

FMIN (scalar)

Floating-point Minimum (scalar). This instruction compares the first and second source SIMD&FP register values, and writes the smaller of the two floating-point values to the destination SIMD&FP register.

When [FPCR.AH](#) is 0, the behavior is as follows:

- Negative zero compares less than positive zero.
- When [FPCR.DN](#) is 0, if either value is a NaN, the result is a quiet NaN.
- When [FPCR.DN](#) is 1, if either value is a NaN, the result is Default NaN.

When [FPCR.AH](#) is 1, the behavior is as follows:

- If both values are zeros, regardless of the sign of either zero, the result is the second value.
- If either value is a NaN, regardless of the value of [FPCR.DN](#), the result is the second value.

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.

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	0	1	1	1	1	0	f	t	y	p	e	1					R	m		0	1	0	1	1	0				R	n			R	d
op																																			

Half-precision (ftype == 11) (FEAT_FP16)

FMIN <Hd>, <Hn>, <Hm>

Single-precision (ftype == 00)

FMIN <Sd>, <Sn>, <Sm>

Double-precision (ftype == 01)

FMIN <Dd>, <Dn>, <Dm>

```
if ftype == '10' || (ftype == '11' && !IsFeatureImplemented(FEAT_FP16))
integer d = UInt(Rd);
integer n = UInt(Rn);
```

```
integer m = UInt(Rm);

constant integer esize = 8 << UInt(ftype EOR '10');
```

Assembler Symbols

<Dd>	Is the 64-bit name of the SIMD&FP destination register, encoded in the "Rd" field.
<Dn>	Is the 64-bit name of the first SIMD&FP source register, encoded in the "Rn" field.
<Dm>	Is the 64-bit name of the second SIMD&FP source register, encoded in the "Rm" field.
<Hd>	Is the 16-bit name of the SIMD&FP destination register, encoded in the "Rd" field.
<Hn>	Is the 16-bit name of the first SIMD&FP source register, encoded in the "Rn" field.
<Hm>	Is the 16-bit name of the second SIMD&FP source register, encoded in the "Rm" field.
<Sd>	Is the 32-bit name of the SIMD&FP destination register, encoded in the "Rd" field.
<Sn>	Is the 32-bit name of the first SIMD&FP source register, encoded in the "Rn" field.
<Sm>	Is the 32-bit name of the second SIMD&FP source register, encoded in the "Rm" field.

Operation

```
CheckFPEnabled64();
bits(esign) operand1 = V[n, esize];
bits(esign) operand2 = V[m, esize];

FPCRType fpcr = FPCR[];
boolean merge = IsMerging(fpcr);
bits(128) result = if merge then V[n, 128] else Zeros(128);

Elem[result, 0, esize] = FPMIn(operand1, operand2, fpcr);
V[d, 128] = result;
```

[Base
Instructions](#)

[SIMD&FP
Instructions](#)

[SVE
Instructions](#)

[SME
Instructions](#)

[Index by
Encoding](#)

[Sh
Pseud](#)

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