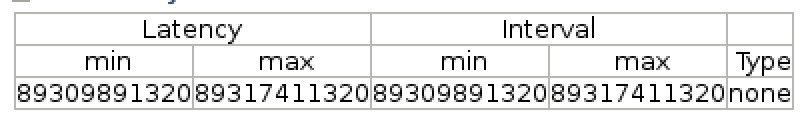
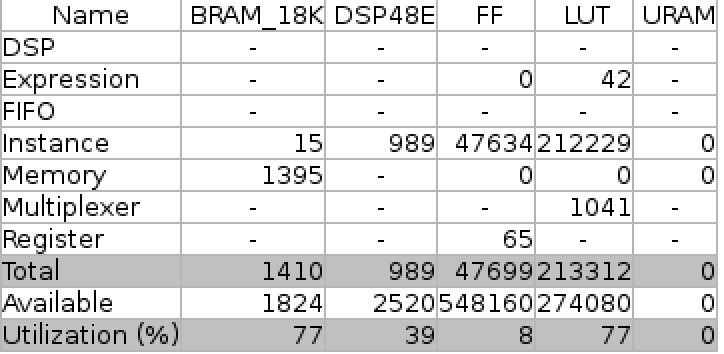
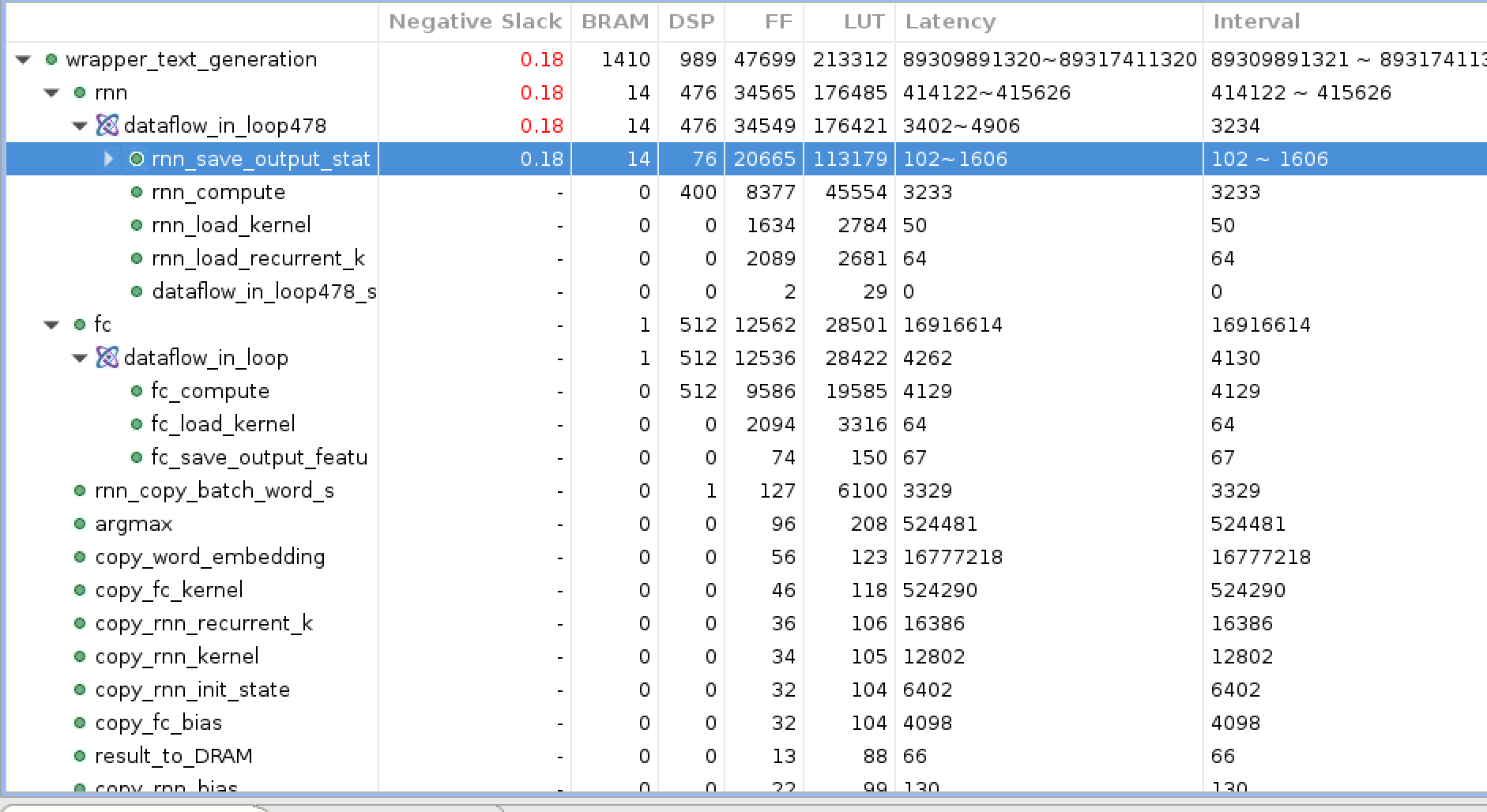
# Baseline

