# Speedup time

原本时间mac\_single\_time \* IN\_STATE\_SIZE \* OUT\_STATE\_SIZE \* BATCH\_SIZE

优化后时间 = (load kernel/ state time + mac\_batch\_time)per\_batch \* OUT\_STATE\_SIZE

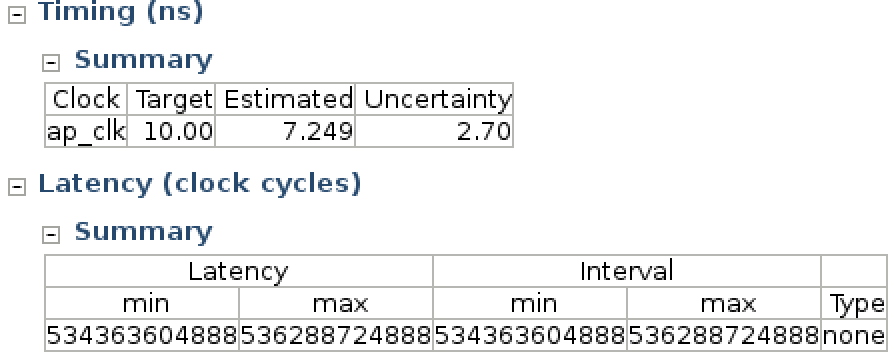
(suppose batch size = 64)

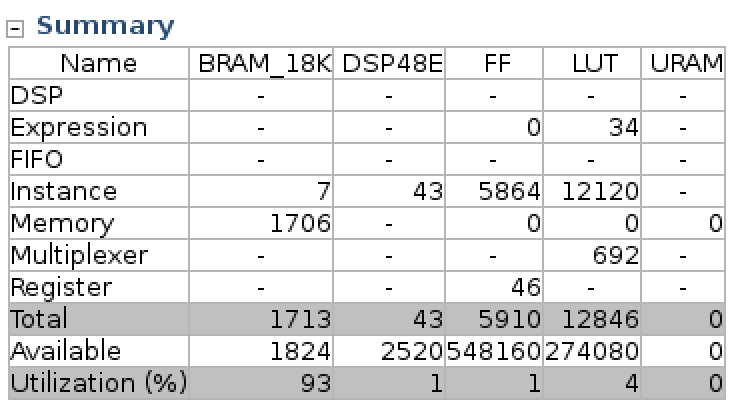
speedup = mac\_single\_time \* IN\_STATE\_SIZE \* BATCH\_SIZE / (load kernel/ state time + mac\_batch\_time)per\_tile

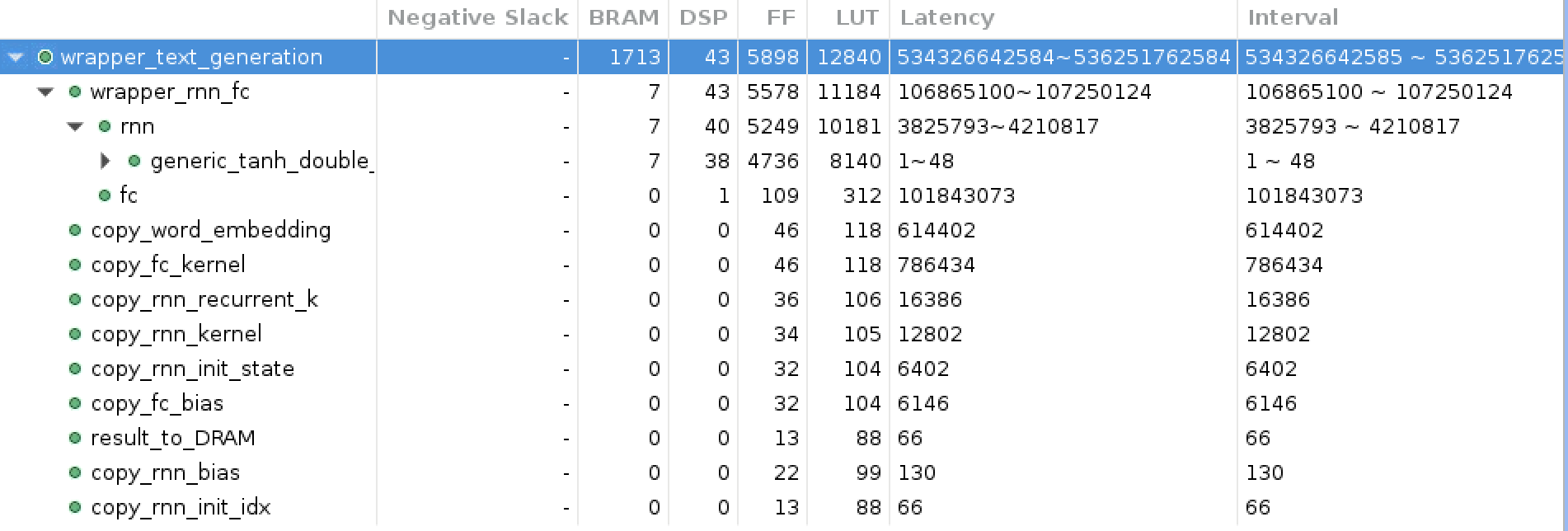
suppose mac\_single\_time = 1, mac\_batch\_time=64, TILE=64, BATCH=64, IN\_STATE\_SIZE = 128 -> speedup = 118

