

## Zceb 0.53.2

This document is in the Stable state. Assume anything could still change, but limited change should be expected. For more information see: <https://riscv.org/spec-state>

This extension reuses encodings from the D-extension. Therefore it is *incompatible* with D. It is fully compatible with F and also with Zdinx.

The instructions are all 16-bit versions of existing 32-bit load/store instructions.

RV32	RV64	Mnemonic	Instruction
✓	✓	c.lbu $rd'$ , uimm( $rs1'$ )	Load unsigned byte, 16-bit encoding
✓	✓	c.lhu $rd'$ , uimm( $rs1'$ )	Load unsigned halfword, 16-bit encoding
✓	✓	c.lb $rd'$ , uimm( $rs1'$ )	Load signed byte, 16-bit encoding
✓	✓	c.lh $rd'$ , uimm( $rs1'$ )	Load signed halfword, 16-bit encoding
✓	✓	c.sb $rs2'$ , uimm( $rs1'$ )	Store byte, 16-bit encoding
✓	✓	c.sh $rs2'$ , uimm( $rs1'$ )	Store halfword, 16-bit encoding

# c.lbu

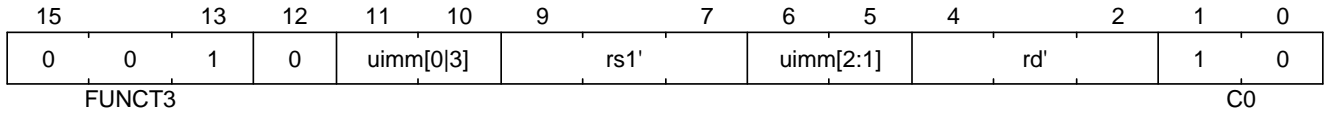
## Synopsis

Load unsigned byte, 16-bit encoding

## Mnemonic

c.lbu *rd'*, *uimm*(*rs1'*)

## Encoding (RV32, RV64)



The immediate offset is formed as follows:

```
uimm[31:4] = 0;
uimm[3]     = encoding[10];
uimm[2:1]   = encoding[6:5];
uimm[0]     = encoding[11];
```

## Description

This instruction loads a byte from the memory address formed by adding *rs1'* to the zero extended immediate *uimm*. The resulting byte is zero extended to XLEN bits and is written to *rd'*.

NOTE

*rd'* and *rs1'* are from the standard 8-register set x8-x15.

## Prerequisites

The C-extension. This encoding conflicts with the D-extension, but there is no conflict with Zdinx if double-precision arithmetic is required.

## 32-bit equivalent

[\[insns-lbu\]](#)

## Operation

```
//This is not SAIL, it's pseudo-code. The SAIL hasn't been written yet.

X(rdc) = EXTZ(mem[X(rs1c)+EXTZ(uimm)] [7..0]);
```

## Included in

Extension	Minimum version	Lifecycle state
Zceb ( <a href="#">Zceb 0.53.2</a> )	0.53.2	Stable

# c.lhu

## Synopsis

Load unsigned halfword, 16-bit encoding

## Mnemonic

`c.lhu rd', uimm(rs1')`

## Encoding (RV32, RV64)

15	13	12	11	10	9	7	6	5	4	2	1	0
0	0	1	1	uimm[4:3]		rs1'		uimm[2:1]		rd'	1	0
FUNCT3											C0	

The immediate offset is formed as follows:

```

uimm[31:5] = 0;
uimm[4:3]  = encoding[11:10];
uimm[2:1]  = encoding[6:5];
uimm[0]    = 0;

```

## Description

This instruction loads a halfword from the memory address formed by adding *rs1'* to the zero extended immediate *uimm*. The resulting halfword is zero extended to XLEN bits and is written to *rd'*.

**NOTE** *rd'* and *rs1'* are from the standard 8-register set x8-x15.

## Prerequisites

The C-extension. This encoding conflicts with the D-extension, but there is no conflict with Zdinx if double-precision arithmetic is required.