**remove all array partition**

****

****

**tanh used half of the dsp ~35%**



# Optimization

Suppose fixed point tanh

## Only partition prefix sum register 4B

#pragma HLS array\_partition variable=rnn\_input\_state\_BRAM cyclic factor=100

#pragma HLS array\_partition variable=rnn\_state0\_BRAM cyclic factor=128

#pragma HLS array\_partition variable=rnn\_state1\_BRAM cyclic factor=128

compute unroll = 4

rnn:

#pragma HLS array\_partition variable=output\_state\_reg factor=4

#pragma HLS array\_partition variable=kernel\_reg cyclic factor=50

#pragma HLS array\_partition variable=recurrent\_kernel\_reg cyclic factor=64

#pragma HLS array\_partition variable=local\_reg cyclic factor=32 dim=2

#pragma HLS array\_partition variable=local\_reg cyclic factor=4 dim=1

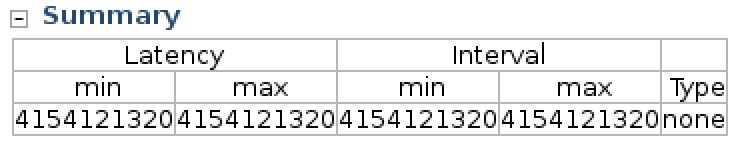
fc:

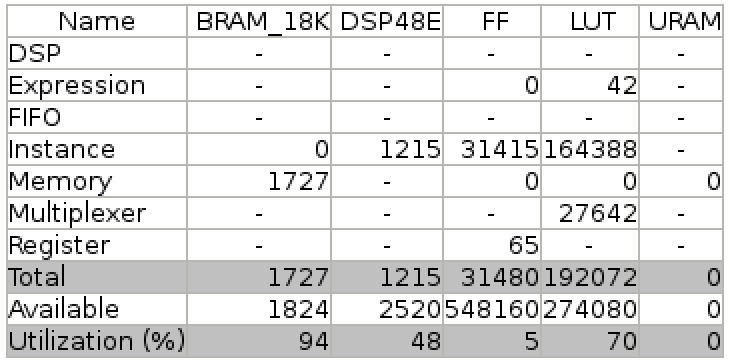
#pragma HLS array\_partition variable=output\_feature\_map\_reg factor=4

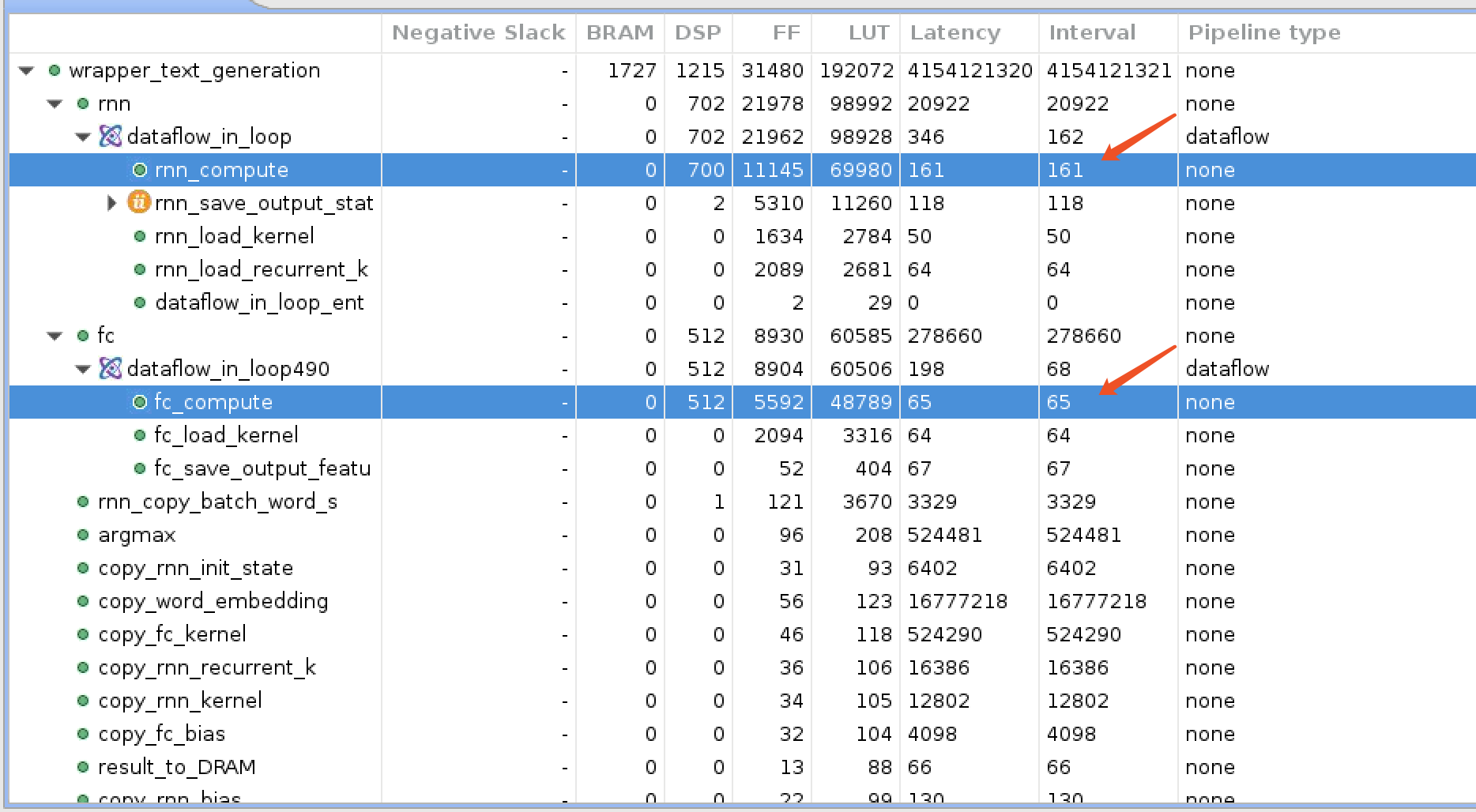
#pragma HLS array\_partition variable=kernel\_reg cyclic factor=64 dim=1

