Operating System Practice

Che-Wei Chang

chewei@mail.cgu.edu.tw

Department of Computer Science and Information Engineering, Chang Gung University

Flash Memory and Phase Change Memory

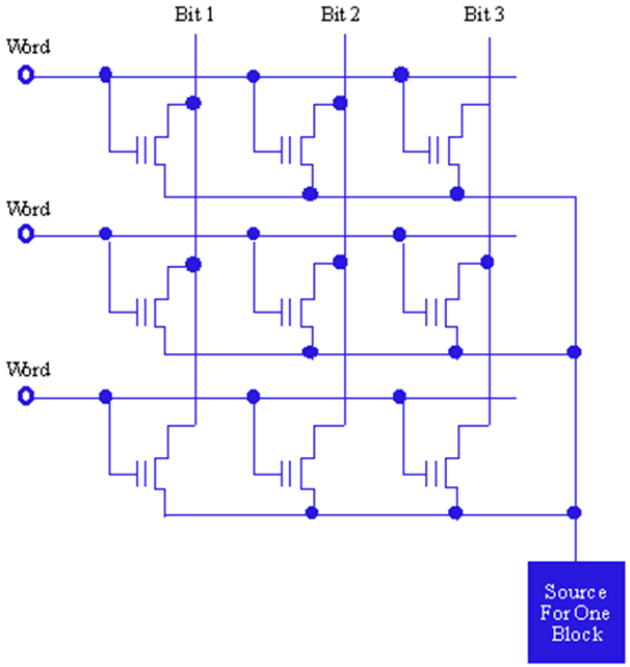
Reference: Prof. Tei-Wei Kuo, NTU and Dr. Yuan-Hao Chang, Academia Sinica

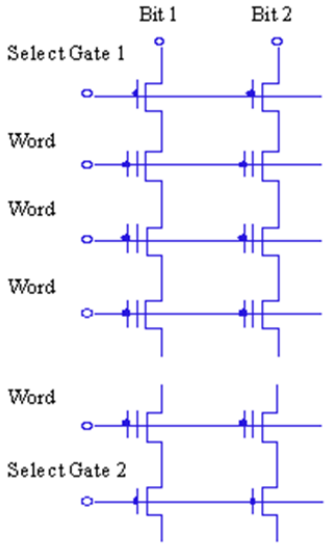
Trends – Market and Technology

- Diversified Application Domains
  - Portable Storage Devices
  - Consumer Electronics
  - Industrial Applications
- Competitiveness in the Price
  - Dropping Rate and the Price Gap with HDDs
- Technology Trend over the Market
  - Improved density
  - Degraded performance
  - Degraded reliability

NOR and NAND Flash

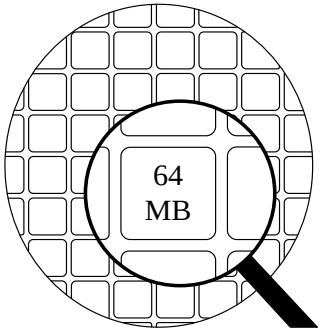
- NAND accesses each cell through adjacent cells, while NOR allows for individual access to each cell
- The cell size of NAND is almost half the size of a NOR cell





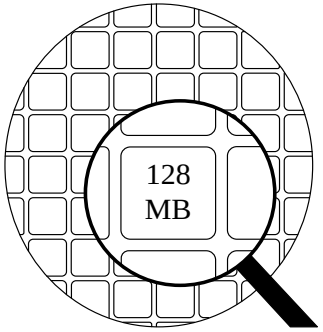
Single-Level Cell (SLC) vs Multi-Level Cell (MLC) Flash

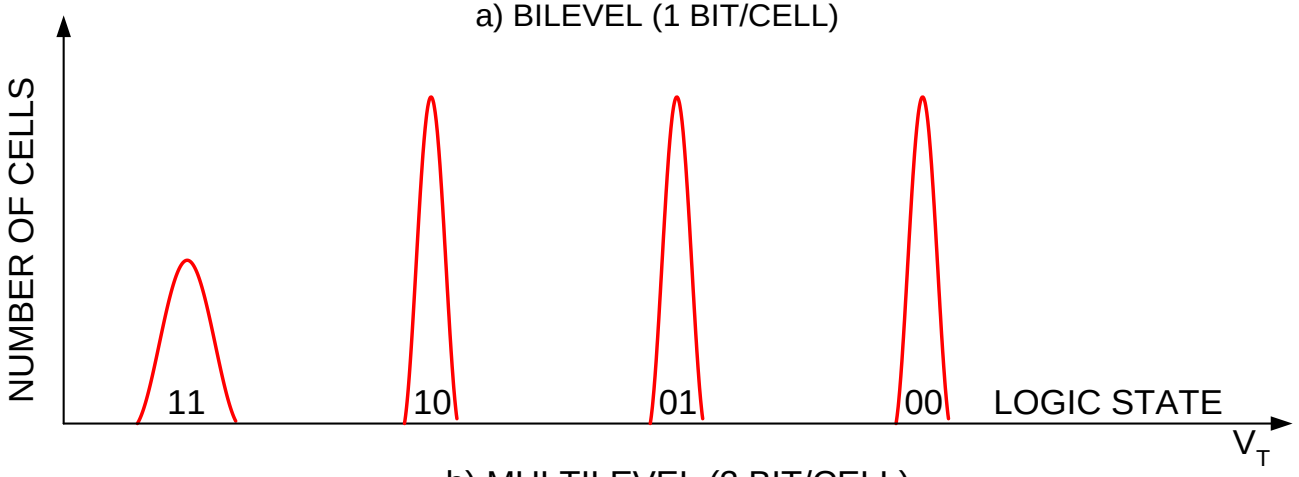
SLC Flash



a) BILEVEL (1 BIT/CELL)

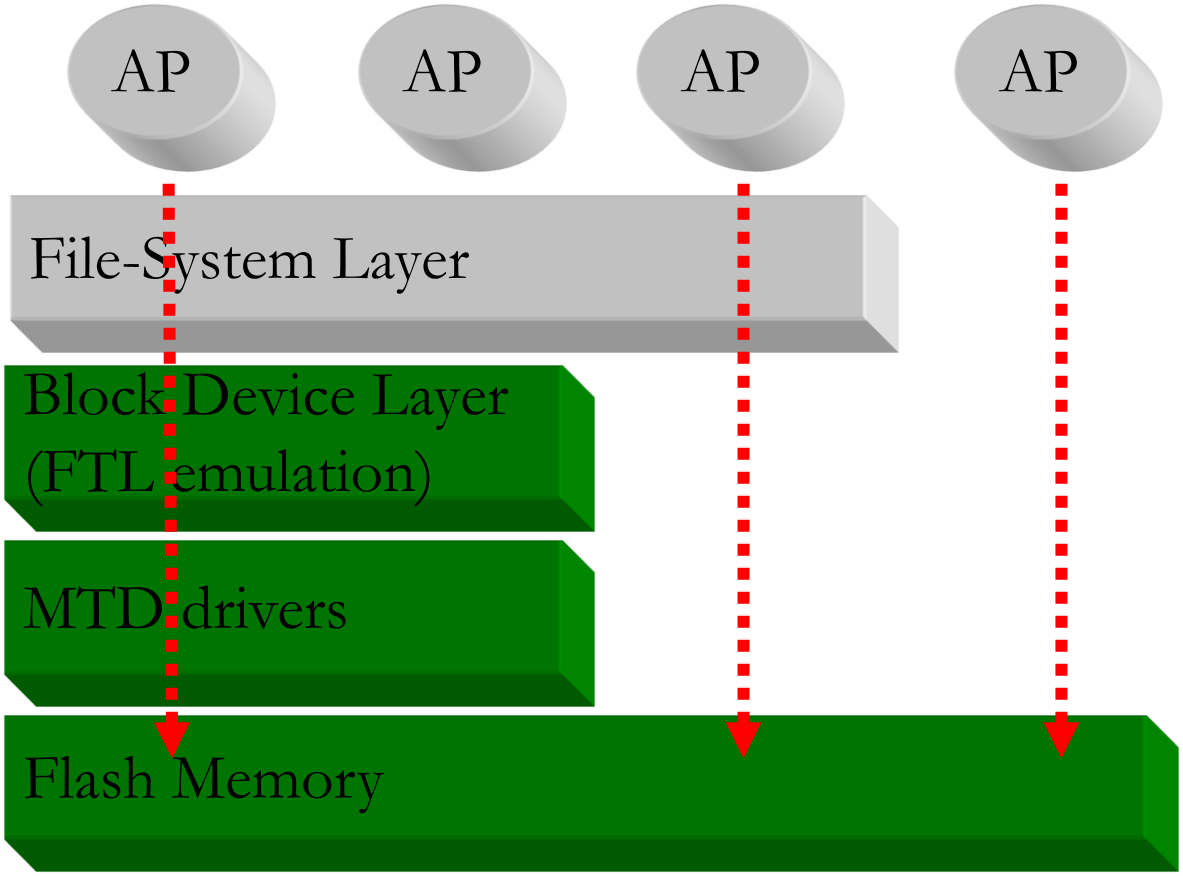
MLC Flash





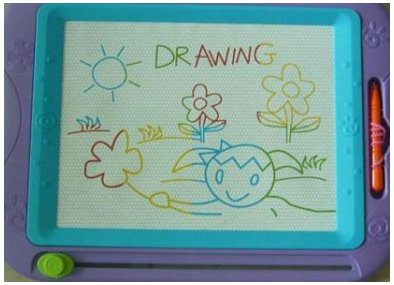
b) MULTILEVEL (2 BIT/CELL)

System Architectures for Flash Management



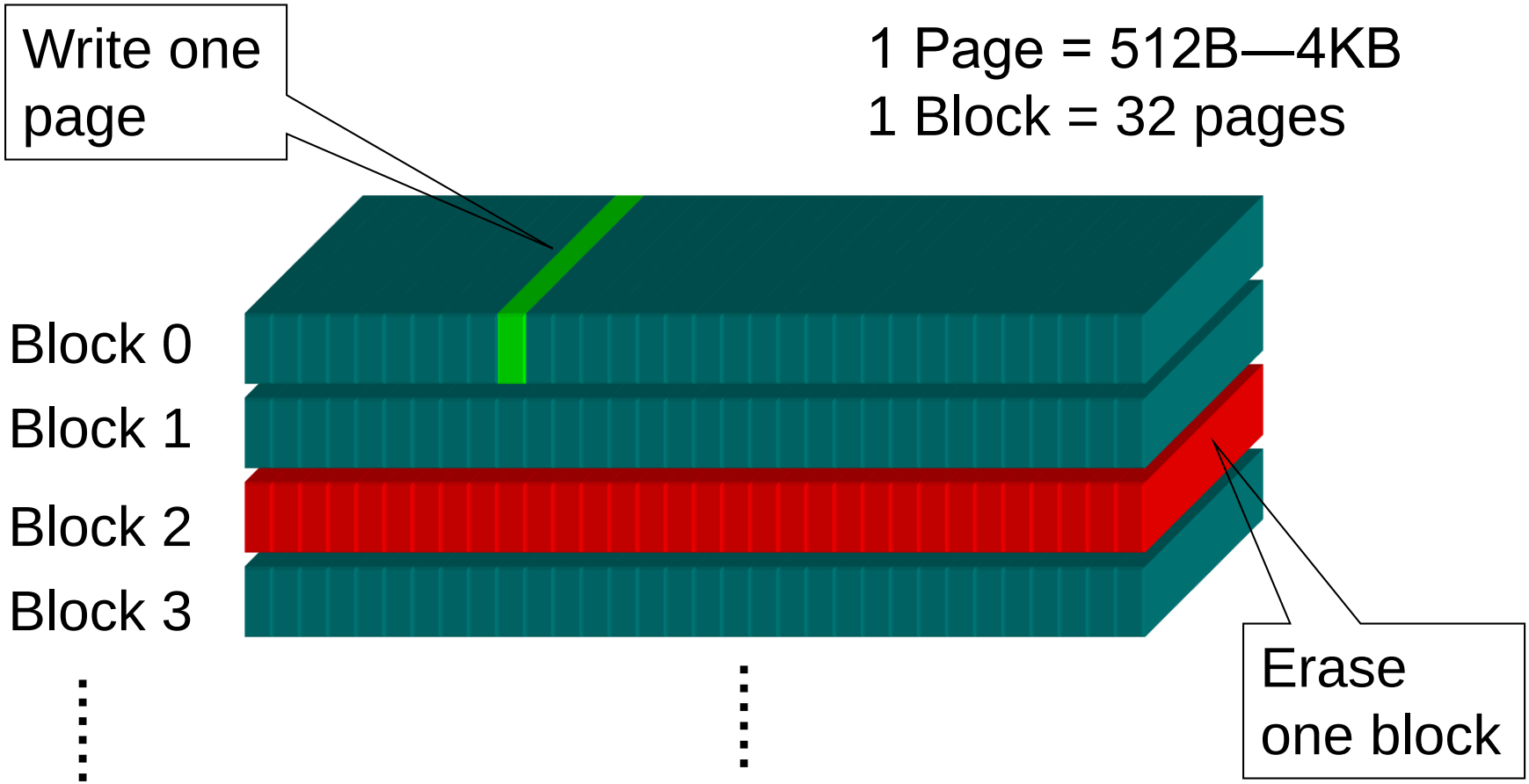
Flash-Memory Characteristics

- Write-Once
  - No writing on the same page unless its residing block is erased
  - Pages are classified into valid, invalid, and free pages
- Bulk-Erasing
  - Pages are erased in a block unit to recycle used but invalid pages

