What is your favorite sitcom out of these options?

clicker.cs.illinois.edu

Q1

~Code~
340

CS 340

Wed craft night

Building Blocks 0b10
(Selection and Information Storage)

survey about exams/readings

optional

Updates

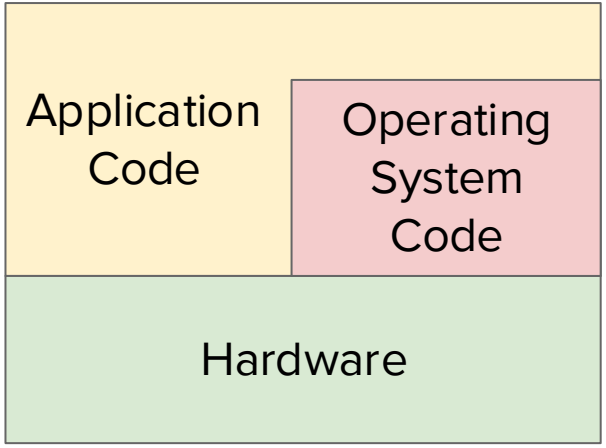
1. MP 2 - Linked-List in C due next Tuesday
2. If you added late, let us know ASAP for possible extensions and if you aren't on Campuswire
3. HW 2 out
   a. Collaboration time today! 2-4

Building Blocks 0b10

Today's LGs:

- See that you can use gates to build math and selection.
- Have a brief understanding of why we use base-2 for storage
- Be able to articulate that
  - Computers have different types of storage hardware
  - We utilize the different types of storage by using caching
  - Caching algorithms rely on spatial and temporal locality
- Be able to identify if code is cache friendly or not

Building Blocks Review



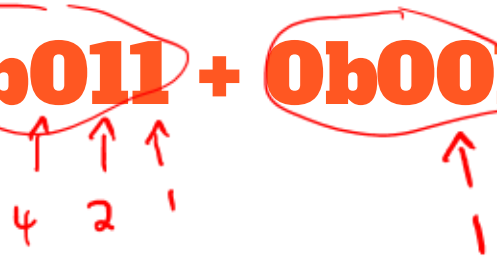
$$C = \neg(\neg(A \& B))$$

Building Blocks

1. Circuit and Gates
2. Binary
3. **Arithmetic Computations**
4. **Selection**
5. **Storage**

Arithmetic Calculations

What is 0b011 + 0b001 in decimal?