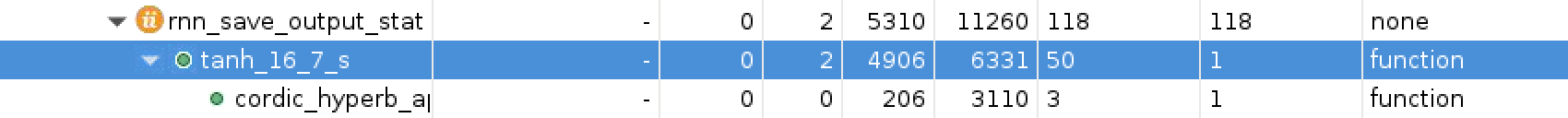
#pragma HLS array\_partition variable=local\_reg cyclic factor=32 dim=2

#pragma HLS array\_partition variable=local\_reg cyclic factor=4 dim=1









one single tanh takes 50 CC

## 2.More aggressive

more prefix sum unroll:

prefix sum loop unroll factor = **4 -> 8**

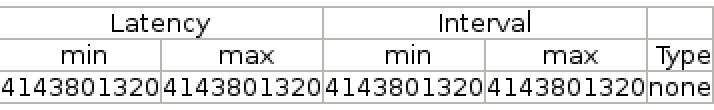
在最外层的Loop加上 #pragma HLS UNROLL factor=2

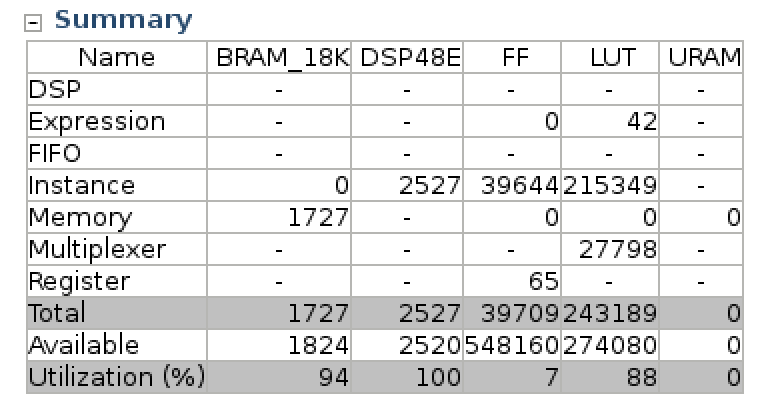
more tanh unroll:

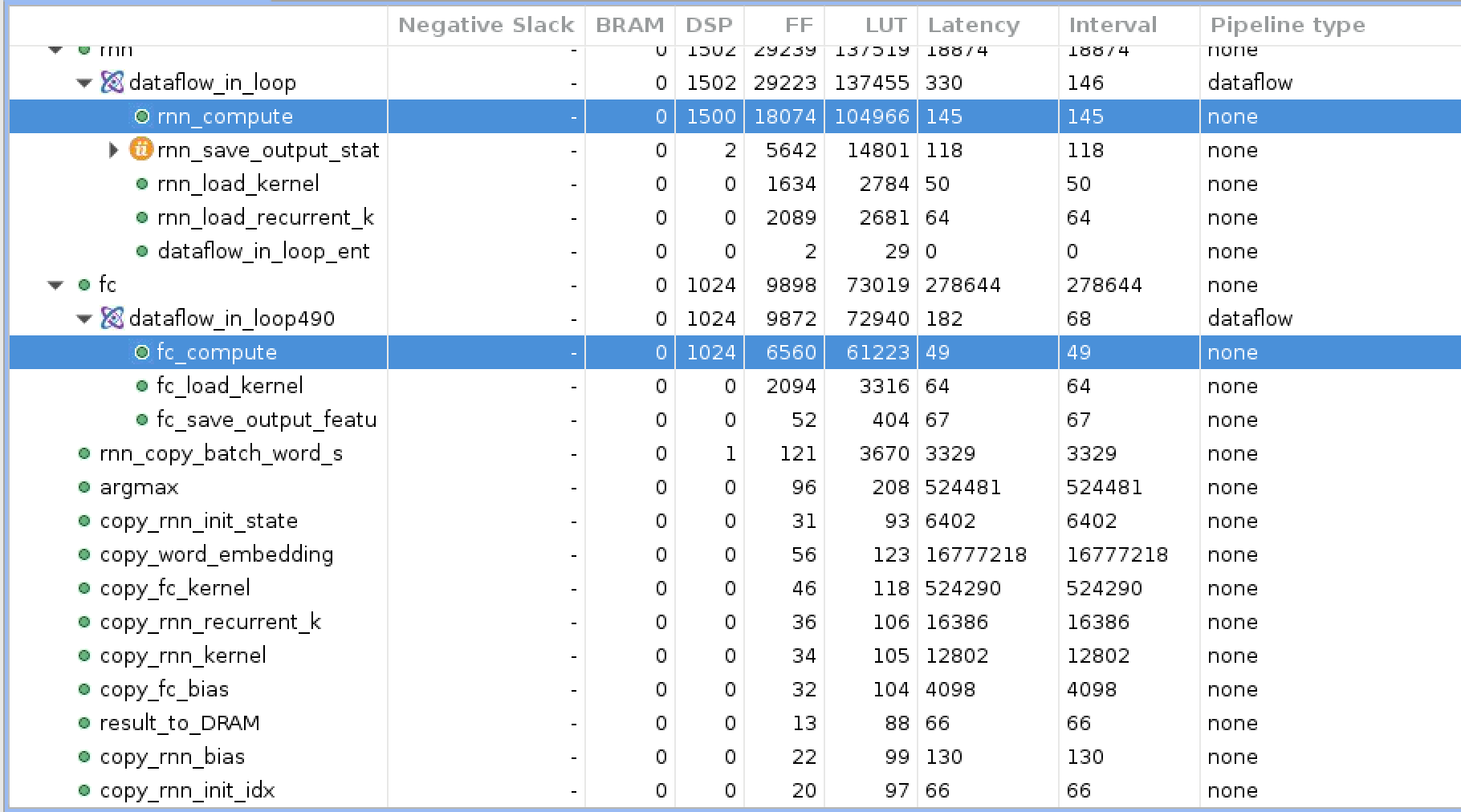
rnn save中：

#pragma HLS UNROLL factor = **4 -> 8**

#pragma HLS PIPELINE







## 3. 改回1000个iter Focus on fc 800M

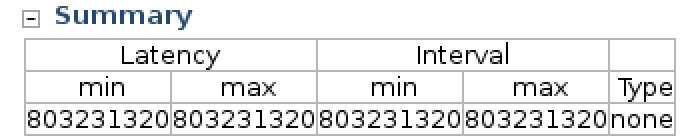
rnn 20k, fc 280k CC -> optimize fc first

