

CPSC 213 Lab 3

Dynamic Arrays & C Pointers

Slides available at randyzhu.com/cpsc213

Randy Zhu

Course Updates

Quiz 1; A3 due Friday

- Assignment 3 is due on Friday, January 30th
- Quiz 1 is running today and tomorrow
 - No questions about quizzes

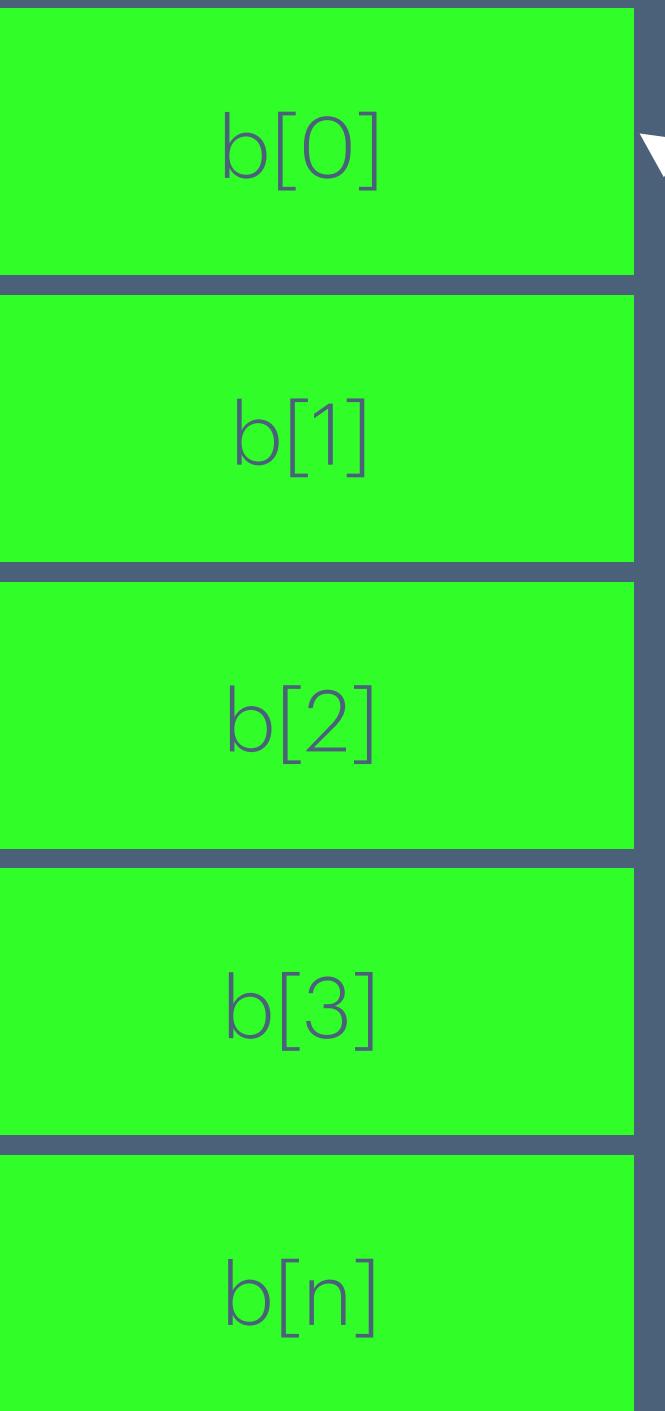
Pointer Review

What is a pointer?

