NYCU-EE IC LAB - Spring 2022

Lab04 Exercise

Design: Convolution Neural Network

Data Preparation

1. Extract test data from TA's directory:

   % tar xvf ~iclabta01/Lab04.tar

2. The extracted LAB directory contains:

   a. **00_TESTBED**

   b. **01_RTL**

   c. **02_SYN**

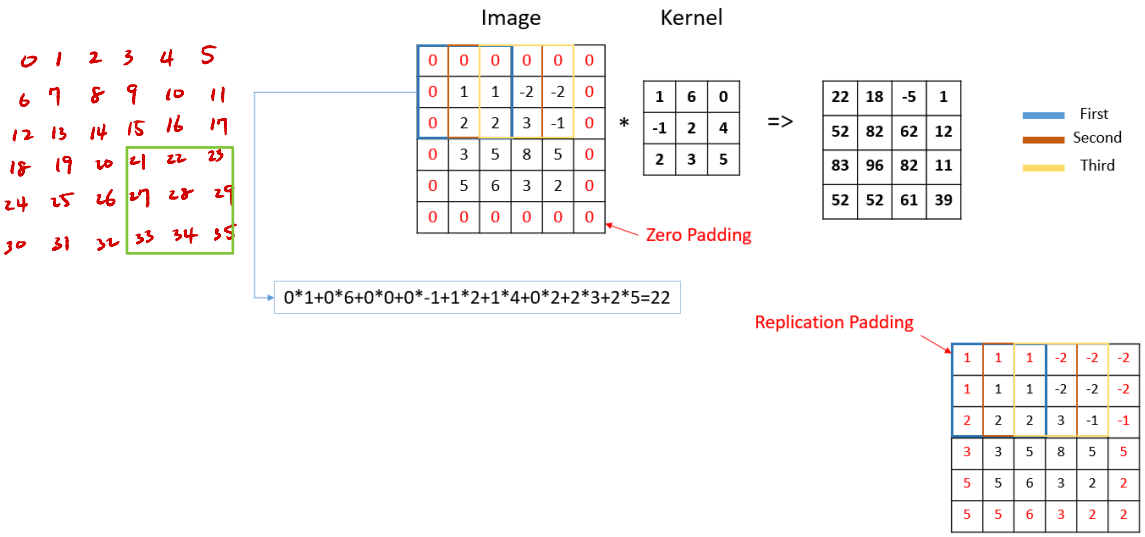
   d. **03_GATE**

Design Description

Convolution Neural Networks (CNN) is a class of artificial neural networks that has become dominant in various computer vision tasks, is attracting interest across a variety of domains, including radiology. CNN is designed to automatically and adaptively learn spatial hierarchies of features through back propagation by using multiple building blocks, such as convolution layers, pooling layers, and fully connected layers.

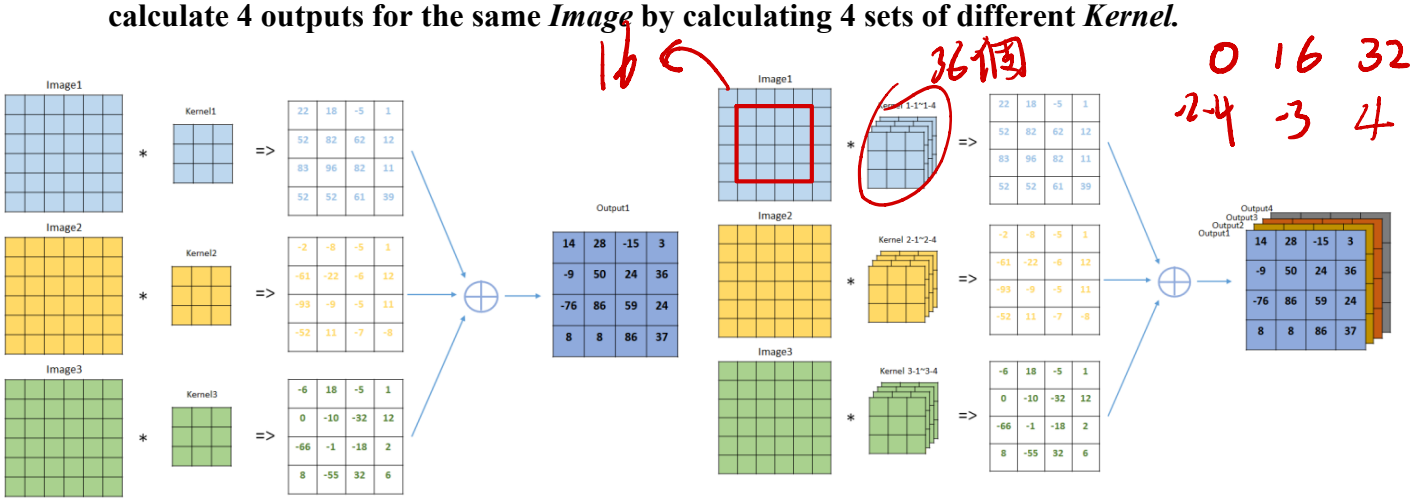
In this exercise, you are asked to use the "Convolution" method, which is a dot product by taking every pair of pixels and multiply, then sum all products. The image size will be 4x4, with stride = 1, and 3x3 kernel size. In order to prevent the convolutional graph from becoming smaller, we will padding at the outermost periphery. The Neural Networks will contain following parts: **Padding, Convolution, Activation function and Pixel Shuffle.**