# Baseline

## 5000 step 536B



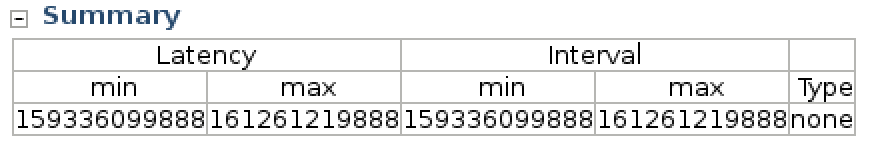


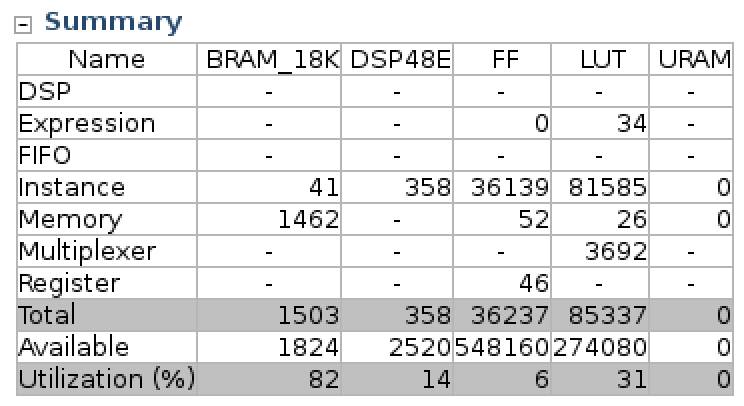


93% of BRAM with FC output stored

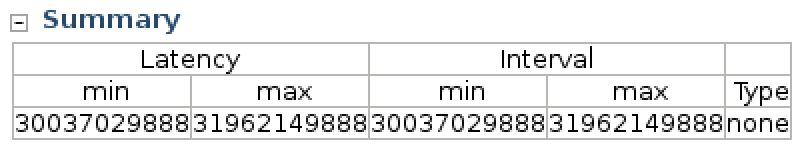
# Only optimize FC

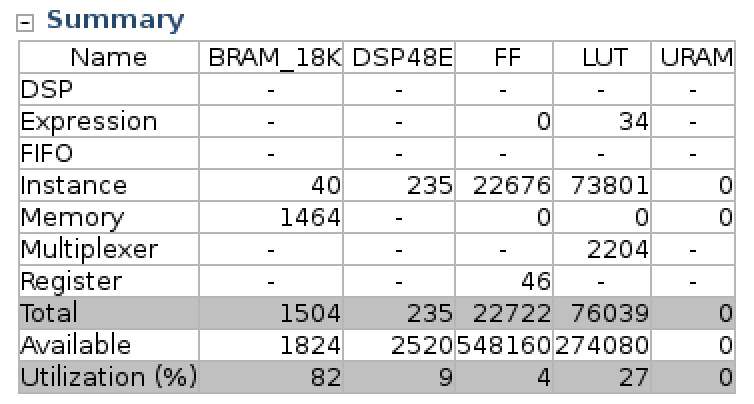
## first try 161B

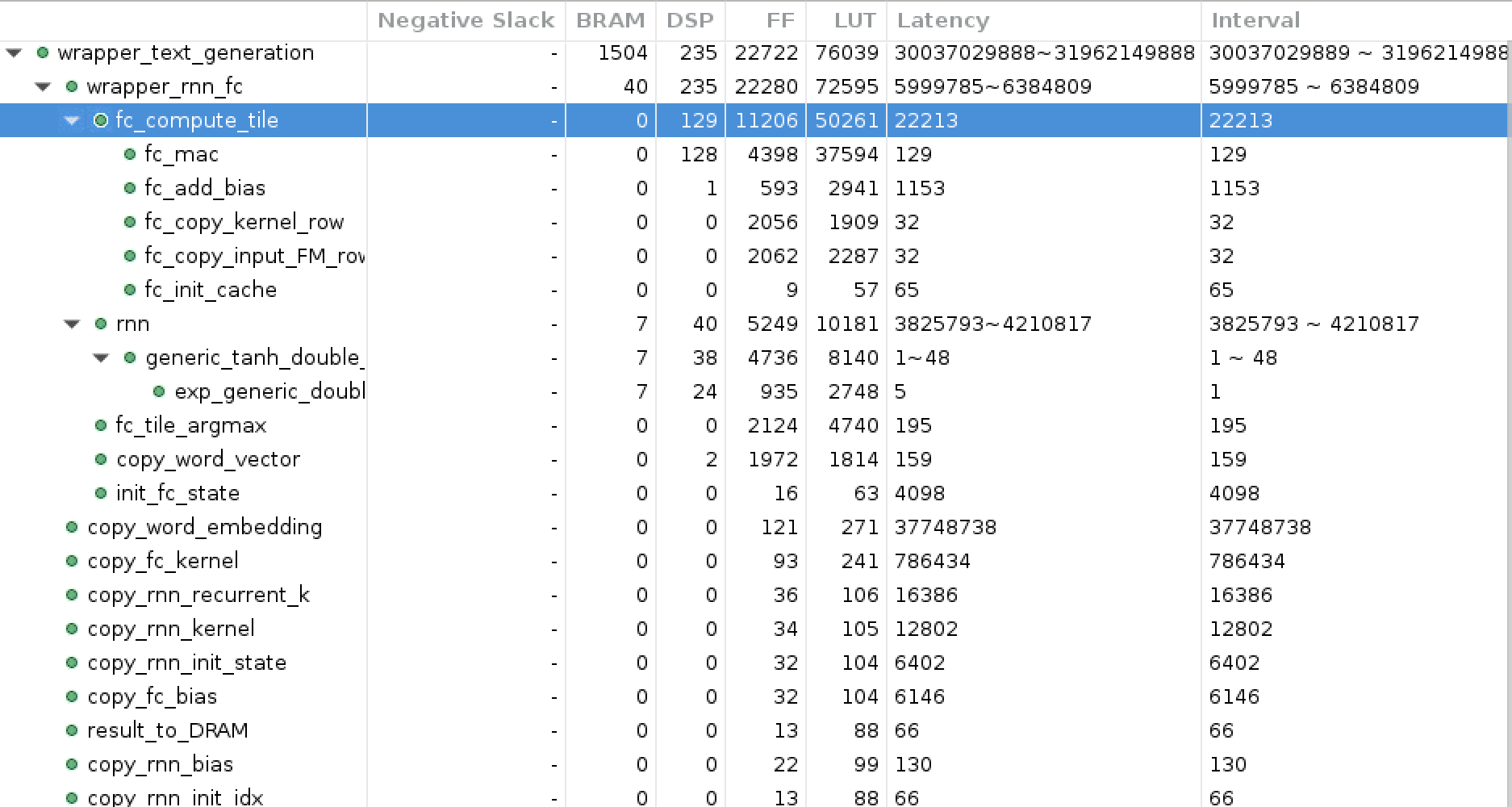




