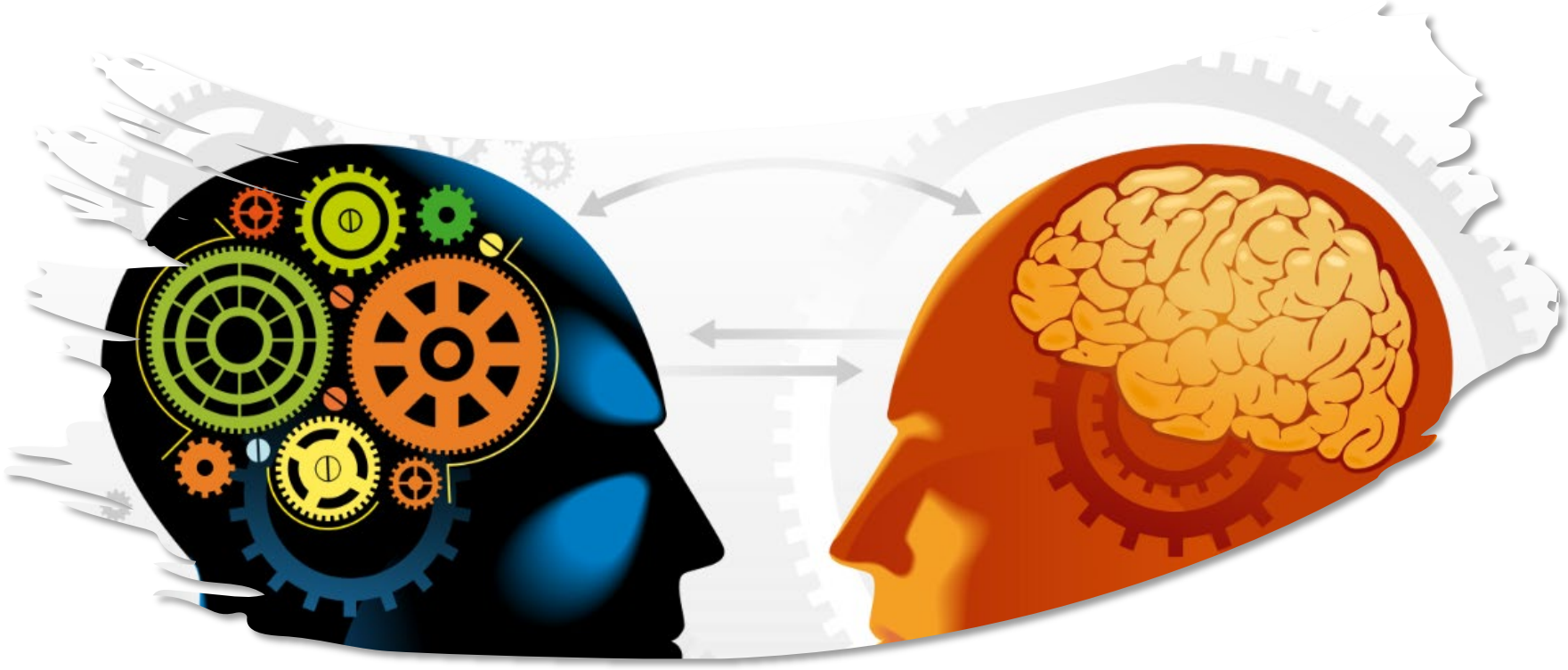


Building the Infinite Brain

COMP 590/790

Raghavendra Pradyumna Pothukuchi

THE UNIVERSITY
of NORTH CAROLINA
at CHAPEL HILL

✉ raghav@cs.unc.edu

Quick Review

Review B3: Ch. 2 and App B.

What is caching?

A technique to minimize the impact of long memory latencies

What are the basic cache parameters?

Associativity (placement), block size, capacity, write through/back, write-allocate or no

What are the types of cache misses?

Compulsory, capacity, conflict

What are some ways in which cache performance can be improved?

Non-blocking, banking, software or hardware prefetching etc.

How to choose the right memory hierarchy design?

