**文章1：DBPS: Dynamic Block Size and Precision Scaling for Efficient DNN Training Supported by RISC-V ISA Extensions---**

**韩国高丽大学 DAC 2023**

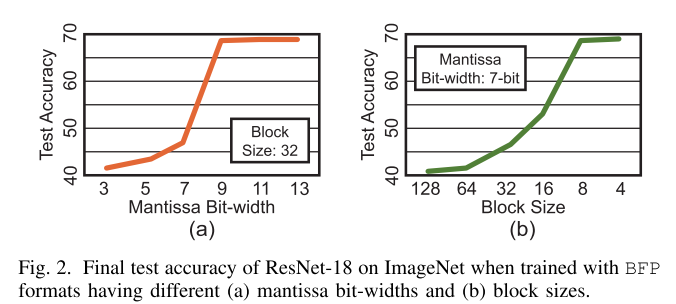
* 研究问题：
* 研究层次：强相关，BFP数据格式改进，MAC+多核间加法树
* 研究重要性：通过动态配置BFP精度和块大小，加速DNN训练
* 研究同异：

同：设计BFP硬件加速器

异：改进了BFP格式，自定义指令扩展了RISC-V ISA，全栈式。

* 挑战：

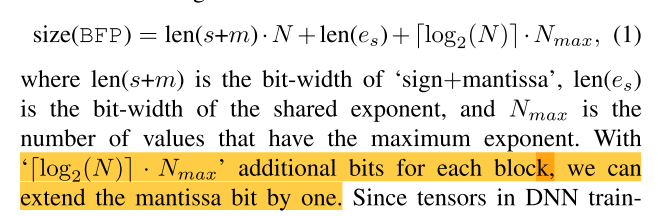
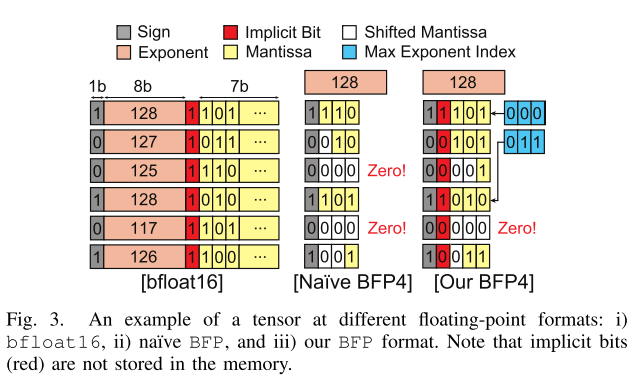
使用BFP格式的挑战在于确定每个张量的块大小和精度。基于BFP的DNN训练的先前工作经验性地确定了块大小及其精度，这限制了其在真实的应用中的使用。FAST 块大小8，尾数很低但FP累加消耗功耗；flexpoint 整个tensor共用一个指数但尾数宽



a尾数9->7精度下降18%，b块大小8->16准确度下降17%，所以需要为下一次训练迭代自动配置所需的精度和块大小

* 解决：

1) 新的BFP数据格式：正常浮点数前导1不存储，但是BFP只有最大指数的是1，其他数是0。本文提出保留每个块内具有最大指数的值的索引，以便可以推断出隐式位而无需存储它们。这有效地将每个操作数的精度提高了一位。存储的N个数的位宽如下，由于DNN训练中的张量具有正态分布，Nmax（有最大指数的值的数量）很小



