CMSC 240
PRINCIPLES OF COMPUTER ORGANIZATION

Deepak Kumar

1

CMSC 240_{10} (11110000_2) (360_8) ($F0_{16}$)
PRINCIPLES OF COMPUTER ORGANIZATION

Pre-requisites: CMSC B 151 or H106 or H107 and CMSC B231

2

ADMINISTRIVIA

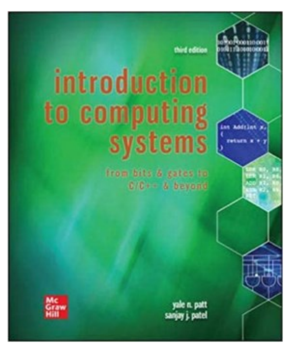
Instructor: Deepak Kumar
Office: Room 202 PSB
Office Hours: TBA

Lectures: Tue-Thu 12:55p to 2:15p
Lecture Room: 245 Park Science Building
Labs: Tuesdays from 2:25p to 3:45p (starts next week)
Lab Room: 231 Park Science Building

Class Website: https://cs.brynmawr.edu/Courses/cs240/Spring2024/

3

TEXT & SOFTWARE



Main Text
Introduction to Computing Systems: From Bits & Bytes to C/C++ and Beyond, **Third Edition**, McGraw Hill 2019.

Software
LC3 Simulator (Windows/MacOS/Linux)

4

LEARNING OUTCOMES

- Describe the major components of a modern computer (CPU, Memory, I/O) and how they are implemented and interact in hardware.
- Understand how programs and data are represented in hardware, how instructions are executed, and how data is stored in and retrieved from memory.
- Design circuits to implement Boolean functions and basic storage/memory constructs using digital logic.
- The von Neumann Model
- Relate the behavior of high-level languages like C or Java to the underlying low-level assembly language.

TOPICS

- What is a Computer?
- Instruction Set Architecture (ISA)
- Bits, Data Types, and Operations
- The von Neumann Model
- The LC3 ISA
- Programming in Assembly using LC3 ISA
- Subroutines and the Stack
- I/O Operations: Service Routines, Traps and Interrupts
- Memory Hierarchy & Caching
- Non-von Neumann Architectures

MY ROLE

- Create educational opportunities for you to achieve the learning outcomes.
- Assess your progress and provide timely feedback
- Provide support so you can successfully complete the course.

YOUR RESPONSIBILITIES

- Put in the effort
- Attend all classes and labs (let me know if you are going to miss any)
- Follow the rules and guidelines in the class website
- Stay connected
  - Check the class website for schedule, updates, readings, and assignments
  - Do the readings as and when they are assigned. This is a key to success.
  - Be familiar with the main concepts before every class meeting
  - Ask for help when you need it!

IMPORTANT DATES & EVALUATION

Important Dates	Activity
January 23	First Class Meeting
February 20	Exam 1
April 2	Exam 2
May 2	Exam 3

Assessments	Weight
Exam 1	20%
Exam 2	20%
Exam 3	20%
Assignments	30%
Labs	10%

9

WHAT IS A COMPUTER?

10

WHAT IS A COMPUTER?

A device with

- A processor (CPU – Central Processing Unit)
- Storage/Memory (RAM, Disk, USB Stick)
- A keyboard
- A mouse
- A Monitor
- A printer

11

THESE ARE ALL COMPUTERS

12

THIS IS ALSO A COMPUTER

FRONTIER

World's Fastest Computer (2023)

