Exception Operations

```
exceptions_send      ( tid, exceptions )
exceptions_return    ( )
```

Clock Operations

```
clock_set            ( clock )
clock_get            ( clock )
clock_tick           (  )
```

Timer Operations

```
timer_wake_after     ( ticks )
timer_wake_when      ( clock )
timer_event_after    ( ticks, event, tmid )
timer_event_when     ( clock, event, tmid )
timer_cancel         ( tmid )
```

D. SUMMARY OF ORKID OPERATIONS

In the following summary, output parameters are underlined.

Task Operations

```
task_create          ( name, priority, stack_size, mode, options, tid )
task_delete          ( tid )
task_ident           ( name, nid, tid )
task_start           ( tid, start_addr, arguments )
task_restart         ( tid, arguments )
task_suspend         ( tid )
task_resume          ( tid )
task_set_priority    ( tid, new_prio, old_prio )
task_set_mode        ( mode, mask, old_mode )
task_read_notepad    ( tid, loc_number, loc_value )
task_write_notepad   ( tid, loc_number, loc_value )
```

Region Operations

```
region_create        ( name, addr, length, granularity, options, rid )
region_delete        ( rid, options )
region_ident         ( name, rid )
region_get_seg       ( rid, seg_size, seg_addr )
region_ret_seg       ( rid, seg_addr )
region_info          ( rid, size, max_segment, granularity )
```

Partition Operations

```
partition_create     ( name, addr, length, block_size, options, pid )
partition_delete     ( pid, options )
partition_ident      ( name, nid, pid, block_size )
partition_get_blk    ( pid, blk_addr )
partition_ret_blk    ( pid, blk_addr )
partition_info       ( pid, blocks, free_blocks, block_size )
```

Semaphore Operations

```
sem_create           ( name, count, options, sid )
sem_delete           ( sid )
sem_ident            ( name, nid, sid )
sem_p                ( sid, time_out )
sem_v                ( sid )
sem_info             ( sit, options, count, tasks_waiting )
```

Queue Operations

```
queue_create         ( name, priv_buff, max_buff, length, options, qid )
queue_delete         ( qid )
queue_ident          ( name, nid, qid )
queue_send           ( qid, message, length )
queue_urgent         ( qid, message, length )
queue_broadcast      ( qid, message, length, count )
queue_receive        ( qid, message, time_out )
```

```
queue_flush          ( qid, count )
queue_info           ( qid, max_buf, length, options, messages_waiting,
                       tasks_waiting )

Event Operations

event_send           ( tid, event )
event_receive        ( events, options, time_out, events_caught )

Exception Operations

exceptions_catch     ( new_XSR, mode, old_XSR, old_mode )
exceptions_send      ( tid, exceptions )
exceptions_return    ( )

Clock Operations

clock_set            ( clock )
clock_get            ( clock )
clock_tick           (  )

Timer Operations

timer_wake_after     ( ticks )
timer_wake_when      ( clock )
timer_event_after    ( ticks, event, tmid )
timer_event_when     ( clock, event, tmid )
timer_cancel         ( tmid )

Interrupt Operations

int_enter            ( )
int_exit             ( )
```

```
#ifndef ORKID_H
#define ORKID_H 1
/*
```

E. ORKID: C LANGUAGE BINDING

This file contains the C language binding standard for VITA's "Open Real-time Kernel Interface Definition", henceforth called **ORKID**. The file is in the format of a C language header file, and is intended to be a common starting point for system developers wishing to produce an **ORKID** compliant kernel.

The **ORKID** C language binding consists of four sections, containing type specifications, function declarations, completion status codes and special symbol codes. The character sequence ??? has been used throughout wherever the coding is implementation dependent.

Of the four sections in this standard, only the function declarations are completely defined. In the other sections, only the type names and constant symbols are defined by this standard - all types and values are implementation dependent. Nevertheless, where possible, example values have been given.

Both ANSI C and non-ANSI C have been used for this header file. Defining the symbol __ANSI__ will cause the ANSI versions to be used, otherwise the non-ANSI versions will be used. Full prototyping has been employed for the ANSI function declarations.

```
*/
```

```
/*

ORKID TYPE SPECIFICATIONS

This section of the ORKID C language binding contains typedef
definitions for the types used in operation arguments in the main ORKID
standard.  The names are the same as those in the ORKID standard.  Only
the names, and in clock_buf the order of the structure members, are
defined by this standard. The actual types are implementation dependent.
*/


typedef unsigned int prio ;
typedef unsigned int lnum ;
typedef unsigned int bit_field ;
typedef struct { ??? } task_id ;
typedef struct { ??? } node_id ;
typedef struct { ??? } region_id ;
typedef struct { ??? } part_id ;
typedef struct { ??? } sema_id ;
typedef struct { ??? } queue_id ;
typedef struct { ??? } timer_id ;
typedef struct {
     ??? cb_year ;
     ??? cb_month ;
     ??? cb_day ;
     ??? cb_hours ;
     ??? cb_minutes ;
     ??? cb_seconds ;
     ??? cb_ticks ;
     ??? cb_time_zone ; } clock_buf ;
```