

# The BubbleWrap Many-Core: Popping Cores for Sequential Acceleration

Ulya Karpuzcu, Brian Greskamp, Josep Torrellas  
University of Illinois

<http://iacoma.cs.uiuc.edu/>

# Problem: The Power Wall

---

## Ideal Scaling

- Dynamic (switching) power density remains constant

$P_{\text{DYN}} = \# \text{Devices} \times \text{Frequency of switching} \times \text{Energy per switching}$



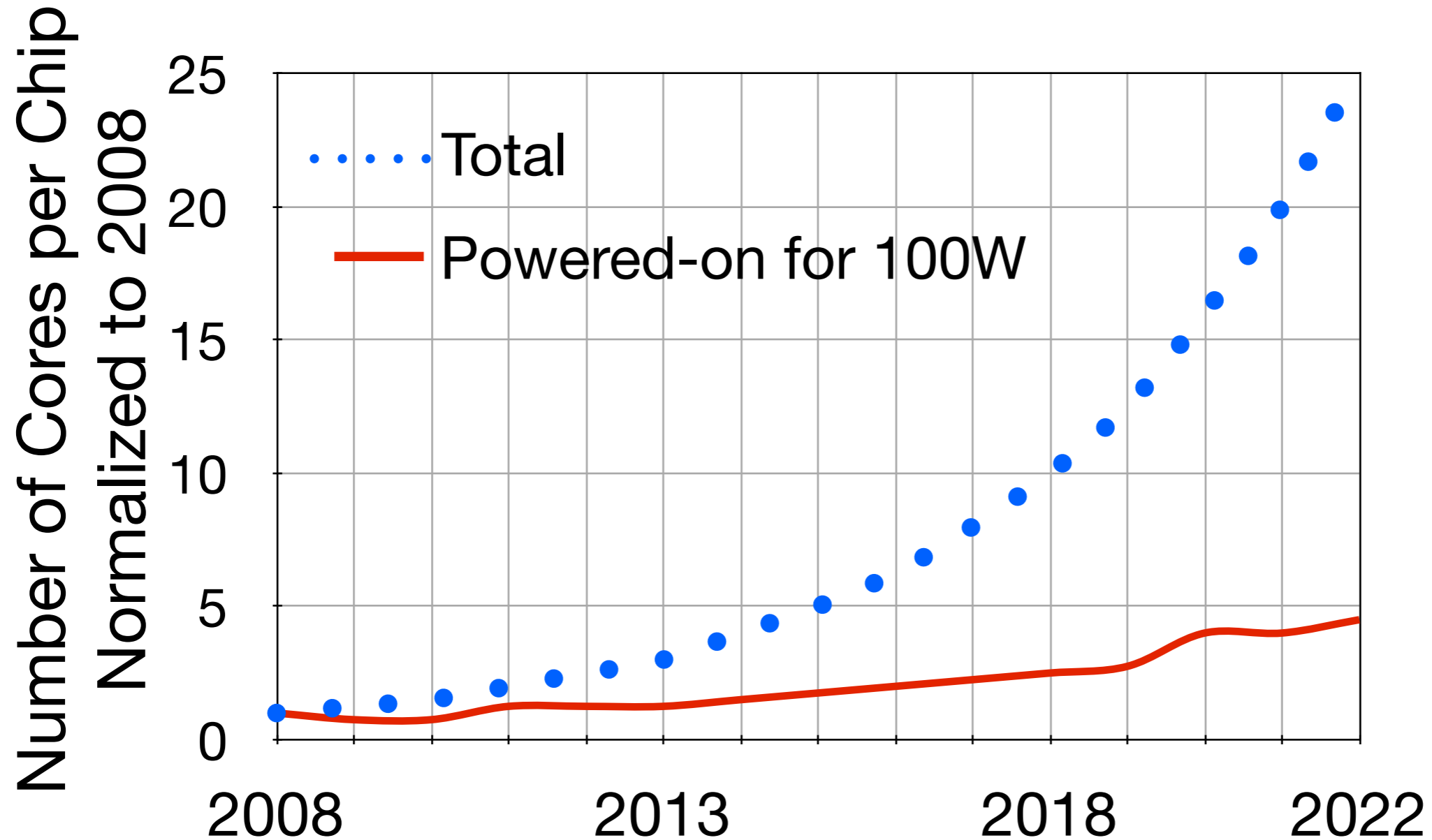
$\propto V_{\text{dd}}^2$

## Practical Scaling

- $V_{\text{dd}}$  has been scaling down slower than ideally
- Historically: Higher  $V_{\text{dd}} \equiv$  Higher performance
- Recently: Practically stagnated  $V_{\text{th}}$  scaling to control static power

