Technion

Electronics
Computers
Communications

Information in Storage Devices

049063 – EE Department, Technion

LECTURE 3: SSD ACCESS

HDD and SSD

- Has been around forever
- Improves, but looks the same
- Predictable performance

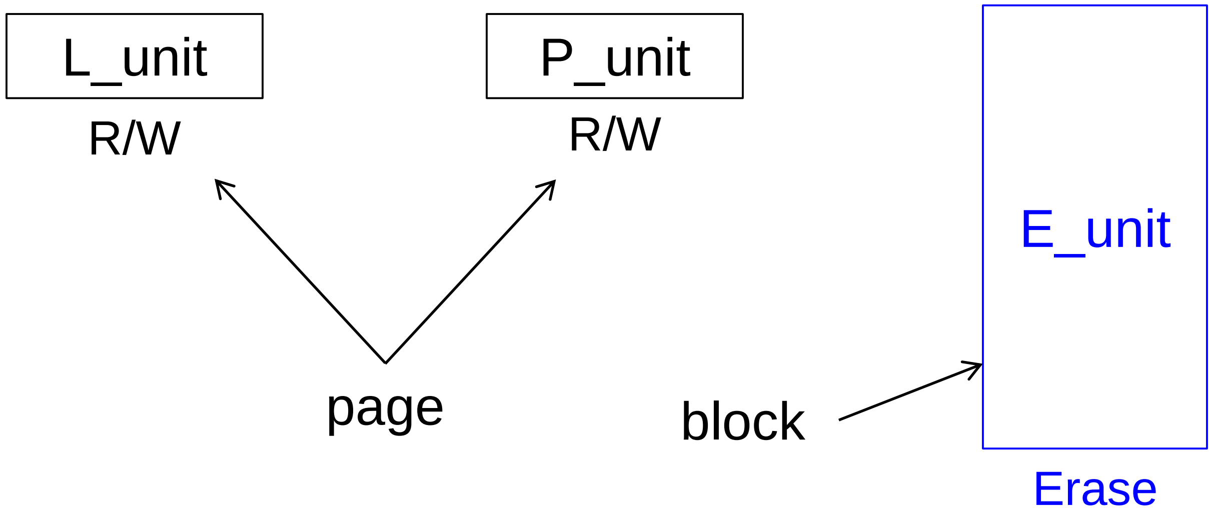
- Fast to respond
- Heavily hyped
  - High media exposure
- You know can do wonders
  - But most encounters less exciting

Solid-State Drive (SSD)

- Silicon-based array of memory cells
  - with standard interface (HDD replacement)
- Invented late '90s (M-Systems, Israel)
  - Uses NAND flash for maximal density
- More expensive, but much faster
- Capacity scales by "Moore's law"

