# NYCU-DCS-2024

HW02

## Design: Convolutional Neural Network (CNN)

### Data Preparation

1. Extract files from TA's directory:
   **% tar xvf ~DCSTA01/HW02.tar**
2. The extracted LAB directory contains:
   a. **00_TESTBED**
   b. **01_RTL**
   c. **02_SYN**
   d. **03_GATE**
   e. **09_SUBMIT**

### Design Description

The **Convolutional Neural Network (CNN)** is a class of deep neural networks, widely used in the fields of computer vision and image processing. They are adept at picking up patterns in spatial data, making them excellent for tasks such as image recognition, object detection, and even video analysis.

**CNNs** mimic the way the human visual cortex operates through the use of convolution. **Convolutional layers** apply various filters to the input data to create feature maps, highlighting important features such as edges, textures, or specific shapes. **Activation functions** are applied after the convolution operations in each layer. The most commonly used activation function is the Rectified Linear Unit (ReLU), which helps the network learn faster. Following the activation function, **pooling layers** are typically used to reduce the dimensionality of the feature maps, enhancing the network's efficiency and focus on the most relevant features.

In this homework, you have to design a **Convolution Layer**, an **Activation Function** and a **Max-Pooling Layer** of CNN.

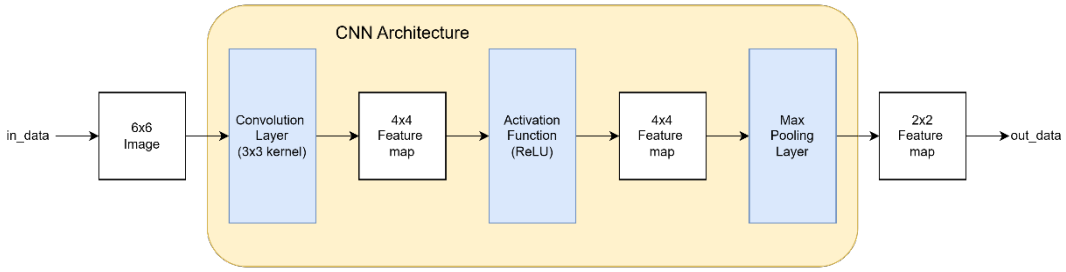



Fig. 1. The Convolutional Neural Network (CNN) architecture

## Functional Description

❖ Convolution

The convolutional formula:

$$\text{FeatureMap}[x, y] = \sum_j \sum_i \text{Image}[x + i, y + j] * \text{Kernel}[i, j]$$

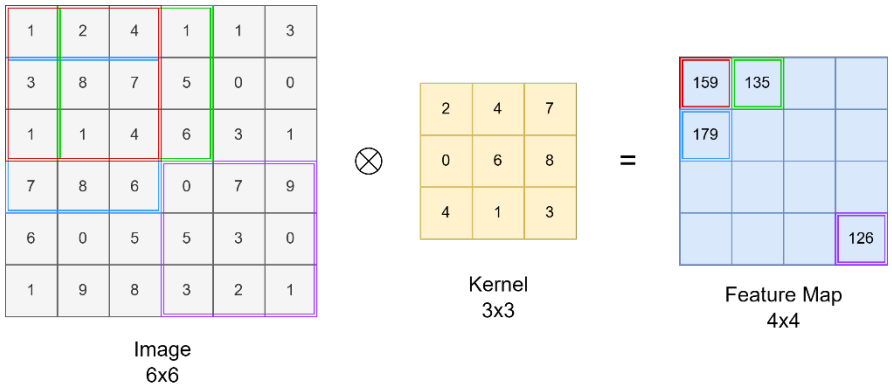



Fig. 2. The example of convolution operation

❖ Activation Function

The ReLU function as activation function applies in this design:

$$\text{ReLU}(x) = \max(0, x)$$

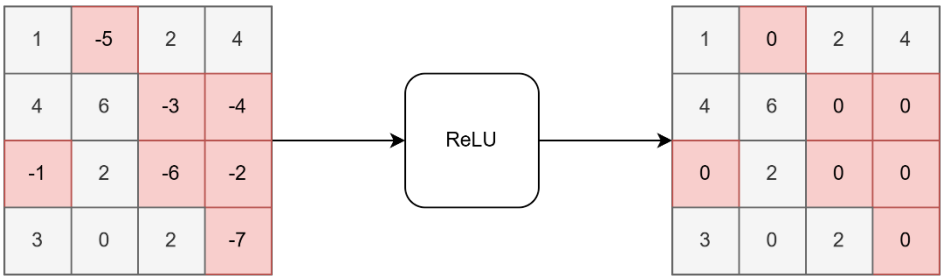



Fig. 3. The example of ReLU function

❖ Max-Pooling

The max-pooling carries out with sliding a 2x2 window over the input, which takes the maximum value in each window as output.

