

Microkernel Construction

Mechanisms for
~~Memory Management~~

Privilege transfer / revocation

Microkernel Construction

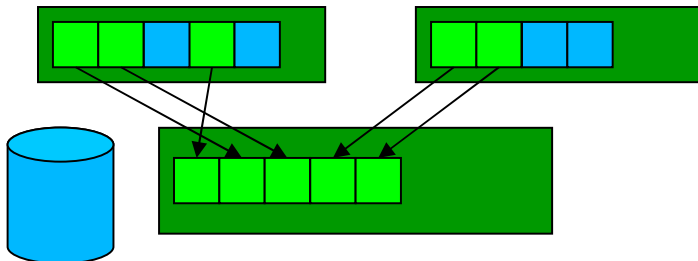
Mechanisms for
~~Memory Management~~

~~Privilege transfer / revocation~~

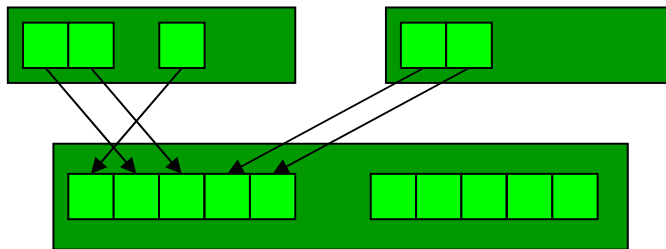
Mapping Database: Taming the Beast

Applications

- **Page Replacement**

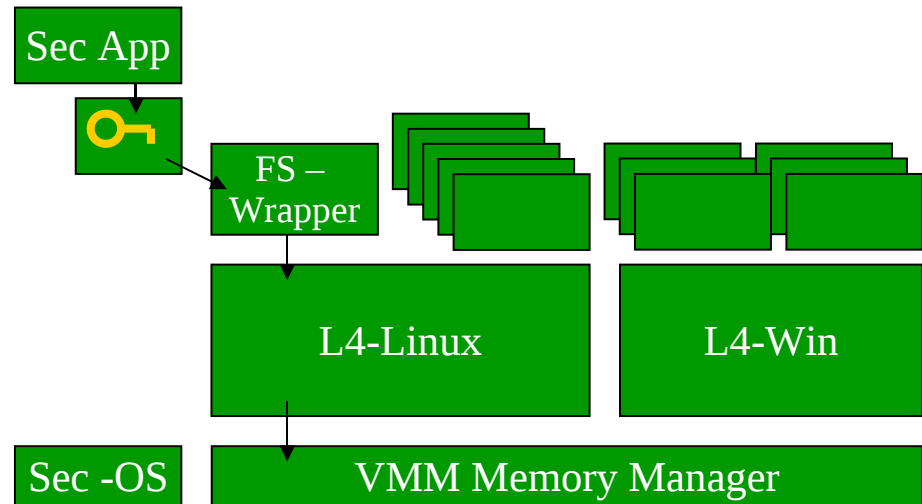


- **Copy on Write / Checkpoint**

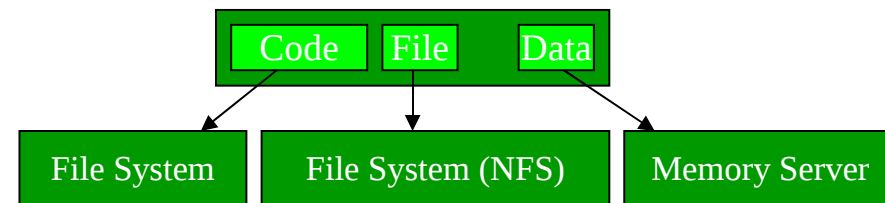


- Permissions (r, w, x)
- Reference Bits (accessed, modified)

