | | n/a | n/a | n/a | 25 | 25 | 25 |
| WaitEvent | <default> | n/a | n/a | n/a | 433 | 434 | 457 |
| | fp | n/a | n/a | n/a | 440 | 441 | 463 |
| GetAlarmBase | | 44 | 43 | 42 | 42 | 46 | 42 |
| GetAlarm | | 48 | 48 | 48 | 48 | 48 | 48 |

| Configuration | | Application Uses | | | | | |
|---|---|---|---|---|---|---|---|
| Events | | No | | | Yes | | |
| Shared Task Priorities | | No | | Yes | No | | Yes |
| Multiple Task Activations | | No | Yes | | No | Yes | |
| SetRelAlarm | | 61 | 60 | 61 | 59 | 61 | 60 |
| SetAbsAlarm | | 60 | 59 | 59 | 61 | 61 | 60 |
| CancelAlarm | | 41 | 42 | 41 | 42 | 41 | 42 |
| InitCounter | | 40 | 40 | 40 | 40 | 40 | 40 |
| GetCounterValue | | 46 | 46 | 46 | 46 | 46 | 46 |
| osek_tick_alarm | <default> | 47 | 48 | 48 | 48 | 47 | 47 |
| | KL | 34 | 34 | 34 | 35 | 35 | 35 |
| osek_incr_counter | | 16 | 16 | 16 | 16 | 16 | 16 |
| GetActiveApplicationMode | | 14 | 14 | 14 | 15 | 15 | 15 |
| StartOS | | 4929 | 4928 | 4928 | 4928 | 4929 | 4951 |
| ShutdownOS | NoHook | n/a | n/a | n/a | n/a | n/a | n/a |
| | Hook | 25 | 25 | 25 | 25 | 25 | 25 |
| InitCOM | | 18 | 18 | 18 | 19 | 19 | 19 |
| CloseCOM | | 18 | 18 | 18 | 18 | 18 | 18 |
| StartCOM | | 37 | 37 | 37 | 103 | 104 | 103 |
| StopCOM | | 22 | 22 | 22 | 22 | 22 | 22 |
| ReadFlag | | n/a | n/a | n/a | 19 | 19 | 19 |
| ResetFlag | | n/a | n/a | n/a | 17 | 17 | 17 |
| ReceiveMessage | | 35 | 35 | 35 | 122 | 134 | 123 |
| GetMessageResource | | n/a | n/a | n/a | 52 | 52 | 52 |
| ReleaseMessageResource | | n/a | n/a | n/a | 63 | 63 | 62 |
| GetMessageStatus | | n/a | n/a | n/a | 35 | 35 | 35 |
| SendMessage | SW | 99 | 118 | 129 | 183 | 196 | 217 |
| | NS | 92 | 110 | 123 | 174 | 186 | 209 |
| | KL | 65 | 84 | 95 | 158 | 158 | 179 |
| ActivateTaskset | SW | 54 | 257 | 301 | 55 | 290 | 277 |
| | NS | 45 | 248 | 292 | 47 | 312 | 298 |
| | KL | 34 | 269 | 281 | 33 | 237 | 286 |
| | SW2 | 54 | 257 | 300 | 55 | 290 | 277 |
| | NS2 | 45 | 248 | 292 | 47 | 312 | 298 |
| | KL2 | 34 | 269 | 281 | 33 | 237 | 286 |
| ChainTaskset | SWL | 281 | 520 | 556 | 308 | 507 | 585 |
| | SWH | 385 | 627 | 662 | 411 | 646 | 660 |
| | NSL | 281 | 521 | 555 | 307 | 539 | 584 |
| | NSH | 382 | 624 | 659 | 407 | 645 | 658 |
| GetTasksetRef | | 43 | 44 | 43 | 43 | 43 | 43 |

| Configuration | | Application Uses | | | | | |
|---|---|---|---|---|---|---|---|
| Events | | No | | | Yes | | |
| Shared Task Priorities | | No | | Yes | No | | Yes |
| Multiple Task Activations | | No | Yes | | No | Yes | |
| MergeTaskset | | 57 | 58 | 58 | 57 | 57 | 57 |
| AssignTaskset | | 40 | 40 | 40 | 40 | 40 | 40 |
| RemoveTaskset | | 55 | 55 | 56 | 55 | 56 | 55 |
| TestSubTaskset | | 57 | 57 | 57 | 57 | 57 | 57 |
| TestEquivalentTaskset | | 54 | 54 | 54 | 54 | 54 | 55 |
| TickSchedule | SW | 72 | 324 | 336 | 90 | 298 | 347 |
| | NS | 65 | 315 | 328 | 81 | 288 | 339 |
| | KL | 51 | 301 | 314 | 67 | 275 | 325 |
| | SW2 | 72 | 324 | 336 | 90 | 294 | 342 |
| | NS2 | 65 | 315 | 328 | 81 | 284 | 334 |
| | KL2 | 51 | 301 | 313 | 67 | 271 | 320 |
| AdvanceSchedule | SW | 71 | 323 | 333 | 89 | 297 | 347 |
| | NS | 61 | 312 | 325 | 79 | 286 | 337 |
| | KL | 50 | 302 | 313 | 67 | 275 | 325 |
| | SW2 | 71 | 322 | 333 | 88 | 293 | 342 |
| | NS2 | 61 | 312 | 325 | 79 | 282 | 332 |
| | KL2 | 50 | 303 | 313 | 67 | 271 | 320 |
| StartSchedule | | 68 | 68 | 68 | 69 | 68 | 68 |
| StopSchedule | | 64 | 64 | 64 | 63 | 64 | 63 |
| GetScheduleStatus | | 70 | 69 | 69 | 69 | 70 | 69 |
| GetScheduleValue | | 65 | 65 | 64 | 64 | 65 | 64 |
| GetScheduleNext | | 44 | 44 | 44 | 44 | 44 | 44 |
| SetScheduleNext | | 41 | 41 | 41 | 41 | 41 | 41 |
| GetArrivalpointDelay | | 42 | 42 | 42 | 43 | 43 | 43 |
| SetArrivalpointDelay | | 39 | 39 | 39 | 39 | 39 | 39 |
| GetArrivalpointTasksetRef | | 40 | 40 | 40 | 40 | 41 | 40 |
| GetArrivalpointNext | | 40 | 40 | 40 | 40 | 40 | 40 |
| SetArrivalpointNext | | 39 | 39 | 39 | 39 | 39 | 39 |
| TestArrivalpointWritable | | 50 | 50 | 50 | 50 | 50 | 50 |
| GetExecutionTime | | 65 | 66 | 65 | 65 | 65 | 66 |
| GetLargestExecutionTime | | 26 | 26 | 26 | 26 | 26 | 26 |
| ResetLargestExecutionTime | | 24 | 24 | 24 | 24 | 24 | 24 |
| GetStackOffset | | 22 | 22 | 22 | 22 | 22 | 22 |