clicker.cs.illinois.edu

Q2

~Code~
340

A) 100
B) 3
C) 4
D) 8

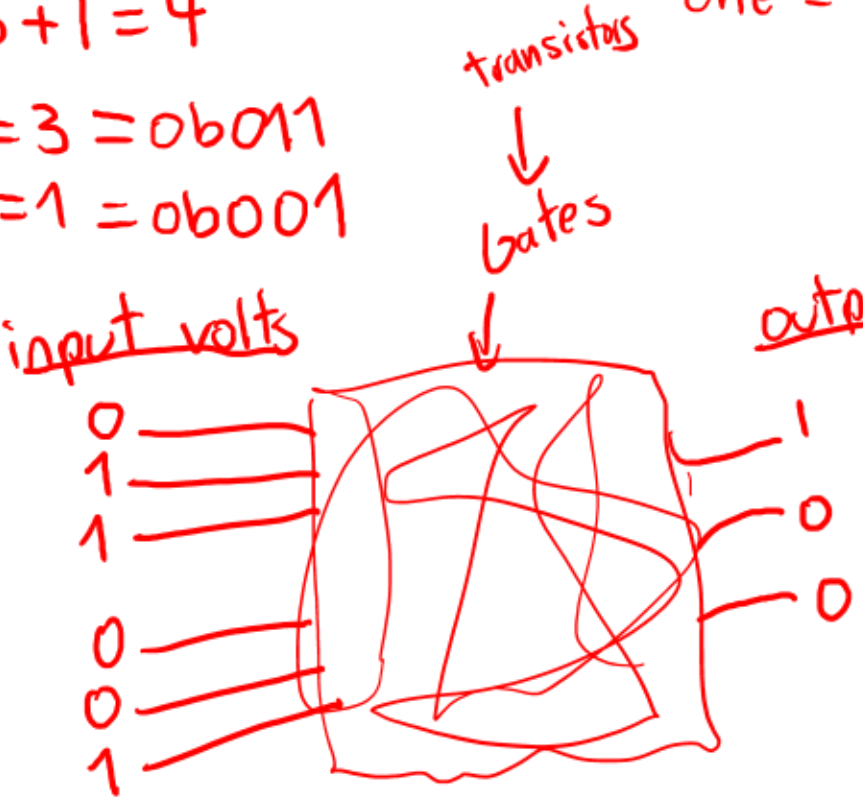
X + Y = Z in Logic

AND = & XOR = ^ or = | not = ~

3 + 1 = 4

x = 3 = 0b011

y = 1 = 0b001



$x = x_2 x_1 x_0 = 0\,1\,1$

$y = y_2 y_1 y_0 = 0\,0\,1$

011 + 001 = 100

$z = z_2 z_1 z_0 = ?$

$z_0 = x_0 \hat{} y_0 = 1 \hat{} 1 = 0$

$z_1 = c_1 \hat{} x_1 \hat{} y_1 = (1 \hat{} 1) \hat{} 0 = 0$

$c_1 = x_0 \& y_0 = 1 \& 1 = 1$

$c_2 = (x_1 \& y_1) | (c_1 \& (x_1 \hat{} y_1))$

$= 1$ (since $1 \hat{} 0$)

$z_2 = c_2 \hat{} x_2 \hat{} y_2 = (1 \hat{} 0) \hat{} 0 = 1$

Addition Formulas

$z_0 = x_0 \oplus y_0$

$c_1 = x_0 \& y_0$

$z_i = c_i \oplus x_i \oplus y_i$

$c_{i+1} = (x_i \& y_i) \mid (c_i \& (x_i \oplus y_i))$

Which diagram would build the logic in the bold box?

