ECE 2400 / ENGRD 2140
Computer Systems Programming
Course Overview

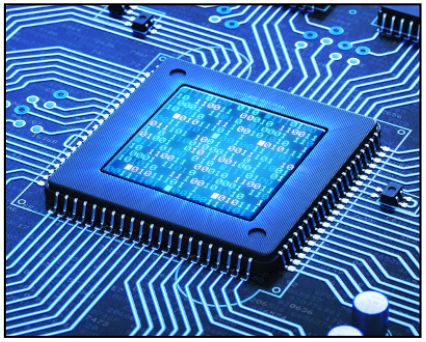
Christopher Batten

School of Electrical and Computer Engineering
Cornell University

`http://www.csl.cornell.edu/courses/ece2400`

Application-Level Software

System-Level Software



ECE 2400 / ENGRD 2140 Computer Systems Programming

What is Computer Systems Programming?

Activity 1: Comparing Algorithms

Trends in Computer Systems Programming

Activity 2: Compiling C to Machine Instructions

Course Logistics

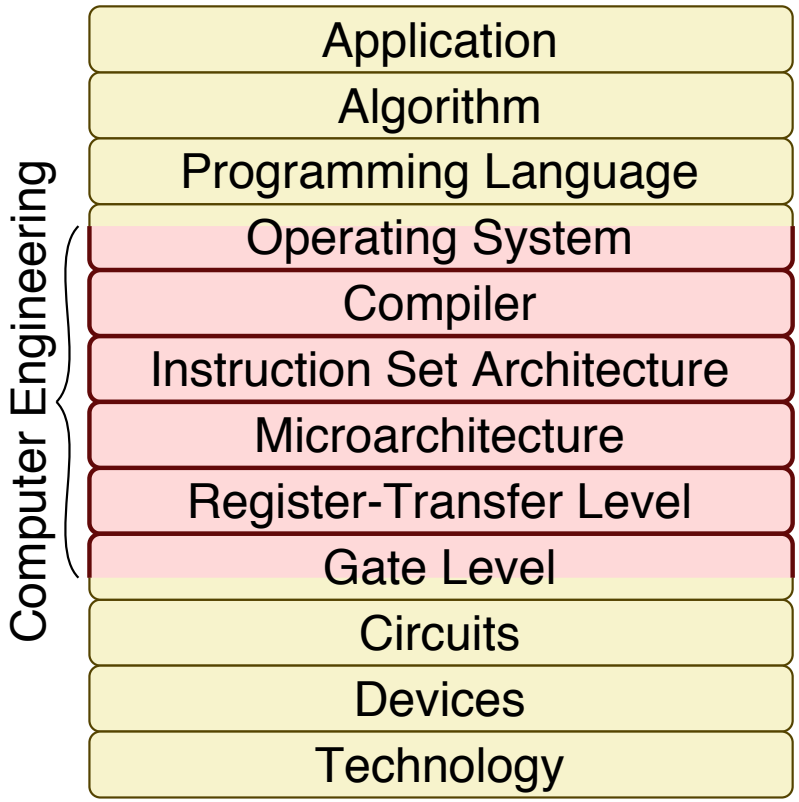
The Computer Systems Stack

Application

Gap too large to bridge in one step
(but there are exceptions,
e.g., a magnetic compass)

Technology

The Computer Systems Stack



Sort an array of numbers

2,6,3,8,4,5 -> 2,3,4,5,6,8

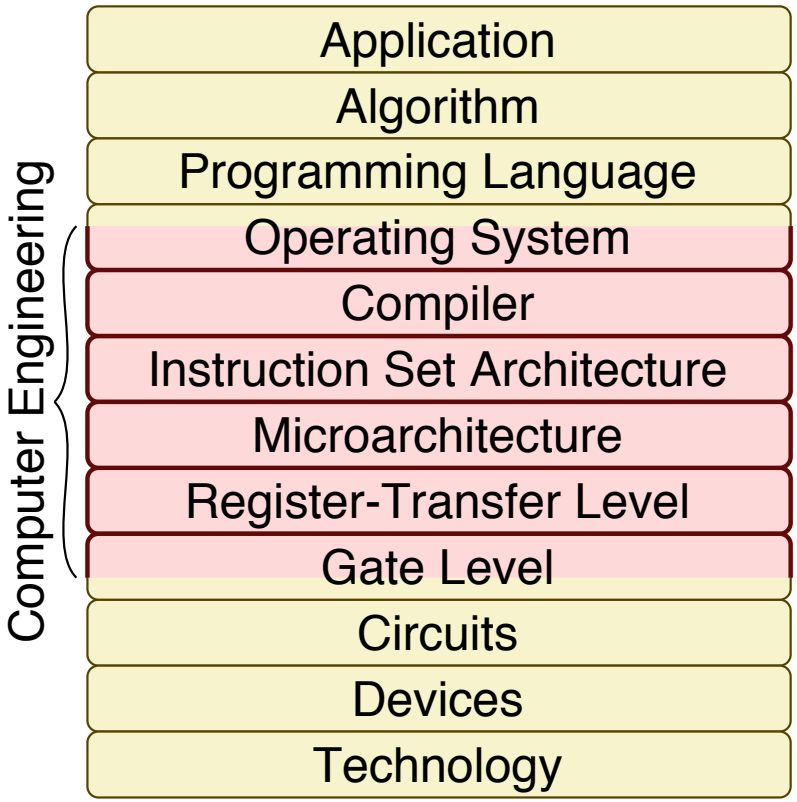
Out-of-place selection sort algorithm

1. Find minimum number in array
2. Move minimum number into output array
3. Repeat steps 1 and 2 until finished

C implementation of selection sort

```
void sort( int b[], int a[], int n ) {
  for ( int idx, k = 0; k < n; k++ ) {
    int min = 100;
    for ( int i = 0; i < n; i++ ) {
      if ( a[i] < min ) {
        min = a[i];
        idx = i;
      }
    }
    b[k]   = min;
    a[idx] = 100;
  }
}
```

