HPCS
High Performance Computer System Lab

Introduction to Cryogenic Computing Architecture Modeling

Jangwoo Kim (and many researchers @ SNU, Kyushu Uni., and Nagoya Uni.)

E-mail: jangwoo@snu.ac.kr ,
Web: https://hpcs.snu.ac.kr/~jangwoo

High Performance Computer System (HPCS) Lab
Department of Electrical and Computer Engineering
Seoul National University

Index

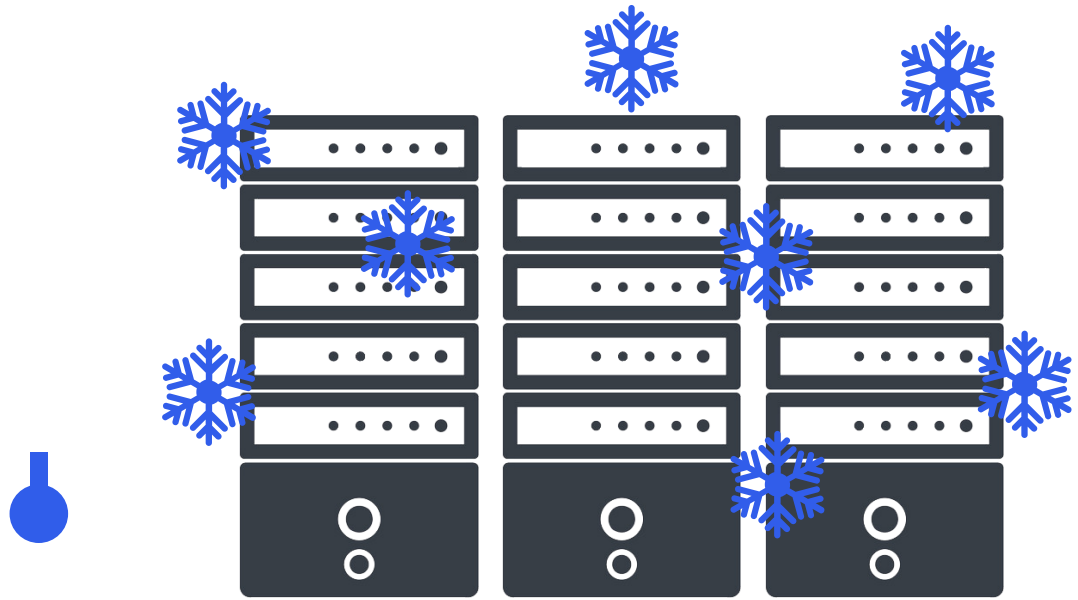
- **Introduction**
- Session #1: 77K CMOS-device computer arch. modeling
- Session #2: 4K SFQ-device computer arch. modeling
- Session #3: 4K+77K quantum computer arch. modeling

Why cryogenic computing?

Conventional Computing

Suffer from the performance wall and power wall problems

Why cryogenic computing?



Cryogenic Computing

Resolve the performance wall and power wall problems

Computing at extremely low temperature is the solution!

Why cryogenic computing?

Two directions to utilize low temperatures
(1) 4K-77K: CMOS-based computer architecture
(2) 4K: SFQ-based computer architecture

Resolve the performance wall and power wall problems

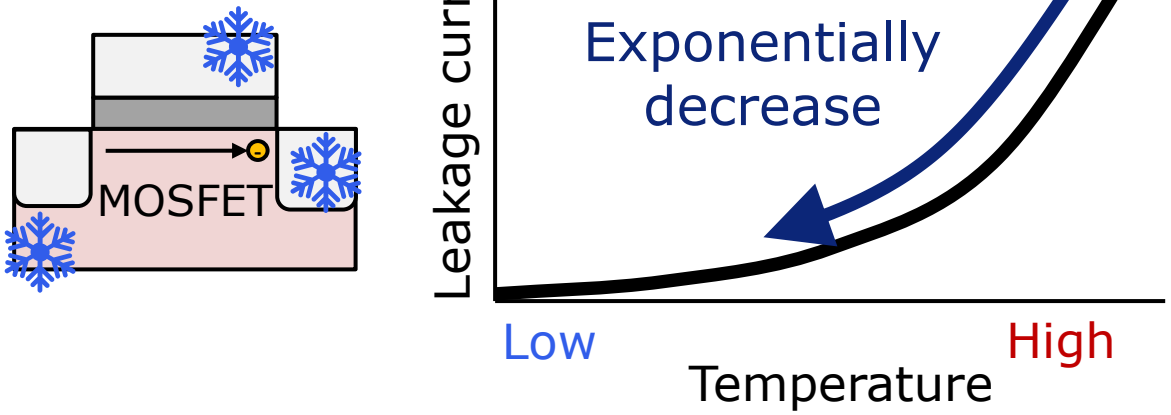
Computing at extremely low temperature is the solution!

What can we do at cryogenic temp.?

#1. Cryogenic CMOS computing @ ~77K

CryoMOSFET

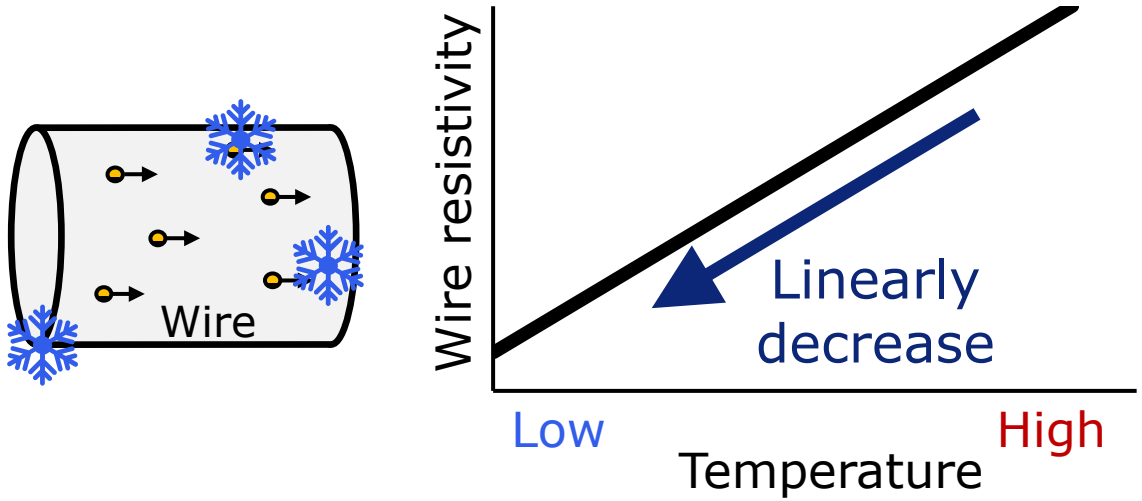
Low leakage current



➡ **Reduce device power**

CryoWire

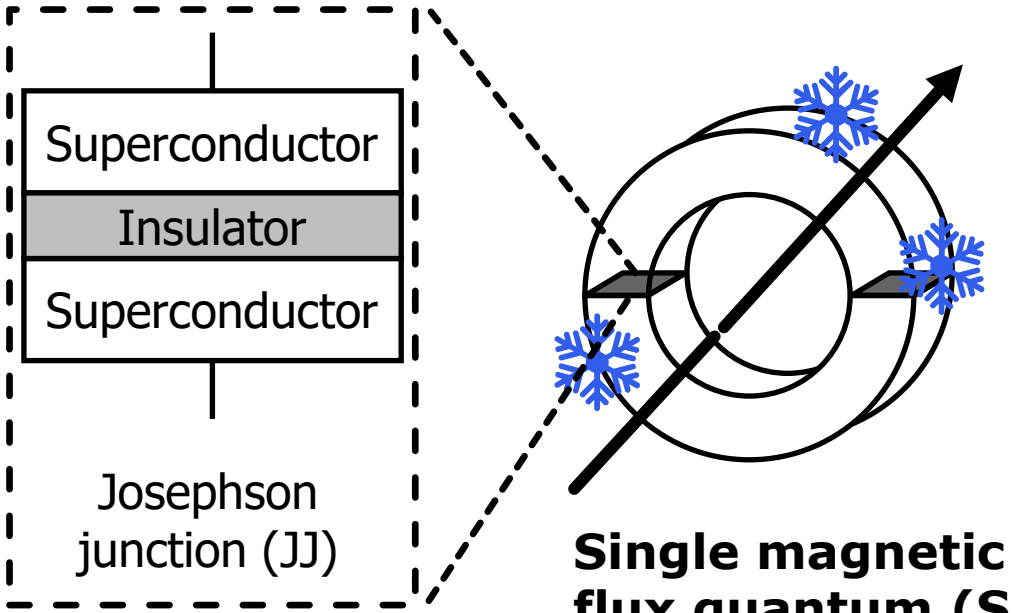
Low wire resistivity



➡ **Reduce wire latency**

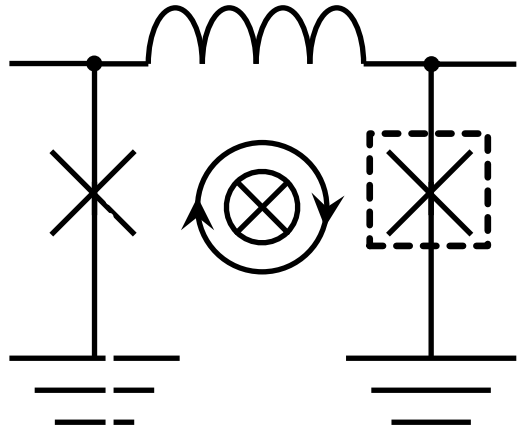
What can we do at cryogenic temp.?

#2. Superconductor computing (e.g., SFQ) @ ~4K



Single magnetic flux quantum (SFQ)

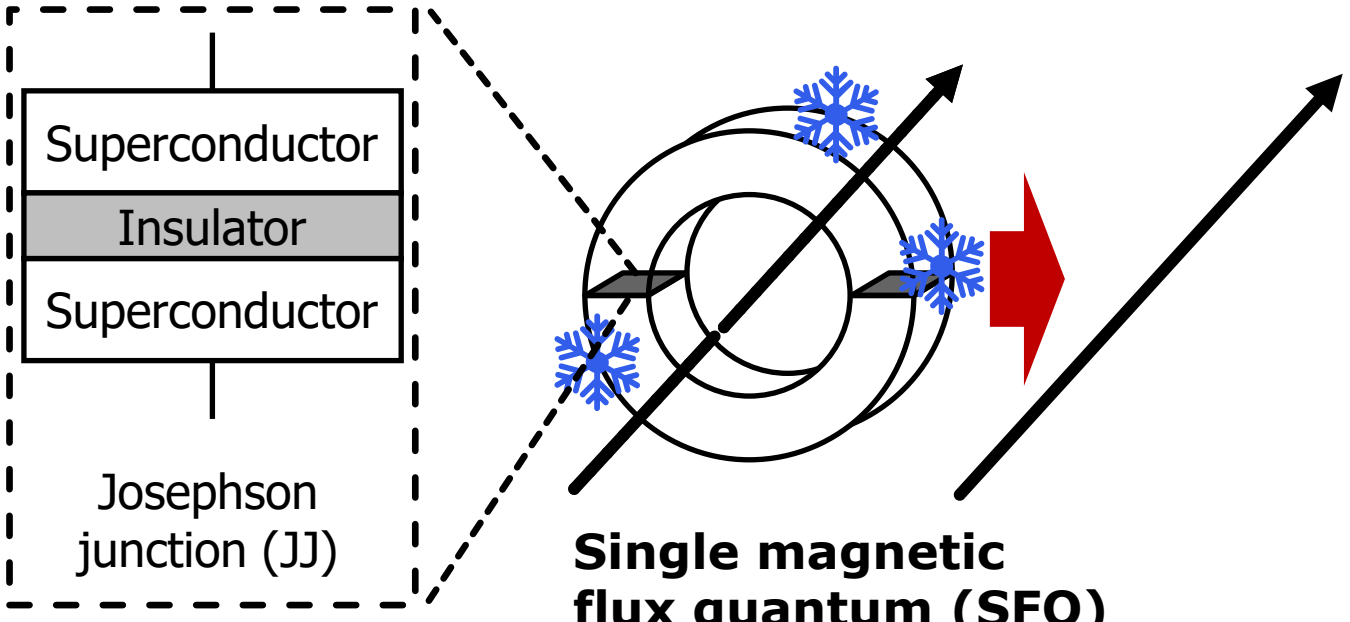
$\Phi_0 = 2.07 \times 10^{-15}\mathrm{Wb}$



SFQ device with superconductor ring

What can we do at cryogenic temp.?

#2. Superconductor computing (e.g., SFQ) @ ~4K

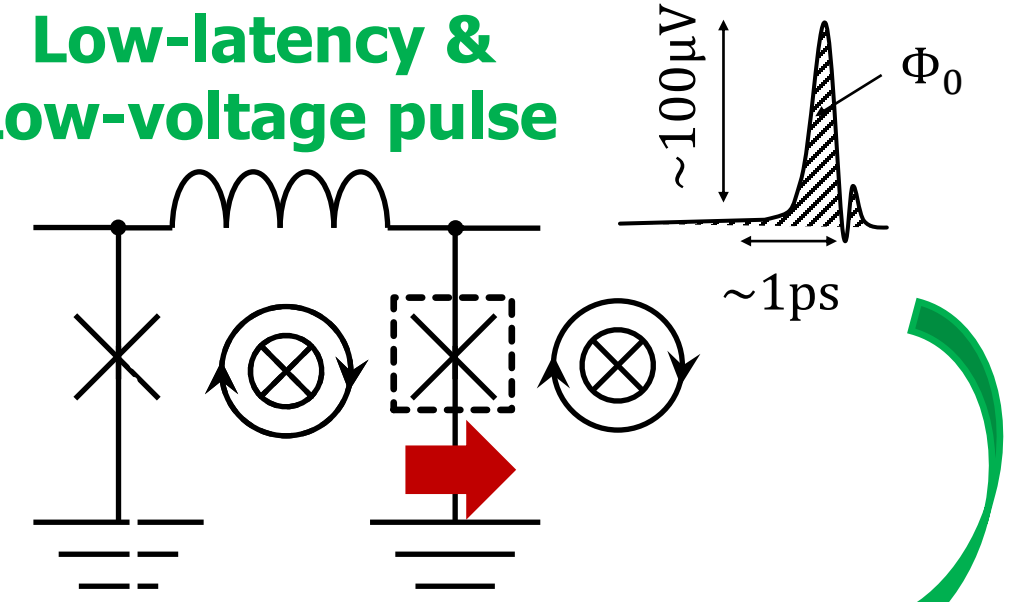


Single magnetic flux quantum (SFQ)

$\Phi_0 = 2.07 \times 10^{-15}\mathrm{Wb}$

SFQ device with superconductor ring

Low-latency & Low-voltage pulse

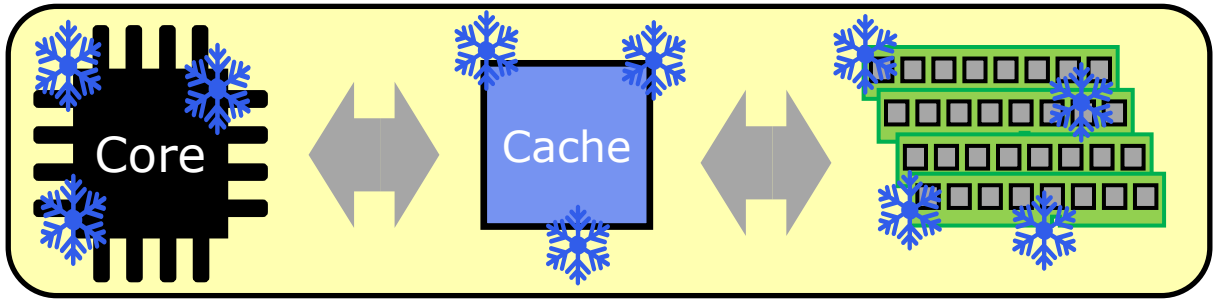


Ultra-fast operation (~100 GHz)

Low-energy switching (~0.1 aJ per switching)

Our research highlights

CMOS-based cryogenic computer

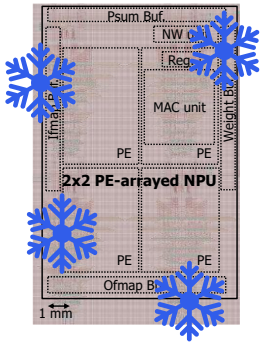


Core
[ISCA'20, ASPLOS'22 Top Picks'21]

Cache
[ASPLOS'20]

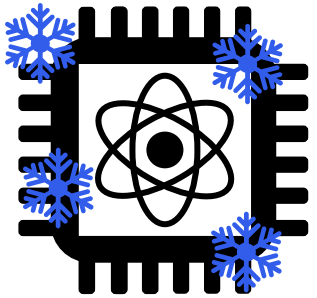
DRAM
[ISCA'19, ISCA'21]

Superconductor computer



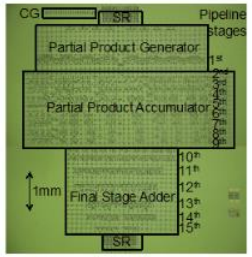
NPU
[MICRO'20, Top Picks'21]

Quantum computer

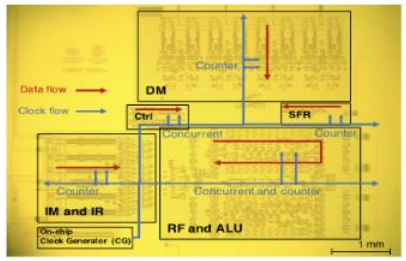


Quantum controller
[ISCA'22]

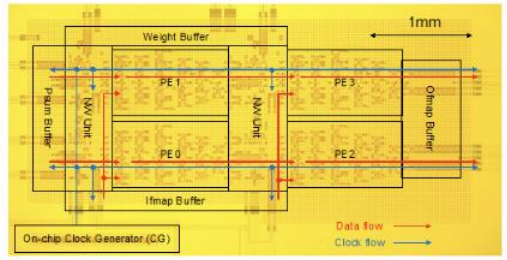
56GHz 1.6mW
ALU
[ISLPED'17]



48GHz 5.6mW
Multiplier
[ISSCC'19]



32GHz 6.2mW
Processor
[VLSI'20]



50GHz
AI Accelerator
[MICRO'20]

Our research highlights

CMOS-based cryogenic computer

Superconductor computer

Quantum computer

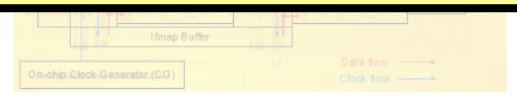
Please refer to the papers for architecture details.

Today's tutorial focus on
"our cryogenic arch. modeling methodologies"

56GHz 1.6mW
ALU
[ISLPED'17]

48GHz 5.6mW
Multiplier
[ISSCC'19]

32GHz 6.2mW
Processor
[VLSI'20]



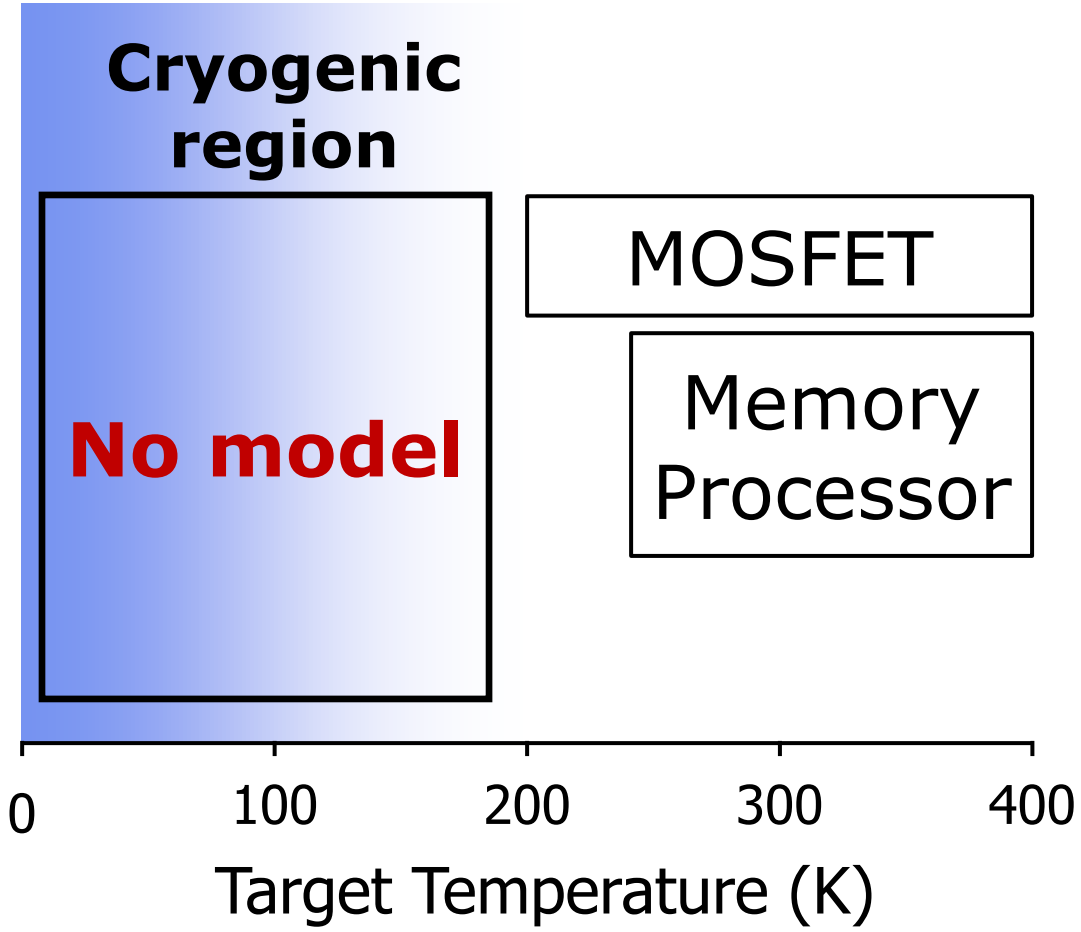
50GHz
AI Accelerator
[MICRO'20]

Index

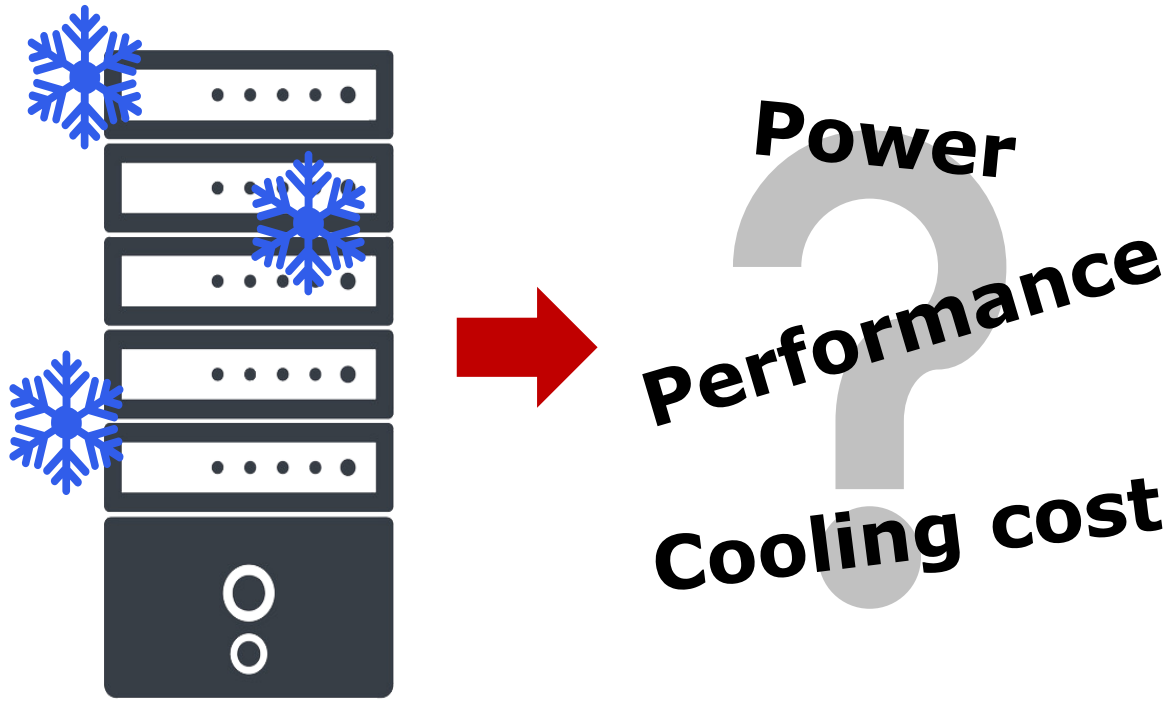
- Introduction
- **Session #1: 77K CMOS-device computer arch. modeling**
- Session #2: 4K SFQ-device computer arch. modeling
- Session #3: 4K+77K quantum computer arch. modeling

Challenges of CryoCMOS computing (1/2)

#1: No modeling tool



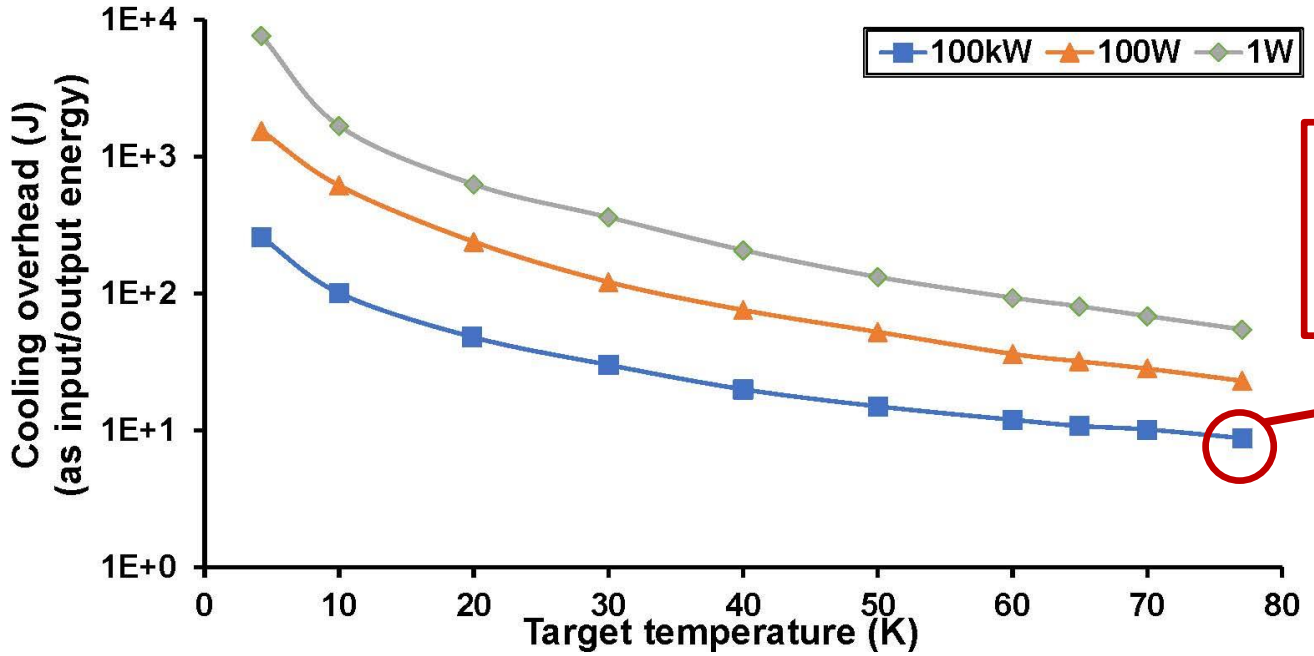
#2: No cryo-optimal designs



Challenges of CryoCMOS computing (2/2)

#3: Cooling cost can be a killer

- $P_{cooling} = P_{device} \times Cooling.Overhead$
  - $Cooling.Overhead = \frac{T_{orignial} - T_{cooling}}{T_{original} \times Cooler\ efficiency}$



With 100KW cooler (efficiency of 0.3)
77K cooling overhead is
9.65x of device power

[Y. Iwasa. 2009. Case studies in superconducting magnets: design and operational issues. Springer Science & Business Media.]

