CryoModel: Cryogenic CMOS Computer Modeling Tools

Dongmoon Min

E-mail: dongmoon.min@snu.ac.kr
Web: https://hpcs.snu.ac.kr/~dongmoon

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

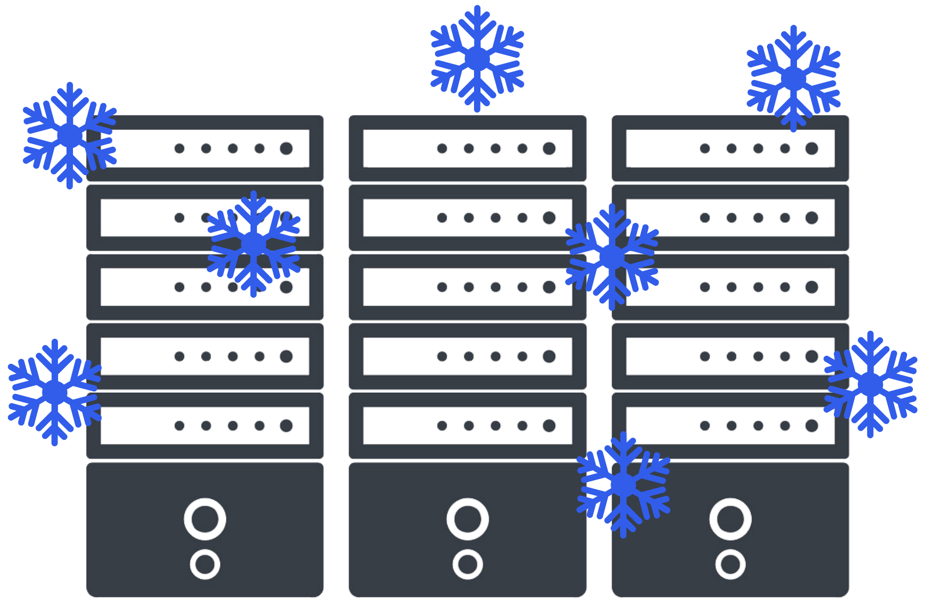
Why cryogenic CMOS computing?

Conventional Computing

Suffer from the power wall and performance wall problems

Why cryogenic CMOS computing?

e.g., 77K, 4K

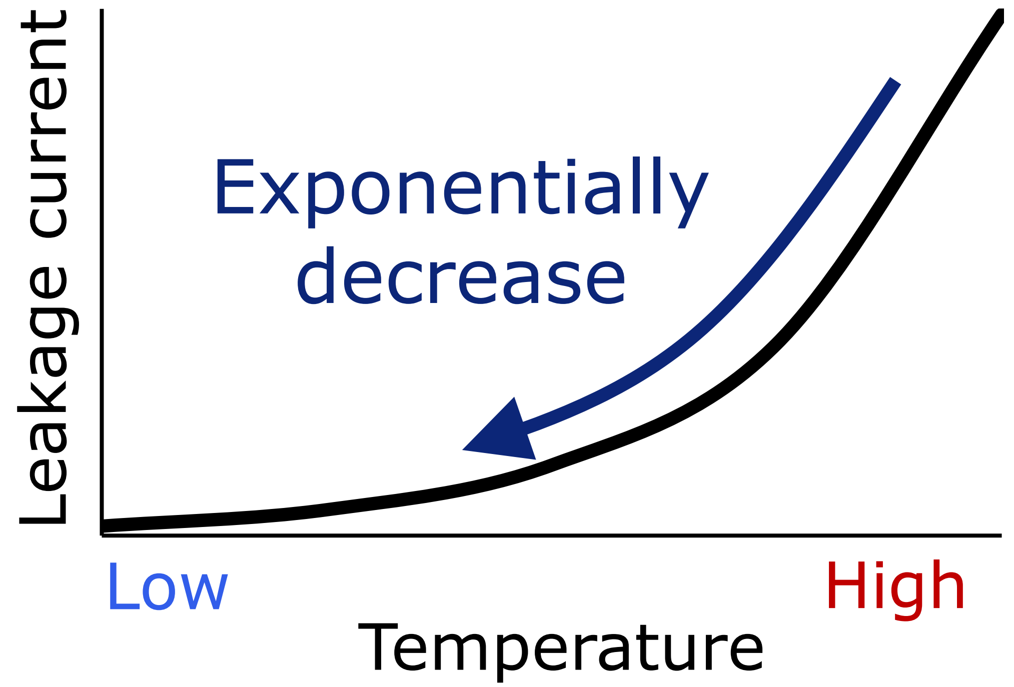


Cryogenic Computing

Resolve the power wall and performance wall problems

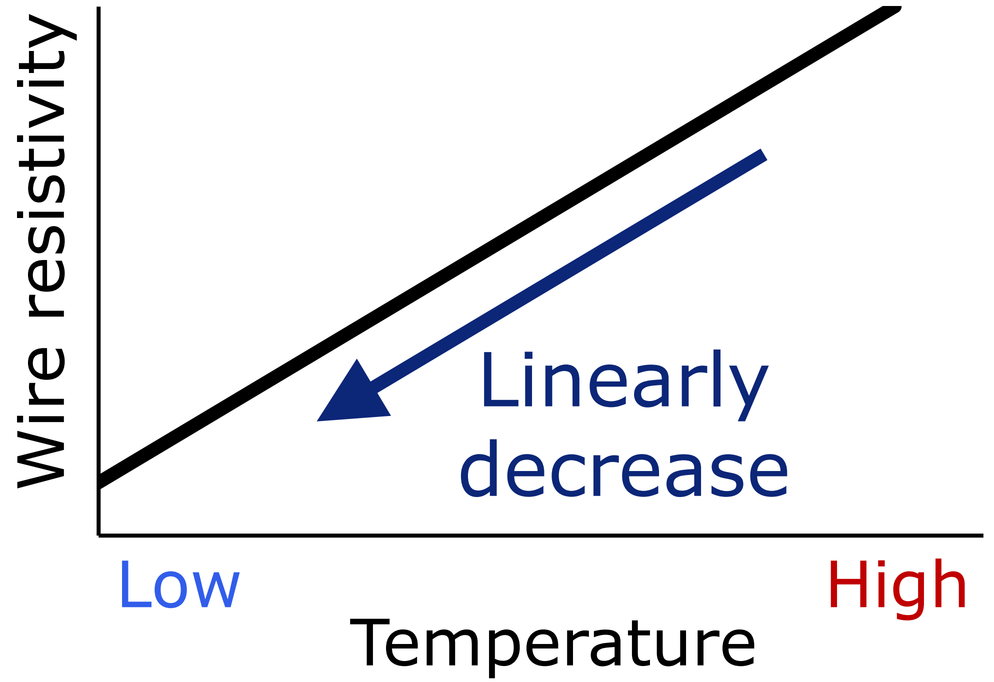
Key benefits of cryogenic CMOS computing

#1: Low leakage current



➡ **Reduce static power**

#2: Low wire resistivity

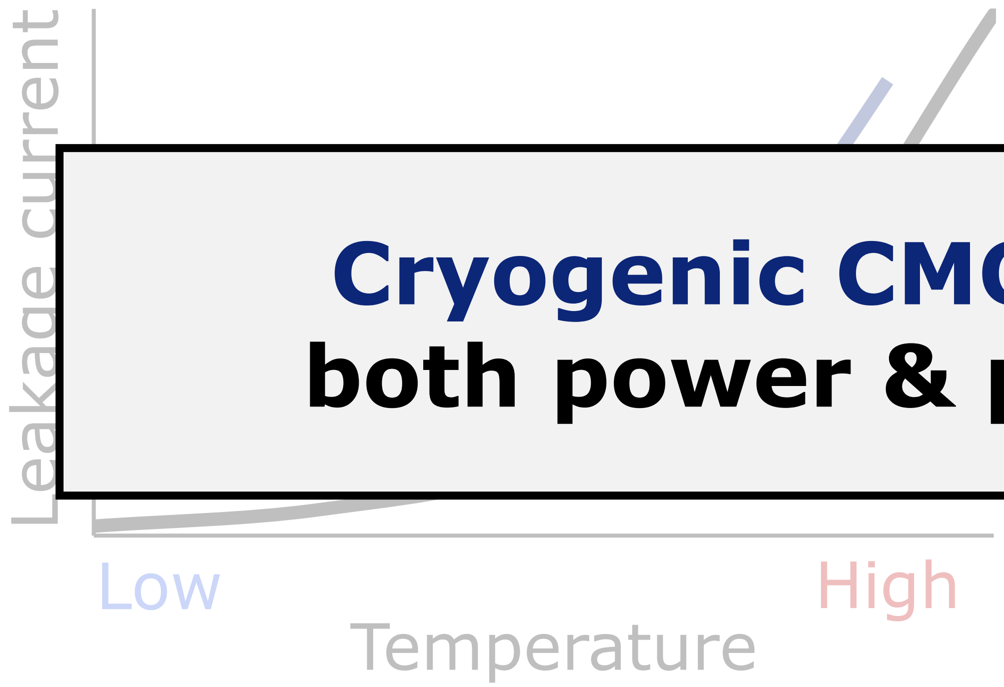


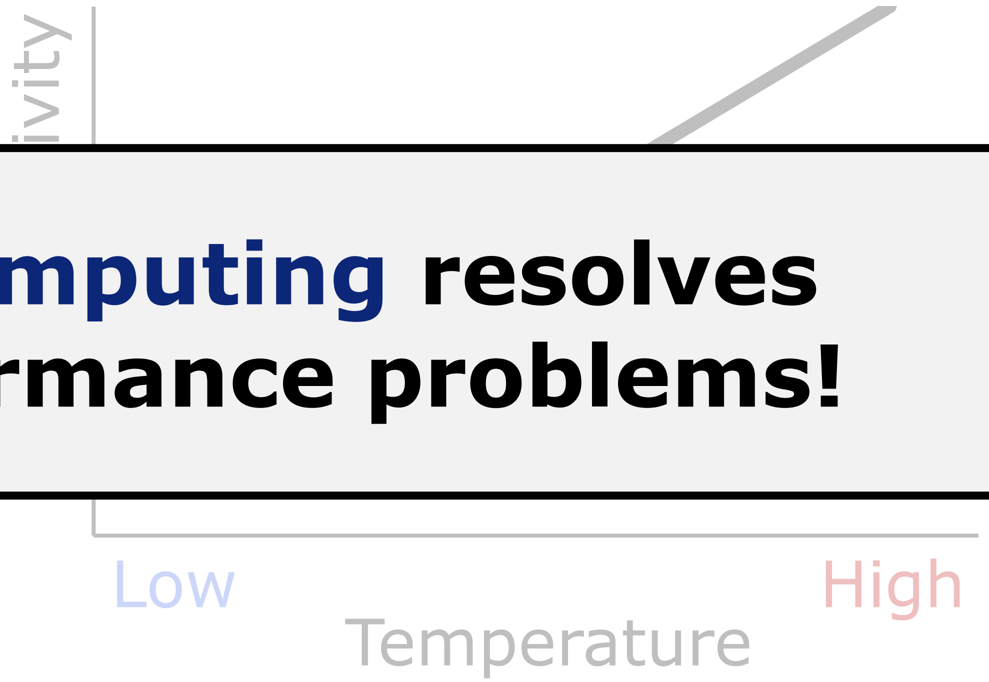
➡ **Reduce transfer latency**

Key benefits of cryogenic CMOS computing

#1: Low leakage current

#2: Low wire resistivity





Cryogenic CMOS computing resolves both power & performance problems!

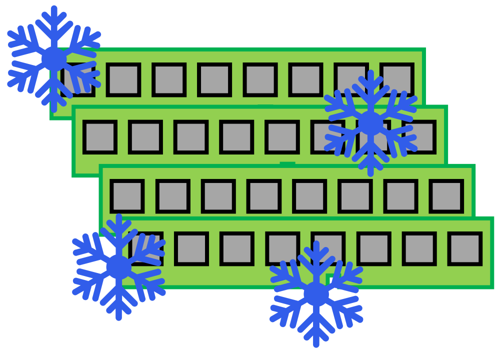
➡ **Reduce static power**

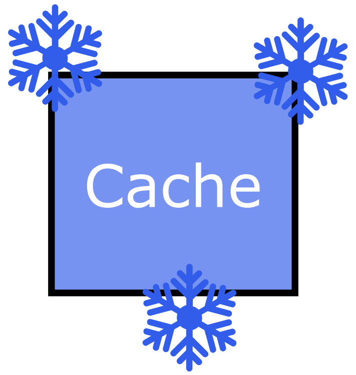
➡ **Reduce transfer latency**

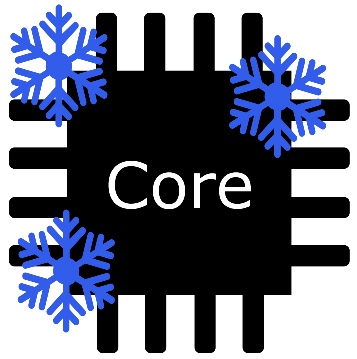
CryoModel overview

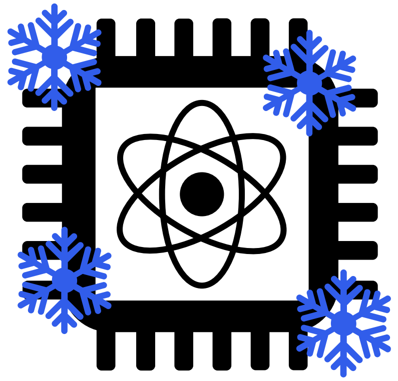
CryoModel covers various cryogenic CMOS-architecture units!

Memory | **Cache** | **Processor** | **Quantum computer**









Cryogenic DRAM
[ISCA’19, ISCA’21]

Cryogenic Cache
[ASPLOS’20]

Cryogenic Core, Network
[ISCA’20, ASPLOS’22]

Quantum controller
[ISCA’22]

77K memory modeling

77K logic modeling

4K memory & logic modeling

CryoModel overview

In this talk, I will cover two sub-models,

(1) 77K/4K CMOS memory modeling

- Case #1: 77K-optimal DRAM [ISCA'19]
- Case #2: 77K-optimal cache architecture [ASPLOS'20]

(2) 77K/4K CMOS logic modeling

- Case #3: Scalability of 4K CMOS QCP [ISCA'22]

77K memory modeling

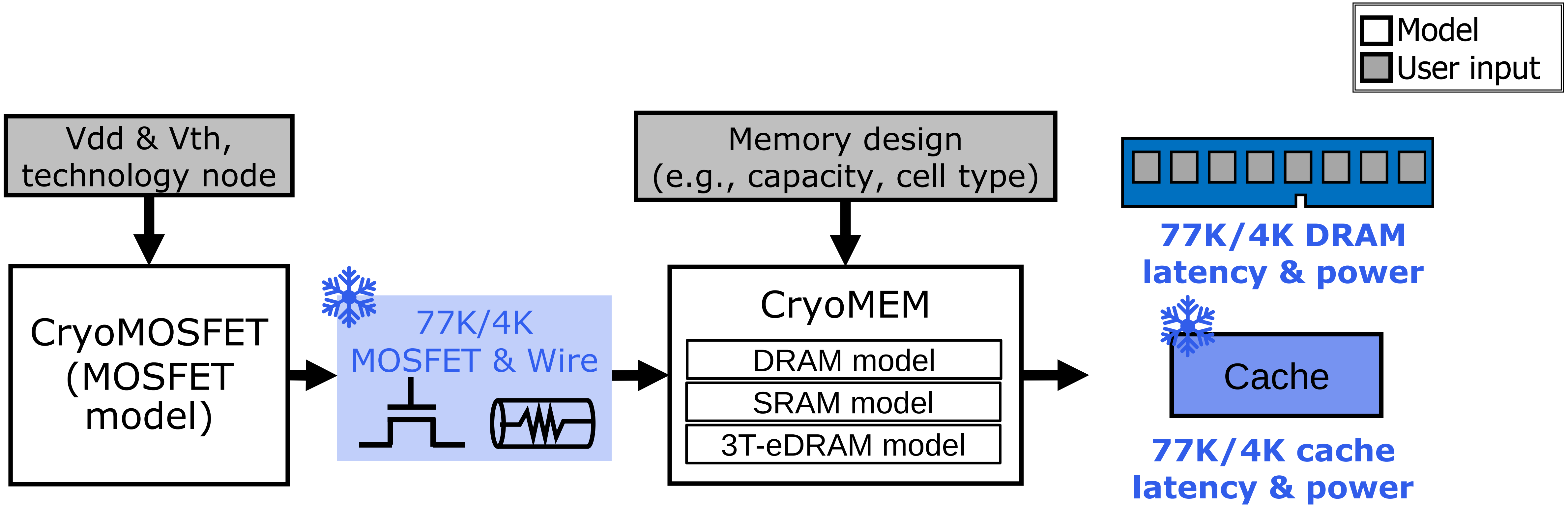
77K logic modeling

4K memory & logic modeling

Index

77K/4K memory modeling overview

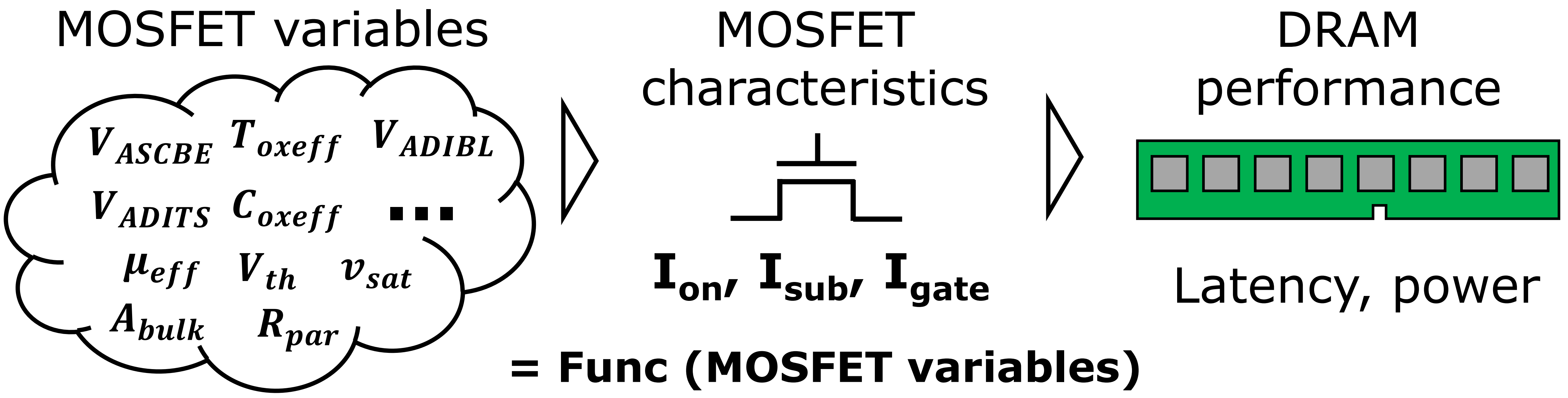
- **Memory model predicts latency and power of 77K/4K memories**
- **Memory model consists of CryoMOSFET and CryoMEM**



☐ Model
▨ User input
Vdd & Vth, technology node
CryoMOSFET (MOSFET model)
77K/4K MOSFET & Wire
Memory design (e.g., capacity, cell type)
CryoMEM
DRAM model
SRAM model
3T-eDRAM model
77K/4K DRAM latency & power
Cache
77K/4K cache latency & power

77K/4K MOSFET modeling (1/2)

- **CryoMOSFET predicts low-temperature I_{on}, I_{sub}, I_{gate}, R_{wire} by modeling three temperature sensitive variables**

