

2020-2021学年秋季学期

第一部分
数制和码制

授课团队：宋威

助 教：薛子涵

- **数字电路建立于二进制之上**
 - **晶体管只能有效模仿开关电路，0/1状态**
 - 晶体管特性的非线性
 - 只有0/1状态能被有效并低噪地表示
 - **开关电路，0/1状态的数学表现为二进制**

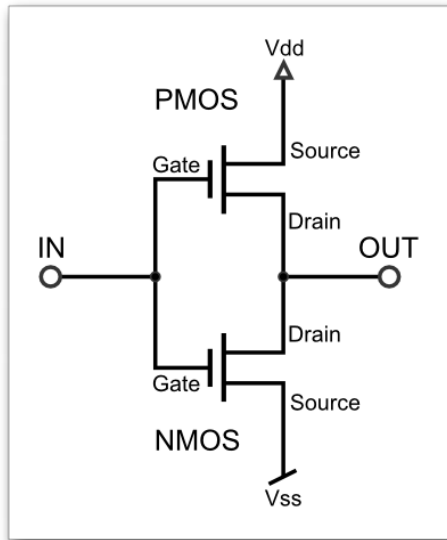
- **人们的日常生活、基础数学建立在十进制之上**
 - 人有10个手指，这其实是习惯
 - 习惯使用十、百、千、万等大数描述

- **数字电路的实际使用必须转换数制**
 - 复杂数字电路完成二进制运算
 - 输入数据需要将十进制转换为二进制
 - 输出数据需要将二进制转换为十进制

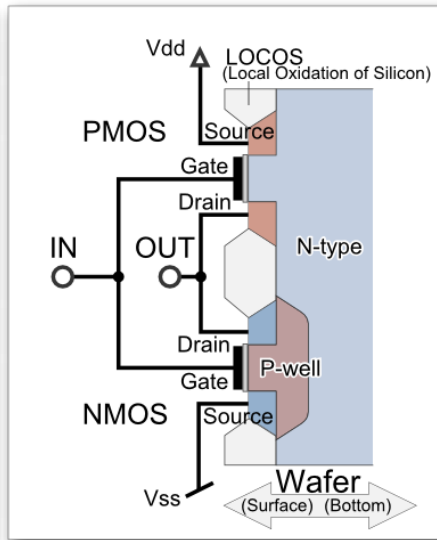
为什么数字电路建立于二进制之上

CMOS inverter

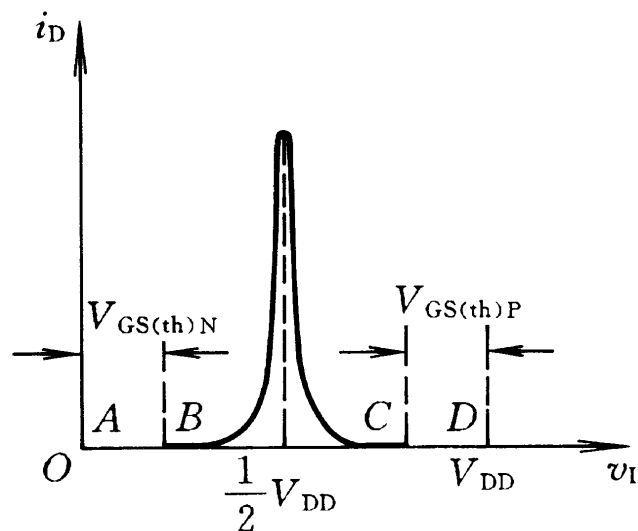
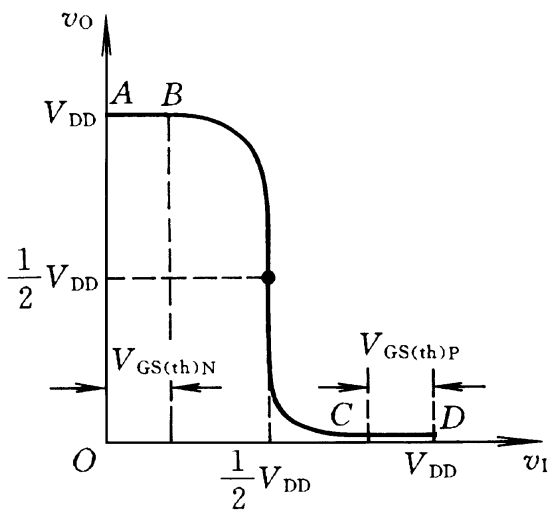
Model chart