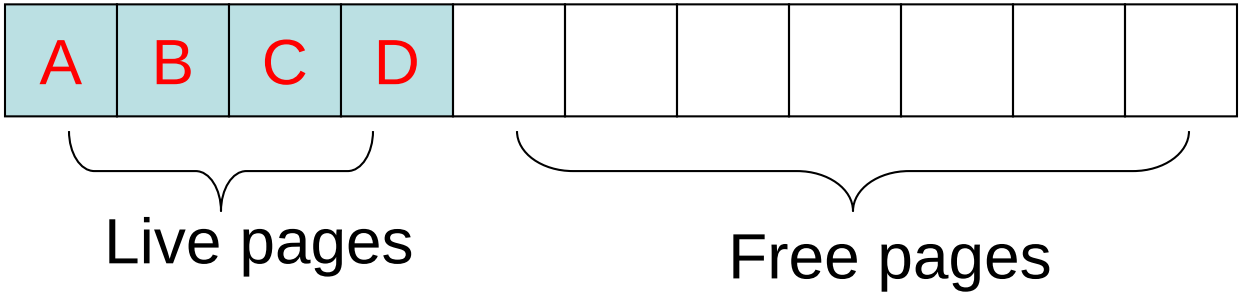


- Wear-Leveling
  - Each block has a limited lifetime in erasing counts

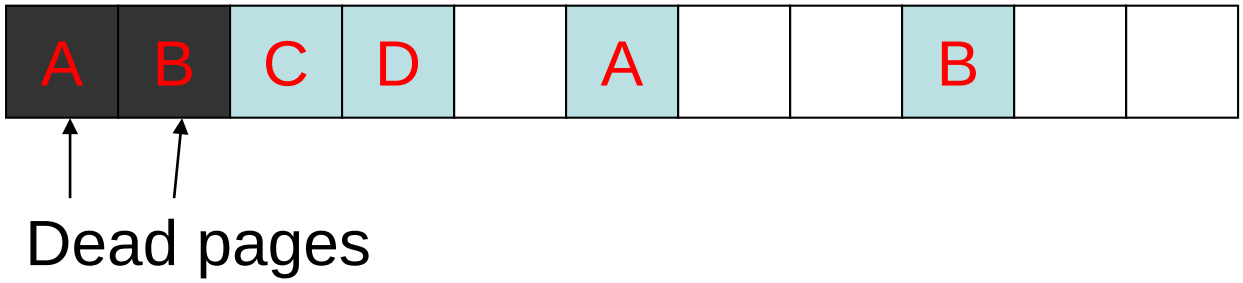
Page Write and Block Erase



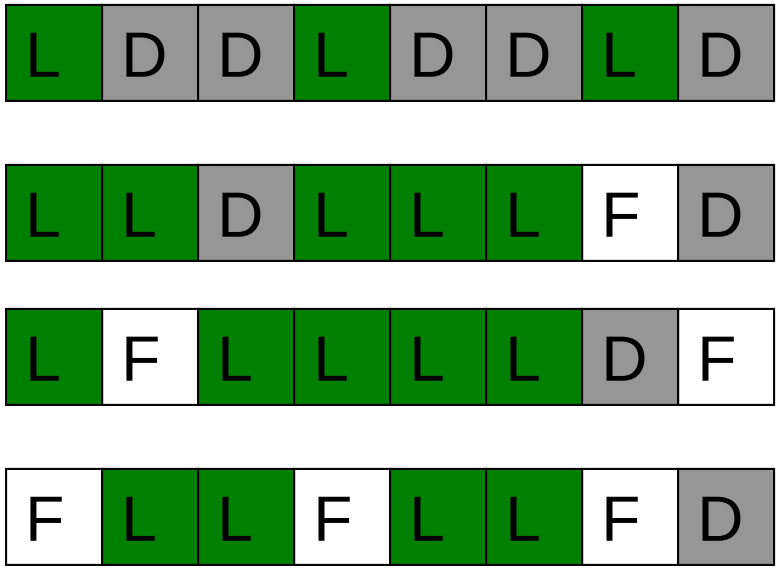
Out-Place Update



Suppose that we want to update data A and B…



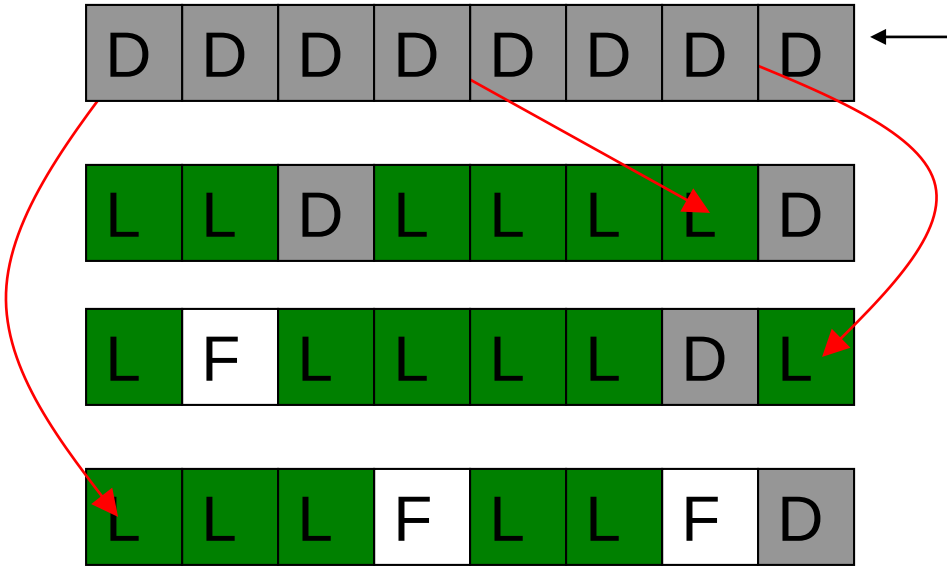
Garbage Collection (1/3)



This block is to be recycled
(3 live pages and 5 dead pages)

- A live page
- A dead page
- A free page

Garbage Collection (2/3)



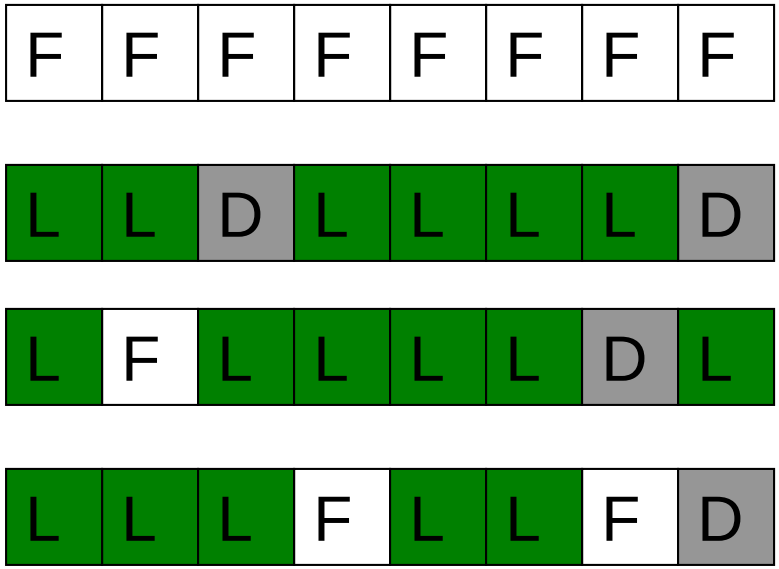
Live data are copied to somewhere else

A live page

A dead page

A free page

Garbage Collection (3/3)