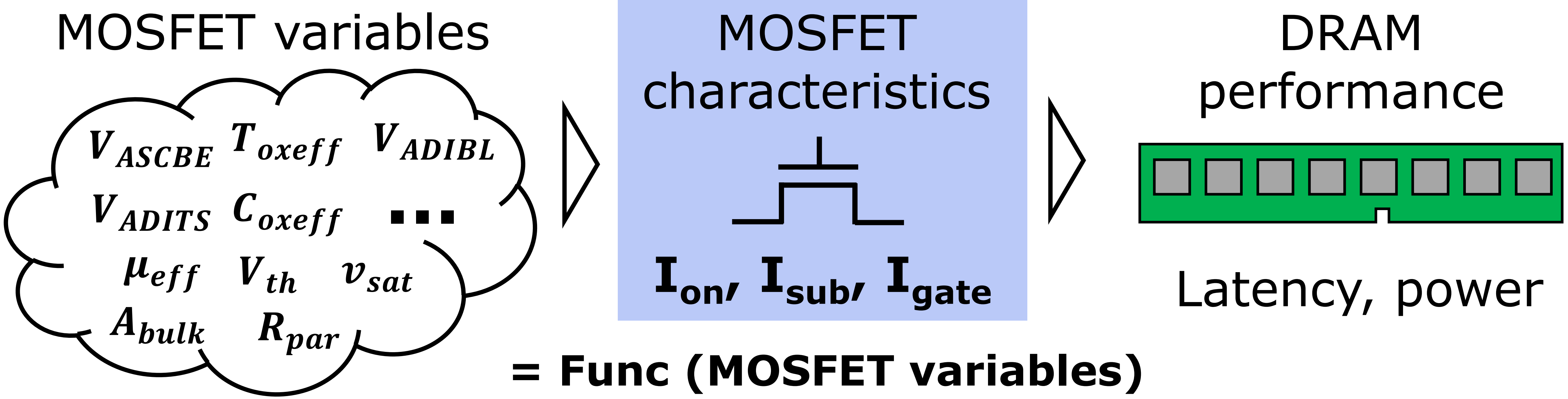


MOSFET variables
V_{ASCBE} T_{oxeff} V_{ADIBL}
V_{ADITS} C_{oxeff} ...
μ_{eff} V_{th} v_{sat}
A_{bulk} R_{par}
MOSFET characteristics
I_{on}, I_{sub}, I_{gate}
= Func (MOSFET variables)
DRAM performance
Latency, power

77K/4K MOSFET modeling (1/2)

- **CryoMOSFET predicts low-temperature I_{on}, I_{sub}, I_{gate}, R_{wire} by modeling three temperature sensitive variables**

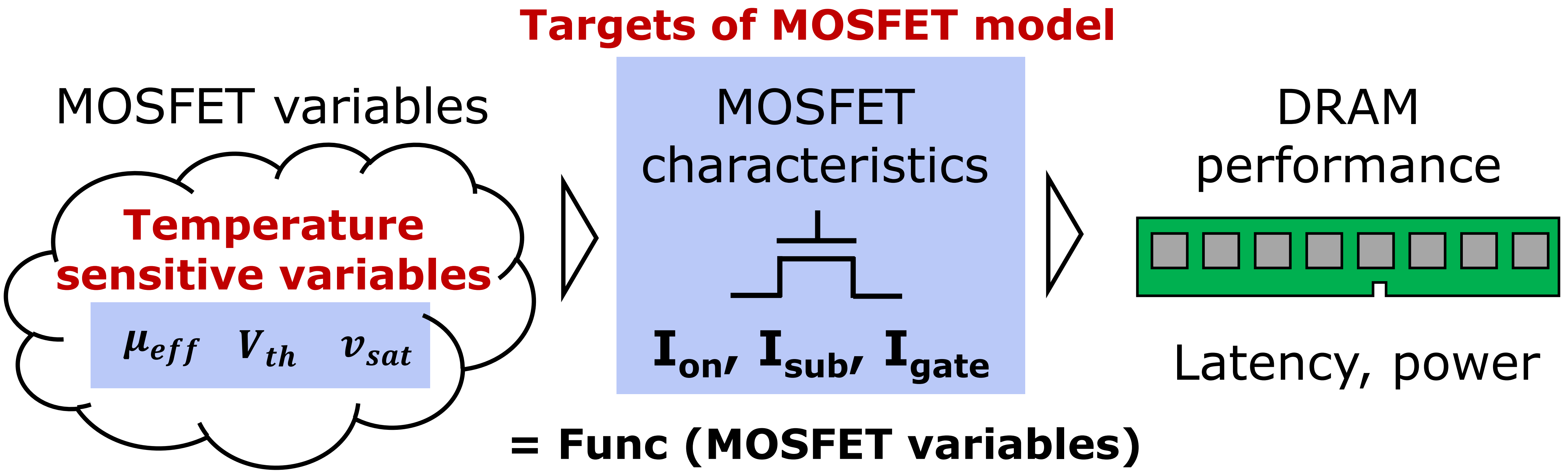
Targets of MOSFET model


MOSFET variables
V_{ASCBE} T_{oxeff} V_{ADIBL}
V_{ADITS} C_{oxeff} ...
μ_{eff} V_{th} v_{sat}
A_{bulk} R_{par}
MOSFET characteristics
I_{on}, I_{sub}, I_{gate}
DRAM performance
Latency, power

= Func (MOSFET variables)

77K/4K MOSFET modeling (1/2)

- **CryoMOSFET predicts low-temperature I_{on}, I_{sub}, I_{gate}, R_{wire} by modeling three temperature sensitive variables**


Targets of MOSFET model
MOSFET variables
Temperature sensitive variables
μ_{eff} V_{th} v_{sat}
MOSFET characteristics
I_{on}, I_{sub}, I_{gate}
DRAM performance
Latency, power
= Func (MOSFET variables)

77K/4K MOSFET modeling (2/2)

Target:
on-current (I_{on})
at 77K

77K/4K MOSFET modeling (2/2)

① Obtain 300K MOSFET variables

Target: on-current (I_{on}) at 77K

①$u_{eff,300K}$

$v_{sat,300K}$

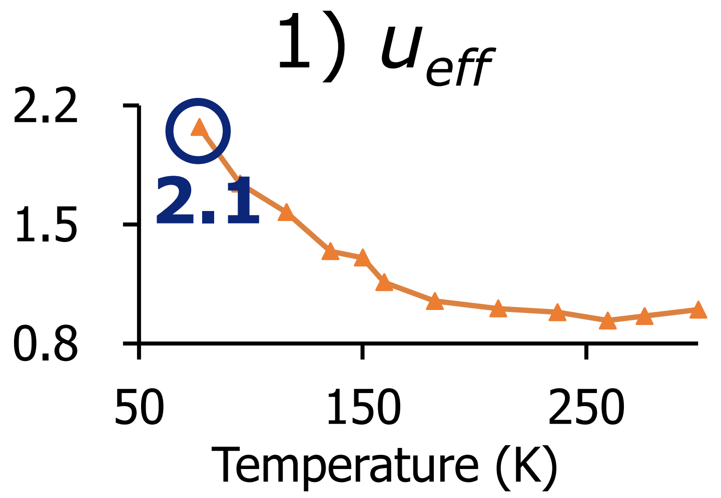
$V_{th,300K}$

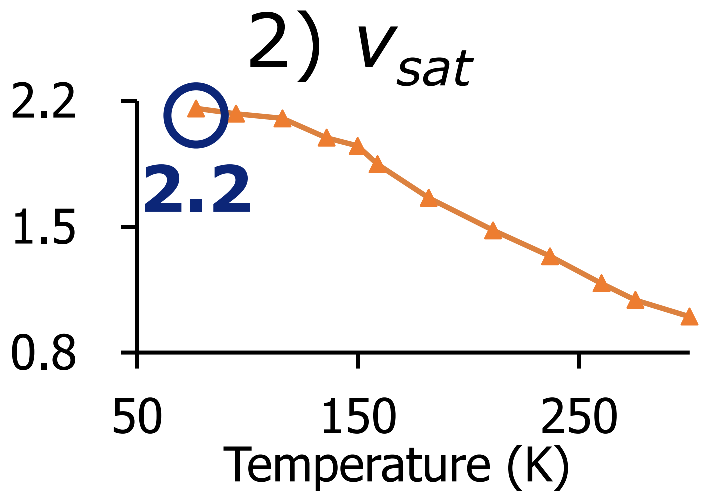
77K/4K MOSFET modeling (2/2)

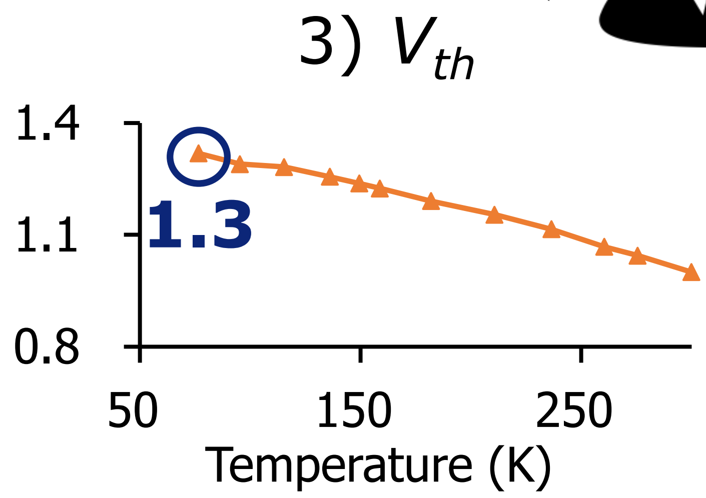
① Obtain 300K MOSFET variables

② Convert 300K variables to 77K variables

Target: on-current (I_{on}) at 77K







② $u_{eff,77K} = u_{eff,300K} \times 2.1$ $v_{sat,77K} = v_{sat,300K} \times 2.2$ $V_{th,77K} = V_{th,300K} \times 1.3$

77K/4K MOSFET modeling (2/2)

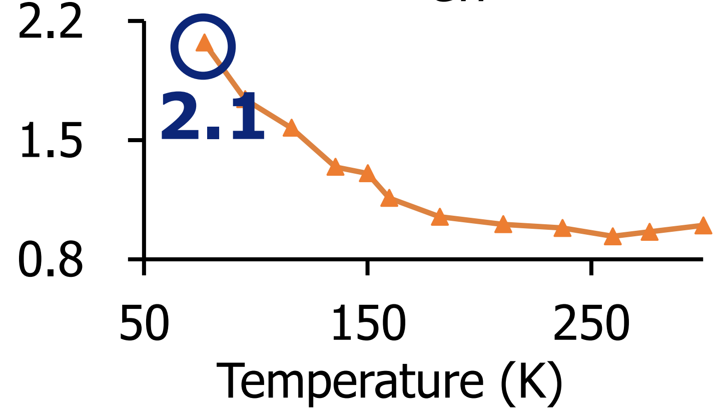
① Obtain 300K MOSFET variables

② Convert 300K variables to 77K variables

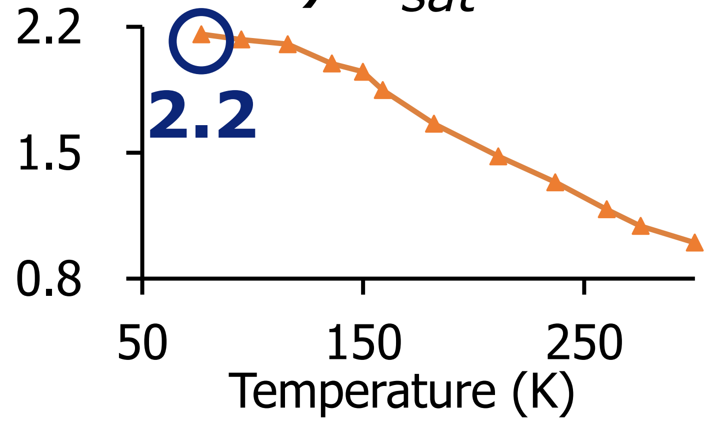
③ Calculate MOSFET characteristics at 77K

Target: on-current (I_{on}) at 77K

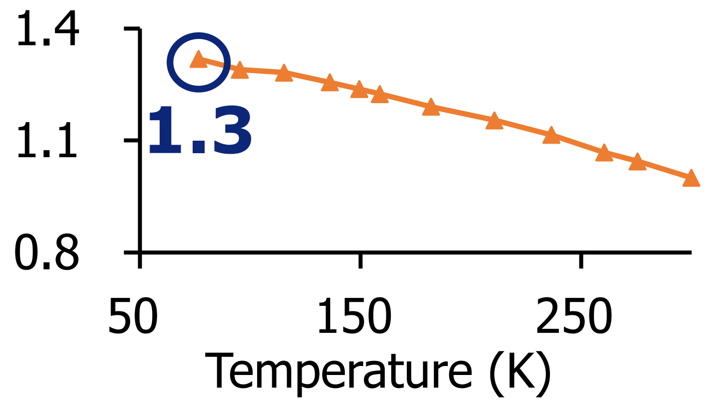
1) u_{eff}



2) v_{sat}



3) V_{th}

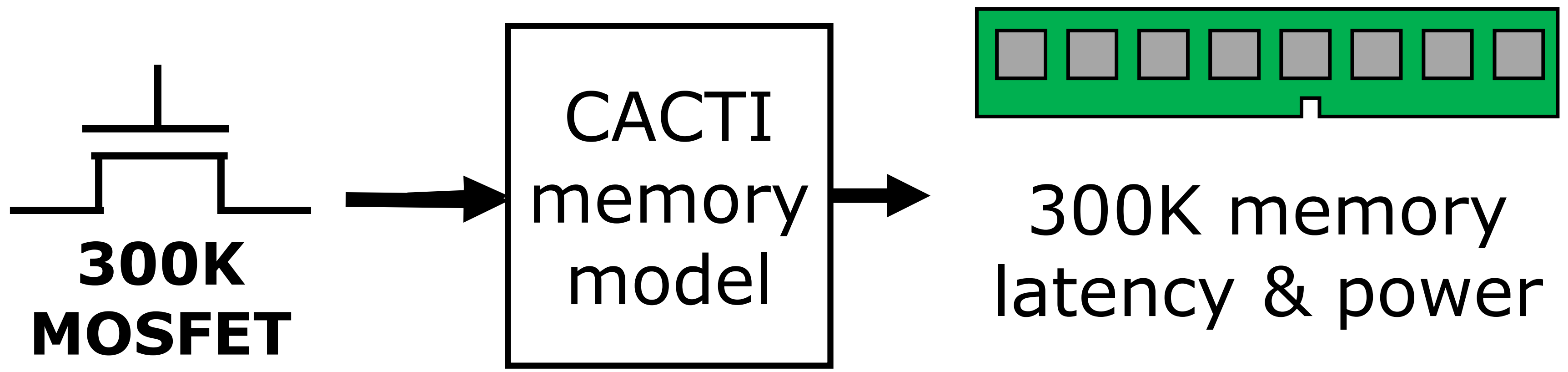


$u_{eff,77K} = u_{eff,300K} \times 2.1$ $v_{sat,77K} = v_{sat,300K} \times 2.2$ $V_{th,77K} = V_{th,300K} \times 1.3$

③ $I_{on} = F_1(u_{eff}, v_{sat}, V_{th}, \ldots) \triangleright I_{on,\ 77K}$

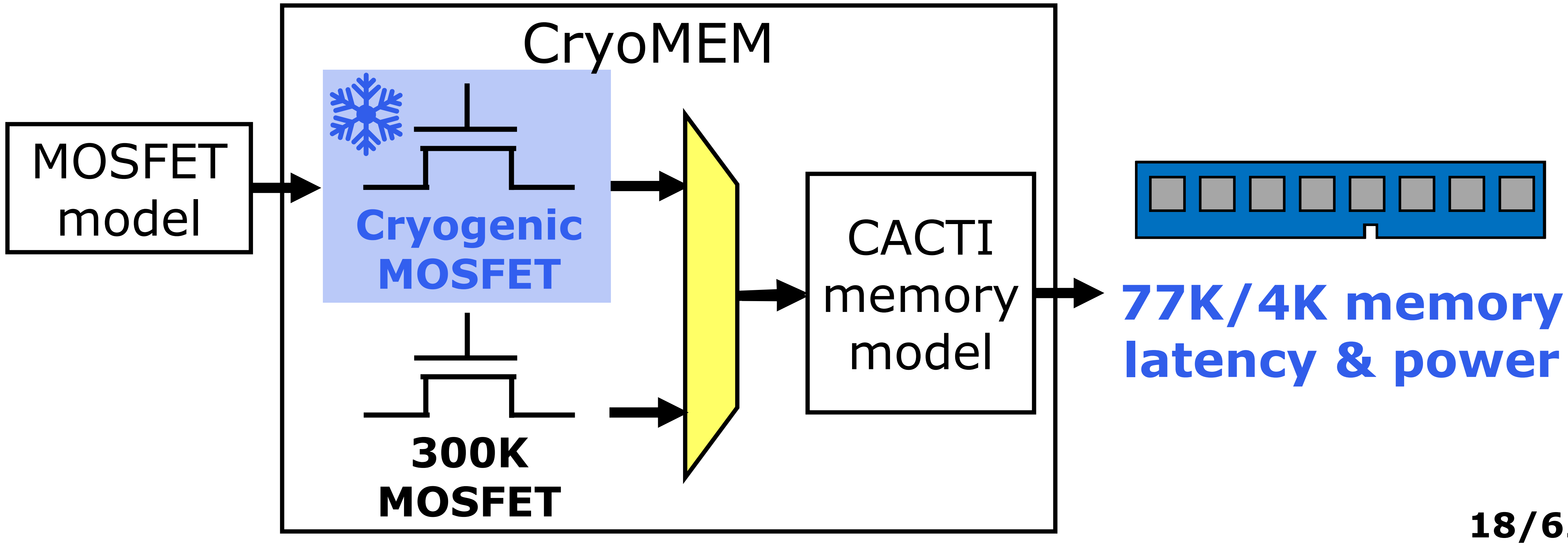
77K/4K memory modeling

- **By using MOSFET characteristics from CryoMOSFET, we can predict latency and power of 77K/4K memories**



77K/4K memory modeling

- **By using MOSFET characteristics from CryoMOSFET, we can predict latency and power of 77K/4K memories**



Installing the pre-required packages

```
> sudo apt install python3.8-dev python3-pip \
  libprotobuf-dev protobuf-compiler libboost1.65-all-dev

> python3.8 –m pip install protobuf cython numpy
```

We believe you already installed the packages.

Downloading/building the memory model

```
# Downloading CryoModel
> git clone https://github.com/SNU-HPCS/CryoModel.git
> cd CryoModel/CryoMEM
```

How to run memory model? (1/3)

```
> ./memory_model.py {config_file} {temperature} {node} \
  {vdd} {vth0} {capacity} {memory-type} {vdd_wl} {vth0_wl}
```

- **{config_file}**
  - Memory model supports **SRAM, 3T-eDRAM, and DRAM**
  - User should edit the config file to change detailed memory config (e.g., bank count, associativity, IO width)
- **{temperature}**
  - We recommend to use **300K, 77K, and 4K** (Parameters at other temperatures are linearly fitted)

How to run memory model? (2/3)

```
> ./memory_model.py {config_file} {temperature} {node} \
  {vdd} {vth0} {capacity} {memory-type} {vdd_wl} {vth0_wl}
```

- **`{node}`**
  - We recommend to use the technology **larger than 22nm** (our MOSFET model supports the planar MOSFET only)
- **`{vdd}`**
  - Vdd should be **higher than Vth0** (our MOSFET model supports the saturation region only)
- **`{vth0}`**
  - **Threshold voltages of both on/off states should be higher than 0V** (CryoMOSFET reports error!)

