SHANDONG UNIVERSITY

# Development of DNN algorithm on Versal AI-engine

Zhao-Zhi Liu

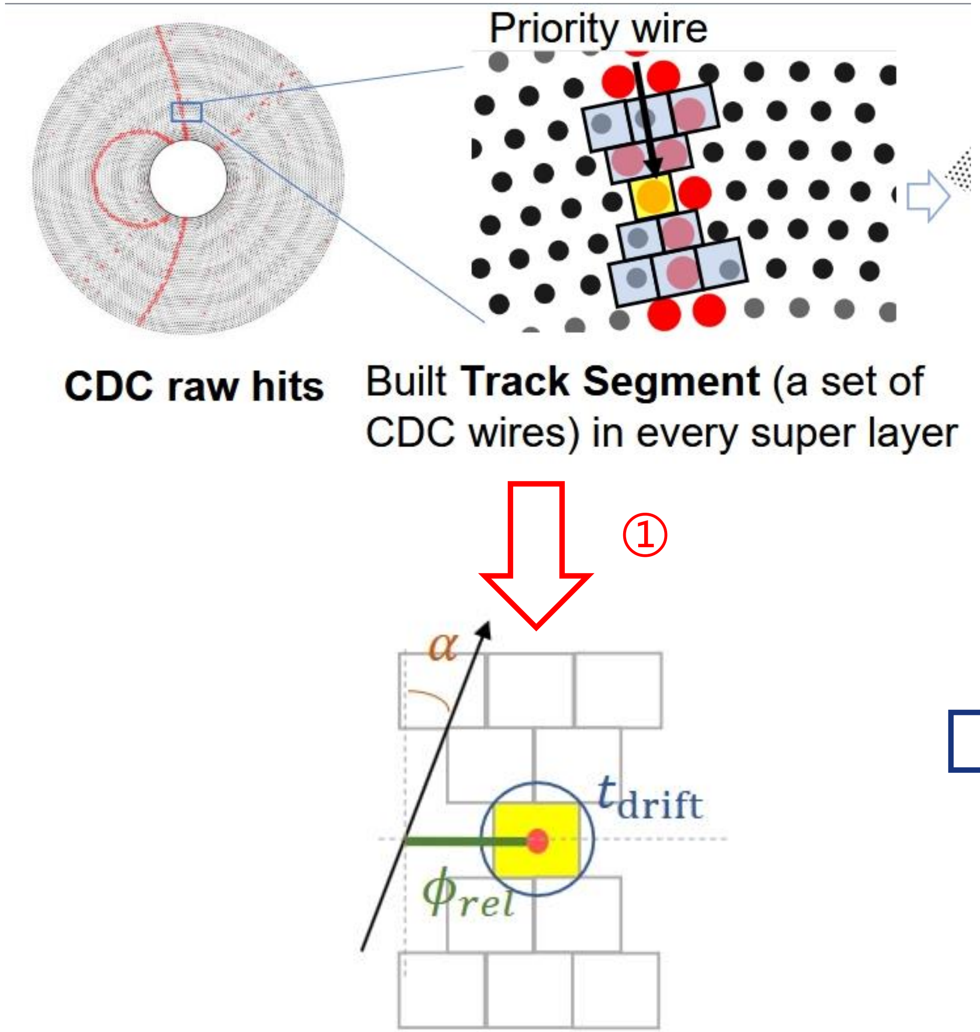
Institute of Frontier and Interdisciplinary Science, Shandong University

Belle II TRG parallel

2025/2/24

# Contents

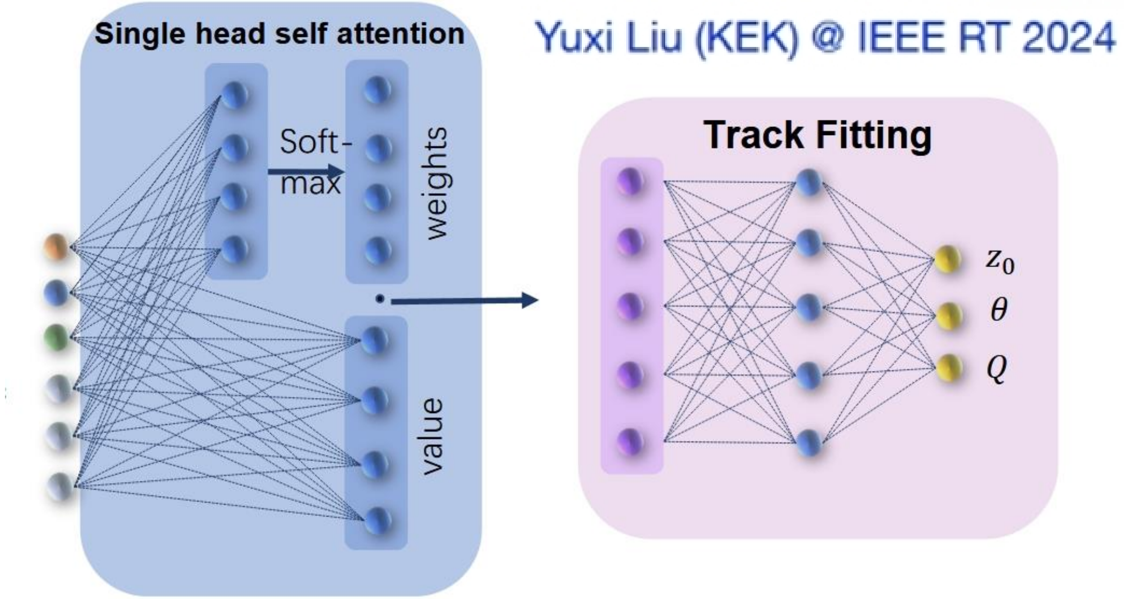
# DNN for tack reconstruction




① Each Track Segment(TS) has a set of α, t and Φ

② Variables in all 9 super layers input into the deep neuro-network

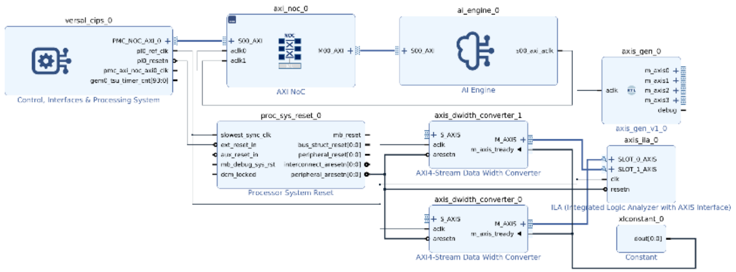
②

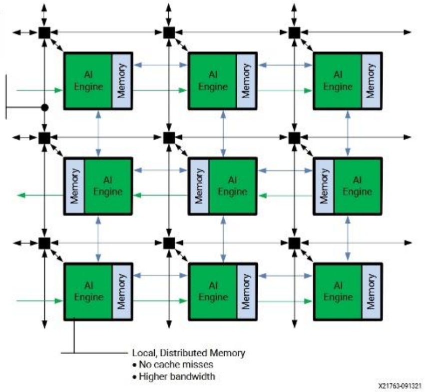


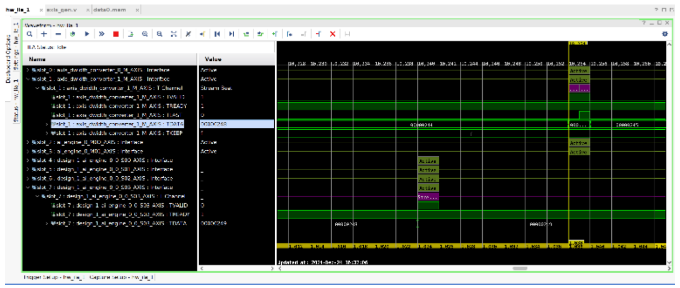

**Input Drift time $t_{drift}$, wires relative location $\phi_{rel}$, Crossing angle $\alpha$**

Output track vertex $z_0$, track $\theta$ and **signal/ background classifier output** $(Q)$

