

## FMUL (vector)

Floating-point Multiply (vector). This instruction multiplies corresponding floating-point values in the vectors in the two source SIMD&FP registers, places the result in a vector, and writes the vector to 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: [Half-precision](#) and [Single-precision and double-precision](#)

### 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	Q	1	0	1	1	1	0	0	1	0	Rm				0	0	0	1	1	1	Rn				Rd						

**FMUL** <Vd>.<T>, <Vn>.<T>, <Vm>.<T>

```
if !IsFeatureImplemented(FEAT_FP16) then UNDEFINED;
```

```
integer d = UInt(Rd);
integer n = UInt(Rn);
integer m = UInt(Rm);
constant integer esize = 16;
constant integer datasize = 64 << UInt(Q);
integer elements = datasize DIV esize;
```

### 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	Q	1	0	1	1	1	0	0	sz	1	Rm				1	1	0	1	1	1	Rn				Rd						

**FMUL** <Vd>.<T>, <Vn>.<T>, <Vm>.<T>

```
integer d = UInt(Rd);
integer n = UInt(Rn);
integer m = UInt(Rm);
if sz:Q == '10' then UNDEFINED;
constant integer esize = 32 << UInt(sz);
constant integer datasize = 64 << UInt(Q);
integer elements = datasize DIV esize;
```

Assembler Symbols

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

<T> For the half-precision variant: is an arrangement specifier, encoded in "Q":

Q	<T>
0	4H
1	8H

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

sz	Q	<T>
0	0	2S
0	1	4S
1	0	RESERVED
1	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;
bits(esize) element1;
bits(esize) element2;

for e = 0 to elements-1
    element1 = Elem[operand1, e, esize];
    element2 = Elem[operand2, e, esize];
    Elem[result, e, esize] = FPMul(element1, element2, FPCR[]);

V[d, datasize] = result;
```

[Base Instructions](#)

[SIMD&FP Instructions](#)

[SVE Instructions](#)

[SME Instructions](#)

[Index by Encoding](#)

[Sh Pseudocode](#)