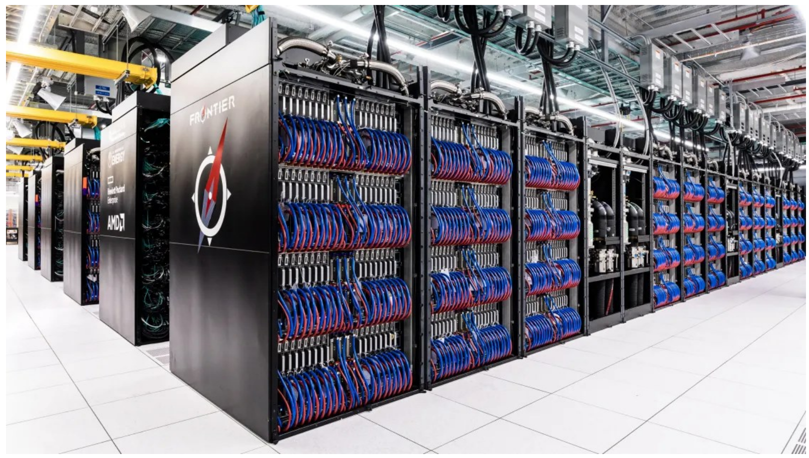
Oak Ridge National Labs, TN

~50,000 Processors

Size of two tennis courts

Costs $600 million

Can execute ~1 ExaFLOP (10^{18} FLOPS)



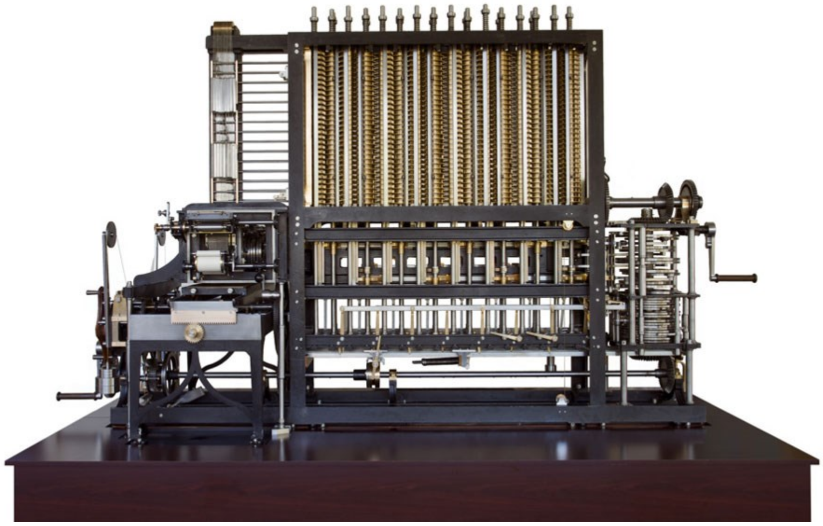
13

WHAT WAS THE FIRST COMPUTER?

14

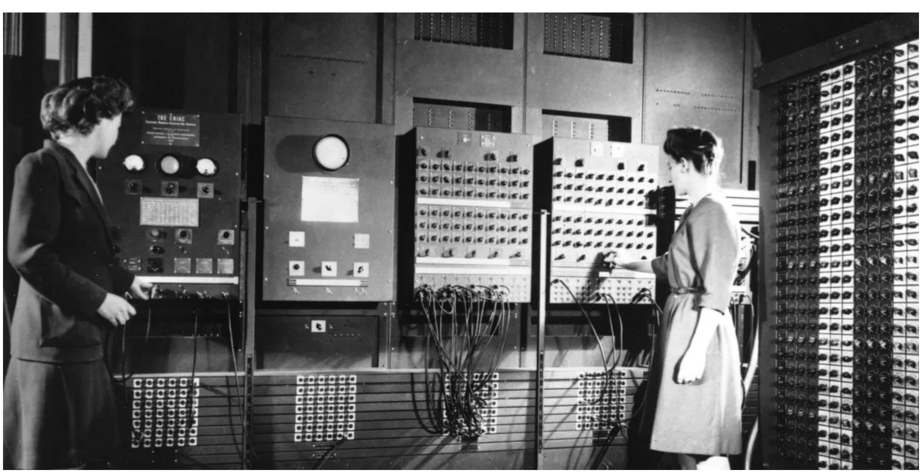
BABBAGE'S DIFFERENCE ENGINE#2 (1832, 2002)

See video at: https://youtu.be/XSkGY6LchJs?si=GzWuDh9yPtsW9DBL



ENIAC FEBRUARY 16, 1946 (U. PENN)

See video at: https://youtu.be/XSkGY6LchJs?si=GzWuDh9yPtsW9DBL



See at: Moore School Building, Corner of 34th & Walnut, Philadelphia.

