Machine Learning Intelligent Chip Design

Homework1 Implementation of AlexNet in SystemC

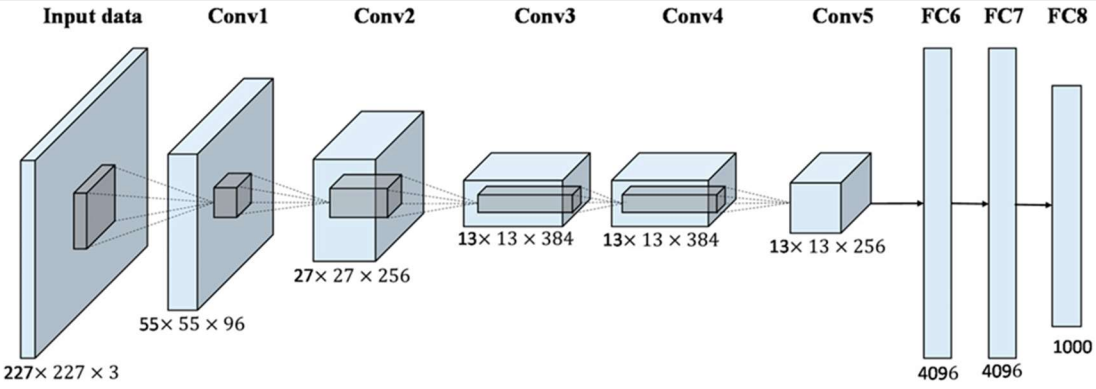
Description

In this task, you are required to implement the AlexNet convolutional neural network architecture using SystemC. AlexNet is a seminal deep learning model that achieved significant breakthroughs in image classification tasks. The network consists of multiple layers, including convolutional layers, max-pooling layers, and fully connected layers.

Implementation Details

- **Network Architecture**:
  You are provided with the network architecture details, including the number of layers, the size of input and output feature maps, filter sizes, and strides.
- **Input Data**:
  Input data for the network will be provided to you. Each input image will have the appropriate dimensions compatible with the network's input layer.
- **Weights and Biases**:
  The weights and biases for each layer of the network will be provided. These parameters are essential for the convolutional and fully connected layers' computations.
- **Simulation**:
  Once you have implemented the network in SystemC and integrated the input data, weights, and biases, you should simulate the network to obtain the predicted output. This will involve passing the input data through the network layers, applying convolutional operations, activation functions, and pooling, followed by fully connected layers, until you get the final output.

Alexnet Training Model



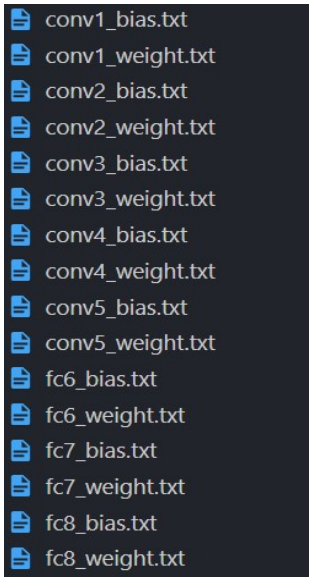
The Pre-trained AlexNet Model Information

```
==========================================================================================================================================
Layer (type:depth-idx)                   Input Shape               Output Shape              Param #                 Kernel Shape            Mult-Adds
==========================================================================================================================================
AlexNet                                  [1, 3, 224, 224]          [1, 1000]                 --                      --                      --
├─Sequential: 1-1                        [1, 3, 224, 224]          [1, 256, 6, 6]            --                      --                      --
│    └─Conv2d: 2-1                       [1, 3, 224, 224]          [1, 64, 55, 55]           23,296                  [11, 11]                70,470,400
│    └─ReLU: 2-2                         [1, 64, 55, 55]           [1, 64, 55, 55]           --                      --                      --
│    └─MaxPool2d: 2-3                    [1, 64, 55, 55]           [1, 64, 27, 27]           --                      3                       --
│    └─Conv2d: 2-4                       [1, 64, 27, 27]           [1, 192, 27, 27]          307,392                 [5, 5]                  224,088,768
│    └─ReLU: 2-5                         [1, 192, 27, 27]          [1, 192, 27, 27]          --                      --                      --
│    └─MaxPool2d: 2-6                    [1, 192, 27, 27]          [1, 192, 13, 13]          --                      3                       --
│    └─Conv2d: 2-7                       [1, 192, 13, 13]          [1, 384, 13, 13]          663,936                 [3, 3]                  112,205,184
│    └─ReLU: 2-8                         [1, 384, 13, 13]          [1, 384, 13, 13]          --                      --                      --
│    └─Conv2d: 2-9                       [1, 384, 13, 13]          [1, 256, 13, 13]          884,992                 [3, 3]                  149,563,648
│    └─ReLU: 2-10                        [1, 256, 13, 13]          [1, 256, 13, 13]          --                      --                      --
│    └─Conv2d: 2-11                      [1, 256, 13, 13]          [1, 256, 13, 13]          590,080                 [3, 3]                  99,723,520
│    └─ReLU: 2-12                        [1, 256, 13, 13]          [1, 256, 13, 13]          --                      --                      --
│    └─MaxPool2d: 2-13                   [1, 256, 13, 13]          [1, 256, 6, 6]            --                      3                       --
├─AdaptiveAvgPool2d: 1-2                 [1, 256, 6, 6]            [1, 256, 6, 6]            --                      --                      --
├─Sequential: 1-3                        [1, 9216]                 [1, 1000]                 --                      --                      --
│    └─Dropout: 2-14                     [1, 9216]                 [1, 9216]                 --                      --                      --
│    └─Linear: 2-15                      [1, 9216]                 [1, 4096]                 37,752,832              --                      37,752,832
│    └─ReLU: 2-16                        [1, 4096]                 [1, 4096]                 --                      --                      --
│    └─Dropout: 2-17                     [1, 4096]                 [1, 4096]                 --                      --                      --
│    └─Linear: 2-18                      [1, 4096]                 [1, 4096]                 16,781,312              --                      16,781,312
│    └─ReLU: 2-19                        [1, 4096]                 [1, 4096]                 --                      --                      --
│    └─Linear: 2-20                      [1, 4096]                 [1, 1000]                 4,097,000               --                      4,097,000
==========================================================================================================================================
Total params: 61,100,840
Trainable params: 61,100,840
Non-trainable params: 0
Total mult-adds (M): 714.68
==========================================================================================================================================
Input size (MB): 0.60
Forward/backward pass size (MB): 3.95
Params size (MB): 244.40
Estimated Total Size (MB): 248.96
==========================================================================================================================================
```