The Computer Systems Stack



Mac OS X, Windows, Linux
Handles low-level hardware management

C Compiler
Transform programs into assembly

```
int a = b + c;
A[i] = a;
```

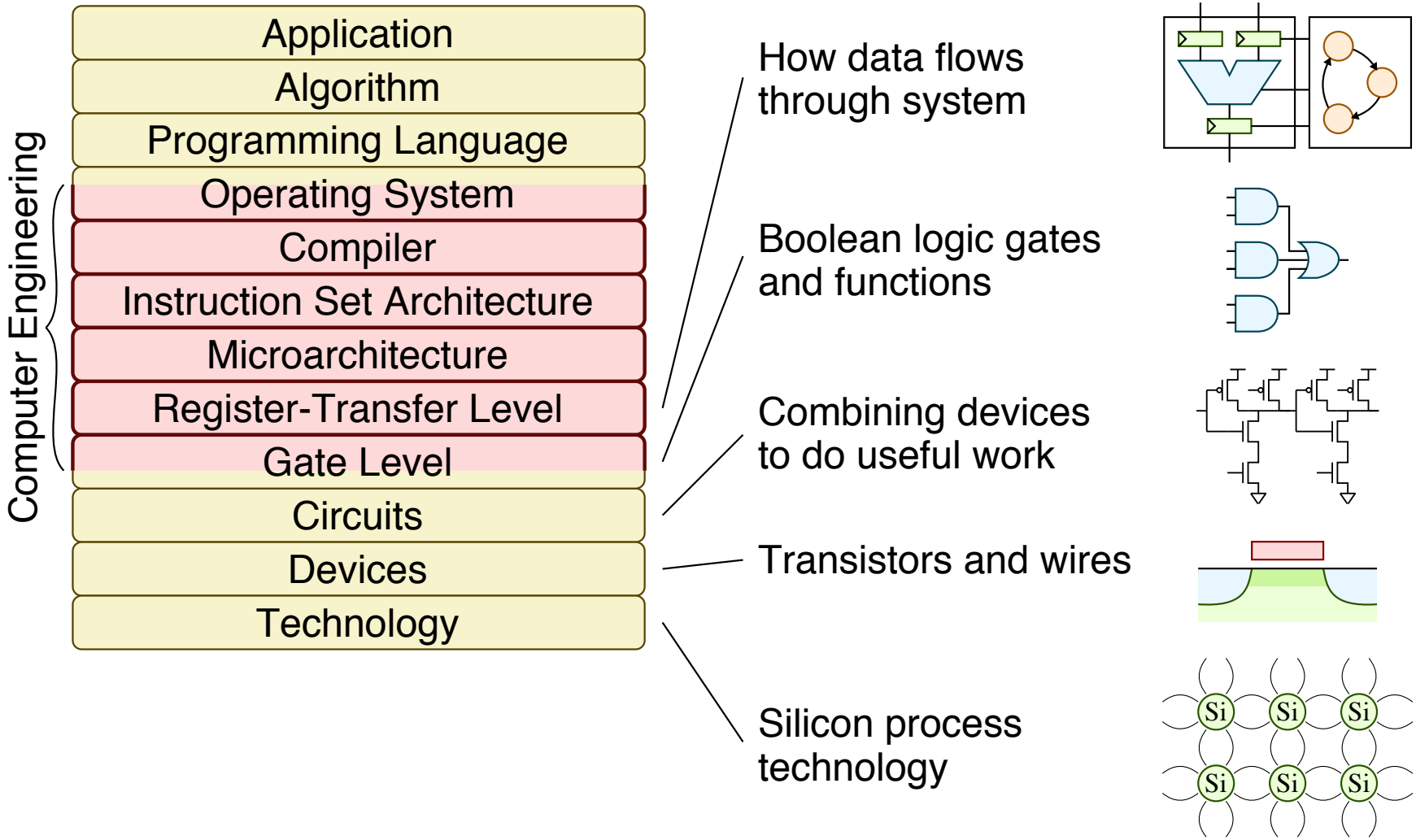
→

```
addu $t0, $t1, $t2
sw   $t0, 0($t3)
```

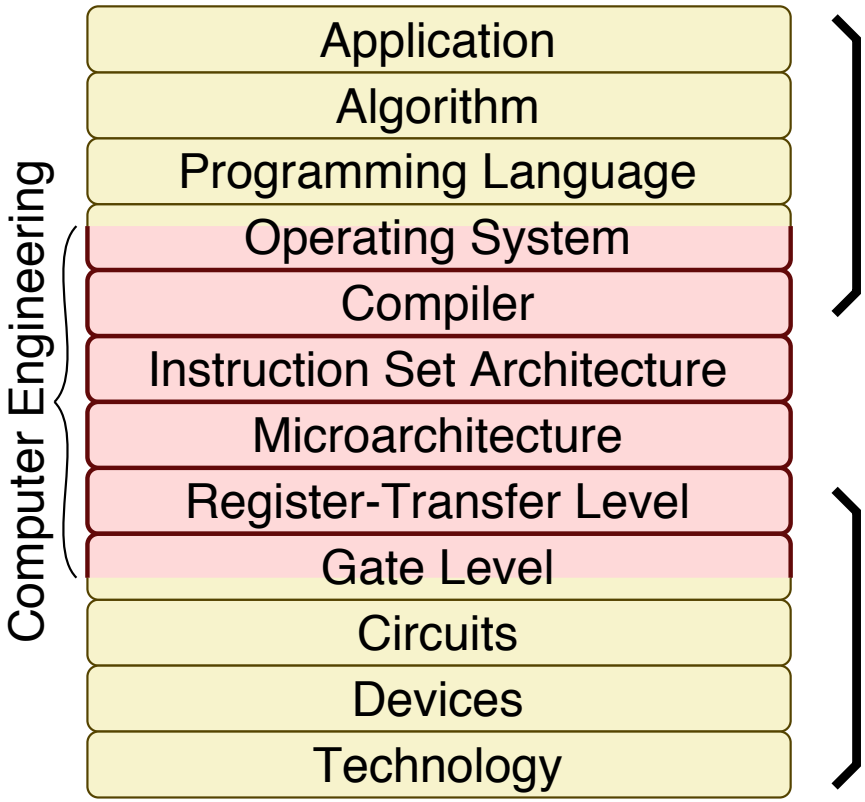
MIPS32 Instruction Set
Instructions that machine executes

```
blez $a2, done
move $a7, $zero
li   $t4, 99
move $a4, $a1
li   $a3, 99
lw   $a5, 0($a4)
```

The Computer Systems Stack



CS vs. Computer Engineering vs. EE



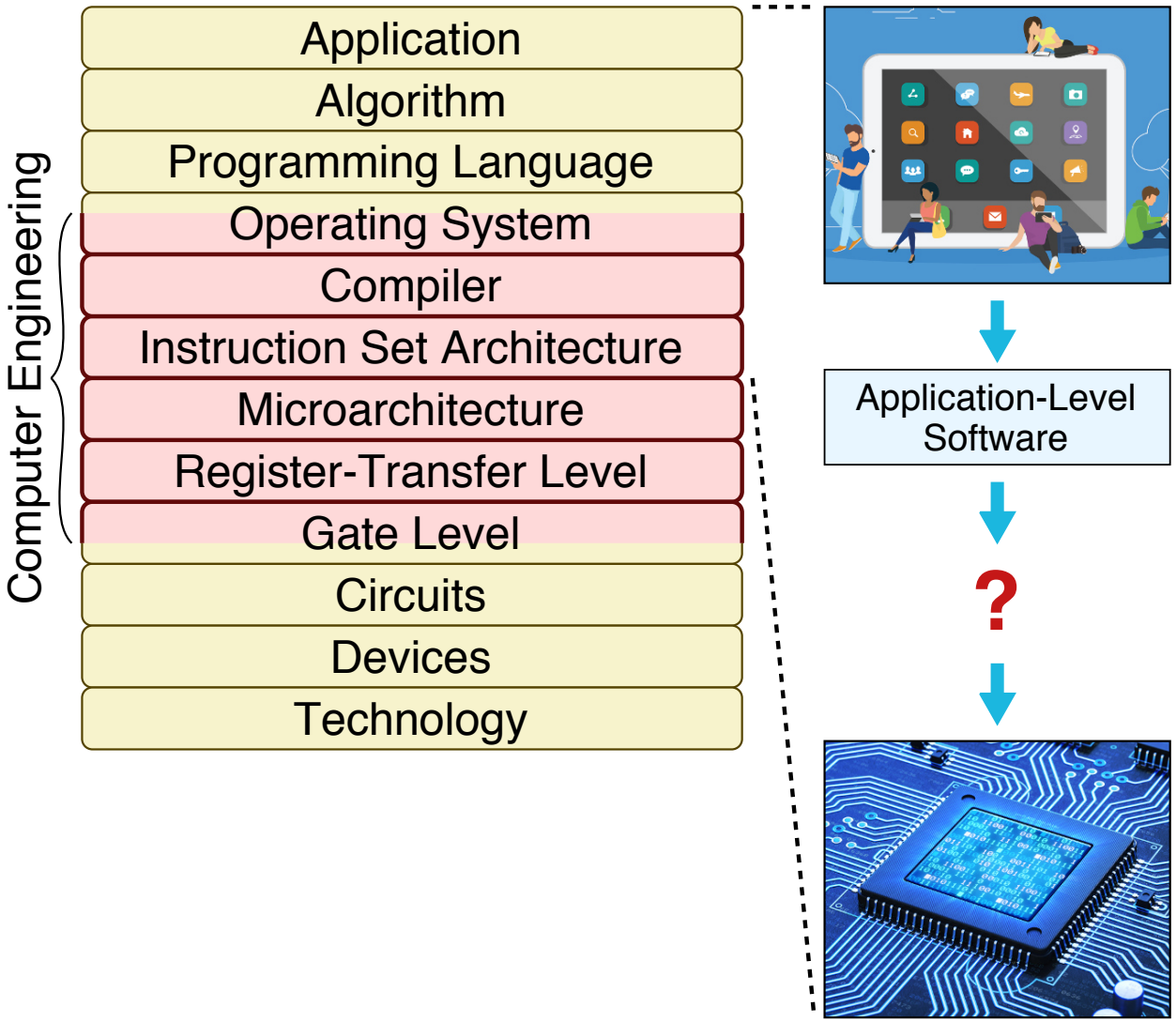
Traditional Computer Science

Computer Engineering is at the interface between hardware and software and considers the entire system

