

LDUMINB, LDUMINAB, LDUMINALB, LDUMINLB

Atomic unsigned minimum on byte in memory atomically loads an 8-bit byte from memory, compares it against the value held in a register, and stores the smaller value back to memory, treating the values as unsigned numbers. The value initially loaded from memory is returned in the destination register.

- If the destination register is not WZR, LDUMINAB and LDUMINALB load from memory with acquire semantics.
- LDUMINLB and LDUMINALB store to memory with release semantics.
- LDUMINB has neither acquire nor release semantics.

For more information about memory ordering semantics, see [Load-Acquire, Store-Release](#).

For information about memory accesses, see [Load/Store addressing modes](#). This instruction is used by the alias [STUMINB, STUMINLB](#).

Integer (FEAT_LSE)

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
0	0	1	1	1	0	0	0	A	R	1						0	1	1	1	0	0										
size											opc																				
Rs									Rn									Rt													

LDUMINAB (A == 1 && R == 0)

LDUMINAB <Ws>, <Wt>, [<Xn|SP>]

LDUMINALB (A == 1 && R == 1)

LDUMINALB <Ws>, <Wt>, [<Xn|SP>]

LDUMINB (A == 0 && R == 0)

LDUMINB <Ws>, <Wt>, [<Xn|SP>]

LDUMINLB (A == 0 && R == 1)

LDUMINLB <Ws>, <Wt>, [<Xn|SP>]

```
if !IsFeatureImplemented(FEAT_LSE) then UNDEFINED;

integer t = UInt(Rt);
integer n = UInt(Rn);
integer s = UInt(Rs);

boolean acquire = A == '1' && Rt != '11111';
boolean release = R == '1';
boolean tagchecked = n != 31;
```

Assembler Symbols

- <Ws> Is the 32-bit name of the general-purpose register holding the data value to be operated on with the contents of the memory location, encoded in the "Rs" field.
- <Wt> Is the 32-bit name of the general-purpose register to be loaded, encoded in the "Rt" field.
- <Xn|SP> Is the 64-bit name of the general-purpose base register or stack pointer, encoded in the "Rn" field.

Alias Conditions

Alias	Is preferred when
STUMINB, STUMINLB	A == '0' && Rt == '11111'

Operation

```
bits(64) address;
bits(8) value;
bits(8) data;

AccessDescriptor accdesc = CreateAccDescAtomicOp(MemAtomicOp_UMIN, acqu

value = X[s, 8];
if n == 31 then
    CheckSPAlignment();
    address = SP[];
else
    address = X[n, 64];

bits(8) comparevalue = bits(8) UNKNOWN; // Irrelevant when not execu
data = MemAtomic(address, comparevalue, value, accdesc);

if t != 31 then
    X[t, 32] = ZeroExtend(data, 32);
```

Operational information

If PSTATE.DIT is 1, the timing of this instruction is insensitive to the value of the data being loaded or stored.

Internal version only: isa v33.64, AdvSIMD v29.12, pseudocode
no_diffs_2023_09_RC2, sve v2023-06_rel ; Build timestamp: 2023-09-18T17:56

Copyright © 2010-2023 Arm Limited or its affiliates. All rights reserved. This
document is Non-Confidential.