WHAT IS THE DIFFERENCE BETWEEN THE ENIAC, IPHONE, AND FRONTIER?

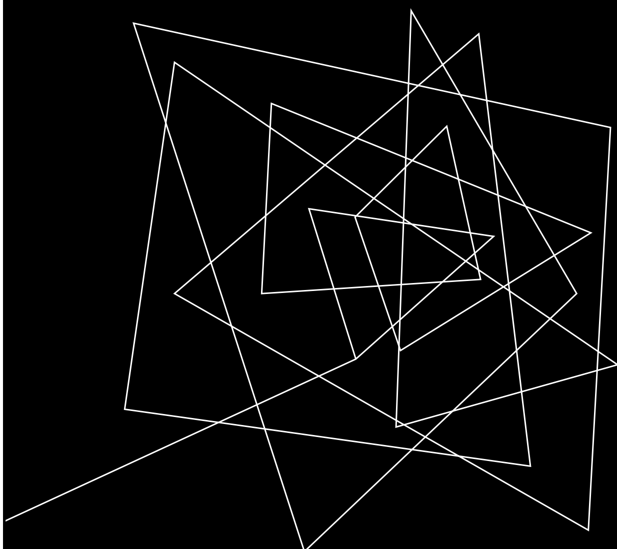
17

WHAT IS THE DIFFERENCE BETWEEN THE ENIAC, IPHONE, AND FRONTIER?

NOTHING!

Besides the size, cost, the speed of the processor(s), and amount of memory.

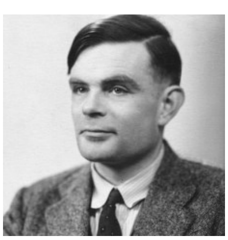
18

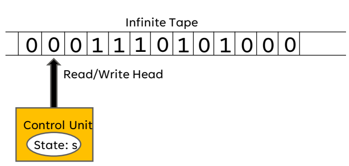


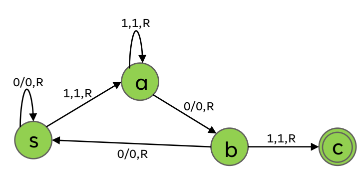
BUT REALLY, WHAT IS A COMPUTER?

19

TURING MACHINE: AN IDEALIZED COMPUTER







	0	1
s	0,R, s	1,R, a
a	0,R, b	1,R, a
b	0,R s	1,R c

20

TURING MACHINES FOR ADDING & MULTIPLYING

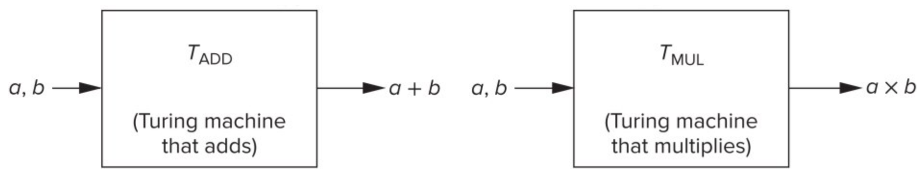


Figure 1.7 Black box models of Turing Machines

21

CHURCH-TURING THESIS* (1936)

EVERY COMPUTATION CAN BE PERFORMED BY SOME TURING MACHINE.

