Digital Design & Computer Arch.

Lab 1 Supplement:
Drawing Basic Circuits

Frank K. Gürkaynak
Seyyedmohammad Sadrosadati
(Presentation by Aaron Zeller)
ETH Zurich

Spring 2024

5 March 2024

Lab Sessions

- Where?
  - On-site

Tuesday	Wednesday	Friday - 1	Friday - 2
HG E19	HG E19	HG D11	HG E19
HG E26.1	HG E26.1	HG D12	HG E26.1
HG E26.3	HG E26.3	HG E26.3	HG E26.3
HG E27	HG E27	HG E27	HG E27

DDCA Course Catalogue Web Page

Supplementary material presentations

- When? (Presentation happens **every week**)
  - Tuesday 16:15-18:00 Start in ML E12, then move to the labs
  - Wednesday 16:15-18:00 Start in HG E19, then move to the labs
  - Friday 08:15-10:00 Start in HG G1, then move to the labs
  - Friday 10:15-12:00 Start in HG G1, then move to the labs

Grading

- **10** labs, **30** points in total
- **70%** final exam grade + **30%** lab grade
- Grading Policy
  - In-class evaluation (70%) and *mandatory* lab reports (30%)
  - Labs are to be finished within 1 week after being announced
  - Late submissions incur a point deduction

General Information

- Read the lab manual carefully as it contains information which will save you time
- All labs are meant to be solved during the lab sessions
  - If you are stuck ask your TAs for help
- For questions
  - digitaltechnik@lists.inf.ethz.ch (Emails are sent to all TAs)
  - Moodle forum (per lab / assignment)