# Design flow





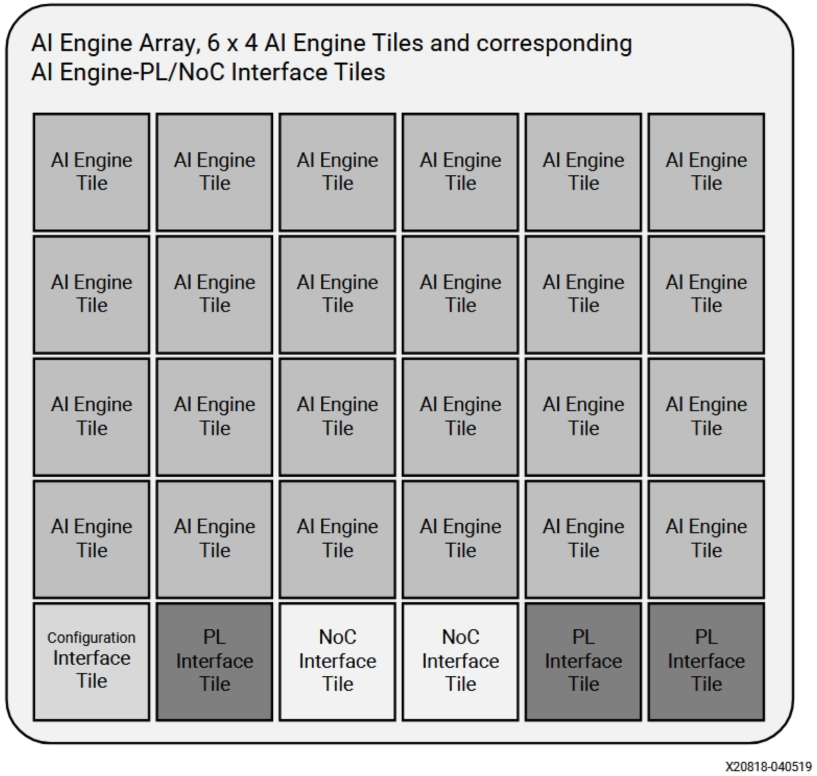



**Step1: Generate hardware platform in vivado(.xsa file)**

**Step2: Neural network deployment in AIE(.bin file)**

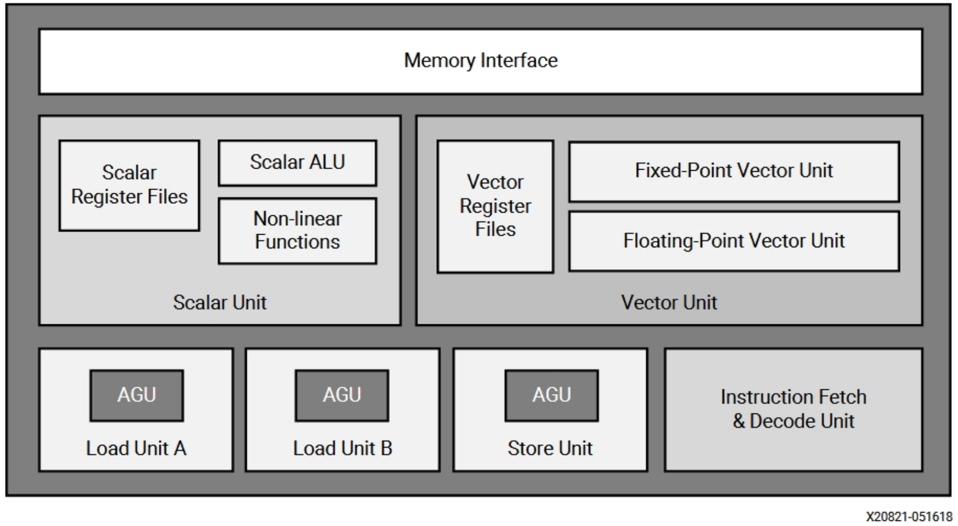
**Step3: Output in ILA**

# AI engine




AIE array

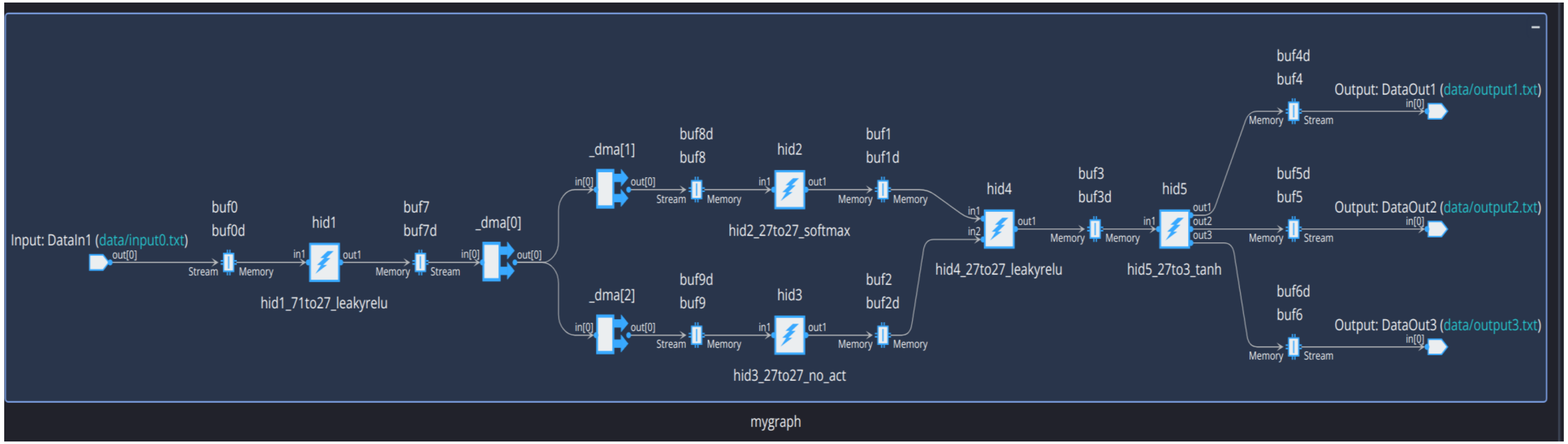

Tile

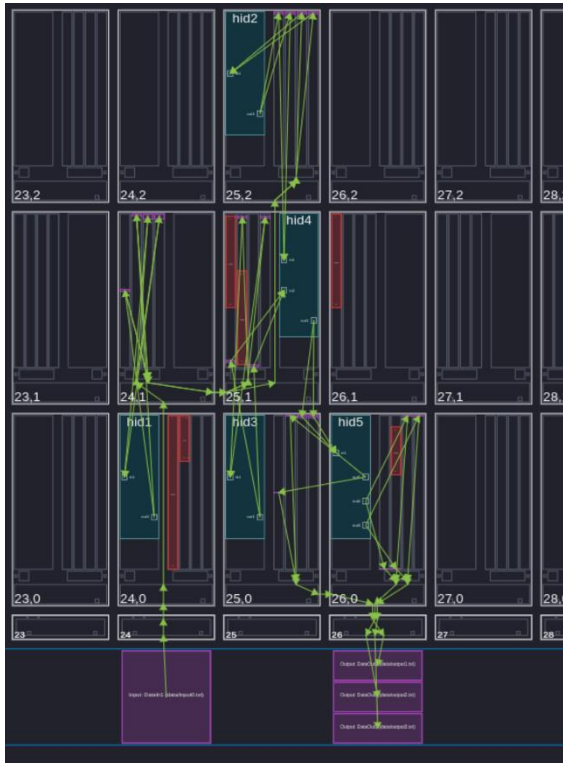



AIE

VCK190 has an AIE array of 400 tiles. Each tile has an AIE which has vector unit and scalar unit for vector algorithm and scalar algorithm.

