第2章 计算系统的基本思维

主要内容

语义符号化的经典案例： 0和1与易经
思维方式与逻辑运算： “0/1与逻辑
二进制与算术运算：0和1与数值信息
编码与符号运算： 0/1与非数值信息
0和1与电子元器件

2.1.1 语义符号化的经典案例： 0和1与易经
(1) 易经是什么？

《易经》是什么？

八卦？ 预测与占卜？ 算命？

自然现象及其变化规律

➔人事现象及其变化规律

➔其他现象及其变化规律

天

日 月

风 雷

泽 山

地

？

2.1.1 语义符号化的经典案例： 0和1与易经

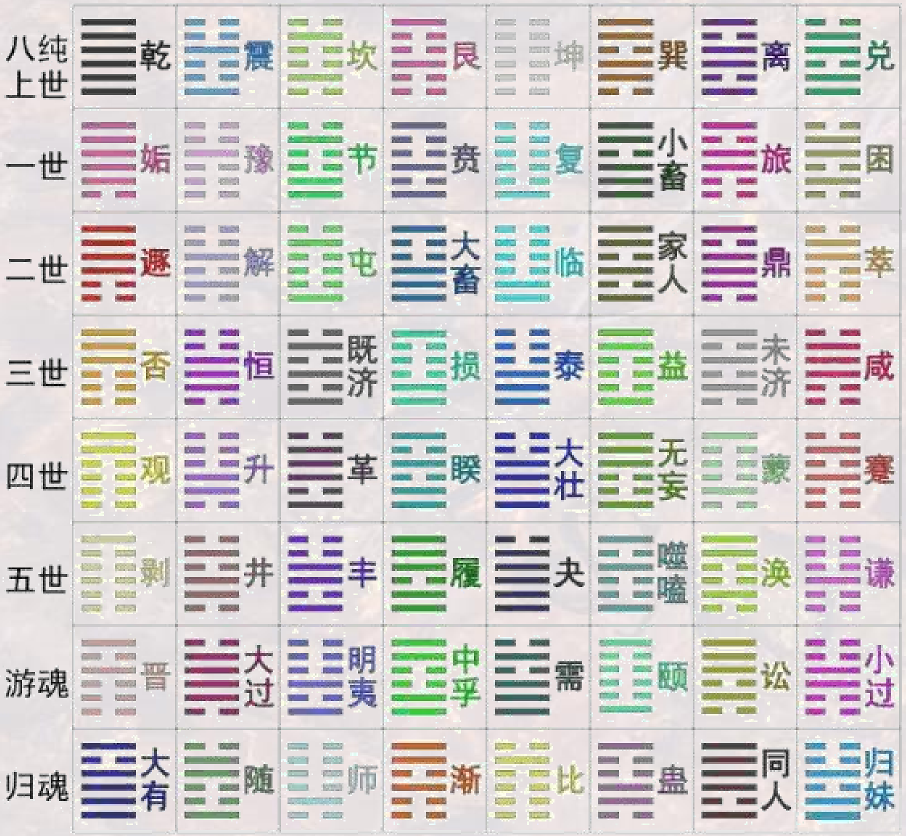
(2) 易经怎样表达自然现象？

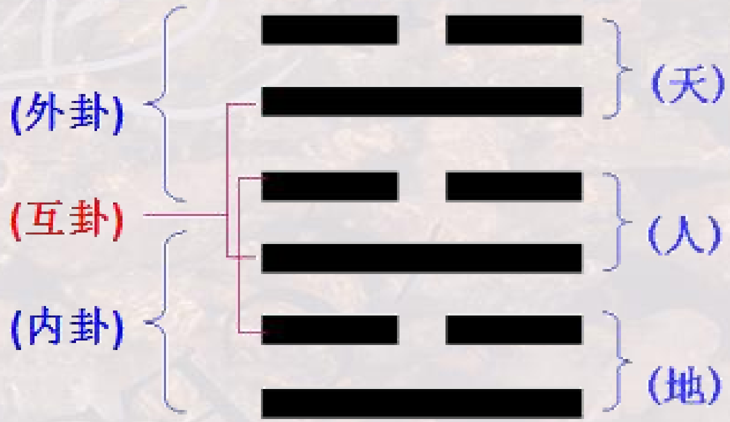
将现象抽象为符号，进行符号组合，利用符号组合表达自然现象

2.1.1 语义符号化的经典案例： 0和1与易经
(2) 易经怎样表达自然现象？

更多的组合，更多的语义，更多的变化

符号化的六十四卦图

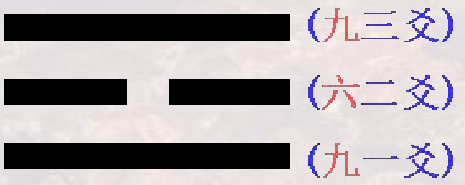




(六画卦)

(3) 易经怎样区分各种组合要素？

符号化的关键是区分与命名---术语体系

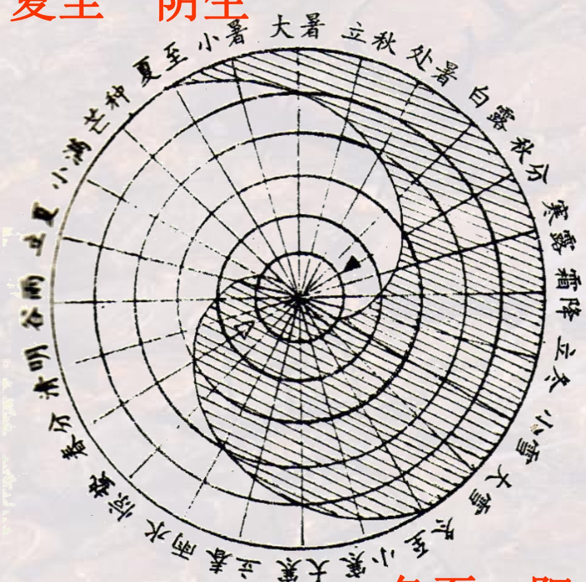


(三画挂)

待区分的	命名
⚋	阴(六)
⚊	阳(九)
三个阴阳构成的一个组合	卦
一个组合中的某一位置	爻
三画阴阳可能出现的八种组合	乾、坤、坎、离、艮、兑、震、巽
一卦中的三个位置	一爻、二爻、三爻
一个位置可能出现阴和阳	阳(九)爻，阴(六)爻
一个位置可能出现阴和阳，结合卦中不同位置组合	九一爻、六一爻、九二爻、六二爻、九三爻、六三爻

2.1.1 语义符号化的经典案例： 0和1与易经
(4) 易经怎样研究自然现象的变化？

符号化的目的是基于符号的演算--符号组合的变化方式



“卦”之间的变化规律是什么？
“卦”及之间变化反映的语义又是什么？

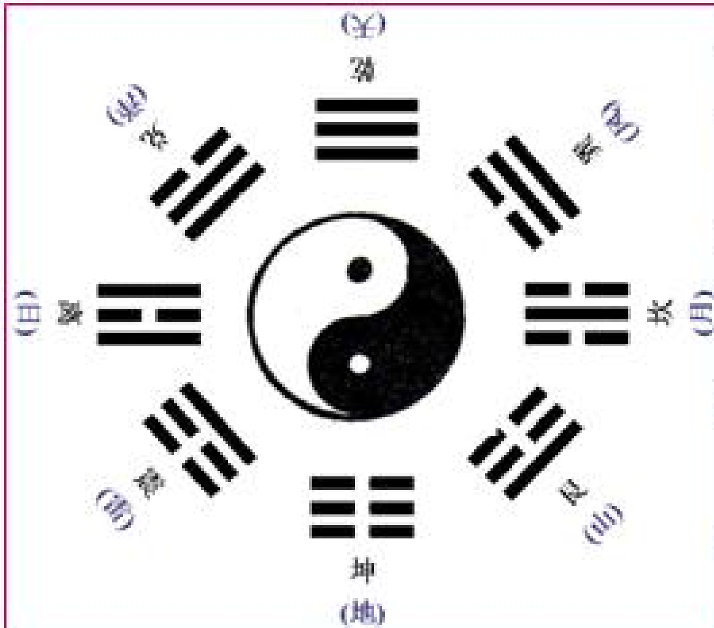
阴、阳
卦、爻
卦变、爻变

生命演化规律

2.1.1 语义符号化的经典案例： 0和1与易经
(5) 易经中为什么称乾坤而不称天地?

