A Heterogeneous Full-stack AI Platform for Performance Monitoring and Hardware-specific Optimizations

Zikang Zhou
State Key Laboratory of ASIC and System, Fudan University
Shanghai, China
zkzhou20@fudan.edu.cn

Chao Fu
State Key Laboratory of ASIC and System, Fudan University
Shanghai, China
cfu19@fudan.edu.cn

Ruiqi Xie
State Key Laboratory of ASIC and System, Fudan University
Shanghai, China
rqxie19@fudan.edu.cn

Jun Han
State Key Laboratory of ASIC and System, Fudan University
Shanghai, China
junhan@fudan.edu.cn

Introduction

- **Rapid development of DNN puts forward higher requirements for hardware computing.**
- **Many general accelerators have been released**
  - GPU
  - Google TPU
  - NVDLA
- Previous works also point out even for the same neural network, the performance and energy will be greatly various when deployed on different hardware accelerators.

What are the hardware-friendly networks targeted specific accelerators?

Is there any optimization toward networks once the hardware is given?

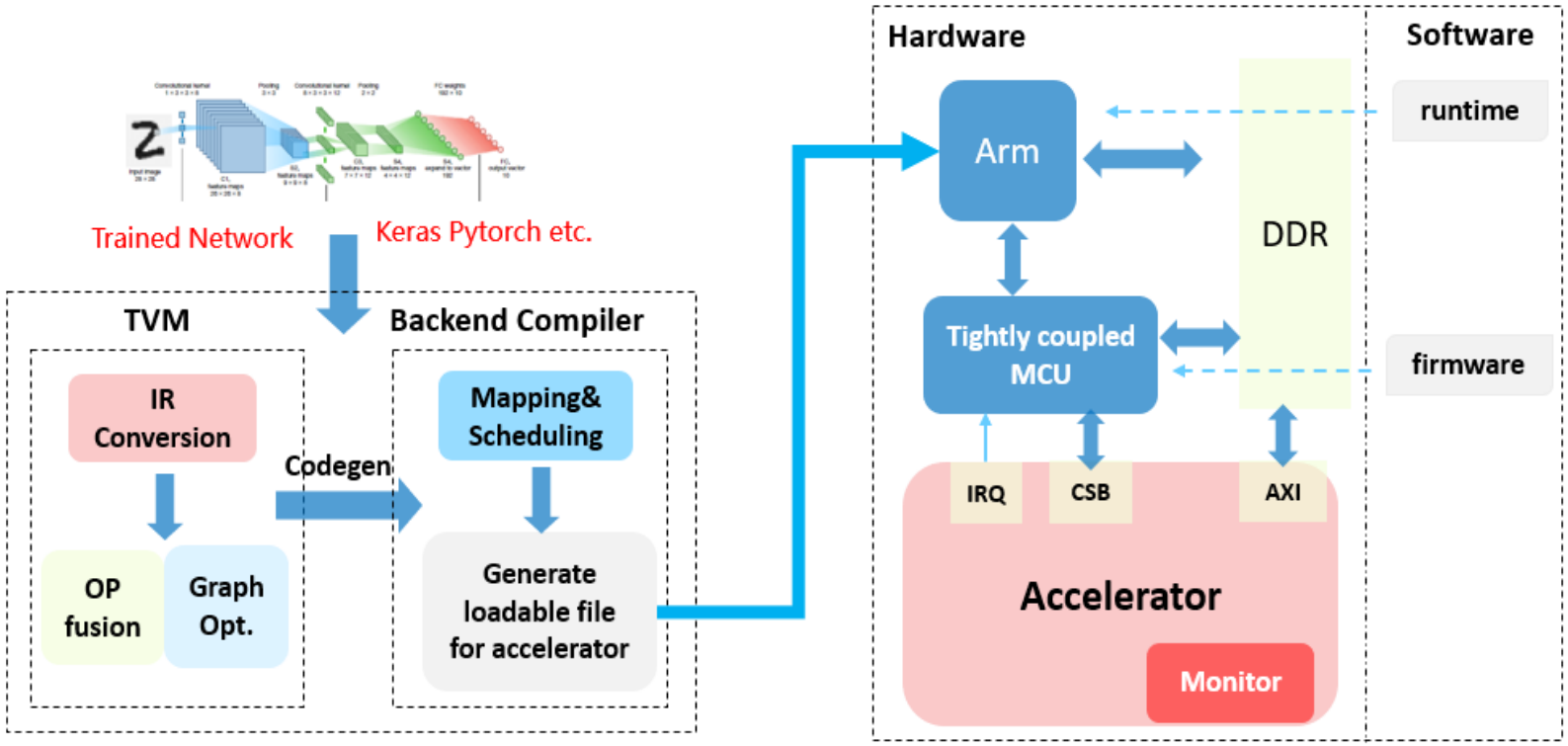
Introduction of Full-stack AI evaluation Platform

