UUNPK

Unpack and zero-extend multi-vector elements

Unpack elements from one or two source vectors and then zero-extend them to place in elements of twice their size within the two or four destination vectors.

This instruction is unpredicated.

It has encodings from 2 classes: Two registers and Four registers

Two registers
(FEAT_SME2)

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	1	size		1	0	0	1	0	1	1	1	1	0	0	0	Zn					Zd				1
																															U

```
UUNPK { <Zd1>.<T>-<Zd2>.<T> }, <Zn>.<Tb>
```

```
if !HaveSME2() then UNDEFINED;
if size == '00' then UNDEFINED;
constant integer esize = 8 << UInt(size);
integer n = UInt(Zn);
integer d = UInt(Zd:'0');
constant integer nreg = 2;
boolean unsigned = TRUE;
```

Four registers
(FEAT_SME2)

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	1	size		1	1	0	1	0	1	1	1	1	0	0	0	Zn				0	Zd			0	1
																															U

```
UUNPK { <Zd1>.<T>-<Zd4>.<T> }, { <Zn1>.<Tb>-<Zn2>.<Tb> }
```

```
if !HaveSME2() then UNDEFINED;
if size == '00' then UNDEFINED;
constant integer esize = 8 << UInt(size);
integer n = UInt(Zn:'0');
integer d = UInt(Zd:'00');
constant integer nreg = 4;
boolean unsigned = TRUE;
```

Assembler Symbols

<Zd1>	For the two registers variant: is the name of the first destination scalable vector register of a multi-vector sequence, encoded as "Zd" times 2.

For the four registers variant: is the name of the first destination scalable vector register of a multi-vector sequence, encoded as "Zd" times 4.

<T>

Is the size specifier, encoded in “size”:

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

<Zd4> Is the name of the fourth destination scalable vector register of a multi-vector sequence, encoded as "Zd" times 4 plus 3.

<Zn1> Is the name of the first scalable vector register of a multi-vector sequence, encoded as "Zn" times 2.

<Zd2> Is the name of the second destination scalable vector register of a multi-vector sequence, encoded as "Zd" times 2 plus 1.

<Zn> Is the name of the 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

<Zn2> Is the name of the second scalable vector register of a multi-vector sequence, encoded as "Zn" times 2 plus 1.

Operation

```
CheckStreamingSVEEnabled();
constant integer VL = CurrentVL;
constant integer elements = VL DIV esize;
constant integer hsize = esize DIV 2;
constant integer sreg = nreg DIV 2;
array [0..3] of bits(VL) results;

for r = 0 to sreg-1
    bits(VL) operand = Z[n+r, VL];
    for i = 0 to 1
        for e = 0 to elements-1
            bits(hsize) element = Elem[operand, i*elements + e, hsize];
            Elem[results[2*r+i], e, esize] = Extend(element, esize, uns
for r = 0 to nreg-1
    Z[d+r, VL] = results[r];
```

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 registers.
    - 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 registers.
    - The values of the NZCV flags.

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