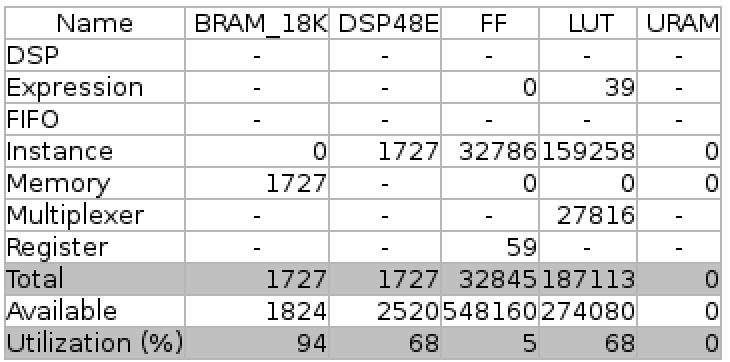
#define FC\_COMPUTE\_UNROLL 8

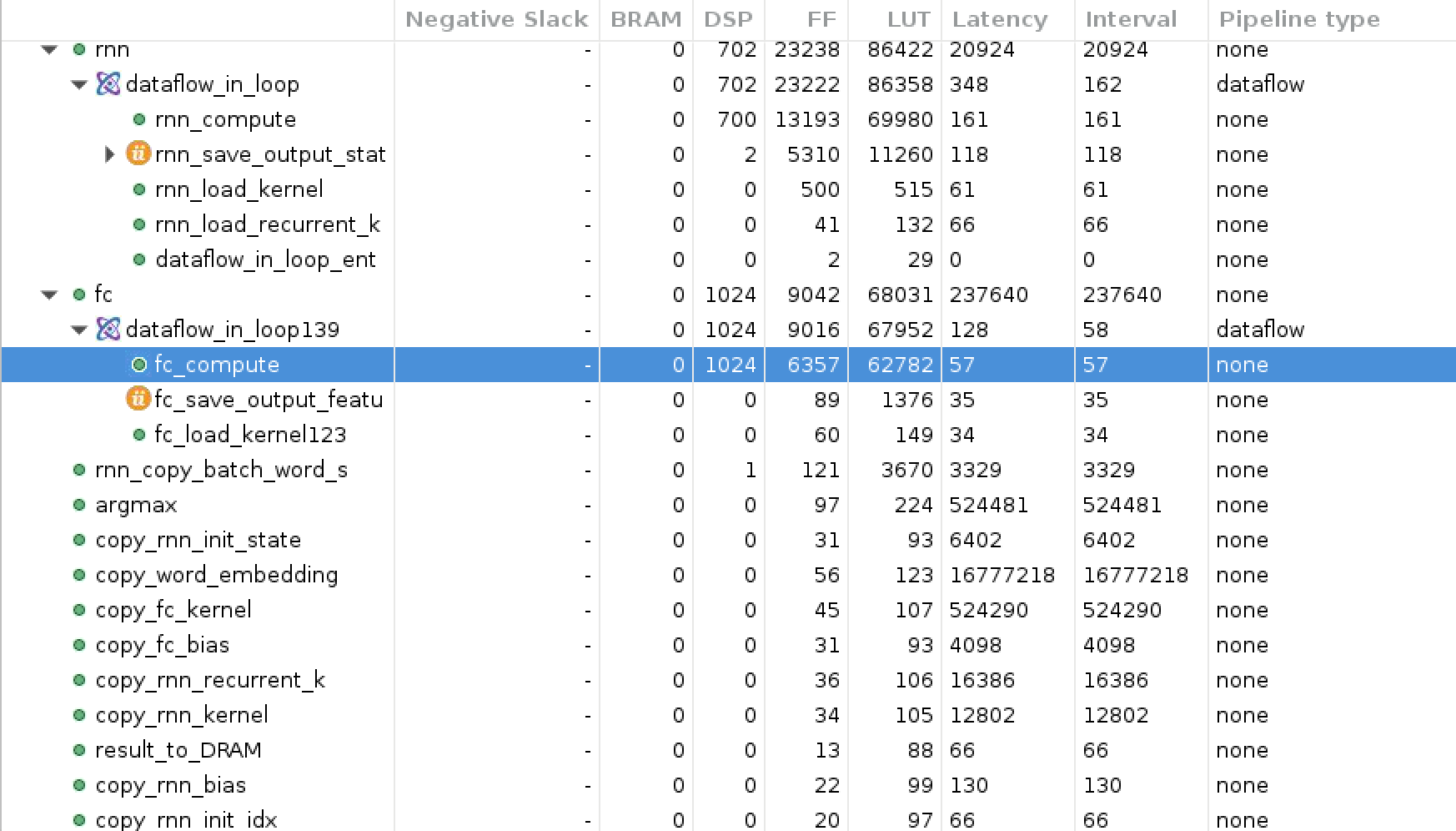
FDATA\_T local\_reg[FC\_COMPUTE\_UNROLL][FC\_INPUT\_SIZE];

