Mc Graw Hill Education

Because learning changes everything.®

From Bits and Gates to C and Beyond

Welcome Aboard

Chapter 1

Introduction to the World of Computing

There is no **magic** to computing!

- Deterministic system – behaves the same way every time.
- Does exactly what we tell it to do: no more, no less.
- Complex system made of very simple parts.
- Even recent advances in AI come from our ability to do many (billions!) simple computations very fast!

Access the text alternative for slide images.

Two Recurring Themes

Abstraction: Productivity Enhancer

You don't need to worry about the details…

- You can drive a car without knowing how an internal combustion engine works.

... until something goes wrong!

- Where's the dipstick? What's a spark plug? Where's that smoke coming from?

Important to understand the components and how they work together.
But thinking at higher levels of abstraction is more efficient.

Hardware **and** Software

It's not either/or – both are essential components of a computer system.

Even if you specialize in one,
you must understand the capabilities and limitations of the other.

Big Idea #1: Universal Computing Device

All computers, given enough time and memory, are capable of computing exactly the same things.

- Smartphone, laptop, supercomputer… limited only by time and memory.

Anything that can be computed, can be computed by a computer.

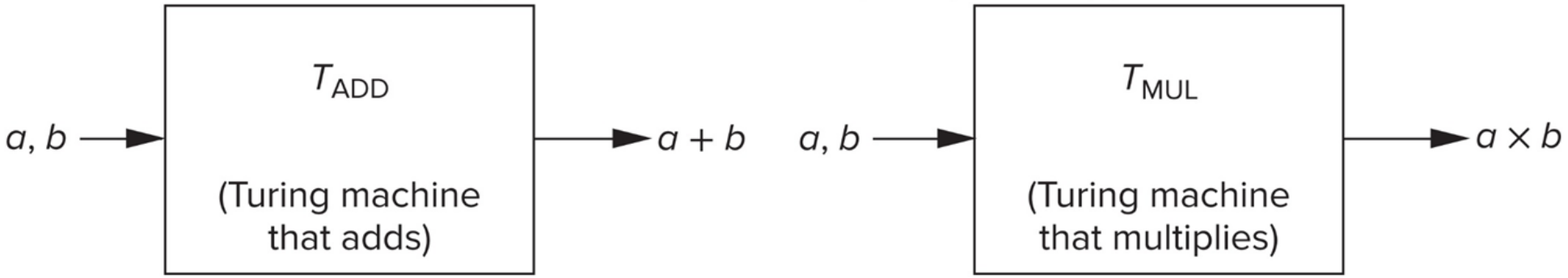
- If you can describe something in terms of computation, it can be done by a computer… again, given enough time and memory.

Turing Machine

Mathematical model of a device that can perform any computation – Alan Turing (1937)

- ability to read/write symbols on an infinite "tape"
- state transitions, based on current state and symbol

Every computation can be performed by some Turing machine. (*Turing's thesis*)



