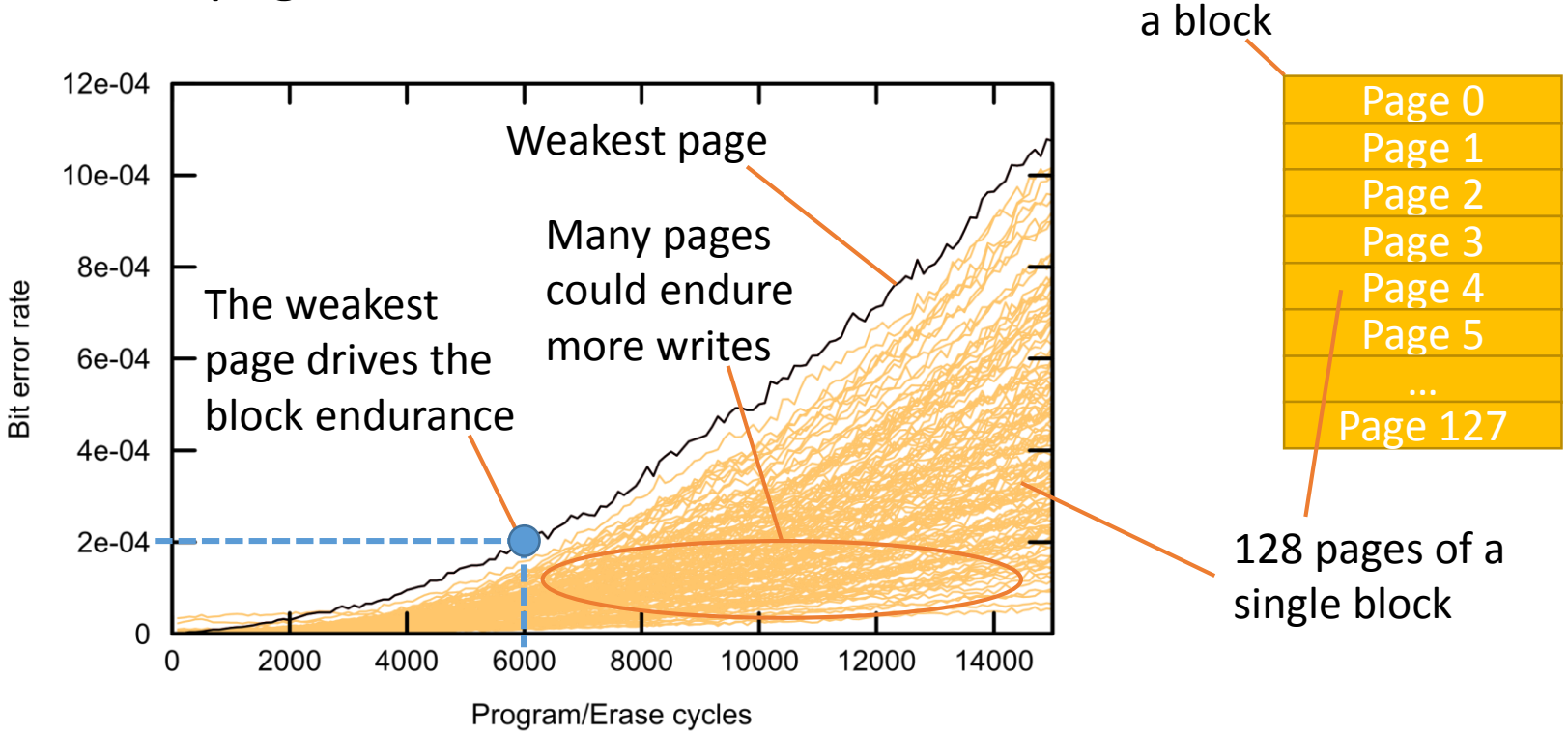
Wear Unleveling: Improving NAND Flash Lifetime by Balancing Page Endurance

Xavier Jimenez, David Novo, and Paolo Ienne

Ecole Polytechnique Fédérale de Lausanne (EPFL)
School of Computer and Communication Sciences
CH-1015 Lausanne, Switzerland

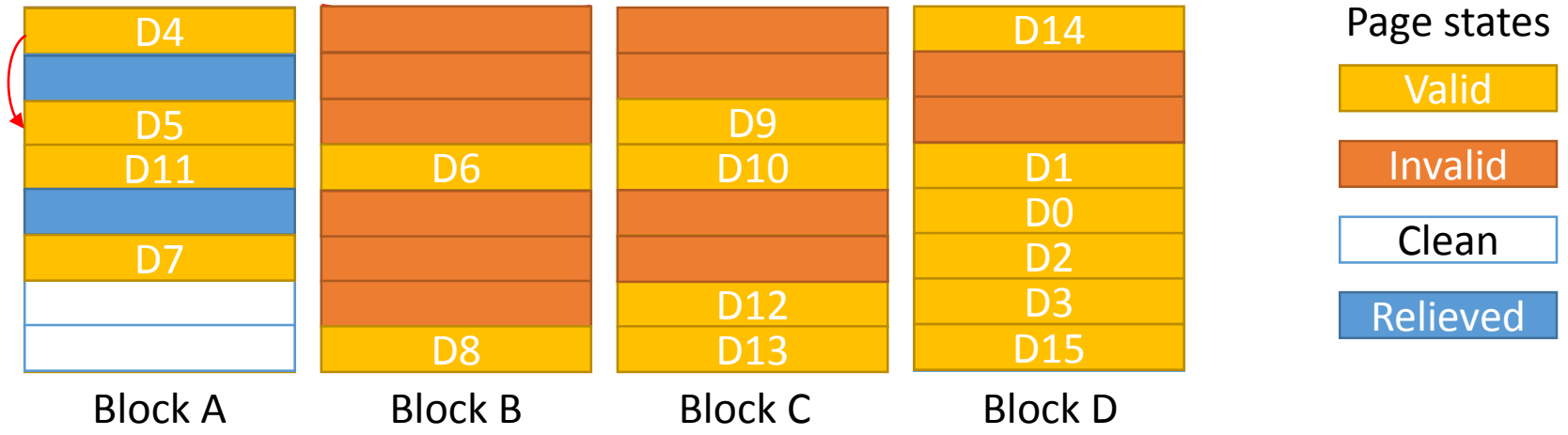
Flash limited lifetime and endurance variance

- NAND Flash is organized in blocks of hundreds pages
- Some pages wear out faster than others



Some background on NAND flash

- Electron tunneling to add/remove charges into/from floating gate
  - Adding charges = Programming (**page**)
  - Removing charges = Erasing (**block**)
  - Out-of-place updates
- Memory cells degrade over Program/Erase (P/E) cycles
  - ECC units correct limited number of errors
  - Spare bytes in each page to store the codes and metadata



