1 Organization and Introduction

- The Art of managing complexity
  - Abstraction: Hiding details when they are not important
  - Discipline: Intentionally restricting your design choices to that you can work more productively at higher abstraction levels
  - The three -Y's
    - Hierarchy: A system is divided into modules of smaller complexity
    - Modularity: Having well defined functions and interfaces
    - Regularity: Encouraging uniformity, so modules can be easily re-used
- Bit: **B**inary dig**it**

2 Binary Numbers

- Powers of two:

$2^0 = 1$	$2^5 = 32$	$2^{10} = 1024$
$2^1 = 2$	$2^6 = 64$	$2^{11} = 2048$
$2^2 = 4$	$2^7 = 128$	$2^{12} = 4096$
$2^3 = 8$	$2^8 = 256$	$2^{13} = 8192$
$2^4 = 16$	$2^9 = 512$	$2^{14} = 16384$

- Binary to decimal conversion

$$\begin{aligned} 10011_2 &= 2^4 \times 1 + 2^3 \times 0 + 2^2 \times 0 + 2^1 \times 1 + 2^0 \times 1 \\ &= 16 \times 1 + 8 \times 0 + 4 \times 0 + 2 \times 1 + 1 \times 1 \\ &= 16 + 0 + 0 + 2 + 1 = 19_{10} \end{aligned}$$

- Convert decimal to binary (roughly). Example with 47_{10} to binary

$2^6 = 64$	is $64 \leq 47$?	no	0	do nothing
$2^5 = 32$	is $32 \leq 47$?	yes	1	47-32=15
$2^4 = 16$	is $16 \leq 15$?	no	0	do nothing
$2^3 = 8$	is $8 \leq 15$?	yes	1	15-8=7
$2^2 = 4$	is $4 \leq 7$?	yes	1	7-4=3
$2^1 = 2$	is $2 \leq 3$?	yes	1	3-2=1
$2^0 = 1$	is $1 \leq 1$?	yes	1	1-1=0; done!

$\Rightarrow 47_{10}$ to binary is 0101111_2

- Binary values and range
  - $N-$digit decimal number
    - How many values: 10^N
    - Range: $[0, 10^N - 1]$
    - Example (3-digit number): $10^3 = 1000$ possible values, range:$[0, 999]$
  - $N-$bit binary number
    - How many values: 2^N

- * Range: $[0, 2^N - 1]$
  * Example (3-digit number): $2^3 = 8$ possible values, range:$[0, 7] = [000_2 \text{ to } 111_2]$

- Hexadecimal (Base-16) Numbers

Decimal	Hexadecimal	Binary		Decimal	Hexadecimal	Binary
0	0	0000		8	8	1000
1	1	0001		9	9	1001
2	2	0010		10	A	1010
3	3	0011		11	B	1011
4	4	0100		12	C	1100
5	5	0101		13	D	1101
6	6	0110		14	E	1110
7	7	0111		15	F	1111

- Bits, Bytes, Nibbles...

$$\underbrace{1}_{\text{MSB}}001011\underbrace{0}_{\text{LSB}} \quad \overbrace{1001\,\underbrace{0110}_{\text{nibble}}}^{\text{Byte}} \quad \underbrace{\text{CE}}_{\text{MSB}}\,\text{BF9A}\,\underbrace{\text{D7}}_{\text{LSB}}$$

  Where MSB=Most significant Bit and LSB=Least significant Bit

- Addition in base two works exactly the same as in base 10, using carries

- Overflow
  - Digital systems operate on a fixed number of bits
  - Addition overflows when the result is too big to fit in the available number of bits

- Signed Binary Numbers
  - Sign/Magnitude Numbers
    * 1 sign bit, $N - 1$ magnitude bits
    * Sign bit is the most significant (left-most) bit
    * Example: 4-bit sign/mag repr. of ± 6:
      · $+6 = \mathbf{0}110$
      · $-6 = \mathbf{1}110$
    * Range of an $N-$bit sign/magnitude number: $[-\left(2^{N-1} - 1\right), 2^{N-1} - 1]$
    * Problems:
      · Addition doesn't work
      · Two representations of 0 (± 0): 1000 and 0000
      · Introduces complexity in the processor design
  - One's Complement Numbers

∗ A negative number is formed by reversing the bits of the positive number (MSB still indicates the sign of the integer)