clicker.cs.illinois.edu

Q3

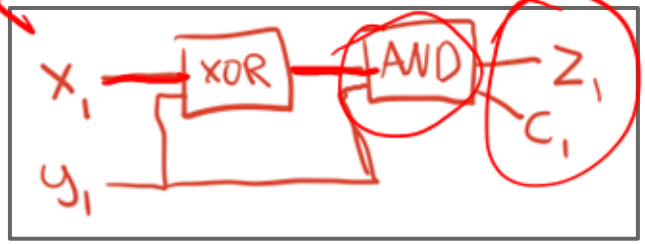
~Code~
340

$z_1 = (x_1 \hat{} y_1) \& y_1$

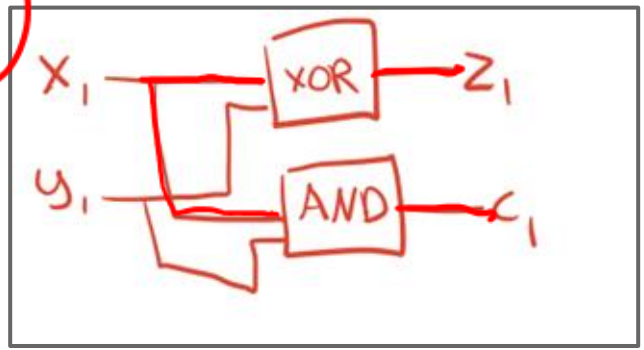
$z_1 = x_1 \hat{} y_1$

$c_1 = x_1 \& y_1$

B

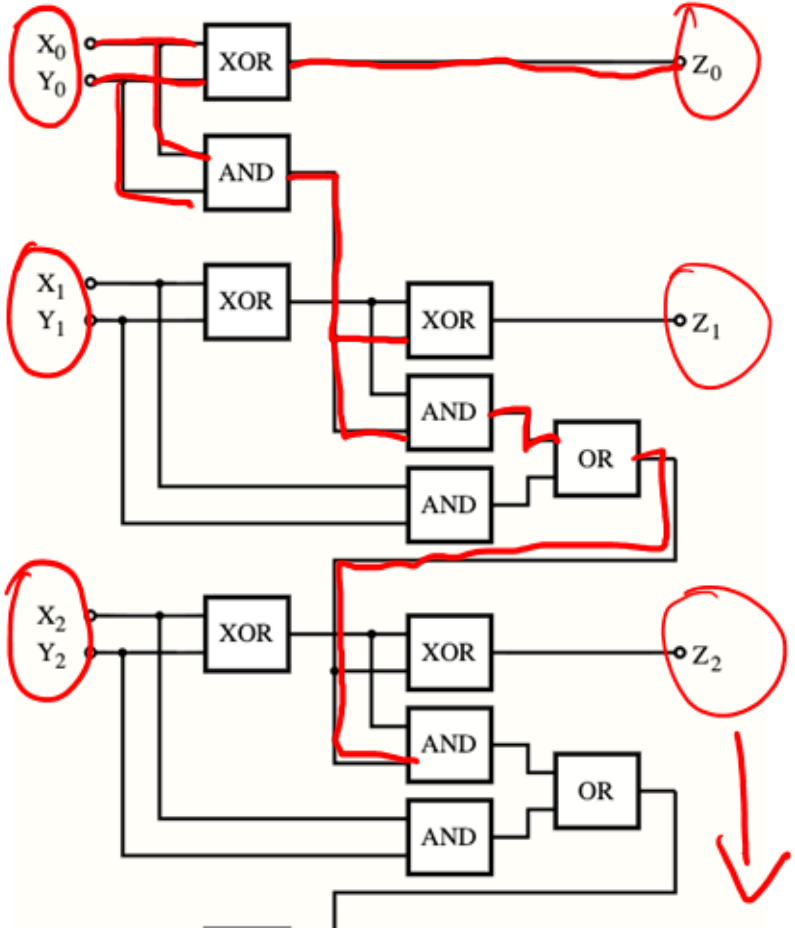


A



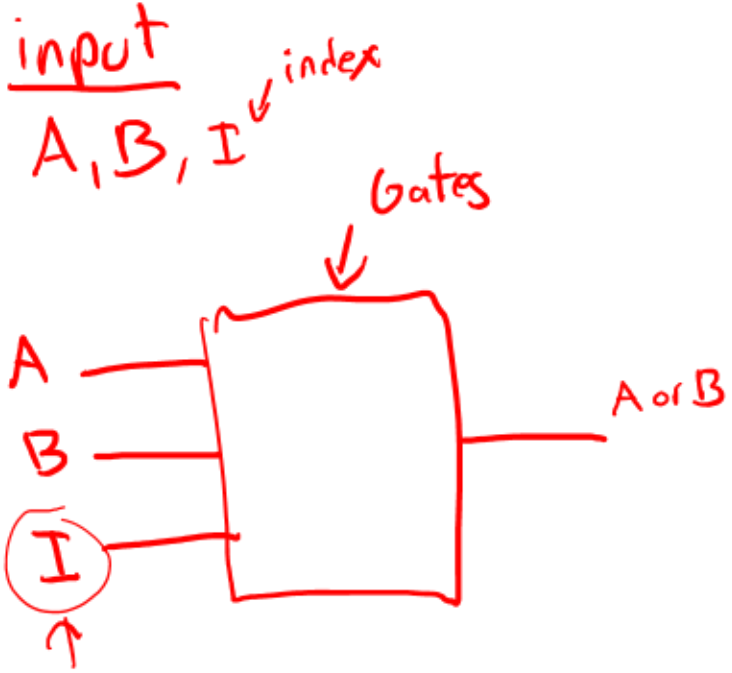
C

Ripple-carry adder circuit



Selection

MUX (sounds like ducks with an m)



logic

$r = (\neg I \,\&\, A) \mid (I \,\&\, B)$

$r = (1 \,\&\, 1) \mid (0 \,\&\, 0) = 1$

[1 0]
A B

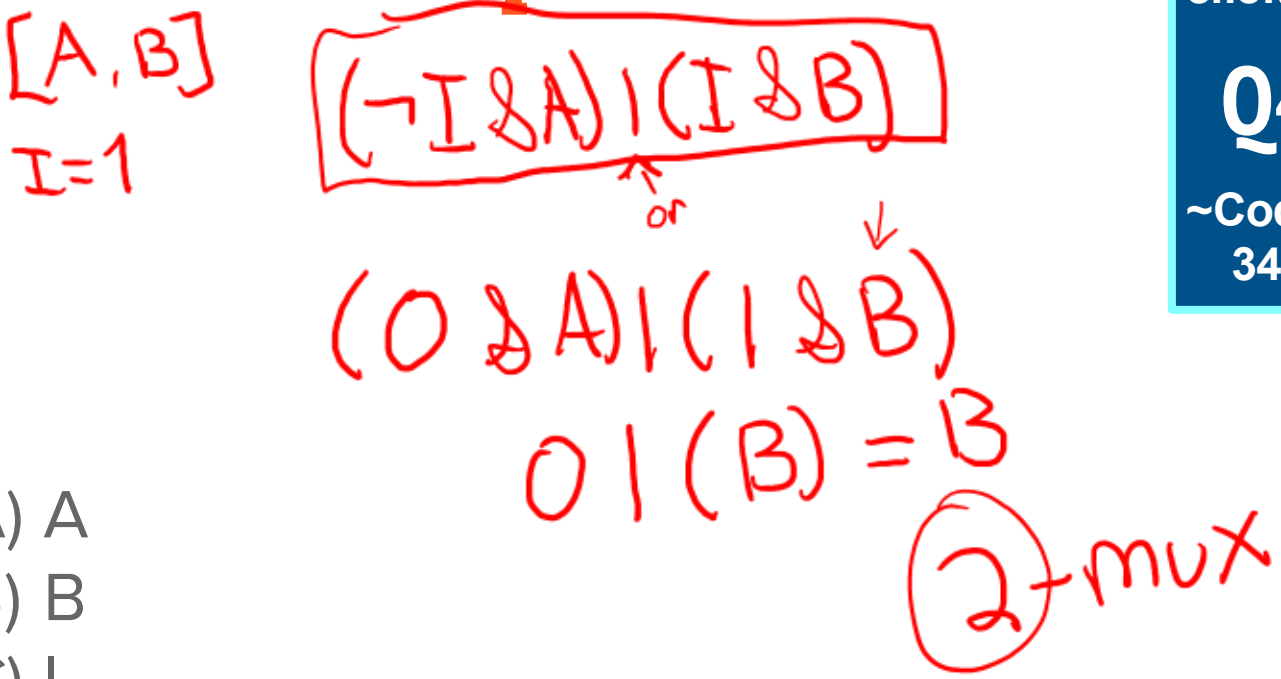
I=0

What is the output when I is 1?

