UQINCP (scalar)

Unsigned saturating increment scalar by count of true predicate elements

Counts the number of true elements in the source predicate and then uses the result to increment the scalar destination. The result is saturated to the general-purpose register's unsigned integer range.

It has encodings from 2 classes: [32-bit](#) and [64-bit](#)

32-bit

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	1	0	0	1	0	1	size	1	0	1	0	0	1	1	0	0	0	1	0	0		Pm	Rdn

D U sf

        UQINCP **<Wdn>**, **<Pm>.<T>**

```
if !HaveSVE() && !HaveSME() then UNDEFINED;
constant integer esize = 8 << UInt(size);
integer m = UInt(Pm);
integer dn = UInt(Rdn);
boolean unsigned = TRUE;
constant integer ssize = 32;
```

64-bit

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	1	0	0	1	0	1	size	1	0	1	0	0	1	1	0	0	0	1	1	0		Pm	Rdn

D U sf

        UQINCP **<Xdn>**, **<Pm>.<T>**

```
if !HaveSVE() && !HaveSME() then UNDEFINED;
constant integer esize = 8 << UInt(size);
integer m = UInt(Pm);
integer dn = UInt(Rdn);
boolean unsigned = TRUE;
constant integer ssize = 64;
```

Assembler Symbols

<Wdn>   Is the 32-bit name of the source and destination general-purpose register, encoded in the "Rdn" field.

<Xdn>   Is the 64-bit name of the source and destination general-purpose register, encoded in the "Rdn" field.

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

<T>

Is the size specifier, encoded in "size":

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

Operation

```
CheckSVEEnabled();
constant integer VL = CurrentVL;
constant integer PL = VL DIV 8;
constant integer elements = VL DIV esize;
bits(ssize) operand1 = X[dn, ssize];
bits(PL) operand2 = P[m, PL];
bits(ssize) result;
integer count = 0;

for e = 0 to elements-1
    if ActivePredicateElement(operand2, e, esize) then
        count = count + 1;

integer element = Int(operand1, unsigned);
(result, -) = SatQ(element + count, ssize, unsigned);
X[dn, 64] = Extend(result, 64, unsigned);
```

Operational information

If FEAT_SME is implemented and the PE is in Streaming SVE mode, then any subsequent instruction which is dependent on the general-purpose register written by this instruction might be significantly delayed.