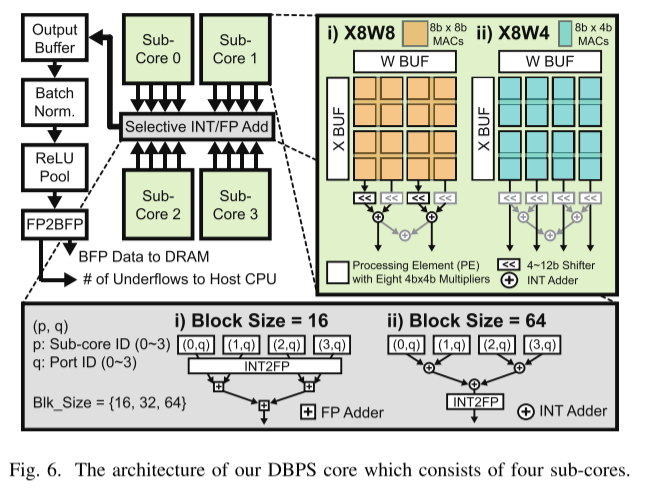
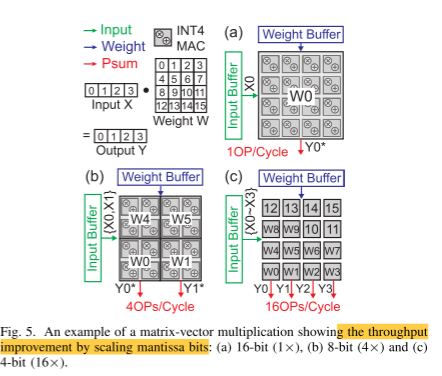
2)DBPS控制：提出了一种动态块大小和精度控制（尾数允许4/8/16b）

在开始时保持块大小较小，然后每当学习率衰减时（例如，每20个时期）将块大小加倍。；

增加尾数位宽提高吞吐量（计算下溢的个数配置精度）

结构图 四个子核组成，每个核包括精度可缩放的MAC

选择性INT/FP加法树：当计算部分和时，块大小16，因为每个子核处理具有不同共享指数的操作数 需要转换FP后在FP-ADDER累加。块大小64，则四个子核处理共享一个指数值的相同块 用INT32累加器



* 作用：

对于尾数的低位宽的处理，精度可缩放的MAC

**文章2：Optimizing FPGA-Based DNN Accelerator With Shared Exponential Floating-Point Format**

**TCAS 西交2023**

* 研究问题：
* 研究层次：弱相关，BFP数据格式改进，用FPGA中DSP资源优化MAC操作和FP-ACC
* 研究重要性：新格式减小存储开销，降低了由于低位宽尾数移位的精度损失。
* 研究同异：

同：BFP的格式改进

异：在BFP基础上增加校正位，用于对数字的指数E与共享指数值Es之间的差异进行编码。

* 挑战：

1)精度：用INT8精度损失大；定点量化，神经网络的每一层的缩放系数需要预先确定，当输入。量化需要微调但是有些数据隐私场景 难重训，

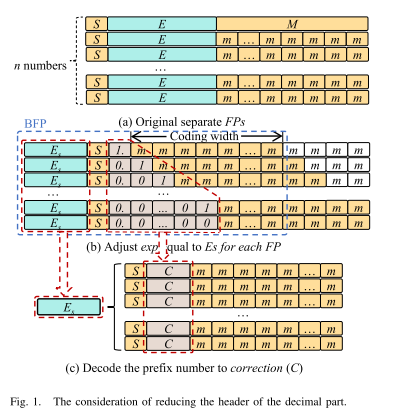
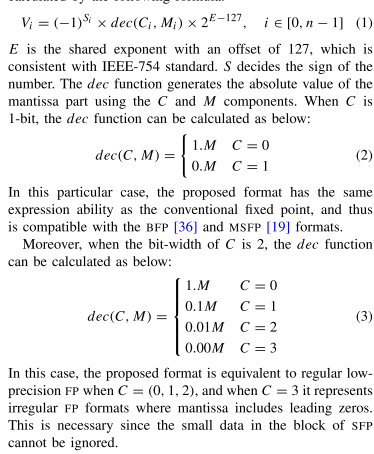
而在FPGA上部署浮点资源开销大；使用更低位宽，FP 8格式不能直接从全精度神经网络转换，因为用户必须手动确定每层使用哪种格式