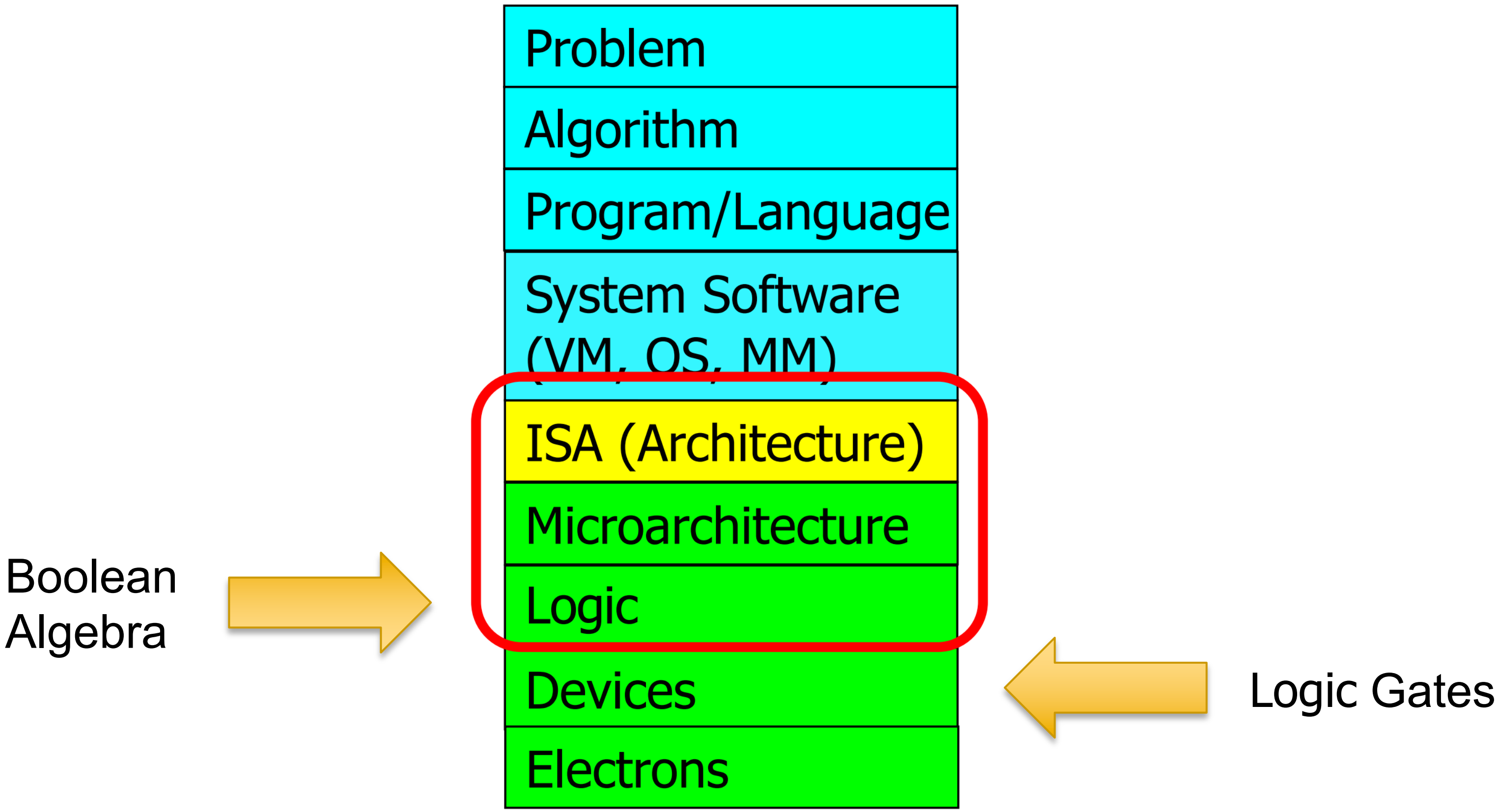
Follow the Lab Manuals

- Solutions may work but they might break when they are included inside of a top-module in future labs

(Do not attempt this)

- Read carefully before implementing

What We Will Learn?



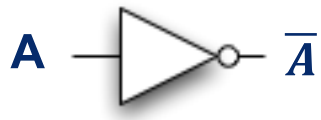
What We Will Learn?

- In Lab 1, you will design simple combinatorial circuits
- We will cover a tutorial about:
  - Boolean Equations
    - Logic operations with binary numbers
  - Logic Gates
    - Basic blocks that are interconnected to form larger units that are needed to construct a computer

Boolean Equations and Logic Gates

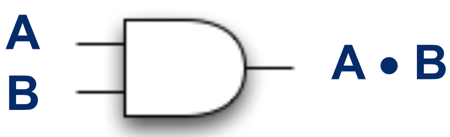
Simple Equations: NOT / AND / OR

$\overline{A}$ *(reads "not A")* is 1 iff A is 0



A	$\overline{A}$
0	1
1	0

A • B *(reads "A and B")* is 1 iff A and B are both 1



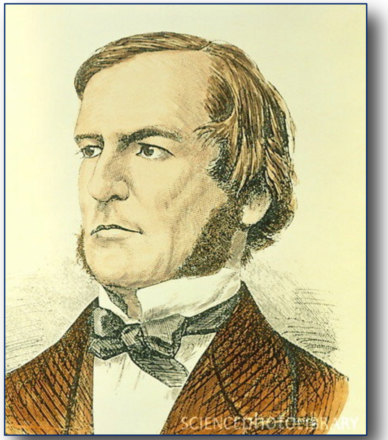
A	B	$A \bullet B$
0	0	0
0	1	0
1	0	0
1	1	1

A + B *(reads "A or B")* is 1 iff either A or B is 1

A	B	$A + B$
0	0	0
0	1	1
1	0	1
1	1	1

Boolean Algebra: Big Picture

- An algebra on 1's and 0's
  - with AND, OR, NOT operations
- What you start with
  - Axioms: basic stuff about objects and operations you just assume to be true at the start
- What you derive first
  - Laws and theorems: allow you to manipulate Boolean expressions
  - …also allow us to do some simplification on Boolean expressions
- What you derive later
  - More "sophisticated" properties useful for manipulating digital designs represented in the form of Boolean equations