Platform Overview

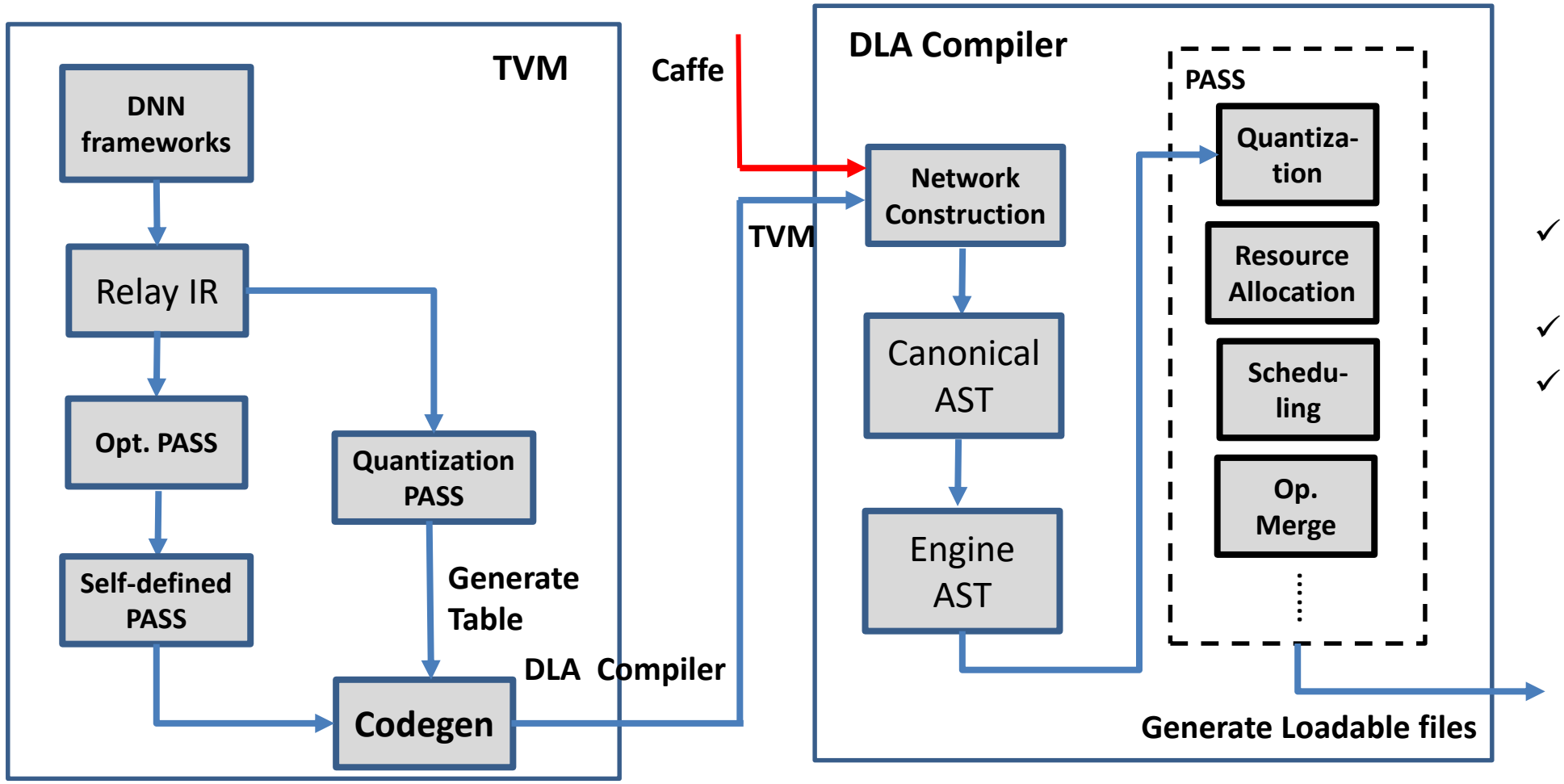
- **This platform is based on opensource TVM and NVDLA, in order to realize representative and versatility.**

Components：

- **Compiler:** compile the network model under common frameworks into a loadable file readable by the subsequent hardware platform.
- **Runtime：** Schedule computing tasks and perform user interaction.
- **Hardware：** Accelerate AI computing as well as monitoring.
- Further support various neural networks, like: Yolo and LSTM etc.