How to run memory model? (3/3)

```
> ./memory_model.py {config_file} {temperature} {node} \
  {vdd} {vth0} {capacity} {memory-type} {vdd_wl} {vth0_wl}
```

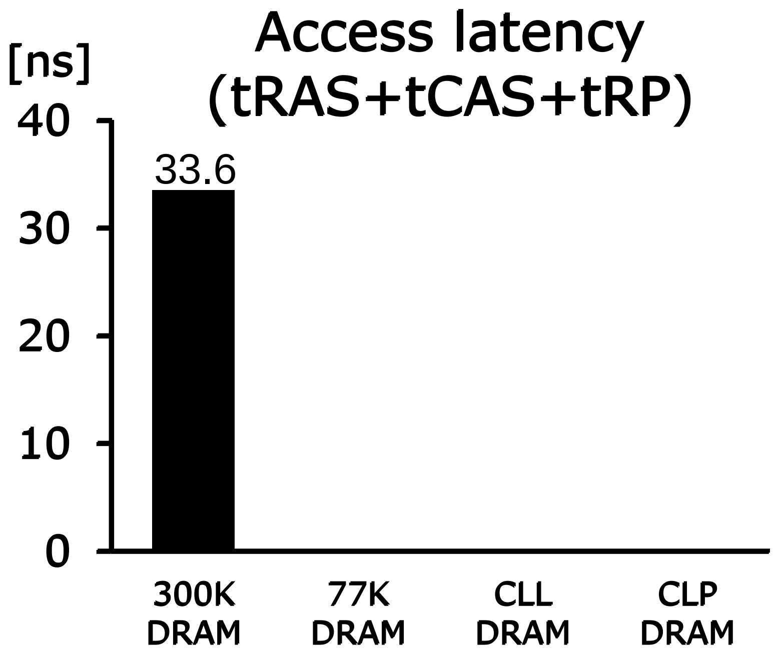
- **`{vdd_wl}`**
  - Source voltage of the DRAM wordline transistor
  - `vdd_wl` can be lower than `vth0_wl`
    (“vdd_wl+vth0_wl" is applied to the gate and it guarantees the saturation region)
- **`{vth0_wl}`**
  - Threshold voltage of the DRAM wordline transistor
  - **Threshold voltages of both on/off states should be higher than 0V**
    (CryoMOSFET will report error!)

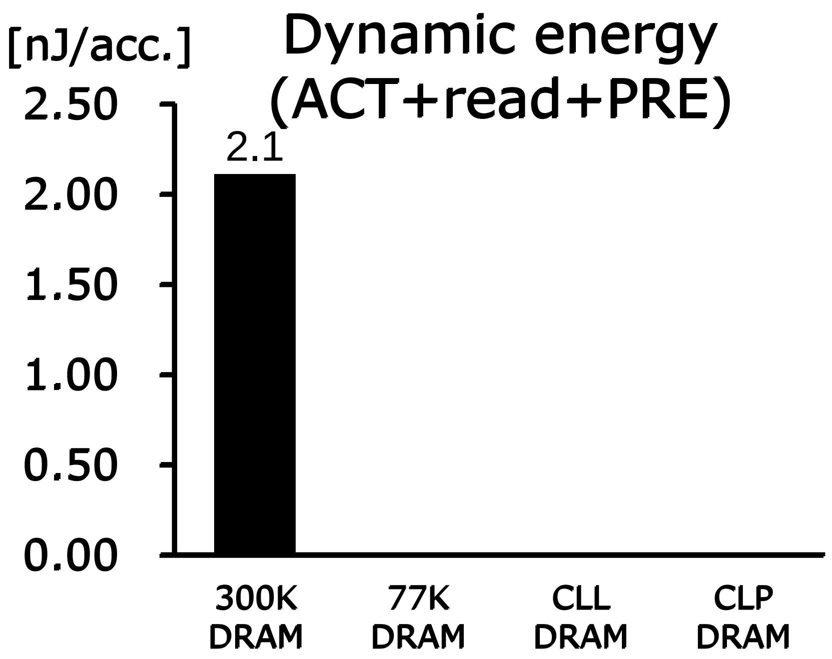
Case study #1.

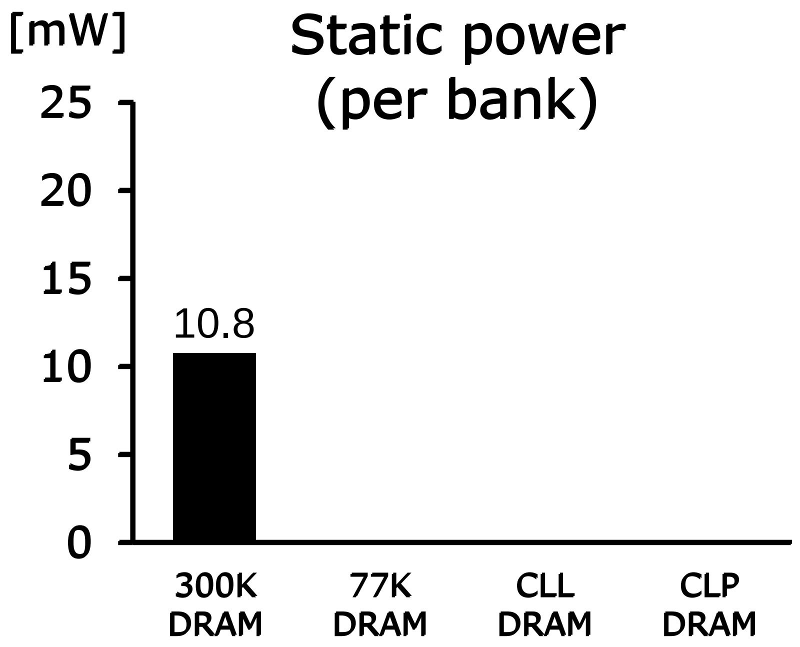
Let’s show the maximum performance gain and power reduction of 77K DRAM! [ISCA’19]

Evaluating the 77K DRAM memory (1/5)

```
# 300K 8Gb DRAM device (baseline)
> ./memory_model.py ./configs/DRAM.cfg 300 32 \
  1.0 0.4 8 dram 1.0 1.0
```

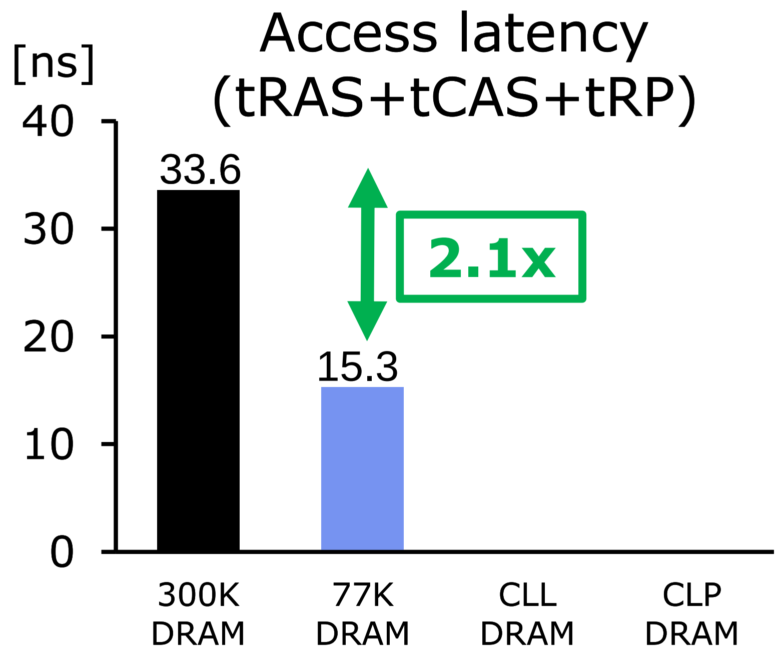


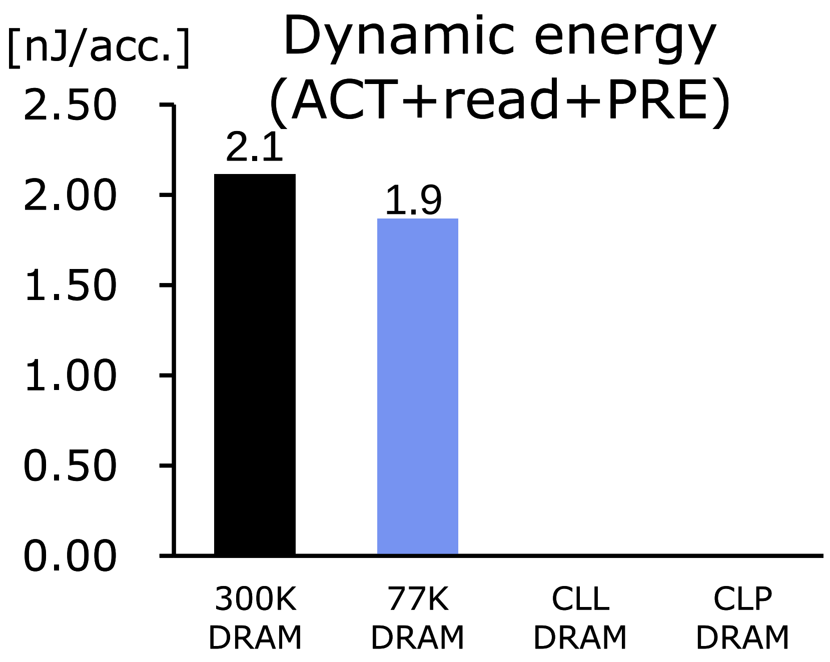


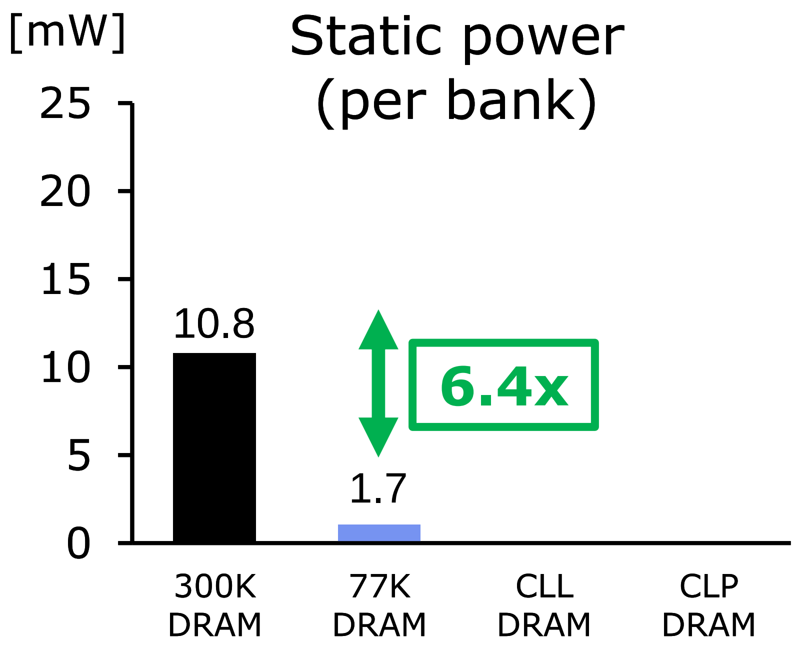


Evaluating the 77K DRAM memory (2/5)

```
# 77K 8Gb DRAM device
> ./memory_model.py ./configs/DRAM.cfg 77 32 \
  1.0 0.4 8 dram 1.0 1.0
```



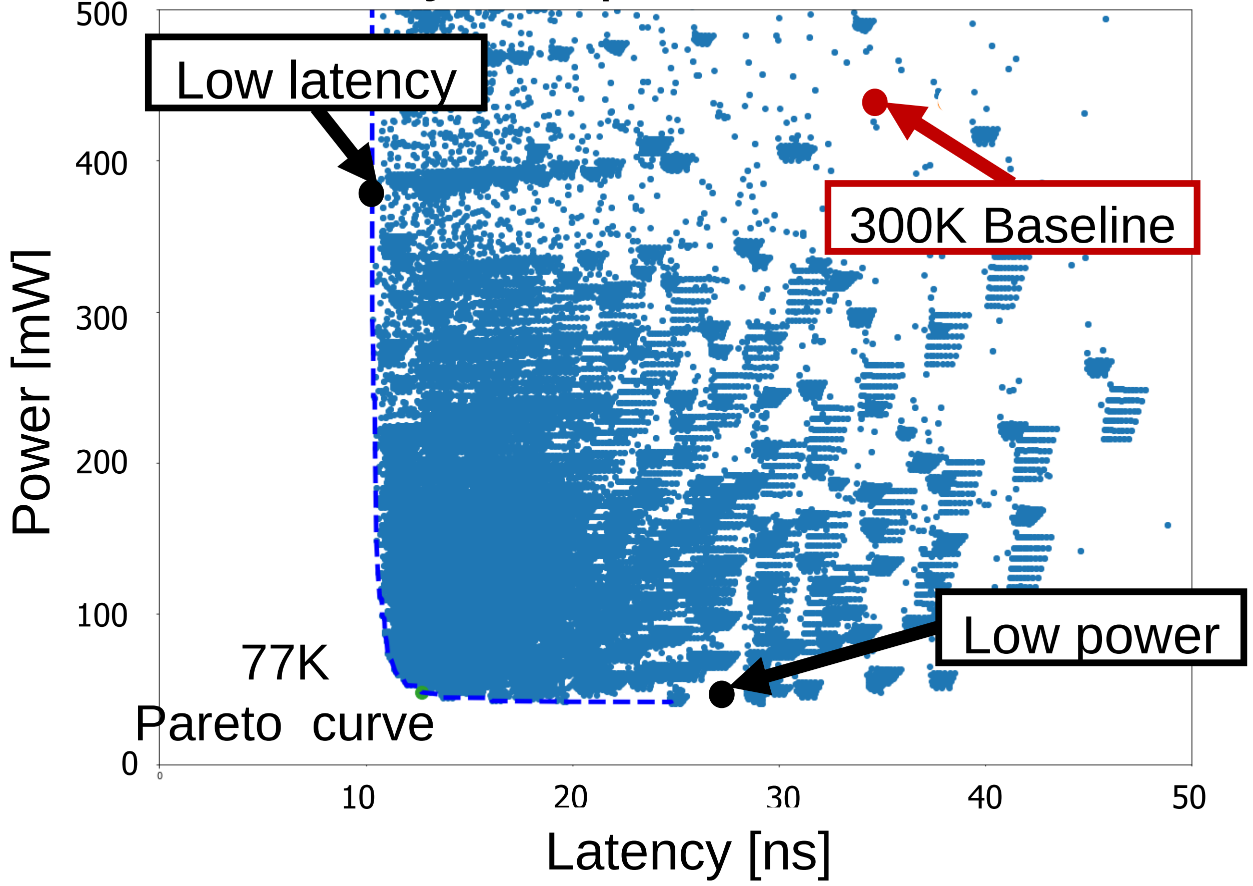




Evaluating the 77K DRAM memory (3/5)

- **With CryoModel, we find the perf.- or power-optimal Vdd & Vth**

Latency and power of DRAMs

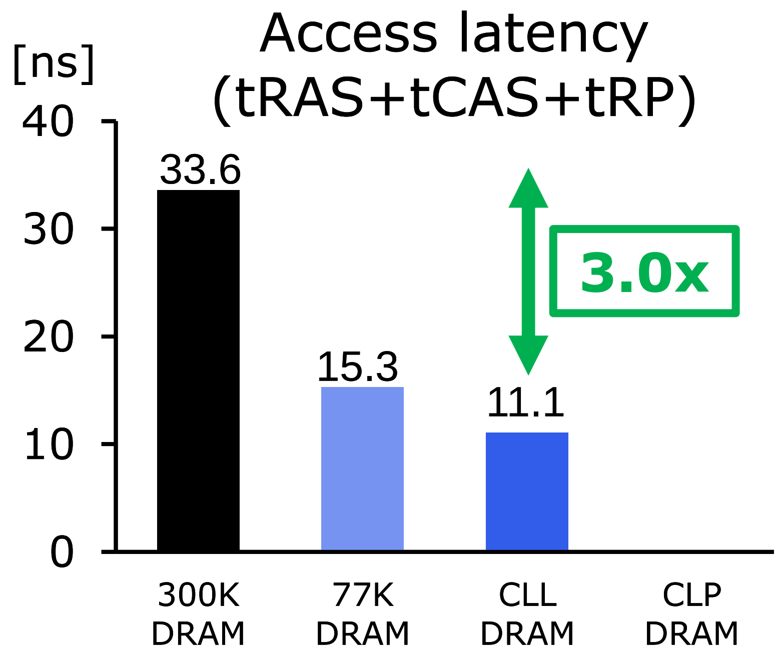


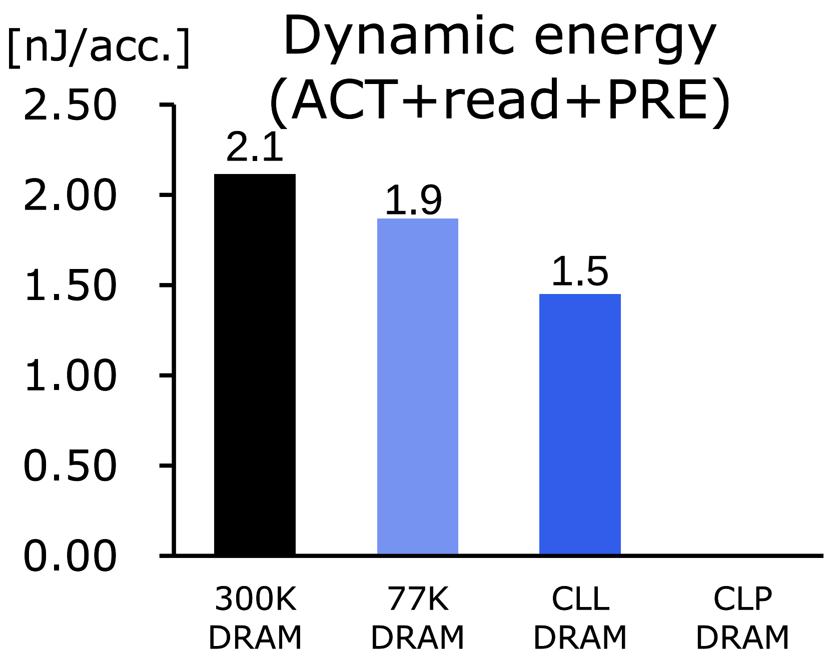
- **Power-optimized DRAM**
  - **Lowest power** for same performance
  - Maximum power-reduction potential

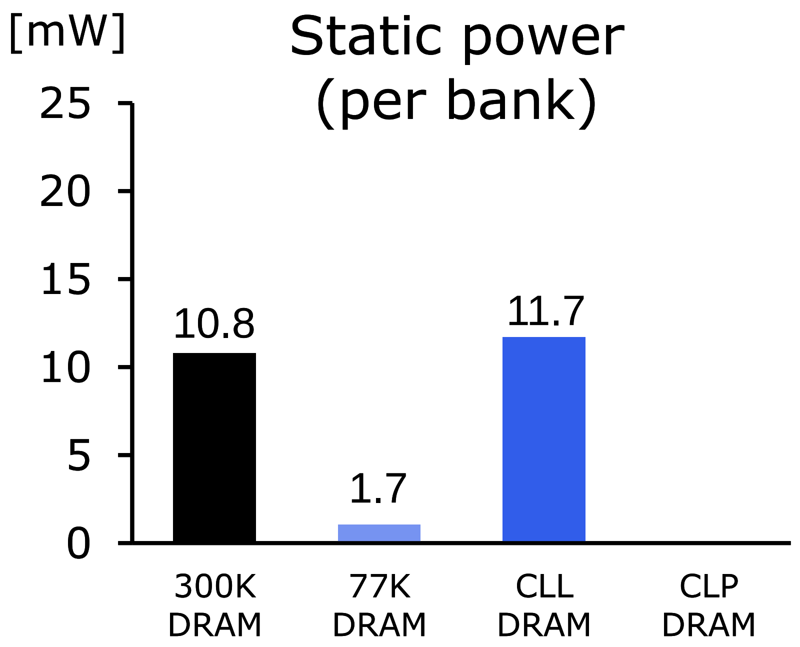
- **Performance-optimized DRAM**
  - **Highest performance** for same power
  - Maximum performance potential