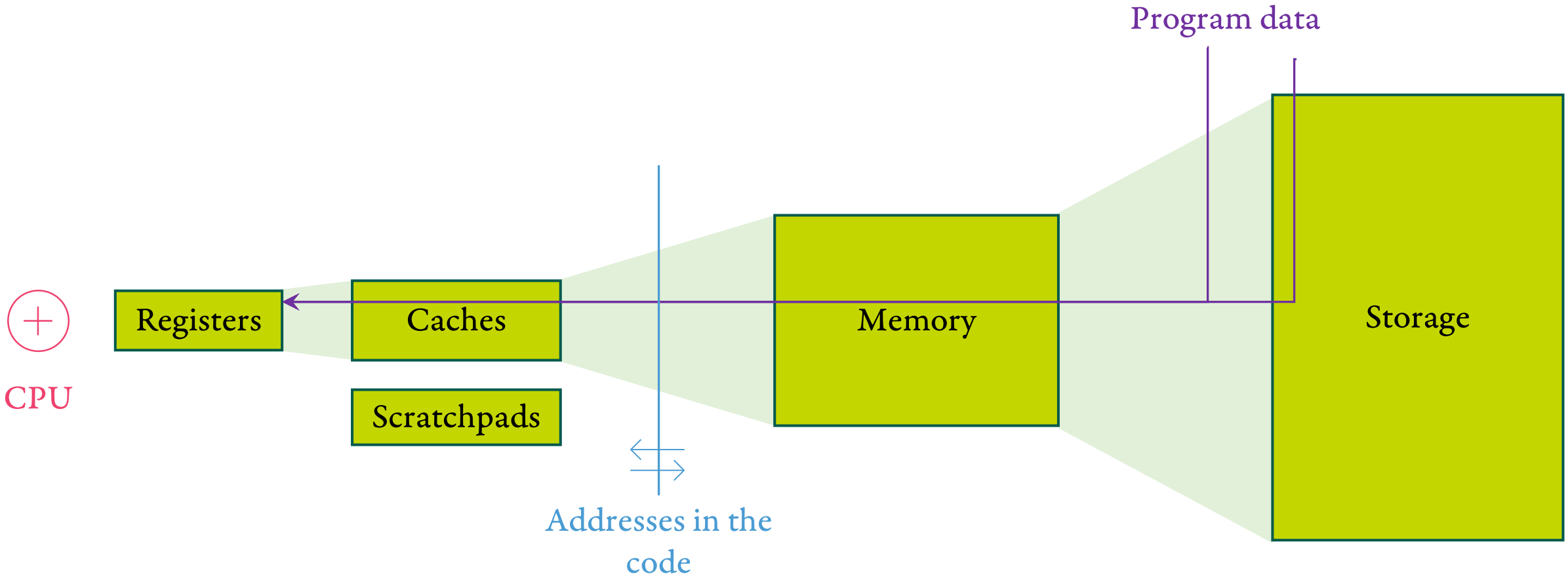
Tailor it to the access patterns and layout: general purpose to domain specific

Understand the relevant "systems" problems and identify solutions

For Today

- Quick review
- **Memory**

Memory Hierarchy



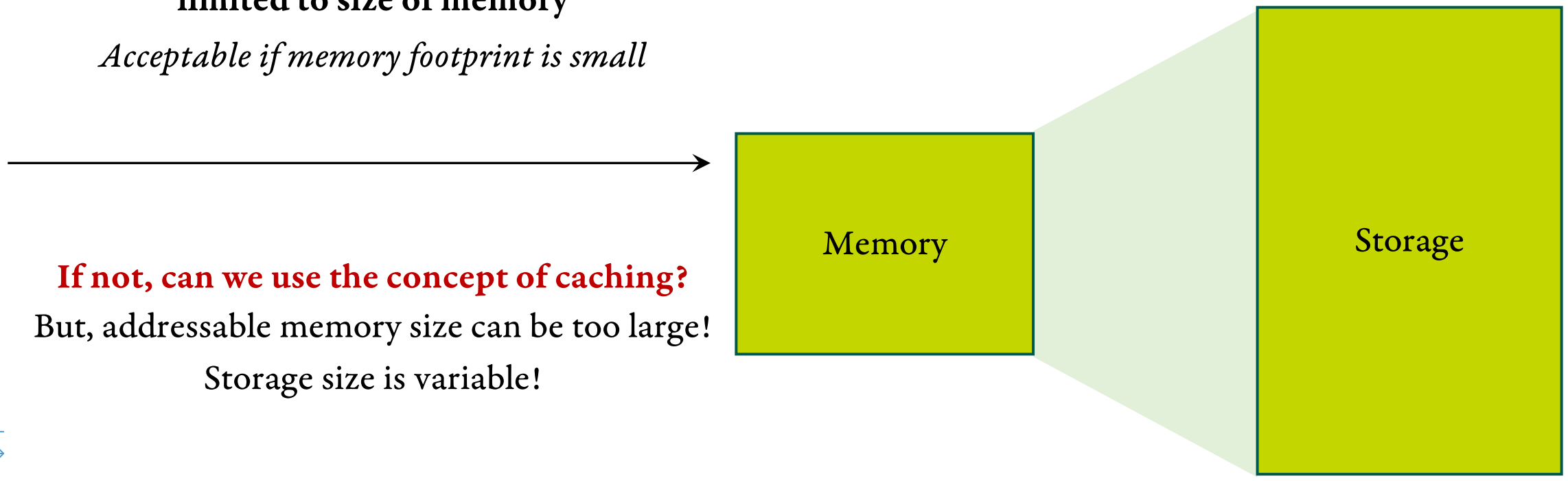
What are these addresses?

