# Mach 3.0 Cthreads Internals

Randall W. Dean

Carnegie
Mellon

Mach

# Old Implementations

- Task
- Coroutine
- Thread

# New Interface Calls

- cthread_set_kernel_limit
- cthread_kernel_limit
- cthread_mach_msg
- cthread_msg_active
- cthread_msg_busy
- cthread_wire
- cthread_unwire

# Modules

- malloc.c : simple multithread safe malloc
- sync.c : spin_lock implementation
- machine/lock.s : spin_try_lock implementation
- mig_support.c
- stack.c
- machine/thread.c
- cthreads.c
- cprocs.c

# mig_support.c

- mig_init(cproc)
- mig_get_reply_port
- mig_dealloc_reply_port

# stack.c

- Probe for size
- Size if power of 2
- Align appropriately
- cproc_self at top

# machine/thread.c

- Internal Routines
    - cproc_setup
    - cthread_sp

# cthreads.c

- Manager of struct cthread

```
typedef struct cthread {
        struct cthread *next;
        struct mutex lock;
        struct condition done;
        int state;
        jmp_buf catch;
        cthread_fn_t func;
        any_t arg;
        any_t result;
        char *name;
        any_t data;
        struct ur_cthread *ur;
} *cthread_t;
```

# cthreads.c <2>

- Internal Routines
    - cthread_alloc
    - cthread_free
    - cthread_body
- Exported Routines
    - cthread_init
    - cthread_fork
    - cthread_detach
    - cthread_join
    - cthread_exit
    - cthread_set_name
    - cthread_name
    - cthread_set_limit
    - cthread_limit

# cprocs.c

- Manager of struct cproc

```
typedef struct cproc {
        struct cproc *next;
        cthread_t incarnation;
        struct cproc *list;
#ifdef  WAIT_DEBUG
        char *waiting_for;
#endif  WAIT_DEBUG
        mach_port_t reply_port;
        int context;
        spin_lock_t lock;
        int state;
#define CPROC_RUNNING   0
#define CPROC_SWITCHING 1
#define CPROC_BLOCKED   2
#define CPROC_CONDWAIT  4
        mach_port_t wired;
        int busy;
        mach_msg_header_t msg;
        unsigned int stack_base;
        unsigned int stack_size;
} *cproc_t;
```

# cprocs.c <2>

- Manager of struct port_entry typedef struct port_entry { struct port_entry *next; mach_port_t port; struct cthread_queue queue; int min; int max; int held; spin_lock_t lock; } *port_entry_t;

# cprocs.c <3>

- Internal Routines
  - cproc_init
  - cproc_alloc
  - cproc_ready
  - cproc_waiting
  - cproc_waiter
  - cproc_block
  - cproc_create
  - get_port_entry

# cprocs.c <4>

- Exported Routines
  - cthread_set_kernel_limit
  - cthread_kernel_limit
  - cthread_wire
  - cthread_unwire
  - condition_wait
  - condition_signal
  - condition_broadcast
  - cthread_yield
  - mutex_lock
  - mutex_unlock
  - cthread_mach_msg
  - cthread_msg_busy
  - cthread_msg_active