Evaluating the 77K DRAM memory (4/5)

```
# 77K 8Gb DRAM device with voltage scaling
> ./memory_model.py ./configs/DRAM.cfg 77 32 \
  1.05 0.275 8 dram 0.5 0.35
```

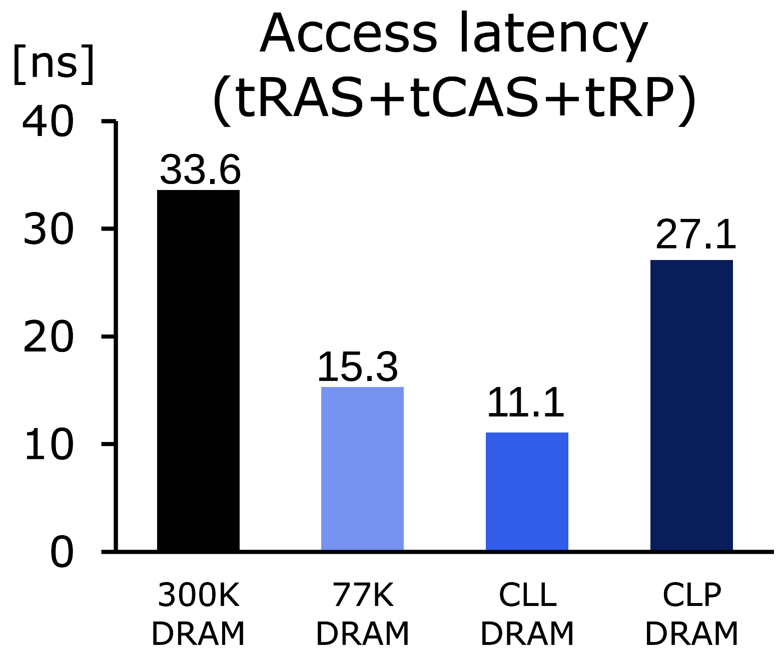


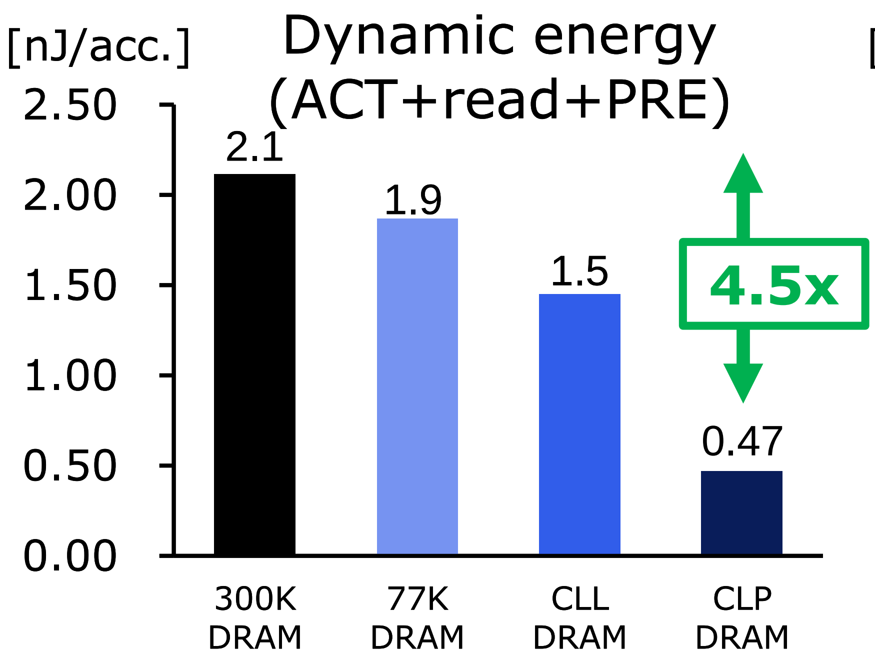


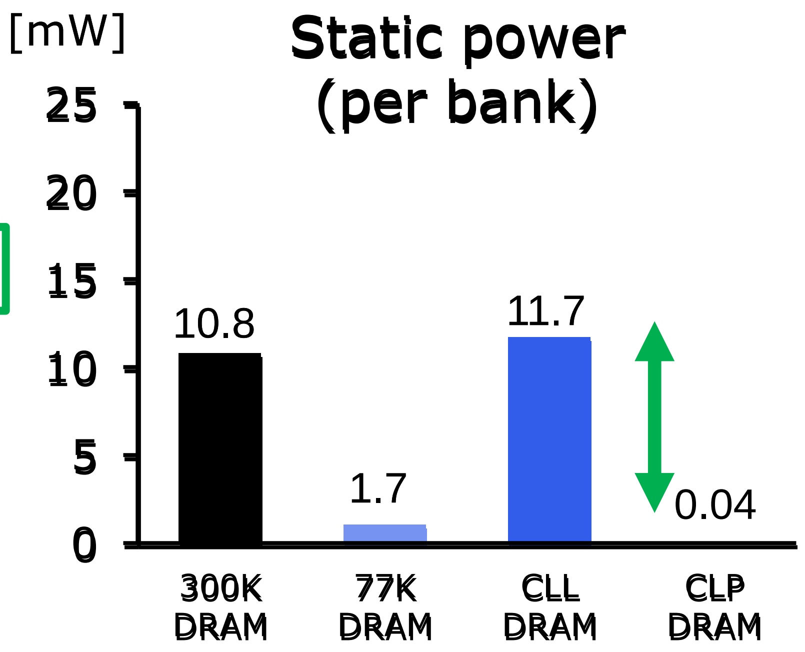


Evaluating the 77K DRAM memory (5/5)

```
# 77K 8Gb DRAM device with voltage scaling
> ./memory_model.py ./configs/DRAM.cfg 77 32 \
  0.25 0.2 8 dram 0.35 0.35
```







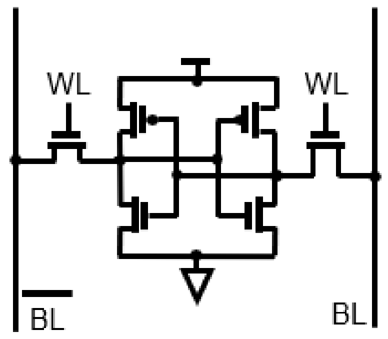
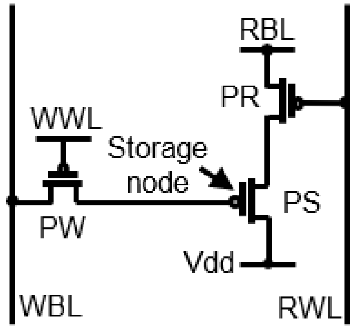
Case study #2.

Let’s architect the 77K-optimal L1, L2, and L3 caches with various cell technologies! [ASPLOS’20]

Optimal cell technology at 77K

SRAM and 3T-eDRAM is a promising cell technology at 77K

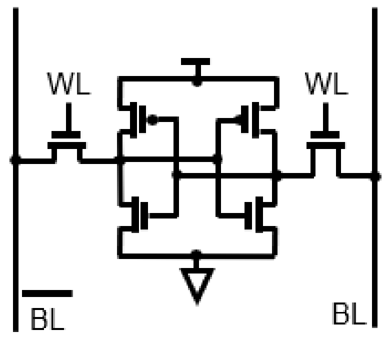
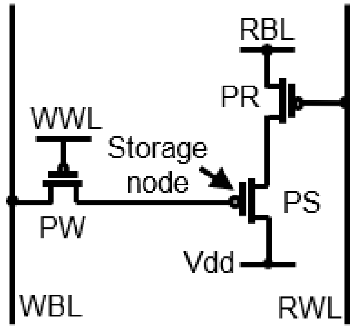
300K

	SRAM	3T-eDRAM
Schematic		 RBL PR WWL Storage node PS PW Vdd WBL RWL
Density	1	**2.13**
Speed	**Fastest**	**Fast**
Power	-	**Low static power**
Refresh overhead	-	Huge refresh overhead

Optimal cell technology at 77K

SRAM and 3T-eDRAM is a promising cell technology at 77K

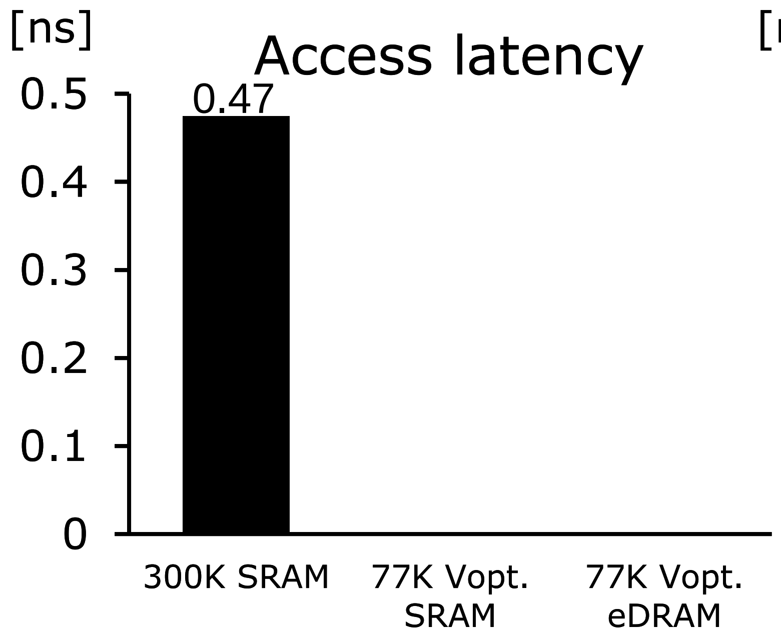
77K

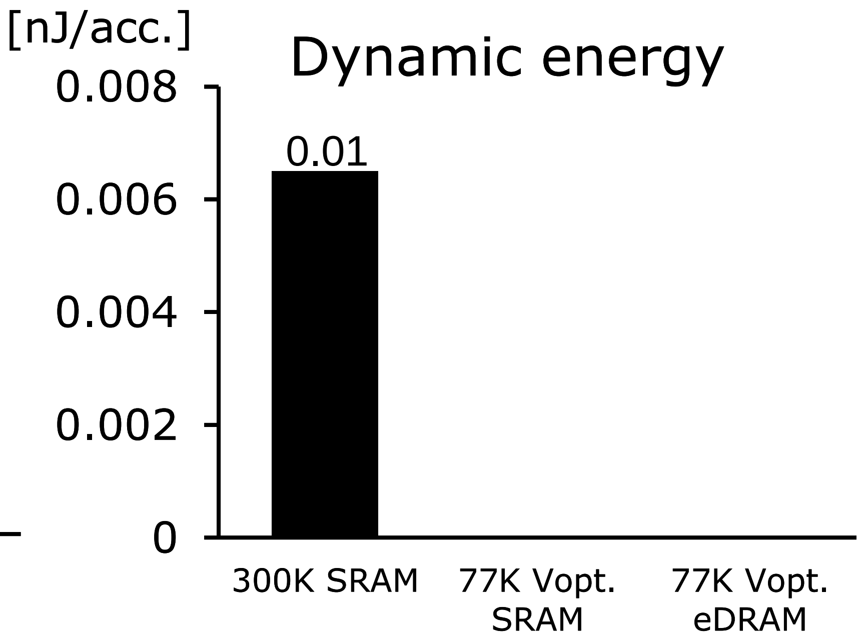
	❄ SRAM	❄ 3T-eDRAM
Schematic		 RBL PR WWL Storage node PS PW Vdd WBL RWL
Density	1	**2.13**
Speed	**Fastest**	**Fast**
Power	-	**Low static power**
Refresh overhead	-	**Negligible refresh overhead**

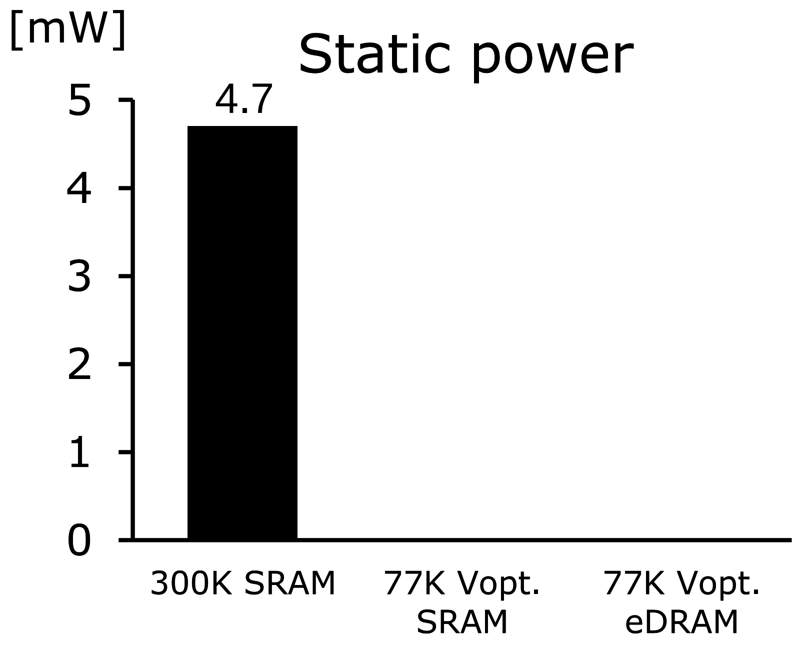
SRAM & 3T-eDRAM for L1 caches (1/3)

# 300K 64KB SRAM (baseline)

```
> ./memory_model.py configs/cache-sram.cfg 300 \
  22 0.8 0.5 65536 cache
```



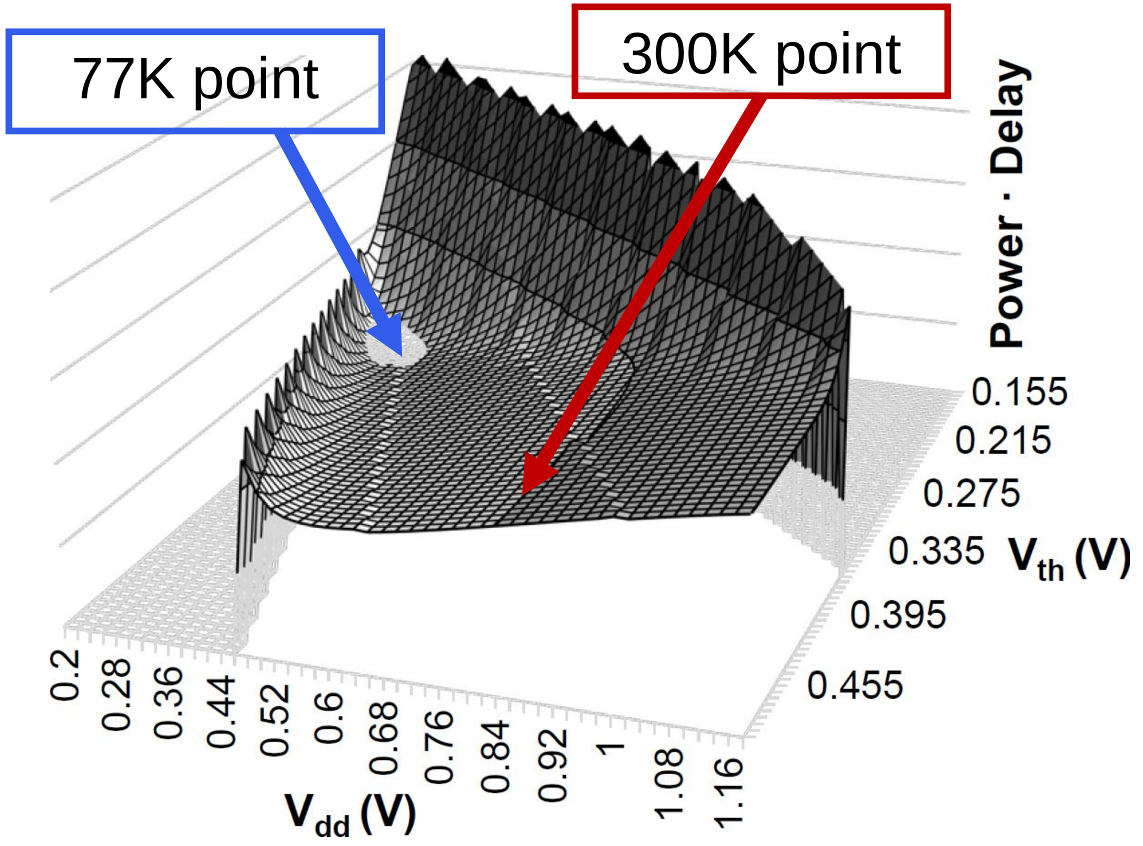




Finding the PDP-optimal voltage level

- **With CryoModel, we can find the 77K-optimal Vdd & Vth**

PDP of 77K SRAM caches



300K point: Vdd=0.8V / Vth=0.5V

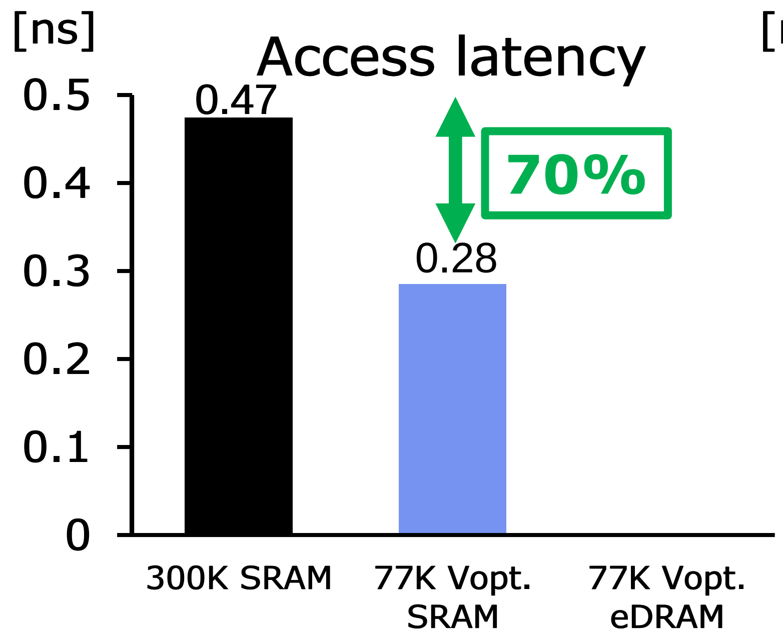
Reduced by ~50%

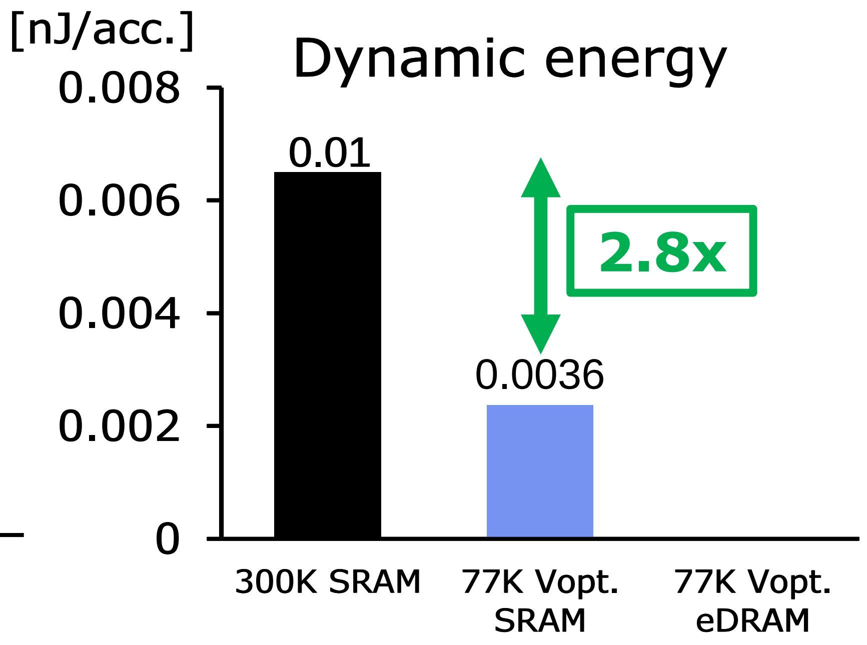
77K point: **Vdd=0.44V / Vth=0.26V**

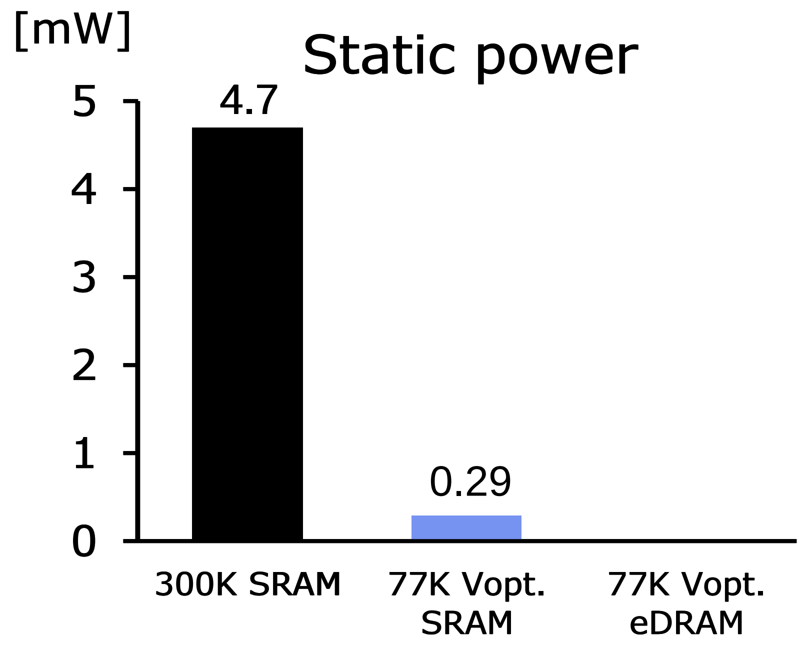
HPCS High Performance Computer System Lab

