ECE/ENGRD 2300
Digital Logic & Computer Organization
Spring 2024

Course Overview

Zhiru Zhang
School of Electrical and Computer Engineering
Today’s Lecture

Part 1. About this course

Part 2. Digital abstraction
Digital Computers are Everywhere
Changing Every Aspect of Our Life

many more (to come)
Digital (Super)Intelligence?

Mar 12\textsuperscript{th}, 2016: Google’s AlphaGo defeated top Go player Lee Sedol

Source: [www.economist.com/science-and-technology/2016/03/12/showdown](www.economist.com/science-and-technology/2016/03/12/showdown)

AlphaGo training is reported to use 1920 CPUs, 280 GPUs, and additional TPUs
Digital (Super)Intelligence?

OpenAI’s new language generator GPT-3 is shockingly good—and completely mindless

The AI is the largest language model ever created and can generate amazing human-like text on demand but won’t bring us closer to true intelligence.

by Will Douglas Heaven

July 20, 2020

Training GPT-3 is estimated to cost at least $4.6 million

Q. Mario Klingemann @quasimondo

Repeating to @quasimondo

Maybe I should turn these into a book called "Artificial Thoughts of an Artificial Fellow" before people stop buying books altogether when #gpt3 makes them according to their personal demands.

"On Giving Unsolicited Advice"

drive.google.com/file/d/1Twez_k...

On Giving Unsolicited Advice
not by Jerome K. Jerome
GPT-3, Summer 2020

There are very few people who really want advice, and those few will pay liberally for it. The greater number of people who, when they want to give advice, don’t want to give it to anyone in particular, but they just want to unload their opinions. If you want to receive that kind of advice, and give it, and you want to do both of these things free, you should subscribe to the columns of "Fashionable Intelligence." That way you will find abundant opportunities of doing these things free of charge.

The trouble about unasked-for advice is that nine times out of ten it is likely to be dished out by people who have no right to give it, but who are very willing to give it all the same. There is a man I know who thinks he has a right to give advice to
The “Intelligent” Chatbot!

Try talking with ChatGPT, our new AI system which is optimized for dialogue. Your feedback will help us improve it.

Here’s how much it costs to run OpenAI’s ChatGPT chatbot per day
It might cost around $3 million per month for OpenAI to run ChatGPT.

We’ve trained a model called ChatGPT which interacts in a conversational way. The dialogue format makes it possible for ChatGPT to answer followup...
How Do We Build a Complex Computer System?

Computer system (iPad v1)

Circuit Board (PCB)
- ≈8 PCBs / system

Integrated Circuit (IC):
- .25M-16M devices
- ≈8-16 ICs / PCB

Module:
- 100K devices
- ≈8-16 modules / IC

Device: Transistor

Cell: 16-64 devices
- ≈1K-10K cells / module

Gate: 2-8 devices
- ≈2-16 gates / cell

Total 1-2B transistors

Scheme for representing information
Our Plan of Attack …

• Understand how things work, bottom-up

• Encapsulate our understanding using appropriate abstractions

• Develop hands-on experience by implementing digital circuits on a real FPGA board
Required Textbook

- Get 2\textsuperscript{nd} edition (MIPS, not the ARM version)
Class Resources

• **Course Website**
  – [https://www.csl.cornell.edu/courses/ece2300/](https://www.csl.cornell.edu/courses/ece2300/)
  – Lecture slides, syllabus, and other handouts

• **Ed Discussion**
  – Announcements and Q&A
  – Links to lecture recordings

• **CMSX: Course Management System**
  – Assignments and grades
  – Electronic submissions required
Teaching Assistants

- **PhD TAs**
  - Niansong Zhang (nz264)
  - Yulun Yao (yy665)

- **MEng TAs**
  - Christopher Bakhos (cmb524)
  - Crystal Shi (cys37)
  - Raleigh Clemens (rjc422)

- **Undergrad TAs**
  - Amy Le (aml399)
  - Bahaa Kotb (bmk89)
  - Mohammad Al-Labadi (maa366)
  - Nita Kattimani (nsk62)
  - Paige Shelton (pes99)
  - Rachel Yan (sy625)
  - Vanessa Fang (vf72)
  - Wei Zheng (wz328)
  - Vivian Fan (xf37)
  - Xiaoyu Liang (xl434)
  - Anjelica Bian (yb265)
  - Nicole Li (yl3558)
  - Zarif Karim (zk67)
  - Zach Jessup (zsj5)