## Extended

| Configuration | | Application Uses | | | | | |
|---|---|---|---|---|---|---|---|
| Events | | No | | | Yes | | |
| Shared Task Priorities | | No | | Yes | No | | Yes |
| Multiple Task Activations | | No | Yes | | No | Yes | |
| Service | Variant | | | | | | |
| ActivateTask | SW | 186 | 206 | 220 | 191 | 201 | 220 |
| | NS | 206 | 224 | 238 | 208 | 220 | 238 |
| | KL | 159 | 180 | 193 | 163 | 174 | 191 |
| TerminateTask | LExt | 238 | 239 | 238 | 239 | 236 | 238 |
| | H | 333 | 333 | 335 | 333 | 331 | 332 |
| ChainTask | SWL | 451 | 480 | 517 | 481 | 491 | 541 |
| | SWH | 552 | 582 | 618 | 582 | 595 | 640 |
| | NSL | 474 | 504 | 538 | 503 | 515 | 567 |
| | NSH | 573 | 602 | 642 | 604 | 615 | 661 |
| Schedule | SW | 79 | 77 | 89 | 79 | 78 | 87 |
| GetTaskID | | 34 | 34 | 34 | 53 | 53 | 53 |
| GetTaskState | | 192 | 192 | 192 | 211 | 210 | 212 |
| EnableAllInterrupts | | 36 | 36 | 36 | 36 | 36 | 36 |
| DisableAllInterrupts | | 34 | 34 | 34 | 34 | 34 | 34 |
| ResumeAllInterrupts | | 47 | 47 | 47 | 49 | 48 | 47 |
| SuspendAllInterrupts | | 34 | 34 | 35 | 34 | 34 | 34 |
| ResumeOSInterrupts | | 46 | 46 | 46 | 48 | 47 | 46 |
| SuspendOSInterrupts | | 34 | 34 | 36 | 34 | 34 | 34 |
| GetResource | Task | 278 | 277 | 150 | 301 | 299 | 176 |
| | Combined | 136 | 137 | 136 | 160 | 160 | 161 |
| | CLEx | 188 | 190 | 188 | 214 | 215 | 215 |
| ReleaseResource | Task | 140 | 141 | 140 | 165 | 165 | 163 |
| | Combined | 144 | 146 | 145 | 169 | 168 | 168 |
| | CLEx | 157 | 156 | 157 | 181 | 180 | 180 |
| SetEvent | SW | n/a | n/a | n/a | 198 | 197 | 197 |
| | NS | n/a | n/a | n/a | 205 | 206 | 205 |
| | KL | n/a | n/a | n/a | 175 | 175 | 175 |
| ClearEvent | | n/a | n/a | n/a | 51 | 51 | 52 |
| GetEvent | | n/a | n/a | n/a | 160 | 160 | 160 |
| WaitEvent | <default> | n/a | n/a | n/a | 520 | 518 | 541 |
| | fp | n/a | n/a | n/a | 527 | 526 | 547 |
| GetAlarmBase | | 126 | 127 | 132 | 126 | 127 | 135 |
| GetAlarm | | 127 | 128 | 135 | 128 | 127 | 136 |

