数字量和模拟量

- 数字量：在时间上和数量上都是离散、不连续的。
  （存在一个最小数量单位△）
- 模拟量：数字量以外的物理量。

Continuous versus Discrete

- Which are "continuous"?
  - Color
  - English letters
  - Light
  - Cars
  - Sound
  - Height and weight
  - Dogs
  - Electric current and voltage

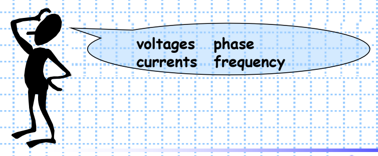
Many natural phenomena are continuous

数字量和模拟量

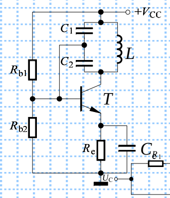
- 数字量：在时间上和数量上都是离散、不连续的。（存在一个最小数量单位△）
- 模拟量：数字量以外的物理量。

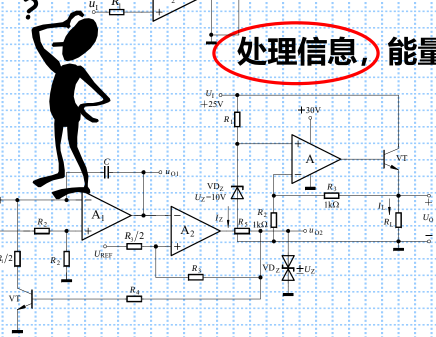
- 数字电路和模拟电路：

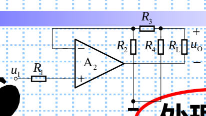
工作信号、研究的对象、分析/设计方法以及所用的数学工具都有显著的不同。



- 电子电路 ？







处理信息，能量转换

- 模拟电路：用连续的模拟电压/流值来表示信息
- 数字电路：用离散的电压序列来表示信息

电子技术的发展

Electronic Design Automation

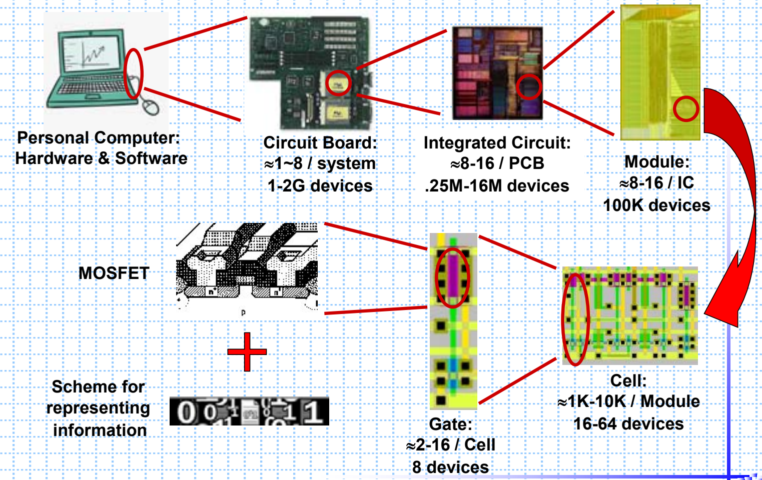
- 48年　贝尔实验室制成第一只晶体管
- 58年　集成电路　(4-12-100-1000)
- 69年　大规模集成电路（10万）
- 75年　超大规模集成电路(15万)
- …

EDA技术

SSI　MSI　LSI　VLSI　ULSI　GLSI

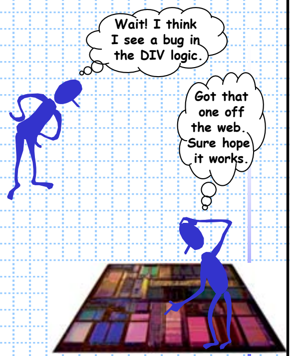
第一片集成电路只有4个晶体管，而97年一片集成电路上有40亿个晶体管。预测集成度按10倍/6年的速度----

How do you build systems with >1G components?



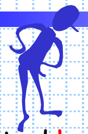
What do we see?