**Dynamic Power Density is increasing**

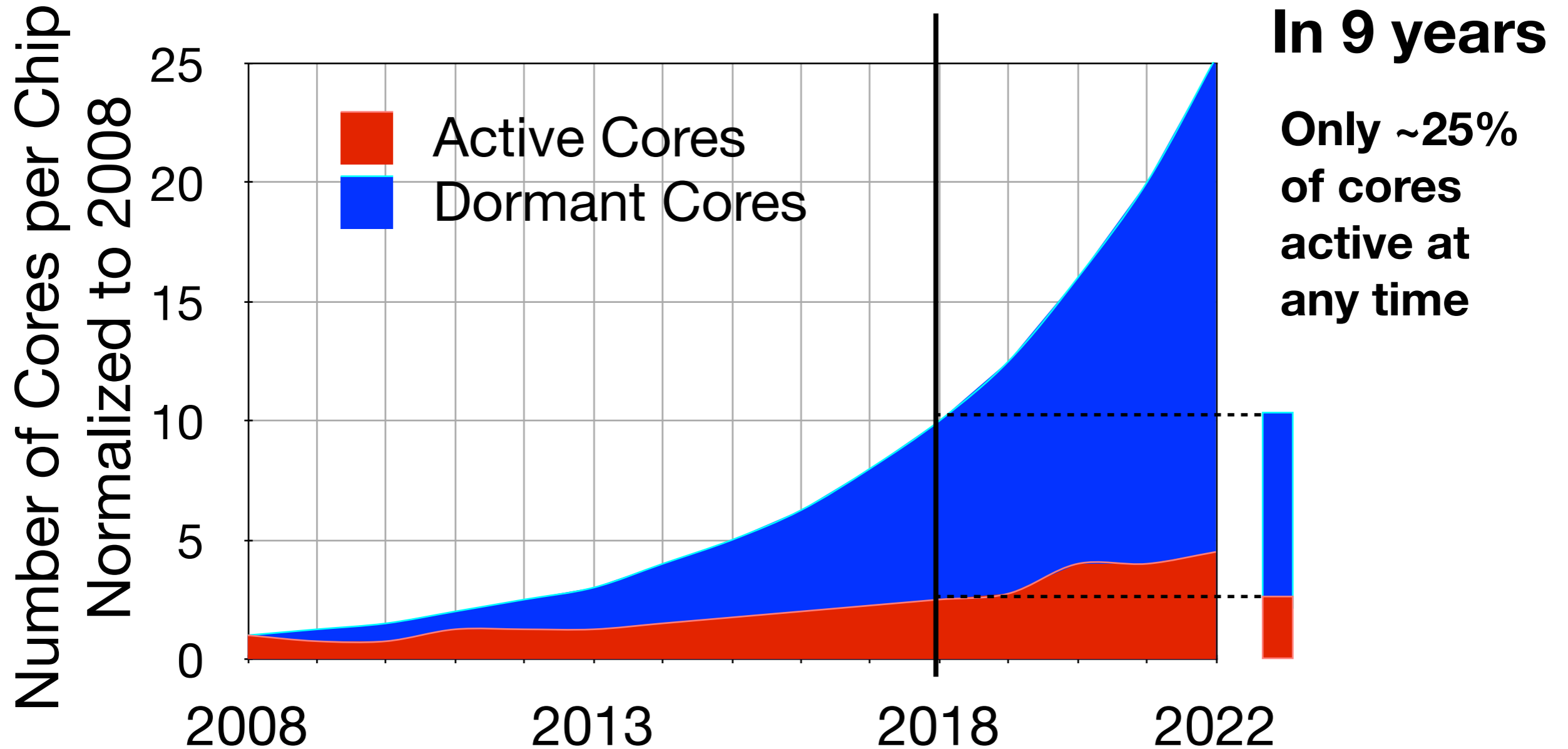
# The Power Wall



ITRS 2008



# The Power Wall

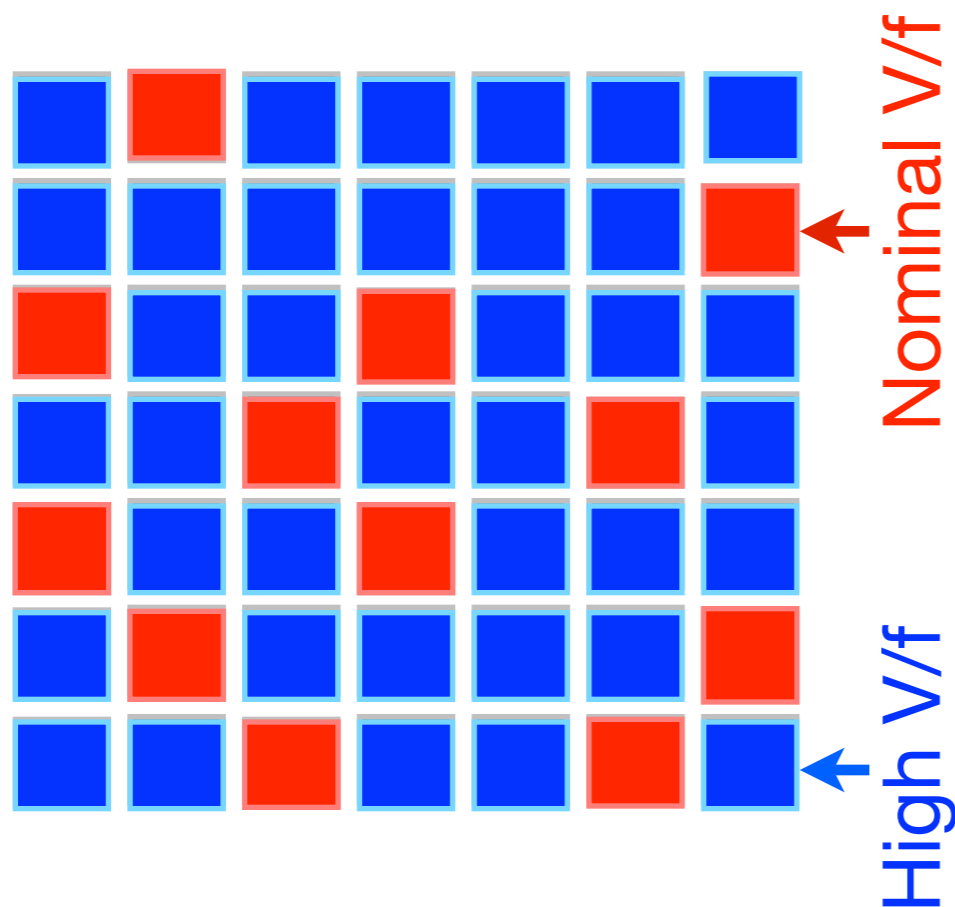


ITRS 2008



# The BubbleWrap Many-Core

- Exploit dormant cores to accelerate sequential sections at the cost of a shorter per-core service life



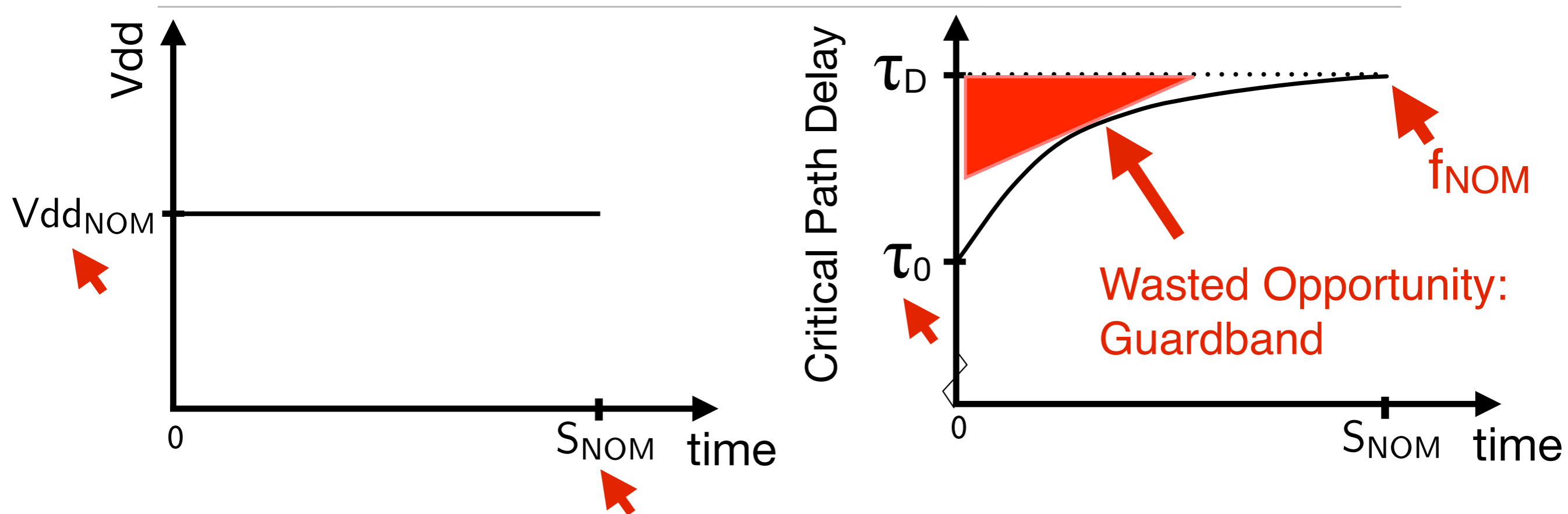
- Base: A homogeneous many-core
- **Throughput Cores**
  - Most energy-efficient cores
  - Run parallel sections
  - Operate at nominal V/f
- **Expendable Cores**
  - Run sequential sections
  - Operate at elevated V/f
  - Discarded early due to shorter service life (Popped like BubbleWrap)

# Core Aging

---

- Manifestation: Progressive slow-down in logic as the core is being used
- Main contributor: Bias Temperature Instability (BTI)
  - Induces increase in critical path delays  $\propto \text{time}^{\text{const.} < 1}$
  - Aging rate: **Exponential** dependence on Vdd and T

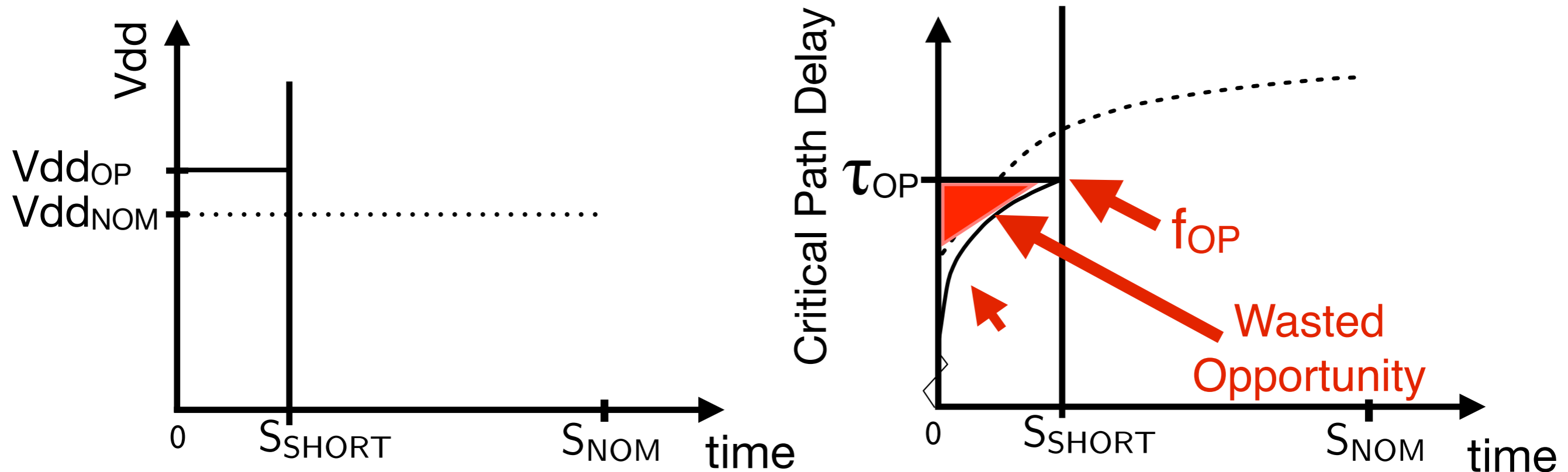
# Aging-induced Degradation



- BTI-induced increase in critical path delays  $\propto \text{time}^{\text{const.} < 1}$
- $f_{NOM}$  set by the delay at the end of the service-life ( $S_{NOM}$ )