| Configuration | | Application Uses | | | | | |
|---|---|---|---|---|---|---|---|
| Events | | No | | | Yes | | |
| Shared Task Priorities | | No | | Yes | No | | Yes |
| Multiple Task Activations | | No | Yes | | No | Yes | |
| SetRelAlarm | | 147 | 146 | 155 | 147 | 147 | 155 |
| SetAbsAlarm | | 141 | 141 | 149 | 142 | 142 | 150 |
| CancelAlarm | | 119 | 119 | 127 | 119 | 119 | 127 |
| InitCounter | | 186 | 186 | 196 | 187 | 187 | 196 |
| GetCounterValue | | 121 | 122 | 123 | 121 | 122 | 120 |
| osek_tick_alarm | <default> | 61 | 61 | 61 | 61 | 61 | 61 |
| | KL | 33 | 33 | 33 | 33 | 33 | 33 |
| osek_incr_counter | | 15 | 15 | 15 | 15 | 15 | 15 |
| GetActiveApplicationMode | | 14 | 14 | 14 | 14 | 14 | 14 |
| StartOS | | 5101 | 5101 | 5099 | 5100 | 5099 | 5101 |
| ShutdownOS | NoHook | n/a | n/a | n/a | n/a | n/a | n/a |
| | Hook | 26 | 26 | 27 | 26 | 26 | 27 |
| InitCOM | | 18 | 18 | 18 | 19 | 19 | 19 |
| CloseCOM | | 18 | 18 | 18 | 18 | 18 | 18 |
| StartCOM | | 44 | 44 | 44 | 113 | 110 | 111 |
| StopCOM | | 31 | 30 | 30 | 30 | 30 | 31 |
| ReadFlag | | n/a | n/a | n/a | 32 | 36 | 32 |
| ResetFlag | | n/a | n/a | n/a | 30 | 30 | 30 |
| ReceiveMessage | | 97 | 97 | 97 | 181 | 180 | 183 |
| GetMessageResource | | n/a | n/a | n/a | 238 | 238 | 239 |
| ReleaseMessageResource | | n/a | n/a | n/a | 226 | 225 | 225 |
| GetMessageStatus | | n/a | n/a | n/a | 83 | 82 | 82 |
| SendMessage | SW | 290 | 309 | 323 | 374 | 384 | 415 |
| | NS | 312 | 328 | 344 | 392 | 403 | 424 |
| | KL | 239 | 260 | 273 | 323 | 334 | 352 |
| ActivateTaskset | SW | 387 | 476 | 551 | 368 | 476 | 557 |
| | NS | 402 | 489 | 562 | 381 | 487 | 569 |
| | KL | 358 | 479 | 522 | 337 | 446 | 534 |
| | SW2 | 387 | 476 | 550 | 368 | 475 | 557 |
| | NS2 | 402 | 489 | 562 | 380 | 487 | 569 |
| | KL2 | 358 | 479 | 522 | 337 | 446 | 535 |
| ChainTaskset | SWL | 662 | 753 | 816 | 658 | 764 | 876 |
| | SWH | 762 | 854 | 950 | 788 | 899 | 947 |
| | NSL | 672 | 770 | 866 | 674 | 814 | 893 |
| | NSH | 773 | 903 | 1001 | 777 | 887 | 996 |
| GetTasksetRef | | 170 | 170 | 170 | 170 | 170 | 170 |

| Configuration | | Application Uses | | | | | |
|---|---|---|---|---|---|---|---|
| Events | | No | | | Yes | | |
| Shared Task Priorities | | No | | Yes | No | | Yes |
| Multiple Task Activations | | No | Yes | | No | Yes | |
| MergeTaskset | | 119 | 121 | 120 | 120 | 122 | 122 |
| AssignTaskset | | 87 | 87 | 88 | 90 | 89 | 89 |
| RemoveTaskset | | 118 | 119 | 119 | 118 | 120 | 116 |
| TestSubTaskset | | 120 | 121 | 120 | 120 | 119 | 120 |
| TestEquivalentTaskset | | 118 | 118 | 118 | 118 | 117 | 116 |
| TickSchedule | SW | 135 | 588 | 630 | 442 | 564 | 650 |
| | NS | 153 | 603 | 646 | 459 | 578 | 665 |
| | KL | 84 | 556 | 599 | 412 | 535 | 619 |
| | SW2 | 135 | 588 | 630 | 442 | 553 | 642 |
| | NS2 | 153 | 603 | 646 | 459 | 567 | 658 |
| | KL2 | 84 | 556 | 599 | 412 | 522 | 611 |
| AdvanceSchedule | SW | 139 | 626 | 667 | 480 | 601 | 686 |
| | NS | 154 | 640 | 682 | 497 | 616 | 703 |
| | KL | 107 | 597 | 639 | 453 | 573 | 661 |
| | SW2 | 139 | 625 | 668 | 480 | 590 | 678 |
| | NS2 | 154 | 640 | 682 | 497 | 605 | 695 |
| | KL2 | 106 | 597 | 639 | 453 | 563 | 651 |
| StartSchedule | | 110 | 109 | 110 | 109 | 110 | 110 |
| StopSchedule | | 97 | 98 | 97 | 97 | 97 | 98 |
| GetScheduleStatus | | 103 | 103 | 103 | 103 | 103 | 103 |
| GetScheduleValue | | 97 | 96 | 95 | 95 | 95 | 96 |
| GetScheduleNext | | 62 | 61 | 61 | 61 | 62 | 61 |
| SetScheduleNext | | 89 | 89 | 89 | 91 | 89 | 90 |
| GetArrivalpointDelay | | 72 | 71 | 72 | 72 | 72 | 71 |
| SetArrivalpointDelay | | 74 | 73 | 74 | 74 | 73 | 73 |
| GetArrivalpointTasksetRef | | 61 | 62 | 61 | 61 | 62 | 62 |
| GetArrivalpointNext | | 61 | 60 | 60 | 61 | 60 | 60 |
| SetArrivalpointNext | | 96 | 95 | 96 | 96 | 96 | 96 |
| TestArrivalpointWritable | | 70 | 69 | 69 | 69 | 70 | 70 |
| GetExecutionTime | | 84 | 85 | 85 | 83 | 83 | 83 |
| GetLargestExecutionTime | | 145 | 146 | 146 | 146 | 145 | 146 |
| ResetLargestExecutionTime | | 140 | 139 | 140 | 140 | 139 | 140 |
| GetStackOffset | | 22 | 22 | 22 | 22 | 22 | 22 |

### 4.3.2 OS Start-up Time

OS start-up time is the time from the entry to the `StartOS()` function to the execution of the first instruction in a user task (including the idle task) without any hook routines being called. This time is always application dependent, since `StartOS()` may activate any number of tasks and start any number of user-specified alarms.