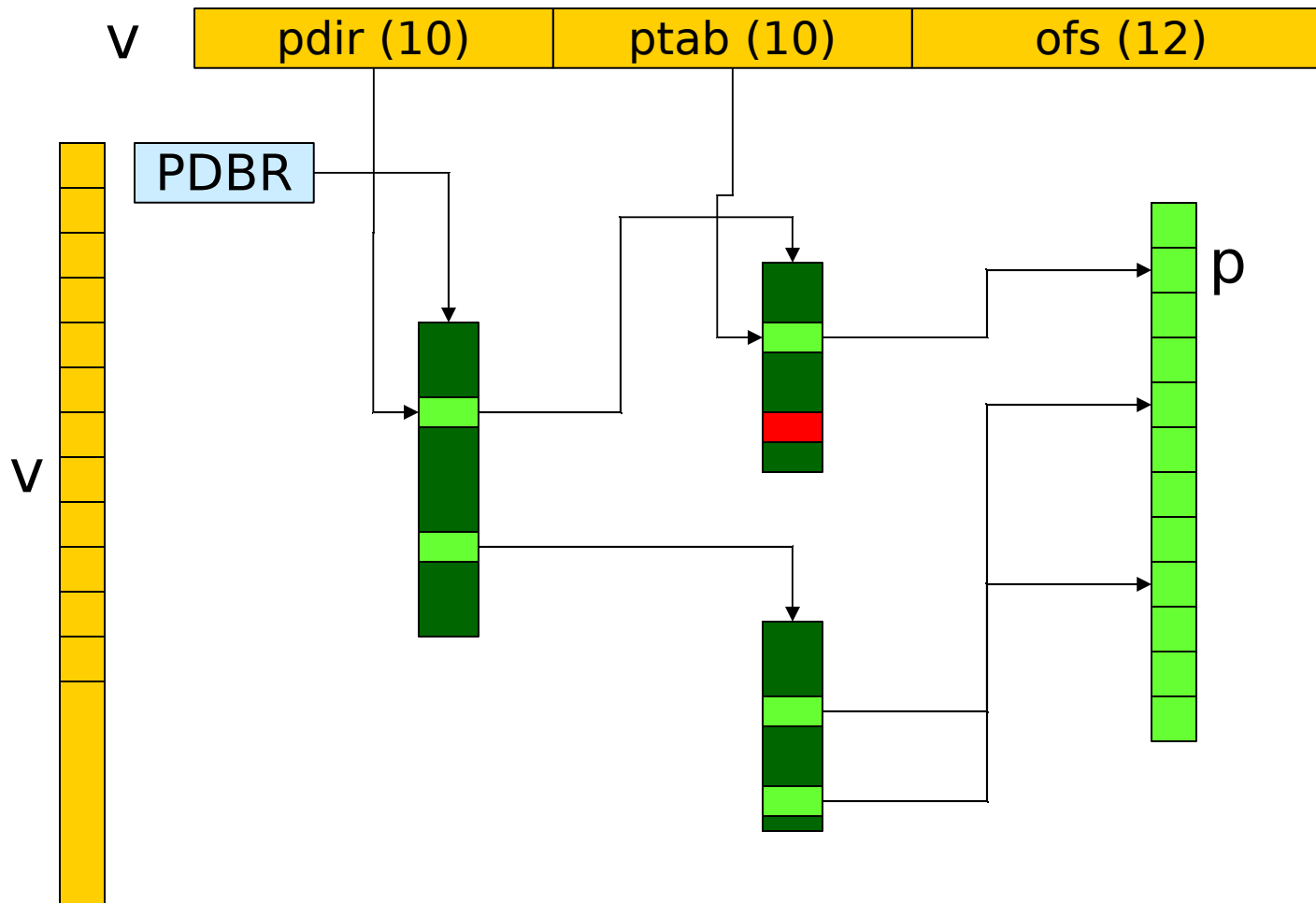
- **Hierarchy**



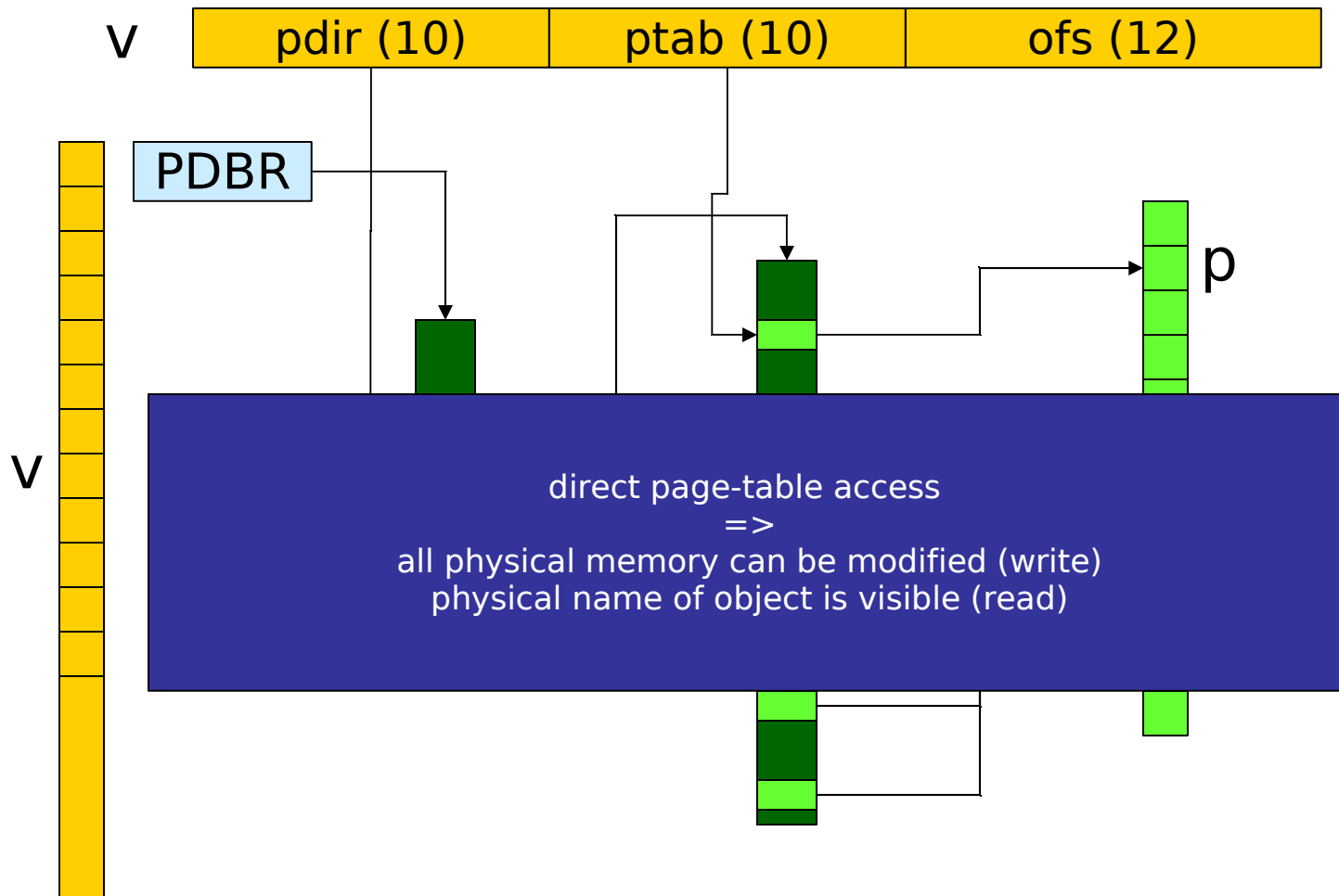
- **Graph**



Address space implementation



Address space implementation



Overview

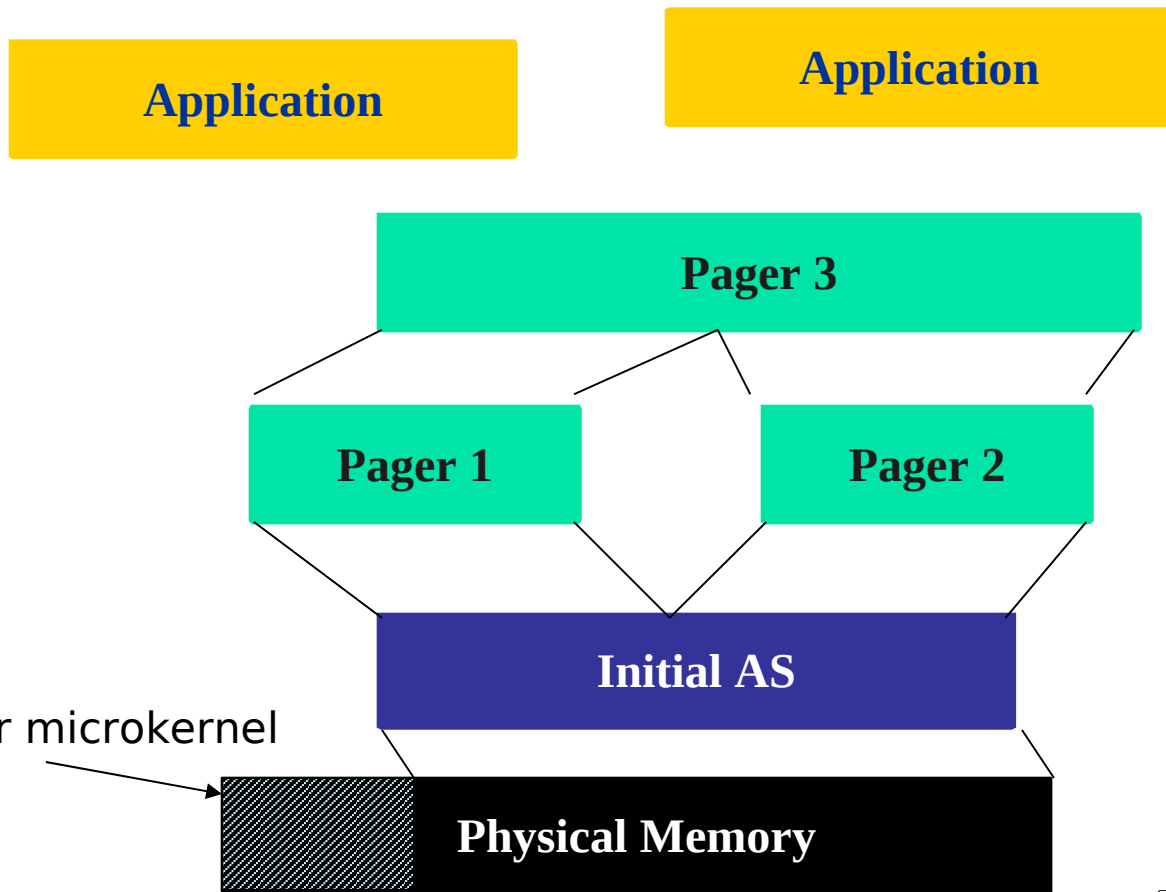
- Introduction
- Address Space Implementation
- Recursive Virtual Address Space Model
 - Map
 - Grant / Copy
 - Unmap
- Implementation
 - Implementation Constraints
 - Data Structures
 - Synchronization

Address space implementation

- Restrict pagetable access (Virtual machines)
 - Paravirtualization:
 - insert page, remove page
 - check that only valid entries are inserted
 - Full Virtualization
 - Shadow pagetables
 - Create copy of Guest Pagetable
 - Validate and activate copy
 - Problem: Changes in guest pagetable
 - VTLB
 - Use Host Page Table as second level TLB
 - On miss (page-fault):
 - lookup guest page table
 - validate entry
 - insert translation to second level TLB
- Abstract from PT model
 - Recursive virtual address space model

Recursive virtual address space model

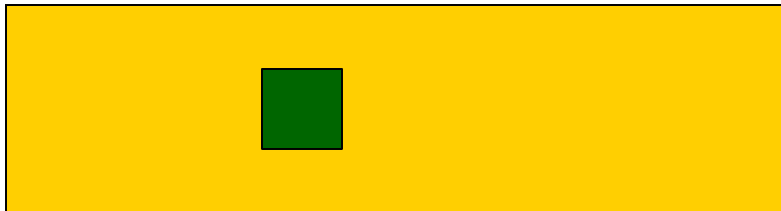
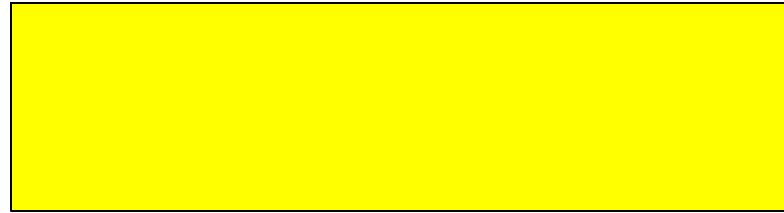
- Address space construction



Reserved for microkernel

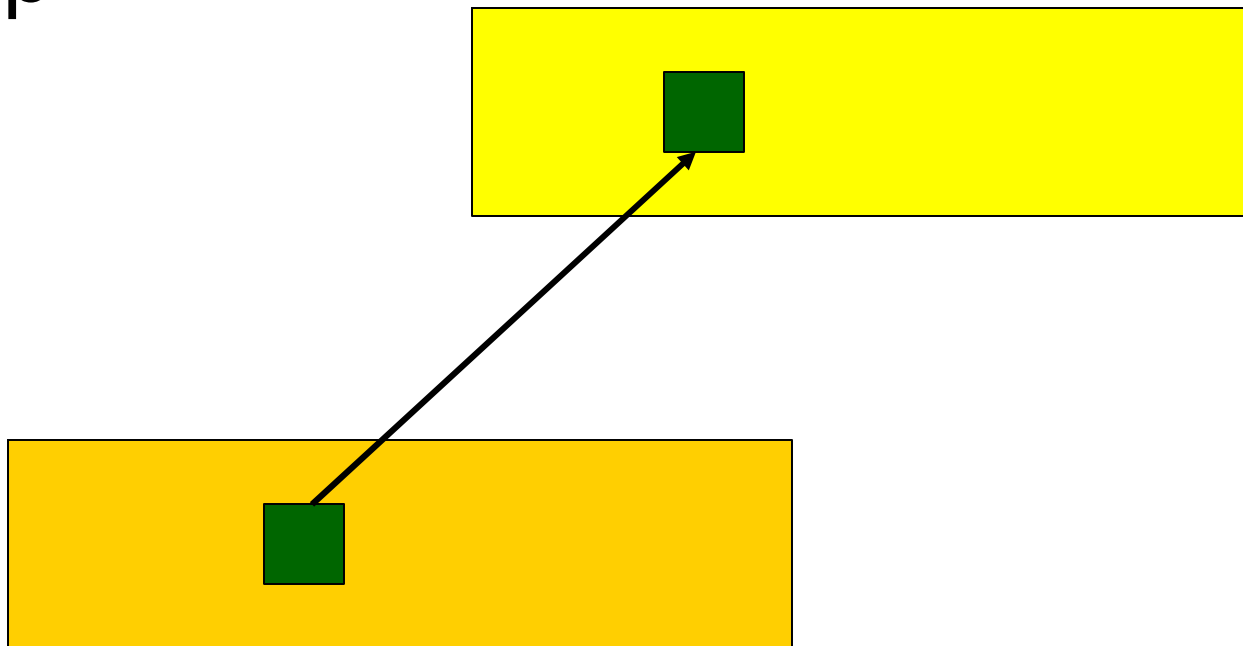
Recursive virtual address space model

- Map



Recursive virtual address space model

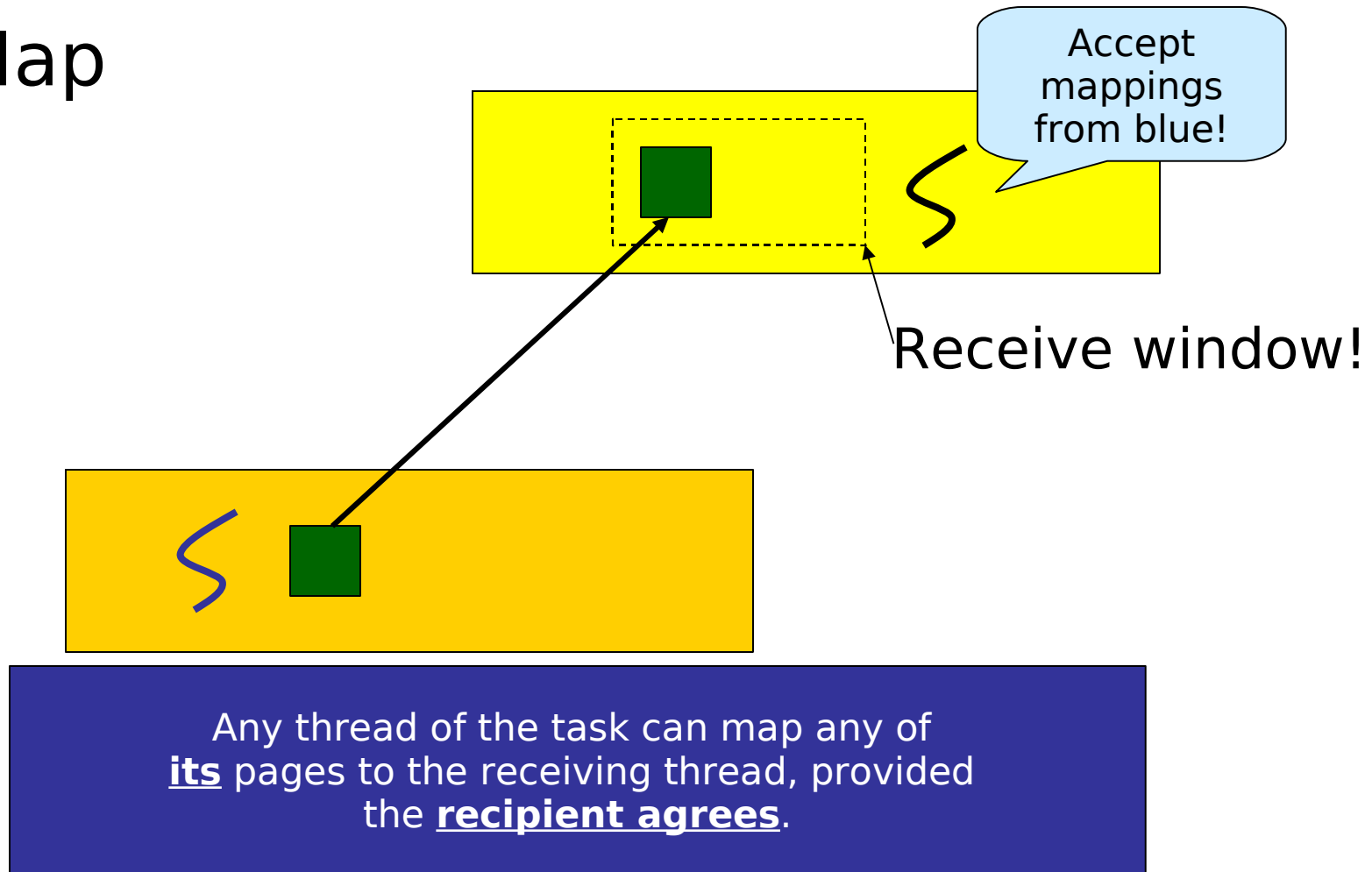
- Map



Any thread of the task can map any of its pages to the receiving thread.