To reach entire staff: <ece2300-staff@csli.cornell.edu>
Seeking Help After Class

• **Ed Discussion**
  – Questions on lectures, assignments, and labs
  – Monitored by instructor & TAs

• **Staff email**
  – Grading related questions to instructor & TAs

• **Instructor email**
  – Private matters/appointment

• **Office hours**
  – Instructor: Thursday, 4:30-6:00pm (Online)
  – TA office hours to be announced soon (In-person)
Grading

- Participation: 3%
- Quizzes: 5%
- Homework: 12%
- Labs: 30%
- Prelim 1: 14%
- Prelim 2: 16%
- Final: 20%
Participation (3%)

- Participating in-class activities
  - Asking & answering questions in class
  - Live challenges, polls, and bonus problems

- Contributing to online discussion forum
  - Posting questions & helping other students

- A rough rubric
  - Active = 3pts
  - Somewhat engaged = 2pts
  - Little impression = 0-1pt
Labs and Homework (42%)

• Labs (30%)
  – Five labs in total
  – Prelab: write-up of your (partial) design
  – Lab section: implement and test your design
  – Report (labs 3 & 4): write up your findings

• Homework (12%)
  – Eight problem sets in total
Exams and Quizzes (55%)

• **Prelims (30%)**
  – Tuesday, March 5, in class
  – Thursday, April 11, 7:30pm @ Phillips 101

• **Final Exam (20%)**
  – Date TBD

• **Quizzes (5%)**
  – You will need to answer pop quiz questions in most lectures, using itempool (https://itempool.com/ece2300/live)
    • Make-up quizzes can be arranged if you miss lectures due to legitimate reasons
  – Four lowest scores will be dropped
Important Policies

• **Late Policy**
  – We collect assignments the instant they are due
    • Late submissions = 0 points
    • Applies to homework, prelabs, and lab reports
  – Total **EIGHT slip days**, intended to cover minor illnesses or “crunch time”
    • At most TWO slip days for prelab
    • If you have a serious illness or family emergency, contact me

• **Regrade Policy**
  – Submit regrade form to course staff
    <ece2300-staff@csli.cornell.edu> within one week if you feel a grading mistake has been made
How to Do Well in This Class

• Attend every lecture and participate
• Read the book sections before class
• Keep up with the week to week assignments
• Seek help if necessary
Academic Integrity

- [http://cuinfo.cornell.edu/aic.cfm](http://cuinfo.cornell.edu/aic.cfm)

- Discussion of homework and lab concepts? YES

- Misrepresenting someone else’s work as your own is prohibited
  - Getting someone else’s work? NO
  - Sharing your work with others? NO
  - Finding solutions on the web? NO
  - Outsourcing lab/homework to AI? NO

- Buying or selling course materials to commercial vendors (including Internet sites)? NO
# Course Schedule

<table>
<thead>
<tr>
<th>Date</th>
<th>Lecture</th>
<th>Reading</th>
<th>Lab</th>
<th>HW/Exam</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tue 1/23</td>
<td><strong>1: Course Overview</strong> [slides] [syllabus]</td>
<td>1.1-1.4.2, 1.5-1.6.2, 2.1-2.3</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Thu 1/25</td>
<td><strong>2: Boolean Algebra</strong> [slides]</td>
<td>2.4-2.7</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Fri 1/26</td>
<td></td>
<td></td>
<td>Lab 1 out</td>
<td></td>
</tr>
<tr>
<td>Tue 1/30</td>
<td><strong>3: Combinational Logic Minimization</strong> [slides]</td>
<td>1.7</td>
<td>HW 1 out</td>
<td></td>
</tr>
<tr>
<td>Thu 2/1</td>
<td><strong>4: CMOS Logic</strong> [slides]</td>
<td>2.8</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Tue 2/6</td>
<td><strong>5: Combinational Building Blocks</strong> [slides]</td>
<td>3.1-3.2</td>
<td>HW 2 out</td>
<td></td>
</tr>
<tr>
<td>Wed 2/7</td>
<td></td>
<td>Due: Lab 1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Thu 2/8</td>
<td><strong>6: Sequential Logic: Clocks, Latches, FFs</strong> [slides]</td>
<td>4.1-4.5 (skip VHDL), 5.4</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Fri 2/9</td>
<td></td>
<td>Lab 2 out</td>
<td>Due: HW 1</td>
<td></td>
</tr>
<tr>
<td>Tue 2/13</td>
<td><strong>7: More Sequential Logic, Verilog</strong> [slides]</td>
<td>3.4, 4.6</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Wed 2/14</td>
<td></td>
<td>Due: Prelab 2a</td>
<td>HW 3 out</td>
<td></td>
</tr>
<tr>
<td>Thu 2/15</td>
<td><strong>8: Finite State Machines (FSMs) 1</strong> [slides]</td>
<td>4.9</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Fri 2/16</td>
<td></td>
<td>Due: HW 2</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Tue 2/20</td>
<td><strong>9: FSMs 2, FSMs in Verilog</strong> [slides]</td>
<td>2.9, 4.6, 4.9</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Complete schedule is on course website.
Course Content