## 2.rearrange fc alignment 30B





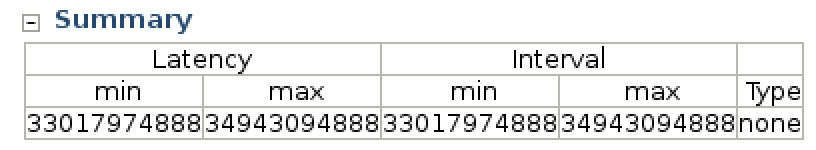


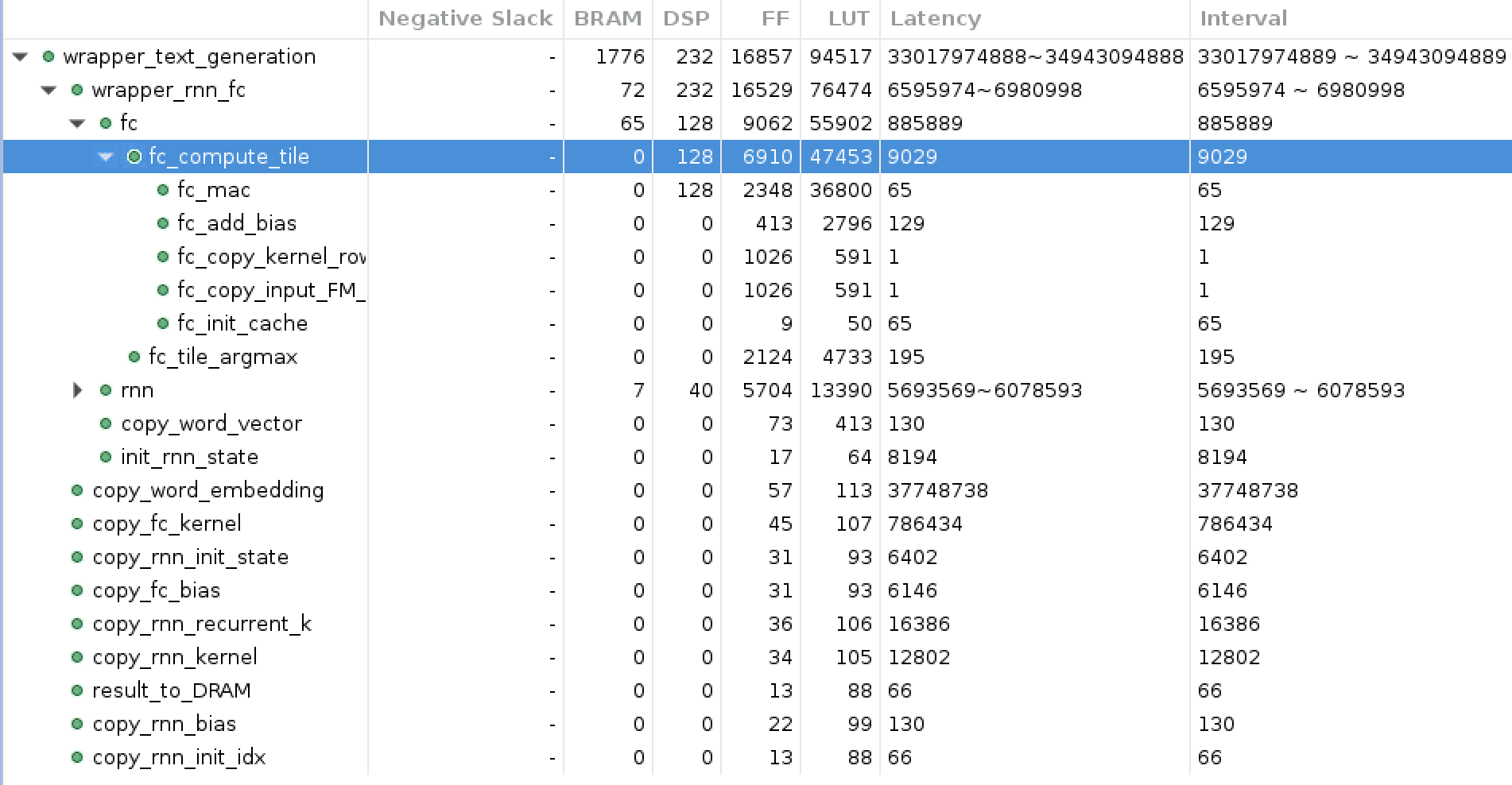
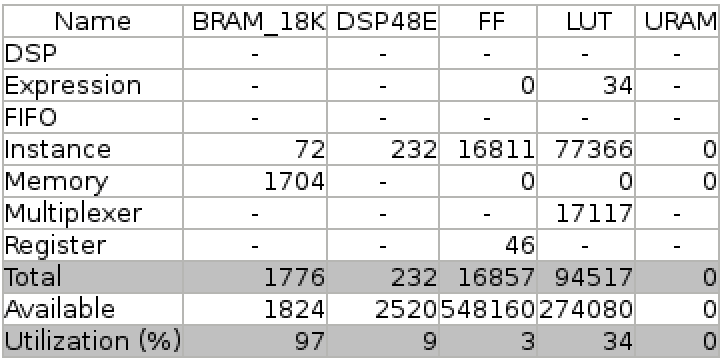
平均一个fm 350个cycle，最好的prefix sum -> ~110

## 3.tuned fc new baseline 34B

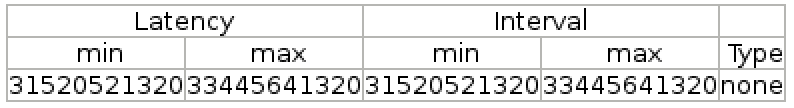
标准fc tile version -> 基于这个version再做微调

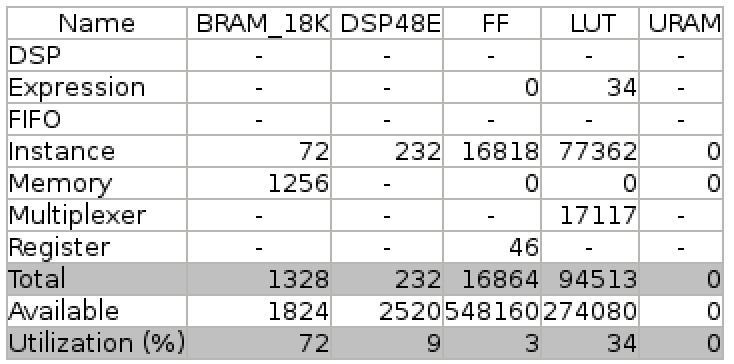
因为unroll了rnn的state，所以降低了rnn的速度，总CC反而比以前多一点