clicker.cs.illinois.edu

Q4

~Code~
340



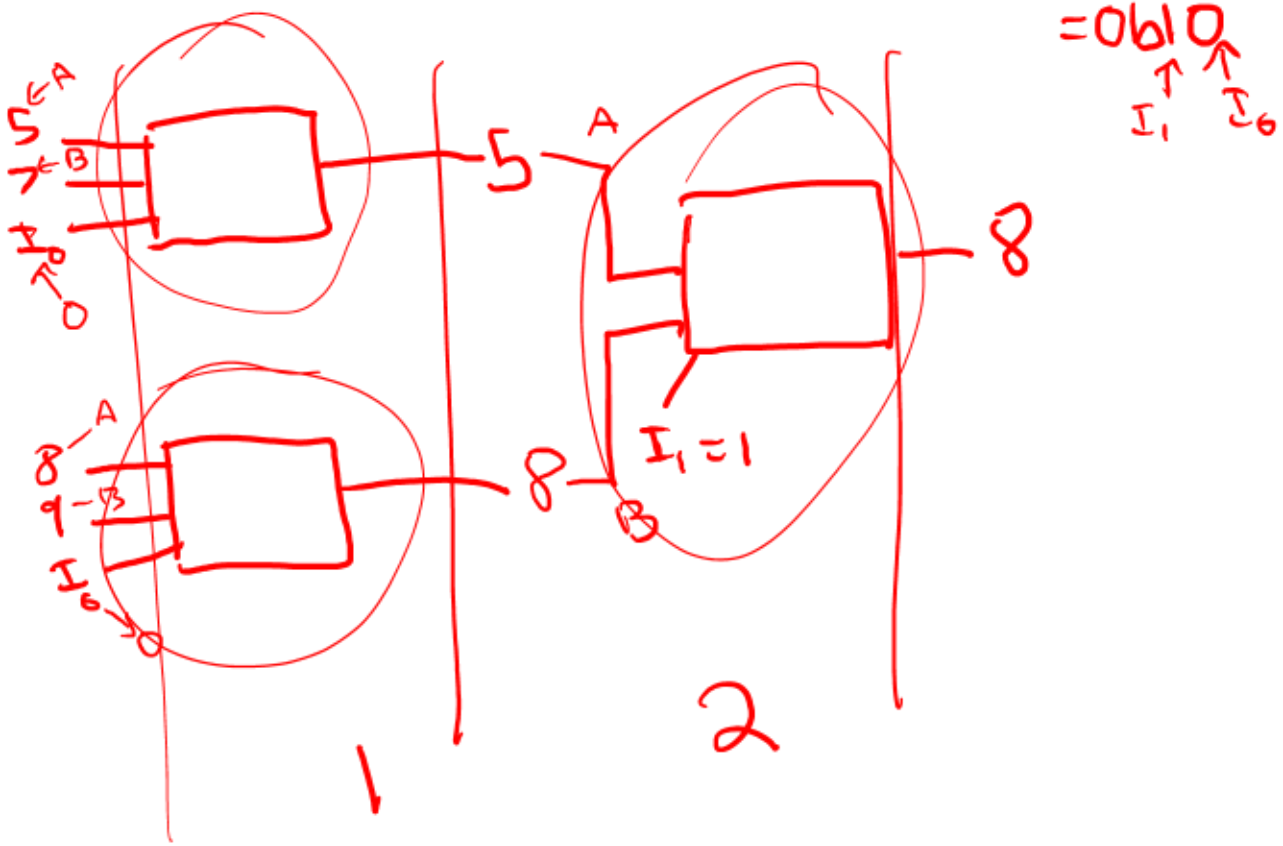
A) A
B) B
C) I

A MUX can scale!