Silicon wafer



饱和区
低电流



数制的基本概念

表示数时，仅用一位数码往往不够用，必须用进位计数的方法组成多位数码。多位数码每一位的构成以及从低位到高位进位规则称为进位计数制，简称数制。

○ 基数

○ 进位制的基数，就是在该进位制中可能用到的数码个数。

○ 权数

○ 在某一进位制的数中，每一位的大小都对应着该位上的数码乘上一个固定的数，这个固定的数就是这一位的权数。权数是一个幂。

○ 系数

○ 每一位上的数，也是相应权数的系数

什么是数制：十进制自然数到N进制自然数

$$1997 = 1 \times 10^3 + 9 \times 10^2 + 9 \times 10^1 + 7 \times 10^0$$

$$1997 = [1 \quad 9 \quad 9 \quad 7] \cdot \begin{bmatrix} 10^3 \\ 10^2 \\ 10^1 \\ 10^0 \end{bmatrix}$$

$$D = \sum_{i=0}^{+\infty} k_i \times 10^i \quad 1997 = (1997)_{10}$$

十进制数是以**10**为基数**k**为系数的数字表述。

N进制数是以**N**为基数**k**为系数的数字表述。

$$D = \sum_{i=0}^{+\infty} k_i \times N^i = (k_{+\infty}, \dots, k_i, \dots, k_0)_N$$

十进制正实数到N进制正实数

$$1997.75 = [1 \ 9 \ 9 \ 7 \ 7 \ 5] \cdot \begin{bmatrix} 10^3 \\ 10^2 \\ 10^1 \\ 10^0 \\ 10^{-1} \\ 10^{-2} \end{bmatrix}$$

$$D = \sum_{i=-\infty}^{+\infty} k_i \times 10^i \quad 1997.75 = (1997.75)_{10}$$

$$D = \sum_{i=-\infty}^{+\infty} k_i \times N^i = (k_{+\infty}, \dots, k_i, \dots, k_{-\infty})_N$$

十进制正实数的定点表示(附加)

$$1997.53 = [1 \ 9 \ 9 \ 7 \ 7 \ 5] \cdot \begin{bmatrix} 10^3 \\ 10^2 \\ 10^1 \\ 10^0 \\ 10^{-1} \\ 10^{-2} \end{bmatrix}$$

$$199.753 = [1 \ 9 \ 9 \ 7 \ 7 \ 5] \cdot \begin{bmatrix} 10^2 \\ 10^3 \\ 10^0 \\ 10^{-1} \\ 10^{-2} \\ 10^{-3} \end{bmatrix}$$

8位定点数(5,3):

$$19977.5 = \{19977500\}$$

$$1997.75 = \{01997750\}$$

$$199.775 = \{00199775\}$$

$$19.9775 = \{00019977\} \text{ 精度丢失! 需要浮点表示}$$

十进制正实数的浮点表示(附加)

8位定点数(5,3): 5位整数位, 3位小数位

$$19977.5 = \{19977500\}$$

$$1997.75 = \{01997750\}$$

$$199.775 = \{00199775\}$$

$$19.9775 = \{00019977\} \text{ 精度丢失! 需要浮点表示}$$

8位浮点数(1,7): 1位指数位, 7位小数位

$$19977.5 = \{60199775\} = 0.0199775 * 10^6$$

$$1997.75 = \{50199775\} = 0.0199775 * 10^5$$

$$199.775 = \{40199775\} = 0.0199775 * 10^4$$

$$19.9775 = \{30199775\} = 0.0199775 * 10^3$$

浮点表示并不唯一

$$19977.5 = \{51997750\} = 0.1997750 * 10^5$$



○N进制

$$D = \sum_{i=-\infty}^{+\infty} k_i \times N^i = (k_{+\infty}, \dots, k_i, \dots k_{-\infty})_N$$