22

UNIVERSAL TURING MACHINES

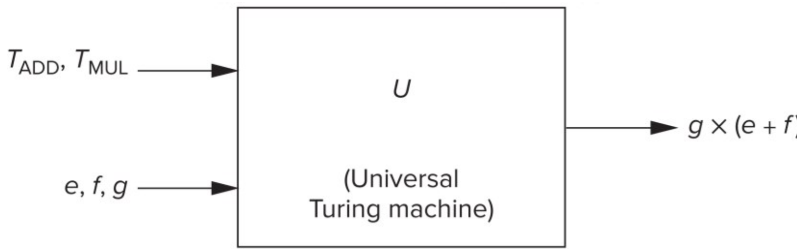


Figure 1.8 Black box model of a universal Turing Machine

UNIVERSAL TURING MACHINE = PROGRAMMABLE COMPUTER

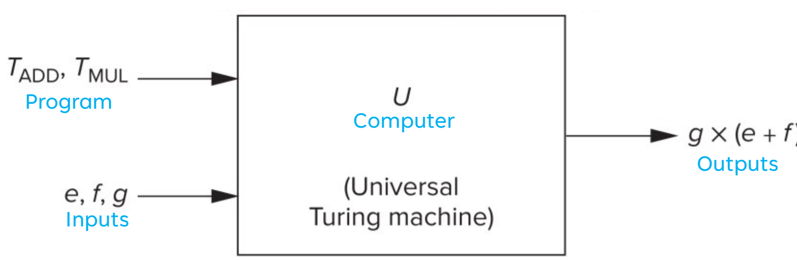


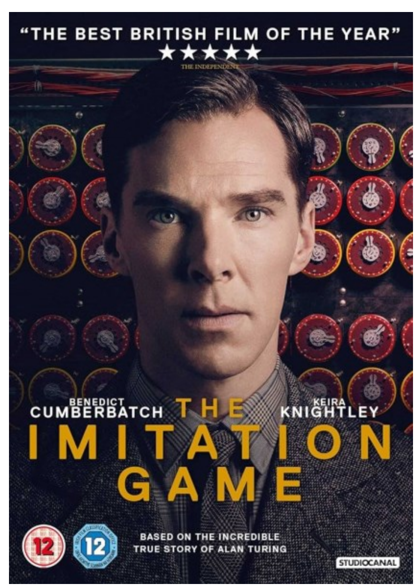
Figure 1.8 Black box model of a universal Turing Machine

KEY IDEA

A COMPUTER IS ESSENTIALLY A UNIVERSAL TURING MACHINE.

25

PUT THIS ON YOUR WATCH LIST



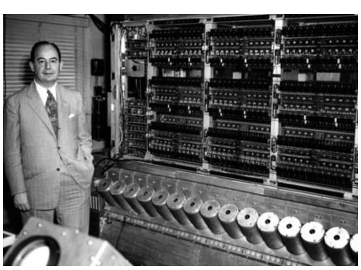
26

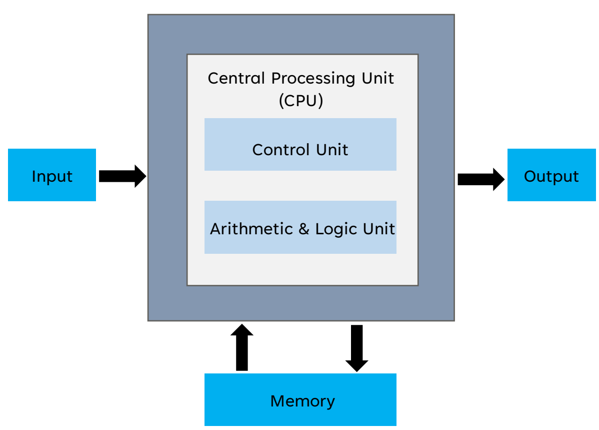
OK, SO…

HOW DO WE ACTUALLY BUILD A UNIVERSAL TURING MACHINE (COMPUTER)???

27

VON NEUMAN ARCHITECTURE (1945)

