

## SQADD

Signed saturating Add. This instruction adds the values of corresponding elements of the two source SIMD&FP registers, places the results into a vector, and writes the vector to the destination SIMD&FP register.

If overflow occurs with any of the results, those results are saturated. If saturation occurs, the cumulative saturation bit *FPSR.QC* is set.

Depending on the settings in the *CPACR\_EL1*, *CPTR\_EL2*, and *CPTR\_EL3* registers, and the current Security state and Exception level, an attempt to execute the instruction might be trapped.

It has encodings from 2 classes: [Scalar](#) and [Vector](#)

### Scalar

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		1		0		1		1		1		1		0		size		1		Rm				0				0				0				1				1				Rn				Rd			
U																																																			

U

**SQADD** [<V><d>](#), [<V><n>](#), [<V><m>](#)

```
integer d = UInt(Rd);
integer n = UInt(Rn);
integer m = UInt(Rm);
constant integer esize = 8 << UInt(size);
constant integer datasize = esize;
integer elements = 1;
boolean unsigned = (U == '1');
```

### Vector

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	Q	0	0	1	1	1	0	size	1	Rm			0	0	0	0	1	1	Rn			Rd									
U																															

U

**SQADD** [<Vd>.<T>](#), [<Vn>.<T>](#), [<Vm>.<T>](#)

```
integer d = UInt(Rd);
integer n = UInt(Rn);
integer m = UInt(Rm);
if size:Q == '110' then UNDEFINED;
constant integer esize = 8 << UInt(size);
constant integer datasize = 64 << UInt(Q);
integer elements = datasize DIV esize;
boolean unsigned = (U == '1');
```

## Assembler Symbols

<V>

Is a width specifier, encoded in "size":

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

<d>

Is the number of the SIMD&FP destination register, in the "Rd" field.

<n>

Is the number of the first SIMD&FP source register, encoded in the "Rn" field.

<m>

Is the number of the second SIMD&FP source register, encoded in the "Rm" field.

<Vd>

Is the name of the SIMD&FP destination register, encoded in the "Rd" field.

<T>

Is an arrangement specifier, encoded in "size:Q":

size	Q	<T>
00	0	8B
00	1	16B
01	0	4H
01	1	8H
10	0	2S
10	1	4S
11	0	RESERVED
11	1	2D

<Vn>

Is the name of the first SIMD&FP source register, encoded in the "Rn" field.

<Vm>

Is the name of the second SIMD&FP source register, encoded in the "Rm" field.

## Operation

```
CheckFPAdvSIMDEnabled64();
bits(datasize) operand1 = V[n, datasize];
bits(datasize) operand2 = V[m, datasize];
bits(datasize) result;
integer element1;
integer element2;
integer sum;
boolean sat;

for e = 0 to elements-1
```

```

element1 = Int(Elem[operand1, e, esize], unsigned);
element2 = Int(Elem[operand2, e, esize], unsigned);
sum = element1 + element2;
(Elem[result, e, esize], sat) = SatQ(sum, esize, unsigned);
if sat then FPSR.QC = '1';

```

```

V[d, datasize] = 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.