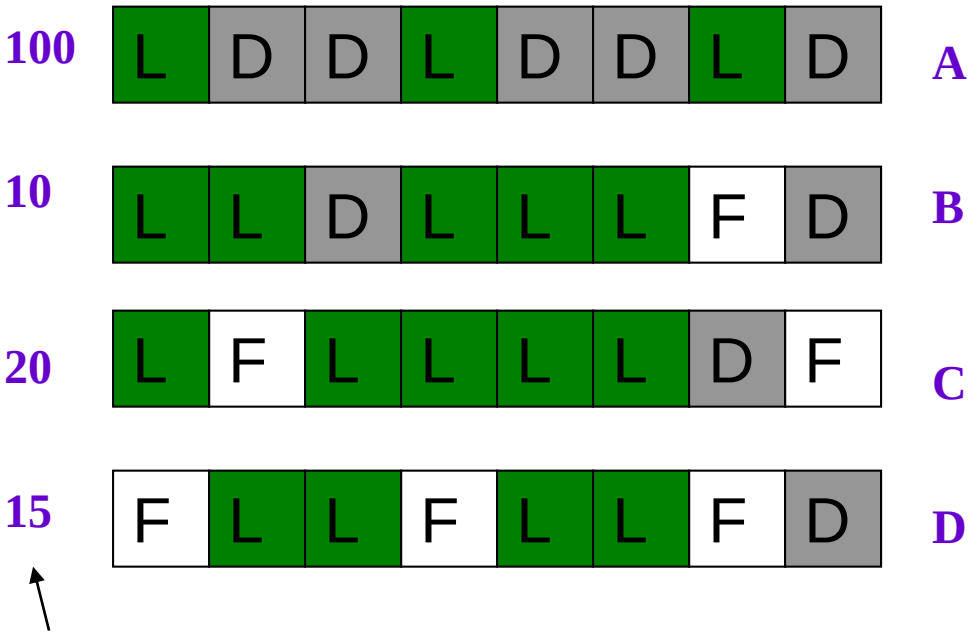


The block is then erased

Overheads:
- live data copying
- block erasing

A live page
A dead page
A free page

Wear-Leveling

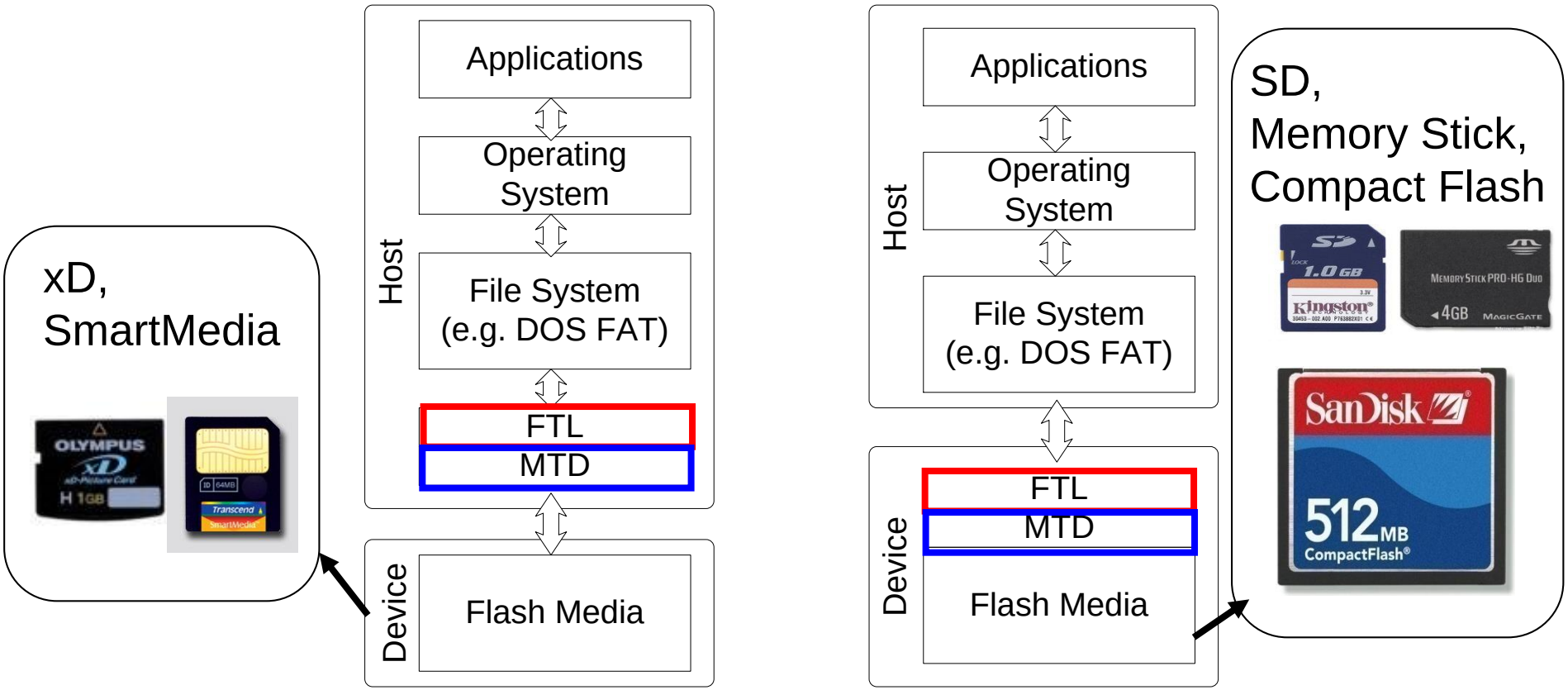


Erase cycle counts

Wear-leveling might interfere with the decisions of the block-recycling policy

- A live page
- A dead page
- A free page

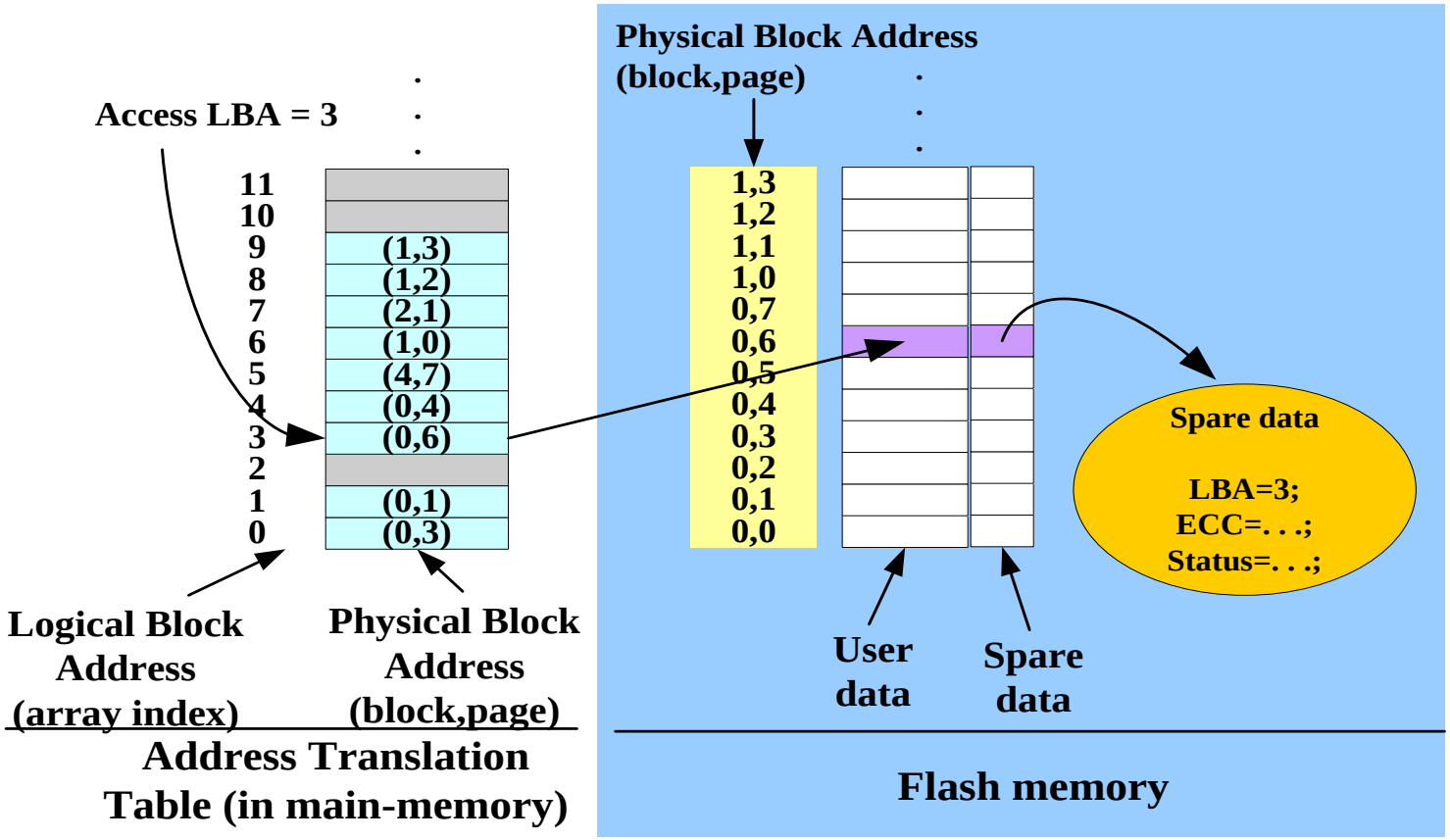
Flash Translation Layer



*FTL: Flash Translation Layer, MTD: Memory Technology Device

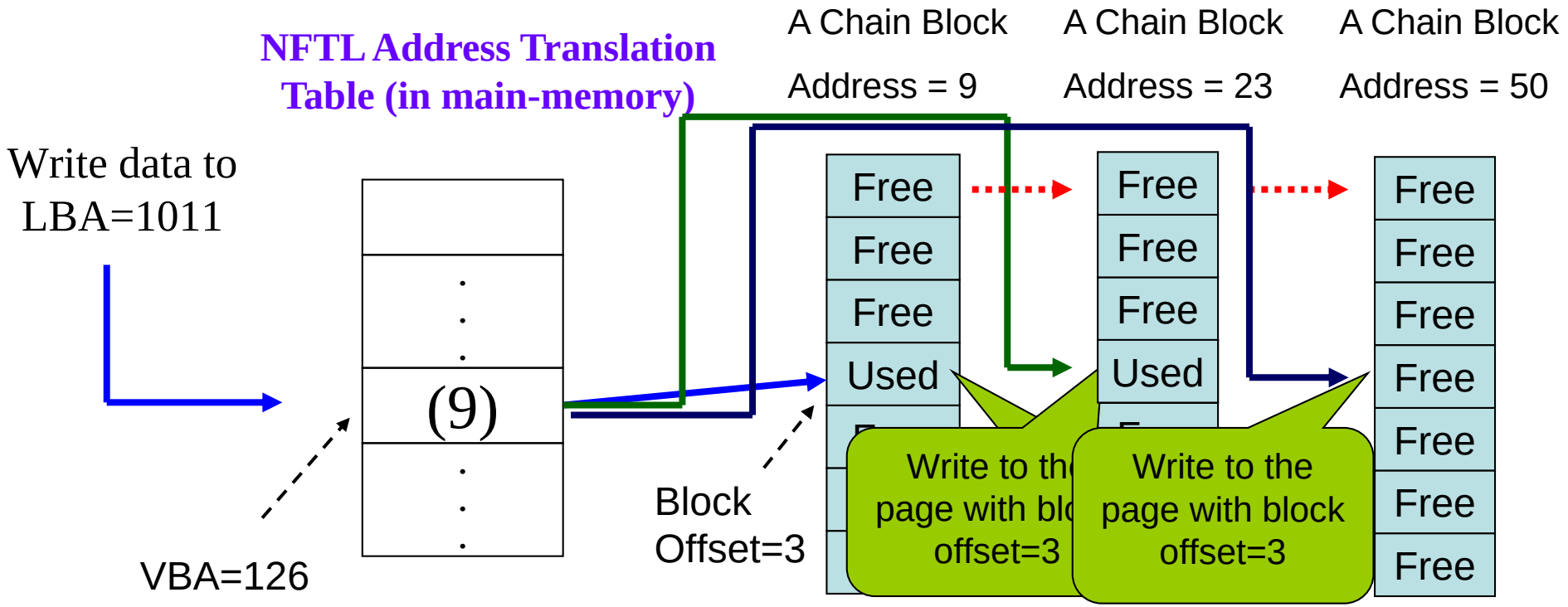
Policies – FTL

- FTL adopts a page-level address translation mechanism



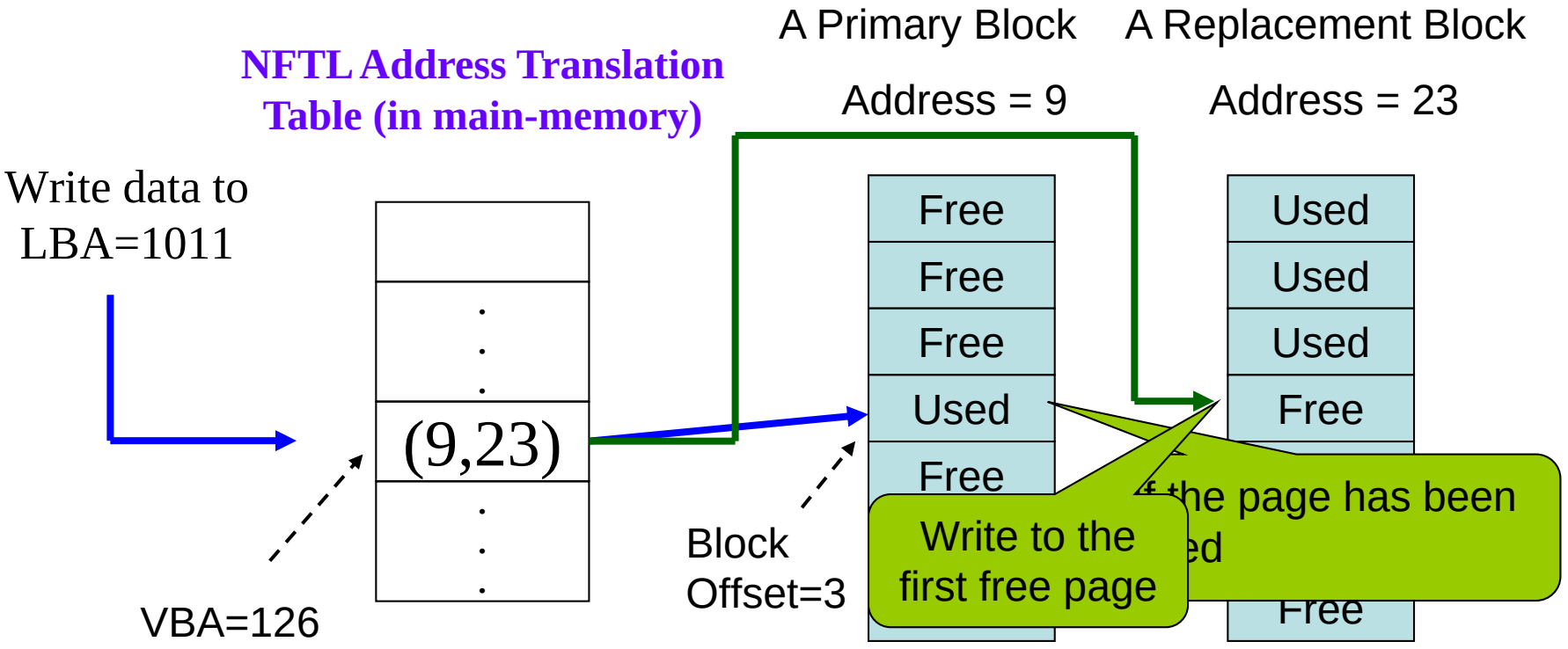
Policies – NFTL (Type 1)

- A logical address under NFTL is divided into a virtual block address and a block offset, e.g., LBA=1011 => virtual block address (VBA) = 1011 / 8 = 126 and block offset = 1011 % 8 = 3



Policies – NFTL (Type 2)

- A logical address under NFTL is divided into a virtual block address and a block offset, e.g., LBA=1011 => virtual block address (VBA) = 1011 / 8 = 126 and block offset = 1011 % 8 = 3