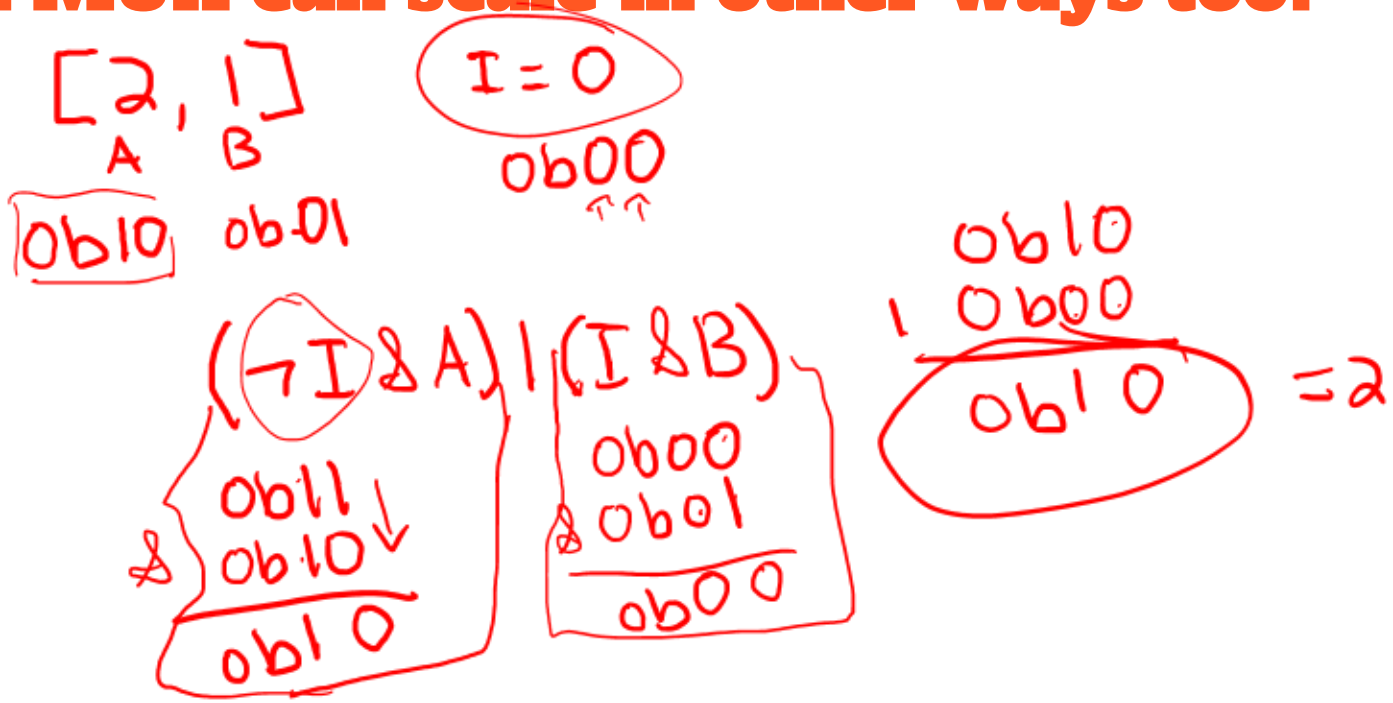
[5, 7, 8, 9]

I = 2

= 0 b 1 0



A MUX can scale in other ways too!



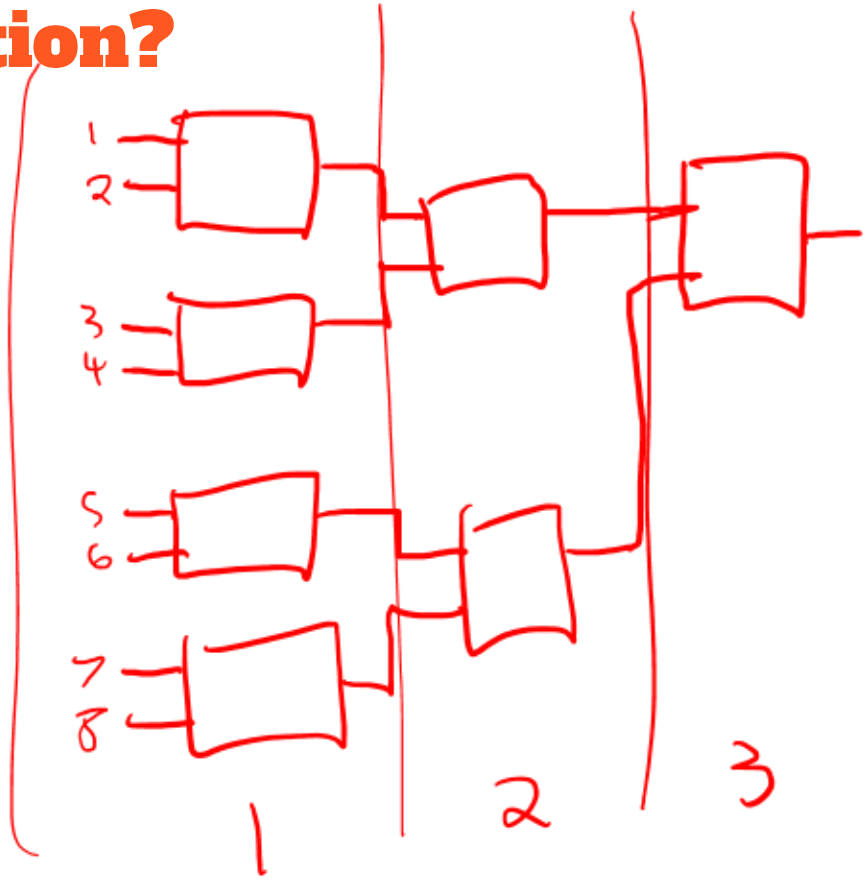
What depth of 2-MUX for a 8 selection?

