3.2.9 Q_URGENT

NAME

q_urgent -- "Place an Urgent Message at the Head of a Message Queue"

SYNOPSIS

```
#include <message.h>
uint q_urgent ( qid, buffer )

        uint qid;           /* message queue id returned from q_create or q_ident */
        long (*buffer)[4];  /* pointer to message buffer */
```

DESCRIPTION

The *q_urgent* directive sends a message to the queue identified by the *qid*. This call is the same as the *q_send* call, except, if there are other messages at the queue, this message is put at the head of the queue.

If a task is already waiting at the queue, the message is copied to that task's indicated receiving buffer. The task is then made ready. If there is no task waiting, the message is copied to a system buffer which is then placed at the head of the message queue.

Once sent, the task's message area may be reused immediately. A message is fixed length, 16-bytes.

The message queue may exist on the local processor or any remote processor in a multiprocessor configuration, as long as the queue was created with the GLOBAL flags value set (see *q_create*).

RETURN VALUE

If the *q_urgent* directive successfully sent a message, then 0 is returned.

If the call was not successful, an error code is returned.

ERROR CONDITIONS

Message *qid* is invalid.

Out of system message buffers.

Message queue at maximum count.

ISR cannot reference remote node.

NOTES

Can be called from within an ISR, except when the queue was not created from the local node.

May cause a preempt if a task waiting at the message queue has a higher priority than the running task, and the preempt mode is in effect. A preempt will not occur if a task waiting exists on

a remote processor in a multiprocessor configuration.

3.2.10 Q_BROADCAST

NAME

q_broadcast -- "Broadcast N Identical Messages to a Message Queue"

SYNOPSIS

```
#include <message.h>
uint q_broadcast ( qid, buffer, &count )

        uint qid;           /* message queue id returned from q_create or q_ident */
        long (*buffer)[4];  /* pointer to message buffer */
        uint count;         /* number of tasks made ready - returned by this call */
```

DESCRIPTION

The *q_broadcast* directive sends as many messages as necessary to make ready all tasks waiting on the queue identified by the *qid*. The number of tasks readied is returned to the caller in *count*.

Once sent, the task's message buffer may be reused immediately.

The message queue may exist on the local processor or any remote processor in a multiprocessor configuration, as long as the queue was created with the GLOBAL flags value set (see *q_create*).

RETURN VALUE

If the *q_broadcast* directive succeeds, the *count* is filled in with the number of tasks readied, and 0 is returned.

If the call was not successful, an error code is returned.

ERROR CONDITIONS

Message *qid* is invalid.

ISR cannot reference remote node.

NOTES

Can be called from within an ISR, except when the queue was not created from the local node.

May cause a preempt if a task waiting at the message queue has a higher priority than the running task, and the preempt mode is in effect. A preempt will not occur if a task waiting exists on a remote processor in a multiprocessor configuration.

3.2.11 Q_RECEIVE

NAME

q_receive -- "Receive a Message from a Message Queue"

SYNOPSIS

```
#include <message.h>
uint q_receive ( qid, buffer, flags, timeout )

        uint qid;           /* message queue id returned from q_create or q_ident */
        long (*buffer)[4];  /* pointer to message buffer */
        uint flags;         /* options */
        uint timeout;       /* number of ticks to wait */
                            /* 0 indicates wait forever */
```

The flags values are:

NOWAIT	set	if the task is to return immediately
	clear	if the task is to wait for a message

DESCRIPTION

The *q_receive* directive allows a task to request a message from the message queue identified by *qid*.

If there is a message at the message queue, it is copied into the requester's buffer.

If there is no message at the message queue, then the NOWAIT flag determines what to do. If the NOWAIT flags value is set, the task returns immediately with -1 and the no message at queue error number. If the NOWAIT flags value is clear, the task is put on a wait list for the message queue, according the queue's attributes (FIFO or priority).

The *timeout* field is used to determine how long to wait. A zero in the *timeout* field indicates no timeout -- wait forever. A non-zero entry in the *timeout* field indicates that the task will run after that many ticks, if a message has not been received, or before if a message is received.

When *q_receive* is called from an ISR, the no wait option is forced by the executive. Thus there will be no waiting for a message. An error will be returned if there is no message.

The message queue may exist on the local processor or any remote processor in a multiprocessor configuration, as long as the queue was created with the GLOBAL flags value set (see *q_create*).

RETURN VALUE

If the *q_receive* directive succeeds, then 0 is returned.

If the call was not successful, an error code is returned.

ERROR CONDITIONS

Message *qid* is invalid.

No message at queue (if no wait is selected).

Message queue deleted.

Timed out with no message (if wait and timeout is selected).

ISR cannot reference remote node.

NOTES

Can be called from within an ISR, except when the queue was not created from the local node. The executive will force the options to no wait.

The requesting task may be blocked if there is no message available, and the wait option is selected.