### 4.3.3 Interrupt Latencies

Interrupt latency is the time between an interrupt request being recognized by the target hardware and the execution of the first instruction of the user provided handler function. The following tables give the interrupt latencies (in CPU cycles).

## Standard

| Configuration | | Application Uses | | | | | |
|---|---|---|---|---|---|---|---|
| Events | | No | | | Yes | | |
| Shared Task Priorities | | No | | Yes | No | | Yes |
| Multiple Task Activations | | No | Yes | | No | Yes | |
| Operation | ISR Category | | | | | | |
| ISR Latency | Cat 1 | 31 | 31 | 31 | 31 | 31 | 31 |
| | Cat 2 | 29 | 46 | 46 | 46 | 46 | 46 |

## Timing

| Configuration | | Application Uses | | | | | |
|---|---|---|---|---|---|---|---|
| Events | | No | | | Yes | | |
| Shared Task Priorities | | No | | Yes | No | | Yes |
| Multiple Task Activations | | No | Yes | | No | Yes | |
| Operation | ISR Category | | | | | | |
| ISR Latency | Cat 1 | 31 | 31 | 31 | 31 | 31 | 31 |
| | Cat 2 | 155 | 166 | 166 | 166 | 166 | 168 |

## Extended

| Configuration | | Application Uses | | | | | |
|---|---|---|---|---|---|---|---|
| Events | | No | | | Yes | | |
| Shared Task Priorities | | No | | Yes | No | | Yes |
| Multiple Task Activations | | No | Yes | | No | Yes | |
| Operation | ISR Category | | | | | | |
| ISR Latency | Cat 1 | 31 | 31 | 31 | 31 | 31 | 31 |
| | Cat 2 | 154 | 167 | 166 | 166 | 166 | 166 |

### 4.3.4 Task Switching Times

Task switching time is the time between the last instruction of the previous task and the first instruction of the next task. The switching time differs, depending on the switching contexts (e.g. an `ActivateTask()` versus a `ChainTask()`).

RTA-OSEK sub-task types also affect the switching time. The tables in this section show the switching times (in CPU cycles) for all system classes for basic, lightweight tasks and for basic and extended heavyweight tasks.

Figures 1 to 8 show the RTA-OSEK switching contexts measured.