Relieve pages to extend the endurance

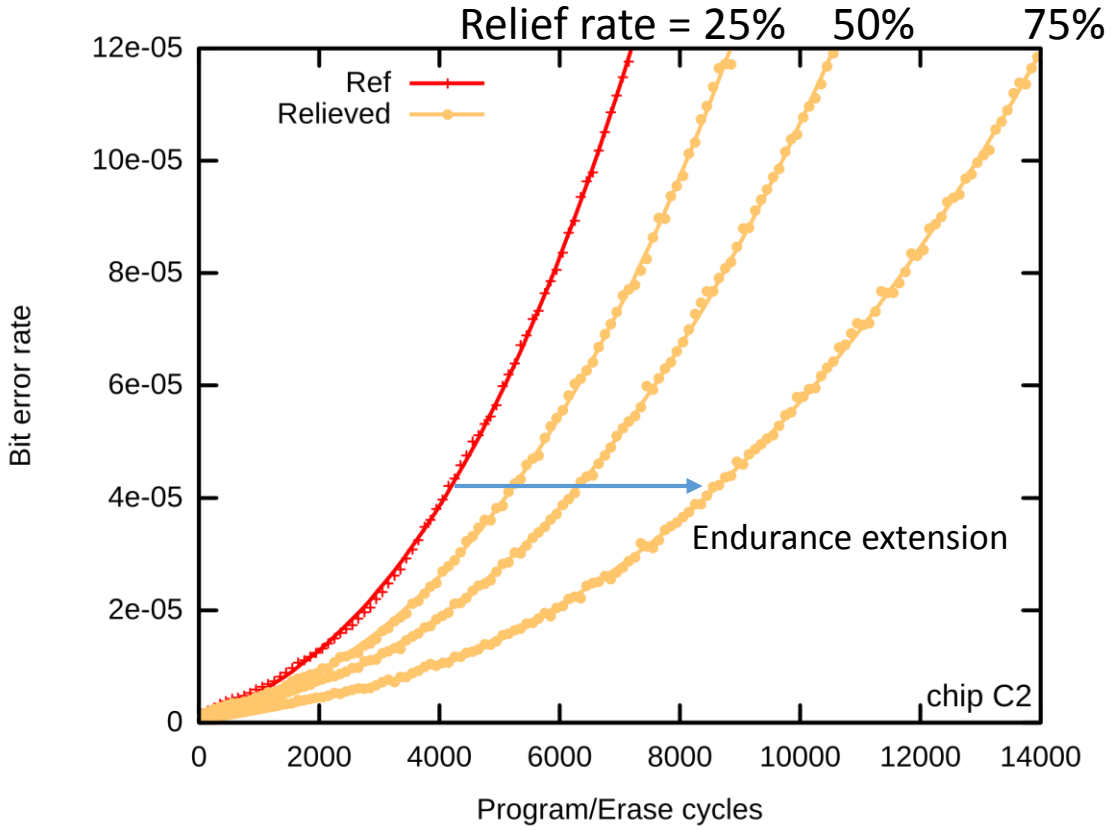
- Page relief is characterized on two NAND flash chips
- Endurance ≠ lifetime

Endurance:

Total P/E cycles

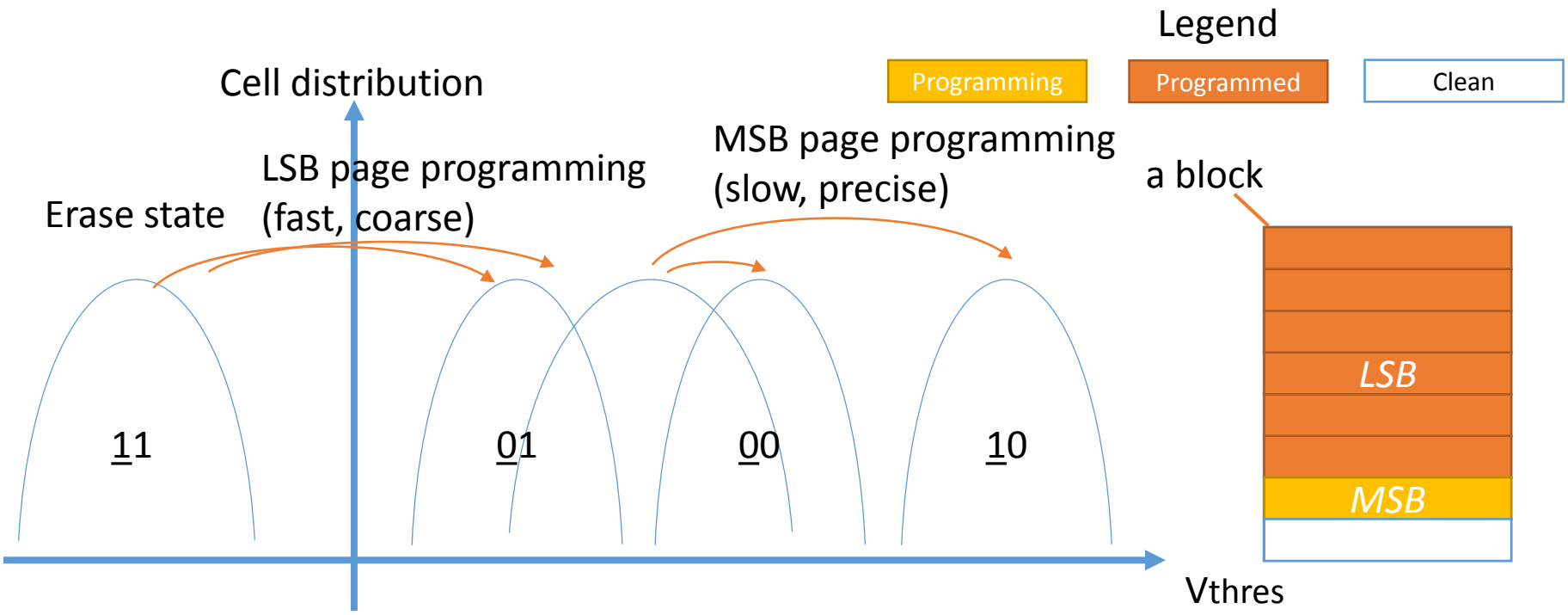
Lifetime:

Total data written



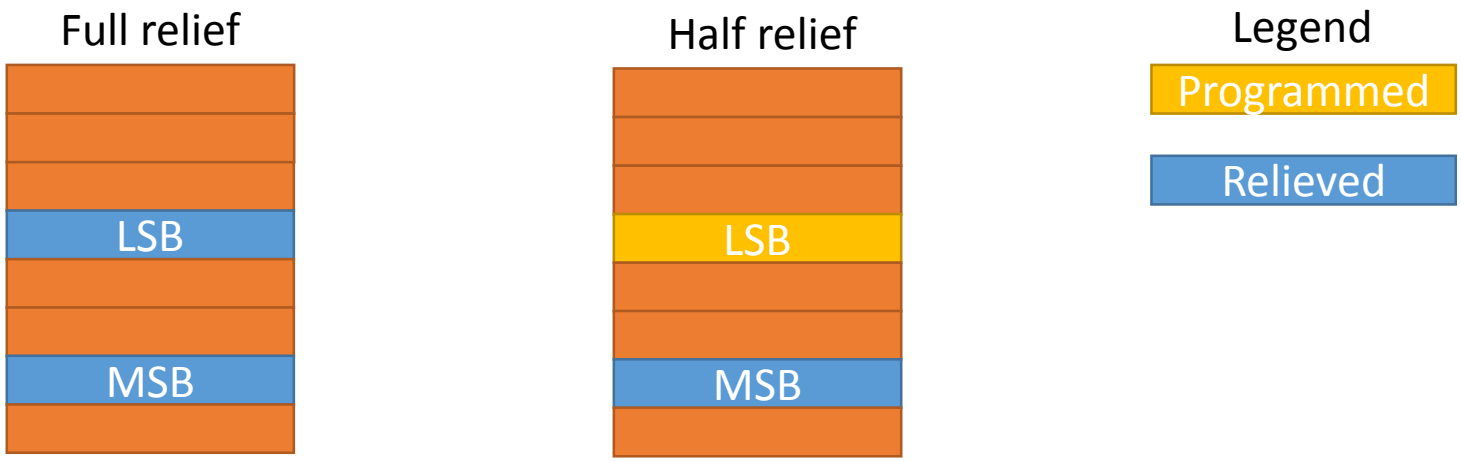
Page relief on multi-level cells: page pairs

