NCTU-EE IC LAB – FALL 2019

Lab06 Exercise

Design: Convolution & Max pooling

Data Preparation

1. Extract test data from TA's directory:

   % tar xvf ~iclabta01/Lab06.tar

2. The extracted LAB directory contains:

   a. **00_TESTBED**

   b. **01_RTL**

   c. **02_SYN**

   d. **03_GATE**

Design Description

Recently, convolutional neural network(CNN) had shown its power in image recognition and signal processing field. In CNN models, different layers such as convolution, max pooling and fully connected are included.

1. **Convolution**

   Convolution layer can extract features of image by doing convolution with different kernels. Zero padding is used to remain the output feature map have the same size of origin image.

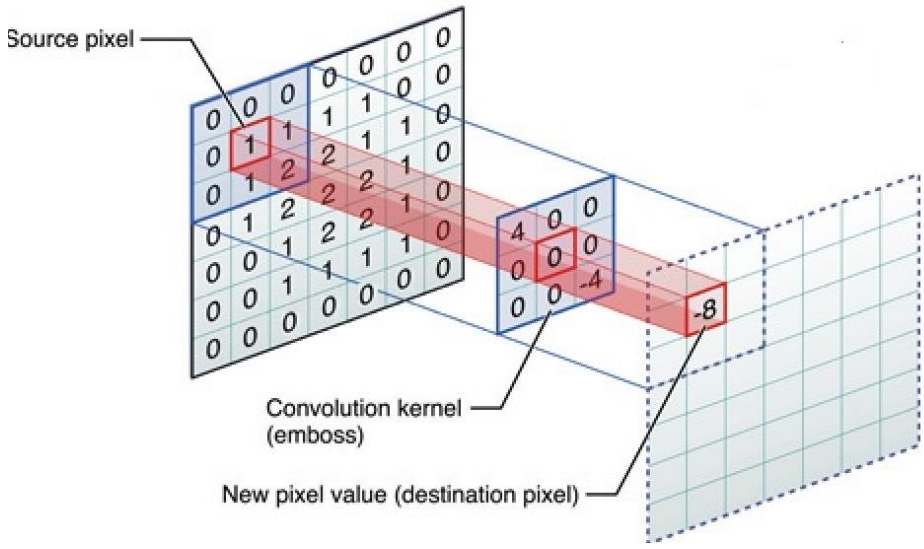


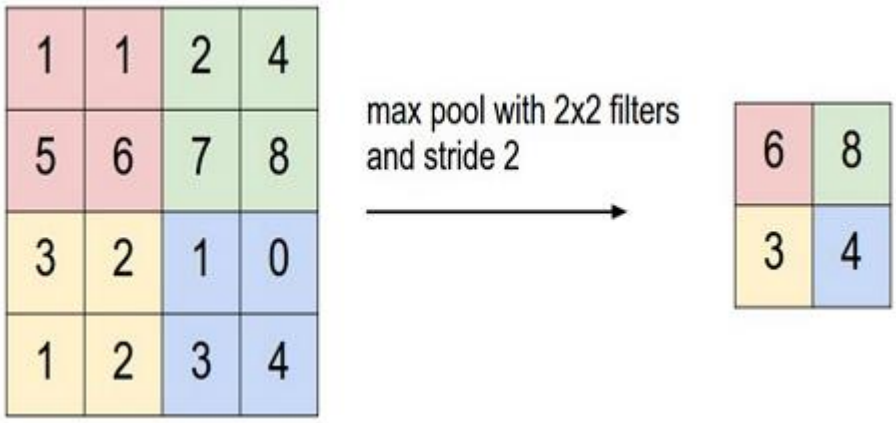
Fig1. Image size 5, Zero padding size 1, kernel size 3, stride 1

Notice : In CNN ,the convolution operation is actually the correlation operation in traditional image processing field which means you don't need toggle the kernel.

2. Max pooling

After you get a convolution result , there usually will be followed by a pooling layer to reduce data size in order to decrease total parameters of network.

The most common setting of pooling layer is max pooling with filter size 2 and stride 2, so each side of feature map will be down size by 2.



In this lab, you need to first design a 2D convolution soft IP and use it to achieve 2D convolution followed by max pooling sequentially. The image size is **4 X 4** and will be fed in **raster scan** order in 16 cycles. **Three** kernel size is **3 X 3** and stride is **1** and will be fed in **raster scan** order in 9 cycles. You need to do zero padding **1** around the image to get the same size of original image. The max pooling size is **2** and the stride is **2.** So , the max pooling result should be **2X2** ,and you should output **three** results correspond to three kernels in **raster scan** order for 4 cycles.