Refer to data locations, but where?

Memory Design Goals: Capacity

Capacity: Program data (and code) limited to size of memory

Acceptable if memory footprint is small



If not, can we use the concept of caching?

But, addressable memory size can be too large!

Storage size is variable!

Addresses in the code

M-bit address

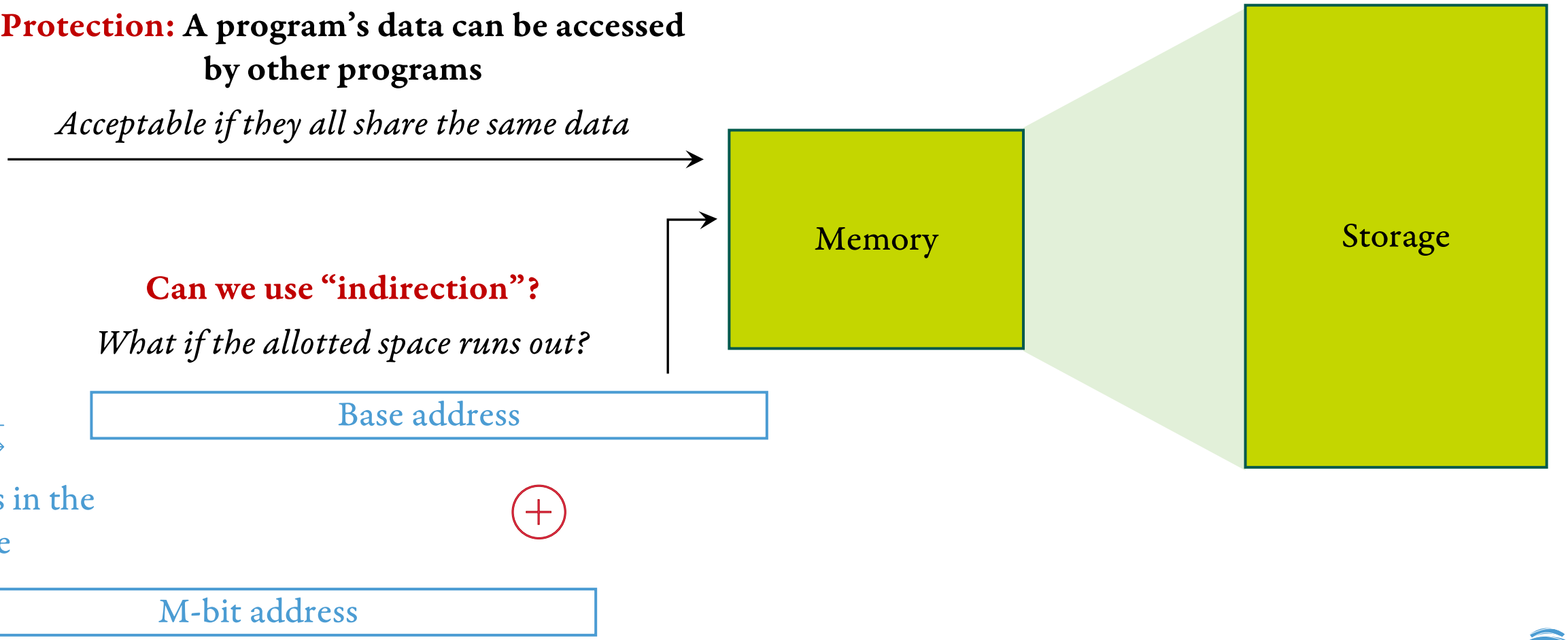
Change Memory-Storage mapping in each machine?

Memory Design Goals: Abstraction and Control

Abstraction: Addresses of a program need to consider other programs and memory layout