Compiler

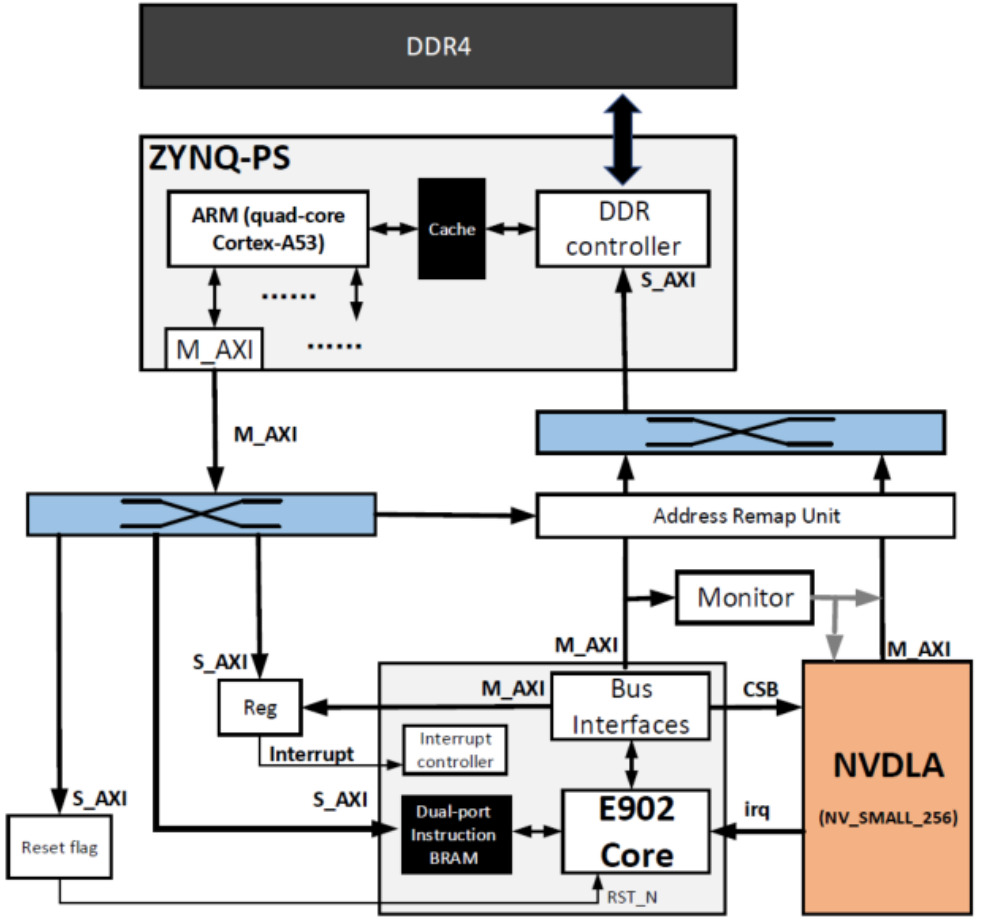


- ✓ Expand the optimization space.
- ✓ Have better versatility.
- ✓ Have low development cost.