Formula of Convolution : $R(x,y) = \sum_{x',y'} (Kernel(x',y') \times Image(x+x', y+y'))$



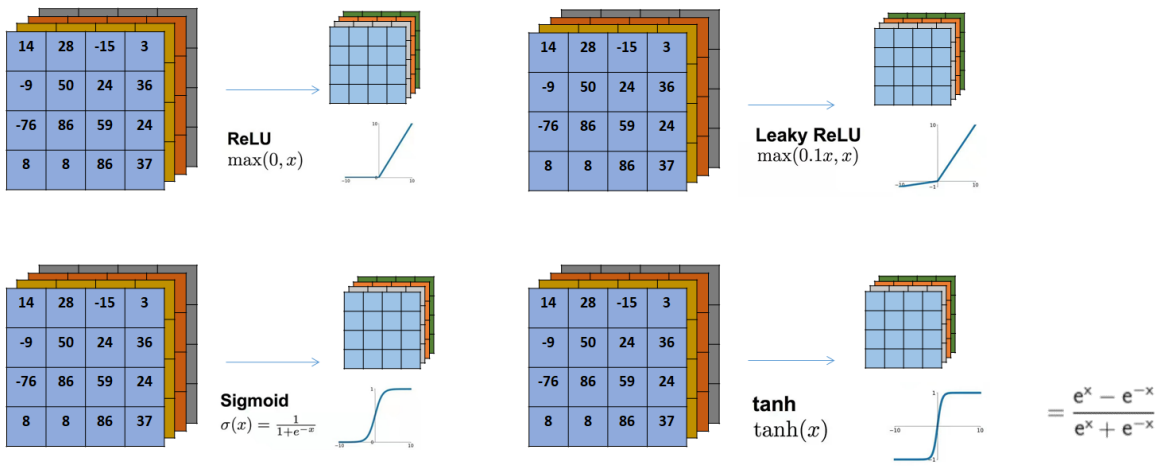
- Description

When in_valid_i, in_valid_k is high, you will get 3 different images: *Image1, Image2, Image3,* and their corresponding kernels: *Kernel1, Kernel2, Kernel3*. Also, these Kernels will give four different styles. **In other words, each set of *Kernel* will receive 9x4=36 same or different signals.** Before doing the convolution, in order to keep the size of the feature map consistent with the original image, you must perform **Replication Padding** or **Zero Padding** according to the information given by Opt. Then, you have to calculate the convolution from the image and its corresponding kernel and add up the same positions in the calculated three sets of results to get the first Output. The schematic diagram is the same as bellow. Please follow the same method to calculate Output2~Output4 for different Kernels. **In general, you must calculate 4 outputs for the same *Image* by calculating 4 sets of different *Kernel*.**

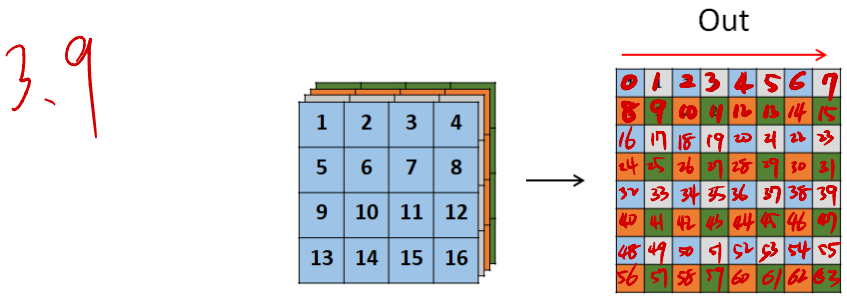


After getting the four Outputs, please do one of Relu, Leaky Relu, Sigmoid, tanh operations on the four Outputs according to the instructions given by **2bits of Opt Signals (You'll get Opt same as in_valid_o is high). The formula is in the figure below.**