# Baseline development on VCK190

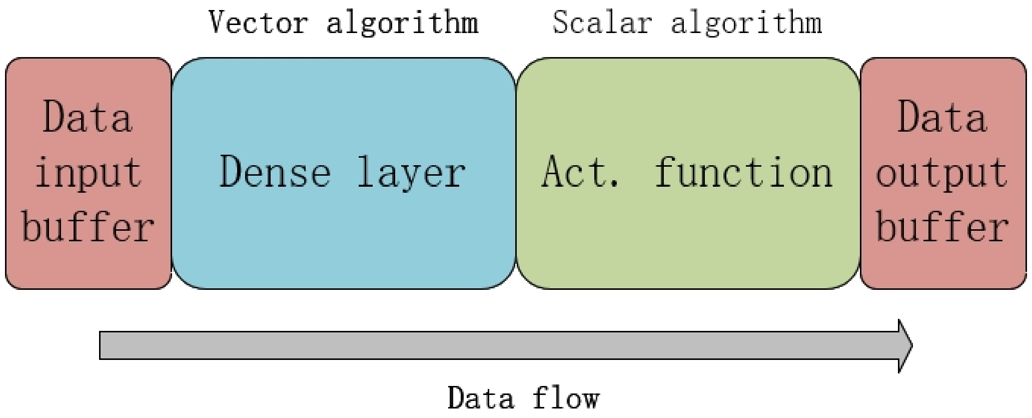


1.One kernel represent for one hidden layer and its input dense layer and mapped inside one single tile.

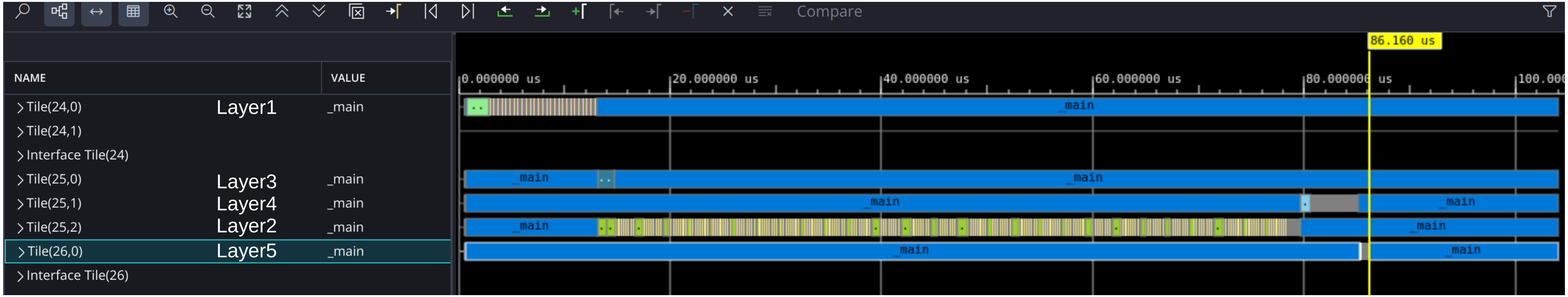
2.The in/output of the kernels are buffer types, so one buffer between two kernels.

3.All kernels inside one graph.




Each kernel's logic

# Simulation latency



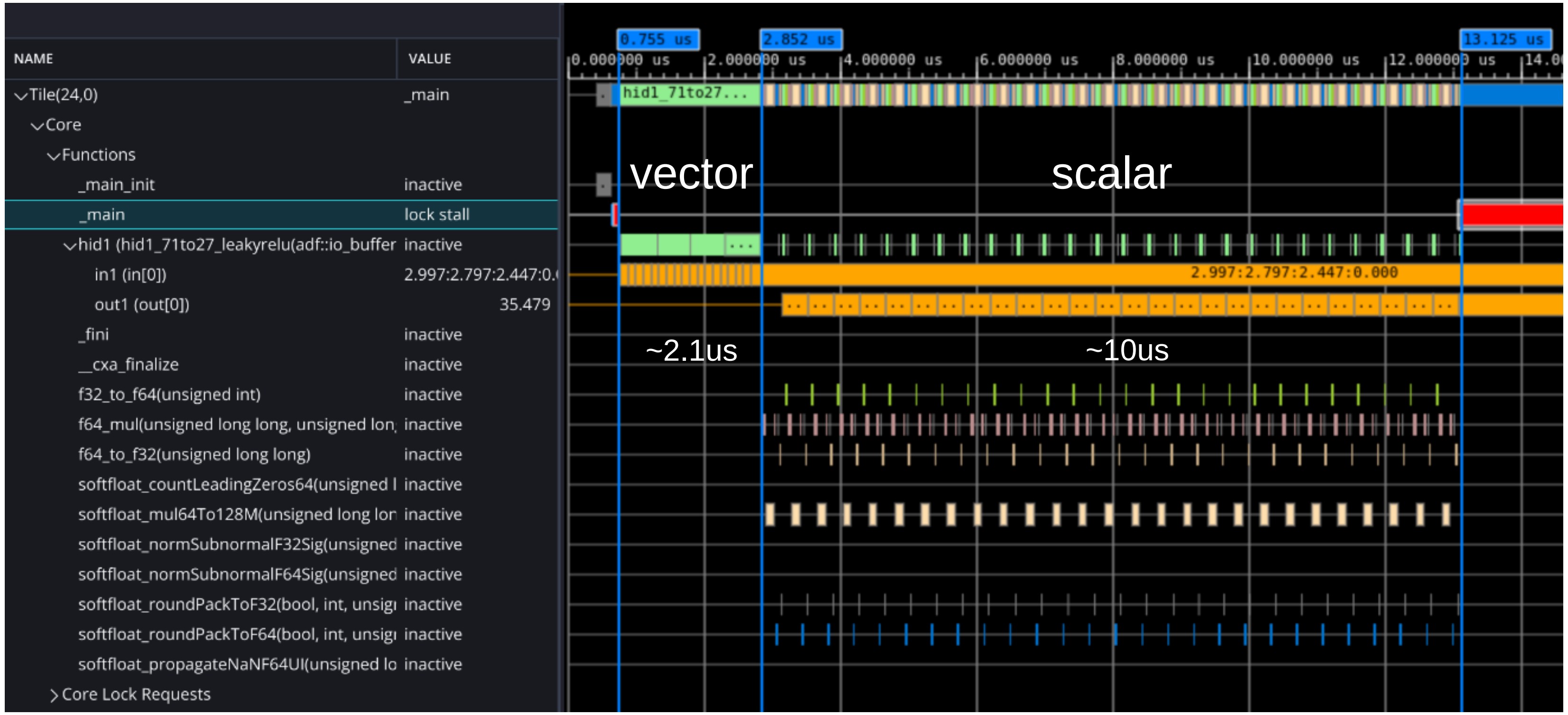

*Non-blue modules represent operational status

| | Layer1 (leakyrelu) | Layer2 (softmax) | Layer3 (no act.) | Layer4 (leakyrelu) | Layer5 (tanh) | total |
|---|---|---|---|---|---|---|
| Latency | ~12us | ~66us | ~1.5us | ~5.5us | ~0.9us | ~86us |

*Buffer port used, which need time to ready before usage.