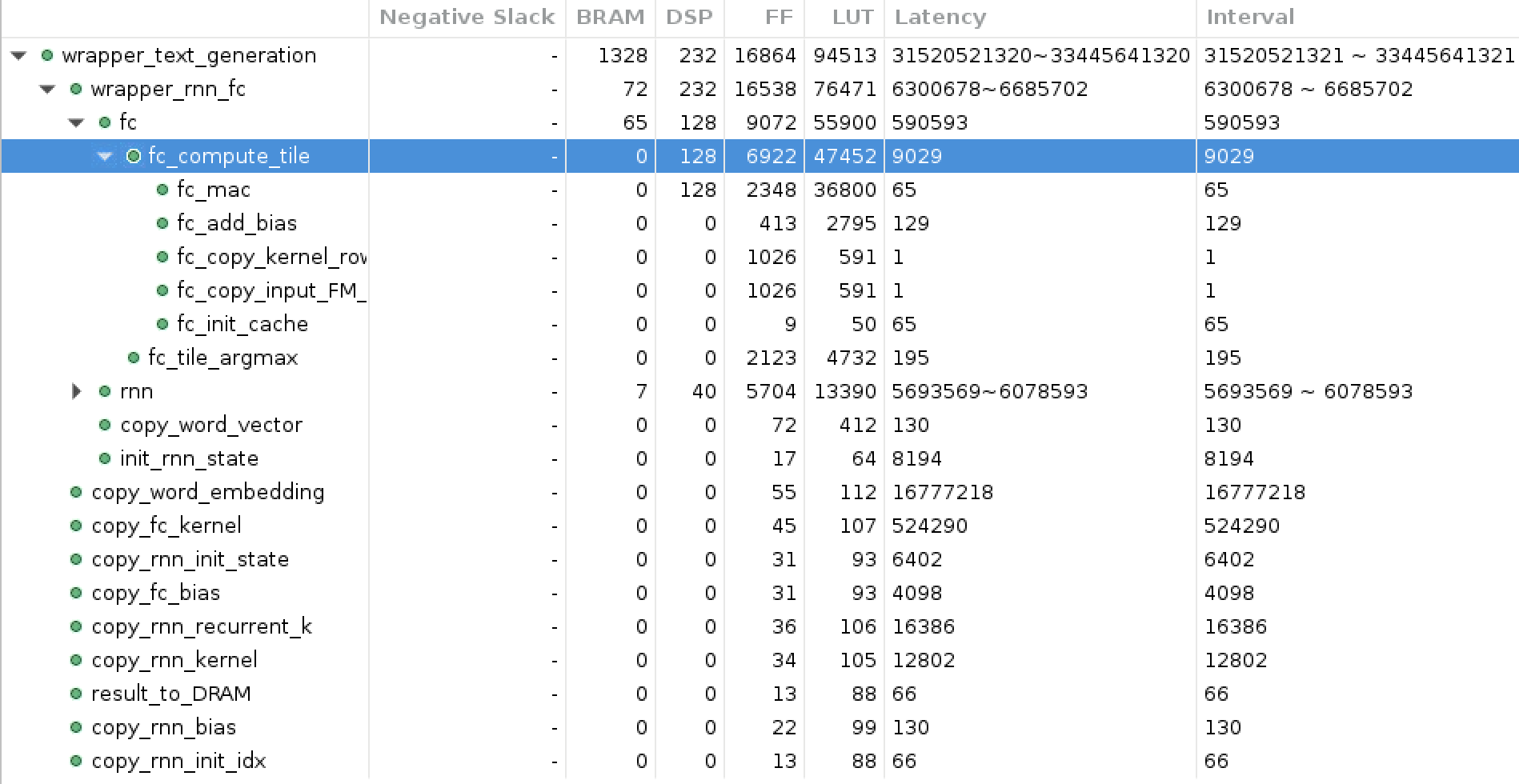




## 4.调整word num到4096 31B







(65 + 1 + 1) \* 128 + 129 + 195 + 65 = 8,965

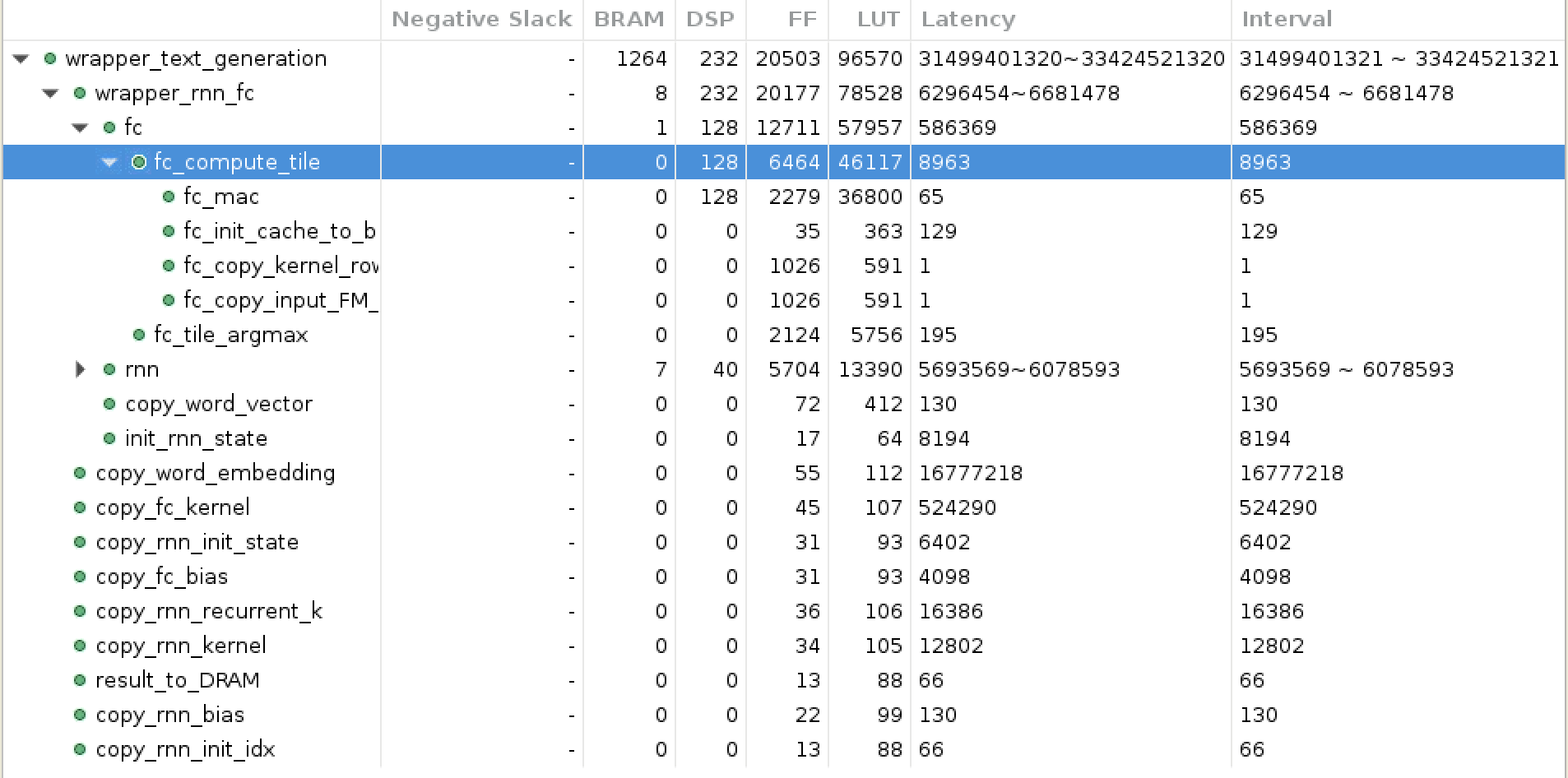
65 \* 128 = 8,320 -> optimize here

combine init and add bias -> init to bias value

## 5.more partition on output\_FM\_cache 31B

#pragma HLS array\_partition variable=output\_feature\_map\_cache cyclic factor=64 -> **128**