将符号再赋予语义 ---“本”与“用”：抽象与具体化

现象	本体	用体
天(自然空间)	乾(抽象空间)	父(家庭空间)， 首 (身体空间)， 马 (动物空间)



本义： 乾为天，坤为地，震为雷为动，巽为风为入，坎为月为水为险，离为日为火，艮为山为止，兑为泽为悦。----自然现象

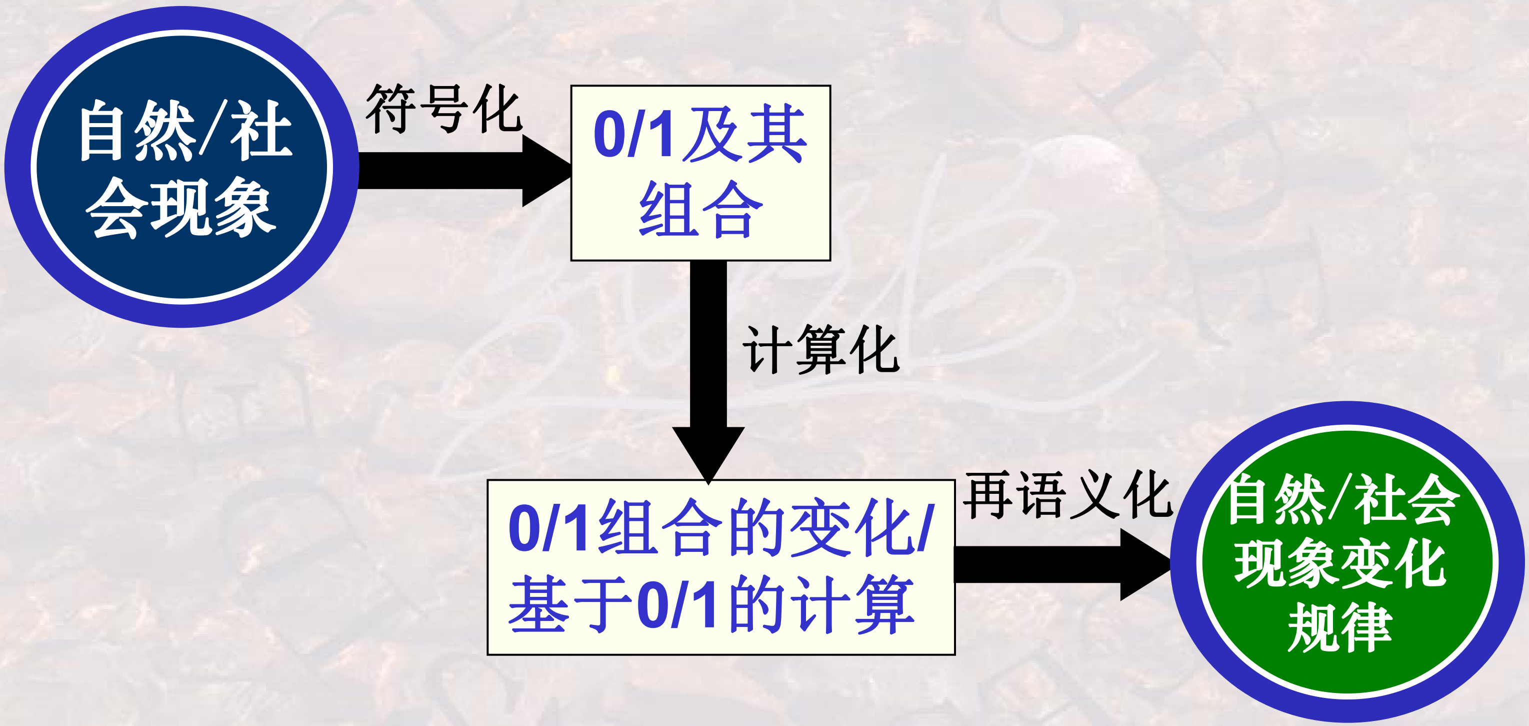
用1：乾为父，坤为母，震为长男，巽为长女，坎为中男，离为中女，艮为少男，兑为少女。---“家庭”空间

用2：乾为首，坤为腹，震为足，巽为股，坎为耳，离为目，艮为手，兑为口。 ----“身体”空间

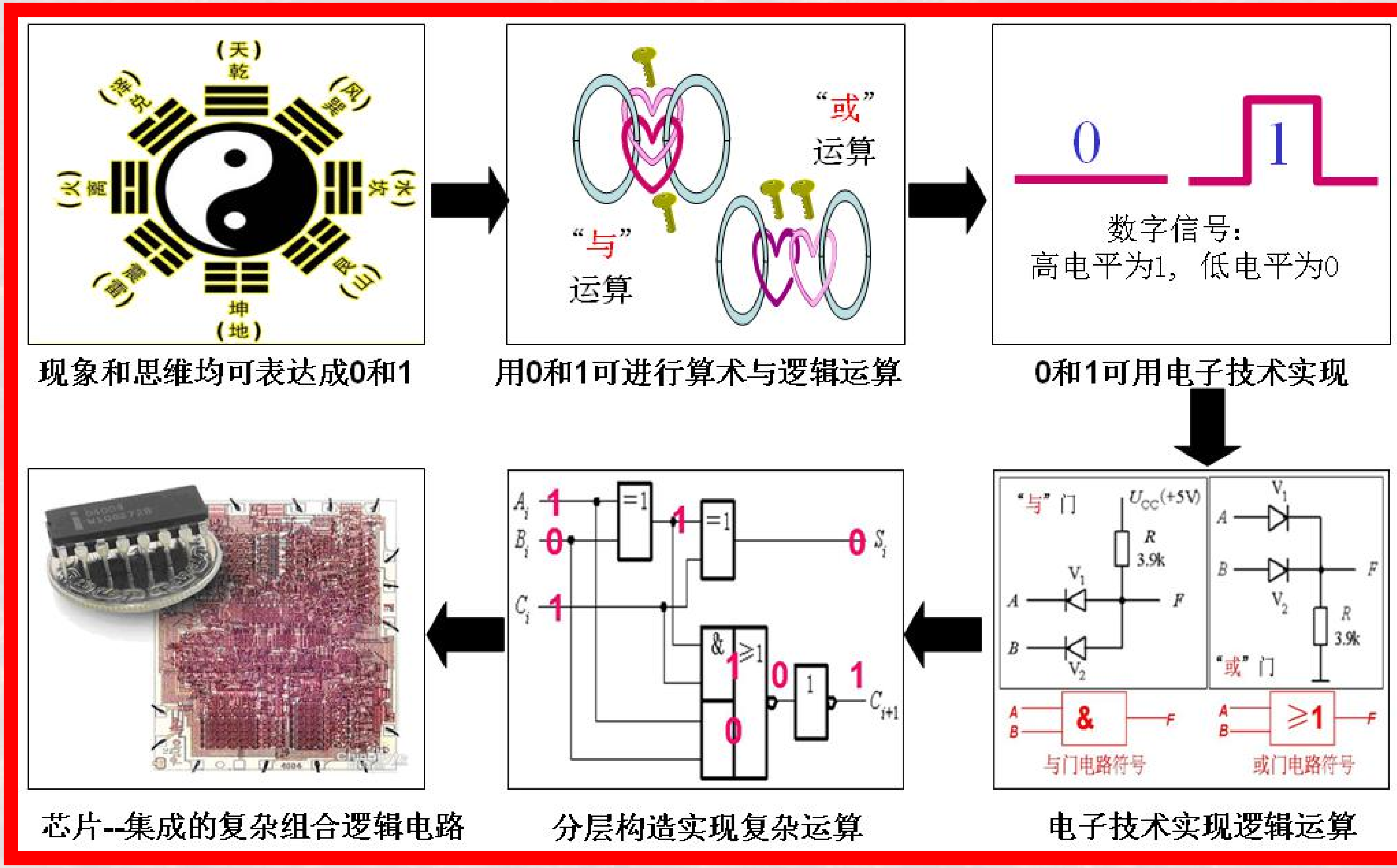
用3：乾为马，坤为牛，震为龙，巽为鸡，坎为豕，离为雉，艮为狗，兑为羊。 ----“动物世界”空间

2.1.1 语义符号化的经典案例： 0和1与易经
(6) 小结?

由“易经”看“符号化及符号计算”？



2.1.1 语义符号化的经典案例： 0和1与易经
(6) 易经在“符号化-计算化-自动化”思维中的位置？



现象和思维均可表达成0和1　　用0和1可进行算术与逻辑运算　　0和1可用电子技术实现

芯片--集成的复杂组合逻辑电路　　分层构造实现复杂运算　　电子技术实现逻辑运算

语义符号化 ➔ 符号计算化 ➔ 计算0(和)1化

➔ 0(和) 1自动化➔ 分层构造化➔ 构造集成化；

2.1.2 思维方式与逻辑运算：“0/1与逻辑

2.1.2 思维方式与逻辑运算：0和1与逻辑
(1) 什么是逻辑?

逻辑是指事物因果之间所遵循的规律，是现实中普适的思维方式

◆ 逻辑的基本表现形式是命题与推理，推理即依据由简单命题的判断推导得出复杂命题的判断结论的过程。命题由语句表述，即内容为“真”或为“假”的一个判断语句！

例如　在一次中学生测验中，有三位老师做了预测：A.学习委员及格（改为：有人及格）；B.有人不及格；C.全班都不及格。在考试后证明只有一个老师的预测是对的，请问谁对谁错？

命题A：“学习委员及格(改为：有人及格)”；

命题B：“有人不及格”；

命题C：“全班都不及格”；

由题目假设和命题之间关系得出“已知”：A、B、C只有一个为真

如果A真，则C假；如果C真，则A假；

如果B真，而A, C可能有一个为真，与题矛盾，所以B为假。

如果B假，则“全班都及格”为真，而由此推断C为假。

由上“已知”，推理：A为真。

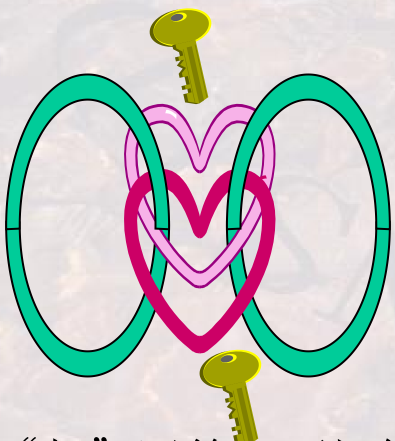
2.1.2 思维方式与逻辑运算：0和1与逻辑
(2) 有哪些基本的逻辑运算操作？

