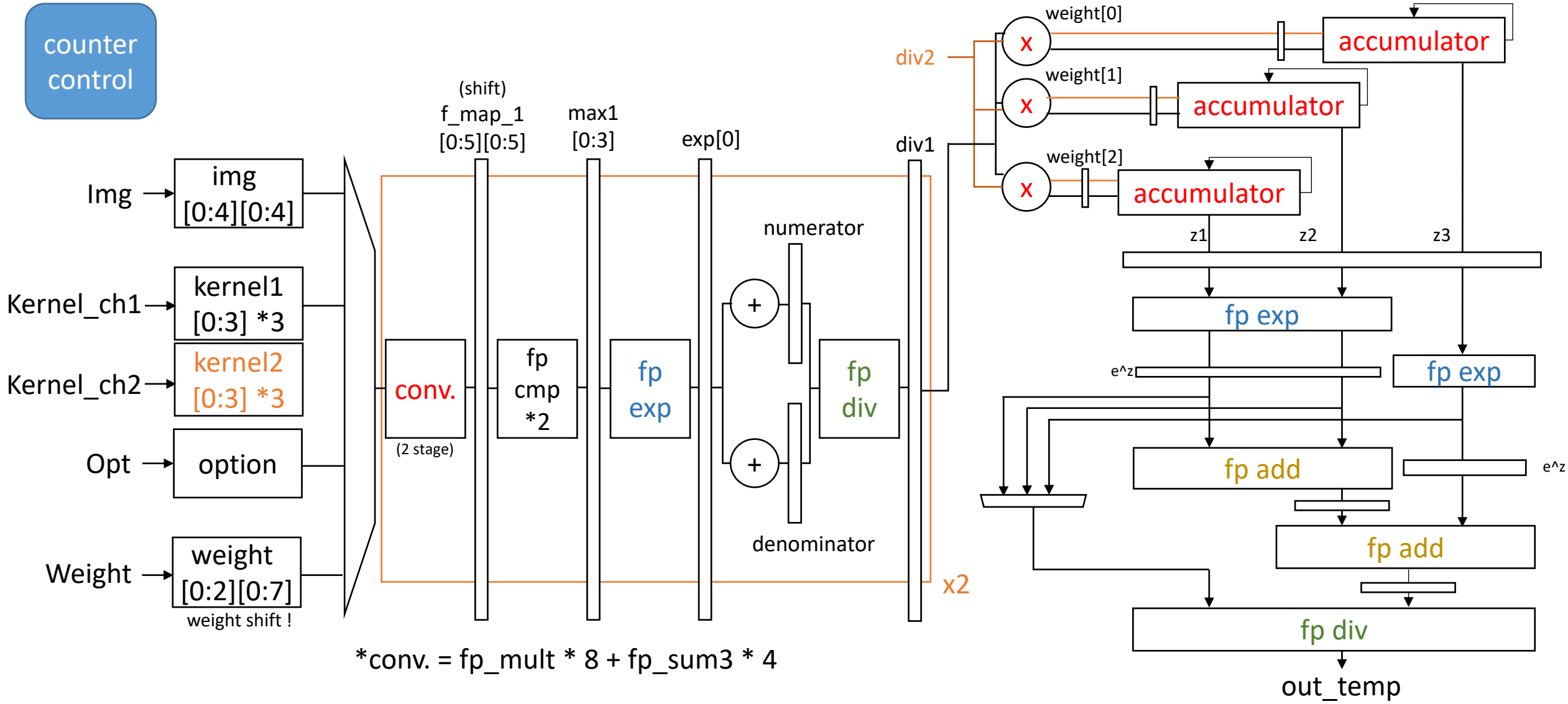
Lab04 Code Review

分享者: 邱士豪

OUTLINE

- Block Diagram
- Design
  - latency decrease
  - shifting feature map
  - activation function
  - fully connected layer

BLOCK DIAGRAM (take kernel1 for example)



Latency Decrease

- input image needs 75 cycles.(25 cycles / image channel)
- convolution:

  1 pixel per cycle = 36 cycles to finish an image channel

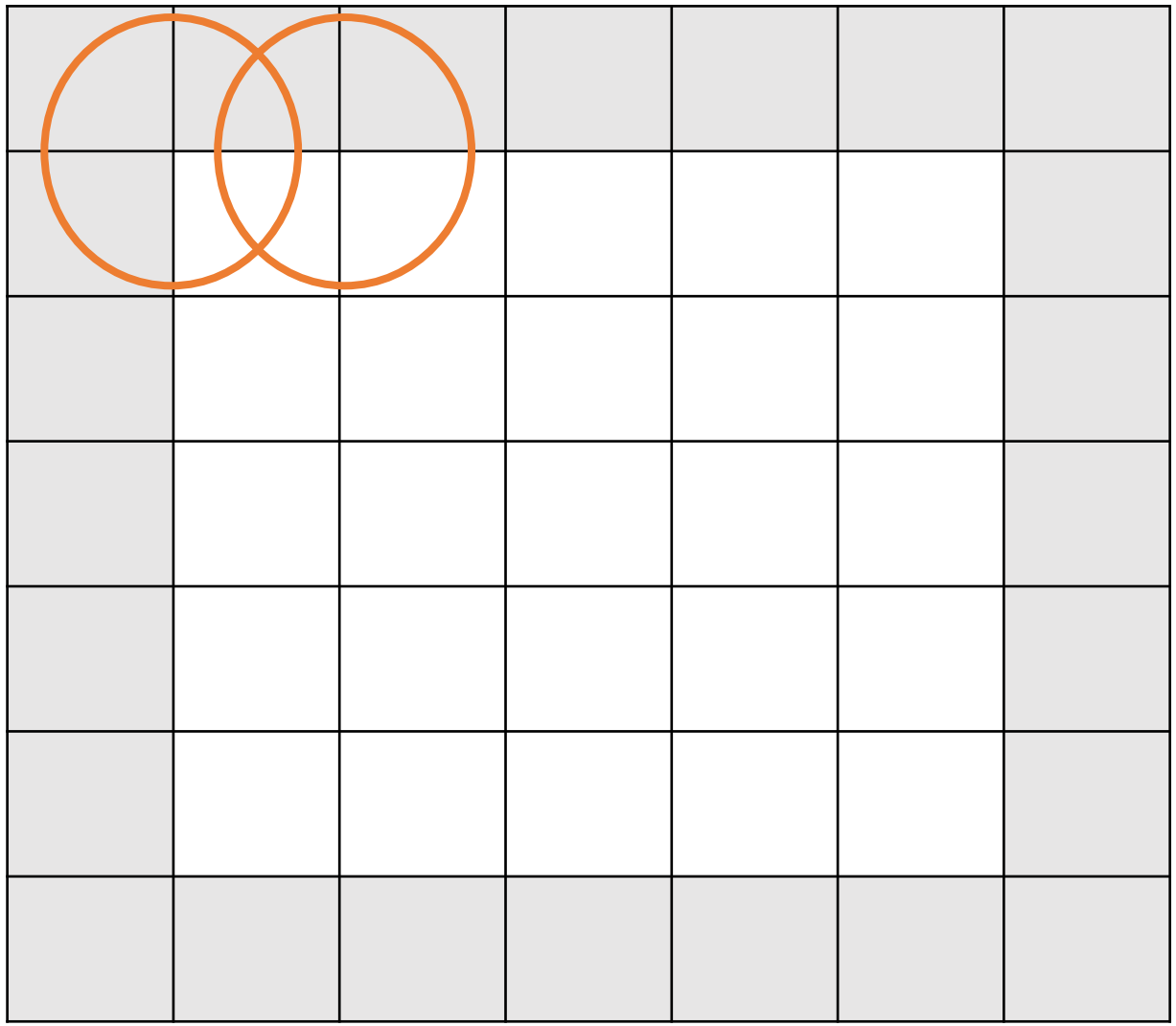
  => latency > 36*3 – 75 = 33 ↑
- 2 pixels per cycle = 18 cycles to finish an image channel

