

## UQCVT (four registers)

Multi-vector unsigned saturating extract narrow

Saturate the unsigned integer value in each element of the four source vectors to quarter the original source element width, and place the results in the quarter-width destination elements.

This instruction is unpredicated.

### SME2

(FEAT\_SME2)

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
1	1	0	0	0	0	0	1	sz	0	1	1	0	0	1	1	1	1	1	0	0	0		Zn	0	1				Zd		
																								N		U					

**UQCVT** <Zd>.<T>, { <Zn1>.<Tb>-<Zn4>.<Tb> }

```
if !HaveSME2() then UNDEFINED;
constant integer esize = 8 << UInt(sz);
integer n = UInt(Zn:'00');
integer d = UInt(Zd);
```

### Assembler Symbols

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

<T> Is the size specifier, encoded in "sz":

sz	<T>
0	B
1	H

<Zn1> Is the name of the first scalable vector register of a multi-vector sequence, encoded as "Zn" times 4.

<Tb> Is the size specifier, encoded in "sz":

sz	<Tb>
0	S
1	D

<Zn4> Is the name of the fourth scalable vector register of a multi-vector sequence, encoded as "Zn" times 4 plus 3.

### Operation

```

CheckStreamingSVEEnabled();
constant integer VL = CurrentVL;
constant integer elements = VL DIV (4 * esize);
bits(VL) result;

for r = 0 to 3
    bits(VL) operand = Z[n+r, VL];
    for e = 0 to elements-1
        integer element = UInt(Elem[operand, e, 4 * esize]);
        Elem[result, r*elements + e, esize] = UnsignedSat(element, esize);

Z[d, VL] = result;

```

[Base  
Instructions](#)

[SIMD&FP  
Instructions](#)

[SVE  
Instructions](#)

[SME  
Instructions](#)

[Index by  
Encoding](#)

[Sh  
Pseudocode](#)

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

Copyright Â© 2010-2023 Arm Limited or its affiliates. All rights reserved. This  
document is Non-Confidential.