

**NAME**

`BN_set_bit`, `BN_clear_bit`, `BN_is_bit_set`, `BN_mask_bits`, `BN_lshift`, `BN_lshift1`, `BN_rshift`, `BN_rshift1` - bit operations on `BIGNUMs`

**SYNOPSIS**

```
#include <openssl/bn.h>

int BN_set_bit(BIGNUM *a, int n);
int BN_clear_bit(BIGNUM *a, int n);

int BN_is_bit_set(const BIGNUM *a, int n);

int BN_mask_bits(BIGNUM *a, int n);

int BN_lshift(BIGNUM *r, const BIGNUM *a, int n);
int BN_lshift1(BIGNUM *r, BIGNUM *a);

int BN_rshift(BIGNUM *r, BIGNUM *a, int n);
int BN_rshift1(BIGNUM *r, BIGNUM *a);
```

**DESCRIPTION**

`BN_set_bit()` sets bit `n` in `a` to 1 ( $a | = (1 < n)$ ). The number is expanded if necessary.

`BN_clear_bit()` sets bit `n` in `a` to 0 ( $a \&= \sim (1 < n)$ ). An error occurs if `a` is shorter than `n` bits.

`BN_is_bit_set()` tests if bit `n` in `a` is set.

`BN_mask_bits()` truncates `a` to an `n` bit number ( $a \&= \sim ( \sim 0 ) >> n$ ). An error occurs if `a` already is shorter than `n` bits.

`BN_lshift()` shifts `a` left by `n` bits and places the result in `r` ( $r = a * 2^n$ ). `BN_lshift1()` shifts `a` left by one and places the result in `r` ( $r = 2 * a$ ).

`BN_rshift()` shifts `a` right by `n` bits and places the result in `r` ( $r = a / 2^n$ ). `BN_rshift1()` shifts `a` right by one and places the result in `r` ( $r = a / 2$ ).

For the shift functions, `r` and `a` may be the same variable.

**RETURN VALUES**

`BN_is_bit_set()` returns 1 if the bit is set, 0 otherwise.

All other functions return 1 for success, 0 on error. The error codes can be obtained by [ERR\\_get\\_error\(3\)](#).

**SEE ALSO**

[bn\(3\)](#), [BN\\_num\\_bytes\(3\)](#), [BN\\_add\(3\)](#)

**HISTORY**

`BN_set_bit()`, `BN_clear_bit()`, `BN_is_bit_set()`, `BN_mask_bits()`, `BN_lshift()`, `BN_lshift1()`, `BN_rshift()`, and `BN_rshift1()` are available in all versions of SSLeay and OpenSSL.