clicker.cs.illinois.edu

Q5

~Code~
340

A) 1
B) 2
C) 3
D) 4
E) 5



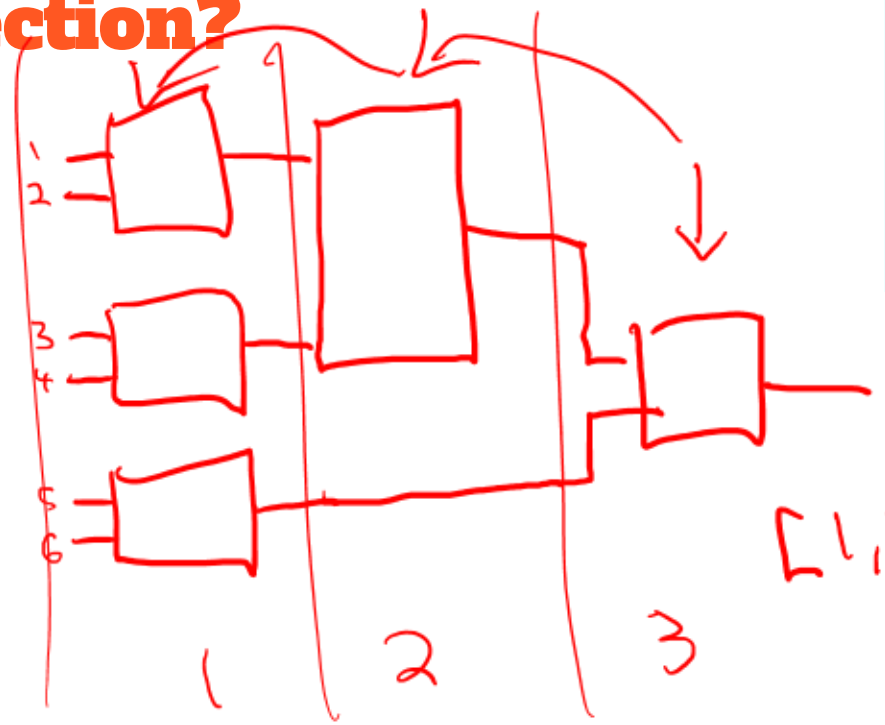
What min depth of 2-MUX's for a 6 input selection?

clicker.cs.illinois.edu

Q6

~Code~
340

A) 1
B) 2
C) 3
D) 4
E) 5



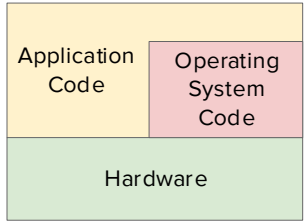
[1,2,3,4,5,6]

MUX Takeaways

more depth = slower

powers of 2 are used for data storage

Summary Slide



1. Circuits and Gates
2. Binary
3. Arithmetic Computations
4. Selection
5. **Storage**

Gates

Gates

Storage

Types of Storage

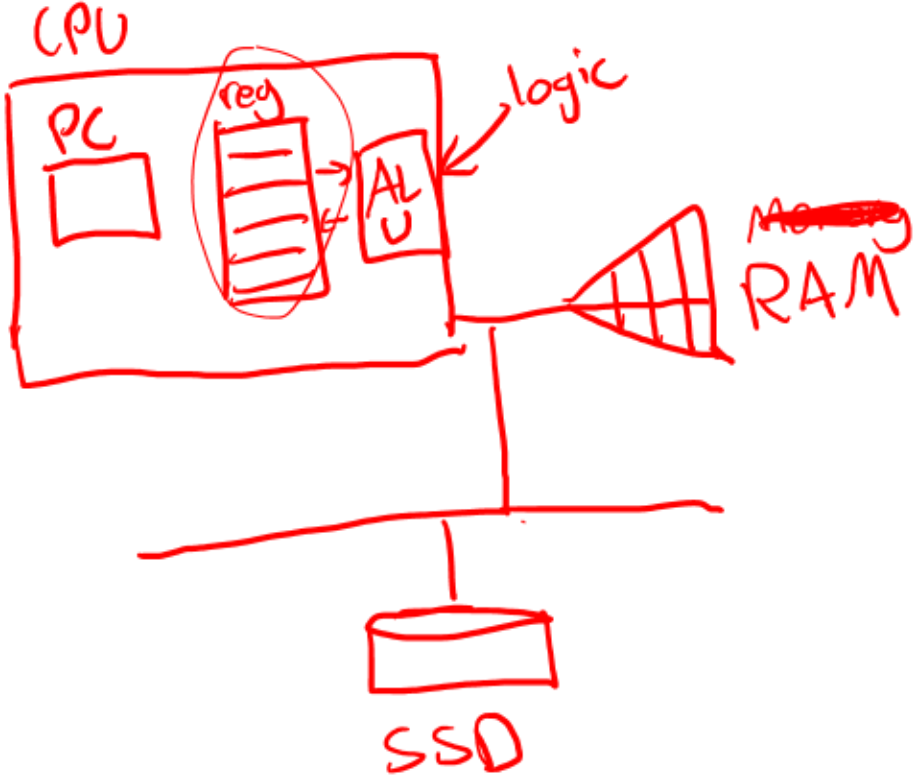
non-Volatile ← if it persists once power is off