2)BFP转换时，M的前导数并不总是等于1，因此必须显式存储M的前导数，从而导致额外的存储开销。其次，由于尾数的右移，块中的小数字的精度显著降低。

* 解决：

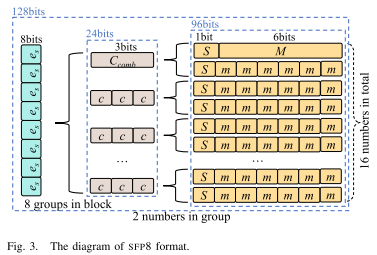
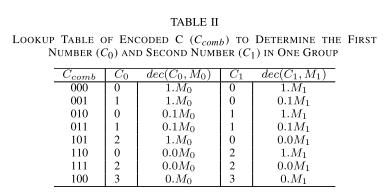
1）shared-exponent-FP共享指数实现高数据压缩率，同时在低精度格式上保持良好的DNN模型精度。

图c将M的头部编码为值C，表示M中前导零的数量。C的这个值被称为校正位； dec函数使用C和M分量生成尾数部分的绝对值。当C1bit，具有与传统定点相同的表达能力，因此与BFP和MSFP格式兼容；；当C2bit，C =（0，1，2）时，所提出的格式等效于常规低精度FP，而当C = 3时，它表示不规则FP格式 尾数包括前导零

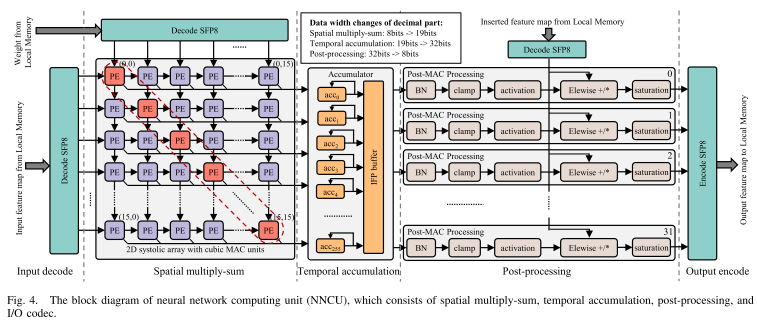
RESNET18 块大小16，对于大多数层共享指数和原来指数差值是0/1，最后一层>=3，所以分成=0 =1 >1考虑

SFP块包含16个浮点数，块中的所有数字共享8位指数Es。进一步分为8组，每个组具有用于由两个数字共享的组合校正分量的3位Ccomb。此外，每个数字有1位的符号和6位的尾数。因此，一个SFP 8块包含总共128位。整体密度相当于INT 8和FP 8。设计了一个查找表，以尽量简化Ccomb和C之间的转换电路。

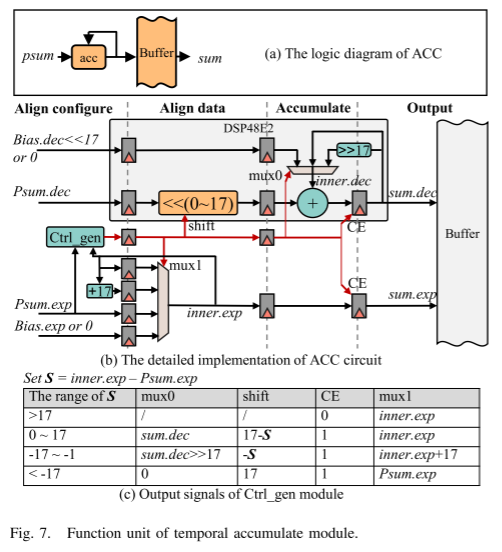
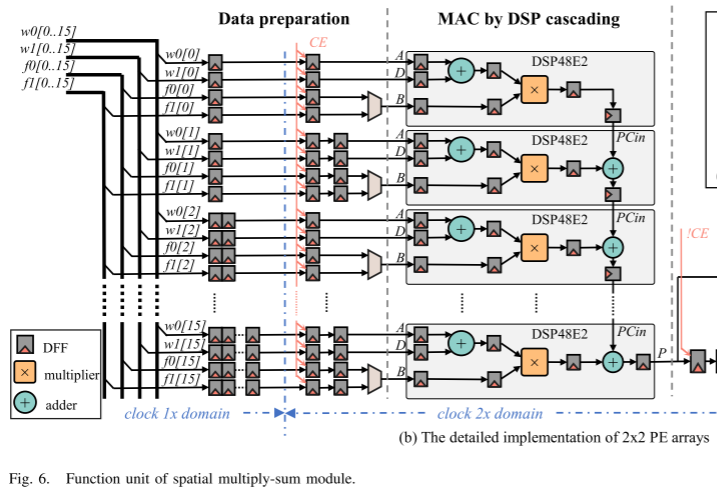
2）提出了一种高效的基于FPGA的浮点神经网络加速器，支持INT8和SFP