- A variable that contains a memory address
- Syntax examples:
 - `int* ptr;`
 - `unsigned char* m;`
 - `int** matrix;`
- in general: `<type>* ptr;`
- Pointers can point to any type, *including pointers*

```
int* b;  
int a[10];
```

0x????:



0x1000:

b:

0x????

a:

a[1]

a[2]

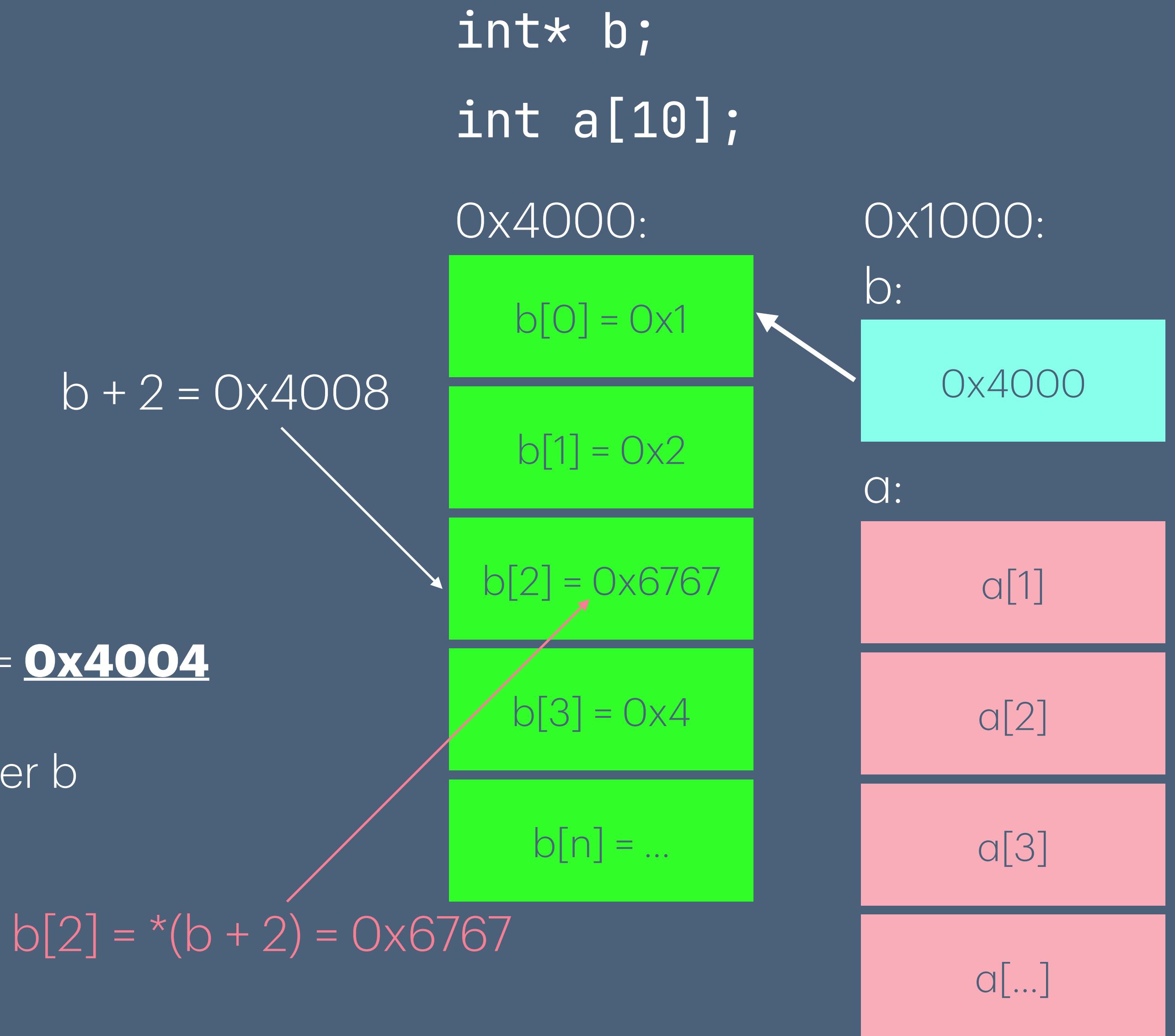
a[3]

a[...]