Protection: A program's data can be accessed by other programs

Acceptable if they all share the same data



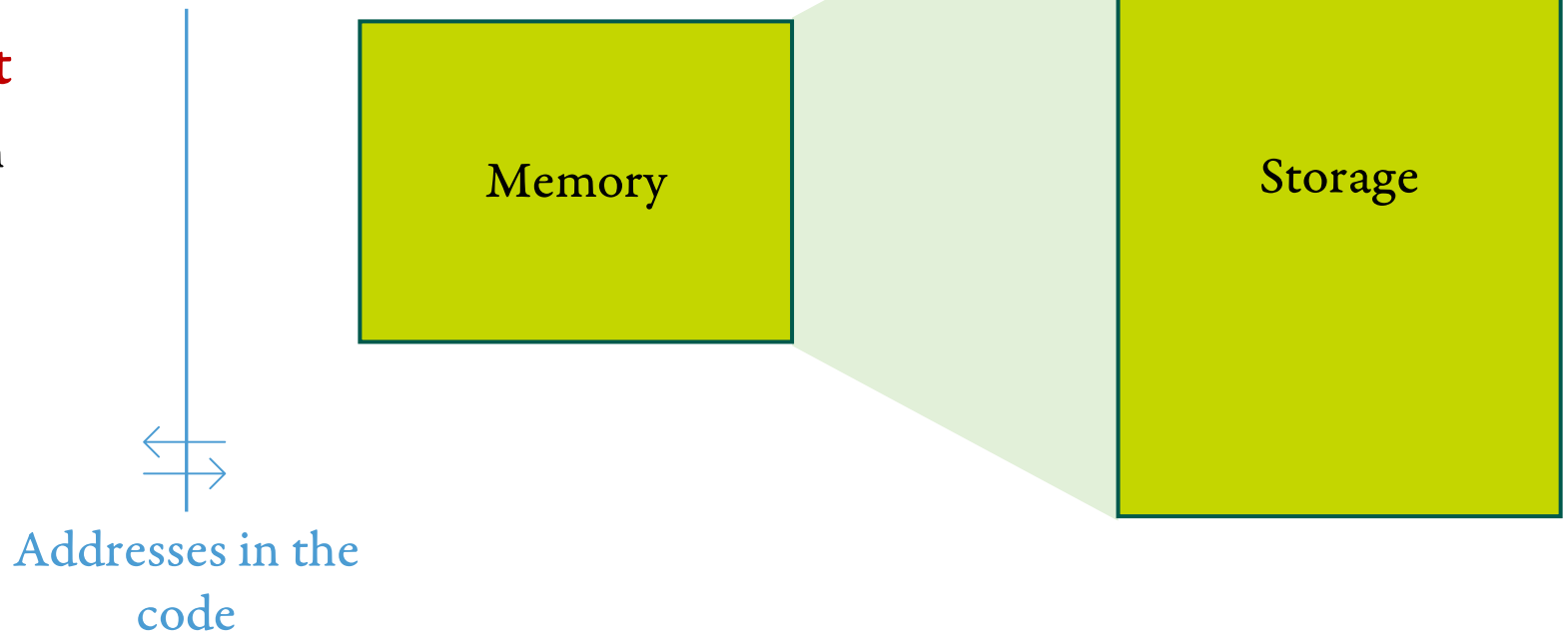
Memory Management

> Dynamic Storage Allocation in the Atlas Computer, Including an Automatic Use of a Backing Store
> https://dl.acm.org/doi/10.1145/366786.366800