Research goals

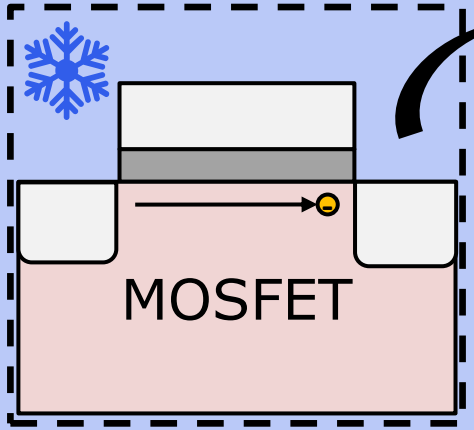
Session #1

Goal 1	Build a CryoCMOS architecture modeling tool (e.g., device, circuit, architecture simulators, …)
Goal 2	Build cryogenic-optimal architectures (e.g., core, cache, memory, server, …)

CryoModel: Overview

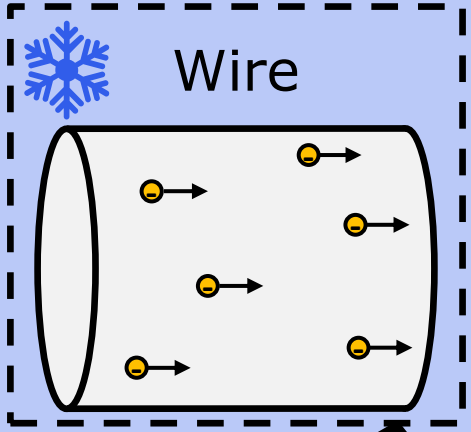
1) MOSFET model

Low-temperature
MOSFET characteristics

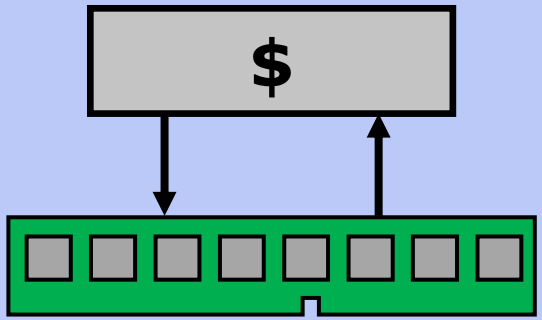


2) Wire model

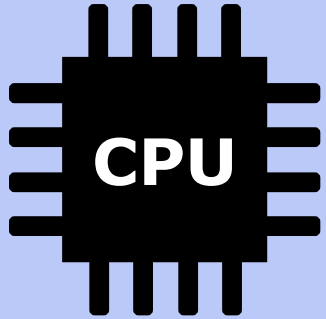
Low-temperature
wire characteristics



3) Memory/Processor model



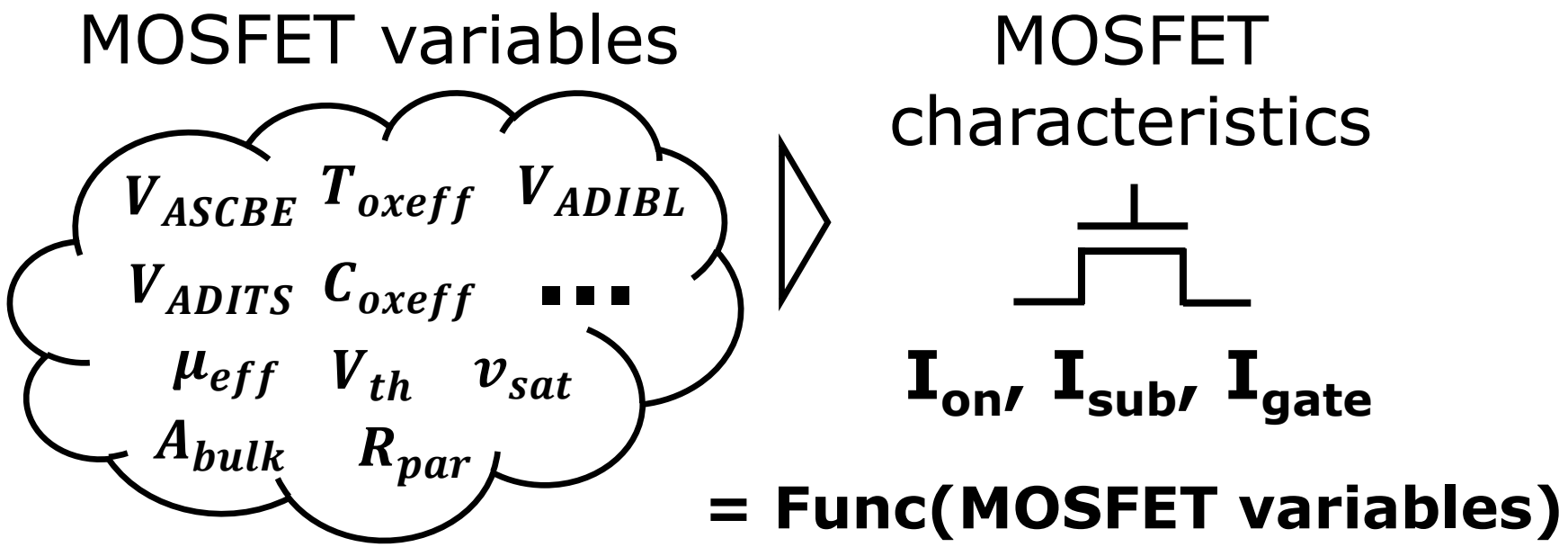
Latency & Power of
cryogenic memory



Critical-path delays of
cryogenic processor

CryoModel: MOSFET model

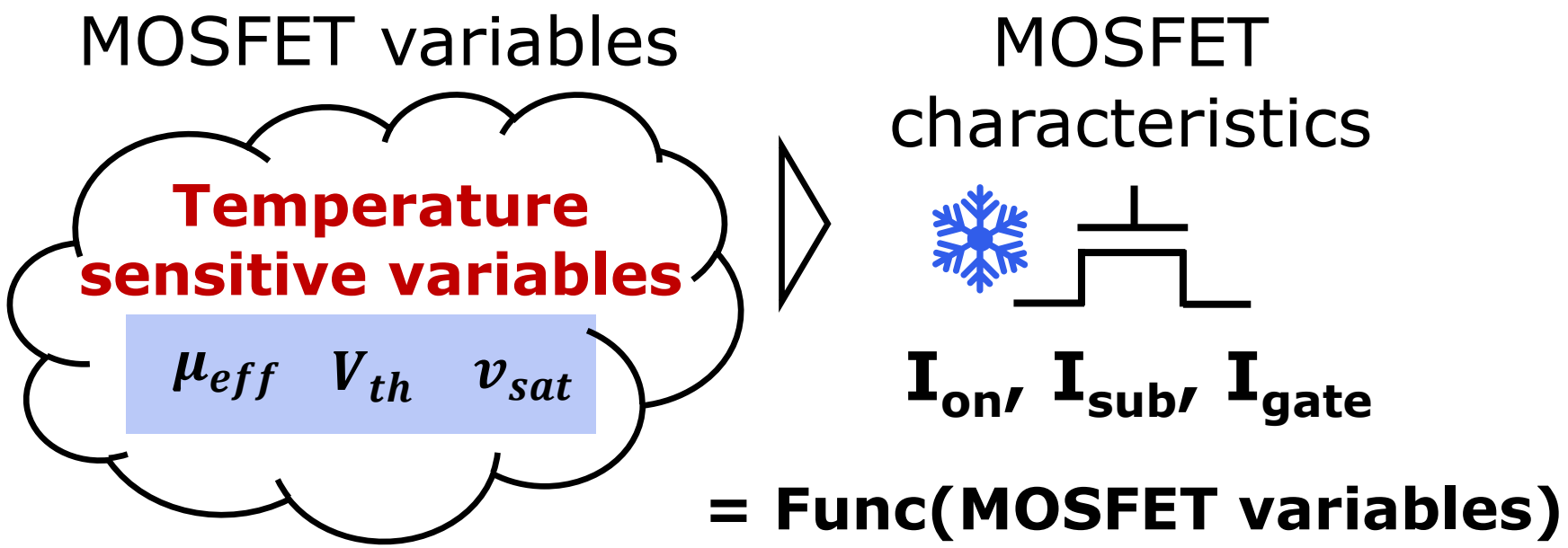
- **MOSFET model predicts low-temperature I_{on}, I_{sub}, I_{gate} by modeling three temperature sensitive variables**



= Func(MOSFET variables)

CryoModel: MOSFET model

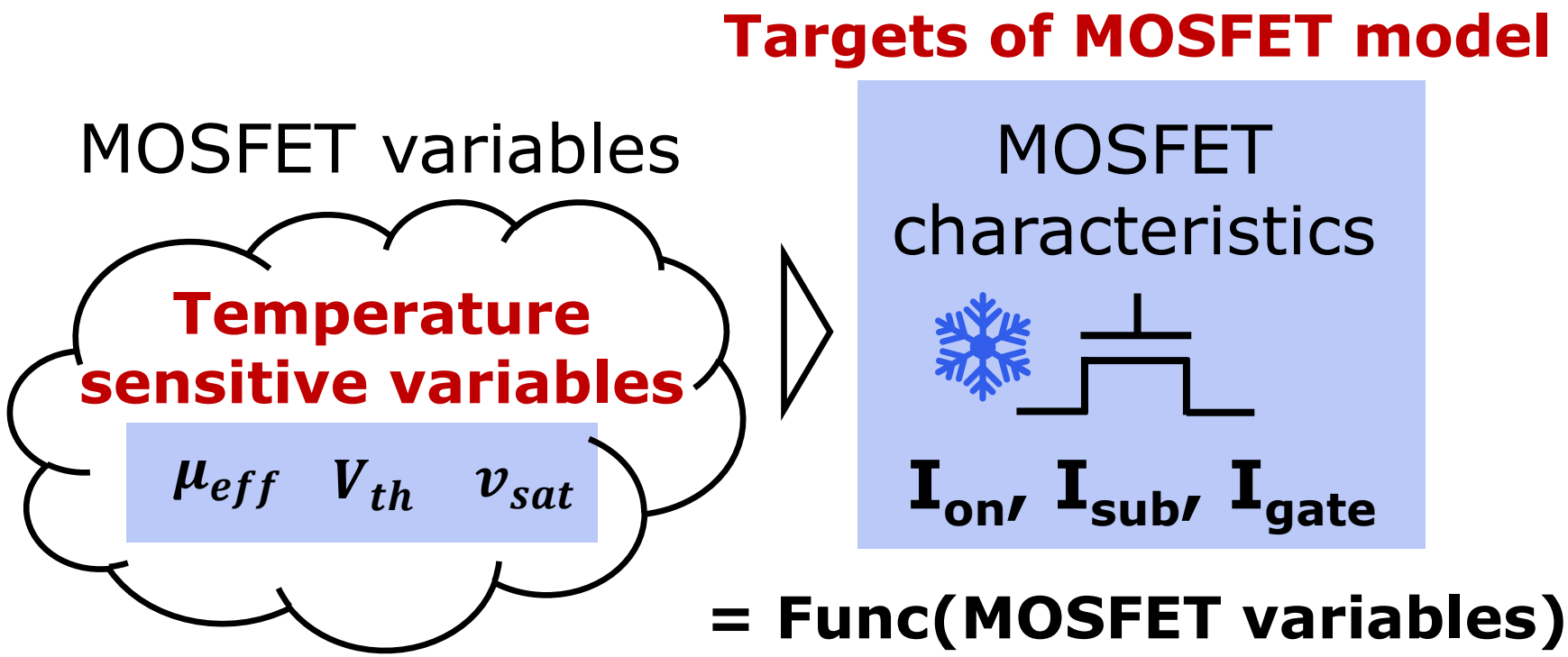
- **MOSFET model predicts low-temperature I_{on}, I_{sub}, I_{gate} by modeling three temperature sensitive variables**



= Func(MOSFET variables)

CryoModel: MOSFET model

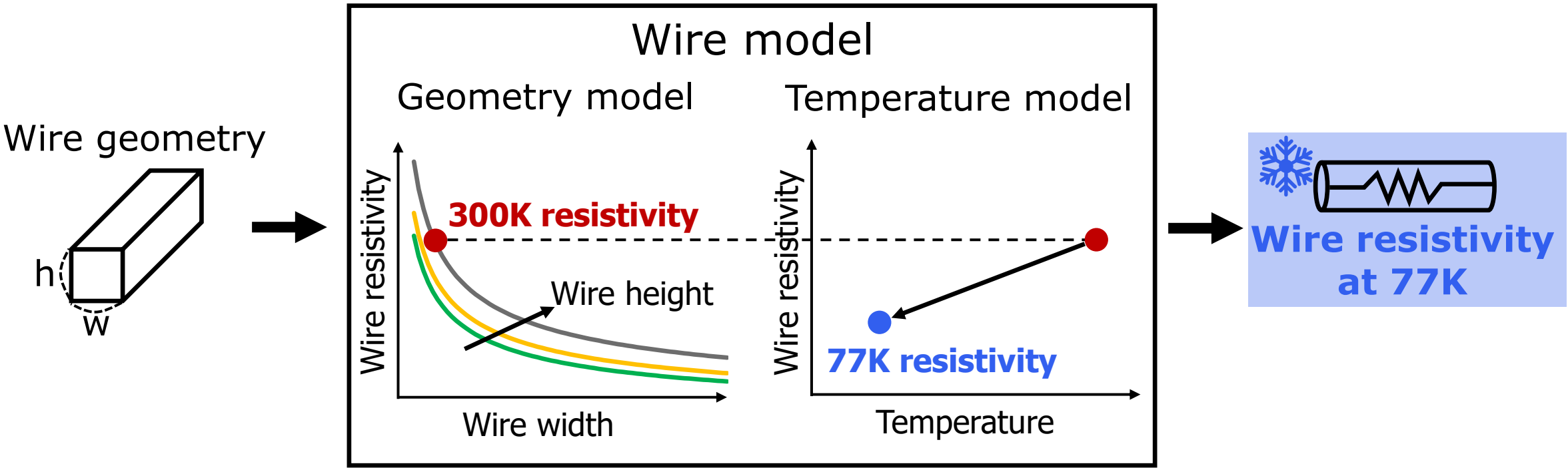
- **MOSFET model predicts low-temperature I_{on}, I_{sub}, I_{gate} by modeling three temperature sensitive variables**



= Func(MOSFET variables)

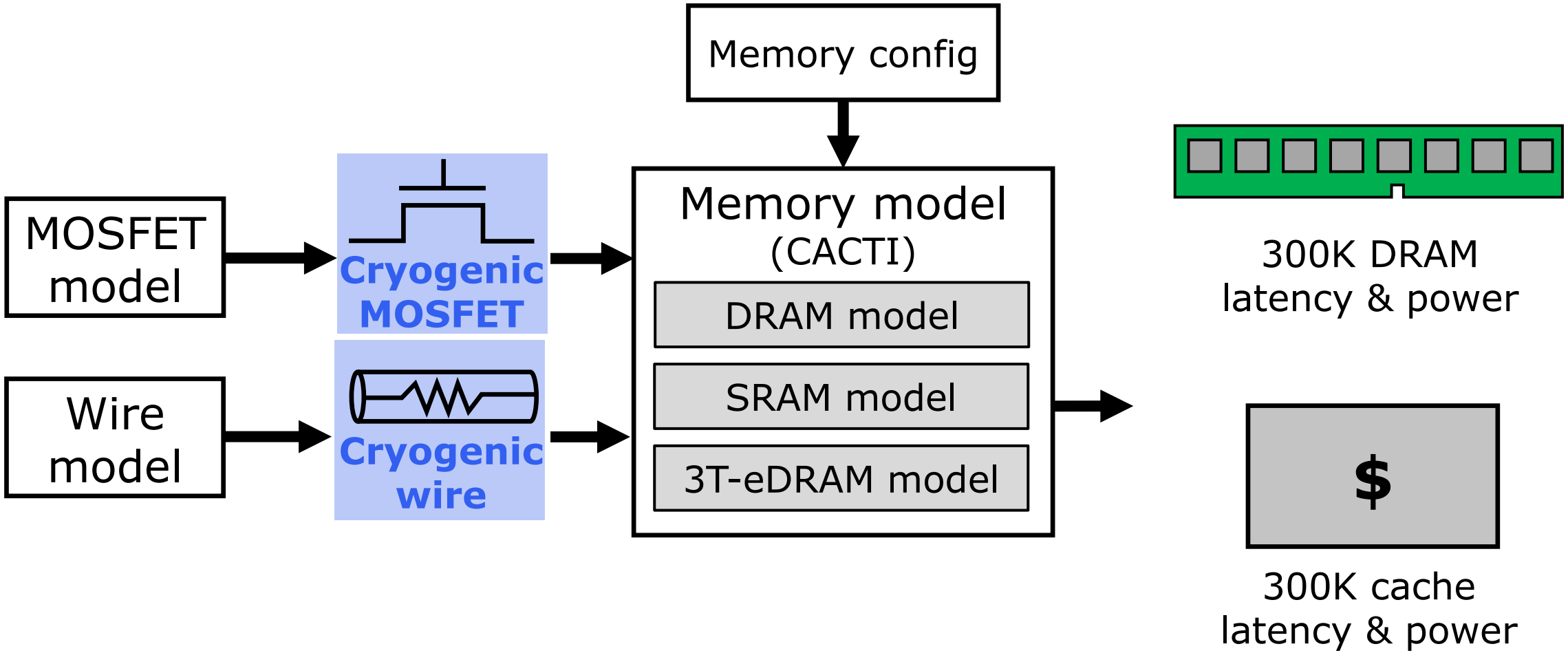
CryoModel: Wire model

- **Wire model predicts low-temperature wire resistivity**
  - Geometry model: derive 300K resistivity based on wire width and height
  - Temperature model: linearly scale the 300K resistivity to 77K resistivity