Traditional Electrical Engineering

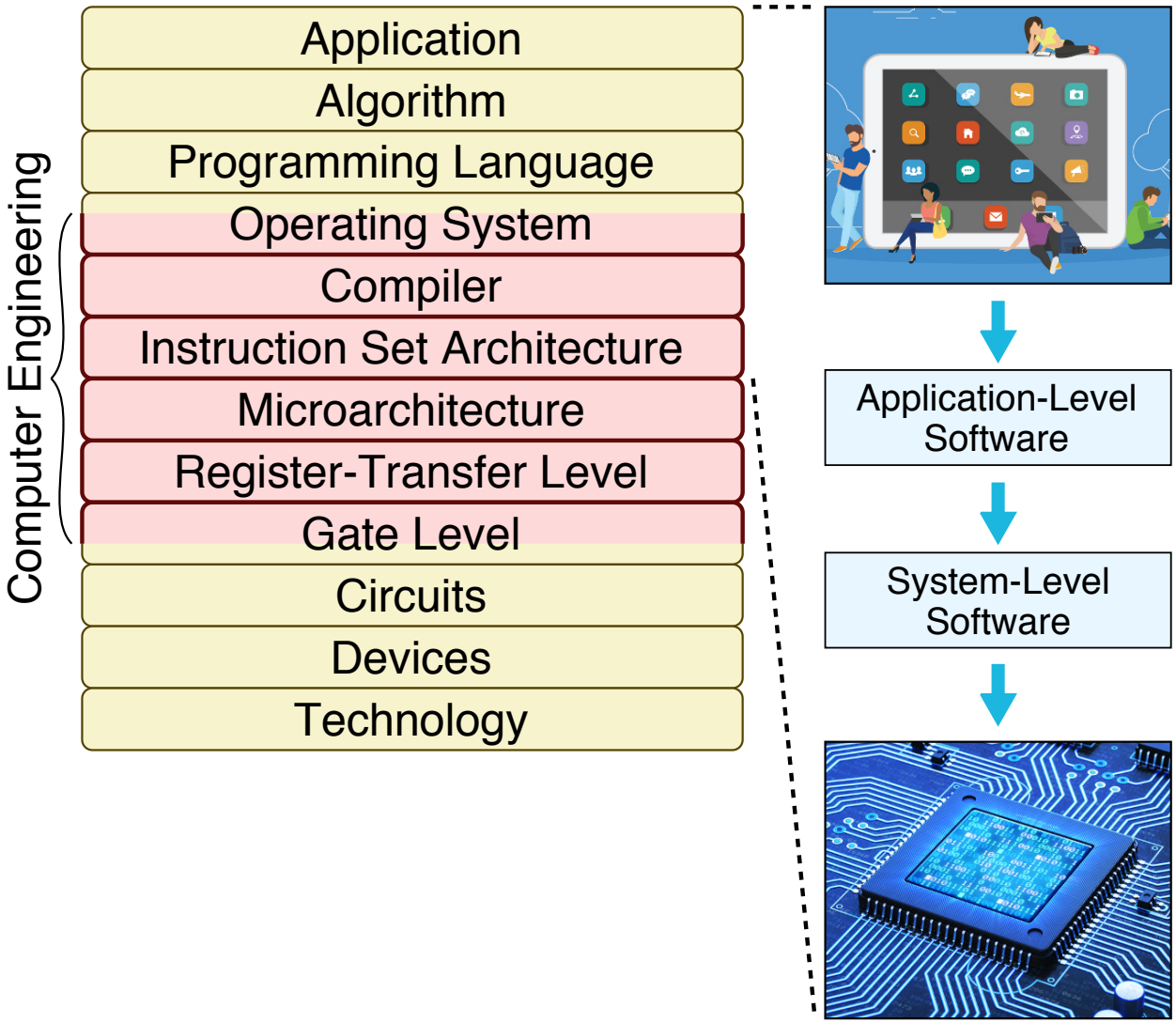
In its broadest definition, computer engineering is the development of the abstraction/implementation layers that allow us to execute information processing applications efficiently using available manufacturing technologies

Python for Application-Level Programming



- High-level, user-facing software
- Enable productively developing applications that provide new functionality to users
- Enable productively collecting, analyzing, visualizing data
- Sometimes called a productivity-level language