John Fotheringham, *Pioneering computer architect (Atlas)*

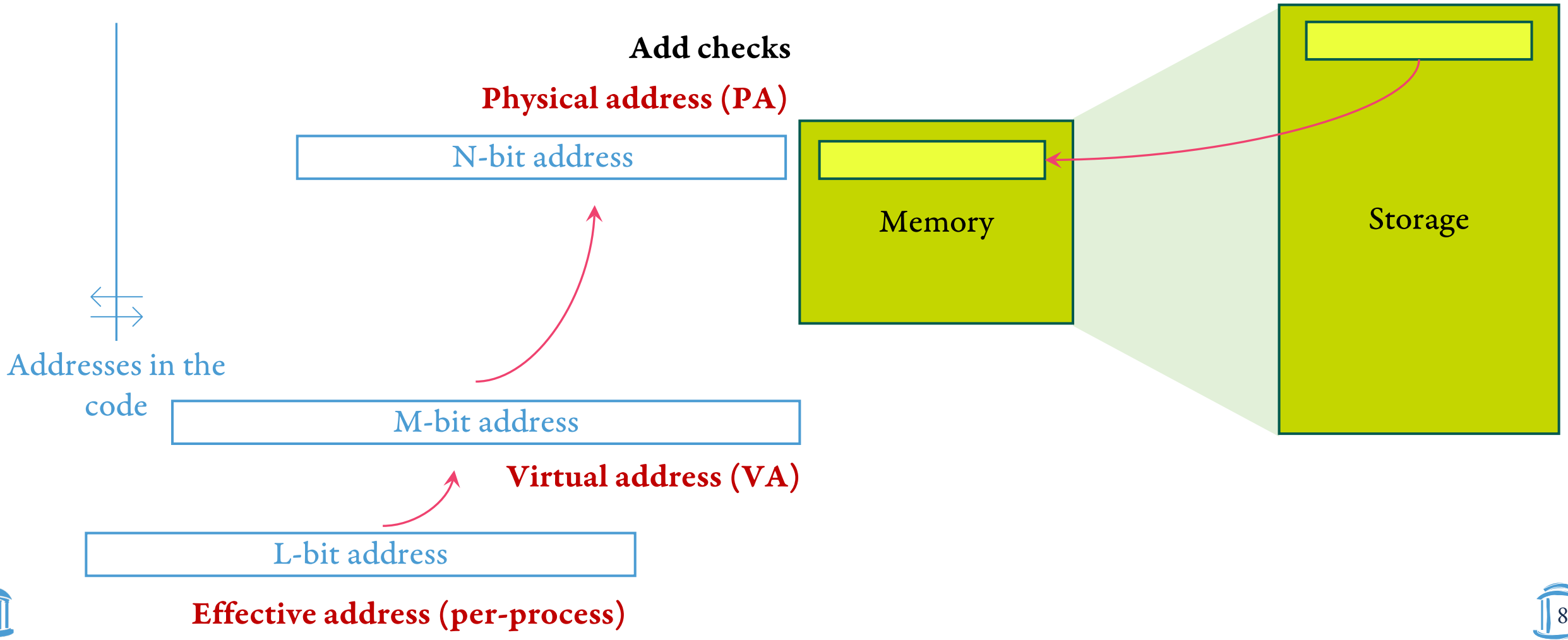
Redefine what an address meant

Reference to information, not a location



Virtual Memory

Large capacity, single program abstraction, and protection



Virtual Memory

Mechanisms

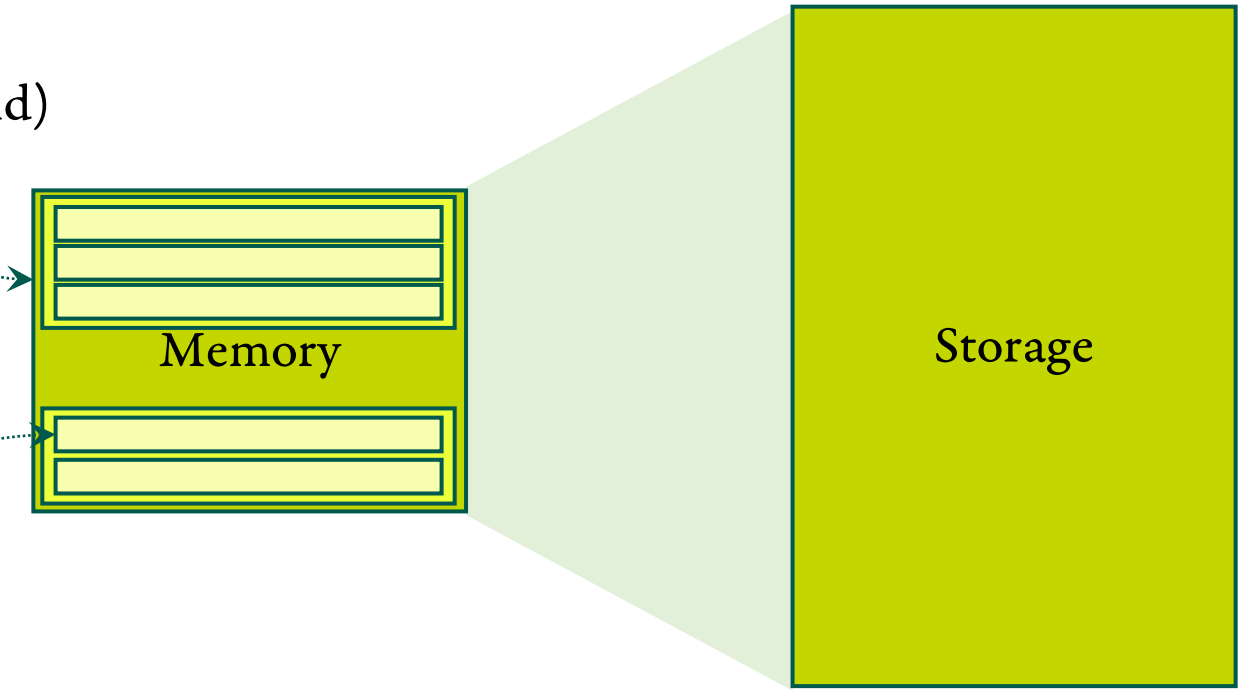
Segments

Variable size contiguous address space blocks (base, bound)

Semantic meaning: code, data etc.

Pages

Fixed size address space blocks



Policies

Placement, Replacement, Write, and Write miss…

Address Translation

Fully associative placement

M-bit address

Virtual Page Number

Page Frame Number

N-bit address

Per-process!