George Boole

Common Logic Gates

Buffer

A	Z
0	0
1	1

AND

A	B	Z
0	0	0
0	1	0
1	0	0
1	1	1

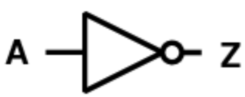
OR

A	B	Z
0	0	0
0	1	1
1	0	1
1	1	1

XOR

A	B	Z
0	0	0
0	1	1
1	0	1
1	1	0

Inverter



A	Z
0	1
1	0

NAND

A	B	Z
0	0	1
0	1	1
1	0	1
1	1	0

NOR

A	B	Z
0	0	1
0	1	0
1	0	0
1	1	0

XNOR

A	B	Z
0	0	1
0	1	0
1	0	0
1	1	1

Boolean Algebra: Axioms

Formal version	*English version*
1. B contains at least two elements, 0 and 1, *such that* $0 \neq 1$	Math formality...
2. *Closure* $a,b \in B$, (i) $a + b \in B$ (ii) $a \bullet b \in B$	Result of AND, OR stays in set you start with
3. *Commutative Laws*: $a,b \in B$, (i) $a + b = b + a$ (ii) $a \bullet b = b \bullet a$	For primitive AND, OR of 2 inputs, order doesn't matter
4. *Identities*: $0, 1 \in B$ (i) $a + 0 = a$ (ii) $a \bullet 1 = a$	There are identity elements for AND, OR, give you back what you started with
5. *Distributive Laws*: (i) $a + (b \bullet c) = (a + b) \bullet (a + c)$ (ii) $a \bullet (b + c) = a \bullet b + a \bullet c$	• distributes over +, just like algebra …but + distributes over •, also (!!)
6. *Complement*: (i) $a + a' = 1$ (ii) $a \bullet a' = 0$	There is a complement element, ANDing, ORing give you an identity

Boolean Algebra: Duality

- Interesting observation
  - All the axioms come in "dual" form
  - Anything true for an expression also true for its dual
  - So any derivation you could make that is true, can be flipped into dual form, and it stays true
- Duality -- More formally
  - A dual of a Boolean expression is derived by replacing
    - Every AND operation with... an OR operation
    - Every OR operation with... an AND
    - Every constant 1 with... a constant 0
    - Every constant 0 with... a constant 1
    - But don't change any of the literals or play with the complements!

Example

$$a \cdot (b + c) = (a \cdot b) + (a \cdot c)$$

$$\rightarrow \; a + (b \cdot c) = (a + b) \cdot (a + c)$$

Boolean Algebra: Useful Laws

Dual

Operations with 0 and 1:

1. $X + 0 = X$ — 1D. $X \bullet 1 = X$
2. $X + 1 = 1$ — 2D. $X \bullet 0 = 0$

AND, OR with identities gives you back the original variable or the identity

Idempotent Law:

3. $X + X = X$ — 3D. $X \bullet X = X$

AND, OR with self = self

Involution Law:

4. $\overline{(\overline{X})} = X$

double complement = no complement

Laws of Complementarity:

5. $X + \overline{X} = 1$ — 5D. $X \bullet \overline{X} = 0$

AND, OR with complement gives you an identity

Commutative Law:

6. $X + Y = Y + X$ — 6D. $X \bullet Y = Y \bullet X$

Just an axiom…

Useful Laws (cont.)

Associative Laws:

7. $(X + Y) + Z = X + (Y + Z)$
$= X + Y + Z$

7D. $(X \bullet Y) \bullet Z = X \bullet (Y \bullet Z)$
$= X \bullet Y \bullet Z$

Parenthesis order doesn't matter

Distributive Laws:

8. $X \bullet (Y + Z) = (X \bullet Y) + (X \bullet Z)$

8D. $X + (Y \bullet Z) = (X + Y) \bullet (X + Z)$

Axiom

Simplification Theorems:

9. $X \bullet Y + X \bullet \overline{Y} = X$

9D. $(X + Y) \bullet (X + \overline{Y}) = X$

10. $X + X \bullet Y = X$

10D. $X \bullet (X + Y) = X$

11. $(X + \overline{Y}) \bullet Y = X \bullet Y$

11D. $(X \bullet \overline{Y}) + Y = X + Y$

Useful for simplifying expressions

Actually worth remembering — they show up a lot in real designs…

DeMorgan's Law

DeMorgan's Law:

$$12.\ \overline{(X + Y + Z + \cdots)} = \bar{X}.\bar{Y}.\bar{Z}. \ldots$$

$$12D.\ \overline{(X . Y . Z. \ldots)} = \bar{X} + \bar{Y} + \bar{Z} + \ \ldots$$

- **Think of this as a transformation**
  - Let's say we have:

$$F = A + B + C$$

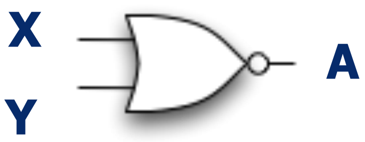
  - Applying DeMorgan's Law (12), gives us:

$$F = \overline{\overline{(A + B + C)}} = \overline{(\bar{A}.\bar{B}.\bar{C})}$$

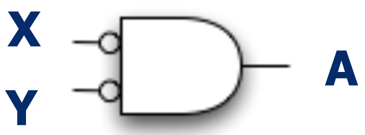
DeMorgan’s Law (cont.)