2'b00	2'b01	2'b10	2'b11
Relu	Leaky Relu	Sigmoid	tanh
Replication Padding	Replication Padding	Zero Padding	Zero Padding



At last, you will ask to use **Pixel Shuffling** to enlarge the image before output the result. **Notice that the priority order of output image is from left to right and then from top to bottom.**



Inputs and Outputs

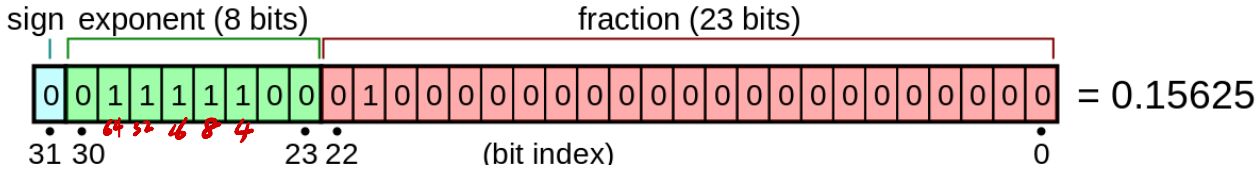
The following are the definitions of input signals

Input Signals	Bit Width	Definition
clk	1	Clock.
rst_n	1	Asynchronous active-low reset.
in_valid_i	1	High when image is valid.
in_valid_k	1	High when kernel is valid.
in_valid_o	1	High when option is valid
Image1,Image2,Image3	32	The input image signals. The arithmetic representation follows the IEEE-754 floating number format.
Kernel1,Kernel2,Kernel3	32	There are 9x4=36 signals for each Kernels. The arithmetic representation follows the IEEE-754 floating number format.
Opt	2	2'd0 : **Relu & {Replication}** 2'd1 : **Leaky Relu & {Replication}** 2'd2 : **Sigmoid & {Zero}** 2'd3 : **tanh & {Zero}**

The following are the definitions of output signals

Output Signals	Bit Width	Definition
out_valid	1	High when out is valid.
out	32	The forward result for current data. The arithmetic representation follows the IEEE-754 floating number format.

Each time you output the result, our pattern will check the correctness of it. Basically, if you follow the formula and use IEEE floating point number IP, you should get same result as our answer. **However, we release the constraint; you may have an error under 0.0009 for the result after converting to float number. This means that we will convert your output from binary format into real float number, and compare with our answer. Error will be calculated by '(golden-ans)/golden'. If the error is higher than the value, you will fail this lab.**



$$\text{sign} = +1$$
$$\text{exponent} = (-127) + 124 = -3$$
$$\text{fraction} = 1 + 2^{-2} = 1.25$$
$$\text{value} = (+1) \times 1.25 \times 2^{-3} = +0.15625$$

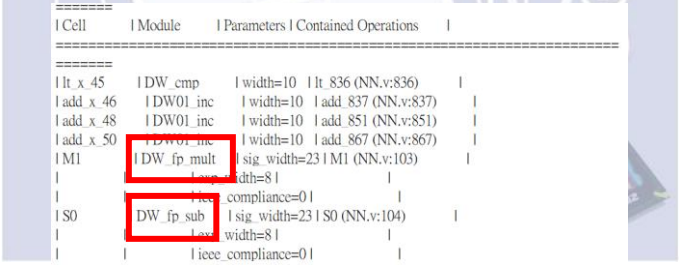
Binary form: 00111101101100101010000001000101
IEEE floating number: 0.08721975
nWave:

nWave will round the number for display, but the computation will not be affected. The following numbers are all from nWave, thus the computations are performed by IPs in Verilog.
※ e-02 = 10^{-2}

1. The input signal **Image1, Image2, Image3** are delivered for **16 cycles** continuously. When **in_valid_i** is low, input is tied to unknown state.
2. The input signal **Kernel1, Kernel2, Kernel3** is delivered for **36 cycles**. When **in_valid_k** is low, input is tied to unknown state.
3. The input signal **Opt** is delivered for **1 cycle** continuously. When **in_valid_o** is low, input is tied to unknown state.
4. All input signals are synchronized at negative edge of the clock.
5. The output signal **out** must be delivered for **64 cycles**, and **out_valid** should be high simultaneously.
6. The **in_valid_k** will come in **2 cycles** after **in_valid_i** is pulled down and **in_valid_i** will come in **2 cycles** after **in_valid_o** is pulled down.(**in_valid_o ➞ in_valid_i ➞ in_valid_k**)