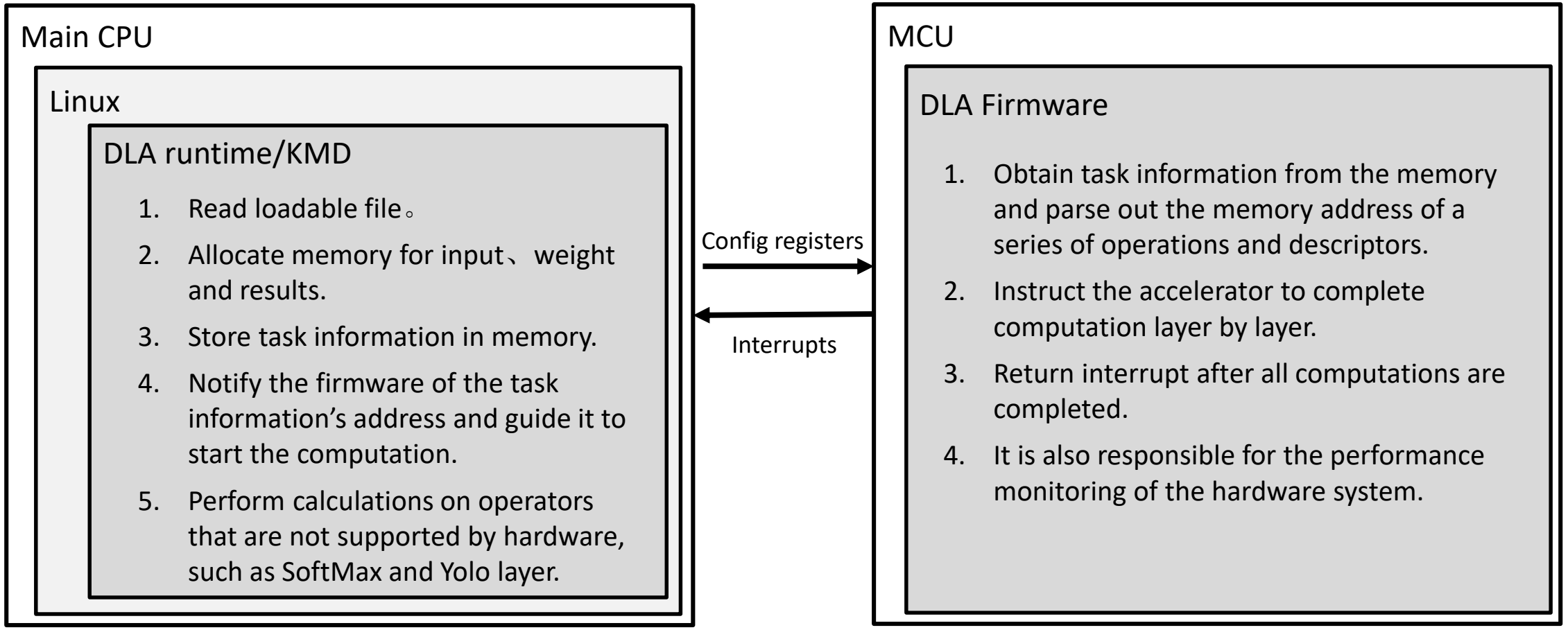
Hardware

Hardware introduction

- This platform is a multi-core heterogeneous system, consisting of a main CPU, an accelerator, and an MCU tightly coupled with the accelerator.
- Arm runs the Linux and DLA KMD/runtime.
- E902 runs the firmware.
- Computing units which supports various DNN operators are integrated in accelerator.
- System **performance monitor** monitoring calculation time, utilization and bandwidth of each computing unit of the accelerator.



Software


Main CPU
Linux
DLA runtime/KMD
1. Read loadable file。
2. Allocate memory for input、 weight and results.
3. Store task information in memory.
4. Notify the firmware of the task information's address and guide it to start the computation.
5. Perform calculations on operators that are not supported by hardware, such as SoftMax and Yolo layer.
Config registers
Interrupts
MCU
DLA Firmware
1. Obtain task information from the memory and parse out the memory address of a series of operations and descriptors.
2. Instruct the accelerator to complete computation layer by layer.
3. Return interrupt after all computations are completed.
4. It is also responsible for the performance monitoring of the hardware system.

