Fav sitcom out of Prof. Schatz's favorites

clicker.cs.illinois.edu

Q1

~Code~

340

CS 340

Building Blocks 0b10
(Selection and Information Storage)

Updates

1. MP 2 - due next Tuesday.
2. HW 3 due next Thursday 1:59pm
   a. Building Blocks
   b. If you need help with gates (see video posted on campus wire and the website)
3. Exam 1 - September 23rd
   a. Sign up now! By the 18th.

Tuesday - no class

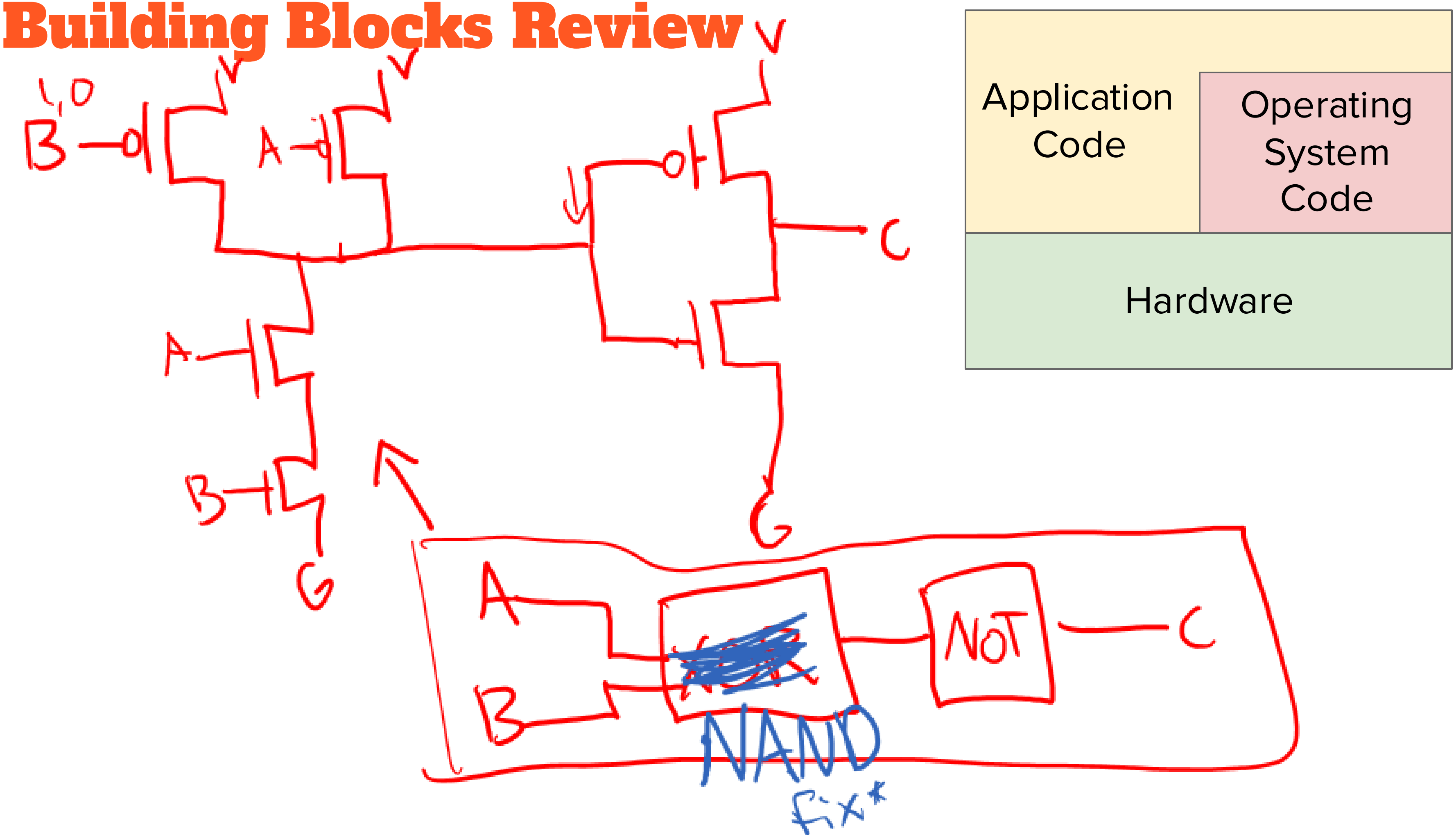
mp0 - mp2

Building Blocks 0b10

Today's LGs:

- Be convinced you can use gates to build calculations 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



Application Code

Operating System Code

Hardware

Building Blocks

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

Arithmetic Calculations

Arithmetic Calculations in Logic

$5+3=8 \Rightarrow 2$

$x=5=\underbrace{0101}_{x_3\, x_2\, x_1\, x_0}$

$y=3=\underbrace{0011}_{y_3\, y_2\, y_1\, y_0}$

$$\begin{array}{rcccc} & 1 & 1 & 1 & \\ & 0 & 1 & 0 & 1 \\ + & 0 & 0 & 1 & 1 \\ \hline & 1 & 0 & 0 & 0 \end{array}$$

$$\begin{array}{c|cccc} x & 0 & 1 & 0 & 1 \\ y & 0 & 0 & 1 & 1 \\ \hline z & 1 & 0 & 0 & 0 \\ & z_3 & z_2 & z_1 & z_0 \\ \hline c & 0 & 1 & 1 & 0 \\ & & & c_1 & c_0 \end{array}$$

