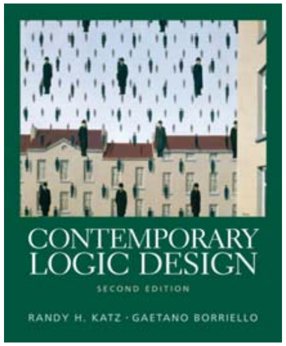
CSE 370 Spring 2006
Introduction to Digital Design

Lecture 2: Binary Number Systems



Last Lecture

- Course Overview
- The Digital Age

Today

- Binary numbers
- Base conversion
- Negative binary numbers
- Switches/CMOS

Administrivia

Make sure

- Signed up to the mailing list

Homework

- Will be assigned on Friday prior to due date
(so that it can haunt you over the weekend!)

- Homework guru: Adrienne Wang (axwang@cs)
Office hours: W 3-5pm in CSE 218

Digital

Digital = Discrete

- Decimal digits
- DNA nucleotides
- Binary codes
  - symbols mapped to bits

Digital Computers

- I/O is digital
  - ASCII, decimal, binary, etc.
- Internal representation
  - binary

BCD

Number Systems

Bases In This Class

- Binary (2), Octal (8), Decimal (10), Hexadecimal(16)
- Positional numbering systems ("significant digits")

$101_2 =$
$67_8 =$
$AB_{16} =$
$41.7_8 =$

Adding, Subtracting

"There are 10 kinds of people in the world—those who understand binary numbers, and those who don't."

1011_2
$+1110_2$

52_{16}
$+ AF_{16}$

10111_2
$- 00101_2$

Conversions

Binary to Octal and Hexadecimal

$1011011001_2 =$

$1011011001_2 =$

Octal and Hexadecimal to Binary

$401_8 =$

$B10_{16} =$

Decimal to Others

Decimal to Binary

58

Decimal to Octal

58

- Why does this work?

Negative Numbers

- Negative binary numbers?
- Historically
  - sign/magnitude
  - ones-complement
  - twos-complement
- For all three:
- most significant bit (msb) is the sign
  - 0=positive 1=negative
- twos-complement universally most used
- simplifies arithmetic

Sign/Magnitude

- most significant bit is sign
  - 0=positive, 1=negative
- remaining bits are magnitude

$0101_2=$
$1101_2=$

- Problem 1: two zeros!

$0000_2 = 0_{10}$ and $1000_2 = -0_{10} = 0_{10}$

- Problem 2: arithmetic is <u>messy</u> (hard to implement)

$4_{10} = 0100_2$	$4_{10} = 0100_2$	$-4_{10} = 0100_2$
$+3_{10} = 0011_2$	$-3_{10} = 1011_2$	$+3_{10} = 1011_2$

Ones-Complement

- most significant bit is sign
  - 0=positive, 1=negative
- negative number is positive numbers bitwise complement

$3_{10} =$
$-3_{10} =$

- Problem 2: arithmetic is clean (add carry)

$4_{10} = 0100_2$
$+3_{10} = 0011_2$

$4_{10} = 0100_2$
$-3_{10} = 1100_2$

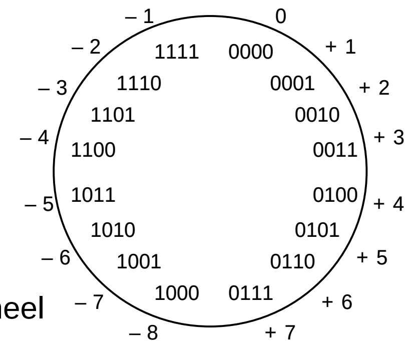
$-4_{10} = 1011_2$
$+3_{10} = 0011_2$

- Problem 1: still two zeros!
  $0000_2 = 0_{10}$ and $1111_2 = -0_{10} = 0_{10}$

Twos-Complement

- most significant bit is sign
  - 0=positive, 1=negative
- negative number is bitwise complement plus 1

$3_{10} =$
$-3_{10} =$



number wheel

Twos-Complement Math

- arithmetic works (drop carry)

$4_{10} = 0100_2$
$+3_{10} = 0011_2$

$4_{10} = 0100_2$
$-3_{10} = 1101_2$

$-4_{10} = 1011_2$
$+3_{10} = 0011_2$

	sign/magnitude	ones-complement	twos-complement
#0's	2	2	1
negative	easy	easy	medium
addition	hard	medium	easy

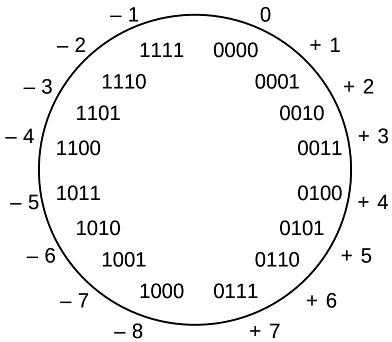
Twos-Complement Exercise

test your skills convert 1_{10} and -5_{10} to 4 bit twos-compelemnt binary and then add them

$1_{10} =$
$-5_{10} =$

Twos-Complement Overflow

- Numbers may add out of range (overflow)



+4=0100
+6=0110
+10=1010

Twos-Complement Overflow

- Numbers may add out of range (overflow)

carry bits	0100	11000	10000
	+4=0100	+4=0100	– 4=1011
	+6=0110	– 3=1100	– 3=1100
	+10=1010	+1=1000	+1=0111

Last two carry bits: c_{last} and c_{2last}

Overflow: f

c_{last}	c_{2last}	f
0	0	0
0	1	1
1	0	1
1	1	0

Twos-Complement Misc

- sign extension
  - $+6_{10} = 0110_2$
  - $–6_{10} = 1001_2$
  - extend to eight bits (a byte):
    - $+6_{10} = 00000110_2$
    - $–6_{10} = 11111001_2$
- different binary numbers have different values
  - 11001 = unsigned
  - 11001 = sign/magnitude
  - 11001 = ones-complement
  - 11001 = twos-complement
- The weird number: 1111_2

Machine Independent?