Specifications

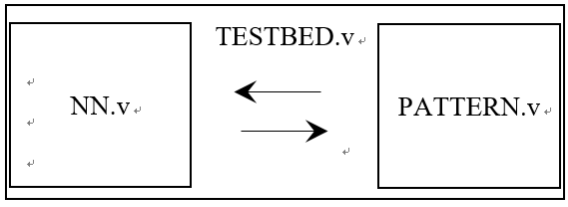
1. Top module name: NN (design file name: NN.v)
2. **You have to check an error under 0.0009 for the result after converting to float number. If the error is higher than the value, you will fail this lab.**
3. **It is asynchronous reset and active-low architecture. If you use synchronous reset (considering reset after clock starting) in your design, you may fail to reset signals.**
4. The reset signal (rst_n) would be given only once at the beginning of simulation. All output signals should be reset after the reset signal is asserted.
5. The **out** should be reset after your **out_valid** is pulled down.
6. The execution latency is limited in **1000 cycles**. The latency is the clock cycles between the falling edge of the **in_valid_k** and the rising edge of the first **out_valid**.
7. The area is limited in **4500000**. Also, the synthesis time should be less than 2 hours.
8. You can adjust your clock period by yourself, but the maximum period is **50 ns**. The precision of clock period is 0.1, for example, 4.5 is allowed, 4.55 is not allowed.
9. The input delay is set to **0.5*(clock period)**.
10. The output delay is set to **0.5*(clock period)**, and the output loading is set to **0.05**.
11. The synthesis result of data type **cannot** include any **latches**.
12. After synthesis, you can check NN.area and NN.timing. The area report is valid when the slack in the end of timing report should be **non-negative (MET)**.
13. In this lab, you must use at least one IEEE floating point number IP from Designware. We will check it at NN.resource in 02_SYN/Report/. The example shows in following figure.



Grading Policy

1. Function Validity: 70%
2. Performance:30 %
   - Area * Computation time: 30%
   - Computation time = (Pattern number+ Latency) * clock cycle time

Block diagram



Note

1. **Please upload the following files on e3 platform before 23:59 p.m. on Mar. 27:**
   - NN_iclab???.v and clock_cycle_iclab???.txt (ex. NN_iclab099.v & 15.5_iclab099.txt), the .txt file contents can be empty, you only need to specify the clock cycle in the file name.
   - The 2nd demo deadline is **23:59 p.m. on APR.1** .
   - Check whether there is any wire / reg / submodule name called “error”, “fail”, “pass”, “congratulation”, if you used, you will fail the lab.
   - If your file violates the naming rule, your will lose 5 point.
2. **Template folders and reference commands:**

   01_RTL/ (RTL simulation) **./01_run**

   02_SYN/ (Synthesis) **./01_run_dc**

   (Check if there is any **latch** in your design in **syn.log)**

   (Check the timing of design in /Report/NN**.timing**)

   03_GATE / (Gate-level simulation) **./01_run**

   ※You should make sure the three clock period values identical in 00_TESTBED/Pattern.v && /02_SYN/syn.tcl**:**

```
`ifdef RTL
    `timescale 1ns/10ps
    `include "NN.v"
    `define CYCLE_TIME 20.0
`endif
`ifdef GATE
    `timescale 1ns/1ps
    `include "NN_SYN.v"
    `define CYCLE_TIME 20.0
`endif
```

```
#=====================
#  Global Parameters
#=====================
set DESIGN "NN"
set CLK_period 20.0
```