Cheap, slow, non volatile
→ SSD

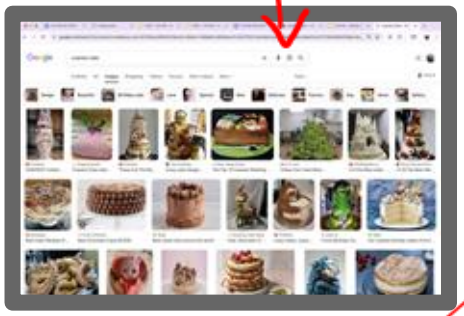
expensive, fast, volatile
flip-flop

↑ RAM
SRAM
DRAM

Hardware for Storing Information



Caching RAM



fast

* most likely to be used

slow

Caching - an algorithm for utilizing fast small memory and slow big memory.

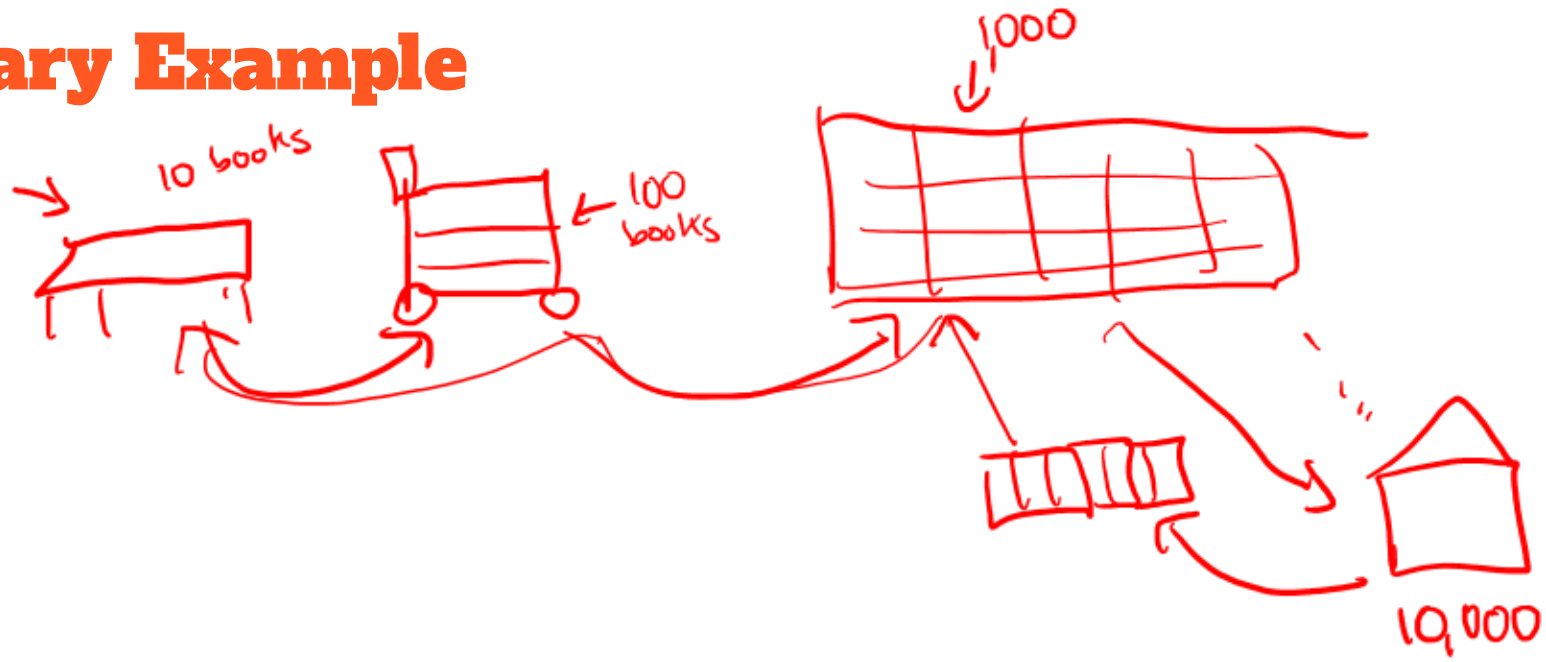
Locality - the idea that computers often use nearby and similar information sequentially.