$z_0 = x_0 \wedge y_0$

$z_0 = 1 \wedge 1$

$z_0 = 0$

$c_1 = x_0 \,\&\, y_0$

$c_1 = 1 \,\&\, 1 = 1$

$z_i = c_i \wedge x_i \wedge y_i$

$c_{i+1} = (x_i \,\&\, y_i) \,|\, (c_i \,\&\, (x_i \wedge y_i))$

Arithmetic Calculations in Logic

$Z_0 = x_0 \hat{}\, y_0$

$C_1 = x_0 \& y_0$

$Z_i = C_i \hat{}\, x_i \hat{}\, y_i$

$C_{i+1} = (x_i \& y_i) \mid (c_i \& (x_i \hat{}\, y_i))$

Arithmetic Calculations in Hardware

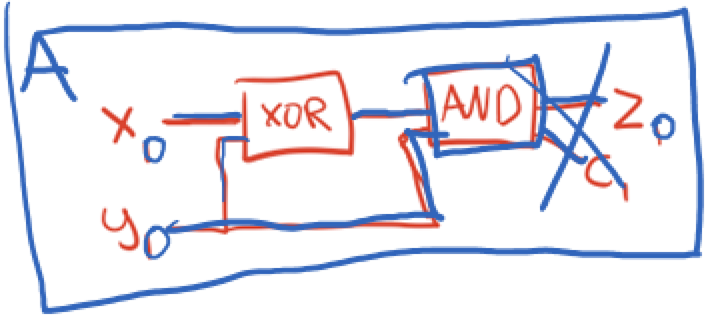
$Z_0 = x_0 \hat{} y_0$

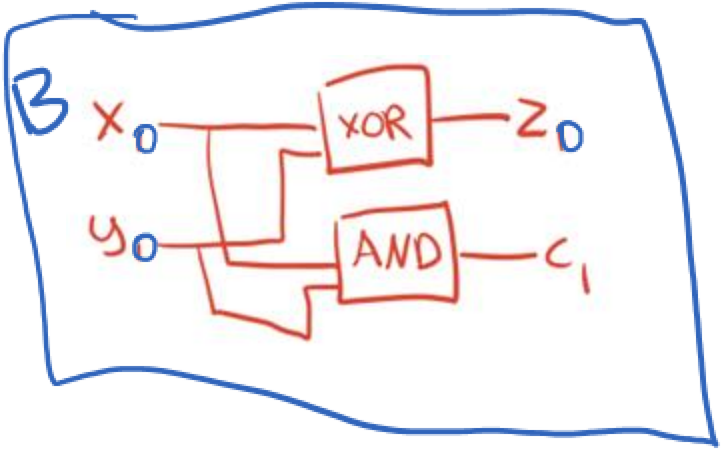
$C_1 = x_0 \,\&\, y_0$

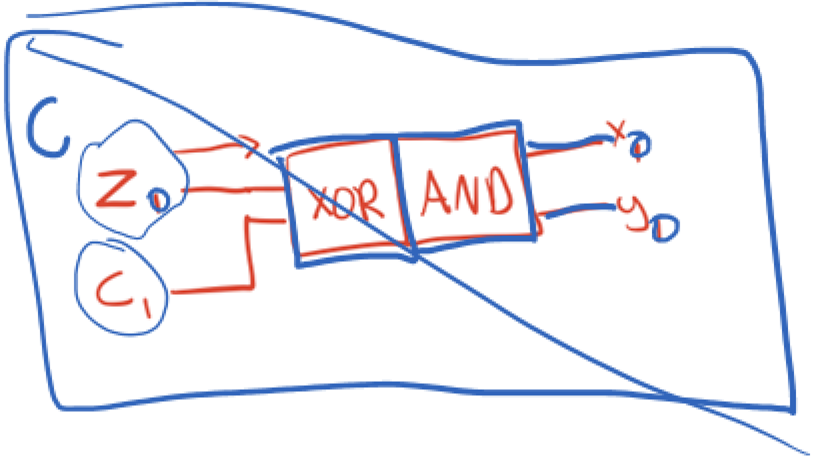
clicker.cs.illinois.edu

Q2

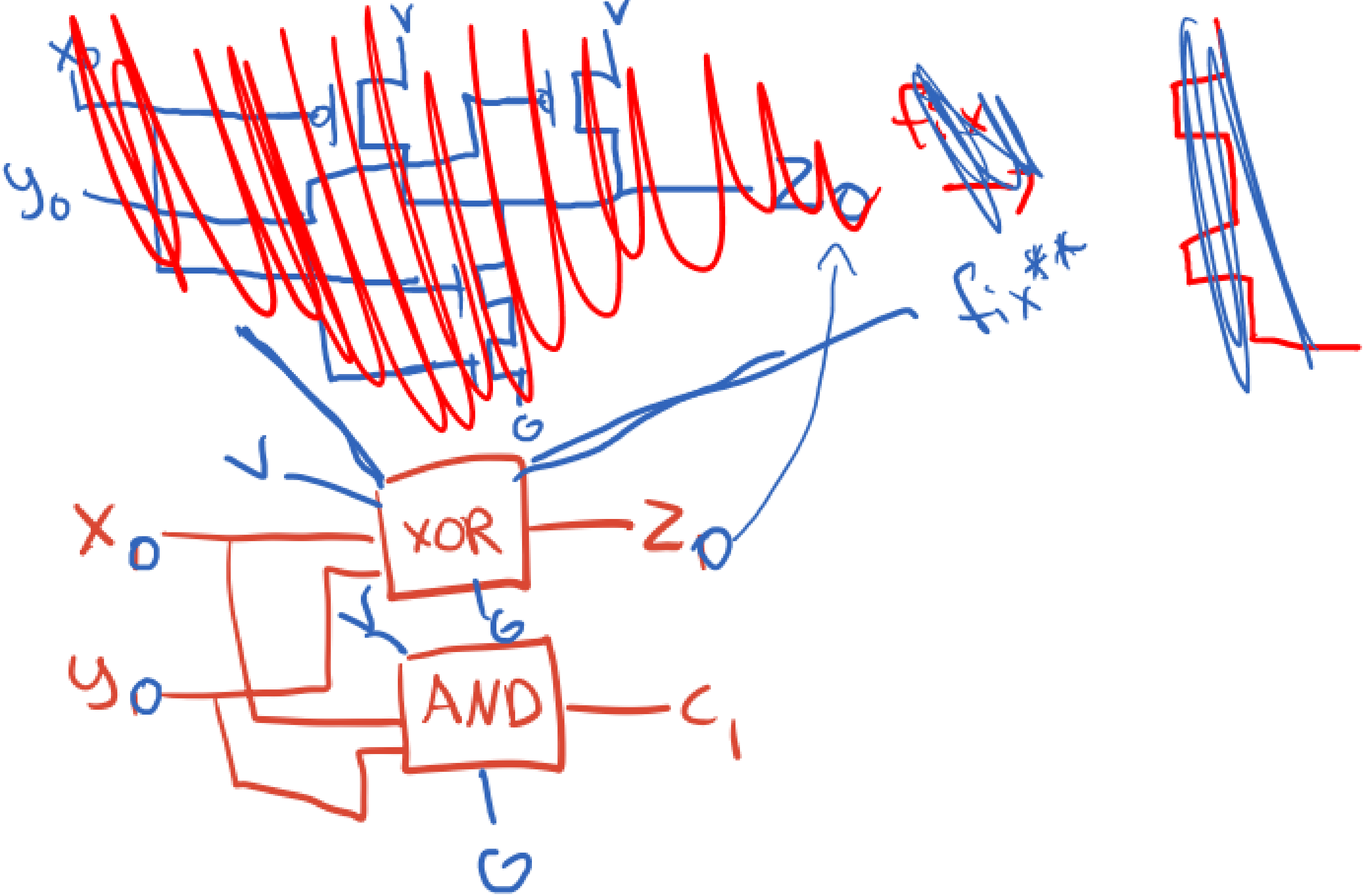
~Code~
340







Arithmetic Calculations in Hardware



What is Z?