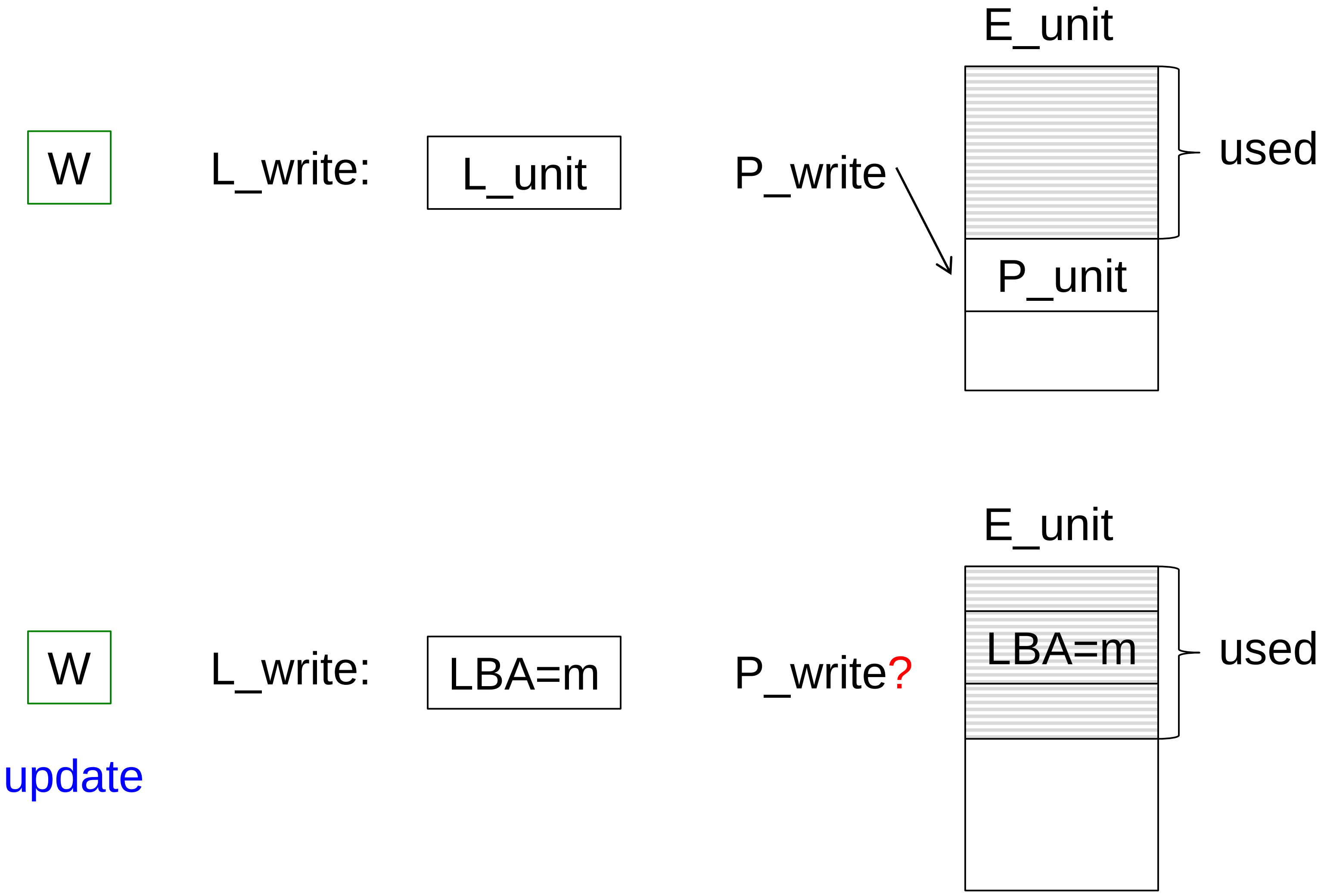
Flash: No Random-Access Erase

- New: Erase unit

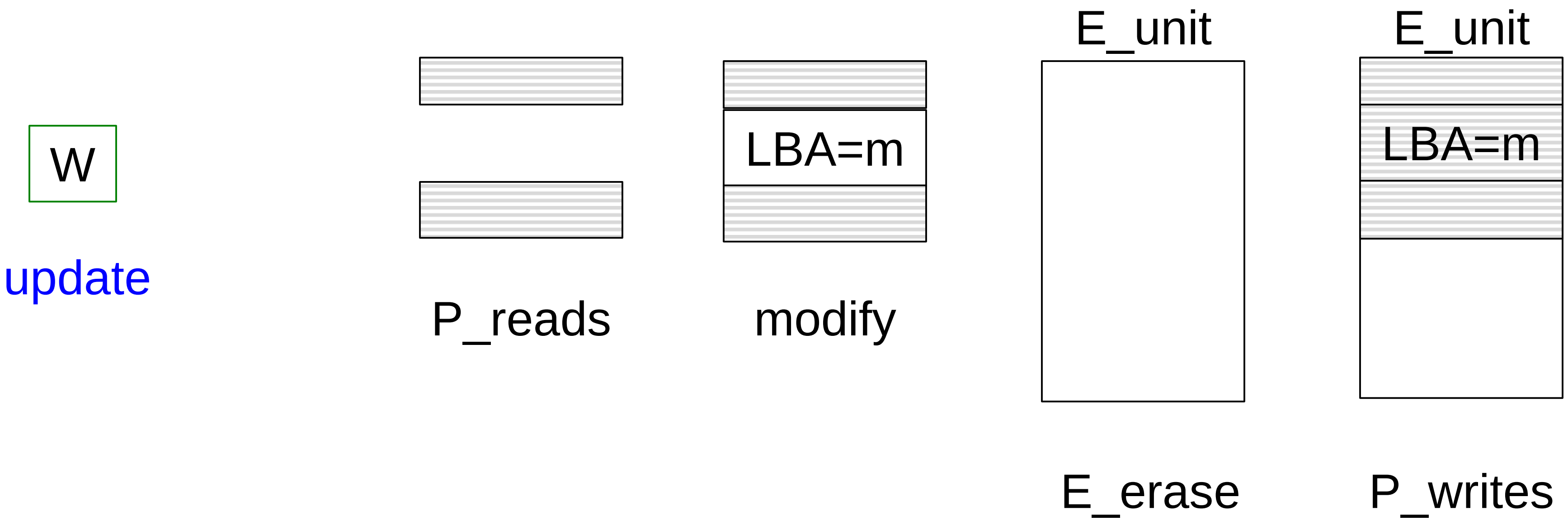


- Physical erase of cells: only full blocks
- Write → program/erase

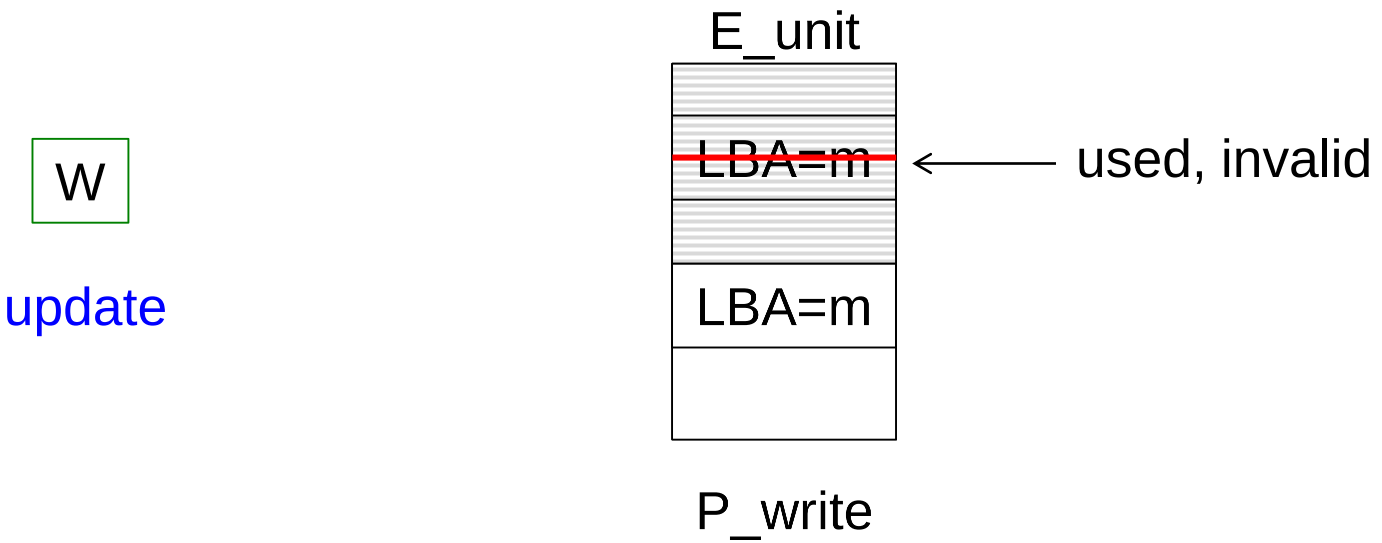
No In-Place Updates



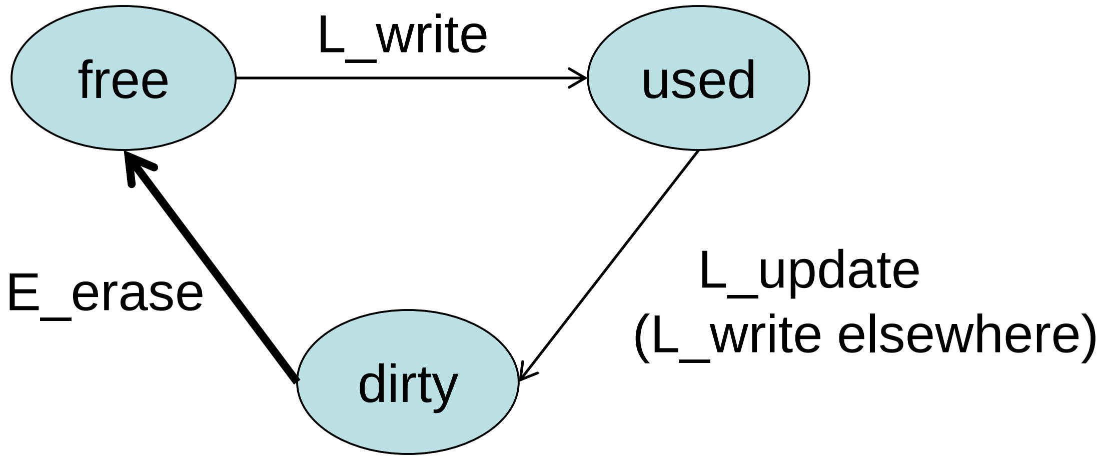
Option 1: RMEW



Option 2: Invalidation



Flash State Diagram



Issues

Option 1: RMEW

- Time
- Wear

Option 2: Invalidation

- Over-provisioning
- Indirection

> SSD
