

# Package: vectorbitops (via r-universe)

September 5, 2024

**Title** Vector Bitwise Operations

**Version** 1.1.2

**Description** A tool for fast, efficient bitwise operations along the elements within a vector. Provides such functionality for AND, OR and XOR, as well as infix operators for all of the binary bitwise operations.

**License** MIT + file LICENSE

**Suggests** spelling, testthat (>= 3.0.0)

**Config/testthat/edition** 3

**Encoding** UTF-8

**Language** en-US

**RoxygenNote** 7.2.3

**NeedsCompilation** yes

**Author** Samuel Sapire [aut, cre, cph]

**Maintainer** Samuel Sapire <sapires@protonmail.com>

**Repository** CRAN

**Date/Publication** 2024-01-08 16:50:02 UTC

## Contents

Infix Bitwise Operators . . . . .	2
Vector Bitops . . . . .	3
<b>Index</b>	<b>4</b>

---

## Infix Bitwise Operators

*Infix operators for bitwise operations.*

---

### Description

Basic infix wrapper around the `base::bitw_OP_` operations.

### Usage

`a %|% b`

`a %&% b`

`a %^% b`

`a %<<% n`

`a %>>% n`

### Arguments

<code>a, b</code>	Integer vectors. Numerics are coerced to integers.
<code>n</code>	Non-negative integer vector of values up to 31.

### Value

An integer vector of length of the longer of the arguments, or zero if one of the arguments is zero-length. NA input makes NA output.

`%|%`: A vector of pairwise ORed values.

`%&%`: A vector of pairwise ANDed values.

`%^%`: A vector of pairwise XORed values.

`%<<%`: A vector of the values on the LHS pairwise left-shifted by the RHS value.

`%>>%`: A vector of the values on the LHS pairwise right-shifted by the RHS value.

### Examples

```
1 %|% 2
1 %&% 2
1 %^% 2
1 %<<% 2
8 %>>% 2
```

**Description**

Functions to apply the same bitwise operation sequentially down a vector of integers. A fast way to AND or OR everything together when a single value is required.

**Usage**

```
bit_vector_AND(vec)
```

```
bit_vector_OR(vec)
```

```
bit_vector_XOR(vec)
```

**Arguments**

`vec`                    A vector of integers. Numeric vectors will be coerced to int.

**Value**

A single integer, the result of applying the operation in question along the vector. Input that cannot be coerced to int returns NA. An empty vector returns 0.

`bit_vector_AND`: A single integer, the result of ANDing each entry in the input vector together.

`bit_vector_OR`: A single integer, the result of ORing each entry in the input vector together.

`bit_vector_XOR`: A single integer, the result of XORing each entry in the input vector together.

**Examples**

```
bit_vector_AND(c(1,3,5,7,9))
```

```
bit_vector_OR(c(1,2,4,8,16))
```

```
bit_vector_XOR(c(1,2,3,4,5))
```

# Index

`<<` (Infix Bitwise Operators), [2](#)  
`>>` (Infix Bitwise Operators), [2](#)  
`&&` (Infix Bitwise Operators), [2](#)  
`^` (Infix Bitwise Operators), [2](#)  
`<<` (Infix Bitwise Operators), [2](#)  
`>>` (Infix Bitwise Operators), [2](#)  
`&&` (Infix Bitwise Operators), [2](#)  
`^` (Infix Bitwise Operators), [2](#)

`bit_vector_AND` (Vector Bitops), [3](#)  
`bit_vector_OR` (Vector Bitops), [3](#)  
`bit_vector_XOR` (Vector Bitops), [3](#)

Infix Bitwise Operators, [2](#)

Vector Bitops, [3](#)