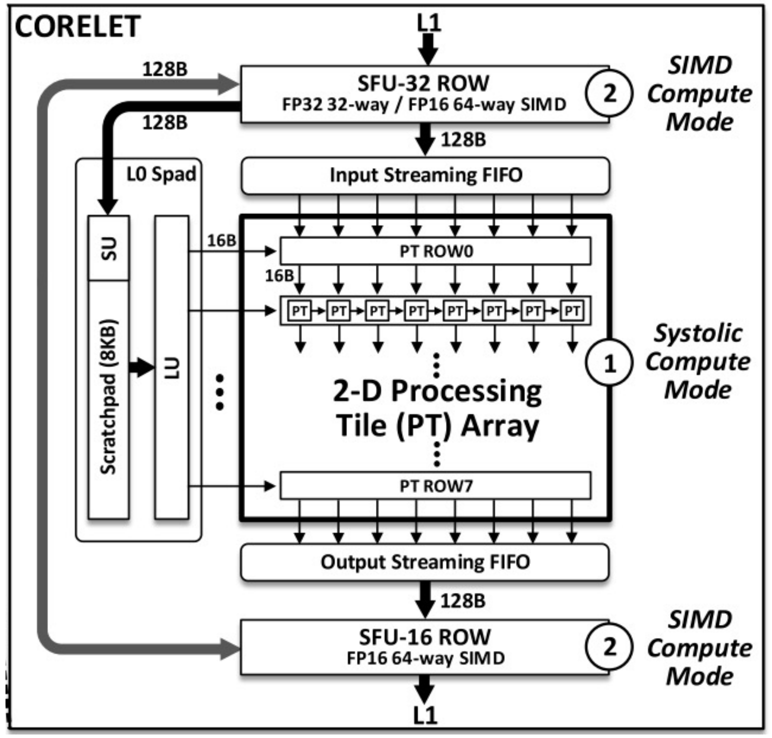
ECE260B Winter 22

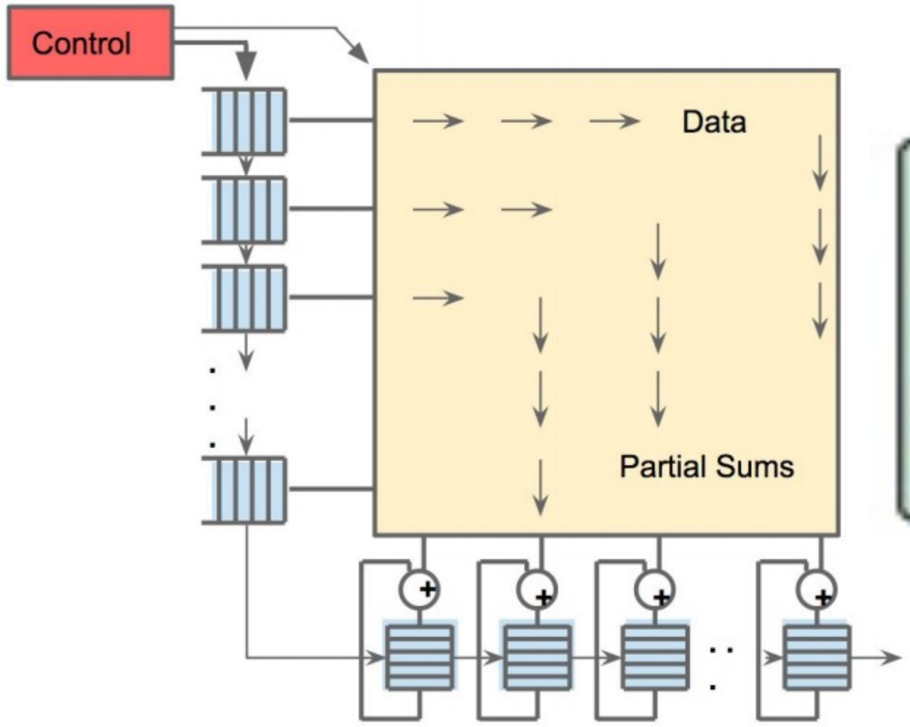
Project Description

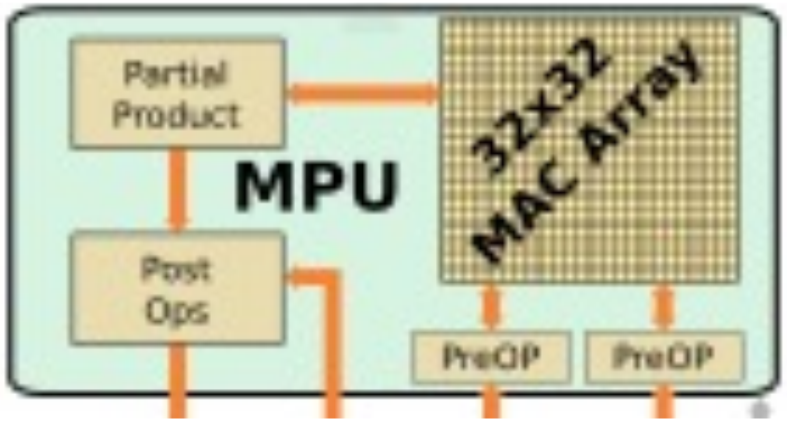
Prof. Mingu Kang

UCSD Computer Engineering

2D Systolic-based Architecture



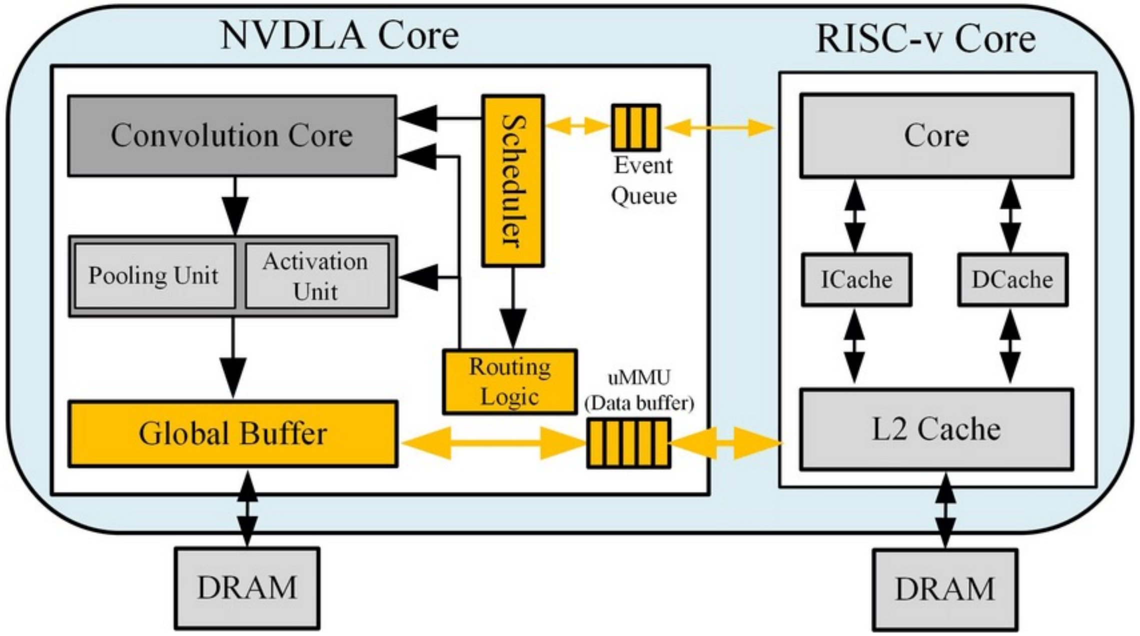




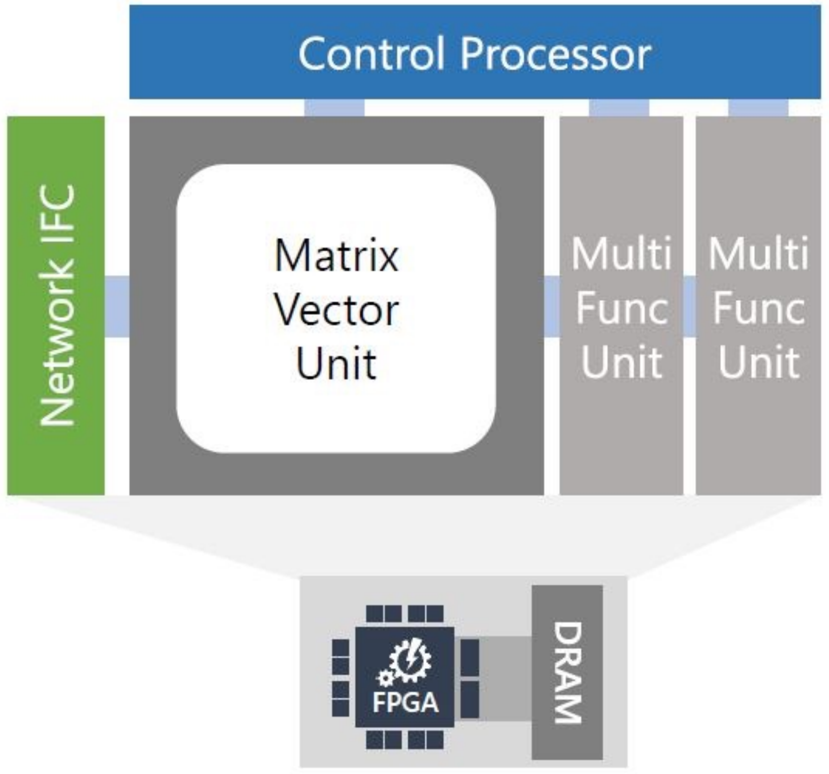
IBM Rapid **Google TPU** **Intel Nervana**

J.Oh, “A 3.0 TFLOPS 0.62V Scalable Processor Core for High Compute Utilization AI Training and Inference“
https://arxiv.org/pdf/1704.04760.pdf
https://fuse.wikichip.org/news/3270/intel-axes-nervana-just-two-months-after-launch/