C/C++ for System-Level Programming

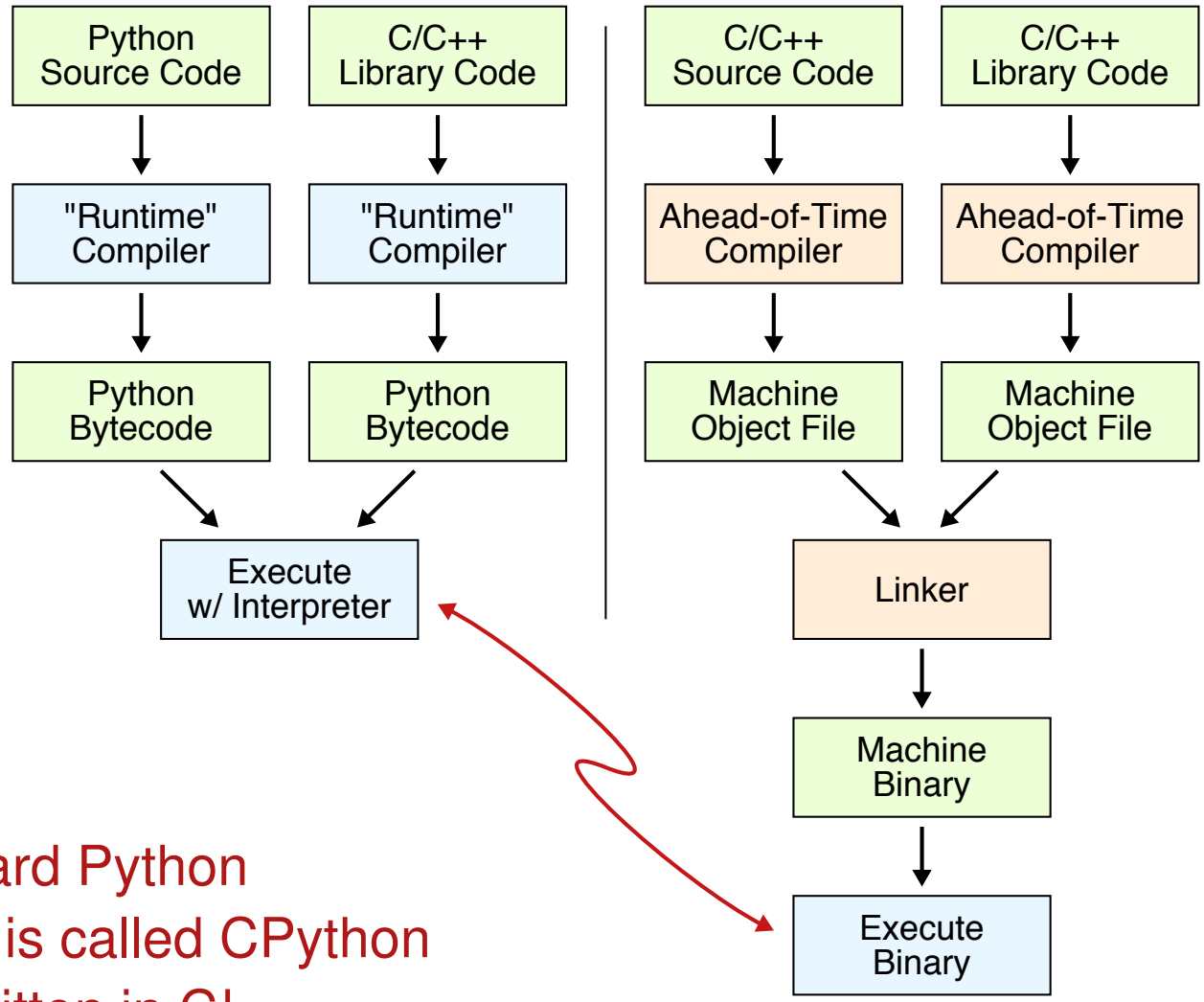


- Connects application software to the low-level computer hardware
- Enables carefully managing performance and resource constraints
- Sometimes called an efficiency-level language

Dynamically Interpreted vs. Statically Compiled

```
def min(a,b):
  if a < b:
    c = a
  else
    c = b
  return c
```

```
LOAD_FAST
LOAD_FAST
COMPARE_OP
POP_JUMP_IF_F
LOAD_FAST
STORE_FAST
JUMP_FORWARD
LOAD_FAST
STORE_FAST
LOAD_FAST
RETURN_VALUE
```



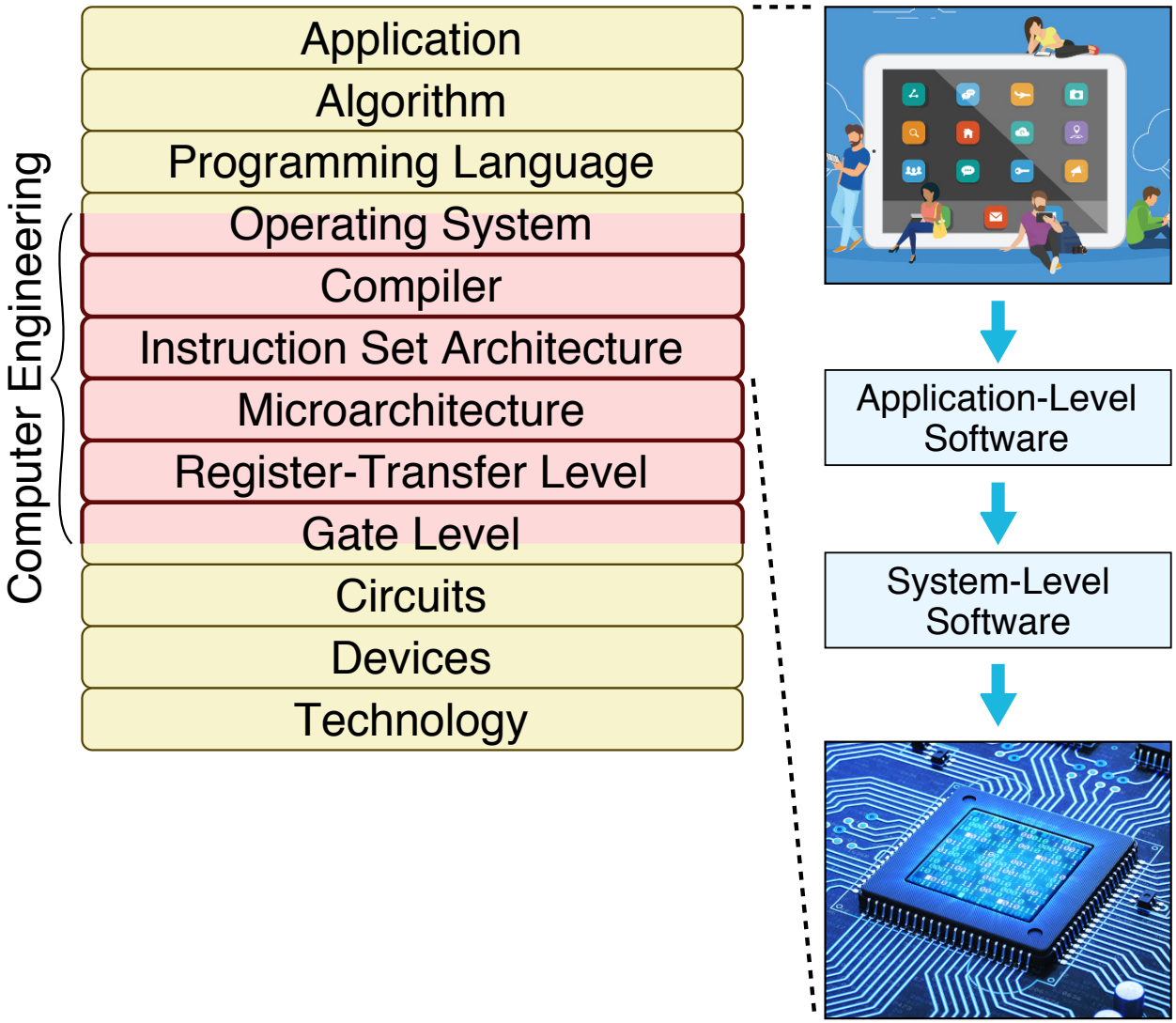

```
int min( int a,
         int b )
{
  int c;
  if ( a < b )
    c = a;
  else
    c = b;
  return c;
}
```

```
pushq %rbp
movq  %rsp,%rbp
cmpl  %esi,%edi
cmovl %edi,%esi
movl  %esi,%eax
popq  %rbp
retq
```

```
1010101
100100010001001
11100111110111
111101001110111
100010011111000
1011101
11000011
```

The standard Python interpreter is called CPython and it is written in C!

