

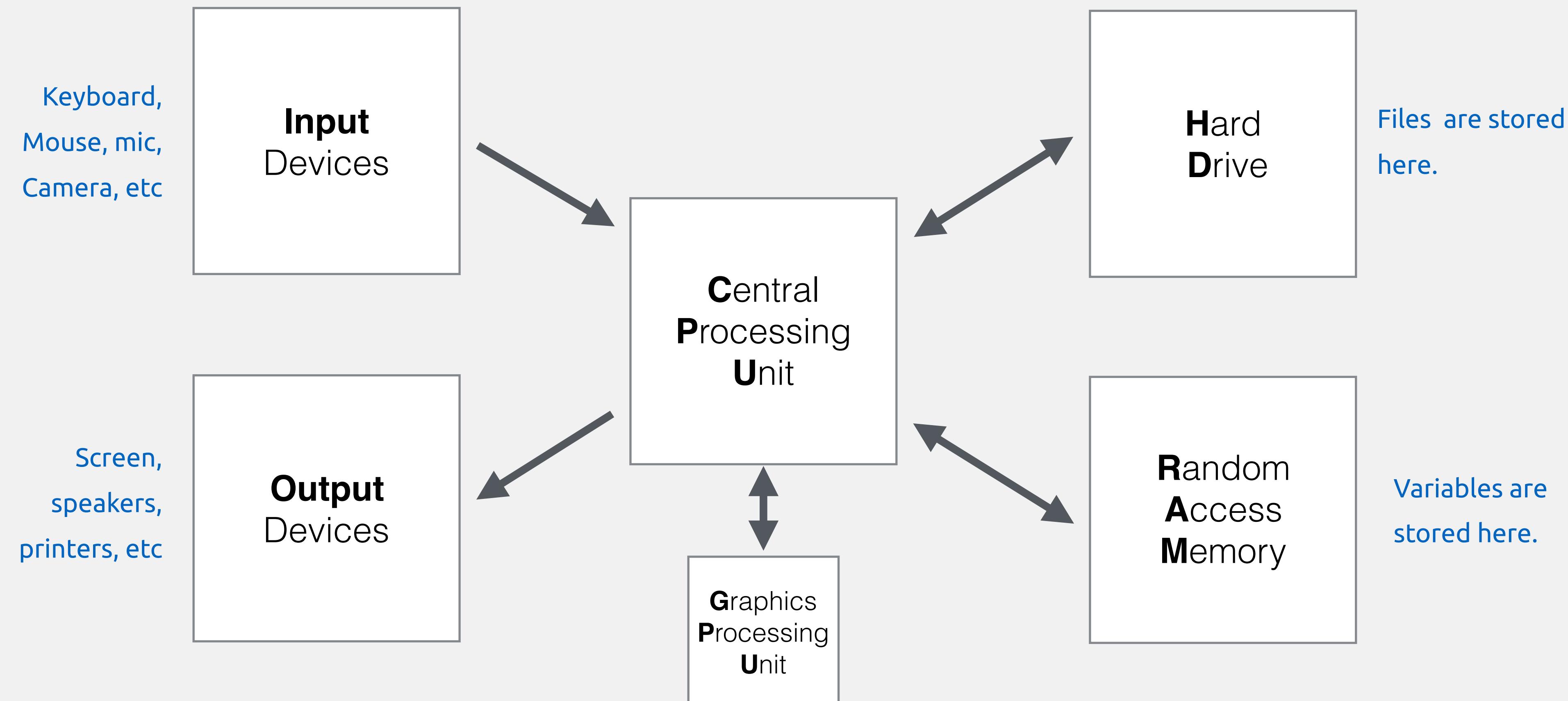
WEEK THREE - DECISION

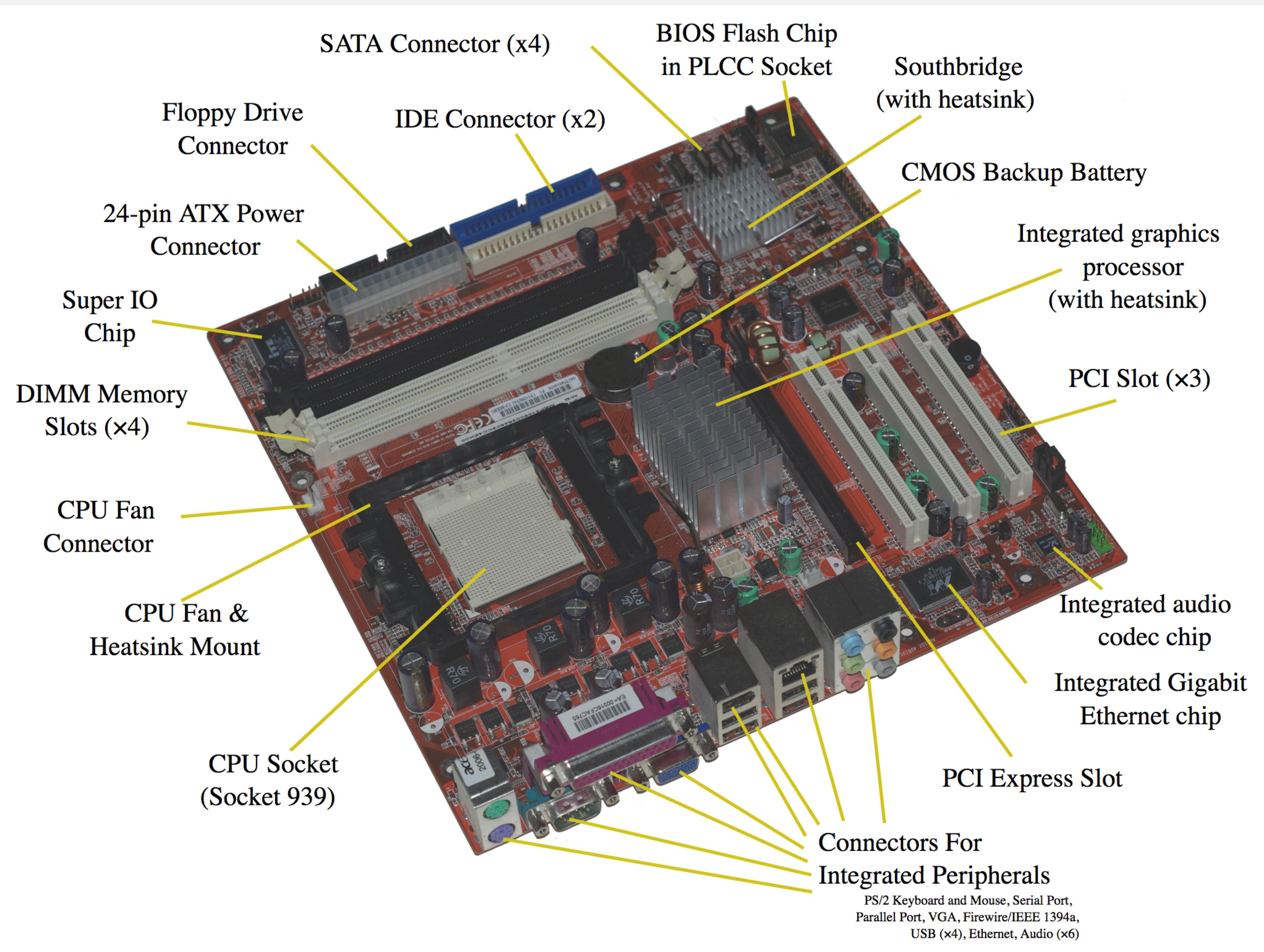
Conditionals / Console / User input