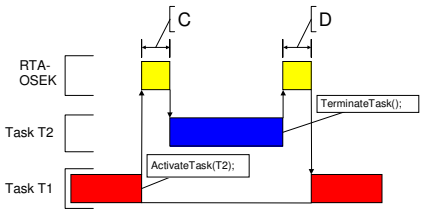


**Figure 1: Task Activates a Higher Priority Task which Terminates Normally**

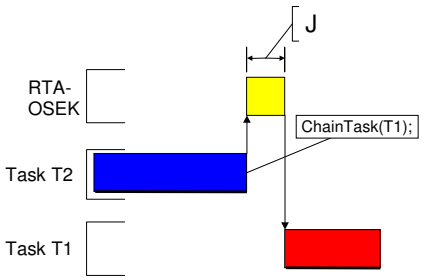


**Figure 2: Task Chaining**

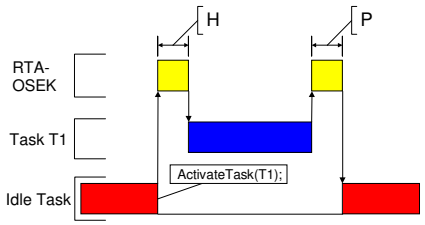


**Figure 3: Task Activation from Idle Task**

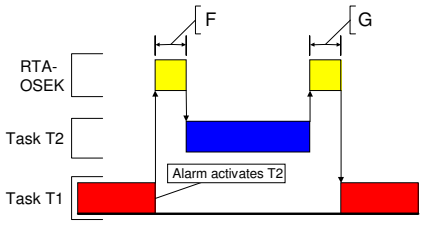


**Figure 4: Task Activation from an Alarm**

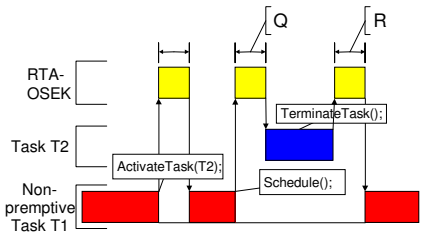


**Figure 5: Non-Premptive Task Calls Schedule()**

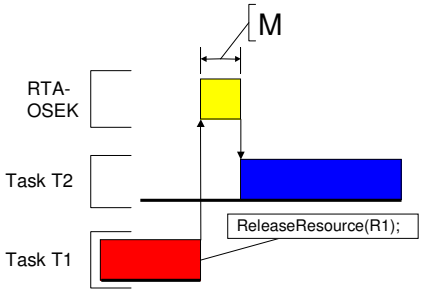


**Figure 6: Blocked Task Activated by ReleaseResource()**

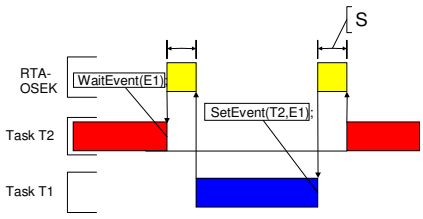


**Figure 7: Waiting Task Activated by SetEvent()**

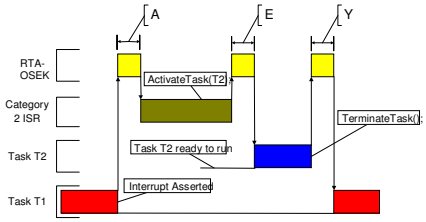


**Figure 8: Category 2 ISR Activates a Higher Priority Task**

## Standard

| Configuration | | Application Uses | | | | | |
|---|---|---|---|---|---|---|---|
| Events | | No | | | Yes | | |
| Shared Task Priorities | | No | | Yes | No | | Yes |
| Multiple Task Activations | Task Attributes | No | Yes | | No | Yes | |
| Normal termination | Light, Basic | 68 | 89 | 101 | 68 | 89 | 100 |
| Figure 1: D | Heavy, Basic/Extended | 158 | 176 | 186 | 183 | 182 | 194 |
| ChainTask | Light, Basic | 96 | 124 | 158 | 98 | 125 | 170 |
| Figure 2: J | Heavy, Basic/Extended | 361 | 407 | 452 | 389 | 413 | 472 |
| Pre-emption | Light, Basic | 92 | 119 | 154 | 93 | 119 | 166 |
| Figure 1: C | Heavy, Basic/Extended | 177 | 201 | 235 | 202 | 214 | 261 |
| From idle task | Light, Basic | 141 | 168 | 204 | 142 | 168 | 215 |
| Figure 3: H | Heavy, Basic/Extended | 226 | 251 | 284 | 251 | 263 | 310 |
| Triggered by alarm | Light, Basic | 137 | 165 | 200 | 138 | 163 | 211 |
| Figure 4: F | Heavy, Basic/Extended | 225 | 250 | 283 | 249 | 263 | 310 |
| Schedule | Light, Basic | 76 | 84 | 116 | 76 | 83 | 115 |
| Figure 5: Q | Heavy, Basic/Extended | 161 | 166 | 197 | 185 | 185 | 216 |
| Release resource | Light, Basic | 81 | 89 | 112 | 82 | 91 | 114 |
| Figure 6: M | Heavy, Basic/Extended | 168 | 171 | 193 | 191 | 193 | 215 |
| SetEvent | | | | | | | |

| Configuration | | Application Uses | | | | | |
|---|---|---|---|---|---|---|---|
| Events | | No | | | Yes | | |
| Shared Task Priorities | | No | | Yes | No | | Yes |
| Multiple Task Activations | Task Attributes | No | Yes | | No | Yes | |
| Figure 7: S | Heavy, Extended | n/a | n/a | n/a | 264 | 263 | 318 |
| From category 2 ISR | Light, Basic | 78 | 100 | 123 | 92 | 100 | 123 |
| Figure 8: E | Heavy, Basic/Extended | 164 | 183 | 205 | 202 | 202 | 225 |

## Timing

| Configuration | | Application Uses | | | | | |
|---|---|---|---|---|---|---|---|
| Events | | No | | | Yes | | |
| Shared Task Priorities | | No | | Yes | No | | Yes |
| Multiple Task Activations | Task Attributes | No | Yes | | No | Yes | |
| Normal termination | Light, Basic | 182 | 197 | 208 | 182 | 196 | 209 |
| Figure 1: D | Heavy, Basic/Extended | 282 | 299 | 307 | 306 | 304 | 315 |
| ChainTask | Light, Basic | 208 | 235 | 273 | 208 | 233 | 281 |
| Figure 2: J | Heavy, Basic/Extended | 604 | 646 | 693 | 625 | 652 | 709 |
| Pre-emption | Light, Basic | 186 | 210 | 249 | 186 | 211 | 261 |
| Figure 1: C | Heavy, Basic/Extended | 269 | 293 | 333 | 295 | 309 | 361 |
| From idle task | Light, Basic | 235 | 259 | 298 | 235 | 260 | 310 |
| Figure 3: H | Heavy, Basic/Extended | 318 | 342 | 378 | 344 | 358 | 408 |
| Triggered by alarm | Light, Basic | 231 | 256 | 297 | 231 | 256 | 306 |
| Figure 4: F | Heavy, Basic/Extended | 316 | 341 | 377 | 342 | 356 | 405 |
| Schedule | Light, Basic | 171 | 176 | 212 | 168 | 175 | 210 |
| Figure 5: Q | Heavy, Basic/Extended | 254 | 259 | 292 | 277 | 278 | 314 |
| Release resource | Light, Basic | 173 | 178 | 207 | 174 | 181 | 206 |
| Figure 6: M | Heavy, Basic/Extended | 256 | 261 | 286 | 283 | 284 | 311 |
| SetEvent | | | | | | | |
| Figure 7: S | Heavy, Extended | n/a | n/a | n/a | 339 | 337 | 396 |
| From category 2 ISR | Light, Basic | 283 | 302 | 331 | 295 | 304 | 331 |
| Figure 8: E | Heavy, Basic/Extended | 364 | 383 | 409 | 402 | 405 | 433 |