**combine init and add bias**

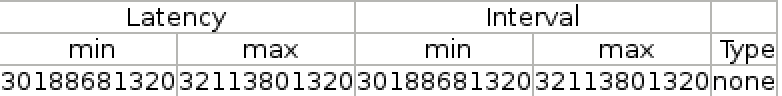
****

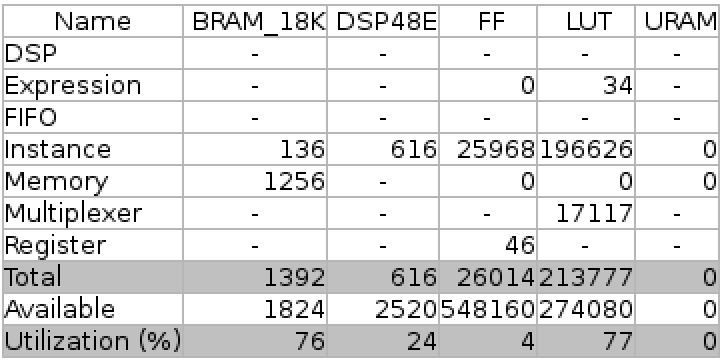
完全一样，还是65 -> more unroll

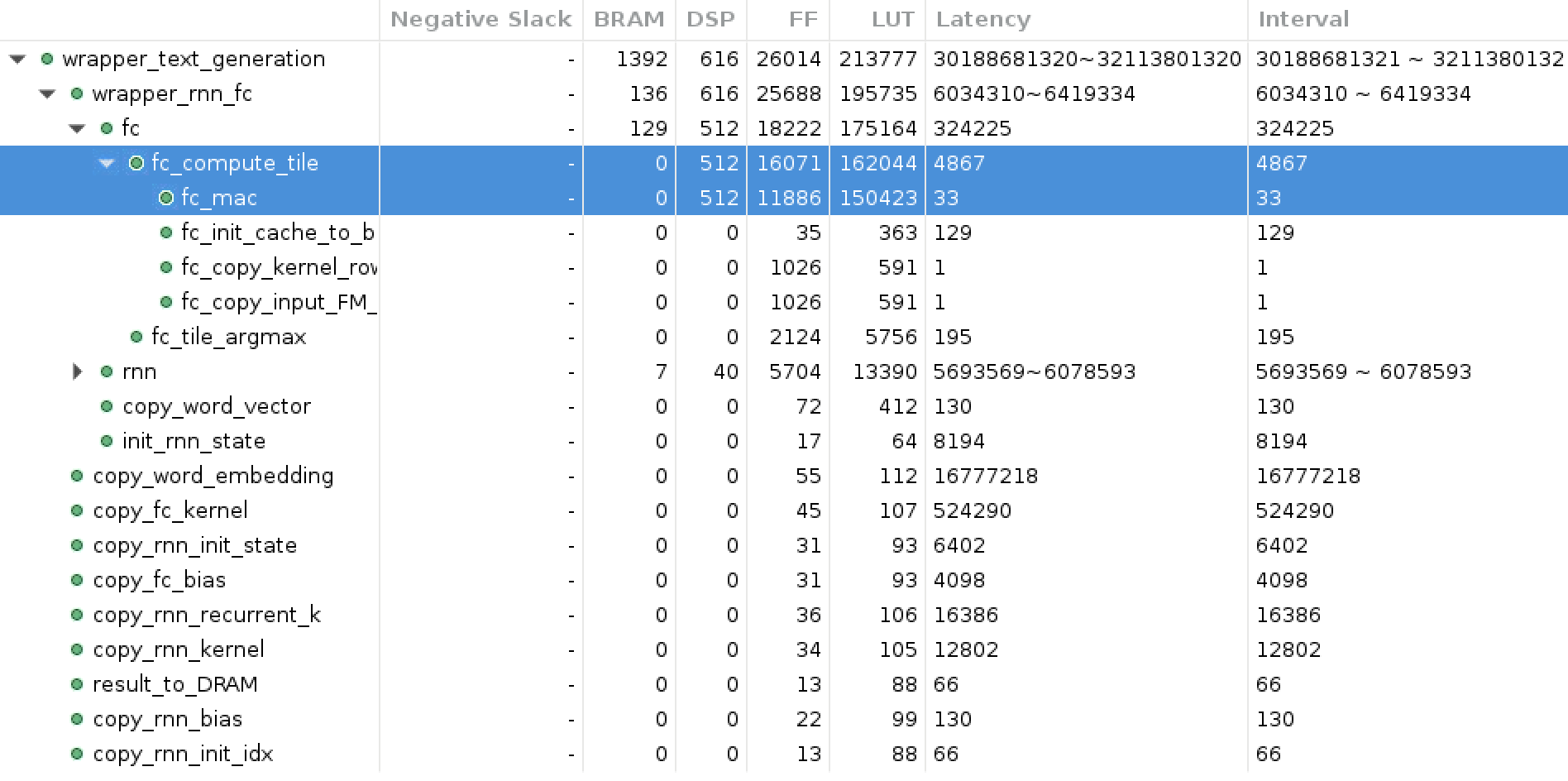
## 6.more unroll in mac 31B

#pragma HLS unroll factor=2 -> **8**

#pragma HLS array\_partition variable=output\_feature\_map\_cache cyclic factor=128

****

****



快了，但是LUT已经到77了。

# Optimize RNN

FC setting:

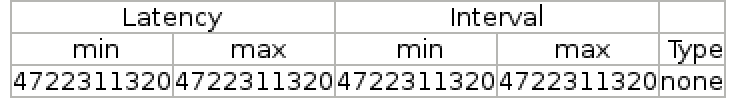
fc\_mac #pragma HLS unroll factor=2

output\_feature\_map\_cache:

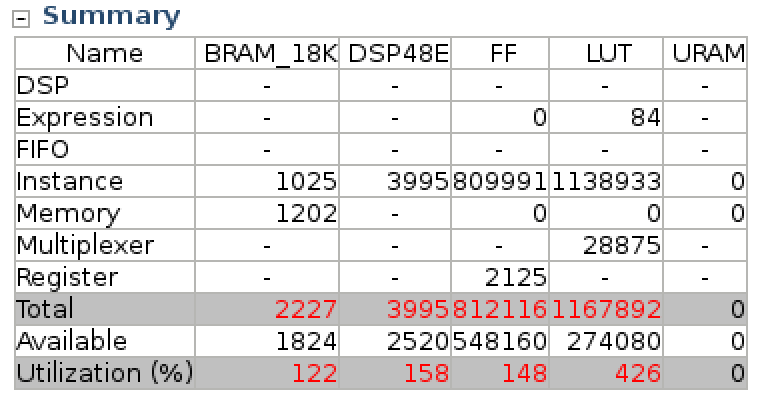
#pragma HLS array\_partition variable=output\_feature\_map\_cache cyclic factor=64