1-D Vector Processor based Architecture



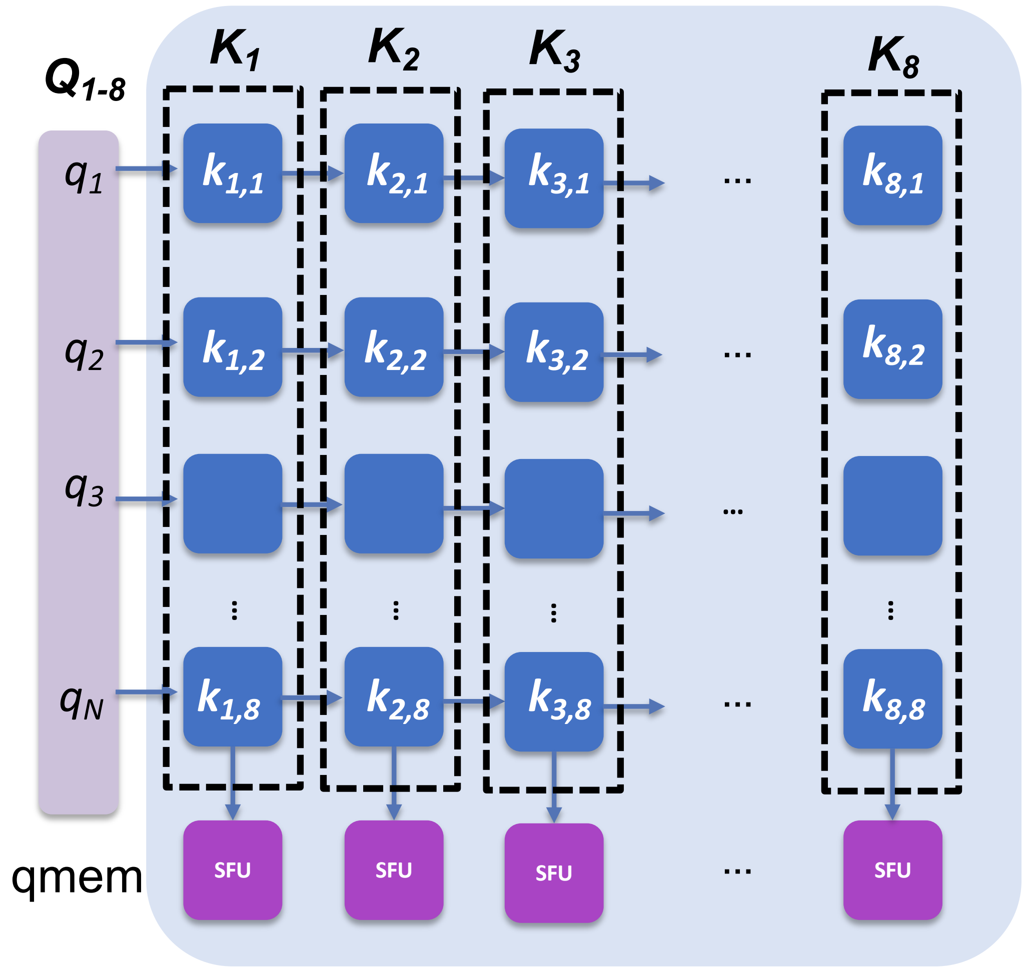
NVIDA NVDLA



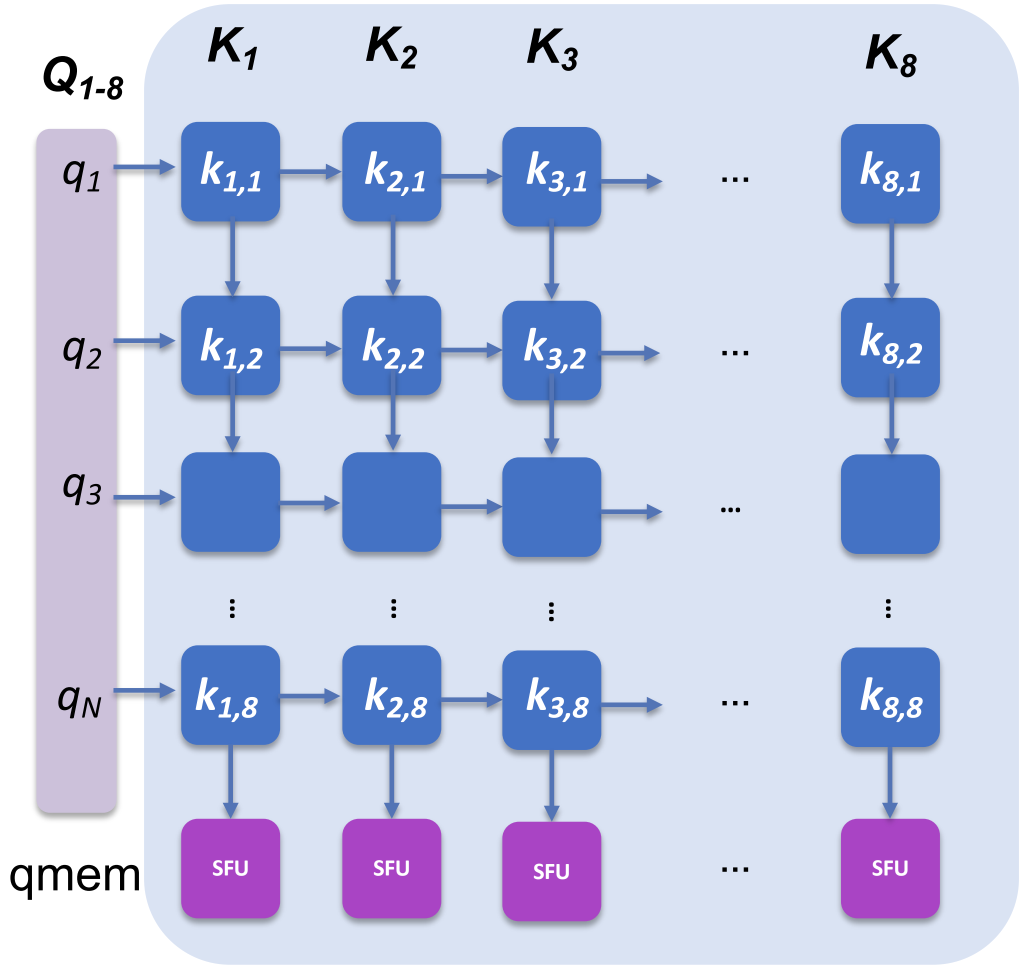
Microsoft Brainwave

https://www.researchgate.net/figure/Elastic-NVDLA-and-RISC-V-implement-DNNGuard-architecture_fig4_339920895
https://www.servethehome.com/microsoft-shows-off-project-brainwave-persistent-inferencing-from-fpga-cache/microsoft-brainwave-soft-dpu-architecture/