  => double the number of multiplier and adder

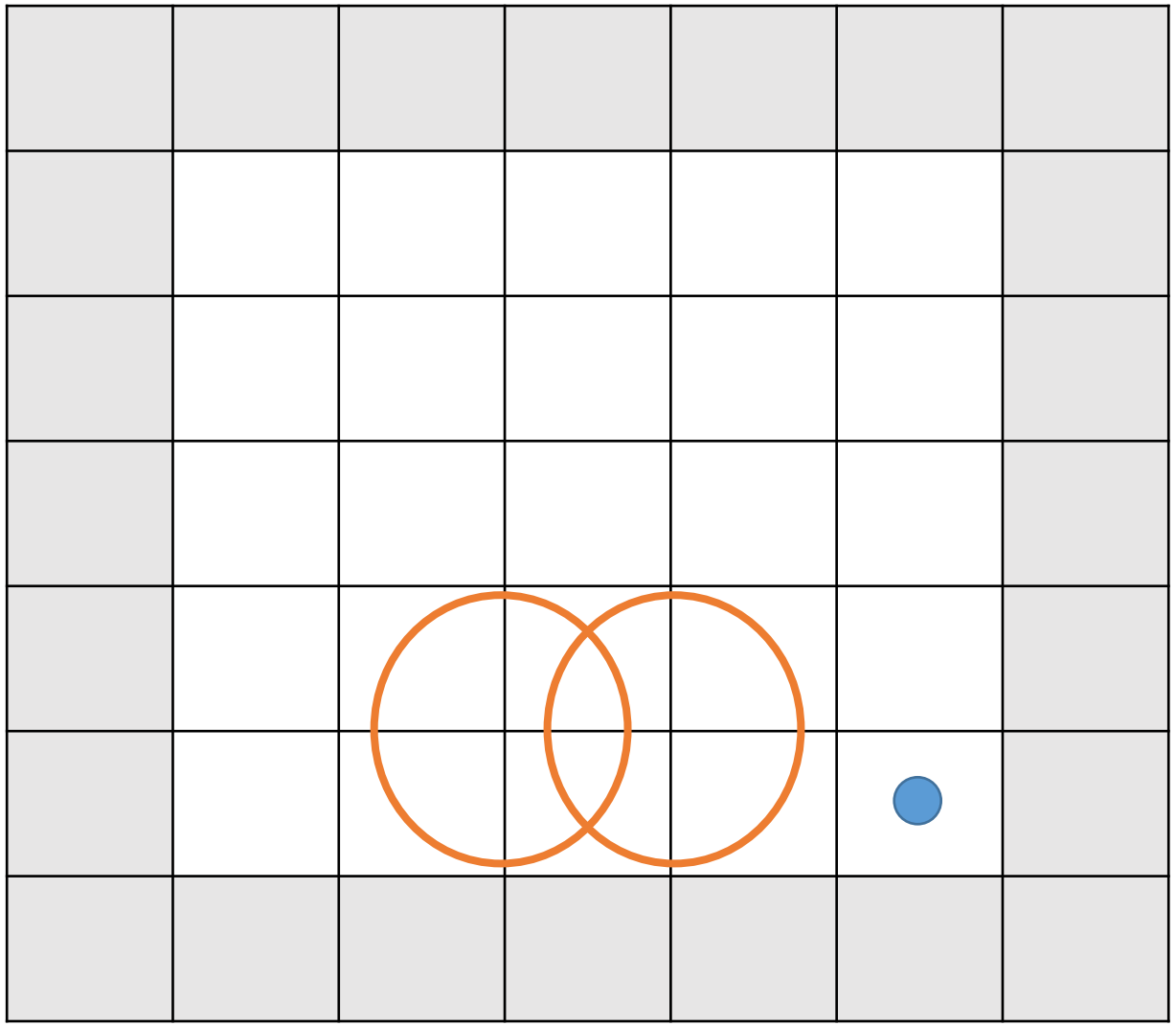
  => faster than image channel

  => larger area but shorter latency !!

Latency decrease depends on how fast you start the convolution.
Notice that the convolution of 2 pixels cannot catch up the unread input.



- WHERE ?



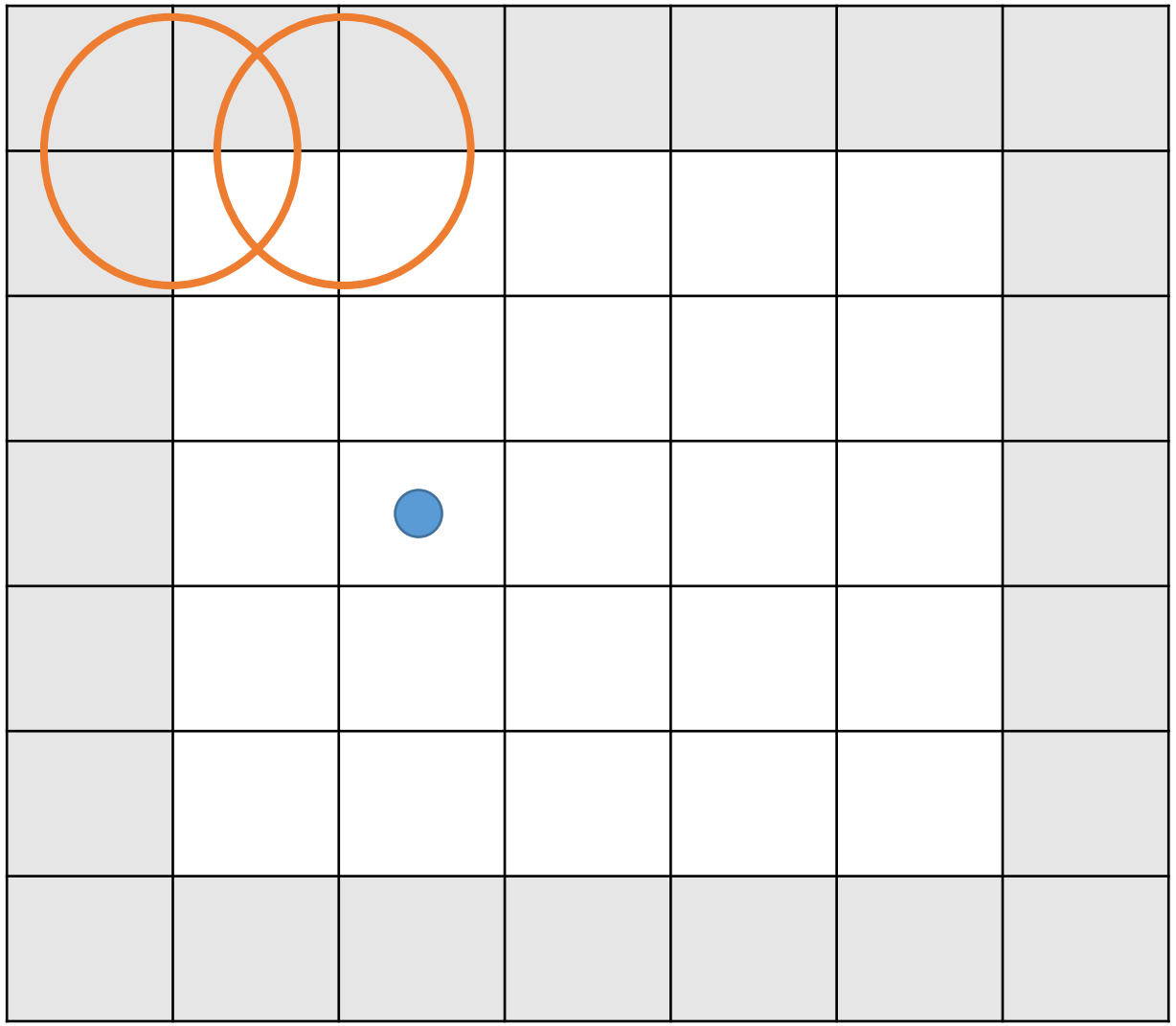
Initial Design:

the 75th input

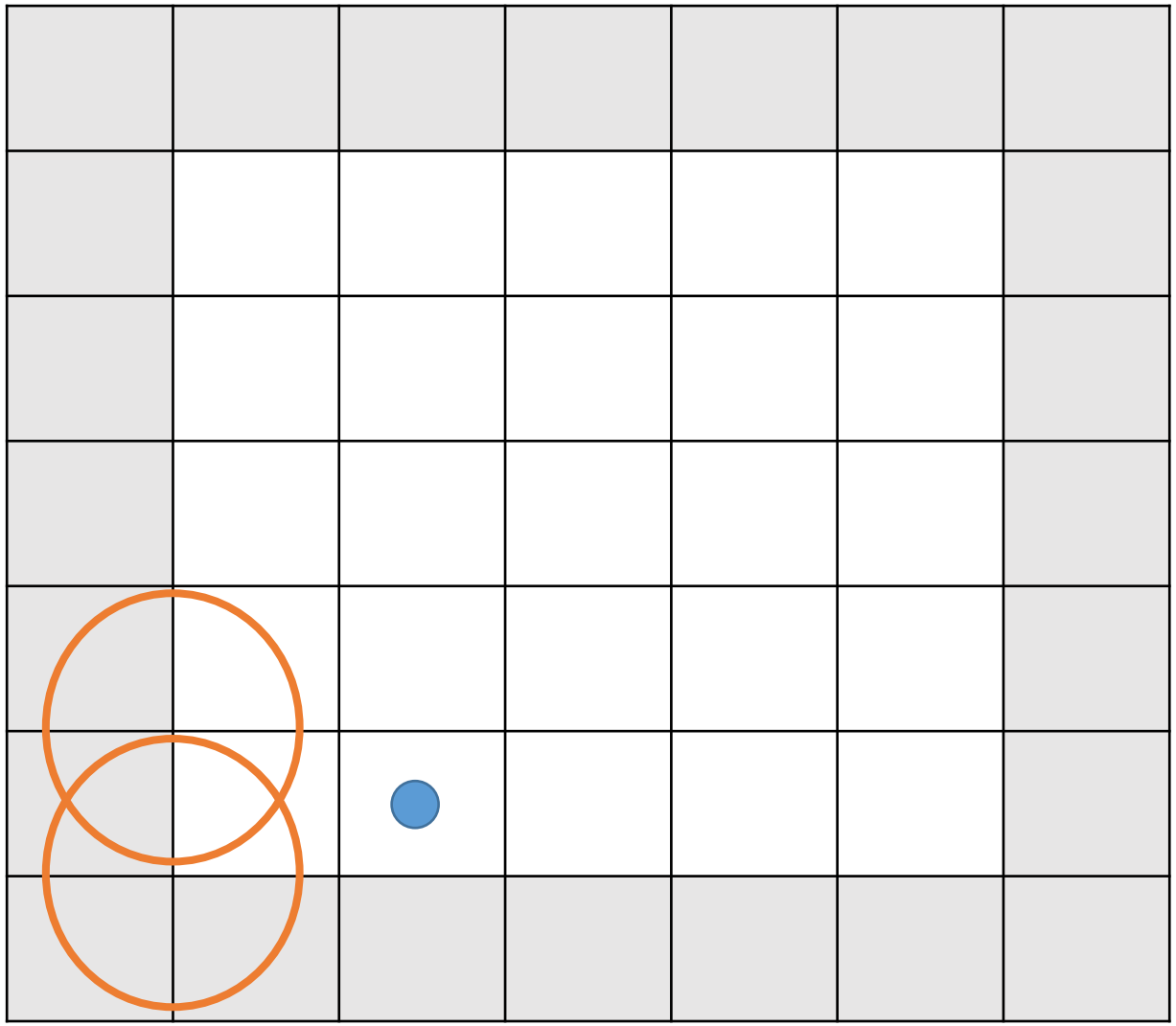
ideal convolution position

4 more cycles to finish, which means latency would be 4

Trace both back to know when to start the convolution

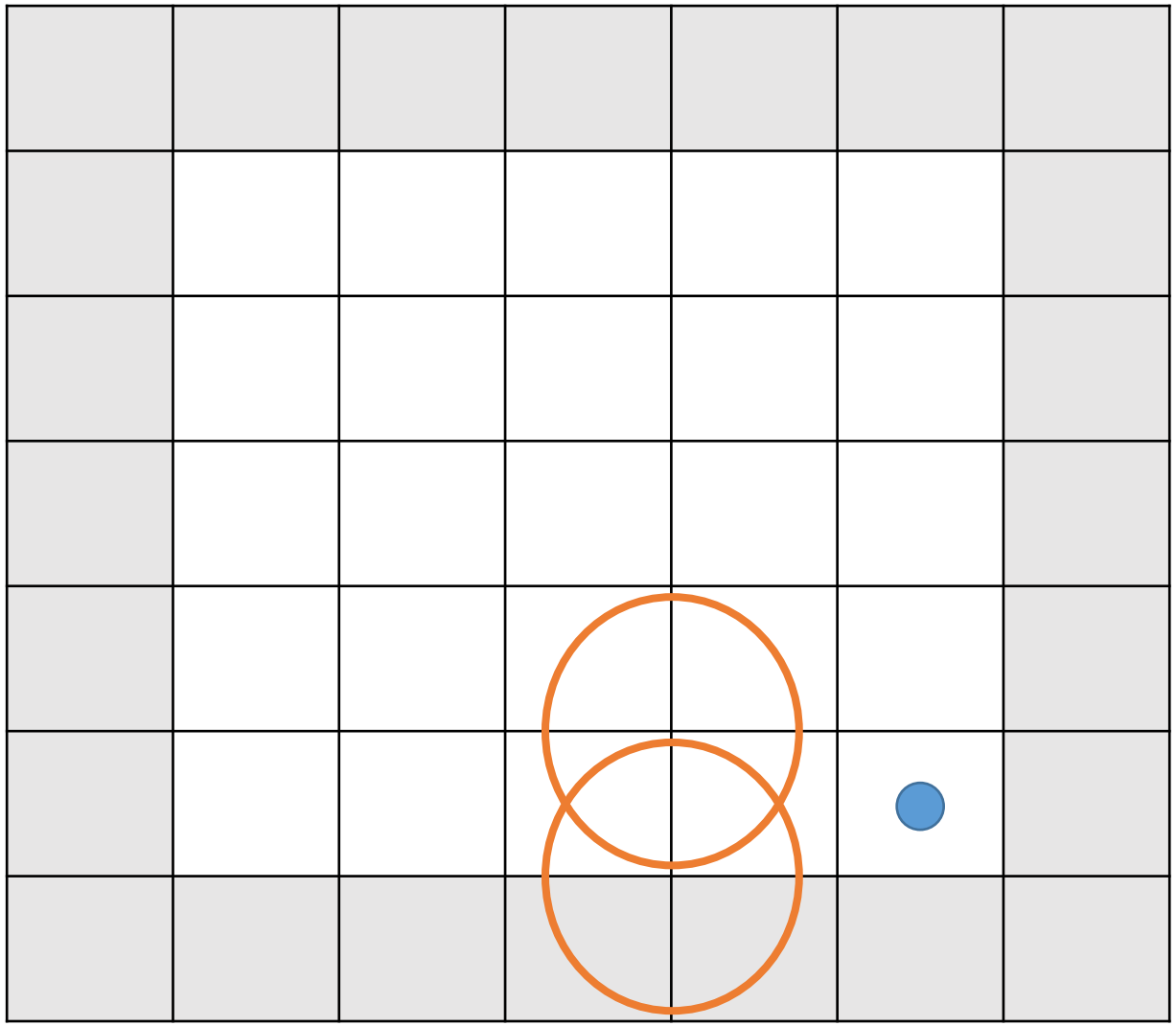


Is this one the best?
Consider “padding” again !



Think about the padding of the bottom left corner.

Change the way how we do the convolution of the last row like this.



Improved Design:

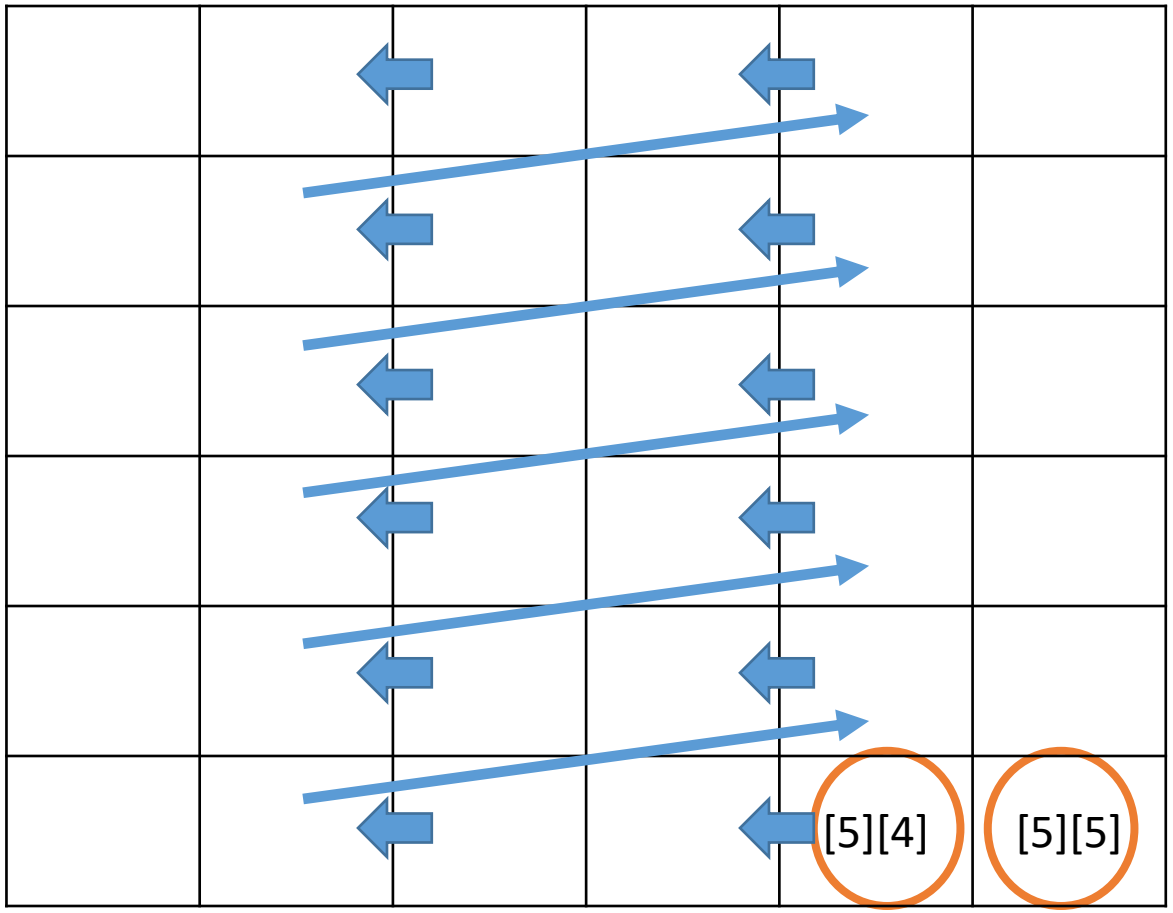
the 75th input

ideal convolution position

2 more cycles to finish, which means latency would be 2!!!

and also trace back to find out when we start.