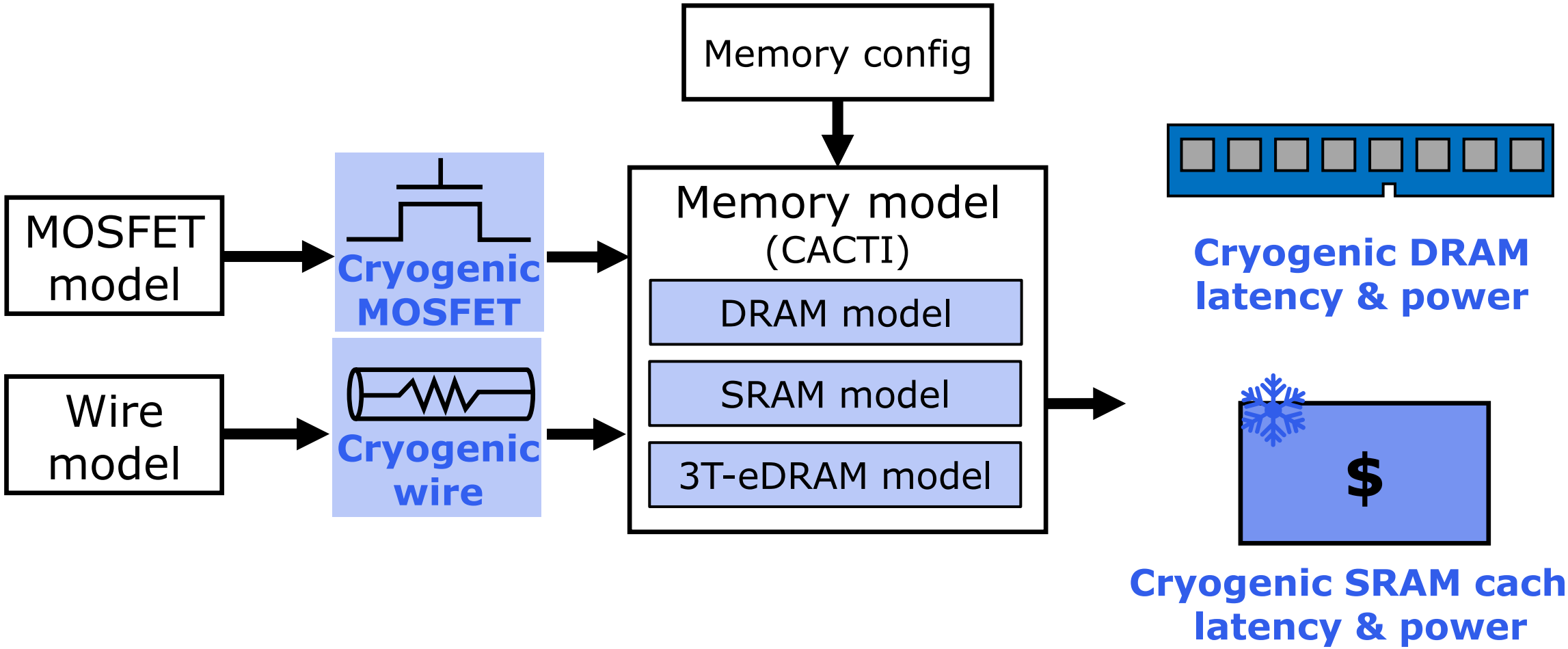
CryoModel: Memory model

- **Can predict latency and power of cryogenic DRAM & cache!**



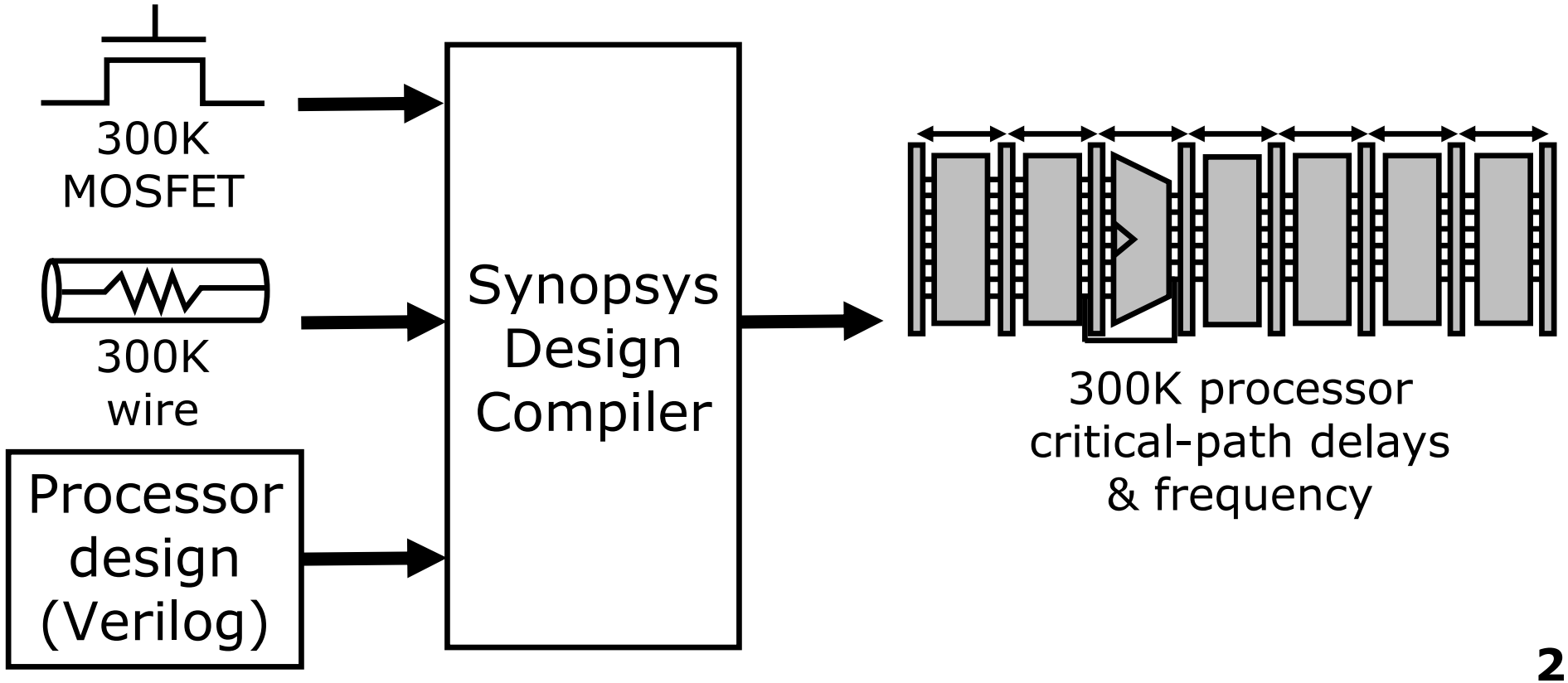
CryoModel: Memory model

- **Can predict latency and power of cryogenic DRAM & cache!**



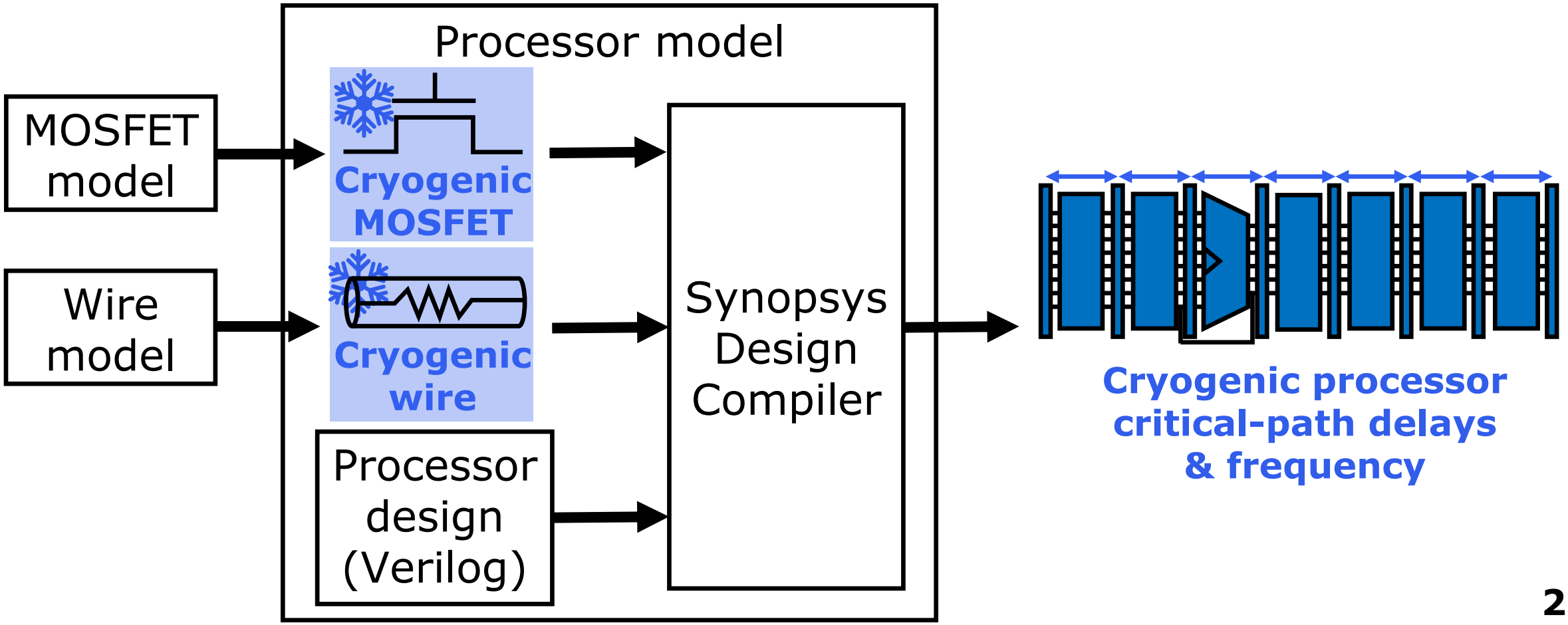
CryoModel: Processor model

- **Can estimate the core's critical-path delays and frequency!**



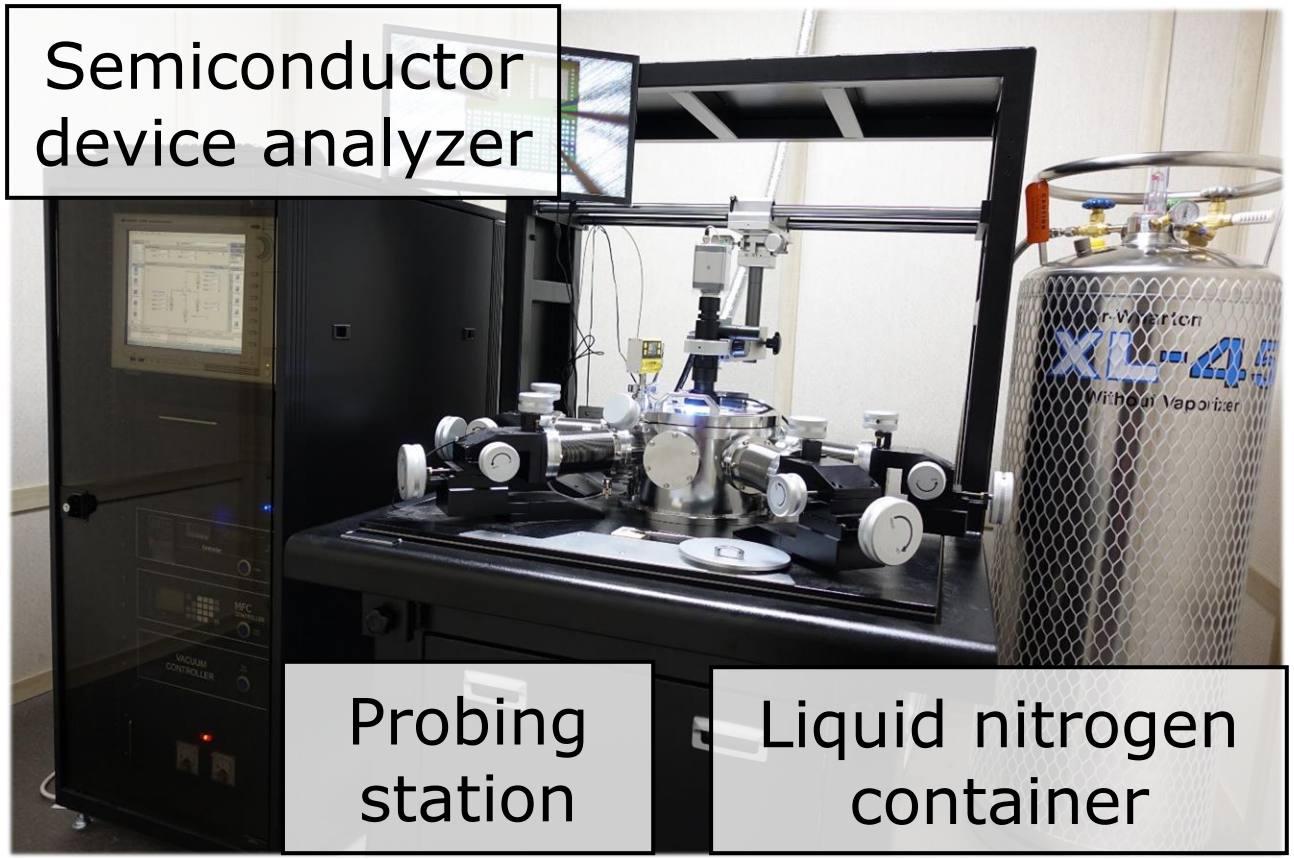
CryoModel: Processor model

- **Can estimate the core's critical-path delays and frequency!**



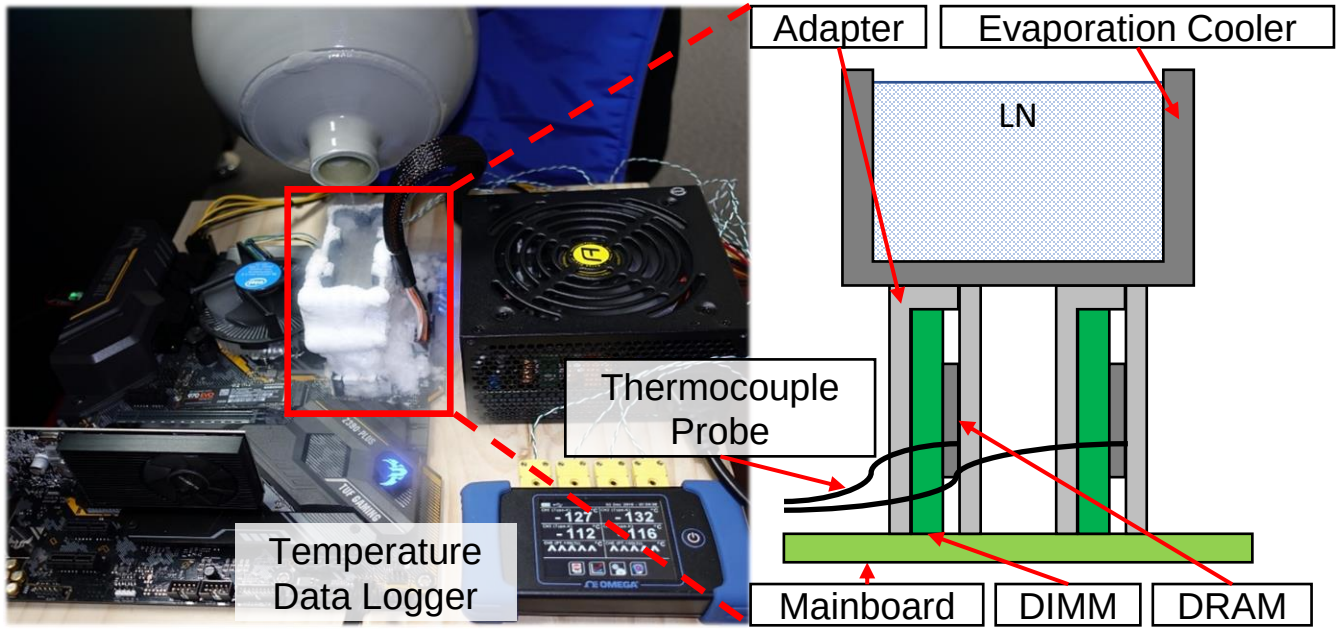
CryoModel: Validation (1/2)

- **In-house probing station**
  - Measure MOSFET characteristics at 77K

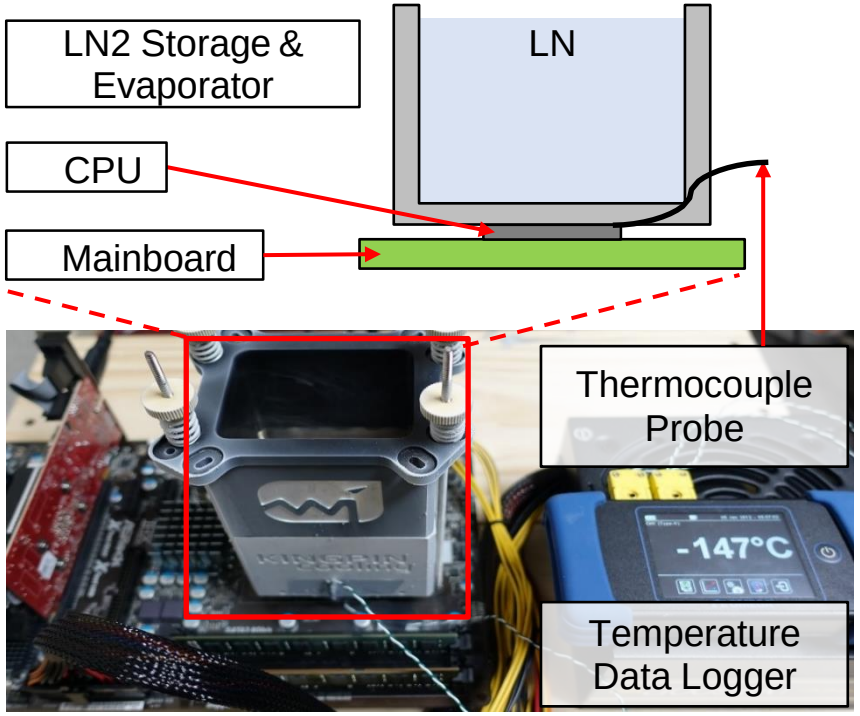


CryoModel: Validation (2/2)

- **Custom-built cooling setup with commodity products**
  - Control and measure the DRAM/processor clock frequency at ~130K



Memory model validation setup

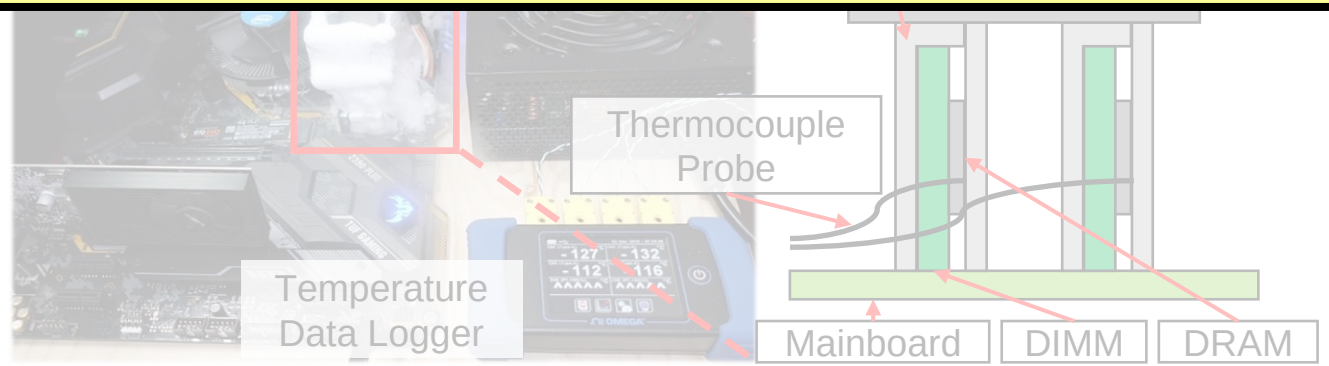


Processor model validation setup

CryoModel: Validation (2/2)

- **Custom-built cooling setup with commodity products**
  - Control and measure the DRAM/processor clock frequency at ~130K

We appreciate Samsung & SK Hynix for their invaluable support!



Memory model validation setup

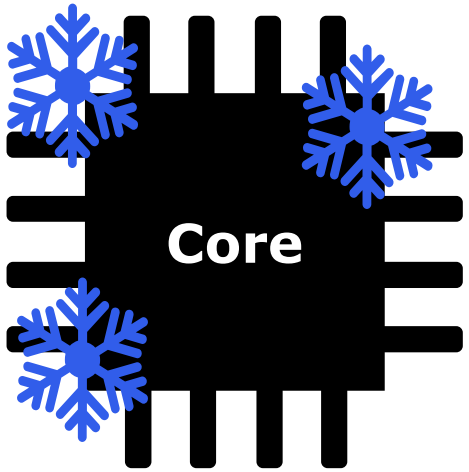
Processor model validation setup

Research goals

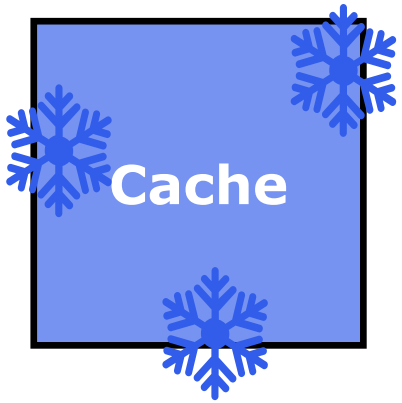
Goal 1	Build a CryoCMOS architecture modeling tool (e.g., device, circuit, architecture simulators, …)
Goal 2	Build cryogenic-optimal architectures (e.g., core, cache, memory, server, …)

Other CryoCMOS architecture research

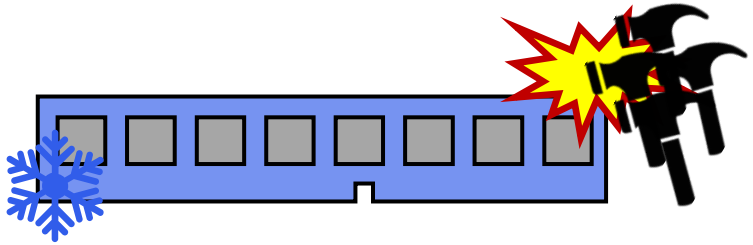
CryoCore/CryoWire
[ISCA'20] / [ASPLOS'22]



CryoCache
[ASPLOS'20]



CryoRAM/Guard
[ISCA'19], [ISCA'21]



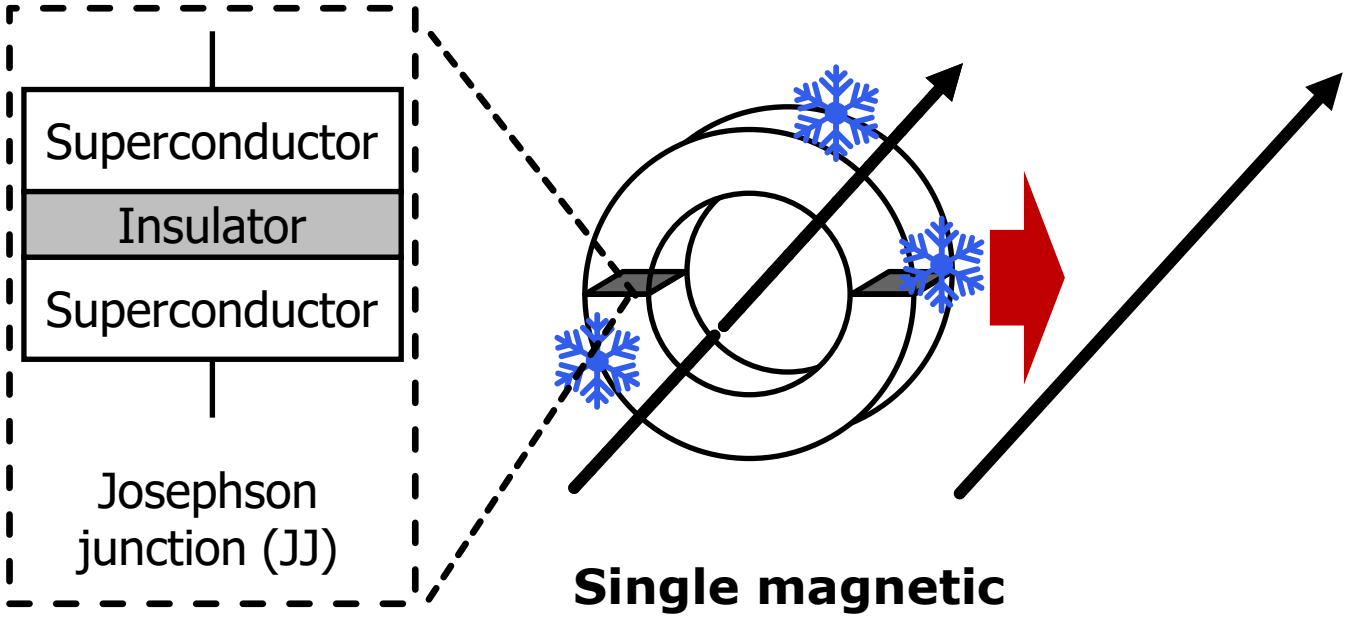
Please refer to the papers for more details!

Index

- Introduction
- Session #1: 77K CMOS-device computer arch. modeling
- **Session #2: 4K SFQ-device computer arch. modeling**
- Session #3: 4K+77K quantum computer arch. modeling

New device for extremely low temp!

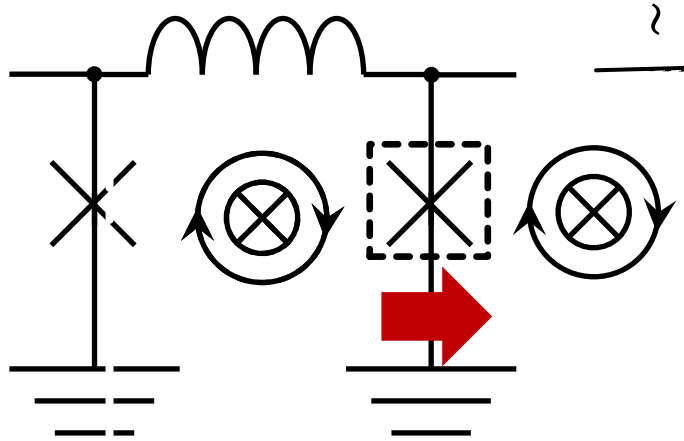
→ Superconductor computing (e.g., SFQ) @ ~4K

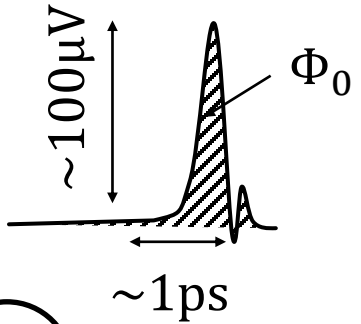


Single magnetic flux quantum (SFQ)

$\Phi_0 = 2.07 \times 10^{-15}\mathrm{Wb}$

Low-latency & Low-voltage pulse

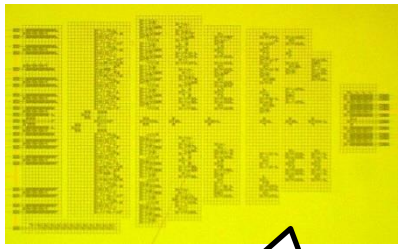




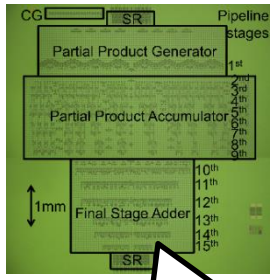
SFQ device with superconductor ring

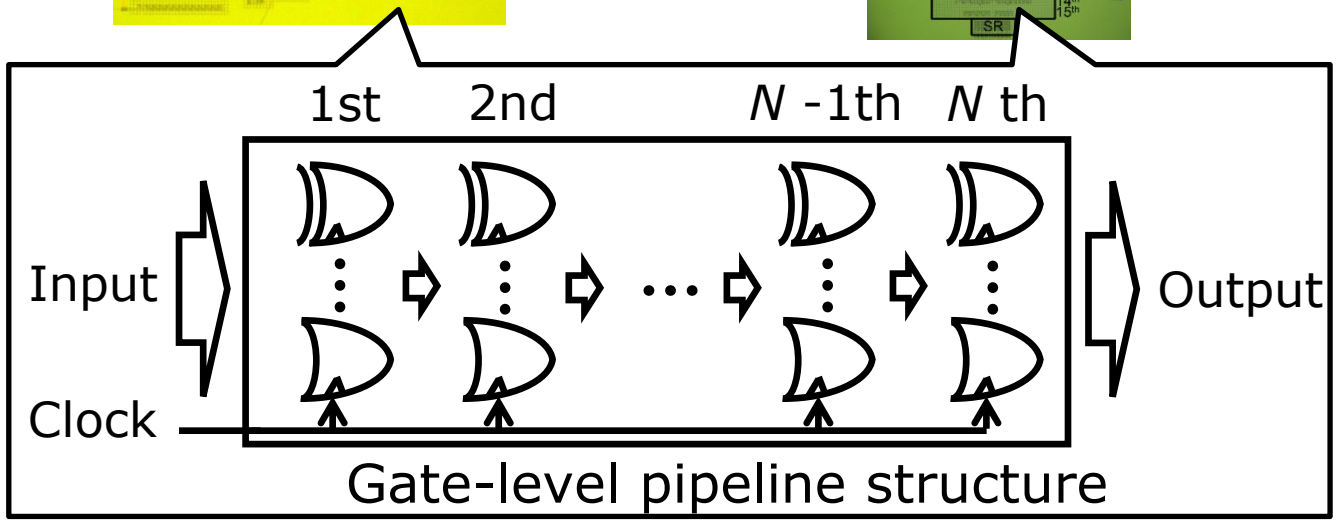
We brought up early circuit designs!