Inputs and Outputs

Input signal	Bit width	Definition
clk	1	Clock
rst_n	1	Asynchronous active-low reset
in_valid1	1	Enable input kernel data
in_valid2	1	Enable input image data
kernel1	3	Input kernel 1
kernel 2	3	Input kernel 2
kernel 3	3	Input kernel 3
Im	8	Input image

Output signal	Bit width	Definition
out_valid	1	Enable output check
out1	19	Output convolution &max pooling result 1
out2	19	Output convolution &max pooling result 2
out3	19	Output convolution &max pooling result 3

1. The **kernel 1~3** is valid only when **in_valid1** is high, and is delivered for 9 cycles continuously .

   When in_valid1 is low, **kernel 1~3** is set to 0.
2. The **im** is valid only when **in_valid2** is high, and is delivered for 16 cycles continuously .

   When in_valid2 is low, **im** is set to 0.
3. **in_valid2** pulls to up right after **in_valid1** pulls to low.
4. All input signals are synchronized at **negative edge** of the clock.
5. There is **only one reset** before the first pattern, **out_valid** and **result** should be low after initial reset
6. **out_valid** should not be raised when **in_valid1 & in_valid2** is high.
7. **out1,out2 out3** should output **continuously 4 cycles** without any interruption.
8. The TA's pattern will capture your output for checking at **clock negative edge**.

Specifications

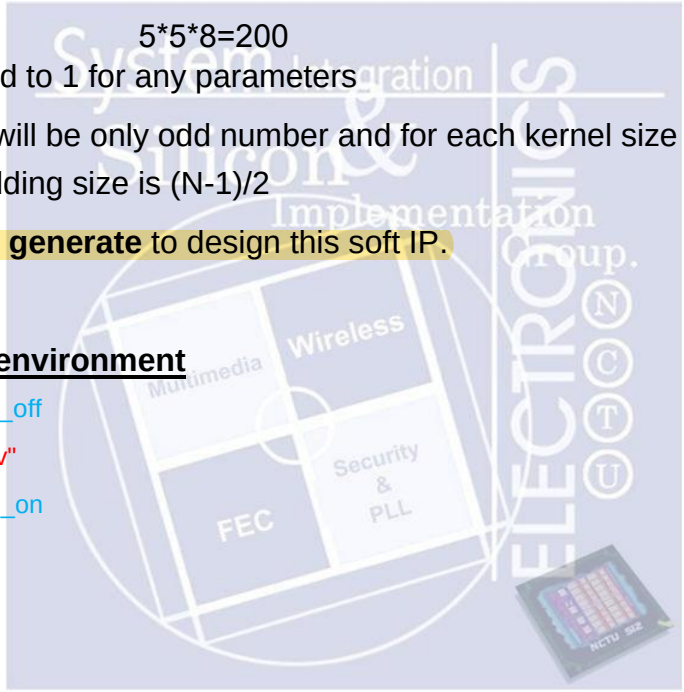
1. Top module name: **CP** (design file name: **CP.v**)
2. It is **asynchronous reset** and **active-low** architecture.
3. The reset signal would be given only once at the beginning of simulation. All output signals should be reset after the reset signal is asserted.
4. The clock period is within **10ns**.
5. The input delay is set to **0.5*(clock period)**.
6. The output delay is set to **0.5*(clock period)**, and the output loading is set to **0.05**.
7. The input delay of **clk** and **rst_n** should be **zero**.
8. The synthesis result (syn.log) of data type **cannot** include any **latches and error.**
9. After synthesis, you can check **CP.area** and **CP.timing**. The area report is valid when the slack in the end of timing report should be **non-negative (MET)**.
10. The gate level simulation **cannot** include any timing violations **without** the **notimingcheck** command.
11. The next group of inputs will come in 2 cycles after your **out_valid** is pulled down.
12. The **out_valid** should be high within **300 cycles** after **in_valid2** pulls to low.
13. The performance is determined by **area** and **latency**. The lower, the better.
14. In this lab, you should write your own **syn.tcl file** and **pattern**.
15. Using **top** wire load mode and **compile ultra**.

Specifications (Soft IP)

1. Top module name: **CONV2D** (design file name: **CONV2D.v**)
2. Input signals：**IMAGE,KERNEL**
3. Output signals：**RESULT**
4. The clock period is **30ns**. Finish convolution within **one** clock cycle.
5. Output loading is set to **0.05**.
6. Using **top** wire load mode and **compile ultra**.
7. Four parameters: M and N indicate the size of image and kernel respectively where M>=N. WIDTH1 and WIDTH2 indicate the bit width of each pixel in image and kernel .
8. Notice that the number sequence should be concatenate together .