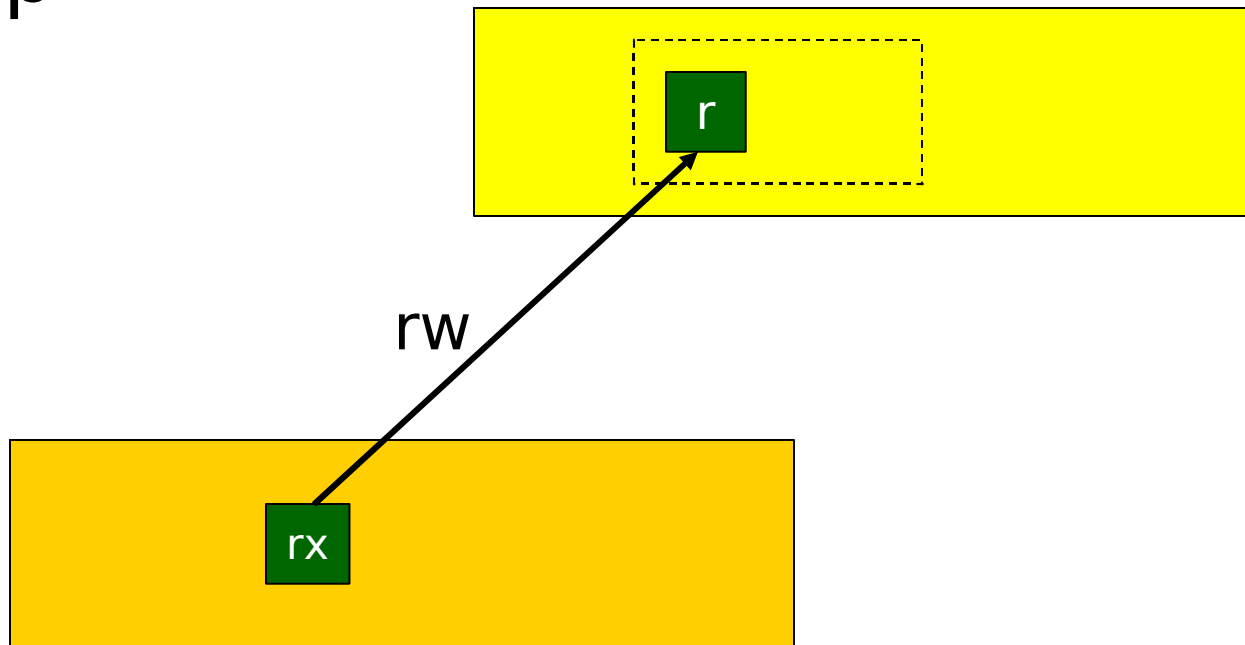
Recursive virtual address space model

- Map



Recursive virtual address space model

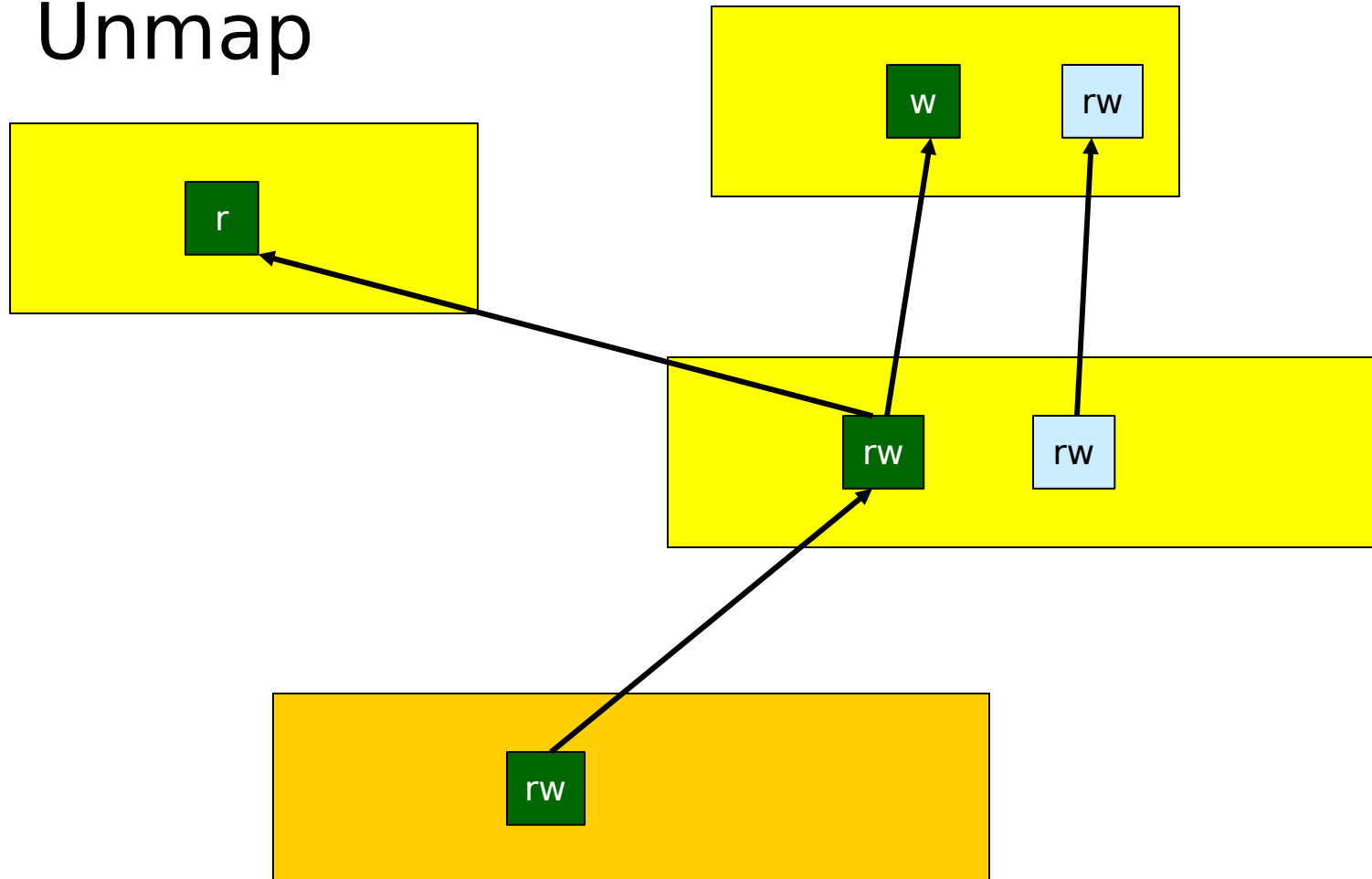
- Map



Map page with minimum of rights.
=> No rights elevation!