$$f_{NOM} = 1/\tau_D$$

# Impact of Operation at Higher V/f on Aging

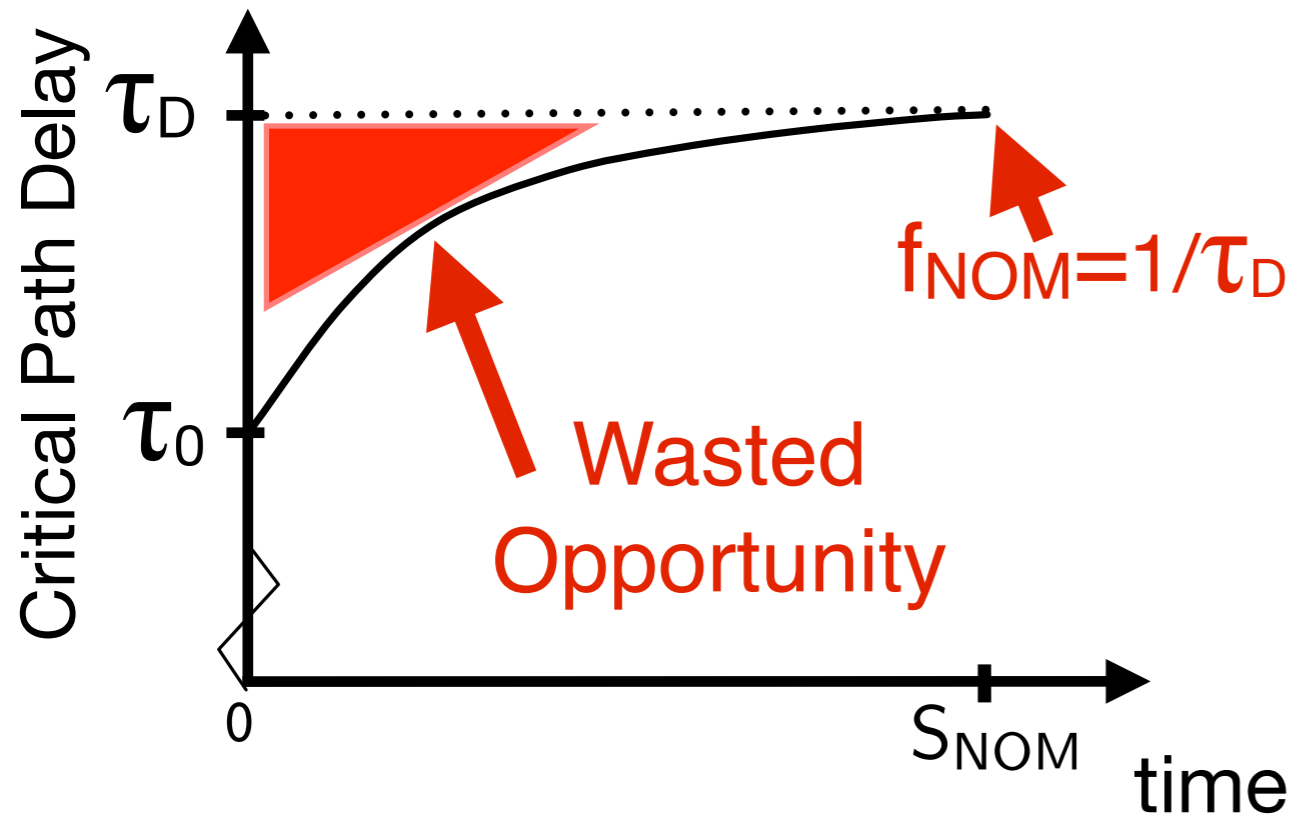


- Higher Vdd:  $Vdd_{OP} \gg Vdd_{NOM}$
- Result: Lower critical path delay; higher aging rate
- Run at constant  $f_{OP} = 1/\tau_{OP}$  until  $S_{SHORT}$ ; then discard



# How to Manage Aging Optimally?

## Contribution: DVS for Aging Management (DVSAM)

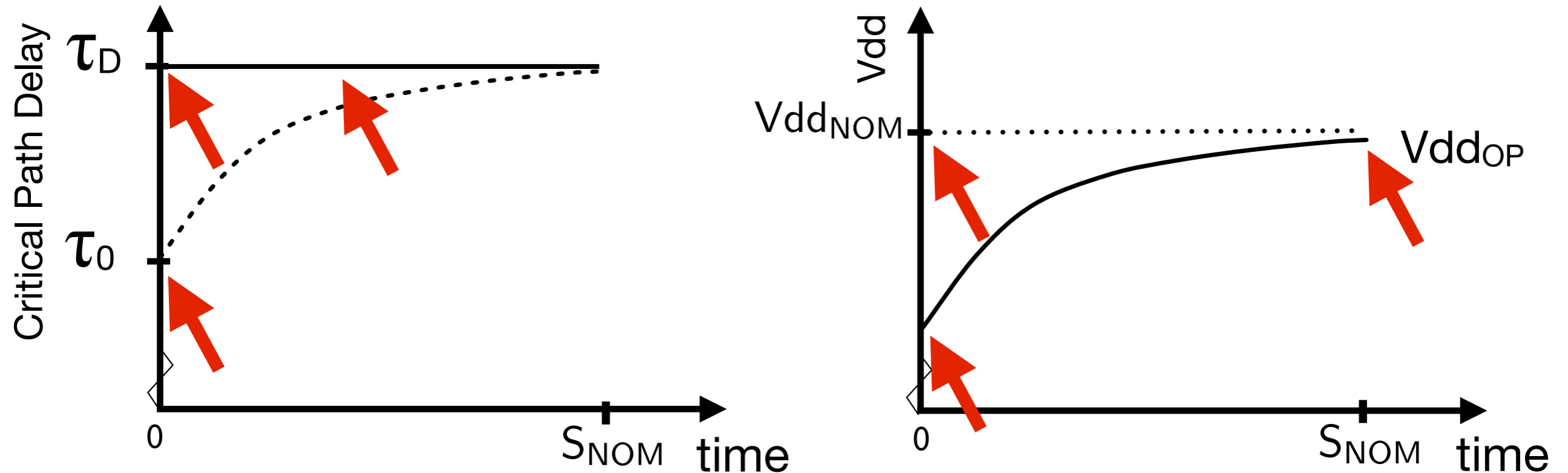


- Change Vdd with time to compensate for critical path degradation
- Enforce minimum Vdd needed for any f-target

- **DVSAM-Pow**: Turn wasted opportunity to power efficiency
- **DVSAM-Perf**: Turn wasted opportunity to higher frequency

# DVSAM-Pow

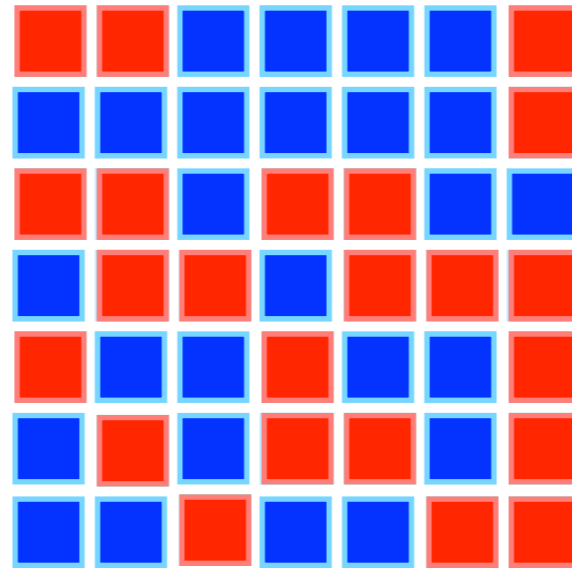
**Idea:** Minimize power consumption at  $f_{\text{NOM}} = 1/\tau_{\text{D}}$



- Critical path delays are kept at  $\tau_{\text{D}}$  until  $S_{\text{NOM}}$ : Run at  $f_{\text{NOM}}$
- Start with low Vdd and increase slowly

# DVSAM-Pow

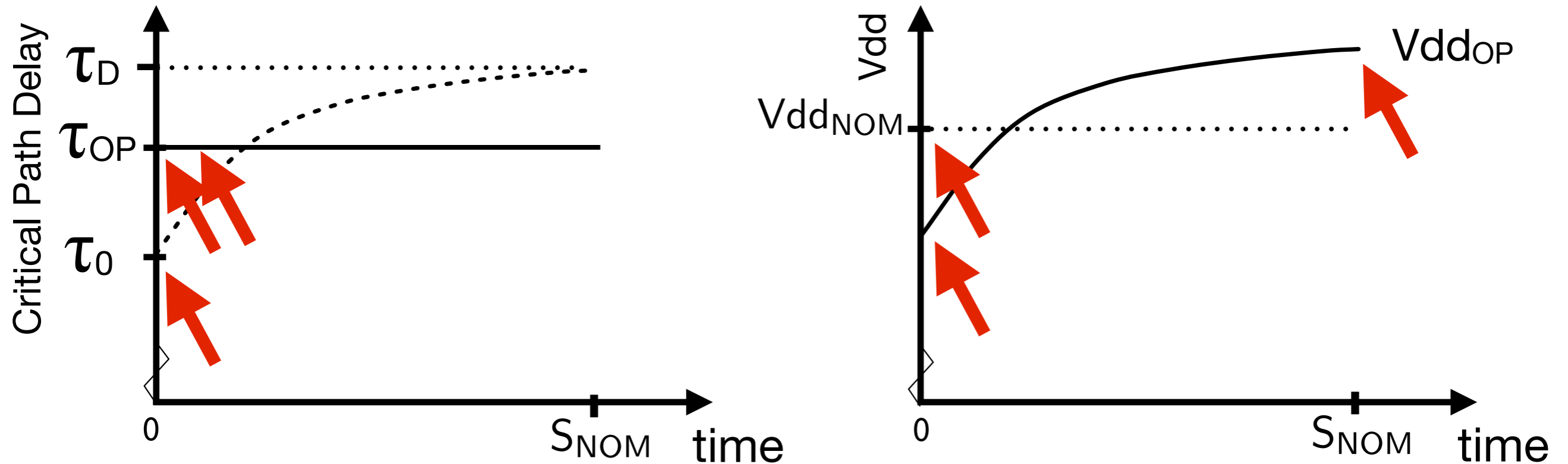
---



- $V_{dd} < V_{dd_{NOM}}$  and  $f = f_{NOM}$  throughout  $S_{NOM}$
- Power savings due to  $V_{dd} < V_{dd_{NOM}}$ 
  - ➔ More cores active for the same P-budget
  - ➔ Increased throughput