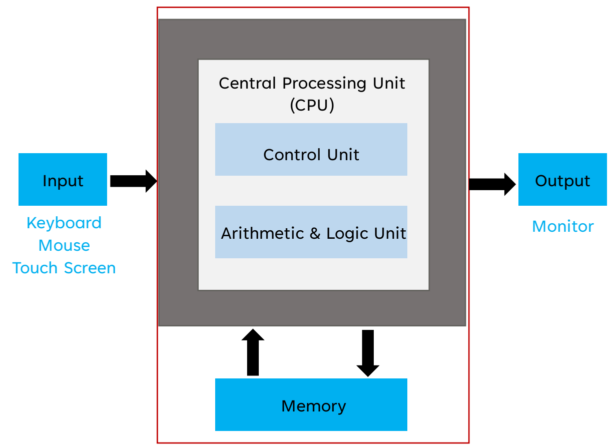




CMSC 240 Principles of Computer Organization 28

28

VON NEUMAN ARCHITECTURE



29

Problems

Algorithms

Language

Machine (ISA) Architecture

Microarchitecture

Circuits

Devices

NON-VON NEUMANN COMPUTERS

Technically, most of today's computers are non-von Neumann computers.

I what way?

To understand/answer this, we will first need to learn what von Neumann computers are.

30

HOW DO WE GET THE COMPUTER TO WORK???

31

LEVELS OF TRANSFORMATION

Problems

Algorithms

Language

Machine (ISA) Architecture

Microarchitecture

Circuits

Devices

CMSC 240 Principles of Computer Organization 32

32

LEVELS OF TRANSFORMATION

Problems

Algorithms

Language

Software

Machine (ISA) Architecture

Hardware

Microarchitecture

Circuits

Devices

LEVELS OF TRANSFORMATION: EXAMPLE

Problems

Algorithms

Language

Machine (ISA) Architecture

Microarchitecture

Circuits

Devices

PROBLEM

Compute the square root of a given number, a.

Problems ⬅
Algorithms
Language
Machine (ISA) Architecture
Microarchitecture
Circuits
Devices

LEVELS OF TRANSFORMATION

PROBLEM

Compute the square root of a given number, a.

A SOLUTION

To compute the square root $x = \sqrt{a}$ do the following:

1. Start with some guess $x_1 > 0$
2. Compute a sequence of guesses $x_1, x_2, \ldots, x_n$ using the equation

$$x_{n+1} = \frac{1}{2}\left(x_n + \frac{a}{x_n}\right)$$

until the numbers produced converge.

Problems
Algorithms ⬅
Language
Machine (ISA) Architecture
Microarchitecture
Circuits
Devices

LEVELS OF TRANSFORMATION

ALGORITHM

1. To compute $\sqrt{a}$
2. Start with some guess $x_i = 1$. This is our initial guess.
3. Compute the next guess $x_{i+1} = \frac{1}{2}\left(x_i + \frac{a}{x_i}\right)$
4. If $x_{i+1} \neq x_i$
   Set x_i to be same as x_{i+1}
   And then repeat from Step 3.

   Otherwise, because $x_{i+1} = x_i$, they have converged.
   Therefore, $\sqrt{a} = x_{i+1}$

Problems

Algorithms

Language

Machine (ISA) Architecture

Microarchitecture

Circuits

Devices

LEVELS OF TRANSFORMATION

ALGORITHM

1. To compute $\sqrt{a}$
2. Start with some guess $x_i = 1$. This is our initial guess.
3. Compute the next guess $x_{i+1} = \frac{1}{2}\left(x_i + \frac{a}{x_i}\right)$
4. If $x_{i+1} \neq x_i$
   Set x_i to be same as x_{i+1}
   And then repeat from Step 3.

   Otherwise because $x_{i+1} = x_i$, they have converged.
   Therefore, $\sqrt{a} = x_{i+1}$

Definition:
An algorithm is a *precise*, *unambiguous*, and *effective* procedure.

37

Problems

Algorithms

Language

Machine (ISA) Architecture

Microarchitecture

Circuits

Devices

LEVELS OF TRANSFORMATION

PROGRAM

```
double sqrt (double a) {
   if (a <= 0) return 0;

   double x0 = 1;
   double x1 = (x0 + a/x0)/2.0;

   while (x0 != x1) {
      x0 = x1;
      x1 = (x1 + (a/x1))/2.0;
   }
   return x1;
} // sqrt()
```


