CPY (immediate, merging)

Copy signed integer immediate to vector elements (merging)

Copy a signed integer immediate to each active element in the destination vector. Inactive elements in the destination vector register remain unmodified.

The immediate operand is a signed value in the range -128 to +127, and for element widths of 16 bits or higher it may also be a signed multiple of 256 in the range -32768 to +32512 (excluding 0).

The immediate is encoded in 8 bits with an optional left shift by 8. The preferred disassembly when the shift option is specified is "#<simm8>, LSL #8". However an assembler and disassembler may also allow use of the shifted 16-bit value unless the immediate is 0 and the shift amount is 8, which must be unambiguously described as "#0, LSL #8".

This instruction is used by the alias MOV (immediate, predicated, merging).

This instruction is used by the pseudo-instruction FMOV (zero, predicated).

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	size		0	1	Pg				0	1	sh	imm8								Zd				
																	M														

```
CPY <Zd>.<T>, <Pg>/M, #<imm>{, <shift>}
```

```
if !HaveSVE() && !HaveSME() then UNDEFINED;
if size:sh == '001' then UNDEFINED;
constant integer esize = 8 << UInt(size);
integer g = UInt(Pg);
integer d = UInt(Zd);
boolean merging = TRUE;
integer imm = SInt(imm8);
if sh == '1' then imm = imm << 8;
```

Assembler Symbols

<Zd>

Is the name of the destination scalable vector register, encoded in the "Zd" field.

<T>

Is the size specifier, encoded in “size”:

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

<Pg>

Is the name of the governing scalable predicate register, encoded in the "Pg" field.

<imm>	Is a signed immediate in the range -128 to 127, encoded in the "imm8" field.
<shift>	Is the optional left shift to apply to the immediate, defaulting to LSL #0 and encoded in “sh”:

sh	<shift>
0	LSL #0
1	LSL #8

Alias Conditions

Alias	Is preferred when
FMOV (zero, predicated)	Never
MOV (immediate, predicated, merging)	Unconditionally

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) dest = Z[d, VL];
bits(VL) result;

for e = 0 to elements-1
    if ActivePredicateElement(mask, e, esize) then
        Elem[result, e, esize] = imm<esize-1:0>;
    elsif merging then
        Elem[result, e, esize] = Elem[dest, e, esize];
    else
        Elem[result, e, esize] = Zeros(esize);

Z[d, 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.

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