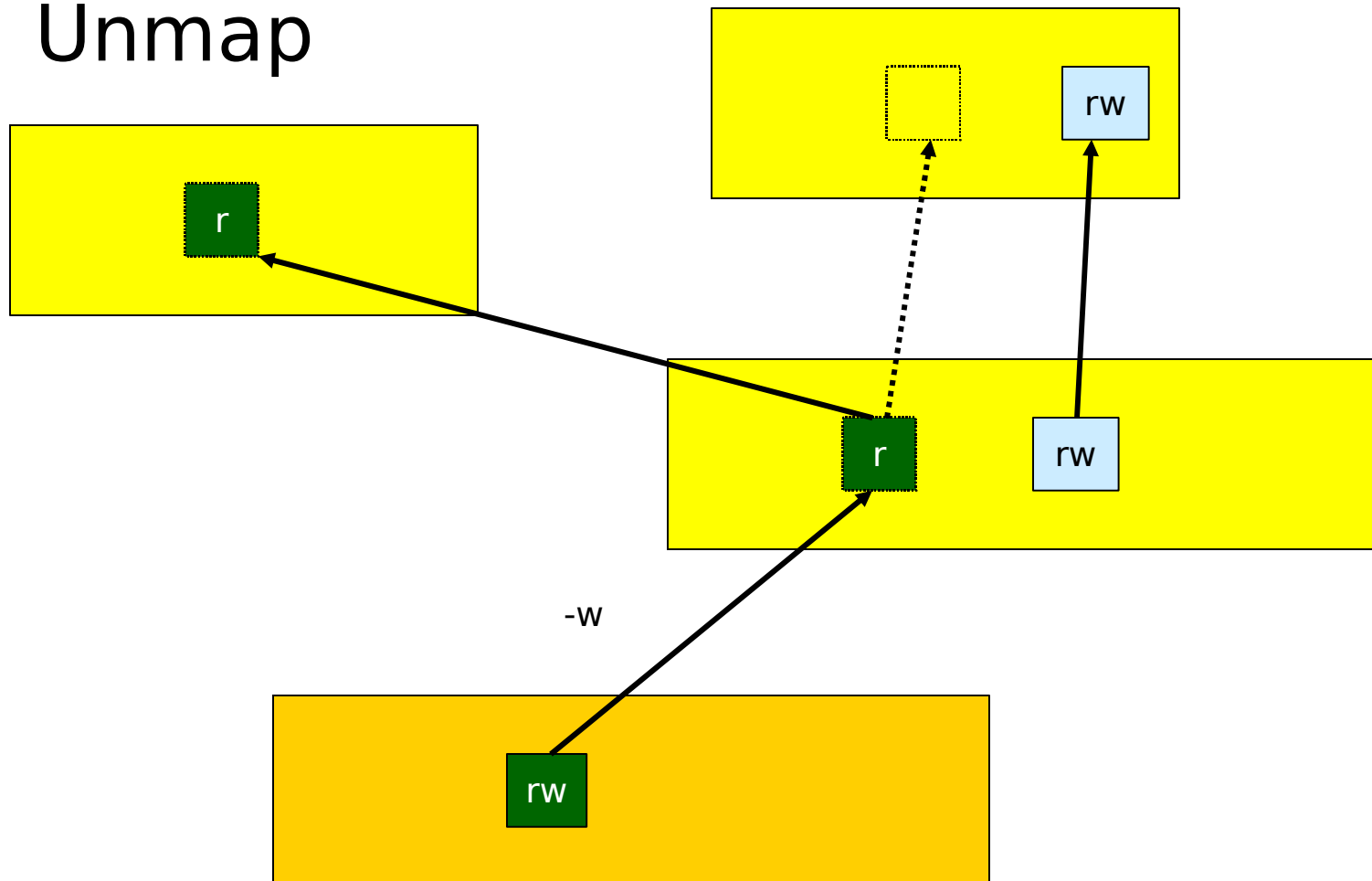
Recursive virtual address space model

■ Unmap



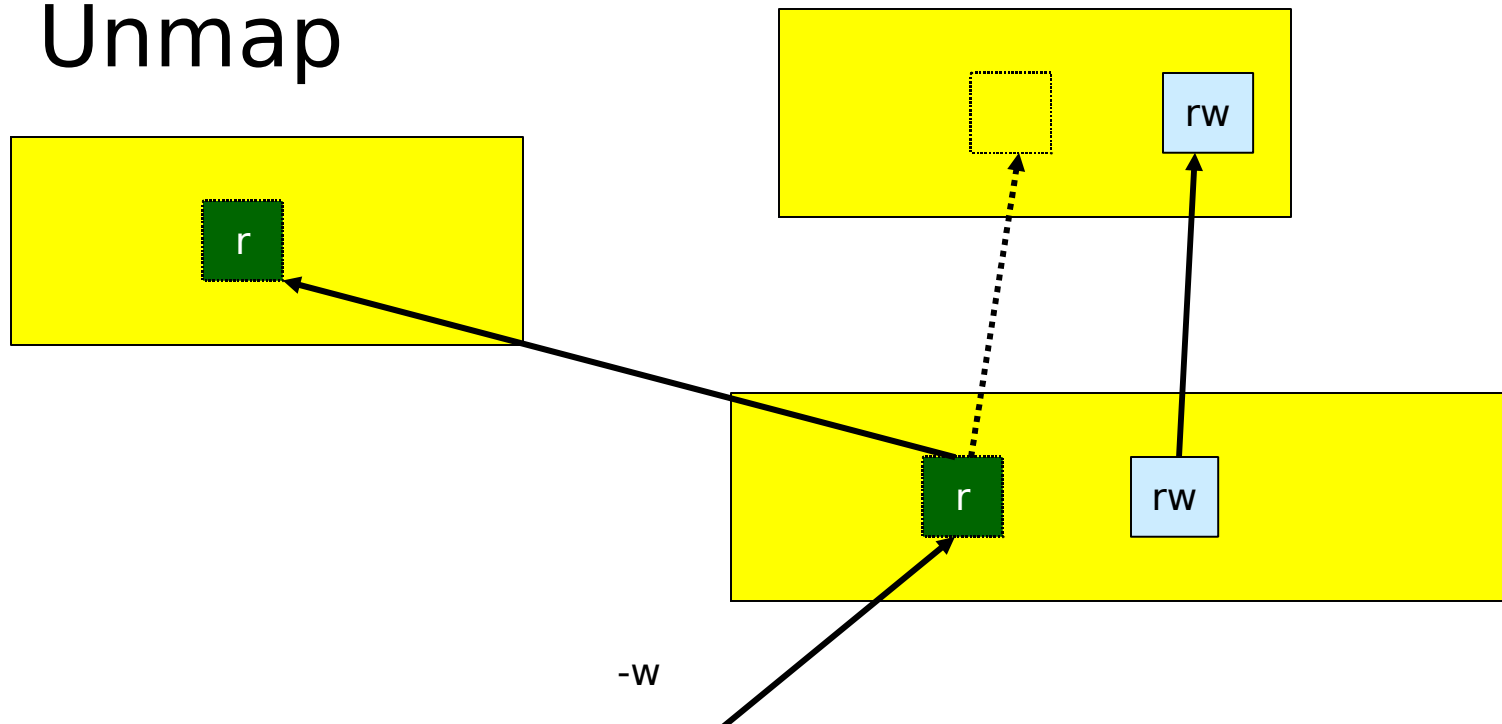
Recursive virtual address space model

■ Unmap



Recursive virtual address space model

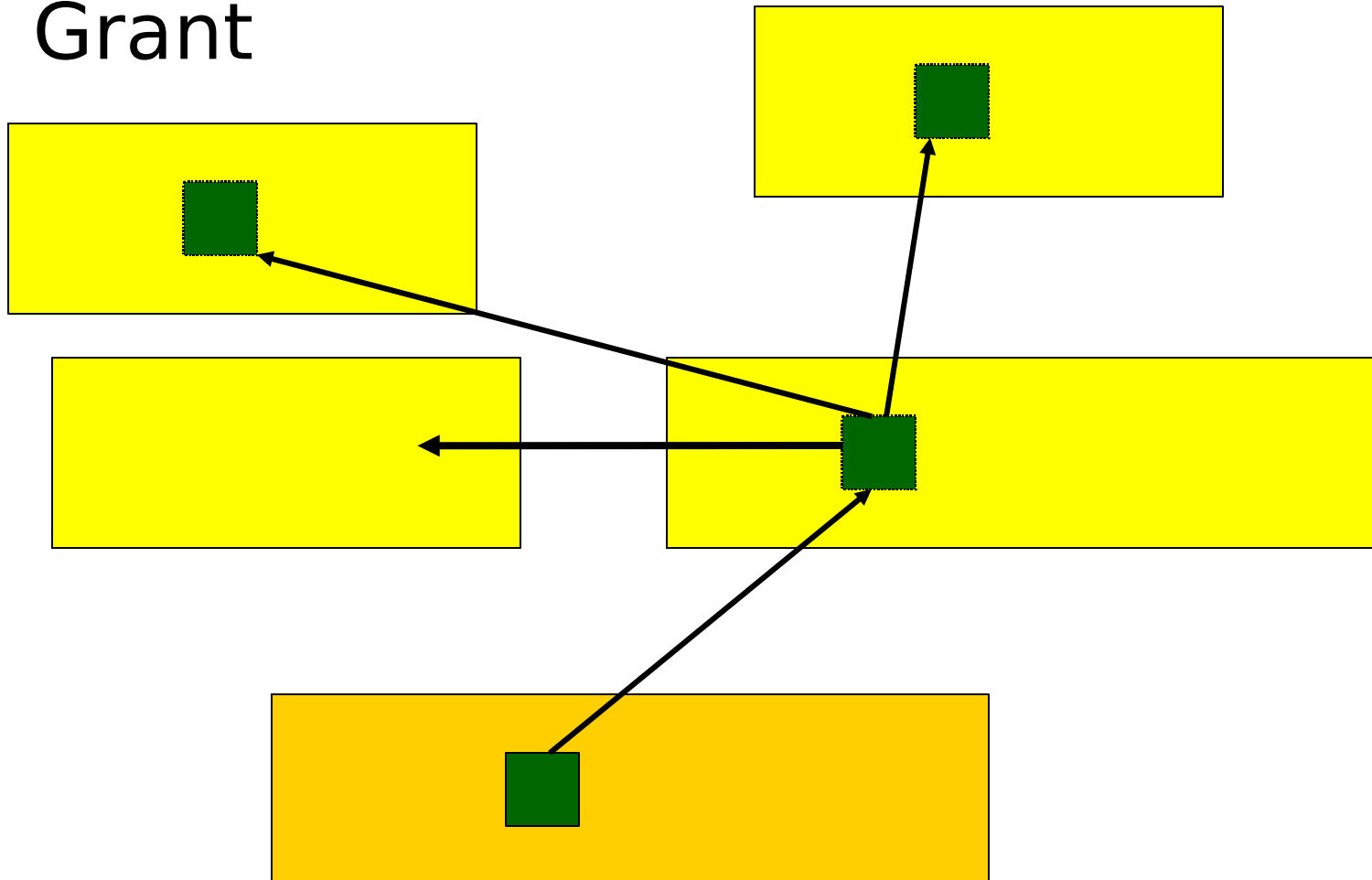
■ Unmap



Recursively revoke access to mapped page.
Without recipient agreement!