Pointer Review

Array-like; pointer arithmetic

- $*b$ gets the value a pointer points to
- Here, $*b \rightarrow 0x1$
- $b + 1$
 - Gets the address of "next" int after $*b$
 - $0x4000 + \text{sizeof(int)} * 1 = 0x4000 + 4 = \text{\textbf{0x4004}}$
 - $*(b + 1)$ gets the value of next int after b
 - Equivalent to **b[1]**
 - In general $*(b + n) = b[n]$



Approaches to counting memory reads

Q1: Static & Dynamic Arrays

- Translate C to ASM
 - Pros
 - Reliable
 - Good practice
 - Cons
 - Slower
 - ASM has to be correct
- Analyze the C code
 - Pros
 - Faster
 - Better understanding of C
 - Cons
 - Less flexible

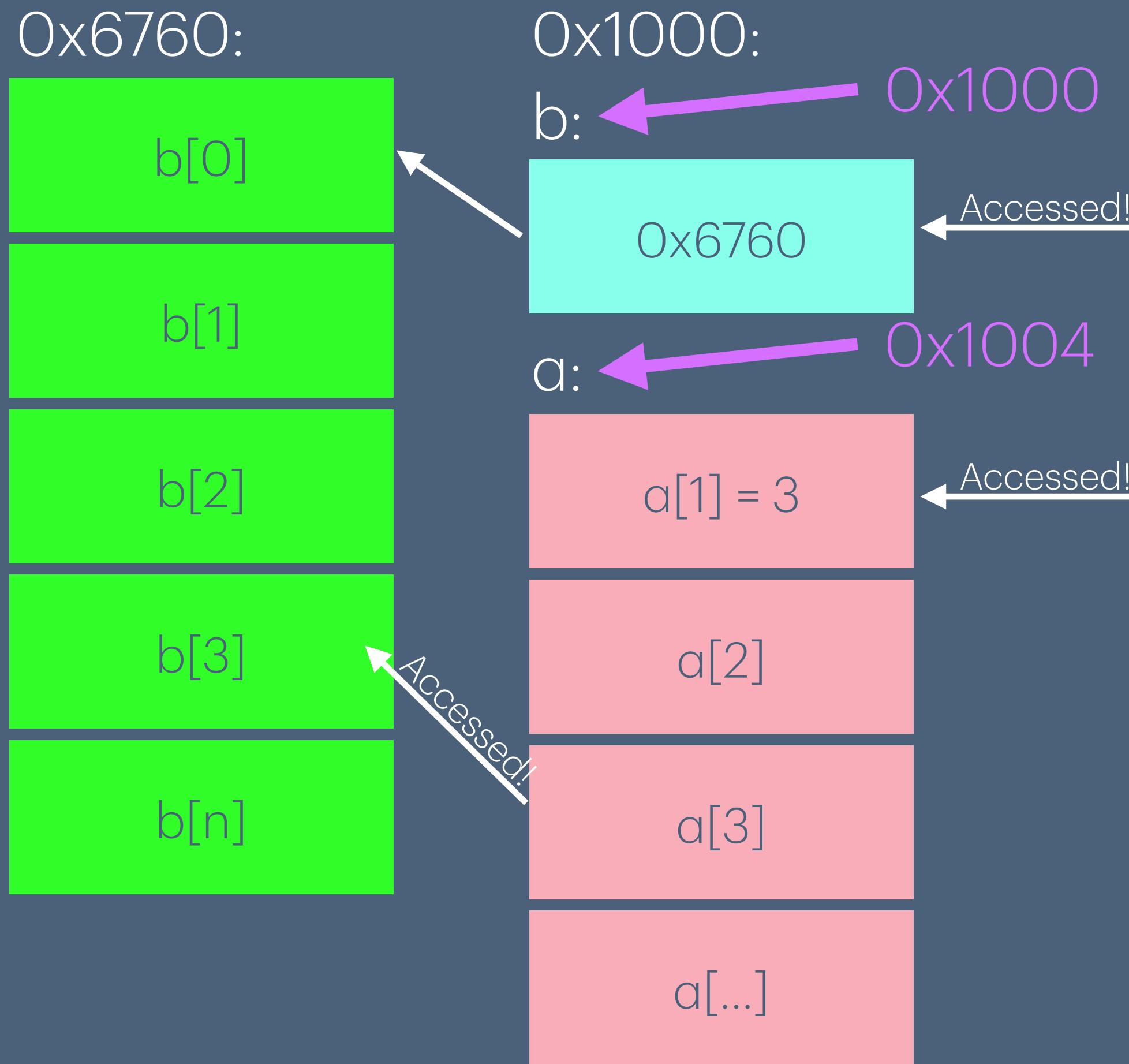
$$b[b[a[1]]] = b[1] + b[a[1]]$$

Dynamic vs Static Arrays

- Analysis approach
 - What do I need?
 - **Address** of RHS
 - **Value** of LHS
 - Values can be shared between RHS and LHS