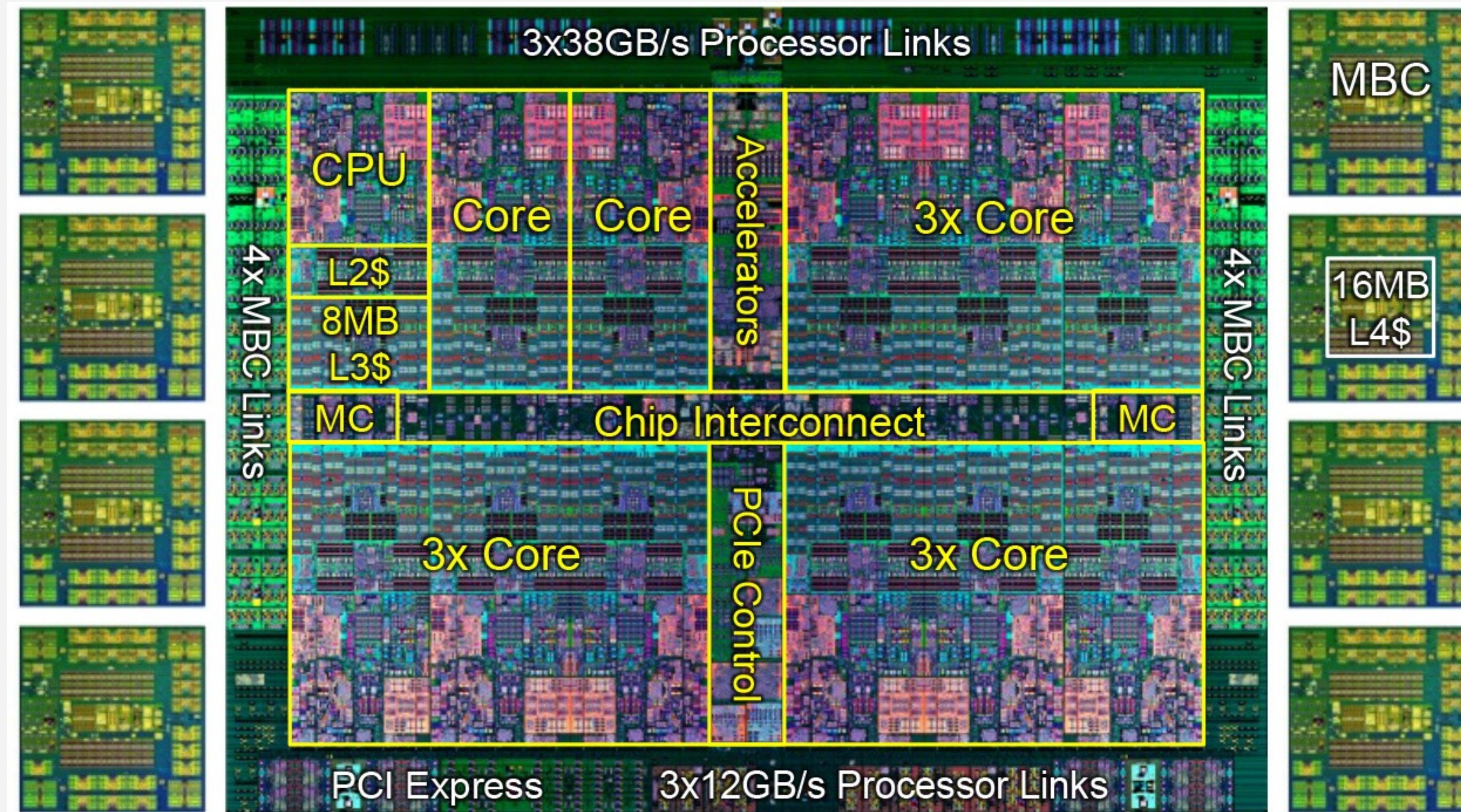
Lior Ben-Gai
November 2019

HIGH LEVEL VIEW OF A COMPUTER

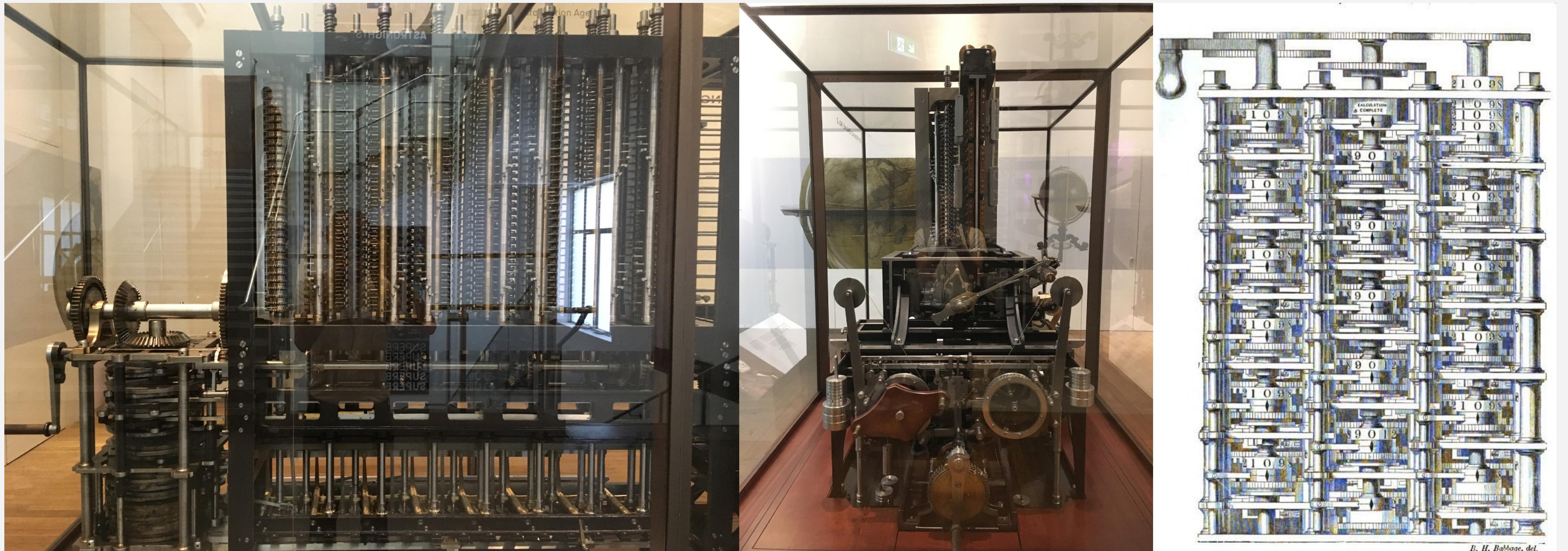
(very high level)