> 1) #PBA > #LBA
> 2) Mapping layer

SSD Performance vs. Technology

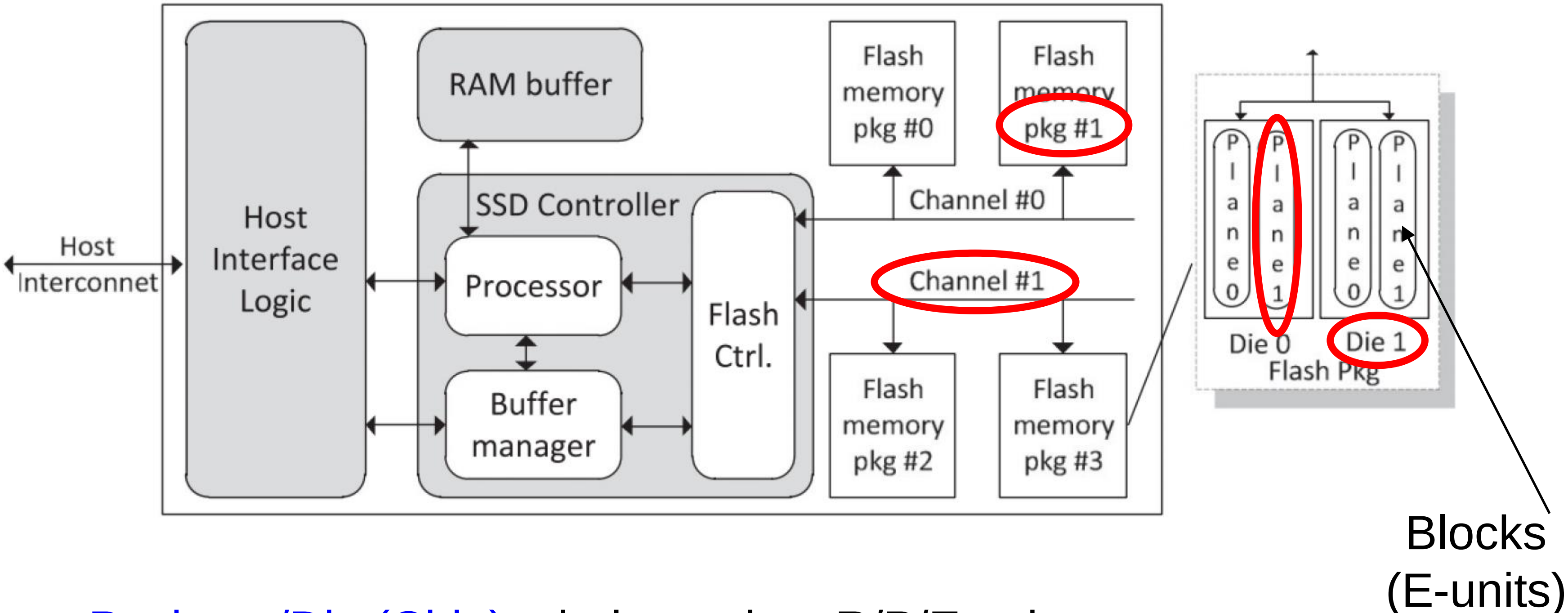
- SLC: 1 bit/cell
  - "single-level cell"
- MLC: 2 bits/cell
  - "multi-level cell"
- TLC: 3 bits/cell
  - "triple-level cell"

TABLE I

READ/PROGRAM/ERASE TIMES FOR SLC/MLC/TLC FLASH CHIPS [10].

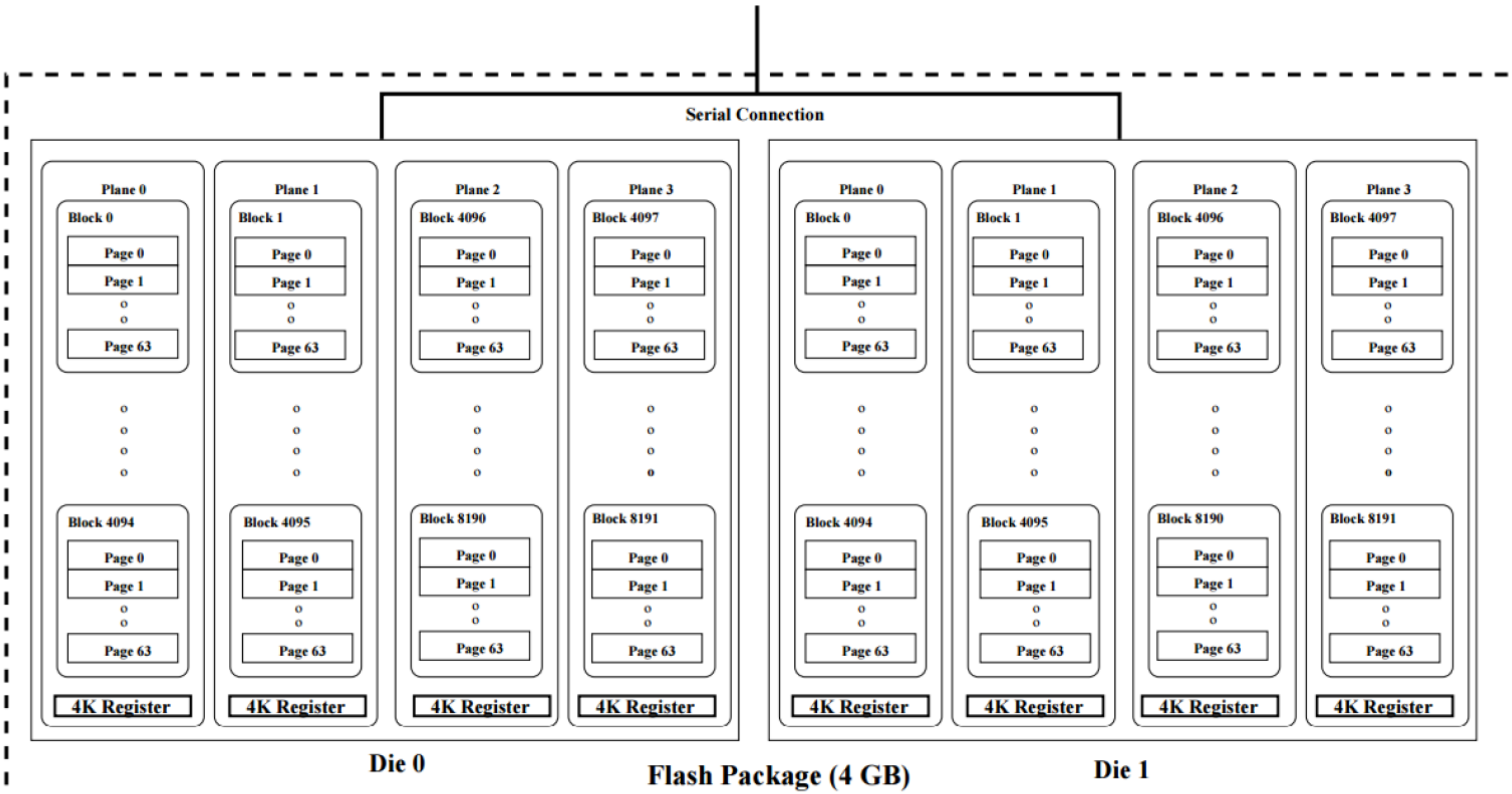
Operations	SLC chips	MLC chips	TLC chips
Random Read	20us	40us	80us
Program	260us	900us	2.3ms
Erase	2ms per block	2ms per block	10ms per block

