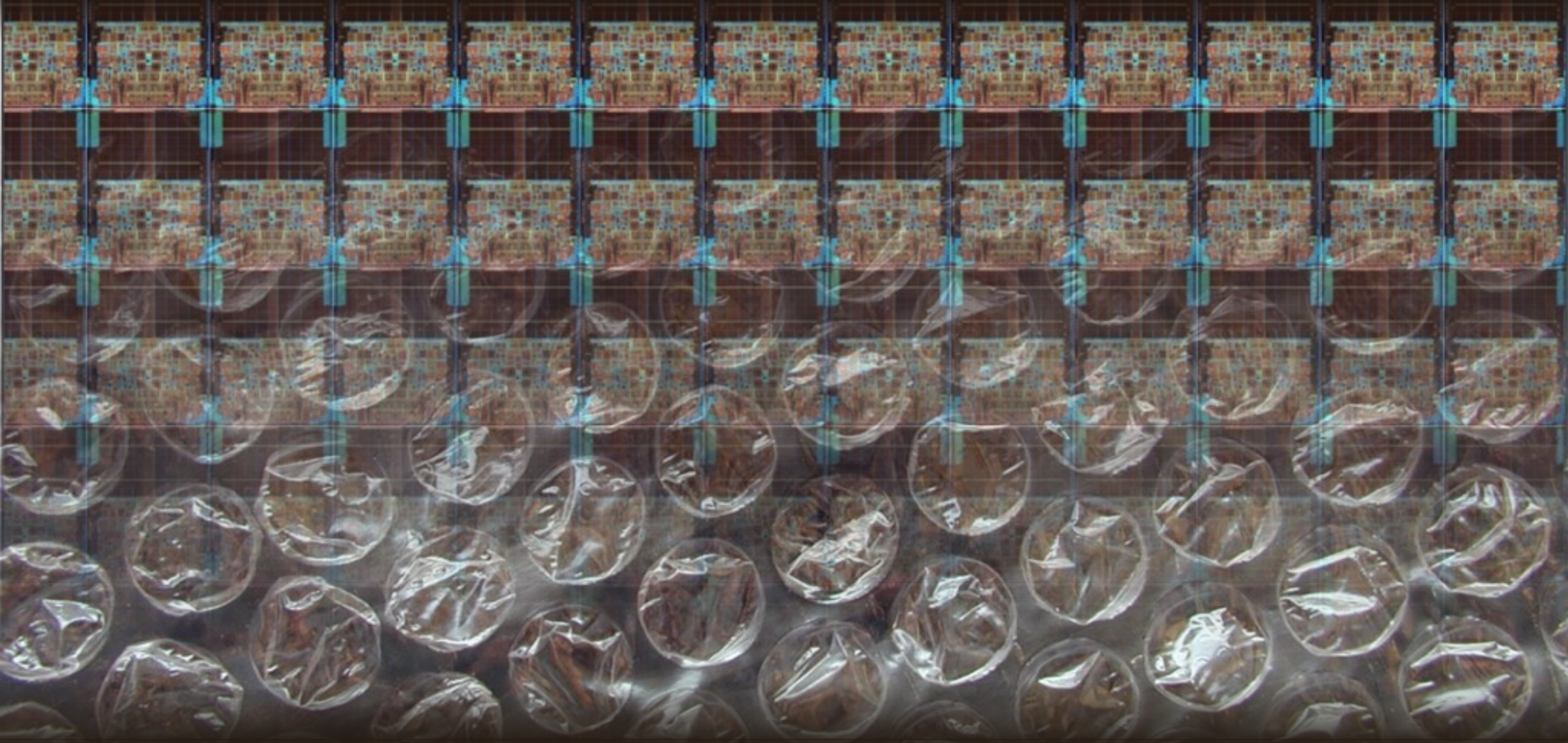
BubbleWrap: Popping CMP Cores for Sequential Acceleration



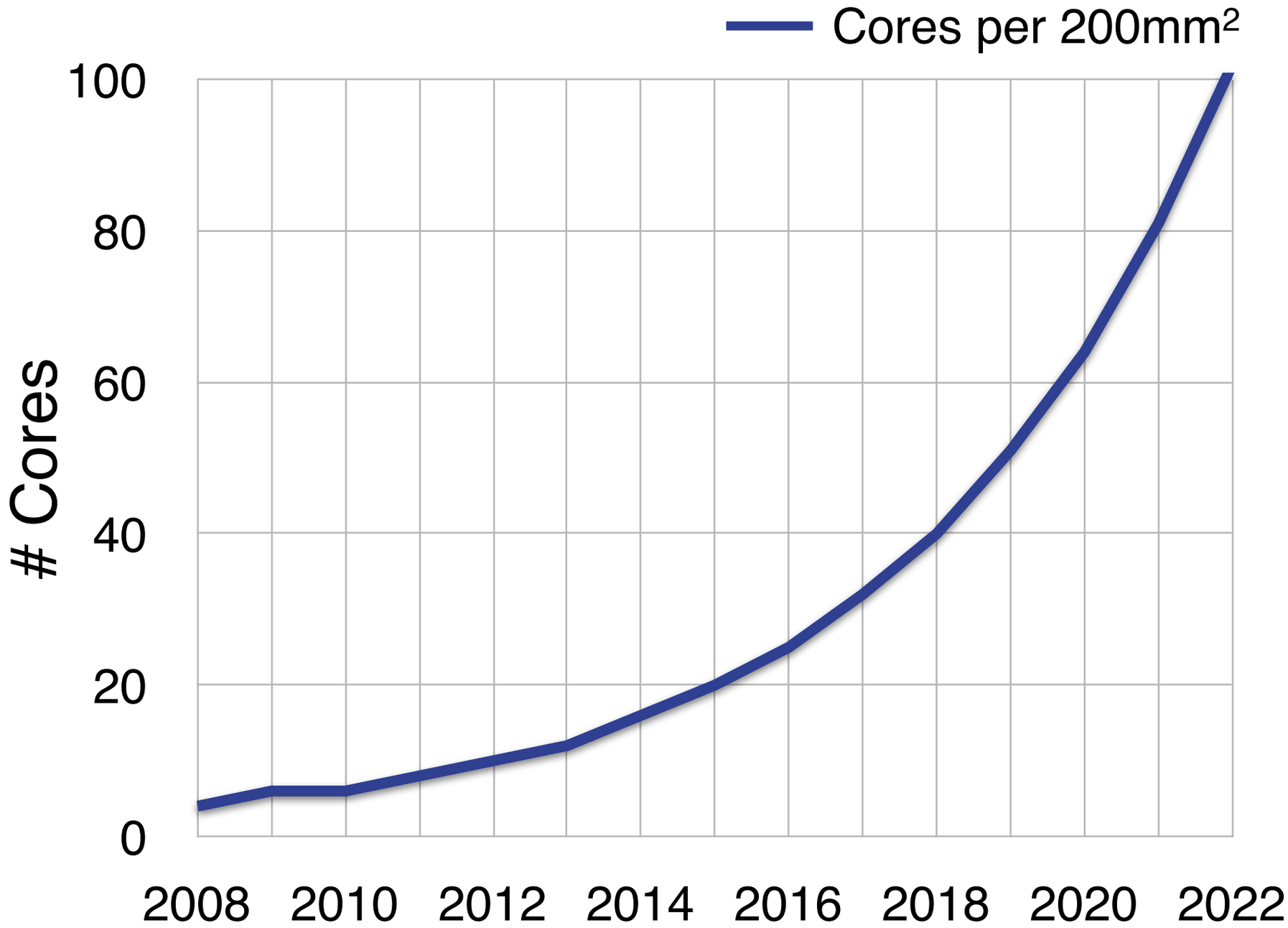
Brian Greskamp, Ulya Karpuzcu, and Josep Torrellas