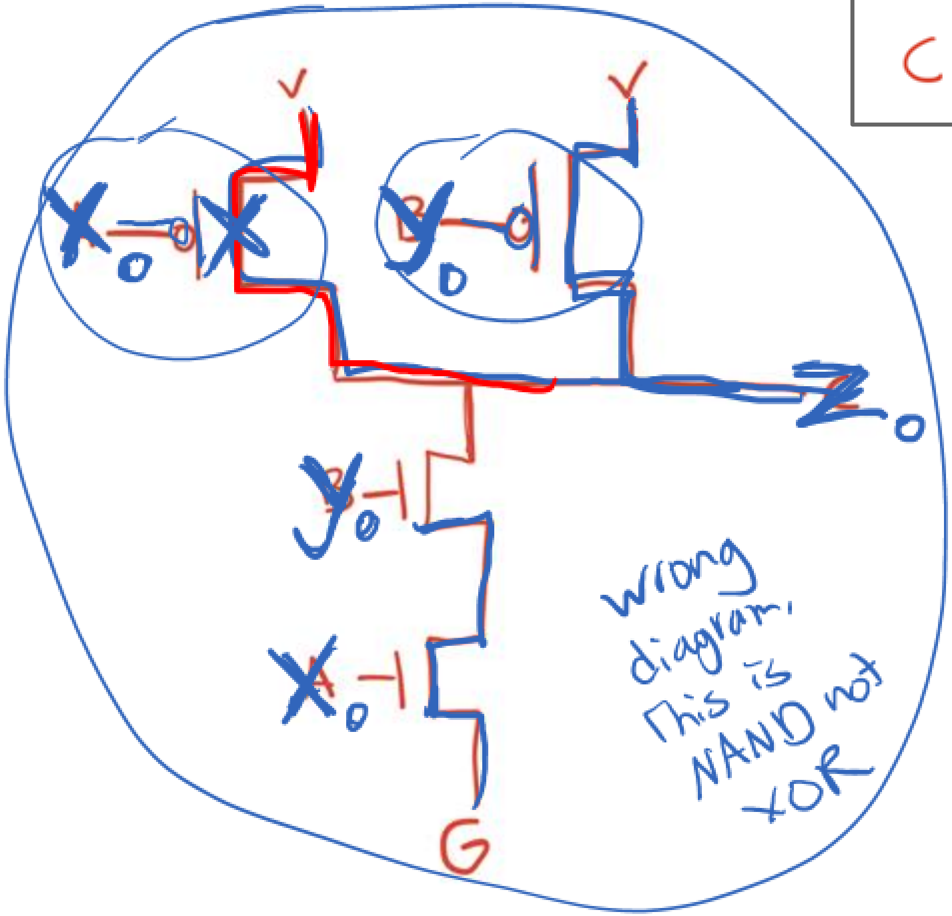
$Z_0 = x_0 \wedge y_0$

$C_1 = x_0 \& y_0$

clicker.cs.illinois.edu

Q3

~Code~
340



wrong diagram, this is NAND not XOR

$x_0 \quad y_0$

$\underline{1+0} = Z_0 ?$

$0+0 = \boxed{1}$

Selection

MUX (sounds like ducks with an m)

$[\overset{0}{\underset{0}{a}}, \overset{1}{\underset{1}{b}}]$ $s = {}^{0 \text{ or } 1}$

$$(\neg s \,\&\, a) \,||\, (\overset{1\,\&\,1}{s \,\&\, B})$$

if $s = 1$ $0 \,\&\, a$

$(0) \,||\, 1$

$= 1, b$

What is the output?