SHANDONG UNIVERSITY

# Simple analyze on the layer 1

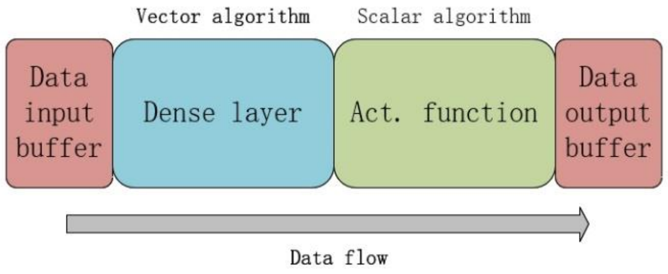



Vector algorithm cost about 2.1us

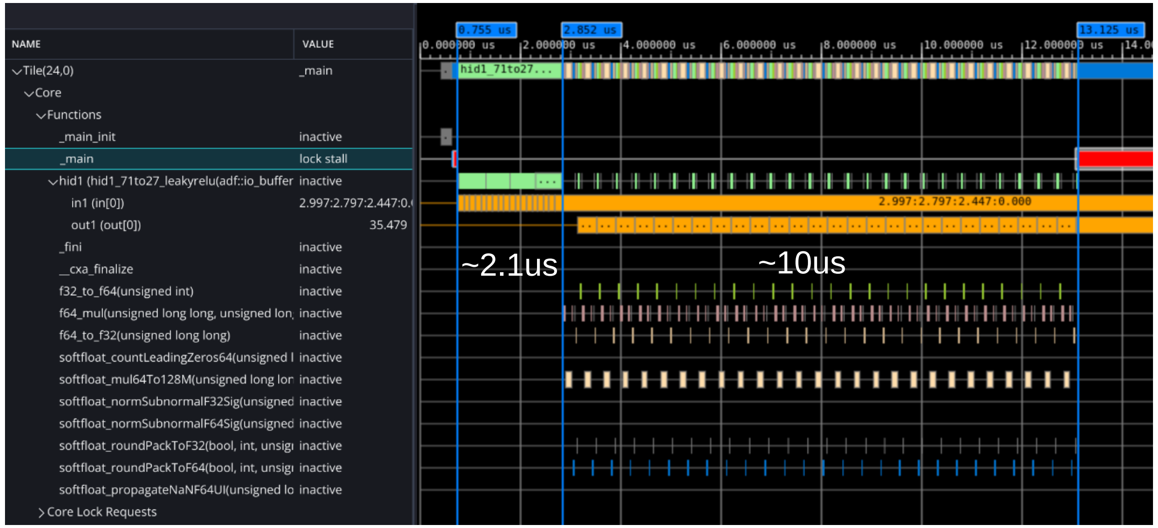
Scalar spends about 10us

Layer one has **71 inputs** and **27 outputs**

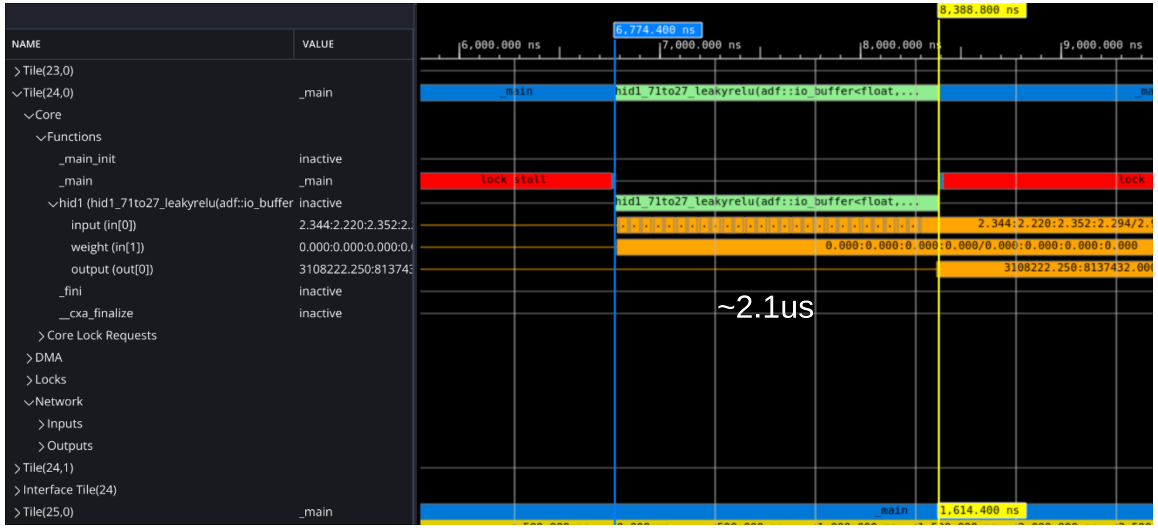
In dense layer, 71*27=1917 times multiplications .(vector part)
In act. function, 27 times compare and multiplications. (scalar part)



# Vectorize optimization comparison




Before opt.




After opt.

```
*outIter1++ = act_leakyrelu(sum[i]);


inline float act_leakyrelu(float x ) {

  if (x < 0) {return 0.01*x;}
  else {return x;}
```

```
aie::vector<float,32>sum2 = aie::mul(sum,0.01f);
aie::vector<float,32>act_out = aie::select( sum, sum2, aie::ge(sum, 0.0f));
```

## c/c++

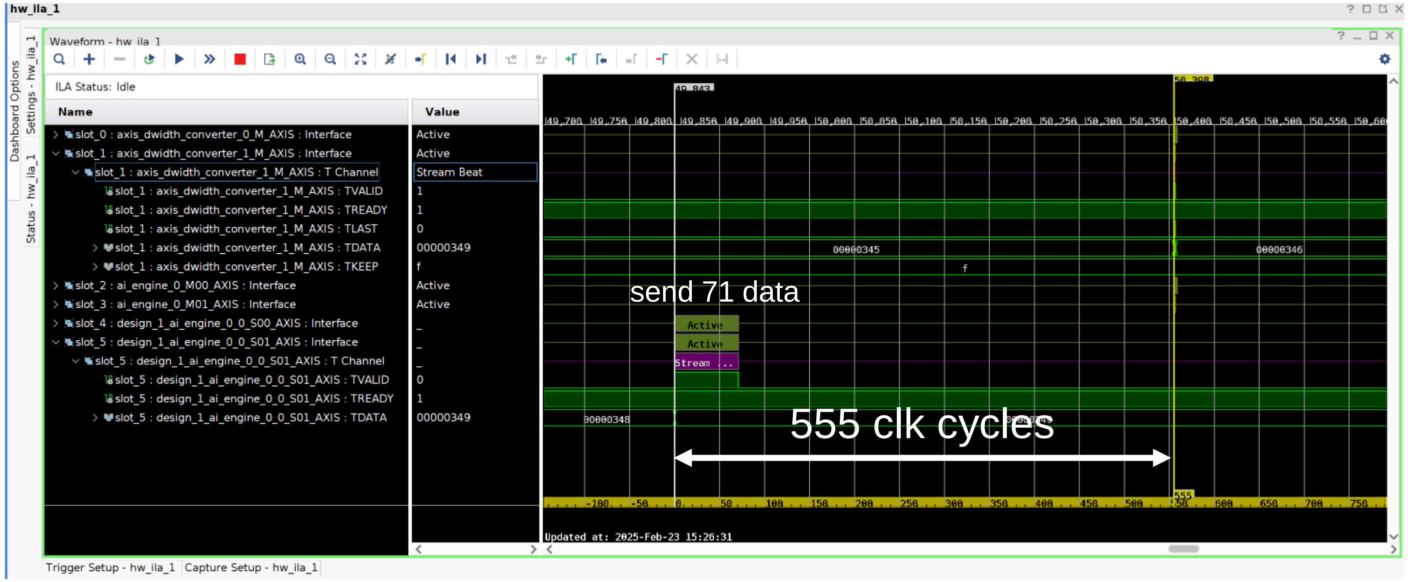
## aie

# Optimization result

| | | Layer 1 (leakyrelu) | Layer 2 (softmax) | Layer 3 (no act.) | Layer 4 (leakyrelu) | Layer 5 (tanh) | total |
|---|---|---|---|---|---|---|---|
| Before opt. | Latency | ~12us | ~66us | ~1.5us | ~5.5us | ~0.9us | ~86us |
| After opt. | Latency | ~2.1us | ~1.3us | ~1.5us | ~0.9us | ~0.2us | ~5us |

Abandon all the scalar algorithm, all use vector algorithm instead.