2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0		One's Compl.	Unsigned
0	0	0	0	0	0	0	0	=	0	0
0	0	0	0	0	0	0	1	=	1	1
0	0	0	0	0	0	1	0	=	2	2
...	...	...	...	...	...	...	...	...	...	...
0	1	1	1	1	1	1	1	=	127	127
1	0	0	0	0	0	0	0	=	-127	128
1	0	0	0	0	0	0	1	=	-126	129
...	...	...	...	...	...	...	...	...	...	...
1	1	1	1	1	1	0	1	=	-2	253
1	1	1	1	1	1	1	0	=	-1	254
1	1	1	1	1	1	1	1	=	-0	255

∗ Range of $n-$bit number: $[-2^{n-1}-1, 2^{n-1}-1]$, 8 bits:$[-127, 127]$

∗ Addition: Done using binary addition with end-around carry. If there is a carry out of the MSB of the sum, this bit must be added to the LSB of the sum

– Two's Complement Numbers

∗ Don't have same problems as sign/magnitude numbers:

· addition works

· Single representation for 0

∗ Has advantages over one's complement:

· Has a single 0 representation

· Eliminates the end-around carry operation required in one's complement addition.

∗ A negative number is formed by reversing the bits of the positive number (MSB still indicates the sign of the integer) and adding 1:

2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0		Two's Compl.	Unsigned
0	0	0	0	0	0	0	0	=	0	0
0	0	0	0	0	0	0	1	=	1	1
0	0	0	0	0	0	1	0	=	2	2
...	...	...	...	...	...	...	...	...	...	...
0	1	1	1	1	1	1	1	=	127	127
1	0	0	0	0	0	0	0	=	-128	128
1	0	0	0	0	0	0	1	=	-127	129
...	...	...	...	...	...	...	...	...	...	...
1	1	1	1	1	1	0	1	=	-3	253
1	1	1	1	1	1	1	0	=	-2	254
1	1	1	1	1	1	1	1	=	-1	255

∗ Same as unsigned binary, but the most significant bit (MSB) has value of -2^{N-1}

· Most positive 4-bit number: 0111

· Most negative 4-bit number: 1000

∗ The most significant bit still indicates the sign (1=neg., 0=pos.)

∗ Range of an $N-$bit two's comp. number: $[-2^{N-1}, 2^{N-1}-1]$, 8 bits:$[-128, 127]$

- Increasing bit width (assume from N to M, with $M > N$):
  - Sign-extension
    - Sign bit is copied into MSB
    - Number value remains the same
    - Give correct result for two's compl. numbers
    - Example 1:
      - 4-bit representation of 3 = **0**011
      - 8-bit sign-extended value: **00000**011
    - Example 2:
      - 4-bit representation of -5 = **1**011
      - 8-bit sign-extended value: **11111**011
  - Zero-extension
    - Zeros are copied into MSB
    - Value will change for negative numbers
    - Example 1:
      - 4-bit value: $0011_2 = 3_{10}$
      - 8-bit zero-extended value: $\mathbf{0000}0011_2 = 3_{10}$
    - Example 2:
      - 4-bit value: $1011_2 = -5_{10}$
      - 8-bit zero-extended value: $\mathbf{0000}1011_2 = 11_{\mathbf{10}}$

3 Short Introduction to Electrical Engineering (EE Perspective)

- The goal of circuit design is to optimize:
  - Area: Net circuit area is proportional to the cost of the device
  - Speed/Throughput: We want circuits that work faster, or do more
  - Power/Energy
    - Mobile devices need to work with a limited power supply
    - High performance devices dissipate more than 100W/cm^2
  - Design time
    - Designers are expensive
    - The competition will not wait for you
- (Frank's) Principles for engineering
  - Good engineers are lazy: They do not want to work unnecessarily, be creative
  - They know how to ask the question "why"?: take nothing for granted
  - Engineering is not a religion: Use what works best for you
  - Keep it simple and stupid: Engineers' job is to manage complexity

- Building blocks for microchips
  - Conductors: Metals (Aluminium, Copper)
  - Insulators: Glass (SiO_2), Air
  - Semiconductors: Silicon (Si), Germanium (Ge)
- N-type Doping: Add extra electron (negatively charged), zone becomes negatively charged