### Extended

| Configuration | | Application Uses | | | | | |
|---|---|---|---|---|---|---|---|
| Events | | No | | | Yes | | |
| Shared Task Priorities | | No | | Yes | No | | Yes |
| Multiple Task Activations | Task Attributes | No | Yes | | No | Yes | |
| Normal termination | Light, Basic | 237 | 252 | 263 | 238 | 249 | 264 |
| Figure 1: D | Heavy, Basic/Extended | 333 | 346 | 360 | 352 | 351 | 363 |
| ChainTask | Light, Basic | 374 | 403 | 441 | 377 | 400 | 449 |
| Figure 2: J | Heavy, Basic/Extended | 818 | 861 | 911 | 841 | 861 | 920 |
| Pre-emption | Light, Basic | 305 | 332 | 371 | 308 | 331 | 377 |
| Figure 1: C | Heavy, Basic/Extended | 388 | 415 | 453 | 417 | 429 | 474 |
| From idle task | Light, Basic | 354 | 381 | 420 | 357 | 380 | 426 |
| Figure 3: H | Heavy, Basic/Extended | 437 | 464 | 502 | 467 | 479 | 523 |
| Triggered by alarm | Light, Basic | 367 | 394 | 434 | 369 | 392 | 438 |
| Figure 4: F | Heavy, Basic/Extended | 452 | 480 | 517 | 480 | 493 | 537 |
| Schedule | Light, Basic | 195 | 201 | 236 | 194 | 199 | 235 |
| Figure 5: Q | Heavy, Basic/Extended | 278 | 284 | 318 | 303 | 303 | 338 |
| Release resource | Light, Basic | 258 | 266 | 290 | 283 | 288 | 315 |
| Figure 6: M | Heavy, Basic/Extended | 341 | 349 | 372 | 392 | 392 | 418 |
| SetEvent | | | | | | | |
| Figure 7: S | Heavy, Extended | n/a | n/a | n/a | 518 | 516 | 570 |
| From category 2 ISR | Light, Basic | 302 | 324 | 350 | 314 | 320 | 349 |
| Figure 8: E | Heavy, Basic/Extended | 383 | 405 | 430 | 421 | 422 | 450 |

## 4.4 Configuration of Run-time Context

The run-time contexts of all tasks reside on the same stack and are recovered when the task terminates. As a result, run-time contexts of mutually exclusive tasks are effectively overlaid. The RTA-OSEK GUI is able to calculate the worst-case stack requirement for the entire application, based on the declared stack usage, the priorities and the resource occupation of individual tasks.

The size of the run-time context of a task depends on the task type and the system configuration. The following tables give the sizes (in bytes) for different OS status and configurations:

## Standard

| Configuration | Application Uses | | | | | |
|---|---|---|---|---|---|---|
| Events | No | | | Yes | | |
| Shared Task Priorities | No | | Yes | No | | Yes |
| Multiple Task Activations | No | Yes | | No | Yes | |
| **Pre- and Post-Task hooks not used** | | | | | | |
| Task type | | | | | | |
| BCC1 lightweight, integer | 12 | 12 | 13 | 12 | 12 | 13 |
| BCC1 lightweight, floating-point | 13 | 13 | 14 | 13 | 13 | 14 |
| BCC1 heavyweight, integer | 28 | 28 | 29 | 28 | 28 | 29 |
| BCC1 heavyweight, floating-point | 28 | 28 | 29 | 28 | 28 | 29 |
| BCC2 lightweight, integer | n/a | 13 | 14 | n/a | 13 | 14 |
| BCC2 lightweight, floating-point | n/a | 13 | 14 | n/a | 13 | 14 |
| BCC2 heavyweight, integer | n/a | 28 | 29 | n/a | 28 | 29 |
| BCC2 heavyweight, floating-point | n/a | 28 | 29 | n/a | 28 | 29 |
| ECC1 heavyweight, integer | n/a | n/a | n/a | 30 | 30 | 31 |
| ECC1 heavyweight, floating-point | n/a | n/a | n/a | 30 | 30 | 31 |
| ECC2 heavyweight, integer | n/a | n/a | n/a | n/a | n/a | 31 |
| ECC2 heavyweight, floating-point | n/a | n/a | n/a | n/a | n/a | 31 |
| | | | | | | |
| **Pre- and/or Post-Task hooks used** | | | | | | |
| Task type | | | | | | |
| BCC1 lightweight, integer | 13 | 13 | 13 | 13 | 13 | 13 |
| BCC1 lightweight, floating-point | 14 | 14 | 14 | 14 | 14 | 14 |
| BCC1 heavyweight, integer | 29 | 29 | 29 | 29 | 29 | 29 |
| BCC1 heavyweight, floating-point | 29 | 29 | 29 | 29 | 29 | 29 |
| BCC2 lightweight, integer | n/a | 14 | 14 | n/a | 14 | 14 |
| BCC2 lightweight, floating-point | n/a | 14 | 14 | n/a | 14 | 14 |
| BCC2 heavyweight, integer | n/a | 29 | 29 | n/a | 29 | 29 |
| BCC2 heavyweight, floating-point | n/a | 29 | 29 | n/a | 29 | 29 |
| ECC1 heavyweight, integer | n/a | n/a | n/a | 31 | 31 | 31 |
| ECC1 heavyweight, floating-point | n/a | n/a | n/a | 31 | 31 | 31 |
| ECC2 heavyweight, integer | n/a | n/a | n/a | n/a | n/a | 31 |
| ECC2 heavyweight, floating-point | n/a | n/a | n/a | n/a | n/a | 31 |