$$b[b[a[1]]] = b[1] + b[a[1]]$$

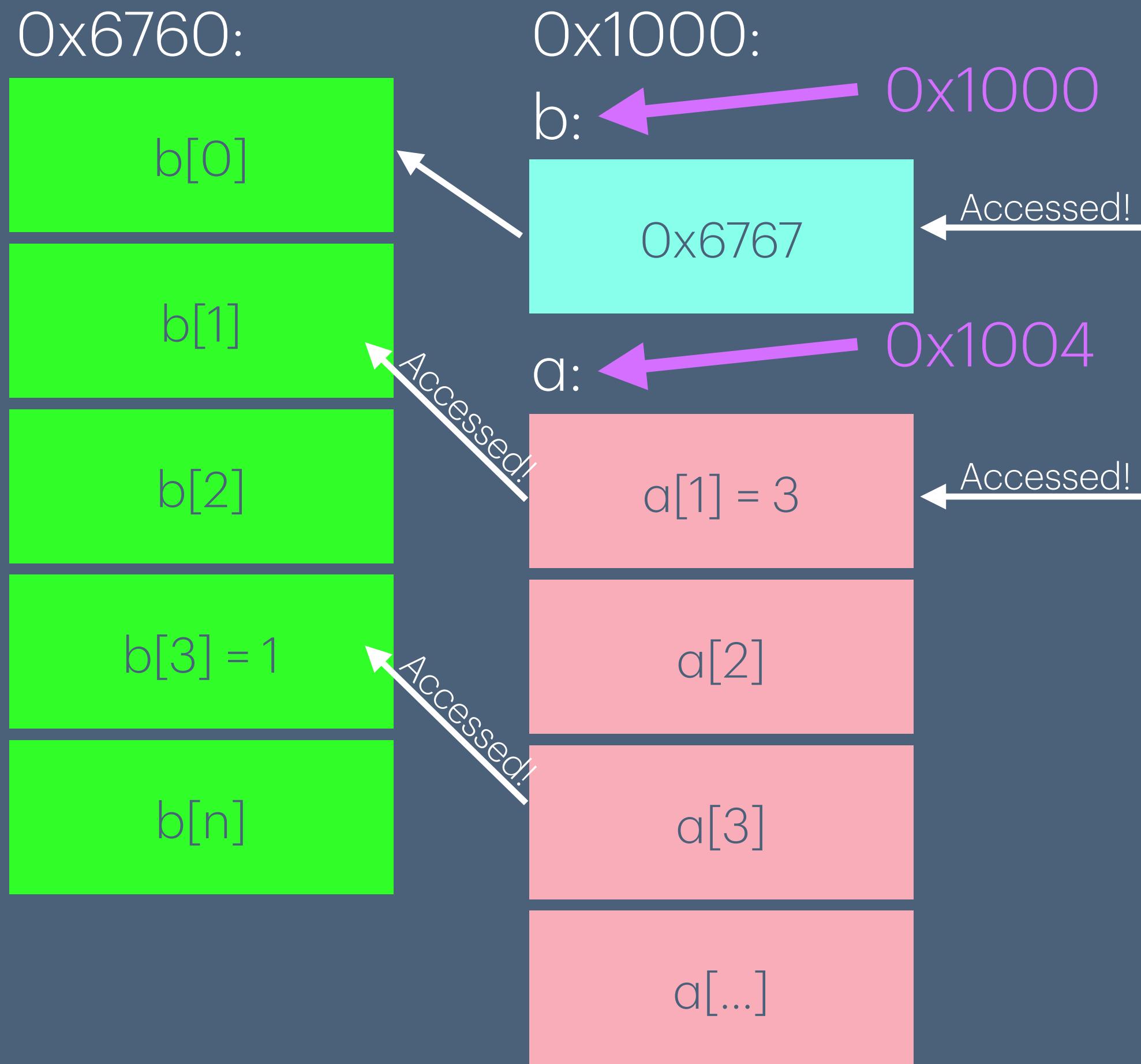
Known at compile time



- Address of $b[b[a[1]]]$
- Value of $b[a[1]]$
- Need $a[1]$
- $a[1]$ known at compile time
- 1 memory read to get $a[1]$
- Need value of b (where does b point to?)
- 1 memory read
- Finally, $b[a[1]]$, 1 more memory read