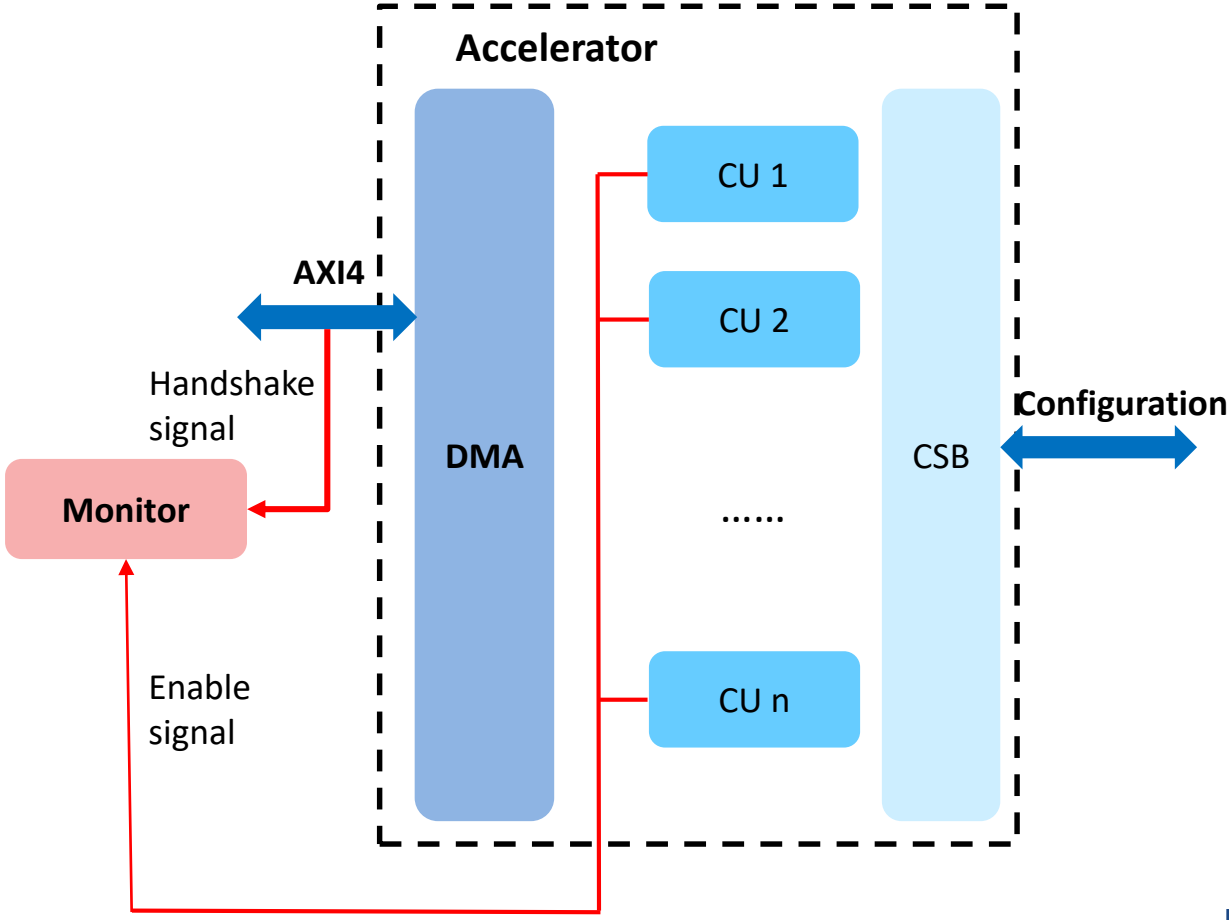
Monitor

Monitor introduction

- Running time and effective computation time of DLA's computation units.
- The number of effective memory accesses and the total memory access time of the accelerator.

Performance profiling

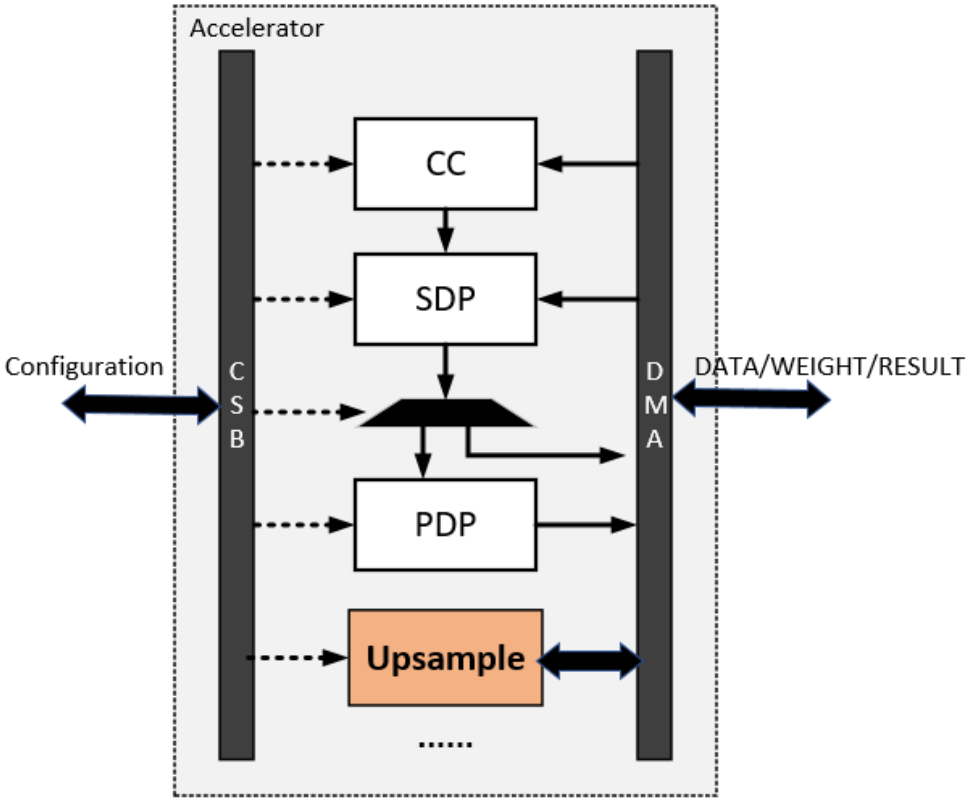
- Latency
- Computation density
- DLA's memory access time and bandwidth
- Running time of each unit



Accelerator

Accelerator introduction

- A variety of computing units are integrated in the accelerator to perform calculations on different operators in the network.
- CC is mainly for performing operations on matrix multiplications such as convolution, deconvolution, and full connection.
- SDP calculates elementwise operations such as biasing, activation and regularization.
- PDP is for pooling operations.
- New operators' supports can be added as needed. Currently on this platform we additionally support LSTM and up-sample.



How the network structures effect performance on target accelerator?

Experiments setup

1、The hardware part is implemented in Xilinx ZCU104 board.

2、Arm Cortex-A53 runs at 1.25GHz，RISC-V E902 runs at 20MHz，accelerator runs at 100MHz.