## Timing

| Configuration | Application Uses | | | | | |
|---|---|---|---|---|---|---|
| Events | No | | | Yes | | |
| Shared Task Priorities | No | | Yes | No | | Yes |
| Multiple Task Activations | No | Yes | | No | Yes | |
| **Pre- and Post-Task hooks not used** | | | | | | |
| Task type | | | | | | |
| BCC1 lightweight, integer | 22 | 22 | 23 | 22 | 22 | 23 |
| BCC1 lightweight, floating-point | 23 | 23 | 24 | 23 | 23 | 24 |
| BCC1 heavyweight, integer | 40 | 40 | 41 | 40 | 40 | 41 |
| BCC1 heavyweight, floating-point | 40 | 40 | 41 | 40 | 40 | 41 |
| BCC2 lightweight, integer | n/a | 23 | 24 | n/a | 23 | 24 |
| BCC2 lightweight, floating-point | n/a | 23 | 24 | n/a | 23 | 24 |
| BCC2 heavyweight, integer | n/a | 40 | 41 | n/a | 40 | 41 |
| BCC2 heavyweight, floating-point | n/a | 40 | 41 | n/a | 40 | 41 |
| ECC1 heavyweight, integer | n/a | n/a | n/a | 41 | 41 | 42 |
| ECC1 heavyweight, floating-point | n/a | n/a | n/a | 41 | 41 | 42 |
| ECC2 heavyweight, integer | n/a | n/a | n/a | n/a | n/a | 42 |
| ECC2 heavyweight, floating-point | n/a | n/a | n/a | n/a | n/a | 42 |
| | | | | | | |
| **Pre- and/or Post-Task hooks used** | | | | | | |
| Task type | | | | | | |
| BCC1 lightweight, integer | 31 | 31 | 31 | 31 | 31 | 31 |
| BCC1 lightweight, floating-point | 32 | 32 | 32 | 32 | 32 | 32 |
| BCC1 heavyweight, integer | 49 | 49 | 49 | 49 | 49 | 49 |
| BCC1 heavyweight, floating-point | 49 | 49 | 49 | 49 | 49 | 49 |
| BCC2 lightweight, integer | n/a | 32 | 32 | n/a | 32 | 32 |
| BCC2 lightweight, floating-point | n/a | 32 | 32 | n/a | 32 | 32 |
| BCC2 heavyweight, integer | n/a | 49 | 49 | n/a | 49 | 49 |
| BCC2 heavyweight, floating-point | n/a | 49 | 49 | n/a | 49 | 49 |
| ECC1 heavyweight, integer | n/a | n/a | n/a | 50 | 50 | 50 |
| ECC1 heavyweight, floating-point | n/a | n/a | n/a | 50 | 50 | 50 |
| ECC2 heavyweight, integer | n/a | n/a | n/a | n/a | n/a | 50 |
| ECC2 heavyweight, floating-point | n/a | n/a | n/a | n/a | n/a | 50 |

## Extended

| Configuration | Application Uses | | | | | |
|---|---|---|---|---|---|---|
| Events | No | | | Yes | | |
| Shared Task Priorities | No | | Yes | No | | Yes |
| Multiple Task Activations | No | Yes | | No | Yes | |
| **Pre- and Post-Task hooks not used** | | | | | | |
| Task type | | | | | | |
| BCC1 lightweight, integer | 22 | 22 | 23 | 22 | 22 | 23 |
| BCC1 lightweight, floating-point | 23 | 23 | 24 | 23 | 23 | 24 |
| BCC1 heavyweight, integer | 40 | 40 | 41 | 40 | 40 | 41 |
| BCC1 heavyweight, floating-point | 40 | 40 | 41 | 40 | 40 | 41 |
| BCC2 lightweight, integer | n/a | 23 | 24 | n/a | 23 | 24 |
| BCC2 lightweight, floating-point | n/a | 23 | 24 | n/a | 23 | 24 |
| BCC2 heavyweight, integer | n/a | 40 | 41 | n/a | 40 | 41 |
| BCC2 heavyweight, floating-point | n/a | 40 | 41 | n/a | 40 | 41 |
| ECC1 heavyweight, integer | n/a | n/a | n/a | 49 | 49 | 50 |
| ECC1 heavyweight, floating-point | n/a | n/a | n/a | 49 | 49 | 50 |
| ECC2 heavyweight, integer | n/a | n/a | n/a | n/a | n/a | 50 |
| ECC2 heavyweight, floating-point | n/a | n/a | n/a | n/a | n/a | 50 |
| | | | | | | |
| **Pre- and/or Post-Task hooks used** | | | | | | |
| Task type | | | | | | |
| BCC1 lightweight, integer | 31 | 31 | 31 | 31 | 31 | 31 |
| BCC1 lightweight, floating-point | 32 | 32 | 32 | 32 | 32 | 32 |
| BCC1 heavyweight, integer | 49 | 49 | 49 | 49 | 49 | 49 |
| BCC1 heavyweight, floating-point | 49 | 49 | 49 | 49 | 49 | 49 |
| BCC2 lightweight, integer | n/a | 32 | 32 | n/a | 32 | 32 |
| BCC2 lightweight, floating-point | n/a | 32 | 32 | n/a | 32 | 32 |
| BCC2 heavyweight, integer | n/a | 49 | 49 | n/a | 49 | 49 |
| BCC2 heavyweight, floating-point | n/a | 49 | 49 | n/a | 49 | 49 |
| ECC1 heavyweight, integer | n/a | n/a | n/a | 58 | 58 | 58 |
| ECC1 heavyweight, floating-point | n/a | n/a | n/a | 58 | 58 | 58 |
| ECC2 heavyweight, integer | n/a | n/a | n/a | n/a | n/a | 58 |
| ECC2 heavyweight, floating-point | n/a | n/a | n/a | n/a | n/a | 58 |

