

tensor.bitwise_and

tensor.bitwise_and

...

```
Copy fnbitwise_and(self:@Tensor, other:@Tensor)->Tensor;
```

...

Computes the bitwise AND of two tensors element-wise. The input tensors must have either:

- Exactly the same shape
- The same number of dimensions and the length of each dimension is either a common length or 1.
-

Args

- self
- (@Tensor
-) - The first tensor to be compared
- other
- (@Tensor
-) - The second tensor to be compared
-

Panics

- Panics if the shapes are not equal or broadcastable
-

Returns

A newTensor with the same shape as the broadcasted inputs.

Example

...

```
Copy usecore::array::{ArrayTrait,SpanTrait};
```

```
useorion::operators::tensor::{TensorTrait,Tensor,U32Tensor};
```

```
fnand_example()->Tensor { lettensor_1=TensorTrait::new( shape:array![3,3].span(), data:array![0,1,2,3,4,5,6,7,8].span(), );
```

```
lettensor_2=TensorTrait::new( shape:array![3,3].span(), data:array![0,1,2,0,4,5,0,6,2].span(), );
```

```
returntensor_1.bitwise_and(@tensor_2); }
```

```
        [0,1,2,0,4,5,0,6,2]
```

...

[Previous tensor.where](#) [Next tensor.bitwise_xor](#)

Last updated3 months ago