Sample Waveform

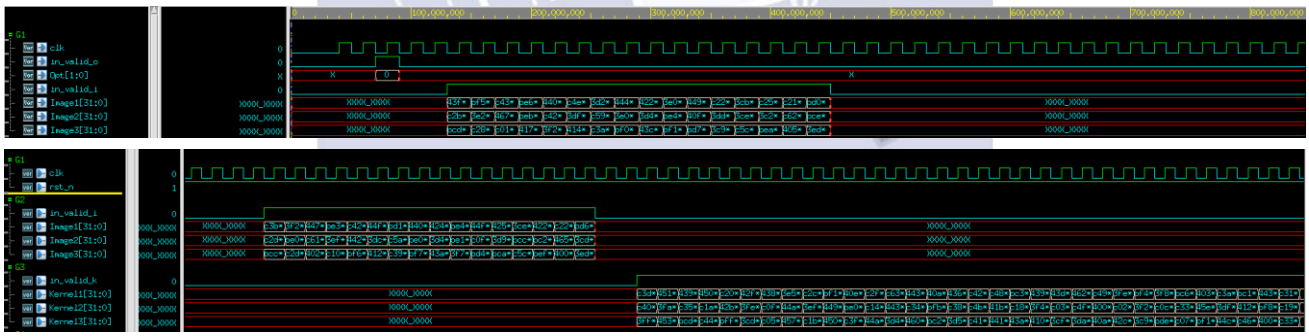


Fig1. Input waveform

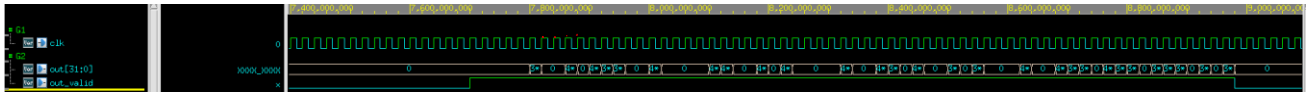
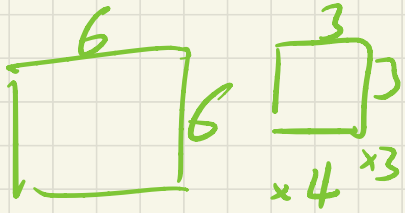
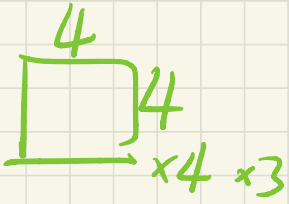


Fig2. Output waveform

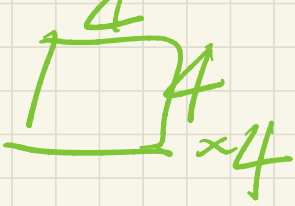
① LOAD (Padding)



