7.5. QUEUE_URGENT

Send a message to head of queue.

Synopsis

```
queue_urgent( qid, message, length )
```

Input Parameters

qid	: queue_id	kernel defined queue identifier
message	: address	message starting address
length	: integer	message length in bytes

Output Parameters

<none>

Completion Status

OK	queue_urgent operation successful
INVALID_PARAMETER	a parameter refers to an illegal address
INVALID_ID	queue does not exist
OBJECT_DELETED	queue specified has been deleted
INVALID_LENGTH	message length greater than queue's buffer length
QUEUE_FULL	no more buffers available
NODE_NOT_REACHABLE	node on which semaphore resides is not reachable

Description

This operation sends a priority message to a queue.

If the queue's wait queue contains a number of tasks waiting on messages, then the action is exactly the same as for queue send. The message is delivered to the task at the head of the wait queue. This task is then removed from the wait queue, unblocked and will be returned a successful completion status along with the message.

Otherwise the message is inserted at the head of the message queue. If there is no memory available for the buffer, then the NO_MORE_MEMORY completion status is returned.

7.6. QUEUE_BROADCAST

Broadcast message to all tasks blocked on a queue.

Synopsis

```
queue_broadcast( qid, message, length, count )
```

Input Parameters

qid	: queue_id	kernel defined queue identifier
message	: address	message starting address
length	: integer	message length in bytes

Output Parameters

count	: integer	number of unblocked tasks

Completion Status

OK	queue_broadcast operation successful
ILLEGAL_USE	operation not callable from ISR
INVALID_PARAMETER	a parameter refers to an illegal address
INVALID_ID	queue does not exist
OBJECT_DELETED	queue specified has been deleted
INVALID_LENGTH	message length greater than queue's buffer length
NODE_NOT_REACHABLE	node on which semaphore resides is not reachable

Description

This operation sends a message to all tasks waiting on the queue. If the wait queue is empty, then no messages are sent, no tasks are unblocked and the count returned will be zero. If the wait queue contains a number of tasks waiting on messages, then the message is delivered to each task in the wait queue. All tasks are then removed from the wait queue, unblocked and returned a successful completion status. The number of tasks unblocked is returned in the count parameter.

This operations is atomic with respect to other operations on the queue.

7.7. QUEUE_RECEIVE

Receive a message from a queue.

Synopsis

```
queue_receive( qid, message, options, time_out )
```

Input Parameters

qid	: queue_id	kernel defined queue identifier
message	: address	address to put message
options	: bit_field	queue receive options
time_out	: integer	max number of ticks to wait

Output Parameters

<none>

Literal Values

options	+ NOWAIT	do not wait - return immediately if no message in queue
time_out	= FOREVER	wait forever - do not time out

Completion Status

OK	queue_receive operation successful
ILLEGAL_USE	operation not callable from ISR
INVALID_PARAMETER	a parameter refers to an illegal address
INVALID_ID	queue does not exist
OBJECT_DELETED	queue specified has been deleted
INVALID_ADDRESS	message refers to an illegal address
INVALID_OPTIONS	invalid options value
TIME_OUT	queue-receive operation timed out
QUEUE_DELETED	queue deleted while blocked in queue_receive operation
QUEUE_EMPTY	queue empty with NOWAIT option
NODE_NOT_REACHABLE	node on which semaphore resides is not reachable

Description

This operation receives a message from a given queue. If there are one or more messages on the queue, then the buffer at the head is removed from the queue, its message is copied into the given area, the buffer is deallocated, and a successful completion status returned.

If the queue is empty, and NOWAIT was not specified in the options, then the task is blocked and put on the queue's wait queue in order of task priority or first in first out. If NOWAIT was specified and the queue is empty, then the QUEUE_EMPTY completion status is returned. If the queue is deleted while the task is waiting on a message from it, then the QUEUE_DELETED completion status is returned. If the

timeout expires, then the TIME_OUT completion status is returned. Otherwise, when the task reaches the head of the queue and a message is sent, or if a message is broadcast while the task is anywhere in the queue, then the task receives the message and is returned a successful completion status.

7.8. QUEUE_FLUSH

Flush all messages on a queue.

Synopsis

```
queue_flush( qid, count )
```

Input Parameters

qid	: queue_id	kernel defined queue identifier

Output Parameters

count	: integer	number of flushed messages

Completion Status

OK	queue_flush operation successful
ILLEGAL_USE	operation not callable from ISR
INVALID_PARAMETER	a parameter refers to an illegal address
INVALID_ID	queue does not exist
OBJECT_DELETED	queue specified has been deleted
NODE_NOT_REACHABLE	node on which semaphore resides is not reachable

Description

If there were one or more messages in the specified queue, then they are removed from the queue, their buffers deallocated and their number returned in count. If there were no messages in the queue, then a count of zero is returned.