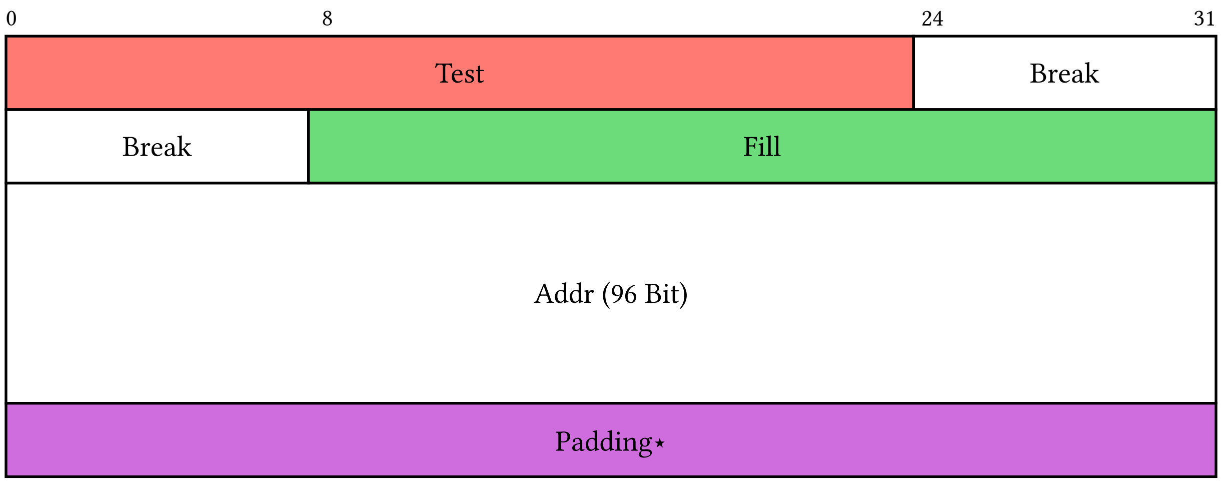
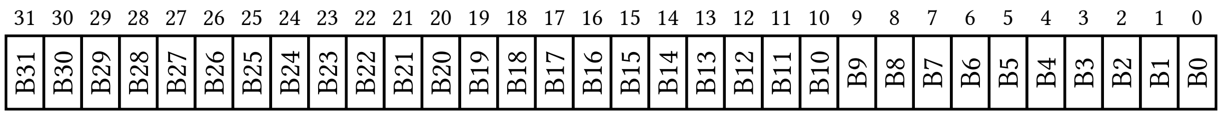
Bytefield

Colored Example



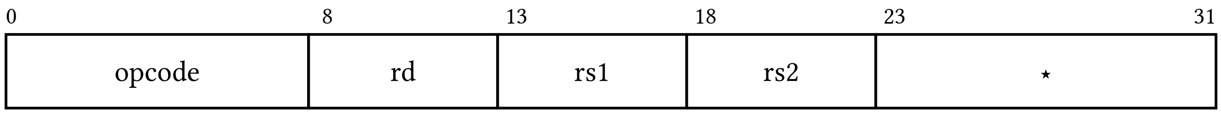
Show all bits in the bitheader

Show all bit headers with `bitheader: "all"`


31 30 29 28 27 26 25 24 23 22 21 20 19 18 17 16 15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0
B31 B30 B29 B28 B27 B26 B25 B24 B23 B22 B21 B20 B19 B18 B17 B16 B15 B14 B13 B12 B11 B10 B9 B8 B7 B6 B5 B4 B3 B2 B1 B0

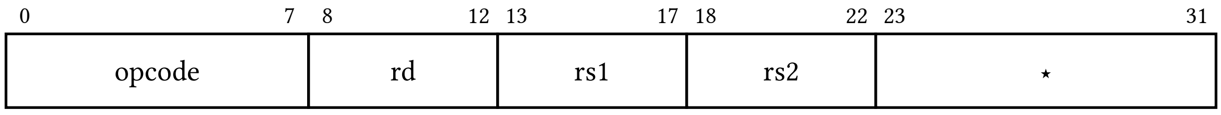
Smart bit header

Show start bit of each bitbox with `bitheader: "smart"`.



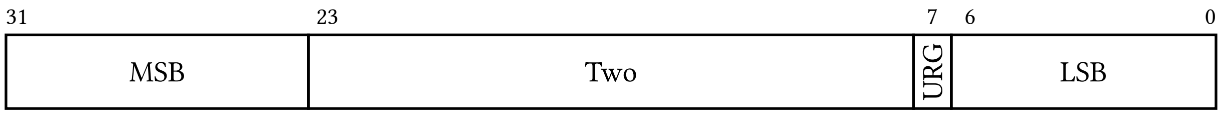
Bounds bit header

Show start bit of each bitbox with `bitheader: "bounds"`.