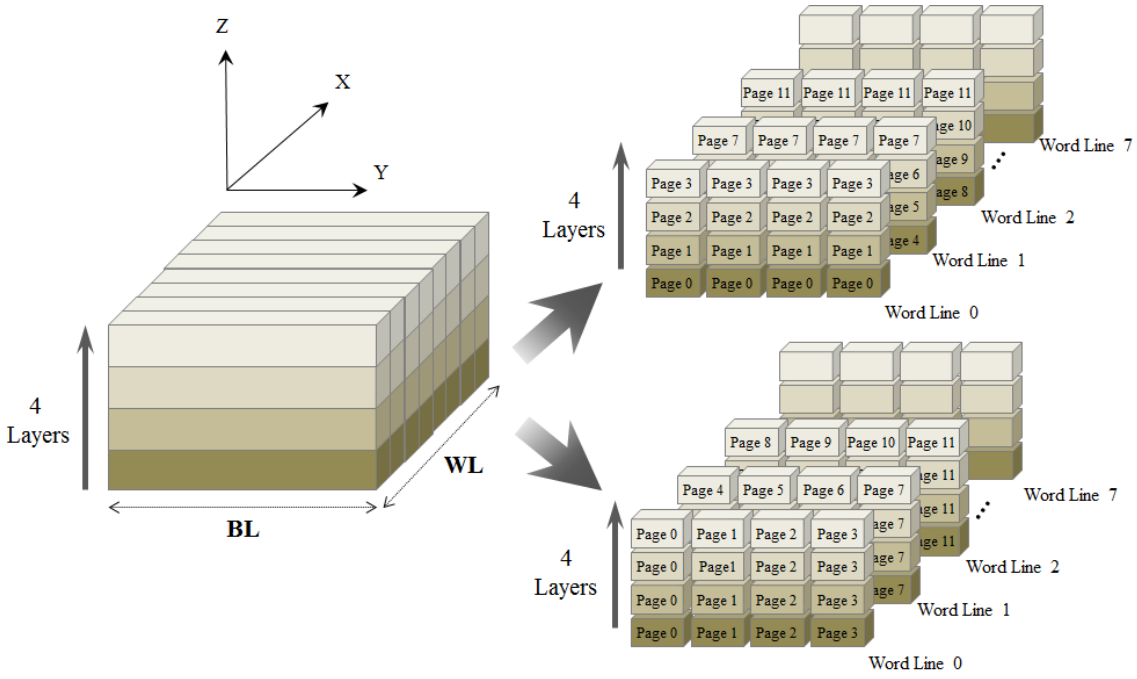


Challenges and Research Topics of Flash Memory Designs

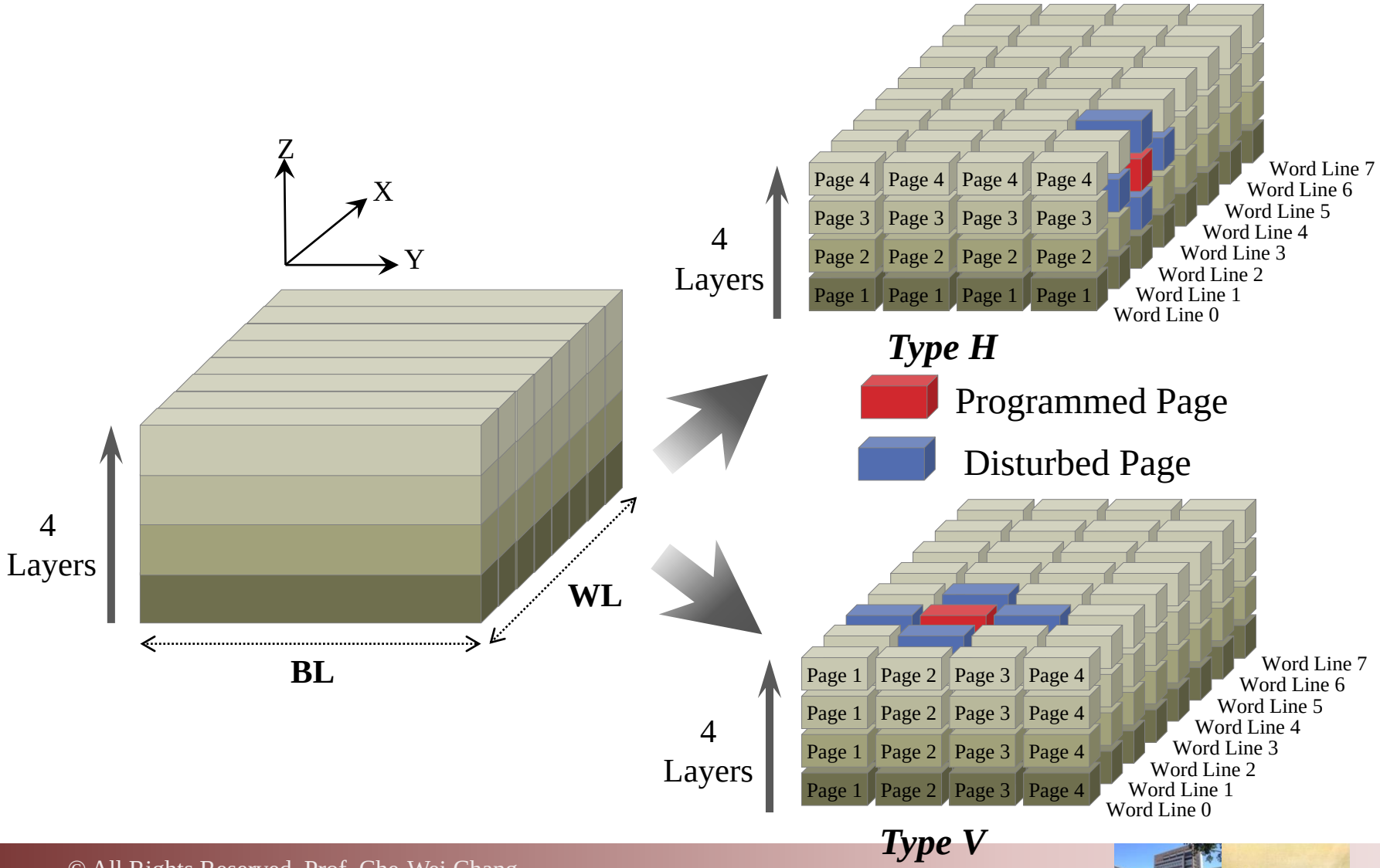
- Performance
  - Reduce the overheads of Flash management
  - Reduce the access time to data
  - Reduce the garbage collection time
- Reliability
  - Error correcting codes
  - Log systems
- Endurance
  - Dynamic wear-leveling
  - Static wear-leveling

3D Flash Memory

- 3D flash memory provides a good chance to further scale down the feature size and to reduce the bit cost.
  - Deliver very large storage space
  - Worsen program disturbance

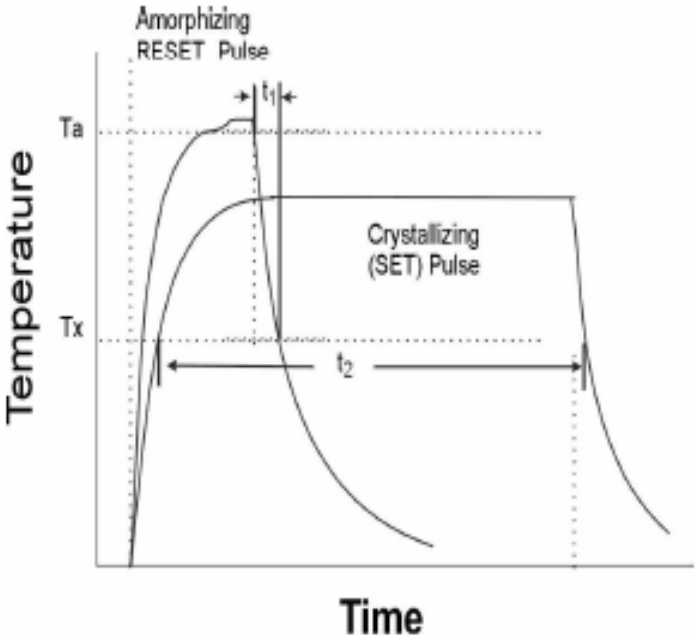


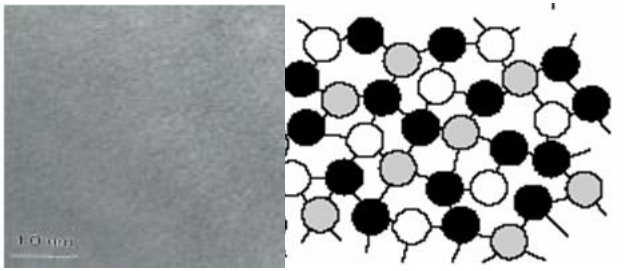
Deteriorated Disturb on 3D Flash



Phase Change Memory (PCM)

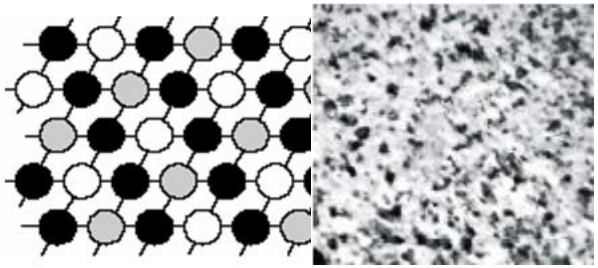
- PCM is a non-volatile memory (NVRAM)
- PCM employs a reversible phase change in materials to store information.
- PCM exploits differences in the electrical resistivity of a material in different phases





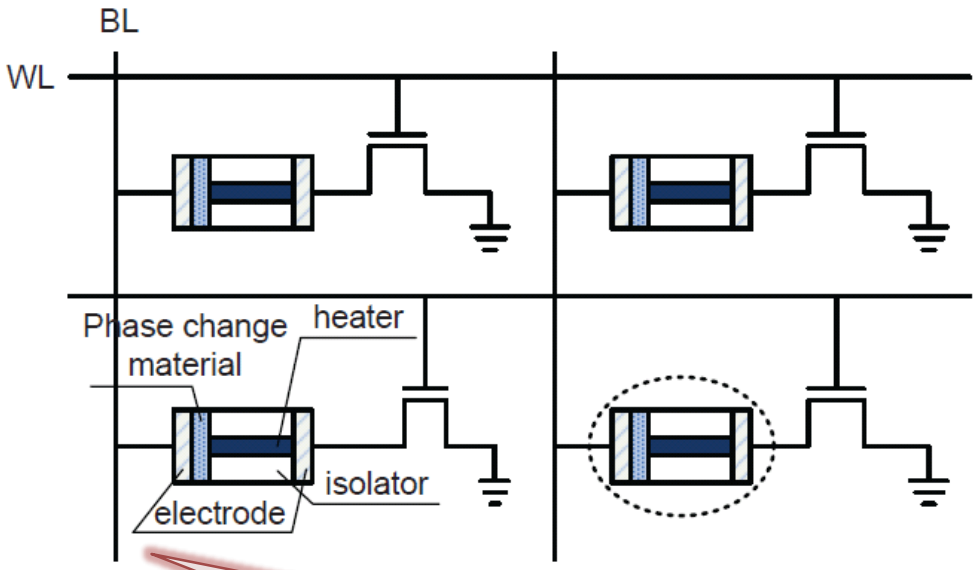
Reset - Power Limiter

Set - Performance Limiter



PCM Cell Array and Characteristics

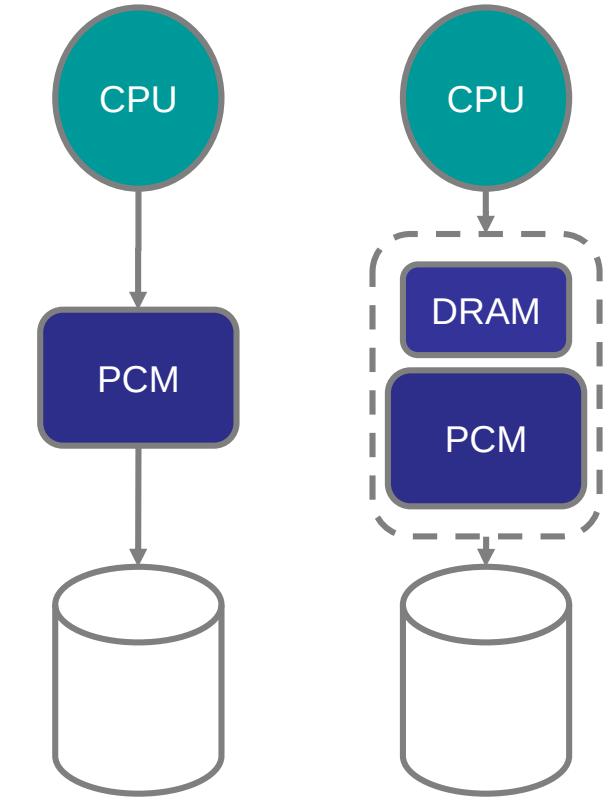
- Pros of PCM
  - Non-volatility
  - Bit-addressability
  - High scalability
  - No dynamic power
- Cons of PCM (compared to DRAM)
  - Low performance on writes
  - High energy consumption on writes
  - Low endurance



The read and write (SET and RESET) operations of a PCM cell require different current and voltage levels on the bitline, and take different amount of time to complete.