University of Illinois

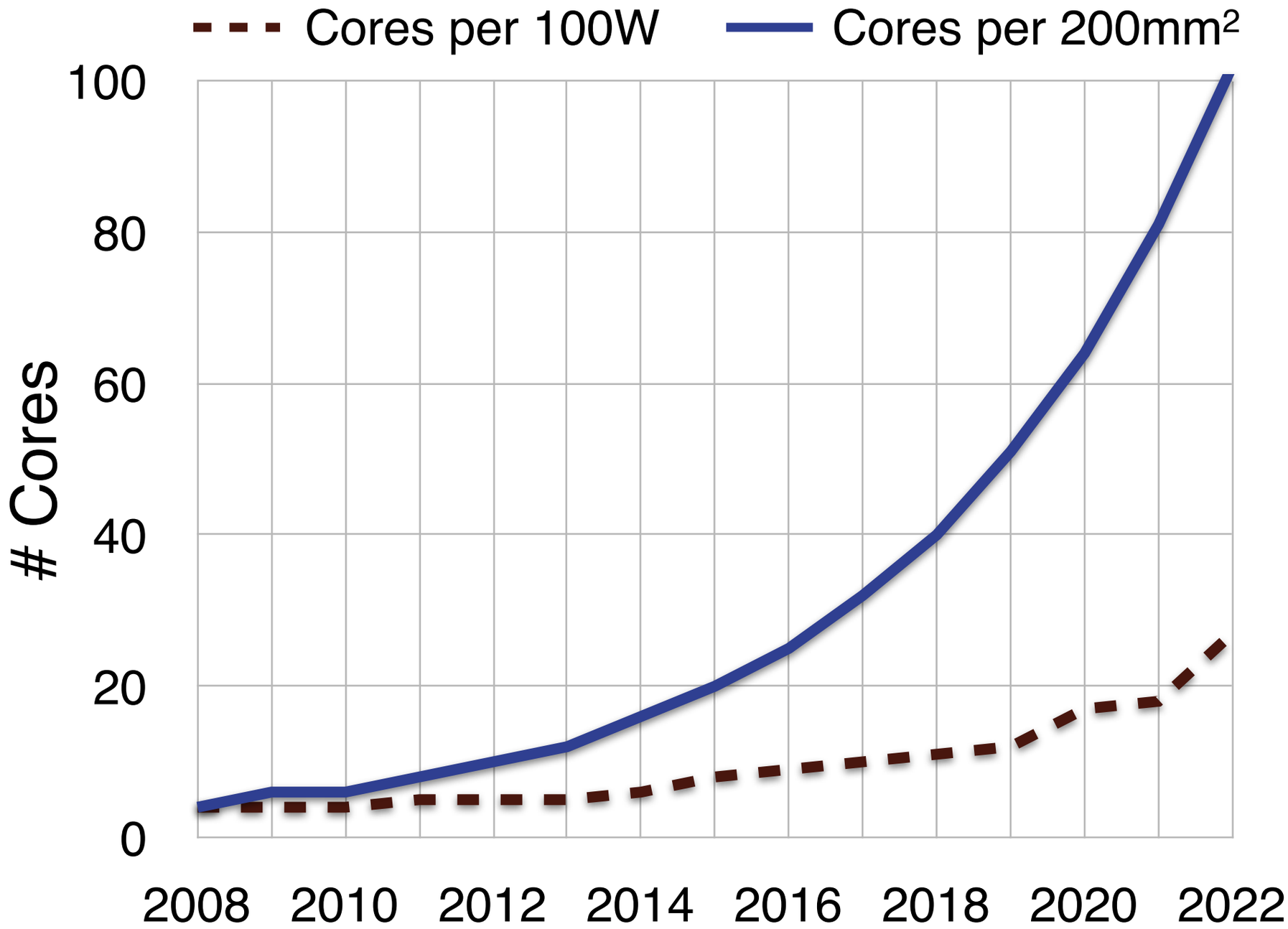
http://iacoma.cs.uiuc.edu/

i-acoma group

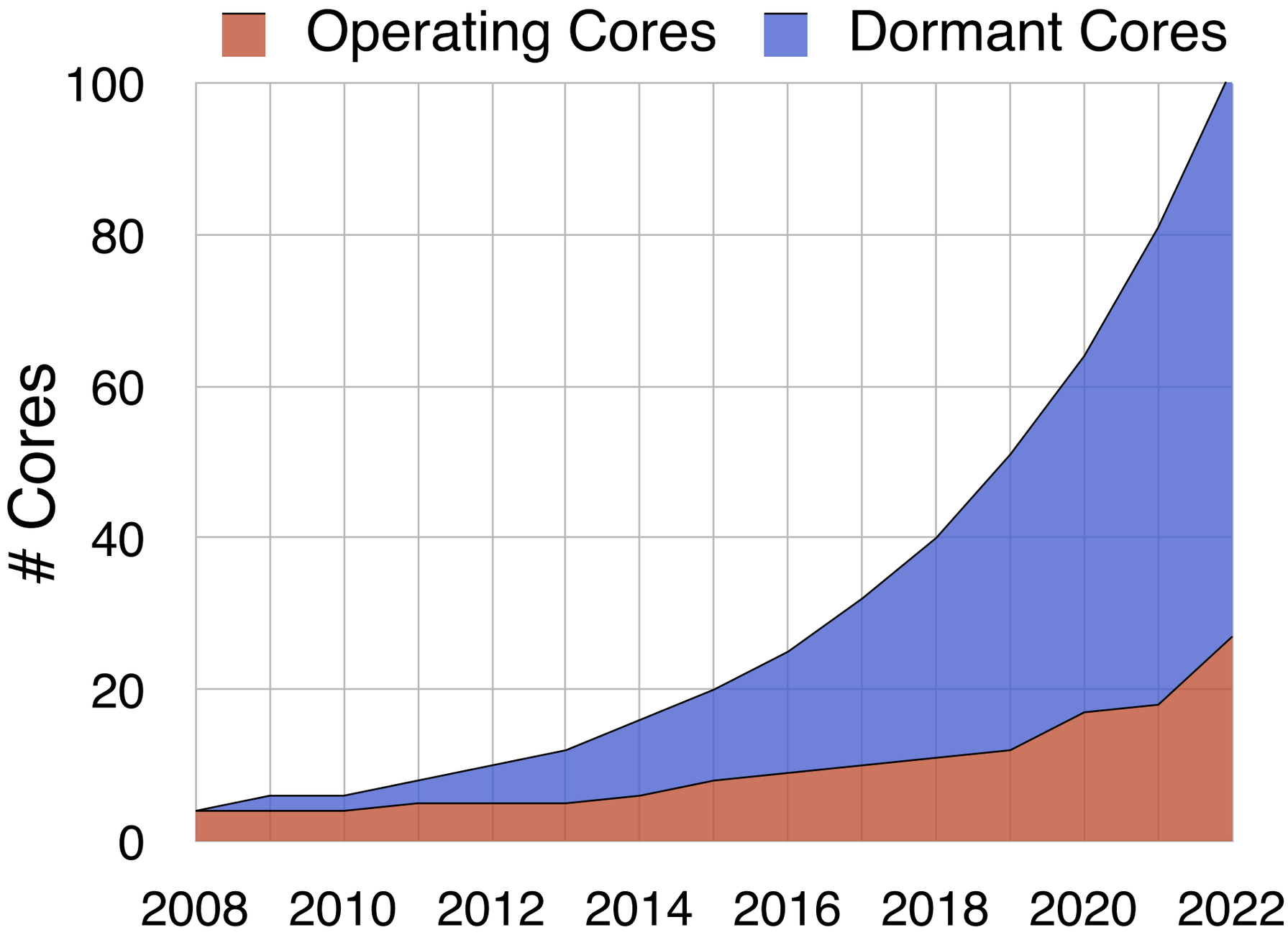
ITRS Power Scaling



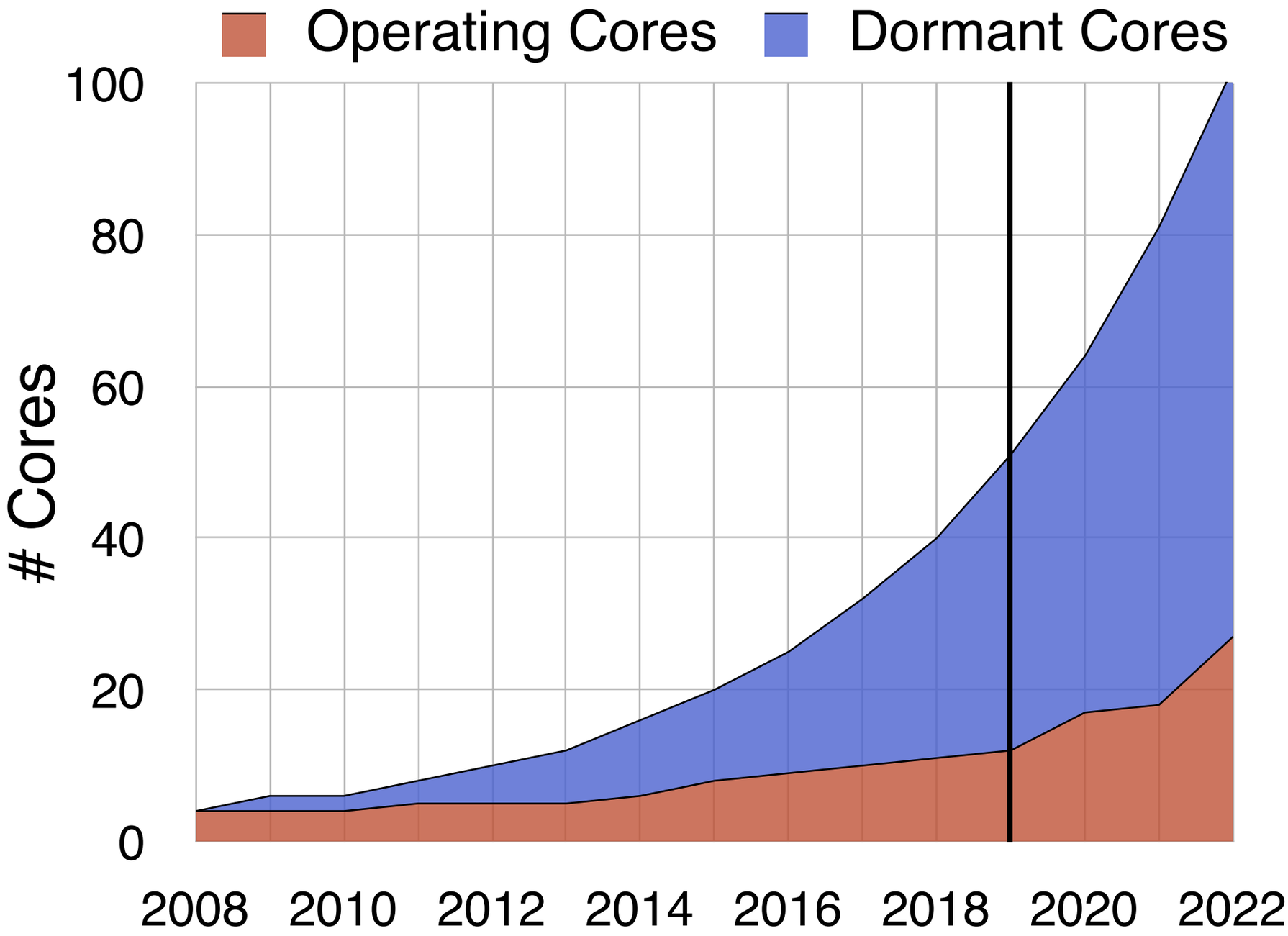
ITRS Power Scaling



ITRS Power Scaling

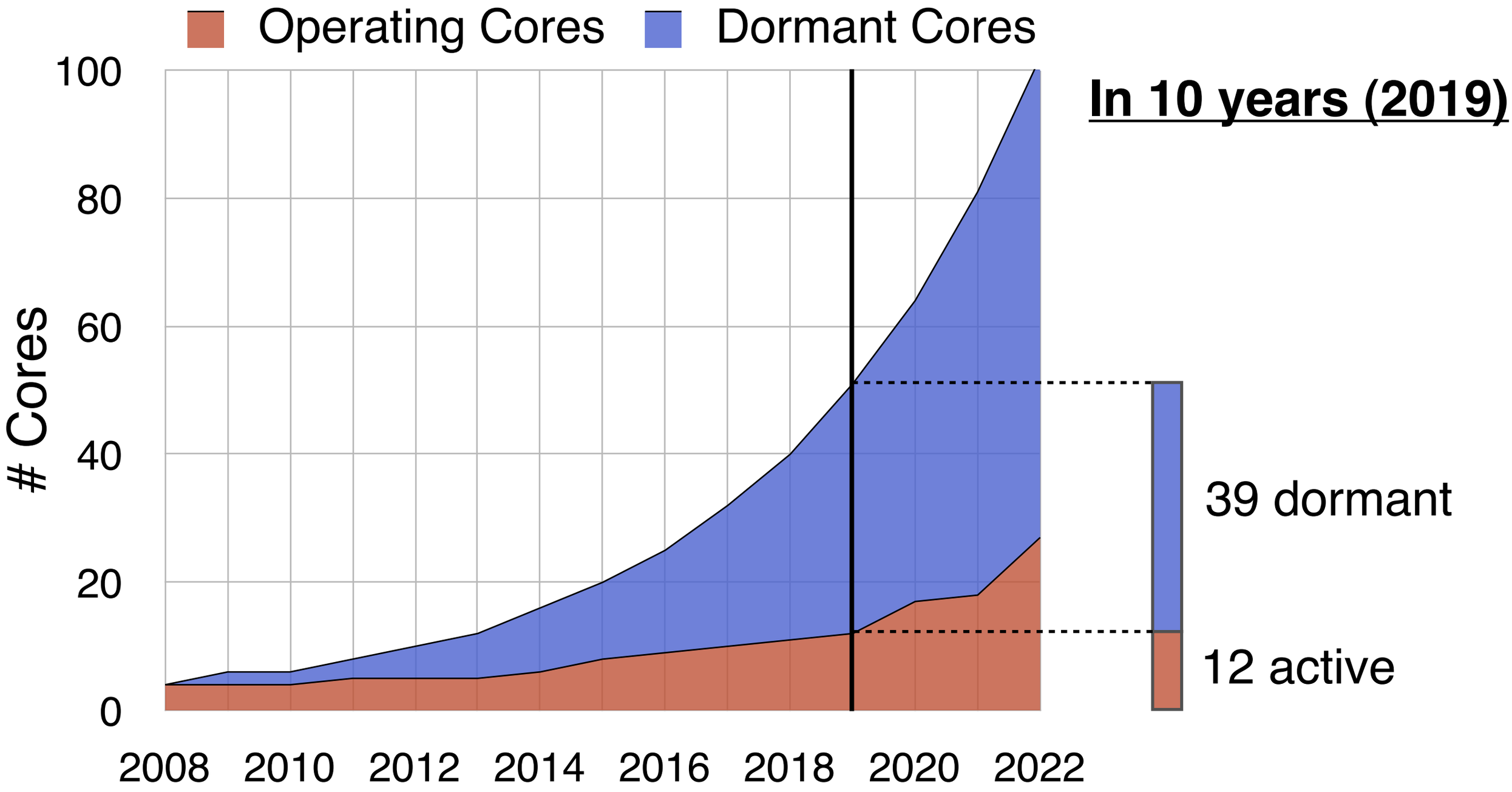


ITRS Power Scaling

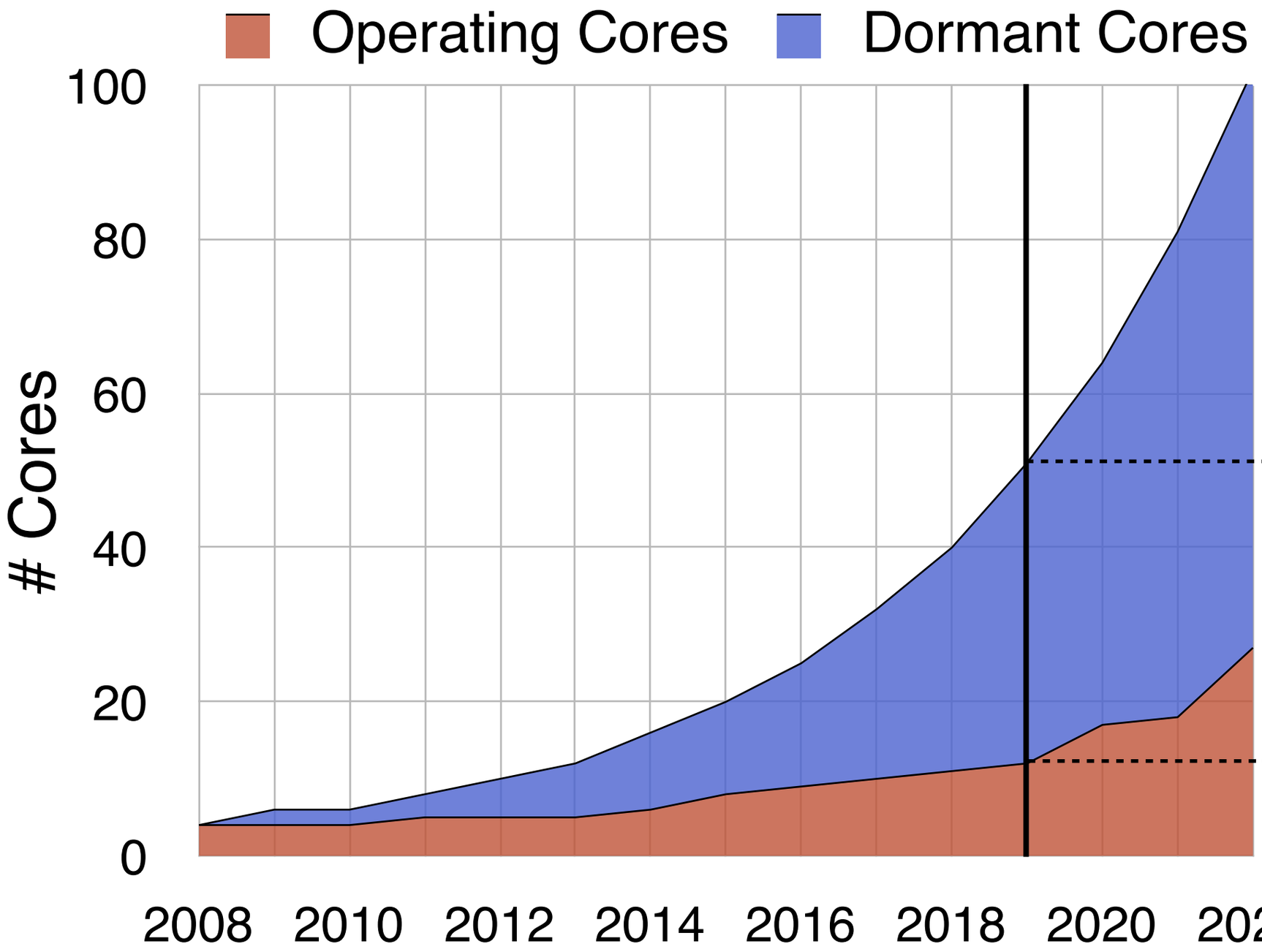


In 10 years (2019)

ITRS Power Scaling



ITRS Power Scaling



In 10 years (2019)

Only 1/4 of cores active at any time

39 dormant

12 active

Exploiting Dormant Cores

Exploiting Dormant Cores

- Chips are designed to last ~7 years

Exploiting Dormant Cores