For more info about Turing machines, see

http://www.wikipedia.org/wiki/Turing_machine

For more about Alan Turing, see

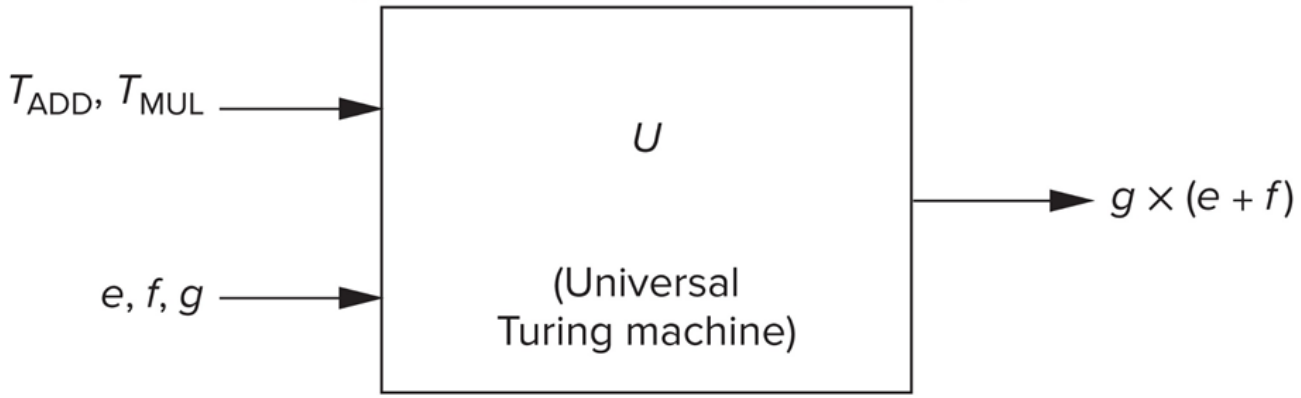
http://www.turing.org.uk/

Access the text alternative for slide images.

Universal Turing Machine

A machine that can implement all Turing machines
-- this is also a Turing machine!

- inputs: data, plus a description of computation (other TMs).



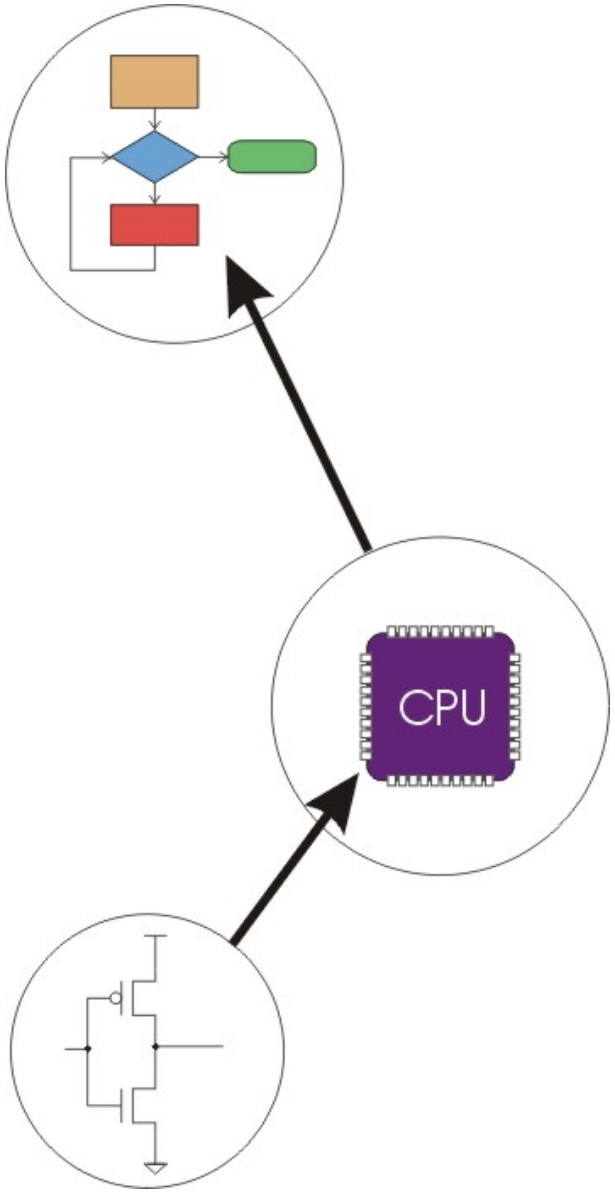
U is programmable – so is a computer!

- **Instructions are part of the input data.**
- **A computer can emulate a Universal Turing Machine.**

A computer is a universal computing device.

Access the text alternative for slide images.

