

ADDHA

Add horizontally vector elements to ZA tile

Add each element of the source vector to the corresponding active element of each horizontal slice of a ZA tile. The tile elements are predicated by a pair of governing predicates. An element of a horizontal slice is considered active if its corresponding element in the second governing predicate is TRUE and the element corresponding to its horizontal slice number in the first governing predicate is TRUE. Inactive elements in the destination tile remain unmodified.

ID_AA64SMFR0_EL1.I16I64 indicates whether the 64-bit integer variant is implemented.

It has encodings from 2 classes: [32-bit](#) and [64-bit](#)

32-bit
(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	0	0	0	0	0	0	1	0	0	1	0	0	0	0	Pm		Pn							Zn		0	0	0	ZAda	
V																															

ADDHA <ZAda>.S, <Pn>/M, <Pm>/M, <Zn>.S

```
if !HaveSME() then UNDEFINED;
constant integer esize = 32;
integer a = UInt(Pn);
integer b = UInt(Pm);
integer n = UInt(Zn);
integer da = UInt(ZAda);
```

64-bit
(FEAT_SME_I16I64)

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	0	0	0	0	0	0	1	1	0	1	0	0	0	0	Pm		Pn							Zn		0	0	ZAda		
V																															

ADDHA <ZAda>.D, <Pn>/M, <Pm>/M, <Zn>.D

```
if !HaveSMEI16I64() then UNDEFINED;
constant integer esize = 64;
integer a = UInt(Pn);
integer b = UInt(Pm);
integer n = UInt(Zn);
integer da = UInt(ZAda);
```

Assembler Symbols

<ZAda>	For the 32-bit variant: is the name of the ZA tile ZA0-ZA3, encoded in the "ZAda" field. For the 64-bit variant: is the name of the ZA tile ZA0-ZA7, encoded in the "ZAda" field.
<Pn>	Is the name of the first governing scalable predicate register P0-P7, encoded in the "Pn" field.
<Pm>	Is the name of the second governing scalable predicate register P0-P7, encoded in the "Pm" field.
<Zn>	Is the name of the source scalable vector register, encoded in the "Zn" field.

Operation

```
CheckStreamingSVEAndZAAEnabled();  
constant integer VL = CurrentVL;  
constant integer PL = VL DIV 8;  
constant integer dim = VL DIV esize;  
bits(PL) mask1 = P[a, PL];  
bits(PL) mask2 = P[b, PL];  
bits(VL) operand_src = Z[n, VL];  
bits(dim*dim*esize) operand_acc = ZAtile[da, esize, dim*dim*esize];  
bits(dim*dim*esize) result;  
  
for col = 0 to dim-1  
  bits(esize) element = Elem[operand_src, col, esize];  
  for row = 0 to dim-1  
    bits(esize) res = Elem[operand_acc, row*dim+col, esize];  
    if (ActivePredicateElement(mask1, row, esize) &&  
        ActivePredicateElement(mask2, col, esize)) then  
      res = res + element;  
      Elem[result, row*dim+col, esize] = res;  
  
ZAtile[da, esize, dim*dim*esize] = result;
```

Operational information

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 registers contain 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 registers contain the same value for each execution.
 - The values of the NZCV flags.

[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

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