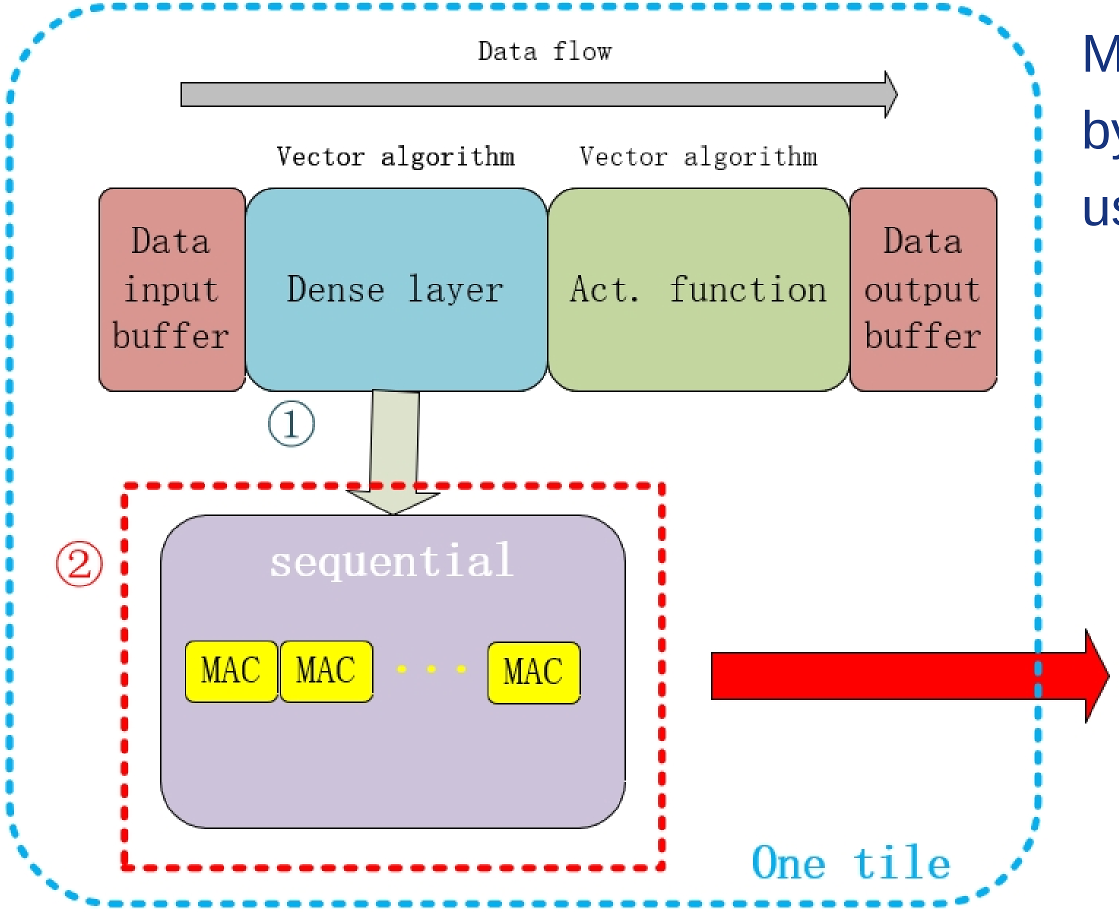
# ILA result



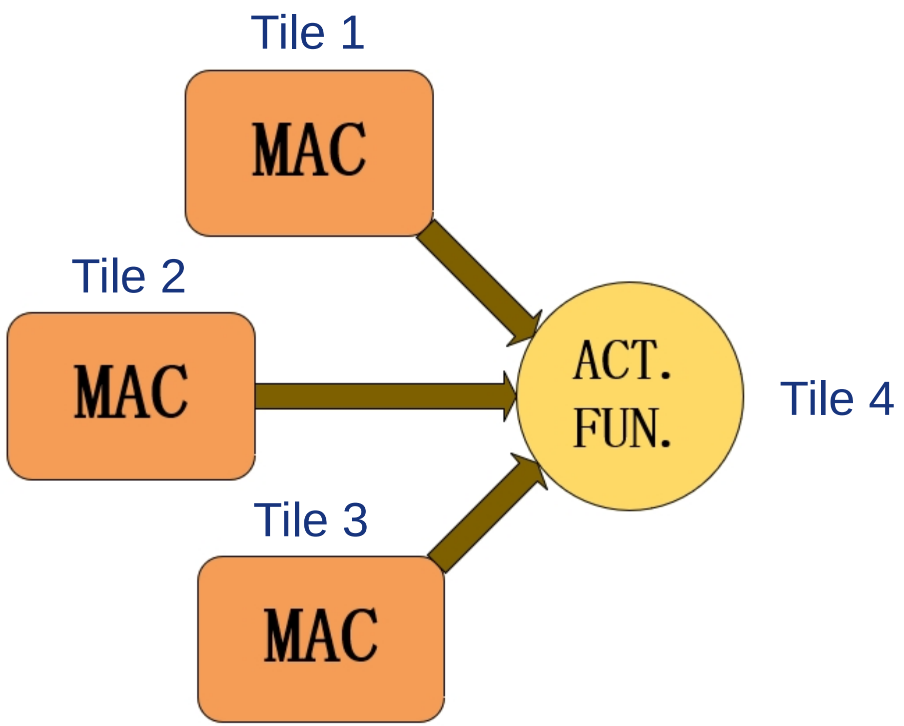


Total **555 clk cycles** one instance. Clk period **10ns**. So the latency in ila is **5.55us**.