[a, b]
0 1

S = 0 or 1

(¬s & a) | (s & B) =

S = 0

[1, 1]
A B
0 1

(1 & 0) | (0 & 1)
0 1 0 = 0

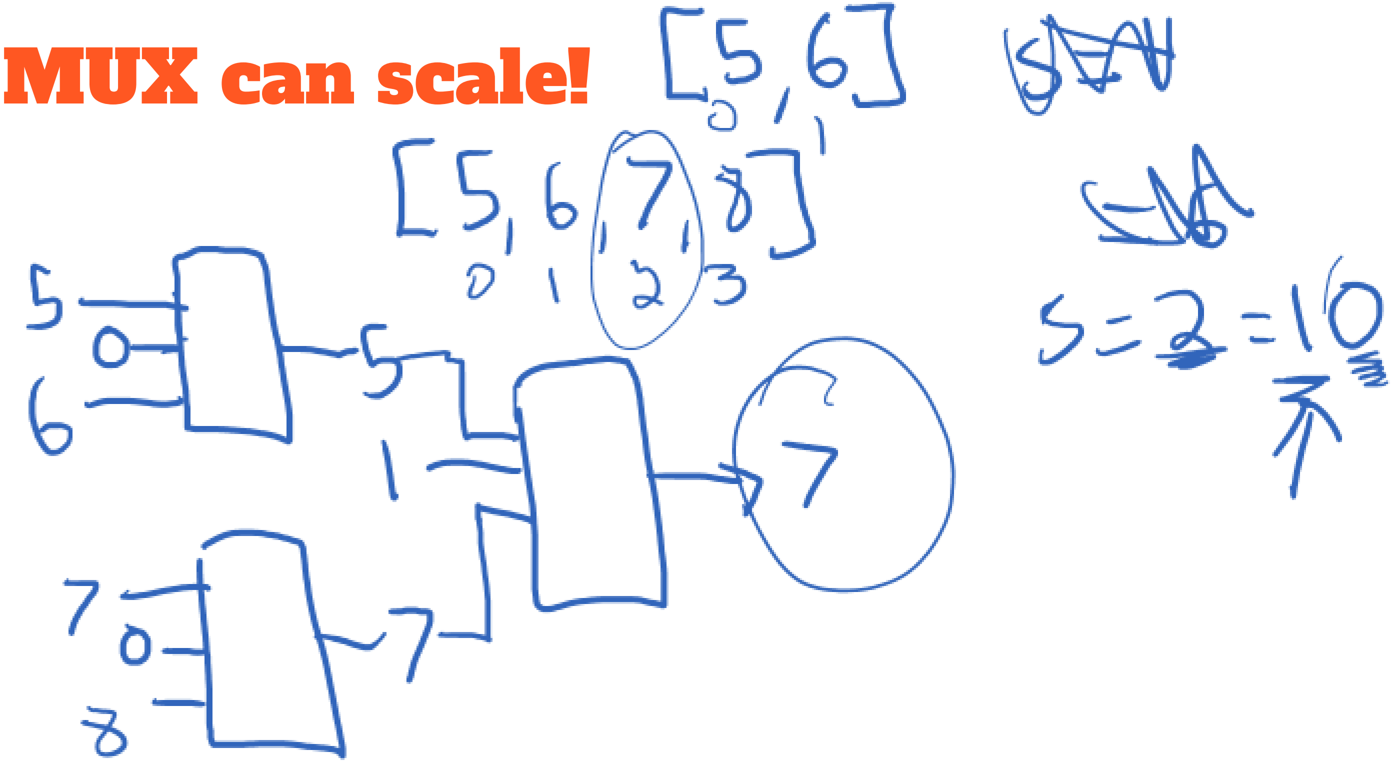
(1 & 1) | (0 & 1)
1 1 0 = 1

clicker.cs.illinois.edu

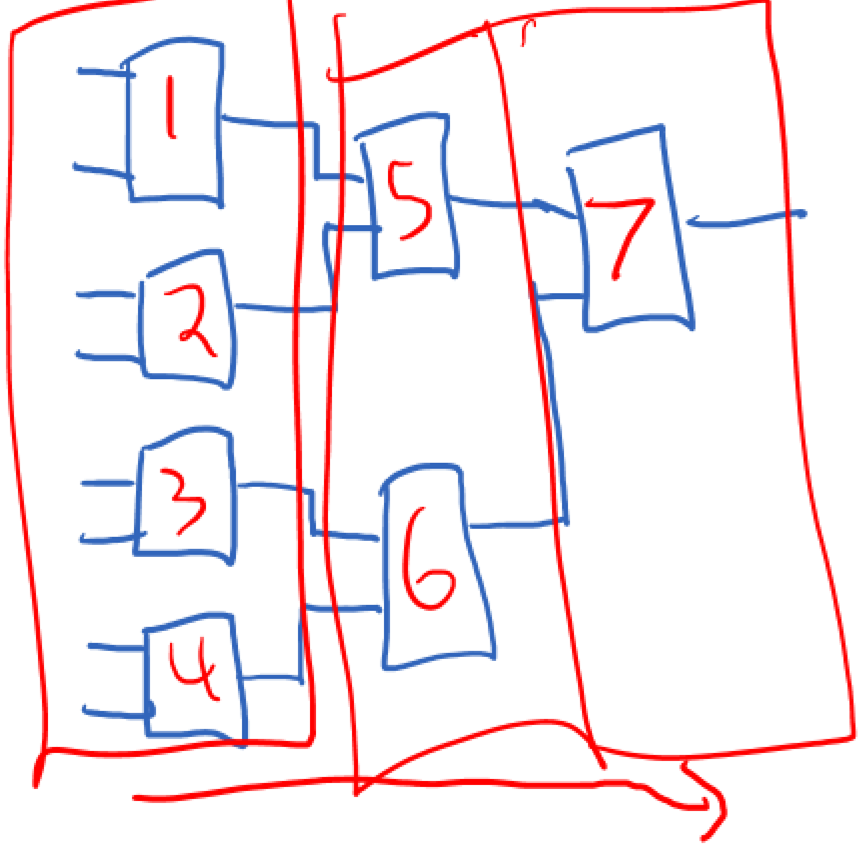
Q4

~Code~
340

MUX can scale!



How many 2-MUX for a 8 selection?



7

depth = 3

clicker.cs.illinois.edu

Q5

~Code~
340

What depth of 2-MUX's for a 16 input selection?

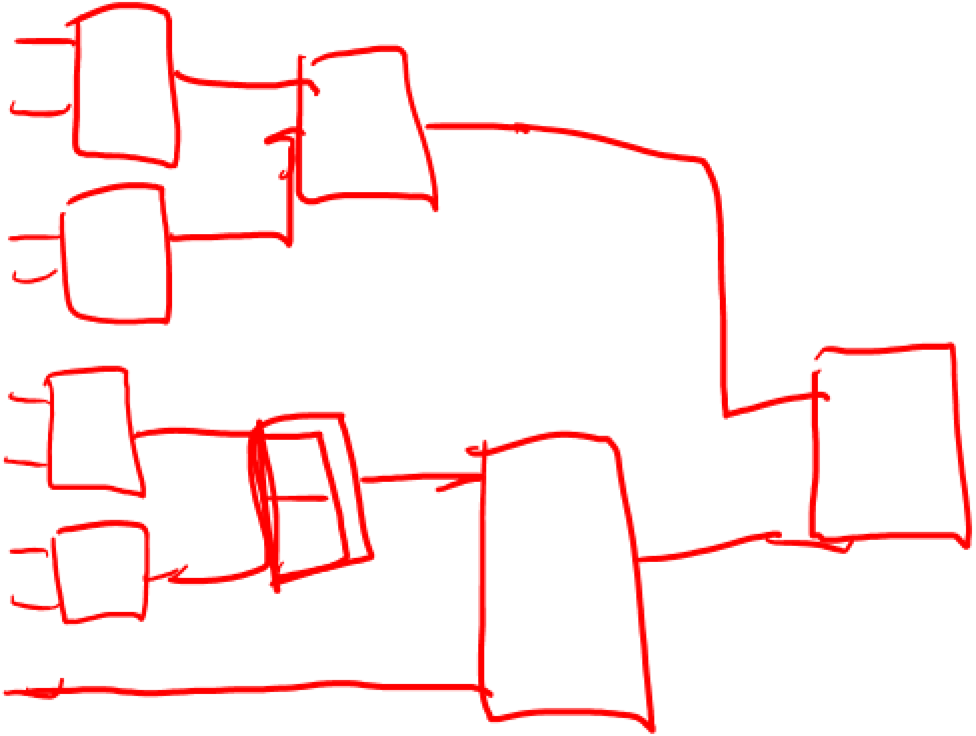
$\log_2 16 = 4$

clicker.cs.illinois.edu

Q6

~Code~
340

What depth of 2-MUX's for a 9 input selection?