# DVSAM-Perf

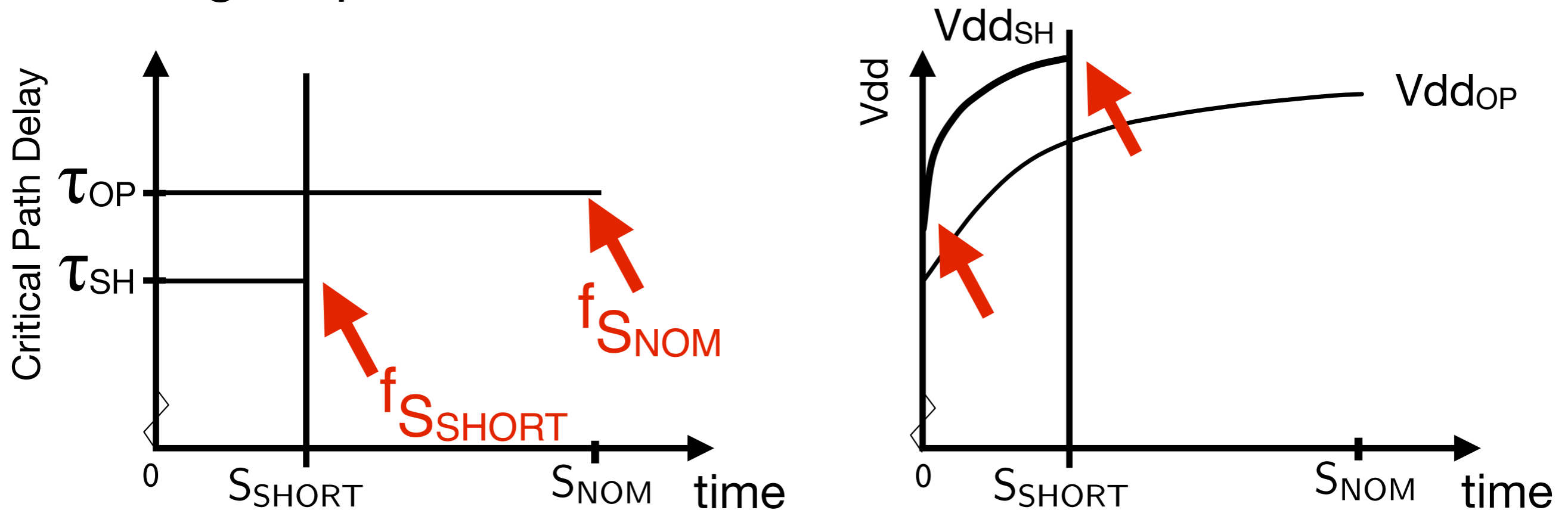
**Idea:** Maximize frequency for the same service life



- Shorter critical path delay  $\tau_{OP}$  until  $S_{NOM}$ : Run at higher  $f = 1 / \tau_{OP}$
- Start with low Vdd and increase rapidly

# DVSAM-Perf for $S_{SHORT}$

**Idea:** Aggressive DVSAM-Perf for a short service life to get even higher performance

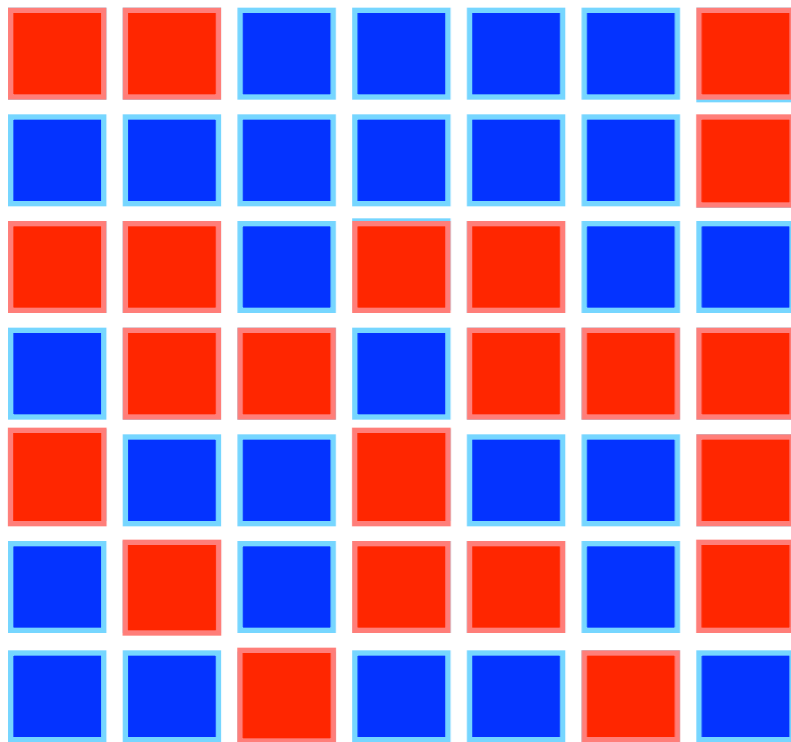


- Even higher frequency than DVSAM-Perf for short service life

# BubbleWrap Environments

---

## Throughput Cores



## Expendable Cores

- Two choices for Throughput Cores
  - Nominal operation
  - Use **DVSAM-Pow** and **expand** the set of throughput cores for the same power budget
- Two choices for Expendable Cores
  - Higher, constant  $V_{dd}$  until  $S_{SHORT}$ ; then discard
  - **DVSAM-Perf** until  $S_{SHORT}$ ; then discard

# Hardware Support?

---

- No change in the core architecture
- Need circuits to measure aging
- Need high-precision DVS
- Clock and power distribution
  - Two separate V/f domains: One for Expendable and one for Throughput Cores



# BubbleWrap Evaluation

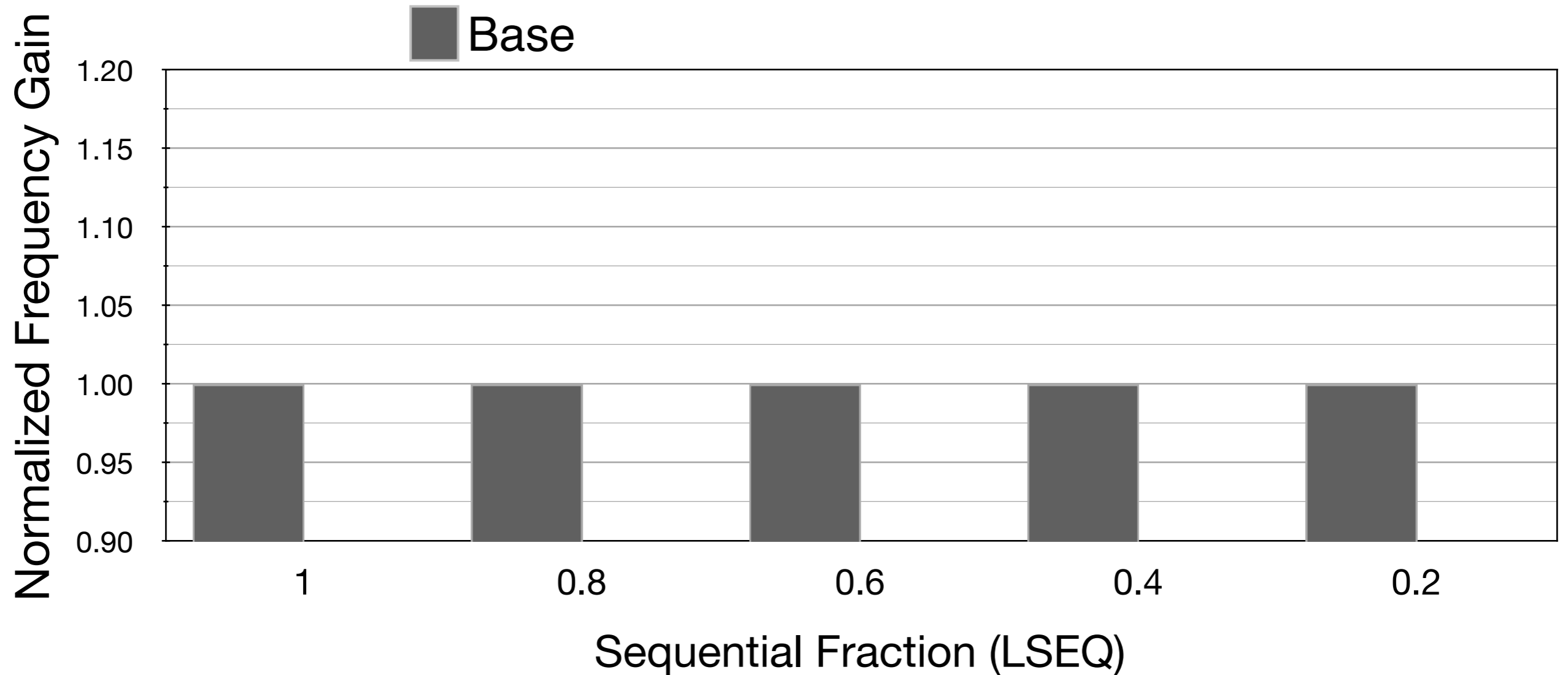
---

- 32 core chip:  $N_T = 16$  Throughput and  $N_E = 16$  Expendable cores
- 22nm high-k metal-gate process
- Multiprogrammed workload synthesized from SPEC2000
- SESC enhanced by a power & thermal model