# 5 Inline Interrupt Control API Calls

The RTA-OSEK Component for the TriCore17x6/Tasking supports two variations of the OSEK interrupt handling API calls. In addition to the API calls contained within the RTA-OSEK run-time libraries, inline versions are also supported. Using these inline versions will result in faster code and a reduced Context Save Area usage. The inline versions of these API calls are all have the "os" prefix.

The inline API calls are restricted to the standard build applications that do not use RTA-TRACE. Inline calls contained within application code for other configurations will be automatically substituted with calls to the library API during compilation.

To take advantage of the inline versions in application code the substitutions in the following table should be used:

| Library API call | Inline API call |
|---|---|
| `DisableAllInterrupts()` | `osDisableAllInterrupts()` |
| `EnableAllInterrupts()` | `osEnableAllInterrupts()` |
| `SuspendOSInterrupts()` | `osSuspendOSInterrupts()` |
| `ResumeOSInterrupts()` | `osResumeOSInterrupts()` |
| `SuspendAllInterrupts()` | `osSuspendAllInterrupts()` |
| `ResumeAllInterrupts()` | `osResumeAllInterrupts()` |

# 6 Compatibility with v5.0.0

## 6.1 Support for v1.3.1 CPUs

Version 5.0.1 is the first version that can support the new v1.3.1 CPU cores (including the TC1767 and TC1797). The behavior of the PCXI.PCPN register has been modified between the CPU variants. Earlier versions of RTA-OSEK can only support the PCXI.PCPN behavior of v1.3 cores. RTA-OSEK has been modified to support both of these CPU core types.

## 6.2 Number of tasks

To improve performance v5.0.1 now only supports 32 tasks (in addition to the idle task).

## 6.3 Silicon bug CPU_TC.048

Following discussions with Infineon the interrupt entry latency has been reduced by removing unnecessary `nop` instructions. These instructions had been inserted to counter errata CPU_TC.048. As this issue only affects loads into an address register from memory and not from immediate values these can be safely removed. However, the Tasking-VX toolset will incorrectly generate a warning when assembling the file `osgen.asm` regarding this issue but these can be safely ignored.

## 6.4 Direct function calling

To improve performance (i.e. reduce the instruction count) v5.0.1 now supports direct addressing of code objects. Earlier versions applied a strict policy of only using indirect (i.e. 32-bit) addressing so that the entire memory space could be safely used. Direct addressing requires that all code objects must be located either within a relative signed 24-bit displacement address range of the current address or an absolute disp24 address range (See the Infineon v1.3 Instruction set manual for further details). These address ranges cover a ±16 Mbytes region from the current address or the first 2 Mbytes of every 256 Mbytes. As a result direct addressing cannot be used between code objects located in the internal FLASH memory and the external memory regions or in the CSRAM of v1.3 CPUs. To make use of direct addressing application code should be compiled with the macro `OS_DIRECT_CALLS` defined.

## 6.5 A0 variable addressing

In v5.0.0 there is an issue where the variable `osek_orti_call` is incorrectly referenced using A0 addressing in the extended build. This has now been resolved and it can now be safely located in the general data region.

## 6.6 Section Names

The section names used by RTA-OSEK have been modified in v5.0.1 so that variables that must be initialized by the c start-up code can now be easily differentiated from variables that are initialized by the `StartOS()` API call. The sections os_pnir2, os_pir2 and ps_pur2 are now additionally used containing data that is initialized by the C start-up code.

# 7 Compatibility with v5.0.1

## 7.1 Workaround for silicon problem

Version 5.0.2 has a workaround for a race condition concerning the MTCR instruction and the contents of the PCXI condition.

Version 5.0.2 libraries cannot be used with applications configured for 5.0.1; and version 5.0.1 libraries cannot be used with applications configured for 5.0.2. This avoids a problem that it was possible to link 5.0.1 libraries with applications configured for 5.0.0, giving an invalid configuration.

# 8 Compatibility with Pre-v5 Kernels

## 8.1 Updating the application version

To convert an existing v3.x OIL configuration file to v5.0.0, load the file into the v5 RTA-OSEK GUI, select the 'OS Configuration' option in the 'Application' menu and change the 'Kernel Version' to v5.00. When the OIL configuration file is saved it will then use the v5.0.0 format and the v5.00 kernel libraries. This process can be reversed to move back to earlier kernel versions.

# Support

For product support, please contact your local ETAS representative.

Office locations and contact details can be found at the front of this manual and on the ETAS Group website www.etasgroup.com.