clicker.cs.illinois.edu

Q7

~Code~
340

4

MUX Takeaways

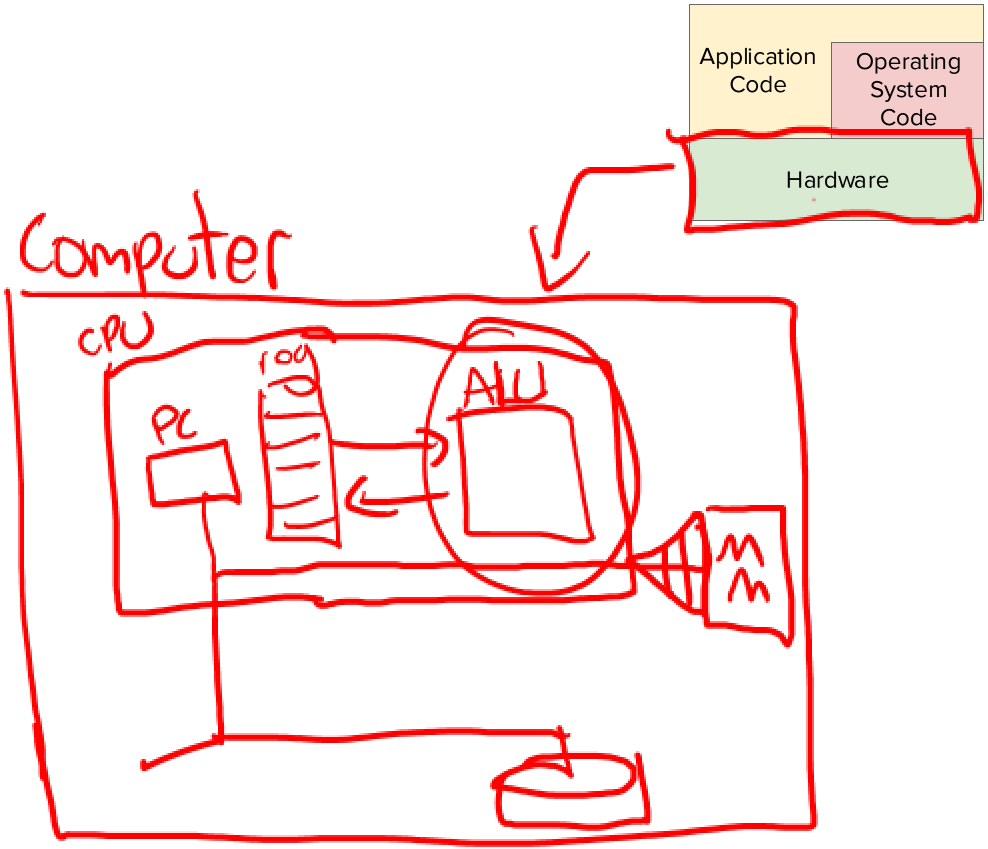
selects 1 from many

more depth = bad

slower

Summary Slide

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



Application Code

Operating System Code

Hardware

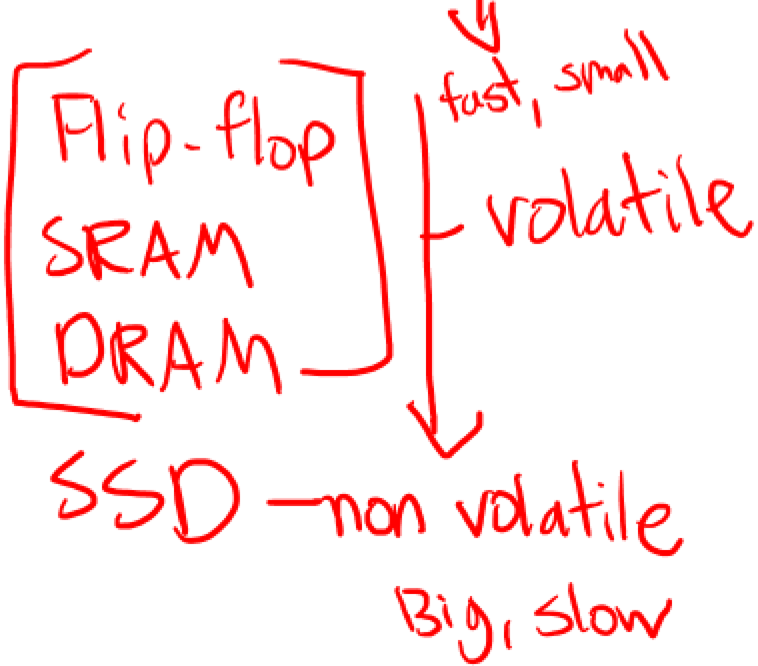
Storage

Hardware for Storing Information

store 1 or 0

volatile - goes away when off

non volatile → stay when off



fast, small

Flip-flop

SRAM

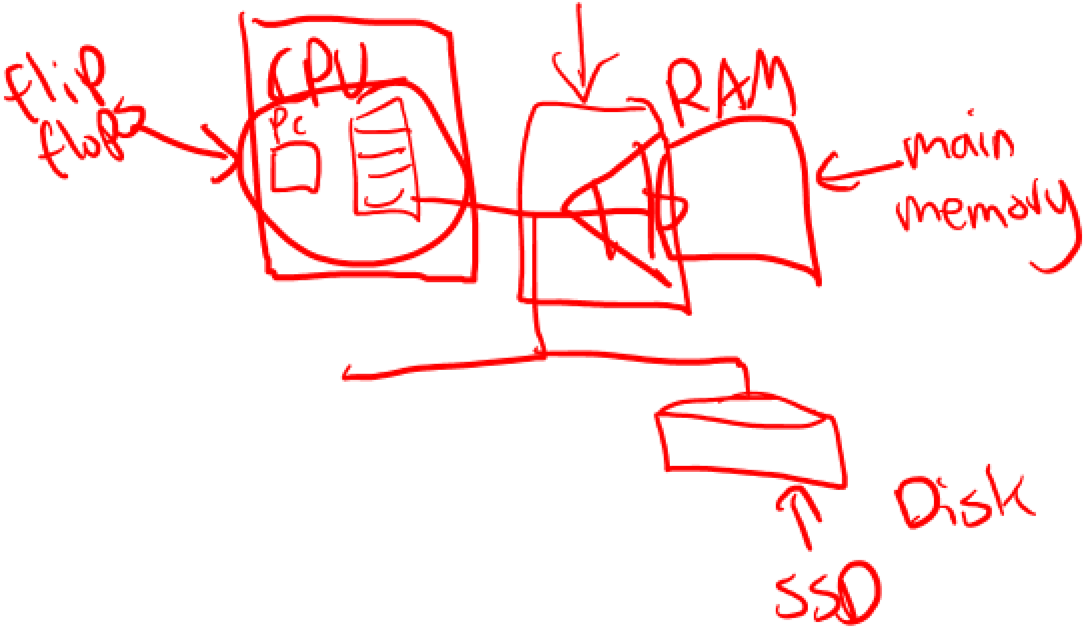
DRAM

volatile

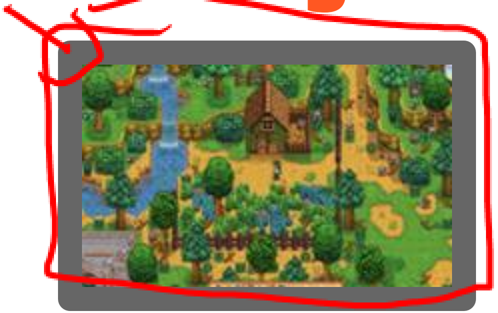
SSD - non volatile

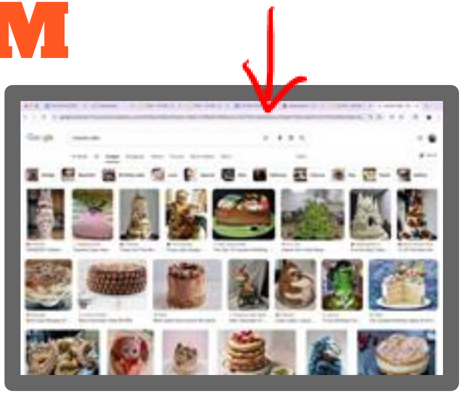
Big, slow

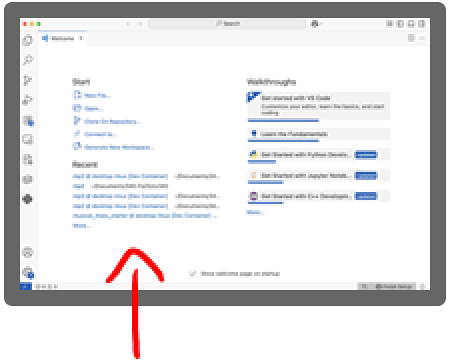
Hardware for Storing Information



Caching RAM







Search

Welcome

Start

New File...

