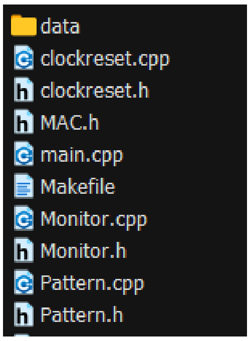
MLCHIP HW1

Implementation of AlexNet in SystemC

Implementation

File list	Description
data/	All input files are put in the data folder, including input feature maps (size: 224*224*3), weights, bias and classes.
clockreset.h	Define the Clock module and the Reset module.
clockreset.cpp	Include clockreset.h, describe the functions in it and detail how they work.
MAC.h	All AlexNet functions are in this file. Read weights and bias at the reset stage. Do the convolution, ReLU, maxpooling or linear function when valid signals for specific layers are high at positive edge of clock. The submodules for AlexNet layers are listed below.
main.cpp	Connect all signals in different modules, like testbench. Define the reset signal to last for 10ns, the clock cycle is 10ns, and the clock would change its value for 60 times (30 cycles). Also define the sc_start to determine the total simulation time.
Makefile	It automates the process of compiling and linking source code files. Use the “make” instruction to run the codes and use the “make clean” instruction to clear the previous result.
Monitor.h	Receive input signals that are needed from other modules. Define the Monitor module, including the constructor of the module.
Monitor.cpp	Read the imagenet_class.txt by line at the reset stage. Print out the results of intermediate layers during the calculation. Deal with SoftMax calculation based on the results from the last layer of AlexNet. Finally, sort the SoftMax results and print the top 5 results for 2 input images.
Pattern.h	Define the Pattern module. This module is triggered at the negative edge of clock.
Pattern.cpp	Detail the function of the Pattern module. Read the input images and generate the valid signal for the first layer of AlexNet.



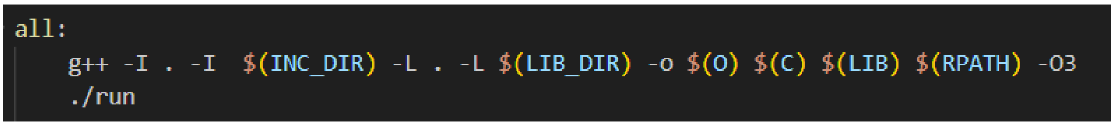
data
clockreset.cpp
clockreset.h
MAC.h
main.cpp
Makefile
Monitor.cpp
Monitor.h
Pattern.cpp
Pattern.h

AlexNet Layers

My SC_MODULE	AlexNet Layer
CONV_RELU_1	Conv2d (in_channel=3, out_channel=64, kernel_size=11, stride=4, padding=2) ReLU
MAX_POOLING_1	MaxPool2d (kernel_size=3, stride=2)
CONV_RELU_2	Conv2d (64, 192, kernel_size=5, padding=2) ReLU
MAX_POOLING_2	MaxPool2d (kernel_size=3, stride=2)
CONV_RELU_3	Conv2d (in_channel=192, out_channel=384, kernel_size=3, padding=1) ReLU
CONV_RELU_4	Conv2d (in_channel=384, out_channel=256, kernel_size=3, padding=1) ReLU
CONV_RELU_5	Conv2d (in_channel=256, out_channel=256, kernel_size=3, padding=1) ReLU
MAX_POOLING_3	MaxPool2d (kernel_size=3, stride=2)
No need for implementation	AdaptiveAvgPool2d ((6, 6)) Dropout
LINEAR_RELU_1	Linear (256 * 6 * 6, 4096) ReLU
No need for implementation	Dropout
LINEAR_RELU_2	Linear (4096, 4096) ReLU
LINEAR_3	Linear (4096, num_classes=1000)

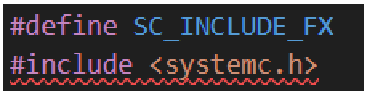
Observation and Optimization

- Add -O3 in the command to speed up the execution.

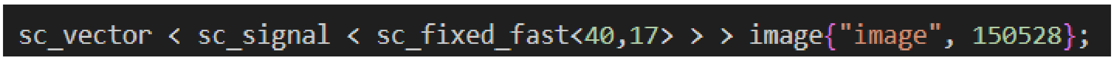


```
all:
    g++ -I . -I  $(INC_DIR) -L . -L $(LIB_DIR) -o $(O) $(C) $(LIB) $(RPATH) -O3
    ./run
```

- Read input feature maps in pipeline: read cat.txt at cycle 2 and read dog.txt at cycle 3. By doing so, the 2 images can pass through all the layers and grab the results for 2 continuous cycles.
- Use sc_fixed_fast<40,17> (40: total wl, 17: integer wl) to connect fixed point data from different modules. The sc_fixed_fast datatype is faster than sc_fixed during the simulation.