思维的符号化及其计算----基本逻辑运算

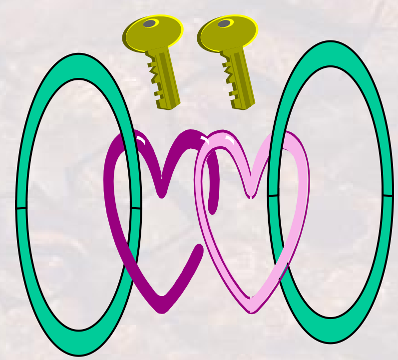
一个命题由**X, Y, Z**等表示，其值可能为“真”或为“假”。

则两个命题**X, Y**之间是可以进行运算的：

- **“与”运算(AND):** 当X和Y都为真时, X AND Y也为真; 其他情况, X AND Y均为假。
- **“或”运算(OR):** 当X和Y都为假时, X OR Y也为假; 其他情况, X OR Y均为真。
- **“非”运算(NOT):** 当X为真时, NOT X为假;当X为假时, NOT X为真。
- **“异或”运算(XOR):** 当X和Y都为真或都为假时, X XOR Y为假; 否则, X XOR Y为真。



“与”运算：两把钥匙都有才能开门



“或”运算：只要有任何一把钥匙便能开门

2.1.2 思维方式与逻辑运算：0和1与逻辑
(3) 怎样符号化逻辑并进行计算？

用0和1来表示逻辑运算

注: 1表示 真，0表示 假

- “与”运算**AND**：
  有0为0 ，全1为1
- “或”运算**OR**：
  有1为1，全0为0
- “非”运算**NOT**：
  非0则1，非1则0
- “异或”运算**XOR**：
  相同为0，不同为1

0
AND 0
0

0
AND 1
0

1
AND 0
0

1
AND 1
1

0
OR 0
0

0
OR 1
1

1
OR 0
1

1
OR 1
1

NOT 0
1

NOT 1
0

0
XOR 0
0

0
XOR 1
1

1
XOR 0
1

1
XOR 1
0

2.1.2 思维方式与逻辑运算：0和1与逻辑
(3) 怎样符号化逻辑并进行计算?

将逻辑表达为0和1及其运算

1 ---真　　　0 ---假

一个命题用**A**、**B**等符号表达，其中符号的值可能为**0**，也可能为**1**

命题**A**：“学习委员及格(改为：有人及格)”

命题**B**：“有人不及格”

命题**C**：“全班都不及格”

已知：**((A AND (NOT C)) OR ((NOT A) AND C)) = 1**

(B) AND ((A AND (NOT C)) OR ((NOT A) AND C))) = 0

(NOT B) AND (NOT C) = 1

组合形成所有可能解

{<A=1，B=0，C=0>, <A=0，B=1，C=0>，<A=0，B=0，C=1> }

将上述可能解代入已知条件，使所有已知条件都满足的便是问题的解：

<A=1，B=0，C=0>。

2.1.2 思维方式与逻辑运算：0和1与逻辑
(4) 逻辑研究有哪些？

可深入学习

1)Aristotle (亚里士多德)(公元前384－322)。古希腊哲学家：形式逻辑。

典型概念：**命题，推理，三段论**

2)Leibnitz (莱布尼茨)(1646－1716)。德国数学家：数理逻辑。

典型概念：**谓词，谓词演算**

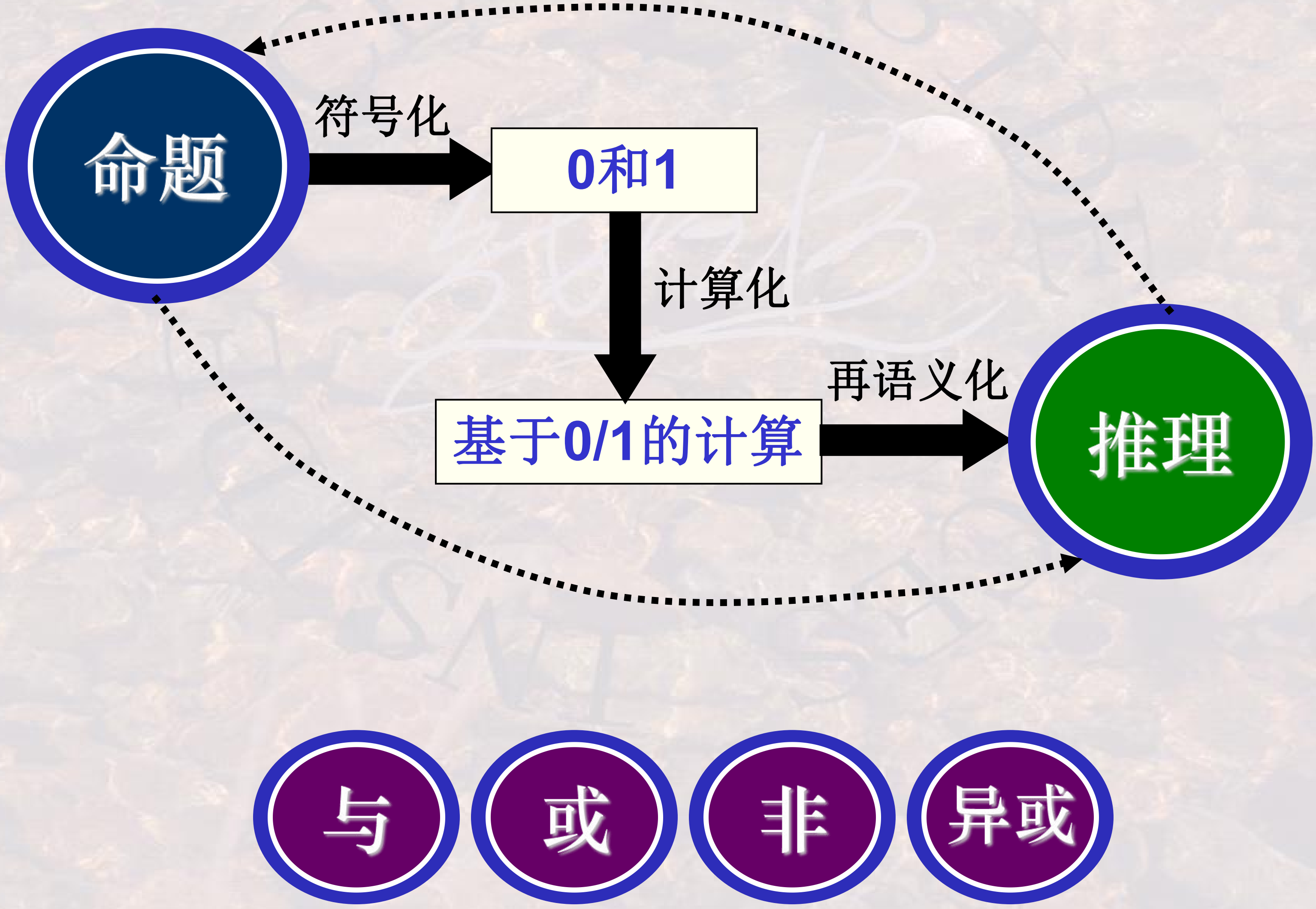
3) Boole (布尔) (1815－1864)。英国数学家，布尔代数

典型概念：**布尔量, 布尔值, 布尔运算, 布尔操作**

4)其他：时序逻辑(Temporal Logics)、模态逻辑(Modal Logics)、归纳逻辑(Inductive Logics)、模糊逻辑(Fuzzy Logics)、粗糙逻辑(Rough Logics)、非单调逻辑等

(5) 小结

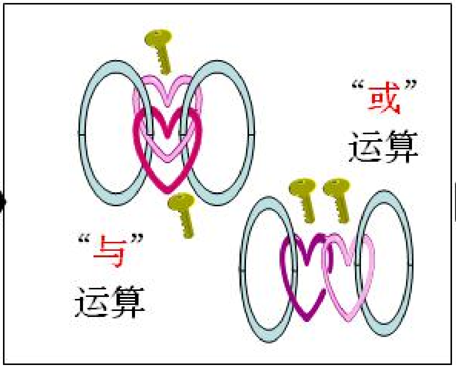
由“逻辑”看“符号化及符号计算”？



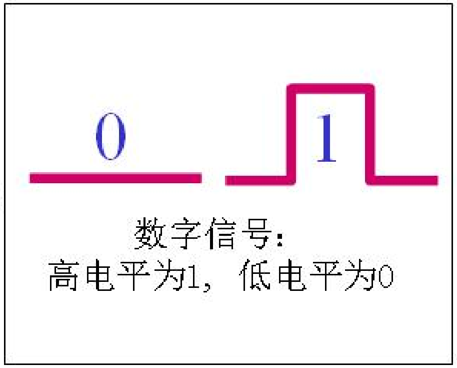
2.1.2 思维方式与逻辑运算：0和1与逻辑

(6) 逻辑在“符号化-计算化-自动化”思维中的位置?