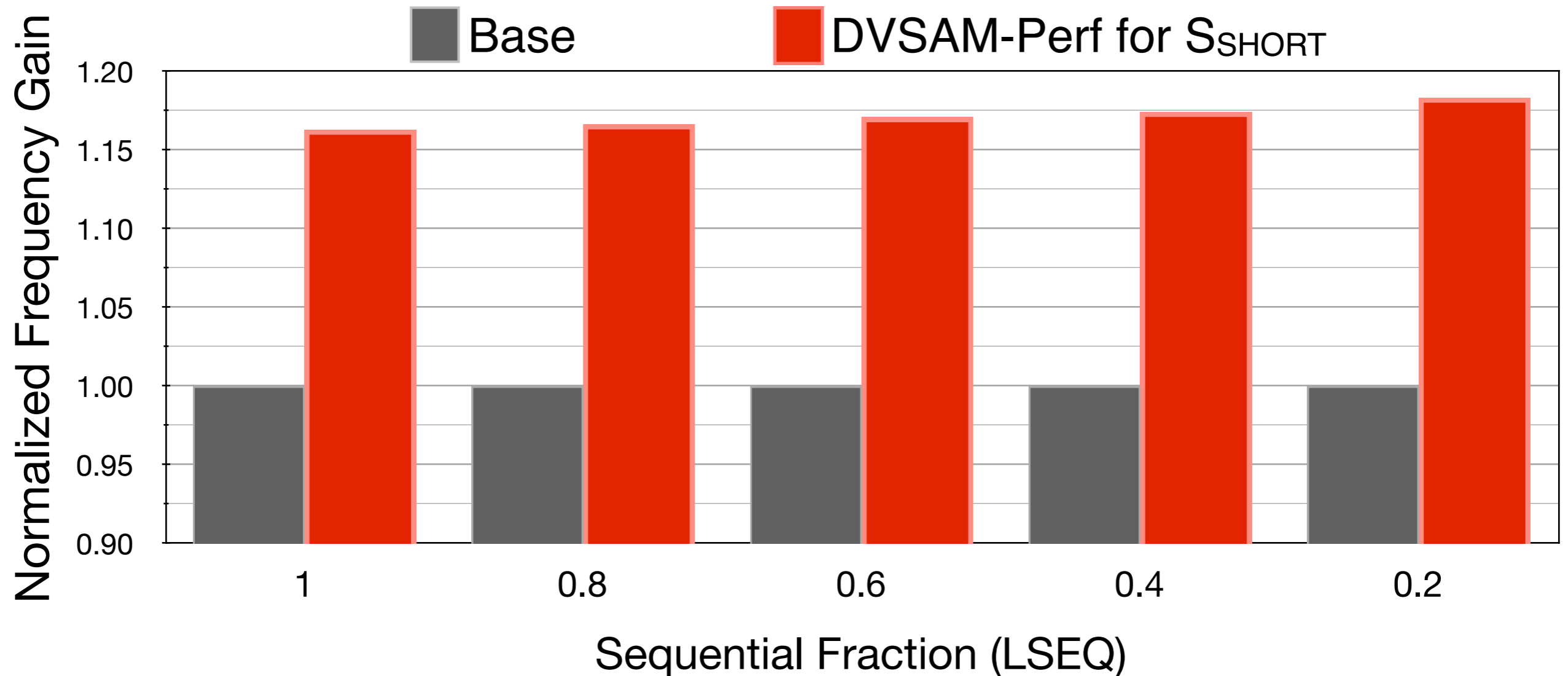




# Frequency Gains of Sequential Section



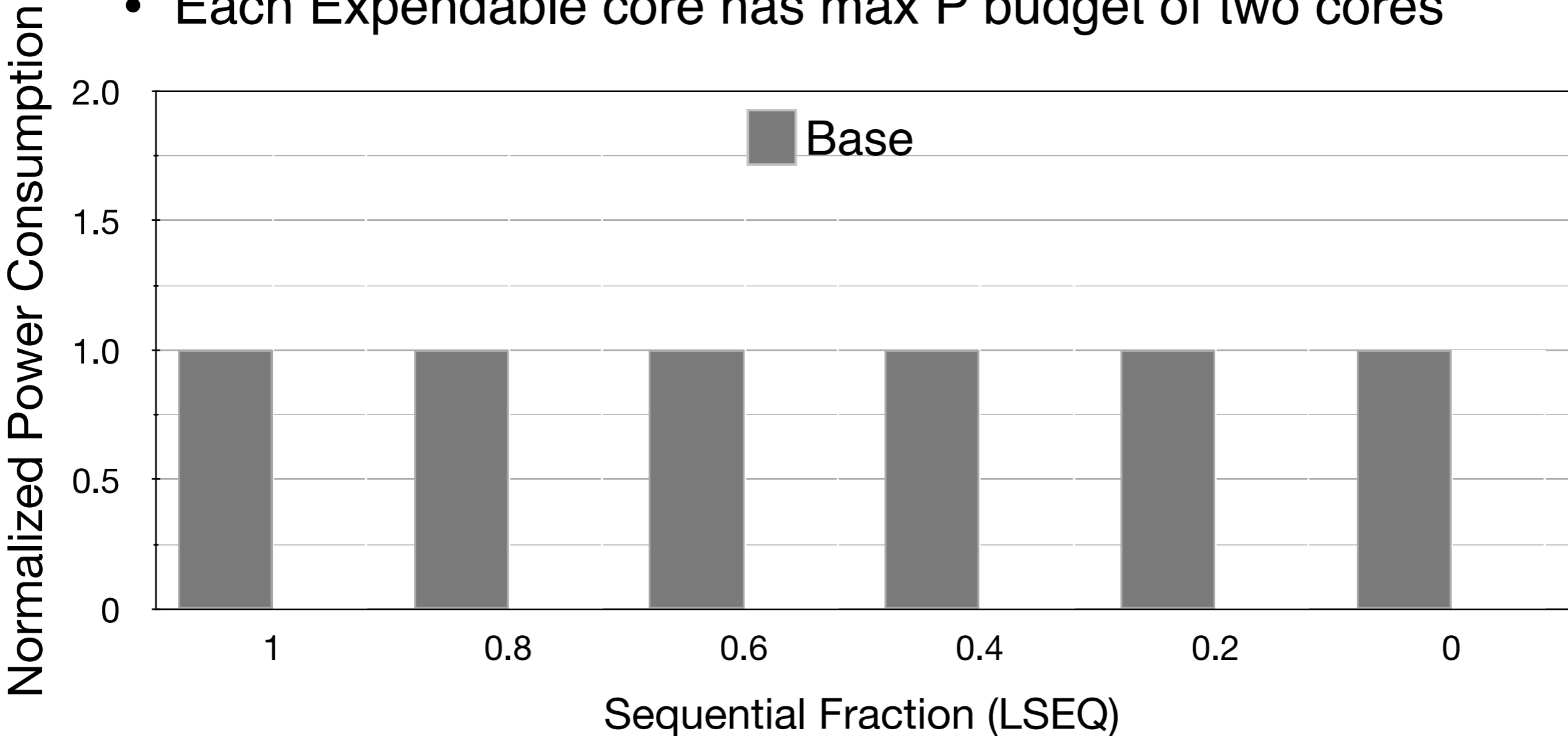
# Frequency Gains of Sequential Section



- Large  $f$  gains are feasible
- $f$  increases with smaller sequential section
- For DVSAM-Perf, each expendable core runs for  $L_{SEQ}/N_E \times S_{NOM}$

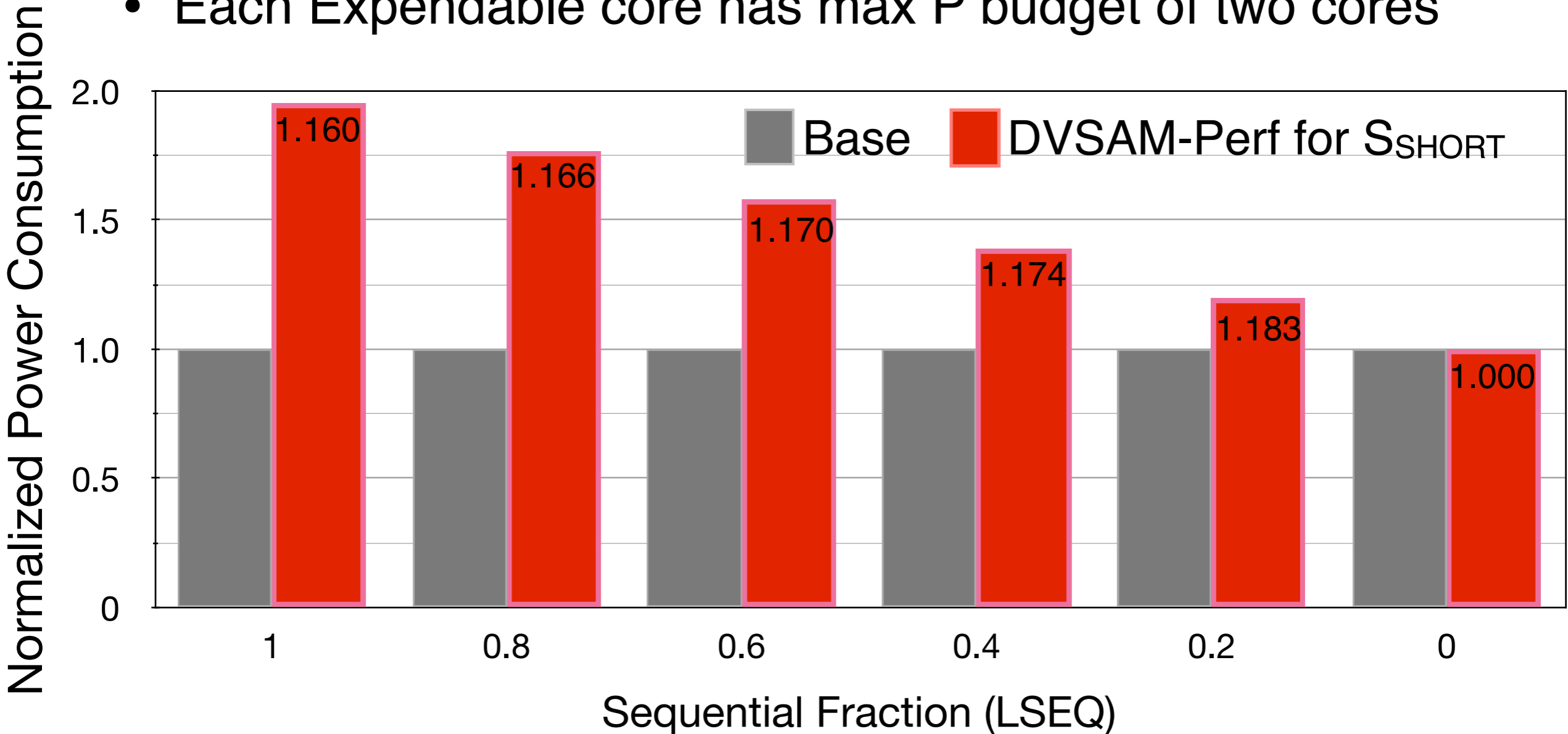
# Power Consumption of Sequential Section