SSD Architecture and parallelization



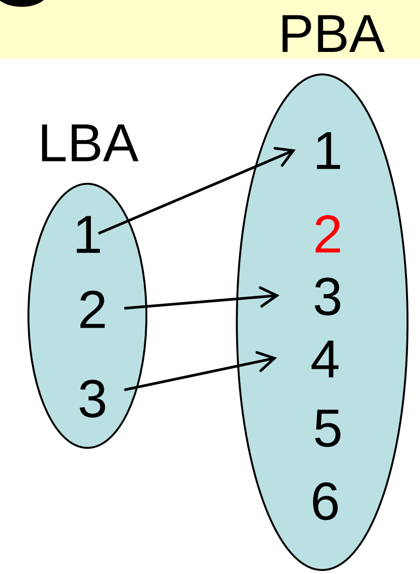
- Package/Die (Chip) – independent R/P/E units
- Plane – multiple planes of the same die can perform the same command (R/P/E) in parallel
- Channel – independent transfers between controller and packages
- Parallelization: perform commands in parallel
- Interleaving: Use channel for one package while the other is busy

Flash Package Example



Address Mapping

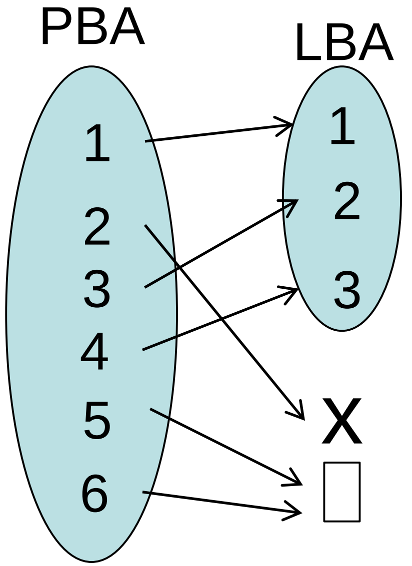
- Direct map (LBA→PBA)
  - Injective (1-1)



- Inverse map (PBA→LBA)
  - Range includes LBAs and

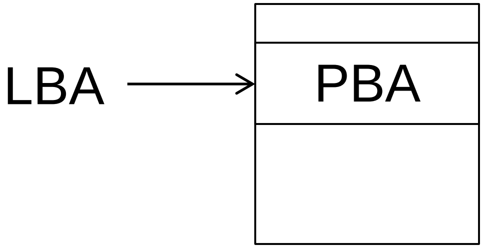
x: dirty

□: free

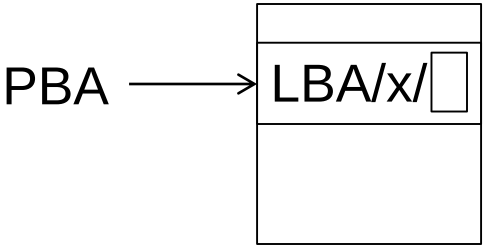


Flash Mapping Layer

- Direct map
  - Inquired on read
  - Updated on write



- Inverse map
  - Find free PBA
  - Updated on write
  - Inquired for “clean” operations



Clean Operations

- Reclaim dirty pages
- Also called **garbage collection**

Procedure:

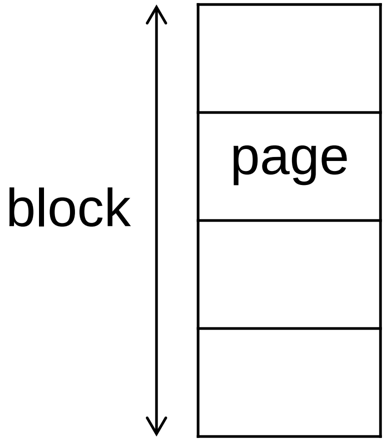
1) Choose an E_unit (how?)
2) Copy all used L_units to other E_unit(s)
3) Erase E_unit

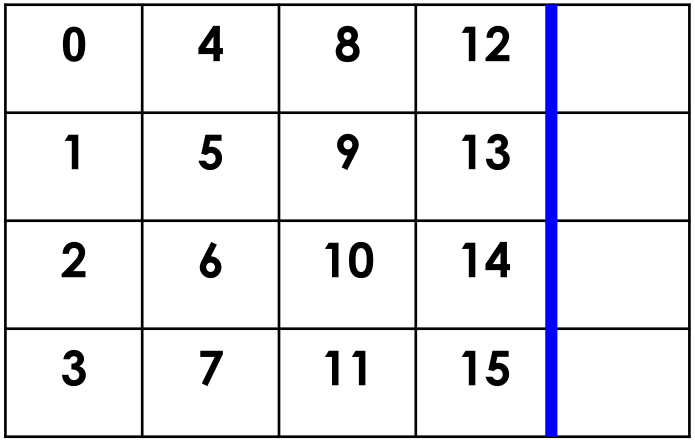
Objectives:

1) Minimize copy operations
2) Level the wear of E_units

Write Amplification (WA)

- SSD with 1 spare block





Write Amplification (WA)

- First 4 writes

- <u>Example:</u>
  - Writes: 0,4,8,12,

~~0~~	~~4~~	~~8~~	~~12~~	0
1	5	9	13	4
2	6	10	14	8
3	7	11	15	12

Write Amplification (WA)

- Incoming write: LBA 1

- <u>Example:</u>
  - Writes: 0,4,8,12,1

~~0~~	~~4~~	~~8~~	~~12~~	0
1	5	9	13	4
2	6	10	14	8
3	7	11	15	12

Choose for cleanup

Full!

