LDARH

Load-Acquire Register Halfword derives an address from a base register value, loads a halfword from memory, zero-extends it, and writes it to a register. The instruction also has memory ordering semantics as described in *Load-Acquire, Store-Release*. For information about memory accesses, see *Load/Store addressing modes*.

Note

For this instruction, if the destination is WZR/XZR, it is impossible for software to observe the presence of the acquire semantic other than its effect on the arrival at endpoints.

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 1	0 0 1 0 0 0	1	1	0	(1)(1)(1)(1)(1)	1	(1)(1)(1)(1)(1)	Rn	Rt
size			L		Rs	o0	Rt2		

```
LDARH <Wt>, [<Xn|SP>{,#0}]
```

```
integer n = UInt(Rn);
integer t = UInt(Rt);

boolean tagchecked = n != 31;
```

Assembler Symbols

<Wt> Is the 32-bit name of the general-purpose register to be transferred, 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.

Operation

```
bits(64) address;
bits(16) data;

AccessDescriptor accdesc;
accdesc = CreateAccDescAcqRel(MemOp_LOAD, tagchecked);

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

data = Mem[address, 2, accdesc];
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.

Base Instructions SIMD&FP Instructions SVE Instructions SME Instructions Index by Encoding Sh Pseu

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