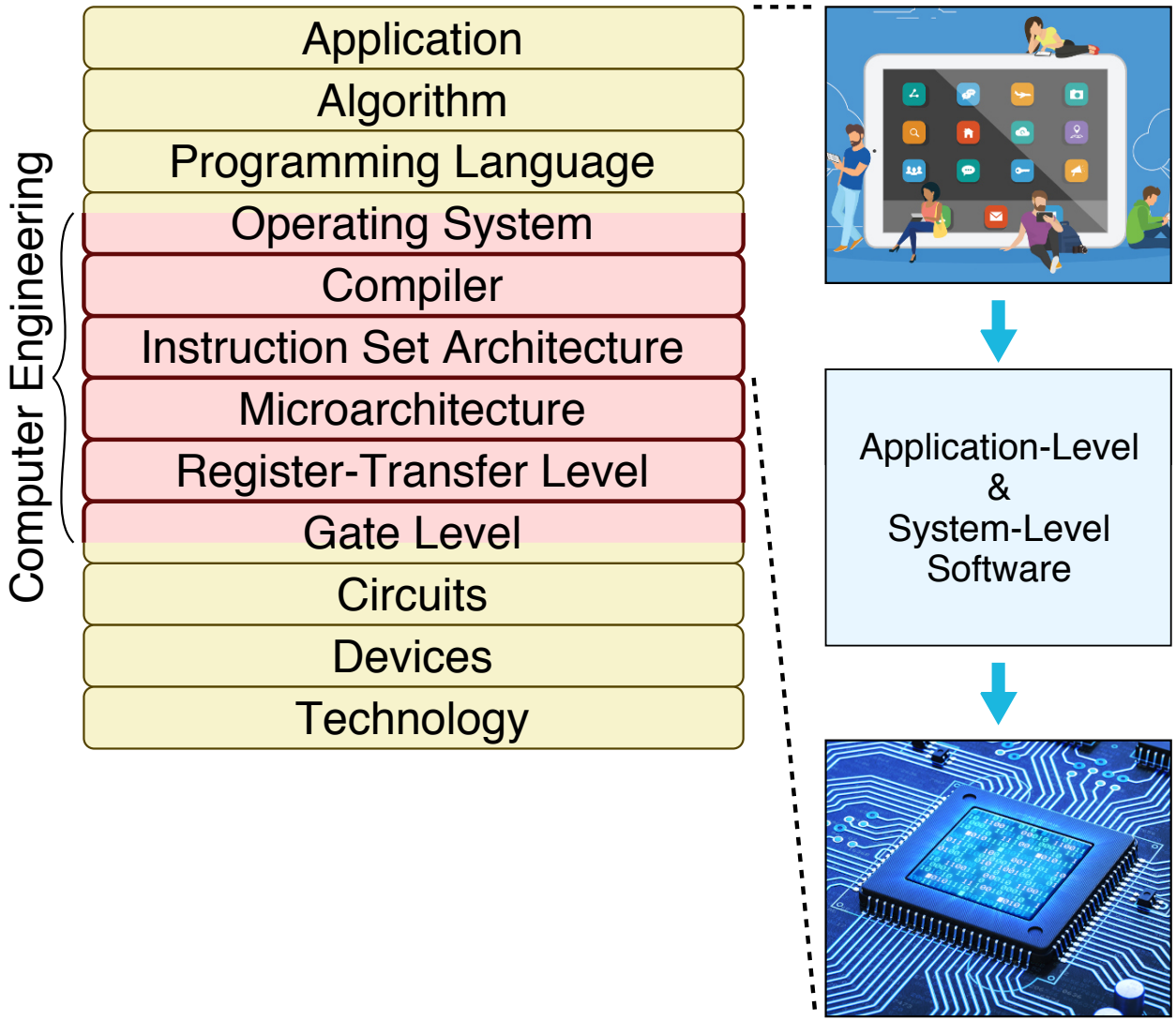
Computer Systems Programming is Diverse

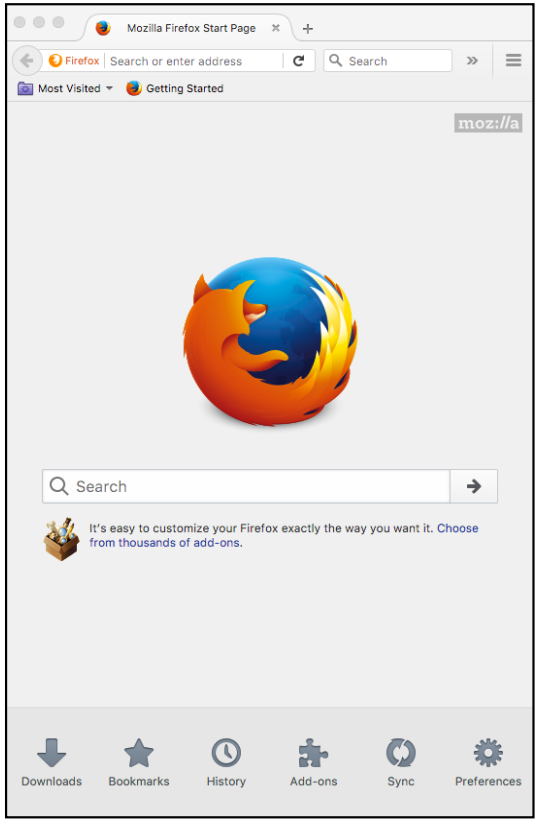


- Python, MATLAB
- Ruby, Javascript
- SQL, LINQ
- NumPy
- GUI frameworks

- Interpreters
- Compilers
- Databases
- Numerical libraries
- Operating systems
- Embedded control

Aside: C/C++ for Application-Level Software





A Tale of Two Programming Languages

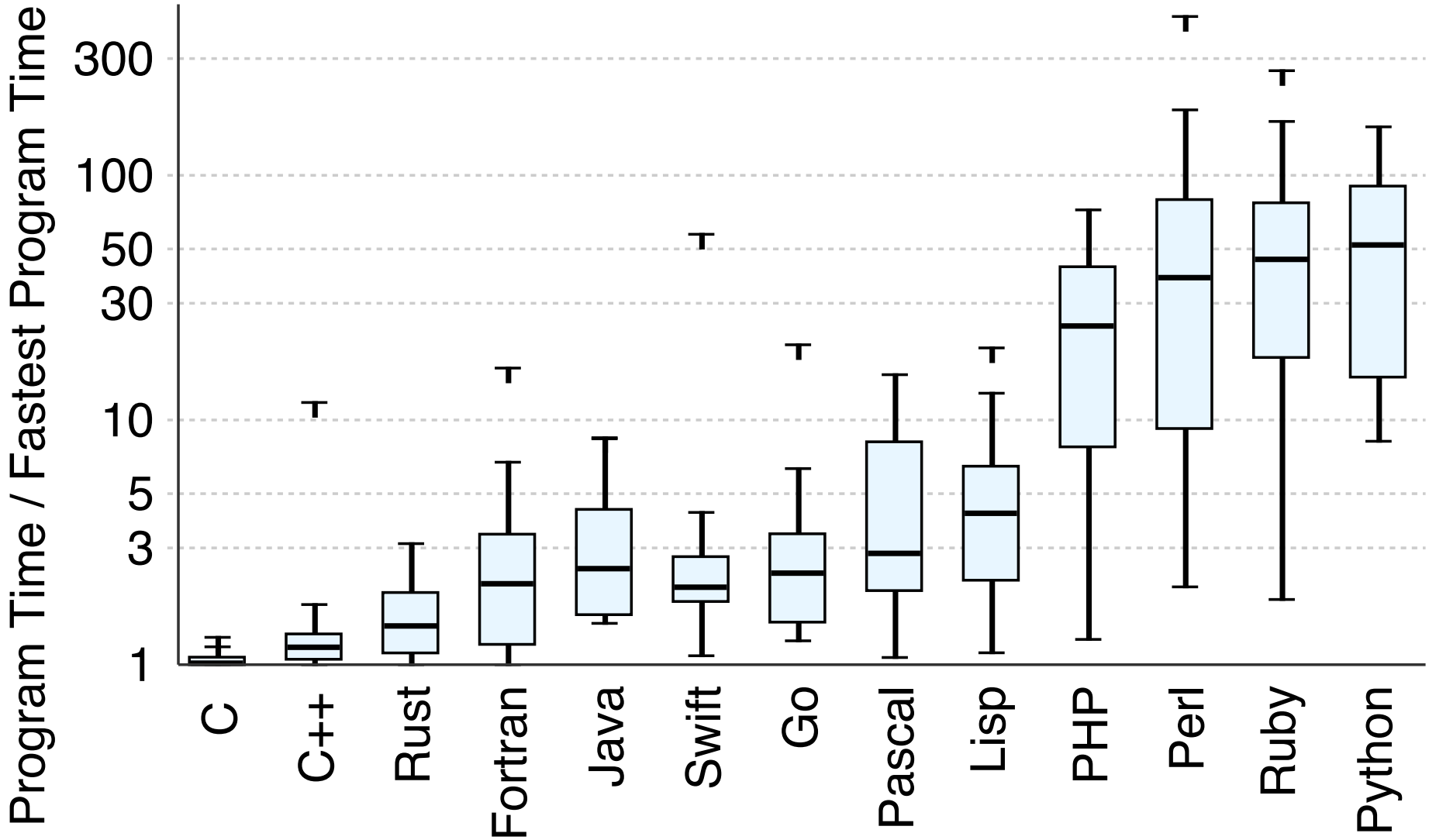
Python Programming Language

- Introduced: 1991
- Most of the machine details are hidden from programmer
- Programmer gives up some control for improved productivity
- Easily supports multiple programming paradigms
- Extensive standard library is included
- Slow and memory inefficient