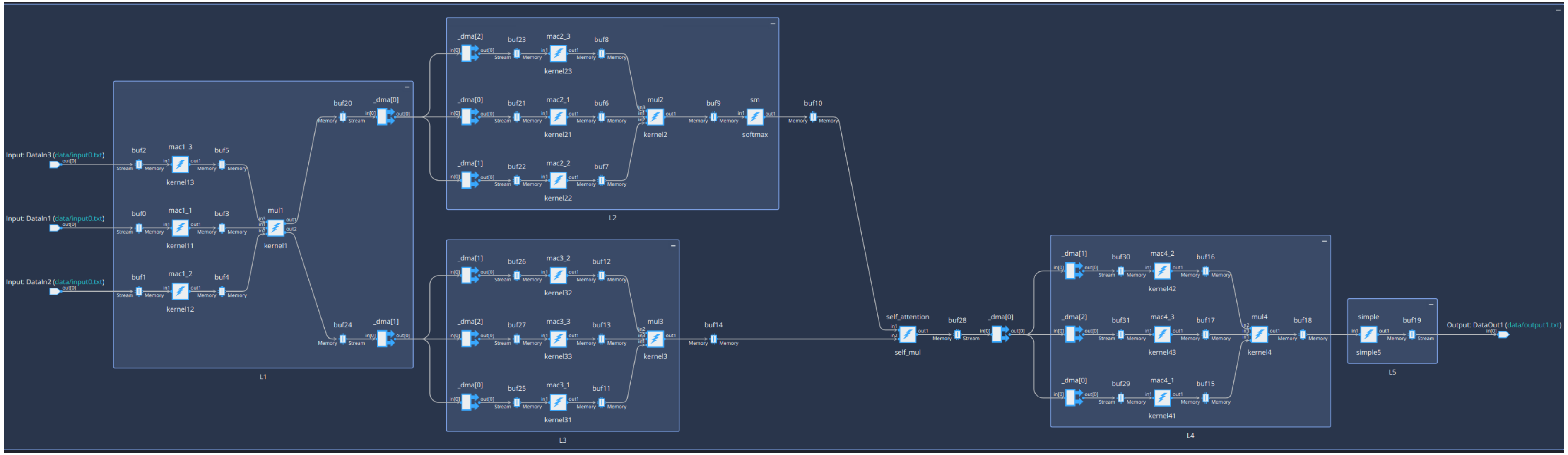
# Location optimization

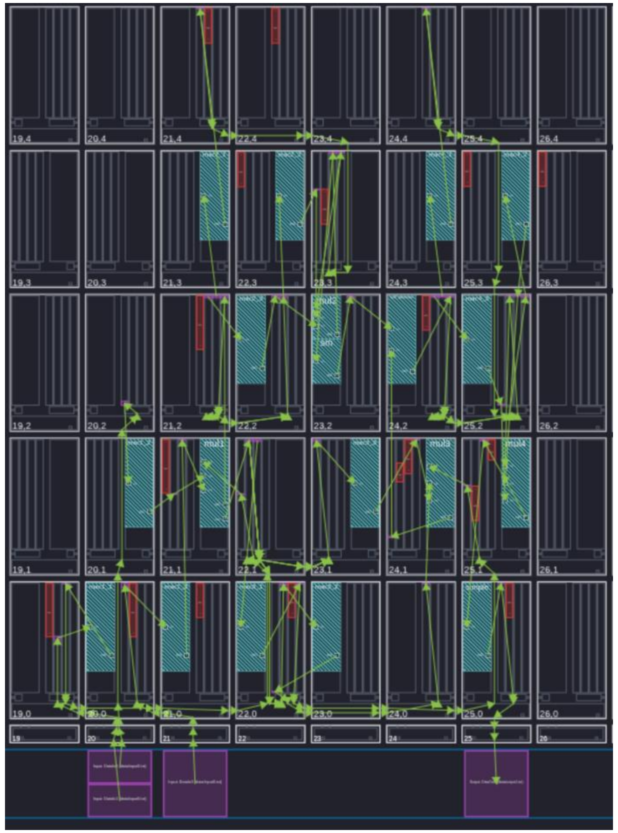


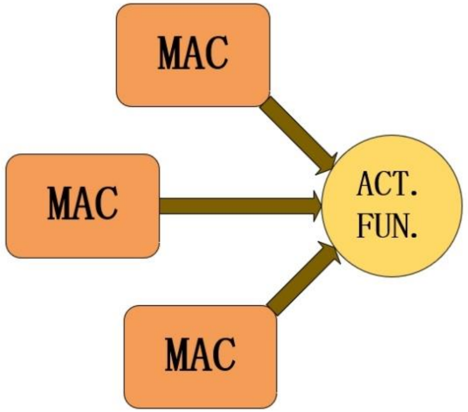

Make the mac operation from **sequential** to **parallel** by disassembling the mac operation into **three aie tiles** using **location constraint**.



# Location optimization








1.one **graph** represents **one layer.**
2.**three mac kernels** form the **dense layer.**
3.**one act. kernel** forming the **hidden layer.**

# Location optimization

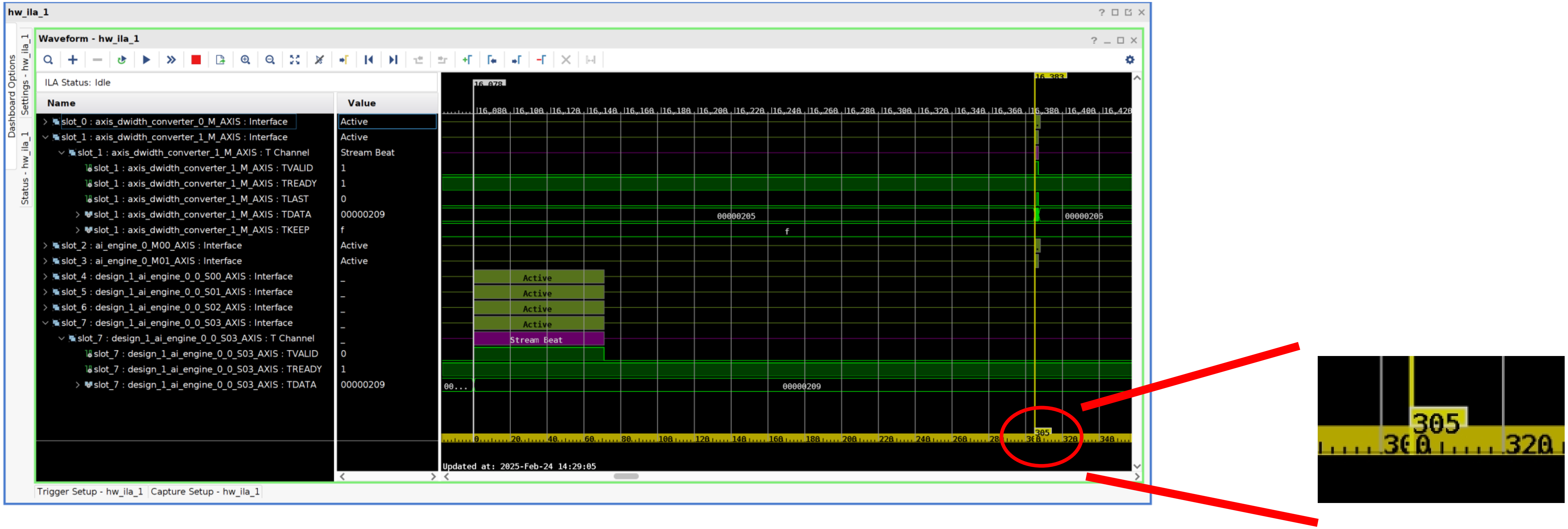
| Latency | Layer 1 (leakyrelu) | Layer 2 (softmax) | Layer 3 (no act.) | Layer 4 (leakyrelu) | Layer 5 (tanh) | total |
|---|---|---|---|---|---|---|
| No opt. | ~12us | ~66us | ~1.5us | ~5.5us | ~0.9us | ~86us |
| Vectorize opt. | ~2.1us | ~1.3us | ~1.5us | ~0.9us | ~0.2us | ~5us |
| Location opt. | ~479ns | ~931ns | ~327ns | ~404ns | ~100ns | ~2.1us |

Note:

① Due to the location constraint, some links between kernel ports have a longer way to move which brings **more delay in data moving**.

So the total latency is a bit larger than kernel combined latency.

# ILA result




Total **305 clk cycles** one instance. Clk period **10ns**. So the latency in ila is **3.05us**.

# Summary

① In **baseline** version, **vector algorithm** is used in **dense layer** while **scalar algorithm** is used in **act. function**.

And total latency in simulation is about **86us**. In this part, it finds out that vector algorithm is fast.

② In the **version 2** with optimization, **act. function** uses **vector algorithm** for replacement.

And total latency in simulation is about **5us**. Ila shows the latency is about **5.5us**.