Shifting Feature Map

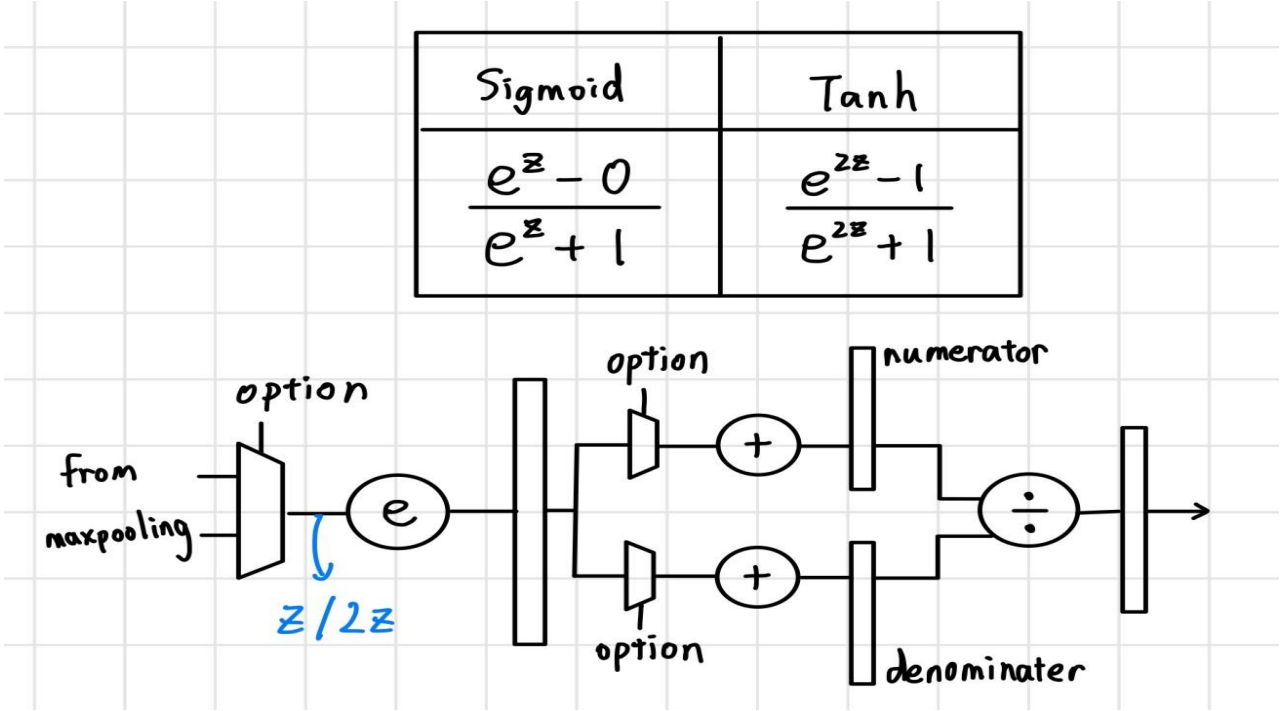


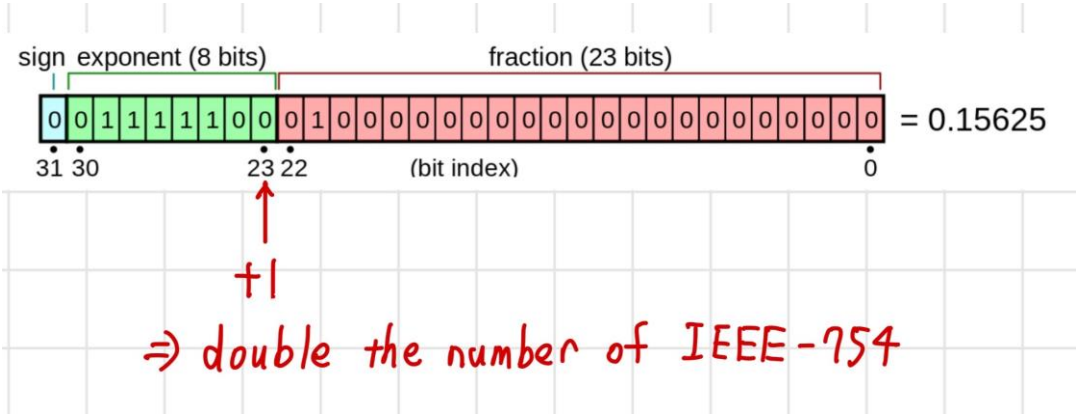
This benefits the complexity of combinational logic of maxpooling part.

The comparing elements will often include [5][5] and [5][4].