PCM as Main Memory (1/2)

- Take advantage of its scalability and byte-addressability
- Challenges
  - Limited PCM endurance
  - Asymmetric read/write performance



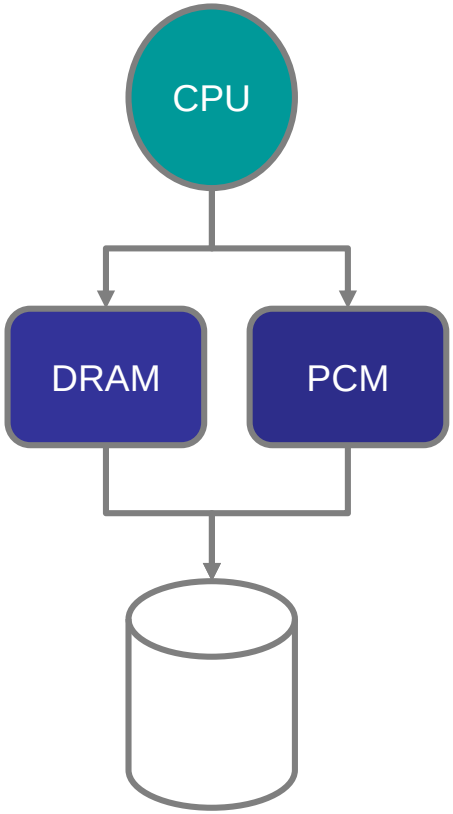
System with PCM

System with hybrid memory : DRAM as cache

PCM as Main Memory (2/2)

- Take advantage of its non-volatility and byte-addressability

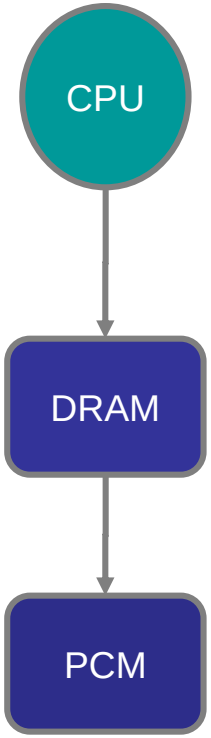
- Challenges:
  - What data should be in DRAM
  - What data should be in PCM
  - How to reuse data after power-off



System with hybrid memory

PCM as Storage

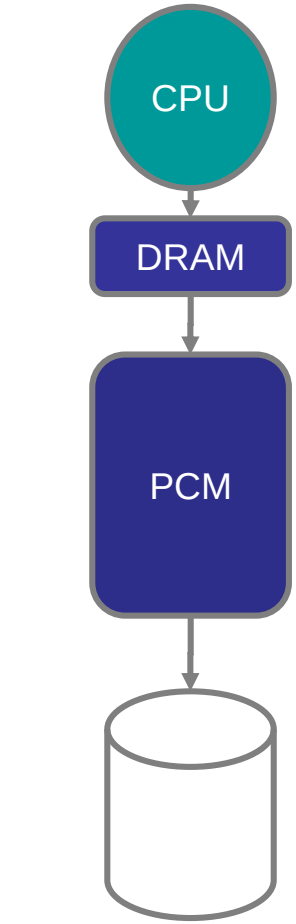
- Take advantage of its non-volatility and high performance
- Challenges
  - Modern file systems have been built around the assumption that persistent storage is accessed via block-based interface
  - How to exploit its properties of persistent, byte-addressable memory



System with PCM

PCM as Storage Class Memory

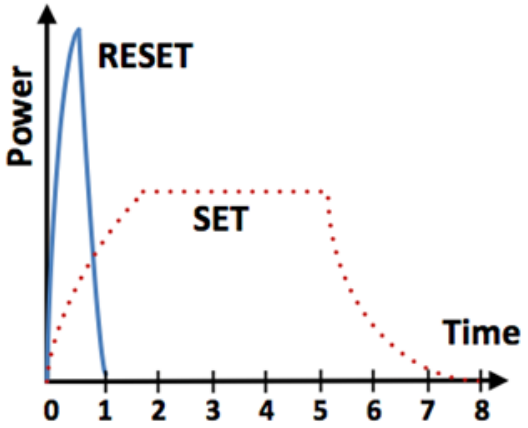
- IBM first proposed the idea of Storage Class Memory (SCM)
- PCM is the candidate of SCM
- SCM blurs the distinction between
  - Memory (fast, expensive, volatile) and
  - Storage (slow, cheap, non-volatile)



System with PCM as SCM

Issues of Using PCM

- Write asymmetry
  - Reset
    - High instant power with short time
  - Set
    - Low power with long time
- Write latency
- Endurance issue

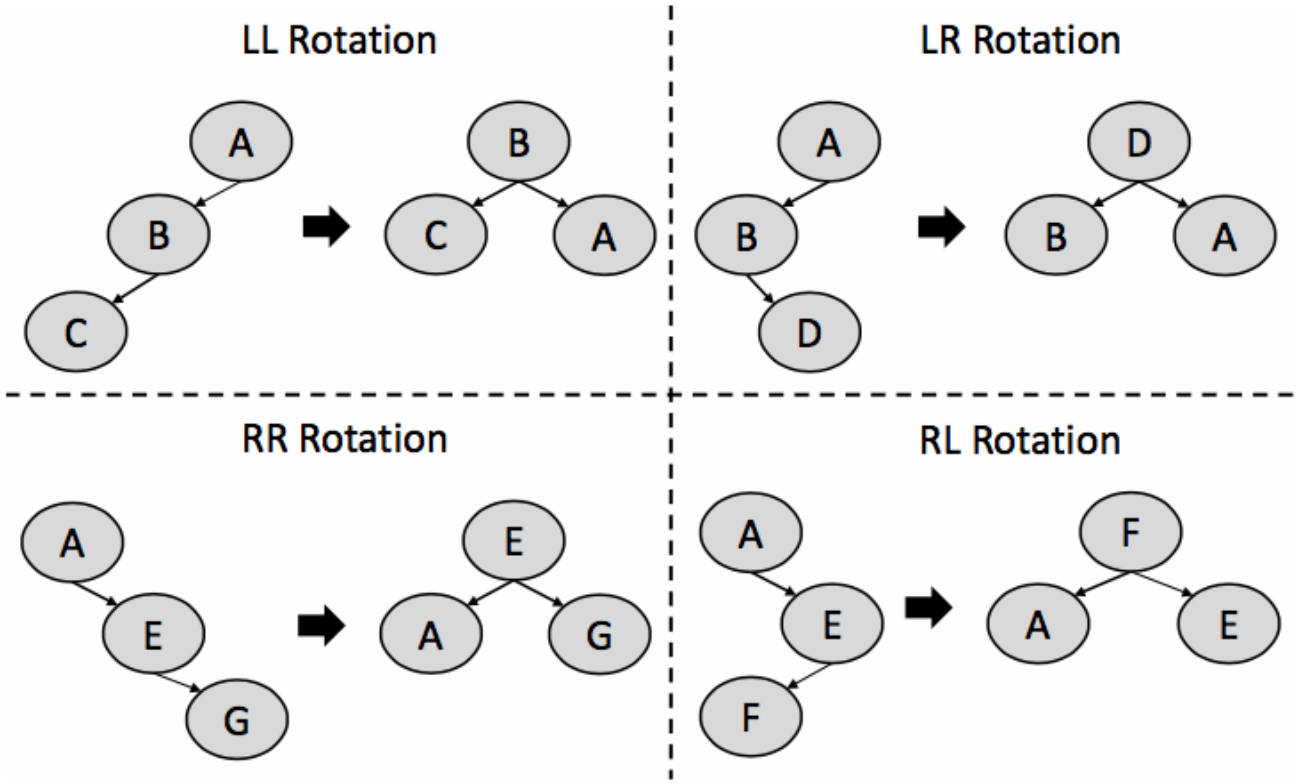


Types & Attributes	DRAM	PCM
Non-volatility	No	Yes
Bit alterability	Yes	Yes
Retention time	$\sim$ 60 ms	$>$ 10 years
Density	20 – 32 nm	$<$ 20 nm
Write endurance	$> 10^{15}$ cycles	$10^{6} - 10^{8}$ cycles
Write latency	20 – 50 ns	150 ns
Read latency	50 ns	50 ns

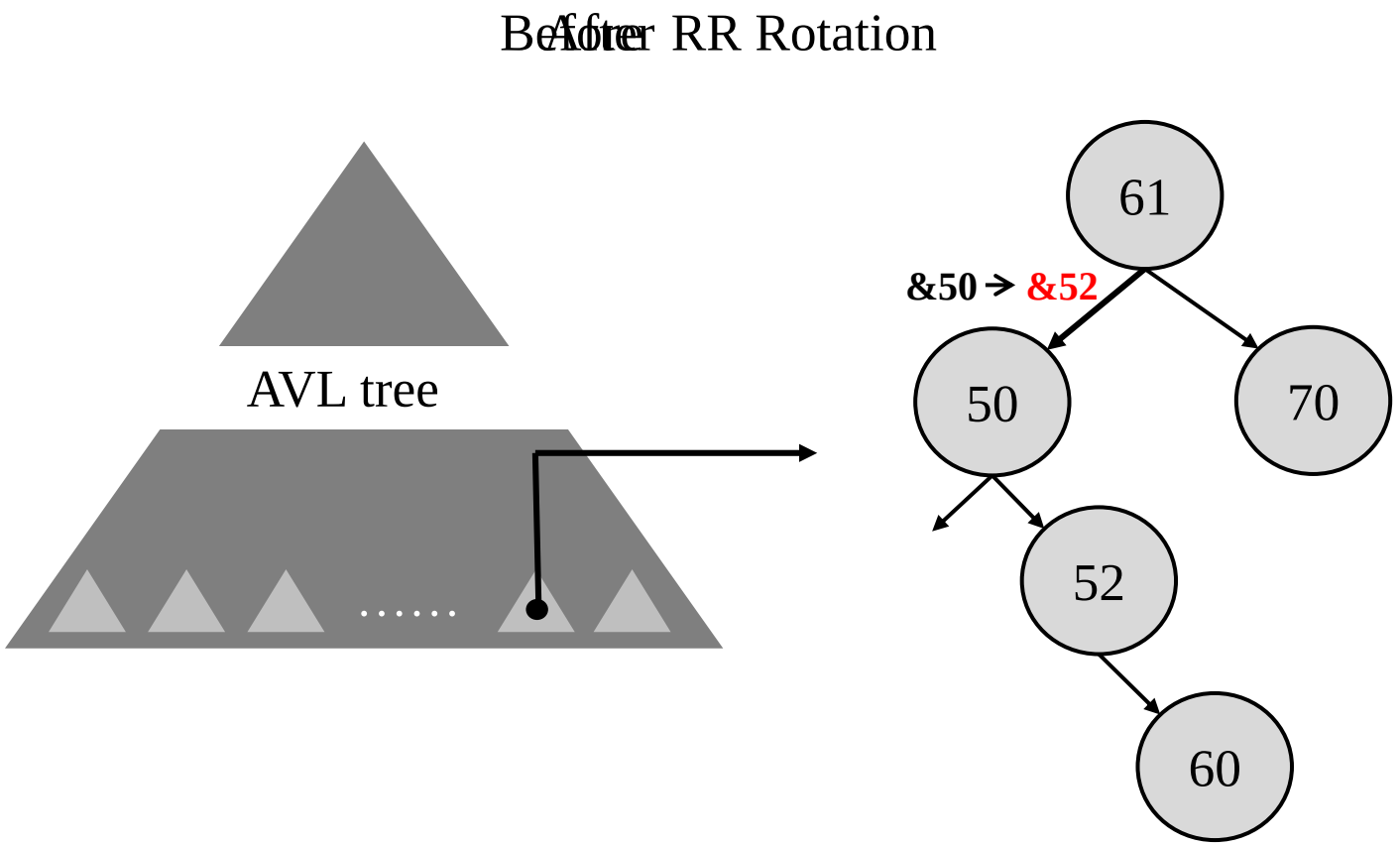
Write Reduction on PCM

- Big/massive data applications demand extremely large main memory space for better performance
- PCM with low leakage power and high density is a promising candidate to replace DRAM
- Write endurance and latency are critical for using PCM
- Exiting studies improve the write mechanism to handle given write patterns on PCM
- Why don't we improve fundamental data structures directly so as to generate more suitable write patterns for PCM