- Each Expendable core has max P budget of two cores



# Power Consumption of Sequential Section

- Each Expendable core has max P budget of two cores

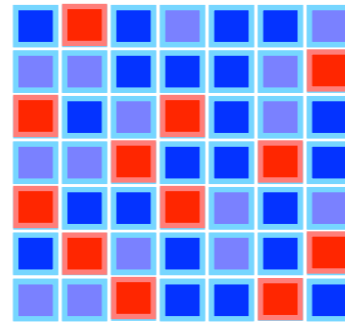


- Tolerable power cost for the frequency gains

# Conclusion

---

The BubbleWrap Many-Core:  
Exploiting dormant cores for sequential acceleration



- Simple homogeneous design
- No architectural or software changes
- Improves sequential and parallel performance
  - Fully sequential applications at 16% higher f
  - Fully parallel applications at 30% higher throughput

# The BubbleWrap Many-Core: Popping Cores for Sequential Acceleration

Ulya Karpuzcu, Brian Greskamp, Josep Torrellas  
University of Illinois

<http://iacoma.cs.uiuc.edu/>

# Also in the Paper

---

- Details of DVSAM-Pow operation; translating power savings to throughput performance
- Discussion and Evaluation of Different BubbleWrap environments based on DVSAM deployment
- A hardware controller to manage BubbleWrap operation

# Issues

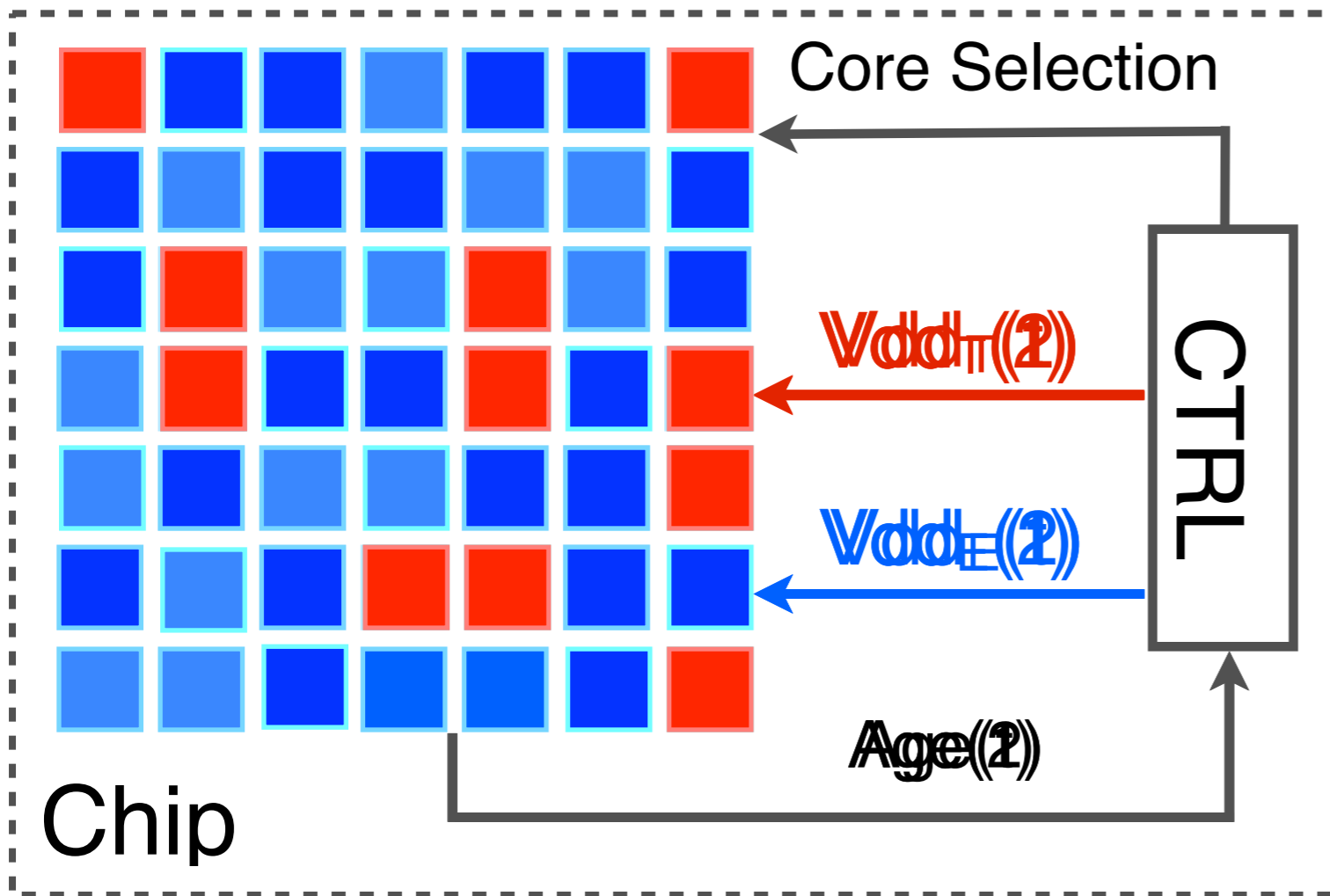
---

- Other uses of die area taken by dormant cores?
  - More on-chip storage
- Alternative architectures for energy efficiency?
  - Heterogeneous systems
  - Accelerator architectures
  - Near- $V_{th}$  operation