SRAM & 3T-eDRAM for L1 caches (2/3)

```
# 77K 64KB SRAM
> ./memory_model.py configs/cache-sram.cfg 77 \
  22 0.44 0.26 65536 cache
```

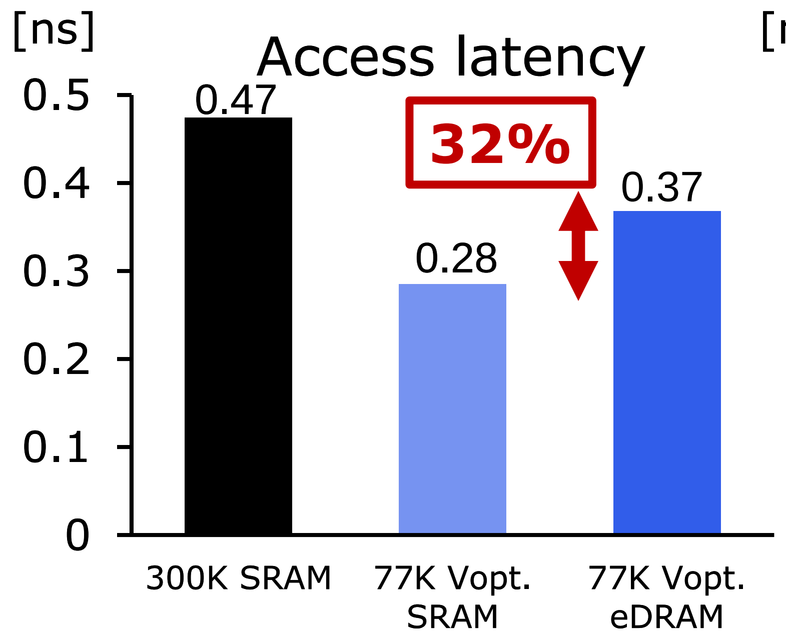


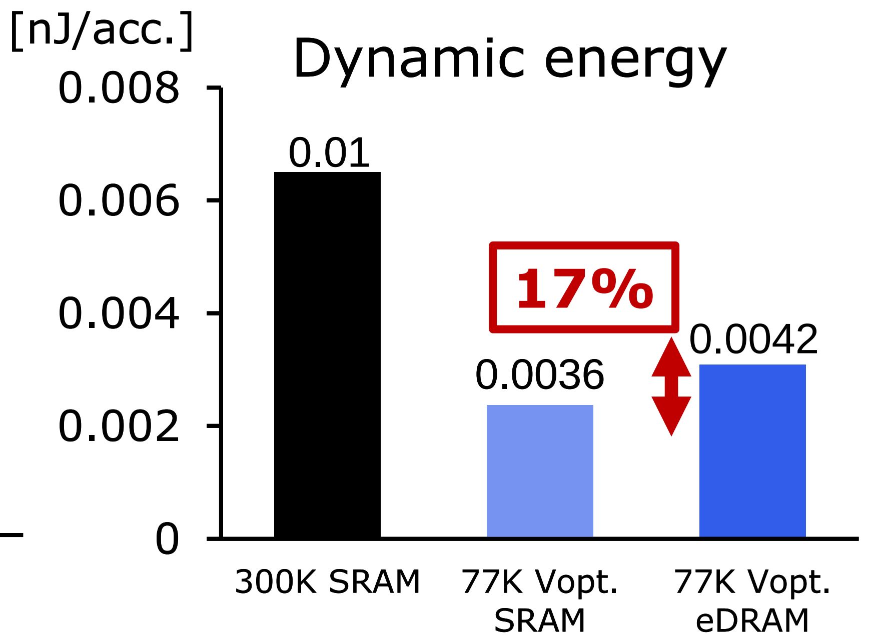


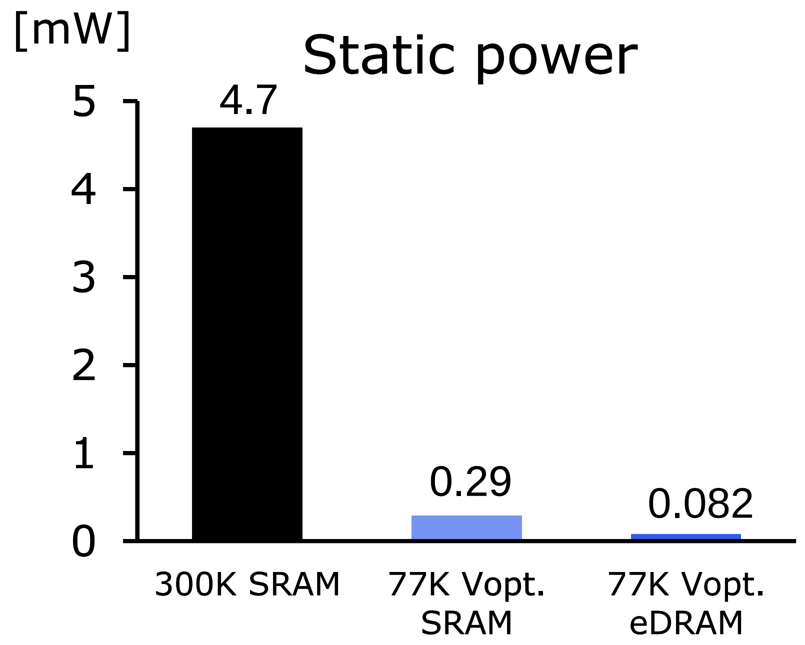


SRAM & 3T-eDRAM for L1 caches (3/3)

```
# 77K 128KB 3T-eDRAM
> ./memory_model.py configs/cache-3tedram.cfg 77 \
  22 0.44 0.26 131072 cache
```



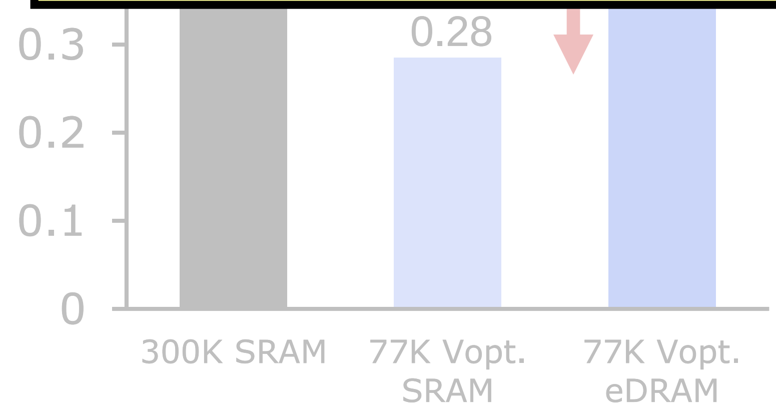


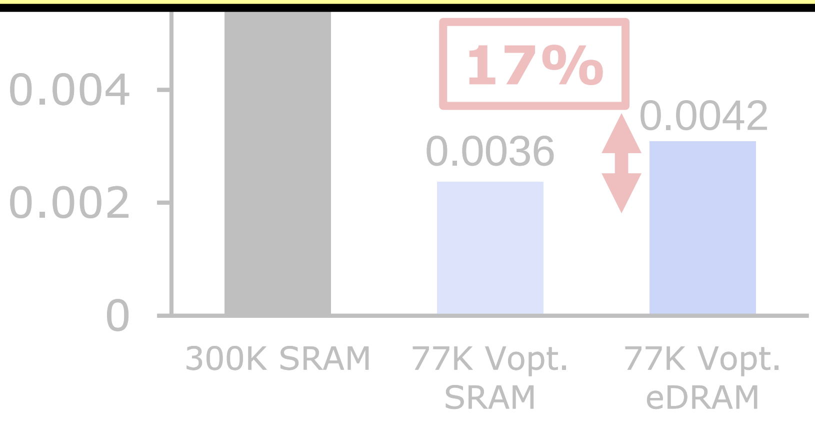


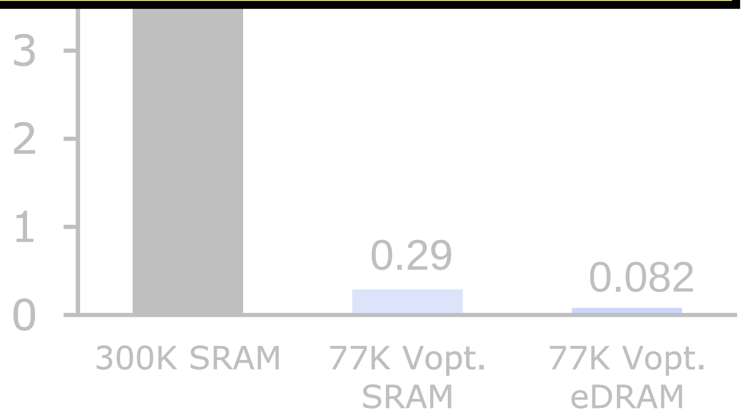
SRAM & 3T-eDRAM for L1 caches (3/3)

```
# 77K 128KB 3T-eDRAM
> ./memory_model.py configs/cache-3tedram.cfg 77 \
```

SRAM is suitable for latency-sensitive dynamic-power dominant L1 design!



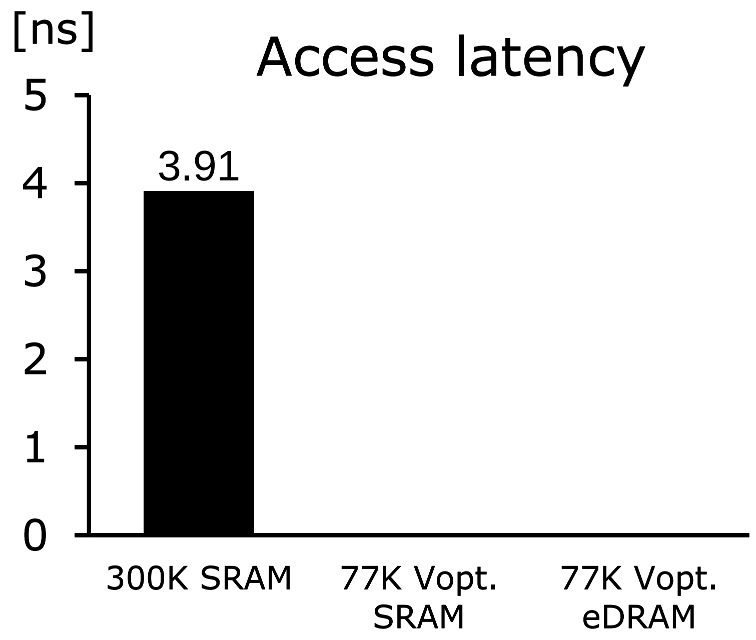


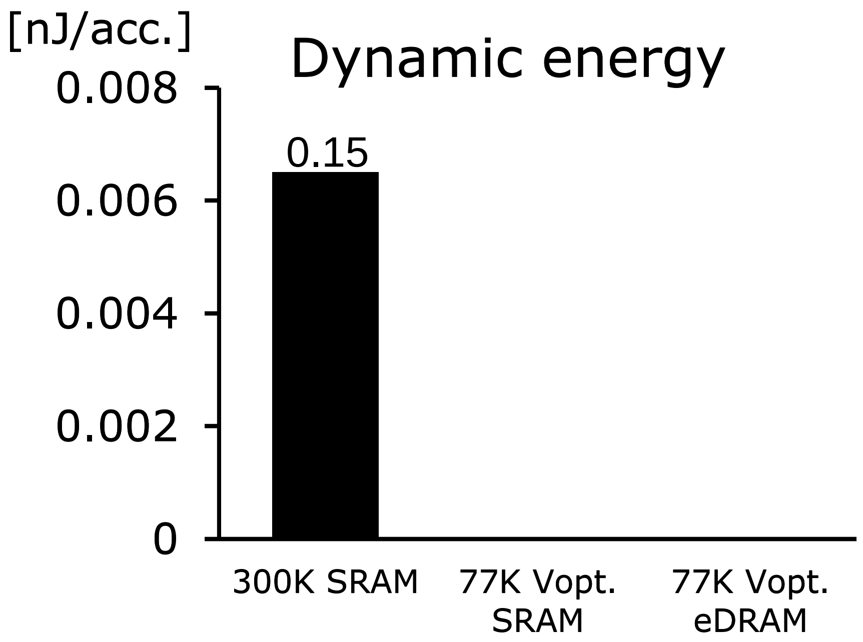


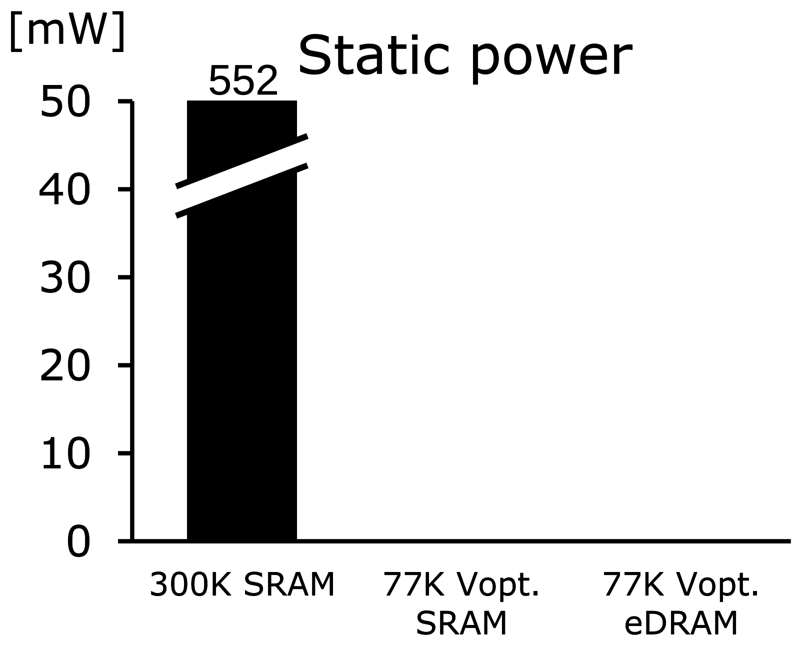
SRAM & 3T-eDRAM for L2/L3 caches (1/3)

# 300K 8MB SRAM cache (baseline)

```
> ./memory_model.py configs/cache-sram.cfg 300 \
  22 0.8 0.5 8388068 cache
```

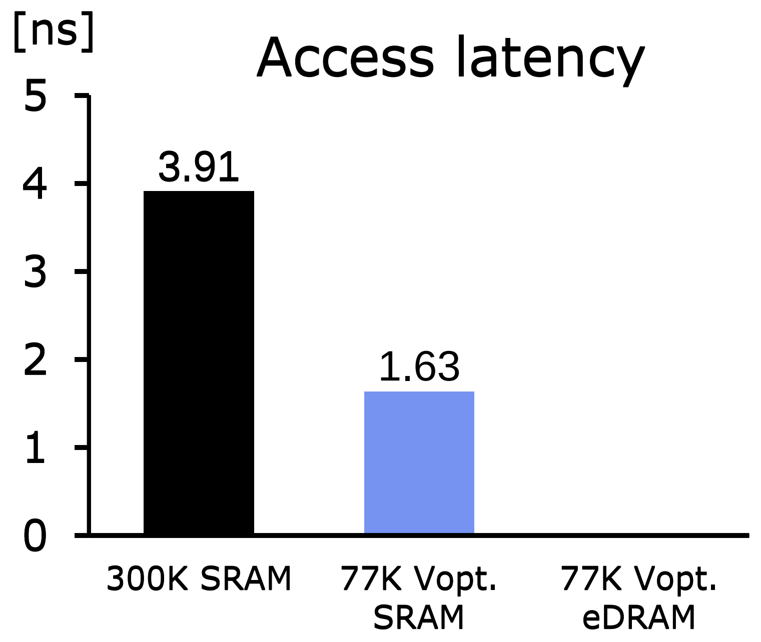


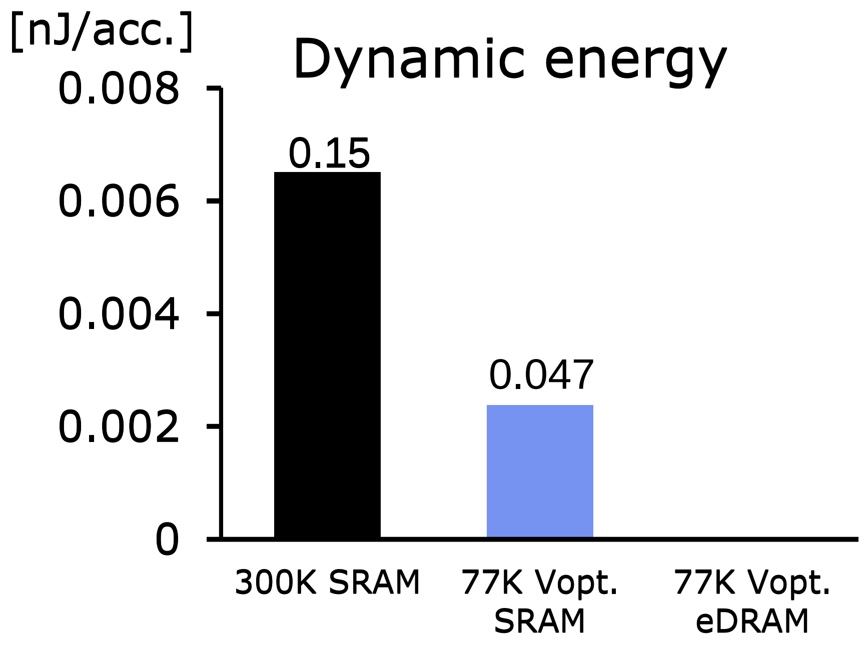


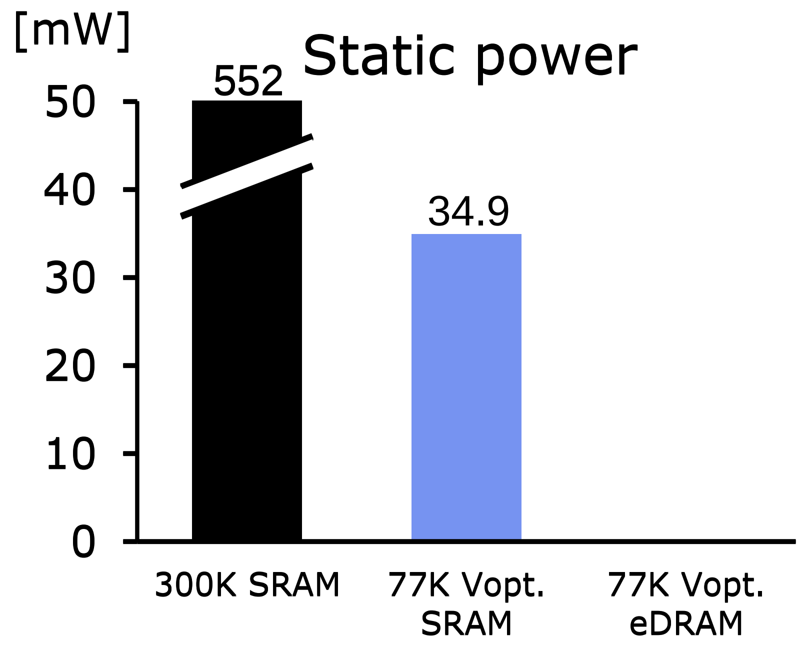


SRAM & 3T-eDRAM for L2/L3 caches (2/3)

```
# 77K 8MB SRAM cache
> ./memory_model.py configs/cache-sram.cfg 77 \
  22 0.44 0.26 8388068 cache
```