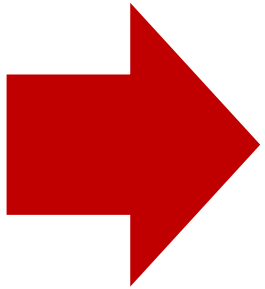
8bit ALU: 56 GHz, 1.6mW [1]

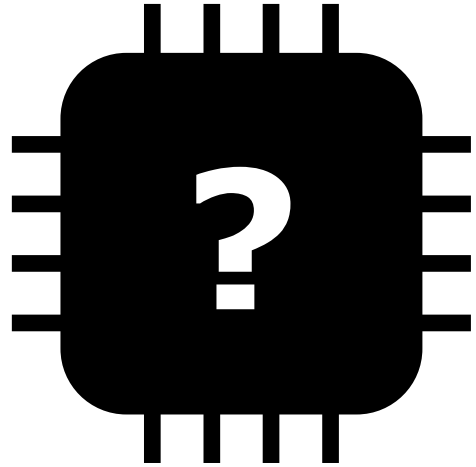


8bit MUL: 48 GHz, 5.6 mW [2]









Recent progress in SFQ circuits

Next: SFQ-based architecture design

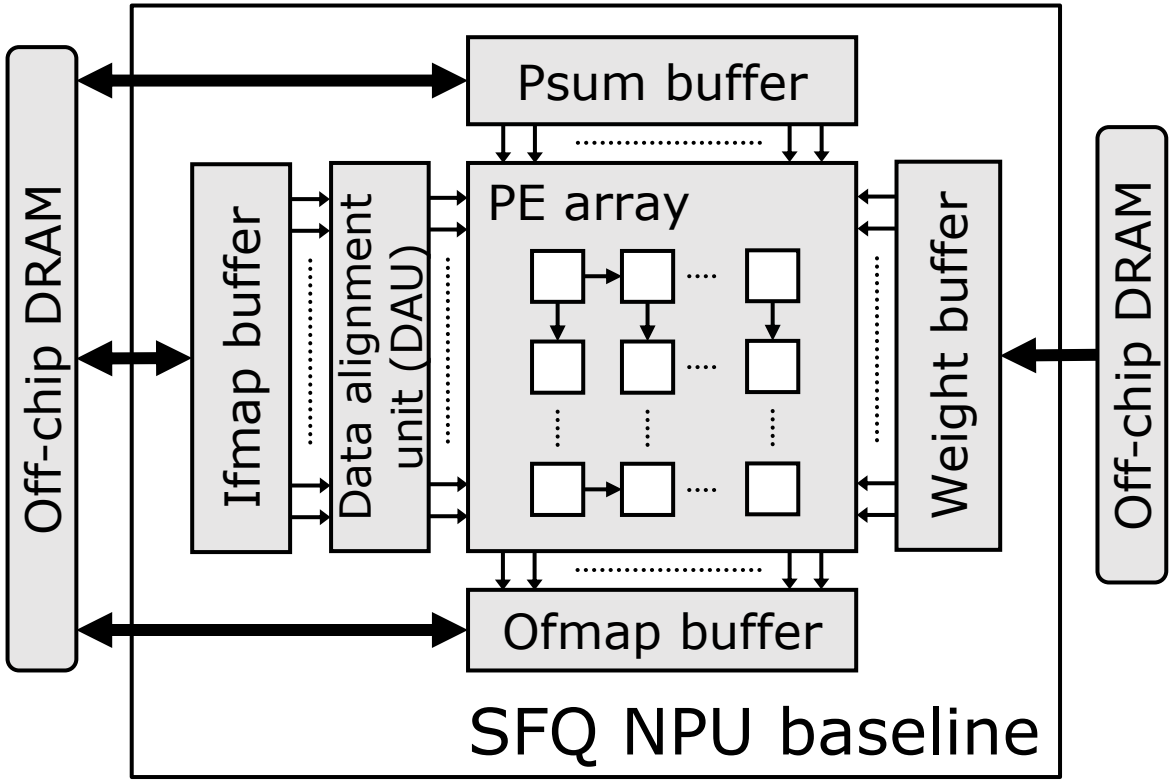
[1] M. Tanaka et al., "High-throughput bit-parallel arithmetic logic unit using rapid single-flux-quantum logic," in Proc. of ISEC, Jun. 2017
[2] I. Nagaoka et al., "A 48 GHz 5.6mW gate-level-pipelined multiplier using single-flux quantum logic," in ISSCC2019, 2019

Research goals

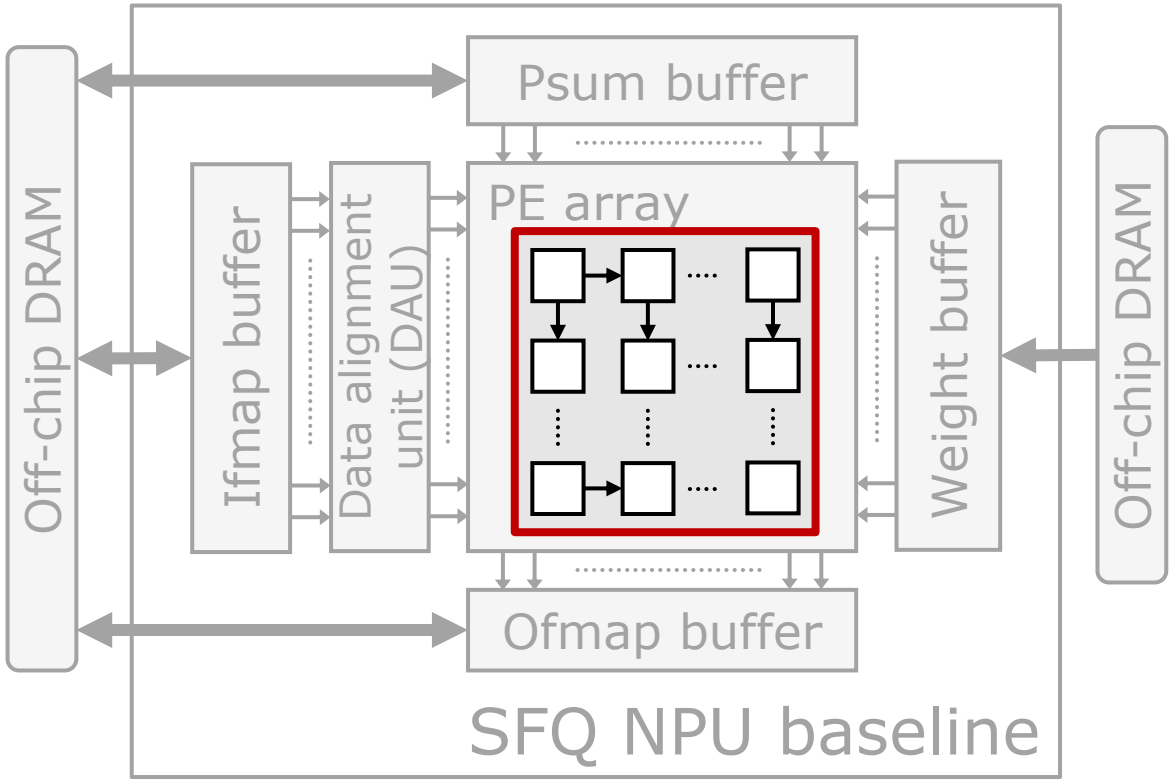
Session #2

Goal 1	Build SFQ modeling/simulation framework
Goal 2	Build SFQ-optimal architectures (e.g., processor, AI accelerator, …)

A good example: SFQ-based NPU

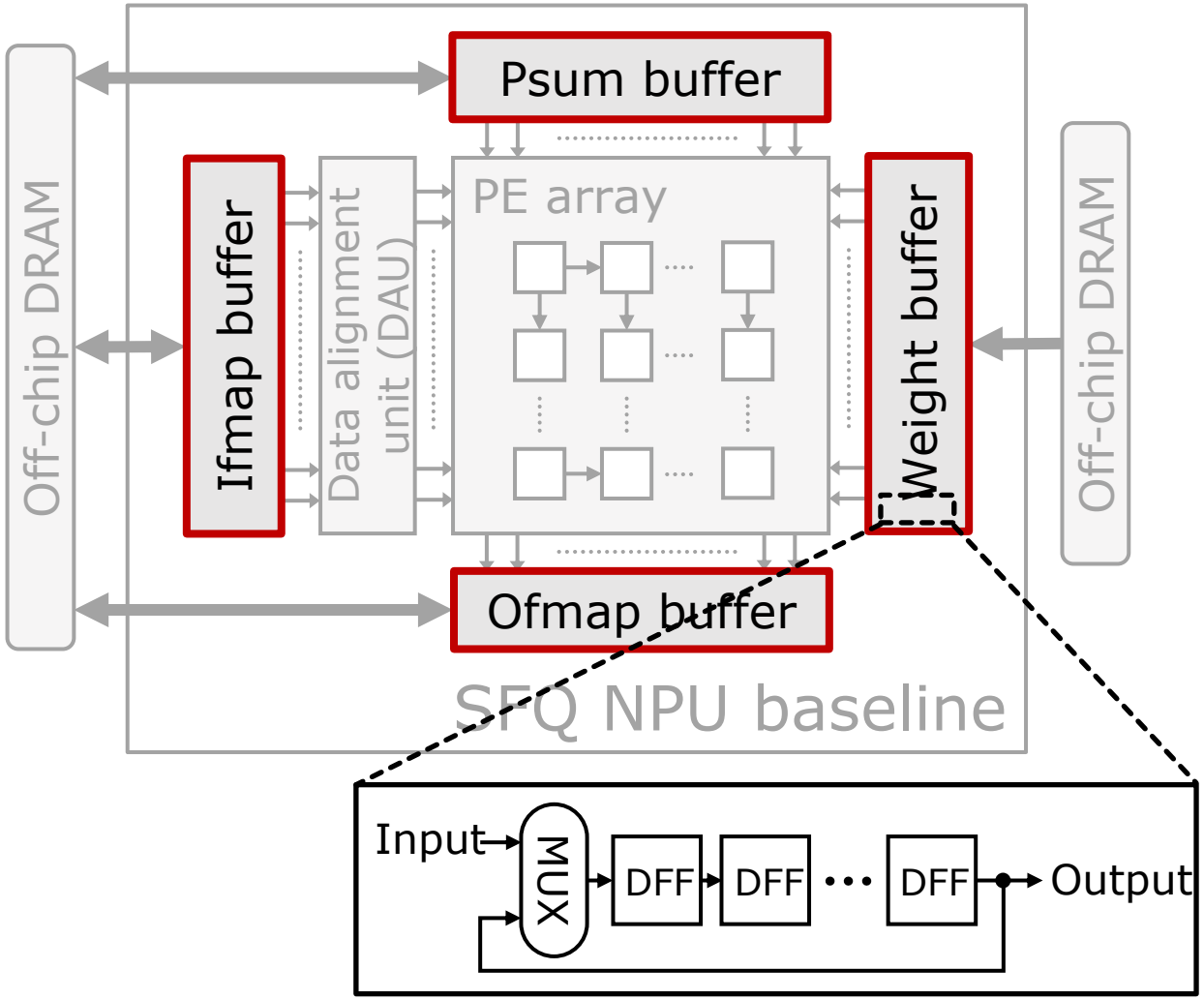


A good example: SFQ-based NPU



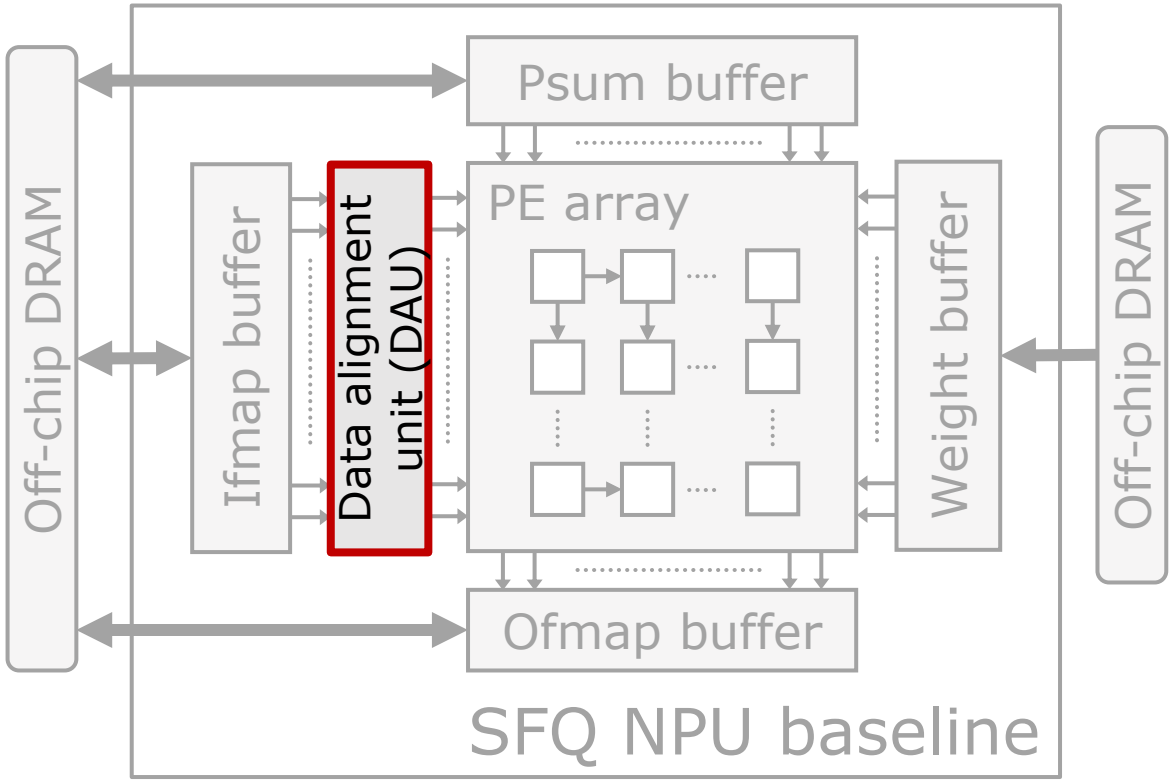
- On-chip network
  - 2D systolic array

A good example: SFQ-based NPU

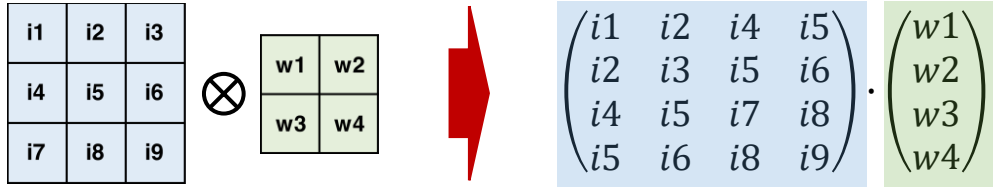


- On-chip network
  - 2D systolic array
- Buffer design
  - Shift-register-based buffers

A good example: SFQ-based NPU

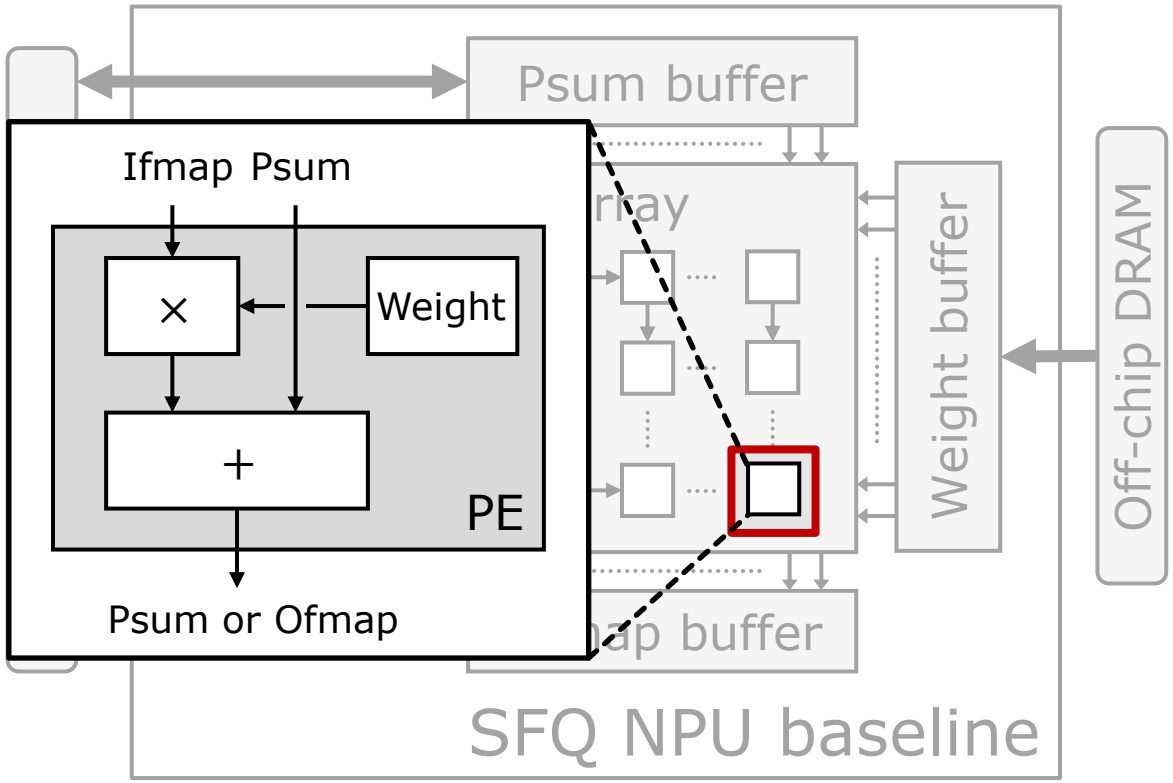


- On-chip network
  - 2D systolic array
- Buffer design
  - Shift-register-based buffers
  - Data alignment unit

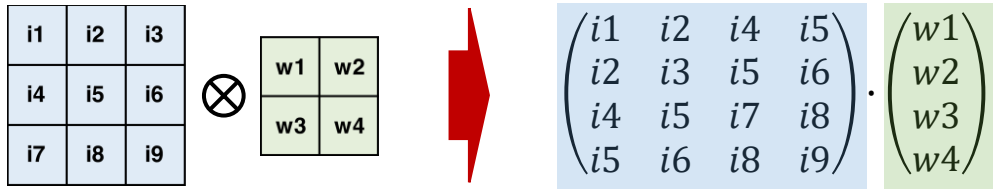


Convolutional operation Matrix multiplication

A good example: SFQ-based NPU



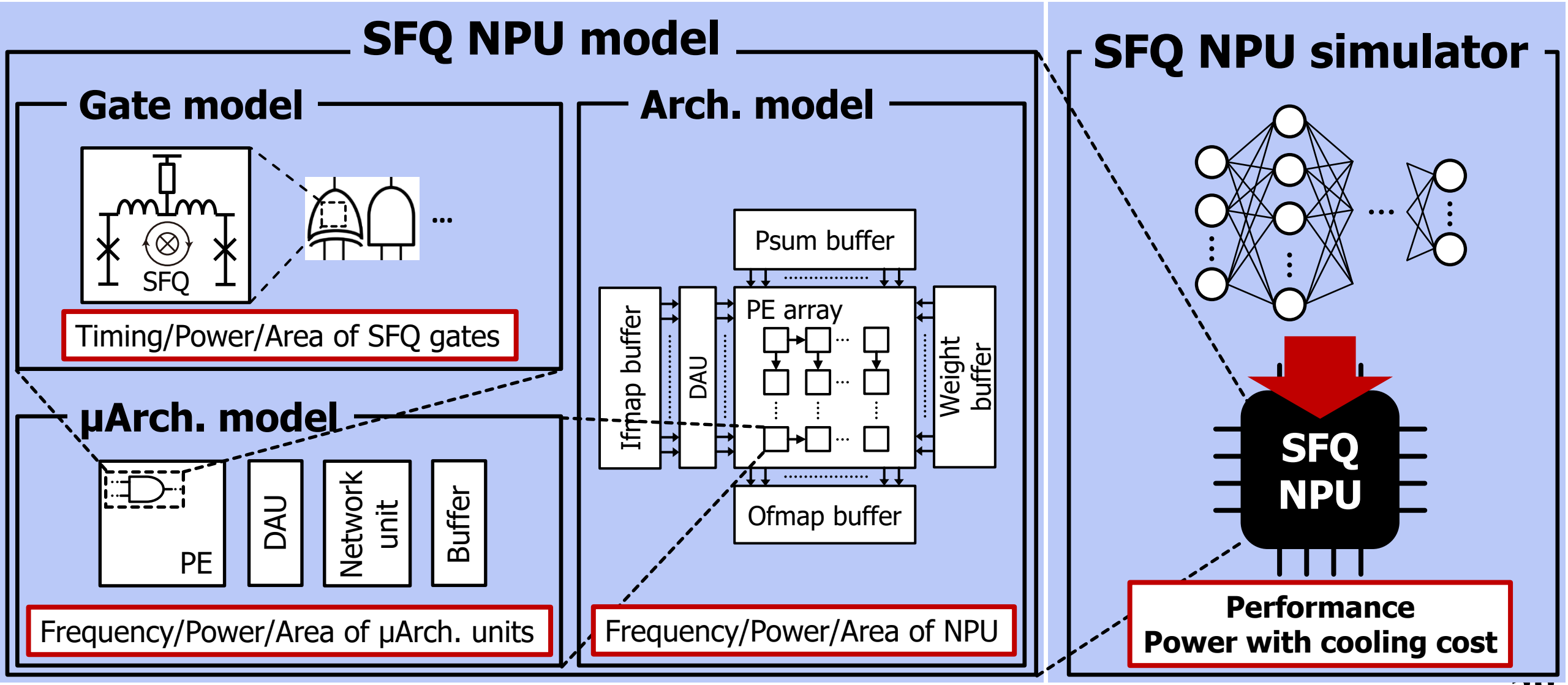
- On-chip network
  - 2D systolic array
- Buffer design
  - Shift-register-based buffers
  - Data alignment unit



Convolutional operation Matrix multiplication

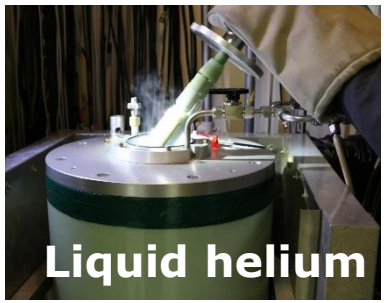
- Processing Element (PE) design
  - Weight stationary PE

SFQ NPU simulation framework: Overview



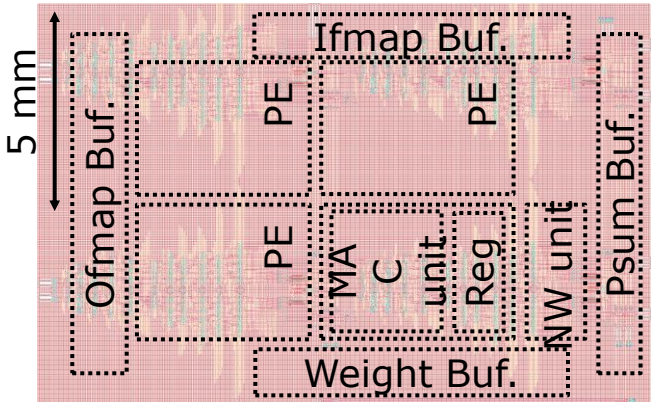
SFQ NPU model: Validation

- **SFQ NPU model accurately estimates freq./power/area**
  - Compared to the post-layout simulation and chips fabricated by 1.0 μm Nb [3]

