测试case

1. 基础测试seed 000：Liu:66 sim\_TS3D\_liu Qiu:40 sim\_TS3D\_2
   1. CFG：按CONV4?但真实数据是CONV2 64个128bit
      1. CONV2 参数的size
         1. FLGACT: **2(patch) X** **4(frame) X 2(block) X 16(row) X 16(col) X 32(chn)**
            1. 放在片上有 4x2x16x16x32 = 8KB = 1块SRAM
         2. ACT; 与FLGACT相同顺序，要考虑稀疏度75%
            1. = 16KB = 2块SRAM
         3. FLGWEI: **2(Ftrgrp) X 2(block) X 16(PEB) X 28(Weight) X 32(chn)**
            1. 片上有2x16x28x32=28672bit = 3.5KB = 一块SRAM
         4. WEI: 2（FtrGrp）X 432(所有weights) x 64(所有通道)，要考虑稀疏度75%
            1. 7KB，但由于增加到16的倍数，超过8KB用两块SRAM
      2. CFGGB的参数
         1. alloc与total的关系？total是否太少？？位宽考虑增加到5bit?
            1. 没有total\_wei是因为根据需要会砍通道到放得下的，比如砍成4块，那么算下一块相当于下一个Ftrgrp
         2. num\_loop\_wei为frame 4 因为此时是复用act固定一个patch（因为loop patch是最外层）
         3. num\_loop\_act为FtrGrp 2 因为复用act要与所有FtrGrp卷积
         4. 给GB的num alloc loop均是次数，即1就表示一块一次
            1. // CONV2: IFCFG\_data =
            2. // { 1'd0,
            3. // 6'd10, 8'd1, 11'd1, 6'd3, 10'd1,
            4. // 4'd1, 4'd2,4'd1, 4'd2,
            5. // 4'd1,4'd1, 4'd2,
            6. // 12'd4, 8'd2,
            7. // { 20'd1, 8'd0, 1'd1, 1'd0, 3'd2 }};
         5. 输入的5个文件统一用二进制，128bit一行，文件内的顺序在草稿本的loop，同时生成相应的十进制文件，wei/act用4d，有符号，地址用12d无符号。
            1. flag 1bit
            2. wei/act 8bit 有符号数
            3. wei\_addr 16bit 无符号数
            4. 目录结构

act\_data

act\_L00.txt // 包含一层的所有act, 是用于TB的总文件

act\_L00\_P00.txt //包含一层的第0个patch的act，用于理想模型读入

act\_flag

flagact\_L00.txt

flagact\_L00\_P00.txt

wei\_addr

weiaddr\_L00.txt

weiaddr\_L00\_F00.txt // 第0个FilterGroup

wei\_data

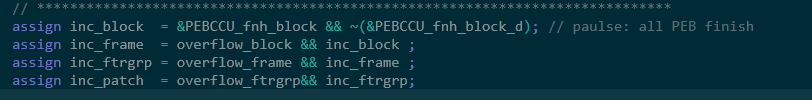
wei\_L00.txt

wei\_L00\_F00.txt

wei\_flag

flagwei\_L00.txt

flagwei\_L00\_F00.txt

* 1. POOL：步长固定为2，没有帧pooling; **没有DELTA的功能**（单模块待测试，CCU\_valdelta=0）；BF\_flg\_data是没有地址的？直接一直放？
  2. 只测试TS3D，IF模块可bypass掉
     1. 片外最开始将pad\_OE拉高->给ASIC的clk使能拉高->ASIC上升沿拉高GBIF\_cfg\_val->FPGA下降沿采到GBIF\_cfg\_val和GBIF\_cfg\_info，根据GBIF\_cfg\_info
        1. ASIC读：拉低pad\_OE, 然后下降沿拉高IFGB\_rd\_val，ASIC上升沿读取到IFGB\_rd\_data，
        2. ASIC写：拉高pad\_OE, 然后下降沿拉高IFGB\_wr\_rdy，FPGA下降沿读取到GBIF\_wr\_data，
        3. FPGA传完所有的数，下降沿拉高pad\_OE，等待10ns，拉高IFGB\_cfg\_rdy，等待GBIF\_cfg\_val
     2. pad直接对接TS3D，pad\_OE是FPGA提前给的。
  3. GB： weight的flag和数量相配；加上pullback\_act
     1. cfg\_info
        1. 0001 config信息
        2. 0111 wei\_addr
        3. 1001 wei\_data
        4. 1011 wei\_flag
        5. 1101 act\_data
        6. 1111 act\_flag
  4. 测试数据用真实的：从神经网络中提取；暂时无所谓
  5. 给GB的SRAM保证8KB写满。
     1. GB修正：修正PEB/SRAM\_WEI，只要datain\_val有效即可写到SRAM，state在scanf状态将intr写入fifo，不输出，写完之后转向WRITE状态时，等GB取instr，同时等GB写的wei写入SRAM。
     2. 修正：配置的num total alloc只给一层开始配置一次，GB只需要等待reset\_all然后拉高GBCFG\_rdy等待配置
  6. 
  7. 测试功能：先在block loop里面block累加，再在loop frame时向一下PEC传累加的结果（因为三维卷积会在帧的维度上再卷积一次），每一帧出来的结果（TS3D.py中是PSUMGB\_data的第二个PEC，RTL中也是第二个PEC的PSUMGB\_data）再与基帧的结果相加，得到真实的结果，然后传给pool做RELU和二维的pooling，结果再稀疏化传给BF
  8. 整个仿真路径在TS3D/zhoucc/sim\_TS3D/
  9. GBPSUM
     1. 修正：CCUGB\_reset\_patch（实际上是一个ftrgrp的计算）必须要等pool完frame+1次（由于frame方向上的padding）后才会给脉冲；CCU必须等CCUGB\_reset\_patch才会进行下一Ftrgrp计算
     2. CCUPOOL\_reset也得等POOL完成才会reset，暂时定为0；
     3. 问题：inc\_block会立即启动下一个ftrgrp，但这时，pooling同时还在读取GBPSUM？会出错吗？（可能不会pool很快，但问题存在）
     4. 问题4，frame转向下一帧太早，还写还未完全写入SRAM：根本上是inc\_block太早了，一是让PEBCCU\_fnh在保证psum被取走才拉高（不可以，无法用PSUMGB\_val来保证，还是要从PEB入手，保证被取走，完成整个过程是干净的IDLE？-> 不要给inc\_block看什么时候），二是延后4个周期：等GBPSUM把数写完
     5. 问题5：最后一帧SRAM1需要帧变化？其实不需要：CCUGB\_frame变化为0->1->2->3此为正常pool出3帧，->4->5 额外的pool出2帧（block为0不变），然后frame->0 并且reset\_patch回到初始状态。

1. 典型测试 seed 100
2. 完整测试 seed 200