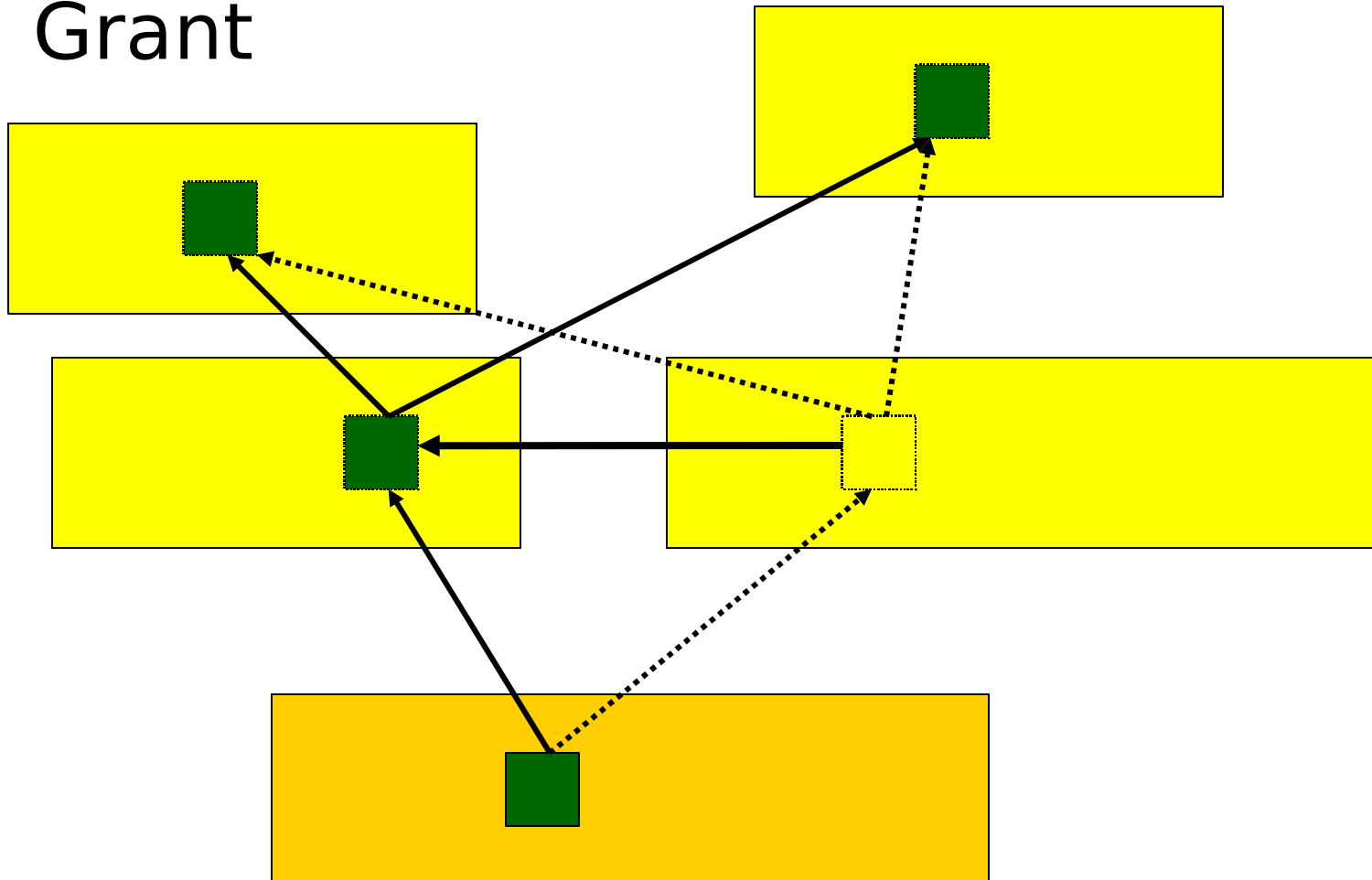
Recursive virtual address space model

- Grant



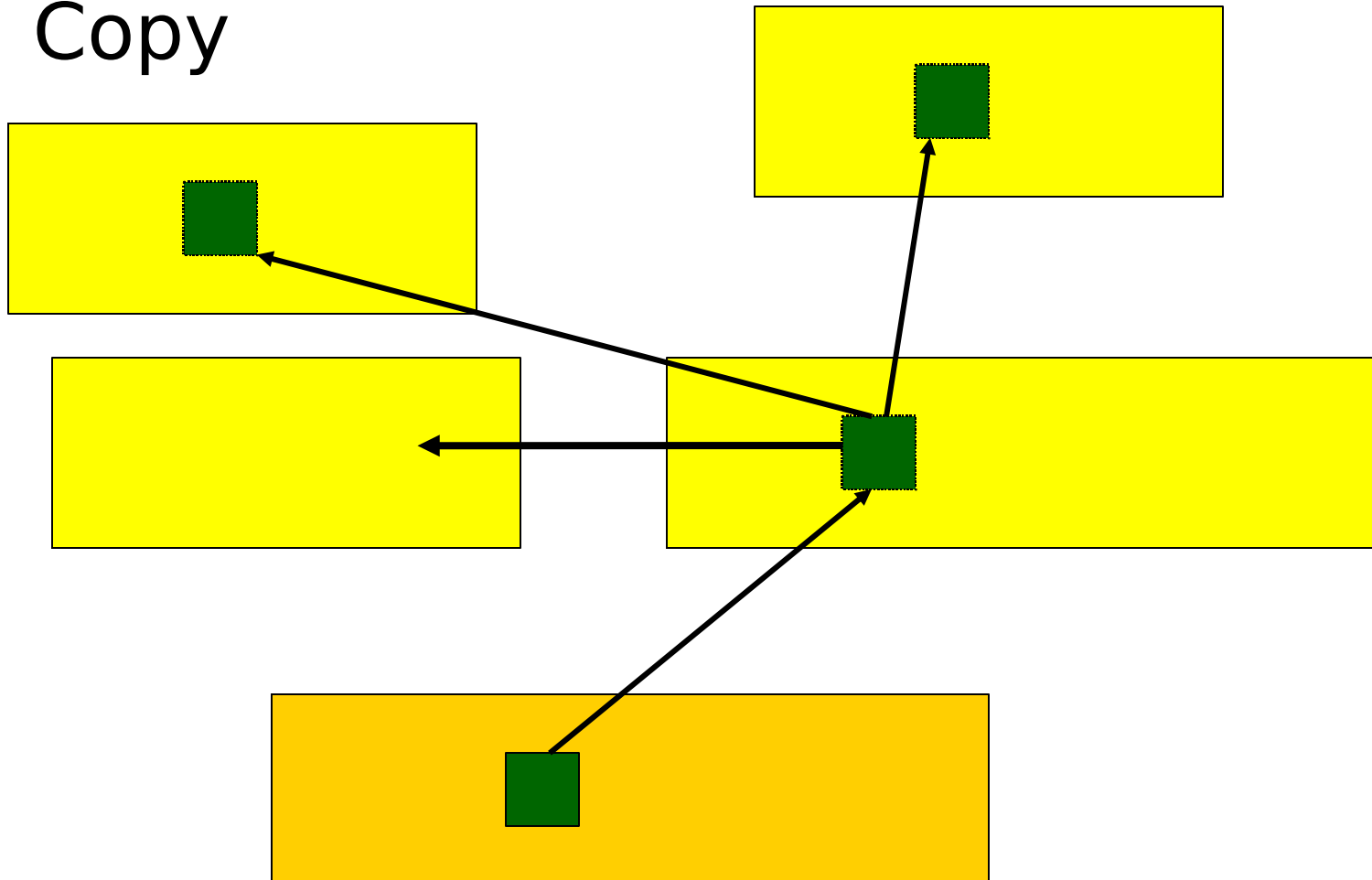
Recursive virtual address space model

- Grant



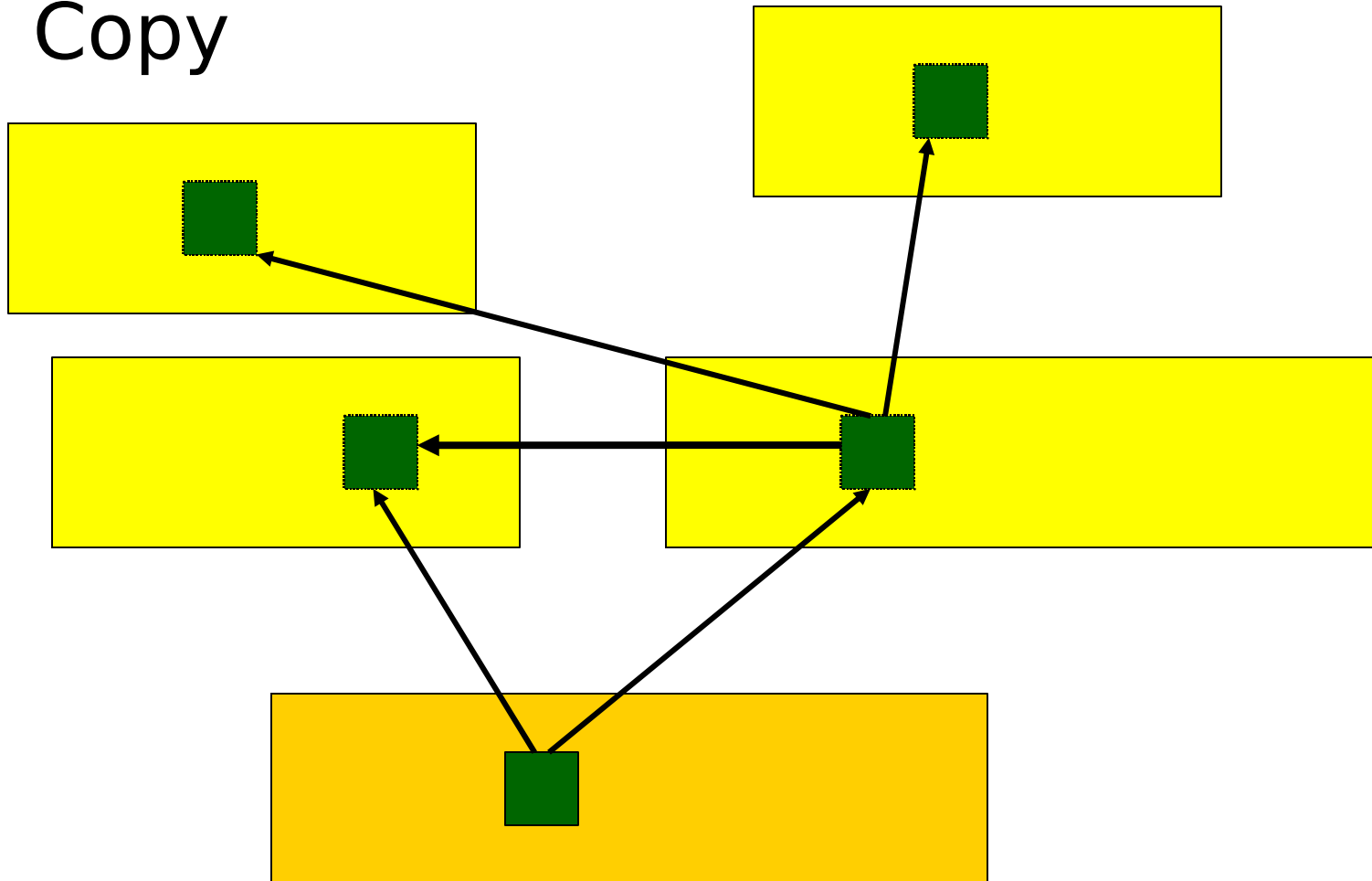
Recursive virtual address space model

- Copy



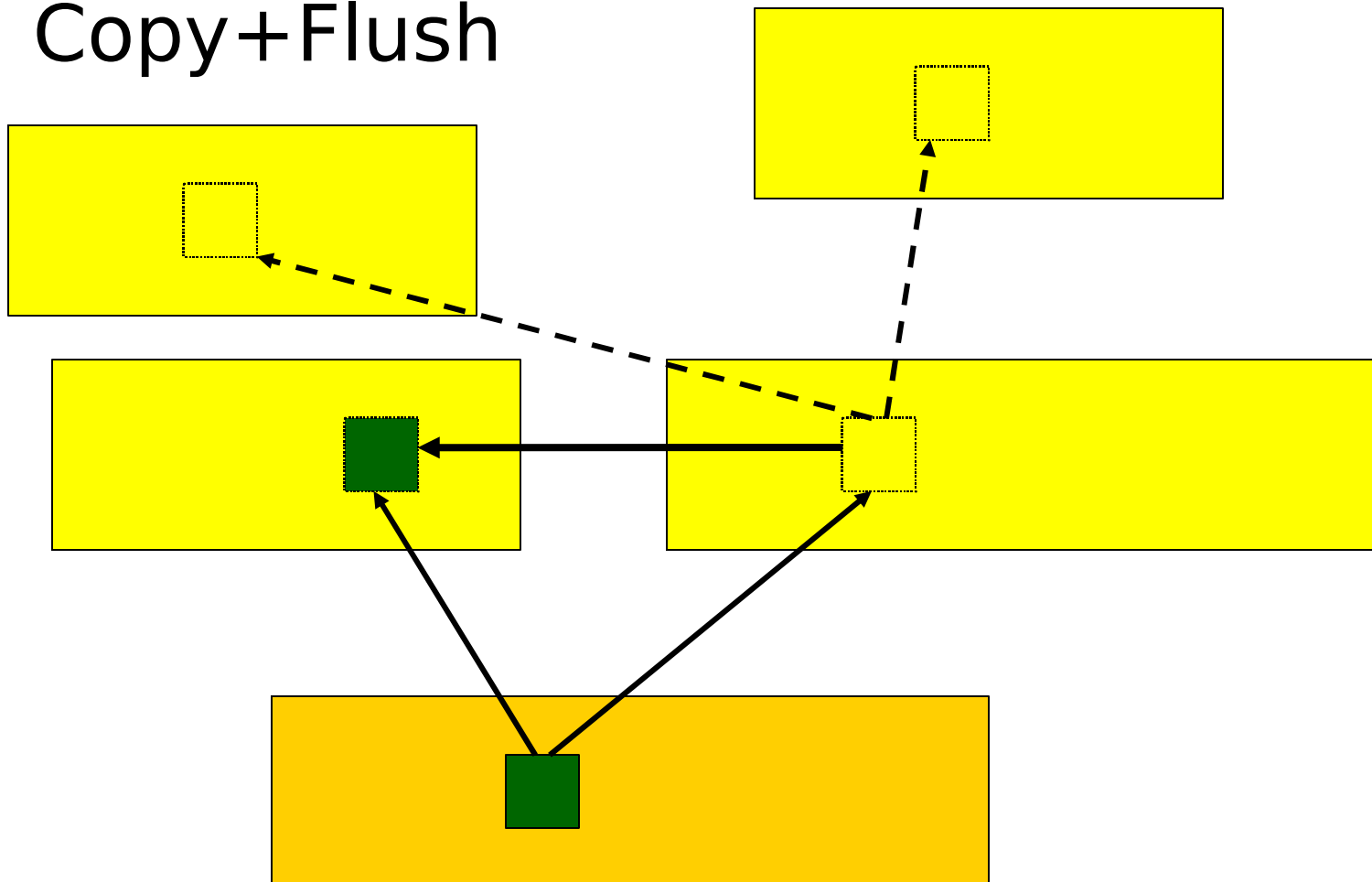
Recursive virtual address space model

- Copy



Recursive virtual address space model

- Copy+Flush



Flexpage

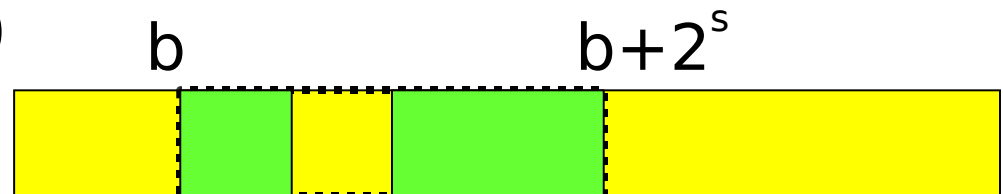
- Abstract from HW page size

- IA 32: 4K, 4M
- IA 64: 4K, 8K, ... , 512M

- Flexpage: b, s

- aligned: $b \% 2^s = 0$

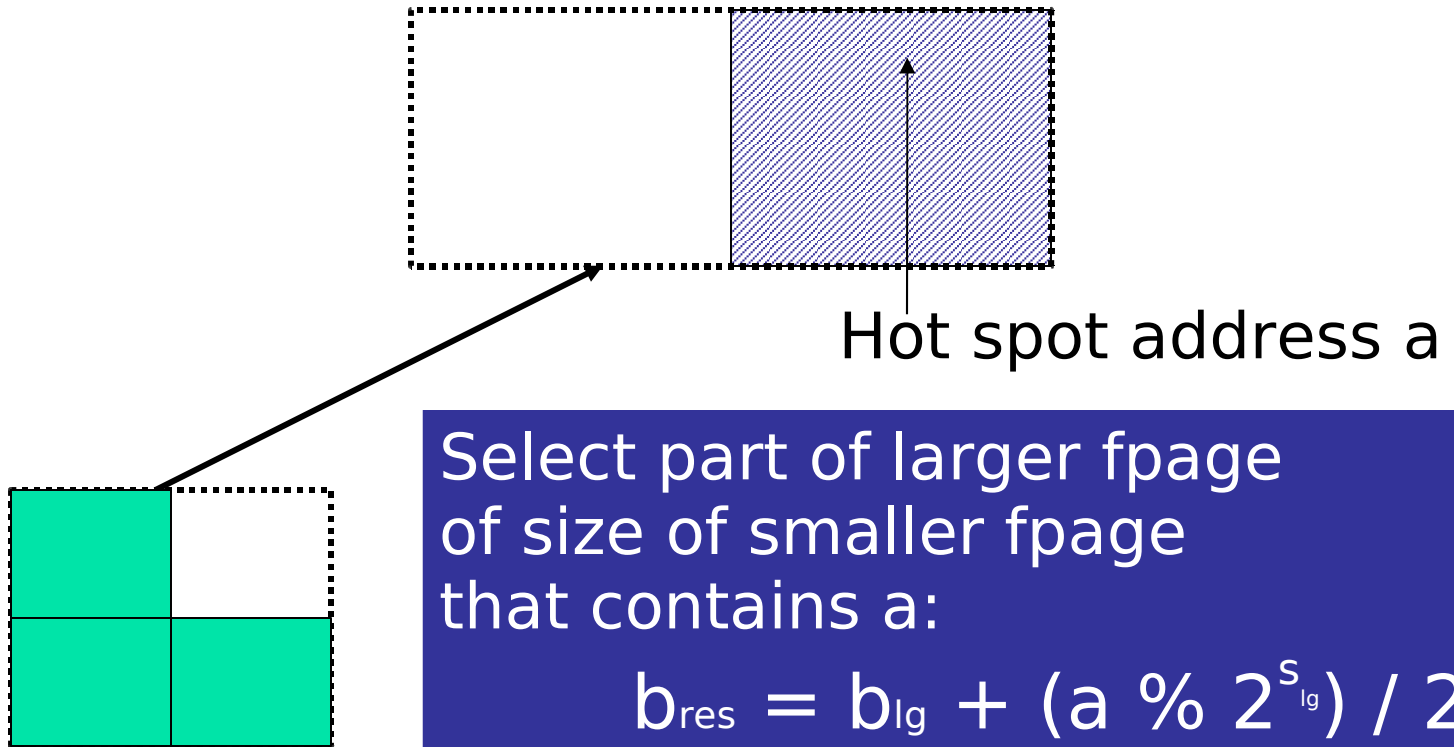
- Size = Power of 2



- Send Flexpage, Receive Flexpage = Rcv Window

Flexpage

- Map / Grant revisited



Select part of larger fpage
of size of smaller fpage
that contains a :