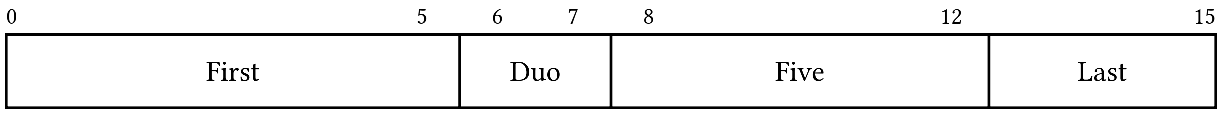
Reversed bit order

Select `msb_first: true` for a reversed bit order.

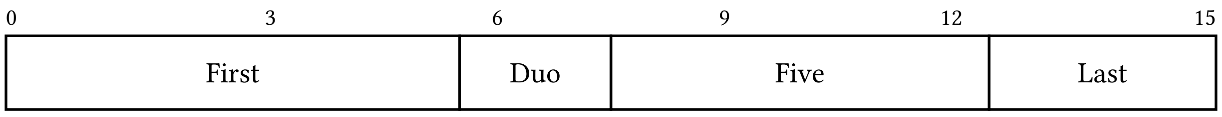


Custom bit header

Pass an `array` to specify each number.



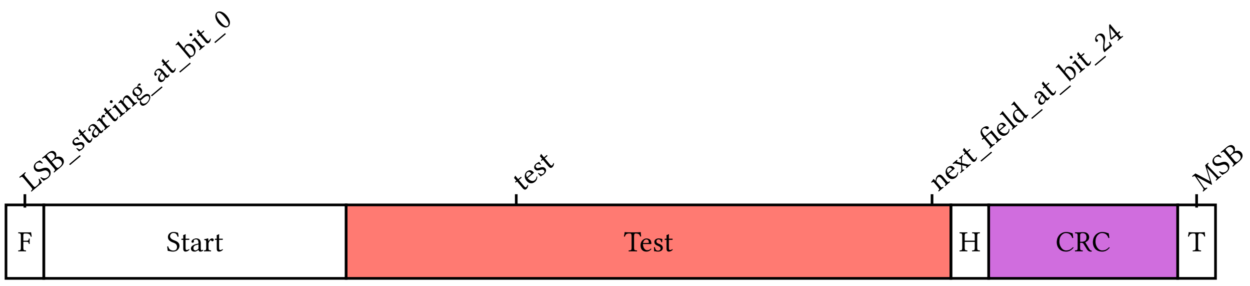
Pass an `integer` to show all multiples of this number.



Text header instead of numbers [WIP]

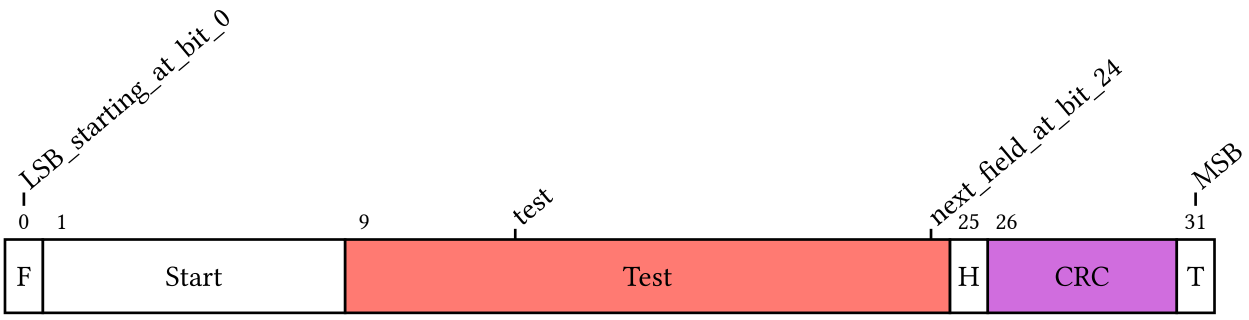
Pass an `dictionary` as bitheader. Example:

```
#bytefield(
  bitheader: (
    "0": "LSB_starting_at_bit_0",
    "13": "test",
    "24": "next_field_at_bit_24",
    "31":"MSB",
    angle: -40deg,
    marker: auto // or none
  ),
  bits: 32,
  bit[F],
  byte[Start],
  bytes(2, fill: red.lighten(30%))[Test],
  bit[H],
  bits(5, fill: purple.lighten(40%))[CRC],
  bit[T],
)
```