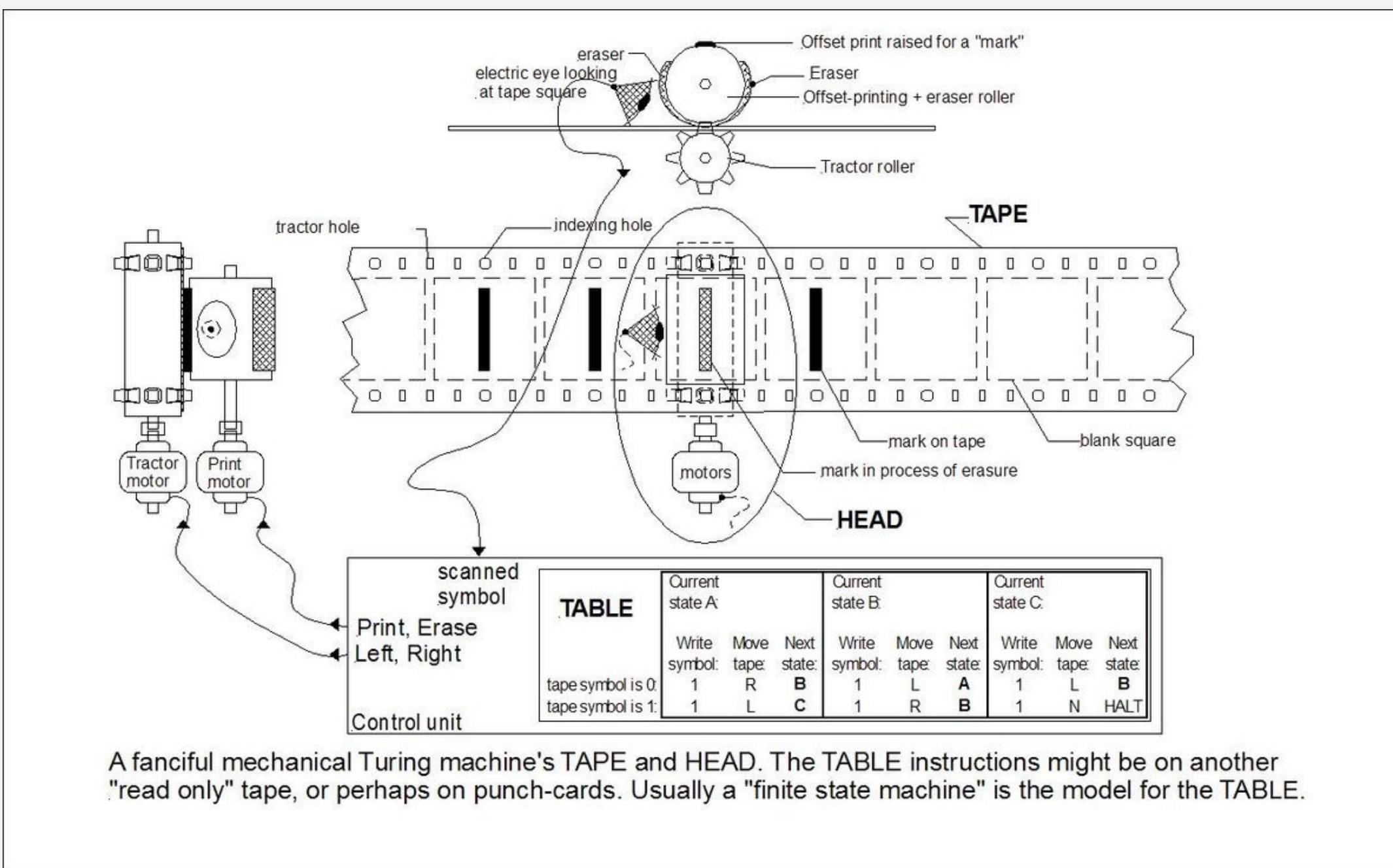
<http://www.ic-cracker.com/reverse-mcu-ic-renesas-r5f2i388cnfp/reverse-mcu-ic-renesas-r5f2i388cnfp-2/>