○二进制

D相同时，系数向量{k}并不相等。

$$D = \sum_{i=-\infty}^{+\infty} k_i \times 2^i = (k_{+\infty}, \dots, k_i, \dots k_{-\infty})_2$$

二进制数转十进制数

$$D = \sum_{i=-\infty}^{+\infty} k_i \times 2^i = (k_{+\infty}, \dots, k_i, \dots, k_{-\infty})_2$$

$$\begin{aligned} & (101110.11)_2 \\ &= 1 \times 2^5 + 1 \times 2^3 + 1 \times 2^2 + 1 \times 2^1 + 1 \times 2^{-1} + 1 \times 2^{-2} \\ &= 32 + 8 + 4 + 2 + 0.5 + 0.25 \\ &= 46.75 \end{aligned}$$

$$(11111001101.11)_2 = (1997.75)_{10}$$

十进制数转二进制数-公式展开

$$D = \sum_{i=-\infty}^{+\infty} k_i \times 10^i = (k_{+\infty}, \dots, k_i, \dots, k_{-\infty})_{10}$$

$$\begin{aligned} & (46)_{10} \\ &= 4 \times 10^1 + 6 \times 10^0 \\ &= 100 \times 1010^1 + 110 \times 1010^0 \\ &= ??? \end{aligned}$$

用公式直接分解然后代入二进制运算并不方便!

十进制数转二进制数-基数连除、连乘法

整数部分:

基数连除,

取余数自下而上.

小数部分:

基数连乘,

取整数自上而下.

| | | | |
|---|----|----------------|----|
| 2 | 44 | 余数 | 低位 |
| 2 | 22 | 0= K_0 | ↑ |
| 2 | 11 | 0= K_1 | |
| 2 | 5 | 1= K_2 | |
| 2 | 2 | 1= K_3 | |
| 2 | 1 | 0= K_4 | |
| | 0 | 1= K_5 | |

| | | |
|------------|-------------------|----|
| 0.375 | 整数 | 高位 |
| $\times 2$ | 0= K_{-1} | ↓ |
| 0.750 | 1= K_{-2} | |
| $\times 2$ | 1= K_{-3} | |
| 1.500 | | |
| 0.500 | | |
| 1.000 | | |

所以: $(44.375)_D = (101100.011)_B$

十进制数转二进制数-其他方法

快速手算

| 2^0 | 2^1 | 2^2 | 2^3 | 2^4 | 2^5 | 2^6 | 2^7 | 2^8 | 2^9 | 2^{10} | 2^{11} |
|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|----------|----------|
| 1 | 2 | 4 | 8 | 16 | 32 | 64 | 128 | 256 | 512 | 1024 | 2048 |

| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
|------|------|------|------|------|------|------|------|------|------|
| 0000 | 0001 | 0010 | 0011 | 0100 | 0101 | 0110 | 0111 | 1000 | 1001 |

$$44 = 32+8+4 \quad 101100$$

$$75 = 64+8+3 \quad 1001011$$

$$31 = 32 - 1 \quad 100000 - 1 = 11111$$

计算器（程序员模式）

演示

- 数制转换可能造成数据精度的损失，特别是对于小数部分

$$(1.3)_{10} = (1.01001100110011\dots)_2$$

- 原因

$$10 \neq 2^i$$

十进制和二进制的权数之间不能互相整除

十进制和二进制总结

- 为什么讨论二进制
 - 数字电路建立在开关电路之上
- 什么是数制
 - 基数、权数、系数
- 二进制到十进制的转换
 - 直接利用数制公式
- 十进制到二进制转换
 - 基数连除、连乘法
 - 手算、计算器
- 定点小数和浮点数（附加）
- 精度丢失问题（附加）