## 32-bit equivalent

[\[insns-lhu\]](#)

## Operation

```
//This is not SAIL, it's pseudo-code. The SAIL hasn't been written yet.
```

```
X(rdc) = EXTZ(load_mem[X(rs1c)+EXTZ(uimm)] [15..0]);
```

## Included in

Extension	Minimum version	Lifecycle state
Zceb ( <a href="#">Zceb 0.53.2</a> )	0.53.2	Stable

# c.lb

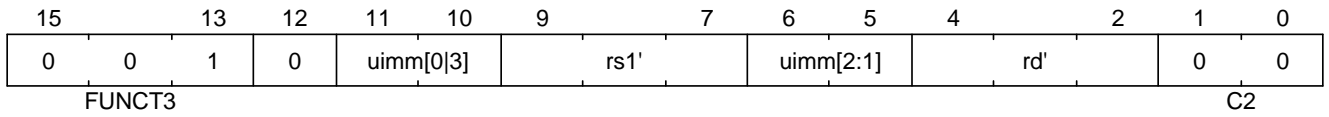
## Synopsis

Load signed byte, 16-bit encoding

## Mnemonic

c.lb *rd'*, *uimm(rs1')*

## Encoding (RV32, RV64)



The immediate offset is formed as follows:

```
uimm[31:4] = 0;
uimm[3]     = encoding[10];
uimm[2:1]   = encoding[6:5];
uimm[0]     = encoding[11];
```

## Description

This instruction loads a byte from the memory address formed by adding *rs1'* to the zero extended immediate *uimm*. The resulting byte is sign extended to XLEN bits and is written to *rd'*.

NOTE

*rd'* and *rs1'* are from the standard 8-register set x8-x15.

## Prerequisites

The C-extension. This encoding conflicts with the D-extension, but there is no conflict with Zdinx if double-precision arithmetic is required.

## 32-bit equivalent

[\[insns-lb\]](#)

## Operation

```
//This is not SAIL, it's pseudo-code. The SAIL hasn't been written yet.

X(rdc) = EXTS(mem[X(rs1c)+EXTZ(uimm)] [7..0]);
```

## Included in

Extension	Minimum version	Lifecycle state
Zceb ( <a href="#">Zceb 0.53.2</a> )	0.53.2	Stable

# c.lh

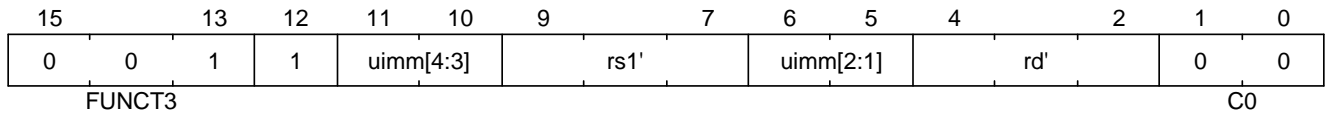
## Synopsis

Load signed halfword, 16-bit encoding

## Mnemonic

c.lh *rd'*, *uimm(rs1')*

## Encoding (RV32, RV64)



The immediate offset is formed as follows:

```
uimm[31:5] = 0;
uimm[4:3]   = encoding[11:10];
uimm[2:1]   = encoding[6:5];
uimm[0]     = 0;
```

## Description

This instruction loads a halfword from the memory address formed by adding *rs1'* to the zero extended immediate *uimm*. The resulting halfword is sign extended to XLEN bits and is written to *rd'*.

NOTE

*rd'* and *rs1'* are from the standard 8-register set x8-x15.

## Prerequisites

The C-extension. This encoding conflicts with the D-extension, but there is no conflict with Zdinx if double-precision arithmetic is required.