Page table: map for address translation

If small, associative search

Large size, use a map

Page Table

M-bit address

VPN

Virtual Page Number: M-P bits
Offset: P bits

N-bit address

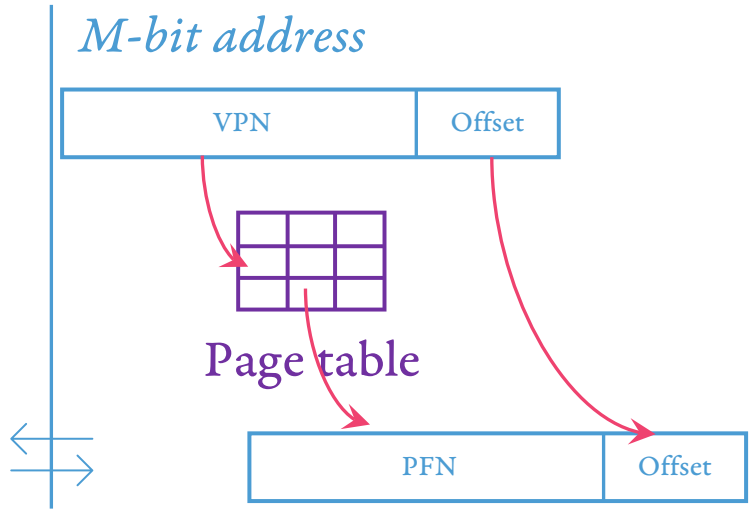
PFN

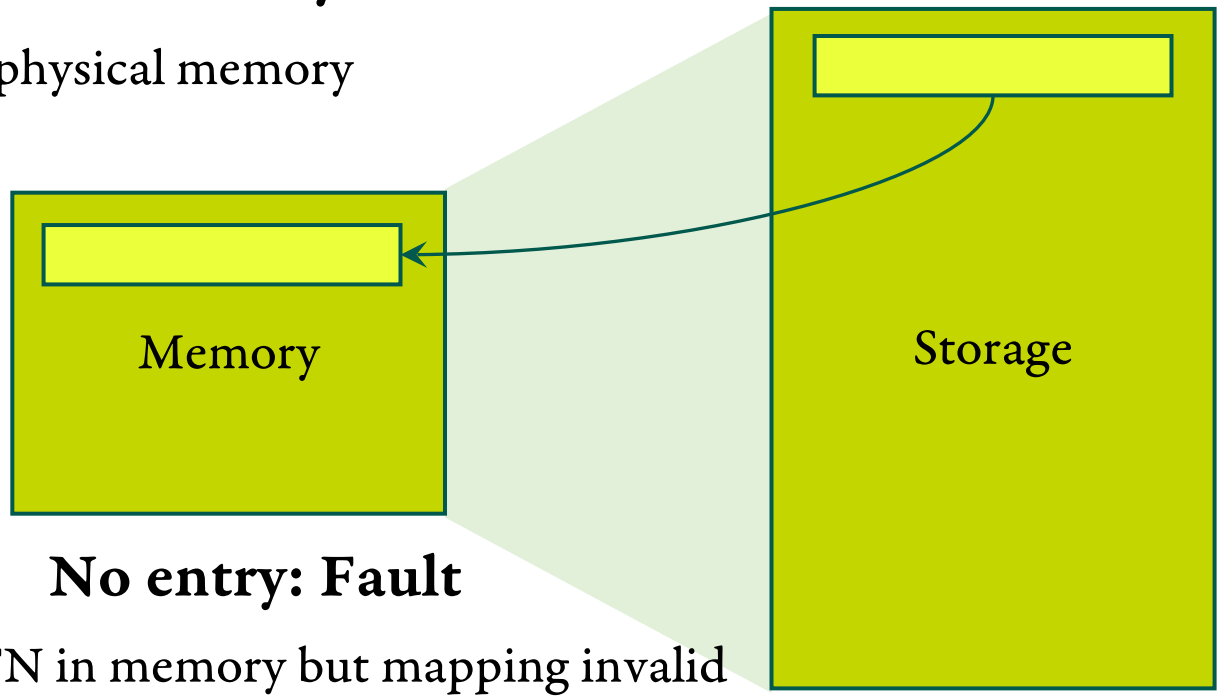
Page Frame Number: N-P bits
Offset: P bits

Access (read, write), and status bits (valid, dirty)

Do we have entries for every VPN?