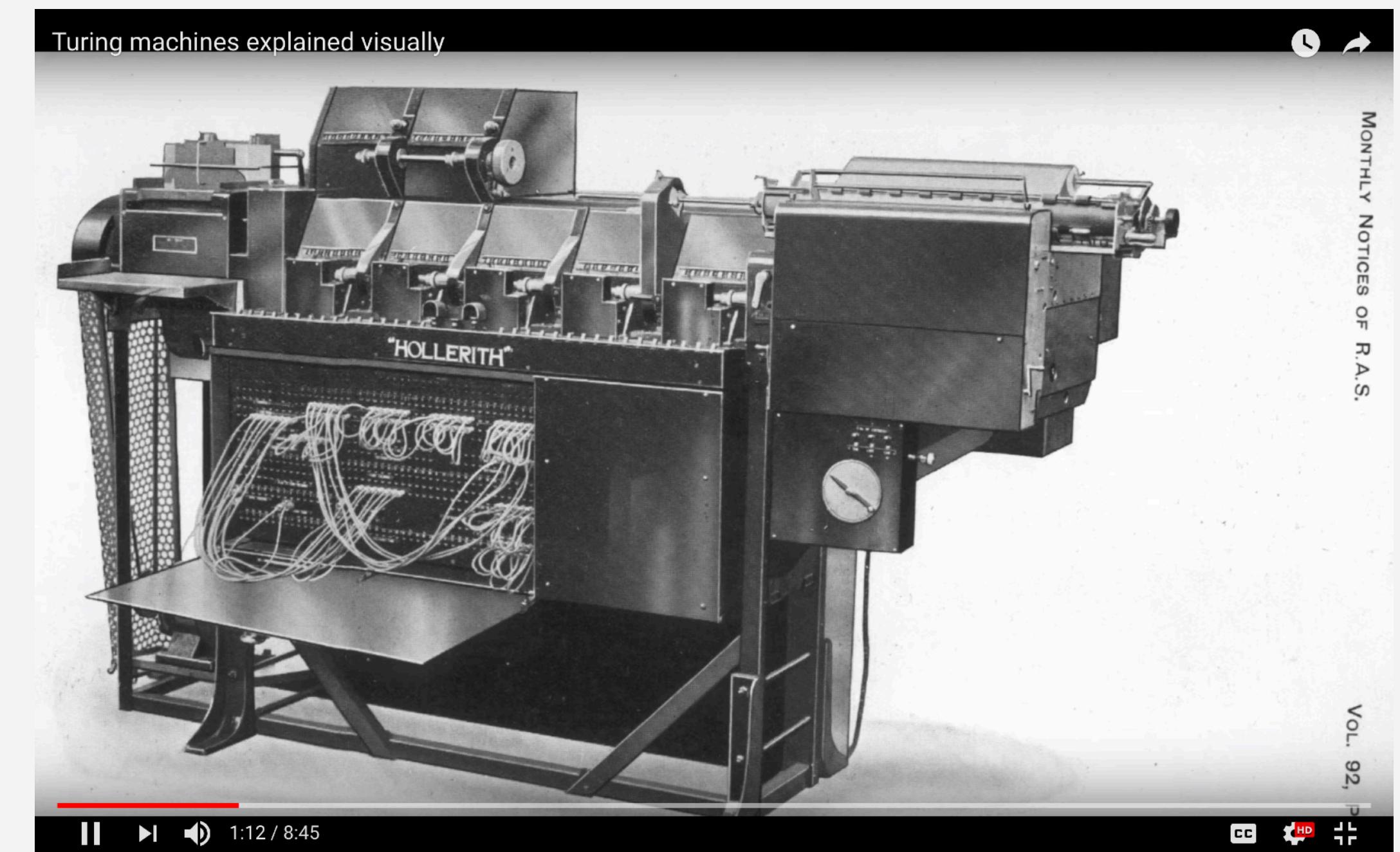
Charles Babbage
Difference Engine
~1822

https://en.wikipedia.org/wiki/Difference_engine

TURING MACHINE



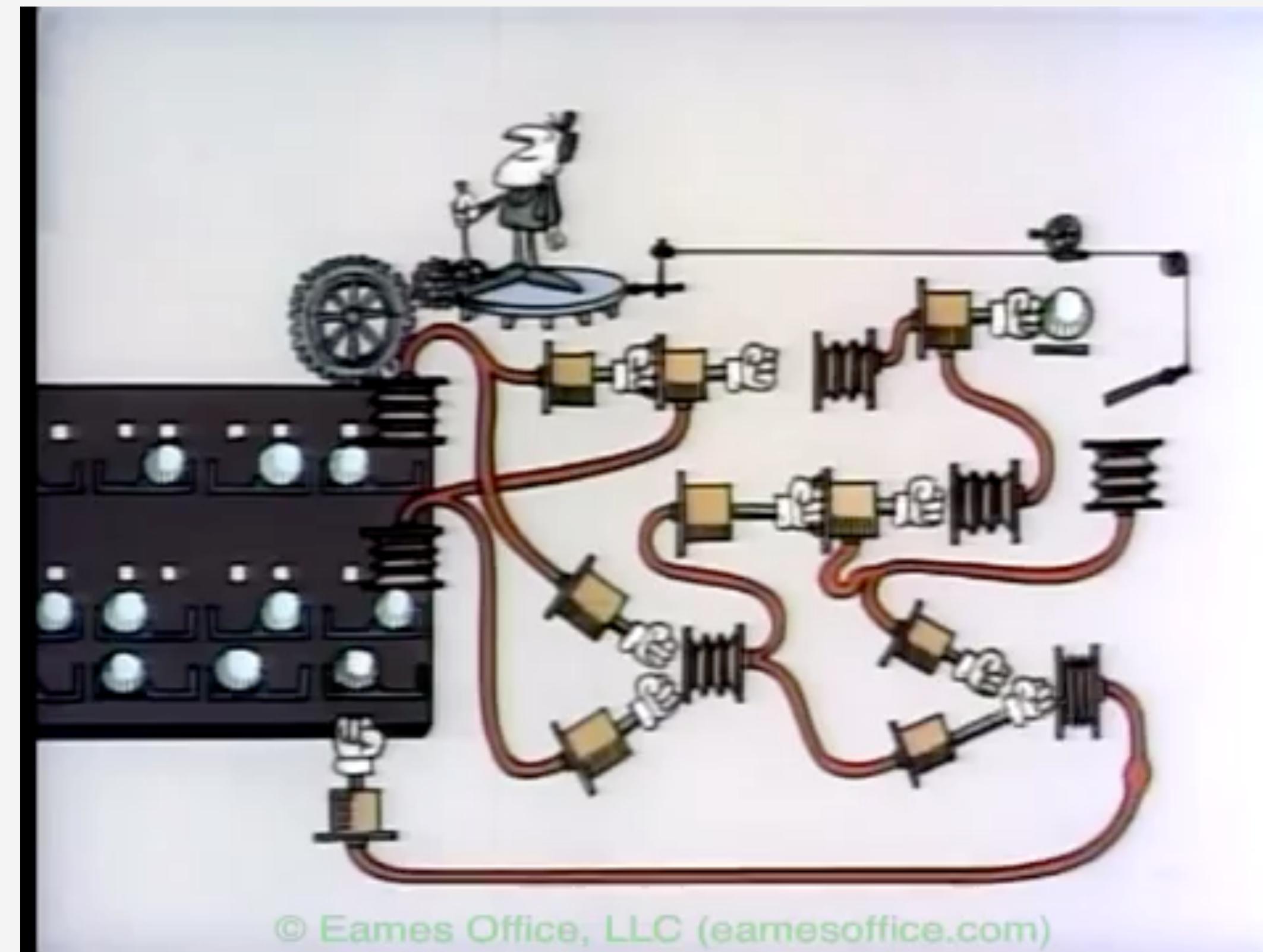
https://en.wikipedia.org/wiki/Turing_machine