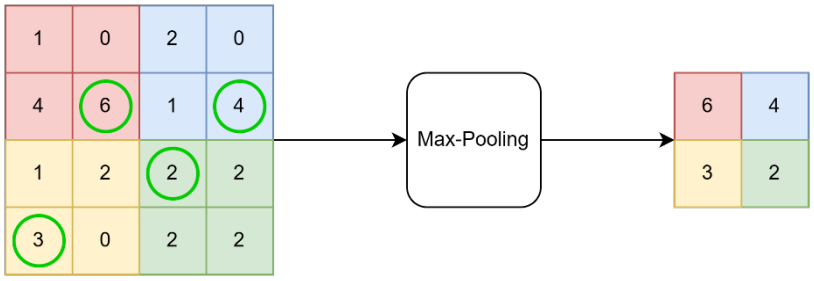



Fig. 3. The example of max-pooling

## Input

The input signals are as follows:

| Input Signal | Bit Width | Description |
|---|---|---|
| clk | 1 | Clock signal |
| rst_n | 1 | Asynchronous active-low reset |
| in_valid | 1 | Be high when input signals are valid. |
| in_data | 16 | The input signals for image and kernel which sent in raster ordering. The arithmetic representation follows the signed 2's complement format. |
| opt | 1 | The option signal<br>1'b0: Activation function: ReLU<br>1'b1: No Activation function |

1. When **in_valid** is high, the 16-bit signed **in_data** is delivered in raster scan order with $\mathbf{6 \times 6 + 3 \times 3 = 45}$ **cycles** continuously.
2. When **in_valid** is low after 45 cycles, the **in_data** is tied to unknown state.
3. The **in_data** is for a $\mathbf{6 \times 6}$ **input image** and $\mathbf{3 \times 3}$ **kernel**. The kernel signal follows the input image.
4. The **opt** is delivered **in the first cycle of in_valid tied to high** and it is **only valid for 1 cycle**.
5. All input signals are synchronized at negative edge of the clock.
6. The next **in_valid** will be valid after 3-5 cycles when previous **out_valid** falls.

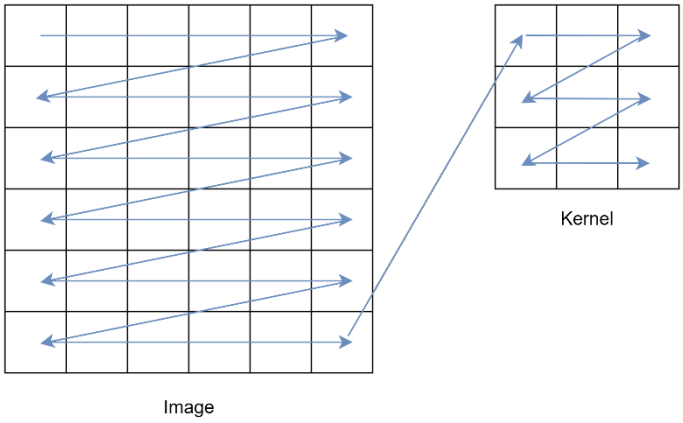



Fig. 4. Input order of the **in_data** signal

## Output

The output signals are as follows:

| Output Signal | Bit Width | Description |
|---|---|---|
| out_valid | 1 | Be high when output signals are valid. |
| out_data | 16 | The output signals for feature map which must be sent in raster ordering. The arithmetic representation follows the signed 2's complement format. |

1. The **out_valid** must be **low** with the **asynchronous reset** when rst_n is **low**.
2. The **out_valid** must be high for **exact 4 consecutive cycles**.
3. The **out_data** must be **zero** with the **asynchronous reset** when rst_n is **low**.
4. The **out_data** must be delivered for **exact 4 consecutive cycles** and **out_valid** should be **high** simultaneously.
5. The **out_data** recommend being **zero** when **out_valid** is **low**.
6. The **out_valid** cannot overlap with **in_valid** at any time.