Open...

Clone Git Repository...

Connect to...

Generate New Workspace...

Recent

More...

Walkthroughs

Get started with VS Code

Customize your editor, learn the basics, and start coding

Learn the Fundamentals

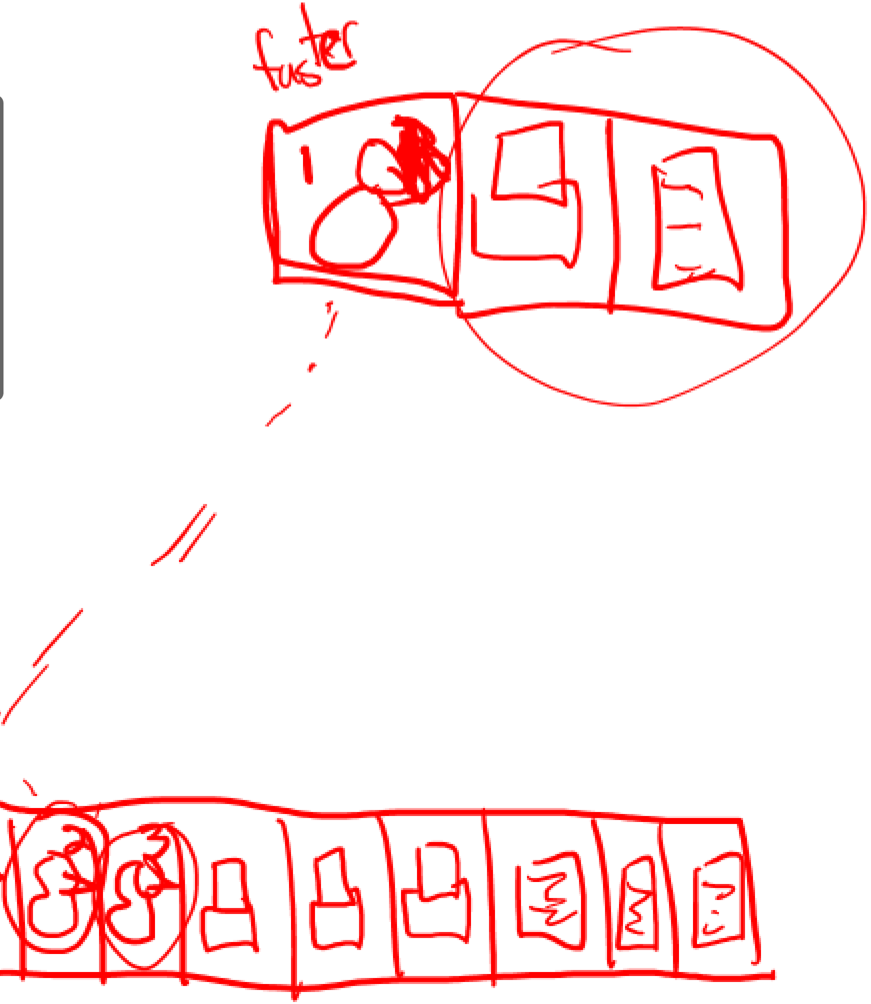
Get Started with Python Develo...

Get Started with Jupyter Noteb...

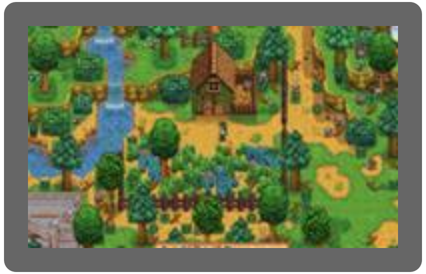
Get Started with C++ Developm...

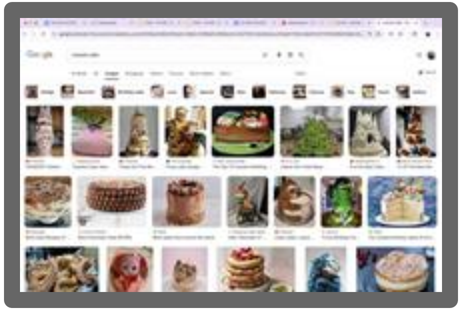
More...

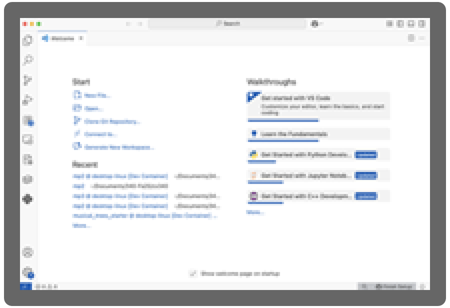
Show welcome page on startup



Caching RAM






Start
New File...
Open...
Clone Git Repository...
Connect to...
Generate New Workspace...
Recent
More...
Walkthroughs
Get started with VS Code
Learn the Fundamentals
Get Started with Python Develo...
Get Started with Jupyter Noteb...
Get Started with C++ Developm...
More...
Show welcome page on startup