how we guess what will be used next

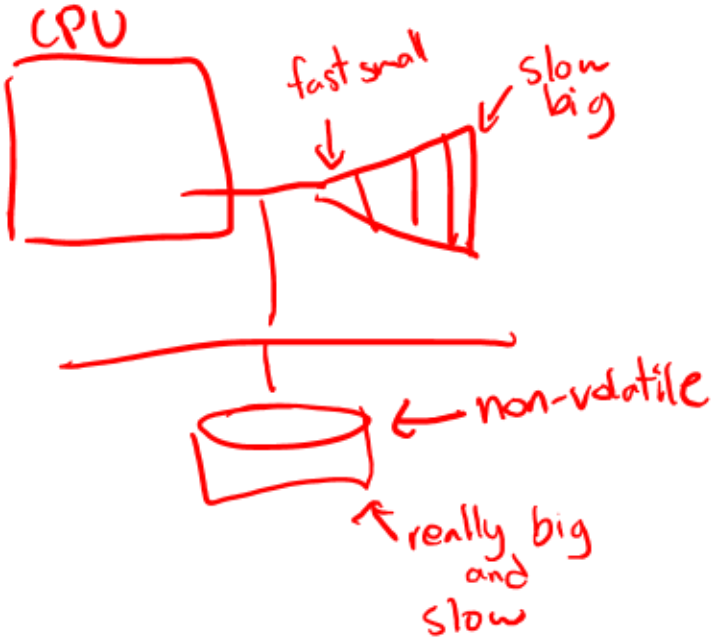
Spatial locality - things nearby

Temporal locality - things used recently

Library Example



Computer Information Storage



How can you change this code for better locality?

clicker.cs.illinois.edu

Q7

~Code~
340

```
6    int arr1[500];
7    int arr2[500];
8    //add stuff to arrays
9    int count = 0;
10   for(int i = 0; i < 500; i++){
11       if(arr1[i]%2 == 0) count++;
12       if(arr2[i]%2 == 0) count++;
13   }
```

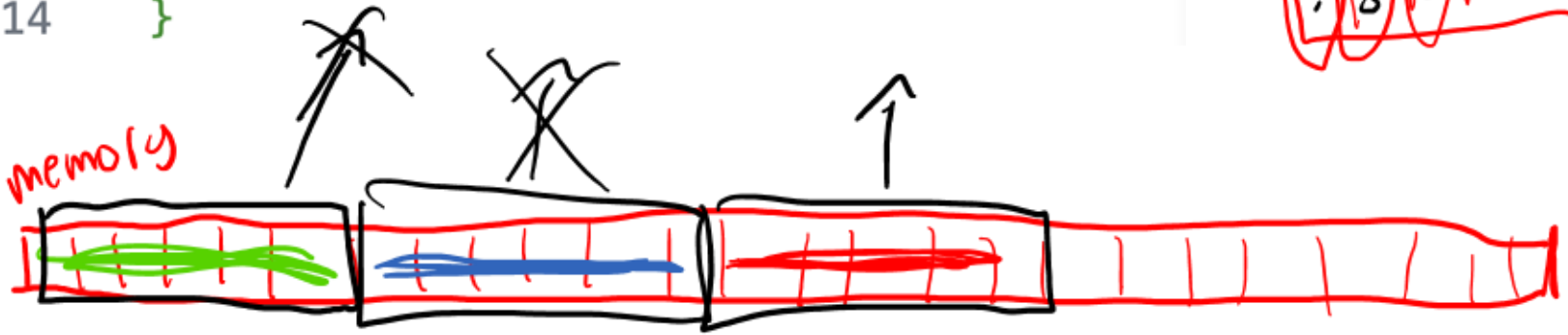
How can you change this code for better locality?

clicker.cs.illinois.edu

Q8

~Code~
340

```
 8    int doub[500][450];
 9    //add stuff to doub
10    for(int col = 0; col < 450; col++){
11        for(int row = 0; row < 500; row++){
12            doub[row][col]++;
13        }
14    }
```



Building Blocks

1. Circuit Basics
2. Gates
3. Binary
4. Arithmetic Computations
5. Selection
6. Storage

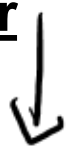
Reading from a file in C

FILE *fopen(const char *_pathname_, const char *_mode_);

size_t fread(void *_ptr_, size_t _size_, size_t _nmemb_, FILE *_stream_);

int fseek(FILE *_stream_, long _offset_, int _whence_);
SEEK_SET, **SEEK_END**, or **SEEK_CUR**

Casting a pointer



```
void * ptr;

int* ptr_i = (int *)ptr;
```