Lecture 1: Introduction to Digital Logic Design

CSE 140: Components and Design Techniques for Digital Systems

Diba Mirza
Dept. of Computer Science and Engineering
University of California, San Diego
What is inside your processor?
Logistics: Resources

Class website:
http://cseweb.ucsd.edu/classes/su15/cse140-a/index.html

- *Approx.* Syllabus
- Detailed schedule
- Readings
- Assignments
- Grading policy
- HW submissions (TED)
- Content/announcements through Piazza
- Grades will be posted on Grade Source
Logistics: Course Components

Grading (grade on style, completeness and correctness)
• 10% Online textbook exercises (zybook buy at $30)
• 10% Home works (4 in total, released via TED)
• 15% In class quizzes using clickers
• 5% In class participation using clickers. You can miss up to 2 classes without penalty
• 30% One midterm
• 30% Final Exam (Tues 7/28, 12:30pm to 1:50pm)
• Note: Final date and time is different from what is indicated in schedule of classes
• If more than 60% of the class fill out CAPE evaluations, lowest quiz score will be dropped
• Grades will be based on standard scale
Logistics: Textbooks

Required:

Online Textbook: Digital Design by F. Vahid
Use the zybook code: UCSDCSE140Summer2015 to subscribe and get the discounted $30 price.

Reference (recommended) texts:


• [Vahid]: Digital Design by Frank Vahid
In class we will use Clickers!

- Lets you vote on multiple choice questions in real time.
Information about TAs

*TAs and tutors

1. Anand Anand
2. Armaiti Ardeshiricham
3. Helen Femmel
4. Chao (Jack) Li
5. Timothy (Tim) Nguyen
6. Anjali Patel

Office hours and emails available on the course website (Check weekly)
Course Problems…Cheating

• What is cheating?
  – Studying together in groups is encouraged
  – Turned-in work must be completely your own.
  – Copying someone else’s solution on a HW or exam is cheating
  – Both “giver” and “receiver” are equally culpable

• Cheating on HW/ exams: in most cases, F in the course.
• Any instance of cheating will be referred to Academic Integrity Office
Motivation

• Microelectronic technologies have revolutionized our world: cell phones, internet, rapid advances in medicine, etc.

• The semiconductor industry has grown from $21 billion in 1985 to $268 billion in 2007.
The Digital Revolution

Integrated Circuit: Many digital operations on the same material

Vacuum tubes

ENIAC

Integrated Circuit

WWII

Stored Program Model

1949

1965

Moore’s Law

Exponential Growth of Computation
Building complex circuits

Processor Graphics
Core
Core
Core
Core
System Agent & Memory Controller
Including SMH, Display and Misc. I/O
Shared L3 Cache**
Memory Controller I/O

Transistor

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

+Vcc

Transistor Switches

Q = A+B

R2
Technology Trends: Moore’s Law

Since 1975, transistor counts have doubled every two years.
How do we handle complexity?

- Big idea: Coordination of many *levels of abstraction*
Scope: Overall Picture of CS140

Data Path Subsystem

- Memory File
- Pointer
- Select
- Mux
- ALU
- Memory
- Register
- Conditions

Control Subsystem

- Conditions
- Sequential machine
- Control
- CLK: Synchronizing Clock
Scope

• The purpose of this course is that we:
  – Learn the principles of digital design
  – Learn to systematically debug increasingly complex designs
  – Design and build digital systems
  – Learn what’s under the hood of an electronic component
## Scope

<table>
<thead>
<tr>
<th>Subjects</th>
<th>Building Blocks</th>
<th>Theory</th>
</tr>
</thead>
<tbody>
<tr>
<td>Combinational</td>
<td>AND, OR, NOT, XOR</td>
<td>Boolean Algebra</td>
</tr>
<tr>
<td>Logic</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Sequential</td>
<td>AND, OR, NOT, FF</td>
<td>Finite State Machine</td>
</tr>
<tr>
<td>Network</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Standard</td>
<td>Operators, Interconnects, Memory</td>
<td>Arithmetics, Universal Logic</td>
</tr>
<tr>
<td>Modules</td>
<td></td>
<td></td>
</tr>
<tr>
<td>System Design</td>
<td>Data Paths, Control Paths</td>
<td>Methodologies</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Combinational Logic Basics
What is a combinational circuit?

- No memory
- Realizes one or more functions
- What are the inputs?
- What are the outputs?

- Inputs and outputs can only have two discrete values:
  - Physical domain (usually, voltages) (0V, 5V)
  - Mathematical domain: Boolean variables (true or false)

Differentiate between different representations:
- physical circuit
- schematic
- mathematical
Representations of combinational circuits: The Schematic

- What is the simplest combinational circuit that you know?
Representations of combinational circuits: Truth Table

<table>
<thead>
<tr>
<th>AND</th>
<th>A</th>
<th>B</th>
<th>Y</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>
Representations of combinational circuits: Boolean Expression/Equation

\[ Y = AB \]

All three forms are equivalent!
Boolean Algebra

Similar to regular algebra but defined on sets with only three basic ‘logic’ operations:

1. Intersection: AND (2-input), Operator: .
2. Union: OR (2 input), Operator: +
3. Complement: NOT (1 input), Operator:
Q: If X and Y are Boolean variables with X=1, Y=0, what is X+X+Y?

A. 0
B. 1
C. 2
Boolean Algebra

Boolean operations satisfy the following laws:

• Commutative laws: $a+b=b+a$, $a\cdot b=b\cdot a$
• Distributive laws: $a+(b\cdot c)=(a+b)\cdot(a+c)$,
  \[ a \cdot (b+c)=a\cdot b+a\cdot c \]
• Identity laws: $a+0=a$, $a\cdot 1=a$
• Complement laws: $a+a'=1$, $a\cdot a'=0$
Boolean algebra and switching functions: Operators and Digital Logic Gates

Two-input operator
AND ($\cdot$)

<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>

AND

Two-input operator
OR ($+$)

<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>1</td>
</tr>
</tbody>
</table>

OR

One-input operator
NOT (Complement, ‘’)

<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>

NOT

0 dominates in AND
0 blocks the output
1 passes signal A

1 dominates in OR
1 blocks the output
0 passes signal A

0 dominates in AND
1 dominates in OR
So, what is the point of representing gates as symbols and Boolean expressions?

- Given the Boolean expression, we can draw the circuit it represents by cascading gates (and vice versa)
Next class

• Designing Combinational circuits