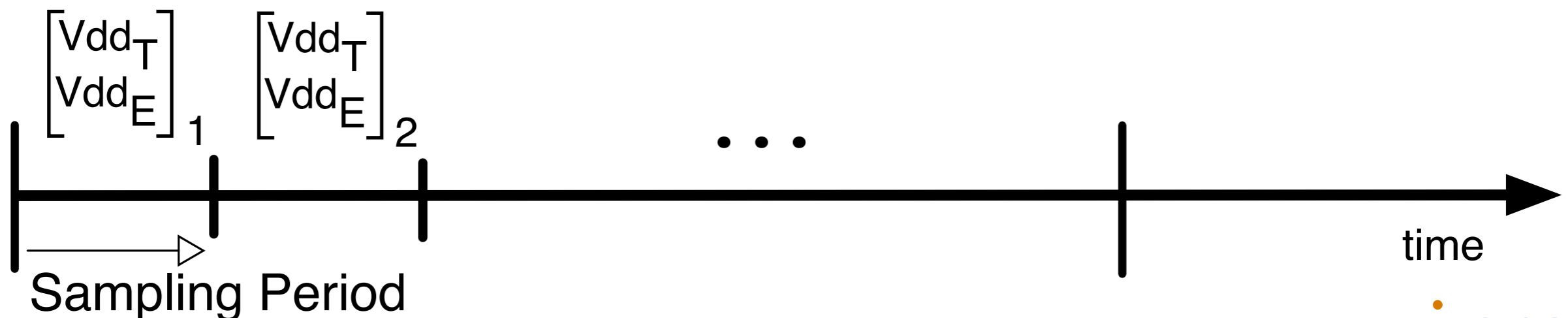


# The Controller

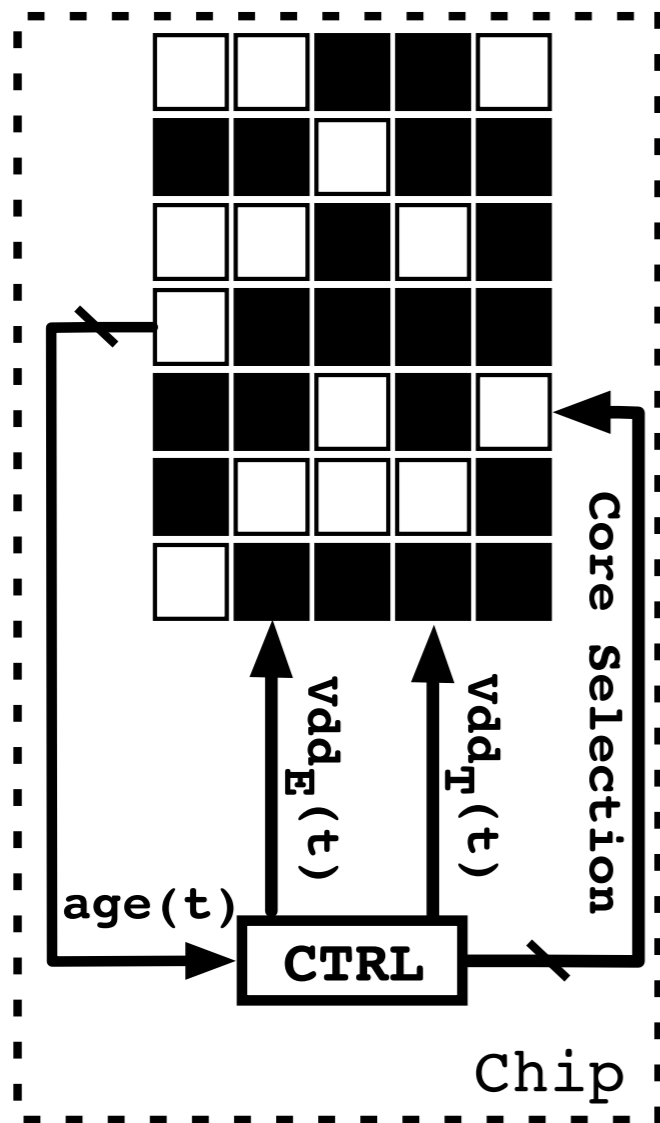


At each scheduling point

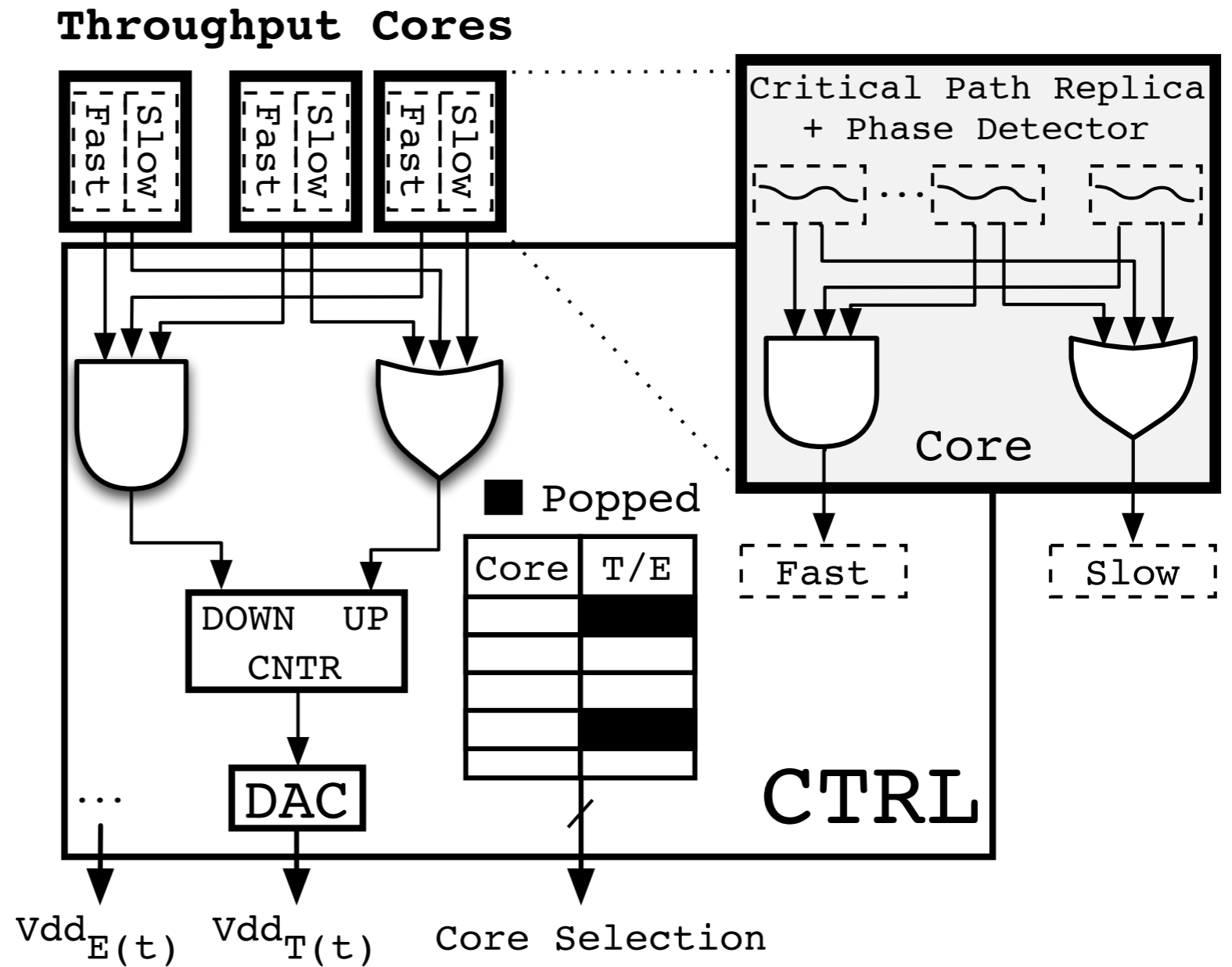
- $N_T$  T-threads
- $N_E$  E-threads
- Resource virtualization
  - Popped cores
  - Core renaming
  - Connection Map



# The Controller

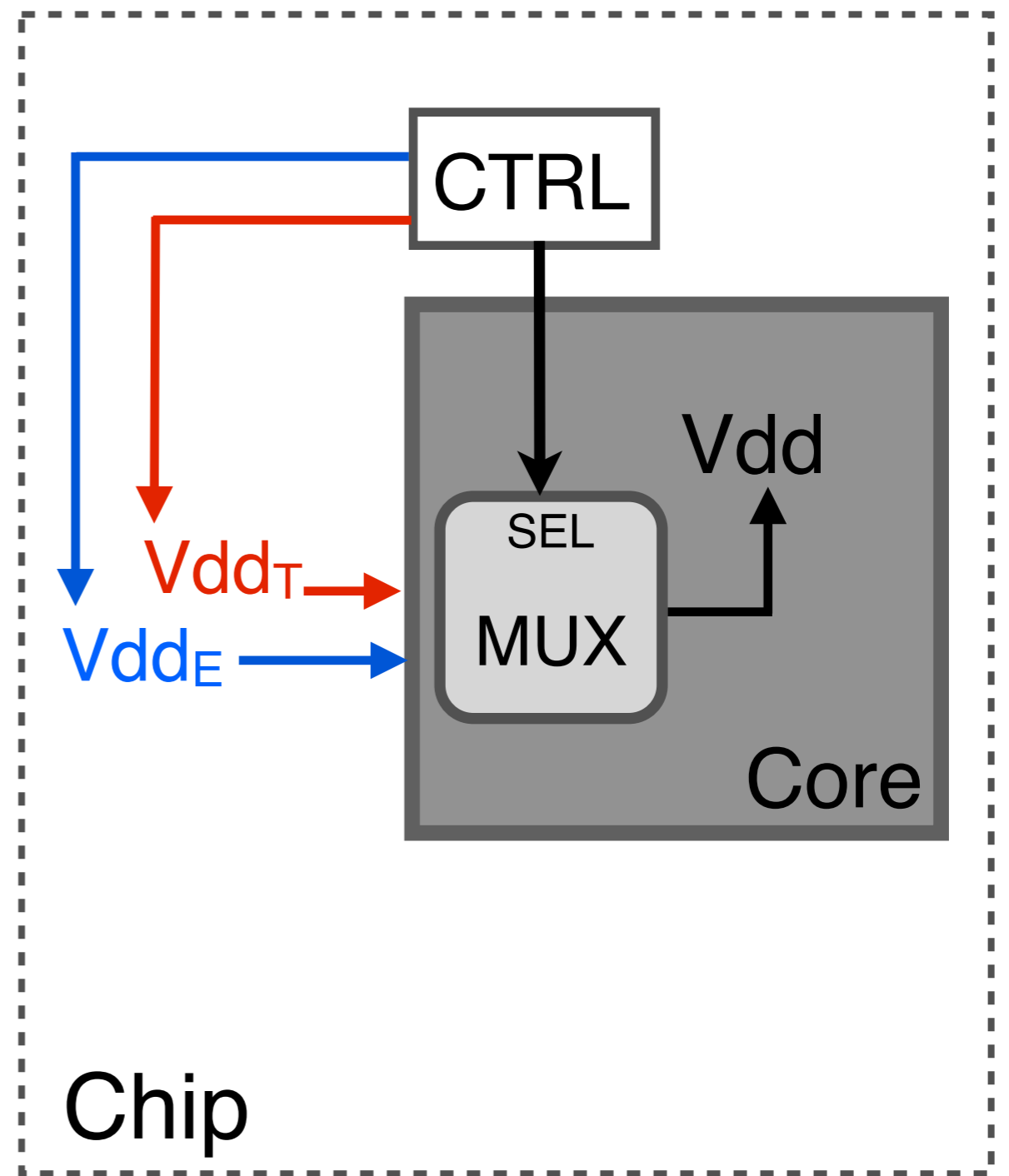
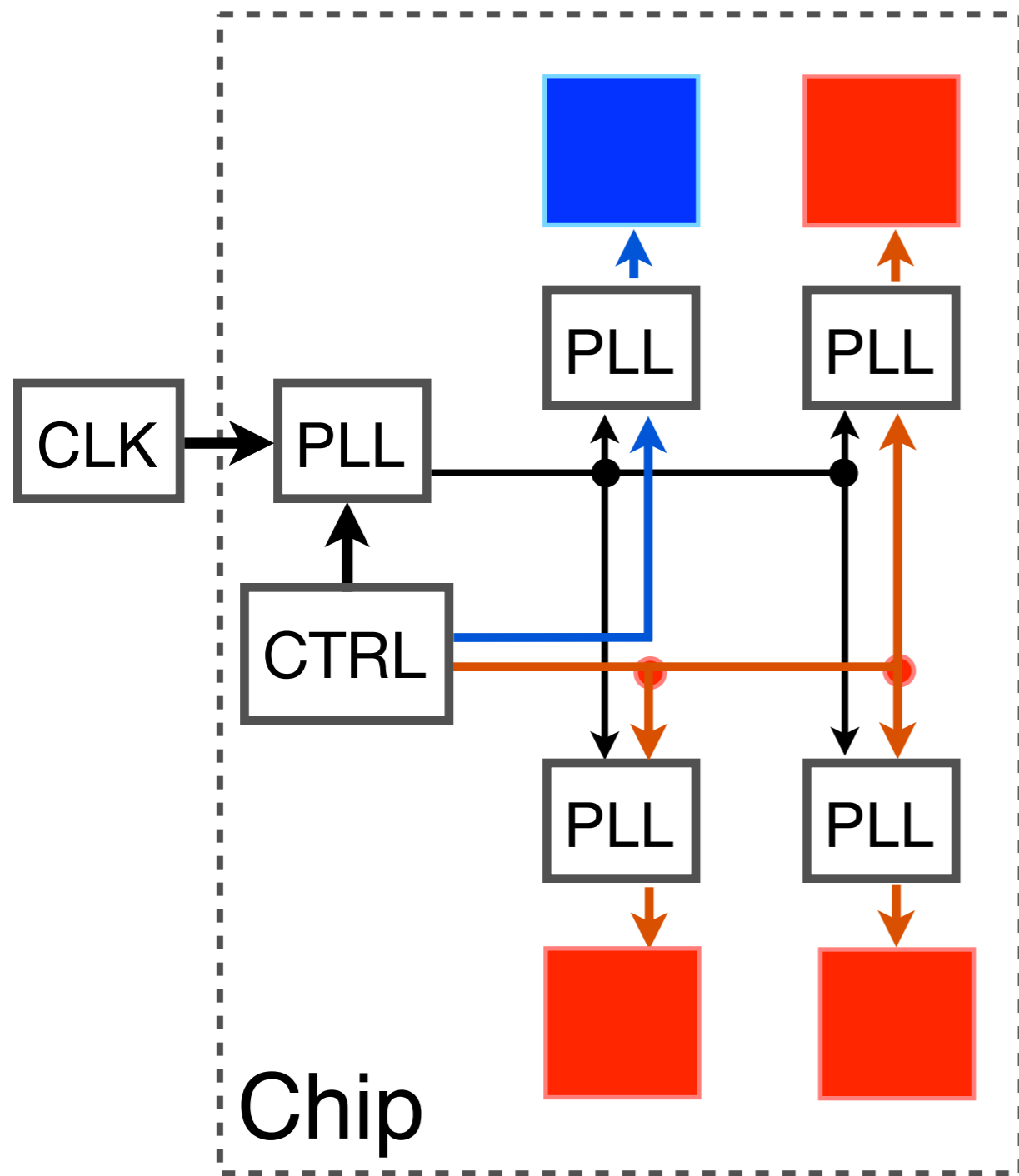