Architecture

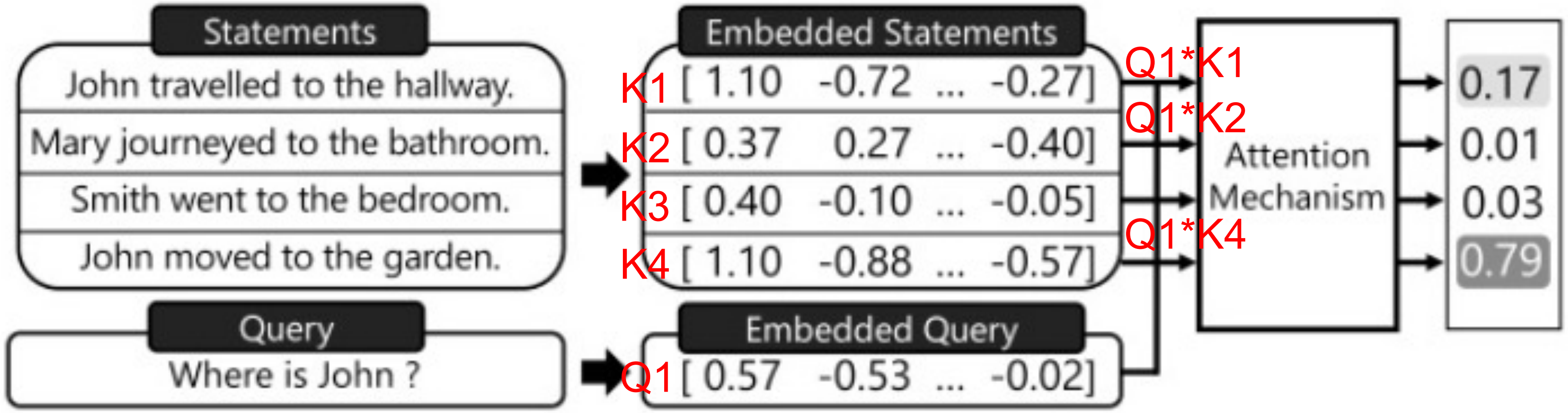


1-D vector processor architecture



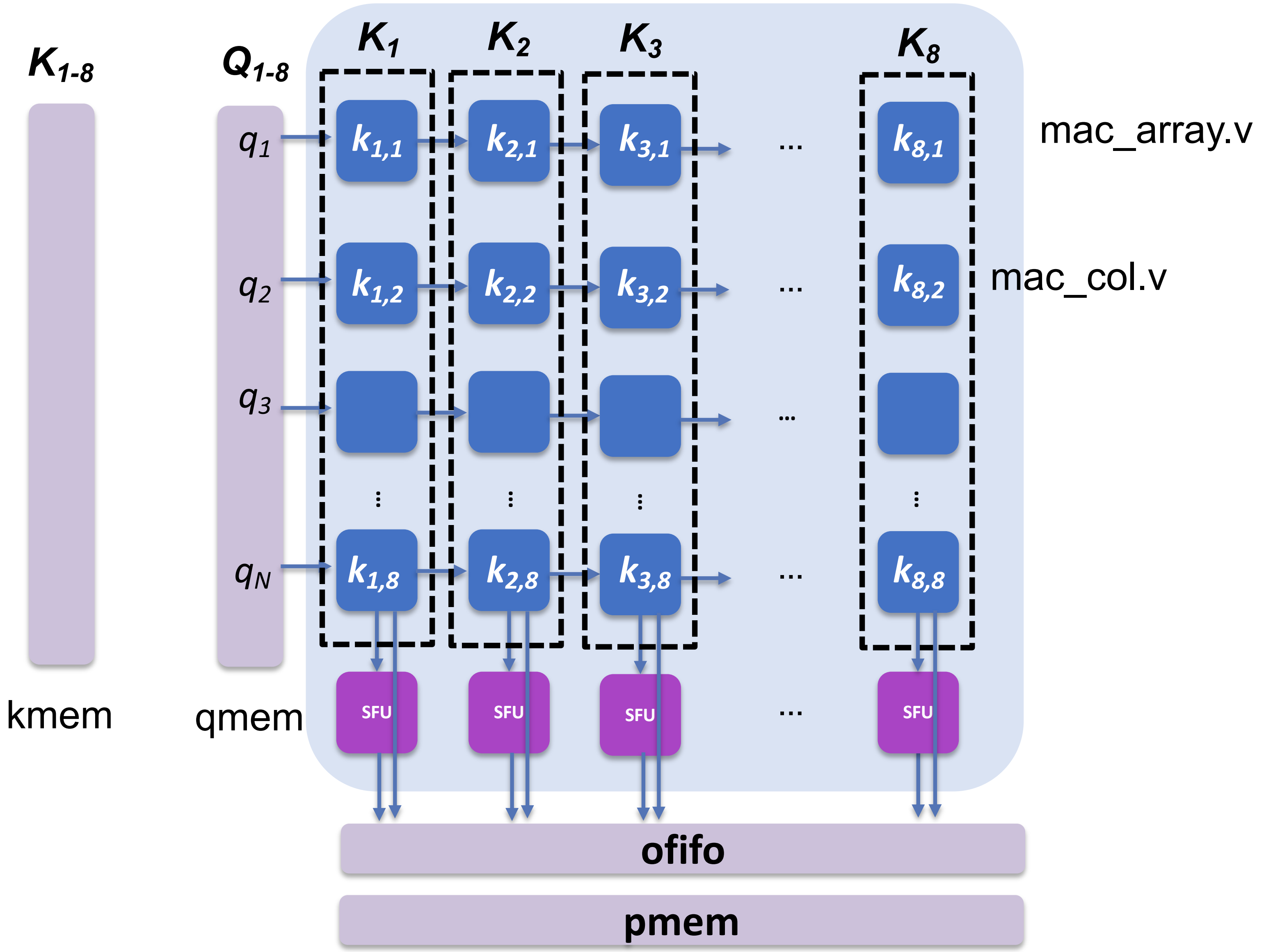
2D-systolic array architecture

Front-end Computation in Transformer Model for Natural Language Processing

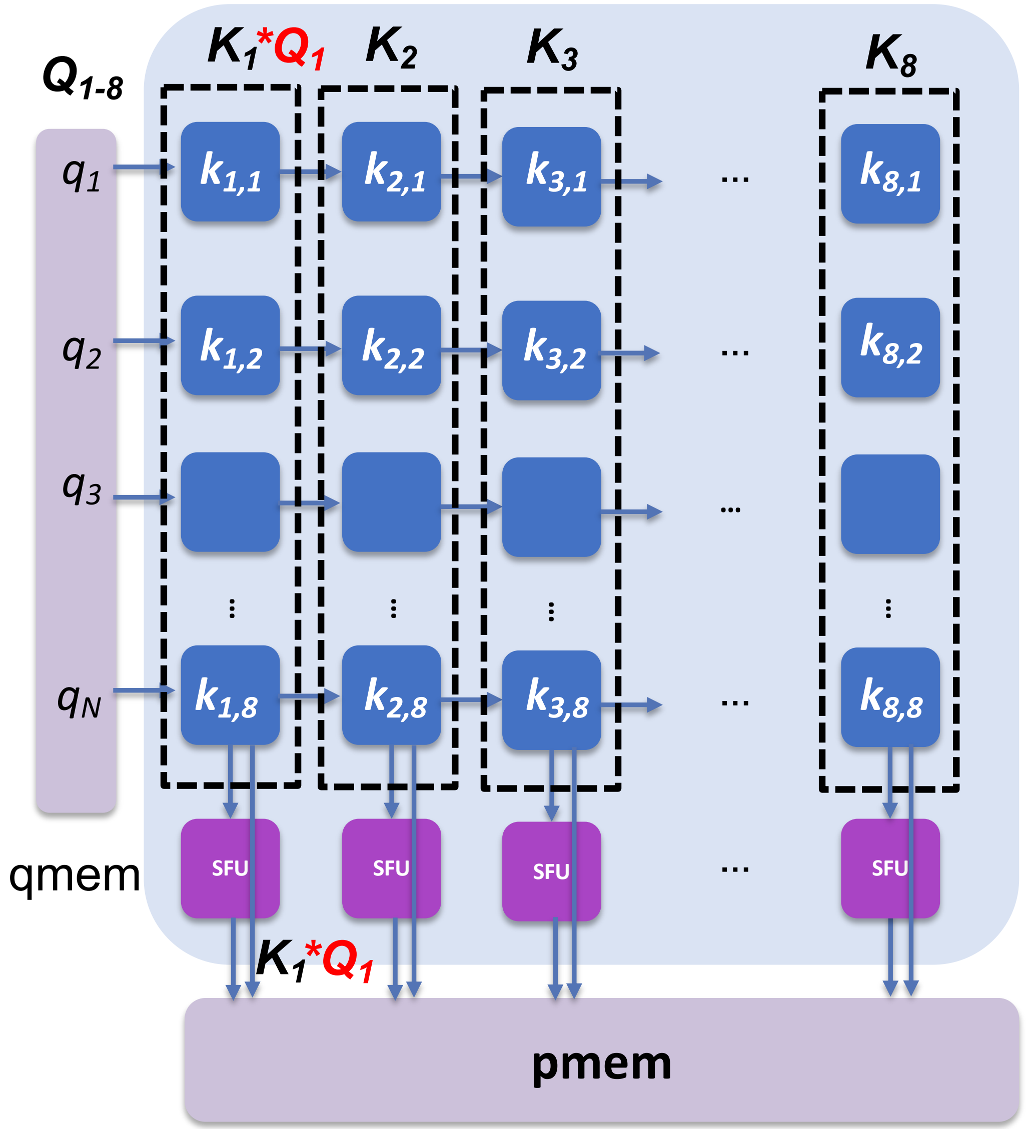

Statements
John travelled to the hallway.
Mary journeyed to the bathroom.
Smith went to the bedroom.
John moved to the garden.
Query
Where is John ?
Embedded Statements
K1 [1.10 -0.72 ... -0.27]
K2 [0.37 0.27 ... -0.40]
K3 [0.40 -0.10 ... -0.05]
K4 [1.10 -0.88 ... -0.57]
Embedded Query
Q1 [0.57 -0.53 ... -0.02]
Q1*K1
Q1*K2
Q1*K4
Attention Mechanism
0.17
0.01
0.03
0.79

