

## Binary constants

1.5

Generated by Doxygen 1.7.3

Fri Dec 9 2011 00:42:43

# Contents

<b>1</b>	<b>File Index</b>	<b>1</b>
1.1	File List . . . . .	1
<b>2</b>	<b>File Documentation</b>	<b>1</b>
2.1	binary_constants.h File Reference . . . . .	1
2.1.1	Detailed Description . . . . .	1
2.1.2	Define Documentation . . . . .	2

## 1 File Index

### 1.1 File List

Here is a list of all files with brief descriptions:

- binary\_constants.h** (Enter binary constants in a legible way that only uses standard language features and thus works with every standard-compliant compiler ) **1**

## 2 File Documentation

### 2.1 binary\_constants.h File Reference

Enter binary constants in a legible way that only uses standard language features and thus works with every standard-compliant compiler.

#### Defines

- #define **HEX\_\_**(n) 0x##n##LU
- #define **B8\_\_**(x)
- #define **B8**(d) ((unsigned char)B8\_\_(HEX\_\_(d)))  
*For up to 8-bit binary constants.*
- #define **B16**(dmsb, dlsb) (((unsigned short)B8(dmsb)<<8) + B8(dlsb))  
*For up to 16-bit binary constants, MSByte first.*
- #define **B32**(dmsb, db2, db1, dlsb)  
*For up to 32-bit binary constants, MSByte first.*
- #define **B64**(dmsb, db6, db5, db4, db3, db2, db1, dlsb)  
*For up to 64-bit binary constants, MSByte first.*

### 2.1.1 Detailed Description

Enter binary constants in a legible way that only uses standard language features and thus works with every standard-compliant compiler. Enter binary constants in C/C++ source like this: `var = B8(01011100);`

All macros evaluate to compile-time constants. Macros:

`B8(01010101) = 0x55 = 85`

`B16(10101010,01010101) = 0xAA55 = 43605 = -21931`

`B32(10000000,11111111,10101010,01010101) = 0x80FFAA55 = 2164238933 = -2130728363`

`B64(11100011,11001100,11100010,11000110,10000000,11111111,10101010,01010101) = 0xE3CCE2C680FFAA55 = 16414744084054256213 = -2031999989655295403`

Obtained from <http://bytes.com/forum/thread216333.html>

(or <http://www.velocityreviews.com/forums/t318127-using-binary-numbers-in-c.html> as macros.h)

Binary constant generator macro. By Tom Torfs - donated to the public domain. Improvements by Volker Kuhlmann - also in the public domain.

Download:

#### See also

<http://volker.top.geek.nz/soft/>

History:

- v. 1.1VK 15 Oct 2007
  - Tidied up into .h file, added `B32()`, `B64()`, improved test code.
- v. 1.2VK 01 Mar 2008
  - Added broken compiler test, improved test code.
- v. 1.3VK 29 Mar 2008
  - Added surrounding conditional.
- v. 1.4VK 02 Apr 2008
  - Added missing `()` in `B8__()` (thanks lint). Version number.
- v. 1.5VK 07 Dec 2011
  - Added doxygen markup.

### 2.1.2 Define Documentation

**2.1.2.1** `#define B16( dmsb, dlsb ) (((unsigned short)B8(dmsb)<<8) + B8(dlsb))`

For up to 16-bit binary constants, MSByte first.

### 2.1.2.2 `#define B32( dmsb, db2, db1, dlsb )`

**Value:**

```
((unsigned long)B8(dmsb)<<24) \
+ ((unsigned long)B8(db2)<<16) \
+ ((unsigned long)B8(db1)<<8) \
+ B8(dlsb)
```

For up to 32-bit binary constants, MSByte first.

### 2.1.2.3 `#define B64( dmsb, db6, db5, db4, db3, db2, db1, dlsb )`

**Value:**

```
((unsigned long long)B32(dmsb,db6,db5,db4)<<32) \
+ B32(db3,db2,db1,dlsb)
```

For up to 64-bit binary constants, MSByte first.

### 2.1.2.4 `#define B8( d )((unsigned char)B8__(HEX__(d)))`

For up to 8-bit binary constants.

### 2.1.2.5 `#define B8__( x )`

**Value:**

```
((x&0x0000000FLU)?1:0) \
+ ((x&0x000000F0LU)?2:0) \
+ ((x&0x00000F00LU)?4:0) \
+ ((x&0x00000F000LU)?8:0) \
+ ((x&0x000F0000LU)?16:0) \
+ ((x&0x000F00000LU)?32:0) \
+ ((x&0x00F000000LU)?64:0) \
+ ((x&0xF0000000LU)?128:0)
```

### 2.1.2.6 `#define HEX__( n )0x##n##LU`

## Index

- B16
  - [binary\\_constants.h](#), 2
- B32
  - [binary\\_constants.h](#), 2
- B64
  - [binary\\_constants.h](#), 3
- B8
  - [binary\\_constants.h](#), 3
- B8\_\_
  - [binary\\_constants.h](#), 3
- [binary\\_constants.h](#), 1
  - B16, 2
  - B32, 2
  - B64, 3
  - B8, 3
  - B8\_\_, 3
  - HEX\_\_, 3
- HEX\_\_
  - [binary\\_constants.h](#), 3