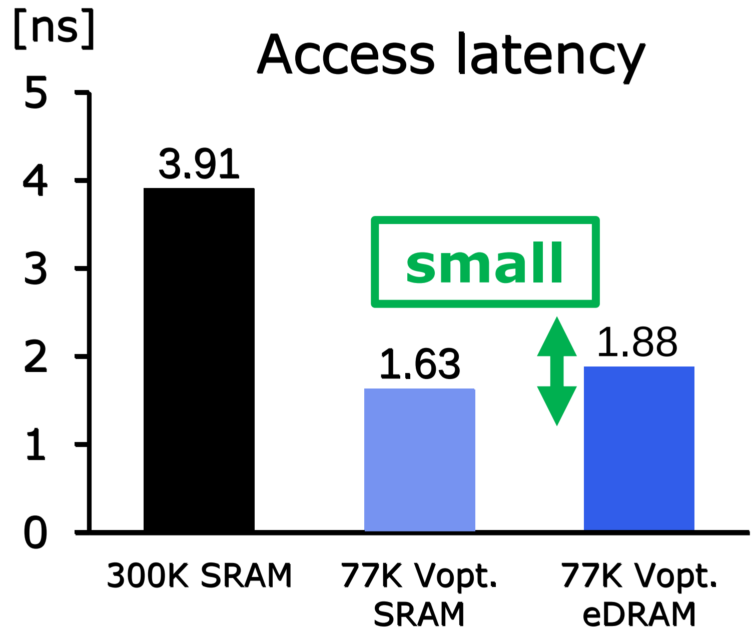


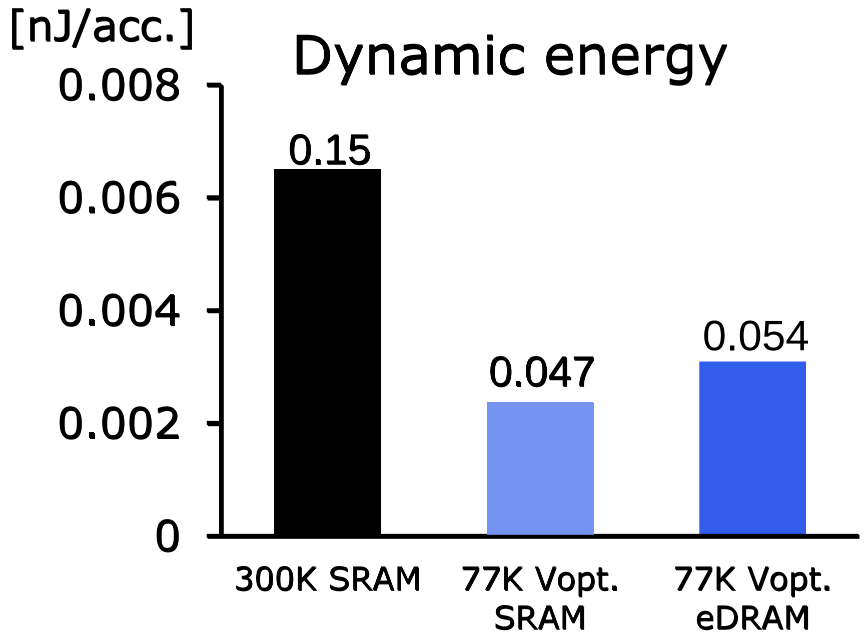


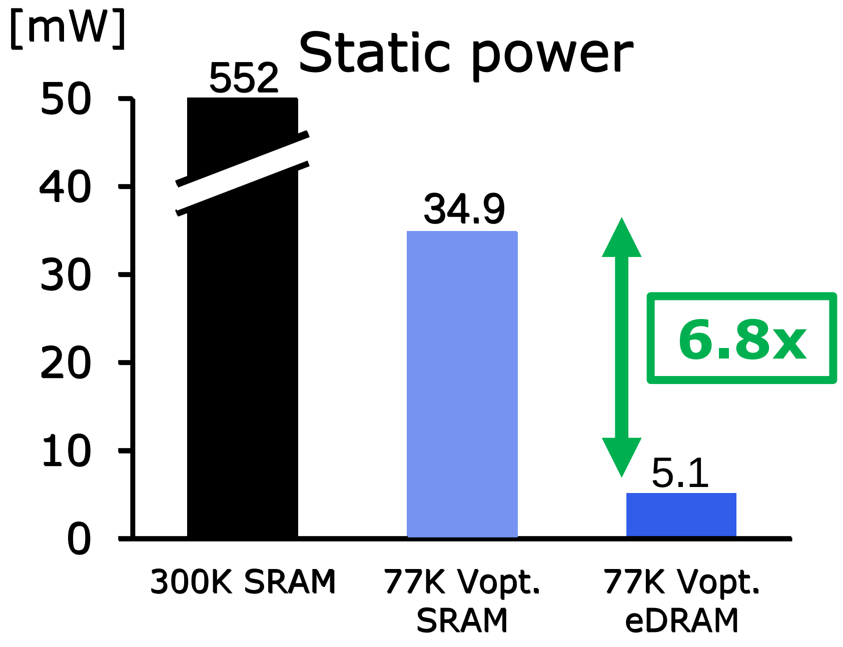


SRAM & 3T-eDRAM for L2/L3 caches (3/3)

```
# 77K 16MB 3T-eDRAM cache
> ./memory_model.py configs/cache-3tedram.cfg 77 \
  22 0.44 0.26 16777216 cache
```



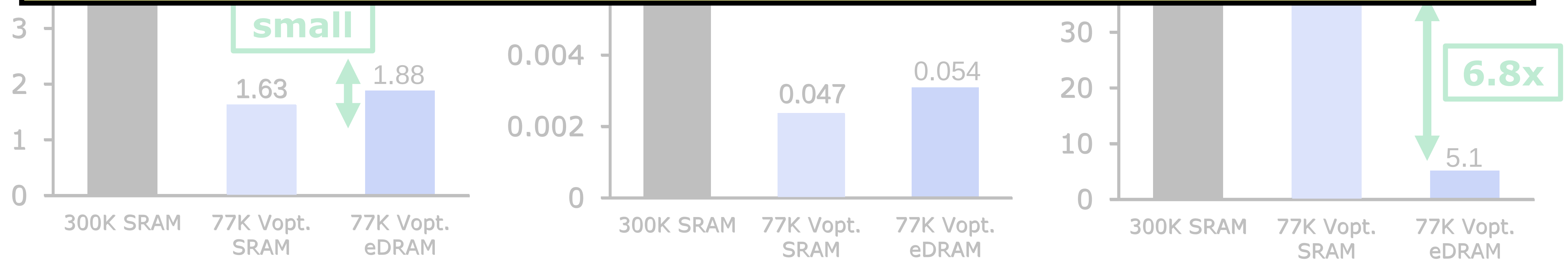




SRAM & 3T-eDRAM for L2/L3 caches (3/3)

```
# 77K 16MB 3T-eDRAM cache
> ./memory_model.py configs/cache-3tedram.cfg 77 \
```

3T-eDRAM is suitable for capacity-sensitive static power dominant L2 and L3 design!



Propose CryoCache [ASPLOS'20]

	SRAM	3T-eDRAM
Latency	Low latency	-
Capacity	-	High capacity
Energy	Low dynamic energy	Low static energy

For L1 design **For L2 & L3 design**

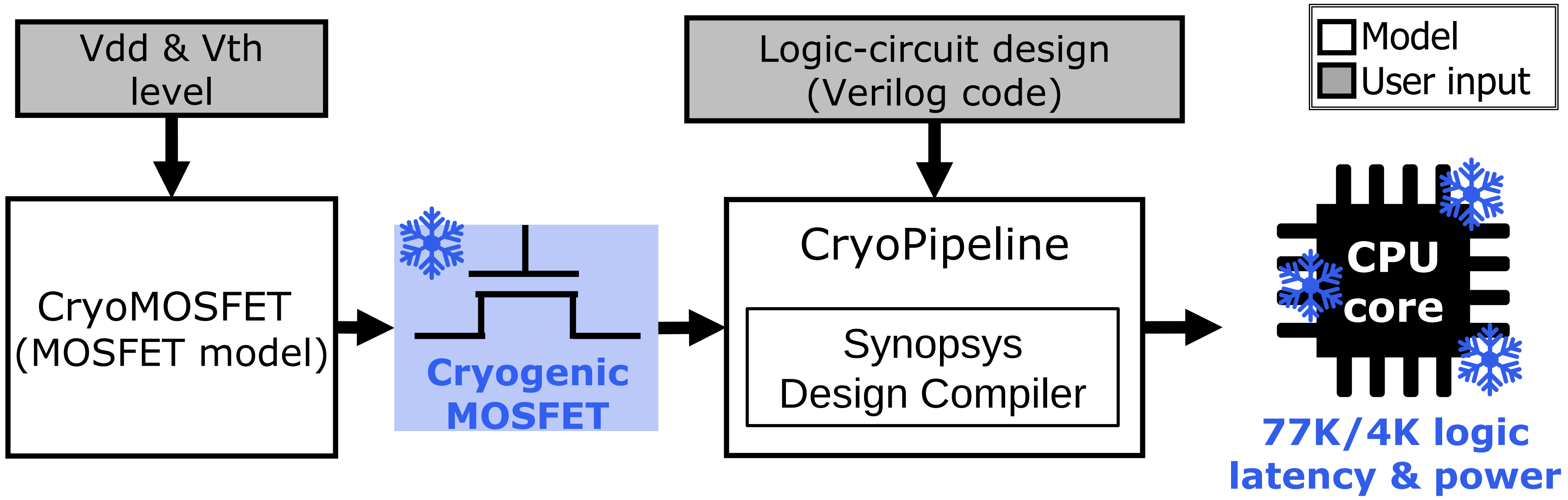
CryoCache: 77K-optimal cache architecture

SRAM for L1 design and 3T-eDRAM for L2 & L3 design

Index

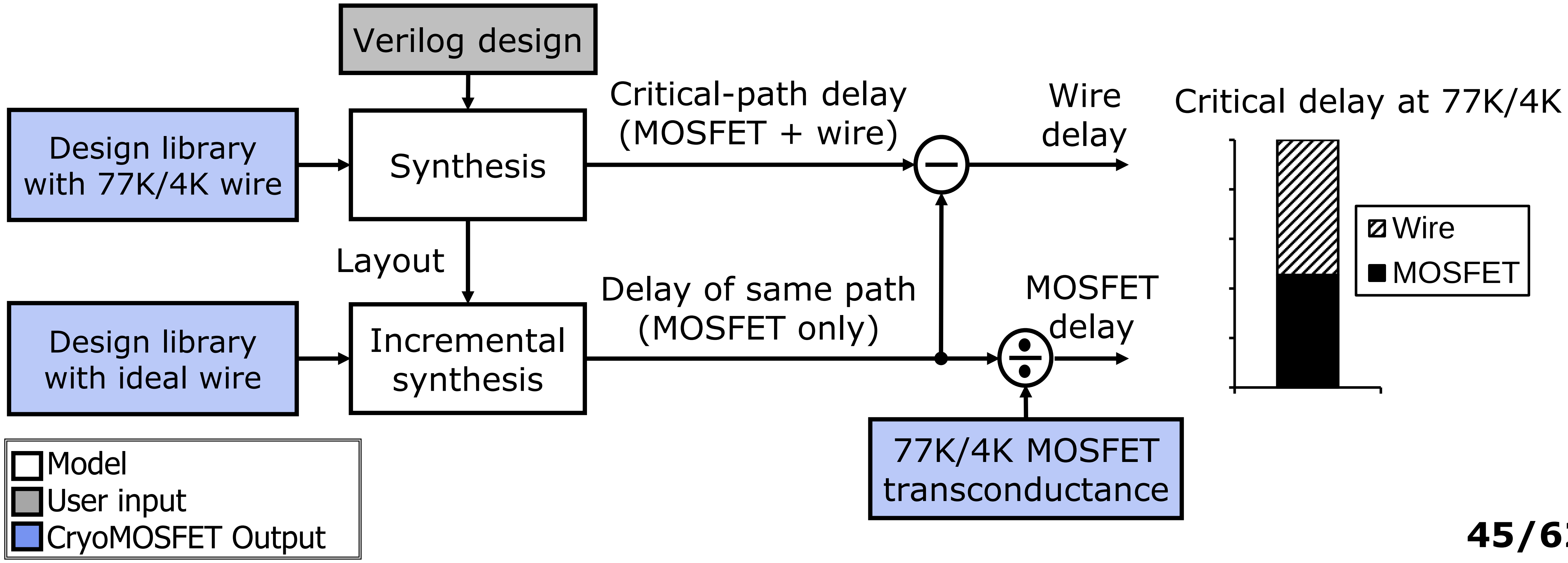
77K/4K logic modeling overview

- **Logic model predicts the latency and power of 77K/4K logics**
- **Logic model supports any Verilog-defined circuit designs**



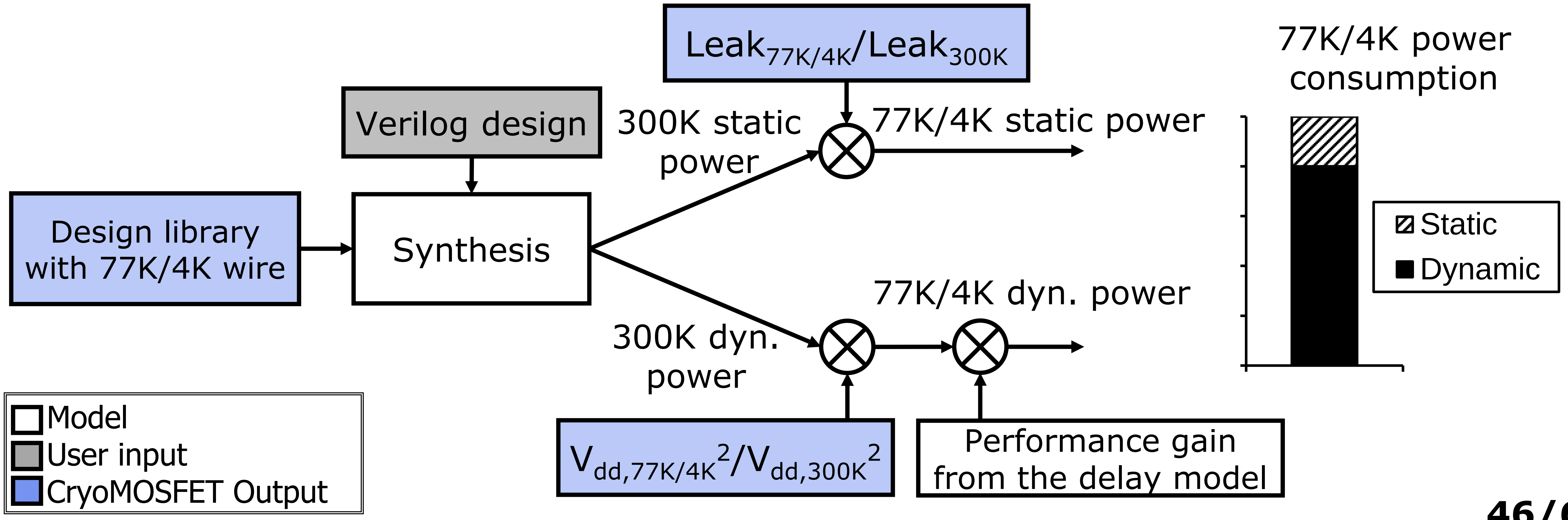
77K/4K logic: Critical-path delay model

- **Logic model predicts the critical-path delay at 77K/4K by calculating the transistor & wire delays separately**


Verilog design
Design library with 77K/4K wire
Synthesis
Critical-path delay (MOSFET + wire)
Wire delay
Layout
Design library with ideal wire
Incremental synthesis
Delay of same path (MOSFET only)
MOSFET delay
77K/4K MOSFET transconductance
Critical delay at 77K/4K
Wire
MOSFET
Model
User input
CryoMOSFET Output

77K/4K logic: Power model

- **Logic model predicts the 77K/4K power consumption by calculating the static and dynamic power separately**



How to run the logic model? (1/3)

```
> cd ../CryoPipeline
> ls
```

- `src_vlg/`
  - Target Verilog code
- `dc_compile/`
  - Design Compiler / Milkyway scripts
- `milky-*/, freepdk-45nm/`
  - Logical / Physical libraries
- `latency_results/, *.ddc`
  - Synthesis results

- `logic_model.py`
  - Orchestrate all the functionality of CryoPipeline (i.e., library generation, synthesis, critical-path analysis, final-result report).