https://www.youtube.com/watch?v=-ZS_zFg4w5k

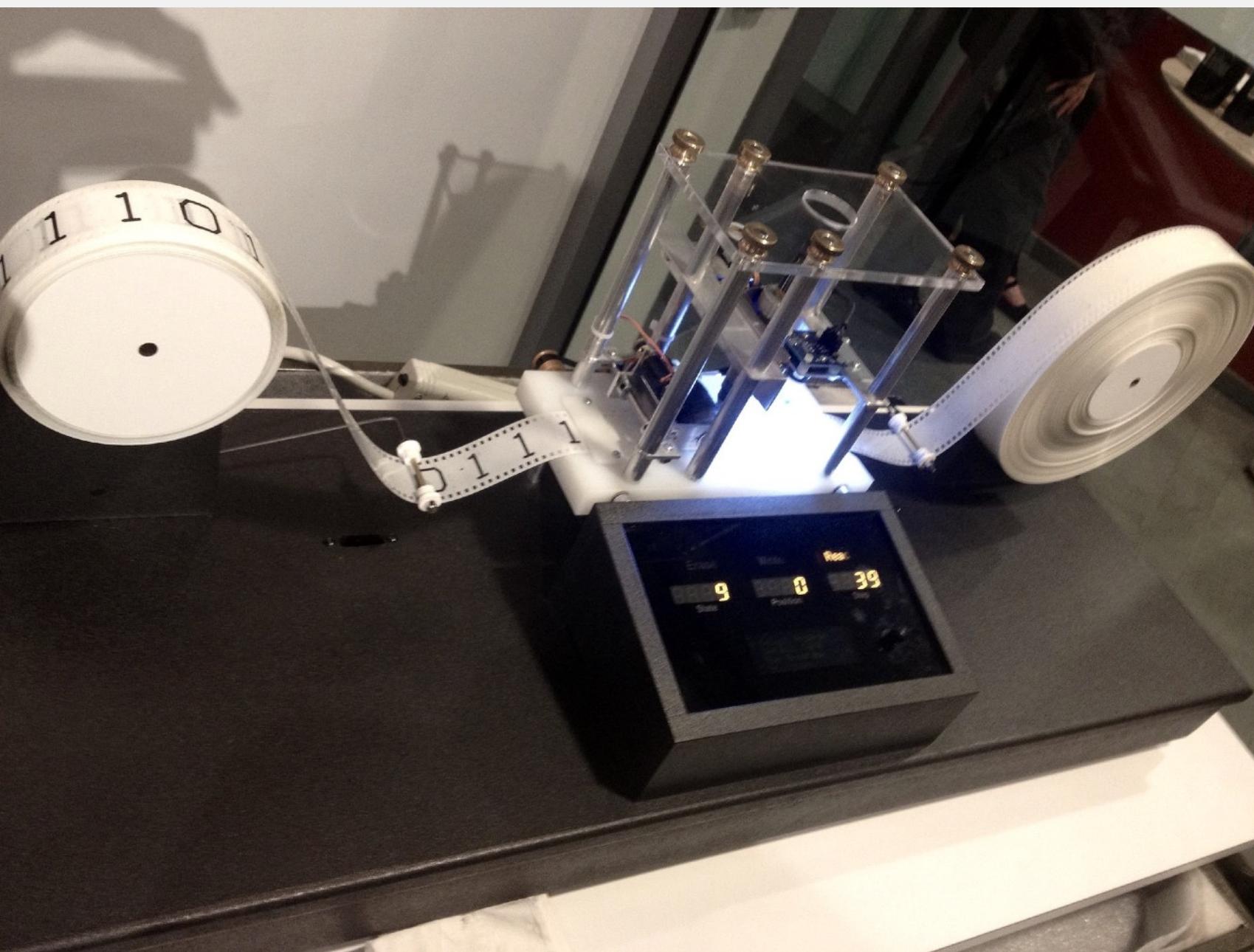
A COMPUTER GLOSSARY

Charles and Ray Eames 1966

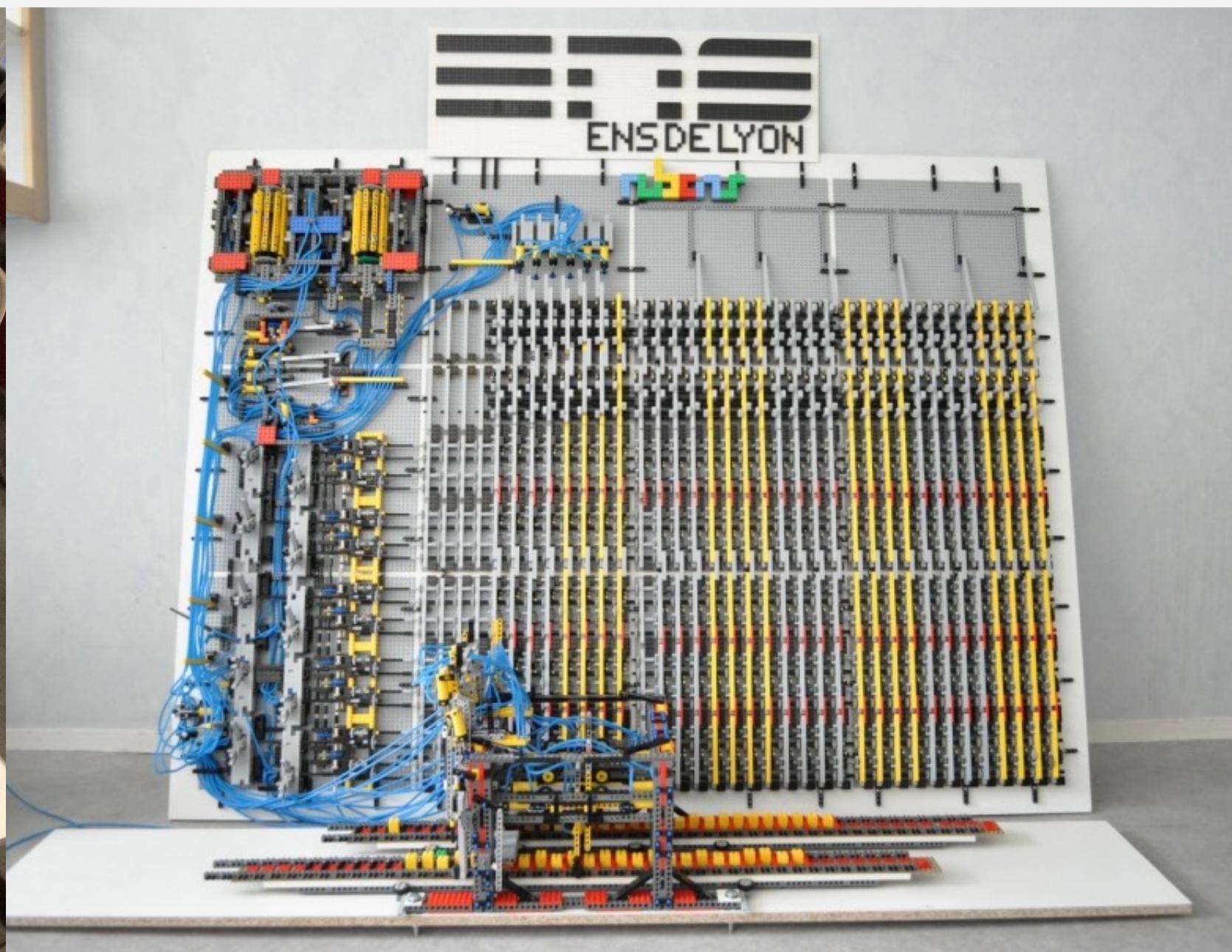


<https://www.youtube.com/watch?v=elgX6sPOqCY>