八进制和十六进制

○二进制 (bin)

$$B = \sum_{i=-\infty}^{+\infty} k_i \times 2^i = (k_{+\infty}, \dots, k_i, \dots, k_{-\infty})_2$$

○八进制 (oct)

$$O = \sum_{i=-\infty}^{+\infty} k_i \times 8^i = (k_{+\infty}, \dots, k_i, \dots, k_{-\infty})_8$$

○十六进制 (hex)

$$H = \sum_{i=-\infty}^{+\infty} k_i \times 16^i = (k_{+\infty}, \dots, k_i, \dots, k_{-\infty})_{16}$$

| | | | | | | | | | | | | | | | | |
|----|---|---|---|---|---|---|---|---|---|---|----|----|----|----|----|----|
| + | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 |
| 十六 | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | A | B | C | D | E | F |

○八进制 (oct)

$$(56)_8 = 5 \times 8^1 + 6 \times 8^0 = (46)_{10}$$

○十六进制 (hex)

$$(56)_{16} = 5 \times 16^1 + 6 \times 16^0 = (86)_{10}$$

$$(CF)_{16} = 12 \times 16^1 + 15 \times 16^0 = (207)_{10}$$

十进制转成八进制和十六进制

○先转成二进制，再由二进制转八/十六进制

○八进制 (oct) : **3位二进制对1位八进制**

| | | | | |
|-----|-----|-----|-----|-----|
| 二进制 | 010 | 111 | 011 | 001 |
| 八进制 | 4 | 7 | 3 | 1 |

$$(010_111_011_001)_2 = (4731)_8$$

○十六进制 (hex) : **4位二进制对1位十六进制**

| | | | |
|-----|------|------|------|
| 二进制 | 0101 | 1101 | 1001 |
| 八进制 | 5 | D | 9 |

$$(0101_1101_1001)_2 = (5D9)_{16}$$

○为什么讨论八进制和十六进制

- 二进制表述系数向量太长，八进制和十六进制有效的缩短了表达长度
- 十六进制是现在计算机内存地址的标准表达方式
 - 1KB = 1024 Byte = 2^{10} Byte
 - 1MB = 1024 KB = 2^{20} Byte
 - 1GB = 1024 MB = 2^{30} Byte
 - 内存地址 0xFFF0_0844

○八/十六进制到十进制的转换

- 直接利用数制公式

○十进制到八/十六进制转换

- 先转换为二进制，然后按位转换

- 和十进制算数运算没有本质区别：**逢二进一**

加法运算

$$\begin{array}{r} 1001 \\ + 0101 \\ \hline 1110 \end{array}$$

减法运算

$$\begin{array}{r} 1001 \\ - 0101 \\ \hline 0100 \end{array}$$

二进制的算术运算

○和十进制算数运算没有本质区别：**逢二进一**

乘法运算

$$\begin{array}{r} 1001 \\ \times 0101 \\ \hline 1001 \\ 0000 \\ 1001 \\ 0000 \\ \hline 0101101 \end{array}$$

除法运算

$$\begin{array}{r} 1.11\dots \\ 0101 \overline{)1001} \\ \underline{0101} \\ 1000 \\ \underline{0101} \\ 0110 \\ \underline{0101} \\ 0010 \end{array}$$

○为什么引入原码、补码与反码

○表达负数并简化（统一）加减法运算

$$D = \sum_{i=0}^n k_i \times 2^i = (k_n, \dots, k_1, k_0)_2$$

○原码 (orig) : s为符号位

$$D_{orig} = (s, k_n, \dots, k_1, k_0)$$

○反码 (inv) : 正数时等于原码, 负数时

$$D_{inv} = (1, \bar{k}_n, \dots, \bar{k}_1, \bar{k}_0)$$

○补码 (comp) 2's complement code : 正数时等于原码, 负数时

$$D_{comp} = D_{inv} + 1$$

○引入补码后:

$$(A - B)_{comp} = A_{comp} + (-B)_{comp}$$

原码并不能正确处理负数加减法

○当结果为正数时，添加符号位不影响结果

○ $2 + 3 = 5$ $(0010)_2 + (0011)_2 = (0101)_2$

○ $7 - 4 = 3$ $(0111)_2 - (0100)_2 = (0011)_2$

○当减法结果为负数时，结果并不正确

○ $4 - 7 = -3$ $(0100)_2 - (0111)_2 = (1101)_2 = (-5)_{10}$

○当负数加正数时，结果也不正确

○ $-2 + 3 = 1$ $(1010)_2 + (0011)_2 = (1101)_2 = (-5)_{10}$

利用补码解决负数计算问题

○当结果为正数时，原码和补码相等，结果仍然正确

○ $2 + 3 = 5$ $(0010)_2 + (0011)_2 = (0101)_2$

○ $7 - 4 = 3$ $(0111)_2 - (0100)_2 = (0011)_2$

○当减法结果为负数时，减数当作补码，变成加法，结果为正确的补码

○ $4 - 7 = -3$

○ $(-7)_{comp} = (-7)_{inv} + 1 = (1000)_2 + (0001)_2 = (1001)_2$

○ $(-3)_{comp} = (-3)_{inv} + 1 = (1100)_2 + (0001)_2 = (1101)_2$

○ $(0100)_2 + (1001)_2 = (1101)_2 = (-3)_{comp}$

○当负数加正数时，结果也正确

○ $-2 + 3 = 1$

○ $(-2)_{comp} = (-2)_{inv} + 1 = (1101)_2 + (0001)_2 = (1110)_2$

○ $(1110)_2 + (0011)_2 = (0001)_2 = (1)_{comp}$

○对于负数来说

○ $d_{orig} + d_{inv} = M - 1$, 其中 $M = 2^n$, 也可以理解成符号位。

$$(-3)_{orig} + (-3)_{inv} = (1011)_2 + (1100)_2 = (0111)_2 = 7$$

○ $d_{comp} = d_{inv} + 1 = M - d_{orig}$ (不过我们这里暂时不考虑符号位)

$$○ (-d)_{comp} = (1, M - d)_2$$

○ 所以, 对于减法 $A - B$ (假设 A, B 都是正数)

$$(0, A)_2 + (1, M - B)_2 = (1, M + A - B)_2$$

○ 同理, 负数的加法 $-A + B$

$$(1, M - A)_2 + (0, B)_2 = (1, M + B - A)_2$$

补码到原码的转换 (附加)

因为:

$$d_{comp} = d_{inv} + 1 = M - d_{orig}$$

所以:

$$d_{comp} + d_{orig} = M$$

$$d_{orig} = M - d_{comp}$$

$$d_{orig} = (d_{comp})_{inv} + 1$$



- 利用原码直接计算

 - 逢二进一

- 原码、反码和补码

 - 引入符号位

 - 带符号位的原码计算不能处理负数

 - 利用补码处理负数，统一加法和减法

○我们只讲BCD码、格雷码和ASCII码

○BCD码： Binary Coded Decimal

- 用二进制编码来表示十进制数，电路系统中输出可读数据的主要方式。

○格雷码： Gray code

- 相邻码字之间的二进制码差为1
- 通讯：减小噪声、码间距（汉明距离）稳定

○ASCII码： American Standard Code for Information Interchange

- 将常见的英文/数字/符号用7(8)位二进制数表示的国际标准。
- 现在的操作系统中使用unicode（16位）表示多语言，是ASCII的超集

○其他的常见编码见教材

- 余3码、2421码、5211码

| 码字 | 编码 |
|----|------|
| 0 | 0000 |
| 1 | 0001 |
| 2 | 0010 |
| 3 | 0011 |
| 4 | 0100 |
| 5 | 0101 |
| 6 | 0110 |
| 7 | 0111 |
| 8 | 1000 |
| 9 | 1001 |

