

LD1Q

Contiguous load of quadwords to 128-bit element ZA tile slice

The slice number in the tile is selected by the slice index register, modulo the number of 128-bit elements in a Streaming SVE vector. The memory address is generated by scalar base and optional scalar offset which is multiplied by 16 and added to the base address. Inactive elements will not cause a read from Device memory or signal a fault, and are set to zero in the destination vector.

SME (FEAT_SME)

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
1	1	1	0	0	0	0	1	1	1	0	Rm			V	Rs	Pg			Rn			0	ZAt								

LD1Q { <ZAt><HV>.Q[<Ws>, <offs>] }, <Pg>/Z, [<Xn|SP>{, <Xm>, LSL #4}]

```
if !HaveSME() then UNDEFINED;
integer n = UInt(Rn);
integer m = UInt(Rm);
integer g = UInt('0':Pg);
integer s = UInt('011':Rs);
integer t = UInt(ZAt);
integer offset = 0;
constant integer esize = 128;
boolean vertical = V == '1';
```

Assembler Symbols

<ZAt> Is the name of the ZA tile ZA0-ZA15 to be accessed, encoded in the "ZAt" field.

<HV> Is the horizontal or vertical slice indicator, encoded in "V":

V	<HV>
0	H
1	V

<Ws> Is the 32-bit name of the slice index register W12-W15, encoded in the "Rs" field.

<offs> Is the slice index offset 0.

<Pg> Is the name of the governing scalable predicate register P0-P7, encoded in the "Pg" field.

<Xn|SP> Is the 64-bit name of the general-purpose base register or stack pointer, encoded in the "Rn" field.

<Xm> Is the optional 64-bit name of the general-purpose offset register, defaulting to XZR, encoded in the "Rm" field.

Operation

```

CheckStreamingSVEAndZAEnabled\(\);
constant integer VL = CurrentVL;
constant integer PL = VL DIV 8;
constant integer dim = VL DIV esize;
bits(64) base;
bits(64) addr;
bits(PL) mask = P[g, PL];
bits(64) moffs = X[m, 64];
bits(32) index = X[s, 32];
integer slice = (UInt(index) + offset) MOD dim;
bits(VL) result;
constant integer mbytes = esize DIV 8;
boolean contiguous = TRUE;
boolean nontemporal = FALSE;
boolean tagchecked = TRUE;
AccessDescriptor accdesc = CreateAccDescSME(MemOp\_LOAD, nontemporal, co

if n == 31 then
    if AnyActiveElement(mask, esize) ||
        ConstrainUnpredictableBool(Unpredictable\_CHECKSPNONEACTIVE) t
        CheckSPAlignment();
    base = SP[];
else
    base = X[n, 64];

for e = 0 to dim - 1
    addr = base + UInt(moffs) * mbytes;
    if ActivePredicateElement(mask, e, esize) then
        Elem[result, e, esize] = Mem[addr, mbytes, accdesc];
    else
        Elem[result, e, esize] = Zeros(esize);
    moffs = moffs + 1;

ZAslice[t, esize, vertical, slice, VL] = result;

```

Operational information

If PSTATE.DIT is 1, the timing of this instruction is insensitive to the value of the data being loaded or stored when its governing predicate register contains the same value for each execution.

[Base
Instructions](#)

[SIMD&FP
Instructions](#)

[SVE
Instructions](#)

[SME
Instructions](#)

[Index by
Encoding](#)

[Sh
Pseud](#)

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