

**NAME**

**ethers, ether\_line, ether\_aton, ether\_aton\_r, ether\_ntoa, ether\_ntoa\_r, ether\_ntohost, ether\_hostton** - Ethernet address conversion and lookup routines

**LIBRARY**

Standard C Library (libc, -lc)

**SYNOPSIS**

```
#include <sys/types.h>
#include <sys/socket.h>
#include <net/ethernet.h>
```

*int*

```
ether_line(const char *l, struct ether_addr *e, char *hostname);
```

*struct ether\_addr \**

```
ether_aton(const char *a);
```

*struct ether\_addr \**

```
ether_aton_r(const char *a, struct ether_addr *e);
```

*char \**

```
ether_ntoa(const struct ether_addr *n);
```

*char \**

```
ether_ntoa_r(const struct ether_addr *n, char *buf);
```

*int*

```
ether_ntohost(char *hostname, const struct ether_addr *e);
```

*int*

```
ether_hostton(const char *hostname, struct ether_addr *e);
```

**DESCRIPTION**

These functions operate on ethernet addresses using an *ether\_addr* structure, which is defined in the header file *<net/ethernet.h>*:

```
/*
 * The number of bytes in an ethernet (MAC) address.
 */
```

```

#define ETHER_ADDR_LEN          6

/*
 * Structure of a 48-bit Ethernet address.
 */
struct ether_addr {
    u_char octet[ETHER_ADDR_LEN];
};

```

The function **ether\_line()** scans *l*, an ASCII string in ethers(5) format and sets *e* to the ethernet address specified in the string and *h* to the hostname. This function is used to parse lines from */etc/ethers* into their component parts.

The **ether\_aton()** and **ether\_aton\_r()** functions convert ASCII representation of ethernet addresses into *ether\_addr* structures. Likewise, the **ether\_ntoa()** and **ether\_ntoa\_r()** functions convert ethernet addresses specified as *ether\_addr* structures into ASCII strings.

The **ether\_ntohost()** and **ether\_hostton()** functions map ethernet addresses to their corresponding hostnames as specified in the */etc/ethers* database. The **ether\_ntohost()** function converts from ethernet address to hostname, and **ether\_hostton()** converts from hostname to ethernet address.

## RETURN VALUES

The **ether\_line()** function returns zero on success and non-zero if it was unable to parse any part of the supplied line *l*. It returns the extracted ethernet address in the supplied *ether\_addr* structure *e* and the hostname in the supplied string *h*.

On success, **ether\_ntoa()** and **ether\_ntoa\_r()** functions return a pointer to a string containing an ASCII representation of an ethernet address. If it is unable to convert the supplied *ether\_addr* structure, it returns a NULL pointer. **ether\_ntoa()** stores the result in a static buffer; **ether\_ntoa\_r()** stores the result in a user-passed buffer.

Likewise, **ether\_aton()** and **ether\_aton\_r()** return a pointer to an *ether\_addr* structure on success and a NULL pointer on failure. **ether\_aton()** stores the result in a static buffer; **ether\_aton\_r()** stores the result in a user-passed buffer.

The **ether\_ntohost()** and **ether\_hostton()** functions both return zero on success or non-zero if they were unable to find a match in the */etc/ethers* database.

## NOTES

The user must ensure that the hostname strings passed to the **ether\_line()**, **ether\_ntohost()** and

**ether\_hostton()** functions are large enough to contain the returned hostnames.

## NIS INTERACTION

If the */etc/ethers* contains a line with a single + in it, the **ether\_ntohost()** and **ether\_hostton()** functions will attempt to consult the NIS *ethers.byname* and *ethers.byaddr* maps in addition to the data in the */etc/ethers* file.

## SEE ALSO

*ethers(5)*, *yp(8)*

## HISTORY

This particular implementation of the **ethers** library functions were written for and first appeared in FreeBSD 2.1. Thread-safe function variants first appeared in FreeBSD 7.0.

## BUGS

The **ether\_aton()** and **ether\_ntoa()** functions returns values that are stored in static memory areas which may be overwritten the next time they are called.

**ether\_ntoa\_r()** accepts a character buffer pointer, but not a buffer length. The caller must ensure adequate space is available in the buffer in order to avoid a buffer overflow.