WHILEGT (predicate pair)

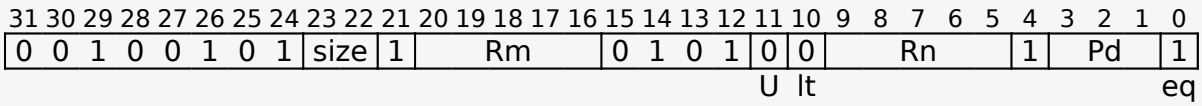
While decrementing signed scalar greater than scalar (pair of predicates)

Generate a pair of predicates that starting from the highest numbered element of the pair is true while the decrementing value of the first, signed scalar operand is greater than the second scalar operand and false thereafter down to the lowest numbered element of the pair.

The full width of the scalar operands is significant for the purposes of comparison, and the full width first operand is decremented by one for each destination predicate element, irrespective of the predicate result element size. The first general-purpose source register is not itself updated.

The lower-numbered elements are placed in the first predicate destination register, and the higher-numbered elements in the second predicate destination register. Sets the first (N), none (Z), !last (C) condition flags based on the predicate result, and the V flag to zero.

SVE2
(FEAT_SVE2p1)




```
WHILEGT { <Pd1>.<T>, <Pd2>.<T> }, <Xn>, <Xm>
```

```
if !HaveSME2() && !HaveSVE2p1() then UNDEFINED;
constant integer esize = 8 << UInt(size);
constant integer rsize = 64;
integer n = UInt(Rn);
integer m = UInt(Rm);
integer d0 = UInt(Pd:'0');
integer d1 = UInt(Pd:'1');
boolean unsigned = FALSE;
SVECmp op = Cmp_GT;
```

Assembler Symbols

<Pd1>	Is the name of the first destination scalable predicate register, encoded as "Pd" times 2.

<T>

Is the size specifier, encoded in “size”:

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

<Pd2> Is the name of the second destination scalable predicate register, encoded as "Pd" times 2 plus 1.

<Xn> Is the 64-bit name of the first source general-purpose register, encoded in the "Rn" field.

<Xm> Is the 64-bit name of the second source general-purpose register, encoded in the "Rm" field.

Operation

```
CheckSVEEnabled();
constant integer VL = CurrentVL;
constant integer PL = VL DIV 8;
constant integer elements = VL DIV esize;
bits(PL*2) mask = Ones(PL*2);
bits(rsize) operand1 = X[n, rsize];
bits(rsize) operand2 = X[m, rsize];
bits(PL*2) result;
boolean last = TRUE;
constant integer psize = esize DIV 8;

for e = (elements*2)-1 downto 0
    boolean cond;
    case op of
        when Cmp_GT cond = (Int(operand1, unsigned) >  Int(operand2, un
        when Cmp_GE cond = (Int(operand1, unsigned) >= Int(operand2, un

    last = last && cond;
    bit pbit = if last then '1' else '0';
    Elem[result, e, psize] = ZeroExtend(pbit, psize);
    operand1 = operand1 - 1;

PSTATE.<N,Z,C,V> = PredTest(mask, result, esize);
P[d0, PL] = result<PL-1:0>;
P[d1, PL] = result<PL*2-1:PL>;
```

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