You can also show labels and numbers

```
#bytefield(
  bitheader: (
  "0": "LSB_starting_at_bit_0",
  "13": "test",
  "24": "next_field_at_bit_24",
  "31":"MSB",
  numbers:"smart", // the numbers to show
  angle: -40deg,
  marker: auto // or none
),
  bits: 32,
  bit[F],
  byte[Start],
  bytes(2, fill: red.lighten(30%))[Test],
  bit[H],
  bits(5, fill: purple.lighten(40%))[CRC],
  bit[T],
)
```

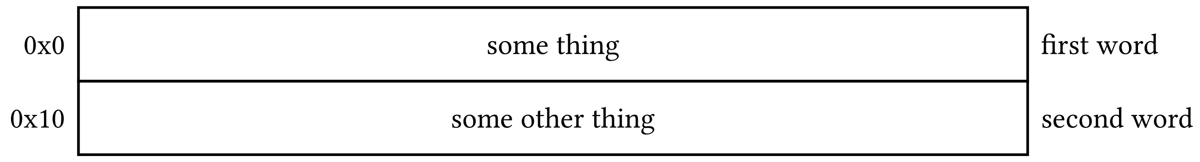


Pre/Post columns

Define additional columns with before the bitfield with `pre` or behind the bitfield with `post` and pass any tablex object.

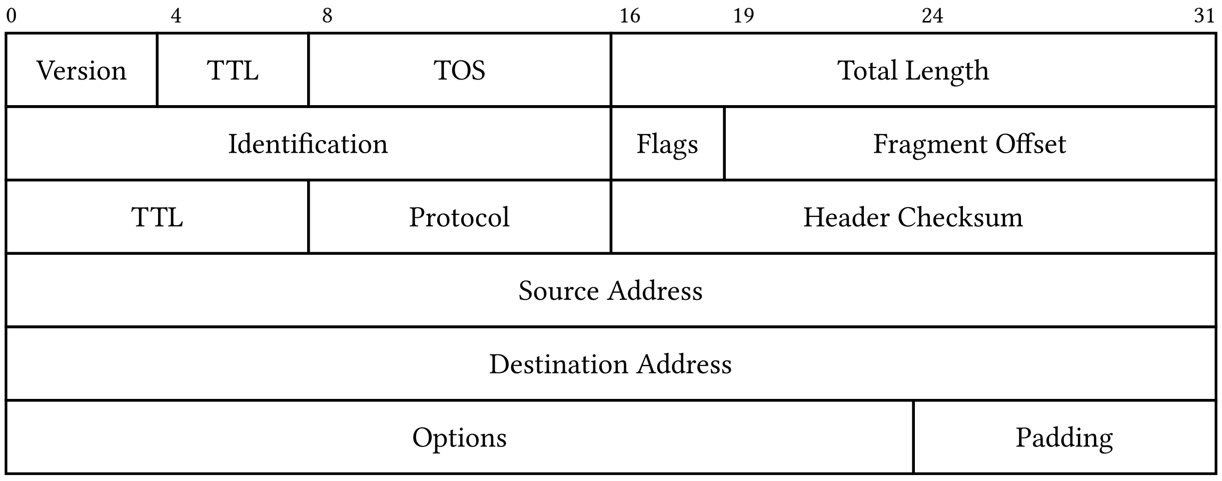
You can use the helpers `left_aligned` and `right_aligned` for left and right aligned text.

```
#bytefield(
  bits:1,
  pre:(auto,),
  post:(auto,),
  right_aligned[0x0],
  bit[some thing],
  left_aligned[first word],
  right_aligned[0x10],
  bit[some other thing],
  left_aligned[second word],
)
```

