## BFDOT (vector)

BFloat16 floating-point dot product (vector). This instruction delimits the source vectors into pairs of BFloat16 elements.

If FEAT_EBF16 is not implemented or *FPCR*.EBF is 0, this instruction:

- Performs an unfused sum-of-products of each pair of adjacent BFloat16 elements in the source vectors. The intermediate single-precision products are rounded before they are summed, and the intermediate sum is rounded before accumulation into the single-precision destination element that overlaps with the corresponding pair of BFloat16 elements in the source vectors.
- Uses the non-IEEE 754 Round-to-Odd rounding mode, which forces bit 0 of an inexact result to 1, and rounds an overflow to an appropriately signed Infinity.
- Flushes denormalized inputs and results to zero, as if *FPCR*.{FZ, FIZ} is {1, 1}.
- Disables alternative floating point behaviors, as if *FPCR*.AH is 0.

If FEAT_EBF16 is implemented and *FPCR*.EBF is 1, then this instruction:

- Performs a fused sum-of-products of each pair of adjacent BFloat16 elements in the source vectors. The intermediate single-precision products are not rounded before they are summed, but the intermediate sum is rounded before accumulation into the single-precision destination element that overlaps with the corresponding pair of BFloat16 elements in the source vectors.
- Follows all other floating-point behaviors that apply to single-precision arithmetic, as governed by *FPCR*.RMode, *FPCR*.FZ, *FPCR*.AH, and *FPCR*.FIZ.

Irrespective of FEAT_EBF16 and *FPCR*.EBF, this instruction:

- Does not modify the cumulative *FPSR* exception bits (IDC, IXC, UFC, OFC, DZC, and IOC).
- Disables trapped floating-point exceptions, as if the *FPCR* trap enable bits (IDE, IXE, UFE, OFE, DZE, and IOE) are all zero.
- Generates only the default NaN, as if *FPCR*.DN is 1.

*ID_AA64ISAR1_EL1*.BF16 indicates whether this instruction is supported.

### Vector

**(FEAT_BF16)**

| 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 | | | | | 1 | 1 | 1 | 1 | 1 | 1 | Rn | | | | | Rd | | | | |

```
BFDOT <Vd>.<Ta>, <Vn>.<Tb>, <Vm>.<Tb>
```

```
if !IsFeatureImplemented(FEAT_BF16) then UNDEFINED;
integer n = UInt(Rn);
integer m = UInt(Rm);
integer d = UInt(Rd);
constant integer datasize = 64 << UInt(Q);
integer elements = datasize DIV 32;
```

### Assembler Symbols

| | |
|---|---|
| <Vd> | Is the name of the SIMD&FP destination register, encoded in the "Rd" field. |
| <Ta> | Is an arrangement specifier, encoded in “Q”: |

| Q | <Ta> |
|---|---|
| 0 | 2S |
| 1 | 4S |

| | |
|---|---|
| <Vn> | Is the name of the first SIMD&FP source register, encoded in the "Rn" field. |
| <Tb> | Is an arrangement specifier, encoded in “Q”: |

| Q | <Tb> |
|---|---|
| 0 | 4H |
| 1 | 8H |

| | |
|---|---|
| <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) operand3 = V[d, datasize];
bits(datasize) result;

for e = 0 to elements-1
    bits(16) elt1_a = Elem[operand1, 2*e+0, 16];
    bits(16) elt1_b = Elem[operand1, 2*e+1, 16];
    bits(16) elt2_a = Elem[operand2, 2*e+0, 16];
    bits(16) elt2_b = Elem[operand2, 2*e+1, 16];

    bits(32) sum = Elem[operand3, e, 32];
    sum = BFDotAdd(sum, elt1_a, elt1_b, elt2_a, elt2_b, FPCR[]);
    Elem[result, e, 32] = sum;

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

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