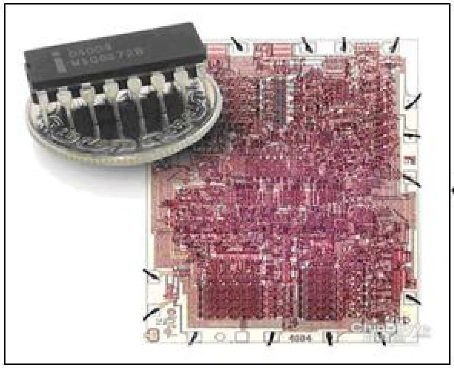
现象和思维均可表达成0和1



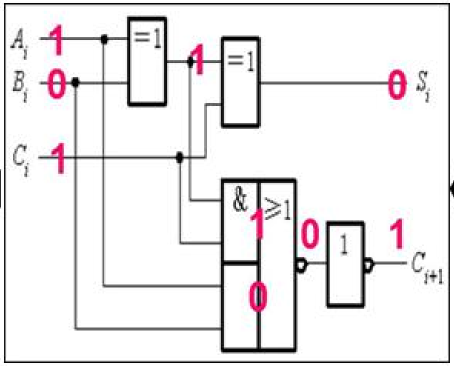
用0和1可进行算术与逻辑运算



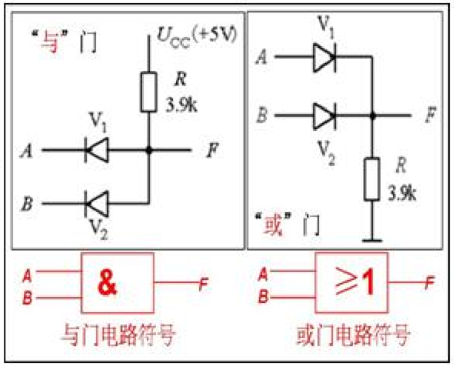
0和1可用电子技术实现



芯片--集成的复杂组合逻辑电路



分层构造实现复杂运算



电子技术实现逻辑运算

语义符号化 ➔ 符号计算化 ➔ 计算0(和)1化

➔ 0(和) 1自动化➔ 分层构造化➔ 构造集成化；

2.1.3 二进制与算术运算：0和1与数值信息

2.1.3 二进制与算术运算：0/1与数值信息
(1) 怎样用0和1表达数值性信息？

数值性信息

◆进位制：用数码和带有权值的数位来表示有大小关系的数值性信息的表示方法。

◆二进制

$2^7\ 2^6\ 2^5\ 2^4\ 2^3\ 2^2\ 2^1\ 2^0\ .\ 2^{-1}\ \ 2^{-2}$ —— *数位的权值*

7 6 5 4 3 2 1 0 .-1 -2 —— *数位*

例如： $(1\ 1\ 1\ 1\ 0\ 1\ 0\ 1\ .\ 0\ 1)_{\text{二}}$ —— *二进制数*

$$=1\times 2^7+1\times 2^6+1\times 2^5+1\times 2^4+0\times 2^3+1\times 2^2+0\times 2^1$$
$$+1\times 2^0+0\times 2^{-1}+1\times 2^{-2} = (245.25)_{\text{十}}$$

2.1.3 二进制与算术运算：0/1与数值信息

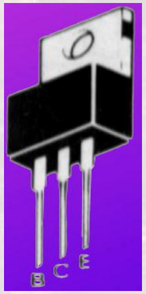
(2) 为什么要用二进制?

基于二进制的算术运算

◆计算规则简单,与逻辑运算能够统一起来；元器件容易实现。

加法运算规则				
	0	1	0	1
	+ 0	+ 0	+ 1	+ 1
	0	1	1	0

减法运算规则				
	0	1	0	1
	− 0	− 0	− 1	− 1
	0	1	1	0



不考虑进位 $\begin{cases} S_i = A_i \text{ XOR } B_i \\ C_{i+1} = A_i \text{ AND } B_i \end{cases}$

考虑进位 $\begin{cases} S_i = (A_i \text{ XOR } B_i) \text{ XOR } C_i \\ C_{i+1} = ((A_i \text{ XOR } B_i) \text{ AND } C_i) \text{ OR } (A_i \text{ AND } B_i) \end{cases}$

$$\begin{array}{rr} & A_i \\ & B_i \\ + & C_i \\ \hline C_{i+1} & S_i \end{array}$$

数值性信息

◆r进制：

r^{n-1} r^{n-2} …. … r^2 r^1 r^0 . r^{-1} r^{-2}…… r^{-m} —— *数位的权值*

n-1 n-2 … 2 1 0 . -1 -2 … -m —— *数位*

$(d_{n-1}d_{n-2}\ldots\ldots d_2d_1d_0 . d_{-1}d_{-2}\ldots\ldots d_{-m})_r$ —— *r进制数*

$=d_{n-1}r^{n-1} + d_{n-2}\, r^{n-2}+\cdots+d_2\, r^2 + d_1\, r^1 + d_0\, r^0 + d_{-1}\, r^{-1} + d_{-2}\, r^{-2} + \cdots + d_{-m}\, r^{-m}$

$$= \sum_{i=-m}^{n-1} d_i r^i$$

- 十六进制：0,1,2,3,4,5,6,7,8,9,A(10), B(11), C(12), D(13), E(14), F(15)
- 八进制：0,1,2,3,4,5,6,7
- 十进制：0,1,2,3,4,5,6,7,8,9

$(365.2)_{10}$, $(11011.01)_2$, $(3460.32)_{八}$, $(596.12)_{十六}$

进位计数制之间的转换

十进制转换为r进制—整数部分转换

已知十进制整数N，求r进制的数码串$d_{n-1}d_{n-2}\ldots d_1d_0$

$$N = (d_{n-1}d_{n-2}\ldots\ldots d_2d_1d_0)_r$$

$$=d_{n-1}r^{n-1} + d_{n-2}\, r^{n-2}+\cdots+d_2\, r^2 + d_1\, r^1 + d_0\, r^0$$

(N/r)的余数为d_0

$((N/r)/r)$的余数为d_1

$(((N/r)/r)/r)$的余数为d_2

……

$(\ldots(((N/r)/r)/r)\ldots/r)$的余数为$d_{n-1}$

“除基取余”

$(245)_{十} = (F5)_{十六}$

$$\begin{array}{r|l}
16 & 245 \\ \hline
16 & 15 \cdots\cdots 5 \\ \hline
 & 0 \cdots\cdots 15(F)
\end{array}$$

进位计数制之间的转换

十进制转换为r进制—小数部分转换

已知十进制小数N，求r进制的数码串$0.d_{-1}d_{-2}\ldots d_{-m}$

$$N = (0.d_{-1}d_{-2}\ldots\ldots d_{-m})_r \quad = d_{-1}r^{-1} + d_{-2}r^{-2} + \cdots + d_{-m}r^{-m}$$

$(N\times r)$的整数部分为d_{-1}

$((N\times r)\times r)$的整数部分为d_{-2}

$(((N\times r)\times r)\times r)$的整数部分为$d_{-3}$

… ...

$(\ldots(((N\times r)\times r)\times r)\ldots\times r)$的整数部分为$d_{n-1}$

注：每次相乘都是去掉整数后的小数部分相乘。

若最终为0，则结束，否则计算到要求位数即可。

$(0.525)_{十}$

$= (0.8666)_{十六}$

$$\begin{array}{rr}
 & 0.525 \\
 & \times\ 16 \\
\hline
8 & .4 \\
 & \times\ 16 \\
\hline
6 & .4 \\
 & \times\ 16 \\
\hline
6 & .4 \\
 & \times\ 16 \\
\hline
6 & .4
\end{array}$$

“乘基取整”

数值性信息

◆示例

$(7\ 5\ 3\ .\ 3\ 7)_{八} =\ 7\ 5\ 3\ .\ 3\ 7\ \ O$

$=7\times 8^{2}+5\times 8^{1}+3\times 8^{0}+3\times 8^{-1}+7\times 8^{-2}$

$=(491.484375)_{十}$

$(7\ 5\ 3\ .\ 3\ 7)_{十六} = 7\ 5\ 3\ .\ 3\ 7\ \ H = 0x\ 7\ 5\ 3\ .\ 3\ 7$

$=7\times 16^{2}+5\times 16^{1}+3\times 16^{0}+3\times 16^{-1}+7\times 16^{-2}$

$=(1875.2148)_{十}$

$(7\ 5\ 3\ .\ 3\ 7)_{十二}$

$=7\times 12^{2}+5\times 12^{1}+3\times 12^{0}+3\times 12^{-1}+7\times 12^{-2}$

$=(1071.2986)_{十}$

同一个数串，由于进位制不同其所表达的数值大小也是不同的

245的十进制表示记为：

245

245的二进制表示记为：

11110101

245的八进制表示记为：

365

245的十六进制表示记为：

F5

同一个数值，用不同进位制表达，结果也是不同的

(4) 数值的符号如何表示呢？

数值的正负符号处理：机器数的原码、反码和补码

真实数值 (带符号的 n 位二进制数)	十进制数	机器数(n+1 位二进制数,其中第 n+1 位表符号,0 表示正号,1 表示负号)		
		原码	反码	补码
+ 11...11	$+(2^n-1)$	0 11...11	0 11...11	0 11...11
+ 10...00	$+2^{n-1}$	0 10...00	0 10...00	0 10...00
+ 00...00	+0	0 00...00	0 00...00	0 00...00
- 00...00	-0	1 00...00	1 11...11	0 00...00
- 10...00	-2^{n-1}	1 10...00	1 01...11	1 10...00
- 11...11	$-(2^n-1)$	1 11...11	1 00...00	1 00...01
-100...00	-2^n	-	-	1 00...00
		正数的原码、反码同补码形式是一样的。最高位为 0 表示正数		
		负数的最高位为 1,表示负数。其余同真实数值的二进制数。	负数的最高位为 1,表示负数。其余在真实数值的二进制数基础上逐位取反。	负数的最高位为 1,表示负数。其余在反码基础上最低位加 1 后形成的。它的负数不包括 0，但包括 -2^n
		机器数由于受到表示数值的位数的限制，只能表示一定范围内的数。超出此范围则为“溢出”		

2.1.3 二进制与算术运算：0/1与数值信息
(5) 使用补码可使减法变加法, 你相信吗?

