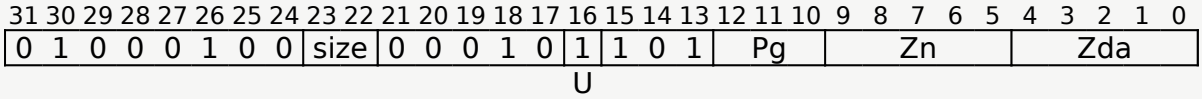
UADALP

Unsigned add and accumulate long pairwise

Add pairs of adjacent unsigned integer values and accumulate the results into the overlapping double-width elements of the destination vector.



UADALP <Zda>.<T>, <Pg>/M, <Zn>.<Tb>

```
if !HaveSVE2() && !HaveSME() then UNDEFINED;
if size == '00' then UNDEFINED;
constant integer esize = 8 << UInt(size);
integer g = UInt(Pg);
integer n = UInt(Zn);
integer da = UInt(Zda);
```

Assembler Symbols

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

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

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

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

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

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

size	<Tb>
00	RESERVED
01	B
10	H
11	S

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) operand_acc = Z[da, VL];
bits(VL) operand_src = if AnyActiveElement(mask, esize) then Z[n, VL] e
bits(VL) result;

for e = 0 to elements-1
    if !ActivePredicateElement(mask, e, esize) then
        Elem[result, e, esize] = Elem[operand_acc, e, esize];
    else
        integer element1 = UInt(Elem[operand_src, 2*e + 0, esize DIV 2]
        integer element2 = UInt(Elem[operand_src, 2*e + 1, esize DIV 2]
        bits(esize) sum = (element1 + element2)<esize-1:0>;
        Elem[result, e, esize] = Elem[operand_acc, e, esize] + sum;

Z[da, 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, or be predicated using the same governing predicate register and source element size as this instruction.
- 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.