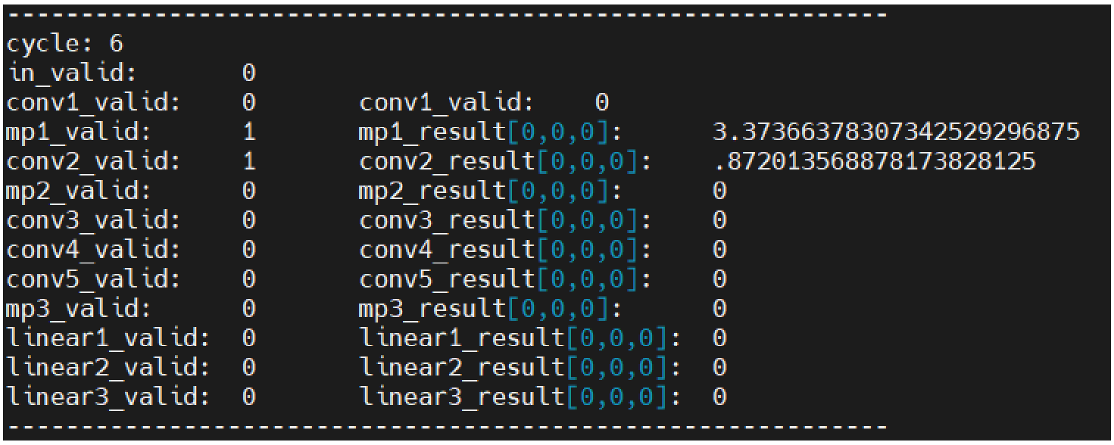
```
#define SC_INCLUDE_FX
#include <systemc.h>
```



```
sc_vector < sc_signal < sc_fixed_fast<40,17> > > image{"image", 150528};
```

- Use in_valid and out_valid signals in the AlexNet implementation layers to ensure only one layer is activated in one cycle for an image. By doing so, the calculation overhead and memory usage are decreased.

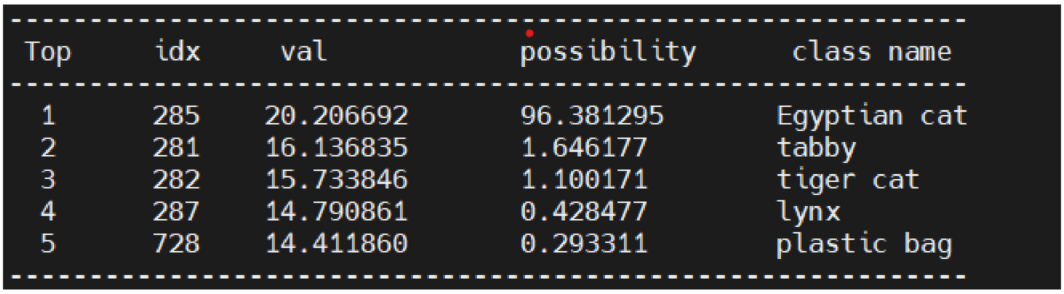
For example, in cycle 6, conv2 is performed for cat.txt, and mp1 is performed for dog.txt.



```
------------------------------------------------------------
cycle: 6
in_valid:       0
conv1_valid:    0       conv1_valid:    0
mp1_valid:      1       mp1_result[0,0,0]:      3.37366378307342529296875
conv2_valid:    1       conv2_result[0,0,0]:    .87201356887817382812 5
mp2_valid:      0       mp2_result[0,0,0]:      0
conv3_valid:    0       conv3_result[0,0,0]:    0
conv4_valid:    0       conv4_result[0,0,0]:    0
conv5_valid:    0       conv5_result[0,0,0]:    0
mp3_valid:      0       mp3_result[0,0,0]:      0
linear1_valid:  0       linear1_result[0,0,0]:  0
linear2_valid:  0       linear2_result[0,0,0]:  0
linear3_valid:  0       linear3_result[0,0,0]:  0
------------------------------------------------------------
```

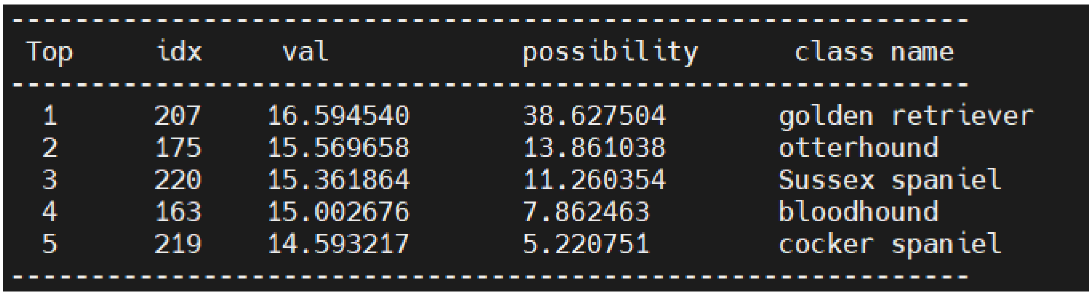
Result

Cat:



```
--------------------------------------------------------------
 Top      idx      val             possibility       class name
--------------------------------------------------------------
  1       285     20.206692        96.381295        Egyptian cat
  2       281     16.136835        1.646177         tabby
  3       282     15.733846        1.100171         tiger cat
  4       287     14.790861        0.428477         lynx
  5       728     14.411860        0.293311         plastic bag
--------------------------------------------------------------
```

Dog:



```
--------------------------------------------------------------
 Top      idx      val             possibility       class name
--------------------------------------------------------------
  1       207     16.594540        38.627504        golden retriever
  2       175     15.569658        13.861038        otterhound
  3       220     15.361864        11.260354        Sussex spaniel
  4       163     15.002676        7.862463         bloodhound
  5       219     14.593217        5.220751         cocker spaniel
--------------------------------------------------------------
```