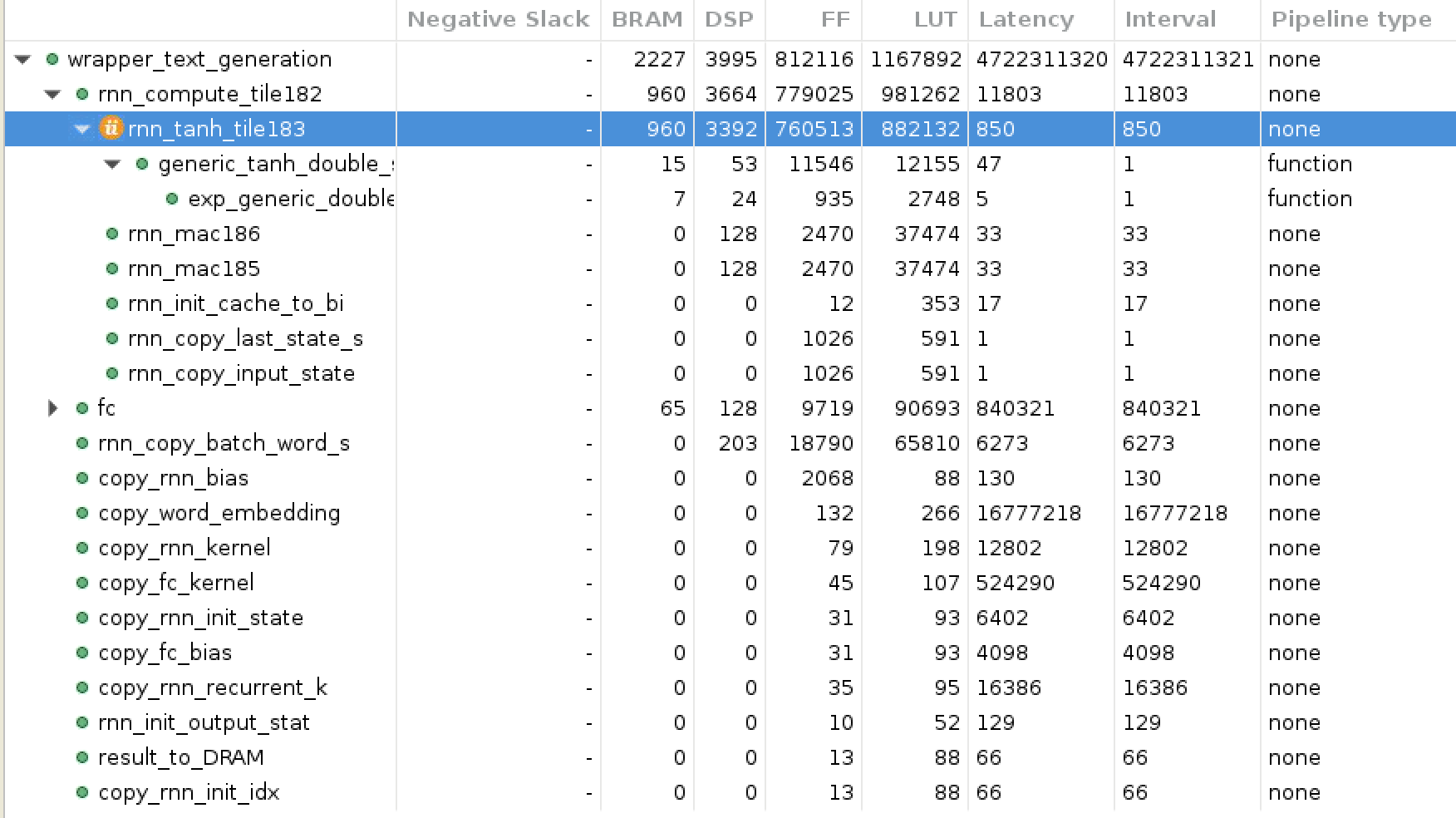
## 1.First try 超出resource

tanh导致了LUT, DSP和 BRAM都严重不足

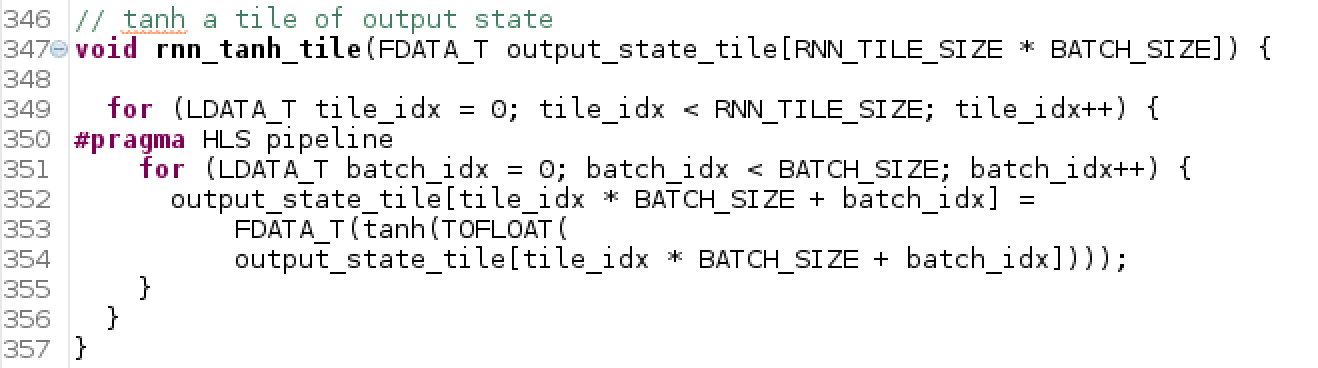


4.7B



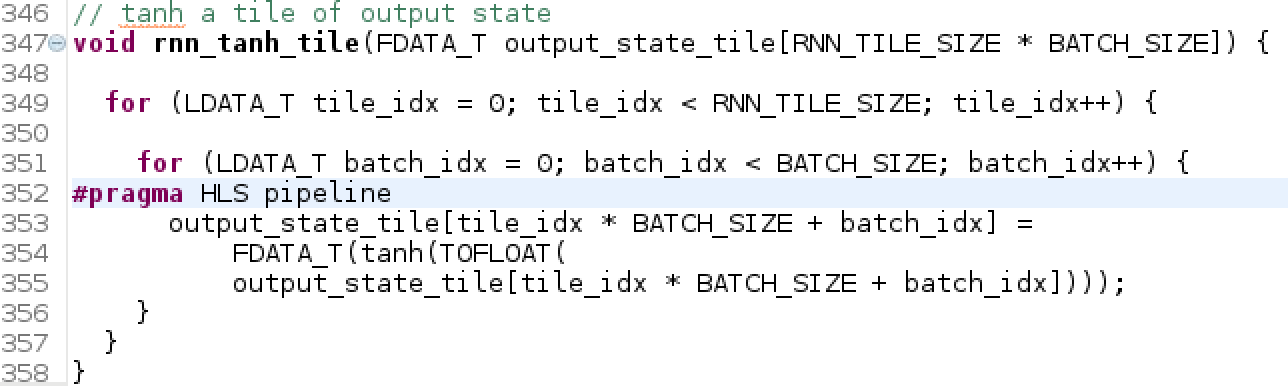


当前implementation: pipeline会把里面的部分unroll

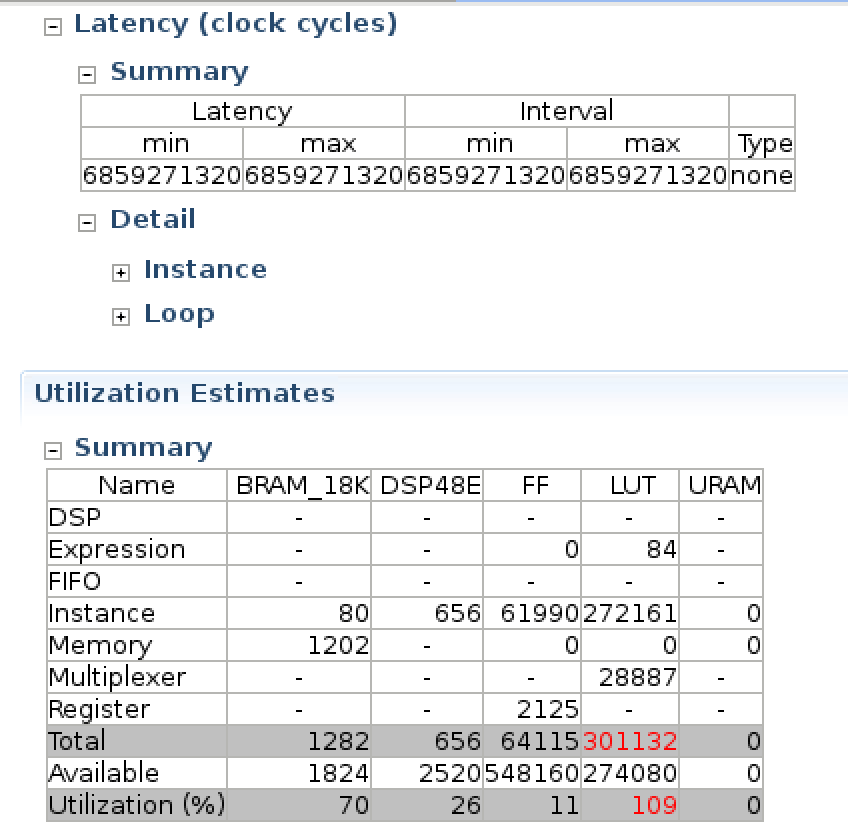


solution: 把pipeline移到里面

## 2.移出tanh的unroll 超出resource

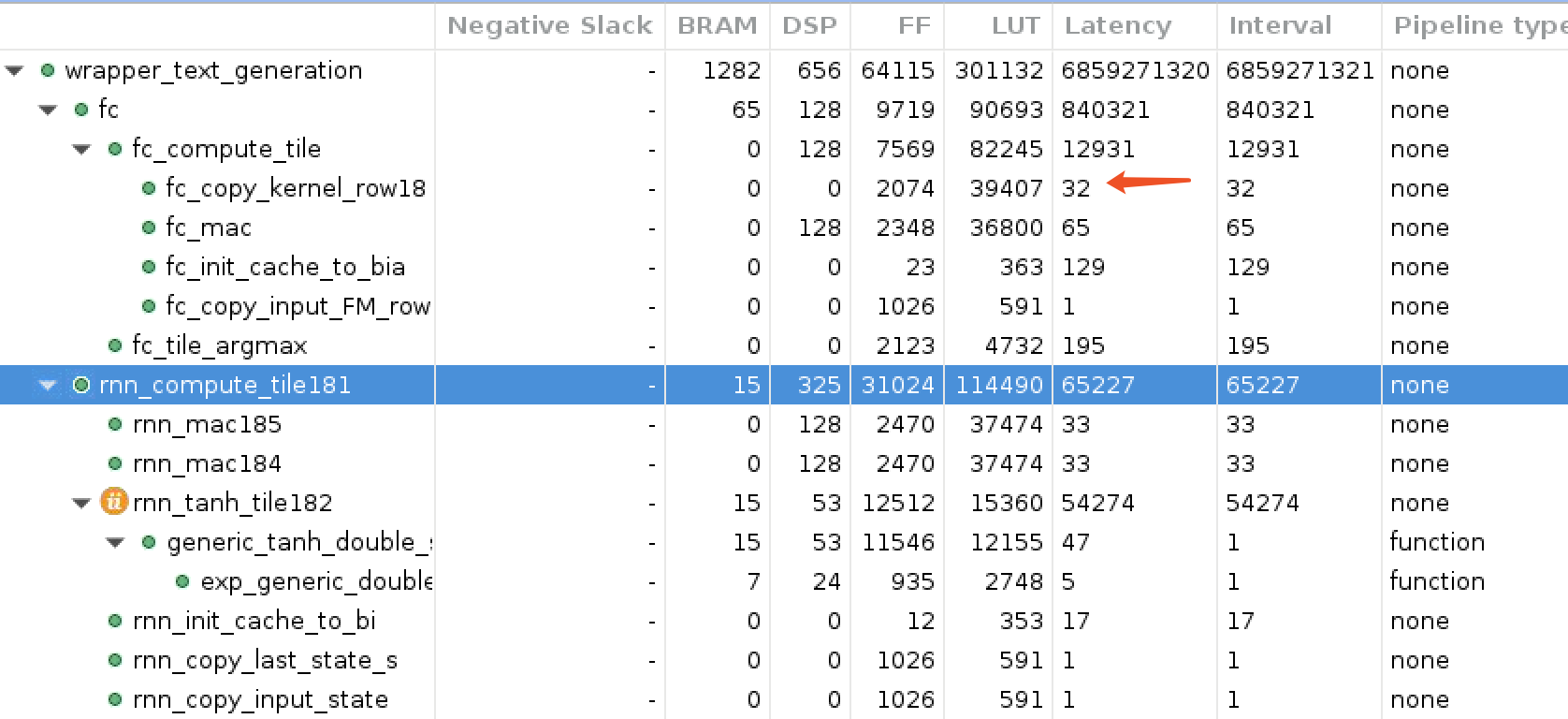


把Pipeline移到最里面

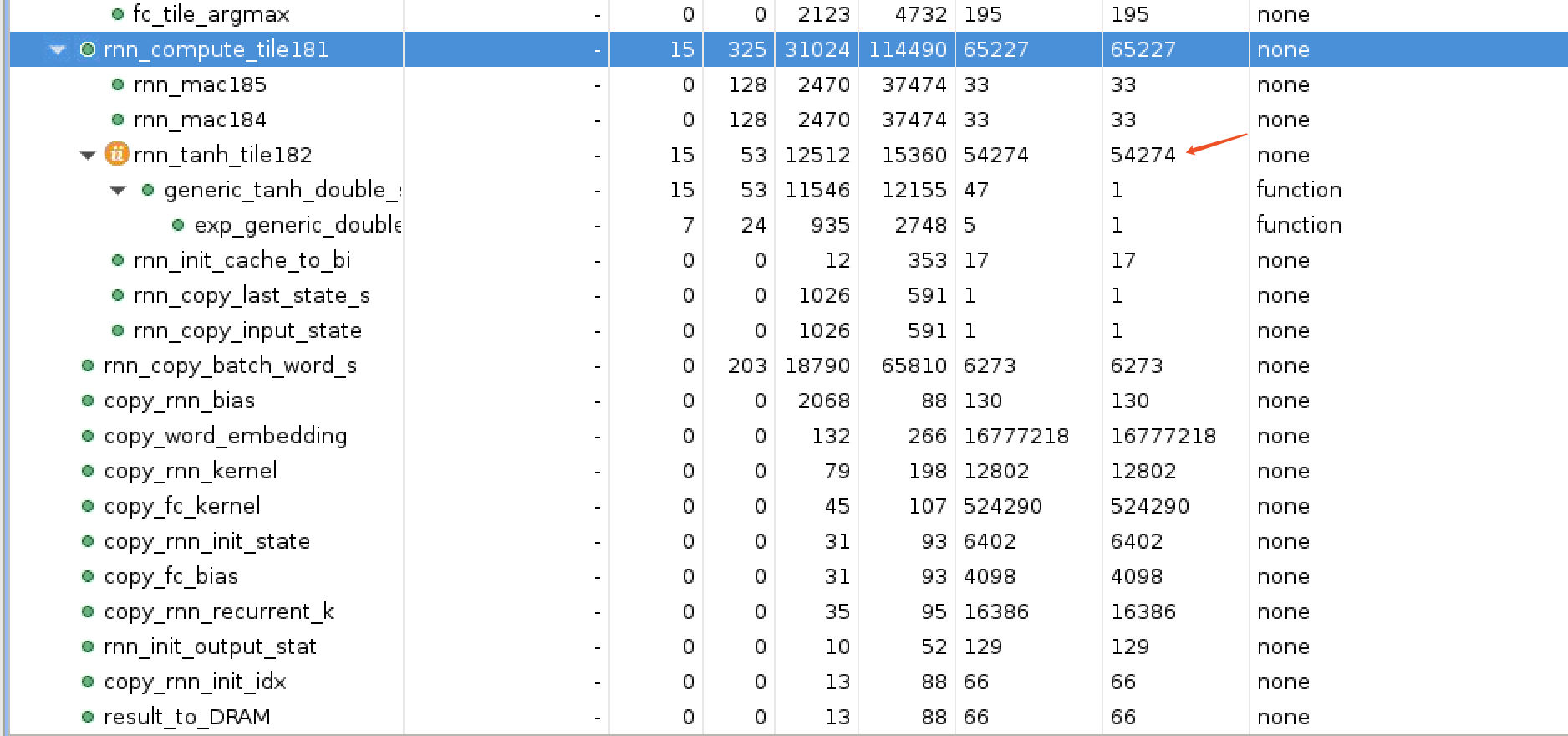


6.8B