38

Problems

Algorithms

Language

Machine (ISA) Architecture

Microarchitecture

Circuits

Devices

LEVELS OF TRANSFORMATION

ASSEMBLY PROGRAM (ISA)

```
        .globl sqrt
sqrt:
.LFB0:
        .cfi_startproc
        pushq %rbp
        .cfi_def_cfa_offset 16
        .cfi_offset 6, -16
        movq %rsp, %rbp
        .cfi_def_cfa_register 6
        movsd %xmm0, -24(%rbp)
        pxor %xmm0, %xmm0
        comisd -24(%rbp), %xmm0
        jb .L8
        pxor %xmm0, %xmm0
        jmp .L4
.L8:
        movsd .LC1(%rip), %xmm0
        movsd %xmm0, -16(%rbp)
        movsd -24(%rbp), %xmm0
        divsd -16(%rbp), %xmm0
        addsd -16(%rbp), %xmm0
        movsd .LC2(%rip), %xmm1
        divsd %xmm1, %xmm0
        movsd %xmm0, -8(%rbp)
        jmp .L5
.L6:
        movsd -8(%rbp), %xmm0
        movsd %xmm0, -16(%rbp)
        movsd -24(%rbp), %xmm0
        divsd -8(%rbp), %xmm0
        addsd -8(%rbp), %xmm0
        movsd .LC2(%rip), %xmm1
        divsd %xmm1, %xmm0
        movsd %xmm0, -8(%rbp)
.L5:
        movsd -16(%rbp), %xmm0
        jp .L6
        movsd -16(%rbp), %xmm0
        ucomisd -8(%rbp), %xmm0
        jne .L6
        movsd -8(%rbp), %xmm0
.L4:
        popq %rbp
        .cfi_def_cfa 7, 8
        ret
        .cfi_endproc
.LFE0:
        .size sqrt, .-sqrt
        .section .rodata
        .align 8
.LC1:
        .long 0
        .long 1072693248
        .align 8
.LC2:
        .long 0
        .long 1073741824
```


Problems

Algorithms

Language

Machine (ISA) Architecture

Microarchitecture

Circuits

Devices

Java Program → compiler → Assembly Program

LEVELS OF TRANSFORMATION

ASSEMBLY PROGRAM (ISA)

Instruction Set Architecture (e.g. x86, ARM, LC3, etc.)

opcode operands

```
        .globl sqrt
sqrt:
.LFB0:
        .cfi_startproc
        pushq %rbp
        .cfi_def_cfa_offset 16
        .cfi_offset 6, -16
        movq %rsp, %rbp
        .cfi_def_cfa_register 6
        movsd %xmm0, -24(%rbp)
        pxor %xmm0, %xmm0
        comisd -24(%rbp), %xmm0
        jb .L8
        pxor %xmm0, %xmm0
        jmp .L4
.L8:
        movsd .LC1(%rip), %xmm0
        movsd %xmm0, -16(%rbp)
        movsd -24(%rbp), %xmm0
        divsd -16(%rbp), %xmm0
        addsd -16(%rbp), %xmm0
        movsd .LC2(%rip), %xmm1
        divsd %xmm1, %xmm0
        movsd %xmm0, -8(%rbp)
        jmp .L5
.L6:
        movsd -8(%rbp), %xmm0
        movsd %xmm0, -16(%rbp)
        movsd -24(%rbp), %xmm0
        divsd -8(%rbp), %xmm0
        addsd -8(%rbp), %xmm0
        movsd .LC2(%rip), %xmm1
        divsd %xmm1, %xmm0
        movsd %xmm0, -8(%rbp)
.L5:
        movsd -16(%rbp), %xmm0
        jp .L6
        movsd -16(%rbp), %xmm0
        ucomisd -8(%rbp), %xmm0
        jne .L6
        movsd -8(%rbp), %xmm0
.L4:
        popq %rbp
        .cfi_def_cfa 7, 8
        ret
        .cfi_endproc
.LFE0:
        .size sqrt, .-sqrt
        .section .rodata
        .align 8
.LC1:
        .long 0
        .long 1072693248
        .align 8
.LC2:
        .long 0
        .long 1073741824
```

