## NAME

BIO_set_flags, BIO_clear_flags, BIO_test_flags, BIO_get_flags, BIO_set_retry_read, BIO_set_retry_write, BIO_set_retry_special, BIO_clear_retry_flags, BIO_get_retry_flags − manipulate and interpret BIO flags

## SYNOPSIS

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

void BIO_set_flags(BIO *b, int flags);
void BIO_clear_flags(BIO *b, int flags);
int  BIO_test_flags(const BIO *b, int flags);
int  BIO_get_flags(const BIO *b);

void BIO_set_retry_read(BIO *b);
void BIO_set_retry_write(BIO *b);
void BIO_set_retry_special(BIO *b);
void BIO_clear_retry_flags(BIO *b);
int  BIO_get_retry_flags(BIO *b);
```

## DESCRIPTION

A **BIO** has an internal set of bit flags that describe its state. These functions and macros are used primarily by **BIO** implementations and by code that builds **BIO** chains to manipulate those flags.

**BIO_set_flags()** sets the bits given in *flags* in the **BIO** *b*. Any bits already set in the **BIO**'s flag word remain set.

**BIO_clear_flags()** clears the bits given in *flags* from the **BIO** *b*. Any other bits in the flag word are left unchanged.

**BIO_test_flags()** tests the bits given in *flags* in the **BIO** *b* and returns a nonzero value if any of them are currently set and zero otherwise.

**BIO_get_flags()** returns the current flag word from the **BIO** *b*. This is equivalent to testing for all bits and returning the result.

The following convenience macros are built on top of these primitives and are used to maintain the retry state of a BIO:

**BIO_set_retry_read()** marks the **BIO** *b* as being in a retryable state by setting the **BIO_FLAGS_SHOULD_RETRY** flag. In addition, it sets the **BIO_FLAGS_READ** flag to indicate that the retry condition is associated with a read operation.

**BIO_set_retry_write()** marks the **BIO** *b* as being in a retryable state by setting the **BIO_FLAGS_SHOULD_RETRY** flag. In addition, it sets the **BIO_FLAGS_WRITE** flag to indicate that the retry condition is associated with a write operation.

**BIO_set_retry_special()** marks the **BIO** *b* as being in a retryable state by setting the **BIO_FLAGS_SHOULD_RETRY** flag. In addition, it sets the **BIO_FLAGS_IO_SPECIAL** flag to indicate that the retry condition is associated with a read operation some "special" condition. The precise meaning of this condition depends on the **BIO** type.

**BIO_clear_retry_flags()** clears all retry−related bits from *b*, i.e. **BIO_FLA GS_READ**, **BIO_FLAGS_WRITE**, **BIO_FLAGS_IO_SPECIAL**, and **BIO_FLAGS_SHOULD_RETRY**.

**BIO_get_retry_flags()** returns retry−related bits that are currently set in *b*. The result is a subset of **BIO_FLAGS_RWS|BIO_FLAGS_SHOULD_RETRY**.

The retry bits are interpreted by the higher level macros **BIO_should_read()**, **BIO_should_write()**, **BIO_should_io_special()**, **BIO_retry_type()** and **BIO_should_retry()**, as documented in **BIO_should_retry**(3). Application code will typically use those macros rather than manipulate the underlying flags directly.

The following flag bits are currently defined for use with **BIO_set_flags()**, **BIO_clear_flags()** and **BIO_test_flags()**:

**BIO_FLAGS_READ**
The last I/O operation should be retried when the **BIO** becomes readable. This flag is normally set by the **BIO** implementation via **BIO_set_retry_read()** after a failed read operation.

**BIO_FLAGS_WRITE**
The last I/O operation should be retried when the **BIO** becomes writable. This flag is normally set by the **BIO** implementation via **BIO_set_retry_write()** after a failed write operation.

**BIO_FLAGS_IO_SPECIAL**
The last I/O operation should be retried when some "special" condition becomes true. The precise meaning of this condition depends on the **BIO** type and is usually obtained via **BIO_get_retry_BIO()** and **BIO_get_retry_reason()** as described in **BIO_should_retry**(3). This flag is normally set by the **BIO** implementation via **BIO_set_retry_special()**.

**BIO_FLAGS_RWS**
The bitwise OR of **BIO_FLAGS_READ**, **BIO_FLAGS_WRITE** and **BIO_FLAGS_IO_SPECIAL**. This mask is used when clearing or extracting the retry−direction bits.

**BIO_FLAGS_SHOULD_RETRY**
Set if the last I/O operation on the **BIO** should be retried at a later time. If this bit is not set then the condition is treated as an error. This flag is normally set by the**BIO** implementation.

**BIO_FLAGS_BASE64_NO_NL**
When set on a base64 filter **BIO** this flag disables the generation of newline characters in the encoded output and causes newlines to be ignored in the input. See also **BIO_f_base64**(3). The flag has no effect on any other built−in **BIO** types.

**BIO_FLAGS_MEM_RDONLY**
When set on a memory **BIO** this flag indicates that the underlying buffer is read only. Attempts to write to such a **BIO** will fail. The flag has no effect on any other built−in **BIO** types.

**BIO_FLAGS_NONCLEAR_RST**
On a memory **BIO** this flag modifies the behaviour of **BIO_reset()**. When it is set, resetting the**BIO** does not clear the underlying buffer but only resets the current read position. The flag has no effect on any other built−in **BIO** types.

**BIO_FLAGS_IN_EOF**
This flag may be used by a **BIO** implementation to indicate that the end of the input stream has been reached. However, **BIO** types are not required to use this flag to signal end−of−file conditions; they may rely on other mechanisms such as system calls or by querying the next **BIO** in a chain. Applications must therefore not test this flag directly to determine whether EOF has been reached, and must use **BIO_eof()** instead.

A range of additional flag values is reserved for internal use by OpenSSL to track kernel TLS (KTLS) state. This range and the corresponding flag macros are not part of the public API and must not be used by applications.

## RETURN VALUES

**BIO_get_flags()** returns a bit mask of the flags currently set on the **BIO**.

**BIO_test_flags()** returns a bit mask consisting of those flags from the argument that are currently set in the **BIO**. Consequently, it returns a nonzero value if and only if at least one of the requested flags is set.

**BIO_get_retry_flags()** returns a bit mask consisting of those flags from **BIO_FLAGS_READ**, **BIO_FLAGS_WRITE**, **BIO_FLAGS_IO_SPECIAL**, and **BIO_FLAGS_SHOULD_RETRY** that are currently set in the *BIO*.

## NOTES

Ordinary application code will rarely need to call **BIO_set_flags()**, **BIO_clear_flags()** or **BIO_test_flags()** directly. They are intended for **BIO** implementations and for code that forwards retry state from one **BIO** in a chain to another. After a failed I/O operation, applications should normally use **BIO_should_retry()** and related macros as described in **BIO_should_retry**(3) instead of inspecting the flags directly.

These functions and macros are not thread−safe. If a single **BIO** is accessed from multiple threads, the caller must provide appropriate external synchronisation.

## SEE ALSO

**BIO_should_retry** (3), **BIO_f_base64** (3), **bio** (7)

## HISTORY

The functions and macros described here have been available in OpenSSL since at least 1.1.0 (**BIO_FLAGS_IN_EOF** since 1.1.1).

## COPYRIGHT