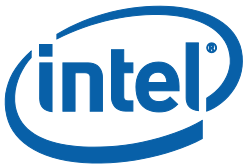


BFLOAT16 – Hardware Numerics Definition

White Paper

November 2018

Revision 1.0

Document Number: 338302-001US

Contents

Figures

No table of figures entries found.

Revision History

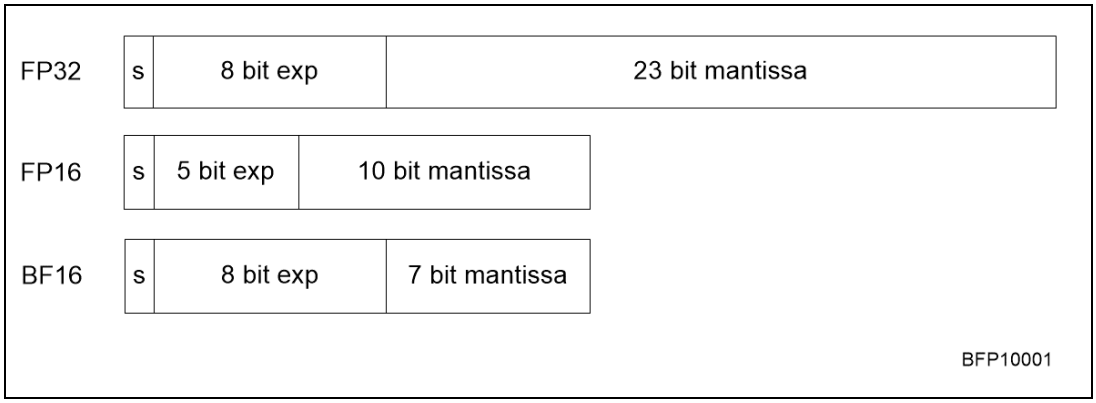
Document Number	Revision Number	Description	Date
338302-001	1.0	Initial release of the document.	November 2018

1 Intel® Deep Learning Boost - Training Numerics Proposal

1.1 Bfloat16 Floating-point Format

Intel® Deep Learning Boost (Intel® DL Boost) uses bfloat16 format (BF16). Figure 1-1 illustrates BF16 versus FP16 and FP32.

Figure 1-1. Comparison of BF16 to FP16 and FP32.



BF16 has several advantages over FP16:

- It can be seen as a short version of FP32, skipping the least significant 16 bits of mantissa.
- There is no need to support denormals; FP32, and therefore also BF16, offer more than enough range for deep learning training tasks.
- FP32 accumulation after the multiply is essential to achieve sufficient numerical behavior on an application level.
- Hardware exception handling is not needed as this is a performance optimization; industry is designing algorithms around checking inf/NaN.

1.2 Architectural Interface – Hardware Proposal for BF16 Units in Intel® Architecture

In order to use BF16 efficiently, it must be implemented in hardware in a unified way. The following sub-sections address the FMA unit with two BF16 input operands and one FP32 input/output operand, and conversions between FP32 and BF16.

1.2.1 FMA Unit

Based on the findings presented above we can propose a standardized unit which fulfills all required items:

- Treat denormal source as zero by default (only this mode is supported).
- Flush denormal results to zero by default (only this mode is supported).
- Set rounding to RNE by default (only this mode is supported).
- All exceptions/traps are masked by default (only this mode is supported).
- NaNs and Infinities are supported normally (with SNaNs getting masked response).
- The BF16*BF16 multiplication is performed without loss of precision; its result is passed to a general FP32 accumulator with the aforementioned settings.

In detail, the proposed multiplication-addition units are shown in Figure 1-2.