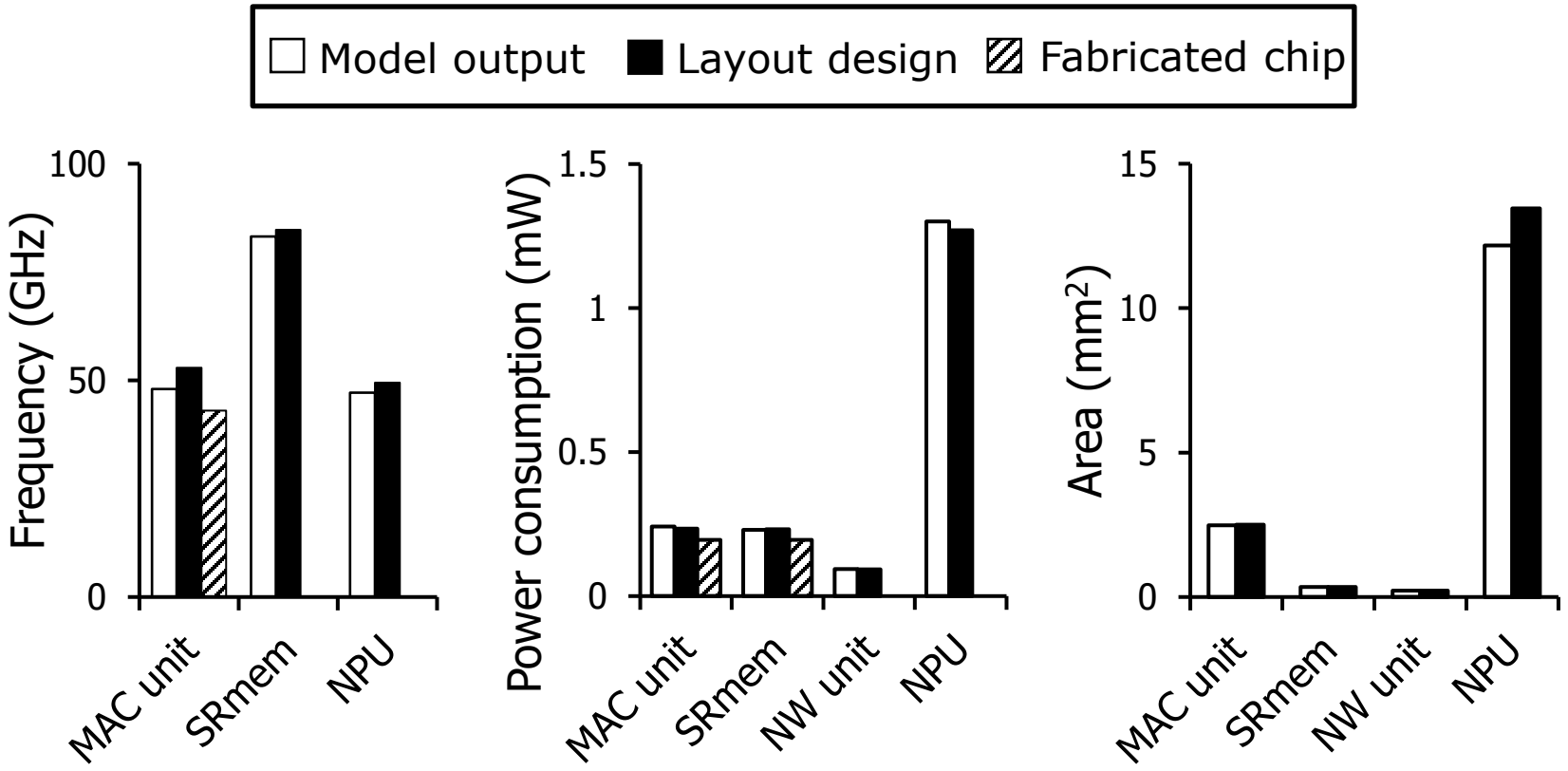


Validation setup

Chip of MAC



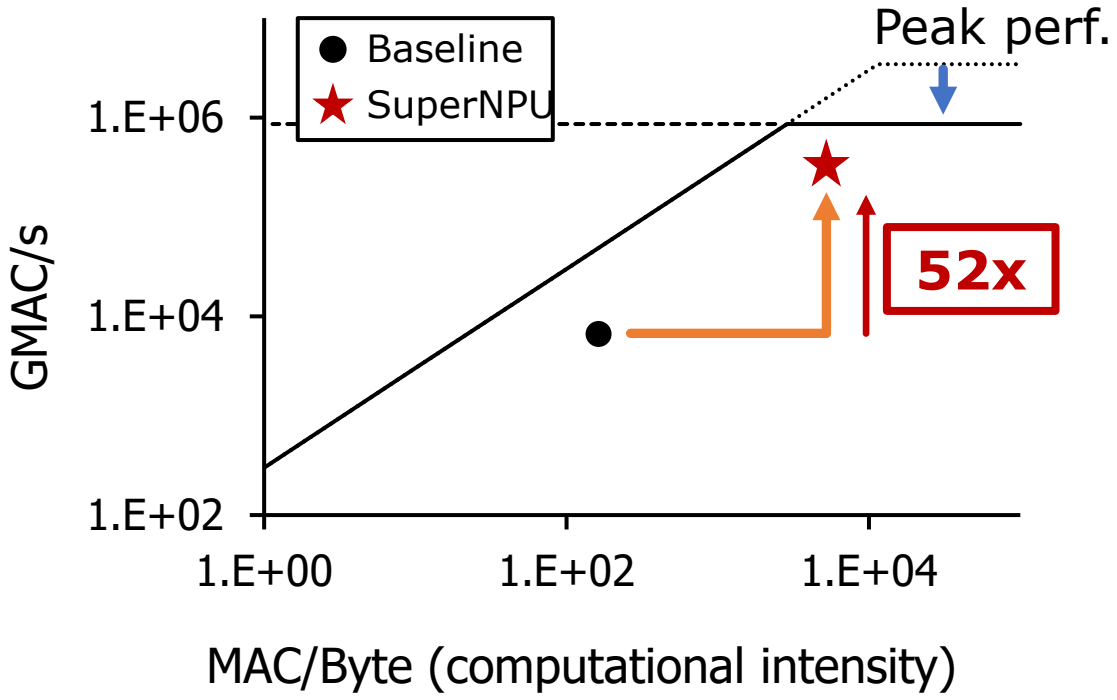
Layout of 4-bit 2×2 NPU

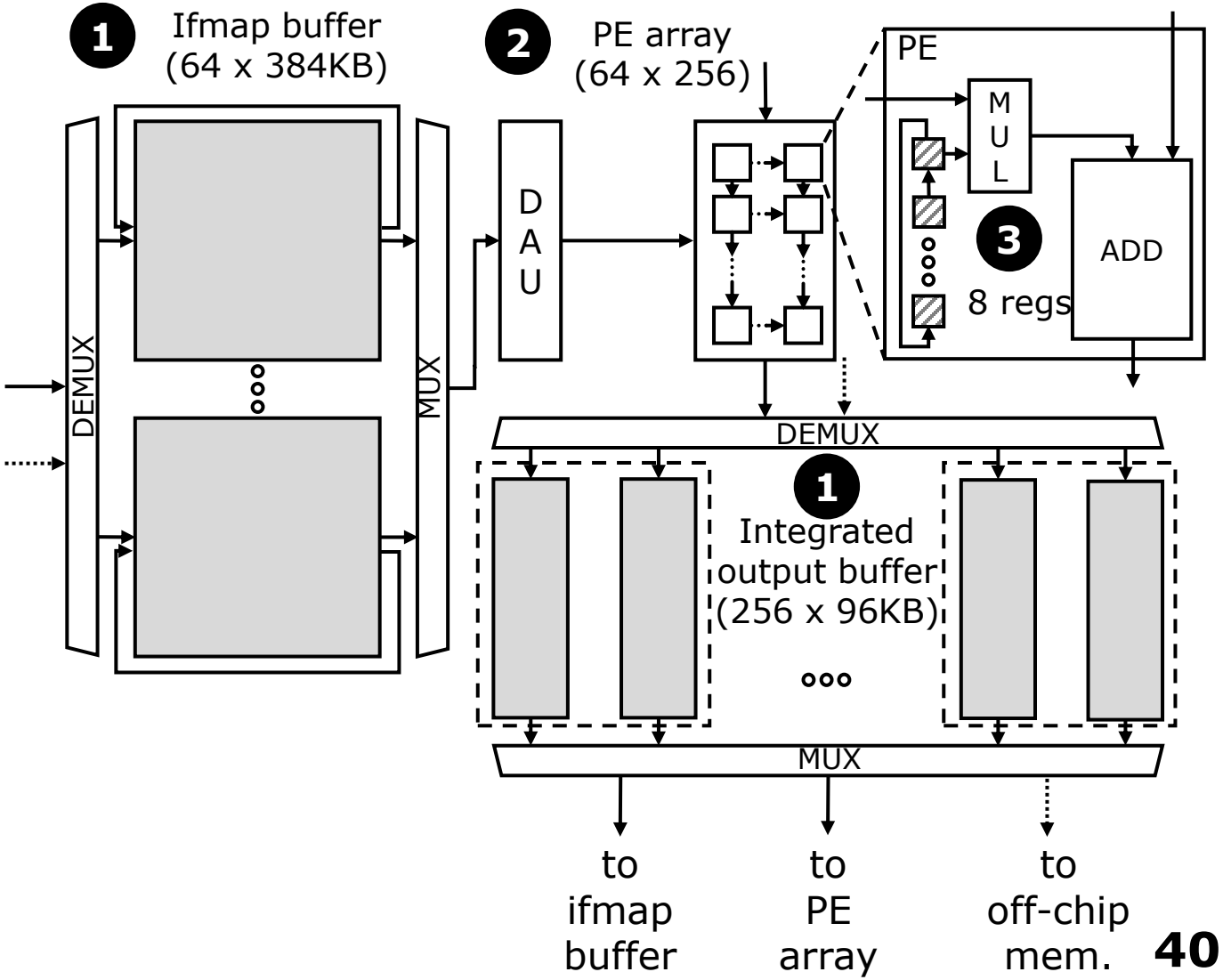


[3] S. Nagasawa et al., "Nb 9-layer fabrication process for superconducting large-scale SFQ circuits and its process evaluation," IEICE Tran. on Electron., 2014

SuperNPU: Optimized SFQ NPU architecture

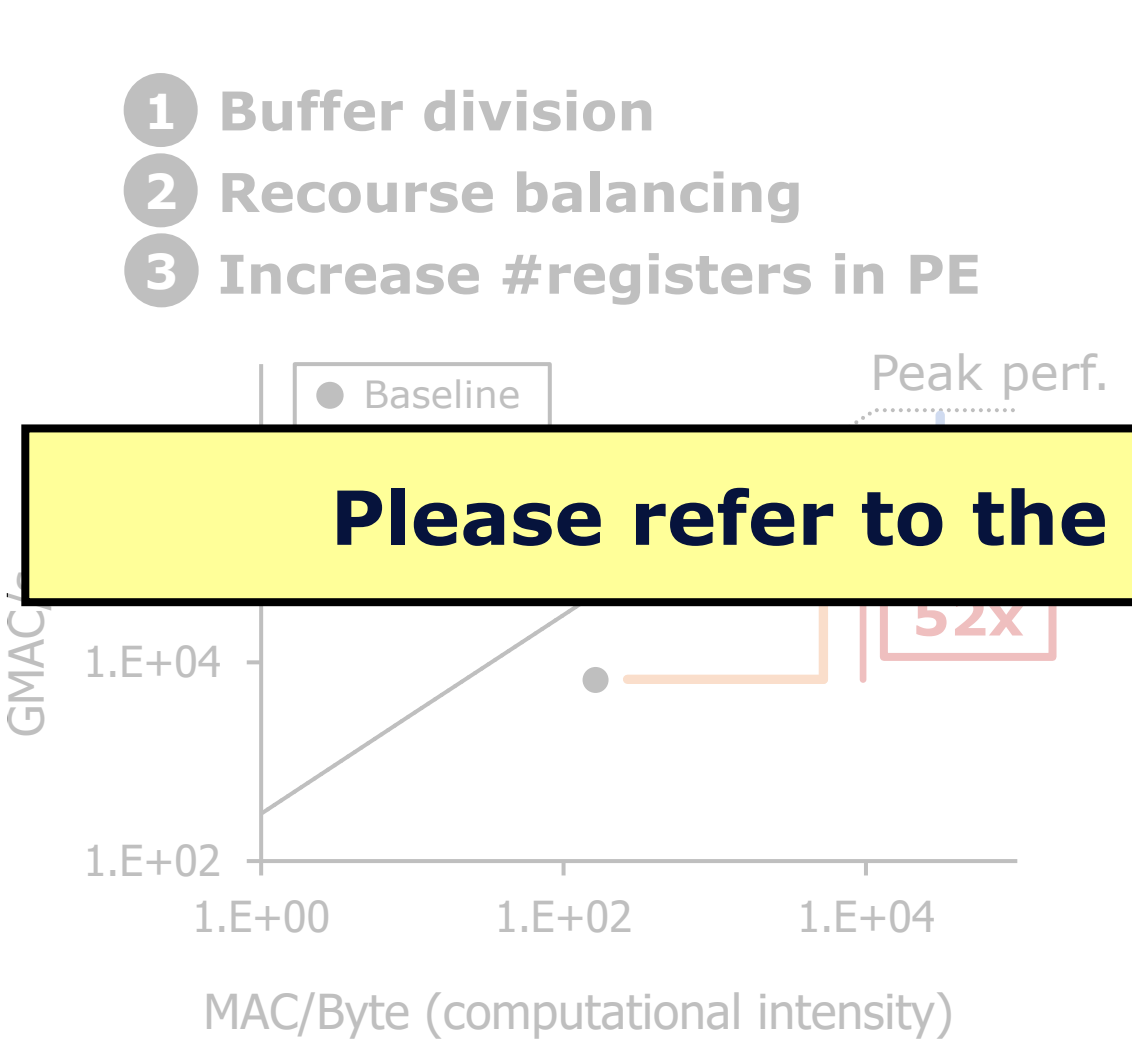
1. **Buffer division**
2. **Recourse balancing**
3. **Increase #registers in PE**

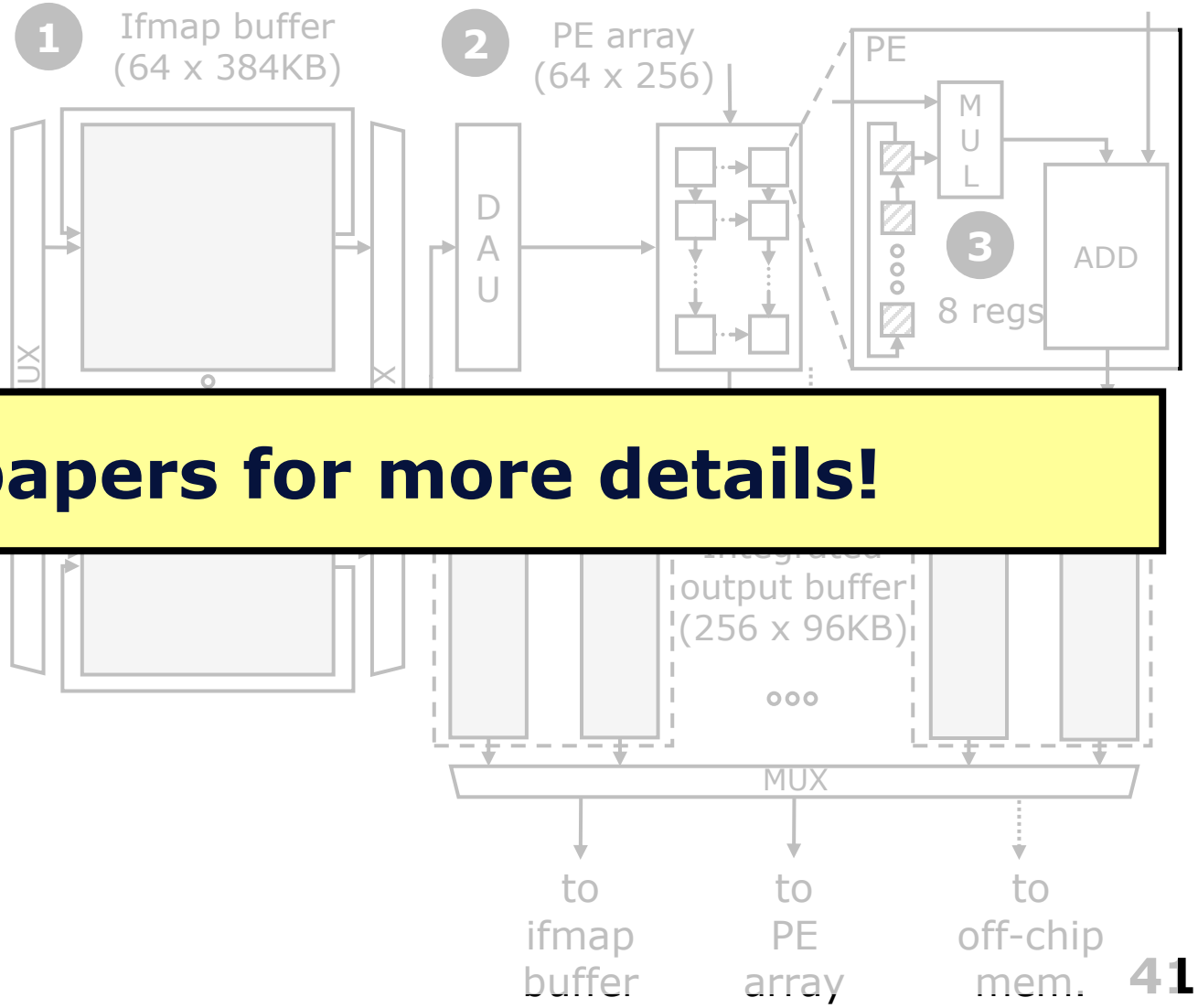




SuperNPU: Optimized SFQ NPU architecture

1. Buffer division
2. Recourse balancing
3. Increase #registers in PE



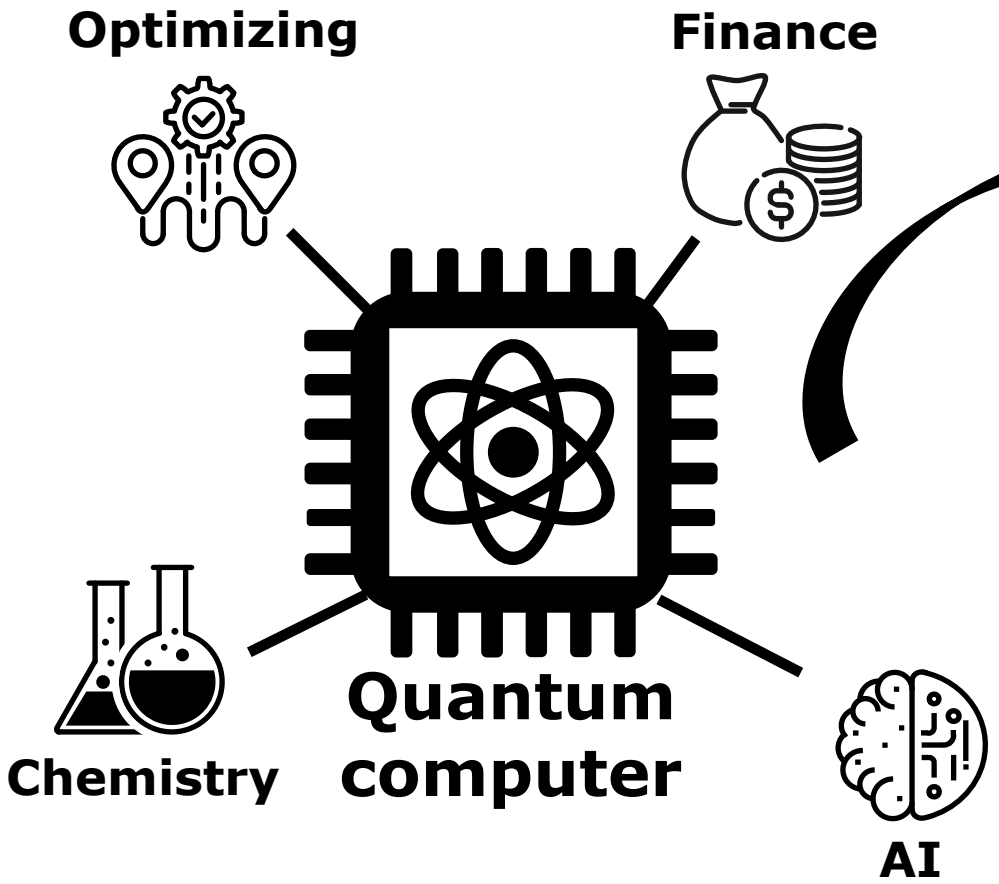


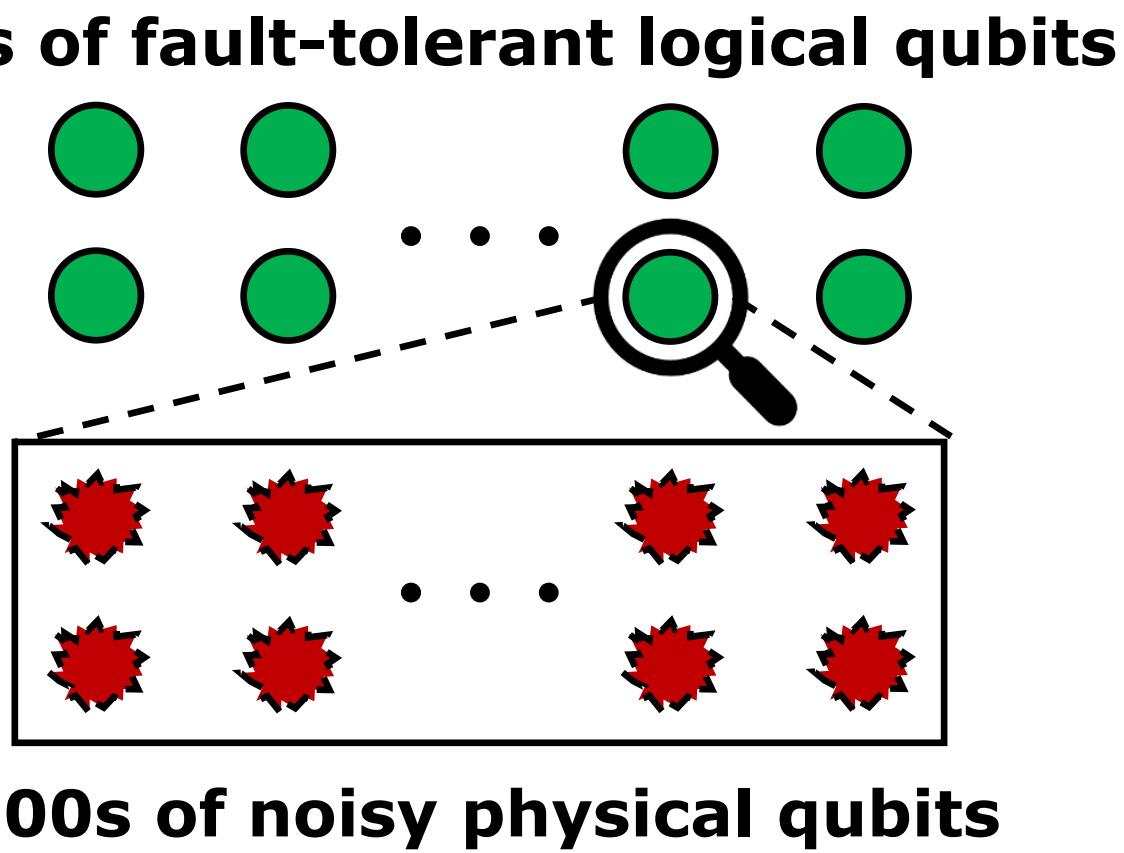
Please refer to the papers for more details!

Index

- Introduction
- Session #1: 77K CMOS-device computer arch. modeling
- Session #2: 4K SFQ-device computer arch. modeling
- **Session #3: 4K+77K quantum computer arch. modeling**

Toward large-scale quantum computer

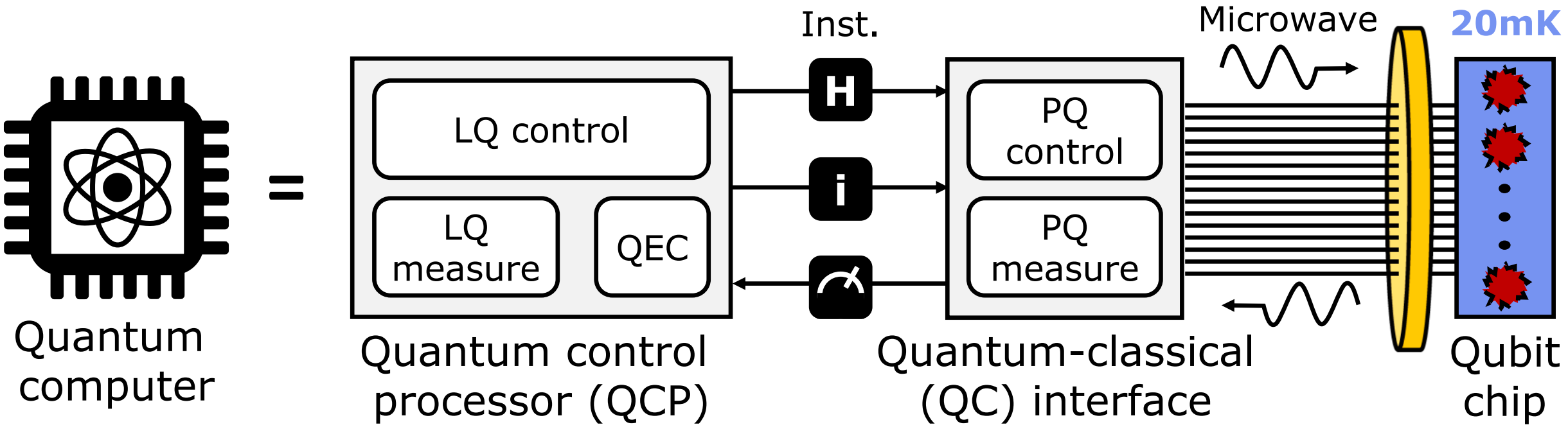




We need a fault-tolerant quantum computer using 10+K physical qubits!

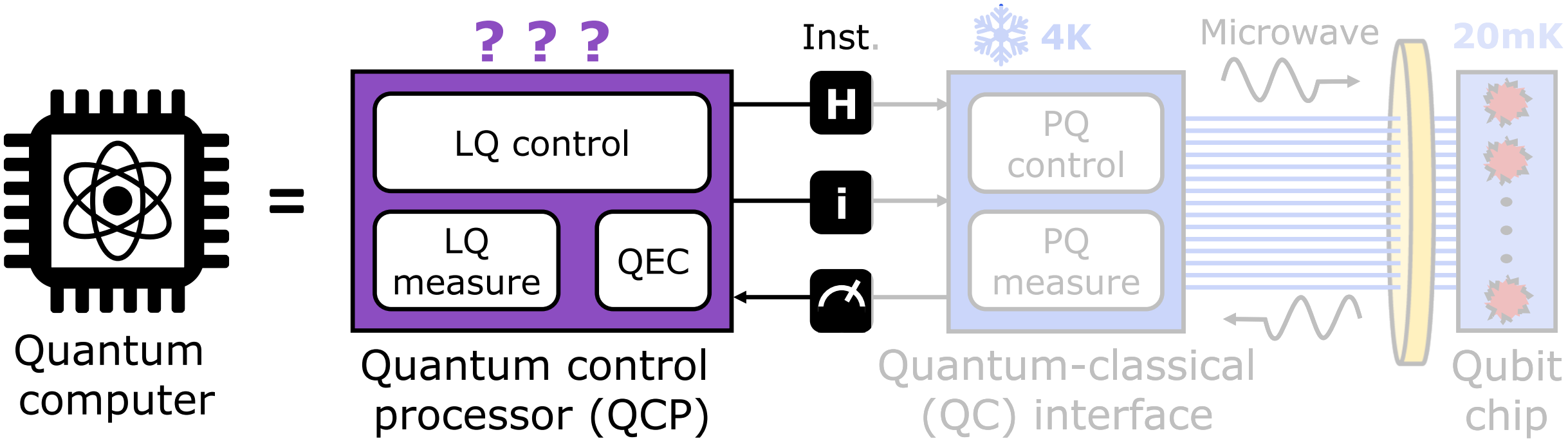
Scalable quantum control system

- **a scalable quantum-control processor & interface!**



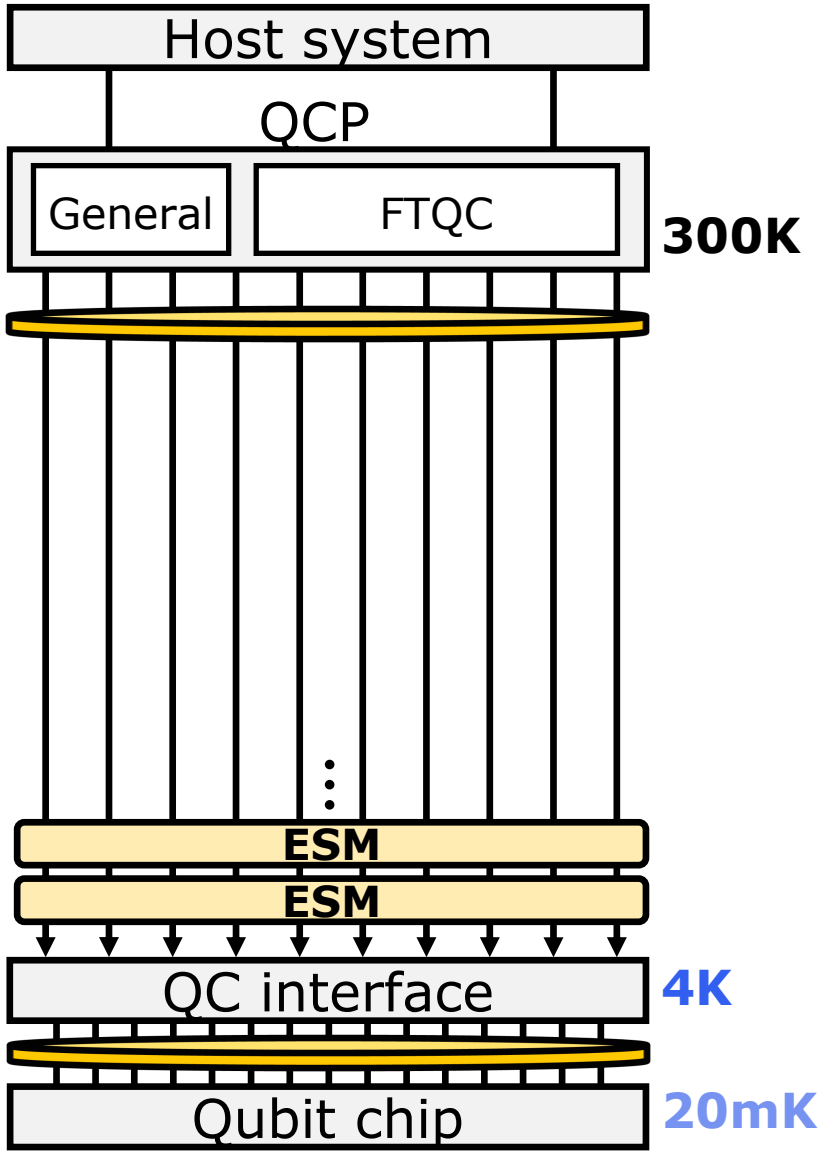
Architecting "scalable control processor"

- **Scalable QCP has not been actively explored yet**



Our research target: a scalable QCP architecture!