数值的正负符号也可和数值一样参与运算：补码运算示意

$(+7) + (+3) = (+10)$

```
    0 0111
+)  0 0011
----------
    0 1010
```

$(-5) + (-7) = (-12)$

```
    1 1011
+)  1 1001
----------
    1 0100
```

$(10) + (-3) = (7)$

```
    0 1010
+)  1 1101
----------
    0 0111
```

$(-7) + (-12) =$ 溢出

```
    1 1001
+)  1 0100
----------
    0 1101
```

加减乘除都可转换成加法来实现, 加法又可由与、或、非、异或等逻辑运算来实现

---只要实现了基本逻辑运算，便可实现任何的计算

基于二进制的算术运算

◆机器可以采用移位、逻辑运算等进行加减乘除运算。

101010B

例1: 10111 + 10011=?

$$
\begin{array}{r}
10111 \\
+)\ 10011 \\
\hline
\mathbf{101010}
\end{array}
$$

01110

例2: 00111 × 00010 = ?

$$
\begin{array}{l}
00111 \\
\xleftarrow{\quad} \text{左移一位} \\
01110
\end{array}
$$

投票 最多可选1项

设置

补码运算的符号位和数值位一样可以参与运算

A 对

B 错

C 不知道

提交

投票 最多可选1项

设置

什么情况下会产生补码运算的溢出

A 同符号数相加

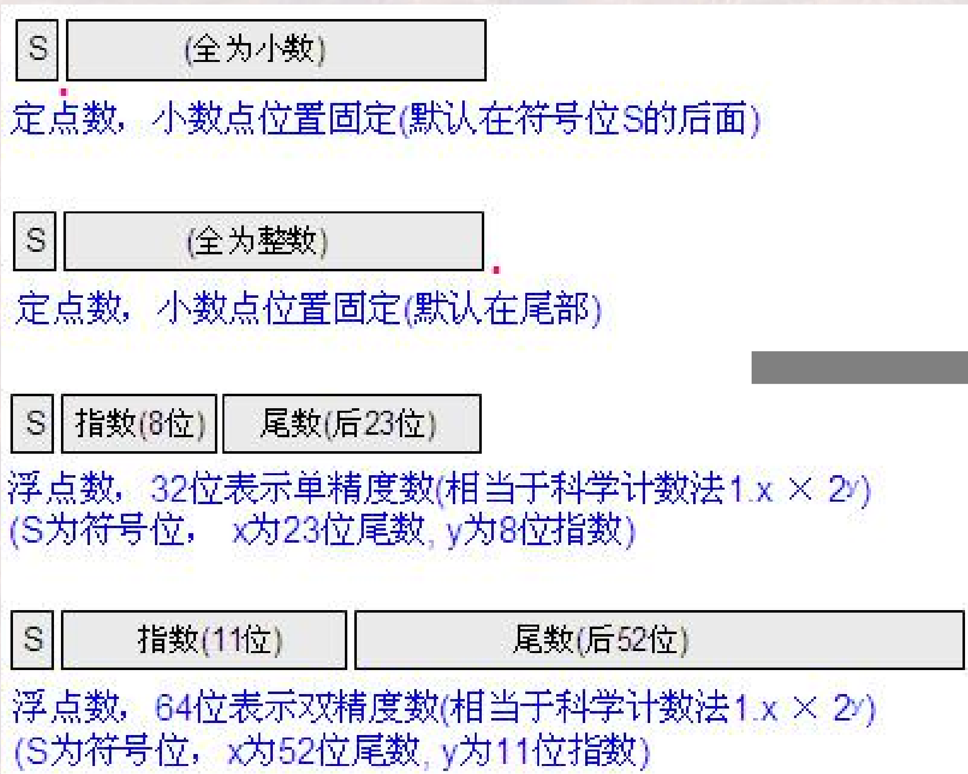
B 异符号数相加

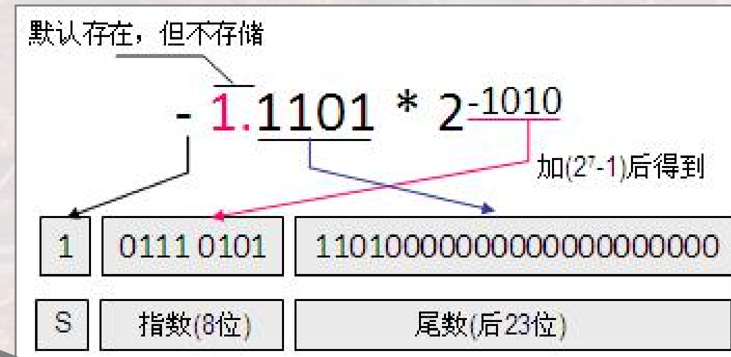
C 同符号数相加，异符号数相加

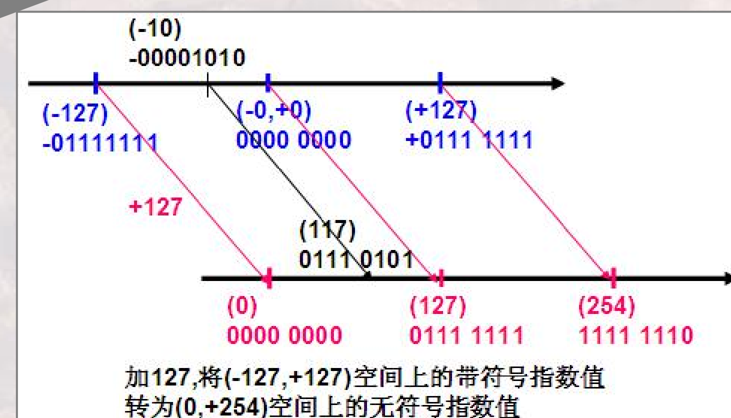
提交

(7) 小数点如何处理呢？

数值的小数点的处理：定点数与浮点数







2.1.3 二进制与算术运算：0/1与数值信息
(8) 信息的度量单位是什么？

信息的基本度量单位

bit　Binary Digit/1位二进制位/0和1

Byte 字节, **8**位二进制位

1KB = 2^{10}字节 **(**市场约**1,000**字节**)**

1MB = 2^{10}KB (市场约**1,000,000**字节**)**

1GB = 2^{10}MB (市场约**1,000,000,000**字节**)**

1TB = 2^{10}GB = 2^{20}MB

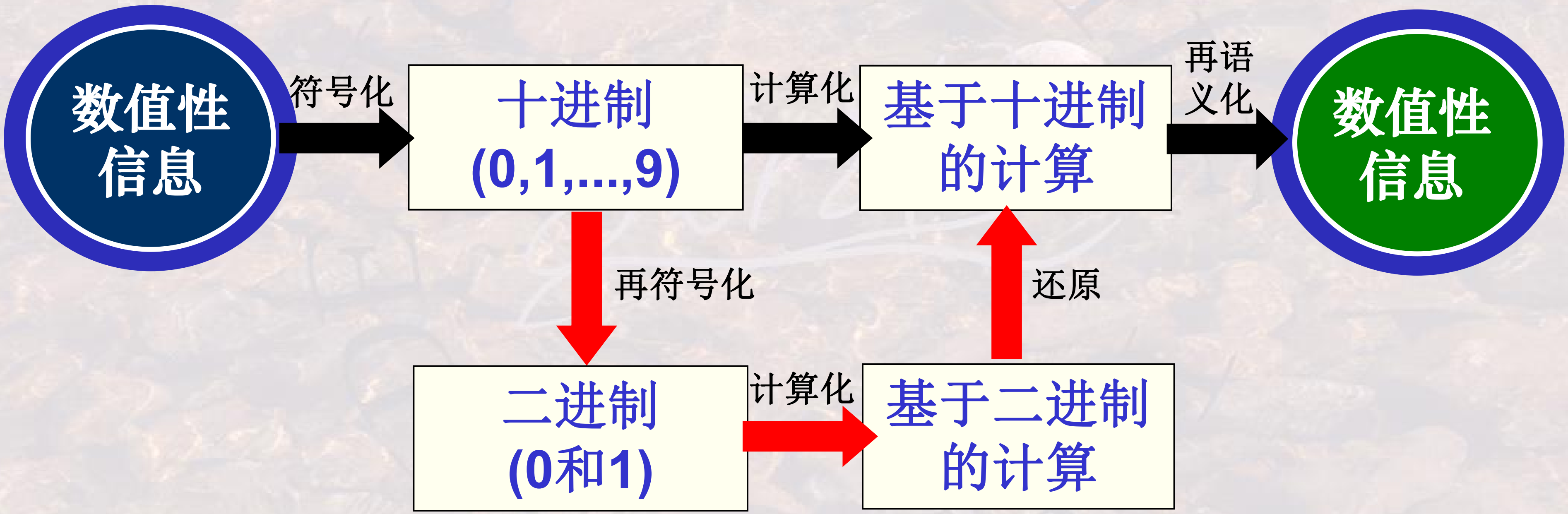
1PB = 2^{10}TB = 2^{30}MB

1EB = 2^{10}PB = 2^{40}MB

注意：2的幂次方为计算单位

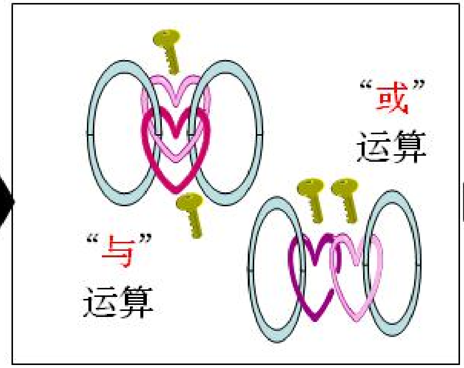
2.1.3 二进制与算术运算：0/1与数值信息
(9) 小结

由数值性信息看符号化及其计算

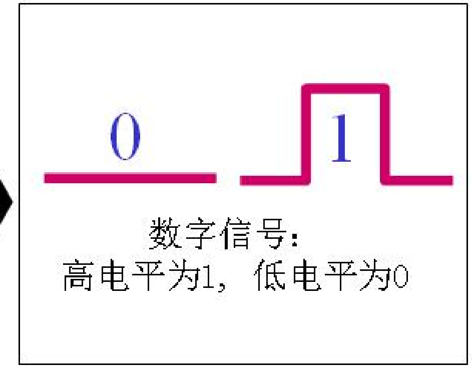


2.1.3 二进制与算术运算：0/1与数值信息
(10) 数值表达与计算在“符号化-计算化-自动化”思维中的位置？

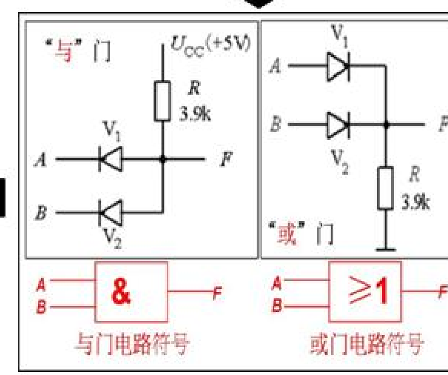
现象和思维均可表达成0和1



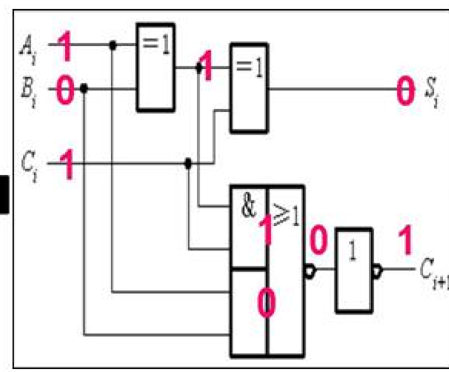
用0和1可进行算术与逻辑运算



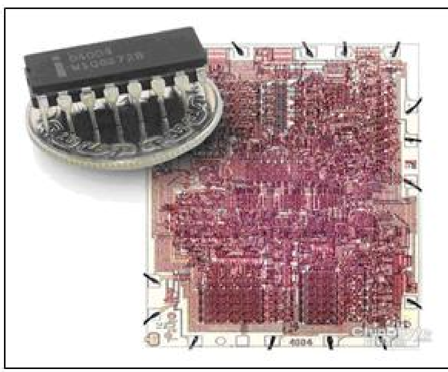
0和1可用电子技术实现



电子技术实现逻辑运算



分层构造实现复杂运算



芯片--集成的复杂组合逻辑电路

语义符号化 ➔ 符号计算化 ➔ 计算0(和)1化

➔ 0(和) 1自动化➔ 分层构造化➔ 构造集成化；

2.1.4 编码与符号运算： 0/1 与非数值信息

2.1.4 编码与符号运算：0/1 与非数值信息
(1) 编码的概念

非数值性信息可以用编码表示

◆**编码：** 编码是以若干位数码或符号的不同组合来表示非数值性信息的方法，它是人为地将若干位数码或符号的每一种组合指定一种唯一的含义。

例如：0----男，1----女

再如：000----星期一　001----星期二　010----星期三
011----星期四　100----星期五　101----星期六
110----星期日

再如：000----一院　001----二院　010----三院