- Chips are designed to last ~7 years

➡ Must operate at conservative *V*, *f*

Exploiting Dormant Cores

- Chips are designed to last ~7 years

  ➡ Must operate at conservative *V*, *f*

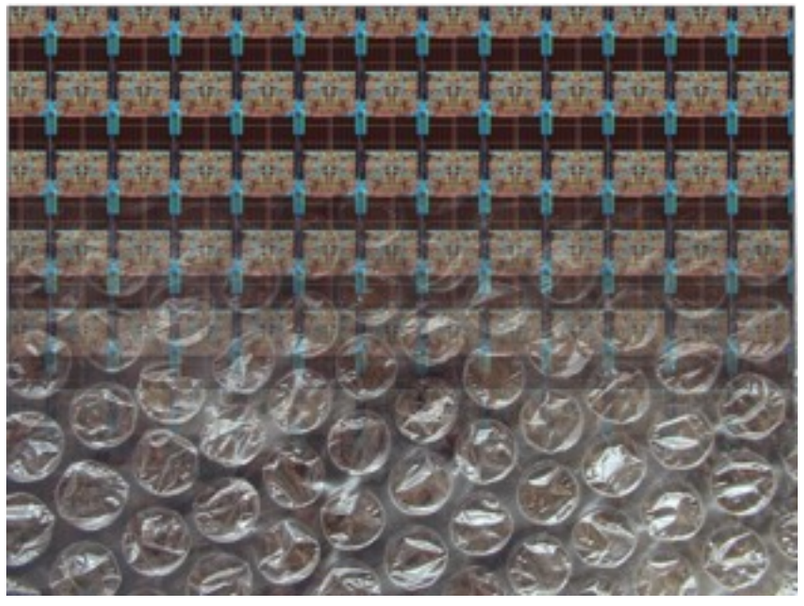
- But we will have plenty of spare cores

Exploiting Dormant Cores

- Chips are designed to last ~7 years

  ➡ Must operate at conservative *V*, *f*

- But we will have plenty of spare cores
  - Pop them like BubbleWrap

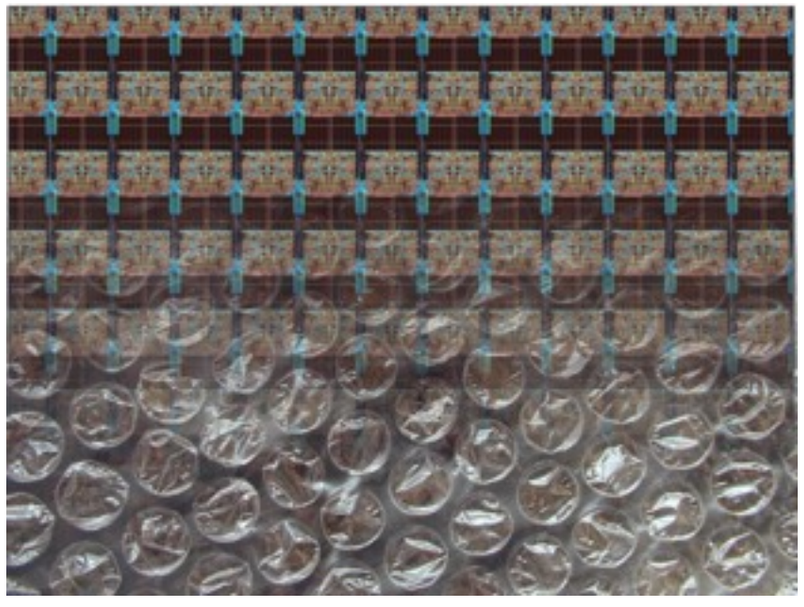


Exploiting Dormant Cores

- Chips are designed to last ~7 years

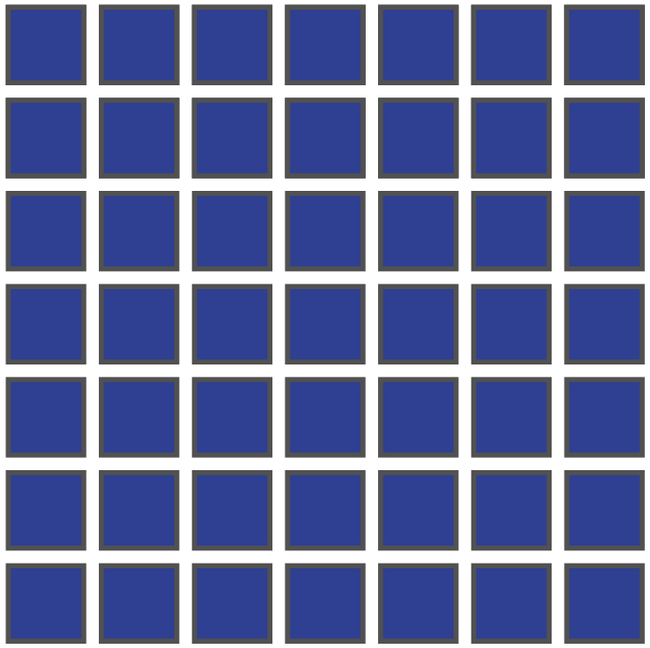
  ➡ Must operate at conservative *V*, *f*