$$b[b[a[1]]] = b[1] + b[a[1]]$$

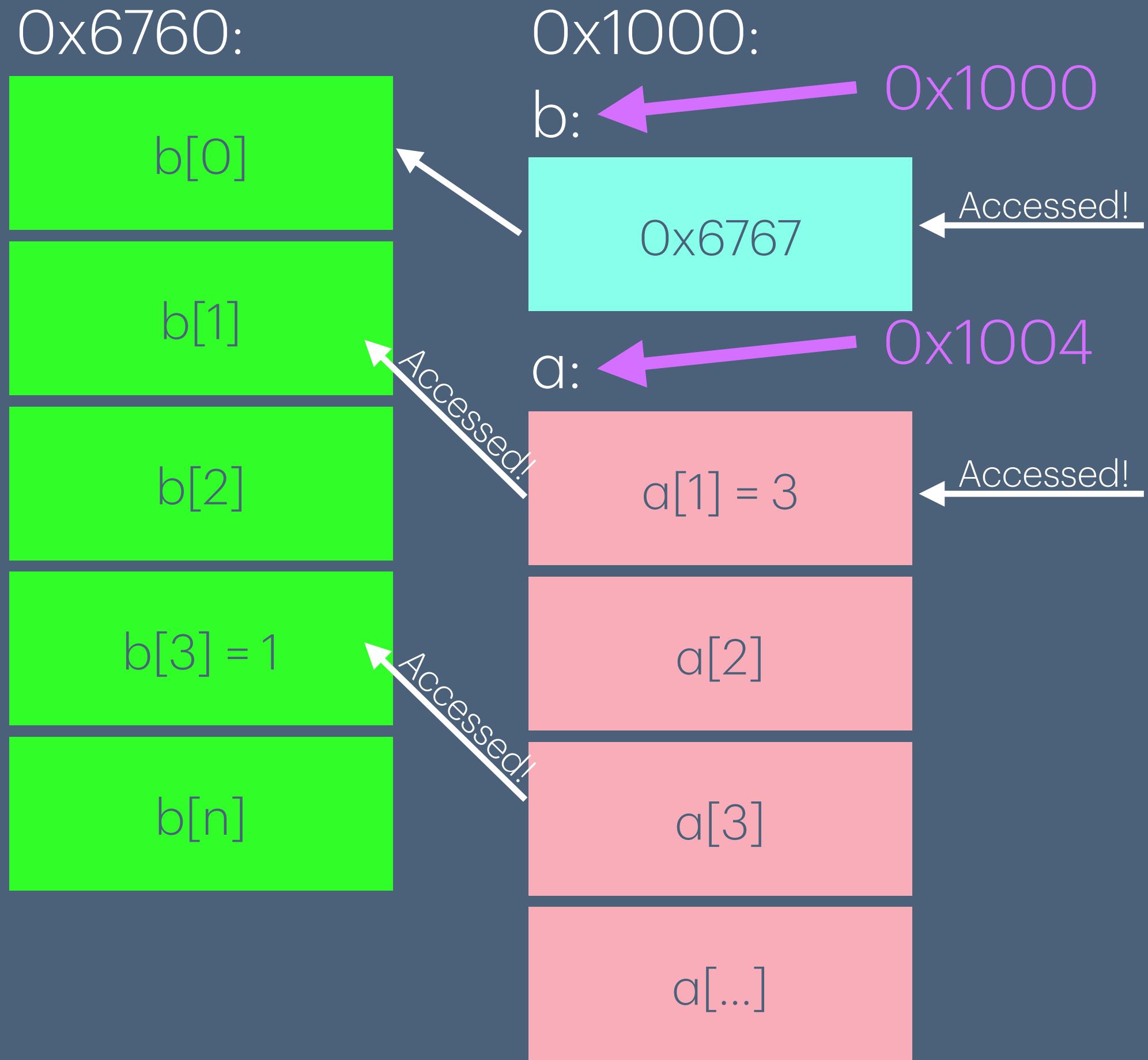
Known at compile time



- What we know from before
 - $a[1]$, $b[a[1]]$, b
 - Need to find where $b[b[a[1]]]$ is still
 - Easy, add b to $b[a[1]] * 4$, no memory needs needed. $b[b[a[1]]] = 0x6760 + 1 * 4 = 0x6764$.
- Calculate $b[1] + b[a[1]]$
- We need $b[1]$, which is 1 memory read

$$b[b[a[1]]] = b[1] + b[a[1]]$$

Known at compile time



- Total memory reads
 - 4
- Write assembly approach is probably much slower
- Count number of instructions in ISA that read from memory

Q2: Pointers

WTF?

```
arr[3][0] = arr[0][2 - arr[1][4]]
```

Q2: Pointers

An algorithm

- Apply this rule:
 - $\text{arr}[0][2 - \text{arr}[1][4]]$
 - $\text{arr}[0][2 - *(\text{arr}[1] + 4)]$