- Multi-level cells (MLCs) store two bits
  - Each bit mapped to a different page: LSB and MSB page pair

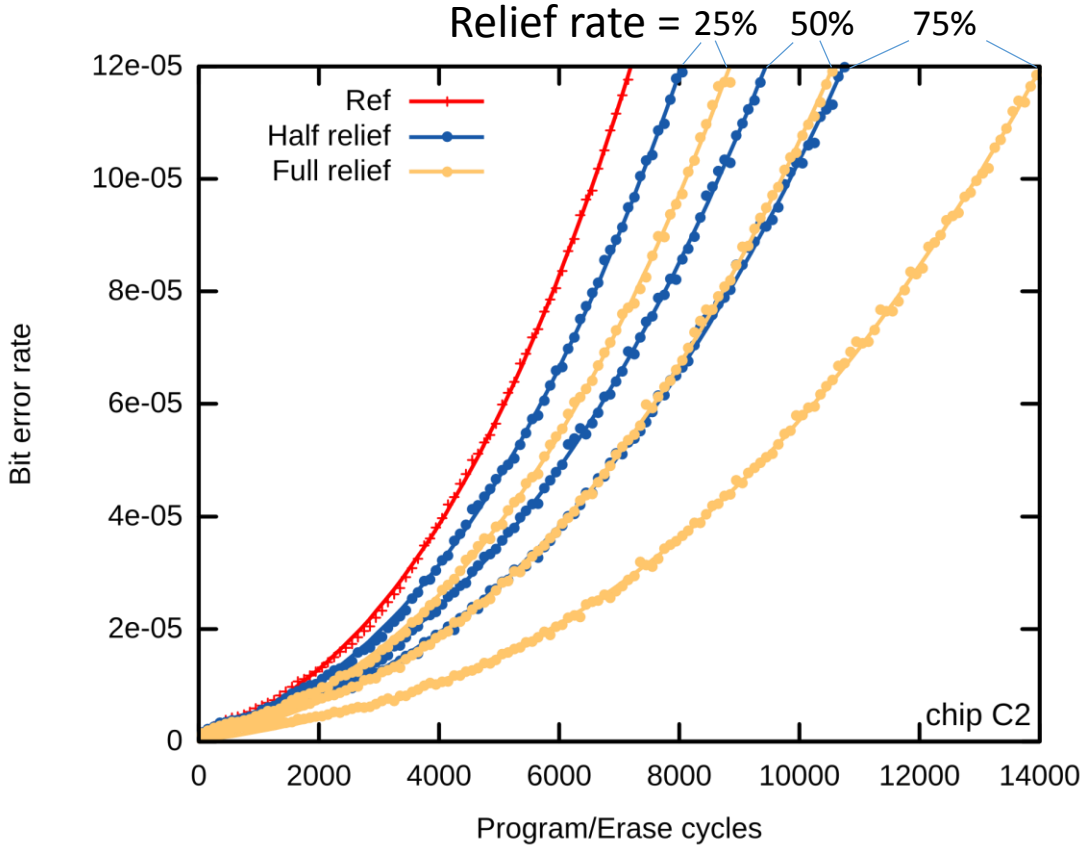


Half and full relief

- Half relief ≡ SLC-mode[Roohparvar, patent 08] [Grupp et al., MICRO'09]
  - LSB programming approx. 3-4x faster than MSB
  - Tradeoffs capacity for lower write latency



Half and full relief characterization



Half relief can be more effective

- Relative wear of relief cycles for 2 chips

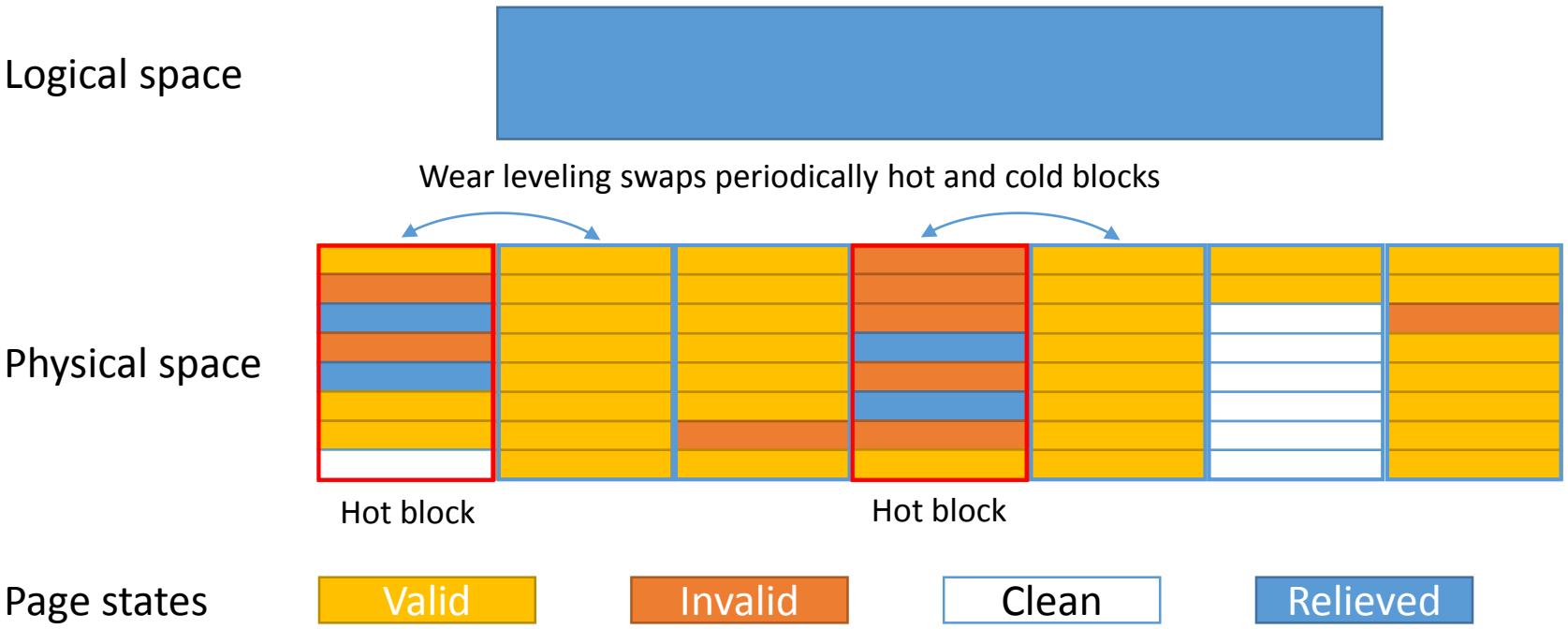
Chip	*Full*	*Half*
C1	**39%**	**61%**
C2	**34%**	**55%**

- Half relief is more effective in terms of written bits per cycle
  - 2 bits written in 2 cycles:

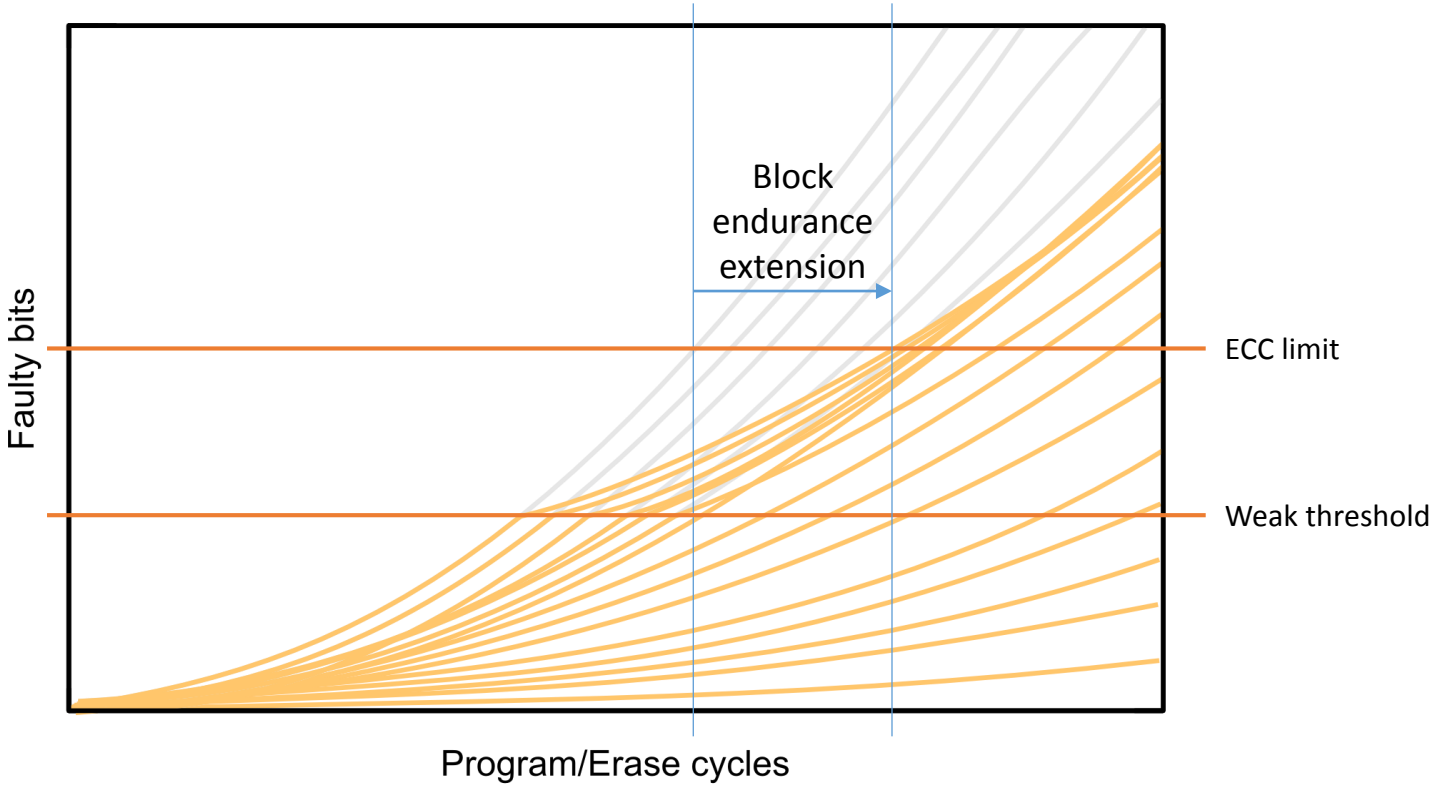
Chip	*Full + Regular*	*2x Half*
C1	39%+100% = **139%**	2 x 61% = **122%**
C2	34%+100% = **134%**	2 x 55% = **110%**

Hot blocks provide control and opportunity

- Flash Translation Layers (FTLs) provide simple interface, similar to magnetic disks
  - Garbage collection, wear leveling, and physical aspects of flash are hidden



Reactive strategy: identifying weak pages on the fly



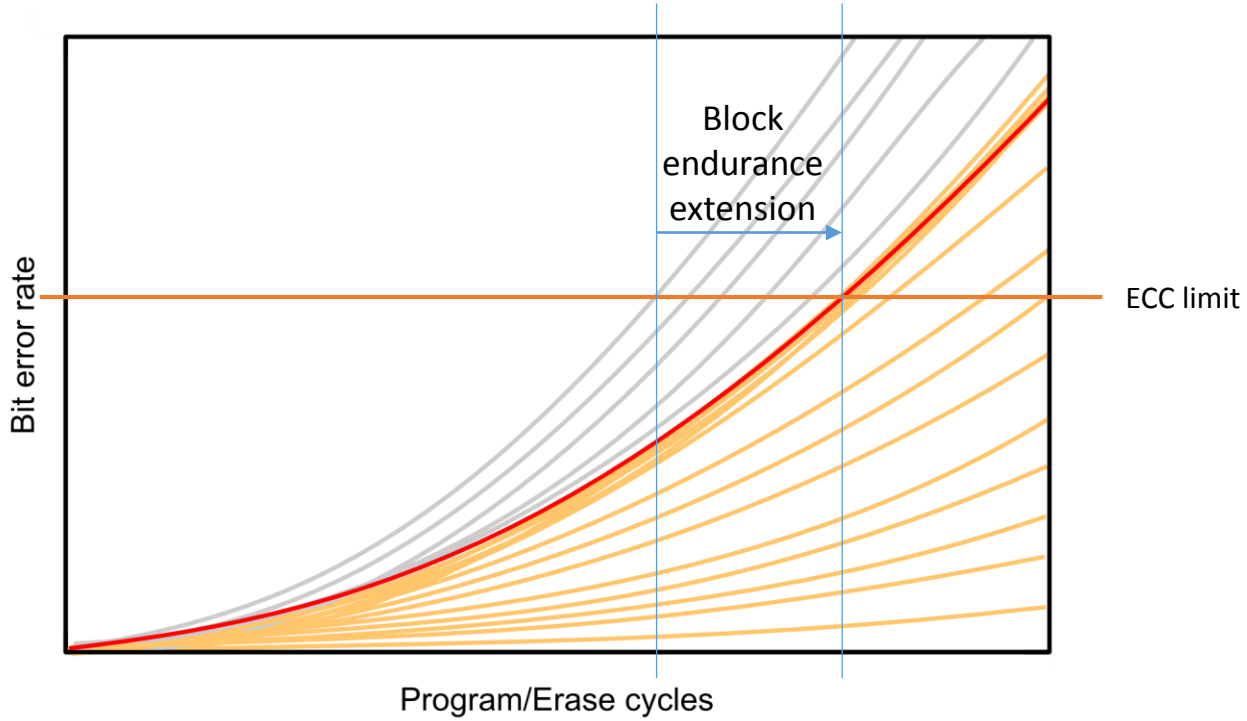
Reactive strategy: overheads

- Storage overhead:
  - 2 bits per page ⟶ cheap
- FTL memory overhead:
  - 2 bits per clean page (up to 32 Bytes per block with clean pages)
- Performance overhead:
  - Error monitoring: at worst, 1 extra read per write
    ⟶ approx. +10% write latency
  - Capacity reduction increases the garbage collection frequency