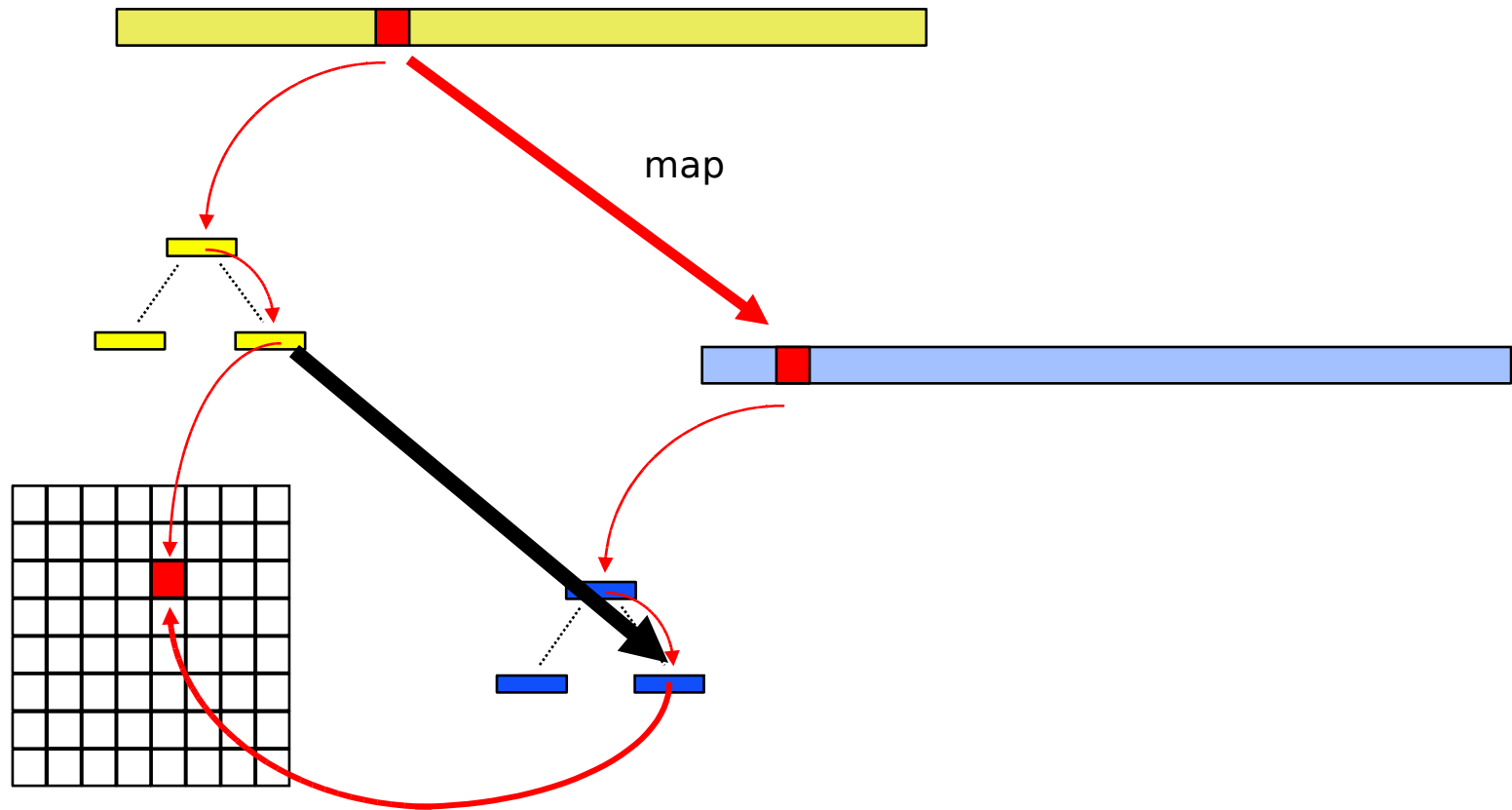
$$b_{\text{res}} = b_{\text{lg}} + (a \% 2^{S_{\text{lg}}}) / 2^{S_{\text{sm}}}$$

$$S_{\text{res}} = S_{\text{sm}}$$

Overview

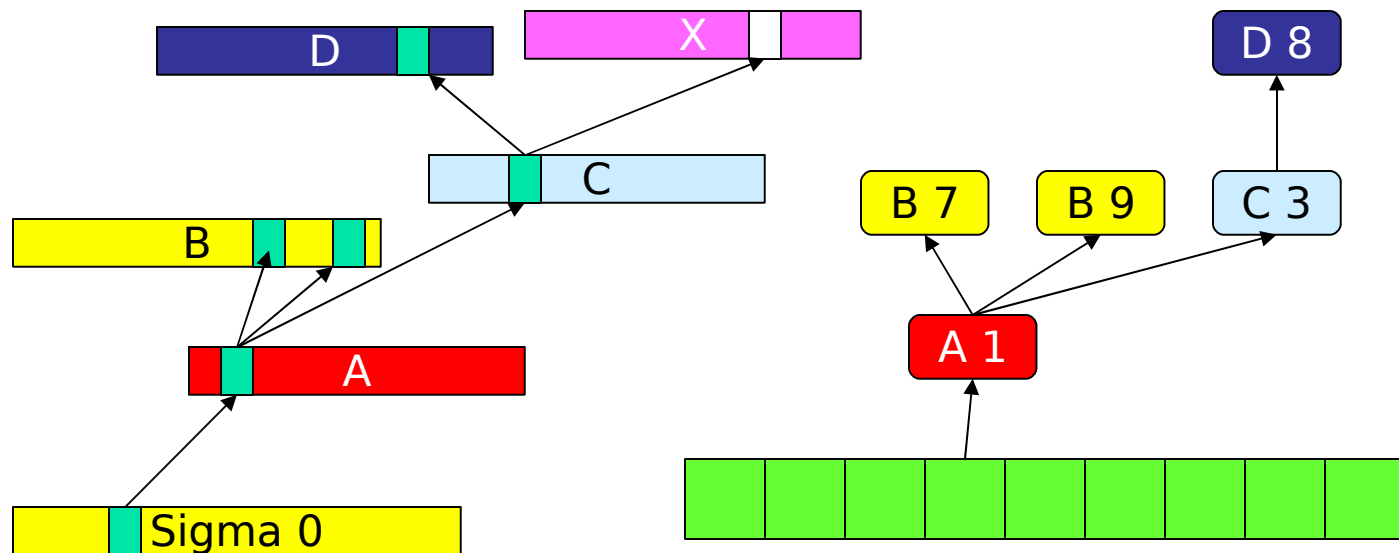
- Introduction
- Address Space Implementation
- Recursive Virtual Address Space Model
 - Map
 - Grant / Copy
 - Unmap
- **Implementation**
 - **Implementation Constraints**
 - **Data Structures**
 - **Synchronization**

Mapping VM - Implementation



Mapping Database

- Mapping database
 - Purpose: iterator (derived mappings)
 - Tile map trees



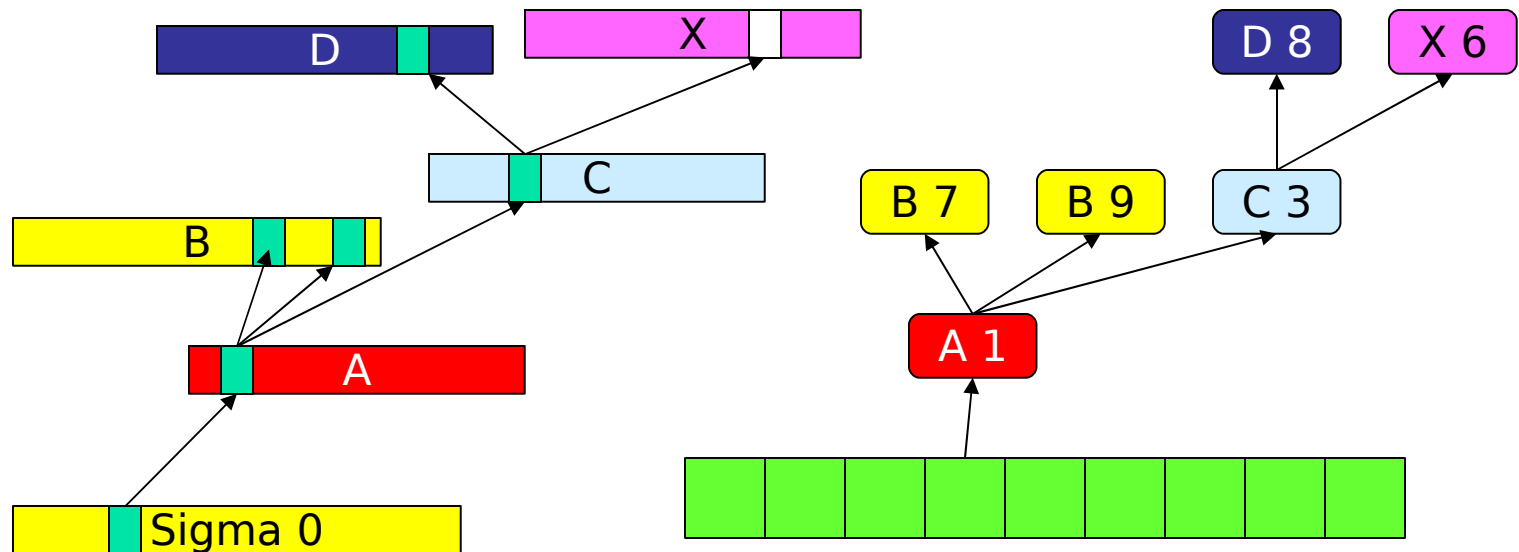
Tiles (physical memory frames)

Implementation Constraints

- **Functionality**
 - Fast Mapping / Overmapping
- **Consistency**
 - No rights elevation
 - Revocability
 - Restartability
- **Real Time**
 - Bounded execution times
 - Avoid long interrupt latencies
- **Resources**
 - Pagetables, Mapping Database
 - Accounting (to mapper / mappee)