- But we will have plenty of spare cores
  - Pop them like BubbleWrap
  - Some run at high *V*, *f* for just a few months



A BubbleWrap CMP

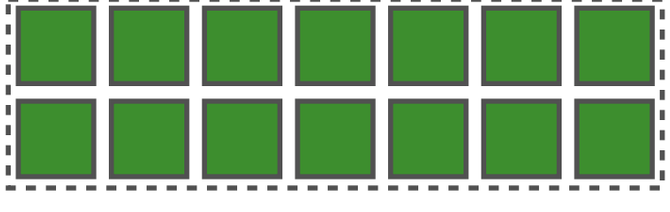
- Start with simple, homogeneous CMP

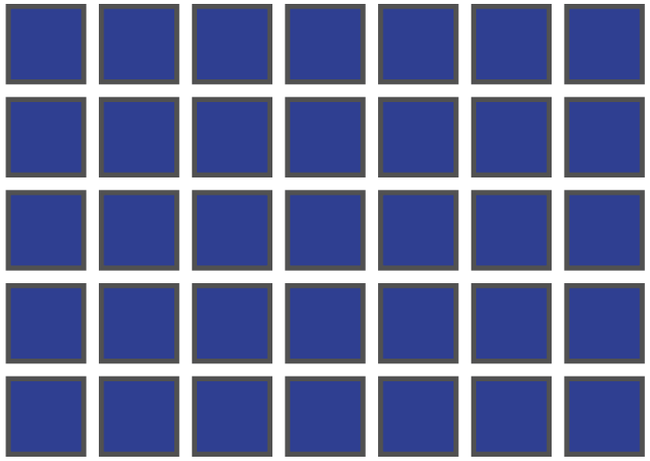


12nm CMP

A BubbleWrap CMP

Efficient, *Select* Cores

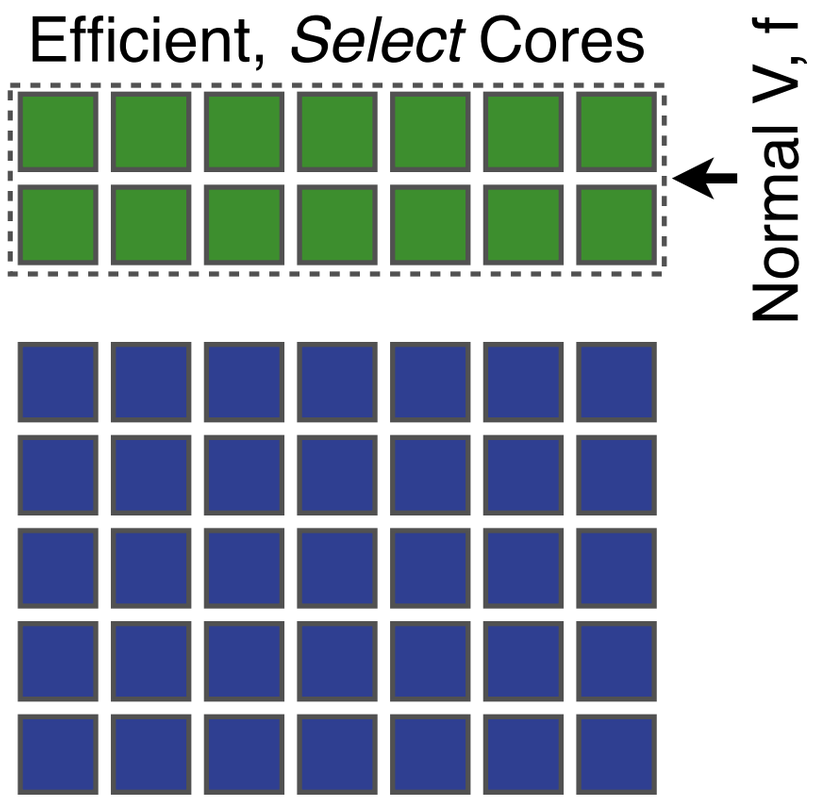




12nm CMP

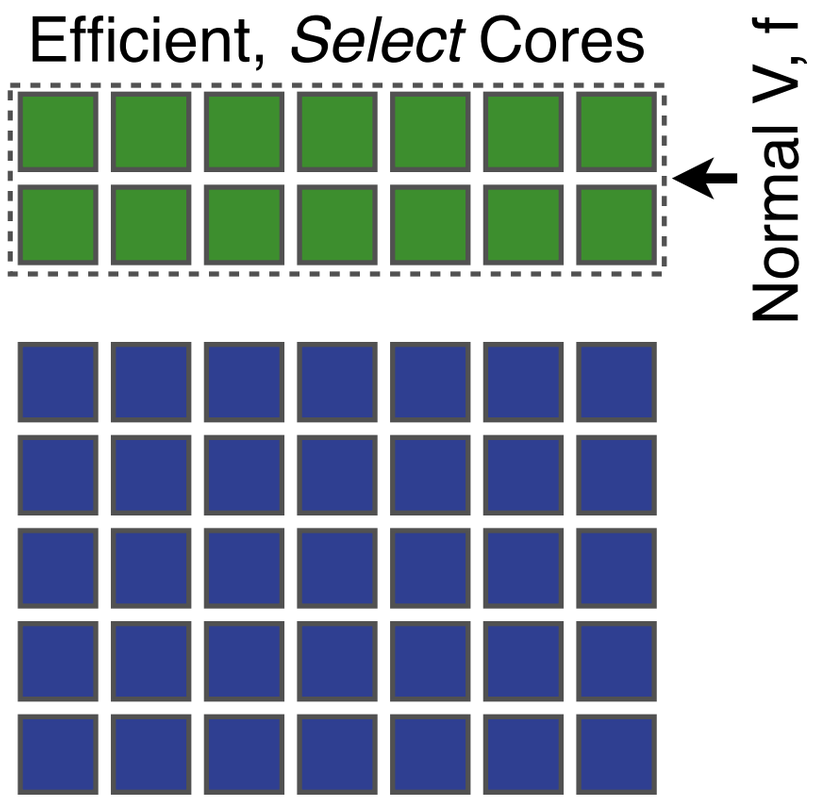
- Start with simple, homogeneous CMP
- Select most efficient cores

A BubbleWrap CMP



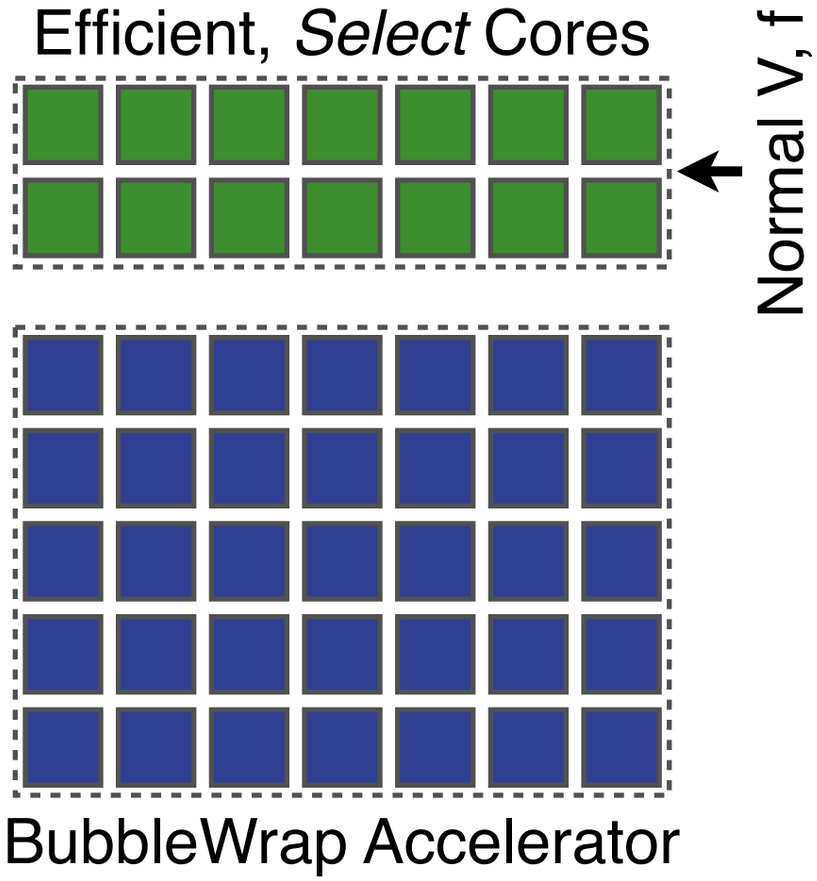
- Start with simple, homogeneous CMP
- Select most efficient cores
  - Run at nominal *V*, *f*

A BubbleWrap CMP



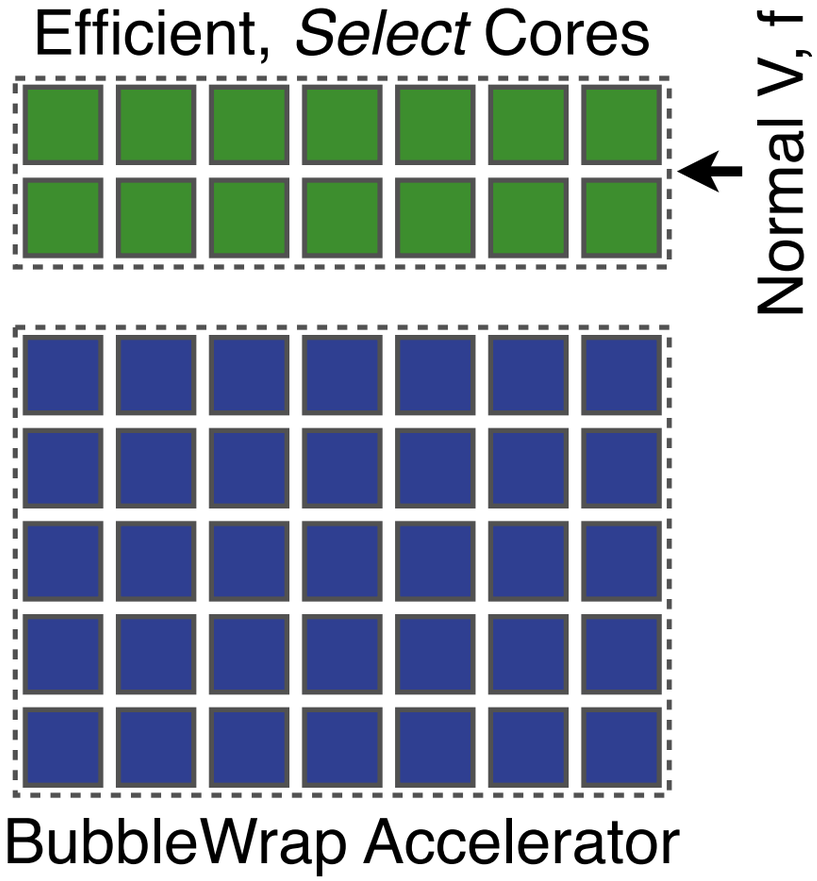
- Start with simple, homogeneous CMP
- Select most efficient cores
  - Run at nominal *V*, *f*
- Use for parallel phases