Simple but slow to react ⟶ less potential

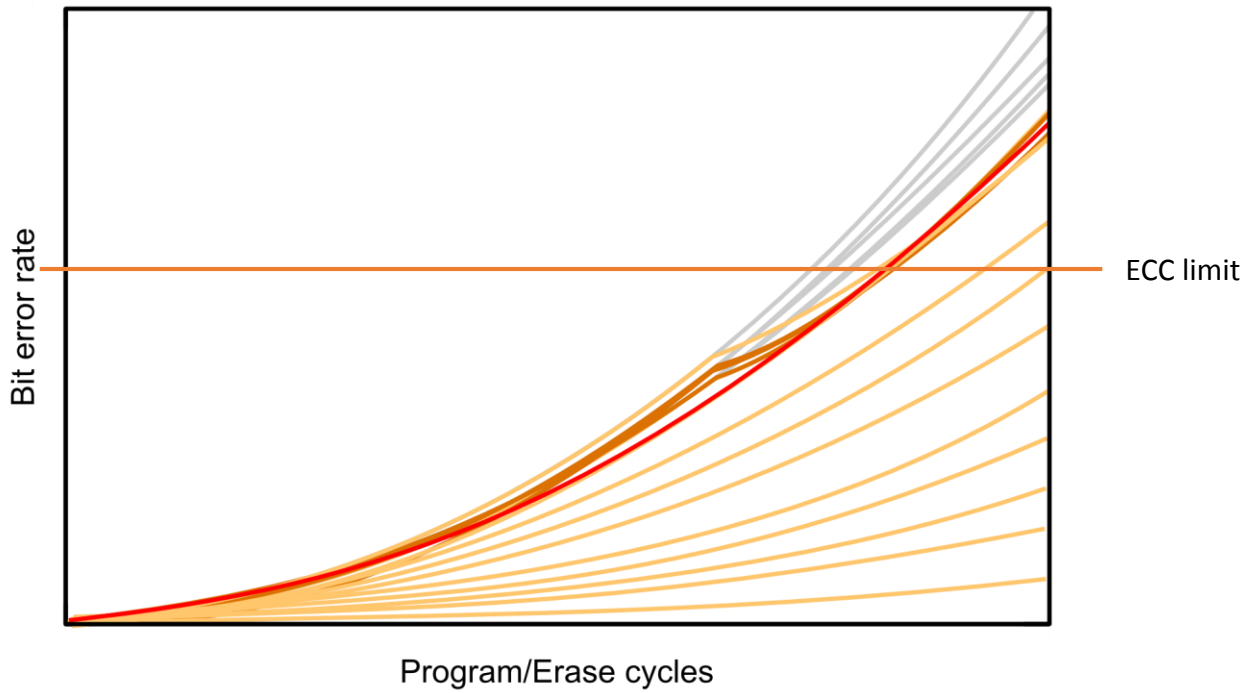
Proactive strategy: planning ahead of time

- Correlation between endurance and page pair number
  - We can compute the number of times weak pages should be relaxed to match the weakest page's extended endurance



Proactive strategy: adaptive planning

- Plan relief rates for multiple total relief cycles
  - Speculate on a number of relief cycles
  - When exceeded, speculate for a higher one



Proactive strategy on look-up tables

Plan 1 (4,000 relief cycles)

Page pair Nb	Half relief	Full relief
0	-	-
1	50%	50%
2	-	-
3	-	100%
4	-	-
5	30%	-
…	…	…
127	-	-

Weakest LSB/MSB page pair

Plan 2 (2,000 relief cycles)

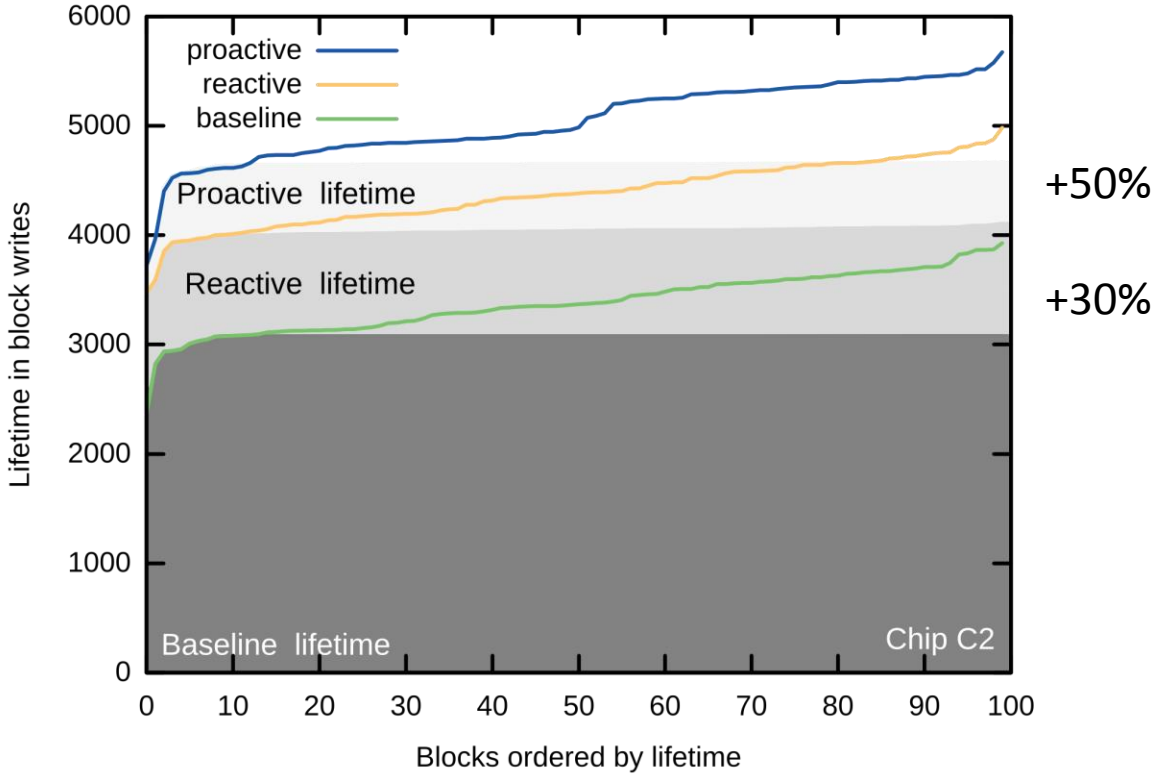
Page pair Nb	Half relief	Full relief
0	-	-
1	-	100%
2	60%	40%
3	-	100%
4	10%	-
5	-	100%
…	…	…
127	-	-

Proactive strategy overheads

- All the computational efforts done offline
- Storage/memory overhead:
  - A 16-bit counter per block to store relief cycles (similar to storing P/E cycles)
  - About 1-2 KB to store the LUTs (typically 2-4 LUTs)
    - LUT are sparse ⟶ can be compressed
- Performance overhead
  - Capacity reduction increases garbage collection frequency