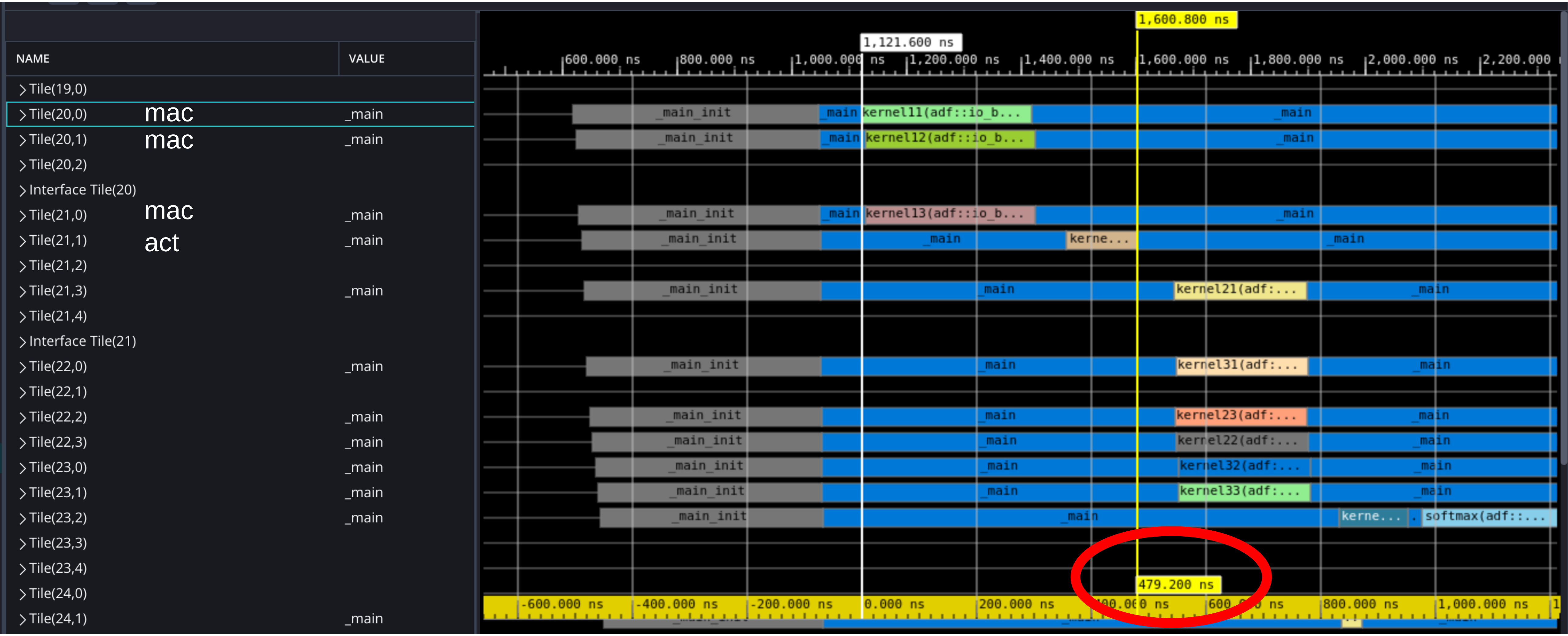
③ In the **version 3** with location optimization, one layer is divided into **multiple** ai-engines instead of one to improve the parallel execution to lower the latency.

Simulation shows the latency is about **2.1us**. Ila shows the latency is about **3.05us**.