A BubbleWrap CMP



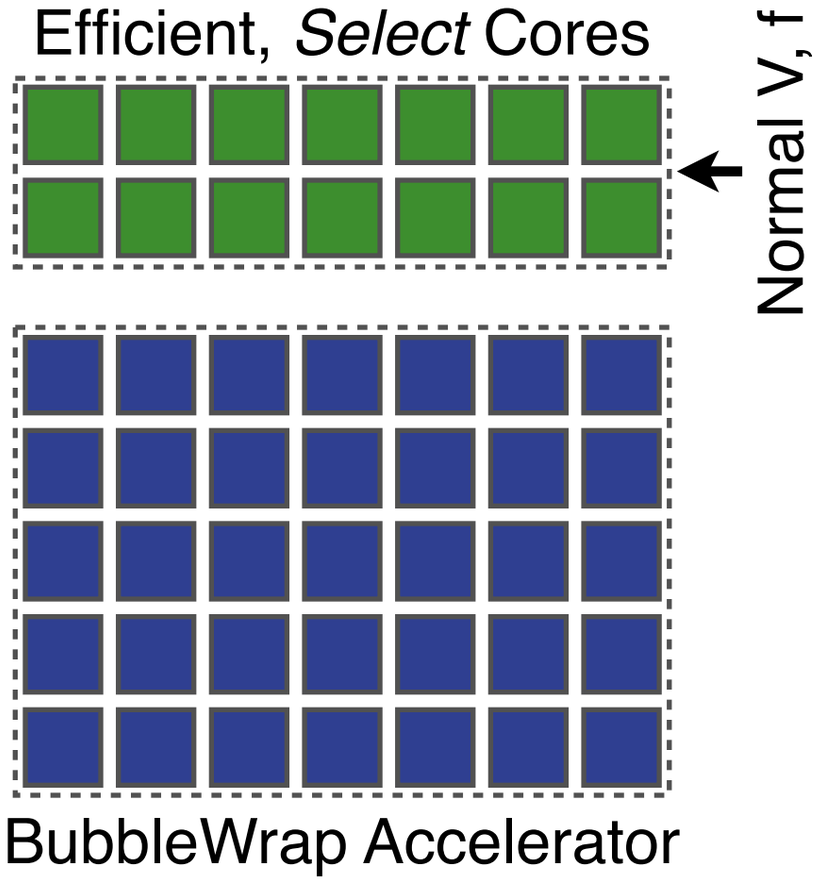
- Start with simple, homogeneous CMP
- Select most efficient cores
  - Run at nominal *V*, *f*
  - Use for parallel phases
- Rest of cores are BubbleWrap

A BubbleWrap CMP



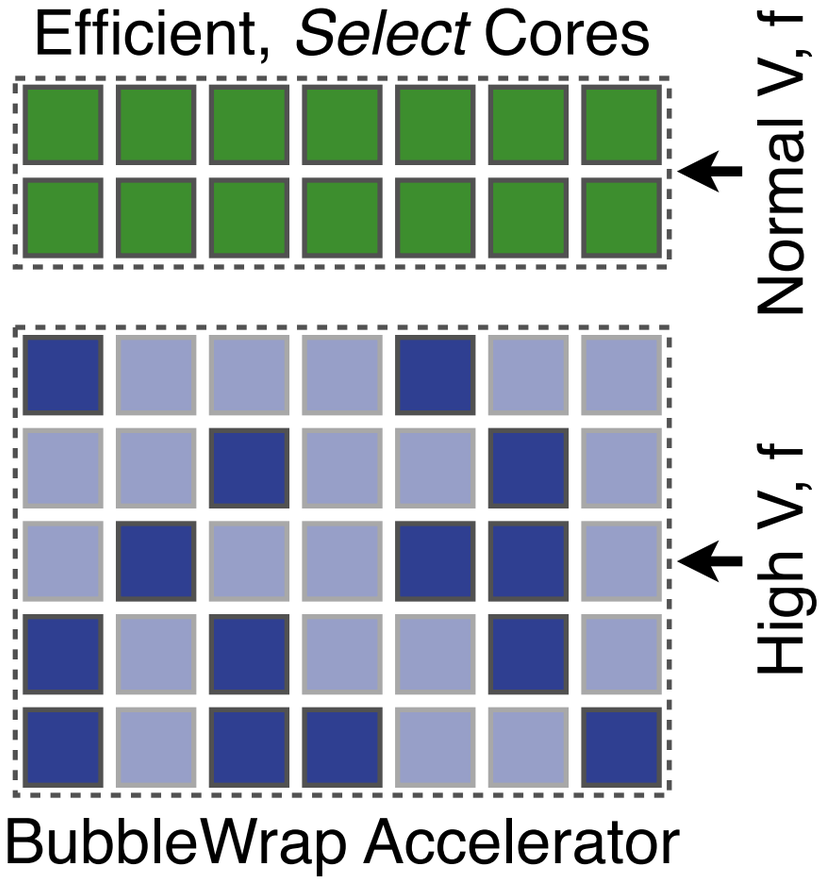
- Start with simple, homogeneous CMP
- Select most efficient cores
  - Run at nominal *V*, *f*
  - Use for parallel phases
- Rest of cores are BubbleWrap
  - "Sequential accelerators"

A BubbleWrap CMP



- Start with simple, homogeneous CMP
- Select most efficient cores
  - Run at nominal *V*, *f*
  - Use for parallel phases
- Rest of cores are BubbleWrap
  - “Sequential accelerators”
  - Free to consume high power

A BubbleWrap CMP

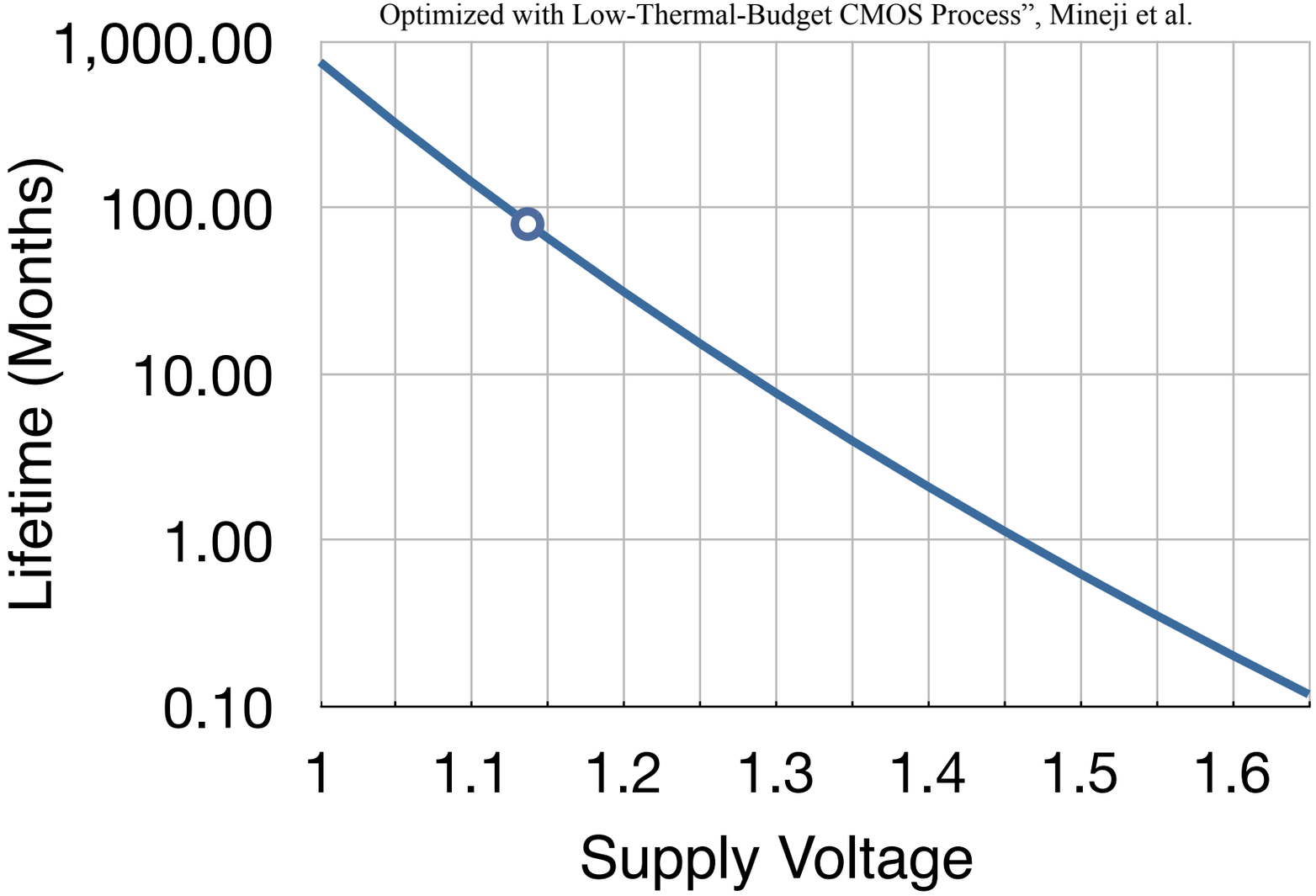


12nm CMP

- Start with simple, homogeneous CMP
- Select most efficient cores
  - Run at nominal *V*, *f*
  - Use for parallel phases
- Rest of cores are BubbleWrap
  - "Sequential accelerators"
  - Free to consume high power
  - Run at high *V*, *f*

