

# Finale

---

CS 2130: Computer Systems and Organization 1

May 1, 2023

# Announcements

- Homework 10 due tonight at 11pm
  - Limited number of submissions, test your code before submitting
  - Can submit through Wednesday at 11pm
- Review session in lab tomorrow
  - Labs are near capacity
- Final Exam: Thursday May 4, 7-10pm, Chem 402
  - Cumulative, see practice tests
- Remember to fill out course evaluations

## Common Memory Problems (from reading)

- Memory leak
- Uninitialized memory
- Accidental cast-to-pointer
- Wrong use of 'sizeof'
- Unary operator precedence mistakes
- Use after free
- Stack buffer overflow
- Heap buffer overflow
- Global buffer overflow
- Use after return
- Uninitialized pointer
- Use after scope

# Vulnerabilities

# Vulnerabilities

**Vulnerability:** a program for which something like this could happen (security holes)

- Ex: stack buffer overflow possibility
- Not necessarily malicious (like when we talked about backdoors)

**Exploit:** a way to use a vulnerability or backdoor that has been created

- Ex: the magic long word to type into our program

# Vulnerabilities

Anytime you can modify memory the programmer did not expect you to be able to modify, there's something you can do to give yourself power or rights the programmer didn't mean to give you

## Common Memory Problems (from reading)

- Memory leak
- Uninitialized memory
- Accidental cast-to-pointer
- Wrong use of 'sizeof'
- Unary operator precedence mistakes
- Use after free
- Stack buffer overflow
- Heap buffer overflow
- Global buffer overflow
- Use after return
- Uninitialized pointer
- Use after scope



# Vulnerabilities

What should you do when you find a vulnerability?

Good practices when finding a vulnerability:

1. Tell the owner
2. Wait (a reasonable amount of time for a fix)
3. Publish





# Where have we been?

# Where have we been?



# Where have we been?



# Where have we been?

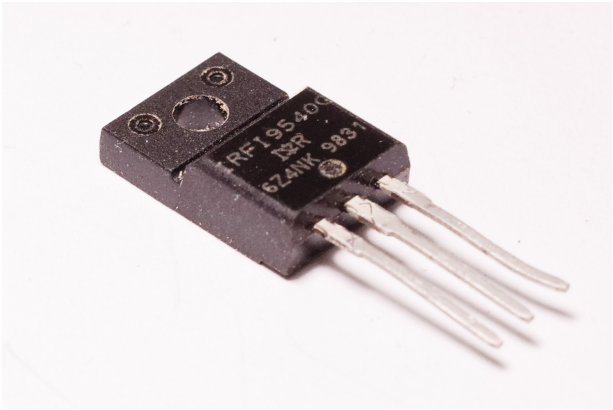
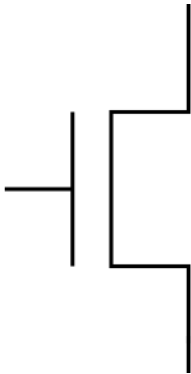




# Where have we been?

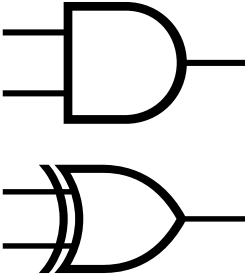


# Where have we been?

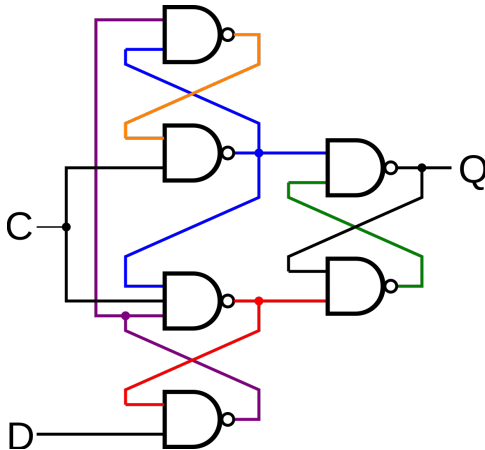
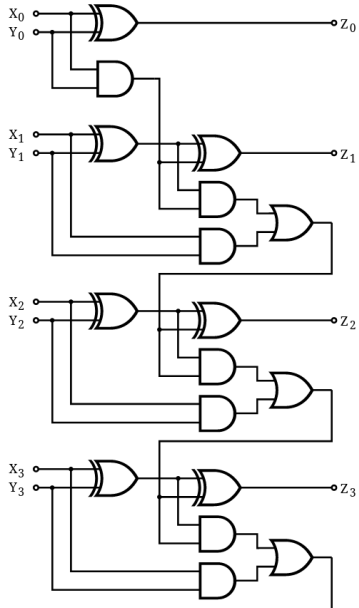


