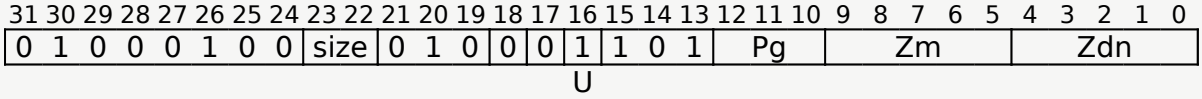
ADDP

Add pairwise

Add pairs of adjacent elements within each source vector, and interleave the results from corresponding lanes. The interleaved result values are destructively placed in the first source vector.



ADDP <Zdn>.<T>, <Pg>/M, <Zdn>.<T>, <Zm>.<T>

```
if !HaveSVE2() && !HaveSME() then UNDEFINED;
constant integer esize = 8 << UInt(size);
integer g = UInt(Pg);
integer m = UInt(Zm);
integer dn = UInt(Zdn);
```

Assembler Symbols

<Zdn> Is the name of the first source and destination scalable vector register, encoded in the "Zdn" field.

<T> Is the size specifier, encoded in “size”:

size	<T>
00	B
01	H
10	S
11	D

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

<Zm> Is the name of the second source scalable vector register, encoded in the "Zm" field.

Operation

```
CheckSVEEnabled();
constant integer VL = CurrentVL;
constant integer PL = VL DIV 8;
constant integer elements = VL DIV esize;
bits(PL) mask = P[g, PL];
bits(VL) operand1 = Z[dn, VL];
bits(VL) operand2 = if AnyActiveElement(mask, esize) then Z[m, VL] else
bits(VL) result;
integer element1;
```

```
integer element2;

for e = 0 to elements-1
    if !ActivePredicateElement(mask, e, esize) then
        Elem[result, e, esize] = Elem[operand1, e, esize];
    else
        if IsEven(e) then
            element1 = UInt(Elem[operand1, e + 0, esize]);
            element2 = UInt(Elem[operand1, e + 1, esize]);
        else
            element1 = UInt(Elem[operand2, e - 1, esize]);
            element2 = UInt(Elem[operand2, e + 0, esize]);
        integer res = element1 + element2;
        Elem[result, e, esize] = res<esize-1:0>;

Z[dn, VL] = result;
```

Operational information

If FEAT_SVE2 is implemented or FEAT_SME is implemented, then if PSTATE.DIT is 1:

- The execution time of this instruction is independent of:
  - The values of the data supplied in any of its operand registers when its governing predicate register contains the same value for each execution.
  - The values of the NZCV flags.
- The response of this instruction to asynchronous exceptions does not vary based on:
  - The values of the data supplied in any of its operand registers when its governing predicate register contains the same value for each execution.
  - The values of the NZCV flags.

This instruction might be immediately preceded in program order by a `MOVPRFX` instruction. The `MOVPRFX` instruction must conform to all of the following requirements, otherwise the behavior of the `MOVPRFX` and this instruction is unpredictable:

- The `MOVPRFX` instruction must be unpredicated.
- The `MOVPRFX` instruction must specify the same destination register as this instruction.
- The destination register must not refer to architectural register state referenced by any other source operand register of this instruction.

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.