Facebook bAbi QA task processing

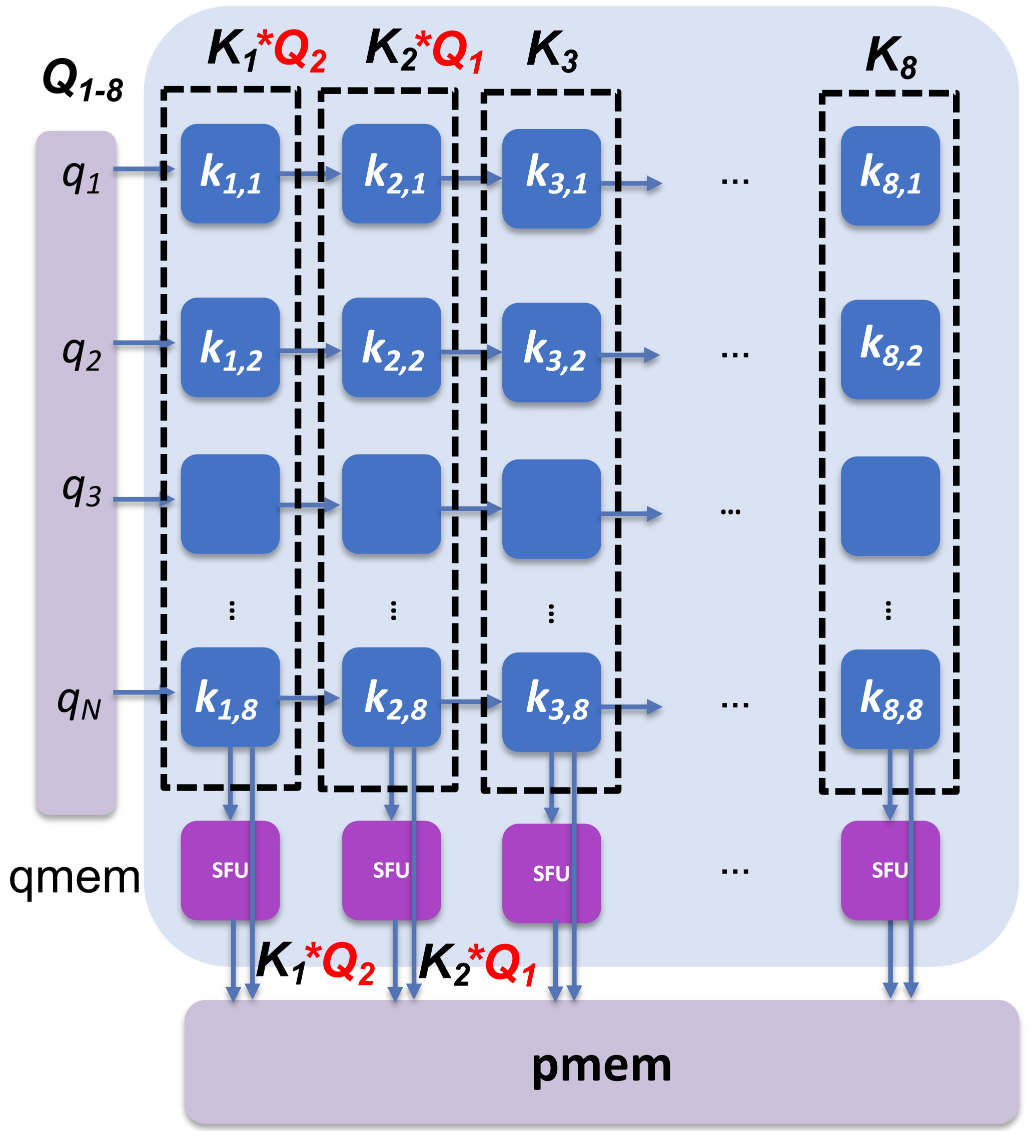
Architecture



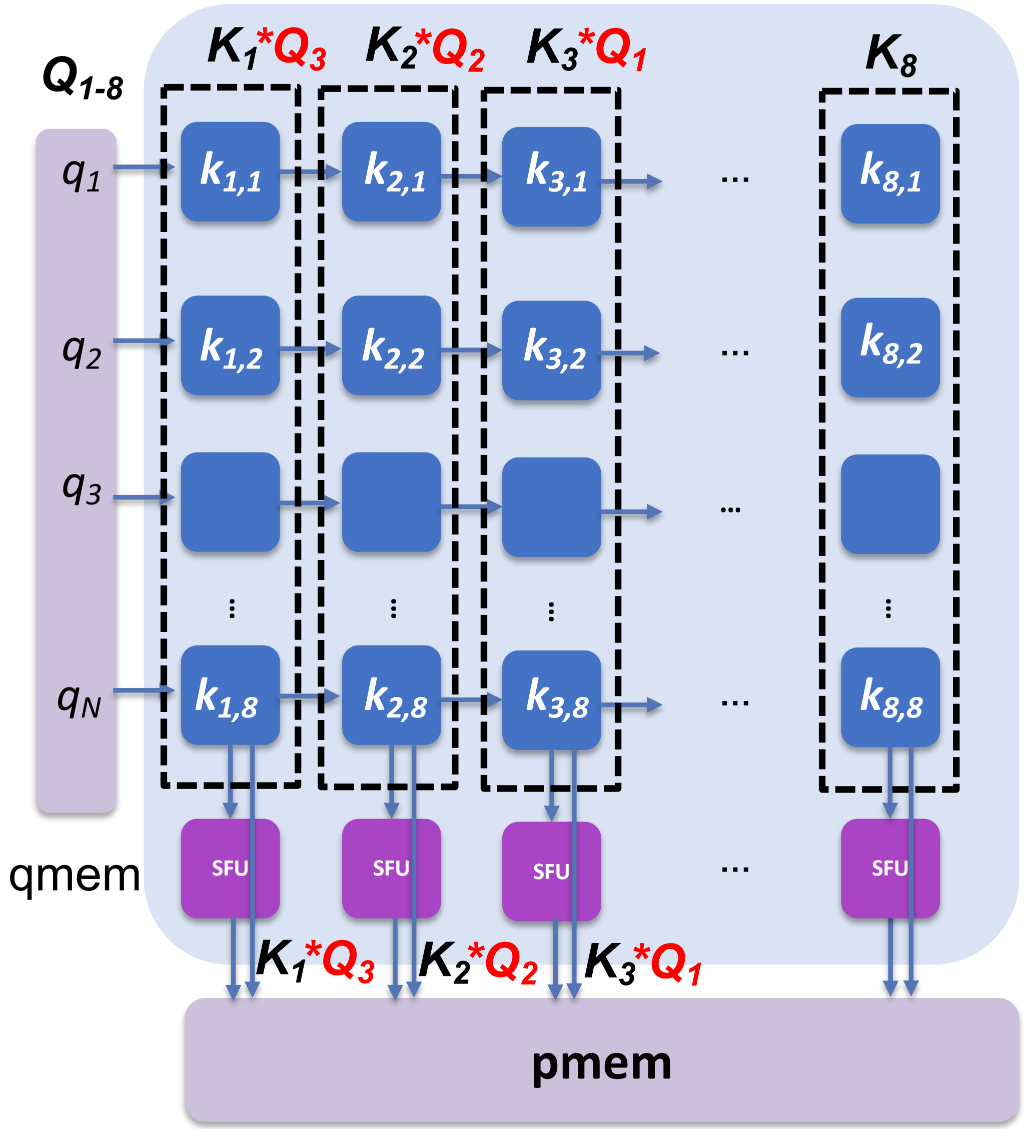
Computation



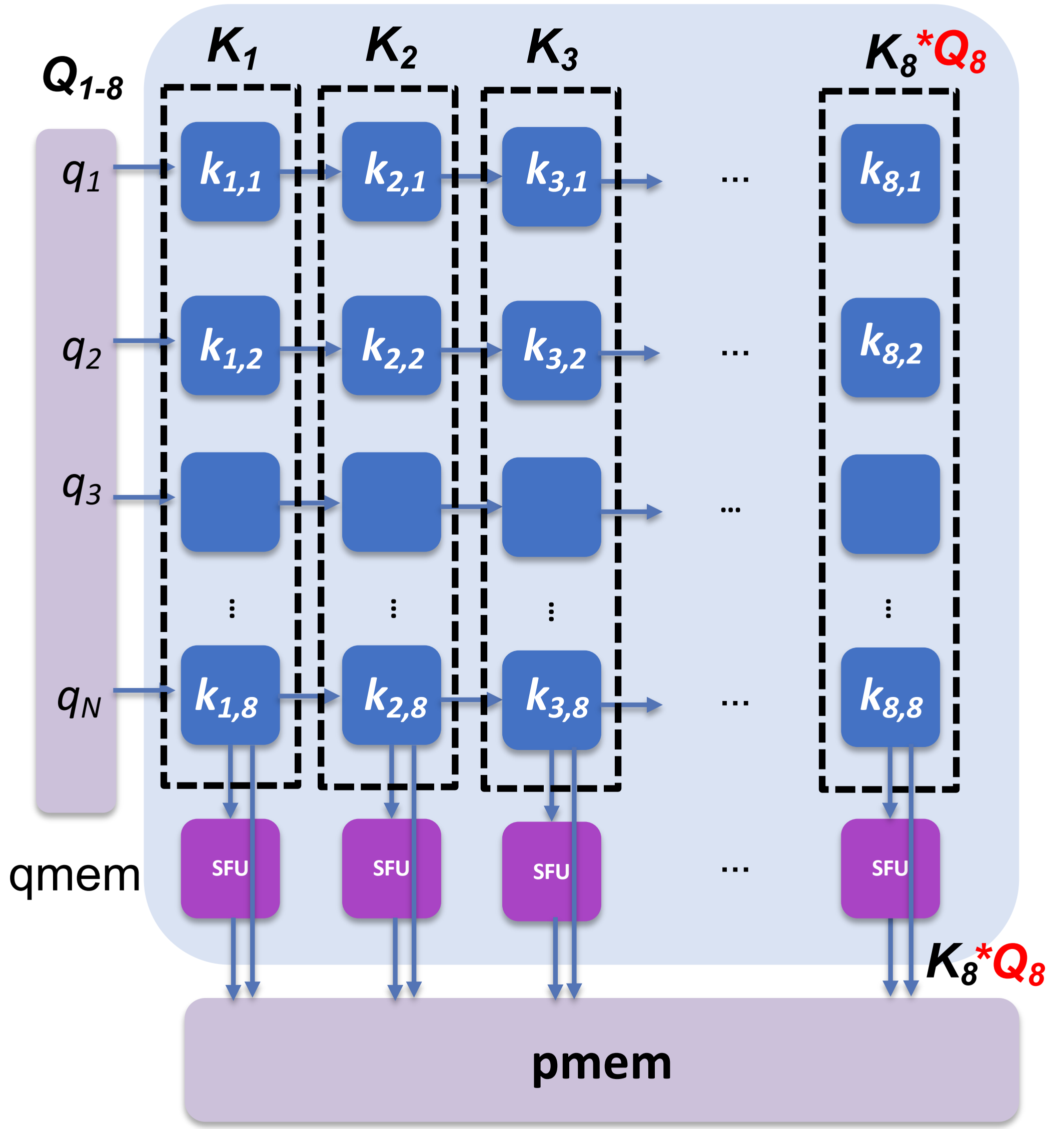
Computation



Computation



Computation



Processing Stage

- Stage1: memory write stage

  - qmem & kmem: fill both memory with “qdata.txt” and “kdata.txt”