- **Structure　结构**
  - hierarchical design
  - limited complexity at each level
  - reusable building blocks
- **Interfaces　接口**
  - Key elements of system engineering
  - Isolate technologies, allow evolution
  - Major abstraction mechanism
- **What makes a good system design?**
  - minimal mechanism, maximal function
  - reliable in a wide range of environments
  - accommodates future technical improvements



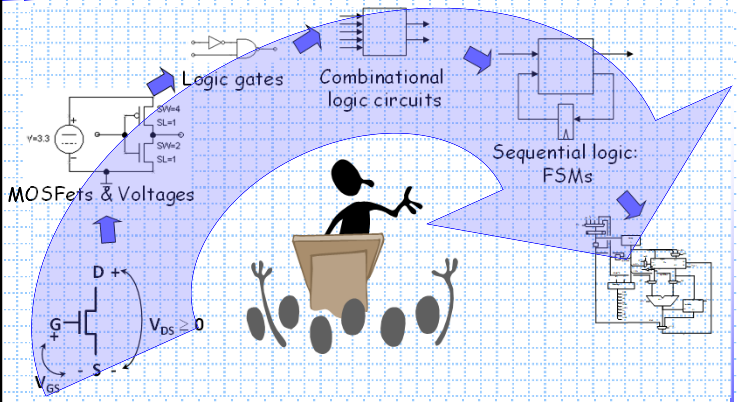
The Art of Managing Complexity

- **Abstraction**
  - Hiding details when they aren't important
- **Discipline**
  - Intentionally restrict design choices
  - Example: Digital discipline
    - Discrete voltages instead of continuous
- **The Three -y's**
  - Hierarchy: A system divided into modules and submodules
  - Modularity: Having well-defined functions and interfaces
  - Regularity: Encouraging uniformity, so modules can be easily reused



Our plan of attack…

- Understand how things work, bottom-up
- Encapsulate our understanding using appropriate abstractions
- Study organizational principles: abstractions, interfaces, API (Application Programming Interface)
- Roll up our sleeves and design at each level of hierarchy
- Learn engineering tricks
  - History
  - Systematic approaches
  - Algorithms
  - Diagnose, fix, and avoid bugs



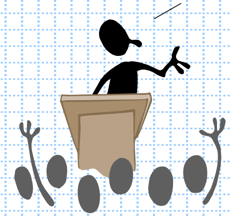
面包板搭建中小规模电路——调试,工具的使用
可编程器件实现大规模电路——EDA工具，硬件描述语言（提前）
PCB板实现成熟电路——EDA工具，焊接调试

编码的目的
二进制编码

第一章 信息和编码

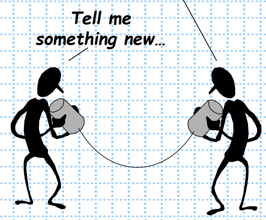
What is “Information 信息”?

“Really, 一点儿都不难!”



information, *n.*
Knowledge, communicated or received concerning a particular fact or circumstance.

中国男足输了



Information resolves uncertainty.
Information is simply that which cannot be predicted.
The less predictable a message is, the more information it conveys!

Quantifying Information
(Claude Shannon, 1948)

Suppose you're faced with N equally probable choices, and I give you a fact that narrows it down to M choices. Then I've given you

$\log_2(N/M)$ bits of information

Information is measured in bits (binary digits) = number of 0/1's required to encode choice(s)

Examples:

- information in one coin flip: $\log_2(2/1)$ = 1 bit
- roll of 2 dice: $\log_2(36/1)$ = 5.2 bits

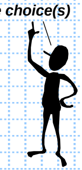
Quantifying Information
(Claude Shannon, 1948)

Suppose you're faced with N equally probable choices, and I give you a fact that narrows it down to M choices. Then I've given you

$\log_2(N/M)$ bits of information

Information is measured in bits (binary digits) = number of 0/1's required to encode choice(s)

data	P_{data}	$\log_2(1/P_{data})$
a heart	13/52	2 bits
not the Ace of spades	51/52	0.028 bits
a face card (J,Q,K)	12/52	2.115 bits
The "suicide king"	1/52	5.7 bits



Get more information when the data resolves more uncertainty about the randomly selected card.

Encoding 编码

- Encoding describes the process of *assigning representations to information*
- Choosing an appropriate and efficient encoding is a real engineering challenge
- Impacts design at many levels
  - Mechanism (devices, # of components used)
  - Efficiency (bits used)
  - Reliability (noise)
  - Security (encryption)

2020011141

- 数制：表示数量的规则
- 码制：表示事物的规则

- **数制：**
  ①每一位的构成
  ②从低位向高位的进位规则

我们常用到的：
十进制，二进制，八进制，十六进制

17

二进制，八进制，十进制，十六进制

- A binary digit has only 2 possibilities —— 逢二进一

0	1

- An octal digit has 8 possibilities —— 逢八进一

0	1	2	3	4	5	6	7

- A decimal digit has 10 possibilities —— 逢十进一

0	1	2	3	4	5	6	7	8	9

- A hexadecimal (hex) digital has 16 possibilities —— 逢十六进一

0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F

18

Encoding numbers

$$v = \sum_{i=0}^{n-1} 2^i b_i$$

2^{11}	2^{10}	2^9	2^8	2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0
0	1	1	1	1	1	0	1	0	0	0	0

$= 2000_{10}$

7 d 0

03720

0x7d0

Oftentimes we will find it convenient to cluster groups of bits together for a more compact notation. Two popular groupings are clusters of 3 bits and 4 bits.