Limited scalability of today's QCP

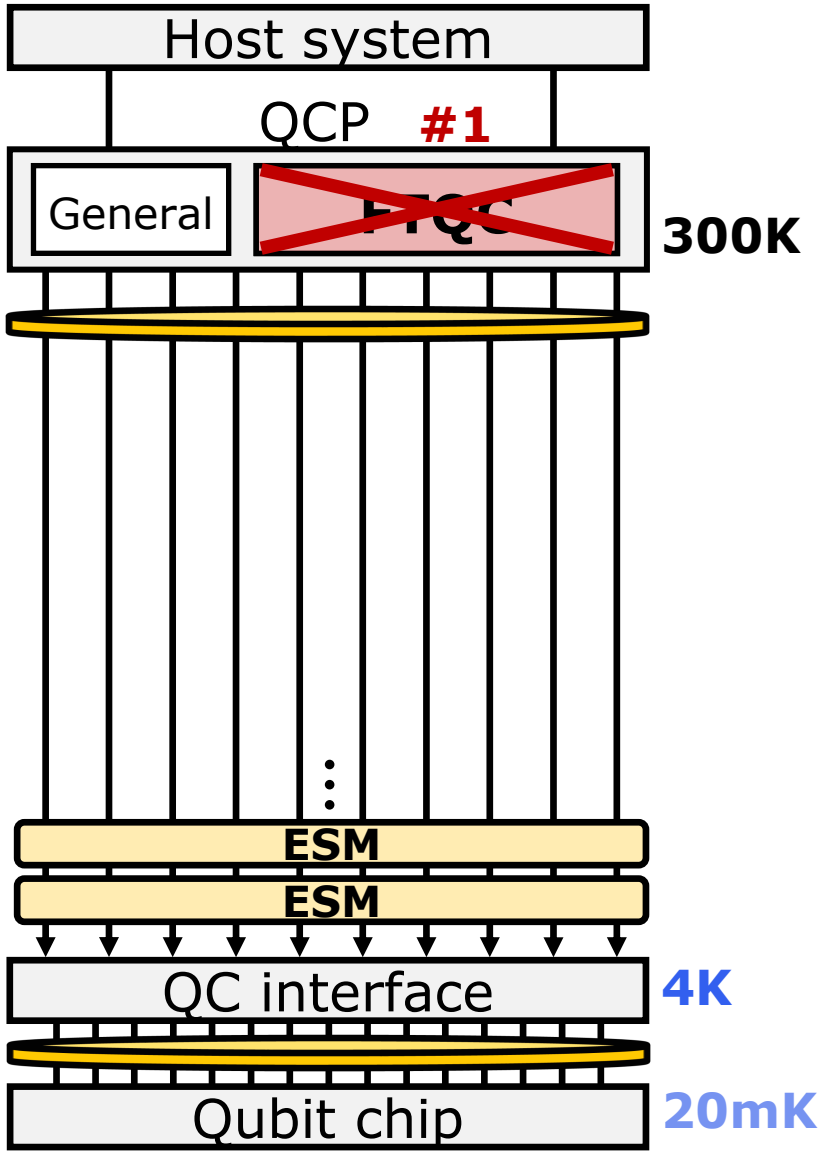


#1. Microarchitecture

#2. Temperature

#3. Technology

Limited scalability of today's QCP



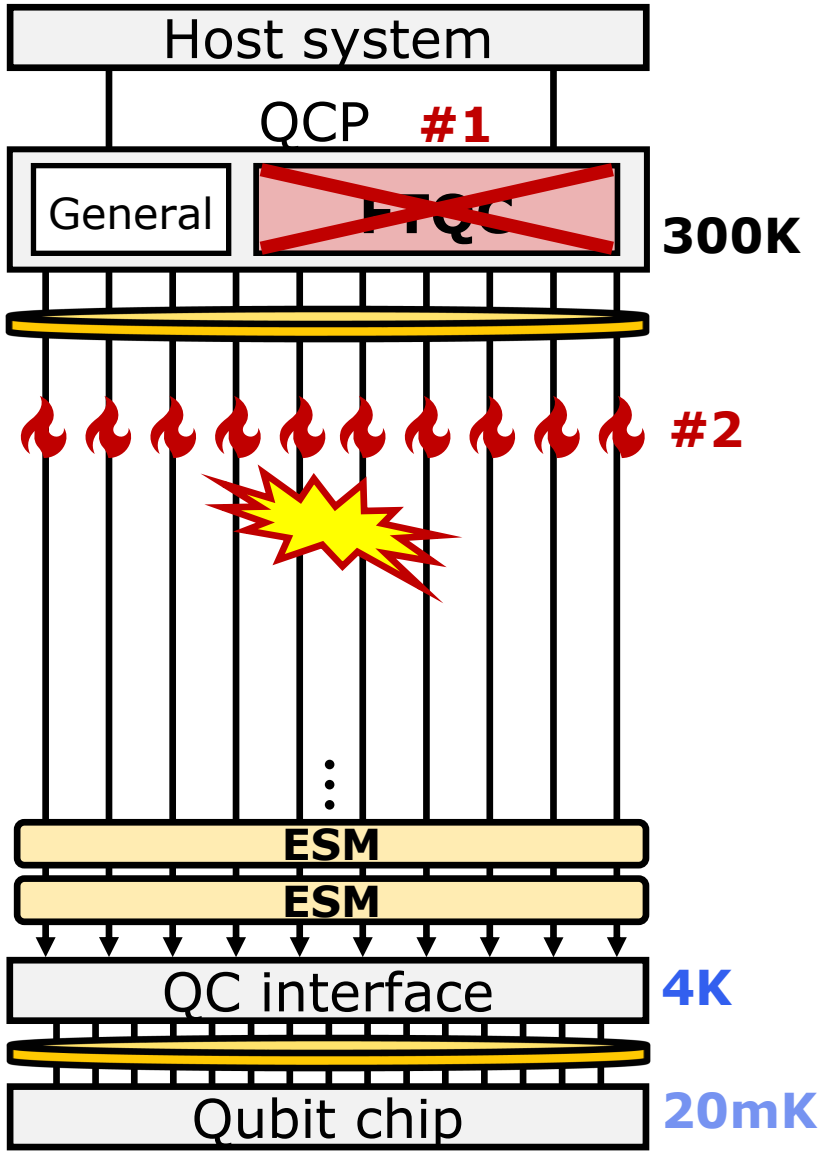
#1. Microarchitecture

No scalable μarch unit for the fault-tolerant quantum computing

#2. Temperature

#3. Technology

Limited scalability of today's QCP



#1. Microarchitecture

No scalable μarch unit for the fault-tolerant quantum computing

#2. Temperature

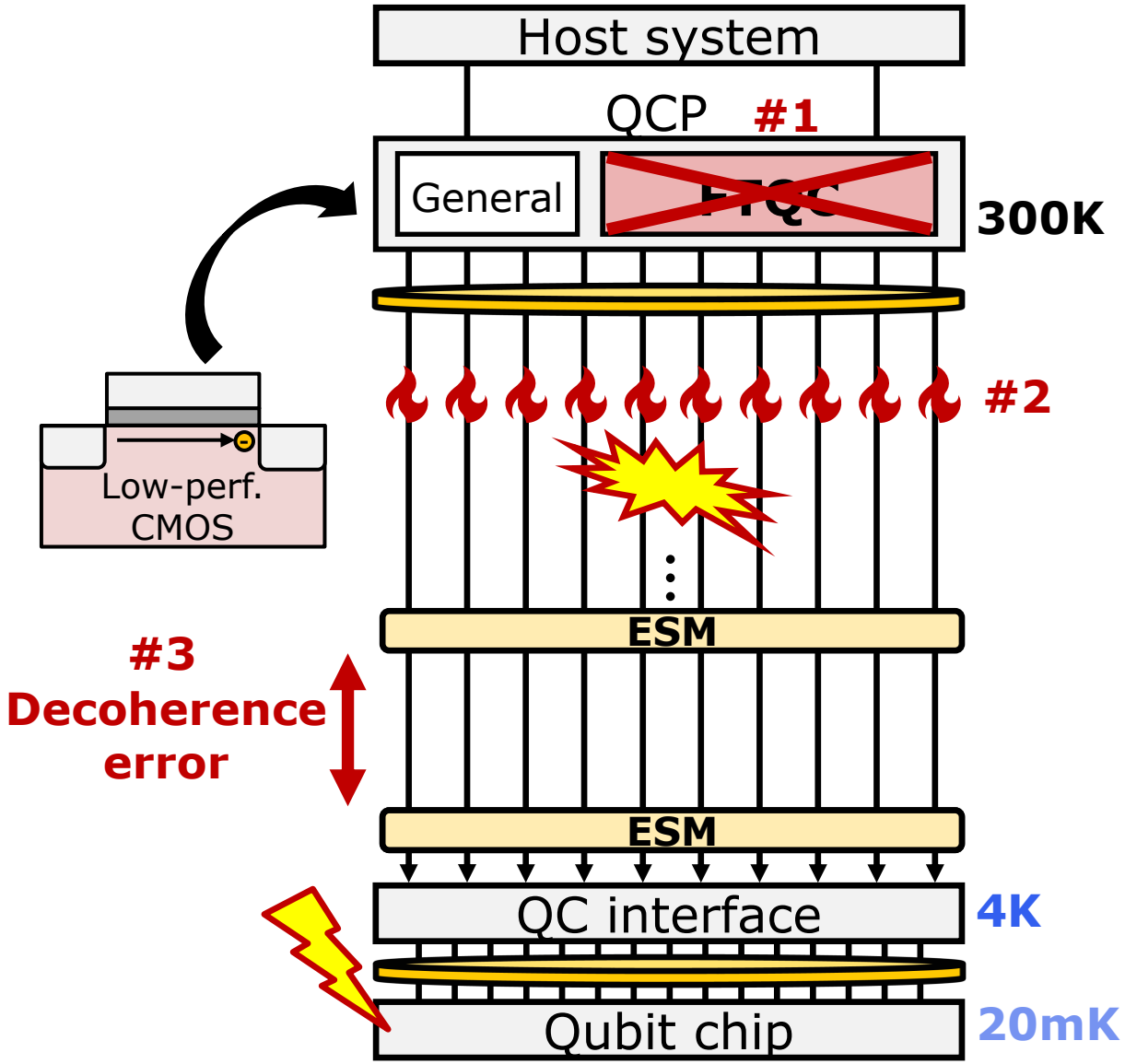
300K operation

→ Huge 300K-4K data transfer

→ Wire heat > 4K power budget

#3. Technology

Limited scalability of today's QCP



#1. Microarchitecture

No scalable μarch unit for the fault-tolerant quantum computing

#2. Temperature

300K operation

→ Huge 300K-4K data transfer

→ Wire heat > 4K power budget

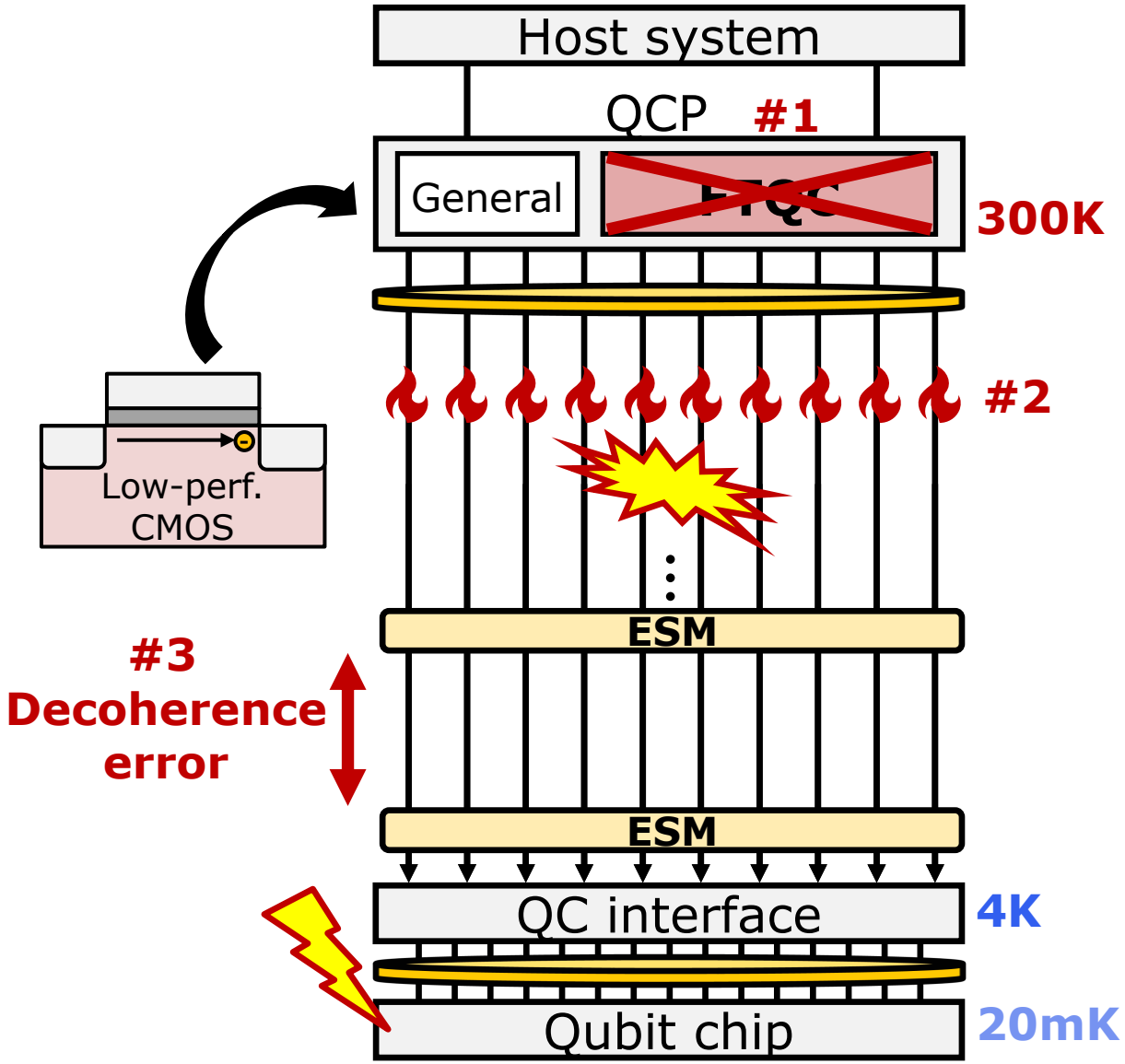
#3. Technology

Performance-limited CMOS

→ Slow QED or Low inst. BW

→ Decoherence error

Recent ideas for scalable QCP

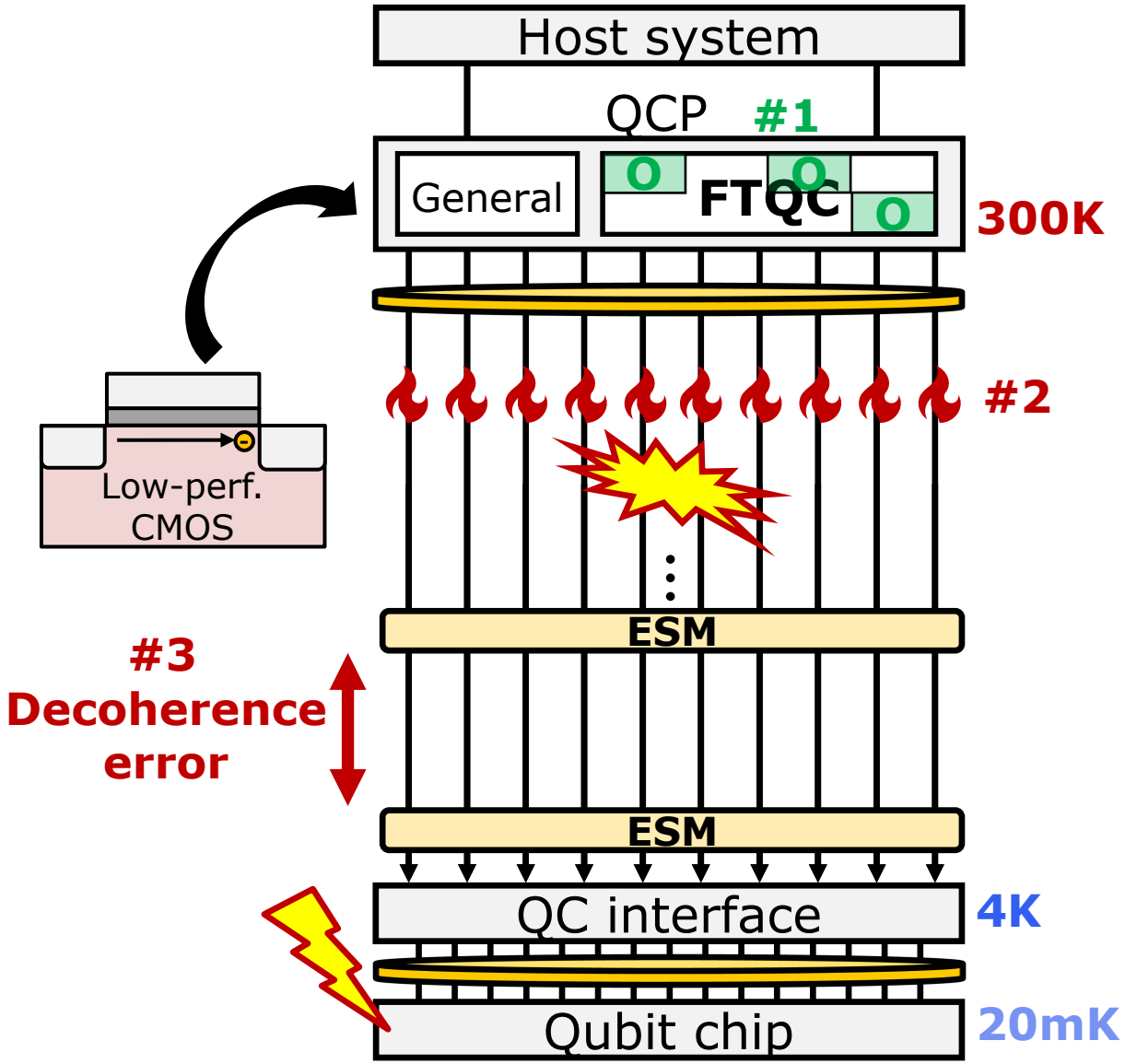


#1. Microarchitecture

#2. Temperature

#3. Technology

Recent ideas for scalable QCP



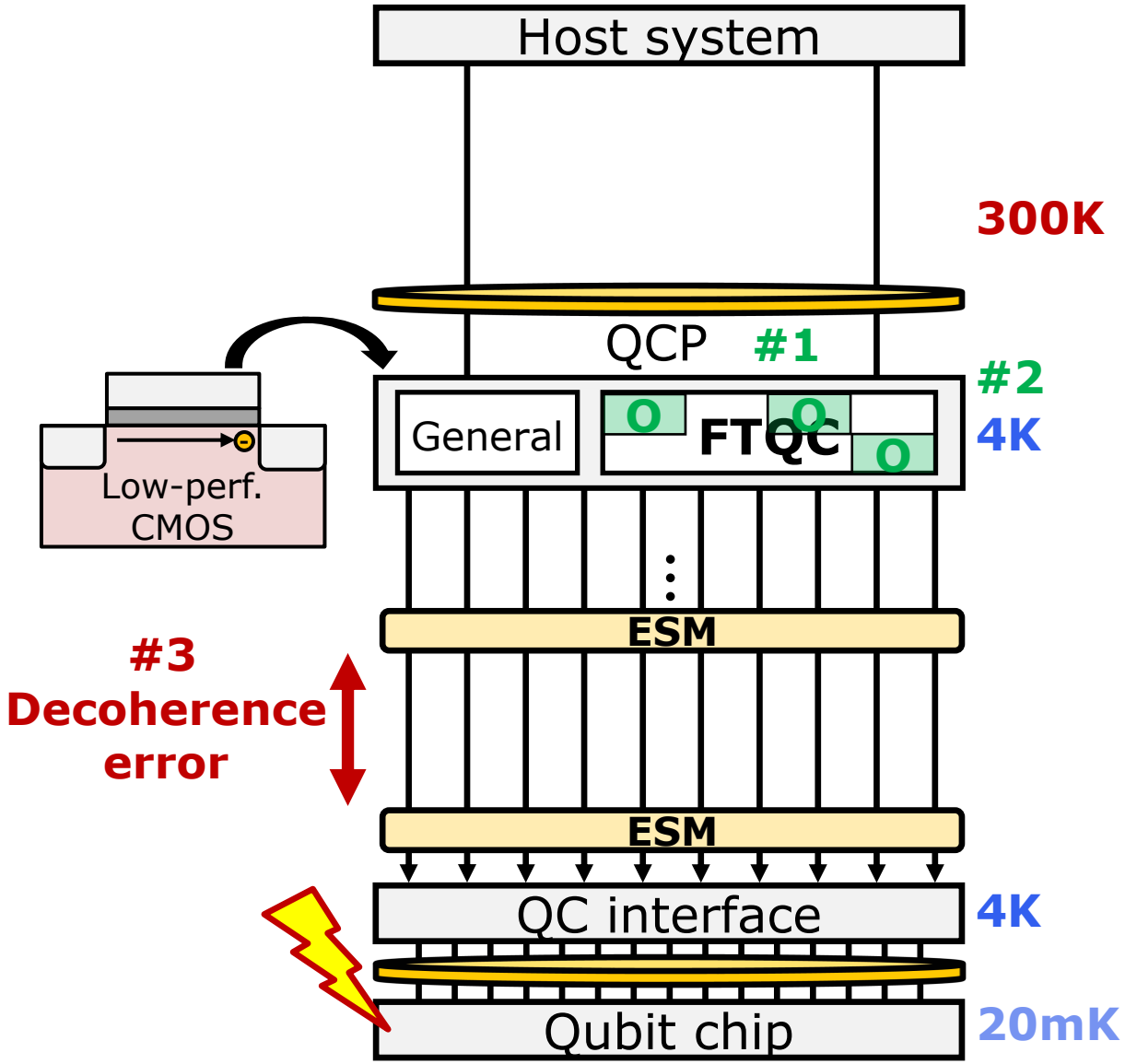
#1. Microarchitecture

(+) Scalable FTQC unit research

#2. Temperature

#3. Technology

Recent ideas for scalable QCP



#1. Microarchitecture

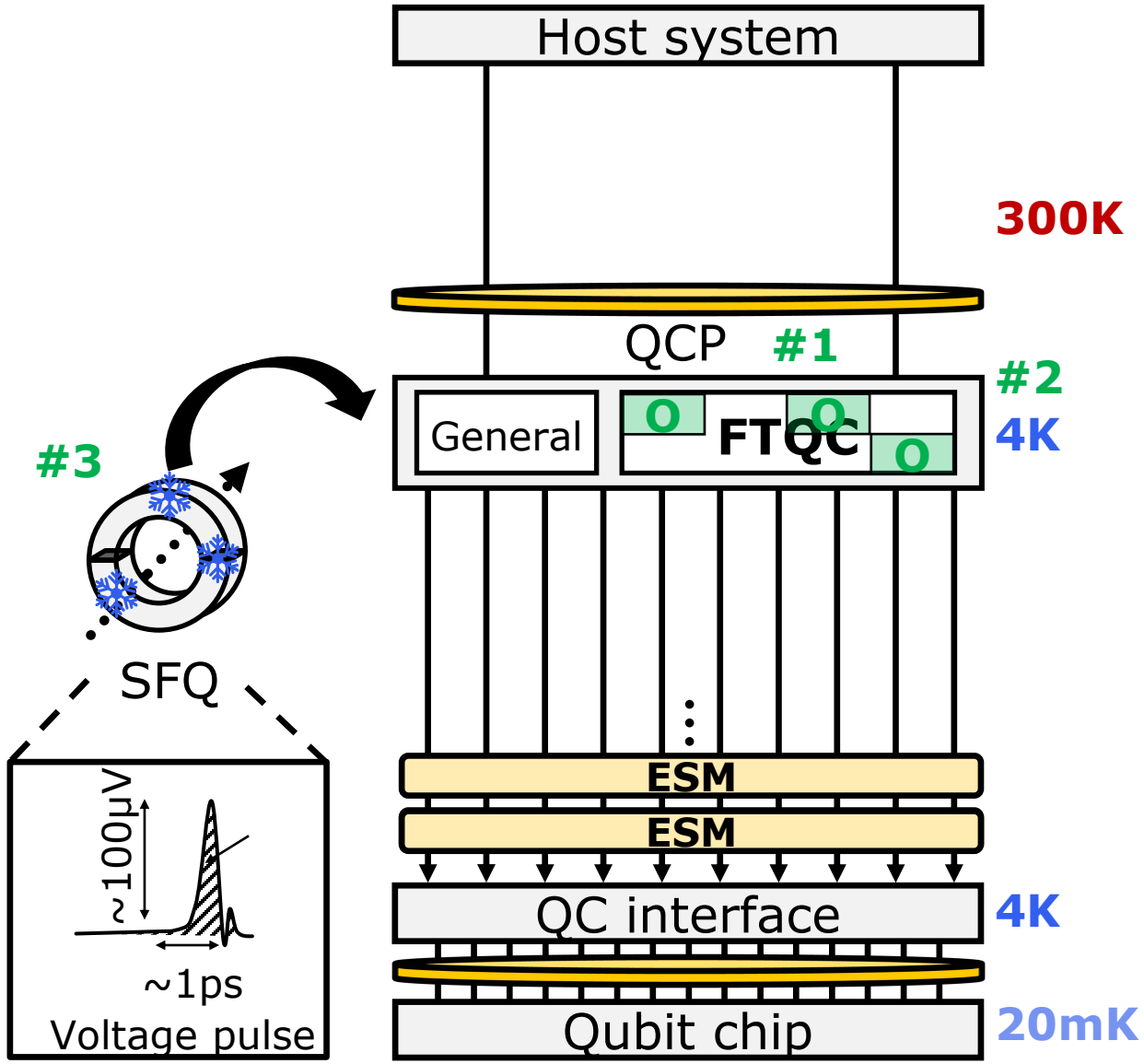
(+) Scalable FTQC unit research

#2. Temperature

(+) 4K operation

#3. Technology

Recent ideas for scalable QCP



#1. Microarchitecture

(+) Scalable FTQC unit research

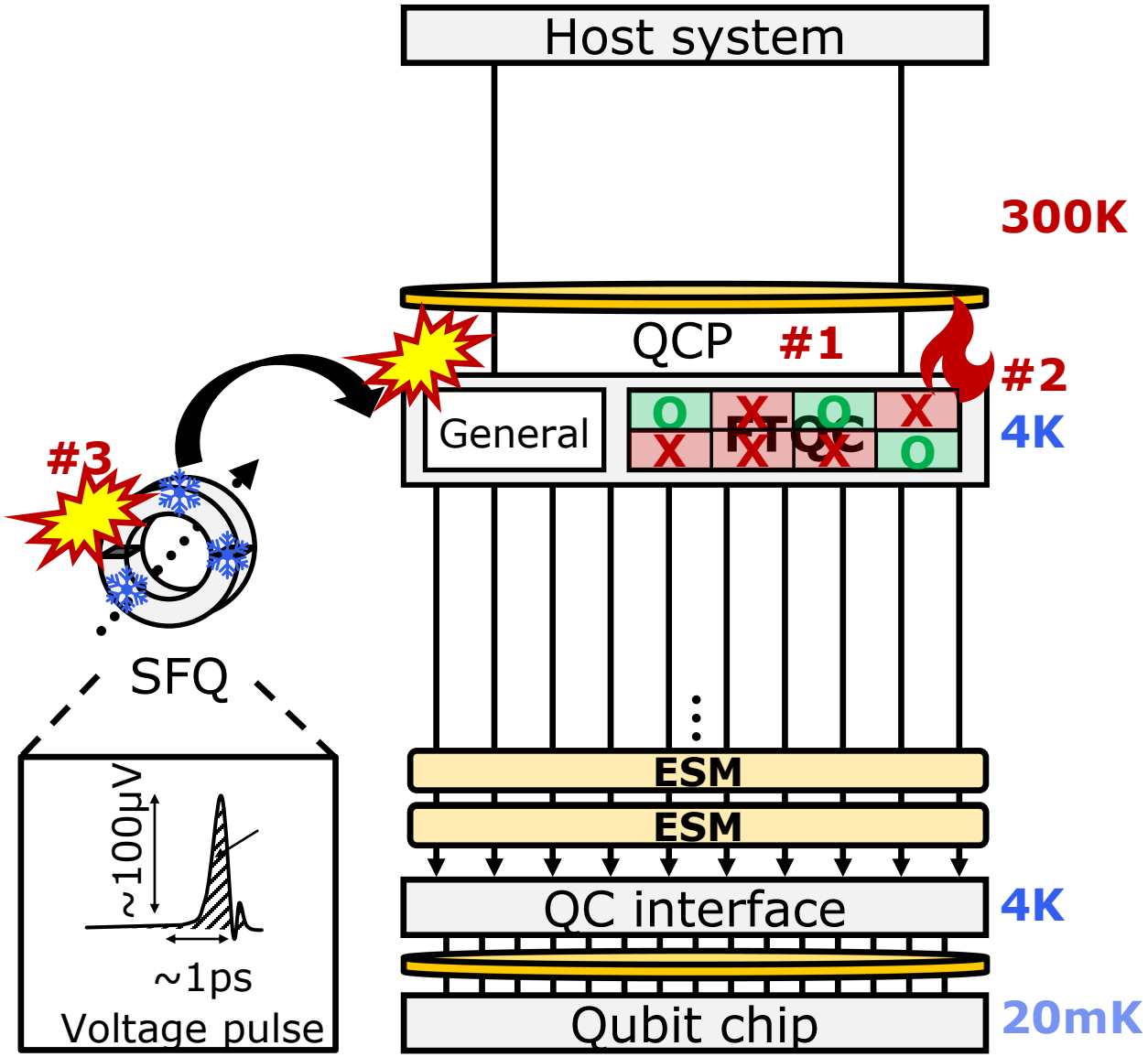
#2. Temperature

(+) 4K operation

#3. Technology

(+) Fast & Low-power SFQ

Recent ideas for scalable QCP



#1. Microarchitecture

(+) Scalable FTQC unit research
(−) Limited μarch coverage

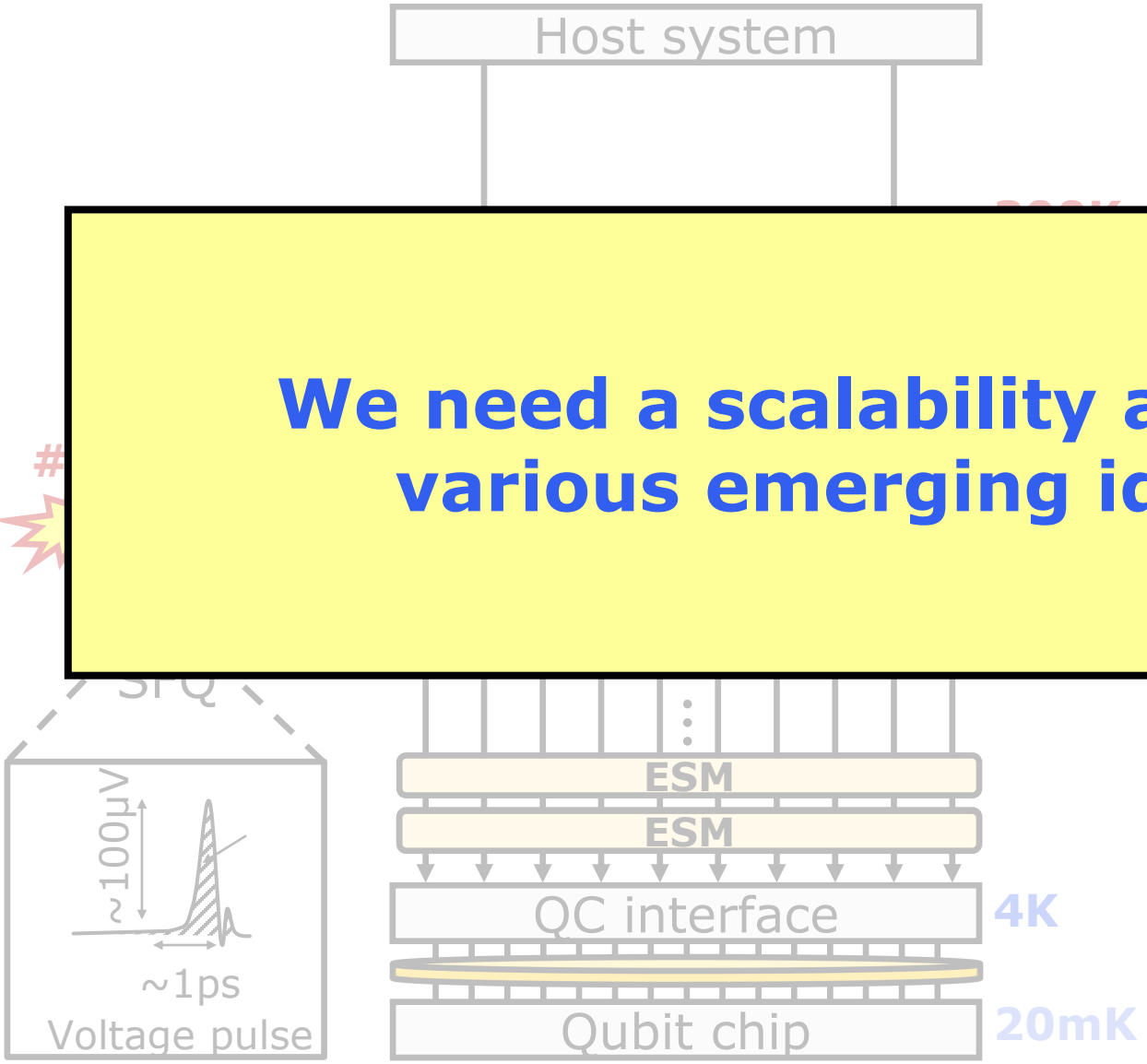
#2. Temperature

(+) 4K operation
(−) 4K device power dissipation

#3. Technology