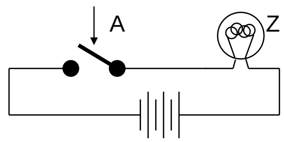
- HAKMEM Item 154 (Bill Gosper)

The myth that any given programming language is machine independent is easily exploded by computing the sum of powers of 2.

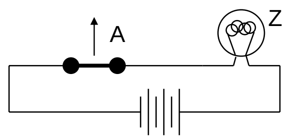
If the result loops with period = 1 with sign +, you are on a sign-magnitude machine.
If the result loops with period = 1 at -1, you are on a twos-complement machine.
If the result loops with period > 1, including the beginning, you are on a ones-complement machine.
If the result loops with period > 1, not including the beginning, your machine isn't binary -- the pattern should tell you the base.
If you run out of memory, you are on a string or Bignum system.
If arithmetic overflow is a fatal error, some fascist pig with a read-only mind is trying to enforce machine independence. But the very ability to trap overflow is machine dependent.

Switches

- Implementing a simple circuit (arrow shows action if wire changes to “1”):



close switch (if A is “1” or asserted) and turn on light bulb (Z)



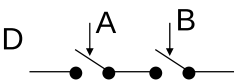
open switch (if A is “0” or unasserted) and turn off light bulb (Z)

$Z \equiv A$

Switches

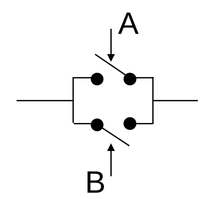
- Compose switches into more complex ones (Boolean functions):

AND



Z ≡ A <u>and</u> B

OR

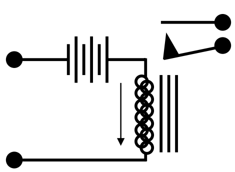


Z ≡ A <u>or</u> B

Switching Networks

- Switch settings determine whether a conducting network to a light bulb
- Larger computations?
  - Use a light bulb (output) to set other switches (input)
  - Example: Mechanical relay

conducting path composed of switches closes circuit



current flowing through coil magnetizes core and causes normally closed (nc) contact to be pulled open

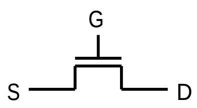
when no current flows, the spring of the contact returns it to its normal position

Transistor Networks

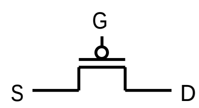
- Relays no more: slow and big
- Modern digital electronics predominately uses CMOS technology
  - MOS: metal-oxcide semiconductor
  - C: complementary (both p and n type transistors arranged so that power is dissipated during switching.)

MOS Transistors

- MOS transistors have three terminals: drain, gate, and source
  - Act as switches: if the voltage on the gate terminal is (some amount) higher/lower than the source terminal then a conducting path will be established between the drain and source terminals.

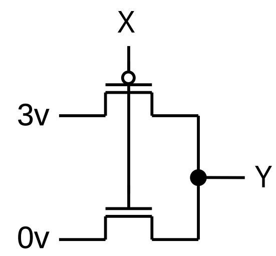


n-channel
open when voltage at G is low
closes when:
voltage(G) > voltage (S) + ε



p-channel
closed when voltage at G is low
opens when:
voltage(G) < voltage (S) – ε

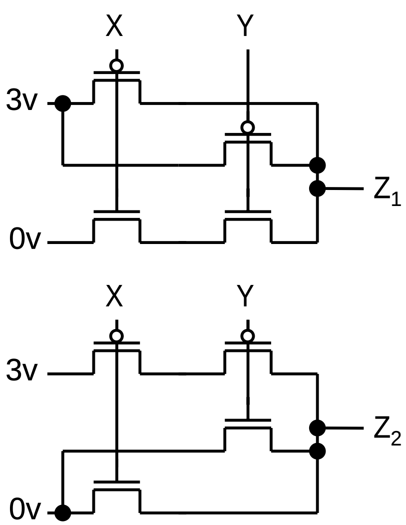
MOS Networks



what is the relationship between x and y?

x	y
0 volts	
3 volts	

Two Input Networks



what is the relationship between x, y and z?

x	y	z1	z2
0 volts	0 volts		
0 volts	3 volts		
3 volts	0 volts		
3 volts	3 volts		

Your To Do List

- Things Internet
  - Sign up for mailing list
- Things Reading
  - Week 1 reading (on website): pp.1-27, Appendix A, pp.33-46
- Things Homework
  - Homework 1 posted on website (due this Friday)
- Things Laboratory
  - Attend first lab session if you haven't already