

NAME

buf_ring, **buf_ring_alloc**, **buf_ring_free**, **buf_ring_enqueue**, **buf_ring_dequeue_mc**,
buf_ring_dequeue_sc, **buf_ring_count**, **buf_ring_empty**, **buf_ring_full**, **buf_ring_peek** - multi-producer,
{single, multi}-consumer lock-less ring buffer

SYNOPSIS

```
#include <sys/param.h>
```

```
#include <sys/buf_ring.h>
```

```
struct buf_ring *
```

```
buf_ring_alloc(int count, struct malloc_type *type, int flags, struct mtx *sc_lock);
```

```
void
```

```
buf_ring_free(struct buf_ring *br, struct malloc_type *type);
```

```
int
```

```
buf_ring_enqueue(struct buf_ring *br, void *buf);
```

```
void *
```

```
buf_ring_dequeue_mc(struct buf_ring *br);
```

```
void *
```

```
buf_ring_dequeue_sc(struct buf_ring *br);
```

```
int
```

```
buf_ring_count(struct buf_ring *br);
```

```
int
```

```
buf_ring_empty(struct buf_ring *br);
```

```
int
```

```
buf_ring_full(struct buf_ring *br);
```

```
void *
```

```
buf_ring_peek(struct buf_ring *br);
```

DESCRIPTION

The **buf_ring** functions provide a lock-less multi-producer and lock-less multi-consumer as well as single-consumer ring buffer.

The **buf_ring_alloc()** function is used to allocate a `buf_ring` ring buffer with *count* slots using `malloc_type` *type* and memory flags *flags*. The single consumer interface is protected by *sc_lock*.

The **buf_ring_free()** function is used to free a `buf_ring`. The user is responsible for freeing any enqueued items.

The **buf_ring_enqueue()** function is used to enqueue a buffer to a `buf_ring`.

The **buf_ring_dequeue_mc()** function is a multi-consumer safe way of dequeuing elements from a `buf_ring`.

The **buf_ring_dequeue_sc()** function is a single-consumer interface to dequeue elements - requiring the user to serialize accesses with a lock.

The **buf_ring_count()** function returns the number of elements in a `buf_ring`.

The **buf_ring_empty()** function returns TRUE if the `buf_ring` is empty, FALSE otherwise.

The **buf_ring_full()** function returns TRUE if no more items can be enqueued, FALSE otherwise.

The **buf_ring_peek()** function returns a pointer to the last element in the `buf_ring` if the `buf_ring` is not empty, NULL otherwise.

RETURN VALUES

The **buf_ring_enqueue()** function return ENOBUFS if there are no available slots in the `buf_ring`.

HISTORY

These functions were introduced in FreeBSD 8.0.