Track only physical memory





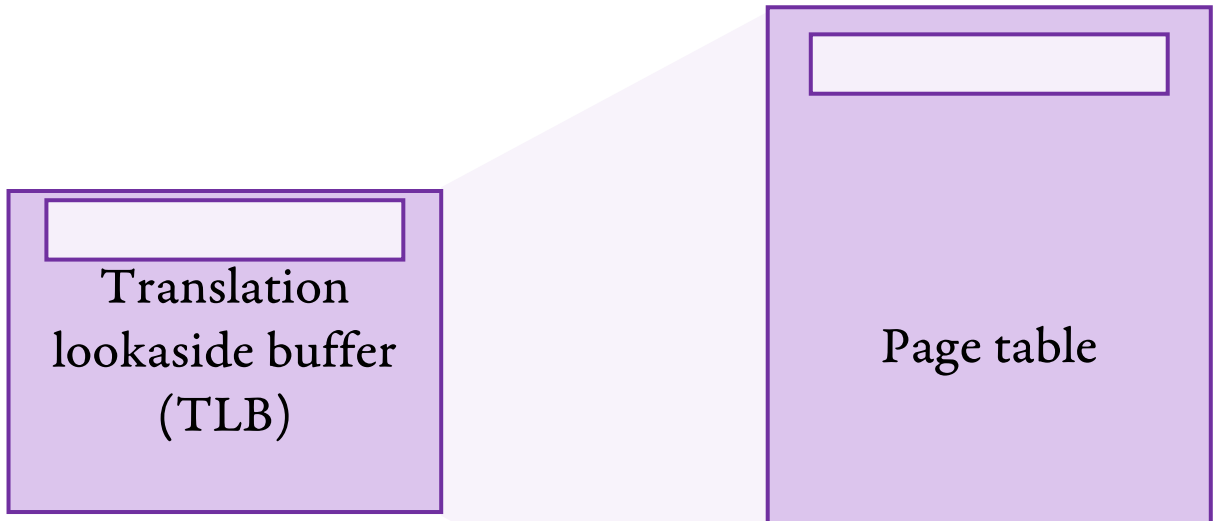
No entry: Fault

Minor: PFN in memory but mapping invalid

Major: VPN not in page table

A Bottleneck

Memory is accessed every cycle

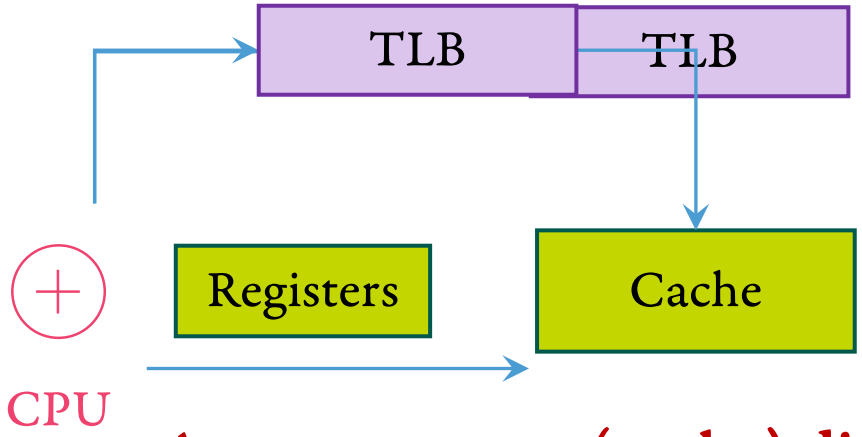


Add a fast path

In hardware, near the processor

In hardware, spread across memory and storage

Execution Flow



Obtain address first?

TLB hit?→Access ok?

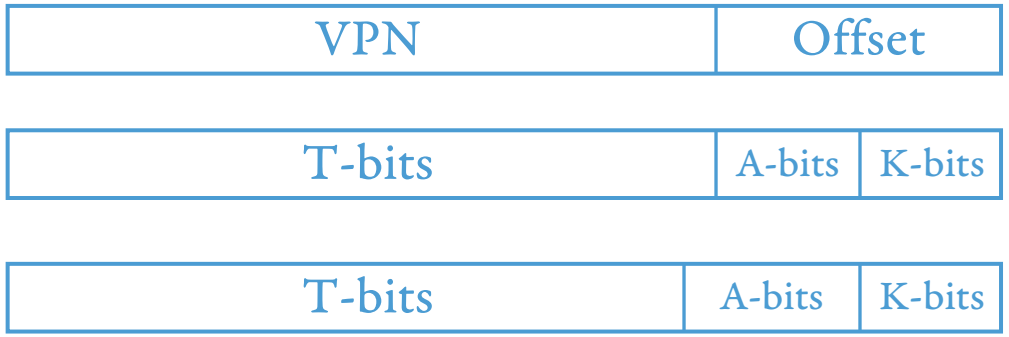
TLB miss?→Hardware page walk→Access ok?

TLB miss?→Hardware page walk →OS kernel→Retry

Latency critical

Access memory (cache) directly?