由0~9的二进制编码直接表示十进制，每4位二进制数代表一位十进制数：

$$(763)_{10} = (0111_0110_0011)_{BCD}$$

↓ ↓ ↓
7 6 3

| 码字 | 编码 |
|----|------|
| 0 | 0000 |
| 1 | 0001 |
| 2 | 0011 |
| 3 | 0010 |
| 4 | 0110 |
| 5 | 0111 |
| 6 | 0101 |
| 7 | 0100 |
| 8 | 1100 |
| 9 | 1101 |
| 10 | 1111 |
| 11 | 1110 |
| 12 | 1010 |
| 13 | 1011 |
| 14 | 1001 |
| 15 | 1000 |

相邻码字之间的码差只有一个比特。

如果用Gray码做计数器，可以避免漏码和插码。

从7到8，Gray码：

$$(0100)_{Gray} \rightarrow (1100)_{Gray}$$

二进制：

$$(0111)_2 \rightarrow (1011)_2 \rightarrow (1000)_2$$

$$(0111)_2 \rightarrow (0010)_2 \rightarrow (1000)_2$$

$$(0111)_2 \rightarrow (0000)_2 \rightarrow (1000)_2$$

但是，4位的Gray码当进位的时候不保持单一比特变化的特性！

$$(0000_1000)_{Gray} \rightarrow (0001_0000)_{Gray}$$

Gray码的特性只在一个码字内有效，多个码字同步仍然需要其他的办法。

ASCII码

| Dec | Hex | Oct | Chr | Dec | Hex | Oct | HTML | Chr | Dec | Hex | Oct | HTML | Chr | Dec | Hex | Oct | HTML | Chr |
|-----|-----|-----|-------|-----|-----|-----|--------|-------|-----|-----|-----|--------|-----|-----|-----|-----|--------|-----|
| 0 | 0 | 000 | NULL | 32 | 20 | 040 | | Space | 64 | 40 | 100 | @ | @ | 96 | 60 | 140 | ` | ` |
| 1 | 1 | 001 | SoH | 33 | 21 | 041 | ! | ! | 65 | 41 | 101 | A | A | 97 | 61 | 141 | a | a |
| 2 | 2 | 002 | SoTxt | 34 | 22 | 042 | " | " | 66 | 42 | 102 | B | B | 98 | 62 | 142 | b | b |
| 3 | 3 | 003 | EoTxt | 35 | 23 | 043 | # | # | 67 | 43 | 103 | C | C | 99 | 63 | 143 | c | c |
| 4 | 4 | 004 | EoT | 36 | 24 | 044 | $ | \$ | 68 | 44 | 104 | D | D | 100 | 64 | 144 | d | d |
| 5 | 5 | 005 | Enq | 37 | 25 | 045 | % | % | 69 | 45 | 105 | E | E | 101 | 65 | 145 | e | e |
| 6 | 6 | 006 | Ack | 38 | 26 | 046 | & | & | 70 | 46 | 106 | F | F | 102 | 66 | 146 | f | f |
| 7 | 7 | 007 | Bell | 39 | 27 | 047 | ' | ' | 71 | 47 | 107 | G | G | 103 | 67 | 147 | g | g |
| 8 | 8 | 010 | Bsp | 40 | 28 | 050 | (| (| 72 | 48 | 110 | H | H | 104 | 68 | 150 | h | h |
| 9 | 9 | 011 | HTab | 41 | 29 | 051 |) |) | 73 | 49 | 111 | I | I | 105 | 69 | 151 | i | i |
| 10 | A | 012 | LFeed | 42 | 2A | 052 | * | * | 74 | 4A | 112 | J | J | 106 | 6A | 152 | j | j |
| 11 | B | 013 | VTab | 43 | 2B | 053 | + | + | 75 | 4B | 113 | K | K | 107 | 6B | 153 | k | k |
| 12 | C | 014 | FFeed | 44 | 2C | 054 | , | , | 76 | 4C | 114 | L | L | 108 | 6C | 154 | l | l |