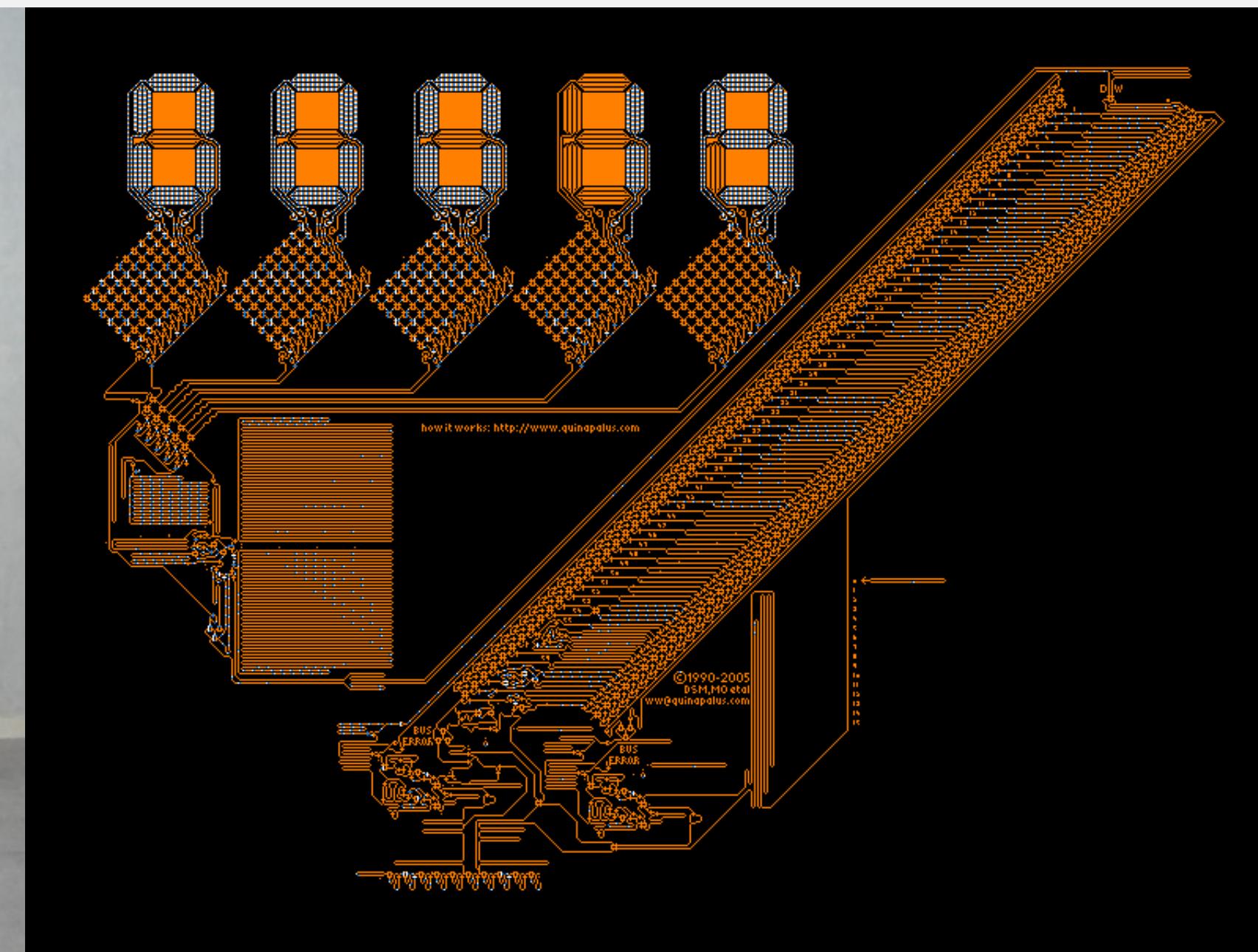
IMPLEMENTING A COMPUTER



Turing machine



in Lego



In software

[Computer Science with AGuy] Minecraft Advanced Quad-Core Redstone Computer v4.0 [200 sub special!]



