## **FADDP** (scalar)

Floating-point Add Pair of elements (scalar). This instruction adds two floating-point vector elements in the source SIMD&FP register and writes the scalar result into the destination SIMD&FP register.

This instruction can generate a floating-point exception. Depending on the settings in *FPCR*, the exception results in either a flag being set in *FPSR* or a synchronous exception being generated. For more information, see *Floating-point exception traps*.

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: <u>Half-precision</u> and <u>Single-precision</u> and <u>double-precision</u>

# Half-precision (FEAT\_FP16)

```
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 1 0 0 sz 1 1 0 0 0 0 1 1 0 1 1 0 Rn Rd
```

```
FADDP <V><d>, <Vn>.<T>
```

```
if !IsFeatureImplemented(FEAT_FP16) then UNDEFINED;
integer d = UInt(Rd);
integer n = UInt(Rn);
constant integer esize = 16;
if sz == '1' then UNDEFINED;
constant integer datasize = 32;
```

#### Single-precision and double-precision

```
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 1 1 1 1 1 1 0 0 0 sz 1 1 0 0 0 0 1 1 0 1 1 0 Rn Rd
```

```
FADDP <V><d>, <Vn>.<T>
```

```
integer d = <u>UInt</u>(Rd);
integer n = <u>UInt</u>(Rn);
constant integer esize = 32 << <u>UInt</u>(sz);
constant integer datasize = esize * 2;
```

### **Assembler Symbols**

<V>

For the half-precision variant: is the destination width specifier, encoded in "sz":

SZ	<v></v>
0	Н
1	RESERVED

For the single-precision and double-precision variant: is the destination width specifier, encoded in "sz":

SZ	<v></v>
0	S
1	D

<d>

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

<Vn>

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

<T>

For the half-precision variant: is the source arrangement specifier, encoded in "sz":

SZ	<t></t>
0	2H
1	RESERVED

For the single-precision and double-precision variant: is the source arrangement specifier, encoded in "sz":

SZ	<t></t>
0	2S
1	2D

#### **Operation**

```
CheckFPAdvSIMDEnabled64();
bits(datasize) operand = V[n, datasize];
V[d, esize] = Reduce(ReduceOp_FADD, operand, esize);
```

Base Instructions SIMD&FP Instructions SVE Instructions SME Instructions Index by Encoding

 $Internal\ version\ only: is a\ v33.64,\ AdvSIMD\ v29.12,\ pseudocode\ no\_diffs\_2023\_09\_RC2,\ sve\ v2023-06\_rel\ ;\ Build\ timestamp:\ 2023-09-18T17:56$ 

Sh Pseu

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