11. CLOCK

Each **ORKID** kernel maintains a node clock. This is a single data object in the kernel data structure which contains the current date and time. The clock is updated at every tick, the frequency of which is node dependent. The range of dates the clock is allowed to take is implementation dependent.

In a multi-node system, the different node clocks will very likely be synchronized, although this is not necessarily done automatically by the kernel. Since nodes could be in different time zones in widely distributed systems, the node clock specifies the local time zone, so that all nodes can synchronize their clocks to the same absolute time.

The data structure containing the clock value passed in clock operations is language binding dependent. It identifies the date and time down to the nearest tick, along with the local time zone. The time zone value is defined as the number of hours ahead (positive value) or behind (negative value) Greenwich Mean Time (GMT).

When the system starts up, the clock may be uninitialised. If this is the case, attempts at reading it before it has been set result in an error completion status, rather than returning a random value.

11.1. CLOCK_SET

Set node time and date.

Synopsis

```
   clock_set( clock )
```

Input Parameters

clock	: clock_buff	current time and date

Output Parameters

<none>

Completion Status

OK	clock_set successful
ILLEGAL_USE	clock_set not callable from ISR
INVALID_PARAMETER	a parameter refers to an invalid address
INVALID_CLOCK	invalid clock value

Description

This operation sets the node clock to the specified value. The kernel checks the supplied date and time in clock_buff to ensure that they are legal. This is purely a syntactic check,_the operation will accept any legal value. The exact structure of the data supplied is language binding dependent.

11.2. CLOCK_GET

Get node time and date.

Synopsis

```
   clock_get( clock )
```

Input Parameters

   <none>

Output Parameters

clock	: clock_buff	current time and date

Completion Status

OK	clock_get successful
INVALID_PARAMETER	a parameter refers to an invalid address
CLOCK_NOT_SET	clock has not been initialized

Description

This operation returns the current date and time in the node clock. If the node clock has not yet been set, then the CLOCK_NOT_SET completion status is returned and the contents of clock are undetermined. The exact structure of the clock_buff data returned is language binding dependent.

11.3. CLOCK_TICK

Announce a tick to the clock.

Synopsis

```
clock_tick(  )
```

Input Parameters

<none>

Output Parameters

<none>

Completion Status

OK	clock_tick successful

Description

This operation increments the current node time by one tick. There are no parameters and the operation always succeeds. Nevertheless, the operation can be meaningless if the clock was not initialized beforehand. Every node must contain a mechanism which keeps the node clock up to date by calling upon clock_tick.

12. TIMERS

ORKID defines two types of timers. The first type is the sleep timer. This type allows a task to sleep either for a given period, or up until a given time, and then wake and continue. Obviously a task can set only one such timer in operation at a time, and once set, it cannot be cancelled. These timers have no identifier.

The second type of timer is the event timer. This type allows a task to send events to itself either after a given period or at a given time. A task can have more than one event timer running at a time. Each event timer is assigned an identifier by the kernel when the event is set. This identifier can be used to cancel the timer.

Timers are purely local objects. They affect only the calling task, either by putting it to sleep or sending it events. Timers exist only while they are running. When they expire or are cancelled, they are deleted from the kernel data structure.