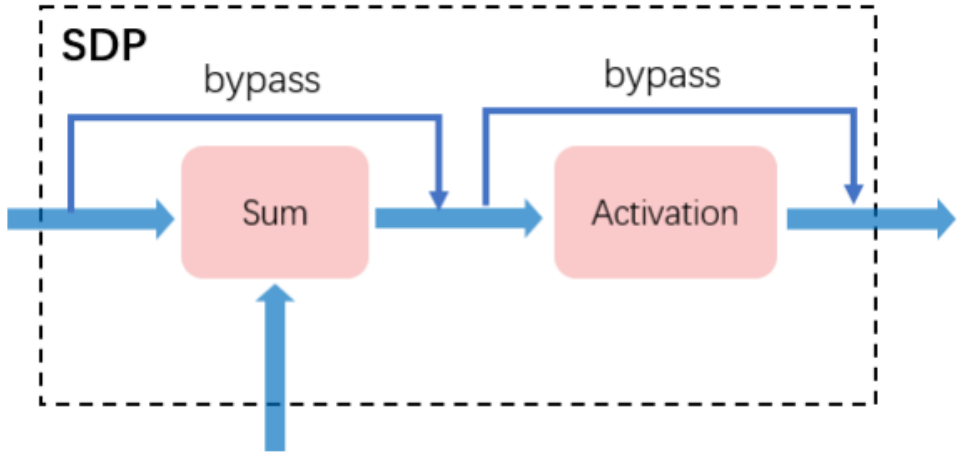
3、Accelerator is in “nv_small” config.

Influence of Operation order

- **Mathmatical**

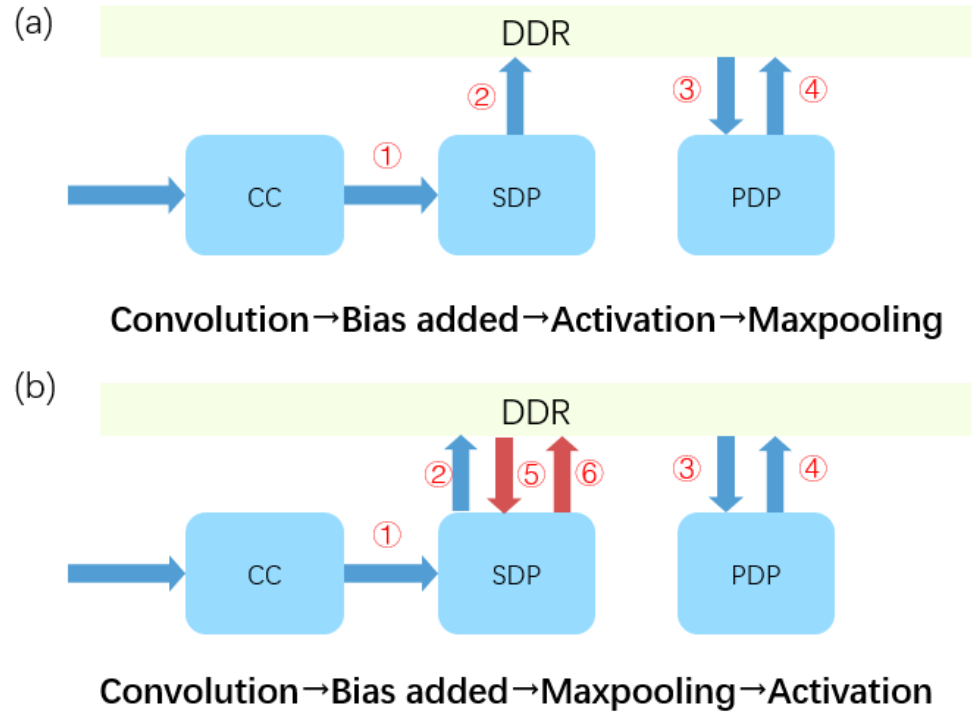
$$Maxpool(Activate(x)) = Activate(Maxpool(x))$$