(a)

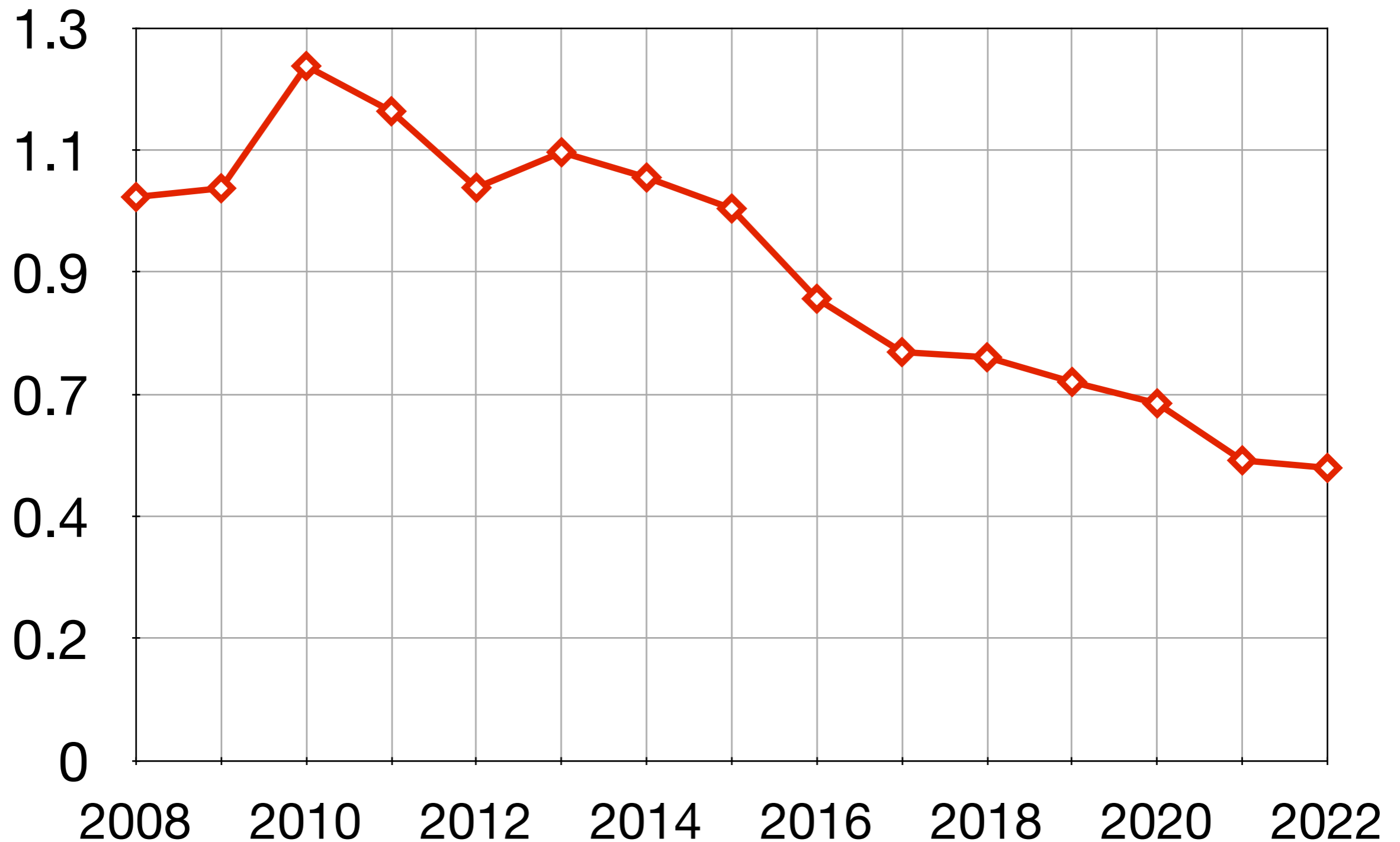


(b)

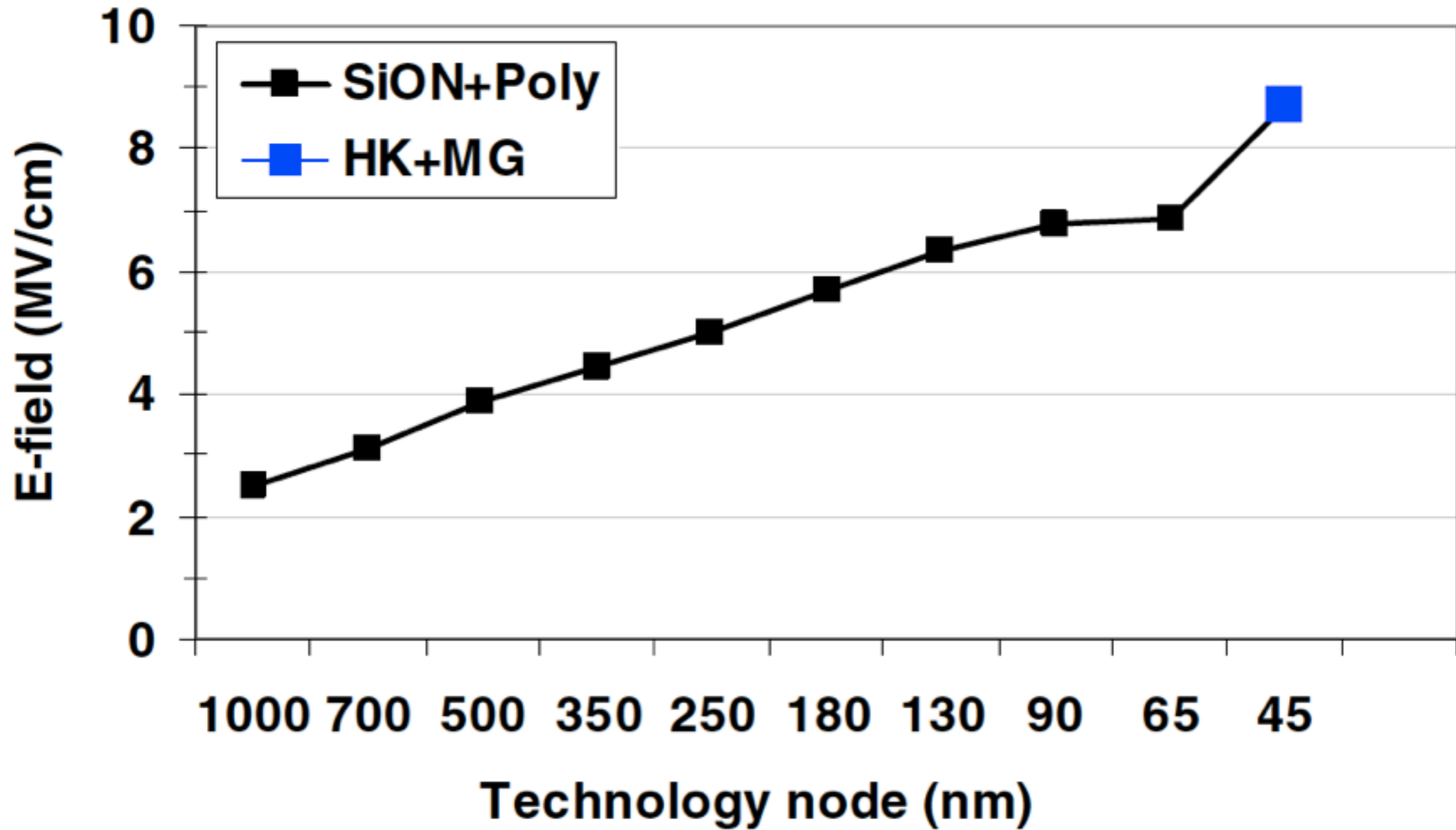
# Clock and Power Distribution



# Energy per Switching



## Gate Dielectric Electric Field



[Bergstrom08]

