PMOV (to predicate)

Move predicate from vector

Copy a packed bitmap, where bit value 0b1 represents TRUE and bit value 0b0 represents FALSE, from part of a source vector register to elements of a destination SVE predicate register.

Because the number of bits in an SVE predicate element scales with the vector element size, the behavior varies according to the specified element size.

- When the predicate element specifier is.B, each bit [N] from the least-significant VL/8 bits in the source vector register is copied to bit [N] of the destination predicate register. The immediate index, if specified, must be 0.
- When the predicate element specifier is.H, each bit [N] within the indexed block of VL/16 bits in the source vector register is copied to bit [N*2] of the destination predicate register, and the other bits in the predicate are set to zero. The immediate index is in the range 0 to 1, inclusive.
- When the predicate elements specifier is.S, each bit [N] within the indexed block of VL/32 bits in the source vector register is copied to bit [N*4] of the destination predicate register, and the other bits in the predicate are set to zero. The immediate index is in the range 0 to 3, inclusive.
- When the predicate element specifier is.D, each bit [N] within the indexed block of VL/64 bits in the source vector register is copied to bit [N*8] of the destination predicate register, and the other bits in the predicate are set to zero. The immediate index is in the range 0 to 7, inclusive.

The immediate index is optional, defaulting to 0 if omitted.

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	0	0	0	1	1	1	0	Zn					0	Pd			

```
PMOV <Pd>.B, <Zn>
```

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

Doubleword
(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	1	i3h	1	0	1	i3l		0	0	0	1	1	1	0	Zn					0	Pd			

PMOV <Pd>.D, <Zn>[<imm>]

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

Halfword
(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	1	i1	0	0	0	1	1	1	0	Zn					0	Pd			

PMOV <Pd>.H, <Zn>[<imm>]

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

Word
(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	1	1	0	1	i2		0	0	0	1	1	1	0	Zn					0	Pd			

PMOV <Pd>.S, <Zn>[<imm>]

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

Assembler Symbols

<Pd>

Is the name of the destination scalable predicate register, encoded in the "Pd" field.

<Zn>

Is the name of the source scalable vector register, encoded in the "Zn" 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.

Operation

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

for e = 0 to elements-1
    Elem[result, e, psize] = ZeroExtend(operand<(elements * imm) + e>,

P[d, PL] = result;
```

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