| 13 | D | 015 | CR | 45 | 2D | 055 | - | - | 77 | 4D | 115 | M | M | 109 | 6D | 155 | m | m |
| 14 | E | 016 | SOut | 46 | 2E | 056 | . | . | 78 | 4E | 116 | N | N | 110 | 6E | 156 | n | n |
| 15 | F | 017 | SIn | 47 | 2F | 057 | / | / | 79 | 4F | 117 | O | O | 111 | 6F | 157 | o | o |
| 16 | 10 | 020 | DLE | 48 | 30 | 060 | 0 | 0 | 80 | 50 | 120 | P | P | 112 | 70 | 160 | p | p |
| 17 | 11 | 021 | DC1 | 49 | 31 | 061 | 1 | 1 | 81 | 51 | 121 | Q | Q | 113 | 71 | 161 | q | q |
| 18 | 12 | 022 | DC2 | 50 | 32 | 062 | 2 | 2 | 82 | 52 | 122 | R | R | 114 | 72 | 162 | r | r |
| 19 | 13 | 023 | DC3 | 51 | 33 | 063 | 3 | 3 | 83 | 53 | 123 | S | S | 115 | 73 | 163 | s | s |
| 20 | 14 | 024 | DC4 | 52 | 34 | 064 | 4 | 4 | 84 | 54 | 124 | T | T | 116 | 74 | 164 | t | t |
| 21 | 15 | 025 | NAck | 53 | 35 | 065 | 5 | 5 | 85 | 55 | 125 | U | U | 117 | 75 | 165 | u | u |
| 22 | 16 | 026 | Syn | 54 | 36 | 066 | 6 | 6 | 86 | 56 | 126 | V | V | 118 | 76 | 166 | v | v |
| 23 | 17 | 027 | EoTB | 55 | 37 | 067 | 7 | 7 | 87 | 57 | 127 | W | W | 119 | 77 | 167 | w | w |
| 24 | 18 | 030 | Can | 56 | 38 | 070 | 8 | 8 | 88 | 58 | 130 | X | X | 120 | 78 | 170 | x | x |
| 25 | 19 | 031 | EoM | 57 | 39 | 071 | 9 | 9 | 89 | 59 | 131 | Y | Y | 121 | 79 | 171 | y | y |
| 26 | 1A | 032 | Sub | 58 | 3A | 072 | : | : | 90 | 5A | 132 | Z | Z | 122 | 7A | 172 | z | z |
| 27 | 1B | 033 | Esc | 59 | 3B | 073 | ; | ; | 91 | 5B | 133 | [| [| 123 | 7B | 173 | { | { |
| 28 | 1C | 034 | FSep | 60 | 3C | 074 | < | < | 92 | 5C | 134 | \ | \ | 124 | 7C | 174 | | | |
| 29 | 1D | 035 | GSep | 61 | 3D | 075 | = | = | 93 | 5D | 135 |] |] | 125 | 7D | 175 | } | } |
| 30 | 1E | 036 | RSep | 62 | 3E | 076 | > | > | 94 | 5E | 136 | ^ | ^ | 126 | 7E | 176 | ~ | ~ |
| 31 | 1F | 037 | USep | 63 | 3F | 077 | ? | ? | 95 | 5F | 137 | _ | _ | 127 | 7F | 177 | | Del |

字符串 “Hello world!”

```
0x00 6C6C6548 Hell
0x04 6F77206F o wo
0x08 21646C72 rld!
0x0C 00000000
```

charstable.com

常见的编码形式总结

- BCD码
 - 十进制的二进制表现形式
- 格雷码
 - 编码距离的概念和用途
- ASCII码
 - 计算机系统中可见字符的编码

总结：考试范围

- 数制
 - 数制的定义
- 二进制
 - 二进制和十进制之间的互相转换
- 八进制和十六进制
 - 二进制、八进制和十六进制之间的互相转换
- 二进制计算
 - 二进制原码的加减乘除手算
 - 基于补码的二进制加减法
- 编码
 - 十进制与BCD编码之间的转换

任何问题？

第一章习题：

4, 5, 6, 7, 12, 15。

○列出下列数字的8位二进制原码

31, 66, 125, 93

○列出下列数字的8位二进制补码

-66, -93

○使用8位二进制补码计算

31+125

66-93