Provided Data Description

Values in the pre-train model in Pytorch are floating points with 16 digits after the decimal. We export these values as txt file for you. Values in these txt files are floating point but rounded to the sixth decimal place.

- **Model layer parameters**

- **imagenet_classes.txt**
  https://gist.github.com/ageitgey/4e1342c10a71981d0b491e1b8227328b
- **Input Data**

  dog.txt

  cat.txt

Reference Result

- **Simulation results of AlexNet executed in Python**

  Dog

```
In [20]: runfile('C:/Users/micha/Desktop/use_dataset_alexnet_model_instance.py', wdir='C:/Users/micha/Desktop')
Predicted class: golden retriever
```

  Cat

```
In [9]: runfile('C:/Users/micha/Desktop/use_dataset_alexnet_model_instance.py', wdir='C:/Users/micha/Desktop')
Predicted class: Egyptian cat
```

- **Simulation results of AlexNet executed in SystemC**

  Dog

idx	val	possibility	class name
207	16.5945	38.62767	golden retriever
175	15.5697	13.86112	otterhound
220	15.3619	11.26034	Sussex spaniel
163	15.0027	7.86242	bloodhound
219	14.5932	5.22078	cocker spaniel

  Cat

idx	val	possibility	class name
285	20.2067	96.38149	Egyptian cat
281	16.1368	1.64618	tabby
282	15.7338	1.10018	tiger cat
287	14.7909	0.42848	lynx
728	14.4119	0.29331	plastic bag

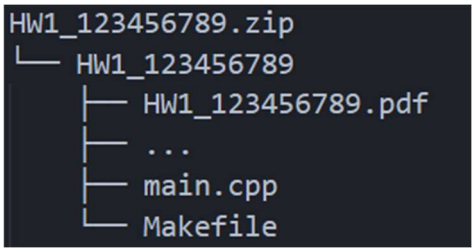
Implement Notes

The purpose of this assignment is just to make students familiar with SystemC, so as long as the execution results are correct, we will not restrict how students implement it.
Here are some tips for your reference:

- You can use one `SC_MODULE` to implement the entire module, or implement each layer with different `SC_MODULE`.
- You can use `sc_signal` to connect different modules.
- We strongly recommend building a monitor module to receive output and print out the execution results.

Submission Guidelines

- **Please compress a folder named HW<ID>_<studend-ID> into a zip file with the same name and upload it to E3.**
- **The folder should include:**
  - Report (Name: HW<ID>_<student ID>.pdf)
  - Codes
  - Makefile
- **Example:**



- **You don’t need to upload parameters.**
- **Ensure that your code is well-commented and organized for clarity and understanding.**
- **Plagiarism is forbidden, otherwise you will get 0 point!!!**

Deliverables

- **SystemC Implementation:**
  Complete implementation of AlexNet architecture using SystemC.
- **Report:**
  A brief report containing
  - Simulation results demonstrate the predicted output for the provided input data.
  - Your implementation approach, challenges faced, and any observations or insights gained during the implementation and simulation process.