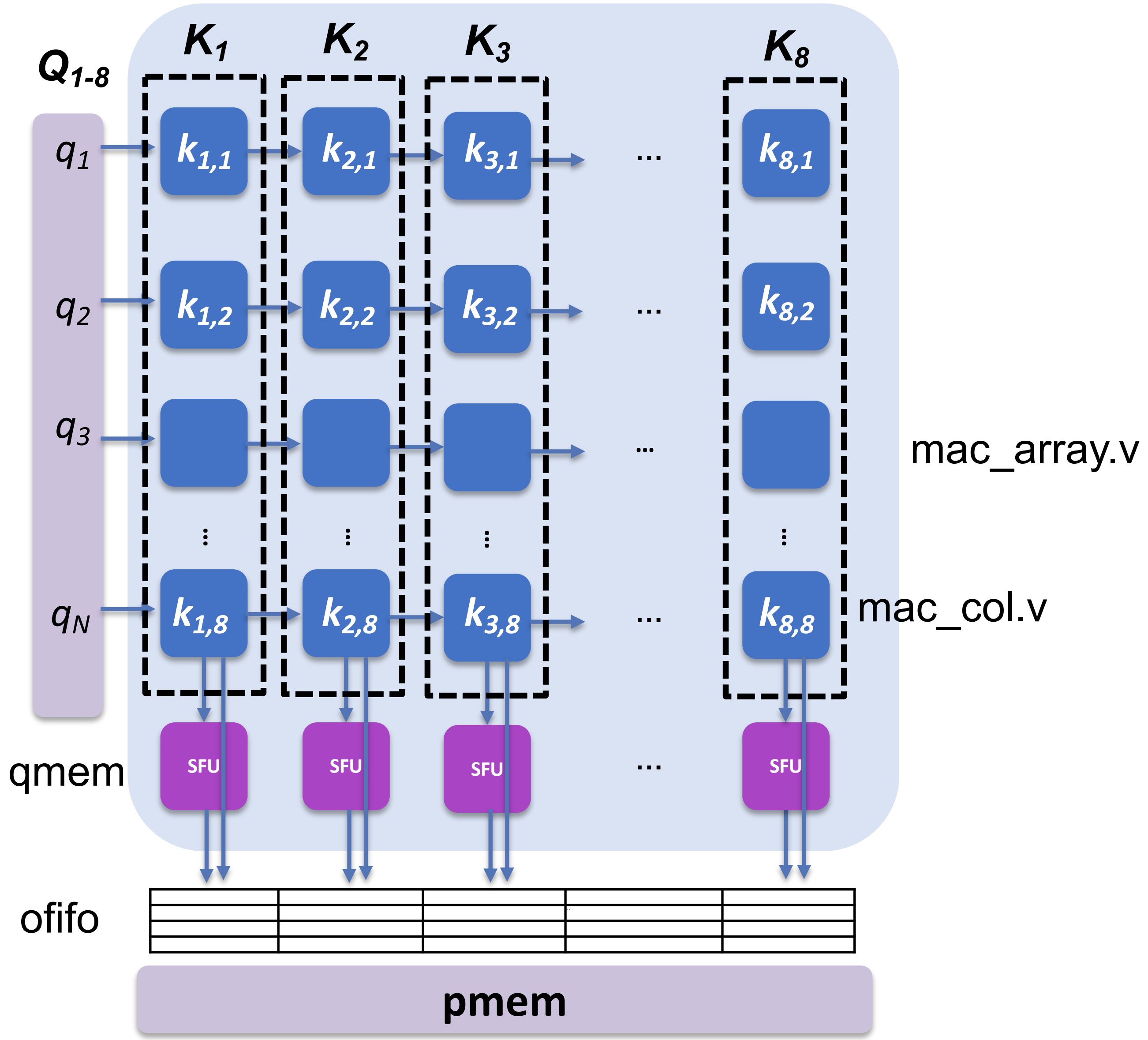
- Stage2: k-data loading stage

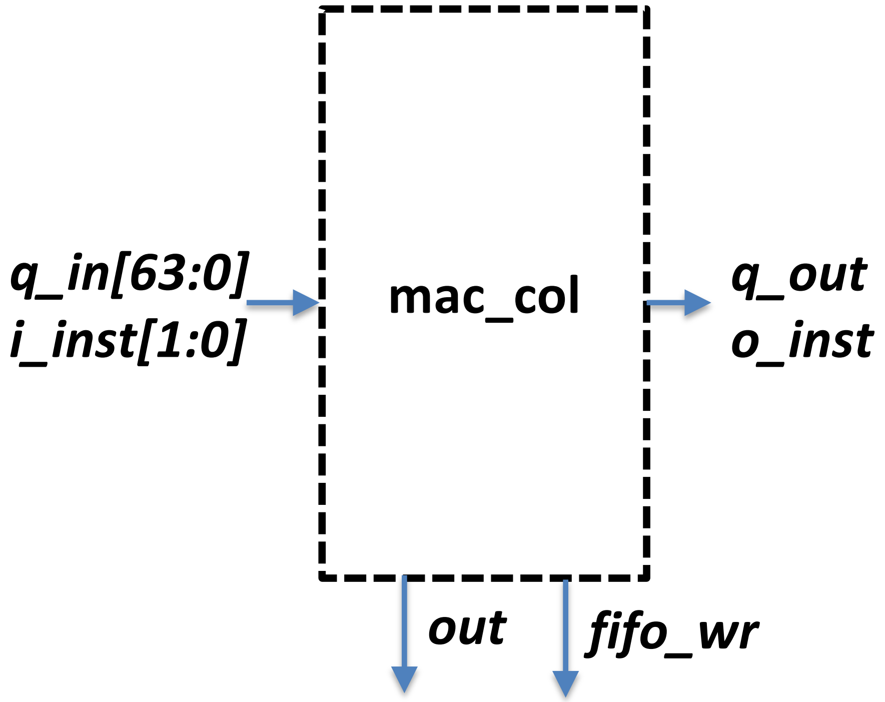
  - move the data from kmem to process’s local register

- Stage3: execute

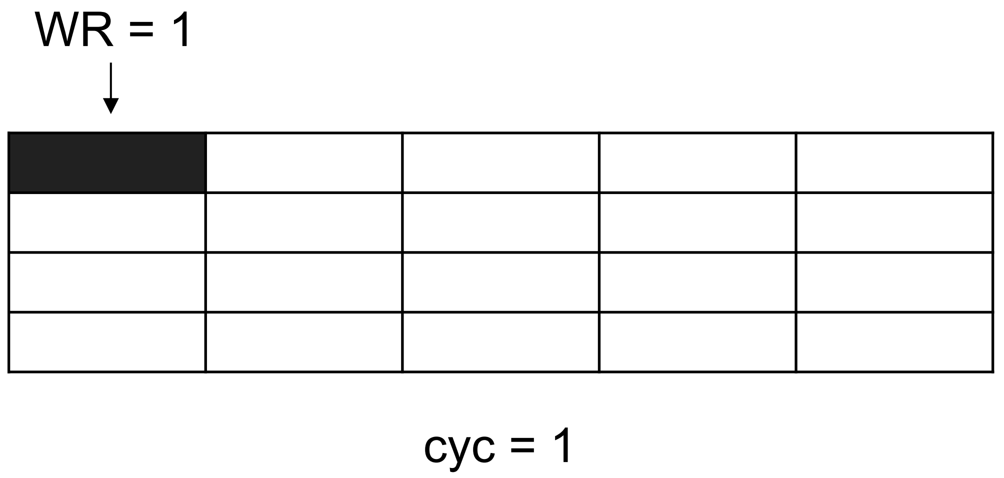
  - move the data from qmem to input of the processor