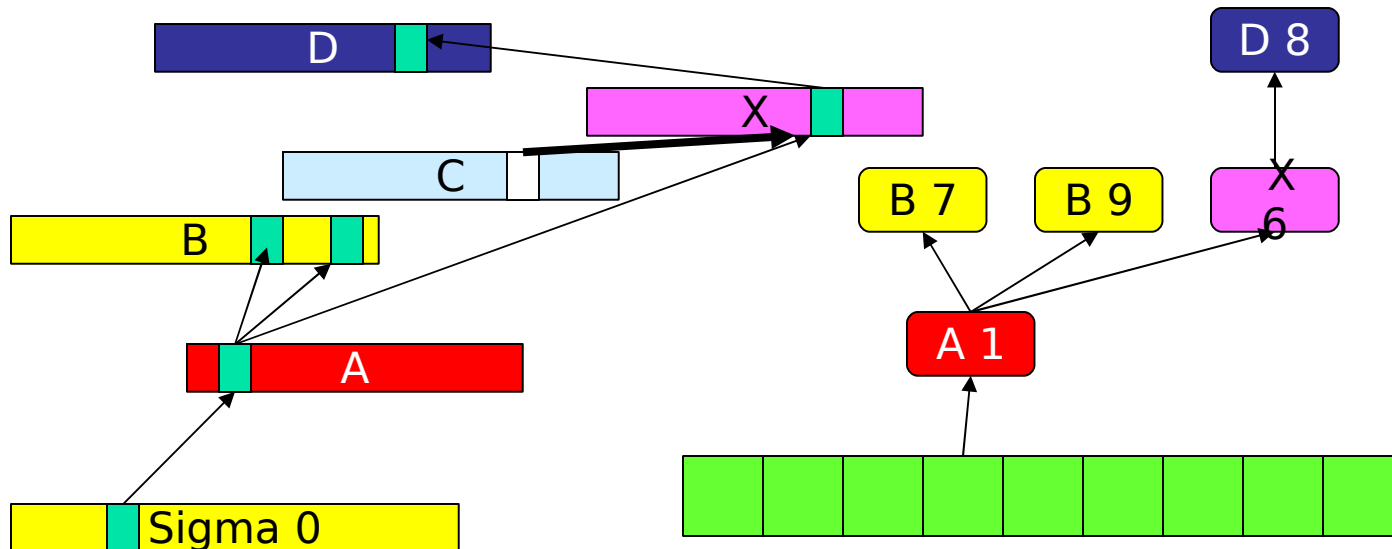
Mapping Database

- Map
 - Insert child node below root node



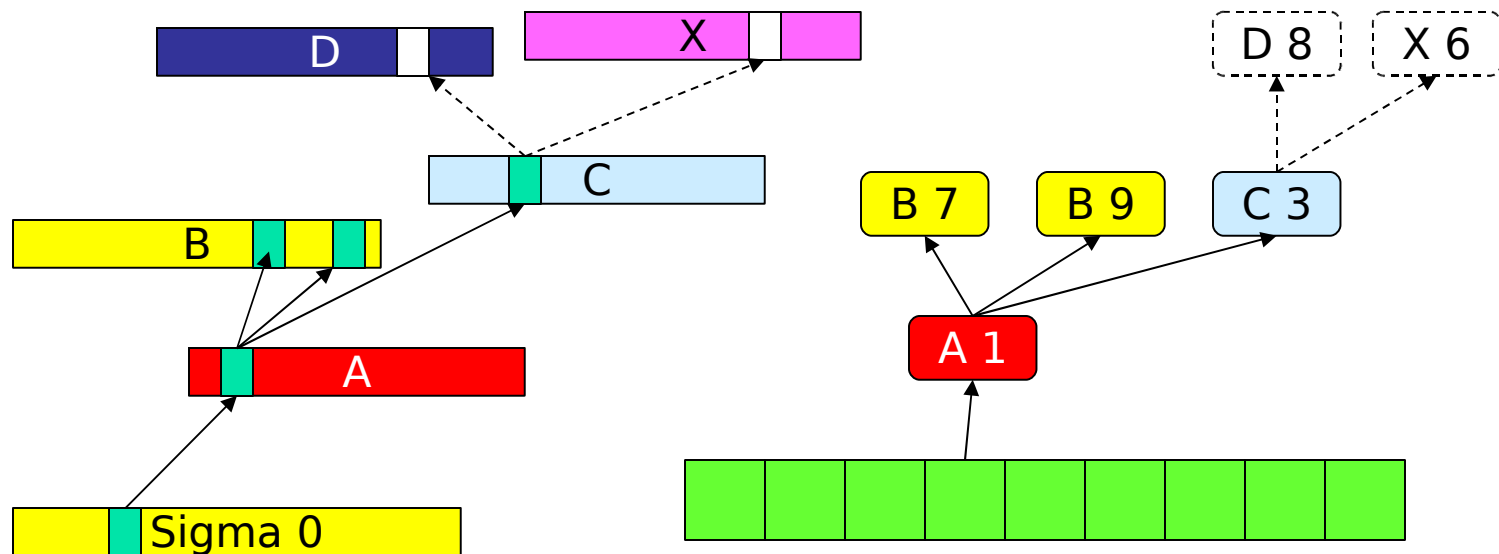
Mapping Database

- Grant
 - Modify root node:
 - Address space
 - Virtual address



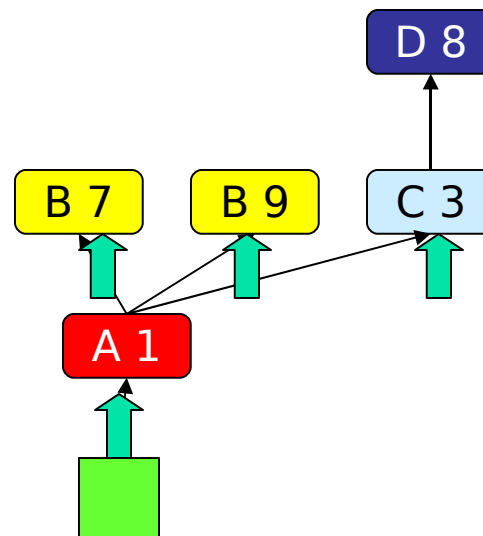
Mapping Database

- Unmap
 - Traverse tree
 - Remove nodes



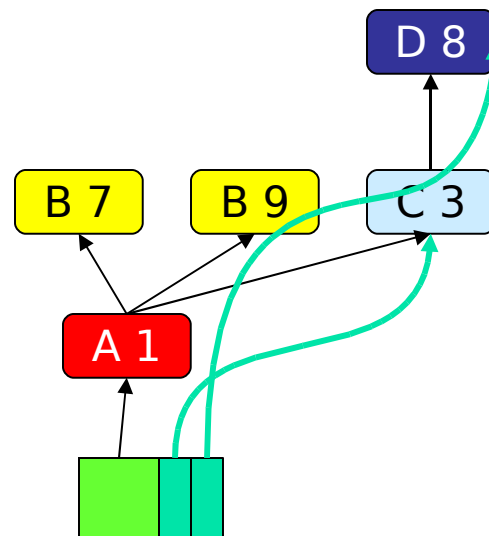
Mapping Database

- How to find the root node
 - Search
 - Problem: Recursion
 - $O(\#nodes)$



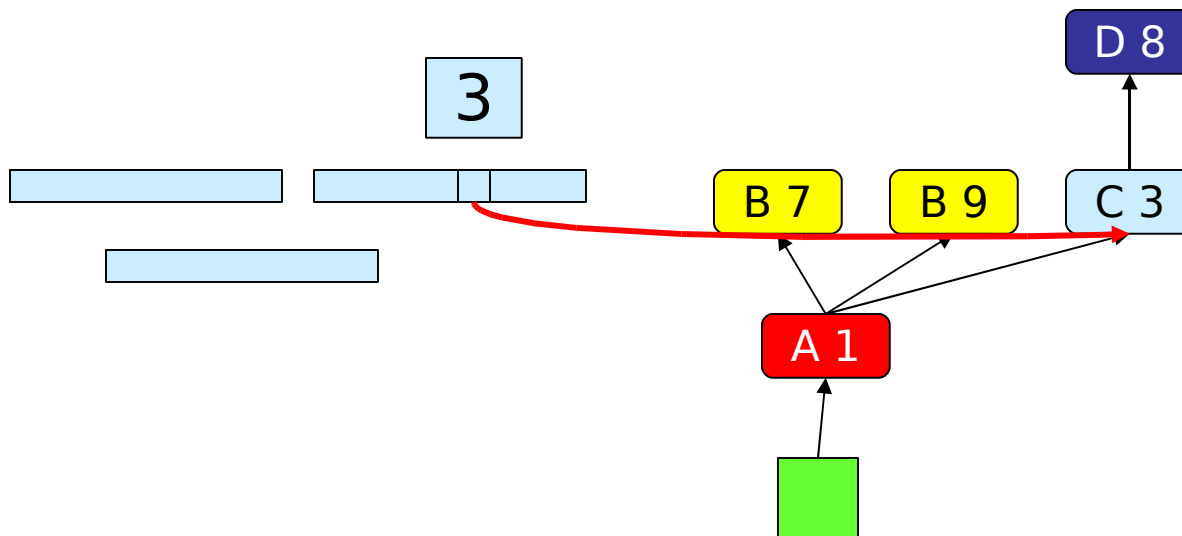
Mapping Database

- How to find the root node
 - Cache



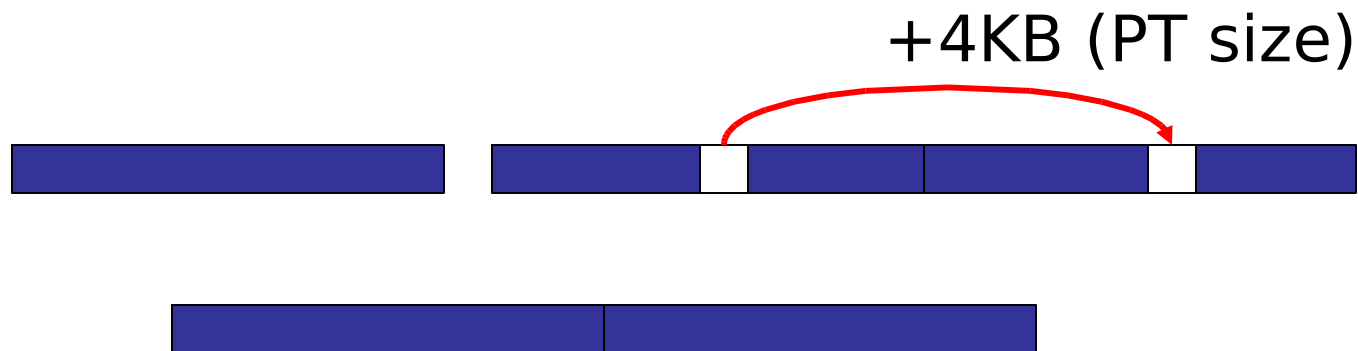
Mapping Database

- How to find the root node
 - PT to Mapnode link



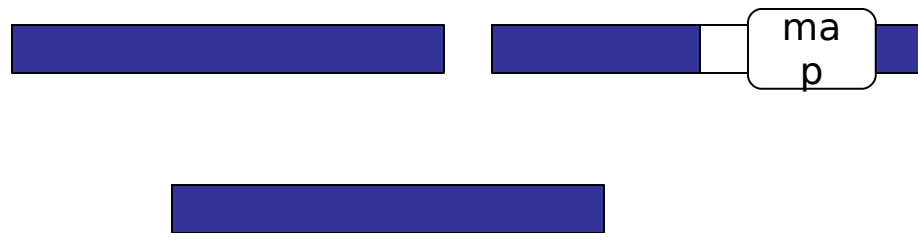
Mapping Database

- How to find the root node
 - PT to Mapnode link



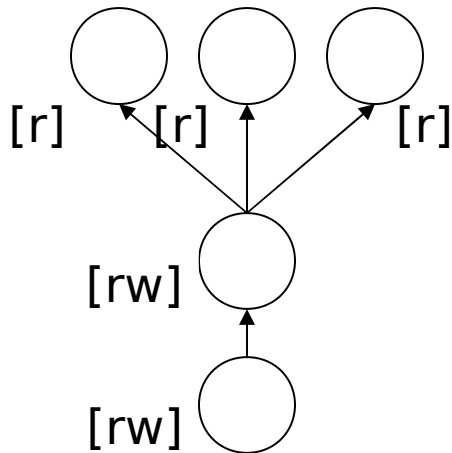
Mapping Database

- Integrating PT + MDB node



Better memory footprint
But: Cache footprint!

Fast overmapping

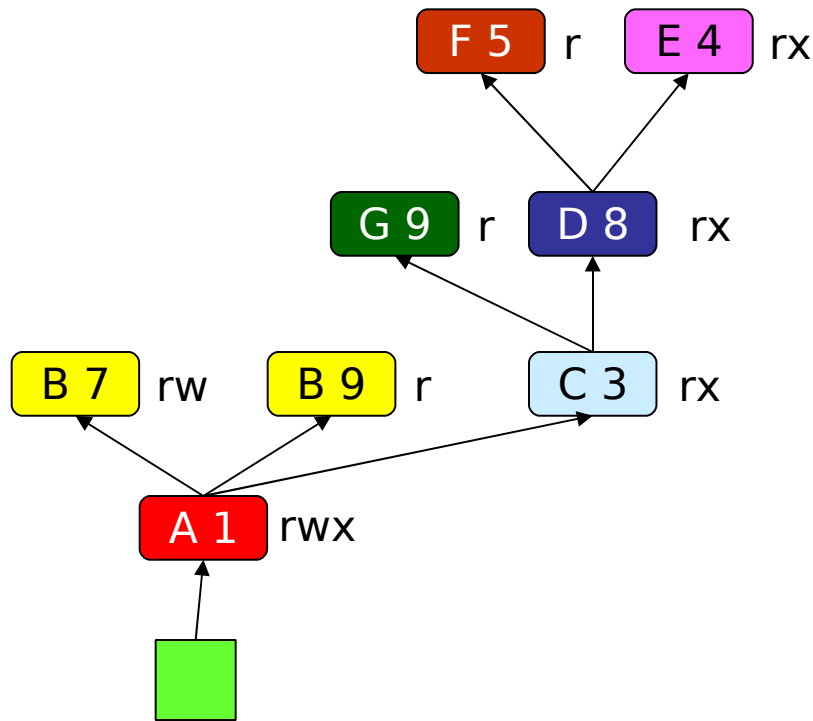


- Same source address space
- Same virtual source address
- Same virtual destination address

Implementation Constraints

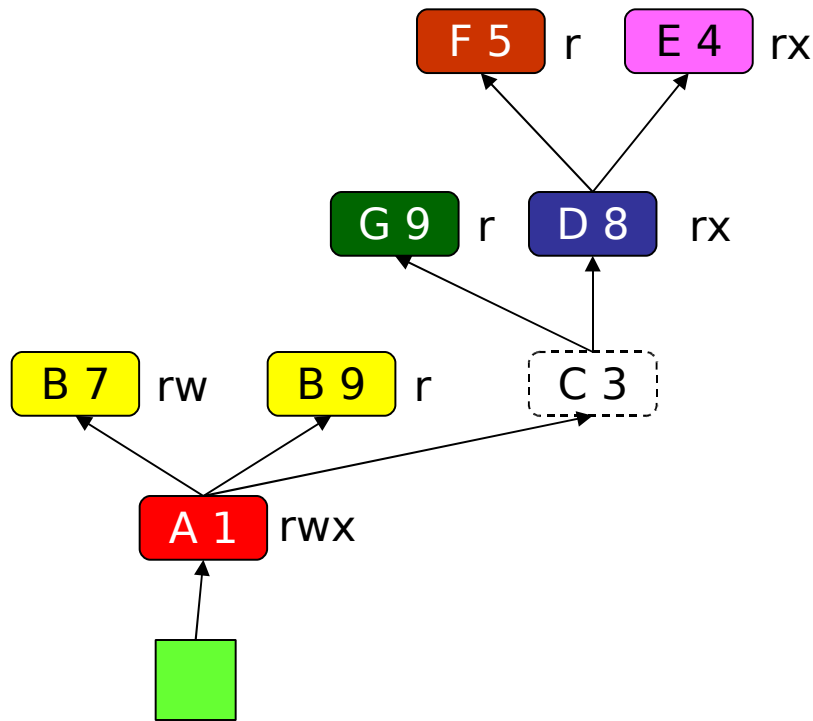
- **Functionality**
 - Fast Mapping / Overmapping
- **Consistency**
 - No rights elevation
 - Revocability
 - Restartability
- **Real Time**
 - Bounded execution times
 - Avoid long interrupt latencies
- **Resources**
 - Pagetables, Mapping Database
 - Accounting (to mapper / mappee)