011----四院　100----五院　101----六院
110----七院　111----其他

编码的三个主要特征

- **唯一性**：每一种组合都有确定的唯一性的含义
- **公共性**：所有相关者都认同、遵守、使用这种编码
- **易于记忆/便于识认性**：有一定规律

ASCII码----英文字母符号的编码

◆**ASCII码**是英文字母与符号的0,1型编码方法，是用7位0和 1 的不同组合来表示10个数字、26个英文大写字母、26个英文小写字母及其一些特殊符号的编码方法，是信息交换的标准编码。

◆ASCII码:American Standard Code for Information Interchange

B_7	B_6	B_5	B_4	B_3	B_2	B_1	B_0	
0	x	x	x	x	x	x	x	
0	0	1	1	0	0	0	1	“1”
0	1	0	0	1	1	1	0	“N”

完整的ASCII码表

$b_6b_5b_4$ \ $b_3b_2b_1b_0$	000	001	010	011	100	101	110	111
0000	NUL	DLE	SP	0	@	P	`	P
0001	SOH	DC1	!	1	A	Q	a	Q
0010	STX	DC2	〃	2	B	R	b	R
0011	ETX	DC3	#	3	C	S	c	S
0100	EOT	DC4	$	4	D	T	d	T
0101	ENQ	NAK	%	5	E	U	e	U
0110	ACK	SYN	&	6	F	V	f	V
0111	BEL	ETB	'	7	G	W	g	W
1000	BS	CAN	(	8	H	X	h	X
1001	HT	EM	)	9	I	Y	i	Y
1010	LF	SUB	*	:	J	Z	j	Z
1011	VT	ESC	+	;	K	[	k	{
1100	FF	FS	,	<	L	]	l	¦
1101	CR	GS	−	=	M	\	m	}
1110	SO	RS	.	>	N	^	n	~
1111	SI	US	/	?	O	_	o	DEL

2.1.4 编码与符号运算：0/1 与非数值信息
(2) 用0、1组合编码字母与符号--- ASCII码

ASCII编码的规律

每**8**位为一个字符，最高位为0

- ✓41H ~ 5AH: “A” ~ “Z”
- ✓61H ~ 7AH: “a” ~ “z”
- ✓0AH: 换行符号LF
- ✓0DH: 回车符号CR
- ✓30H ~ 39H: “0” ~ “9”

01000001	A	41 H
01000010	B	42 H
01000110	F	46 H

信息	存储	解析规则
We are students	01010111 01100101 00100000 01100001 01110010 01100101 00100000 01110011 01110100 01110101 01100100 01100101 01101110 01110100 01110011	0/1串按8位分隔一个字符，查找ASCII码表映射成相应符号

十个数字符号的编码----BCD码

BCD码：**Binary Coded Decimal**(二-十进制编码)是用 4 位 0 和 1 的不同组合，按照与进位制保持一致的关系，来表示**10**个十进制数字的方法。

10个数字，只需4位0/1数码即可

十进制	0	1	2	3	4
BCD码	0000	0001	0010	0011	0100
十进制	5	6	7	8	9
BCD码	0101	0110	0111	1000	1001

2.1.4 编码与符号运算：0/1 与非数值信息
(2续) 信息在计算机中为什么需要区分不同的类型？

同一信息不同表示方法的对比

245的十进制记为**245**

245的二进制记为 **11110101**

245的八进制记为**365**

245的十六进制记为**F5**

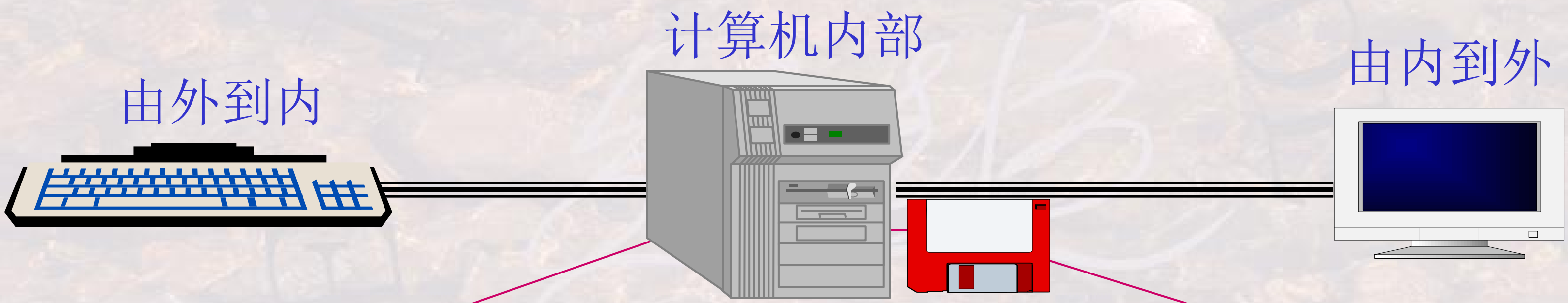
245的BCD码记为**<u>0010</u> <u>0100</u> <u>0101</u>**

245的ASCII码记为**<u>00110010</u> <u>00110100</u> <u>00110101</u>**

2.1.4 编码与符号运算：0/1 与非数值信息
(3)用字母符号组合编码汉字、用0和1编码汉字的字形

汉字的编码

◆**汉字内码**：汉字在计算机内部采用汉字内码存储，汉字内码是一两字节且最高位均为1的0,1型编码



"大"

用0和1编码汉字,每个汉字在计算机内部由 2个字节表示

$b_7\ b_6\ b_5\ b_4\ b_3\ b_2\ b_1\ b_0 \quad b_7\ b_6\ b_5\ b_4\ b_3\ b_2\ b_1\ b_0$