LUT不够 -> less



copy kernel 要32个cycle很奇怪，但是暂时没有发现原因



tanh占用了绝大多数资源和CC

先解决resource问题，尝试把FC\_TILE\_SIZE下调

## 3.下调FC\_TILE\_SIZE

#define FC\_TILE\_SIZE 64 -> **16**

**有些资源可以节省，比如copy一行的CC不一定要是1，可以是4，不会影响太大性能。但是类似于output\_FM\_cache这一类的制约写入速度的一定partition的越多越好**

**FC\_TILE\_SIZE=64: 1 + 1 + 65**

**FC\_TILE\_SIZE=16: (1 + 1 + 17) \* 4**

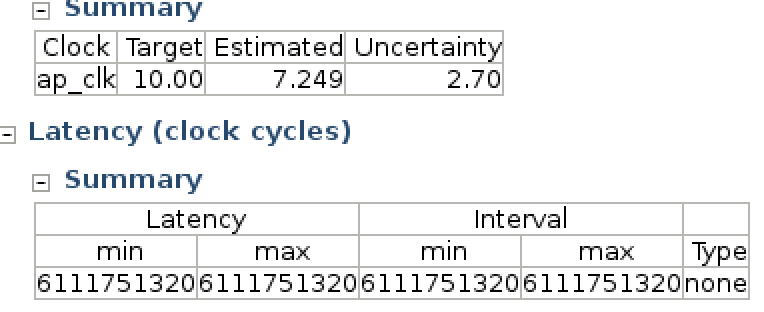
**如果下调kernel的unroll factor，大概(4 + 4 + 17) \* 4，需要原来的1.5倍时间**