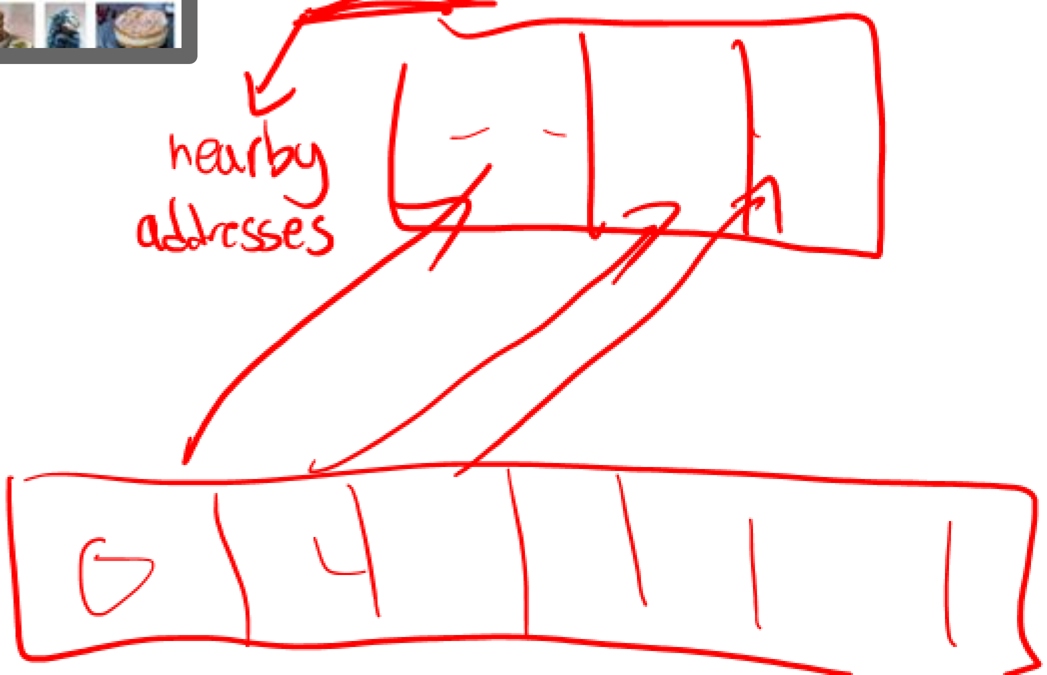
what is most likely

Locality

reuse things

temporal ← time

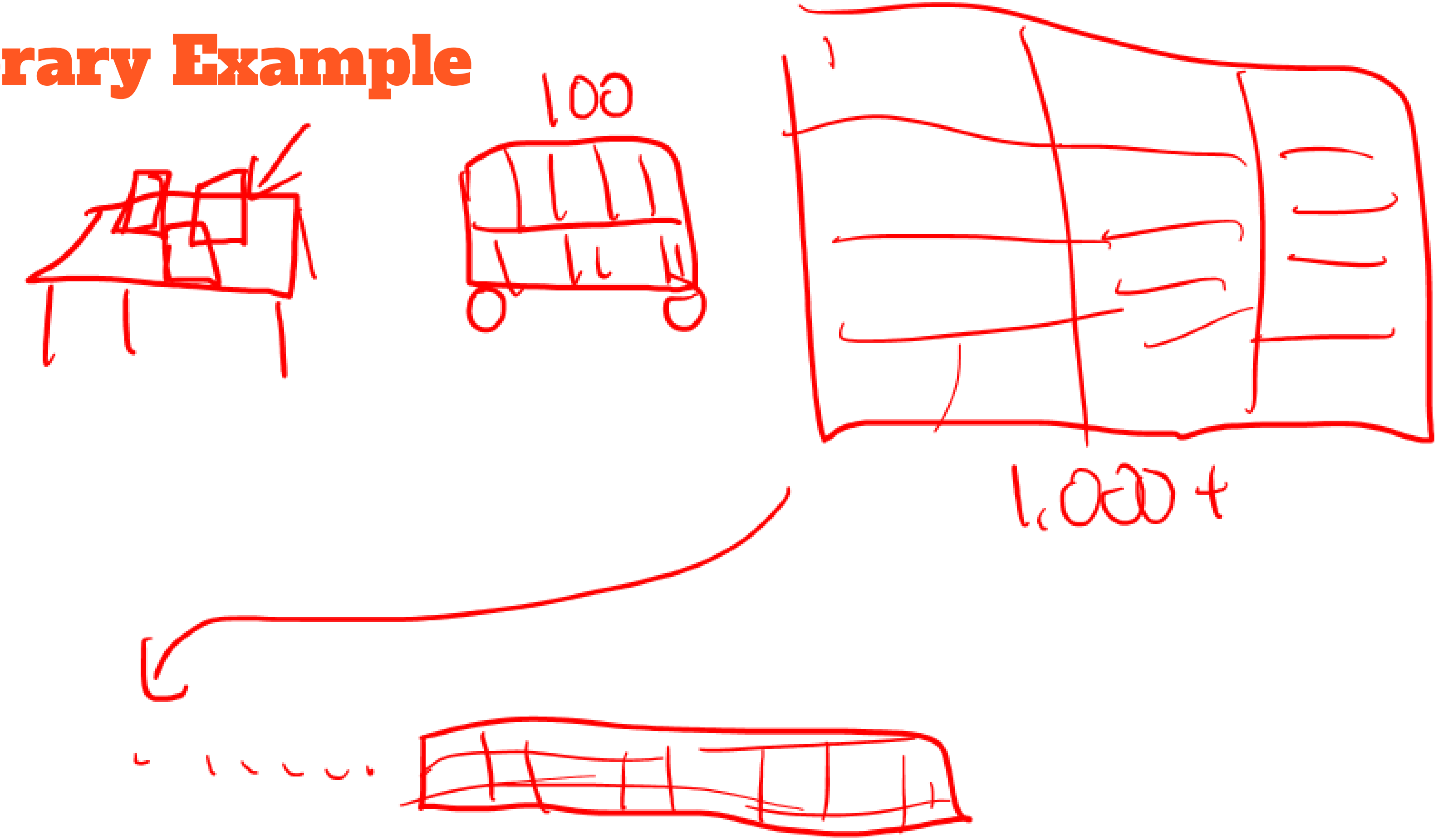
spatial ← space



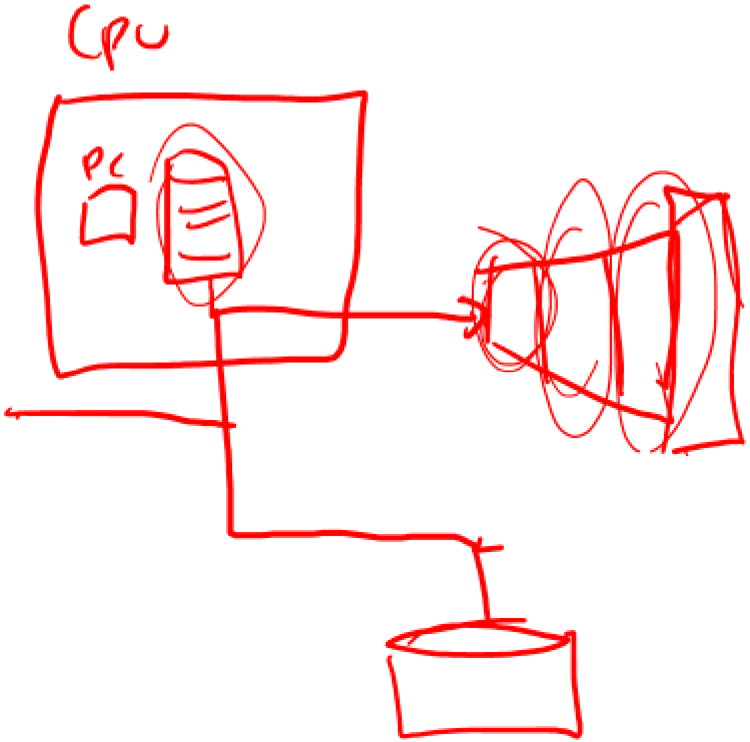
Caching - an algorithm for utilizing fast and small memory and ~~small~~ slow big memory. We keep copies in higher levels for quick access