How to run the logic model? (2/3)

```
> ./logic_model.py --design_name {design name} \
  --temperature {temp.} --node {node} --vdd {vdd} --vth {vth0}
```

- **{design name}**
  - The name of the target Verilog design
- **{temperature}**
  - We support the **300K, 77K, and 4K only**
- **{node}**
  - We support the **45nm technology only**
    (FreePDK45nm is the only editable open-source library)

How to run the logic model? (3/3)

```
> mv {target Verilog design} ./src_vlg
```

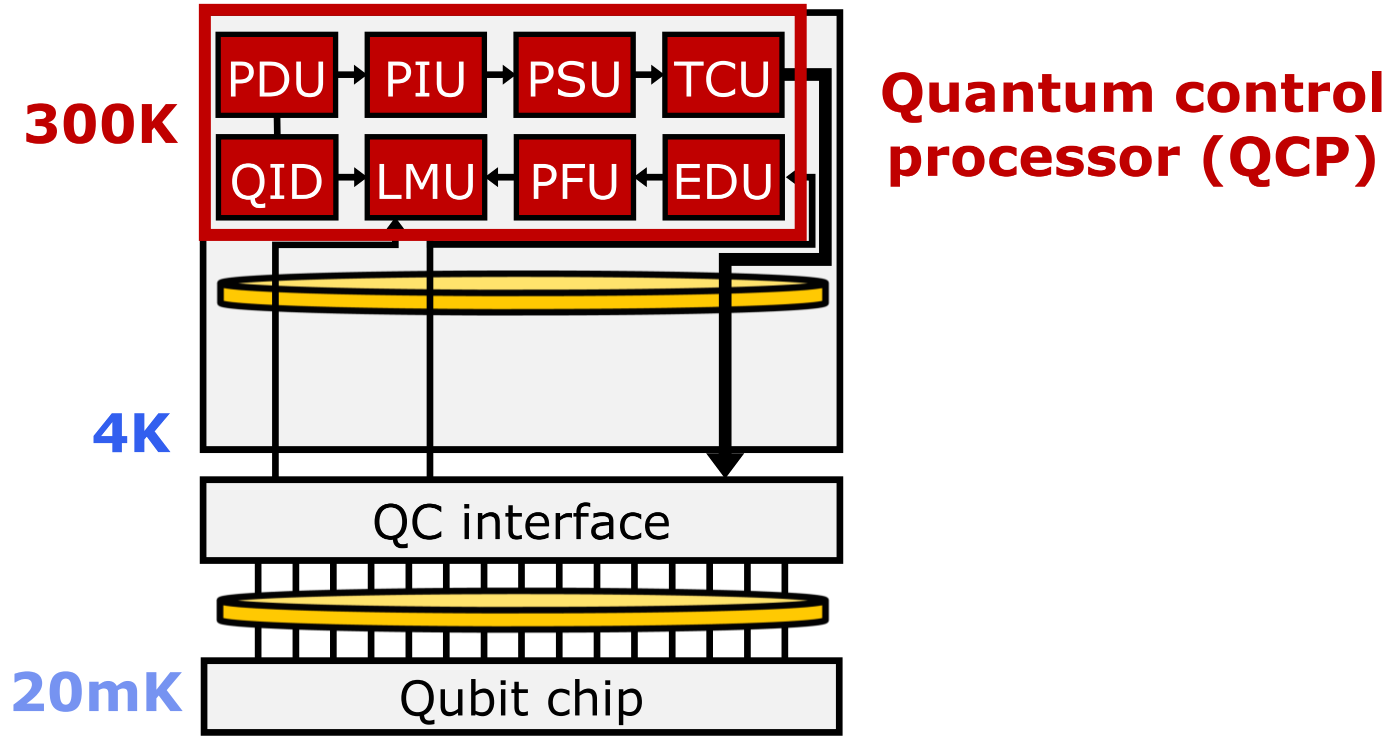
- **User should construct "./src_vlg" with their target Verilog design**
  - For tutorial, we construct "./src_vlg" based on PSU used in XQsim [ISCA'22]
  - You can apply any Verilog design instead of this design for your own research

Case study #3.

Let’s evaluate the scalability of the 4K-CMOS quantum control processor (QCP)! [ISCA’22]

Evaluating the QCP scalability (1/5)

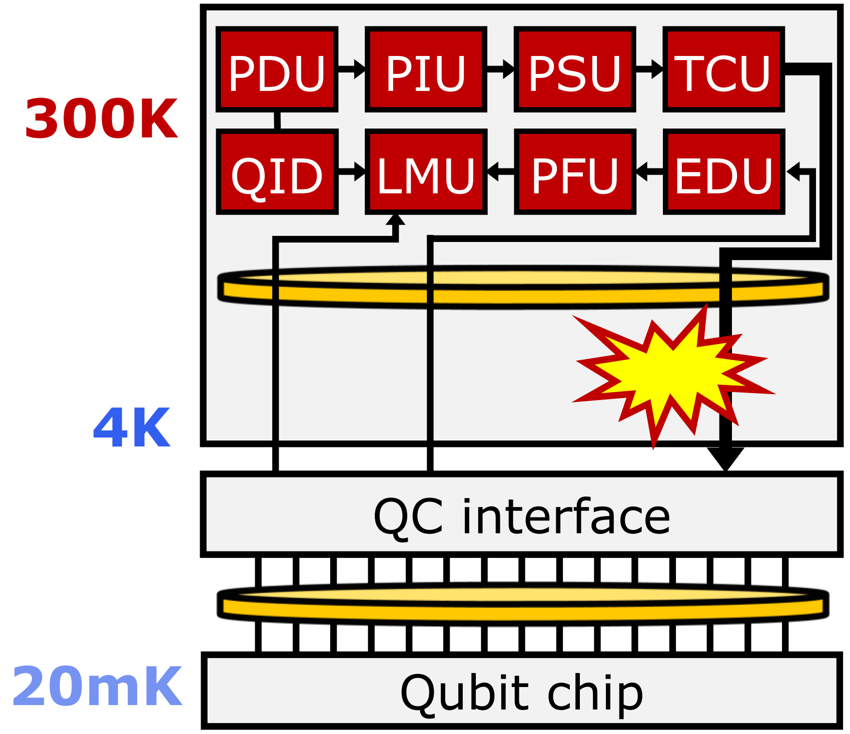
- **4K CMOS-device power can limit the QCP's scalability!**
  - QCP is the hardware for quantum-error correction support.



<300K QCP>

Evaluating the QCP scalability (1/5)

- **4K CMOS-device power can limit the QCP's scalability!**
  - QCP is the hardware for quantum-error correction support.

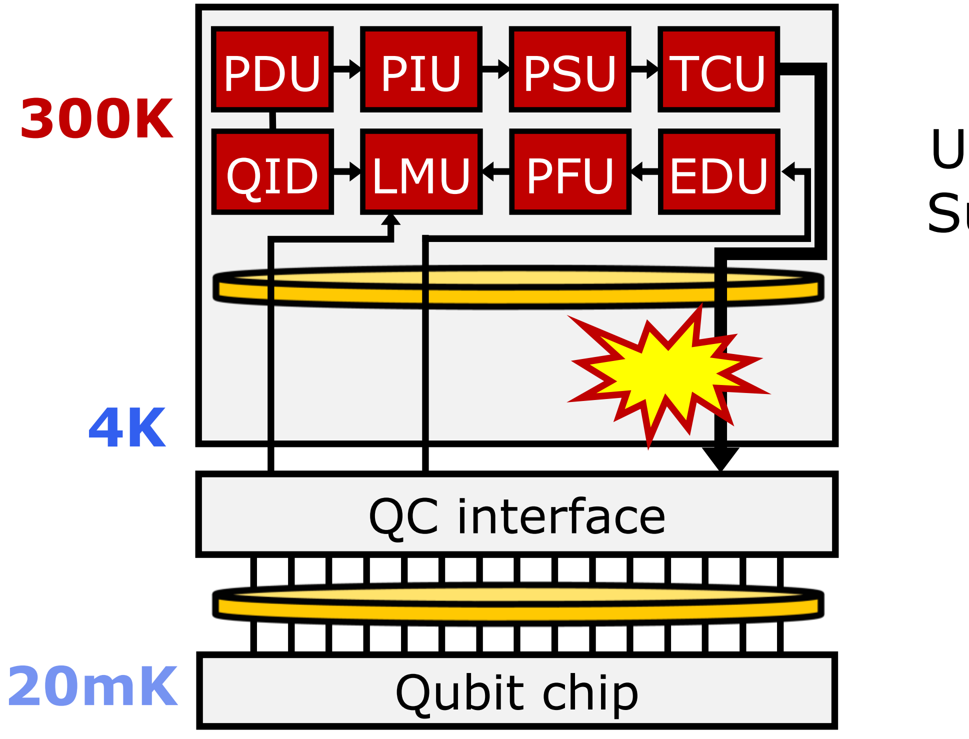


<300K QCP>

Wire heat > 4K power budget

Evaluating the QCP scalability (1/5)

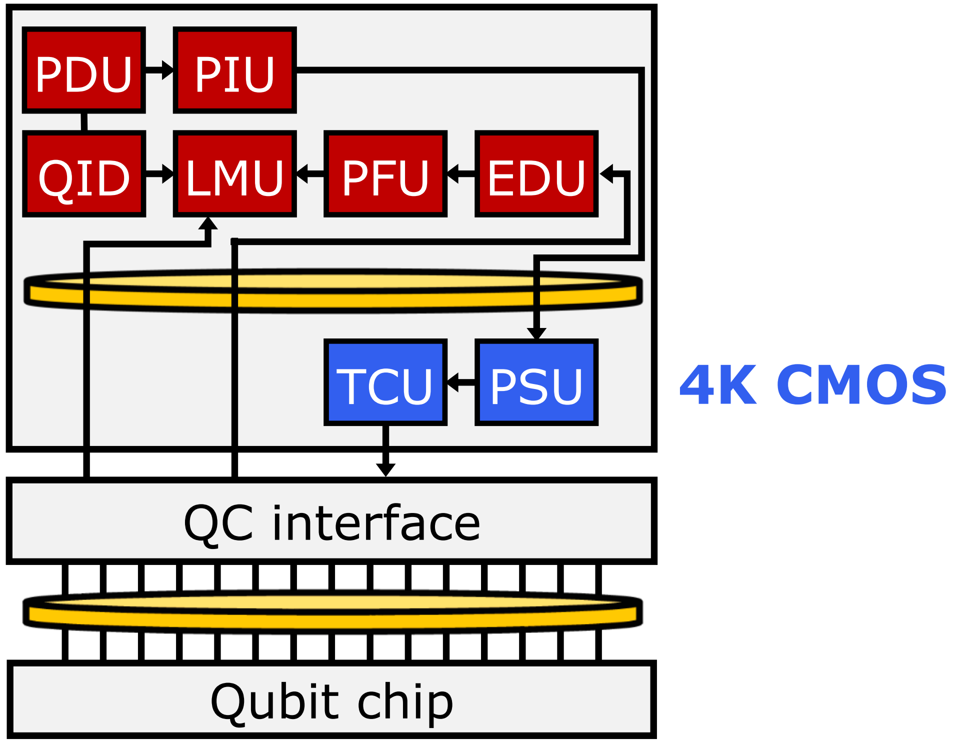
- **4K CMOS-device power can limit the QCP's scalability!**
  - QCP is the hardware for quantum-error correction support.



<300K QCP>

Wire heat > 4K power budget

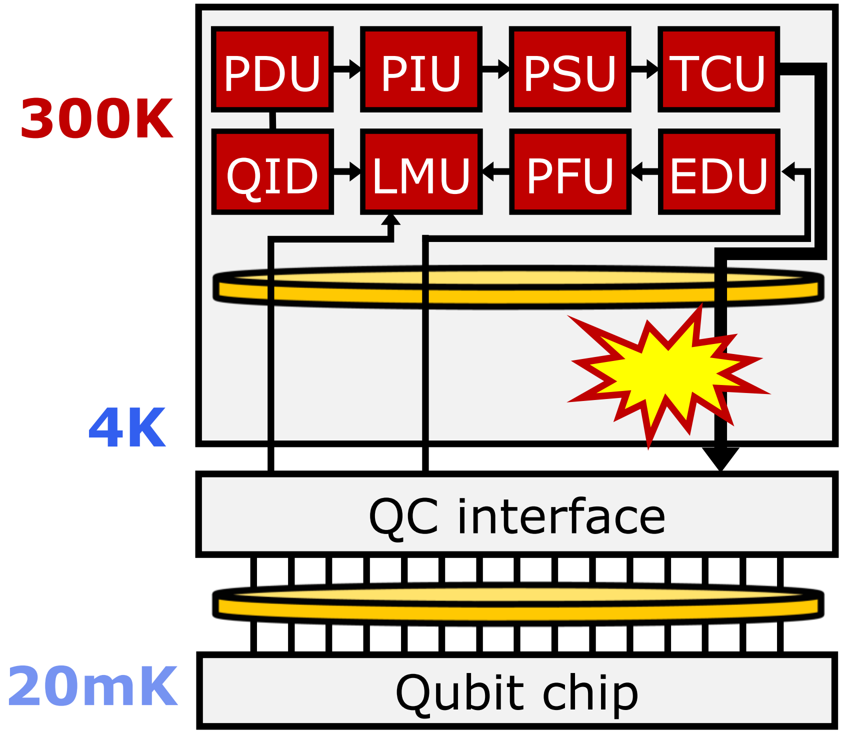
Utilize scalable Supercon. wire



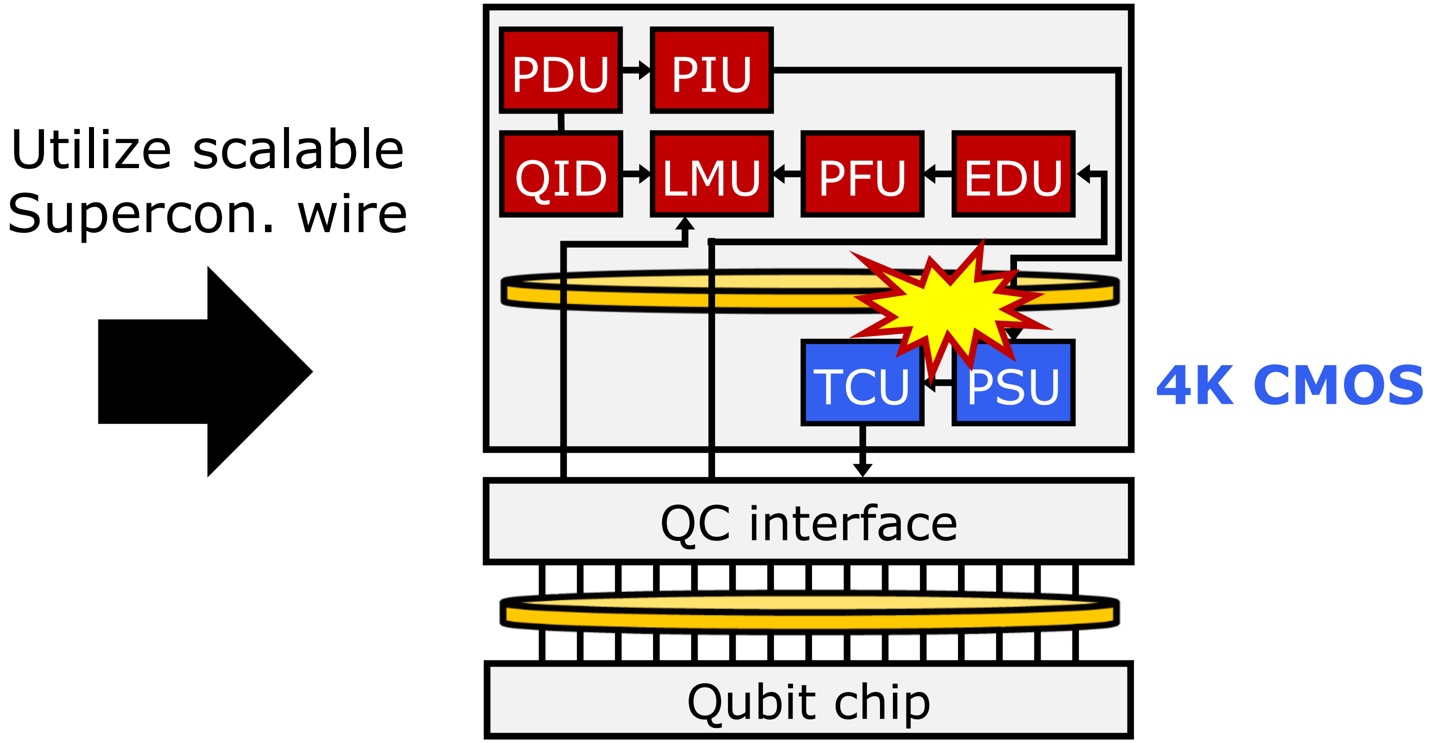
<QCP with 4K CMOS>

Evaluating the QCP scalability (1/5)

- **4K CMOS-device power can limit the QCP's scalability!**
  - QCP is the hardware for quantum-error correction support.



<300K QCP>



<QCP with 4K CMOS>

Wire heat > 4K power budget

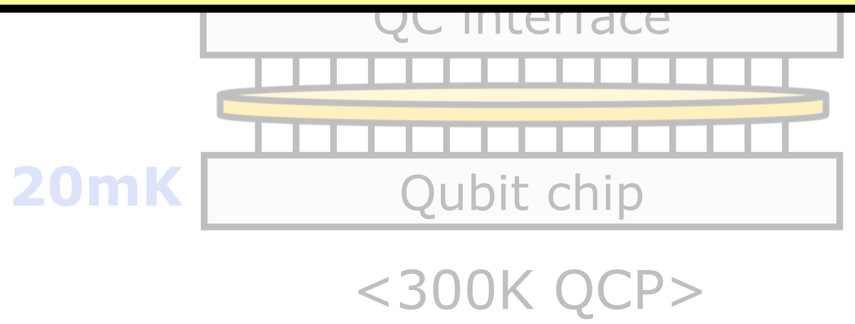
4K CMOS power > 4K power budget

Evaluating the QCP scalability (1/5)

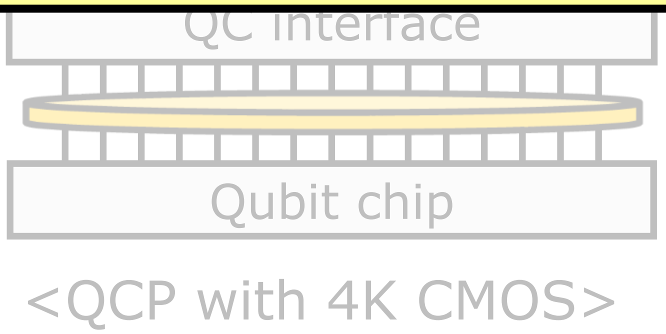
- **4K CMOS-device power can limit the QCP's scalability!**
  - QCP is the hardware for quantum-error correction support.

To evaluate QCP's scalability, we should predict the 4K device power consumption.

→ CryoModel can predict 4K CMOS power consumption!



