PMOV (to vector)

Move predicate to vector

Copy the source SVE predicate register elements into the destination vector register as a packed bitmap with one bit per predicate element, where bit value 0b1 represents a TRUE predicate element, and bit value 0b0 represents a FALSE predicate element.
Because the number of bits in an SVE predicate element scales with the the vector element size, the behavior varies according to the specified element size.

- When the predicate element specifier is.B, every bit in the predicate register is copied to the least-significant VL/8 bits of the destination vector register. The immediate index, if specified, must be 0.
- When the predicate element specifier is.H, every second bit in the predicate register is copied to the indexed block of VL/16 bits in the destination vector register, where the immediate index is in the range 0 to 1, inclusive.
- When the predicate element specifier is.S, every fourth bit in the predicate register is copied to the indexed block of VL/32 bits in the destination vector register, where the immediate index is in the range 0 to 3, inclusive.
- When the predicate element specifier is.D, every eighth bit in the predicate register is copied to the indexed block of VL/64 bits in the destination vector register, where the immediate index is in the range 0 to 7, inclusive.

The immediate index is optional, defaulting to 0 if omitted. When the index is zero, the instruction writes zeroes to the most significant VL-(VL/esize) bits of the destination vector register. When a non-zero index is specified, the packed bitmap is inserted into the destination vector register, and the unindexed blocks remain unchanged.
It has encodings from 4 classes: Byte , Doubleword , Halfword and Word

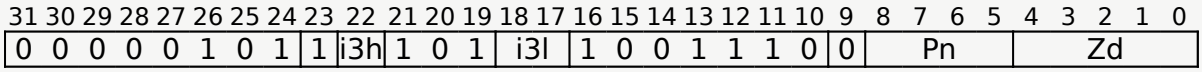
Byte
(FEAT_SVE2p1)

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
0	0	0	0	0	1	0	1	0	0	1	0	1	0	1	1	0	0	1	1	1	0	0	Pn				Zd				

```
PMOV <Zd>, <Pn>.B
```

```
if !HaveSVE2p1() && !HaveSME2p1() then UNDEFINED;
integer n = UInt(Pn);
integer d = UInt(Zd);
constant integer esize = 8;
constant integer imm = 0;
```

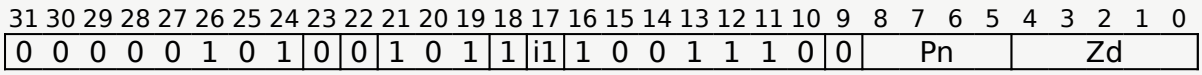
Doubleword
(FEAT_SVE2p1)



PMOV <Zd>[<imm>], <Pn>.D

```
if !HaveSVE2p1() && !HaveSME2p1() then UNDEFINED;
integer n = UInt(Pn);
integer d = UInt(Zd);
constant integer esize = 64;
constant integer imm = UInt(i3h:i3l);
```

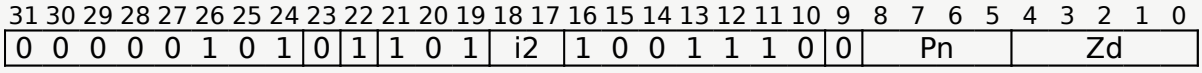
Halfword
(FEAT_SVE2p1)



PMOV <Zd>[<imm>], <Pn>.H

```
if !HaveSVE2p1() && !HaveSME2p1() then UNDEFINED;
integer n = UInt(Pn);
integer d = UInt(Zd);
constant integer esize = 16;
constant integer imm = UInt(i1);
```

Word
(FEAT_SVE2p1)



PMOV <Zd>[<imm>], <Pn>.S

```
if !HaveSVE2p1() && !HaveSME2p1() then UNDEFINED;
integer n = UInt(Pn);
integer d = UInt(Zd);
constant integer esize = 32;
constant integer imm = UInt(i2);
```

Assembler Symbols

<Zd>	Is the name of the destination scalable vector register, encoded in the "Zd" field.
<imm>	For the doubleword variant: is the element index, in the range 0 to 7, encoded in the "i3h:i3l" fields. For the halfword variant: is the element index, in the range 0 to 1, encoded in the "i1" field. For the word variant: is the element index, in the range 0 to 3, encoded in the "i2" field.

<Pn> Is the name of the source scalable predicate register, encoded in the "Pn" field.

Operation

```
CheckSVEEnabled();
constant integer VL = CurrentVL;
constant integer PL = VL DIV 8;
constant integer elements = VL DIV esize;
bits(PL) operand = P[n, PL];
bits(VL) result;

if imm == 0 then
    result = Zeros(VL);
else
    result = Z[d, VL];

for e = 0 to elements-1
    result<(elements * imm) + e> = PredicateElement(operand, e, esize);

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


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