- **Accelerator processing flow in SDP**



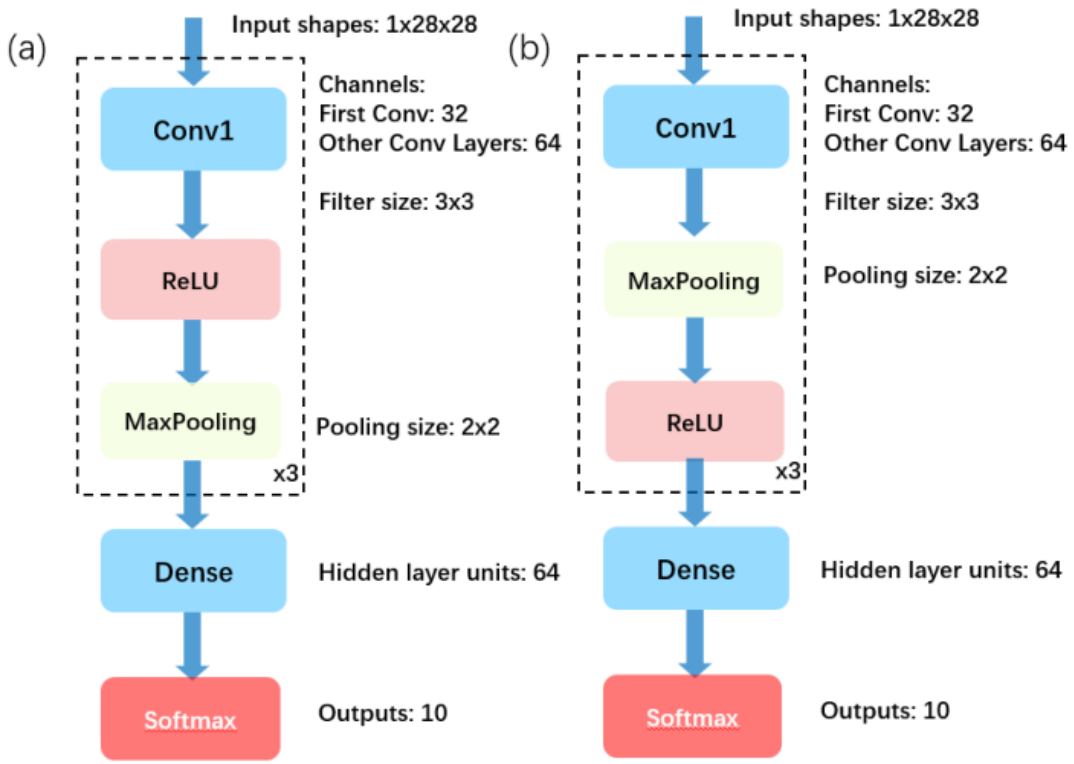
Theoretically, **Activate(Maxpool(x))** may has less computation.

However, **Activate(Maxpool(x))** in this platform will cause extra 2 data transfer.



Influence of Operation order

Experiment models

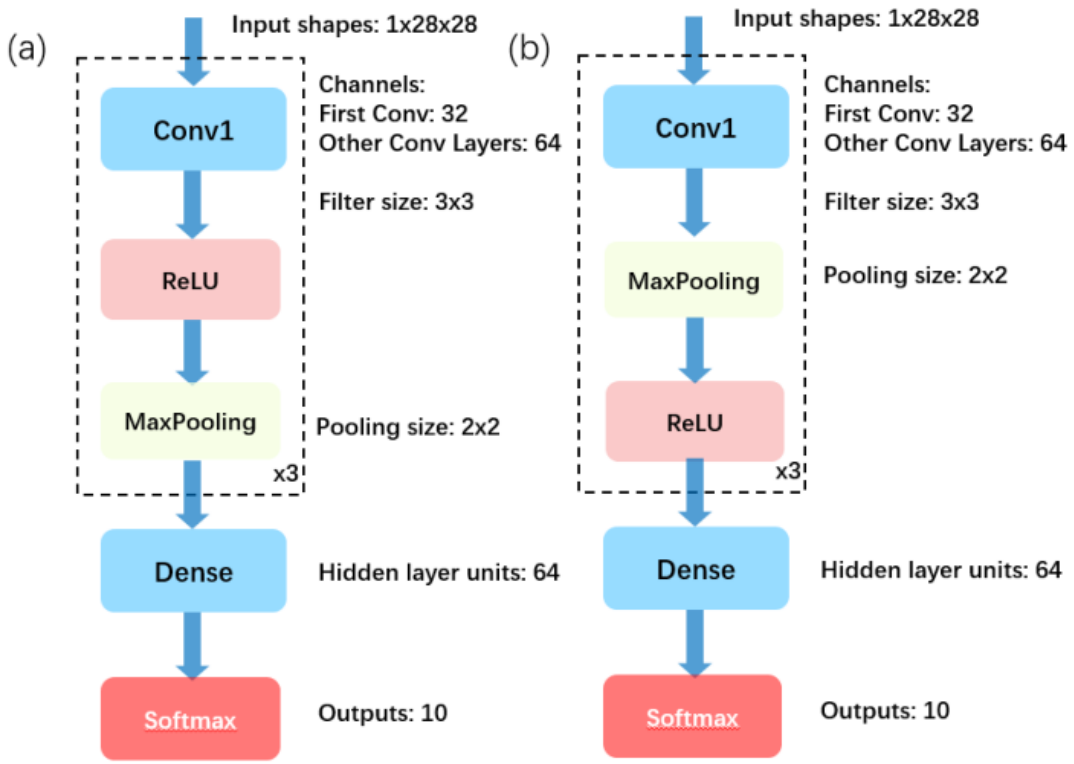


Results

Indicators	Pattern	
	Activation->Pooling	Pooling->Activation
MAC valid time / cycles	34364	34364
SDP valid time / cycles	30032	**37235**
PDP enable time / cycles	38353	38269
Number of DLA's writes	4638	**5522**
Write time / cycles	18552	**22088**
Number of DLA's reads	12872	**13796**
Read time / cycles	141380	**149253**
Accelerator running time / s	0.006617	**0.008179**
Total running time / s	0.011946	**0.014190**