Activation

Sigmoid	Tanh
$\frac{e^{z}-0}{e^{z}+1}$	$\frac{e^{2z}-1}{e^{2z}+1}$

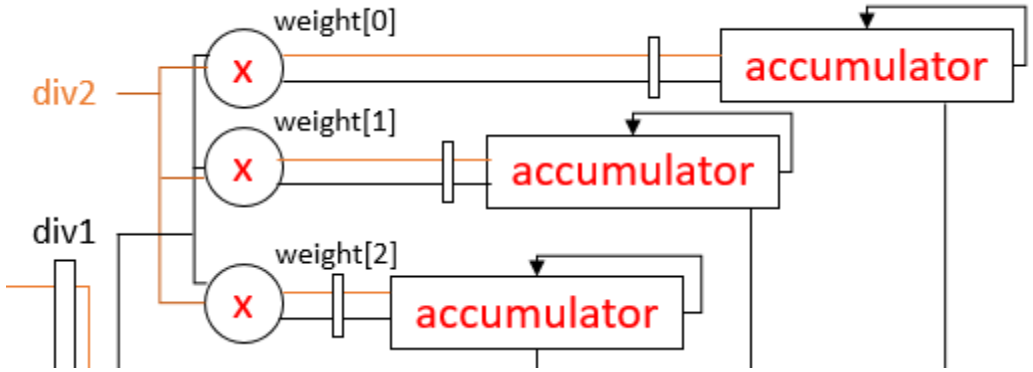




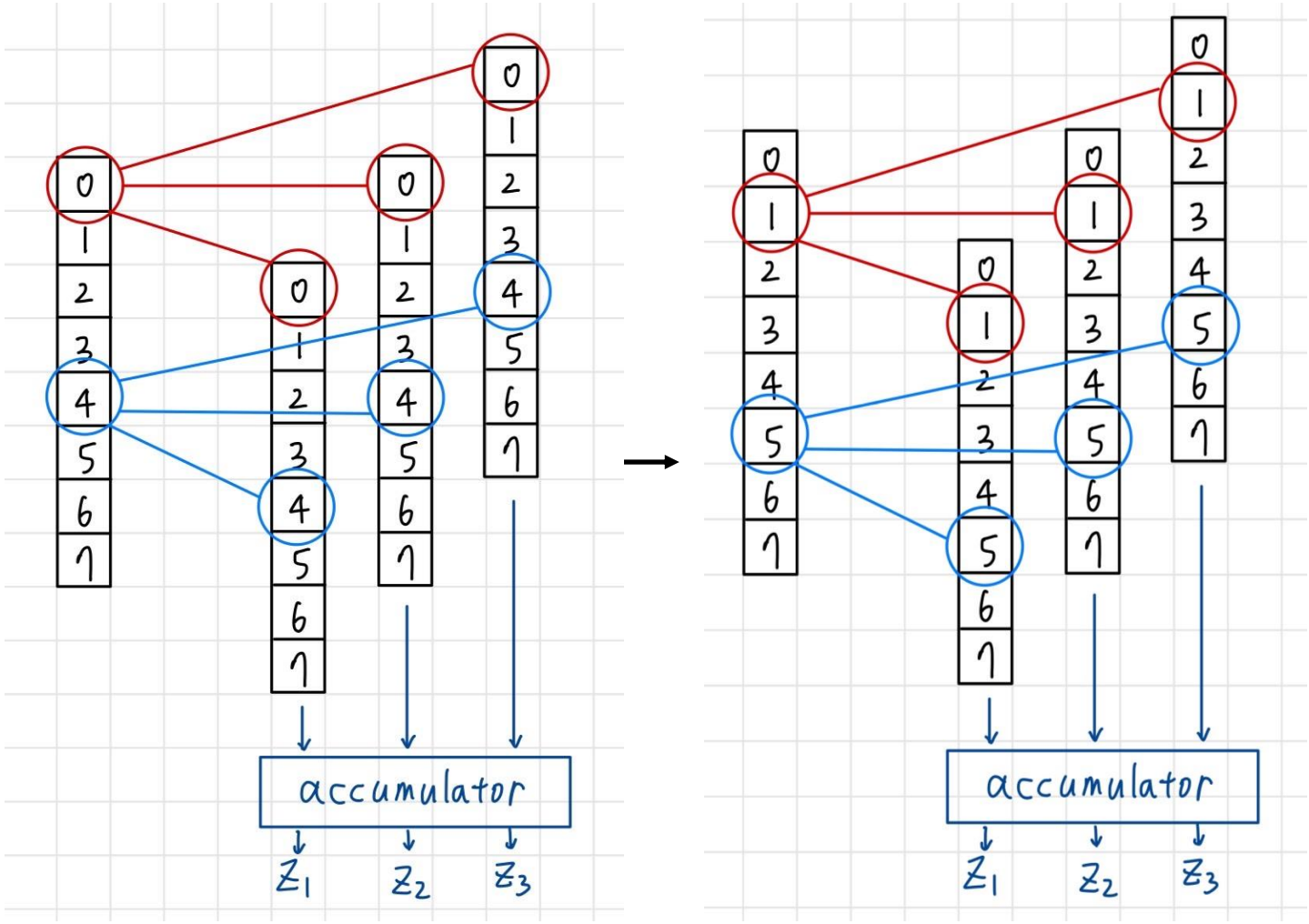

```
max_pooling1 = {max1[1][31], (max1[1][30:23] + 1'b1), max1[1][22:0]};
```