Minecraft 1.11 (1.11/vanilla)
60 fps (51 chunk updates) T: inf VBO
C: 1037/17424 (s) D: 16, L: 0, pC: 000, pU: 0, aB: 60
E: 0/1, B: 0
P: 14, T: All: 1
MultiplayerChunkCache: 1089, 1089

XYZ: -83.302 / 93.13628 / -299.800

Block: -84 93 -300

Chunk: 12 13 4 in -6 5 -19

Facing: west (Towards negative X) (115.8 / 31.7)

Biome: Desert

Light: 15 (15 sky, 0 block)

Local Difficulty: 3.00 // 0.50 (Day 0)

Debug: Pie [shift]: hidden FPS [alt]: hidden

For help: press F3 + 0

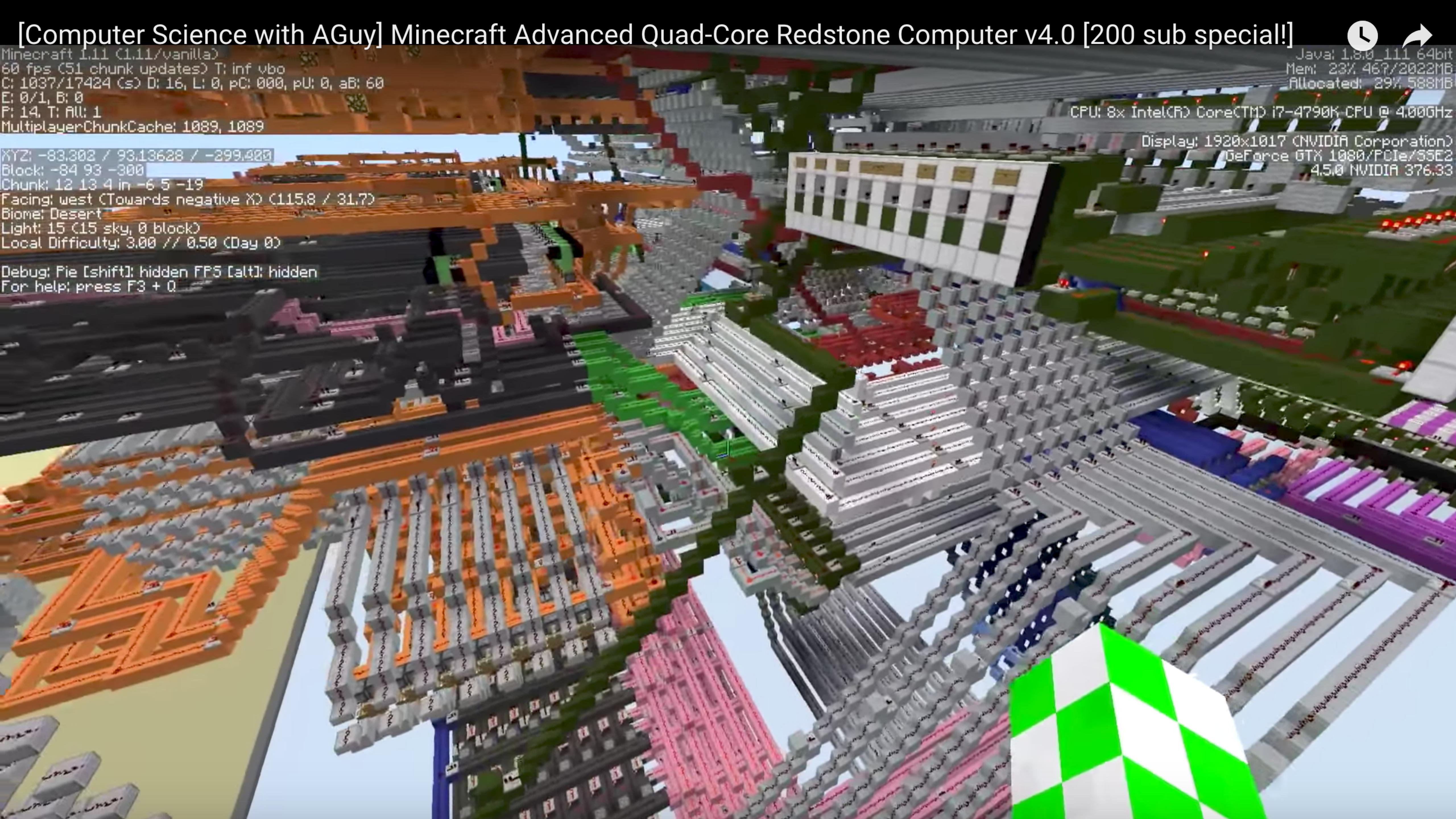
Java: 1.8.0_111 64bit
Mem: 23% 467/2022MB
Allocated: 29% 588MB

CPU: 8x Intel(R) Core(TM) i7-4790K CPU @ 4.00GHz

Display: 1920x1017 (NVIDIA Corporation)

GeForce GTX 1080/PCIe/35E2

4.5.0 NVIDIA 376.33



THE BUILDING BLOCKS OF CODING 01

MEMORY **DECISION** ACTION REPETITION

DECISION

Conditional statements allow the program to **make decisions** while it is running

A decision is always made according to the **true/false** value of a statement

DECISION

(1 + 1 == 2)



true

DECISION

(2 + 2 == 5)



false

DECISION

var a;

a = 10 → put 10 in a

a == 10 → is a equal to 10?