Influence of Operation order

Experiment models

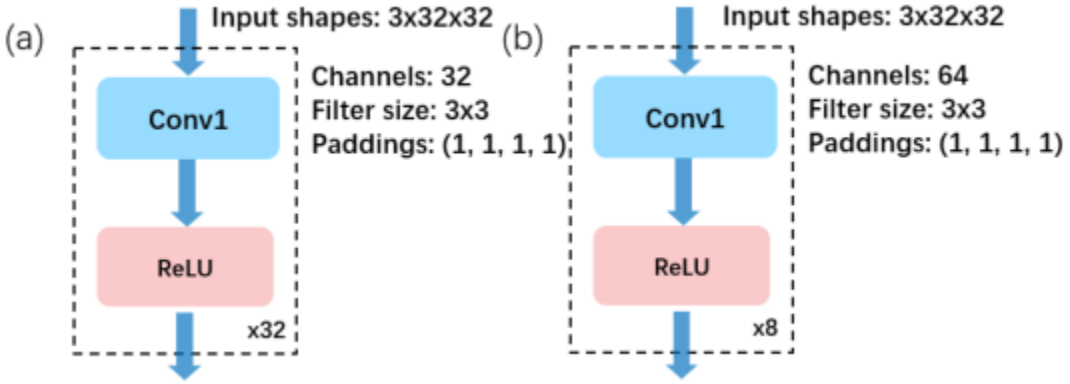


Results

- 7072 elements in total and cause 20us latency deterioration.
- It can be estimated when the feature map is large, and the pooling operations are frequent, the sequence of “Activation-Pooling” will become a more critical factor of performance.
- It can be inferred that for YoloV3-tiny inference, it will cause approximately 4ms performance deterioration.

Effect of Depth and Width of Neural Network

Experiment models

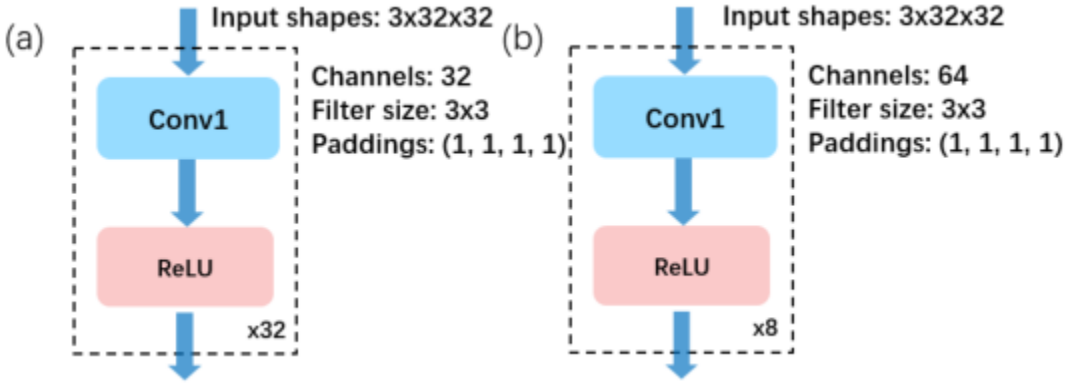


Results

Indicators	Network Structure	
	32 layers- 32 channels	8 layers – 64 channels
MAC valid time / cycles	1179648	1105920
SDP valid time / cycles	**1048705**	524288
PDP enable time / cycles	0	0
Number of DLA's writes	**131072**	65536
Write time / cycles	**973483**	262144
Number of DLA's reads	**164332**	91096
Read time / cycles	**1785690**	992699
Accelerator running time / s	**0.046491s**	0.022934s
Total running time / s	**0.057642s**	0.030127s

Effect of Depth and Width of Neural Network

Experiment models



Results

- For the networks with similar parameters and computation, the one whose depth is deeper having a larger size of feature maps, which increases memory access and operations of bias-adding as well as activation.
- Deeper networks indicates more overhead on data transfer and accelerator configuration.
- Trade deeper network with wider network.

Effect of unsupported operators

Experiment results

Network Type	Accelerator Running Time	Hardware Unsupported Operators Running Time	Total Running Time
Yolov3-tiny	0.40108s	Yolo Layer 0.12816s	0.53924s
Resnet-50	0.93974s	Softmax Layer 0.26228s	1.21202s

- Yolo Layer's running time is about 1/3 of the running time of Yolov3-tiny.
- Softmax Layer's running time is about ¼ of the running time of Resnet50.
- Replace the unsupported operators with the combination of supported operators.

Conclusion and Future works

Conclusion and Future works

- Expand the search space for better performance in specific hardware deployment in NAS.
  - Operation order
  - Trade deeper networks with wider networks
  - Combine operators with operators supported by hardware
- Split the evaluation hardware part into several FPGAs for better scalability.
  - We have split the hardware into two FPGAs using Chip2Chip connection.
  - Have negative effect on bandwidth.

Thanks for listening!