C/C++ Programming Language

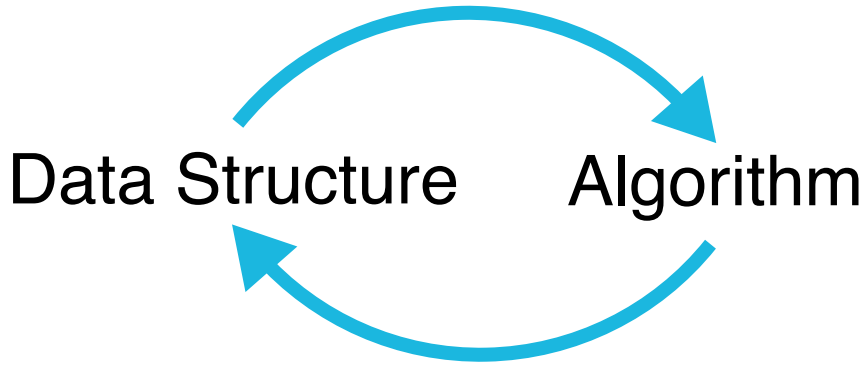
- Introduced: 1972(C), 1979(C++)
- Most of the machine details are exposed to the programmer
- Programmer is in complete control for improved efficiency
- C++ easily supports multiple programming paradigms
- More limited standard library is included
- Fast and memory efficient

Comparing the Performance of Python vs. C/C++



Program = Data Structure + Algorithm

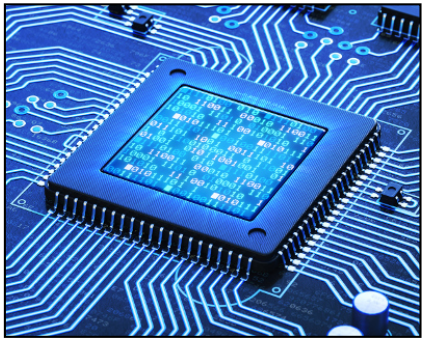
While this course covers C/C++ and system-level programming, this course also builds off of your prior programming experience to further develop your understanding of data-structures and algorithms



- **Data Structure:** Way of efficiently organizing and storing data along with methods for accessing and manipulating this data
- **Algorithm:** Clear set of steps to solve any problem instance in a particular class of problems

Application-Level Software

System-Level Software



ECE 2400 / ENGRD 2140 Computer Systems Programming

What is Computer Systems Programming?

Activity 1: Comparing Algorithms

Trends in Computer Systems Programming

Activity 2: Compiling C to Machine Instructions

Course Logistics

Activity 1: Comparing Algorithms

- **Application:** Sort 16 numbers
- **Activity Steps**
  - 1. Half the class will use Algorithm A, half uses Algorithm B
  - 2. When instructor starts timer, flip over worksheet
  - 3. Sort 16 numbers using assigned algorithm
  - 4. Lookup when completed and write time on worksheet
  - 5. Raise hand
  - 6. When everyone is finished, then analyze data
- **Algorithm A**

```
repeat 16 times
  find smallest number not crossed off in input list
  copy smallest number to next open entry in output list
  cross smallest number off input list
```

Activity 1: Comparing Algorithms

- **Algorithm B**

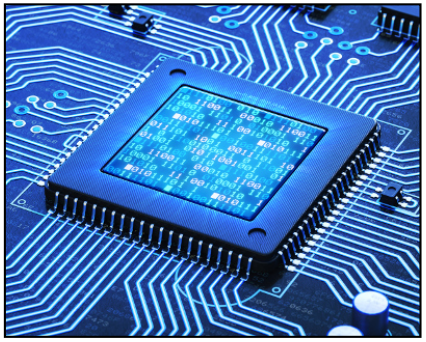
```
repeat 8 times, once for each pair in column 1
  copy smallest into next open entry in next column
  copy largest into next open entry in next column

repeat 4 times, once for group of 4 in column 2
  repeat 4 times
    compare top two numbers not crossed off in both groups
    copy smallest number to next open entry in next column
    cross smallest number off input list

... and so on ...
```

Application-Level Software

System-Level Software



ECE 2400 / ENGRD 2140 Computer Systems Programming

What is Computer Systems Programming?

Activity 1: Comparing Algorithms

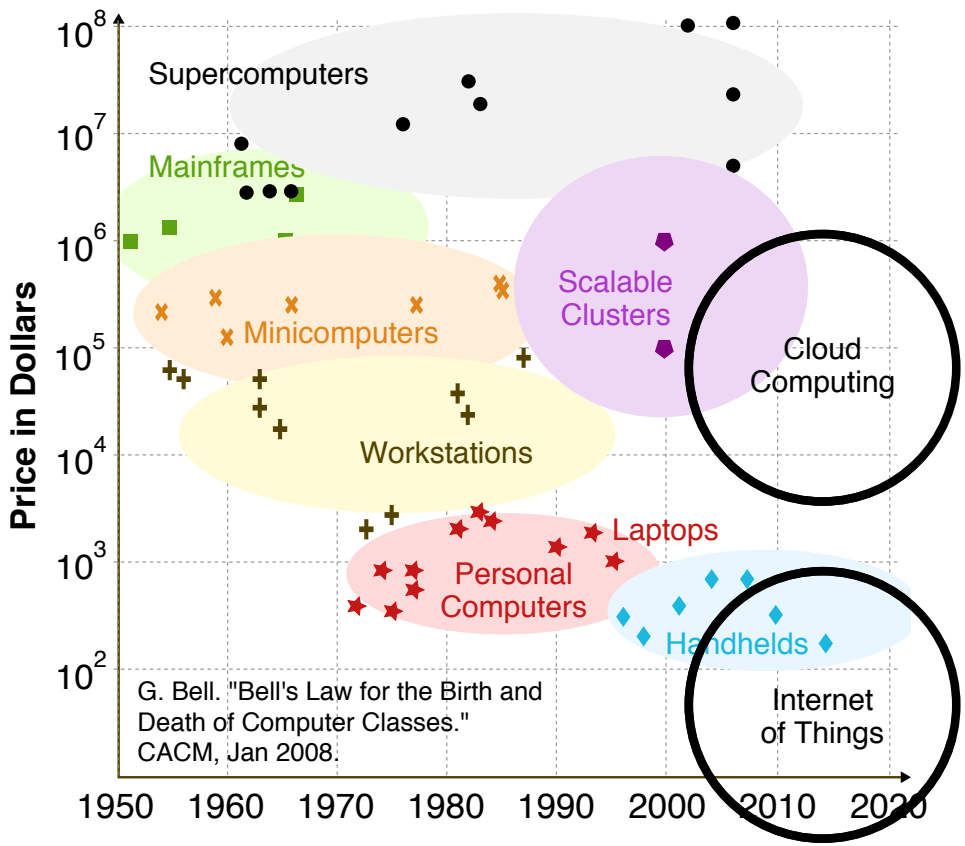
Trends in Computer Systems Programming

Activity 2: Compiling C to Machine Instructions

Course Logistics

Trend towards IoT and Cloud w/ Novel Hardware

Roughly every decade a new, smaller, lower priced computer class forms based on a new programming platform resulting in entire new industries