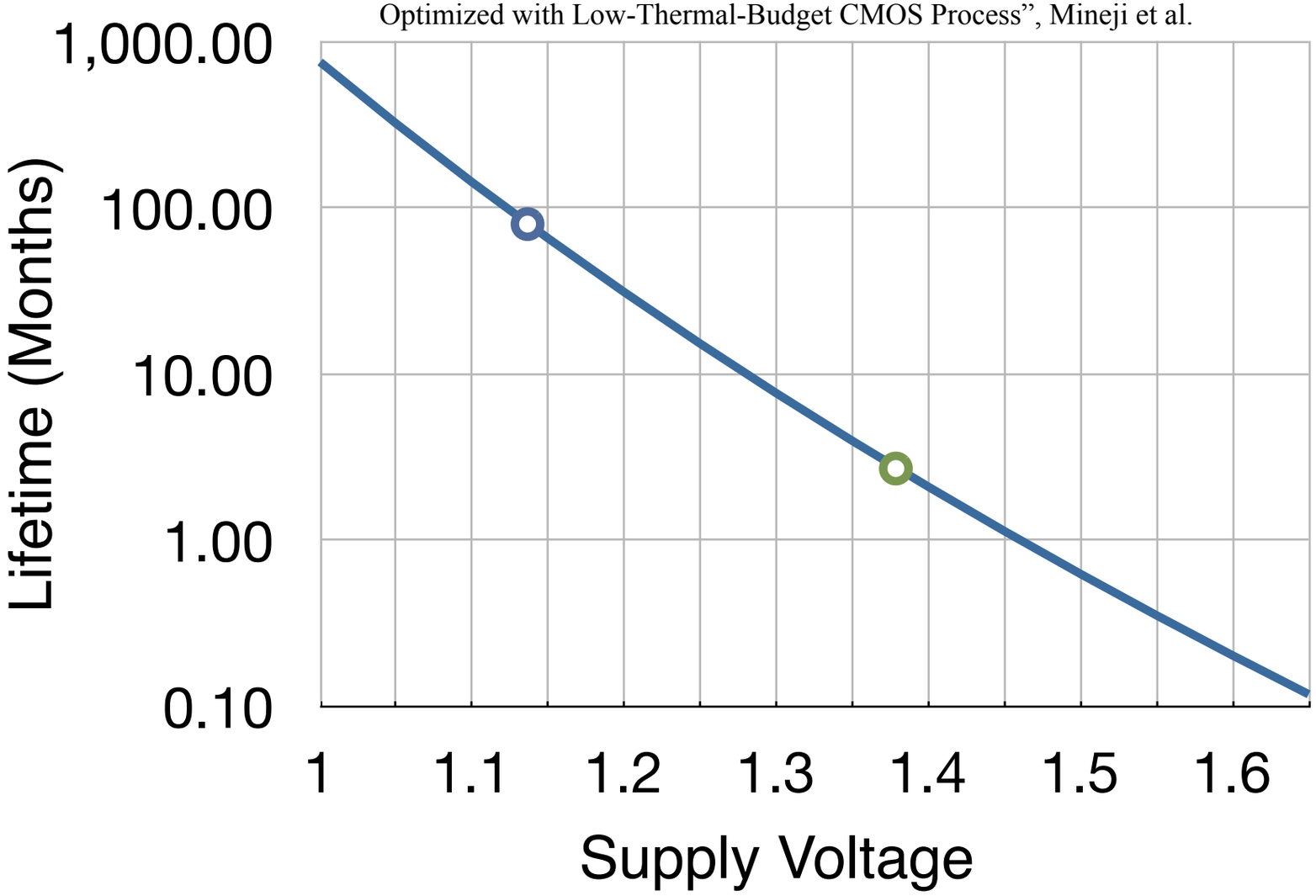
Sequential Benefits

"Integration of Manufacturable 65nm-node HfSiON Transistors Optimized with Low-Thermal-Budget CMOS Process", Mineji et al.



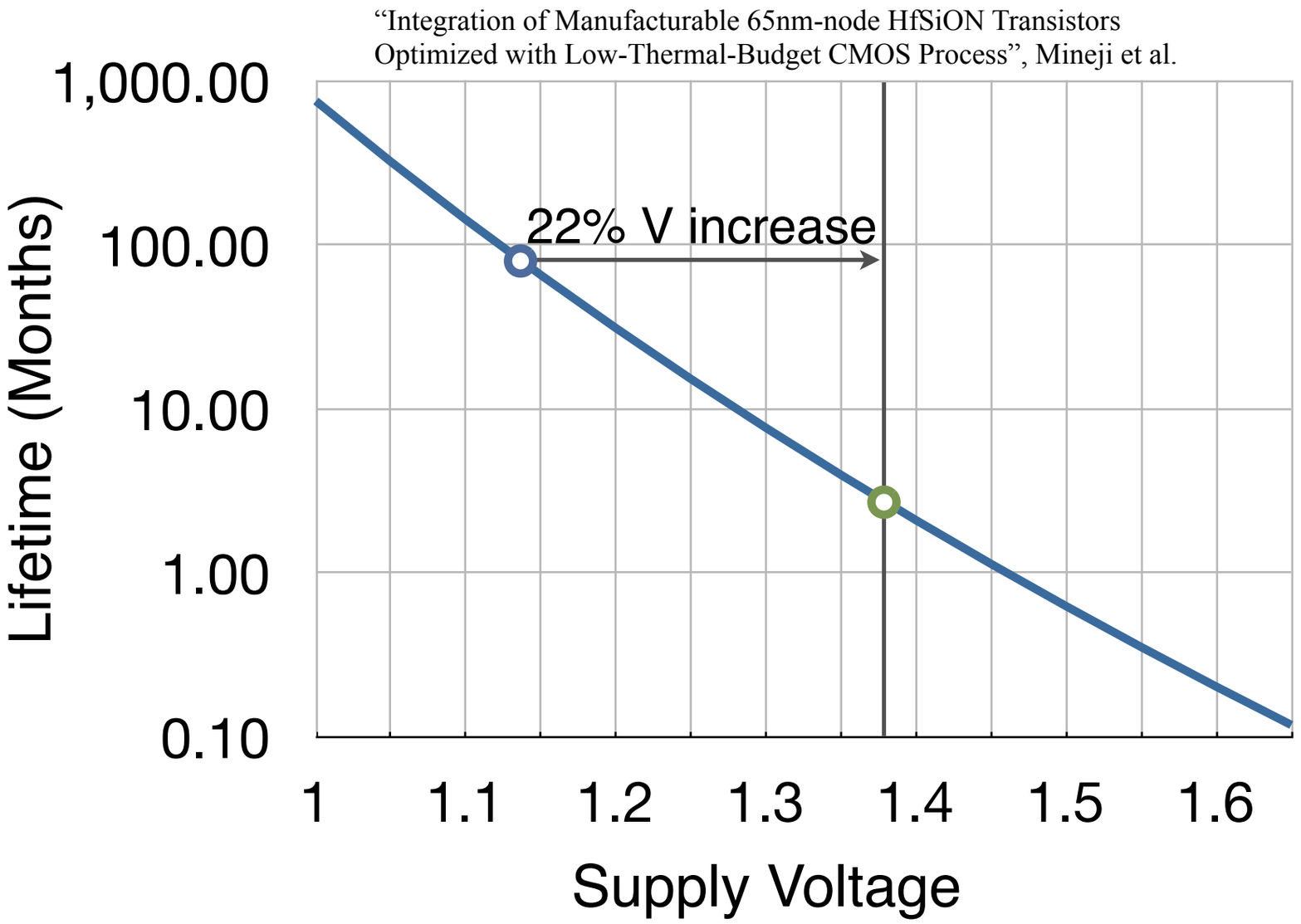
Sequential Benefits

"Integration of Manufacturable 65nm-node HfSiON Transistors Optimized with Low-Thermal-Budget CMOS Process", Mineji et al.



- No BubbleWrap (7 years / core)
- 35 BubbleWrap cores (2.4 months / core)

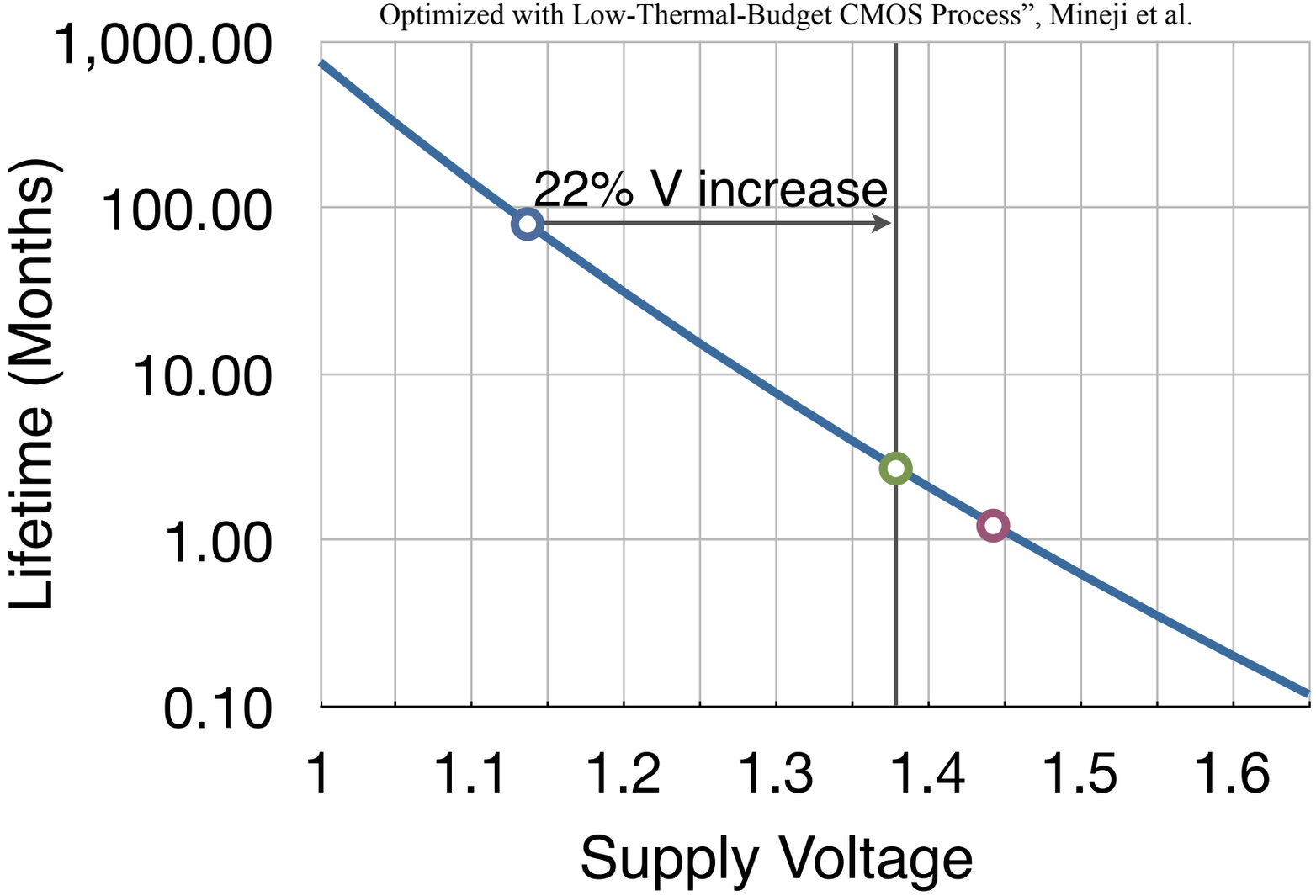
Sequential Benefits



- No BubbleWrap (7 years / core)
- 35 BubbleWrap cores (2.4 months / core)