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

Library Example



Computer Information Storage

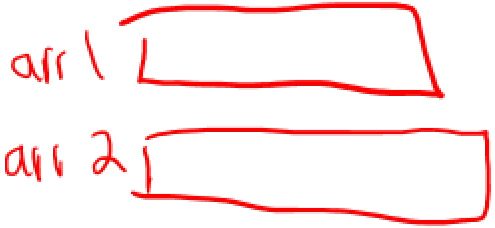


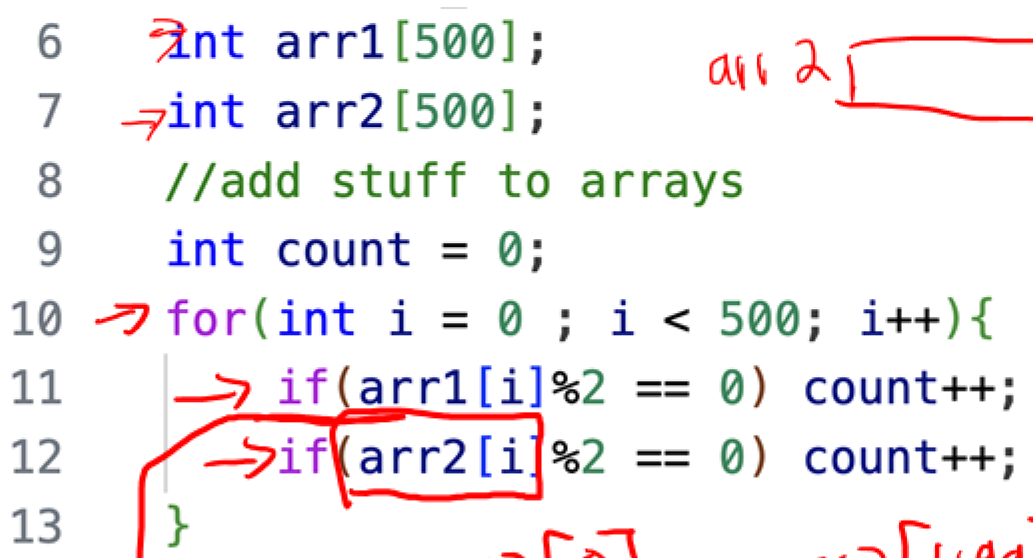
Can you change this code for better locality?

clicker.cs.illinois.edu

Q8

~Code~
340





```
int arr1[500];
int arr2[500];
//add stuff to arrays
int count = 0;
for(int i = 0 ; i < 500; i++){
    if(arr1[i]%2 == 0) count++;
    if(arr2[i]%2 == 0) count++;
}
```

arr2
~~arr~~
~~arr2~~
~~arr1~~
500

for

arr2[0] arr2[499]

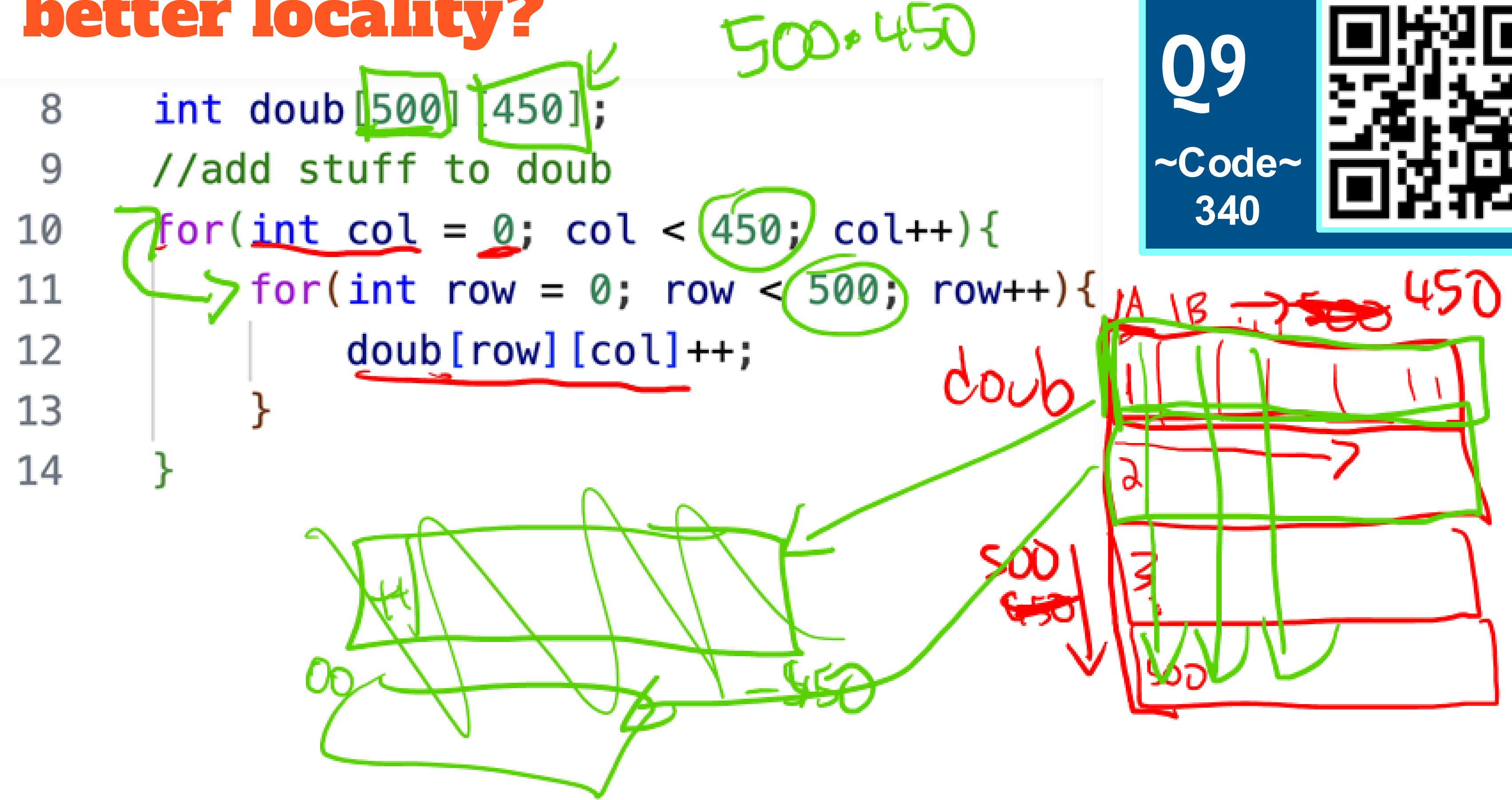
*(arr2+0) *(arr2+499)

Can you change this code for better locality?

clicker.cs.illinois.edu

Q9

~Code~
340



```
int doub[500][450];
//add stuff to doub
for(int col = 0; col < 450; col++){
    for(int row = 0; row < 500; row++){
        doub[row][col]++;
    }
}
```

Building Blocks

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