Interesting — these are conversions between different types of logic
That’s useful given you don’t always have every type of gate

$$A = \overline{(X+Y)} = \bar{X}\bar{Y}$$

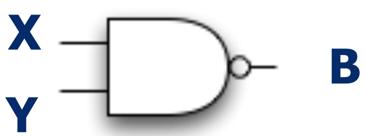


NOR is equivalent to AND with inputs complemented

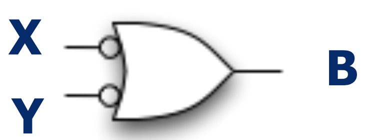


X	Y	$\overline{X+Y}$	$\bar{X}$	$\bar{Y}$	$\bar{X}\bar{Y}$
0	0	1	1	1	1
0	1	0	1	0	0
1	0	0	0	1	0
1	1	0	0	0	0

$$B = \overline{(XY)} = \bar{X} + \bar{Y}$$



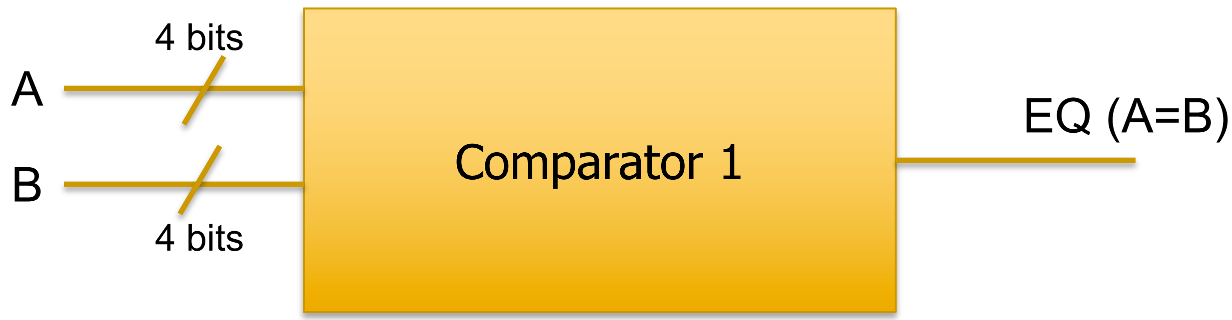
NAND is equivalent to OR with inputs complemented



X	Y	$\overline{XY}$	$\bar{X}$	$\bar{Y}$	$\bar{X}+\bar{Y}$
0	0	1	1	1	1
0	1	1	1	0	1
1	0	1	0	1	1
1	1	0	0	0	0

Part 1: A Comparator Circuit

- Design a comparator that receives two 4-bit numbers A and B, and sets the output bit EQ to logic-1 if A and B are equal



- Hints:
  - First compare A and B bit by bit
  - Then combine the results of the previous steps to set EQ to logic-1 if all A and B are equal

Part 2: A More General Comparator

- Design a circuit that receives two 1-bit inputs A and B, and:
  - sets its first output (O1) to 1 if *A>B,*
  - sets the second output (O2) to 1 if *A=B,*
  - sets the third output (O3) to 1 if *A<B*.

Part 3: Circuits with Only NAND Gates

- Design the circuit of Part 2 using only NAND gates

- Logical Completeness:
  - The set of gates {AND, OR, NOT} is logically complete because we can build a circuit to carry out the specification of any combinatorial logic we wish, without any other kind of gate
  - NAND and NOR are also logically complete

Circuit Drawing

- Circuits can be drawn on paper / tablet / latex / using software
  - Online circuit drawer: logic.ly
    - Uses switches for inputs (on=1, off=0)
    - Uses light bulbs for outputs (on=1, off=0)
  - Latex circuit drawing: tikzmaker

Last Words

- In this lab, you will draw the schematics of some simple operations

- Part 1: A comparator circuit

- Part 2: A more general comparator circuit

- Part 3: Designing circuits using only NAND gates

- You will find more exercises in the lab report

Report Deadline

[22. March 2024 23:59]

Digital Design & Computer Arch.

Lab 1 Supplement: Drawing Basic Circuits

Frank K. Gürkaynak
Seyyedmohammad Sadrosadati
ETH Zurich

Spring 2024

5 March 2024