Cloud Computing

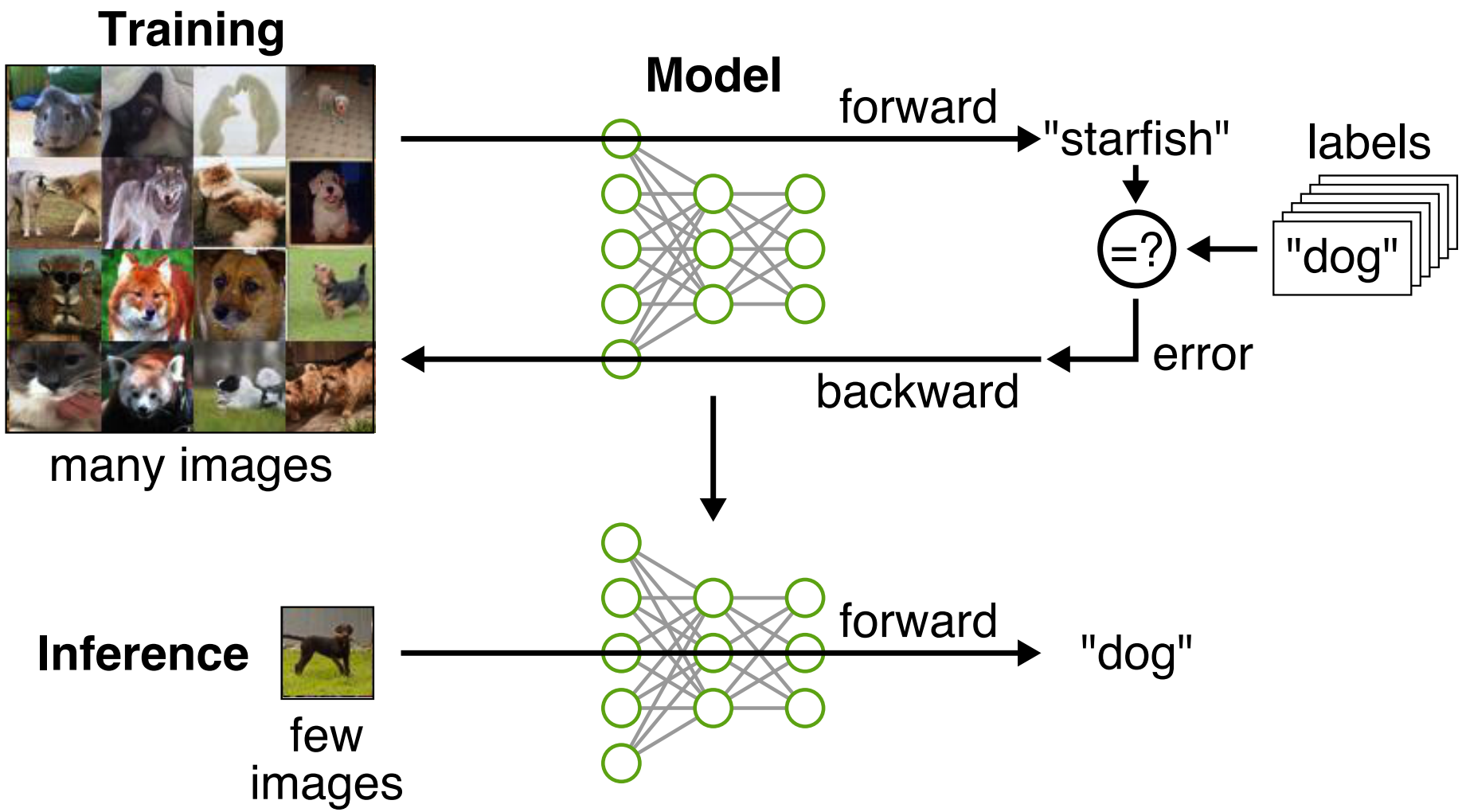
- Often requires low-latency, high-throughput to meet overall application requirements
- Increasingly w/ specialized HW

Internet-of-Things

- Very limited resource constraints (e.g., energy, memory)
- Requires carefully managing these resources to meet overall application requirements
- Increasingly w/ specialized HW

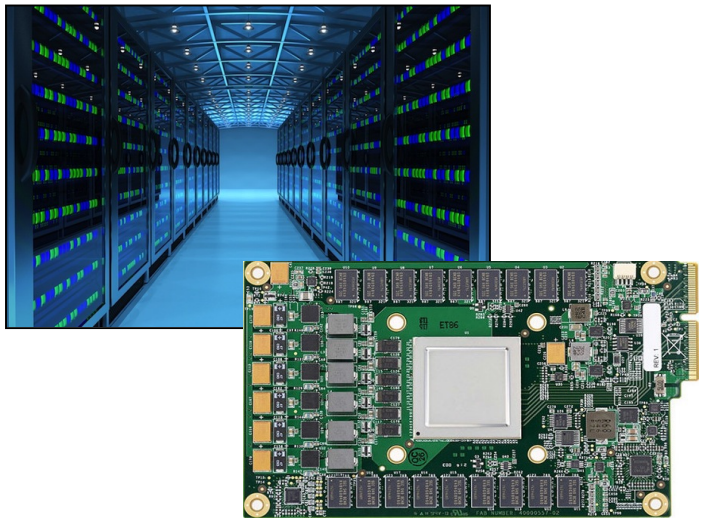
Example Application: Image Recognition

Machine Learning (ML): Training vs. Inference

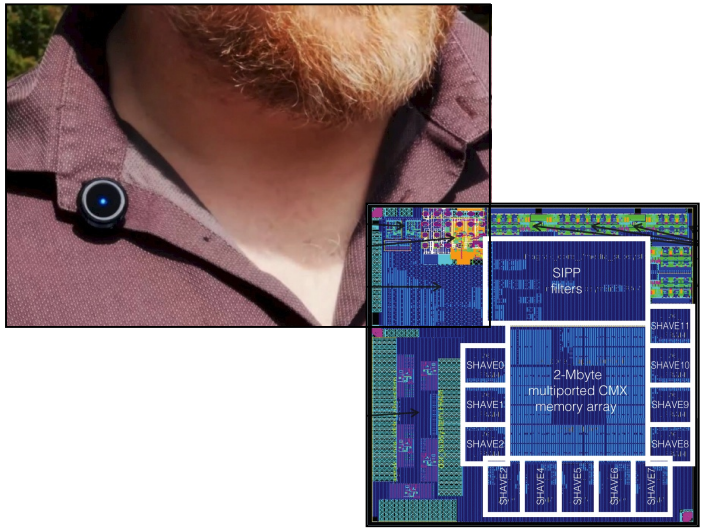


Computer Systems Programming in ML

Cloud Computing



Internet of Things



Google TPU

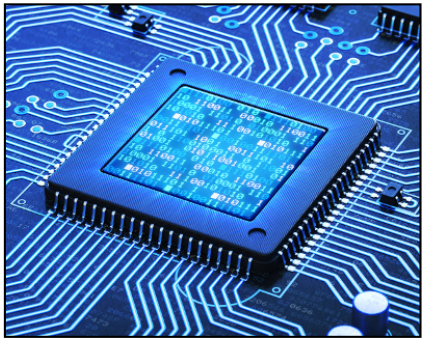
- Training is done using the TensorFlow C++ framework
- Training can take weeks
- Google TPU is custom chip
- High-level ML frameworks use C++ under the hood

Movidius Myriad 2

- Custom chip for ML on embedded IoT devices
- Carefully crafted C/C++ ML libraries for inference
- Embedded control also in C/C++

Application-Level Software

System-Level Software



ECE 2400 / ENGRD 2140 Computer Systems Programming

What is Computer Systems Programming?