输入特征图（fm）、权重（wt）和偏置以SFP8格式存储在片内SRAM中。读数据后，NNCU将其格式转换为单独的小数和指数部分，一个PE包括16个MAC



还进行了几项优化，例如通过最多16个DSP的级联技术（部分和相加）以及优化的高/低位混合（一个DSP相当于2x2 MAC）和2倍频率技术来减少计算资源开销；

通过并行化小数和指数部分的计算来提高浮点累加器运算的计算效率，（DSP内部17b移位器，27x18乘法器，48b累加器）只有Psum.dec被移位任何位，而inner.dec有两个选项：保持其原始值或右移17位，如果差值0-17inner不变，psum对齐；差值-17-0，inner+17，psum对齐；s<-17 inner太小



* 作用：

考虑在尾数很低时，如何改进数据格式

**文章3：Accuracy Boosters: Epoch-Driven Mixed-Mantissa Block Floating-Point for DNN Training**

**韩国高丽大学 Machine Learning 2023**

* 研究问题：
* 研究层次： 强相关，软件层次的HBFP设计空间探索
* 研究重要性：从数学上分析量化对DNN模型和训练时期的不同层的影响，证明了在第一层和最后一层以及精度稍高的最后一个epoch使用较高精度，其他使用较低精度足够准确
* 研究同异：

同：BFP设计空间探索

异：FAST 使用基于BFP的逐层混合精度方法，使用2位和4位尾数。与本文工作不同，FAST需要微调其训练算法的几个额外的超参数，这使得它很难应用于其他DNN模型。FlexBlock使用具有各种块大小的4位和8位尾数，但也需要更高精度的BFP格式来进行权重梯度计算，这会受到量化误差的影响。

* 挑战：

1. 先前的工作[12，13]仅研究了HBFP针对尾数位宽为2的幂（例如，2、4、8位）, 虽然两位幂数自然地与片外存储器接口并消除操作数存储中的碎片，但非两位幂尾数可以进一步提高算术密度.
2. 先前的工作研究了各种独立HBFP格式的训练，缺乏对跨层和跨epoch混合尾数编码的的分析。

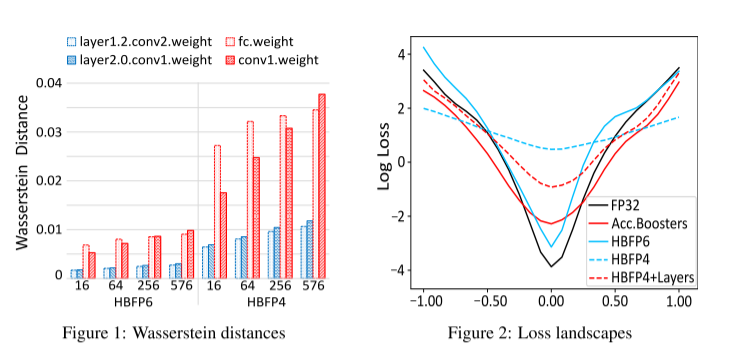
* 解决：

1. 如何在硬件平台上进一步提高算术密度：

a. 可以支持非2的幂数值格式。

b. 允许在保持高精度的同时使用较小的块实现整体更密集的数值格式。不同于其他论文的KL散度，本文使用Wasserstein距离和损失landscapes来研究块大小和尾数位数之间的相互作用，