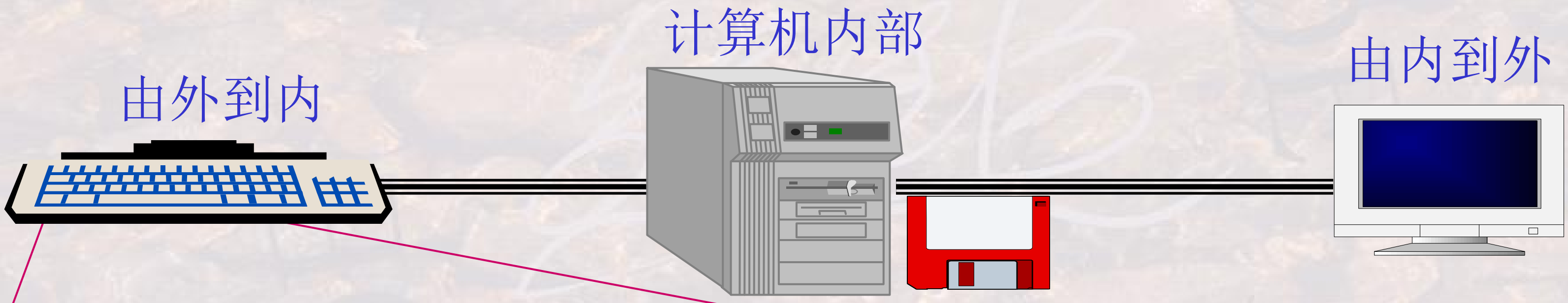
国标码 0 0110100 0 1110111

↓ ↓

(机)内码 1 0110100 1 1110111

汉字的编码

◆**汉字输入码**是用键盘上的字母符号编码每一汉字的编码, 它使人们通过键入字母符号代替键入汉字。



输入码有若干：拼音码、字型码、区位码… …

拼音码： xing

双拼码： x;

其中，‘x’表声母x, 而‘；’表韵母ing

五笔字型码： gajf

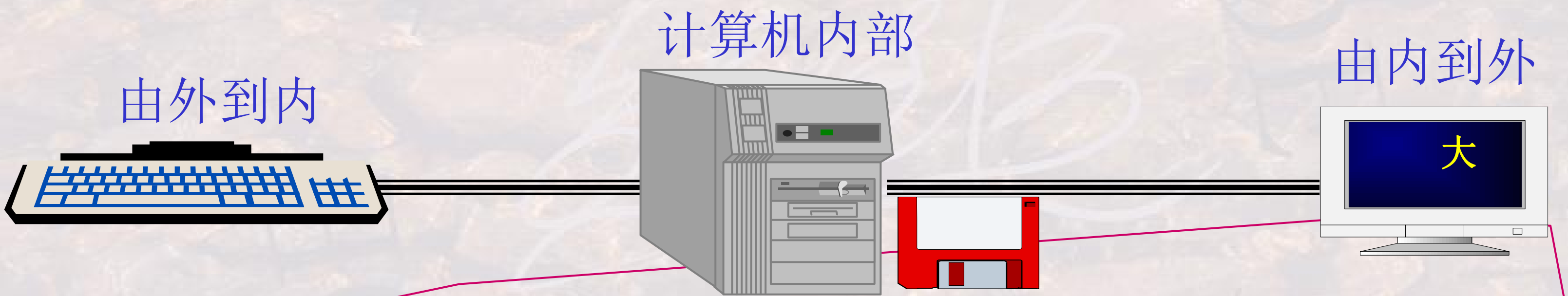
其中, g表字根“一”, a表开下的草字头, j表右侧立刀, f表下面土字

“型”

(3)用字母符号组合编码汉字、用0和1编码汉字的字形

汉字的编码

◆汉字字形码是用0和1编码无亮点和有亮点像素，形成汉字字形的一种编码。依据字形码通过显示器或打印机输出汉字。



用0和1编码无亮点和有亮点形成字形信息，便于显示… …

汉字字形码是一种字模点阵码。也有不同的处理汉字点阵信息的编码，如向量编码等

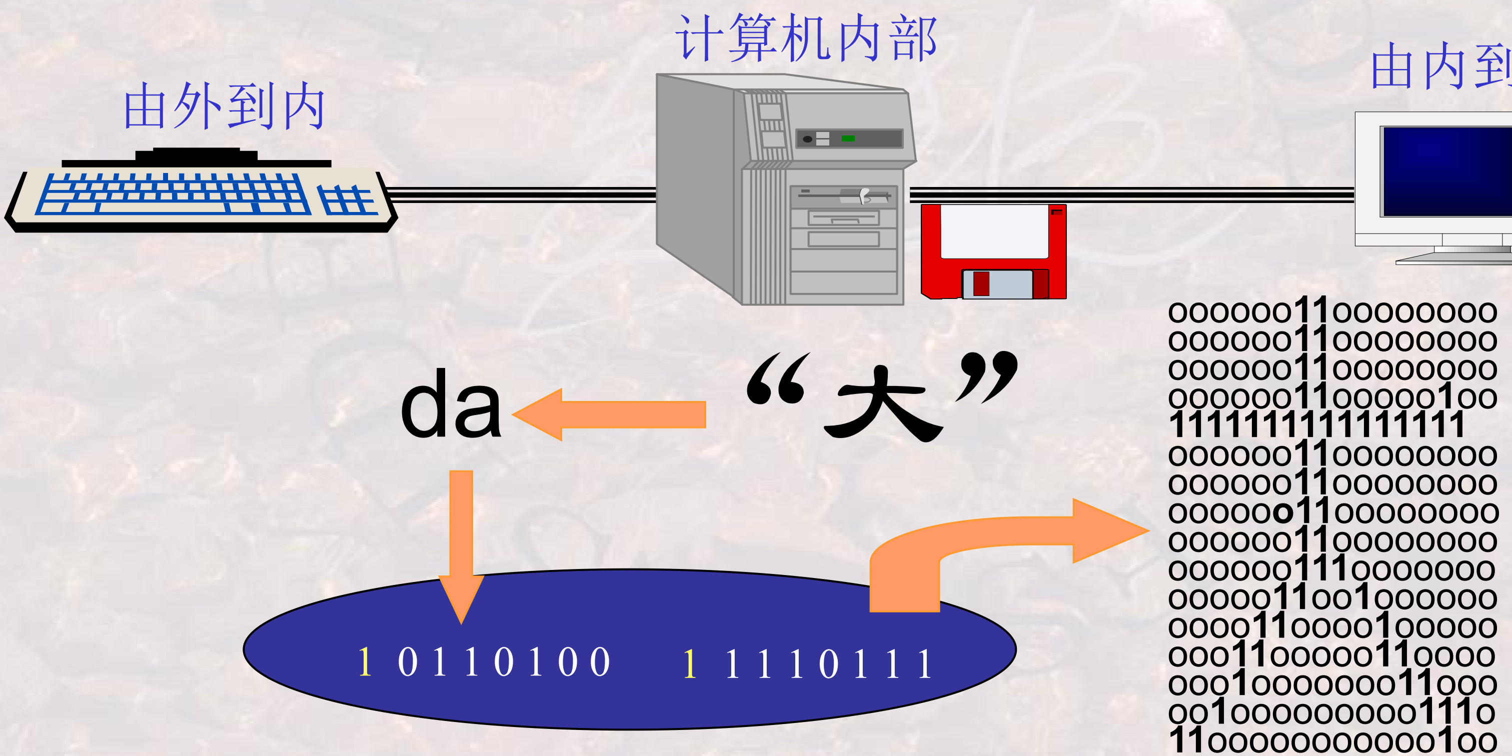
“大”

0000001100000000
0000001100000000
0000001100000000
0000001100000100
1111111111111111
0000001100000000
0000001100000000
0000001100000000
0000001100000000
0000001110000000
0000011001000000
0000110000100000
0001100000110000
0001000000011000
0010000000001110
1100000000000100

(3)用字母符号组合编码汉字、用0和1编码汉字的字形

汉字的编码

◆**汉字处理过程**：通过汉字外码输入, 以汉字内码存储, 以汉字字形码输出



2.1.4 编码与符号运算：0/1 与非数值信息
(3)用字母符号组合编码汉字、用0和1编码汉字的字形

进一步学习:

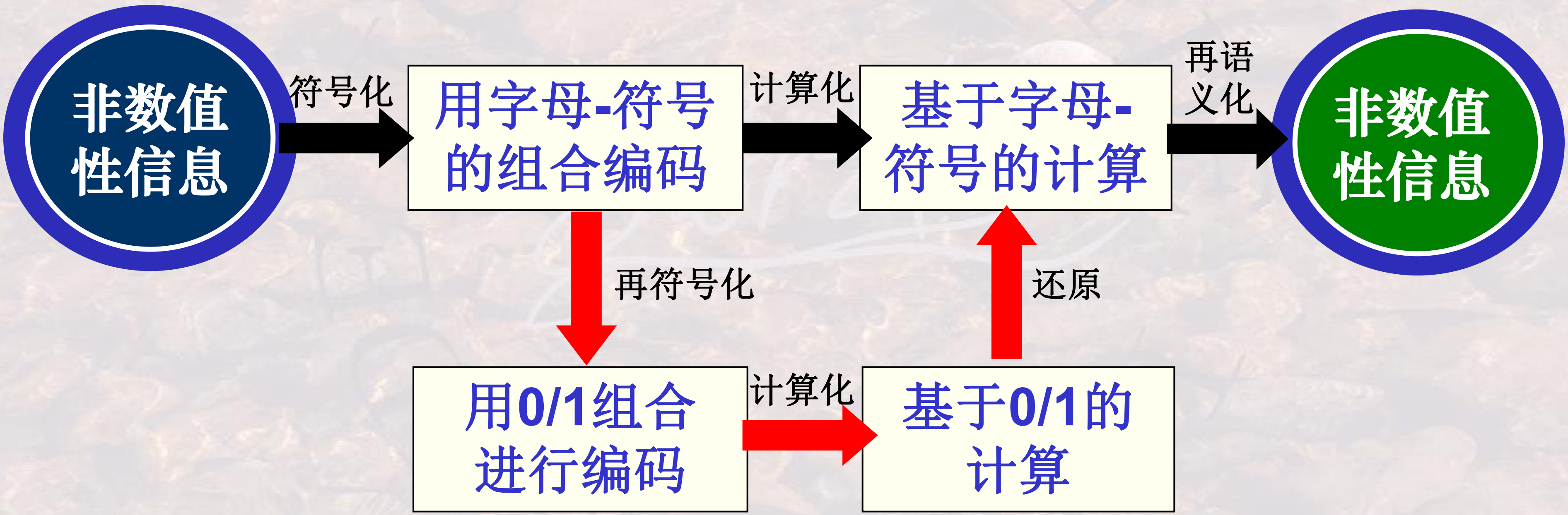
◆**标准ASCII码**：8位0,1型编码，最高位始终为0

◆**扩展ASCII码**：8位0,1型编码，最高位为0时为标准ASCII码；最高位为1时为扩展ASCII码。

◆**UNICODE:** Unicode是国际组织制定的可以容纳世界上所有文字和符号的字符编码方案。Unicode用数字0-0x10FFFF来映射所有的字符(最多可以容纳1114112个字符，或者说有1114112个码位，码位就是可以分配给字符的数字)。具体实现时，再将前述唯一确定的码位按照不同的编码方案映射为相应的编码，有UTF-8、UTF-16、UTF-32等几种编码方案。