Activity 1: Comparing Algorithms

Trends in Computer Systems Programming

Activity 2: Compiling C to Machine Instructions

Course Logistics

https://gcc.godbolt.org

Compiler Explorer - C++ | Christopher

Secure | https://gcc.godbolt.org/#

Compiler Explorer C++ Editor Diff View More Share Other

C++ source #1

```
1  // Type your code here, or load an example.
2  int square(int num) {
3      return num * num;
4  }
```

x86-64 gcc 7.2 (Editor #1, Compiler #1)

x86-64 gcc 7.2 | Compiler options...

11010 .LX0: .text // \s+ Intel

```
1  square(int):
2    push rbp
3    mov rbp, rsp
4    mov DWORD PTR [rbp-4], edi
5    mov eax, DWORD PTR [rbp-4]
6    imul eax, DWORD PTR [rbp-4]
7    pop rbp
8    ret
```

g++ (GCC-Explorer-Build) 7.2.0- 424ms

https://gcc.godbolt.org

- Try entering this example from the textbook and examine the corresponding machine instructions.

```
int myFunction( int x, int y )
{
  int z = x - 2*y;
  return z * x;
}
```

- ECE 2300 uses a MIPS-like architecture. Try choosing MIPS gcc 5.4 (el) from the drop-down menu.

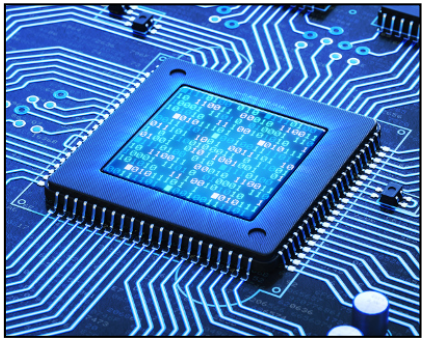
- Now try this example from earlier in today's lecture.

```
int min( int a, int b )
{
  int c;
  if ( a < b )
    c = a;
  else
    c = b;
  return c;
}
```

- Try entering `-O3` into the compiler options text box.

Application-Level Software

System-Level Software



ECE 2400 / ENGRD 2140 Computer Systems Programming

What is Computer Systems Programming?

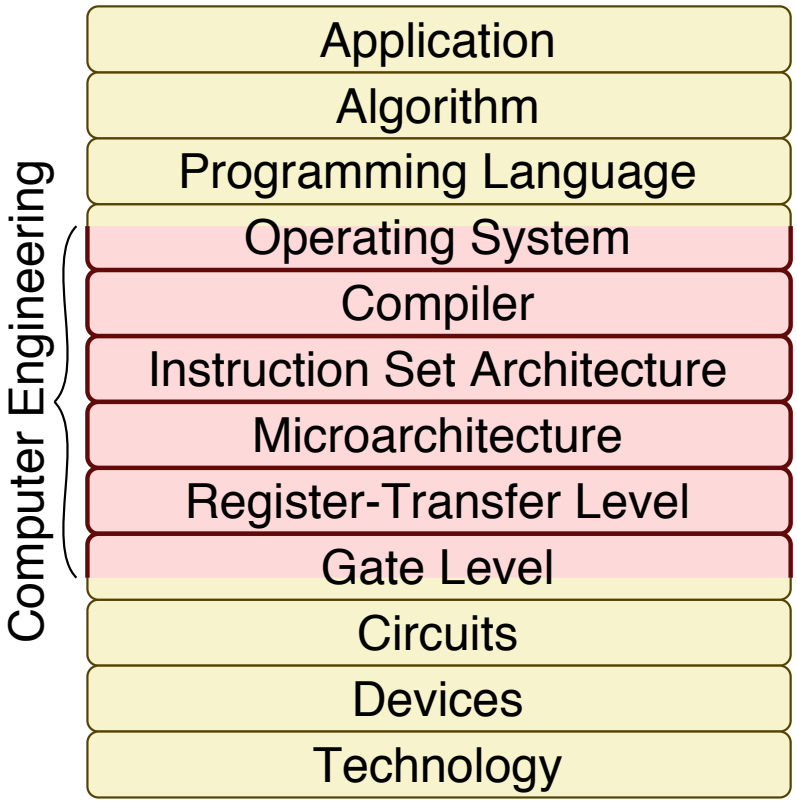
Activity 1: Comparing Algorithms

Trends in Computer Systems Programming

Activity 2: Compiling C to Machine Instructions

Course Logistics

ECE 2400 Within the Engineering Curriculum



CS 1110 / CS 1112 Intro to Computing

- ECE 2400 Computer Systems Programming
  - ECE 3140 Embedded Systems
    - ECE 4760 Design with Microcontrollers
    - ECE 4750 Computer Architecture
- ECE 2300 Digital Logic & Computer Org

ECE 2400 is also an ENGRD and thus satisfies the engineering distribution requirement

ECE 2400 can be an excellent way to generally incorporate programming into your non-ECE engineering curriculum

Course Structure

- **Part 1: C Programming and Basic Data Structures & Algorithms**
  - recursion, types, pointers, arrays, dynamic allocation, computational cost, abstract data types, lists, stacks, queues, sets, maps, sequence sorting, sequence alignment
- **Part 2: C++ Programming and Advanced Data Structures & Algorithms**
  - transition from C to C++, object-oriented programming, template meta-programming, binary search trees, priority queues, hash tables, graphs
- **Part 3: Systems Programming in the UNIX Environment**
  - standard I/O, processes, threads
- **Format**
  - lectures, optional discussion section, short in-class quizzes, readings, programming assignments, two prelim exams, final exam

Programming Assignments

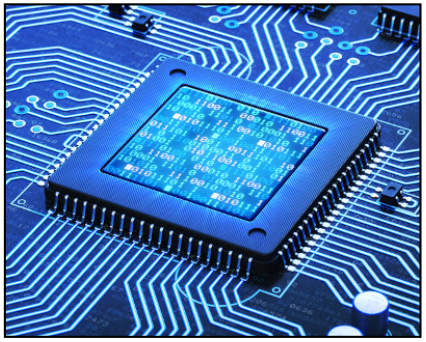
- **PA1–4: Basics**
  - PA1: Complex math functions
  - PA2: Cracking passwords
  - PA3: Searching and sorting
  - PA4: RPN calculator
- **PA5–6: System Software**
  - PA5: In-memory cache for distributed systems
  - PA6: Financial trading system
- **Every programming assignment involves**
  - C/C++ programming
  - Performance measurement
  - Short report

Frequently Asked Questions

- I have not taken CS 1110 nor CS 1112, can I take this class?
  - We assume some basic programming experience, discuss with instructor
- **ECE Majors** – How does ECE 2400 satisfy degree requirements?
  - ECE 2400 can count as your second ENGRD course
  - ECE 2400 can count as an outside-ECE technical elective
  - ECE 2400 satisfies the ECE advanced programming requirement
- **CS Majors** – Can I use ECE 2400 in place of CS 2110?
  - No
- **Other Majors** – How does ECE 2400 satisfy degree requirements?
  - ECE 2400 can count as one of your two required ENGRD courses
- Should I take both ECE 2400 and CS 2110?
  - Maybe? Maybe not?

Application-Level Software

System-Level Software



Take-Away Points

- Computer systems programming involves developing software to connect the low-level computer hardware to high-level, user-facing application software and usually requires careful consideration of performance and resource constraints
- We are entering an exciting era where computer systems programming will play a critical role in enabling both cloud computing and the internet-of-things