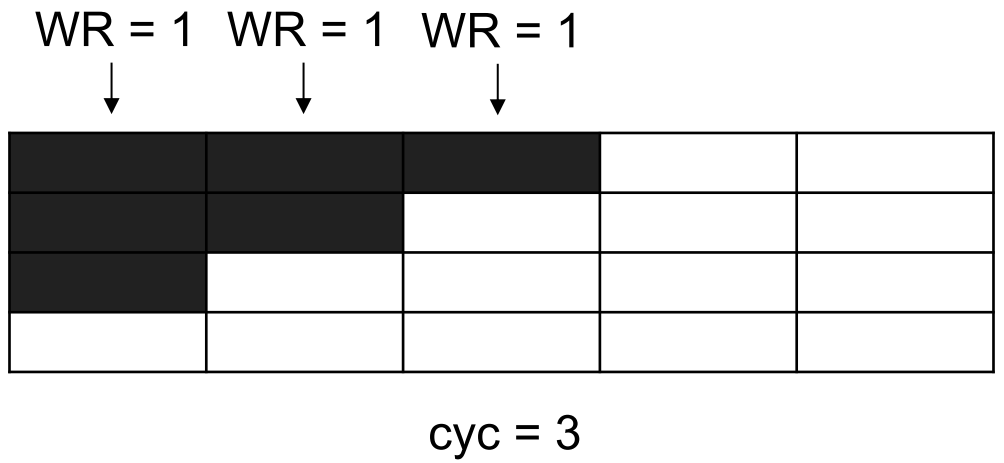
More Detail

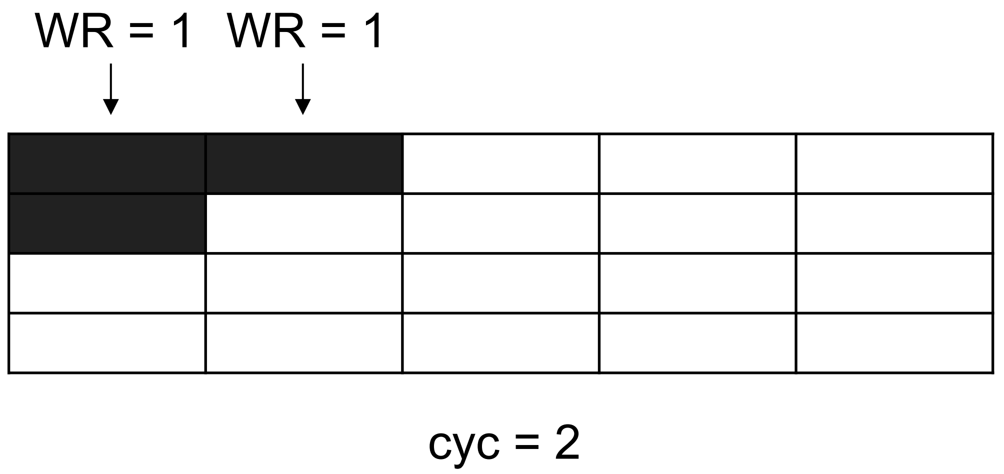

K_1
K_2
K_3
K_8
Q_{1-8}
q_1
q_2
q_3
q_N
$k_{1,1}$
$k_{2,1}$
$k_{3,1}$
$k_{8,1}$
$k_{1,2}$
$k_{2,2}$
$k_{3,2}$
$k_{8,2}$
$k_{1,8}$
$k_{2,8}$
$k_{3,8}$
$k_{8,8}$
mac_array.v
mac_col.v
qmem
SFU
SFU
SFU
SFU
ofifo
pmem