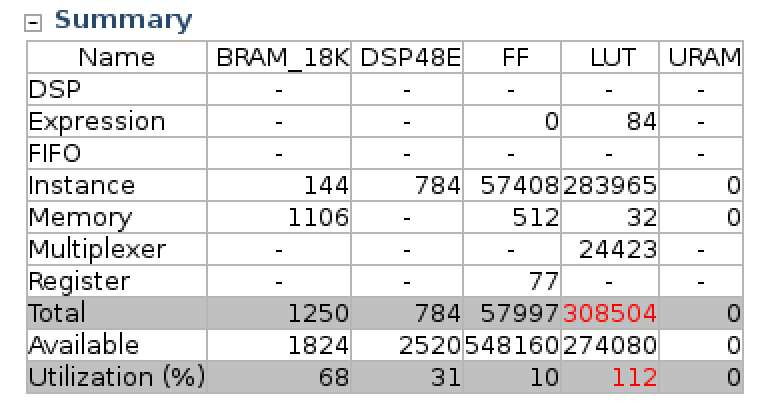
kernel的factor 32/64 -> **16**

#pragma HLS array\_partition variable=output\_feature\_map\_cache cyclic factor=64 -> **128**

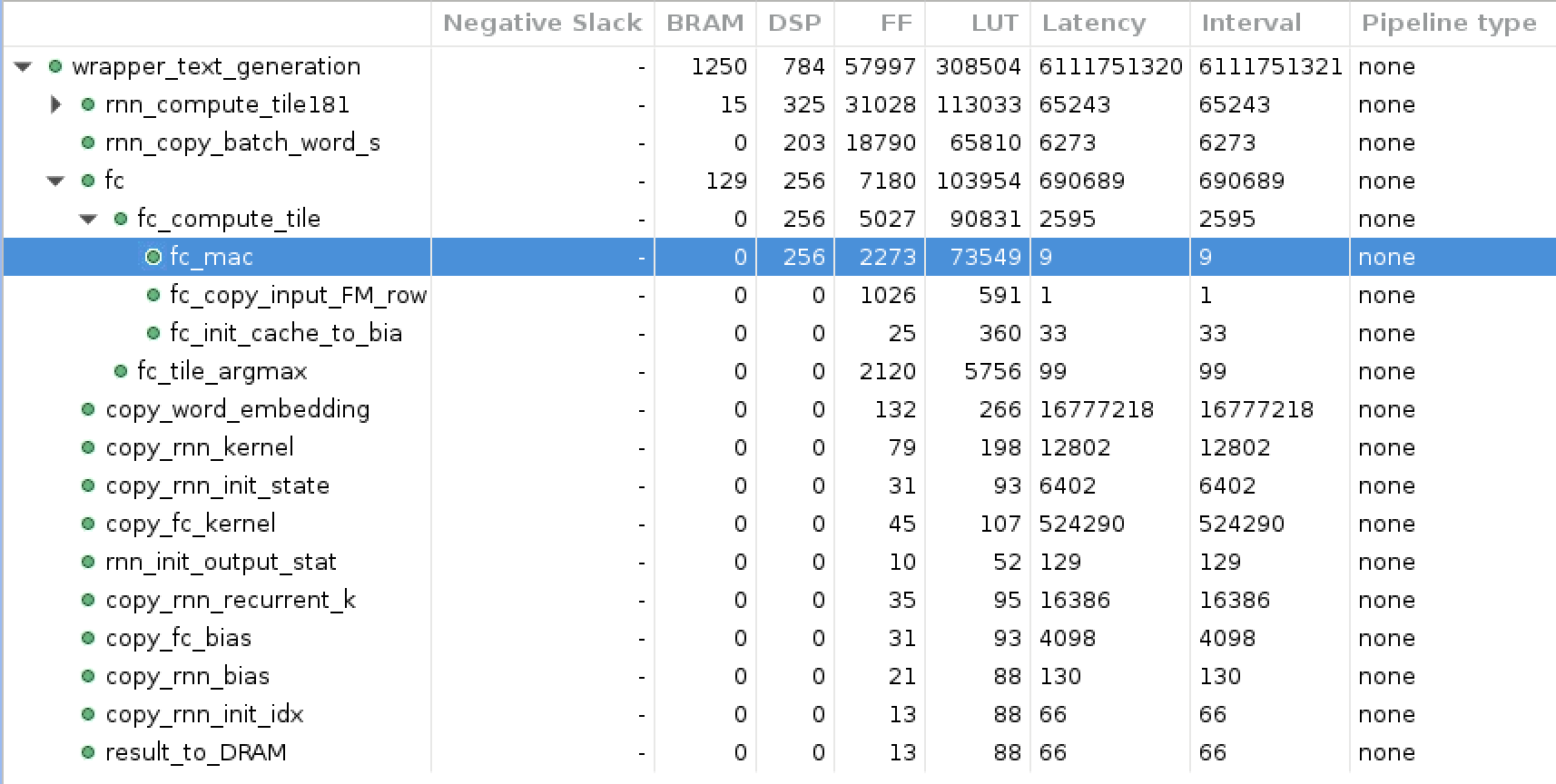
correspondingly, for **fc\_mac**

#pragma HLS unroll factor=**4**

****

****

resource not enough -> copy kernel function -> input whole kernel -> input one row of kernel



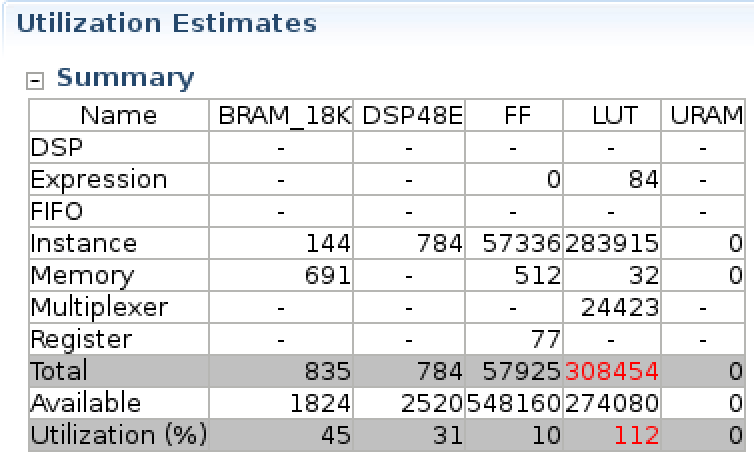
## 4.adjust copy kernel function

previous -> input whole kernel, copy one row

now -> input one row, copy it

**no difference at all**

## 5.Smaller dictionary size -> 2048



LUT和dictionary size无关