Device lifetime evaluation: simple model

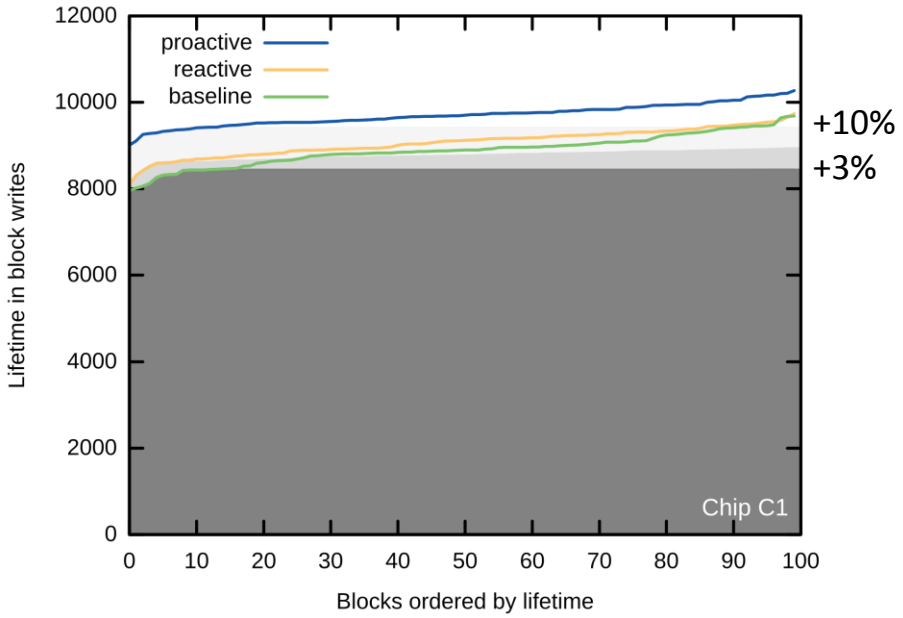
- Max BER of 10^{-4}, max 10% bad blocks, 60% fixed hot write ratio



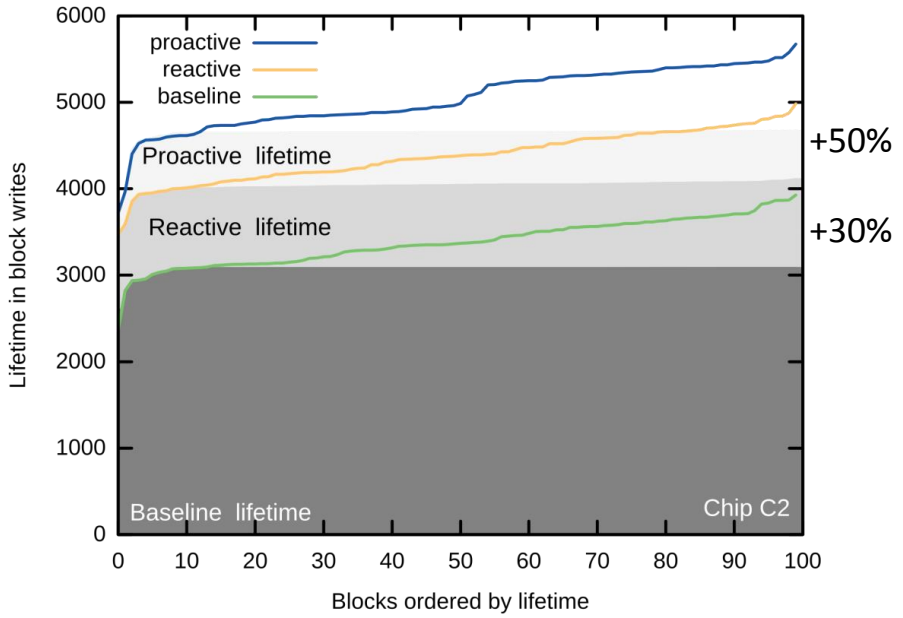
Relief can fix large variances

- Two 30 nm class chips with different architectures

Chip 1
ABL (less interference)

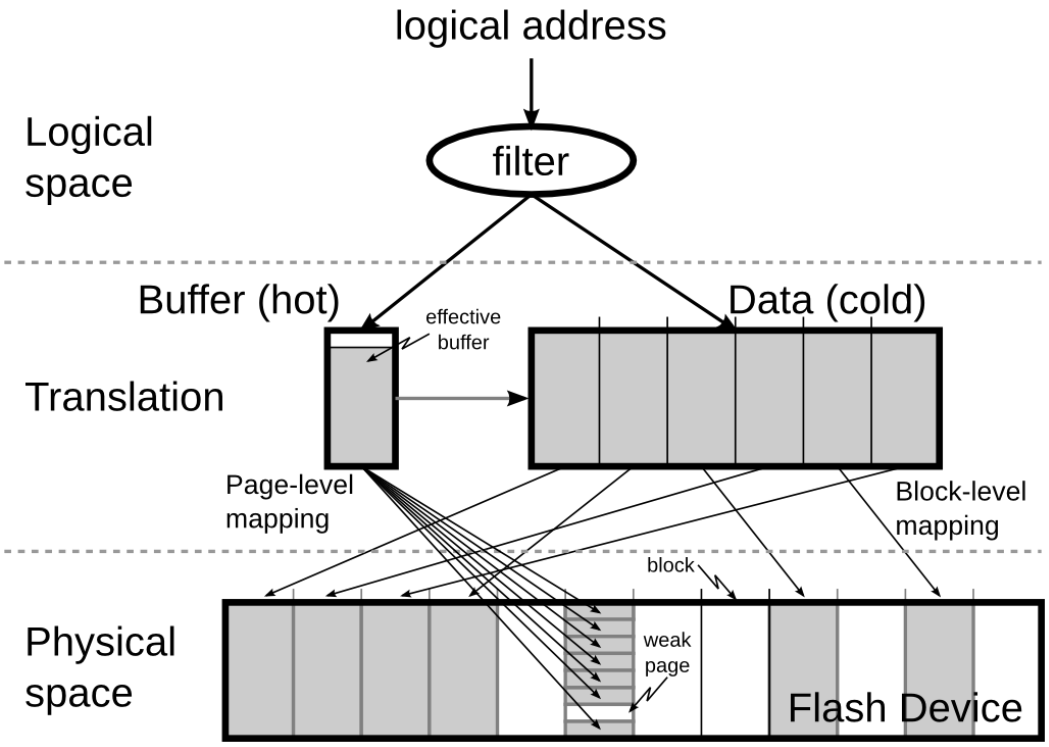


Chip 2
Interleaved (flexible, faster)