#pragma HLS array\_partition variable=local\_reg cyclic factor=32 dim=2

#pragma HLS array\_partition variable=local\_reg cyclic factor=8 dim=1



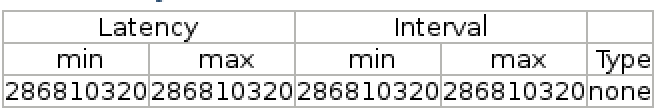


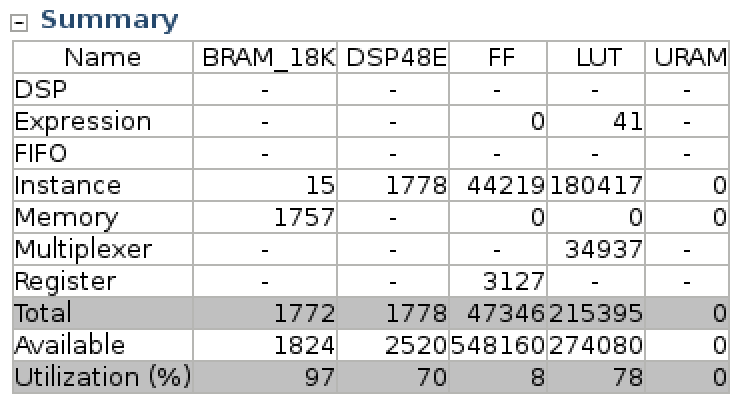


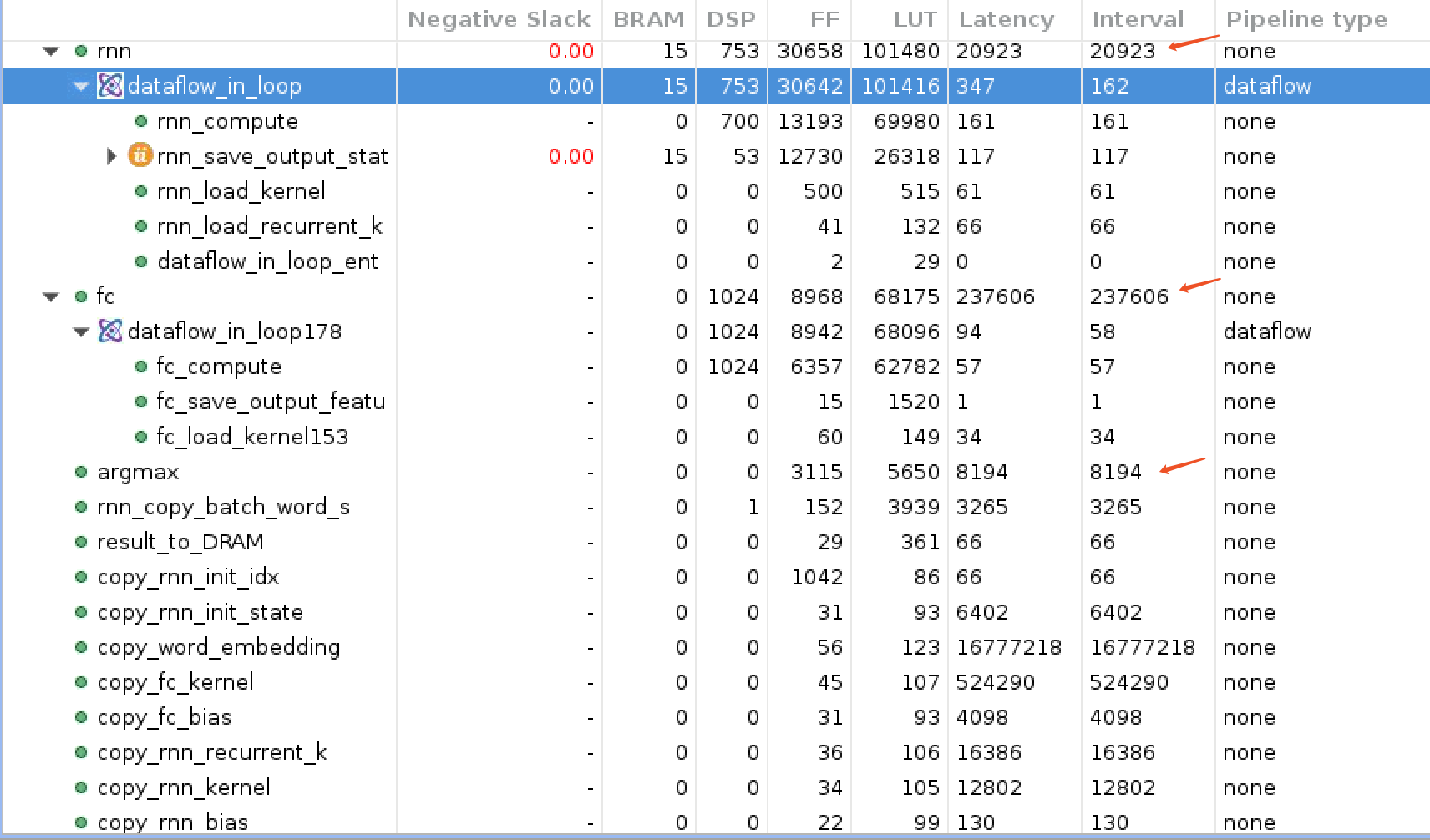