Wire heat > 4K power budget

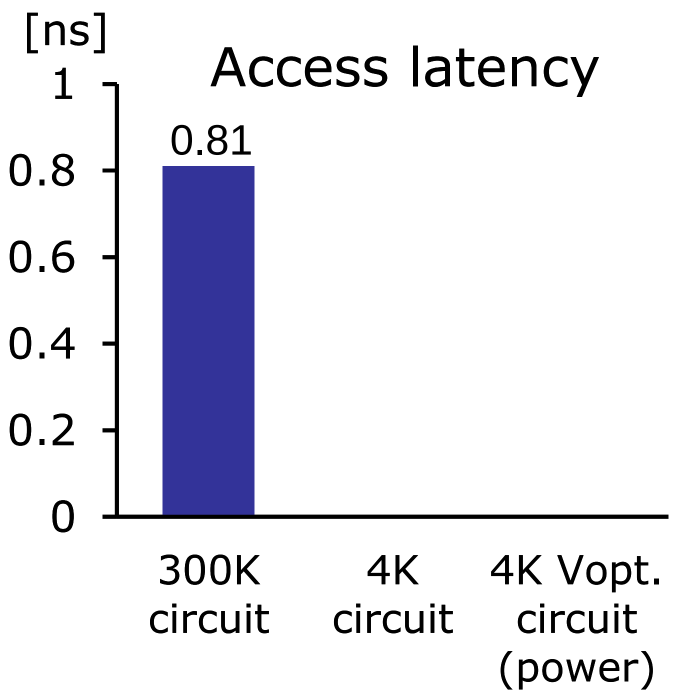


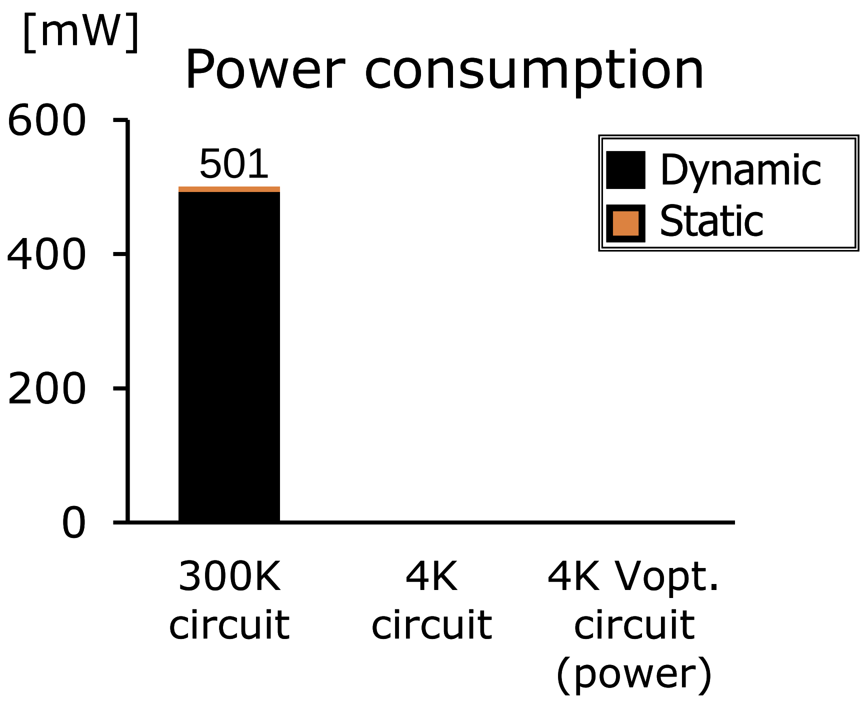
4K CMOS power > 4K power budget

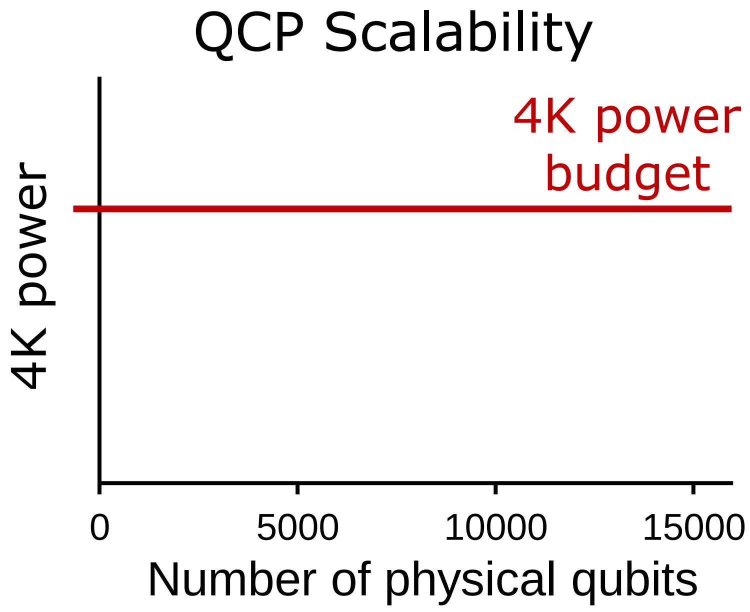
Evaluating the QCP scalability (2/5)

# 300K logic design (baseline)

```
> ./logic_model.py --design_name PSU --temperature 300
```



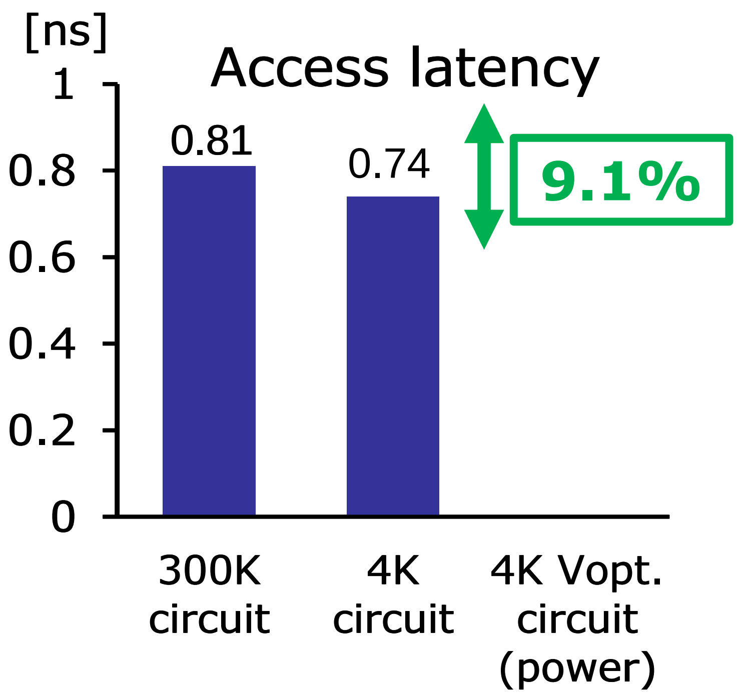


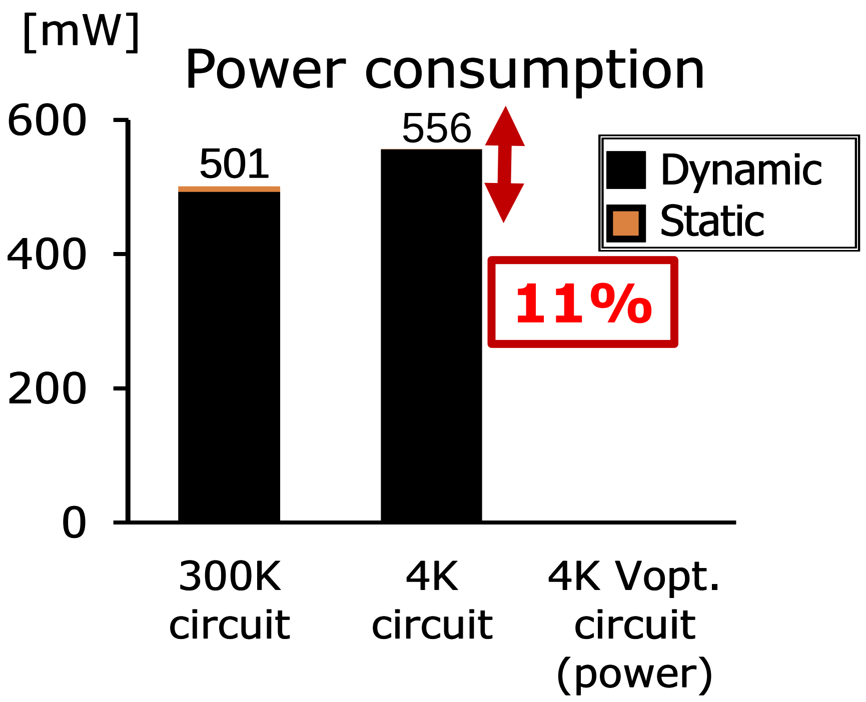


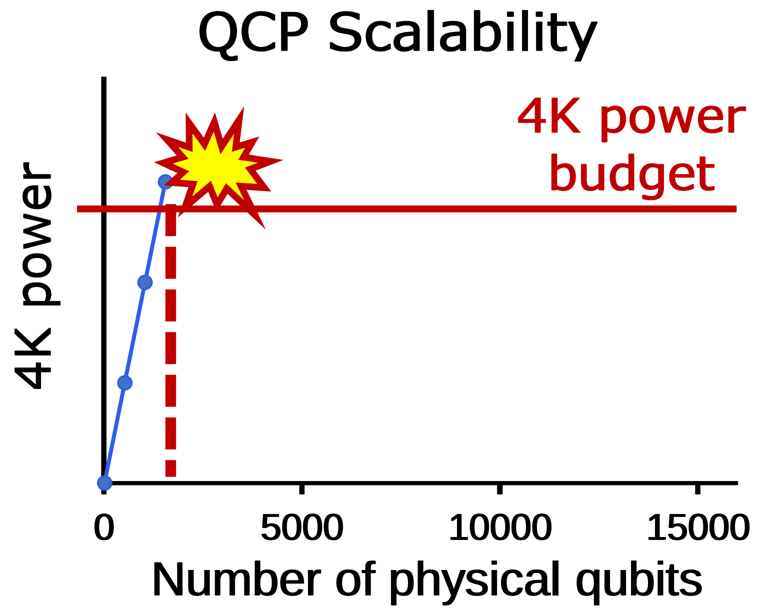
Evaluating the QCP scalability (3/5)

# 77K logic design with circuit-level optimization

```
> ./logic_model.py --design_name PSU --temperature 4
```



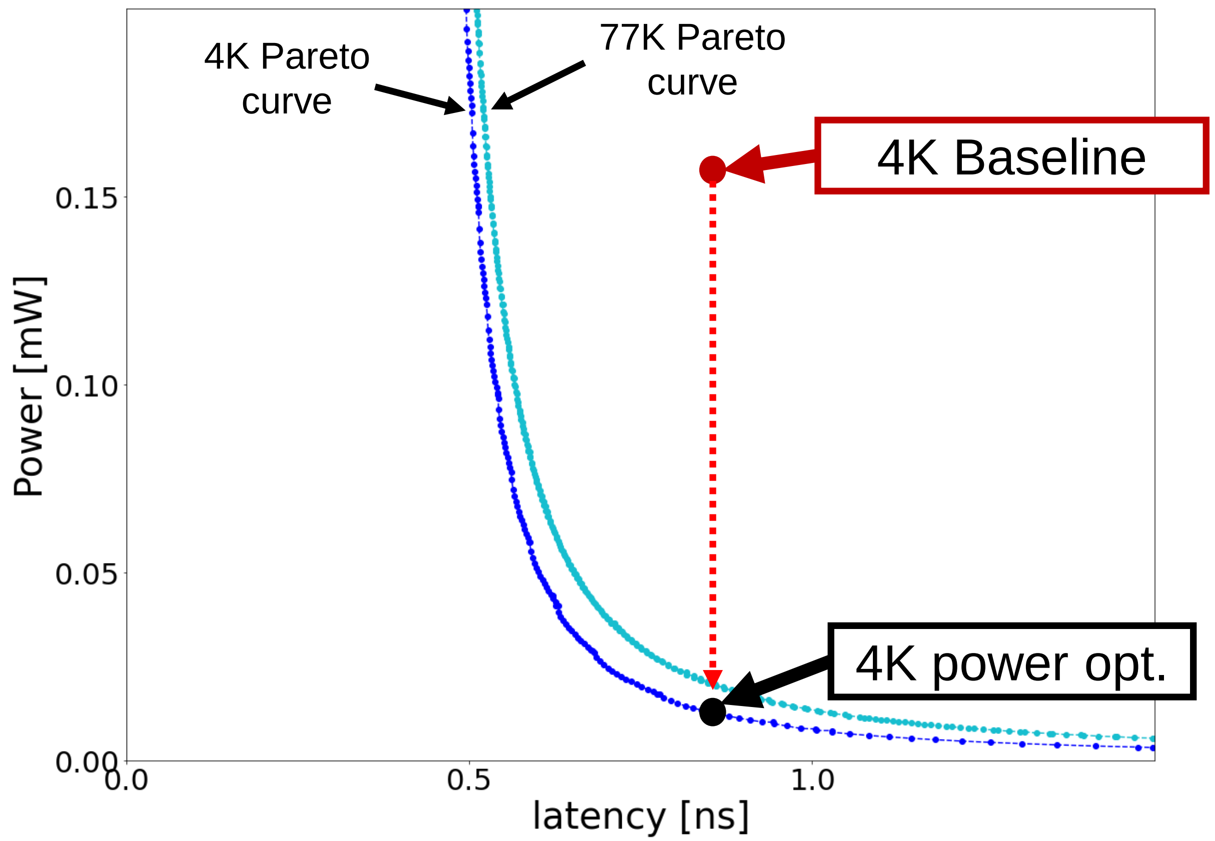




Evaluating the QCP scalability (4/5)

- CryoModel can find the power-optimized design to increase the scalability.

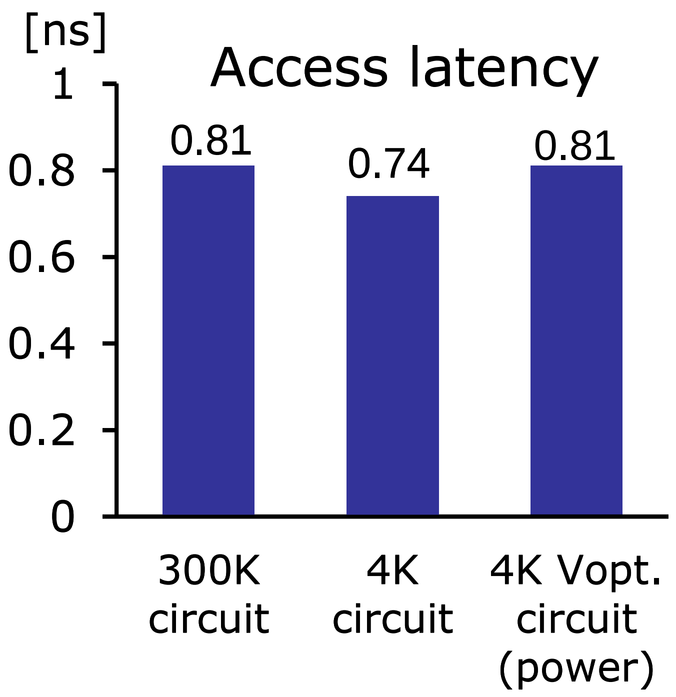
Latency and power for various voltages

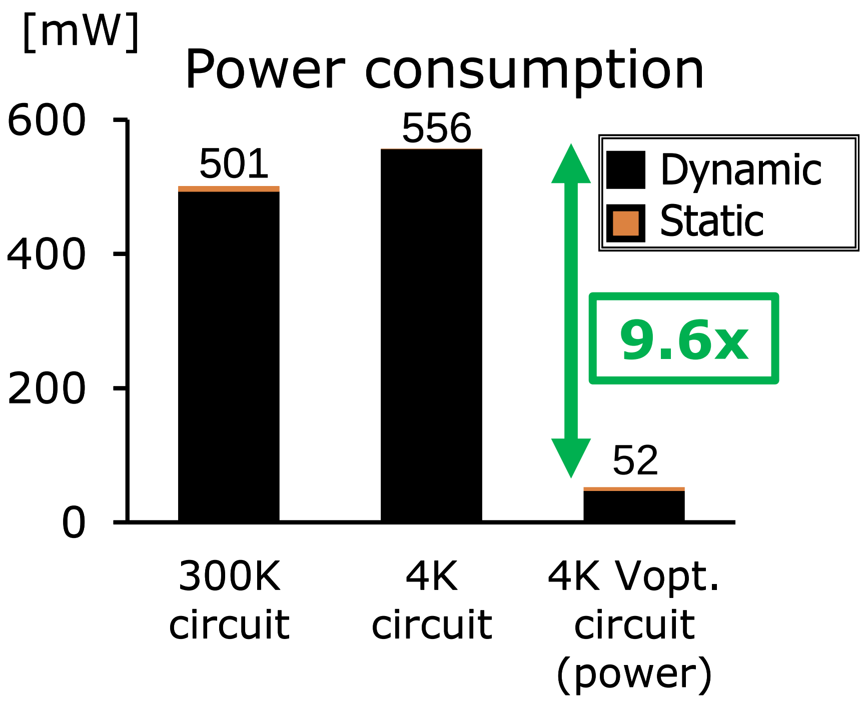


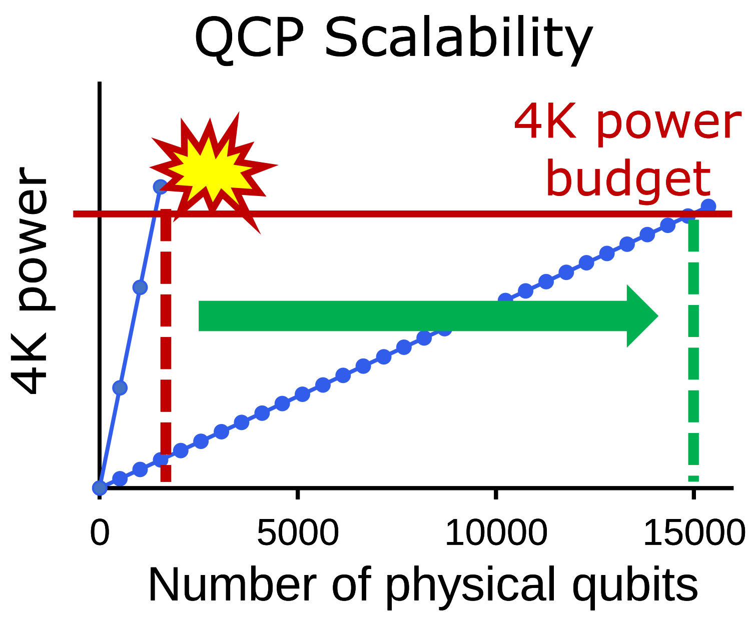
- **Power-optimized design**
  - **Lowest power** for same performance
  - We use this design for our quantum control processor [ISCA'22]

Evaluating the QCP scalability (5/5)

```
# 77K logic design with power-oriented voltage scaling
> ./logic_model.py --design_name PSU --temperature 4 \
  --vdd 0.38 --vth 0.123
```



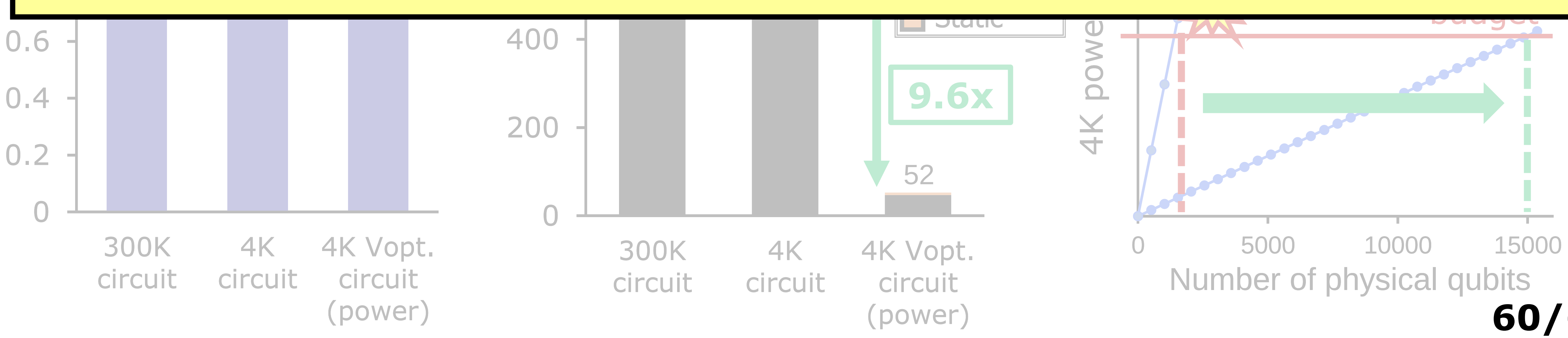




Evaluating the QCP scalability (5/5)

```
# 77K logic design with power-oriented voltage scaling
> ./logic_model.py --design_name PSU --temperature 4 \
    --vdd 0.38 --vth 0.123
```

You can also evaluate the impact of your microarchitectural optimizations with CryoModel!



Index

CryoModel: Summary

- **Cryogenic, superconductor, and quantum computing require CryoCMOS running at 77K and 4K.**

- **CryoModel predicts the latency & power of CryoCMOS.**
  - Memory model supports SRAM, 3T-eDRAM, and DRAM at 77K and 4K
  - Logic model supports Verilog-defined circuit design at 77K and 4K

If you want to use CMOS circuits for cryogenic, superconductor, and quantum computing,

please use CryoModel.

Thank You!

Dongmoon Min

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

E-mail: dongmoon.min@snu.ac.kr
Web: https://hpcs.snu.ac.kr/~dongmoon