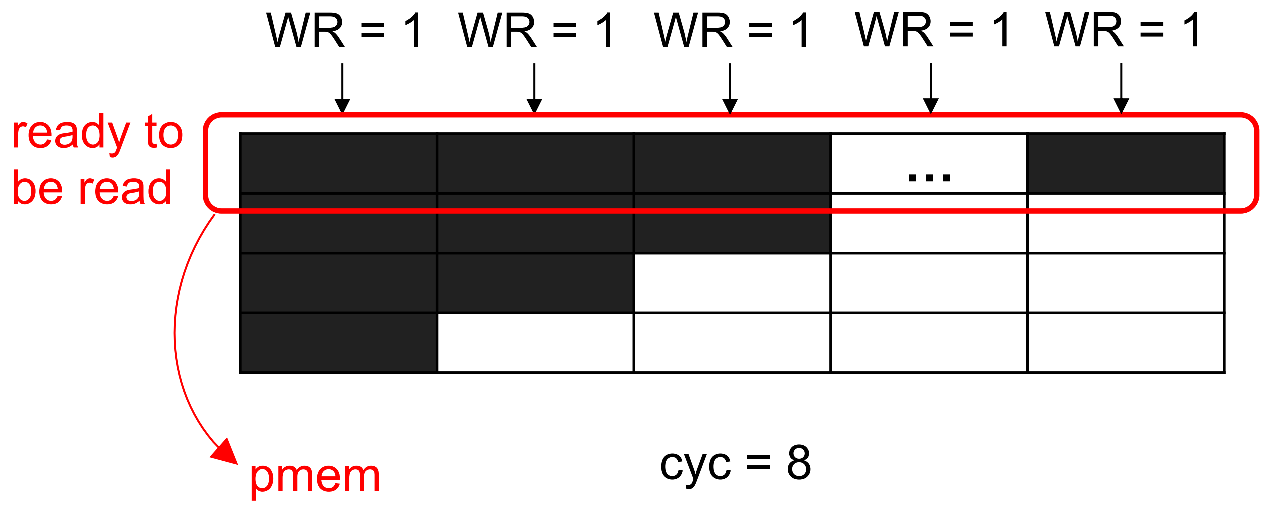

q_in[63:0]
i_inst[1:0]
mac_col
q_out
o_inst
out
fifo_wr

OFIFO Operation

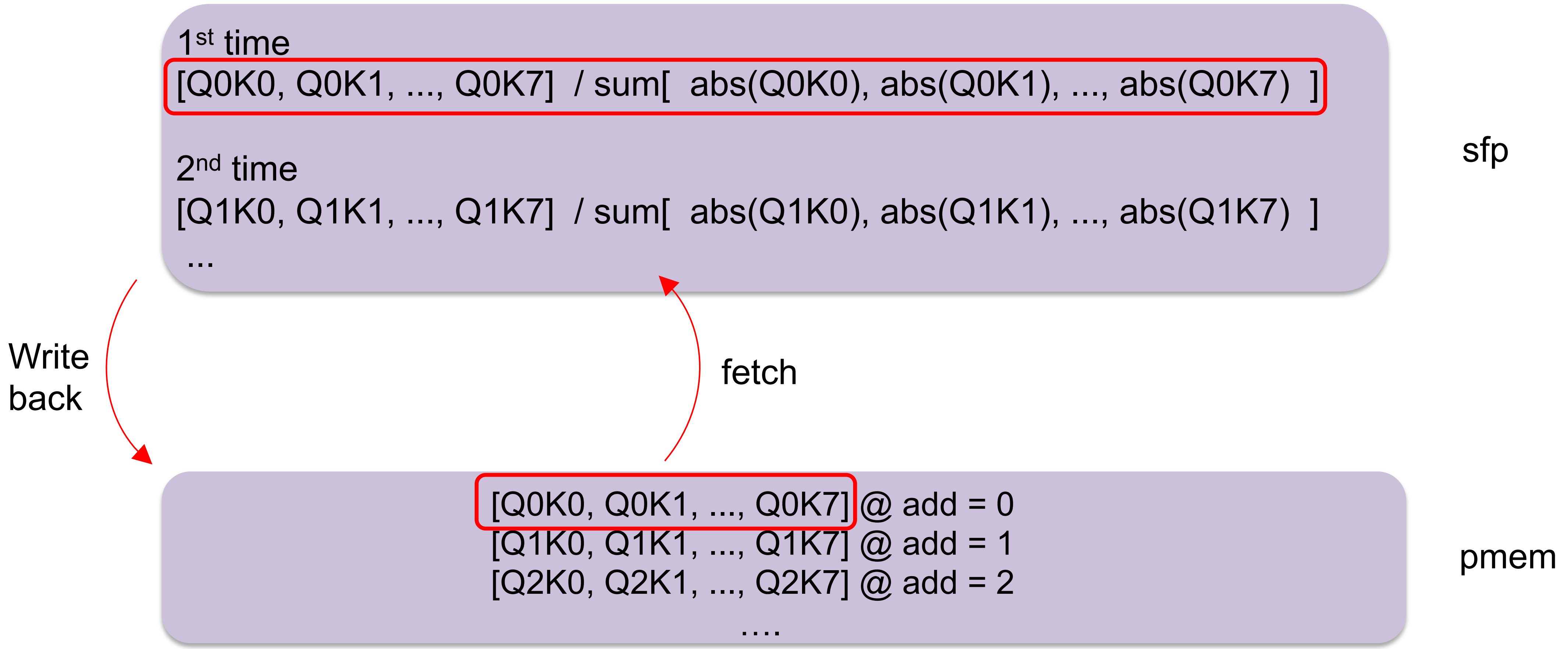








Normalization

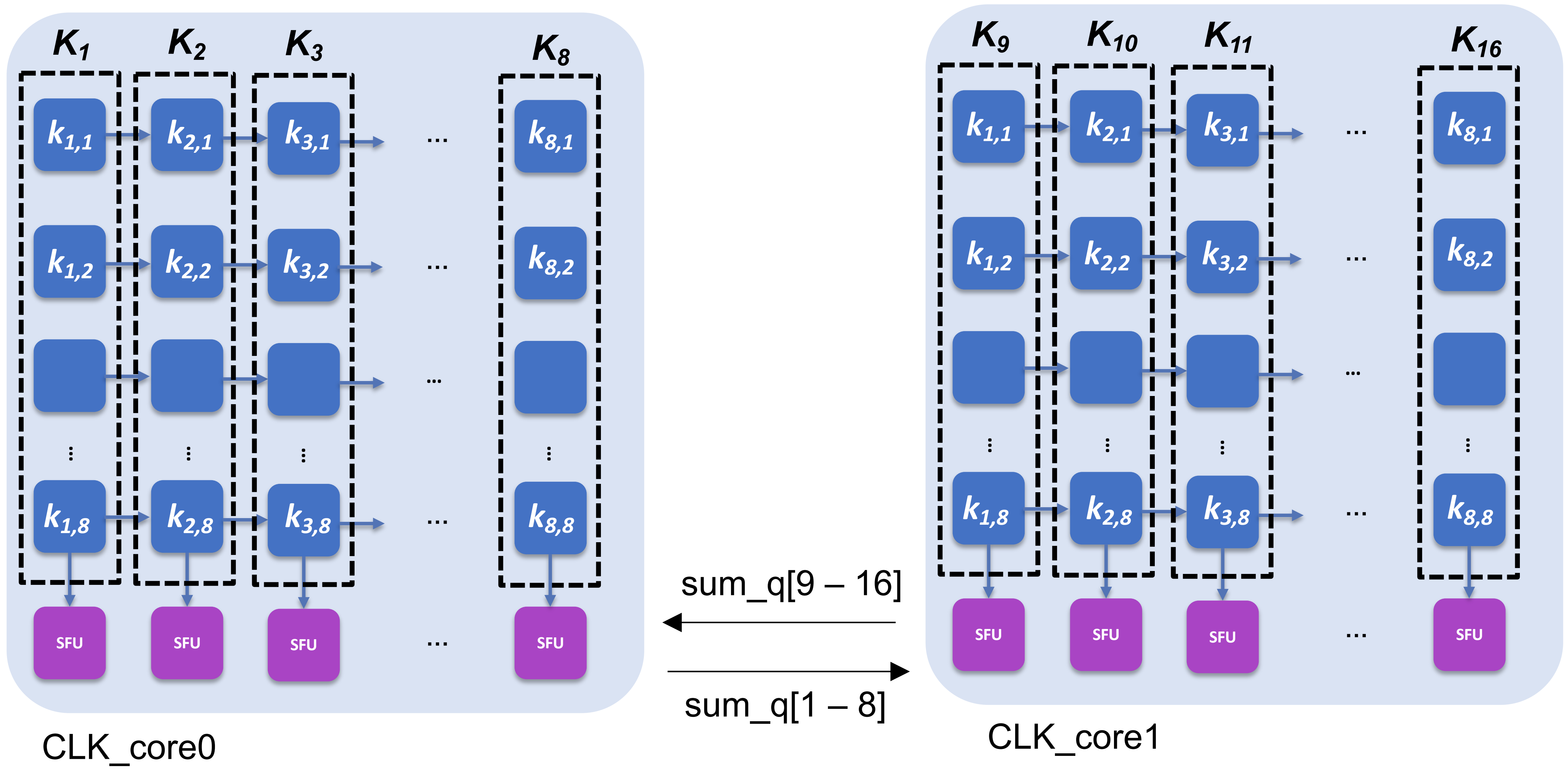


1st time
[Q0K0, Q0K1, ..., Q0K7] / sum[abs(Q0K0), abs(Q0K1), ..., abs(Q0K7)]

2nd time
[Q1K0, Q1K1, ..., Q1K7] / sum[abs(Q1K0), abs(Q1K1), ..., abs(Q1K7)]
...

[Q0K0, Q0K1, ..., Q0K7] @ add = 0
[Q1K0, Q1K1, ..., Q1K7] @ add = 1
[Q2K0, Q2K1, ..., Q2K7] @ add = 2
....

Extension for Dual Core



Normalize by "sum(abs(Q0K0), abs(Q0K1), ..., abs(Q0K7)) from core0
+ sum(abs(Q0K8), abs(Q0K9), ..., abs(Q0K15)) from core1"

Optimization

- Optimize the implementation to maximize the frequency and minimize power
  - However, the project is not a spec competition
- Apply techniques you have learned from this course
- Potential optimization (does not mean you need to do everything below)
  - Pipeline, multi-cycle path, memory double-buffering, clock gating, loop unrolling, more # of cores, any other creative idea
  - Async interface handling (synchronizer, hand shaking, ….)
  - Maximize parallelism between processing stages
  - Design your own controller
  - better verification (with random input)
  - Minimize WNS and DRC errors

How to prove ?

- Prove the efficacy of your idea (why ? how ? how much ?)

  - measure power and check functionality with VCD

    (SDF is not required, but it will be plus alpha)

  - clear comparison before vs. after

  - clear description in the report regarding your design choice

    e.g., why you chose the decision, why you cut the pipeline boundary, ….

  - report is VERY important

Project Policy

- TA & Instructor **will not**
  - solve your project instead of you
  - debug your code (e.g., I got this error with screenshot, can you check my code ?)
  - analyze your issue (e.g., my performance is not so good, do you know why ?)
  - comment on your result (e.g., Can you check my result is correct ?, Can I get A grade with this optimization ?)
  - "fighting against errors" is the biggest part of this project

- However, TA & Instructor will
  - clarify the project problem
  - help any environment issue (e.g., it seems there is no license file, tool does not load, …)
  - provide general guideline for a certain issue

- You can still discuss between students through discussion tab