② MULT

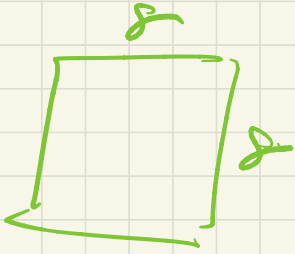


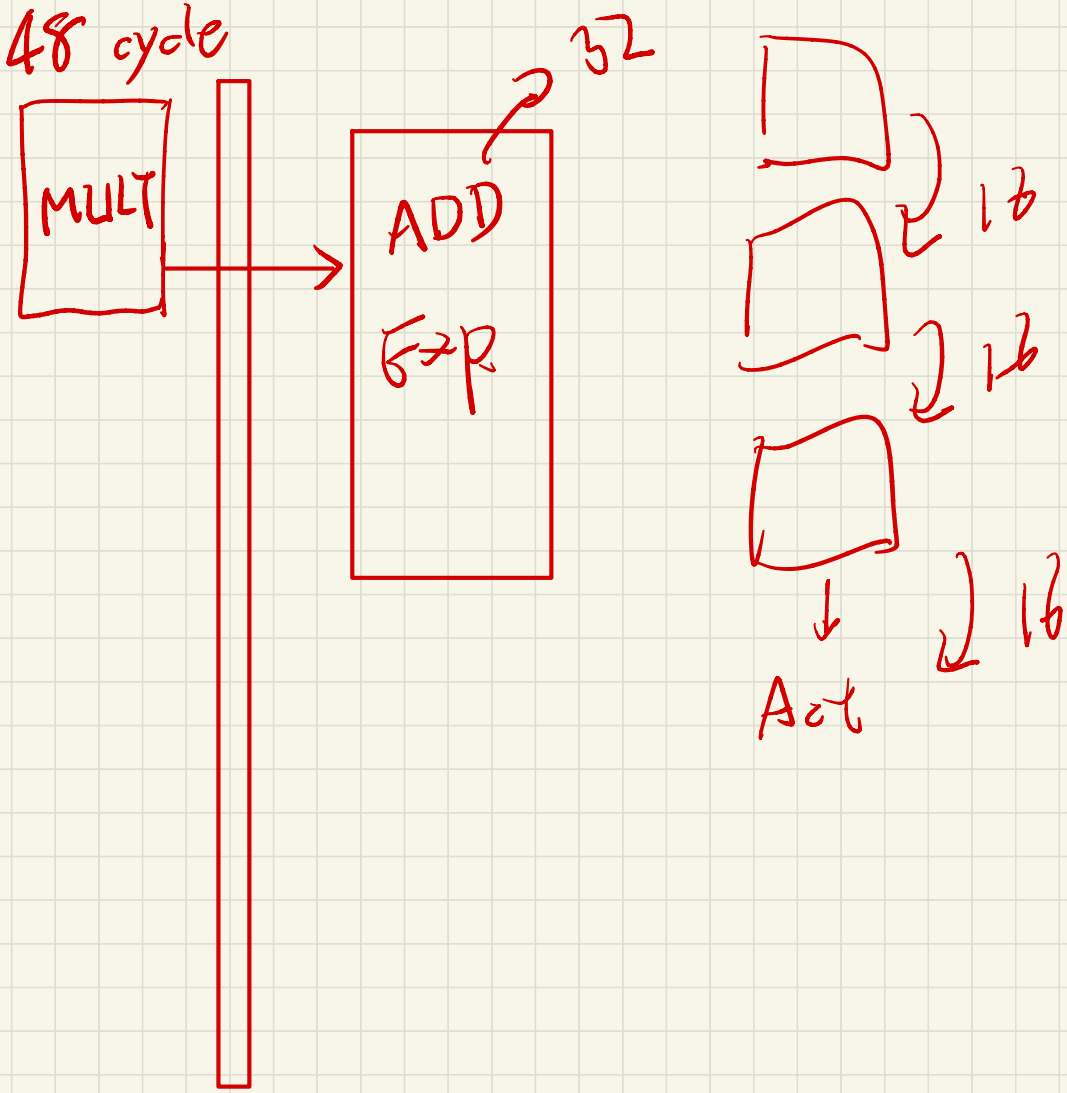
③ PLUS

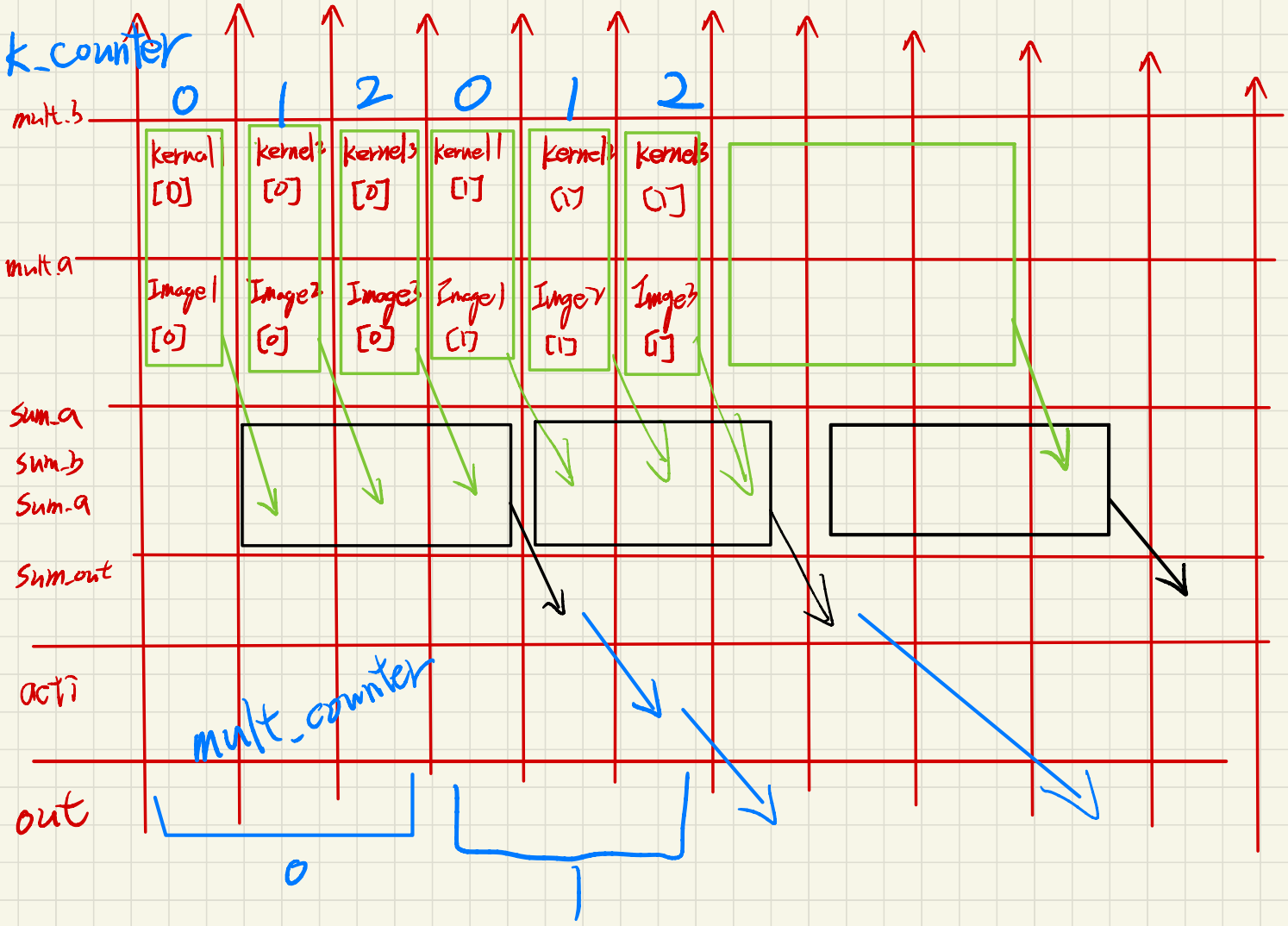


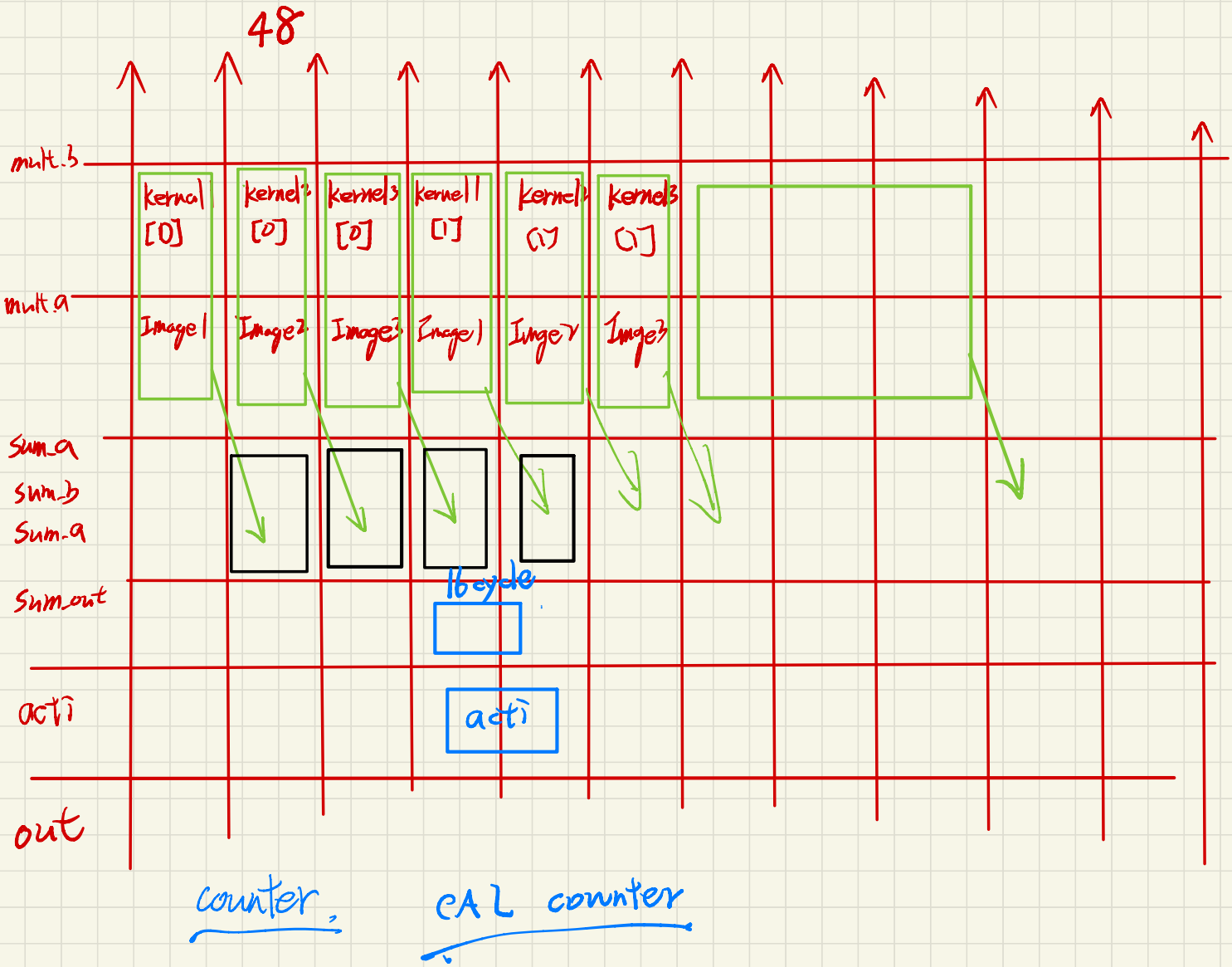
④ EXP

⑤ OUT




48 cycle
MULT
ADD
32
Exp
16
16
16
Act

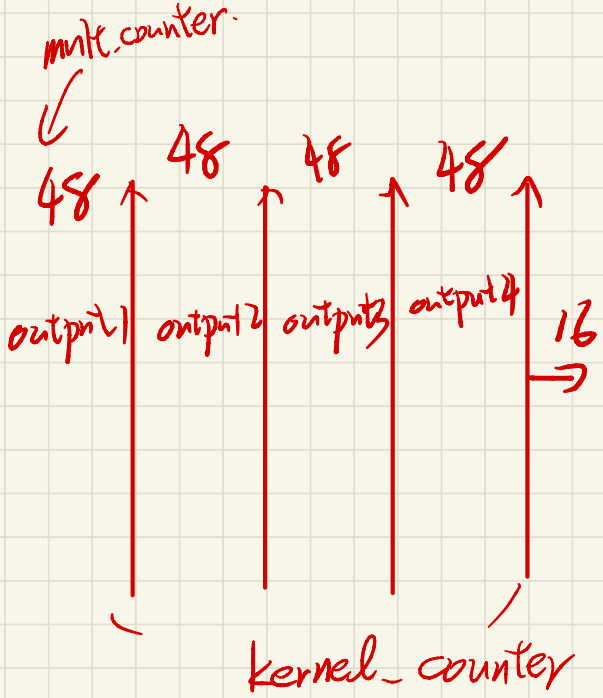

k_counter
0
1
2
0
1
2
mult_b
mult_a
Sum_a
Sum_b
Sum_a
Sum_out