DECISION



(mouseX > width/2)



true

Larger than operator

DECISION

```
( ( 1 == 1 ) && ( 1 == 2 ) )
```



false

AND operator (**&&**) requires both sides to be **true**

DECISION

```
(( 1 == 1 ) || ( 1 == 2 ))
```



true

OR operator (||) requires at least one side to be **true**

DECISION

(1 != 2)



true

NOT Equal operator

DECISION

(!(1 == 2))



true

NOT operator (!) flips any statement

CONDITIONAL STATEMENT BLOCK

```
if( x > width ) {  
    x = 0;  
}
```

CONDITIONAL STATEMENT BLOCK

```
if( mouseIsPressed ){  
    point(mouseX, mouseY);  
}else{  
    background(0);  
}
```

BOOLEAN DATA TYPE

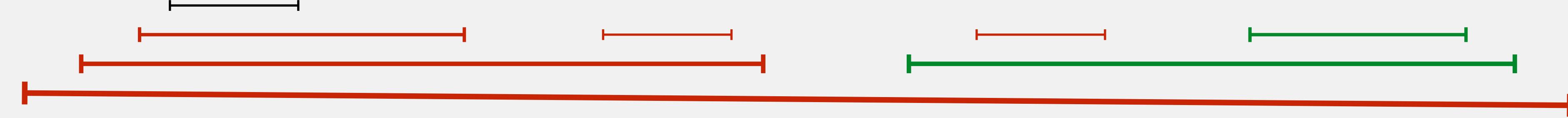
```
var userHasClicked = false;
```

```
if( userHasClicked ){
    // Do something
}else{
    // Do something else
}
```

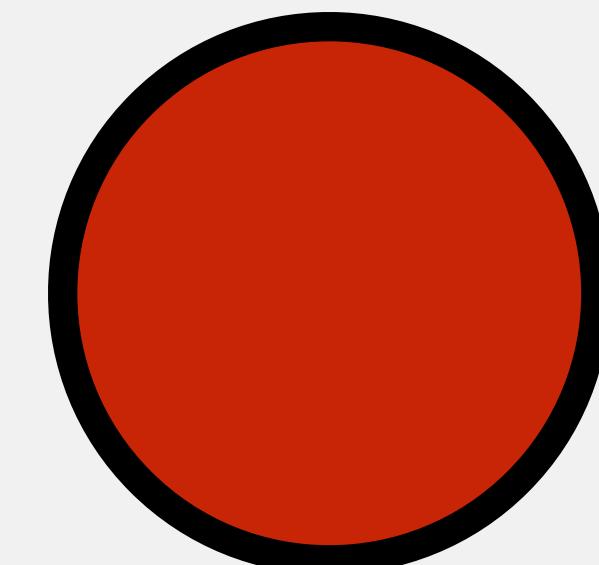
EXERCISE

```
var flagA = true;  
var flagB = false;  
var flagC = ( flagA && flagB );
```

```
var flagD = ( ( ( 1 + 1 == 3 ) || flagC ) && ( (!true) || ( 2 >= 2 ) ) );
```



```
if(flagD){  
    fill(0,255,0);  
} else {  
    fill(255,0,0);  
}  
ellipse(530, 720, 200, 200);
```



OVERVIEW

IF/ELSE STATEMENT

```
if( condition ){
    // do something
}else{
    // do something else
}
```

BOOLEAN DATA TYPE

```
var firstUse = true;
```

ASSIGN VS TEST

```
a = 10; // assign value to var
a == 10; // test if var is equal to value
```

BOOLEAN OPERATORS

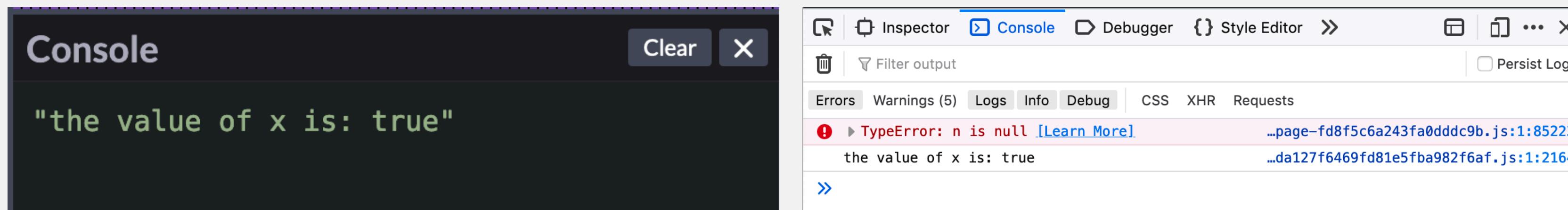
Is Equal to	a == b
Is Unequal to	a != b
Logical AND	a && b
Logical OR	a b
Logical NOT	!a

Is smaller than	a < b
Is larger than	a > b
Smaller OR equal	a <= b
Larger OR equal	a >= b

PRINT / CONSOLE.LOG

```
var a = true;
```

```
print('the value of x is: ' + a);
```



The image shows two side-by-side screenshots of browser developer tools. The left screenshot shows a dark-themed 'Console' tab with the text "the value of x is: true". The right screenshot shows a light-themed 'Console' tab with a 'Logs' tab selected, displaying the same log entry. Both screenshots also show other tabs like 'Inspector', 'Debugger', and 'Style Editor' at the top.

Console (Left)	Console (Right)
"the value of x is: true"	the value of x is: true
	...da127f6469fd81e5fba982f6af.js:1:2164

<https://p5js.org/reference/#/p5/print>

INPUT EVENTS

```
function mousePressed() {  
    point(mouseX, mouseY);  
}
```

<https://p5js.org/reference/#/p5/mousePressed>

INPUT EVENTS

```
function keyTyped( ) {  
    if(key == 's') {  
        save('result.png');  
    }  
}
```

<https://p5js.org/reference/#/p5/keyTyped>

HOMEWORK #3

Everyday activity

Please create an interactive P5 program that represents an everyday activity. It can be anything.

The program must take some **user input** (keyboard or mouse) and use it to perform an action.

- + Make **deliberate use** of programming tools (drawing commands, variables, conditions)
- + Use a distinct **visual language**
- + Do **visual story telling**