- Binary numbers and logic gates
- Boolean algebra and combinational logic
- Sequential logic and state machines
- Binary arithmetic
- Memories
- Instruction set architecture
- Processor organization
- Caches and virtual memory
- Input/output
- Advanced topics
### Where This Course Sits in the “Stack”

<table>
<thead>
<tr>
<th>Applications</th>
</tr>
</thead>
<tbody>
<tr>
<td>Programming languages</td>
</tr>
<tr>
<td>System software (compilers, OS)</td>
</tr>
<tr>
<td>Instruction set architecture</td>
</tr>
<tr>
<td>Microarchitecture</td>
</tr>
<tr>
<td>Digital logic design</td>
</tr>
<tr>
<td>Circuits</td>
</tr>
<tr>
<td>Devices</td>
</tr>
<tr>
<td>Technology (atomic physics)</td>
</tr>
</tbody>
</table>

**Traditional CS**

**Computer Engineering**

**Traditional EE**

Lecture 1: 23
Binary Digital Systems

- **Digital** system: Finite number of values
- **Binary** (base 2) system: Use two states: 0 and 1

**Basic unit of information:** the *binary digit, or bit*
- Two values: 0 and 1

- 0 and 1 represented by voltages
- Key advantage: efficient circuits (cheap, small, fast, low power)
0 and 1 Don’t Have to be Exact

• 0 and 1 represented by voltage ranges (logic levels)

• Electronic circuits do not need to be perfect

• We can tolerate some noise and computers still work
Binary Encoding is Ubiquitous

• **Activity:** Look around and identify one item that can be represented using a binary digit
Can We Represent More Than 2 Values?

- **Yes – just use multiple bits**
  - A collection of 2 bits gives 4 possible values
    - 00, 01, 10, 11
  - A collection of 3 bits gives 8 possible values
    - 000, 001, 010, 011, 100, 101, 110, 111

- **A collection of n bits gives** $2^n$ **possible values**
Positional Number Representation

• Recall positional notation for decimal numbers

\[
329 \quad \text{base 10 (decimal)}
\]

\[
10^2 \quad 10^1 \quad 10^0
\]

\[
3 \times 100 + 2 \times 10 + 9 \times 1 = 329
\]

• Similar positional system for binary

\[
101 \quad \text{base 2 (binary)}
\]

\[
2^2 \quad 2^1 \quad 2^0
\]

\[
1 \times 4 + 0 \times 2 + 1 \times 1 = 5
\]
Logic Gates

- Logic gates are functions: take one or more binary inputs and produce a binary output.

<table>
<thead>
<tr>
<th>A</th>
<th>Y</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
</tr>
</tbody>
</table>

Truth Table

<table>
<thead>
<tr>
<th>A</th>
<th>B</th>
<th>Y</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

NOT Gate: NOT A, A, A’

AND Gate: A AND B, A•B

OR Gate: A OR B, A+B
Build a 1-Bit Adder

- Inputs: A, B and C\textsubscript{in} (carry-in)
- Outputs: S (sum) and C\textsubscript{out} (carry-out)

<table>
<thead>
<tr>
<th>A</th>
<th>B</th>
<th>C\textsubscript{in}</th>
<th>C\textsubscript{out}</th>
<th>S</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

XOR Gate

\[ A \cdot B' + A' \cdot B \]

<table>
<thead>
<tr>
<th>A</th>
<th>B</th>
<th>Y</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
</tbody>
</table>
Build a Multi-Bit Adder

abstraction

Lecture 1: 31
Build a Programmable Microprocessor
Build a Complete Computer
Before Next Class

• Read the syllabus!
• H&H 1.1-1.4.2, 1.5-1.6.2, 2.1-2.3

Next Time

Boolean Algebra