## 4.optimize argmax / float tanh 286M

unroll argmax = 64

改回 float tanh



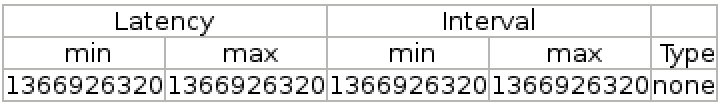


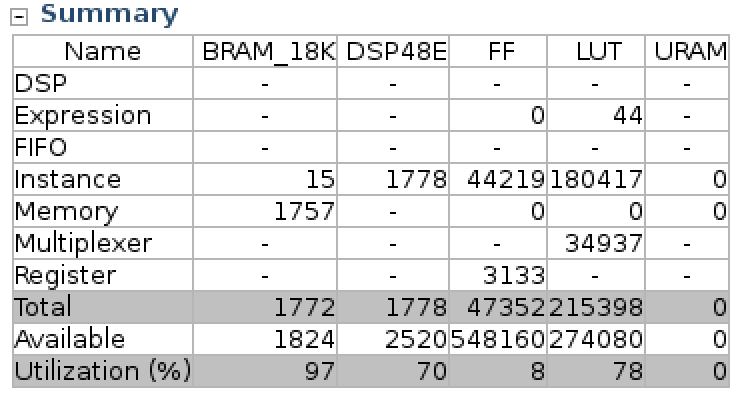


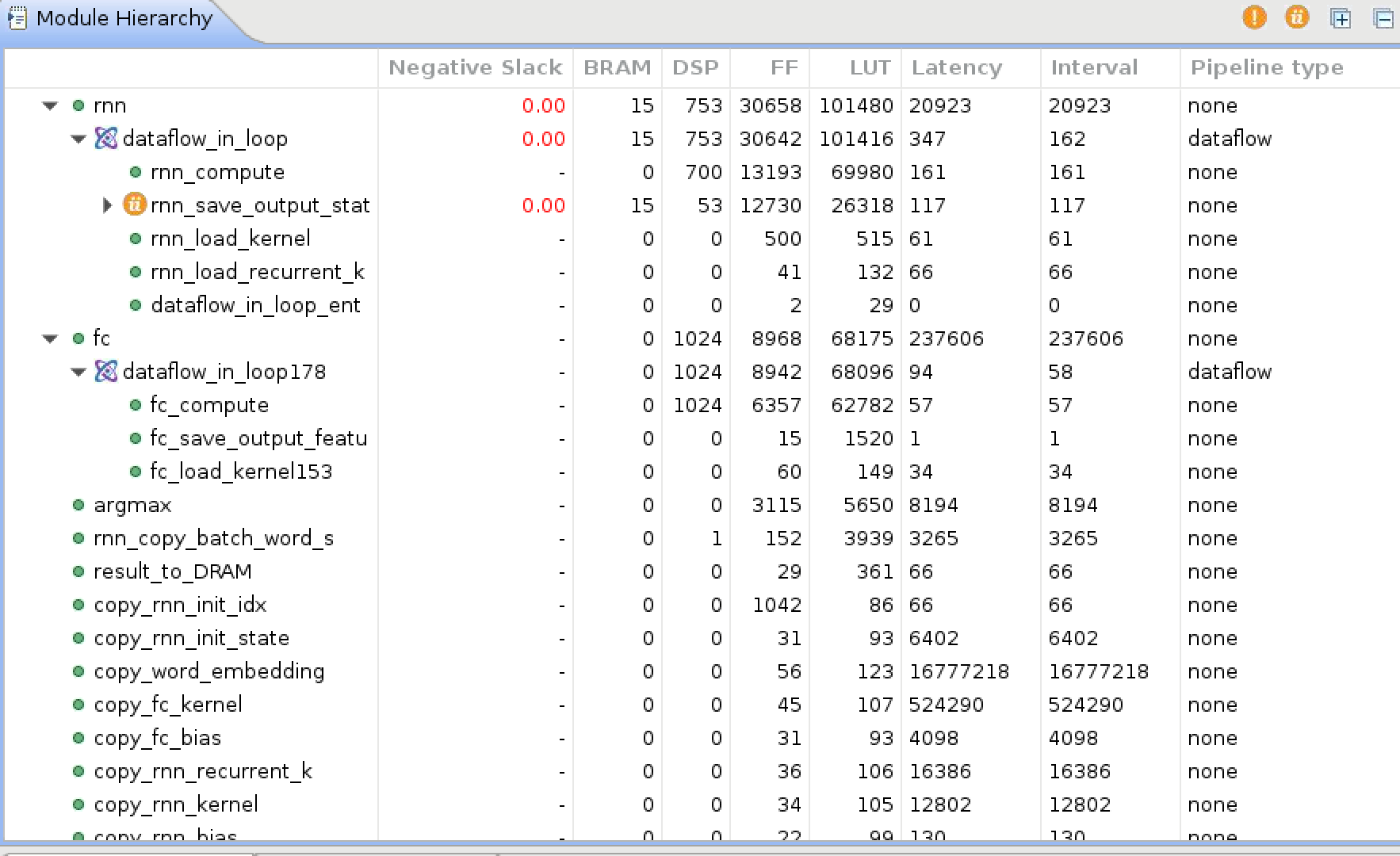
copy部分占了约18M -> x10 = 180M，not negligible