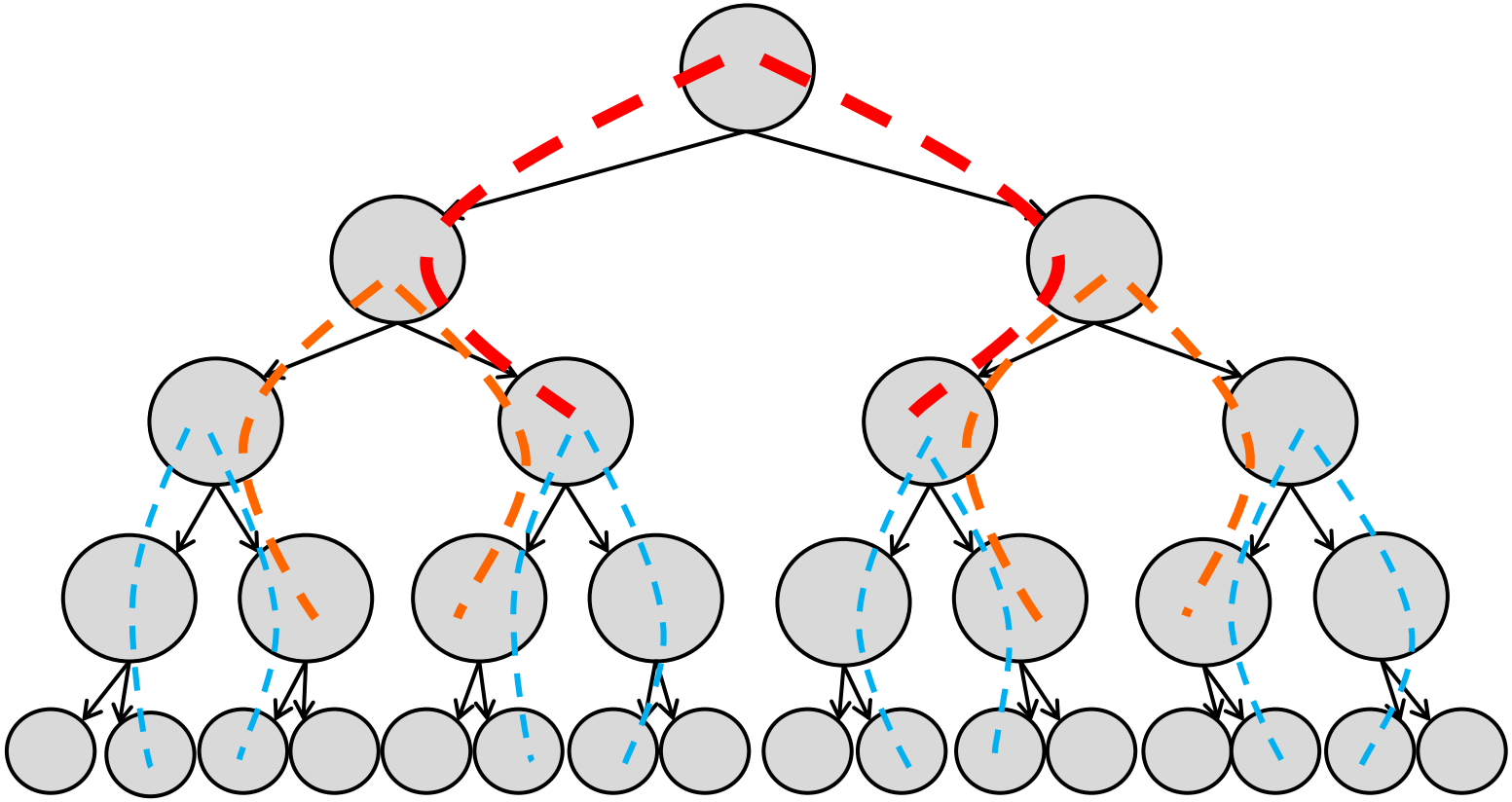
Four Types of AVL Tree Rotations



Relation among Nodes in an RR Rotation

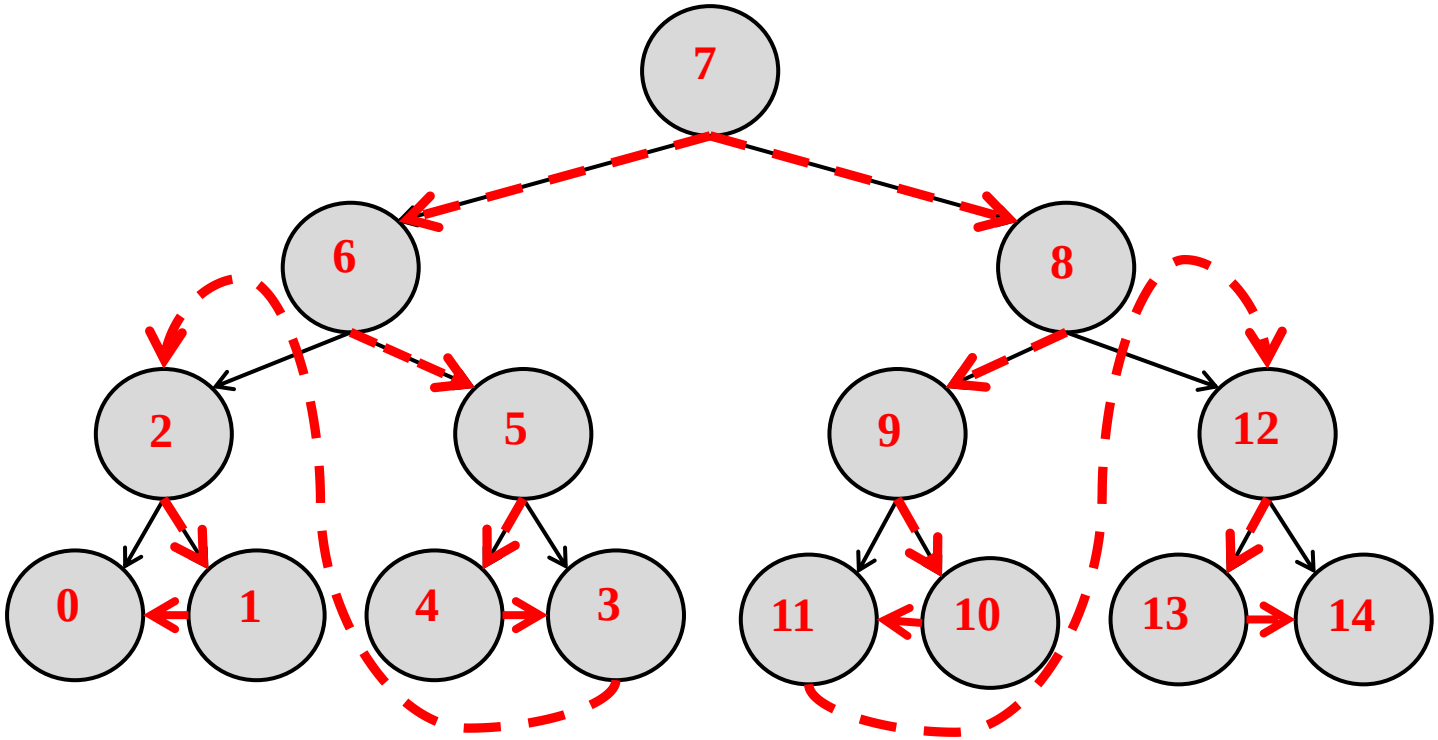


Relation Binding of Tree Nodes



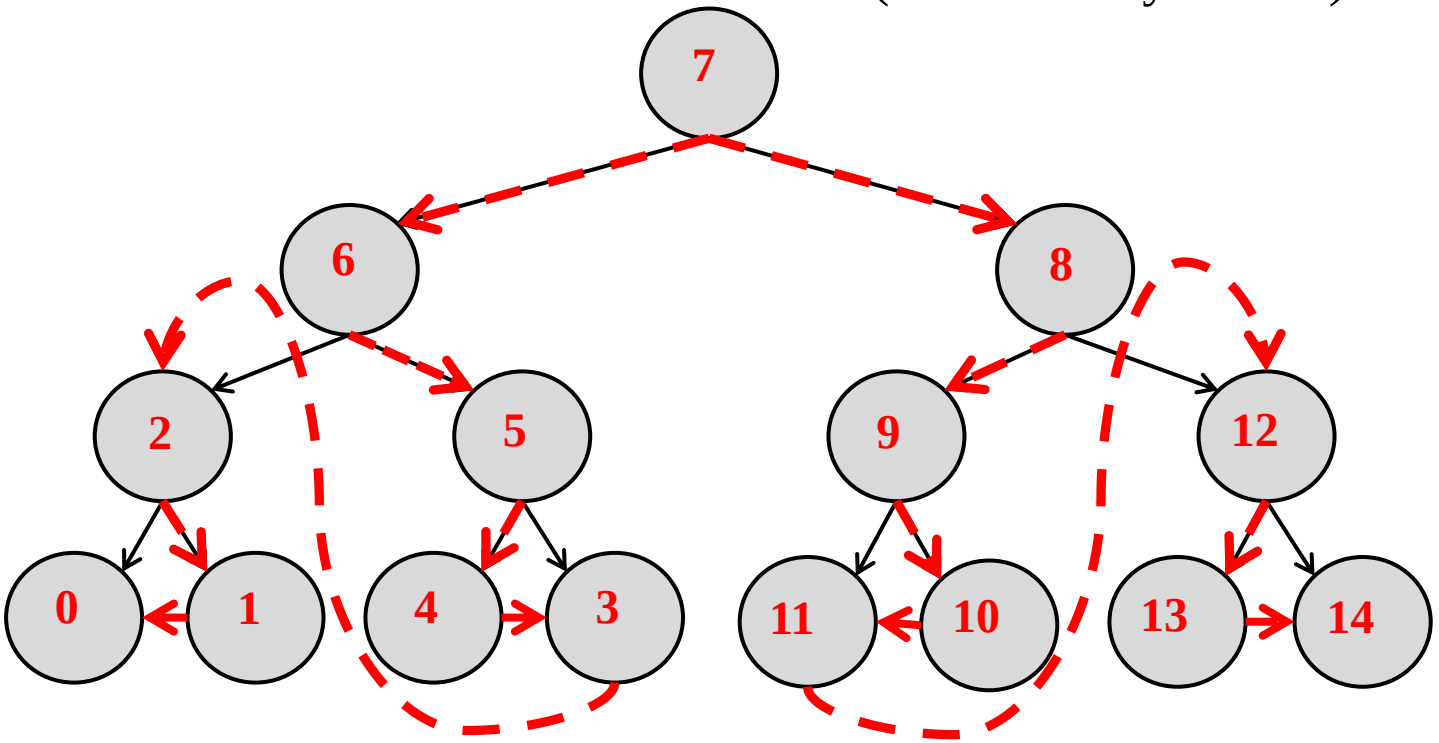
Depth-First-Alternating Traversal (DFAT)

- A systematic approach for indexing all nodes, where nodes having stronger relations will be assigned closer indexes



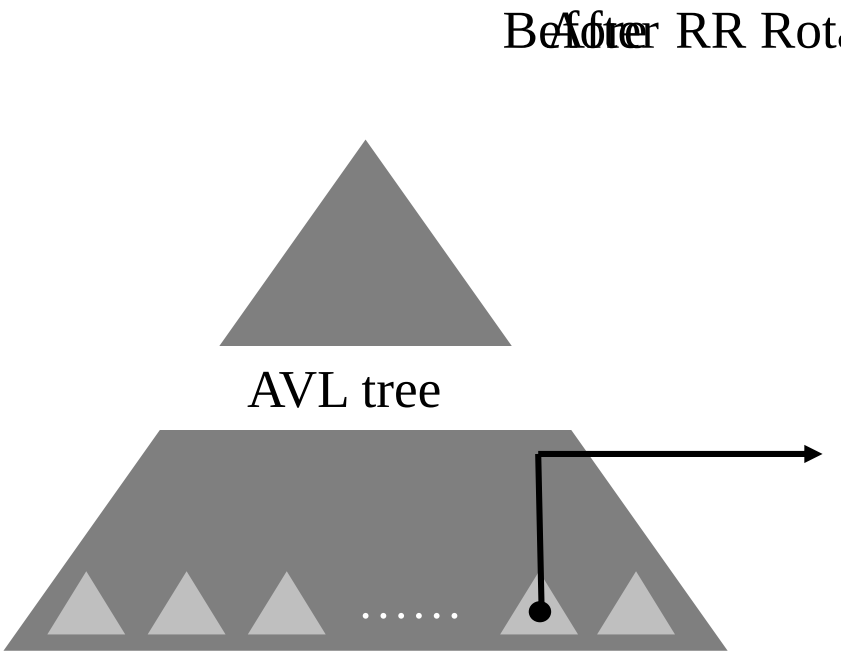
Leveraging Gray Code on DFAT

- Gray code: An ordering of the binary numeral system such that two successive values have the shortest distance (differ in only one bit)



Index	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14
Gray Code	0000	0001	0011	0010	0110	0111	0101	0100	1100	1101	1111	1110	1010	1011	1001

An Example of Running DFAT with Gray Code



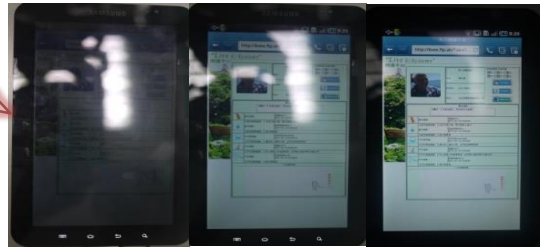
Key value	Binary code address	Gray code address
61	0111111111111101	0100000000000011
70	0111111111111110	0100000000000001
50	*0111111111111111*	*0100000000000000*
52	*1000000000000000*	*1100000000000000*
60	1000000000000001	1100000000000001

Energy Saving Designs

Two Approaches to Reduce the Power Consumption of Devices

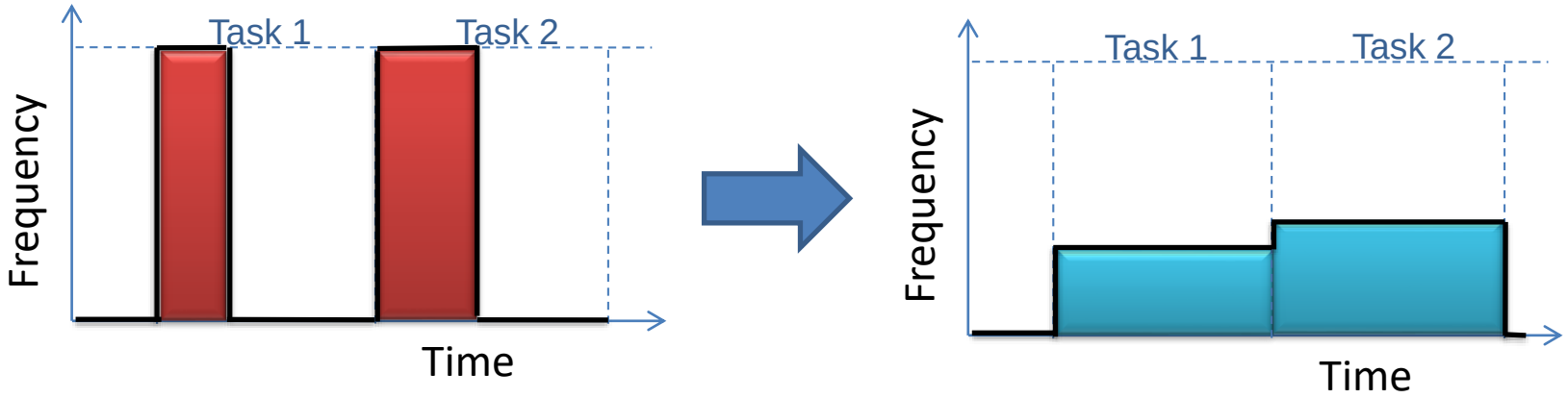
- Dynamic Voltage and Frequency Scaling (DVFS)
  - Scale down the voltage and/or frequency to reduce the processor power consumption
  - An example: reducing 17% of the energy consumption for H.264 decoding over TI DaVince DM6446
- Dynamic Power Management (DPM)
  - Change to an energy-efficient state to reduce the power consumption of peripheral devices
  - An example: reducing 15% of the energy consumption for the browser over Android Galaxy Tab