c. 允许跨层和跨epoch的混合尾数HBFP编码。



2）提出了一种名为Accuracy Boosters的混合尾数DNN训练机制：

由于CNN模型的第一个卷积层和最后一个全连接层对最终模型精度有很大影响，保持这些层的高精度可以降低其余层的精度。所以大多数时期使用HBFP 4（4位尾数）. 仅在DNN模型的最后一个epoch以及第一层和最后一层用HBFP 6进行训练。本文声称这样能将精度提高到FP32水平。

* 作用：

参考论文的设计空间探索方法，讨论block\_size和尾数位宽的影响。

**文章4：LBFP: Logarithmic Block Floating Point Arithmetic for Deep Neural Networks**

**南京大学 APCCAS 2023**

* 研究问题：
* 研究层次：弱相关，改进数值格式LBFP及其MAC结构
* 研究重要性：·提出的改进LBFP降低了DNN中乘法的复杂性。LBFP量化显示出比整数或BFP量化更高的硬件效率。
* 研究同异：

同：BFP格式改进，

异：对数块浮点（LBFP）结合了对数运算的优点和BFP的高效表达。

* 挑战：

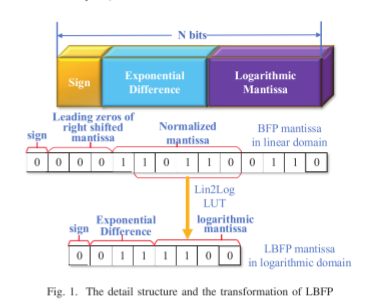
DNN涉及大量乘法运算，与加法运算相比，其计算复杂度要高得多。

* 解决：

1. 对数块浮点（LBFP）:采用对数运算将乘法运算转化为加法和移位运算

LBFP包含符号位（s）、指数差位（ed）和对数尾数位（lm）

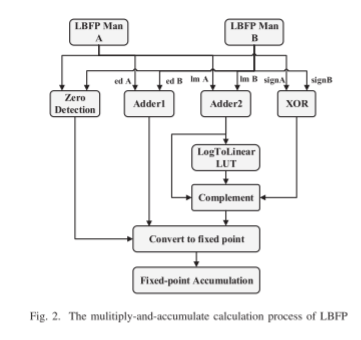
Lm=log2nm 转换通过查找表完成，nm为归一化后的尾数，lm ∈ [0，1），仅保留lm的小数部分，并省略lm的整数位的固定0。Ed和lm宽度可以根据计算精度的要求进行调整。



8位LBFP：ed的宽度为3，lm的宽度为4。共享指数的宽度被设置为5位。

然后，利用3d2位FP目标分布与量化分布之间的KL散度确定共享指数，样本图像来自验证集而不是训练集，整个过程可以在离线几十秒内完成。在运行时不需要额外的计算资源来获得共享指数。

2）乘积的符号由signA和signB的XOR生成。零检测检测A或B是否为0，可以实现精确的零乘法。加法器1用于确定乘积相对于共享指数的指数差。加法器2用于将A和B的对数尾数相加。但是对数域加法复杂，log(2X±2Y)= max(X,Y)+log(1±2-|X−Y|)需要将数据转换回线性域用查找表。根据指数差对乘积进行对齐并与定点数进行累加。



* 作用：

将某项已有的技术方法与BFP结合起来进行计算优化。