Write Amplification (WA)

- Copied 2 valid pages in the cleaned block

- Example:
  - Writes: 0,4,8,12,1

2	~~4~~	~~8~~	~~12~~	0
3	5	9	13	4
1	6	10	14	8
	7	11	15	12

WA Analysis

- Definition – Write Amplification:

The ratio of the total number of internal writes to the number of externally-requested writes.

- Notation:

Write Amplification WA, $WA \geq 1$

- Notes
  1. Large is bad (more time, wear)
  2. Depends on mapping, workload

WA Analysis

- More notation:

T: # physical E_units

N_p: # P_units per E_unit

U: #L_units/N_p

UN_p L_units stored on TN_p P_units

S_f: spare factor $S_f = \frac{T-U}{T}$ $0 \le S_f < 1$

ρ: over-provisioning $\rho = \frac{T}{U} - 1$ $\rho \ge 0$

Effect of Spare

~~0~~	~~4~~	~~8~~	~~12~~	0
1	5	9	13	4
2	6	10	14	8
3	7	11	15	12

Little spare

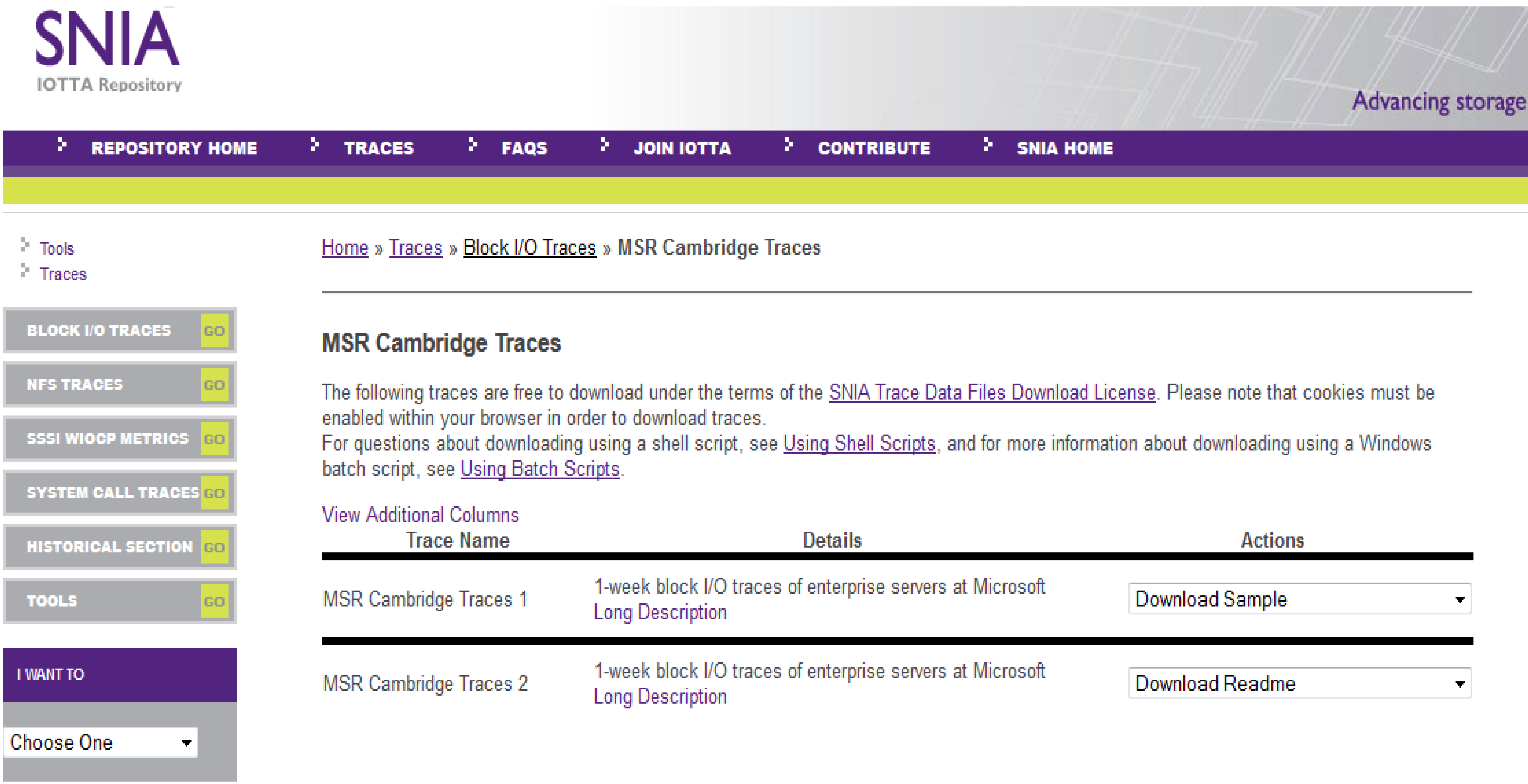
~~0~~	~~4~~	~~8~~	~~12~~	0	5
1	~~5~~	~~9~~	13	4	14
2	6	~~10~~	~~14~~	8	10
3	7	11	15	12	9

