

**NAME**

BN\_mod\_mul\_montgomery, BN\_MONT\_CTX\_new, BN\_MONT\_CTX\_init, BN\_MONT\_CTX\_free, BN\_MONT\_CTX\_set, BN\_MONT\_CTX\_copy, BN\_from\_montgomery, BN\_to\_montgomery - Montgomery multiplication

**SYNOPSIS**

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

BN_MONT_CTX *BN_MONT_CTX_new(void);
void BN_MONT_CTX_init(BN_MONT_CTX *ctx);
void BN_MONT_CTX_free(BN_MONT_CTX *mont);

int BN_MONT_CTX_set(BN_MONT_CTX *mont, const BIGNUM *m, BN_CTX *ctx);
BN_MONT_CTX *BN_MONT_CTX_copy(BN_MONT_CTX *to, BN_MONT_CTX *from);

int BN_mod_mul_montgomery(BIGNUM *r, BIGNUM *a, BIGNUM *b,
BN_MONT_CTX *mont, BN_CTX *ctx);

int BN_from_montgomery(BIGNUM *r, BIGNUM *a, BN_MONT_CTX *mont,
BN_CTX *ctx);

int BN_to_montgomery(BIGNUM *r, BIGNUM *a, BN_MONT_CTX *mont,
BN_CTX *ctx);
```

**DESCRIPTION**

These functions implement Montgomery multiplication. They are used automatically when [BN\\_mod\\_exp\(3\)](#) is called with suitable input, but they may be useful when several operations are to be performed using the same modulus.

*BN\_MONT\_CTX\_new()* allocates and initializes a **BN\_MONT\_CTX** structure. *BN\_MONT\_CTX\_init()* initializes an existing uninitialized **BN\_MONT\_CTX**.

*BN\_MONT\_CTX\_set()* sets up the *mont* structure from the modulus *m* by precomputing its inverse and a value *R*.

*BN\_MONT\_CTX\_copy()* copies the **BN\_MONT\_CTX** from *to* to *from*.

*BN\_MONT\_CTX\_free()* frees the components of the **BN\_MONT\_CTX**, and, if it was created by *BN\_MONT\_CTX\_new()*, also the structure itself.

*BN\_mod\_mul\_montgomery()* computes  $\text{Mont}(a,b) := a*b*R^{-1}$  and places the result in *r*.

*BN\_from\_montgomery()* performs the Montgomery reduction  $r = a*R^{-1}$ .

*BN\_to\_montgomery()* computes  $\text{Mont}(a,R^2)$ , i.e.  $a*R$ . Note that *a* must be non-negative and smaller than the modulus.

For all functions, *ctx* is a previously allocated **BN\_CTX** used for temporary variables.

The **BN\_MONT\_CTX** structure is defined as follows:

```
typedef struct bn_mont_ctx_st
{
    int ri; /* number of bits in R */
    BIGNUM RR; /* R^2 (used to convert to Montgomery form) */
    BIGNUM N; /* The modulus */
    BIGNUM Ni; /* R*(1/R mod N) - N*Ni = 1
    * (Ni is only stored for bignum algorithm) */
    BN_ULONG n0; /* least significant word of Ni */
    int flags;
} BN_MONT_CTX;
```

*BN\_to\_montgomery()* is a macro.

## RETURN VALUES

*BN\_MONT\_CTX\_new()* returns the newly allocated **BN\_MONT\_CTX**, and NULL on error.

*BN\_MONT\_CTX\_init()* and *BN\_MONT\_CTX\_free()* have no return values.

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

## WARNING

The inputs must be reduced modulo **m**, otherwise the result will be outside the expected range.

## SEE ALSO

[bn\(3\)](#), [ERR\\_get\\_error\(3\)](#), [BN\\_add\(3\)](#), [BN\\_CTX\\_new\(3\)](#)

## HISTORY

*BN\_MONT\_CTX\_new()*, *BN\_MONT\_CTX\_free()*, *BN\_MONT\_CTX\_set()*, *BN\_mod\_mul\_montgomery()*, *BN\_from\_montgomery()* and *BN\_to\_montgomery()* are available in all versions of SSLeay and OpenSSL.

*BN\_MONT\_CTX\_init()* and *BN\_MONT\_CTX\_copy()* were added in SSLeay 0.9.1b.