Big Idea #2: Transformations Between Layers



Problems

Algorithms

Language

Instruction Set Architecture

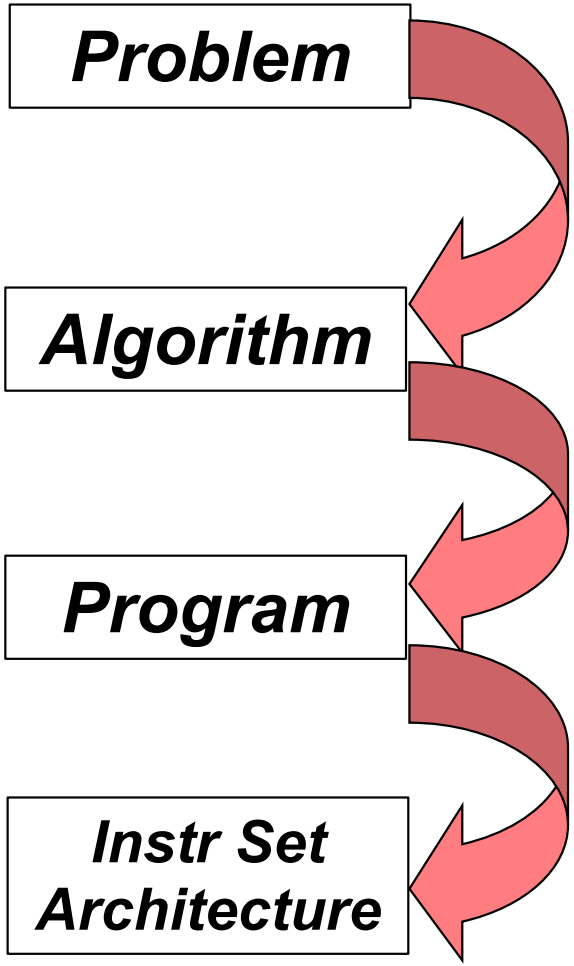
Microarchitecture

Circuits

Devices

How do we solve a problem using a computer?

A systematic sequence of transformations between abstraction layers.



Software Design:

choose algorithms and data structures

Programming:

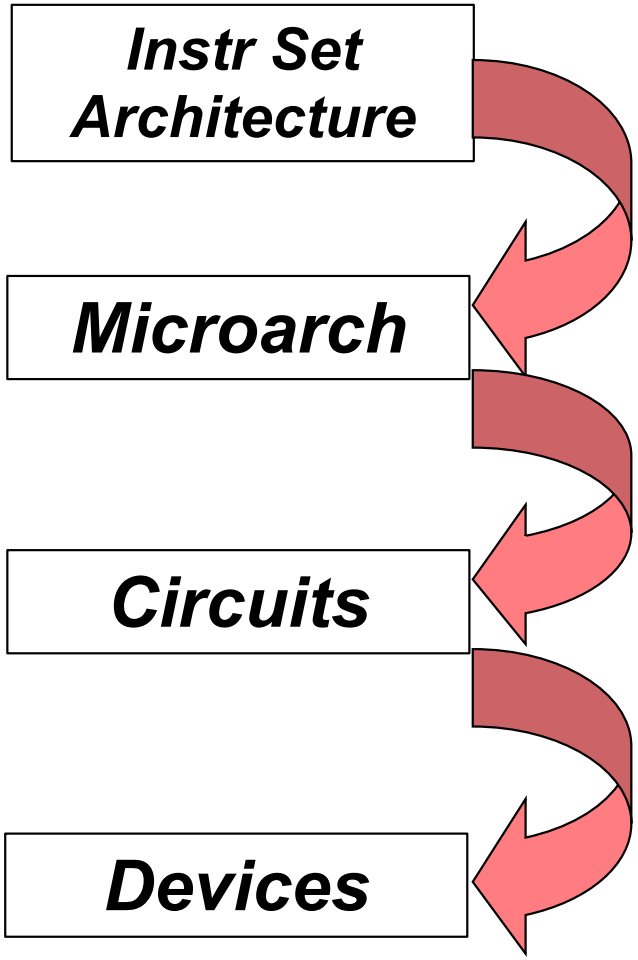
use language to express design

Compiling/Interpreting:

convert language to machine instructions

Access the text alternative for slide images.