   EX: If the input image size is 5 with bit width 8 then the input port should be 200 bits.

   5*5*8=200
9. The stride is fixed to 1 for any parameters
10. The kernel size will be only odd number and for each kernel size the zero padding are needed. Padding size is (N-1)/2
11. You need to use **generate** to design this soft IP.



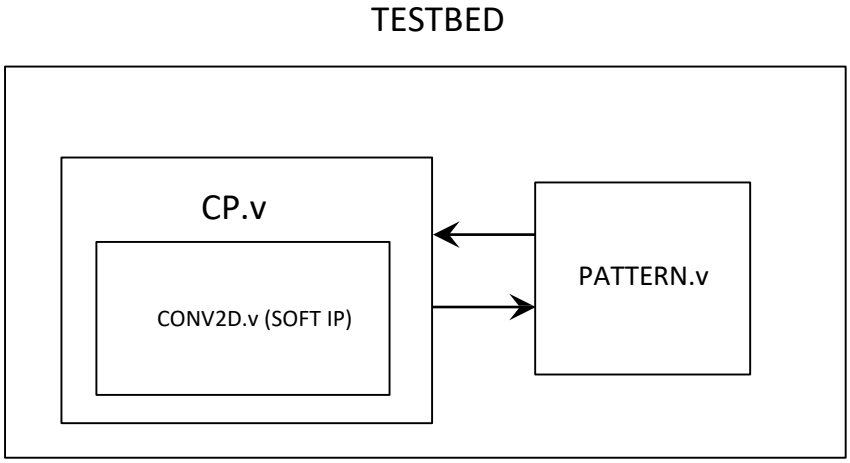
Soft IP Testing environment

```
//synopsys translate_off
`include "CONV2D.v"
//synopsys translate_on

module CP(
// input port
IMAGE,KENEl
// output port
RESULT
);
….

CONV2D #(4,3,8,3) I_CONV2D(
.IMAGE (IMAGE),
.KERNEL (KERNEL)
.RESULT (RESULT)
);
…
endmodule
```

Block diagram



Note

1. Grading policy:

RTL and Gate-level simulation correctness: 45%

Performance: 30%

- Area 20%
- Simulation Time 10%

Soft IP function correctness: 25% (No second demo)

- Randomly choose five combination of #(M,N,WIDTH1,WIDTH2) each 5%
- The range of M : 3~8
- The range of N : 3,5,7
- The range of WIDTH1 : 1~8
- The range of WIDTH2 : 1~3

2. Please upload the following file on e3 platform before noon (12:00 p.m.) on Next Monday

- System Demo : CP_iclabXX.v
- SoftIP : CONV2D_iclabXX.v (XX = Your account)
- CYCLE_iclabXX.txt (If you change the cycle time)

Ex: CP_iclab99.v、CONV2D_iclab99.v、8_iclab99.txt

3. Template folders and reference commands:

01_RTL/ (RTL simulation) ./01_run

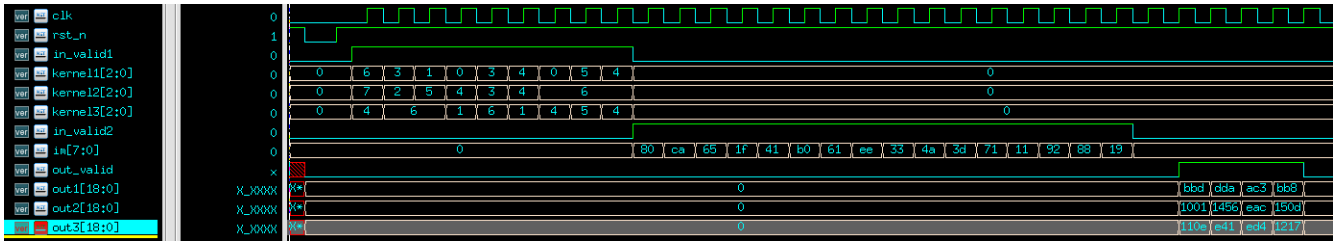
02_SYN/ (Synthesis) ./01_run_dc

(Check the design if there's latch or not in ***syn.log*****)**

(Check the design's timing in /Report/***CP.timing***)

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

Sample Waveform



Version

2019 Fall: Chun-Che Tseng