Octal - base 8

000 - 0
001 - 1
010 - 2
011 - 3
100 - 4
101 - 5
110 - 6
111 - 7

Hexadecimal - base 16

0000 - 0	1000 - 8
0001 - 1	1001 - 9
0010 - 2	1010 - a
0011 - 3	1011 - b
0100 - 4	1100 - c
0101 - 5	1101 - d
0110 - 6	1110 - e
0111 - 7	1111 - f

19

Bits, Bytes, Nibbles… （MSB，LSB）

- **Bits**

10010110

most significant bit　　least significant bit

- **Bytes & Nibbles**

byte

10010110

nibble

- **Bytes**

CEBF9AD7

most significant byte　　least significant byte

20

Powers of 2

- $2^0 = 1$
- $2^1 = 2$
- $2^2 = 4$
- $2^3 = 8$
- $2^4 = 16$
- $2^5 = 32$
- $2^6 = 64$
- $2^7 = 128$
- $2^8 = 256$
- $2^9 = 512$
- $2^{10} = 1024$
- $2^{11} = 2048$
- $2^{12} = 4096$
- $2^{13} = 8192$
- $2^{14} = 16384$
- $2^{15} = 32768$

- 2^{10} = 1 kilo ≈ 1000 (1024)
- 2^{20} = 1 mega ≈ 1 million (1,048,576)
- 2^{30} = 1 giga ≈ 1 billion (1,073,741,824)

wang_hong@tsinghua.edu.cn 清華大學

2022年 秋季学期

Sign/Magnitude Numbers

- 1 sign bit, N-1 magnitude bits
- Sign bit is the most significant (left-most) bit
  - Positive number: sign bit = 0
  - Negative number: sign bit = 1

- Example, 4-bit sign/mag representations of ± 6:
  +6 = 0110
  -6 = 1110
- Range of an N-bit sign/magnitude number:
  $[-(2^{N-1}-1), 2^{N-1}-1]$

wang_hong@tsinghua.edu.cn 清華大學

2022年 秋季学期

Sign/Magnitude Numbers

- **Problems:**
  - Addition doesn't work:

$$\begin{array}{r} 1110 \\ +\,0110 \\ \hline 10100 \end{array} \qquad \begin{array}{r} 1100 \\ +\,0100 \\ \hline 10000 \end{array}$$

  - Two representations of 0 (± 0):
    1000
    0000

wang_hong@tsinghua.edu.cn 清華大學

2022年 秋季学期

N bits

-2^{N-1}	2^{N-2}	...	...	...	2^3	2^2	2^1	2^0

"sign bit"

Range: -2^{N-1} to $2^{N-1}-1$

n bits

- 最高位为符号位（0为正，1为负）
- 正数的补码和它的原码相同
- 负数的补码 = 数值位逐位求反 + 1

wang_hong@tsinghua.edu.cn 清華大學

二进制数的补码：

- 最高位为符号位（0为正，1为负）
- 正数的补码和它的原码相同
- 负数的补码 = 数值位逐位求反 + 1

+5 = (0 101)

-5 = (1 011)

两个补码表示的二进制数相加时的符号位讨论

例：用二进制补码运算求出

13 + 10 、13 - 10 、 - 13 + 10 、 - 13 - 10

注意编码的取值范围

+13	0	01101		+13	0	01101
+10	0	01010		−10	1	10110
+23	0	10111		+3	0	00011

−13	1	10011		−13	1	10011
+10	0	01010		−10	1	10110
−3	1	11101		−23	1	01001

结论：将两个加数的符号位和来自最高位数字位的进位相加，结果就是和的符号

+13	0	1101		+13	0	1101
+10	0	1010		−10	1	0110
+23	1	0111		+3	0	0011

−13	1	0011		−13	1	0011
+10	0	1010		−10	1	0110
−3	1	1101		−23	0	1001

Increasing Bit Width

- Extend number from NtoM bits (M>N) :
  - Sign-extension
  - Zero-extension

- Zero-Extension
- Sign-Extension

Zero-Extension

- Zeros copied to msb's
- Value changes for negative numbers

- Example 1:
  - 4-bit value = $0011_2 = 3_{10}$
  - 8-bit zero-extended value: 00000011 = 3_{10}
- Example 2:
  - 4-bit value = $1011 = -5_{10}$
  - 8-bit zero-extended value: 00001011 = 11_{10}

