Name	Input Parameters						Output Parameters	
rn_create	name	paddr	length	pagesize	flags		&rnid	&bytes
rn_ident	name						&rnid	
rn_delete	rnid							
rn_getseg	rnid	size	flags	timeout			&segaddr	
rn_retseg	rnid	segaddr						
pt_create	name	paddr	length	bsize	flags		&ptid	&bnum
pt_ident	name	node					&ptid	
pt_delete	ptid							
pt_getbuf	ptid						&bufaddr	
pt_retbuf	ptid	bufaddr						
mm_l2p	tid	laddr					&paddr	&length
mm_p2l	tid	paddr					&laddr	&length
mm_pmap	tid	laddr	paddr	length	flags			
mm_unmap	tid	laddr						
mm_pread	paddr	laddr	length					
mm_pwrite	paddr	laddr	length					
mm_ptcreate	name	paddr	length	bsize	laddr	flags	&ptid	&bnum
m_ext2int	external						&internal	
m_int2ext	internal						&external	

TABLE 2. Directive Usage

Name	Remote	ISR	ISR to Remote
t_create	no	no	-
t_ident	yes	yes	yes
t_start	no	no	-
t_restart	no	no	-
t_delete	no	no	-
t_suspend	yes	no	-
t_resume	yes	yes	no
t_setpri	yes	no	-
t_mode	no	no	-
t_getreg	yes	yes	no
t_setreg	yes	yes	no
q_create	no	no	-
q_ident	yes	yes	yes
q_delete	no	no	-
q_send	yes	yes	no
q_urgent	yes	yes	no
q_broadcast	yes	yes	no
q_receive	yes	yes	no
ev_send	yes	yes	no
ev_receive	yes	no	-
as_catch	no	no	-
as_send	yes	yes	no
as_return	no	no	-
sm_create	no	no	-
sm_ident	yes	yes	yes
sm_delete	no	no	-
sm_p	yes	yes	no
sm_v	yes	yes	no
tm_set	yes	yes	no
tm_get	no	yes	no
tm_wkafter	no	no	-
tm_wkwhen	no	no	-
tm_evafter	no	no	-
tm_evwhen	no	no	-
tm_cancel	no	no	-
tm_tick	no	yes	no
i_return	no	yes	-
k_fatal	no	yes	-

Name	Remote	ISR	ISR to Remote
rn_create	no	no	-
rn_ident	yes	yes	yes
rn_delete	no	no	-
rn_getseg	no	no	-
rn_retseg	no	no	-
pt_create	no	no	-
pt_ident	yes	yes	yes
pt_delete	no	no	-
pt_getbuf	yes	yes	yes
pt_retbuf	yes	yes	yes
mm_l2p	no	yes	no
mm_p2l	no	no	-
mm_pmap	no	yes	no
mm_unmap	no	yes	no
mm_pread	no	no	-
mm_pwrite	no	no	-
mm_ptcreate	no	no	-
m_ext2int	no	yes	no
m_int2ext	no	yes	no

NO

3.1 Task Management

A task is a function that can execute concurrently with other functions within a multitasking environment. A task typically accepts one or more inputs, performs some processing function based on the input, and responds with one or more outputs.

A task is created using the *t_create* directive. Once a task is created, other tasks can refer to it and act on its behalf in allocating resources to it. A task is started with the *t_start* directive. Once a task has been started, it can execute its function and vie with other tasks for processor time according to its relative priority.

A task may be deleted with the *t_delete* directive. All knowledge of the task is removed from the system, and other tasks referring to it will be returned an error.

All tasks have a task identifier *(tid)*. The *tid* is assigned to the task at creation time, and must be used in all subsequent calls to the executive to identify that task. The *t_ident* directive may be used to obtain the *tid* of another task when the task name is known.

All tasks have a priority. A task's priority is a measure of the task's importance relative to all other tasks within the system and indicate its "need to run" in a multitasking environment where many tasks may be ready to run at any moment. A task is given a priority at creation time. A task's priority may be changed with the *t_setpri* directive.

A task's mode of execution is set up initially with the *t_start* directive, and may be changed using the *t_mode* directive. The mode of a task specifies its ability to be preempted, timesliced, to execute in user mode, to execute in supervisor mode at an optional interrupt level, and to disable/enable its asynchronous signal routine.

The task manager provides the pair of directives, *t_suspend* and *t_resume*, to control execution of another task.

A task is provided with a set of eight user and eight system defined software registers which may be set with the *t_setreg* directive, and read with the *t_getreg* directive.

The directives provided by the task manager are:

Directive	Function
t_create	Create a task
t_ident	Obtain id of a task
t_delete	Delete a task
t_start	Start a task
t_restart	Restart a task
t_suspend	Suspend a task
t_resume	Resume a task
t_setpri	Set task priority
t_mode	Change task mode
t_getreg	Get task register
t_setreg	Set task register

3.1.1 T_CREATE

NAME

t_create -- "Create a Task"

SYNOPSIS

uint t_create (name, superstk, userstk, priority, flags, &tid)

```
uint name;      /* user defined 4-byte task name */
uint superstk;  /* supervisor stack size in bytes */
uint userstk;   /* user stack size in bytes */
uint priority;  /* task priority */
uint flags;     /* task attributes */
uint tid;       /* task id - returned by this call */
```

Flags is defined as follows:

CMASK		Coprocessor mask 0 = no coprocessor
GLOBAL	set	to indicate the task is a multiprocessor global resource.
	clear	to indicate the task is local

DESCRIPTION

The *t_create* directive creates a task by allocating and initialising a task data structure. A task is created by name. A task id is returned to the caller in the *tid* field. The *tid* must be used in all calls to the executive requiring a *tid*.

The task is allocated a user stack and supervisor stack as determined by the values in the *userstk* and *superstk* fields. A minimum supervisor stack is required, and an error will be returned if the *superstk* value is too small. There is no minimum user stack required.

By setting the GLOBAL value in the flags field, the *tid* will be sent to all processors in the system, to be entered into a global resource table. The system is defined as the collection of interconnected processors. The task is always created on the local node.

The newly created task will be placed in the dormant state. The *t_start* directive will make the task ready, in priority order. The executive will support a minimum of 32 priorities.

The maximum number of tasks is a configuration parameter.

RETURN VALUE

If *t_create* successfully created a task, the *tid* is filled in, and 0 is returned.