## 32-bit equivalent

[\[insns-lh\]](#)

## Operation

```
//This is not SAIL, it's pseudo-code. The SAIL hasn't been written yet.

X(rdc) = EXTS(load_mem[X(rs1c)+EXTZ(uimm)][15..0]);
```

## Included in

Extension	Minimum version	Lifecycle state
Zceb ( <a href="#">Zceb 0.53.2</a> )	0.53.2	Stable

# c.sb

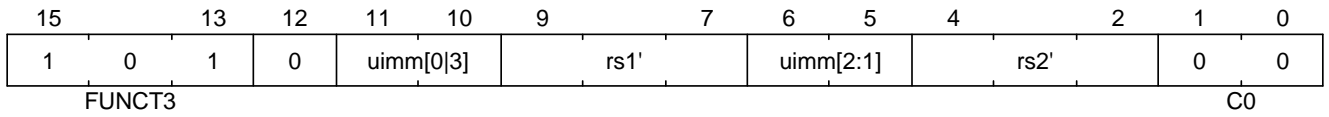
## Synopsis

Store byte, 16-bit encoding

## Mnemonic

c.sb *rd'*, *uimm(rs1')*

## Encoding (RV32, RV64)



The immediate offset is formed as follows:

```
uimm[31:4] = 0;
uimm[3]     = encoding[10];
uimm[2:1]   = encoding[6:5];
uimm[0]     = encoding[11];
```

## Description

This instruction stores the least significant byte of *rs2'* to the memory address formed by adding *rs1'* to the zero extended immediate *uimm*.

NOTE

*rd'* and *rs1'* are from the standard 8-register set x8-x15.

## Prerequisites

The C-extension. This encoding conflicts with the D-extension, but there is no conflict with Zdinx if double-precision arithmetic is required.

## 32-bit equivalent

[\[insns-sb\]](#)

## Operation

```
//This is not SAIL, it's pseudo-code. The SAIL hasn't been written yet.

mem[X(rs1c)+EXTZ(uimm)][7..0] = X(rs2c)
```

## Included in

Extension	Minimum version	Lifecycle state
Zceb ( <a href="#">Zceb 0.53.2</a> )	0.53.2	Stable

# c.sh

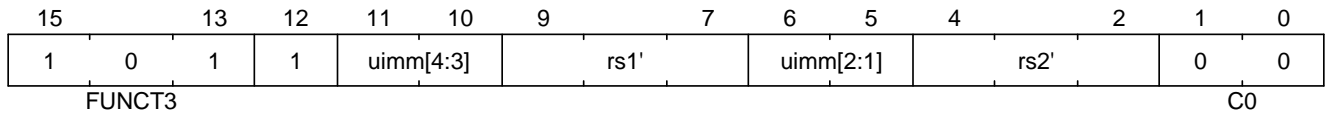
## Synopsis

Store halfword, 16-bit encoding

## Mnemonic

c.sh *rs2'*, *uimm(rs1')*

## Encoding (RV32, RV64)



The immediate offset is formed as follows:

```
uimm[31:5] = 0;
uimm[4:3]   = encoding[11:10];
uimm[2:1]   = encoding[6:5];
uimm[0]     = 0;
```

## Description

This instruction stores the least significant halfword of *rs2'* to the memory address formed by adding *rs1'* to the zero extended immediate *uimm*.

NOTE

*rd'* and *rs1'* are from the standard 8-register set x8-x15.

## Prerequisites

The C-extension. This encoding conflicts with the D-extension, but there is no conflict with Zdinx if double-precision arithmetic is required.

## 32-bit equivalent

[\[insns-sh\]](#)

## Operation

```
//This is not SAIL, it's pseudo-code. The SAIL hasn't been written yet.

mem[X(rs1c)+EXTZ(uimm)][15..0] = X(rs2c)
```

## Included in

Extension	Minimum version	Lifecycle state
Zceb ( <a href="#">Zceb 0.53.2</a> )	0.53.2	Stable