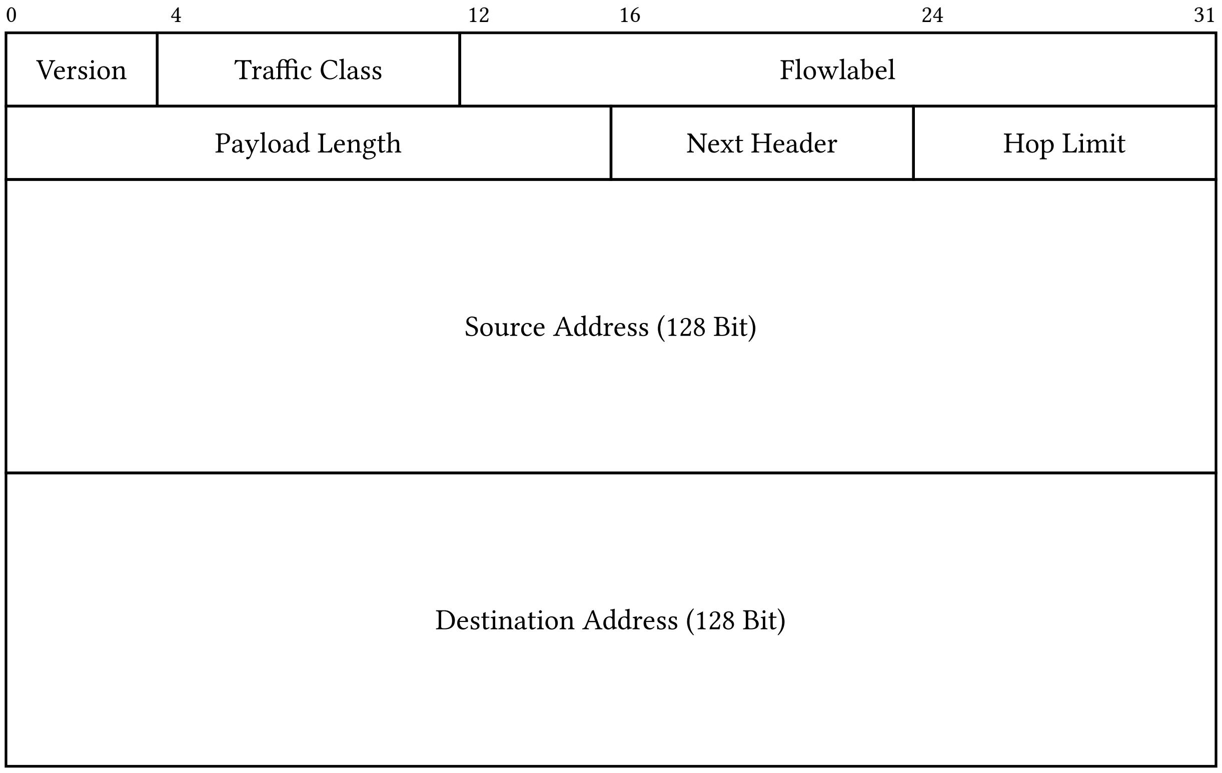


Some predefined network protocols

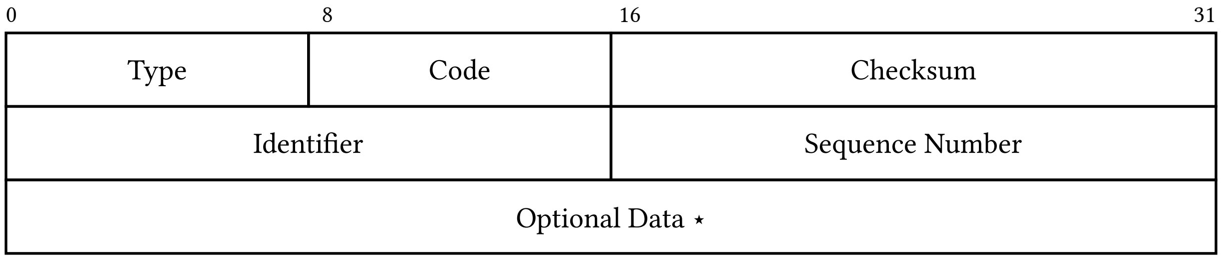
IPv4



IPv6



ICMP



ICMPv6

0	8	16	31
Type	Code	Checksum	
Internet Header + 64 bits of Original Data Datagram ⋆			

DNS

0	16	31
Identification	Flags	
Number of Questions	Number of answer RRs	
Number of authority RRs	Number of additional RRs	
Questions (64 Bit)		
Answers (variable number of resource records) (64 Bit)		
Authority (variable number of resource records) (64 Bit)		
Additional information (variable number of resource records) (64 Bit)		

TCP

0	4	10	16	24	31
Source Port			Destinatino Port		
Sequence Number					
Acknowledgment Number					
Data Offset	Reserved	Flags	Window		
Checksum			Urgent Pointer		
Options				Padding	
...DATA...⋆					

0	4	10	11	12	13	14	15	16	24	31
Source Port								Destinatino Port		
Sequence Number										
Acknowledgment Number										
Data Offset	Reserved	URG	ACK	PSH	RST	SYN	FIN	Window		
Checksum								Urgent Pointer		
Options									Padding	
...DATA...⋆										

UDP

0	16	31
Source Port	Destinatino Port	
Length	Checksum	
...DATA...⋆		