Power Consumption	1625mW	2023mW	3166mW

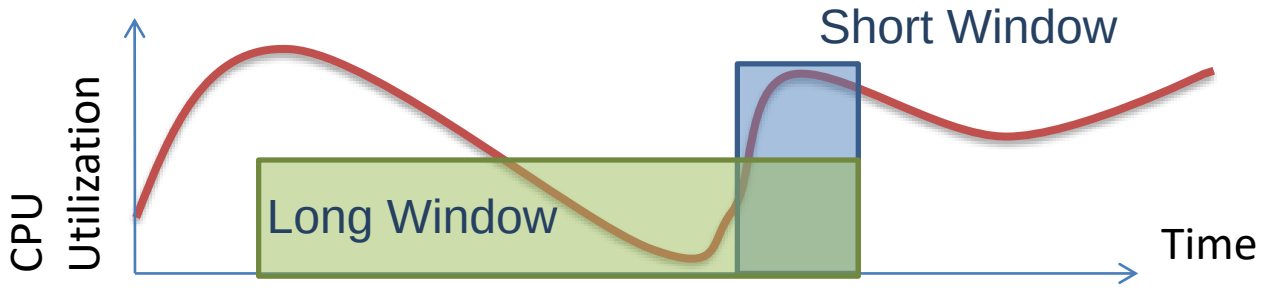


Dynamic Frequency and Voltage Scaling Designs

- The Observation: It is energy-efficient to scale down the processor frequency dynamically to fit current workloads



- The Approach: It keeps a window or buffer to estimate the workload



Energy-Efficient Multimedia Platforms

- Energy-Efficient ARM-DSP Platforms for H.264 Decoding
  - Analyze the workload variance of multimedia jobs
  - Take real platforms for an ARM+DSP design
  - Achieve more than 17% energy saving for the ITRI PAC SoC

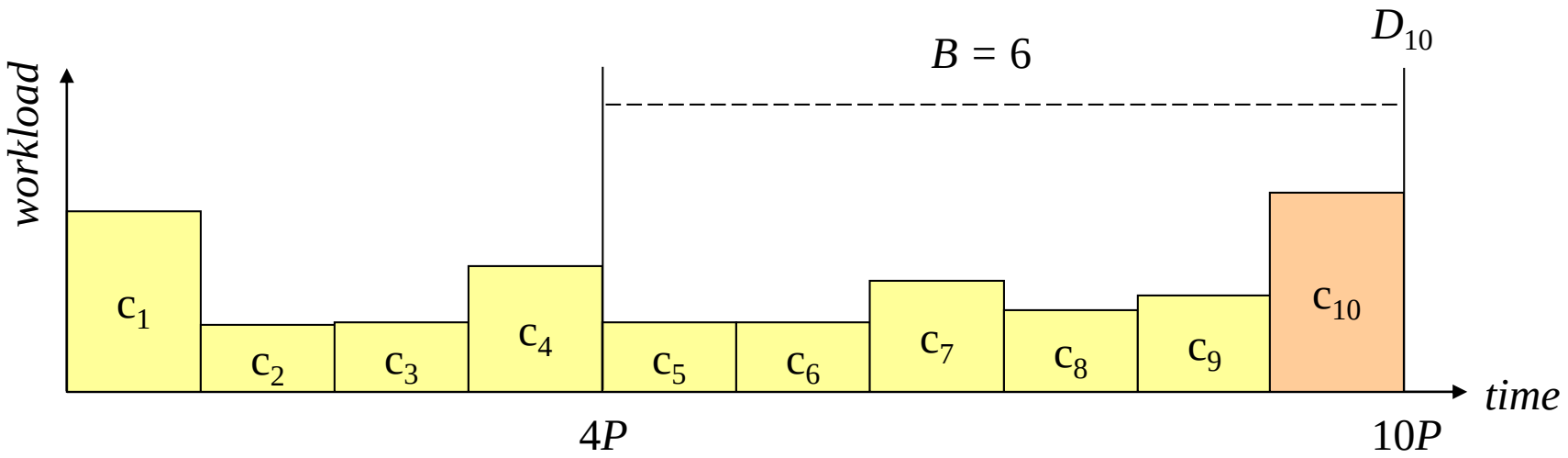
Implementation on a PC with Phenom II

Implementation on an Android Phone

Implementation on the PAC Soc

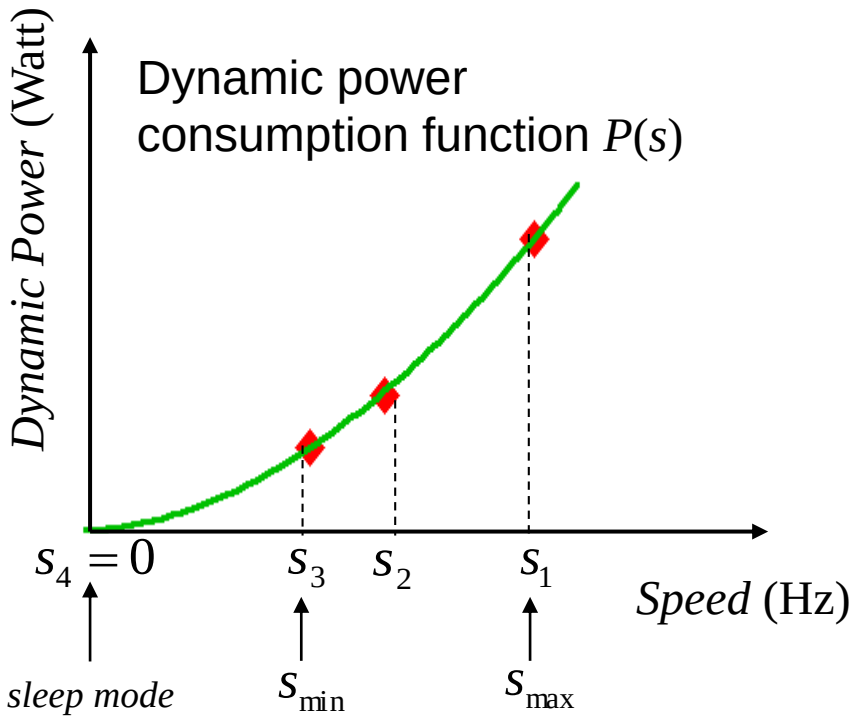
Examples of Energy Saving Designs

Task Model of Multimedia Applications



- A task consists of a sequence of jobs
- Jobs arrive by a period: P time units
- The workload of future job can be predicted based on the log
- Because of the buffer size limitation, the number of pending jobs at any time point is bounded by a fixed integer B

Processor Power Model



$$P(s) = C_{ef} V_{dd}^{\ 2} s,$$

$$s = k \frac{(V_{dd} - V_t)^2}{V_{dd}}$$

where

C_{ef} is the effective switch capacitance

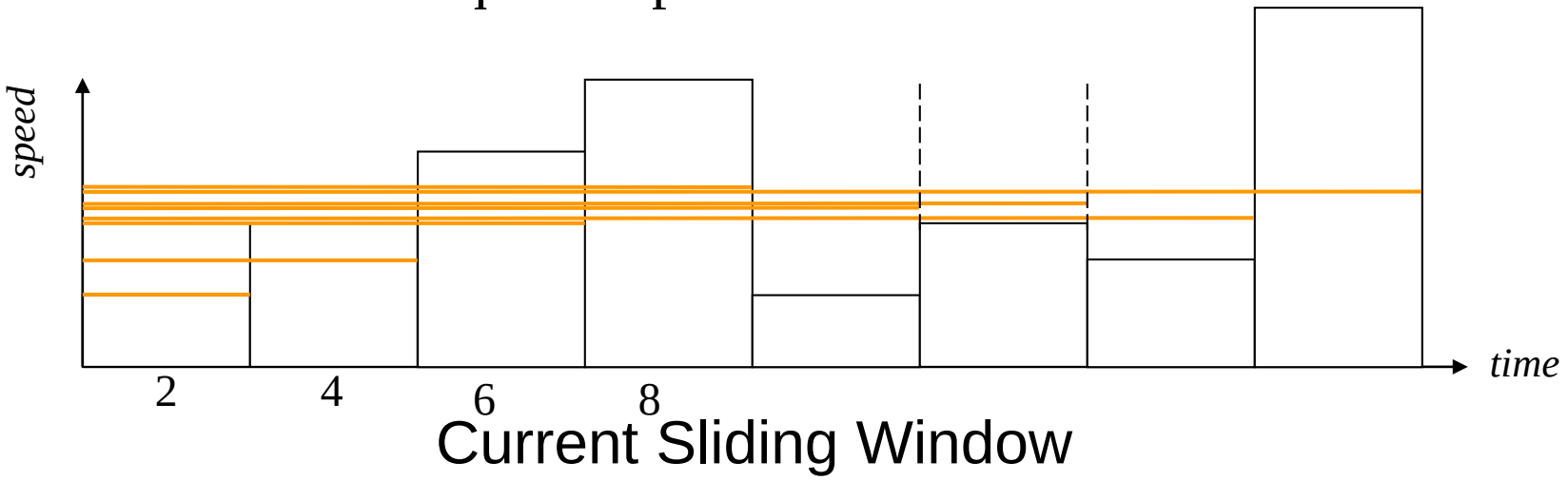
V_t is the threshold voltage

V_{dd} is the supply voltage

k is a hardware-design-specific constant

Concepts of DVFS Algorithm

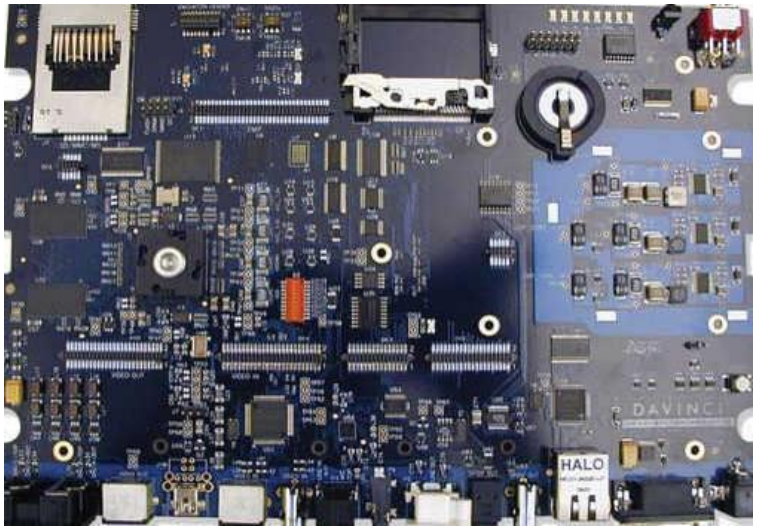
- An Energy-Efficient Multimedia Application
  - ➔Make the frequency as low as possible
  - ➔Meet all performance constraints
  - ➔Find the critical (most demanding) interval by calculating the maximum required speed in all intervals



Hardware Platform

- Platform: Texas Instruments DaVinci DVEVM (The Digital Video Evaluation Module)