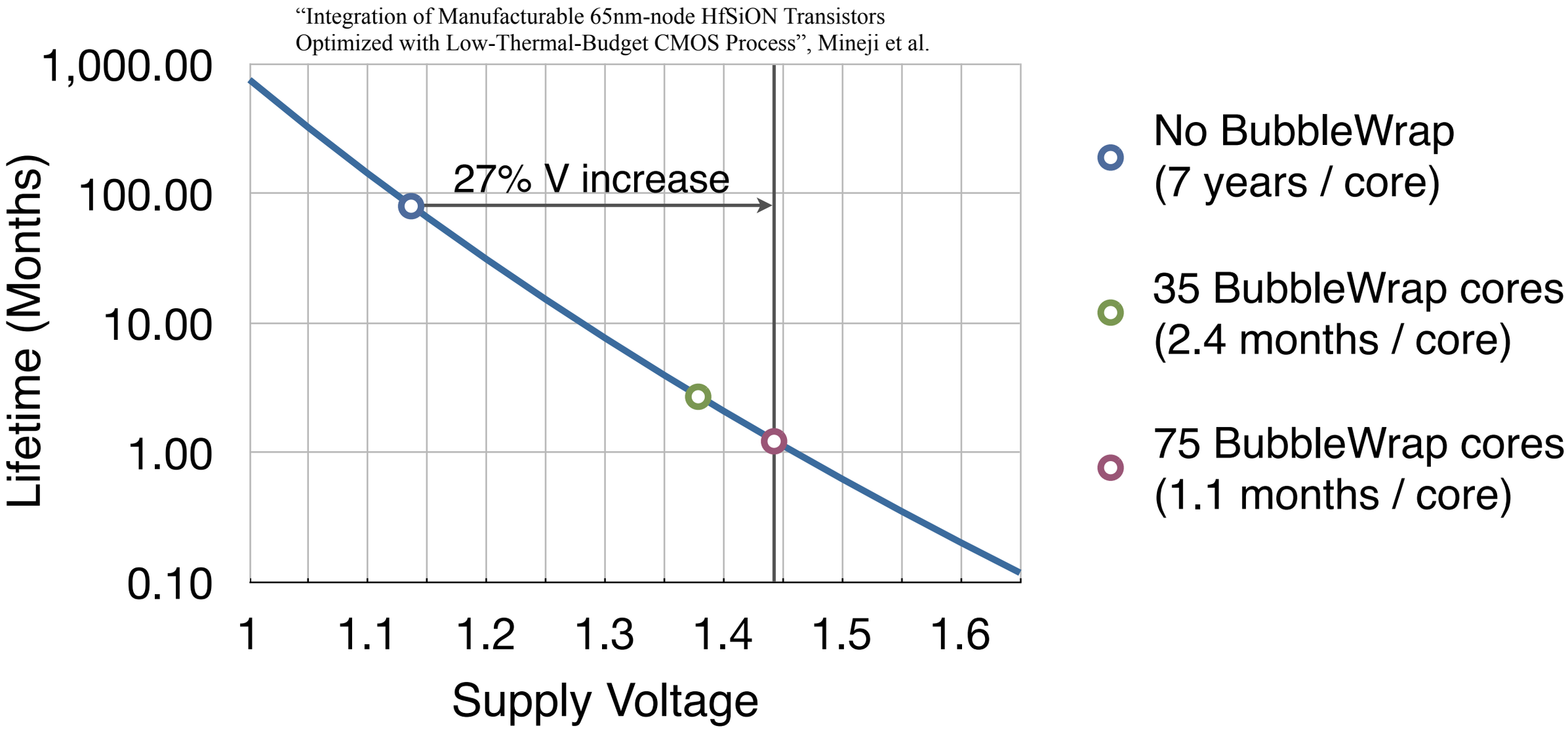
Sequential Benefits

"Integration of Manufacturable 65nm-node HfSiON Transistors Optimized with Low-Thermal-Budget CMOS Process", Mineji et al.

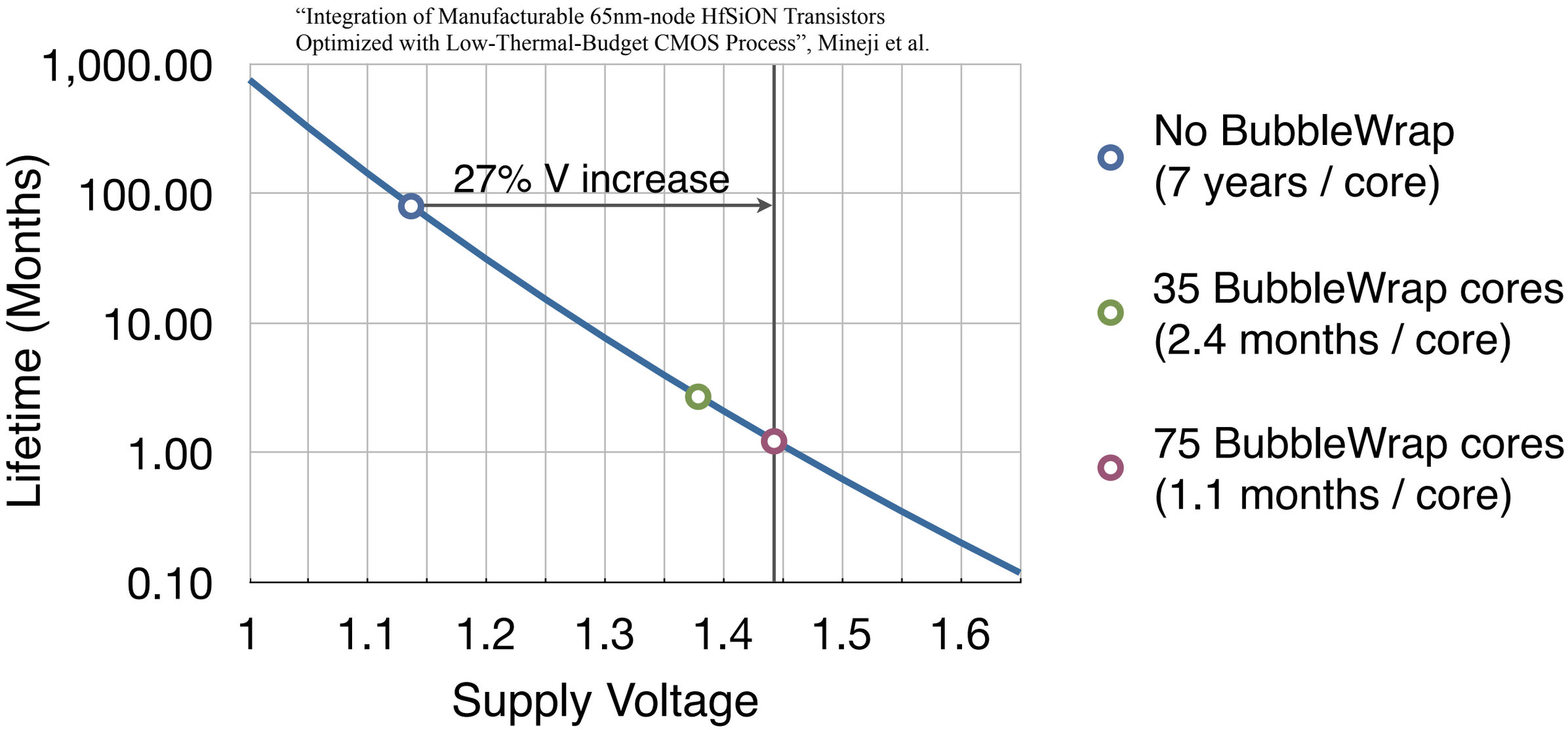


- No BubbleWrap (7 years / core)
- 35 BubbleWrap cores (2.4 months / core)
- 75 BubbleWrap cores (1.1 months / core)

Sequential Benefits



Sequential Benefits



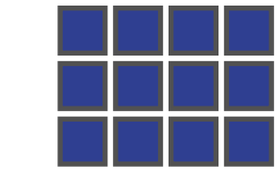
Significant single-thread frequency increase

Throughput Benefits

Throughput Benefits

- Exploit core-to-core variation in power consumption

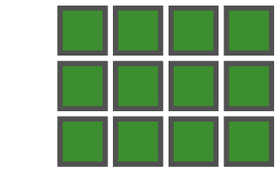
Throughput Benefits



Base CMP

- Exploit core-to-core variation in power consumption
- Base CMP: Only build as many cores as we can power

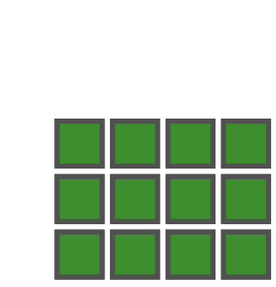
Throughput Benefits



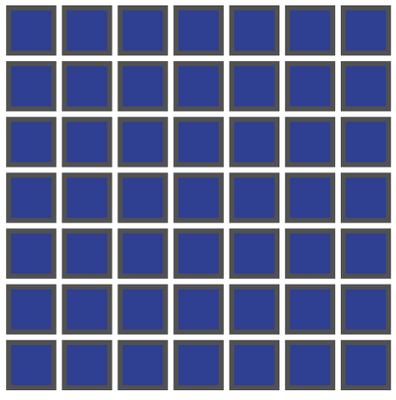
Base CMP

- Exploit core-to-core variation in power consumption
- Base CMP: Only build as many cores as we can power

Throughput Benefits



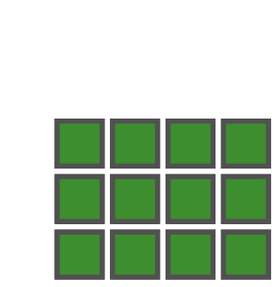
Base CMP



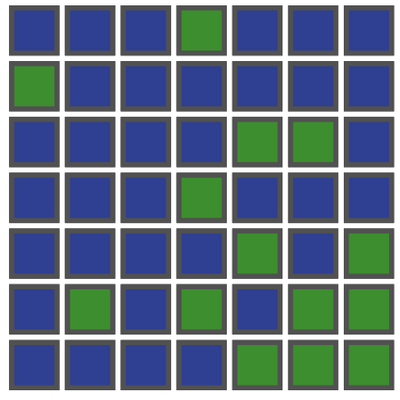
BubbleWrap CMP

- Exploit core-to-core variation in power consumption
- Base CMP: Only build as many cores as we can power
- BubbleWrap: Many cores to choose from