solution: more steps

## 5.改回5000step 1.36B

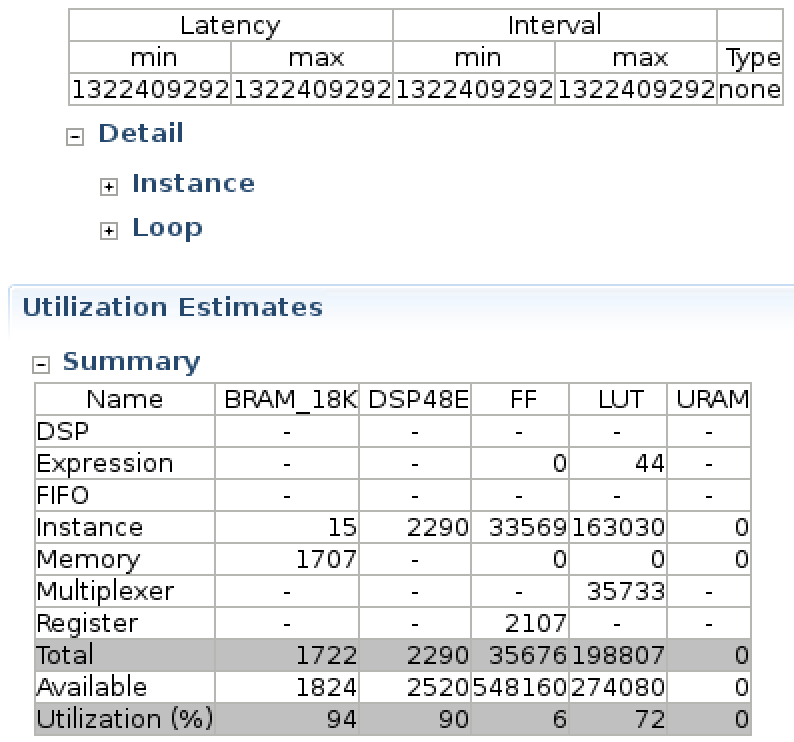


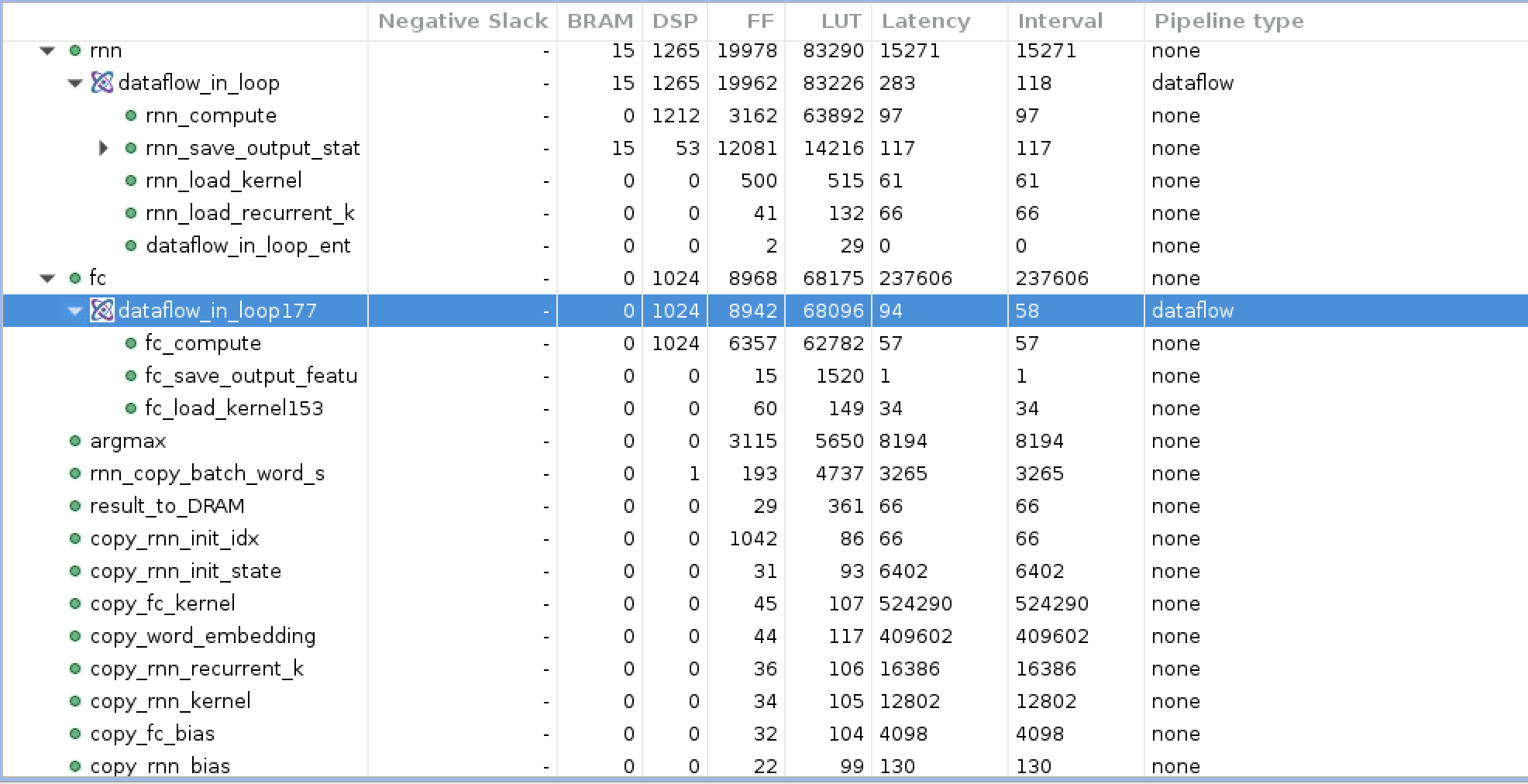
ß



## 6.recompile

RNN input state unroll **100 -> 50，结果居然更快了？**

****



161 -> 97 CC