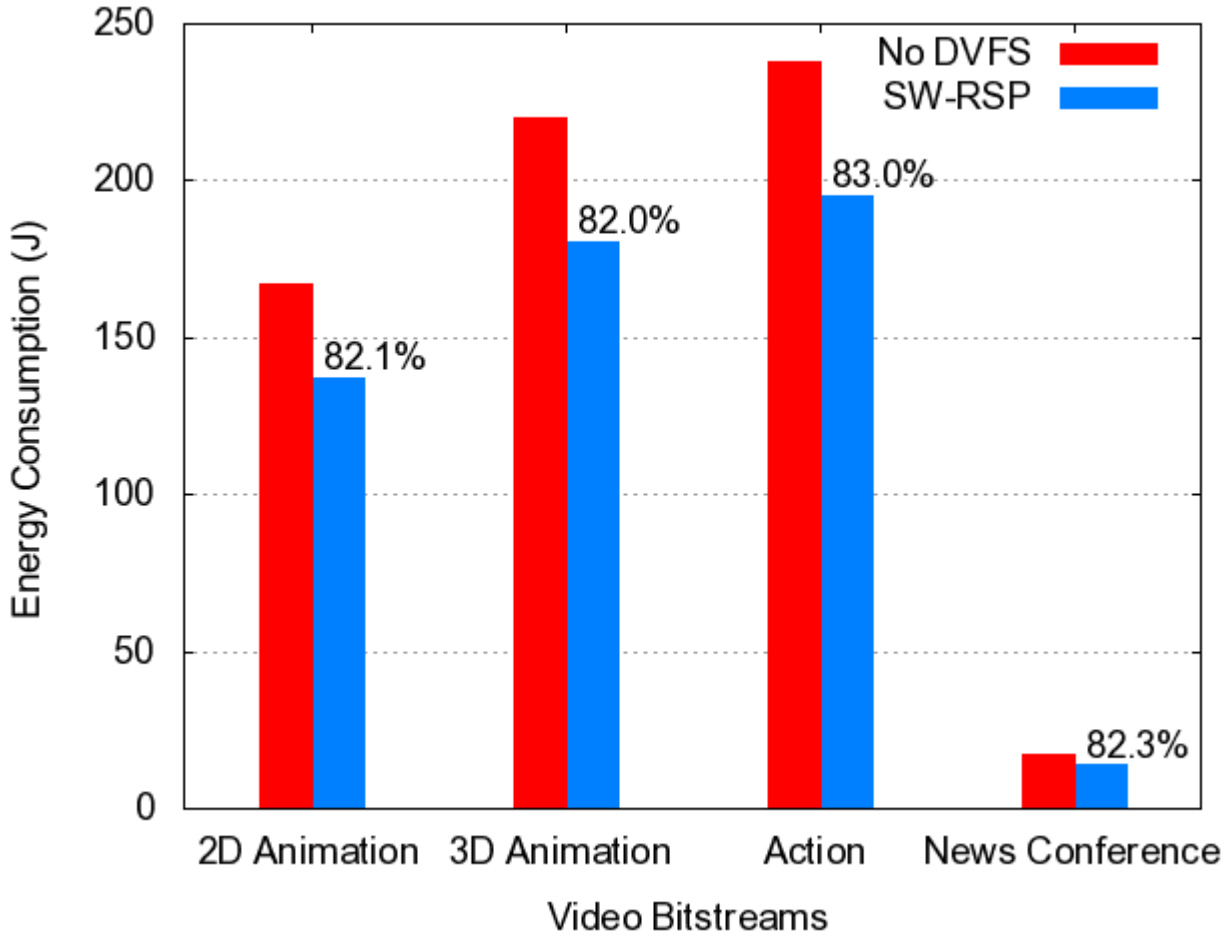
Speed Mode	s_1	s_2	s_3	s_4	s_5	s_6	s_7	s_8	s_9
Frequency (MHz)	594	567	540	513	486	459	432	405	0



Input Video Streams

	2D Animation	3D Animation	Action	News Conf.
				
Number of frames	17985	19182	19182	1751
FPS	29.97	29.97	23.98	23.98
Bit-rate (kbps)	754.99	1237.58	1798.87	631.27
PSNR	42.44	41.85	40.04	42.49
Resolution	720×480	720×480	720×480	720×480

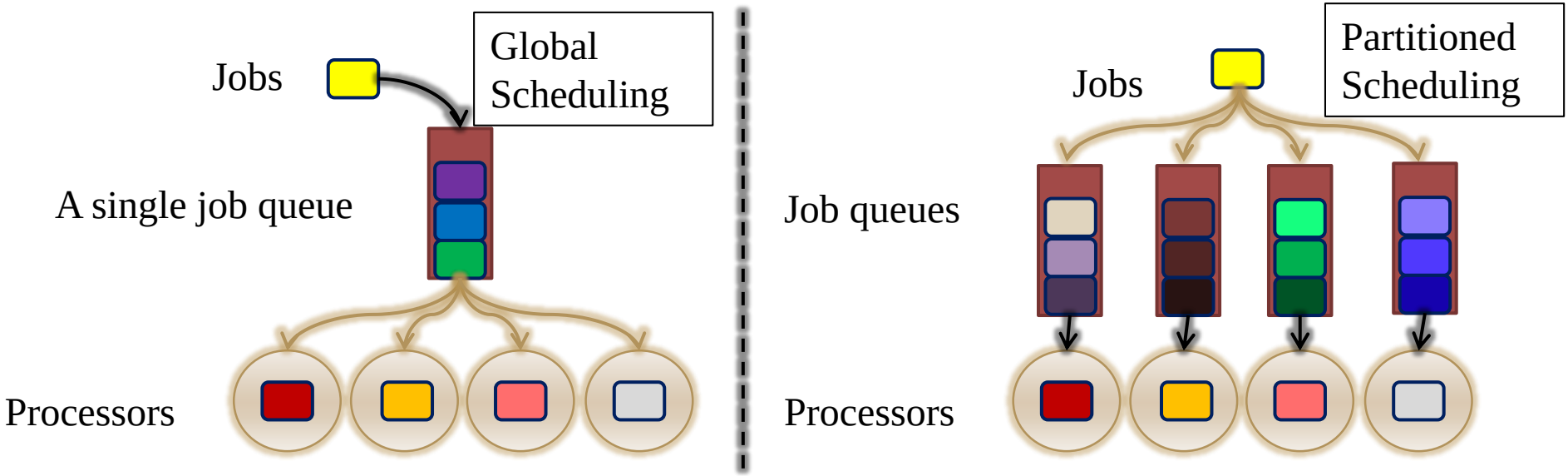
Experimental Results



Multi-Core Scheduling

Scheduling on Multiple Cores (1/2)

- Real-Time Multi-Core Scheduling Algorithms [1]
  - Global scheduling algorithms [2]
  - Partitioned scheduling algorithms [3]



[1] Robert I. Davis and Alan Burns, "A Survey of Hard Real-Time Scheduling for Multiprocessor Systems," in *ACM Computing Surveys*, 2011.
[2] Hennadiy Leontyev and James H. Anderson, "Generalized Tardiness Bounds for Global Multiprocessor Scheduling," in *RTSS*, 2007.
[3] Sanjoy Baruah and Nathan Fisher, "The Partitioned Multiprocessor Scheduling of Sporadic Task Systems," in *RTSS*, 2005.

Scheduling on Multiple Cores (2/2)

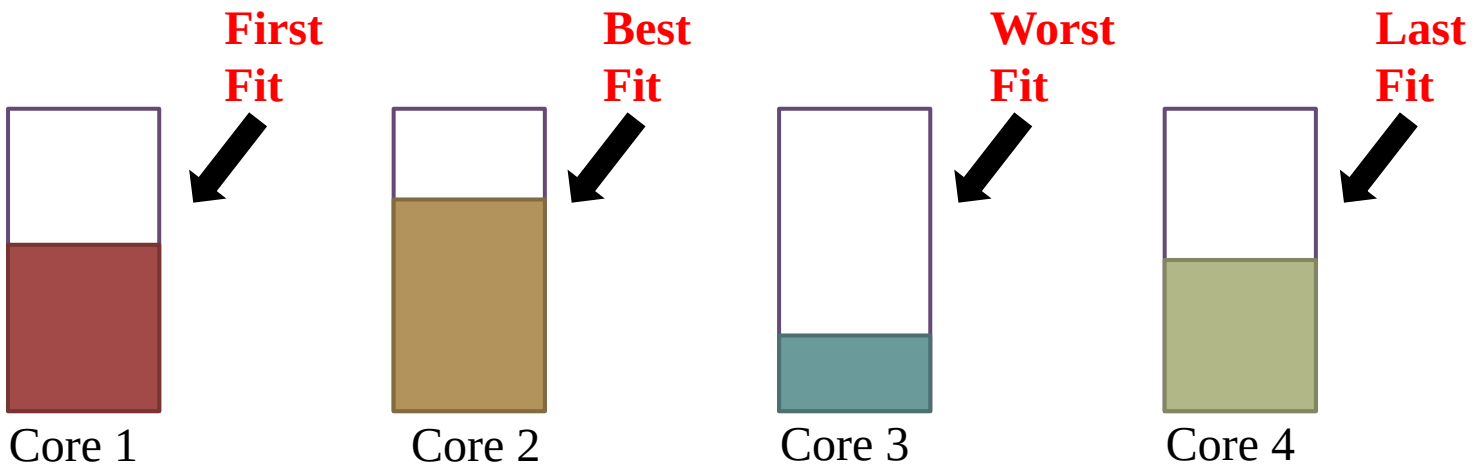
- Settings:
  - 2 cores
  - 2 tasks with execution time 10 arrive at time 0
  - 1 task with execution time 19 arrives at time 1

- Global Scheduling
  - Core 1: 
  - Core 2: 
- Partitioned Scheduling
  - Core 1: 
  - Core 2: 

Different Fitting Approaches for Partitioned Scheduling

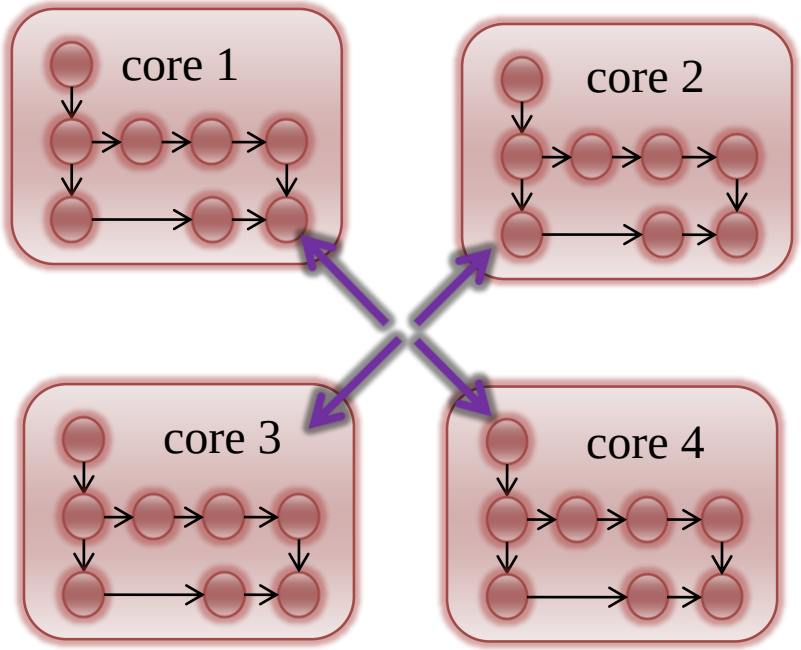
- First-Fit: choose the one with the smallest index
- Last-Fit: choose the one with the largest index
- Best-Fit: choose the one with the maximal utilization
- Worst-Fit: choose the one with the minimal utilization

Assigning a task with utilization equal to 0.1



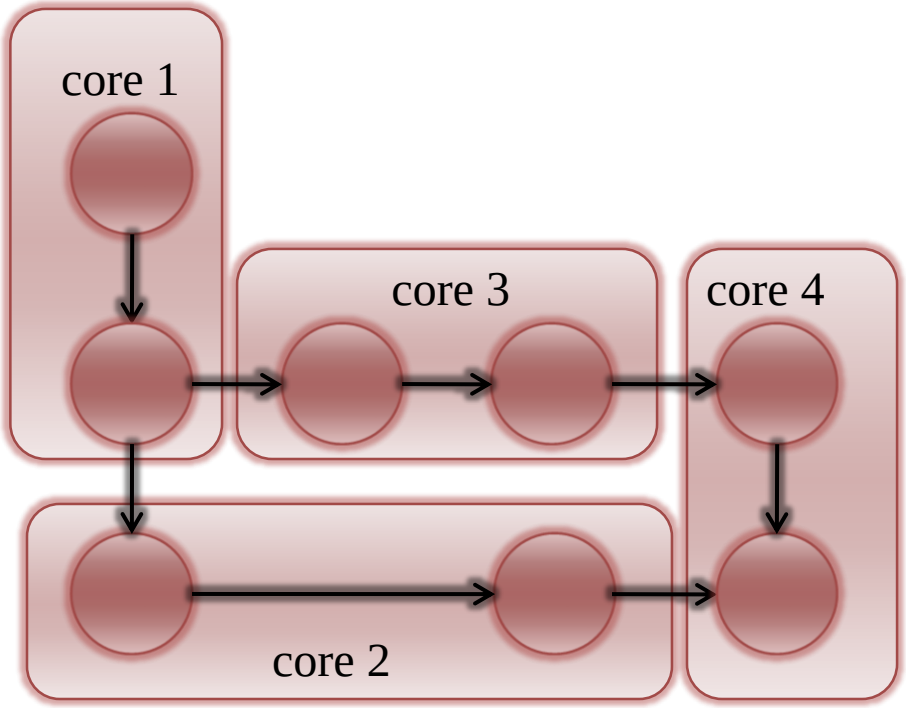
Two Approaches for Using Multiple Cores

- Data Partition



- Flexibility
- Scalability
- Load-balance

- Functional Partition



- Hardware/software co-design
- Parallelism in sequential program