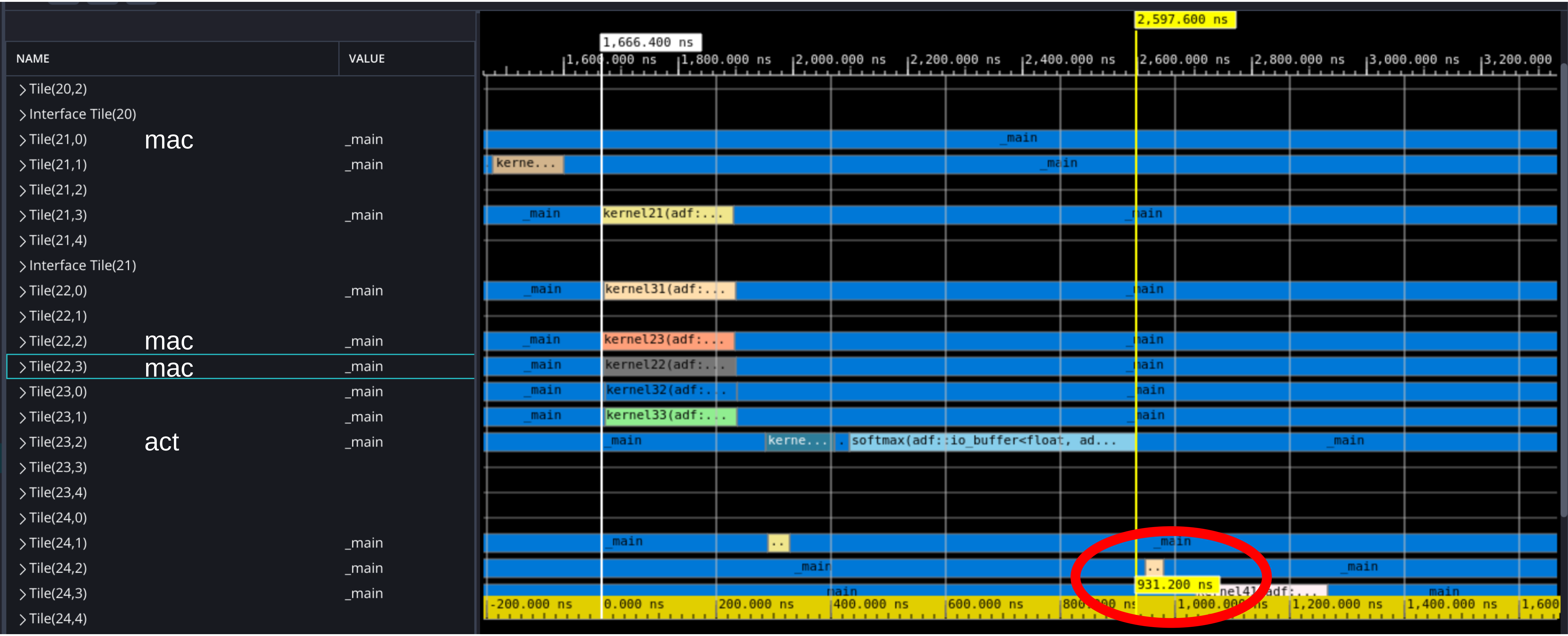
# *END*

# BACK UP

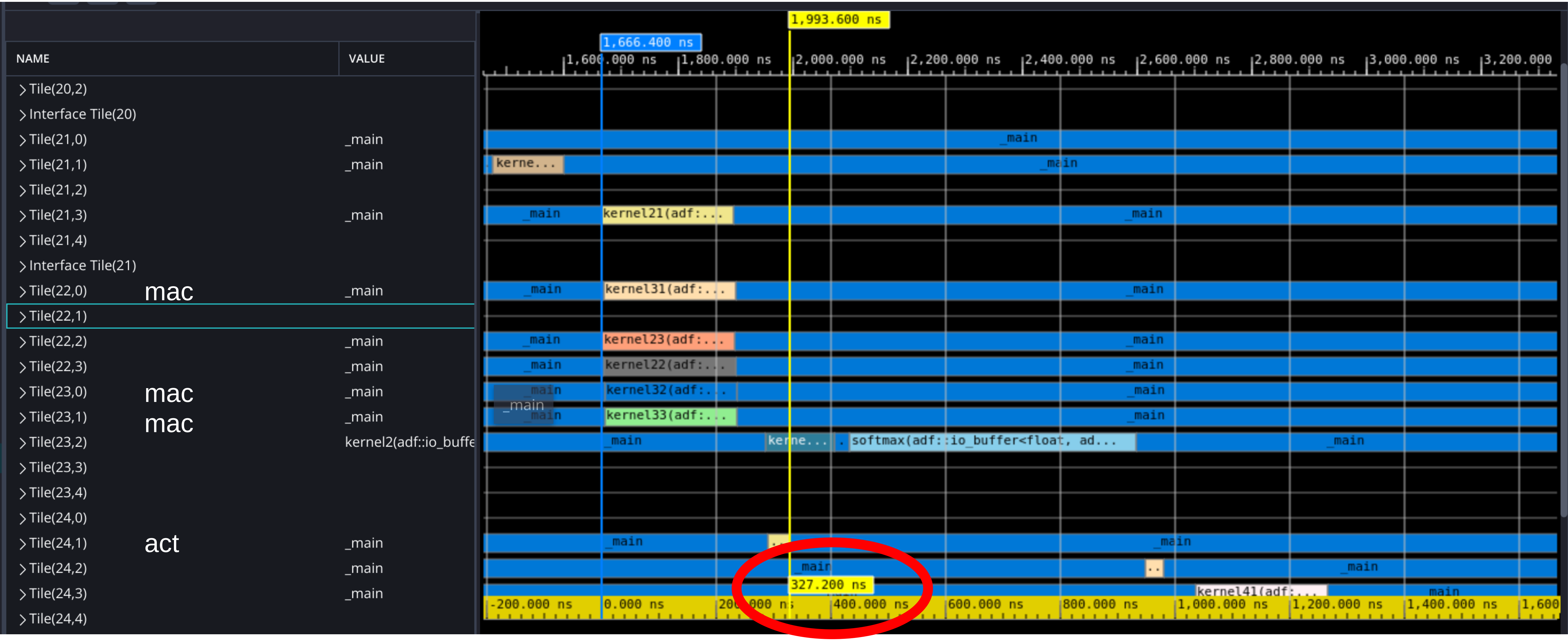
# Layer1 latency



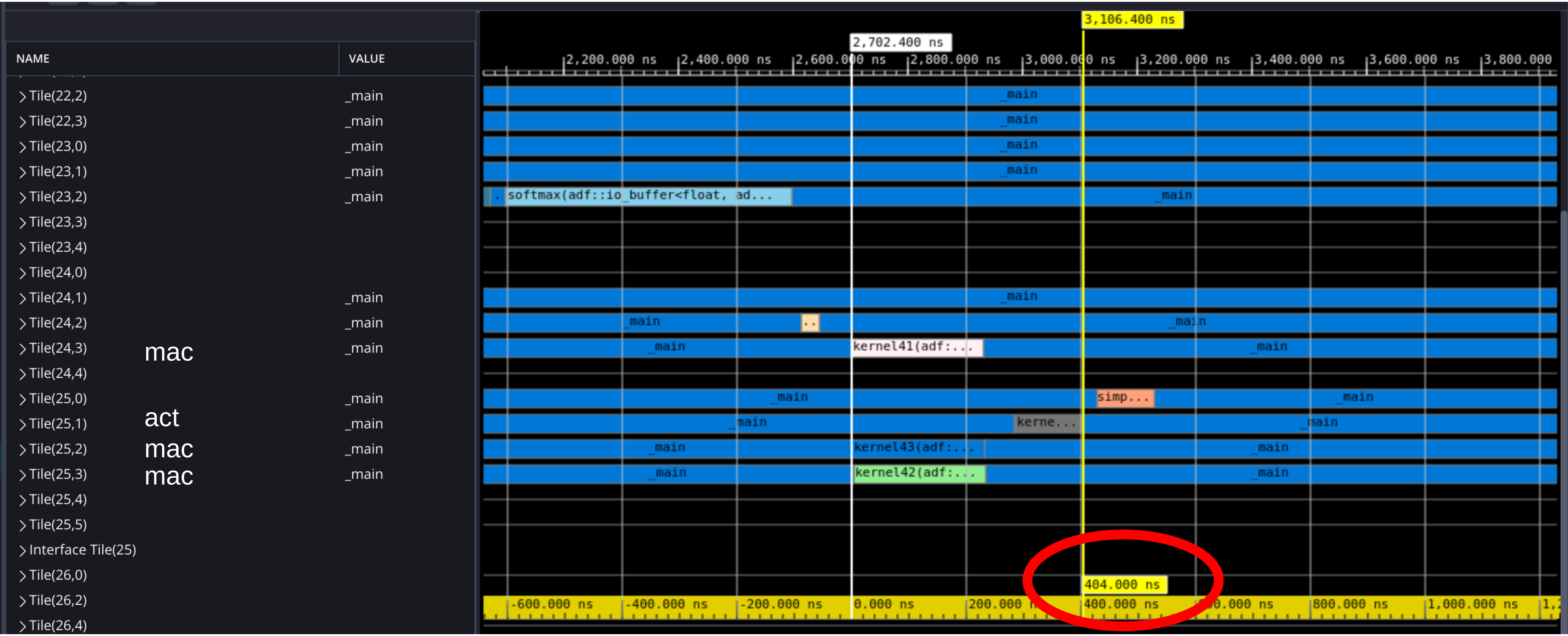
# Layer2 latency



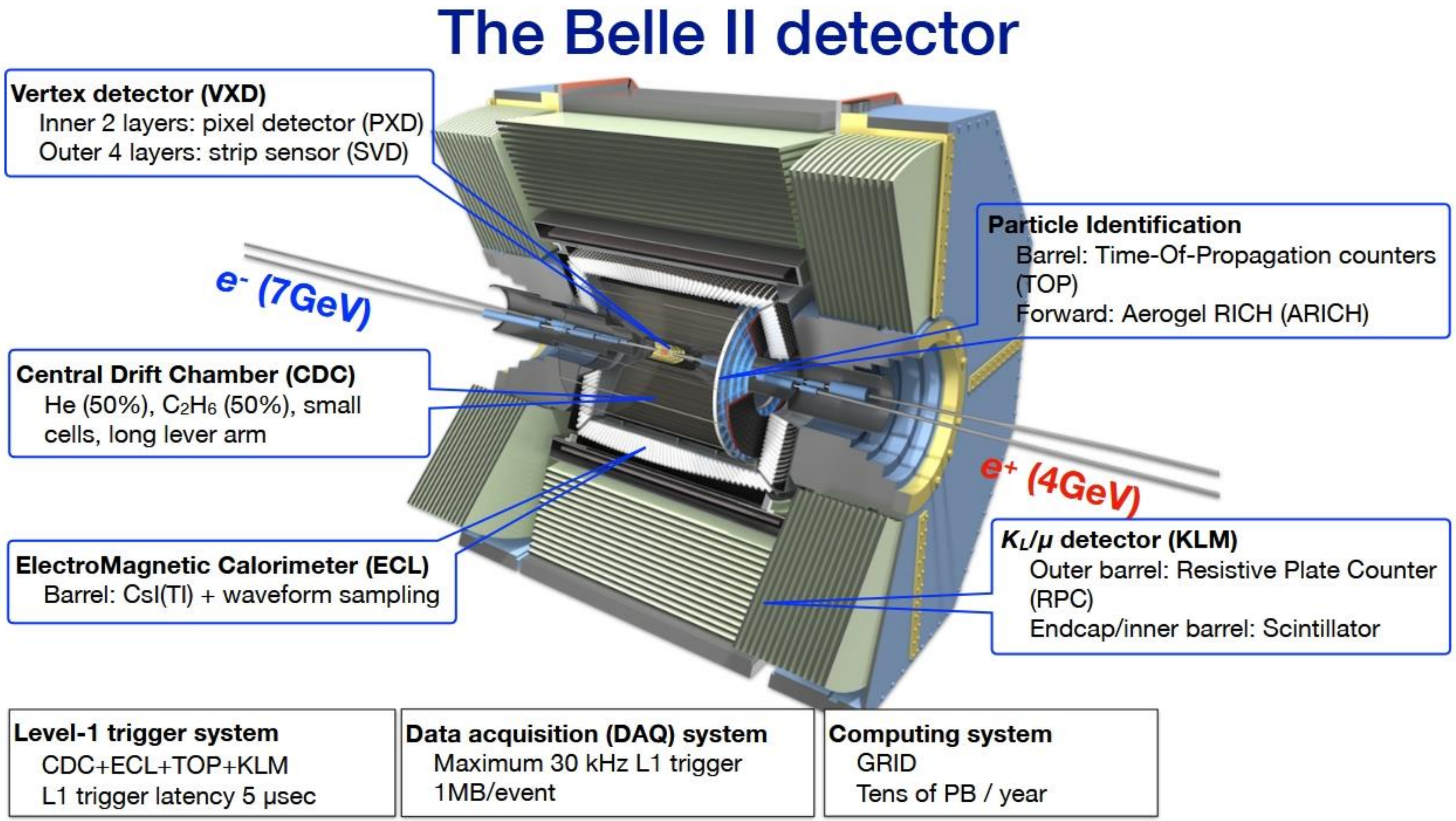
# Layer3 latency



# Layer4 latency



# The Belle II detector




- Tracking: Vertex detectors and CDC
- particle identification: TOP and ARICH
- Calorimeter: ECL.
- KL and muon detector.

First level (L1) trigger, High level trigger (HLT) and DAQ.

# Level one trigger

- CDC, ECL: main triggers for tracks and clusters
- KLM: trigger muon
- TOP: event timing
- GRL: matching of sub-triggers
- GDL: final trigger decision