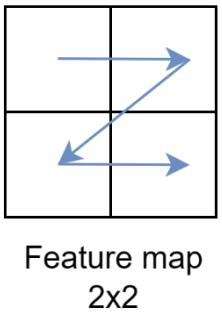



Fig. 5. The order of **out_data** signal

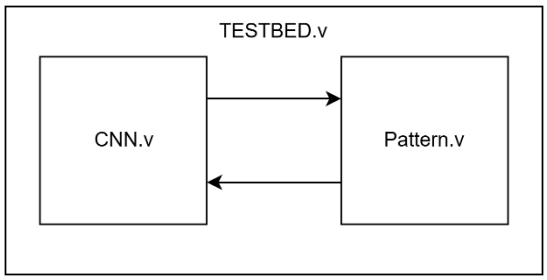
## SPEC

1. Top module name: CNN (File name: CNN.v)
2. It is **asynchronous reset** and **active-low** architecture. If you use synchronous reset (considering reset after clock starting), you may fail to reset signals.
3. 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.
4. The execution latency is limited in **10000 cycles**. The latency is the clock cycles between the falling edge of the in_valid and the rising edge of the out_valid.
5. The synthesis result of data type **cannot** include any **latches** (using ctrl+F to find the term of "Latch" or using the command **./08_check** in 02_SYN/**).**
6. After synthesis, you can check CNN.area and CNN.timing. The timing report should be **non-negative (MET)**.
7. After gate-level simulation, the file of vcs.log cannot include any **timing**

**violation**.

8. Your design **must contain FSM**; if not, you will **fail** this homework.

## Block Diagram




## Homework Upload

1. Please use the commands we provided to tar whole file under 09_SUBMIT/ and to submit your homework **before 15:00 on 4/18 (Thu.)**.
2. Please submit the report file to the new E3. The name of your report file is **report_HW02_DCSXXX.pdf (XXX is your account ID)**. If the file violates the naming rule and the file format, **10 points will be deducted.**
3. Please check the homework file again after you upload it. If you upload the wrong file, you will **fail** this homework.

## Grading Policy

1. Function Validity: total 80%
   - RTL: 70%
   - SYN and GATE: 10%
     - The score **is only based on** the demo result you submit with 09_SUBMIT. Please ensure that you successfully upload the latest version.
     - Please check whether there is any wire/reg/submodule being named as "error", "fail", "pass", "congratulations", "latch". All letters used in the formats of uppercase or lowercase is prohibited. If there is, you will **fail** this homework.
2. Report: total 20 %
   - Your report must contain:
     1. The problem encountered in this homework and how the problem was solved.
     2. Any suggestions about Lab03, Lab04, and HW02.

## Command List

- Verilog RTL simulation (01_RTL/):

- **./01_run_vcs_rtl**

❖ Synthesis (02_SYN/):

- **./01_run_dc_shell**
- **./08_check**

❖ Gate level simulation (03_GATE/):

- **./01_run_vcs_gate**

❖ Submit your files (09_SUBMIT):

- **./00_tar 100**
  - cycle time is 100 in this design. Don't modify it in this homework.
- **./01_submit**
- **./02_check**

❖ Waveform for debug

- **nWave &**
- find *.fsdb
- shift+L for reloading the *.fsdb file after you simulate your design again.

## Sample Waveform

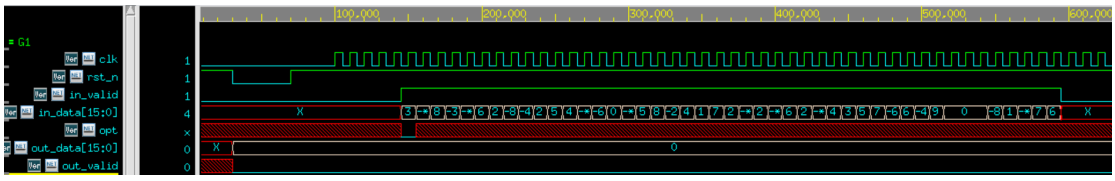



Fig1. The example of input waveform.

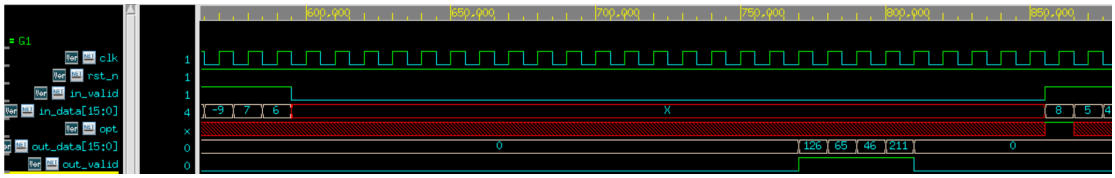



Fig2. The example of output waveform.

## Note

1. You can modify TESTBED.v and PATTERN.v, but we will demo your design through TA's TESTBED.v and PATTERN.v
2. We strongly recommend using the following four states when utilizing an FSM: IDLE, READ, CALC, and OUT. Additional states can be added if necessary.
3. Methods to avoid latches: (1) Avoid multiple driven. (2) Ensure all if-else statements and case with full case. (3) Avoid combinational loops.
4. You must calculate negative integers in this homework, so we recommend use signed wire/reg to store all your data.