Virtually addressed caches



Tag lookup and translation align

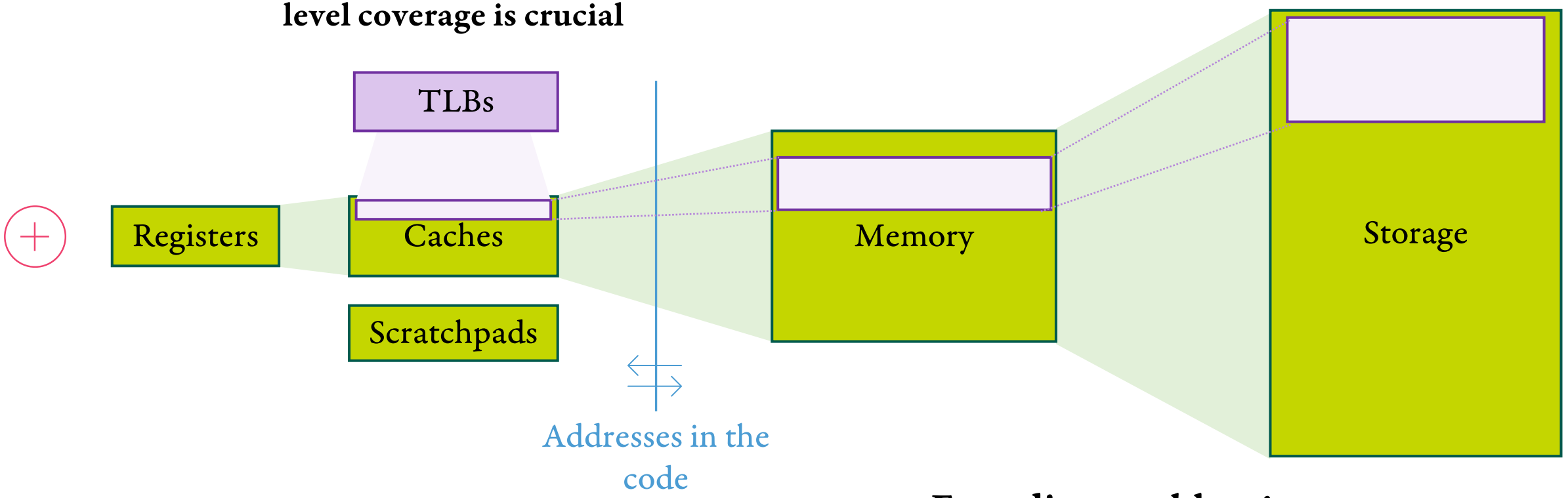
Virtual (physical) index, physical tag

Resolve synonyms (same VA, different PA), homonyms (different VA, same PA)

Virtual index, virtual tag

Memory Hierarchy

Sizing TLB span and corresponding cache level coverage is crucial



Encoding problem!

Many optimizations

Hierarchical, multi-level, hashed page tables etc.

Compressed TLBs, large pages etc.

Takeaways

Review B3: Ch. 2

What are some goals in designing the memory subsystem?

Capacity, program abstraction, protection, and sharing

What is virtual memory?

Abstraction to deliver the goals

What is a bottleneck in virtual memory systems?

Address translation

How can addressing bottlenecks be tackled?

Through fast paths: TLBs

How to choose the right memory hierarchy design?

Tailor it to the access patterns and layout: general purpose to domain specific

Understand the relevant "systems" problems and identify solutions

Image Credits (Educational, Fair Use)

- Title image: VLADGRIN, https://www.istockphoto.com/vector/human_-machine-gm147409511-16840728 (Educational fair use)
- Infinite brain: Science wonder stories, May 1930, Illustrator: Frank R Paul, Editor: Hugo Gernsback
- Brain color, ICs, cloud server, black rat: No attribution required (Hiclipart)
- Hand with spoon: public domain freepng
- Signals: https://www.nature.com/articles/nrn3724
- Thought clouds: F. Willett et al./*Nature* 2021/Erika Woodrum, https://med.stanford.edu/neurosurgery/news/2022/bci-award. https://www.the-scientist.com/news-opinion/brain-computer-interface-user-types-90-characters-per-minute-with-mind-68762
- Picture of scientists: https://www.cs.auckland.ac.nz/~brian/rutherford8.html (original: Pierre de Latil), Bush (Carnegie Science), Others (Wikipedia, National Academies, IEEE, and university profile images)
- Flowchart: Pause08 – flaticon.com; Digital brain: Smashicons – flaticon.com; Quantum processor icons created by Paul J. - Flaticon
- Server rack: upklyak – freepik.com
- Arm, Lotus: Adobe stock
- Quantum processor: Rigetti computing
- Images of implanted users: Top: Case Western Reserve University (https://thedaily.case.edu/man-quadriplegia-employs-injury-bridging-technologies-move-just-thinking/), Bottom: Jan Scheuermann (University of Pittsburgh/UPMC; https://www.upmc.com/media/news/bci-press-release-chocolate)
- Images of wearable BCIs: Cognixion, NextMind
- Types of BCIs: "Brain–computer interfaces for communication and rehabilitation,
- Illustrative BCI: Neuralink
- Electrodes: "Electrochemical and electrophysiological considerations for clinical high channel count neural interfaces", Vatsyayan et al.
- Form factors: Neuropace, Medtronic, Bloomberg, "Fully Implanted Brain–Computer Interface in a Locked-In Patient with ALS" by Vansteensel et al., Blackrock Neurotech
- Jose Delgado's video: Online, various sources (CNN, Youtube)
- Video of Kennedy and Ramsey: Online, various sources (Youtube, Neural signals)
- Code snippet inspiration: ECE 252 slides at Duke (Dan Sorin et al.)
- Apple processor pipeline: https://dougallj.github.io/applecpu/firestorm.html