^ synthesis a smaller adder

Fully Connected



weight will keep shifting



4 cycles to get Zs

Fully Connected (shifting weight)

```
always @(posedge clk) begin
    weight[0][0] <= ns_weight[0][0];
    weight[0][1] <= ns_weight[0][1];
    weight[0][2] <= ns_weight[0][2];
    weight[0][3] <= ns_weight[0][3];
    weight[0][4] <= ns_weight[0][4];
    weight[0][5] <= ns_weight[0][5];
    weight[0][6] <= ns_weight[0][6];
    weight[0][7] <= ns_weight[0][7];
    weight[1][0] <= ns_weight[1][0];
    weight[1][1] <= ns_weight[1][1];
    weight[1][2] <= ns_weight[1][2];
    weight[1][3] <= ns_weight[1][3];
    weight[1][4] <= ns_weight[1][4];
    weight[1][5] <= ns_weight[1][5];
    weight[1][6] <= ns_weight[1][6];
    weight[1][7] <= ns_weight[1][7];
    weight[2][0] <= ns_weight[2][0];
    weight[2][1] <= ns_weight[2][1];
    weight[2][2] <= ns_weight[2][2];
    weight[2][3] <= ns_weight[2][3];
    weight[2][4] <= ns_weight[2][4];
    weight[2][5] <= ns_weight[2][5];
    weight[2][6] <= ns_weight[2][6];
    weight[2][7] <= ns_weight[2][7];
end
```

```
always @(*) begin
    ns_weight[0][0] = weight[0][1];
    ns_weight[0][1] = weight[0][2];
    ns_weight[0][2] = weight[0][3];
    ns_weight[0][3] = weight[0][4];
    ns_weight[0][4] = weight[0][5];
    ns_weight[0][5] = weight[0][6];
    ns_weight[0][6] = weight[0][7];
    ns_weight[0][7] = weight[1][0];
    ns_weight[1][0] = weight[1][1];
    ns_weight[1][1] = weight[1][2];
    ns_weight[1][2] = weight[1][3];
    ns_weight[1][3] = weight[1][4];
    ns_weight[1][4] = weight[1][5];
    ns_weight[1][5] = weight[1][6];
    ns_weight[1][6] = weight[1][7];
    ns_weight[1][7] = weight[2][0];
    ns_weight[2][0] = weight[2][1];
    ns_weight[2][1] = weight[2][2];
    ns_weight[2][2] = weight[2][3];
    ns_weight[2][3] = weight[2][4];
    ns_weight[2][4] = weight[2][5];
    ns_weight[2][5] = weight[2][6];
    ns_weight[2][6] = weight[2][7];
    if(cnt < 'd24)begin
        ns_weight[2][7] = Weight;
    end else begin
        ns_weight[2][7] = weight[0][0];
    end
end
```

```
// 'd79, 'd80, 'd81, 'd82: fc layer acculumation
conv_kernel1[0] = weight[2][1];
conv_kernel1[1] = weight[0][1];
conv_kernel1[2] = weight[1][1];
conv_kernel1[3] = 0;
conv_kernel2[0] = weight[2][5];
conv_kernel2[1] = weight[0][5];
conv_kernel2[2] = weight[1][5];
conv_kernel2[3] = 0;
```

Soft Max