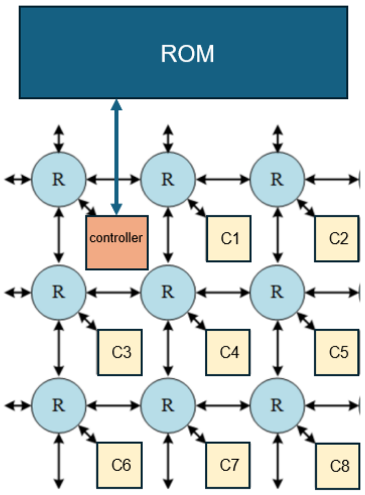
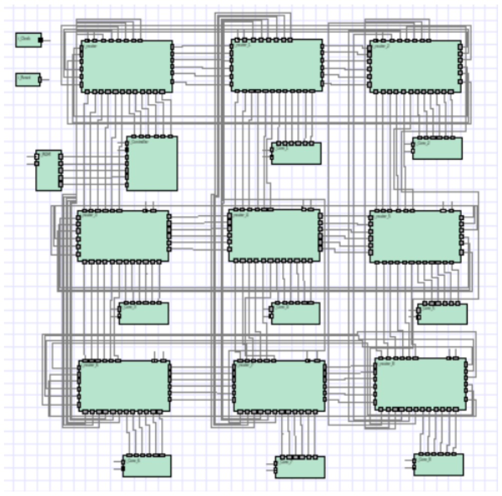
MLCHIP Final Project

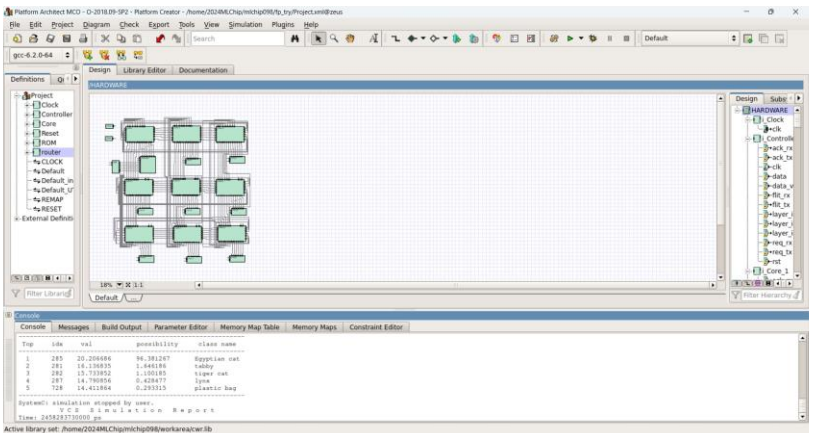
Simulation Result

PA Block Diagram





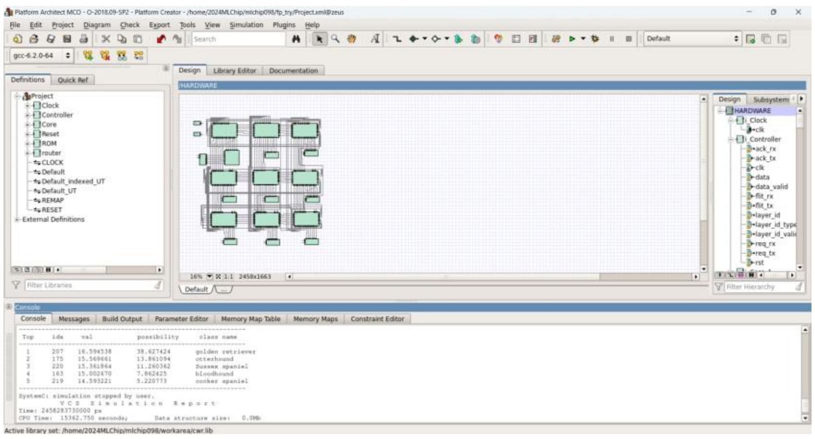
Cat:


Platform Architect MCO - O-2018.09-SP2 - Platform Creator - /home/2024MLChip/mlchip098/fp_try/Project.xml@zeus
File Edit Project Diagram Check Export Tools View Simulation Plugins Help
Console Messages Build Output Parameter Editor Memory Map Table Memory Maps Constraint Editor

```
------------------------------------------------------------------
 Top     idx     val              possibility      class name
------------------------------------------------------------------
  1      285   20.206686         96.381267        Egyptian cat
  2      281   16.136835         1.646186         tabby
  3      282   15.733852         1.100185         tiger cat
  4      287   14.790856         0.428477         lynx
  5      728   14.411864         0.293315         plastic bag
------------------------------------------------------------------
SystemC: simulation stopped by user.
           V C S   S i m u l a t i o n   R e p o r t
Time: 2458283730000 ps
```

Active library set: /home/2024MLChip/mlchip098/workarea/cwr.lib

Dog:


Platform Architect MCO - O-2018.09-SP2 - Platform Creator - /home/2024MLChip/mlchip098/fp_try/Project.xml@zeus
File Edit Project Diagram Check Export Tools View Simulation Plugins Help
Console Messages Build Output Parameter Editor Memory Map Table Memory Maps Constraint Editor

```
------------------------------------------------------------------
 Top     idx     val              possibility      class name
------------------------------------------------------------------
  1      207   16.594538         38.627424        golden retriever
  2      175   15.569661         13.861094        otterhound
  3      220   15.361864         11.260362        Sussex spaniel
  4      163   15.002670         7.862425         bloodhound
  5      219   14.593221         5.220773         cocker spaniel
------------------------------------------------------------------
SystemC: simulation stopped by user.
           V C S   S i m u l a t i o n   R e p o r t
Time: 2458283730000 ps
CPU Time:  15362.750 seconds;       Data structure size:   0.0Mb
```

Active library set: /home/2024MLChip/mlchip098/workarea/cwr.lib

Implementation

NoC Router

Architecture	3x3 Torus NoC architecture
Routing algorithm	XY routing (deadlock free)
Switching	Wormhole switching
Buffer size	one flit (34 bits), depth=1
Virtual channel	No implementation
Flow control	ack/nack protocol (buffered flow control)

In HW4, I replace core0 with the controller and separate the layers into 11 modules. However, to maintain the Torus architecture, I still need to implement 16 routers, and there are 4 cores left unused. Thus, I combine the convolution and max pooling together to achieve the calculation in 8 cores, reducing the core number to 8 and router number to 9. Although the last layer (LINEAR_3) is completed at core6, which seems to need 2 hops to go back to the controller, the Torus architecture helps to reduce the times of data transferring to the least.

The Router Pipeline

At the first cycle, the header flit arrives, and the packet is directed to the buffer of the input port. Then, the routing direction is computed, and the output priority is set. After that, if the packet gets the authority, output the flit and clear the buffer when in_ack is back. Otherwise, the flit stays in the buffer and wait until the output port is free.

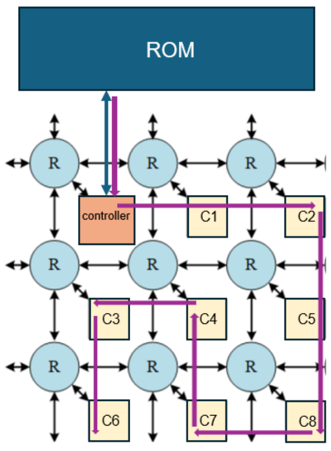
Flit Format

	33	32	31~28	27~24	23	22	21	20~0
header flit	1	0	source_id	dest_id	input	weight	bias	0
body flit	0	0	data value					
tail flit	0	1	data value					

When controller receives data from ROM, it starts organizing the packet. For the header flit, it contains the source_id, dest_id and use 3 bits to indicate whether the data type is input, weight or bias. This enables the cores to distribute them into different space and let PE calculate in a proper way. The controller is placed at the original core0 place, so the source_id is always 0 if the packet is sent by the controller. The dest_id is calculated by the controller or cores to send packet to the correct destination.

AlexNet

PE id	AlexNet Layer (details in HW1 report)
1	CONV_RELU_1 & MAX_POOLING_1
2	CONV_RELU_2 & MAX_POOLING_2
5	CONV_RELU_3
8	CONV_RELU_4
7	CONV_RELU_5 & MAX_POOLING_3
4	LINEAR_RELU_1
3	LINEAR_RELU_2
6	LINEAR_3



With 8 layers, the data is just enough to walk around and back to the controller. This allows fewer data movement, saving time and energy.

File Description

File list	Description
data/	Contain 2 input images (size: 224*224*3), weights, bias and classes.
Makefile	Automate the process of compiling and linking source code files. Use -O3 to speed up the simulation. It takes about 15 minutes to finish processing one image. Not used in PA.
main.cpp	Connect all signals in different modules. Define sc_start. Can create waveform using sc_trace. Not used in PA.
clockreset.h clockreset.cpp	Define clk and rst signals.
ROM.h ROM.cpp	Store data. The controller takes images, weights and bias from it. Change IMAGE_FILE_NAME if you want to process another image.
controller.h	Get data from ROM, packet them and send to appropriate location. After the AlexNet layers are finished, the result will be sent back to it. The controller does the SoftMax and print the result. The simulation will stop by sc_stop when the result of one image is printed.

router.h	Determine the direction of the flit. There may be many approaches, and my implementation is inherited from HW4 with core number decreases to 8. This file is not used for co-sim in PA.
router.v	Rewrite router.h using Verilog.
core.h	Receive data and packet them. Send the packetized data to its PE and output the results when the PE finishes its calculation.
pe.h ~~pe.cpp~~	Different AlexNet layers are mapped to different PEs (listed above). Initially, the dest_id is determined. When the core sends all necessary data to it, the calculation of the layer can begin. Return the result when the computation is finished.
Project.xml	Store the settings in PA, including SystemC modules, Verilog module, parameter settings, port connections, etc.

Observation

- Platform Architect
  - **sc_uint → sc_lv**: When compiling the HDL codes, we need to ensure the data types of input/output ports are matched. Therefore, we have to change vector port mapping type in PA.
  - **Auto connect**: Right click the ports and auto connect ports with the same names (clk, rst).
  - **Parameters**: Use the parameter editor to set up the clk cycle time, reset ticks and core_id.
  - **pe.cpp**: The functions inside pe.cpp are not found when I run the simulation in PA, so I move the codes into pe.h and pe.cpp is not used in this project.
  - **Debug**: It is hard to debug since I did not find the way to dump the waveform when using PA to co-simulate the designs.
  - **Simulation time**: The simulation time is much longer than HW4.