Figure 1-2. Proposed Fused Multiplication-Addition Units

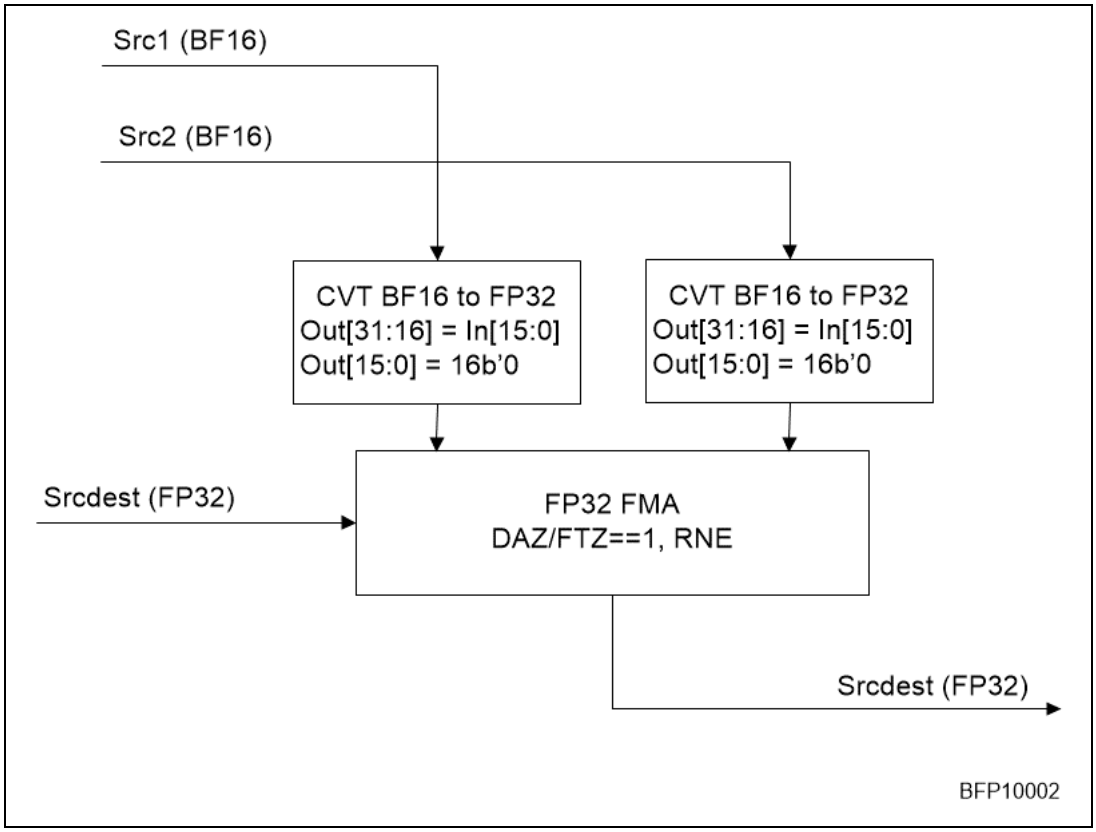


Figure 1-2 is showing an FMA3 unit. This unit takes two BF16 values and multiply-adds (FMA) them as if they would have been extended to full FP32 numbers with the lower 16 bits set to zero (FP32 Mantissa[15:0] = 0). With this convention, the BF16-FMA is defined as a three-way FP32 FMA with DAZ

and FTZ set to “On”. This means that all potential denormals in Src1, Src2, and Srcdest are flushed to zero before being fed into the FMA unit. The rounding mode of the unit is fixed RNE. In the event a denormal is produced as a result, this is also set to 0 (FTZ=1).

RNE rounding is recommended as this is the standard today for FP32 FMA, and the result is equal to or better than ODD rounding.

This BF16 FMA unit is fully aligned with standard FP32 FMA units, hence FP32 units can emulate the proposed BF16 unit bit-accurately.

1.2.2 Conversion Units: FP32 to BF16

Most of the numerical experiments conducted used simple truncation with great success when converting from FP32 to BF16.

However, applying RNE rounding during down converting can result in slightly better numerical performance; e.g., for ResNet50 training, FP32 accuracy is fully reproduced. RNE rounding mode can be implemented with integer math instructions in case of a software-based down-convert sequence. Dedicated convert-rounding units can speed up this operation.