(+) Fast & Low-power SFQ
(−) Non-trivial scalable design

Recent ideas for scalable QCP



We need a scalability analysis tool to evaluate various emerging ideas in all directions!

#1. Microarchitecture

(+) Scalable FTQC unit research

#3. Technology

(+) Fast & Low-power SFQ

(–) Non-trivial scalable design

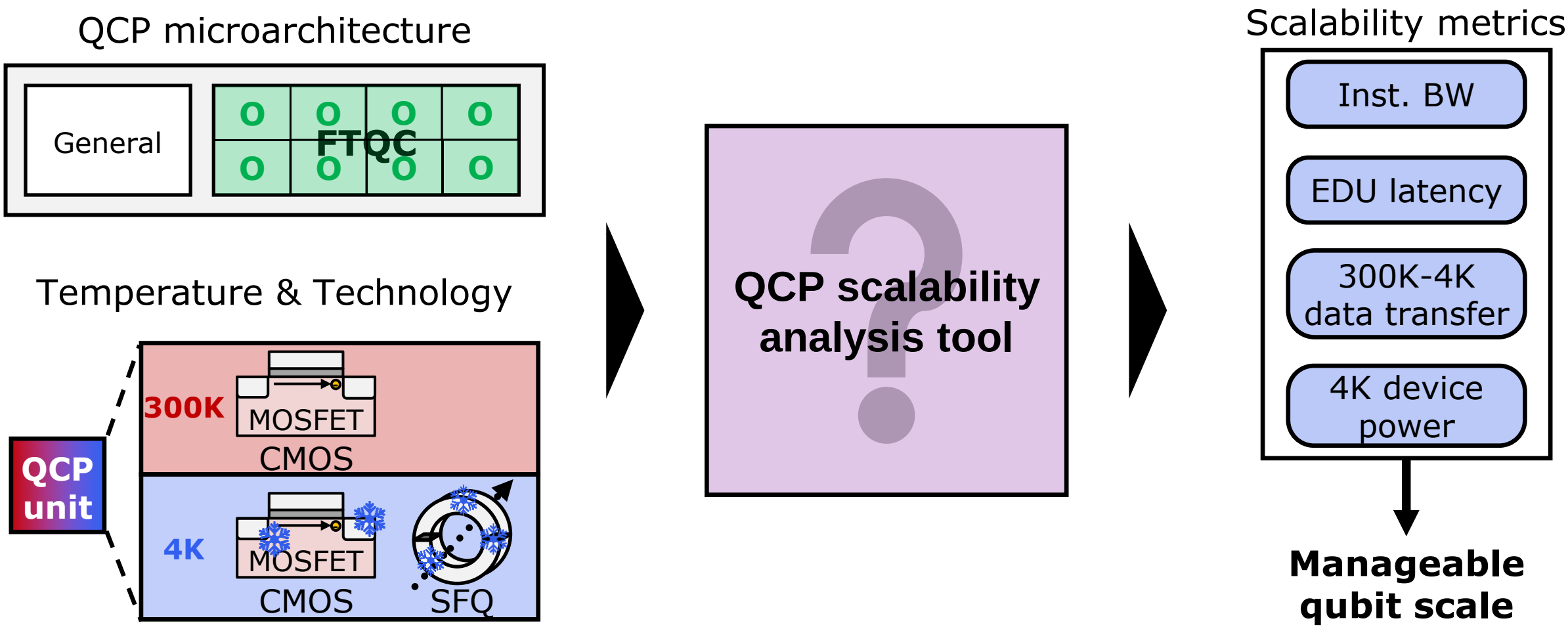
Research goals

Session #3

Step 1	Develop a modeling & simulation tool for QCP
Step 2	Propose a scalable QCP architecture

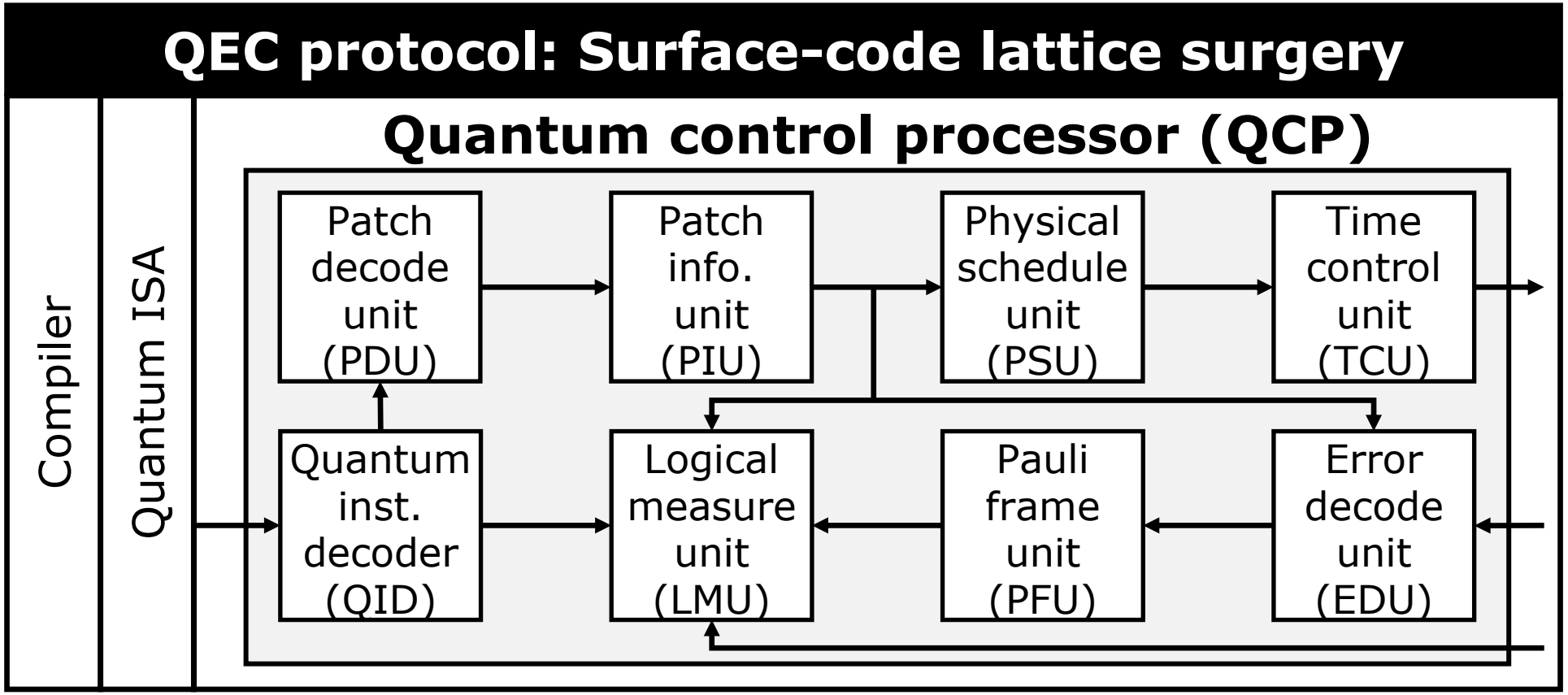
XQsim: Research goal

- **QCP scalability analysis tool to evaluate various ideas in microarchitecture, temperature, and technology**

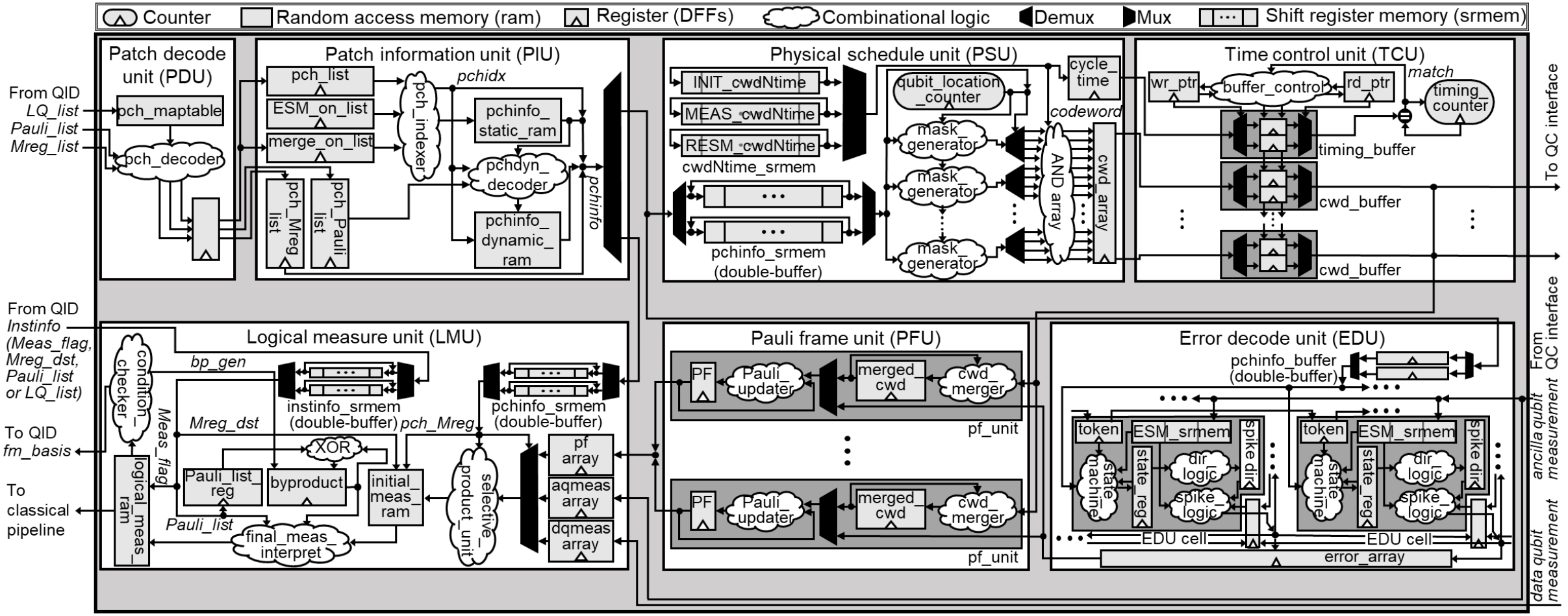


QC microarchitecture overview

- **First full implementation of the fault-tolerant QCP μarch**
  - Implement all the necessary hardware units for fault-tolerant quantum computing in RTL
  - Target surface-code lattice surgery and develop custom compiler & quantum ISA



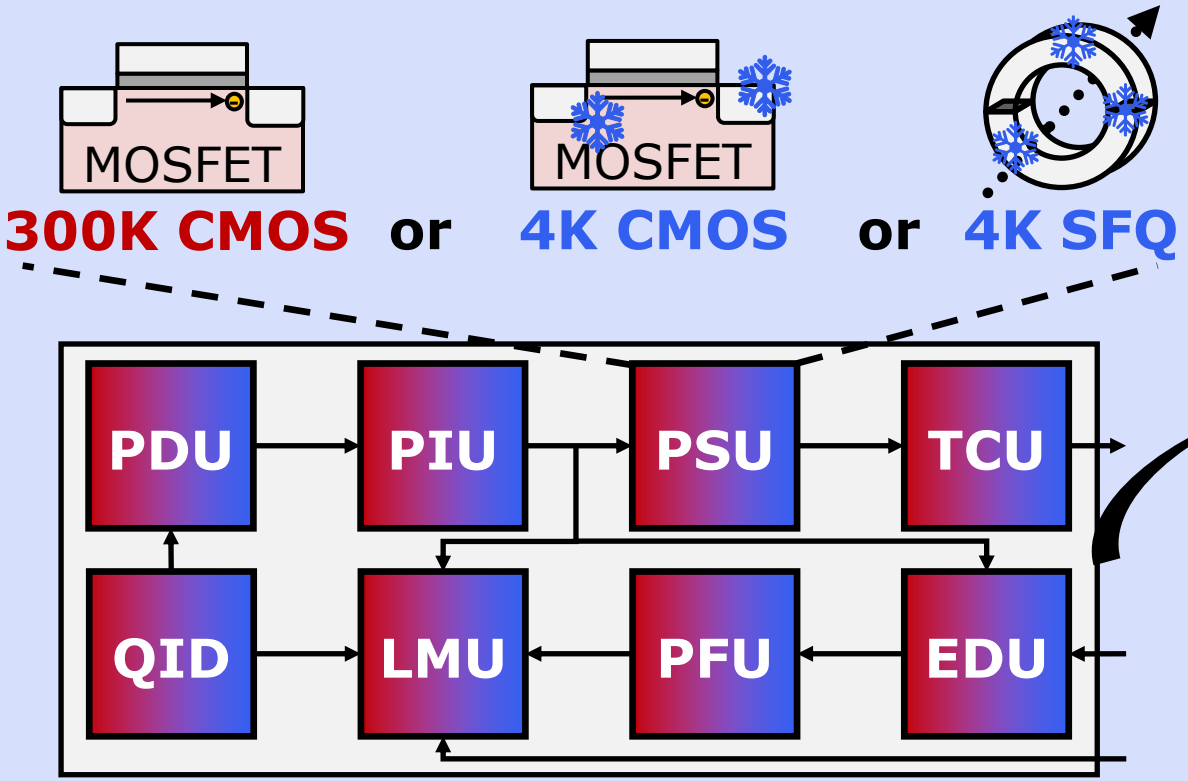
Further microarchitecture details



Please refer to our paper for all these details!

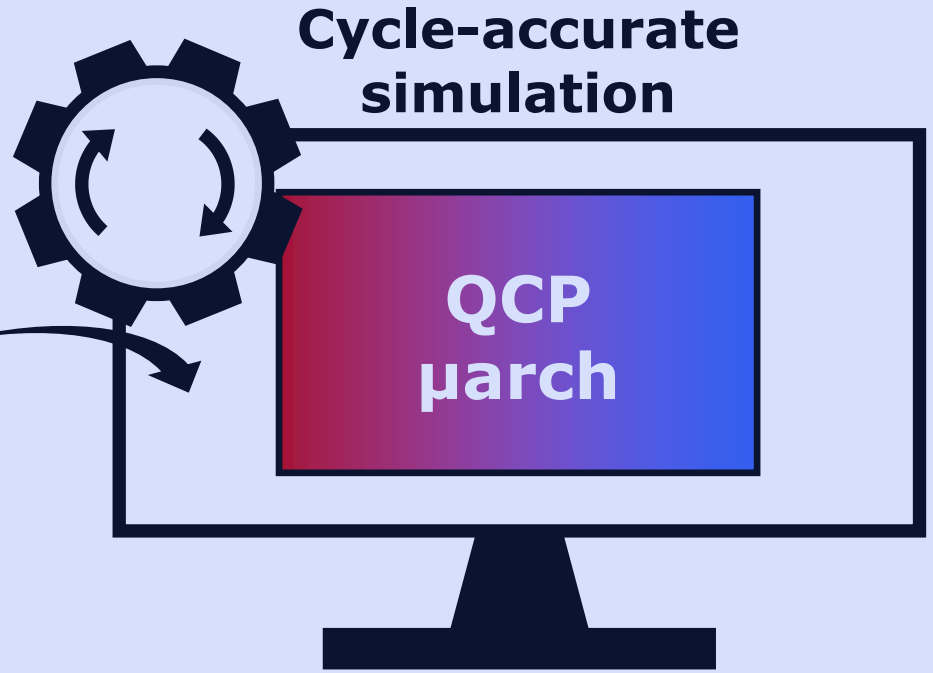
XQsim: Overview

1) XQ-estimator



Frequency and power of each µarch unit

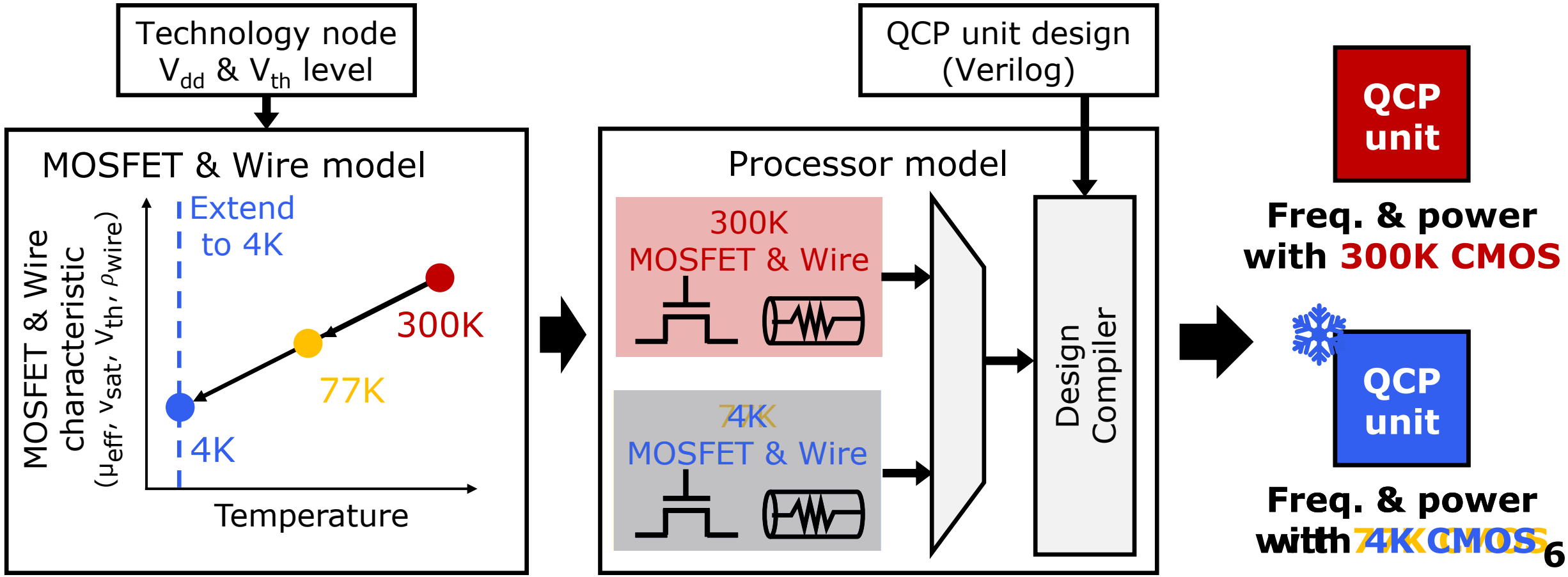
2) XQ-simulator



Manageable qubit scale and scalability bottlenecks

XQ-estimator: CMOS model

- **Build our model on our 77K CryoCMOS model (CryoModel)**
- **Extend MOSFET & Wire model's temperature coverage to 4K**
  - i.e., carrier mobility (μ_{eff}), saturation velocity (v_{sat}), threshold voltage (V_{th}), wire resistivity (ρ_{wire})



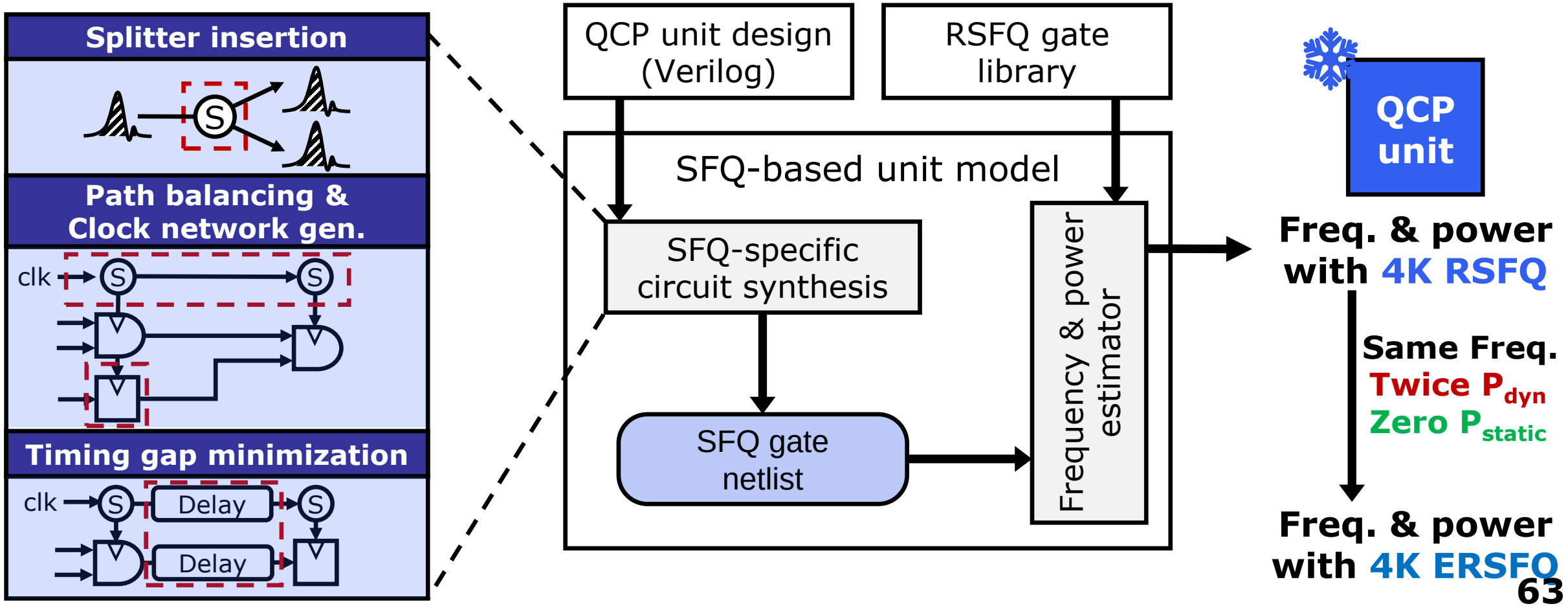
XQ-estimator: CMOS model

- Build our model on our 77K CryoCMOS model (CryoModel)
- Extend MOSFET & Wire model's temperature coverage to 4K
  - i.e., carrier mobility (μ_{eff}), saturation velocity (v_{sat}), threshold voltage (V_{th}), wire resistivity (ρ, ...)