More spare

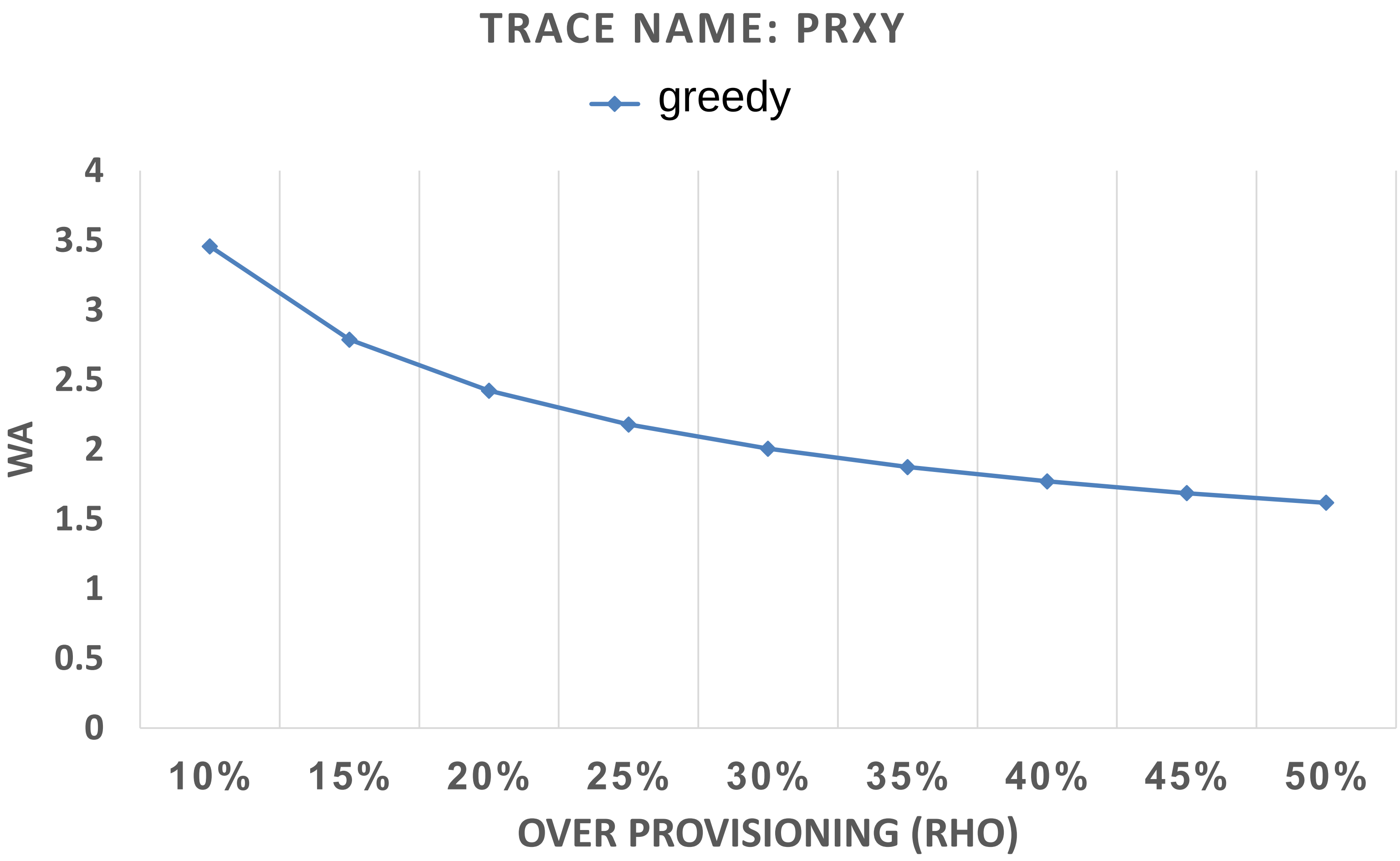
Clean (Garbage Collection) Policies

1. LRU – least-recently written
   – Pick oldest in the "log" of E_units
2. Greedy
   – Pick the E_unit with max # dirty P_units
   – Also called min-valids