Database consistency



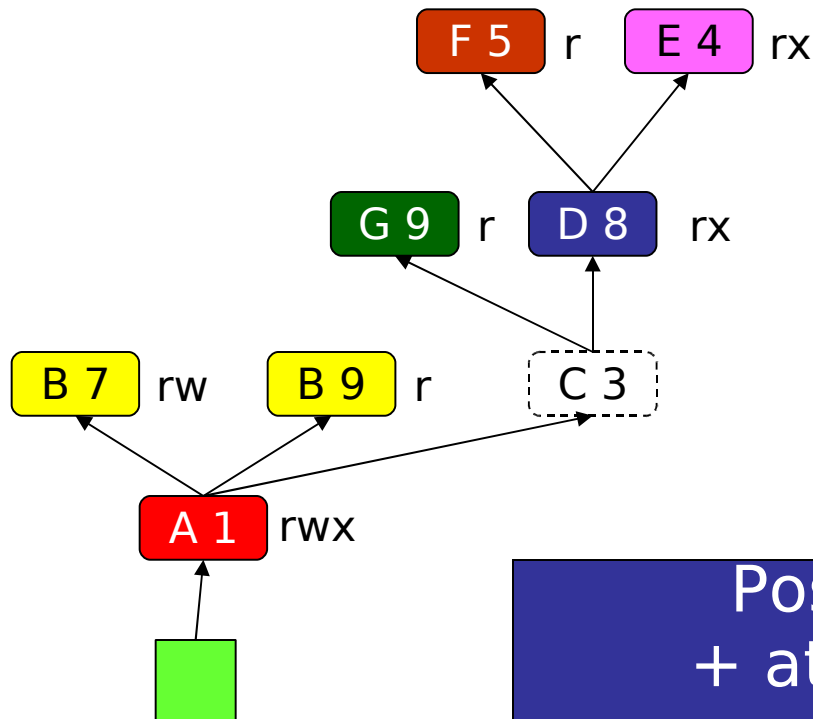
- Map completed:
 - Both tasks have access
- Unmap completed:
 - No access above subtree root-node
- No rights elevation

Database consistency



- Map completed:
 - Both tasks have access
- Unmap completed:
 - No access above subtree root-node
- No rights elevation

Database consistency



- Map completed:
 - Both tasks have access
- Unmap completed:
 - No access above subtree root-node

■ No rights elevation
Post order traversal
+ atomic node ops =>
consistency

Implementation Constraints

- **Functionality**
 - Fast Mapping / Overmapping
- **Consistency**
 - No rights elevation
 - Revocability
 - Restartability
- **Real Time**
 - Bounded execution times
 - Avoid long interrupt latencies (Synchronization)
- **Resources**
 - Pagetables, Mapping Database
 - Accounting (to mapper / mappee)

Real Time

- Bounded Execution Time
 - Limit Size of Subtree
 - complement rights with max size
 - map $(rw, 5) : [rwx, 10] \rightarrow [r, 2] \mid - [rwx, 5] \rightarrow [rw,5],[r,2]$
 - Enforce Flat Hierarchy
 - map privilege
 - Instantaneous Revocation
 - invalidate mappings instantaneous
 - increase version number / use page tables as cache only (EROS)

Resources

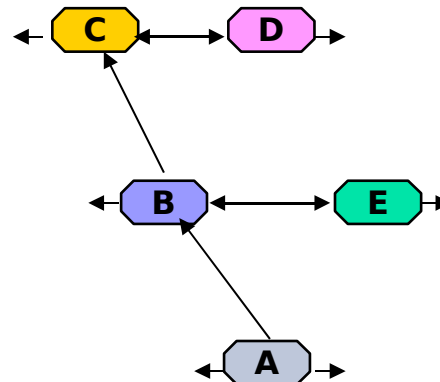
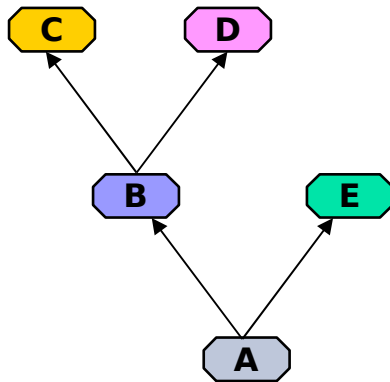
- Pagetables + Nodes in Tree (Mapping Nodes)
- Accounting:
 - Mapper decides what to map
 - page granularity (sequence of small fpages)
 - Mappee benefits from mapping
 - Page tables are used by different servers
 - Code + Files are mapped to same page directory (or same page table if in same 4M area?)
- Solutions:
 - allocate from mapper's quota
 - L4.Sec: mapper defines resource flexpage
 - **Prealloc resources via separate syscall???**

Overview

- Introduction
- Address Space Implementation
- Recursive Virtual Address Space Model
 - Map
 - Grant / Copy
 - Unmap
- Implementation
 - Implementation Constraints
 - Data Structures
 - Synchronization

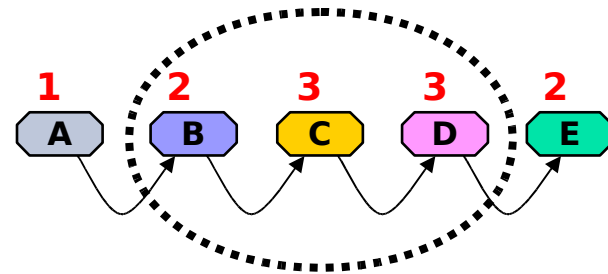
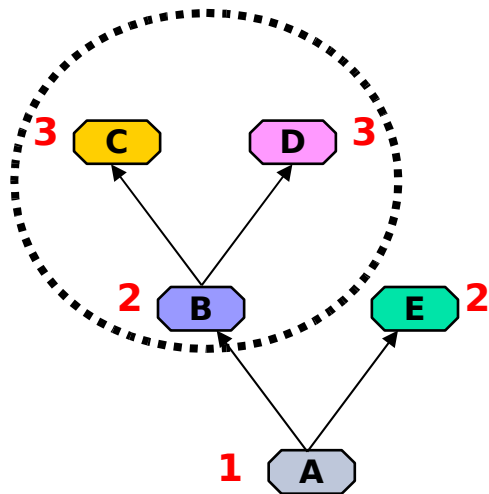
Data Structures

- Tree



Data Structures

- avoid recursion: preorder sorted list



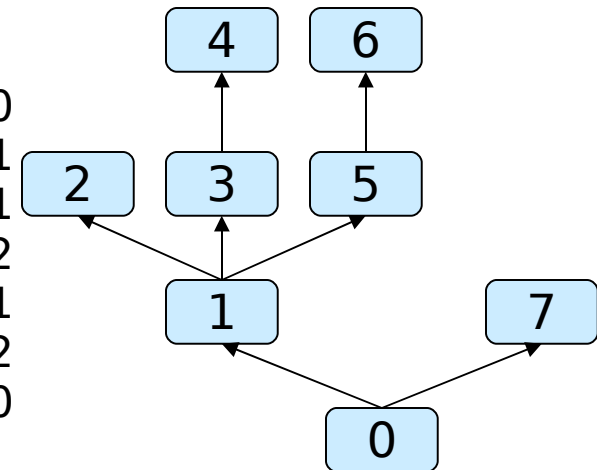
Post order traversal:

- 1) go forward to leaf,
- 2) remove leaf while going backwards

Fiasco MDB

- Densely packed array: pre order
 - Shrink / grow array when needed
(4 ... 4 <<15)

<u>array element</u>	<u>depth value</u>	
0	0	sigma 0 node
1	1	child of element #0 with depth 0
2	2	child of element #1 with depth 1
3	2	child of element #1 with depth 1
4	3	child of element #3 with depth 2
5	2	child of element #1 with depth 1
6	3	child of element #5 with depth 2
7	1	child of element #0 with depth 0



Small memory footprint, but: copying, accounting

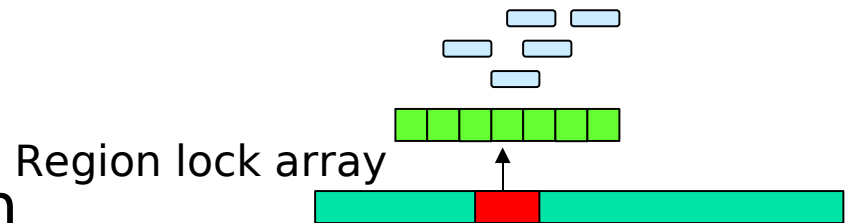
Overview

- Introduction
- Address Space Implementation
- Recursive Virtual Address Space Model
 - Map
 - Grant / Copy
 - Unmap
- Implementation
 - Implementation Constraints
 - Data Structures
 - Synchronization

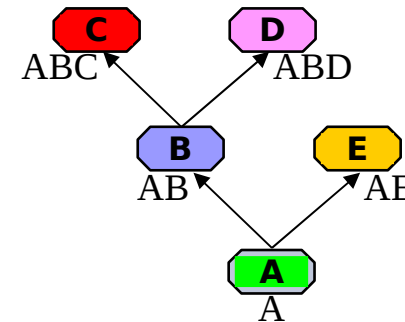
Locking

- Coarse grain locking
 - Lock entire database

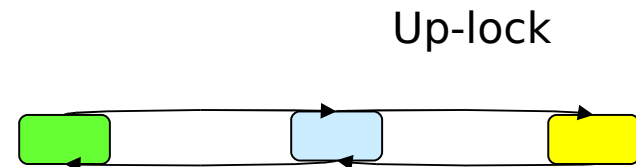
- Region based
 - Lock memory region



- Fine grain
 - Subtree Locks

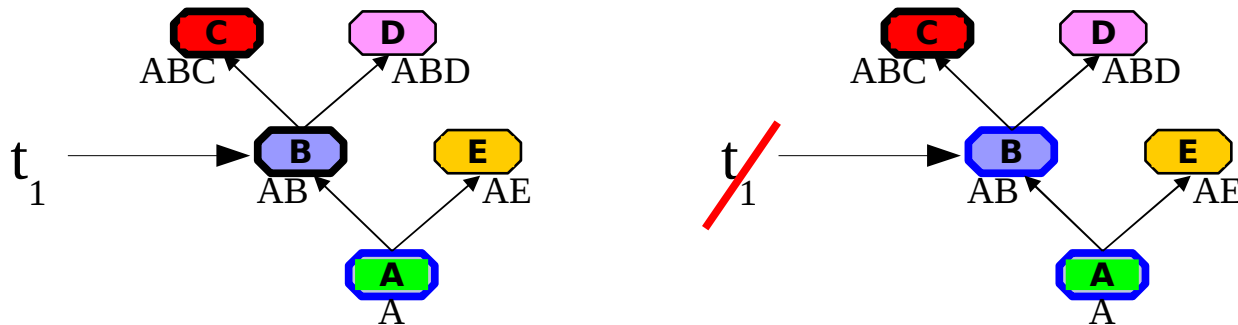


- Lock single nodes



Locking

- More subtree locks:
 - Change semantics to ease locking:
 - Parent unmap precedes child unmap
 - Abort child unmap if parent unmap passes



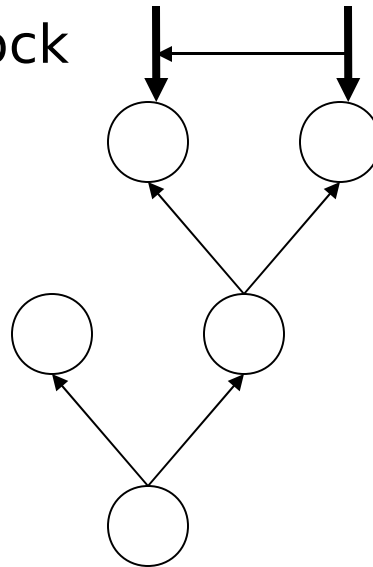
Real Time – Long Interrupt Latency

- Run MDB operations preemptively but synchronize MDB OPs wrt each other!
 - Short interrupt latencies provided interrupt handler does not page fault!
- Preemptive MDB
 - Map waits for unmap to complete
 - Long interrupt latencies when faulting-frame is unmapped
 - Map may surpass unmap
 - Short interrupt latencies in common case
 - Unmap (+ page replacement) is delayed

Guaranteeing Forward Progress

■ A second approach: Helping

Subtree Helping Lock



revoke access



next = remove node



⇒ Helping only for unmap,
implement map as atomic sequence

References

- On Microkernel Construction (J. Liedtke)
- A Framework for VM Diversity (J. Liedtke, et. al)
 - *map, unmap, grant*
- Design and Implementation of the Recursive Virtual Address Space Model for Small Scale Multiprocessor Systems (M. Völp DA)
http://i30www.ira.uka.de/teaching/thesisdocuments/l4ka/2002/voelp_dt_recursive-virtual-address-space-model.pdf
 - *MDB consistency*
- L4. Sec Implementation - Kernel Memory Management (B. Kauer DA)
http://os.inf.tu-dresden.de/papers_ps/kauer-diplom.pdf
 - *L4.Sec / Memory Management Issues*