

# Buddy and Slab Allocators

2024 Fall ECE 344: Operating Systems  
Jon Eyolfson

Lecture 32  
2.0.0

## The Buddy Allocator Restricts the Problem

Typically, allocation requests are of size  $2^n$

e.g. 2, 4, 8, 16, 32, ..., 4096, ...

Restrict allocations to be powers of 2 to enable a more efficient implementation

Split blocks into 2 until you can handle the request

We want to be able to do fast searching and merging

## You Can Implement the Buddy Allocator Using Multiple Lists

We restrict the requests to be  $2^k$ ,  $0 \leq k \leq N$  (round up if needed)

Our implementation would use  $N + 1$  free lists of blocks for each size

For a request of size  $2^k$ , search the free list until we find a big enough block

- Search  $k, k + 1, k + 2, \dots$  until we find one

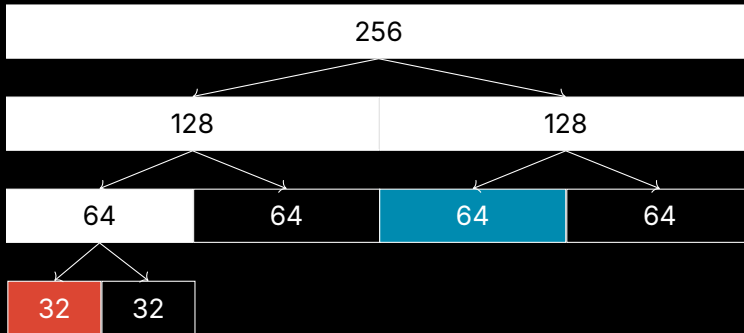
- Recursively divide the block if needed until it's the correct size

- Insert "buddy" blocks into free lists

For deallocations, we coalesce the buddy blocks back together

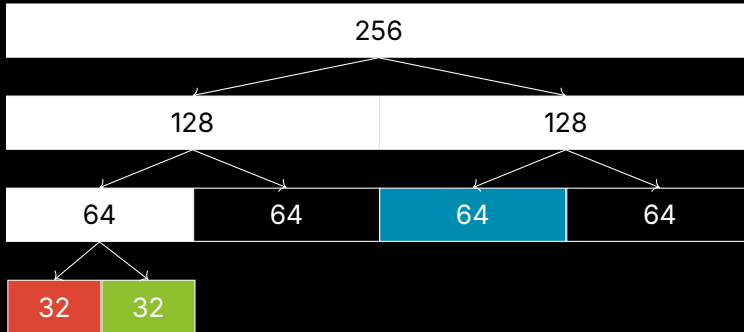
- Recursively coalesce the blocks if needed

## Using the Buddy Allocator (1)



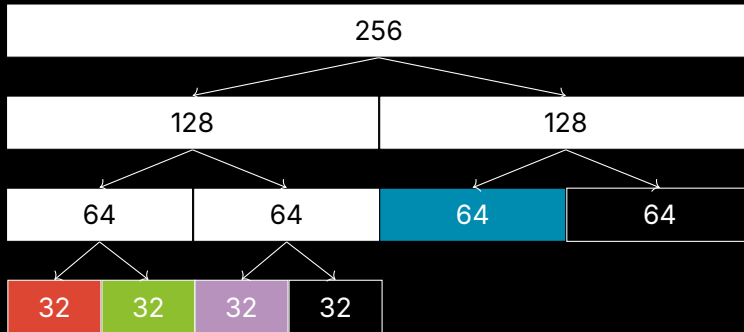
Where do we allocate a request of size 28?

## Using the Buddy Allocator (2)



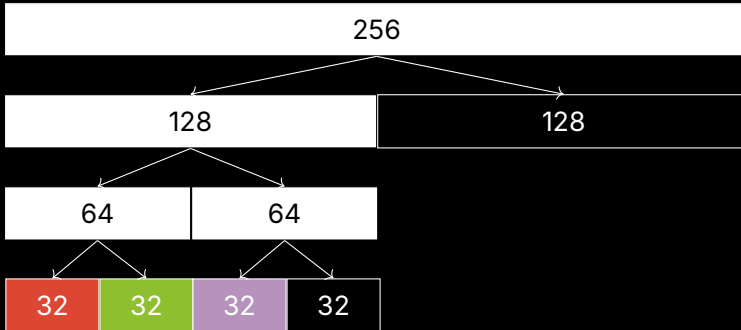
Where do we allocate a request of size 32?

## Using the Buddy Allocator (3)



What happens when we free the size 64 block?

## Using the Buddy Allocator (4)



## Buddy Allocators are Used in Linux

### Advantages

- Fast and simple compared to general dynamic memory allocation
- Avoids external fragmentation by keeping free physical pages contiguous

### Disadvantages

- There's always internal fragmentation
- We always round up the allocation size if it's not a power of 2



## **Slab Allocators Take Advantage of Fixed Size Allocations**

Allocate objects of same size from a dedicated pool

All structures of the same type are the same size

Every object type has it's own pool with blocks of the correct size

This prevents internal fragmentation

## Slab is a Cache of "Slots"

Each allocation size has a corresponding slab of slots  
(one slot holds one allocation)

Instead of a linked list, we can use a bitmap  
(there's a mapping between bit and slot)

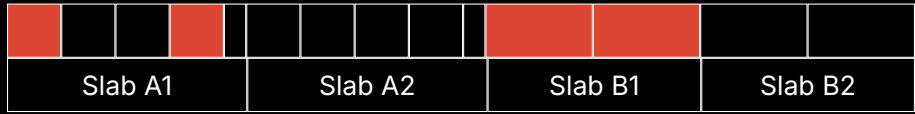
- For allocations, we set the bit and return the slot

- For deallocations, we just clear the bit

The slab can be implemented on top of the buddy allocator

## Each Slab Can Be Allocated using the Buddy Allocator

Consider two object sizes: A and B



We can reduce internal fragmentation if Slabs are located adjacently  
In this example A has internal fragmentation (dark box)

## Even More Memory Allocations

The kernel restricts the problem for better memory allocation implementations

- Buddy allocator is a real-world restricted implementation
- Slab allocator takes advantage of fixed sized objects to reduce fragmentation