Throughput Benefits



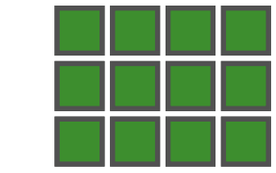
Base CMP



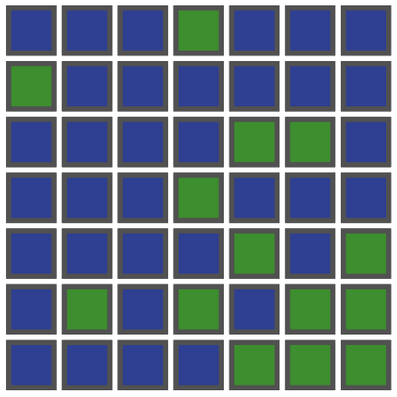
BubbleWrap CMP

- Exploit core-to-core variation in power consumption
- Base CMP: Only build as many cores as we can power
- BubbleWrap: Many cores to choose from
  - Select the most efficient ones for parallel phases

Throughput Benefits



Base CMP



BubbleWrap CMP

- Exploit core-to-core variation in power consumption
- Base CMP: Only build as many cores as we can power
- BubbleWrap: Many cores to choose from
  - Select the most efficient ones for parallel phases

➡ 10% higher throughput than base [Teodorescu08]

Is it a good idea?

Is it a good idea?

No

Is it a good idea?

No

- Additional die area for dormant cores

Is it a good idea?

No

- Additional die area for dormant cores
- Modest speedups -- can’t we do better?

Is it a good idea?

No

- Additional die area for dormant cores
- Modest speedups -- can’t we do better?

Yes

Is it a good idea?

No

- Additional die area for dormant cores
- Modest speedups -- can’t we do better?

Yes

- Simple homogeneous design, no core changes

Is it a good idea?

No

- Additional die area for dormant cores
- Modest speedups -- can’t we do better?

Yes

- Simple homogeneous design, no core changes
- Improves sequential *and* parallel performance

Is it a good idea?

No

- Additional die area for dormant cores
- Modest speedups -- can’t we do better?

Yes

- Simple homogeneous design, no core changes
- Improves sequential *and* parallel performance
- No architectural or software changes

Is it a good idea?

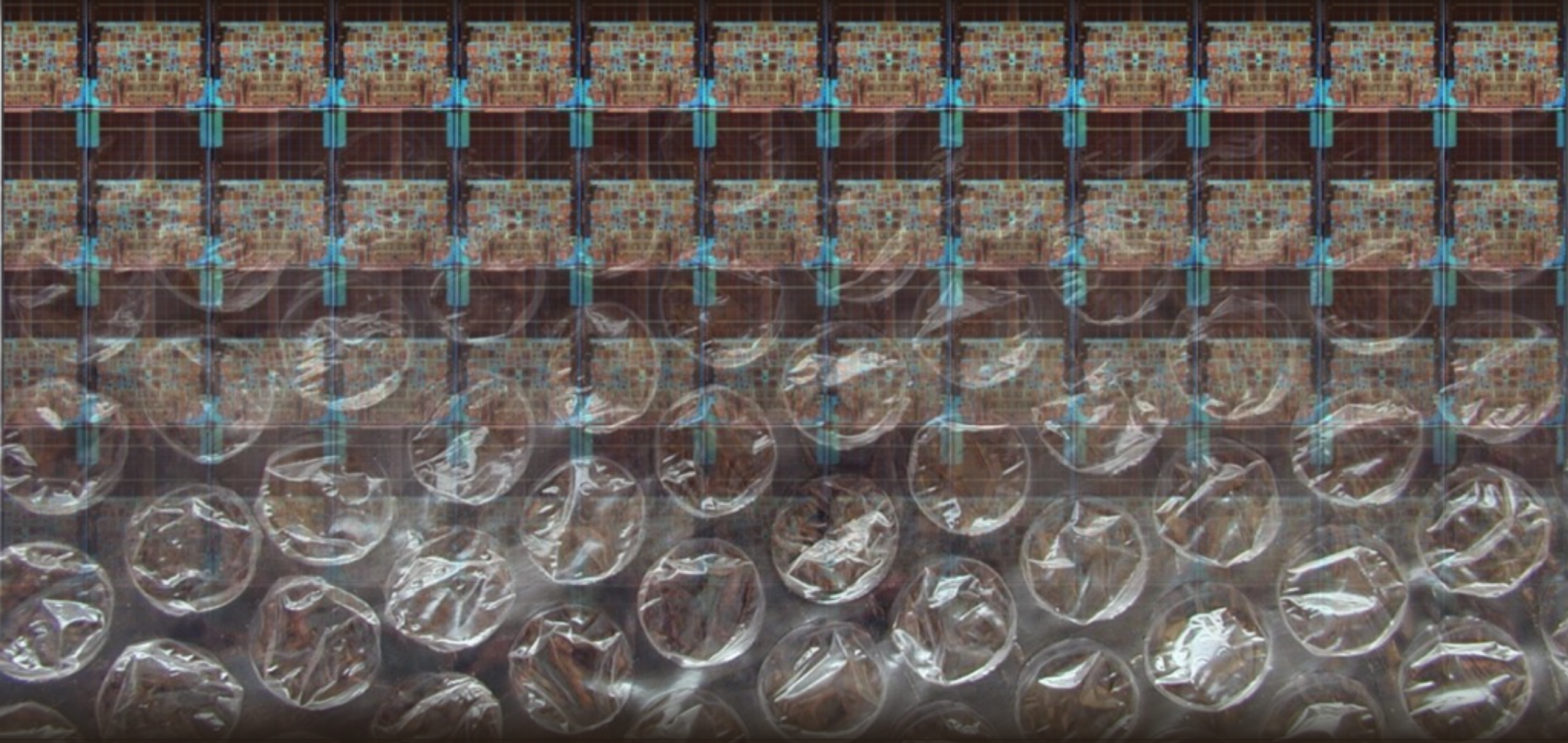
No

- Additional die area for dormant cores
- Modest speedups -- can’t we do better?

Yes

- Simple homogeneous design, no core changes
- Improves sequential *and* parallel performance
- No architectural or software changes
- Accelerates existing binaries

BubbleWrap: Popping CMP Cores for Sequential Acceleration



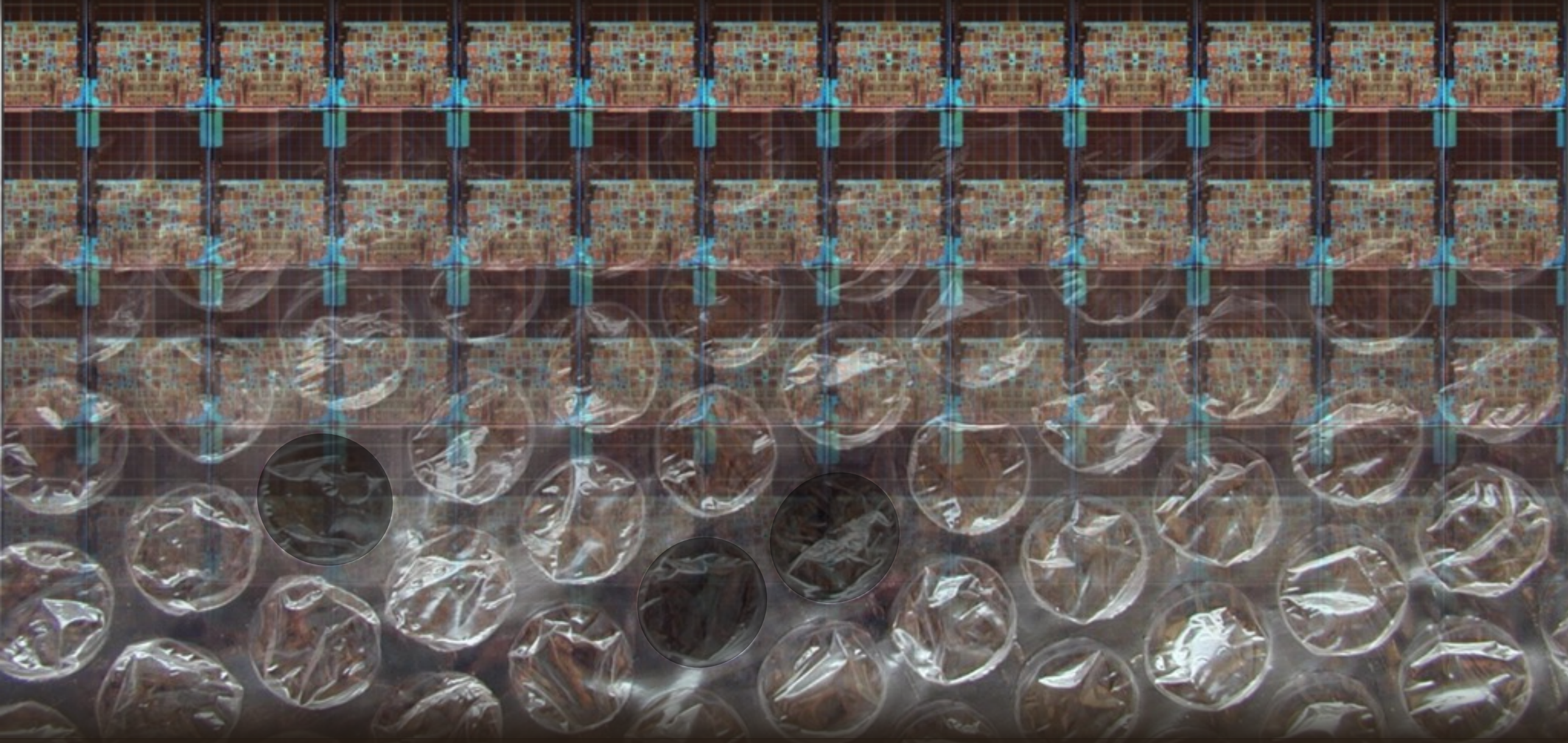
Brian Greskamp, Ulya Karpuzcu, and Josep Torrellas

University of Illinois

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

i-acoma group

BubbleWrap: Popping CMP Cores for Sequential Acceleration



Brian Greskamp, Ulya Karpuzcu, and Josep Torrellas

University of Illinois

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

i-acoma group