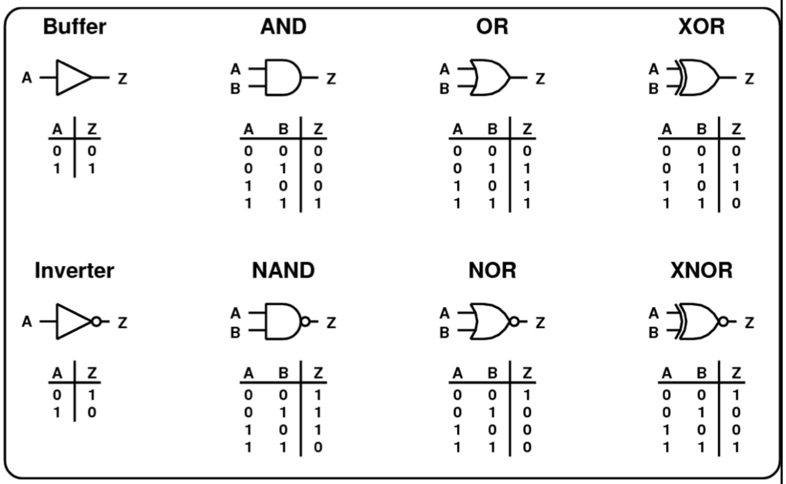
Maybe add a definition or a better explanation

- P-type Doping: Remove electron, zone becomes positively charged
- Semiconductors:
  - You can "Engineer" its properties, i.e.
    * Make it P type by injecting type-III elements (b, Ga, In)
    * Make it N type by injecting elements from type-V (P, As)
  - You can combine P and N regions to each other, from a pure semiconductor
  - Allows you to make interesting electrical devices (Diodes, Transistors, Thrystors)
- pMOS is a P type transistor, nMOS an N type transistors; combined they are a CMOS
- CMOS (Properties)
  - No input current: Capacitive input, no resistive path from the input
  - No current when output is at logic levels: Little static power, current is needed only when switching
  - Electrical properties determined directly by geometry: A transistor that is 2 times larger drives twice the current
  - Very simple to manufacture: pMOS and nMOS can be manufactures on the same substrate
- CMOS Gate Structure
  - The general form used to construct any inverting logic, such as: NOT, NAND, NOR
    * The networks may consist of transistors in series or parallel
    * When transistors are in parallel, the network is ON if either transistor is ON
    * When transistors are in series, the network is ON only if all transistors are ON
  - In a proper logic gate: One of the networks should be ON and the other OFF at any given time
  - Use the rule of conduction complements:
    * When nMOS transistors are in series, the pMOS transistor must be in parallel

* When nMOS transistors are in parallel, the pMOS transistors must be in series

Add picture on slide 34, 03 - EEPerspective

- Logic Gates
  - Perform logic functions: Inversion (NOT), AND, OR, NAND, NOR, etc.
  - Single input: NOT gate, buffer
  - Two-input: AND, OR, XOR, NAND, NOR, XNOR



  - Multiple-Input:
    * 3, 4, or even more input AND, OR, XOR gates
    * Compound gates
      · AND-OR
      · OR-AND
      · AND-OR-INVERT
      · OR-AND-INVERT
    * Other cells: Multiplexers and Adders

- Logic Levels
  - Define ranges of discrete voltages to represent 1 and 0 (i.e. 0 for ground and 1 for 5V (V_{DD})) and allow for noise.

- Noise: Is anything that degrades the signal (i.e. resistance, power supply noise, etc.)

- Moore's Law
  - *"Number of transistors that can be manufactured doubles roughly every 18 months."* - Gordon Moore, 1965

- How do we keep Moore's Law:

- Manufacturing smaller structures: some structures are already a few atoms in size
- Developing materials with better properties
- Optimizing the manufacturing steps
- New technologies

- Power consumption
  - Power = Energy consumed per unit time
  - Two types of power consumption:
    1. Dynamic power consumption: Power to charge transistor gate capacitances
    $$P_{\text{dynamic}} = \frac{1}{2} C V_{DD}^2 f$$
    2. Static power consumption: Power consumed when no gates are switching, caused by the leakage current
    $$P_{\text{static}} = I_{DD} V_{DD}$$

4 Combinational Circuits: Theory

- Circuit elements. A circuit consists of:
  - Inputs
  - Outputs
  - Nodes (wires): Connections between I/O and circuit elements. To count them, look at
    * Outputs of every circuit elements
    * Inputs to the entire circuit
  - Circuit elements
- Types of Logic Circuits
  - Combinational Logic
    * Memoryless
    * Outputs determined by current values of inputs
    * In some books called Combinatorial Logic
  - Sequential Logic
    * Has Memory
    * Outputs determined by previous and current values of inputs
- Rules of Combinational Composition
  - Every circuit element is itself combinational
  - Every node of the circuit is either
    * Designated as an input to the circuit
    * Connects to exactly one output terminal of a circuit element

- The circuit contains no cyclic paths: Every path through the circuit visits each node at most once
- Boolean Equations[1]

[1] For a more in depth look, use the material from Diskrete Mathematik