We exploit our cryo-CMOS device & arch model (Session #1)

XQ-estimator: SFQ model

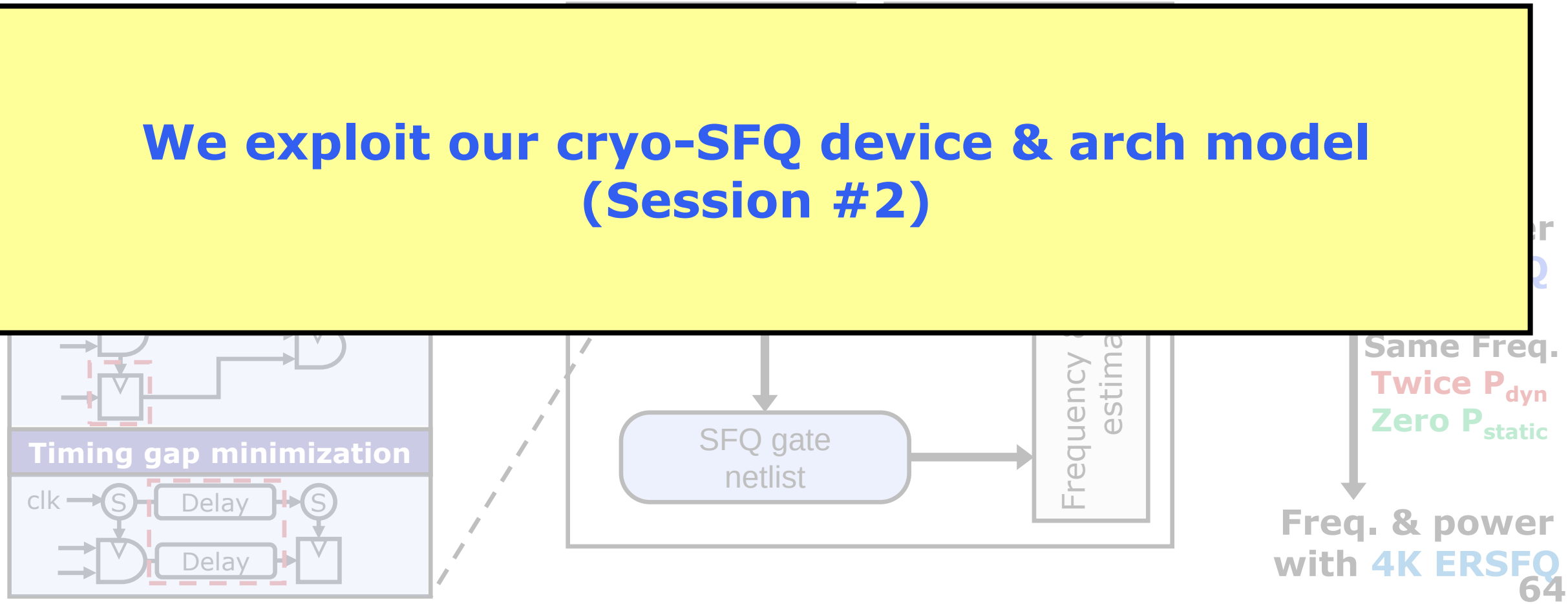
- **Generate an SFQ gate netlist by applying SFQ-specific circuit features**
- **Estimate the frequency and power by using RSFQ gate library data**



XQ-estimator: SFQ model

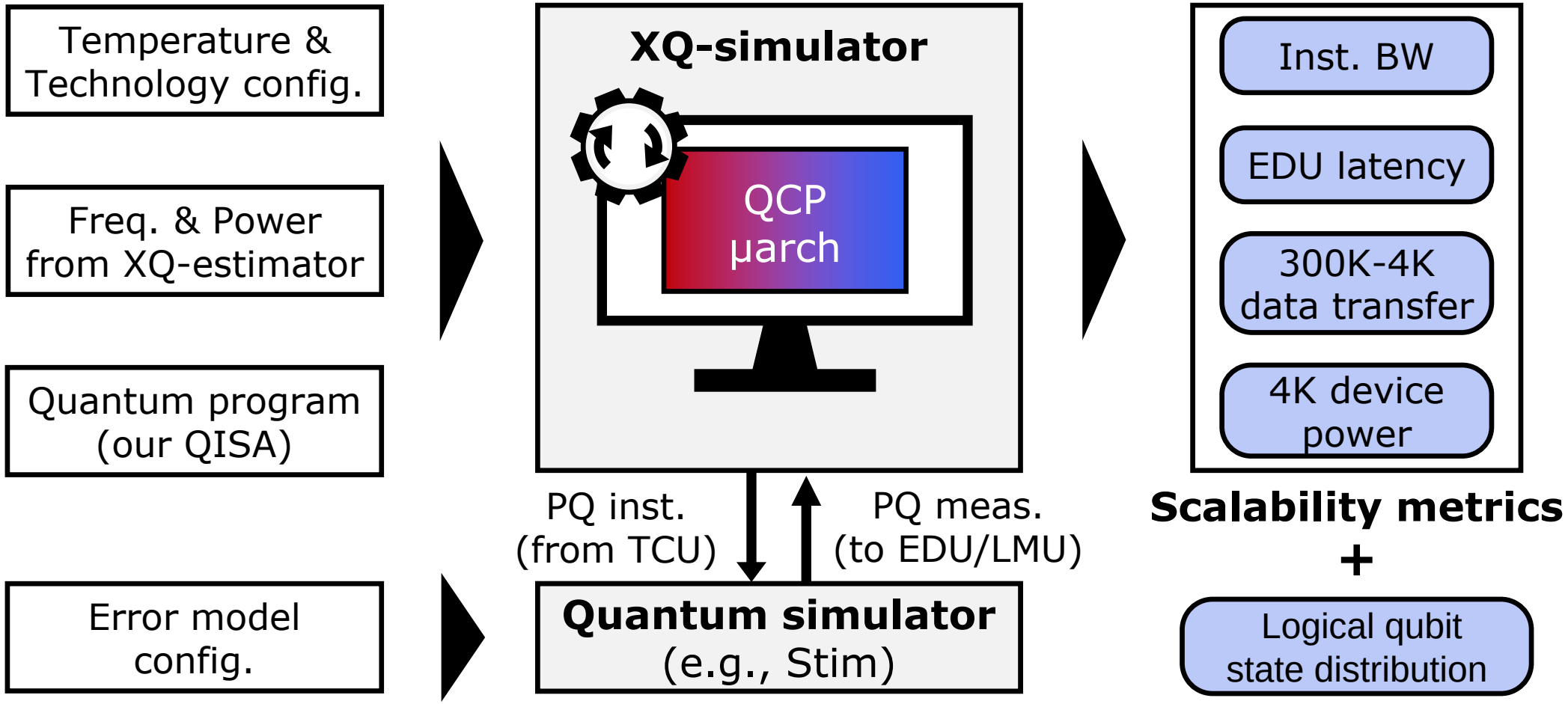
- Generate an SFQ gate netlist by applying SFQ-specific circuit features
- Estimate the frequency and power by using RSFQ gate library data

We exploit our cryo-SFQ device & arch model (Session #2)



XQ-simulator: Overview

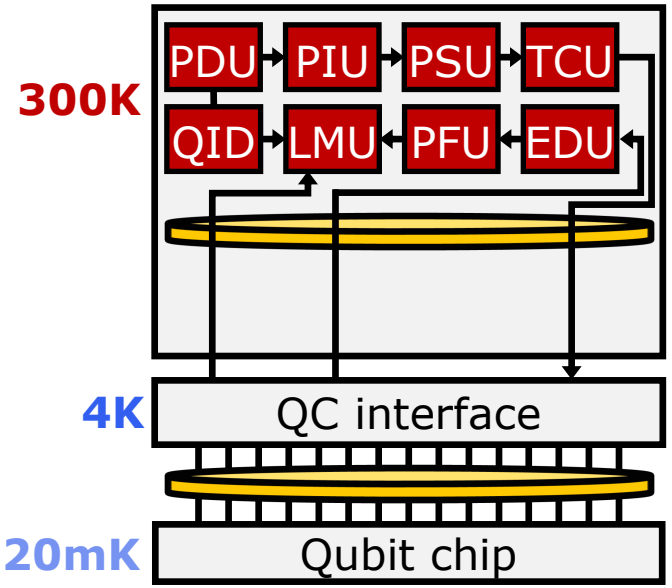
- **Run simulation to report scalability metrics and manageable qubit scale**
- **Integrate a quantum simulator for the functionally correct simulation**



Research goals

Step 1	Develop a modeling & simulation tool for QCP
Step 2	Propose a scalable QCP architecture

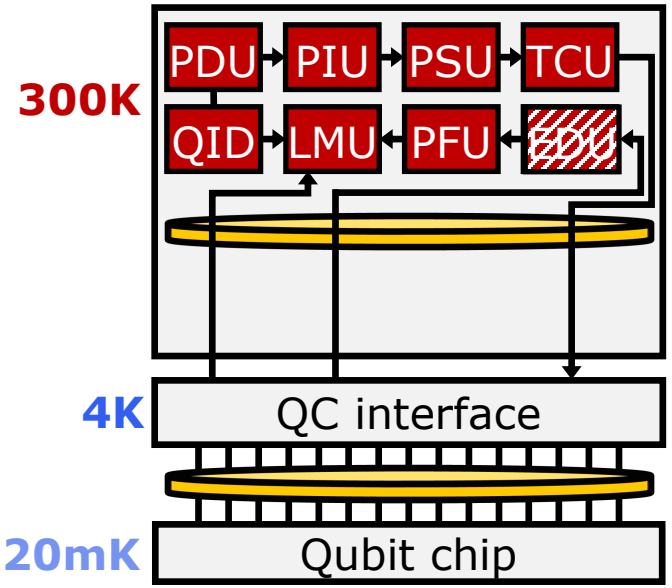
Our 10+K qubit QCP design!



Bottleneck: Slow EDU

Qubit scale: < 250

Our 10+K qubit QCP design!



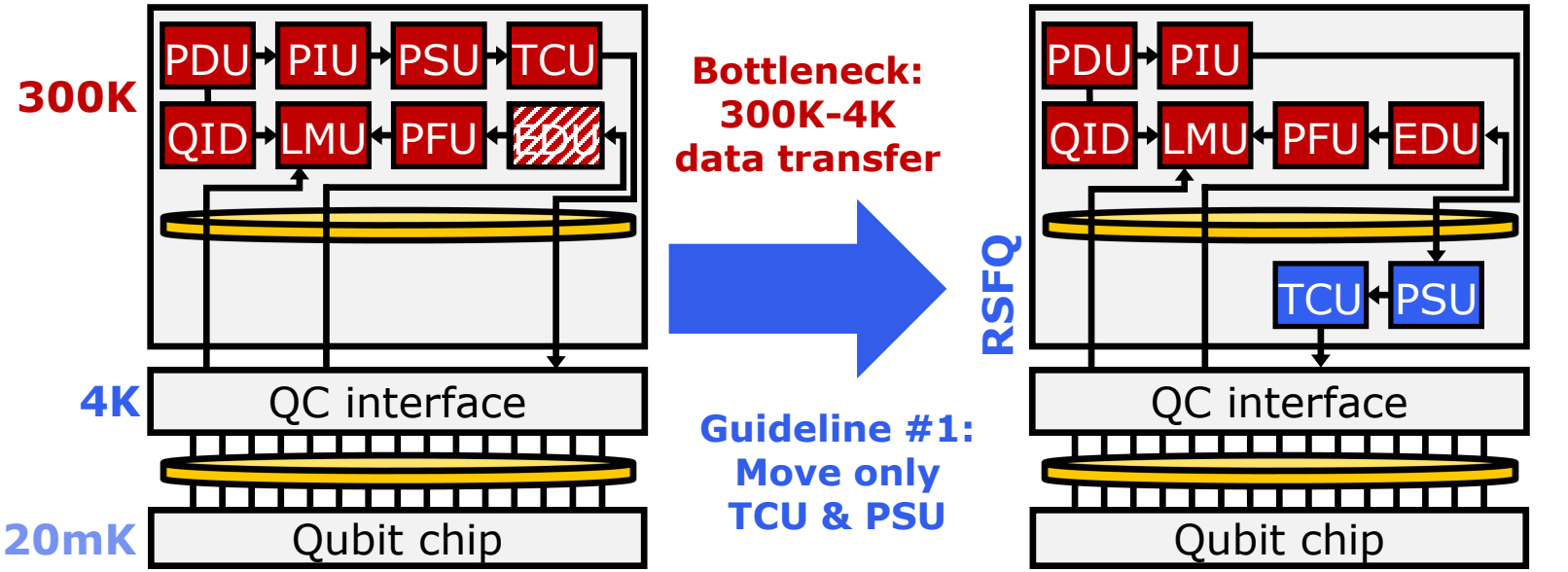
Bottleneck: 300K-4K data transfer

Bottleneck: Slow EDU

μarch #1: Fast EDU

Qubit scale: < 250 → 1,700

Our 10+K qubit QCP design!



Bottleneck: Slow EDU

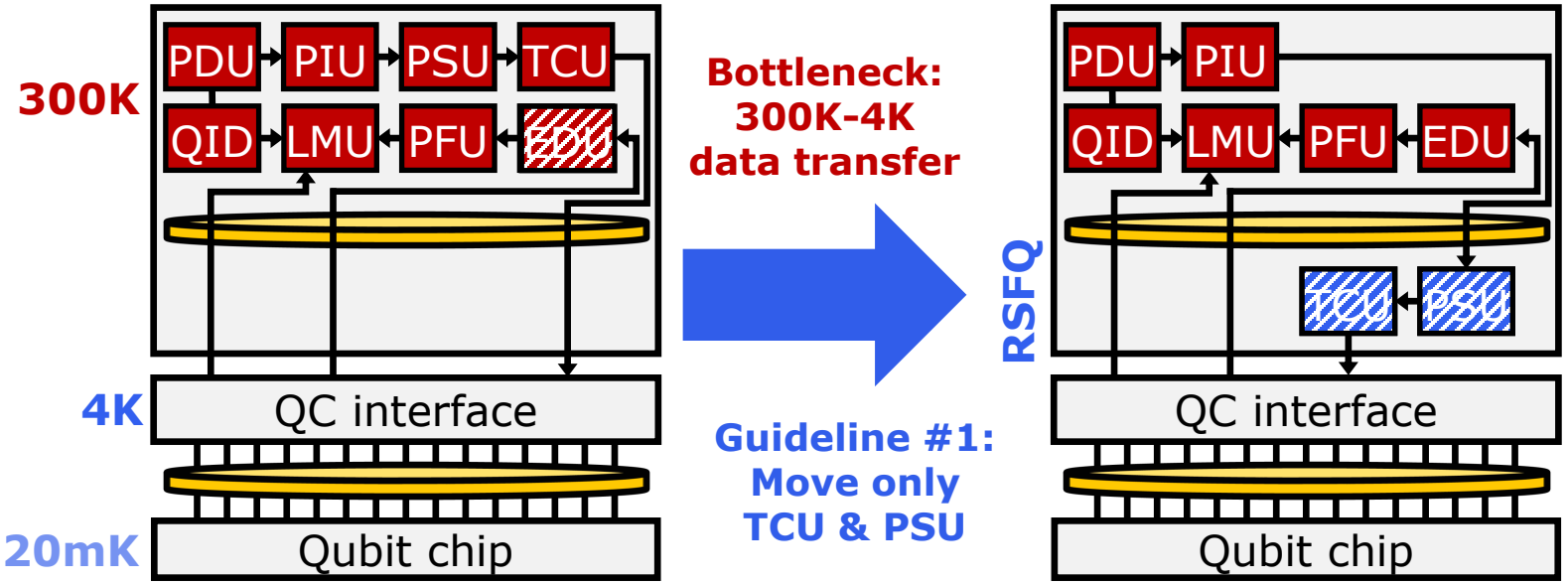
μarch #1: Fast EDU

Qubit scale: < 250 → 1,700

Bottleneck: 4K device power

Qubit scale: 970

Our 10+K qubit QCP design!



Bottleneck: Slow EDU

μarch #1: Fast EDU

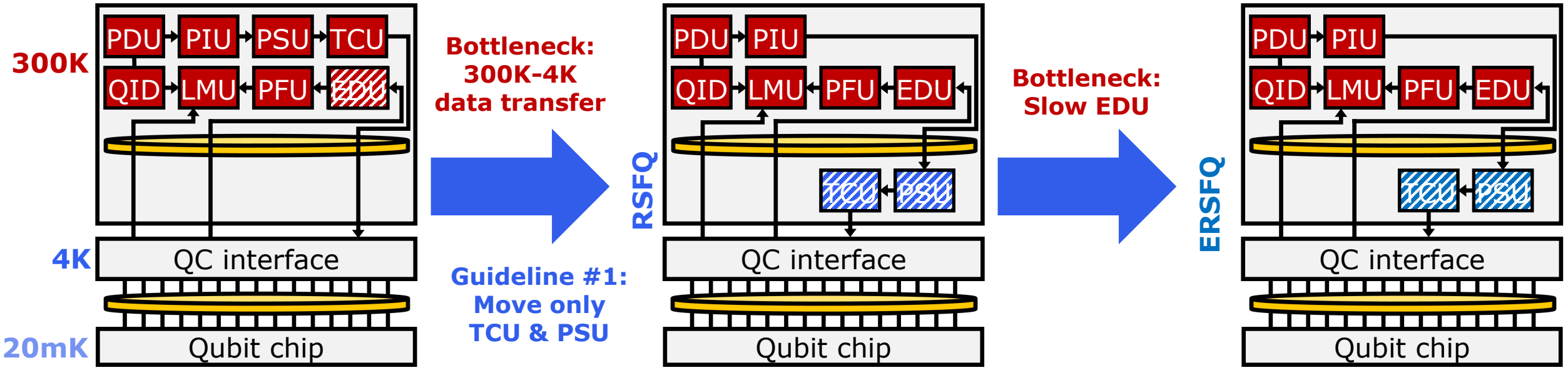
Qubit scale: < 250 → 1,700

Bottleneck: 4K device power

μarch #2 & #3: Low-power PSU & TCU

Qubit scale: 970 → 4,600

Our 10+K qubit QCP design!



Bottleneck: Slow EDU

µarch #1: Fast EDU

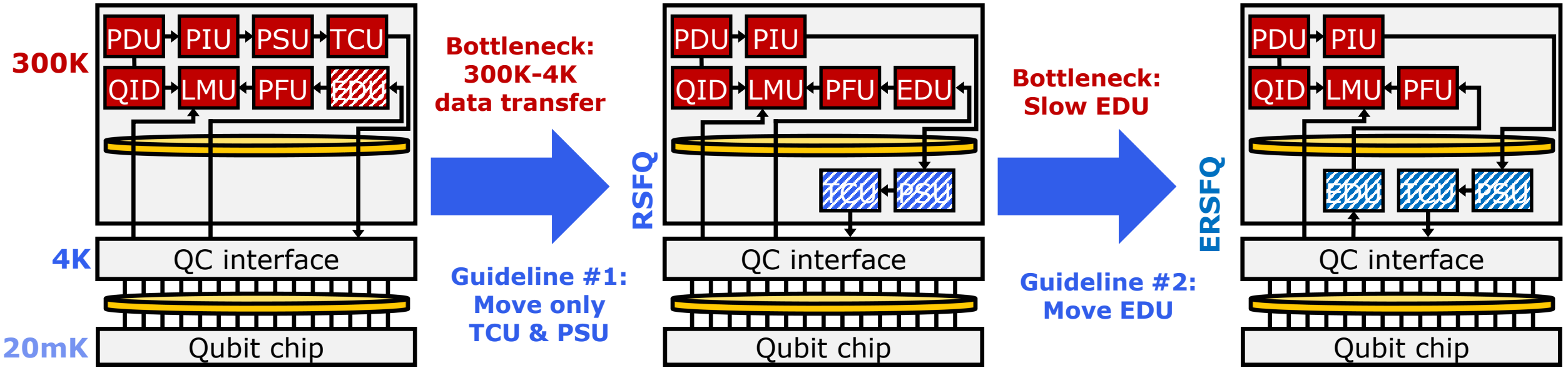
Qubit scale: < 250 → 1,700

Bottleneck: 4K device power

µarch #2 & #3: Low-power PSU & TCU

Qubit scale: 970 → 4,600

Our 10+K qubit QCP design!



Bottleneck: Slow EDU

μarch #1: Fast EDU

Qubit scale: < 250 → 1,700

Bottleneck: 4K device power

μarch #2 & #3: Low-power PSU & TCU

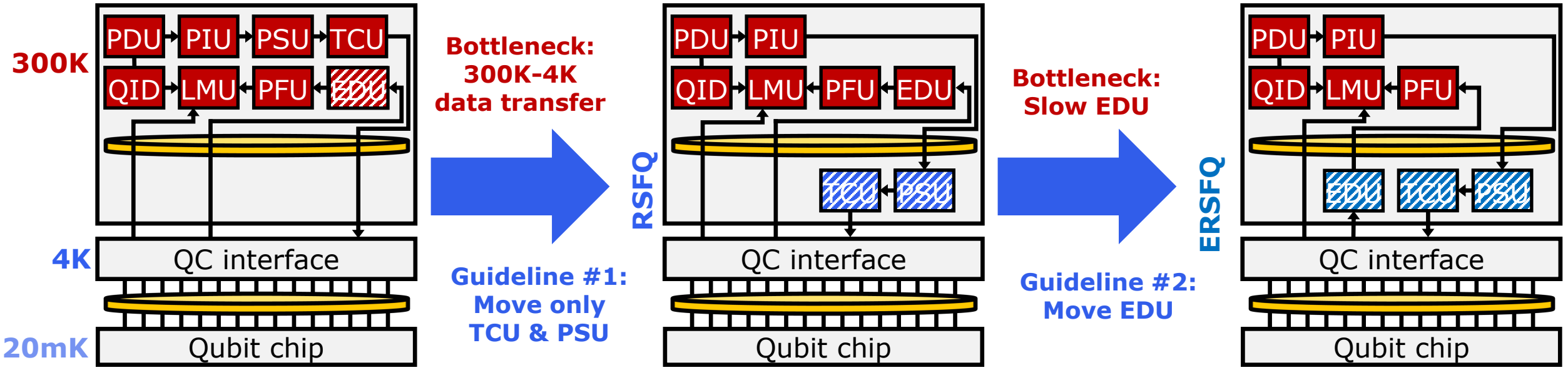
Qubit scale: 970 → 4,600

Bottleneck: 4K device power

μarch #4: Low-power EDU

Qubit scale: 8,100 → 59,000

Our 10+K qubit QCP design!



Bottleneck: Slow EDU

µarch #1: Fast EDU

Qubit scale: < 250 → 1,700

Bottleneck: 4K device power

µarch #2 & #3: Low-power PSU & TCU

Qubit scale: 970 → 4,600

Bottleneck: 4K device power

µarch #4: Low-power EDU

Qubit scale: 8,100 → 59,000

Refer to the paper for more details!

Thanks!

Now the actual tutorials will follow!