- Problems
- Algorithms
- Language
- Machine (ISA) Architecture ←
- Microarchitecture
- Circuits
- Devices

LEVELS OF TRANSFORMATION

MACHINE LANGUAGE PROGRAM

```
0101000000100000    0001010010000001    0110010100000000
0001000000100001    0000010000000101    0000001111111010
0101001001100000    0001100100100001    0101000000100000
0001001001111011    0001011011111111    1111000000100101
0101011011100000    0110010100000000    0011000100000000
0001011011101010    0000001111111010    0101000000100000
0010100000001001    0101000000100000    0001000000100001
0110010100000000    1111000000100101    0101001001100000
0001010010000001    0011000100000000    0001001001111011
0000010000000101    0101000000100000    0101011011100000
0001100100100001    0001000000100001    0001011011101010
0001011011111111    0101001001100000    0010100000001001
0110010100000000    0001001001111011    0110010100000000
0000001111111010    0101011011100000    0001010010000001
0101000000100000    0001011011101010    0000010000000101
1111000000100101    0010100000001001    0001100100100001
0011000100000000    0110010100000000    0001011011111111
                                        ...
```


41

- Problems
- Algorithms
- Language
- Machine (ISA) Architecture ←
- Microarchitecture
- Circuits
- Devices

Assembly Program → assembler → Machine Language Program

LEVELS OF TRANSFORMATION

MACHINE LANGUAGE PROGRAM

← Binary representation of assembly program

```
0101000000100000    0001010010000001    0110010100000000
0001000000100001    0000010000000101    0000001111111010
0101001001100000    0001100100100001    0101000000100000
0001001001111011    0001011011111111    1111000000100101
0101011011100000    0110010100000000    0011000100000000
0001011011101010    0000001111111010    0101000000100000
0010100000001001    0101000000100000    0001000000100001
0110010100000000    1111000000100101    0101001001100000
0001010010000001    0011000100000000    0001001001111011
0000010000000101    0101000000100000    0101011011100000
0001100100100001    0001000000100001    0001011011101010
0001011011111111    0101001001100000    0010100000001001
0110010100000000    0001001001111011    0110010100000000
0000001111111010    0101011011100000    0001010010000001
0101000000100000    0001011011101010    0000010000000101
1111000000100101    0010100000001001    0001100100100001
0011000100000000    0110010100000000    0001011011111111
                                        ...
```

What the CPU does:

```
do forever
   fetch the next instruction
   decode it
   carry it out
```


42

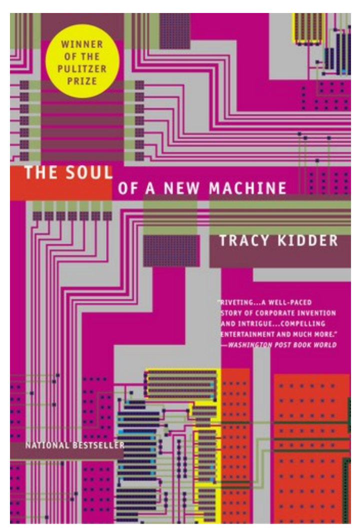
LEVELS OF TRANSFORMATION

Level	
Problems	compute $\sqrt{a}$
Algorithms	1. 2. 3. ...
Language	Python Program
Machine (ISA) Architecture	Machine Code (LC3)
Microarchitecture	
Circuits	
Devices	← We will start here!

CMSC 240 Principles of Computer Organization 43

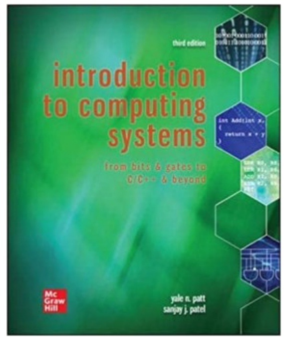
43

PUT THIS ON YOUR READING LIST



CMSC 240 Principles of Computer Organization 44

44



Read
Chapter 1 and Sections 2.1 and 2.2 from your text.

FOR NEXT CLASS…

45

THANK YOU

46