acti
mult_counter
out
0
1


48
mult_b
mult_a
kernel [0]
kernel [0]
kernel [0]
kernel [1]
kernel [1]
kernel [1]
Image1
Image2
Image3
Image1
Image2
Image3
sum_a
sum_b
Sum_a
Sum_out
16cycle
acti
acti
out
counter
CAL counter

next_state = (load_counter == 16 && in_valid_k)
? CAL : LOAD.

next_state = (CAL_counter == 209) ? OUT : CAL;

48+48+48+48+16+1 = 209



0-15 16-31 32-47 48-63

0	0	1	2	3	3
0	0	1	2	3	3
4	4	5	6	7	7
8	8	9	10	11	11
12	12	13	14	15	15
12	12	13	14	15	15

0-8

9-17

18-26

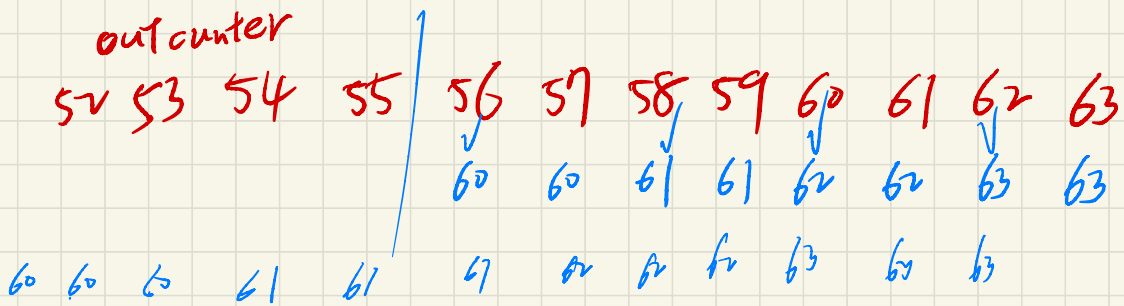
27-35

2^0

12 12 12 12

34 34 34 34

60 61 62 63


outcunter
52 53 54 55 56 57 58 59 60 61 62 63
60 60 61 61 62 62 63 63
60 60 60 61 61 61 62 62 63 63 63