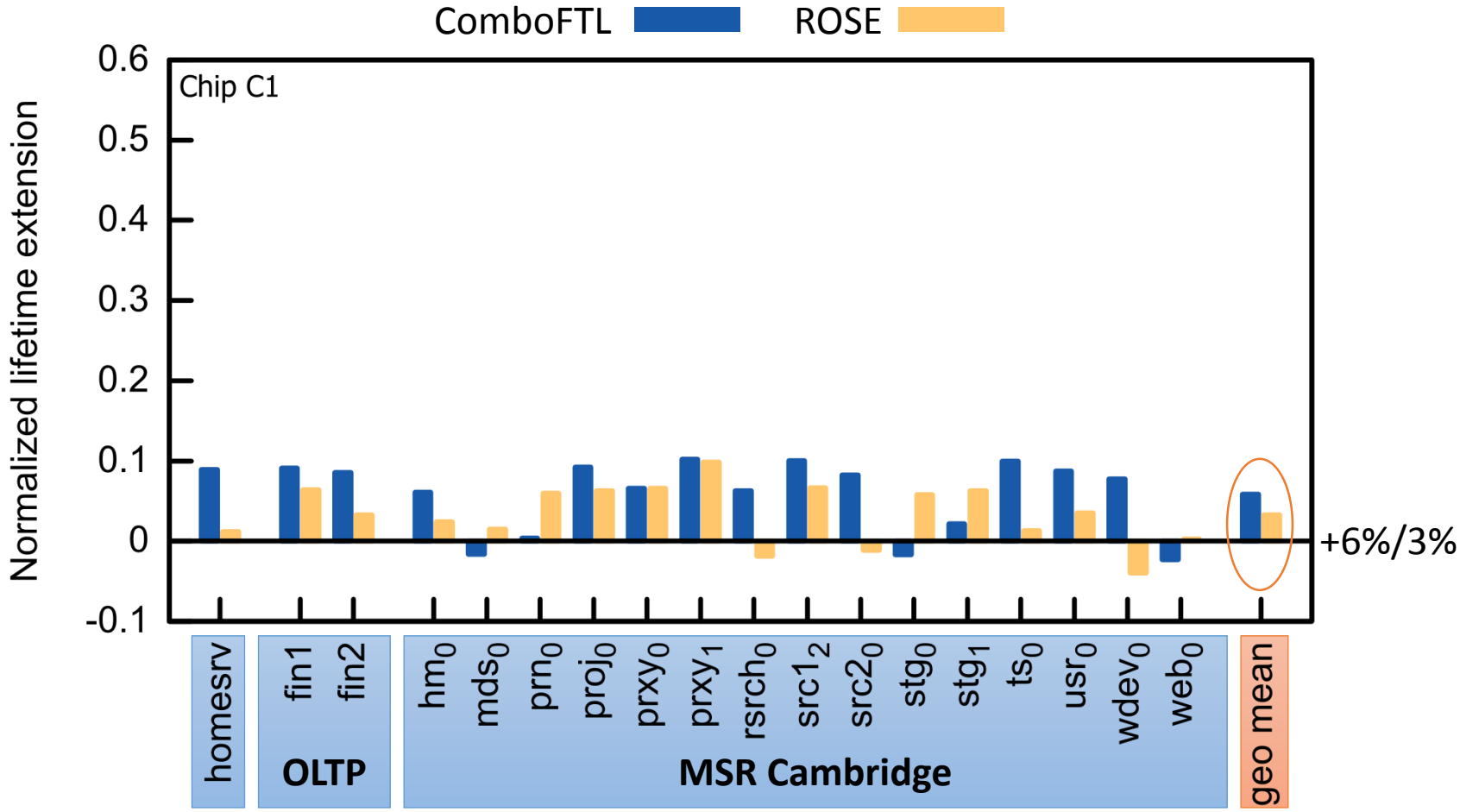


Evaluate the capacity loss impact

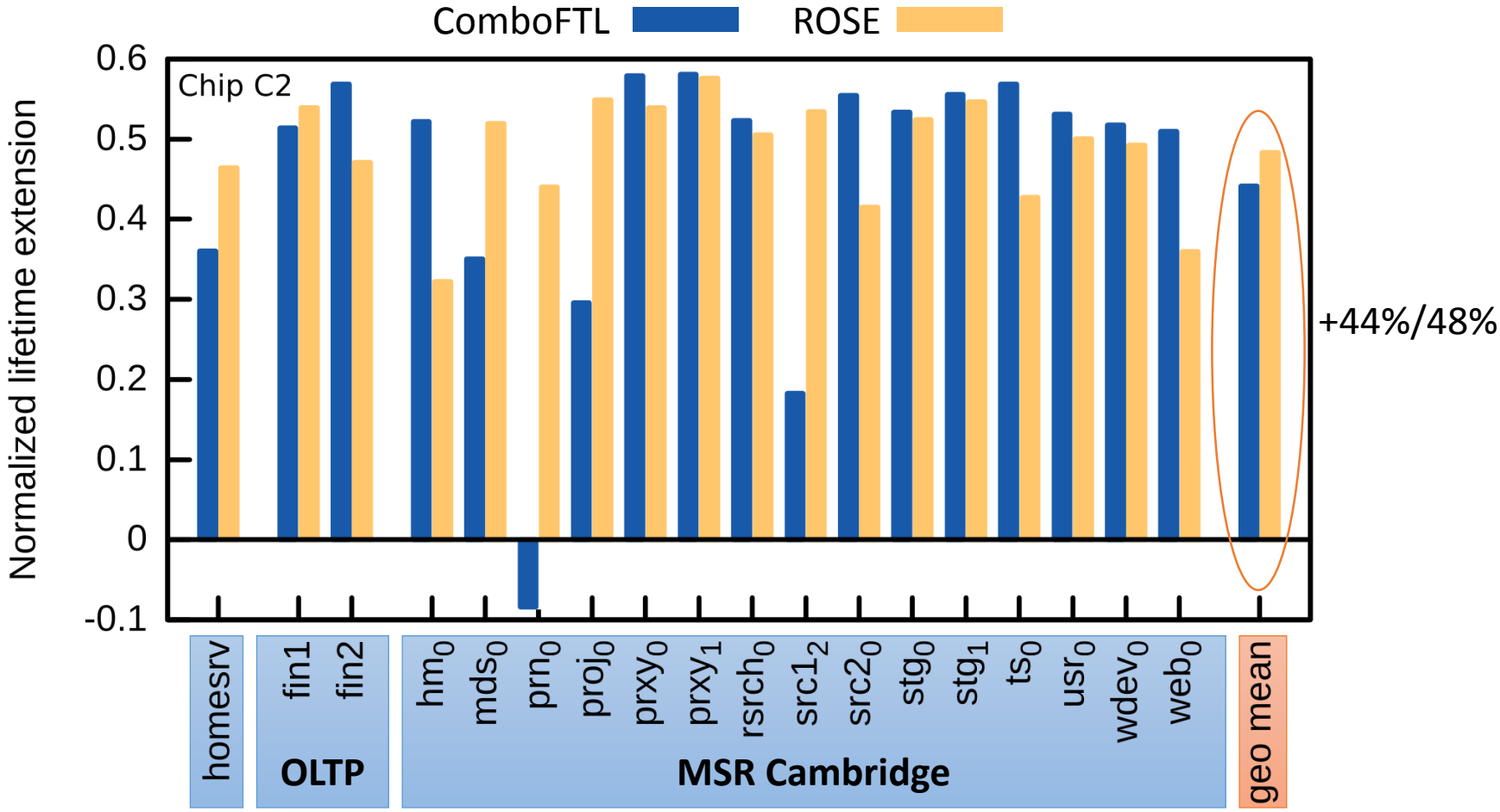
- We implemented the proactive strategy on two Hybrid FTLs
  - ROSE [TC'11], ComboFTL [JSA'10]