- $\underline{*}(a + i) = a[i]$
- Work from right to left
- Simplest first
- Let's start with RHS
 - $\text{arr}[0][2 - *(\text{arr}[1] + 4)]$
 - $\text{arr}[0][2 - *(*(\text{arr} + 1) + 4)]$

Q2: Pointers

RHS continued

- `arr[0][2 - *(*(arr + 1) + 4)]`
 - `*arr[0] + 2 - *(*(arr + 1) + 4)`
- `*(arr[0] + 2 - *(*(arr + 1) + 4))`
 - `*(*arr + 0) + 2 - *(*(arr + 1) + 4))`
 - `*(*arr + 2 - *(*(arr + 1) + 4))`
- DONE!

Q2: Pointers

LHS

- Our original expression: **arr[3][0] = arr[0][2 - arr[1][4]]**
- Convert **arr[3][0]**
 - **arr[3][0]**
 - ***(arr[3] + 0)**
 - ***(arr[3])**
 - ***(*(&arr + 3))**
 - ****(&arr + 3)**

Q2: Pointers

Grand finale

- $\ast\ast(\text{arr} + 3) = \ast(\ast\text{arr} + 2 - \ast(\ast(\text{arr} + 1) + 4))$
- Part 2?
 - Apply the algorithm in reverse