…and even more layers…



Processor Design:

choose structures to implement ISA

Logic/Circuit Design:

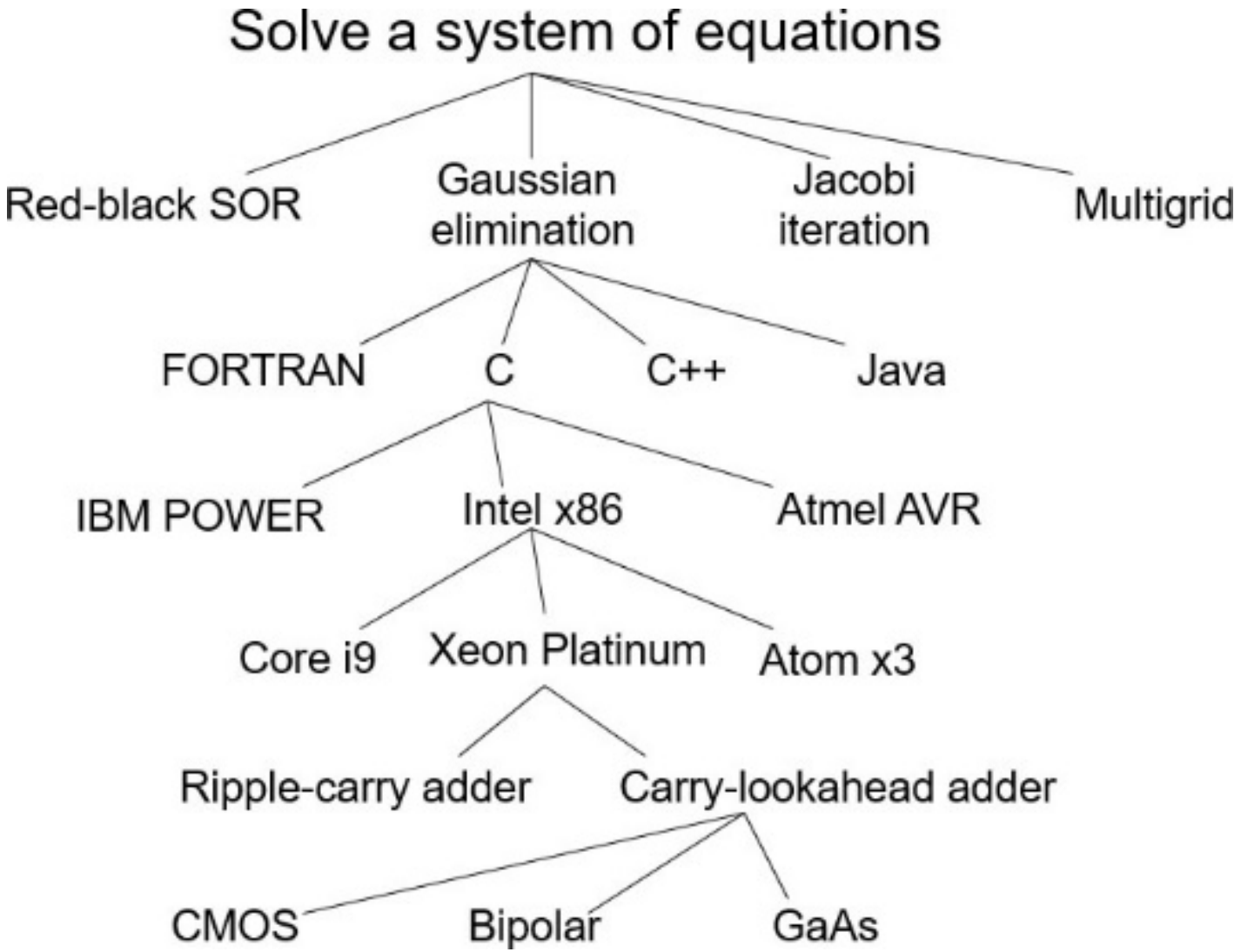
gates and circuits to implement components

Process Engineering & Fabrication:

develop and manufacture transistors, wires, etc.

Access the text alternative for slide images.

Many choices at each layer



Tradeoffs:
cost
performance
power
(etc.)

Access the text alternative for slide images.

Course Outline

Bits and Bytes

- How do we represent information using electrical signals?

Digital Logic

- Leave for next course

Processor and Instruction Set

- How do we build a processor out of logic elements?
- What operations (instructions) will we implement?

Assembly Language Programming

- How do we use processor instructions to implement algorithms?
- How do we write modular, reusable code? (subroutines)

I/O, Traps, and Interrupts

- How does processor communicate with outside world?

C Programming

- How do we write programs in C?
- How do we implement high-level programming constructs?

Because learning changes everything.®

www.mheducation.com