Lifetime extension

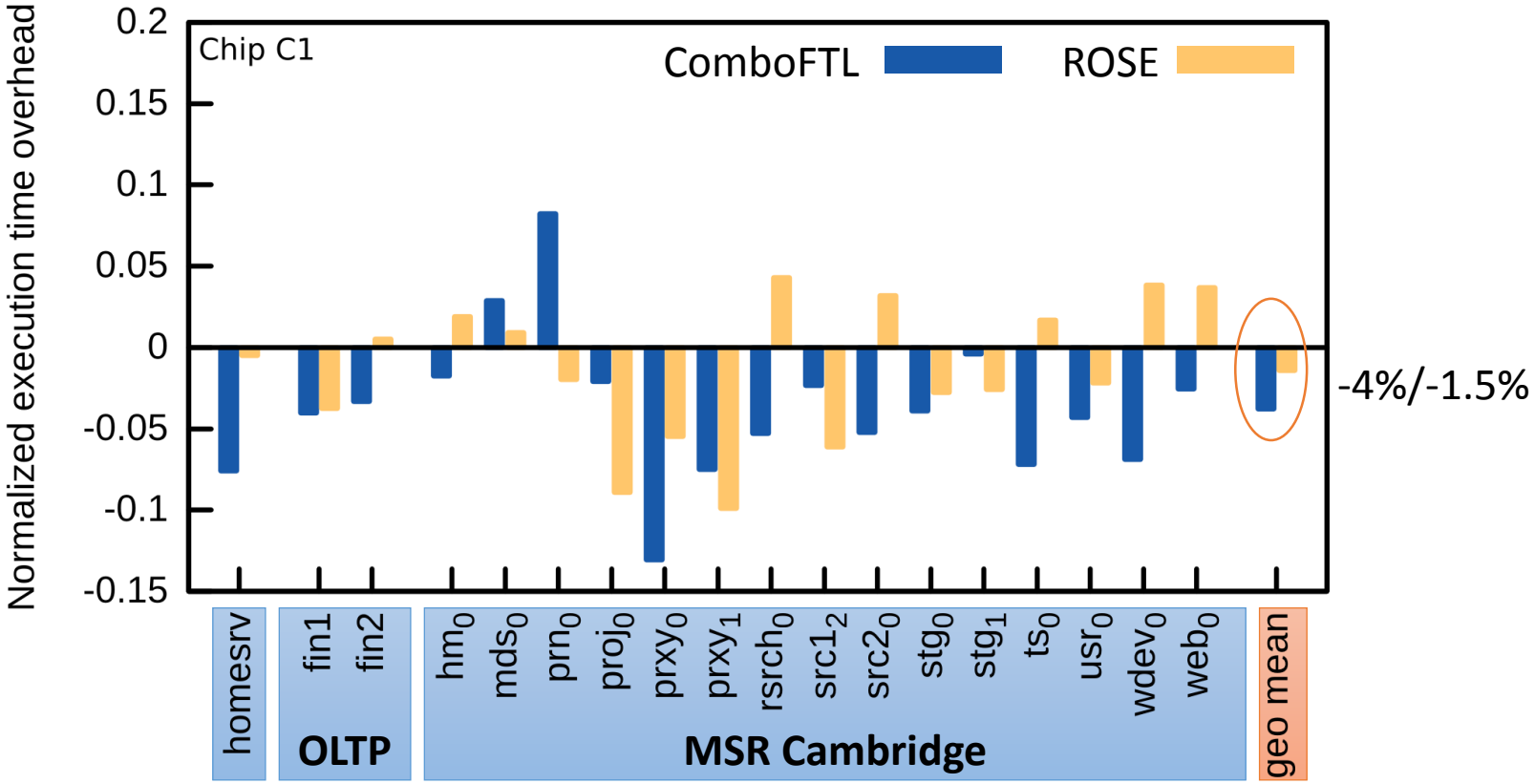


Lifetime extension



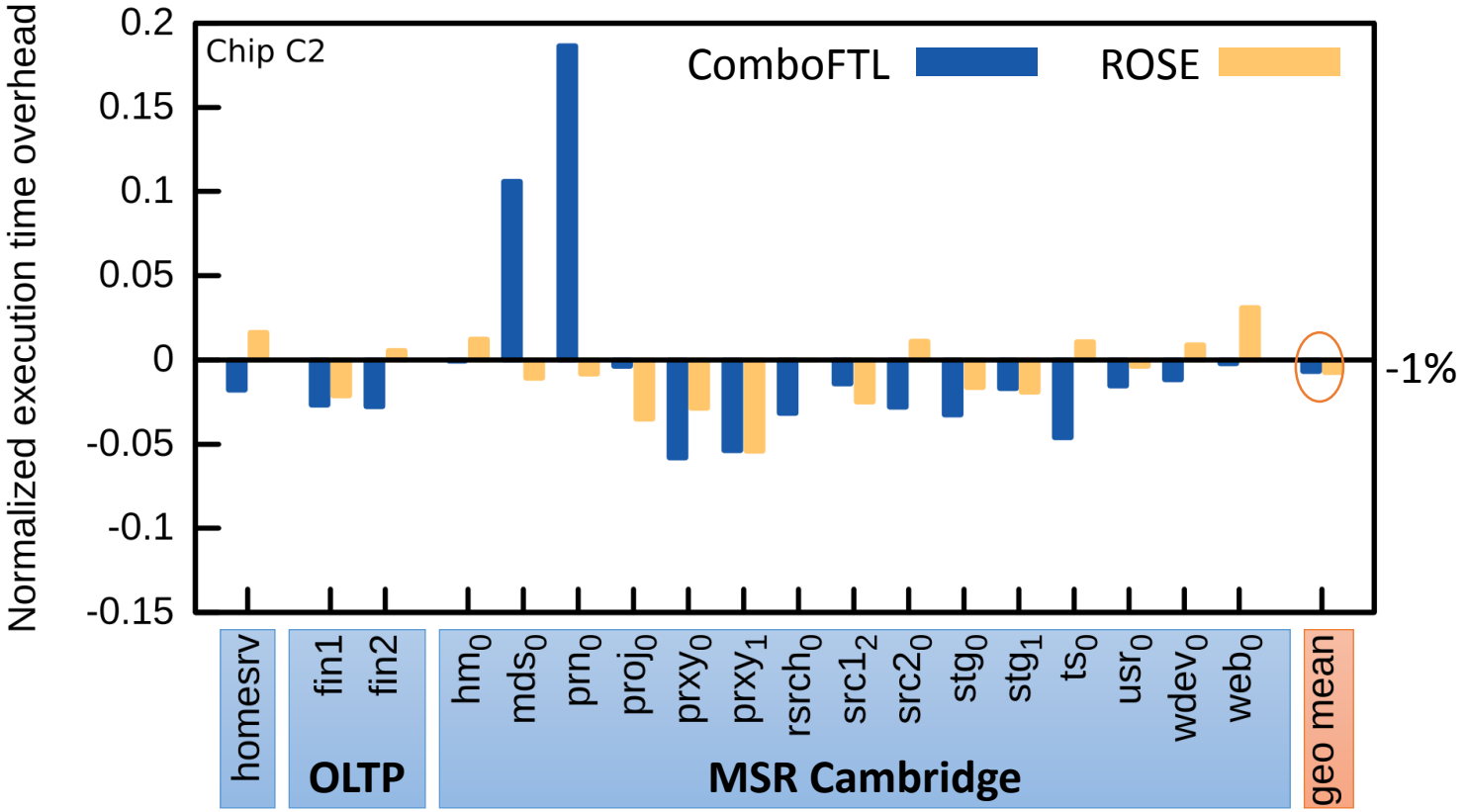
Half relief compensates GC overhead

- Despite the capacity loss, but thanks to half relief, execution time is stable



Half relief compensates GC overhead

- Despite the capacity loss, but thanks to half relief, execution time is stable



Conclusion

- The relief effect was characterized and is significant
- Its exploitability largely depends on the page endurance variance
- We proposed two strategies to integrate this effect into FTLs
- Can be implemented today with off-the-shelf chips
- There is room to develop more efficient approaches
  - Leave more computational efforts to the controller
- Technology scaling will inevitably bring a larger variance
  - Relieving pages might help to overcome the challenge