Sign-Extension

- Sign bit copied to msb's
- Number value is same

- Example 1:
  - 4-bit representation of 3 = 0011
  - 8-bit sign-extended value:
- Example 2:
  - 4-bit representation of -5 = 1011
  - 8-bit sign-extended value:

Sign-Extension

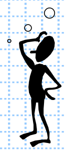
- Sign bit copied to msb's
- Number value is same

- Example 1:
  - 4-bit representation of 3 = 0011
  - 8-bit sign-extended value: 00000011
- Example 2:
  - 4-bit representation of -5 = 1011
  - 8-bit sign-extended value: 11111011

Number System Comparison

- What about fractions?

Number System	Range
Unsigned	$[0, 2^N-1]$
Sign/Magnitude	$[-(2^{N-1}-1), 2^{N-1}-1]$
Two's Complement	$[-2^{N-1}, 2^{N-1}-1]$



For example, 4-bit representation:

	-8	-7	-6	-5	-4	-3	-2	-1	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
Unsigned									0000	0001	0010	0011	0100	0101	0110	0111	1000	1001	1010	1011	1100	1101	1110	1111
Two's Complement	1000	1001	1010	1011	1100	1101	1110	1111	0000	0001	0010	0011	0100	0101	0110	0111								
Sign/Magnitude		1111	1110	1101	1100	1011	1010	1001	0000 1000	0001	0010	0011	0100	0101	0110	0111								

Numbers with Fractions

- Two common notations:
  - Fixed-point: binary point fixed
  - Floating-point: binary point floats to the right of the most significant 1

wang_hong@tsinghua.edu.cn 清华大学

- 例

$$(101.11)_B = 1\times 2^2 + 0\times 2^1 + 1\times 2^0 + 1\times 2^{-1} + 1\times 2^{-2} = (5.75)_D$$

$$(2A.7F)_H = 2\times 16^1 + 10\times 16^0 + 7\times 16^{-1} + 15\times 16^{-2} = (42.4960937)_D$$

$$D = \sum K_i N^i$$ 任意进制。。。

wang_hong@tsinghua.edu.cn 清华大学

Fixed-Point Numbers

- 6.75 using 4 integer bits and 4 fraction bits:

  01101100

  0110.1100

  $2^2 + 2^1 + 2^{-1} + 2^{-2} = 6.75$

  - Binary point is implied
  - The number of integer and fraction bits must be agreed upon beforehand
- Represent 7.5_{10} using 4 integer bits and 4 fraction bits.

  0111.1000

wang_hong@tsinghua.edu.cn 清华大学

Signed Fixed-Point Numbers

- Represent 7.5_{10} using 4 integer bits and 4 fraction bits.

  0111.1000
- Representations:
  - Sign/magnitude
  - Two's complement
- Example: Represent -7.5_{10} using 4 integer and 4 fraction bits
  - Sign/magnitude: 1111.1000
  - Two's complement

wang_hong@tsinghua.edu.cn 清华大学

Signed integers: 2's complement (二进制数的补码)

N bits

-2^{N-1}	2^{N-2}	...	...	...	2^3	2^2	2^1	2^0	2^{-1}	2^{-2}

"sign bit"　　Range: -2^{N-1} to $2^{N-1}-1$　　"decimal" point

By moving the implicit location of "decimal" point, we can represent fractions too:

1101.0110

$= -2^3 + 2^2 + 2^0 + 2^{-2} + 2^{-3}$

$= -8 + 4 + 1 + 0.25 + 0.125$

$= -2.625$

Signed Fixed-Point Numbers

- Represent 7.5_{10} using 4 integer bits and 4 fraction bits.
  0111.1000
- Representations:
  - Sign/magnitude
  - Two's complement
- Example: Represent -7.5_{10} using 4 integer and 4 fraction bits
  - Sign/magnitude: 11111000
  - Two's complement:
    1. +7.5: 0111.1000
    2. Invert bits: 1000.0111
    3. Add 1 to LSB: + 1

    1000.1000

- 码制

  用不同数码表示不同事物时遵循的规则

  例如：学号，身份证号，车牌号。。。

- 目前，数字电路中都采用二进制
- 表示数量时称二进制
- 表示事物时称二值逻辑