0 and 1

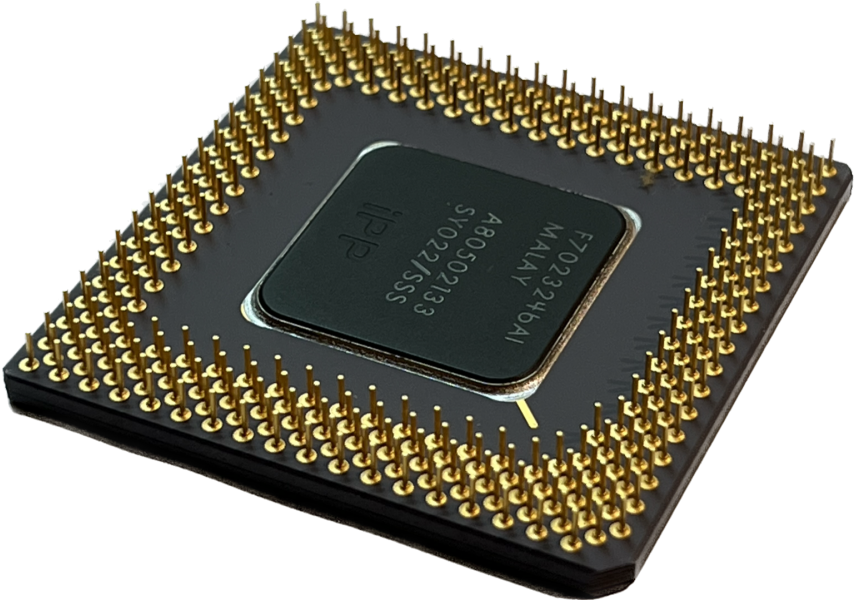
# Where have we been?



# Where have we been?



# Where have we been?



# Where have we been?

```
0000000000000000 <main>:
 0: 55                push  %rbp
 1: 48 89 e5          mov   %rsp,%rbp
 4: 31 c0             xor   %eax,%eax
 6: c7 45 fc 00 00 00 00  movl  $0x0,-0x4(%rbp)
 d: c7 45 f8 03 00 00 00  movl  $0x3,-0x8(%rbp)
14: 48 c7 45 f0 04 00 00  movq  $0x4,-0x10(%rbp)
1b: 00
1c: 48 8d 4d f8       lea  -0x8(%rbp),%rcx
20: 48 89 4d e8       mov  %rcx,-0x18(%rbp)
24: 48 8d 4d f0       lea  -0x10(%rbp),%rcx
28: 48 89 4d e0       mov  %rcx,-0x20(%rbp)
2c: 48 8b 4d e8       mov  -0x18(%rbp),%rcx
30: 48 63 09          movslq (%rcx),%rcx
33: 48 89 4d d8       mov  %rcx,-0x28(%rbp)
37: 48 8b 4d e0       mov  -0x20(%rbp),%rcx
3b: 48 8b 09          mov  (%rcx),%rcx
3e: 89 4d d4          mov  %ecx,-0x2c(%rbp)
41: 5d                pop   %rbp
42: c3                retq
```

# Where have we been?

```
void swap(int *a, int *b) {  
    int tmp = *a;  
    *a = *b;  
    *b = tmp;  
}
```

# Where have we been?

Along the way:

- Interact with the terminal and SSH
- Learn basic command-line tools and editors
- Access command-line documentation
- Practice C and using the C standard library
- Learn how to debug with lldb and the address sanitizer
- Discuss related security and social topics



# Finale

We have covered a LOT

- Electricity on wires
- Transistors to gates (AND, OR, ...)
- Combined gates to make circuits
- Connected circuits and registers to build a 1-byte computer
- Wrote an ISA for that computer (1-byte instructions, Toy ISA)
- Expanded to x86-64 Assembly (saw the binary)
- Concluded with C (how it compiles and connects with Assembly)

Thanks for a great semester!