Evaluation with Traces

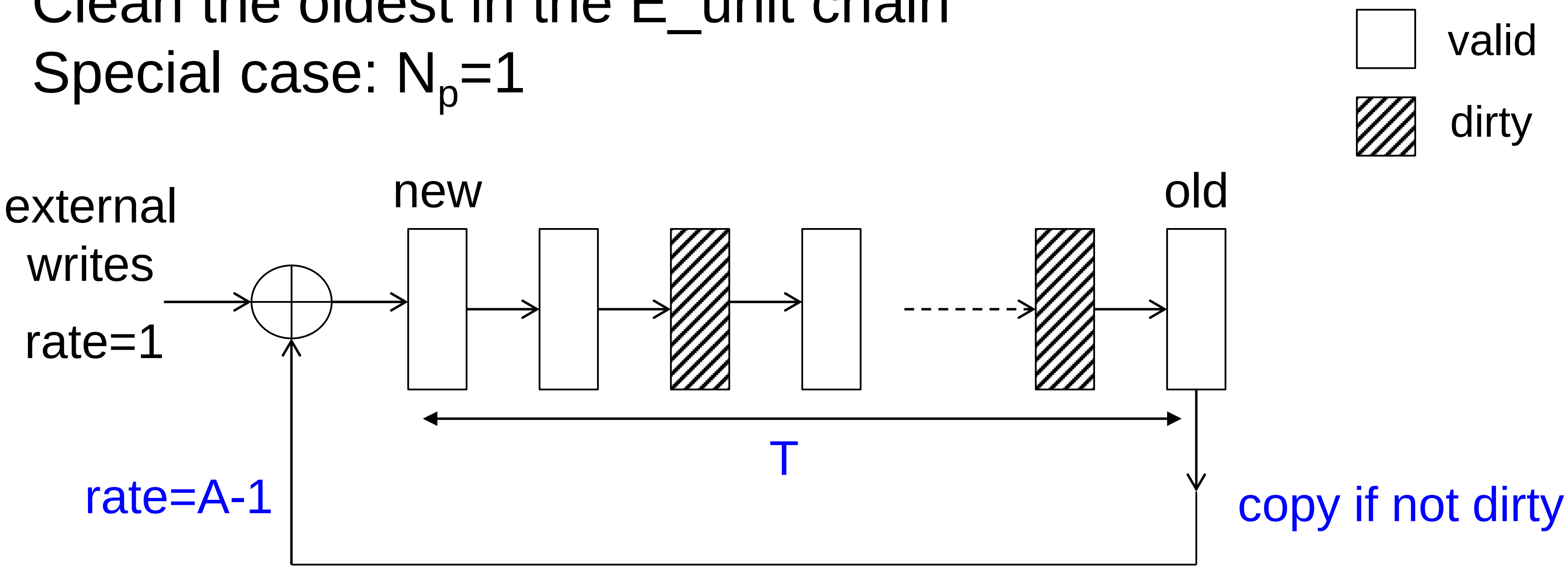


Trace Results - Greedy



LRU Cleaning - Model

- Clean the oldest in the E_unit chain
- Special case: N_p=1

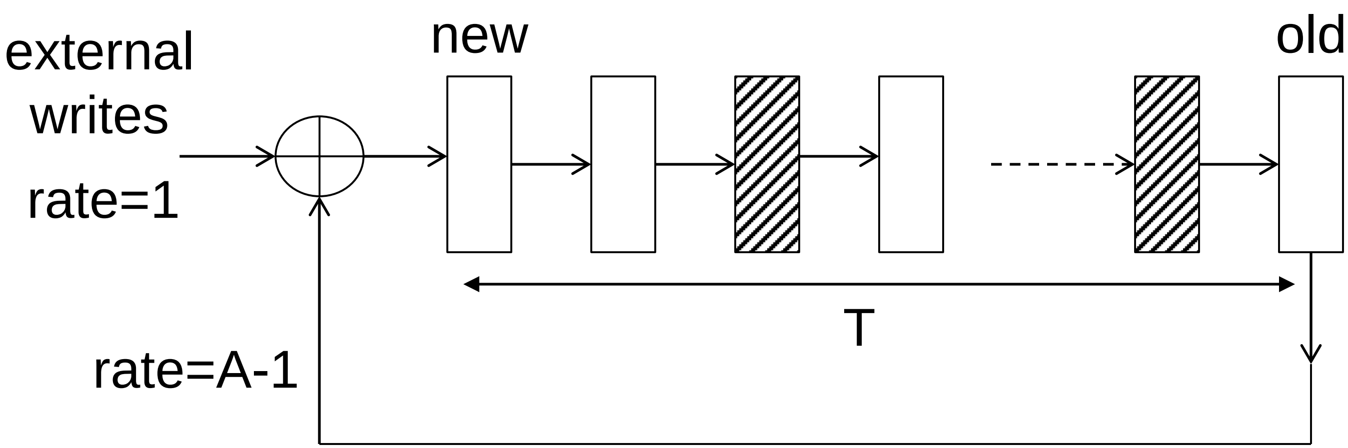


Facts:

- T units in chain, U of them are valid
- A copy of a unit happens if **still** valid when oldest

LRU Cleaning - Analysis

- Special case: N_p=1
- Uniform workload (random write)

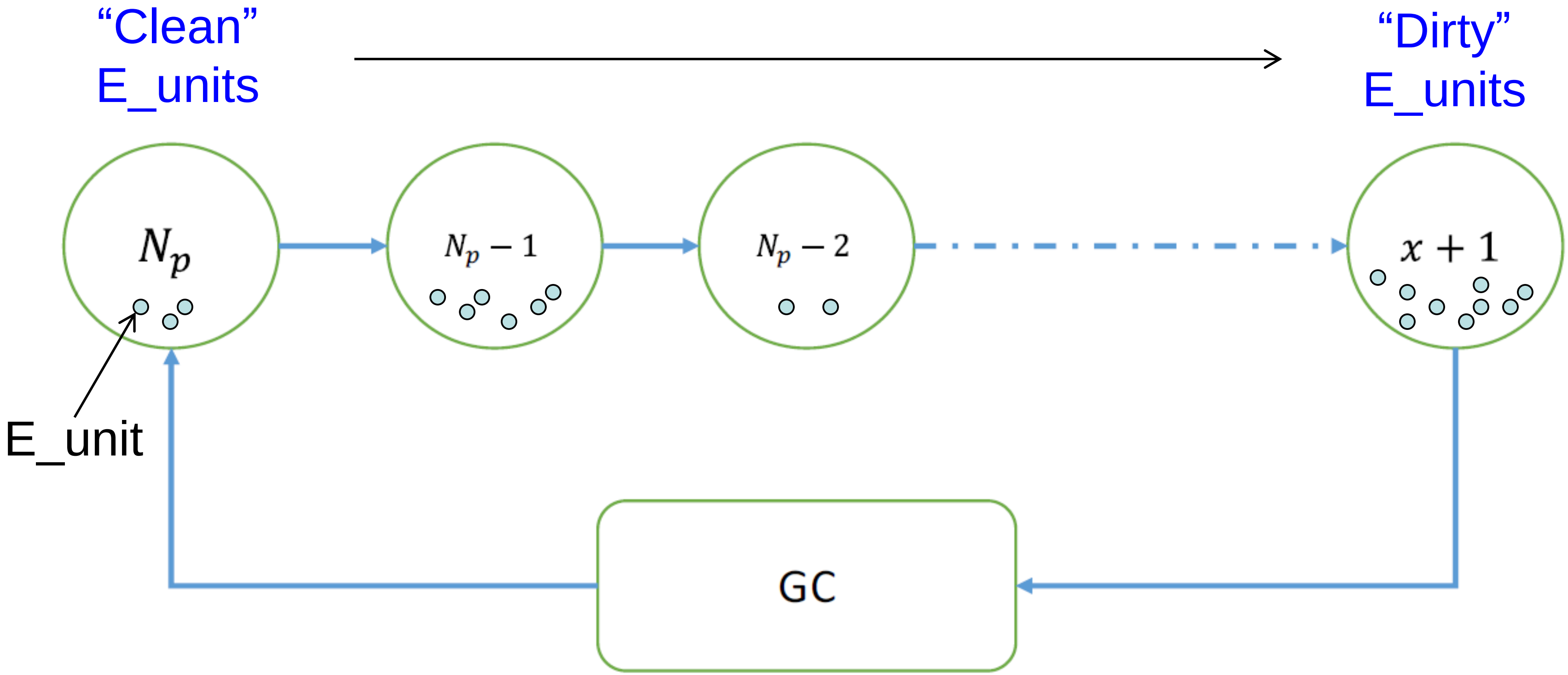


Facts:

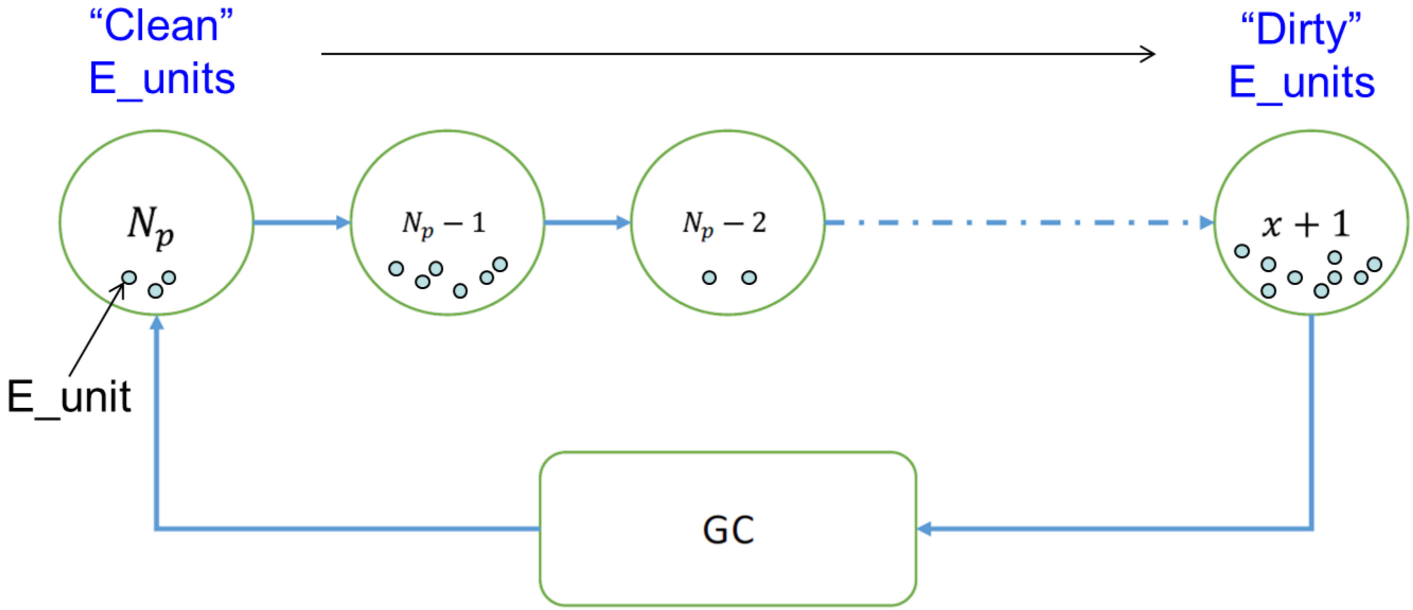
- Pr{host write is addressed to valid E_unit i} **= 1/U**, for all i
- # host writes in a full cycle **M=T/A**
- The same L_unit can cycle many times without host update
- Find A (in expectation) 

Greedy Cleaning - Model

- Clean the E_unit with min # valid (equiv. max # dirty)
- General N_p, uniform workload
- Need to calculate the expected min # valid **x**
- Markov model: states represent # **valid in E_unit**

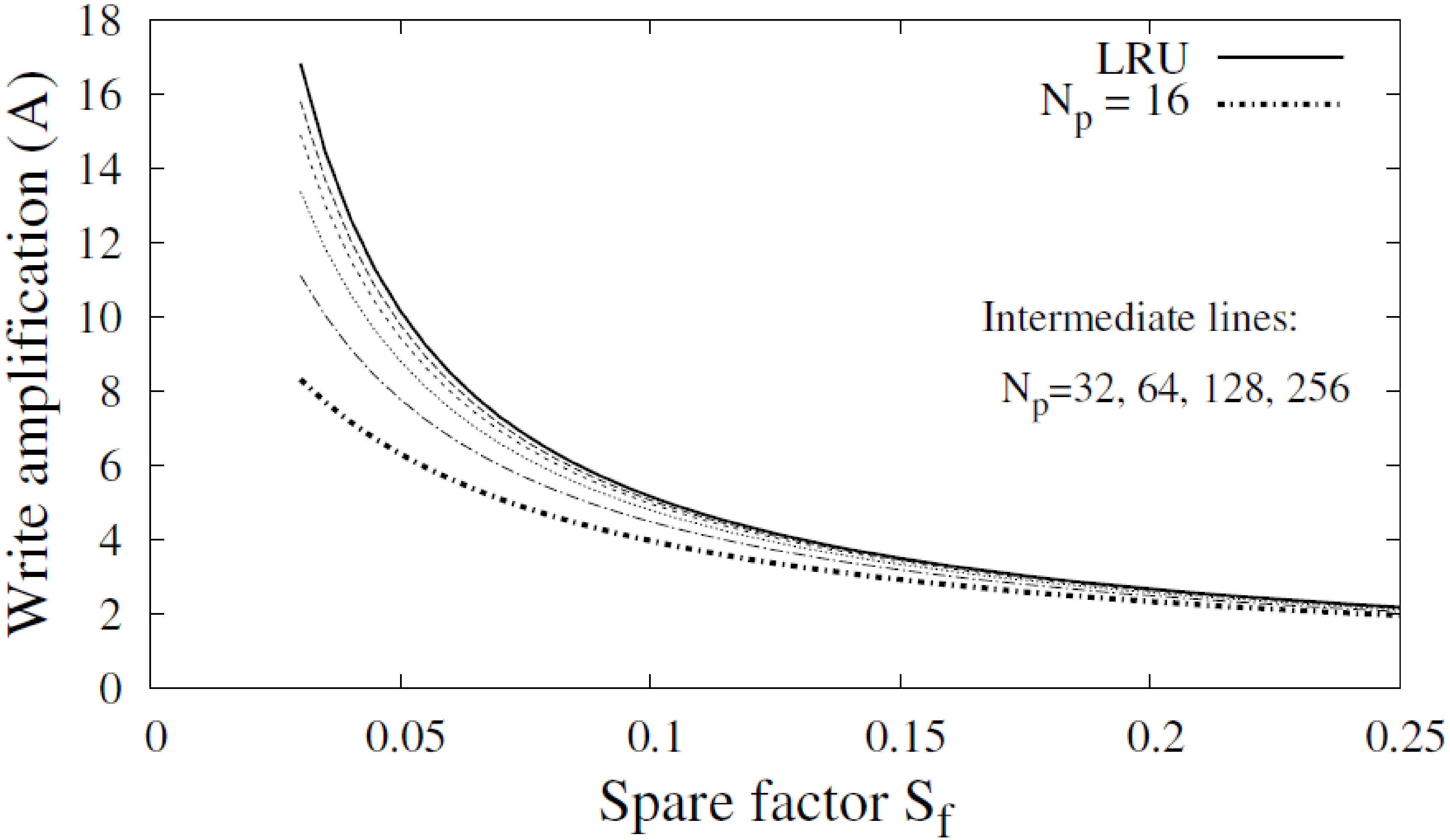


The Greedy Markov Model



- Define fraction of E_units in state i $= f_i$
- Total number of valid L_units in E_units at state i $= iTf_i$
- Random writes → Transition rate from state i to i-1 $= \frac{iTf_i}{UN_p}$
- GC transition rate $= \frac{1}{N_{p-x}}$
- Find A from steady-state considerations

Analytic Results



Summary

- WA reduces with amount of spare
- LRU GC: simple
- Greedy GC: optimal
- Greedy approaches LRU with large N_p
- Another advantage of LRU:

Wear leveling