小结

分层次符号化、分层次编码与计算



用0和1表达

基于0和1计算

编码

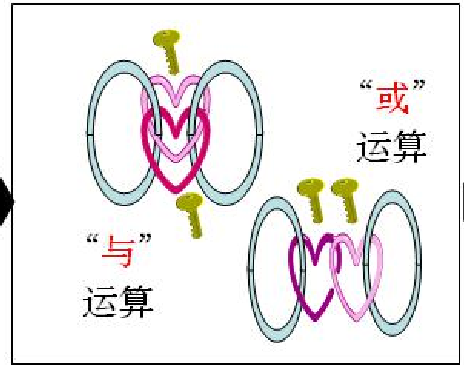
ASCII码

汉字内码

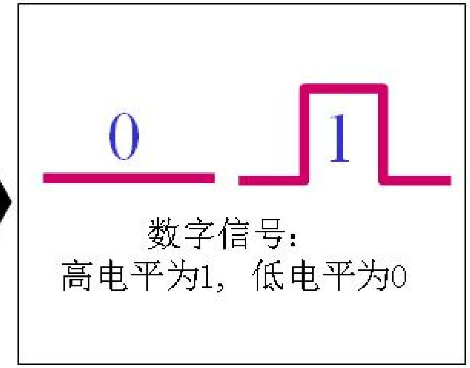
2.1.4 编码与符号运算：0/1 与非数值信息

小结：非数值性信息表达与计算在“符号化-计算化-自动化”思维中的位置

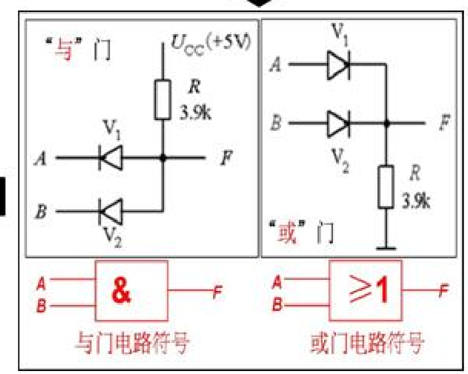
现象和思维均可表达成0和1



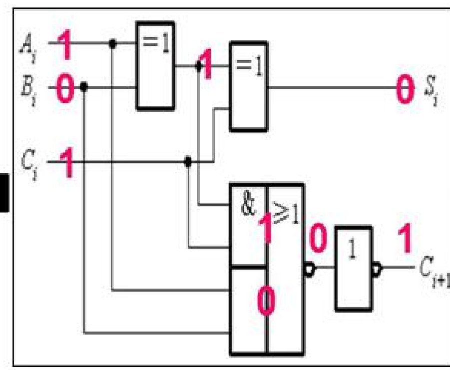
用0和1可进行算术与逻辑运算



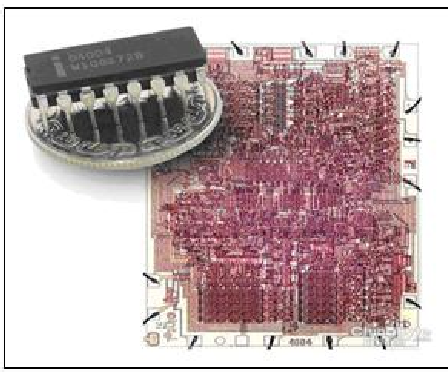
0和1可用电子技术实现



电子技术实现逻辑运算



分层构造实现复杂运算



芯片--集成的复杂组合逻辑电路

语义符号化 ➔ 符号计算化 ➔ 计算0(和)1化

➔ 0(和) 1自动化➔ 分层构造化➔ 构造集成化；

2.1.5 0和1与电子元器件

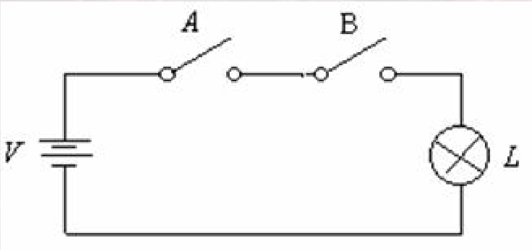
2.1.5 0和1与电子元器件

(1) 如何用电信号及电子元件表达0和1?

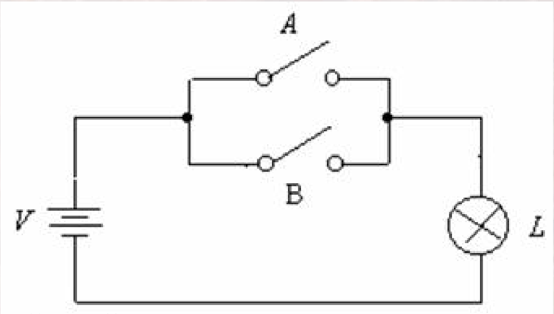
实现0和1的基本元器件: 电信号和继电器开关

数字信号：高电平为1，低电平为0

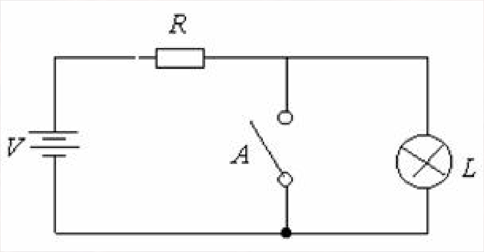
- 用继电器开关实现基本逻辑运算



“与”运算电路



“或”运算电路

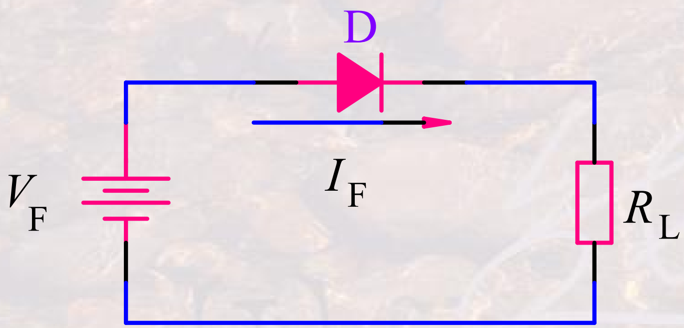


“非”运算电路

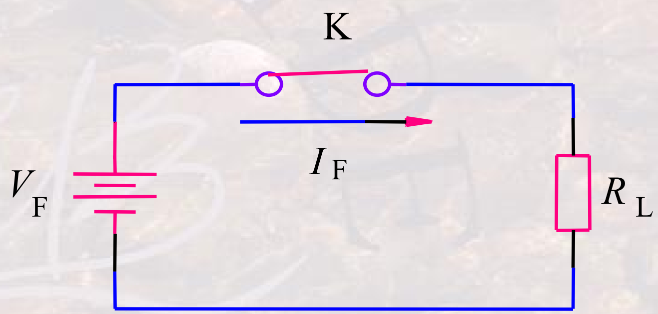
2.1.5 0和1与电子元器件
(2) 处理0和1的基本元件？

实现0和1的基本元器件: 二极管

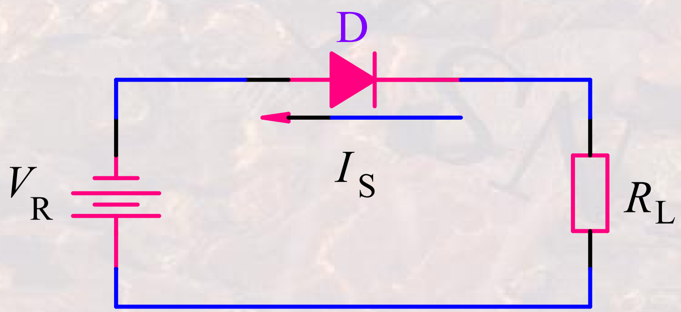
- 二极管的基本特性



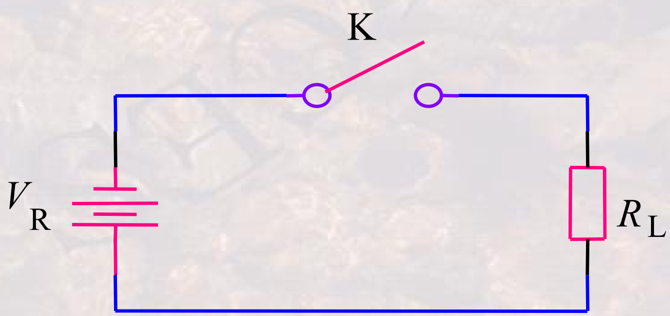
(a)



(b)



(a)



(b)

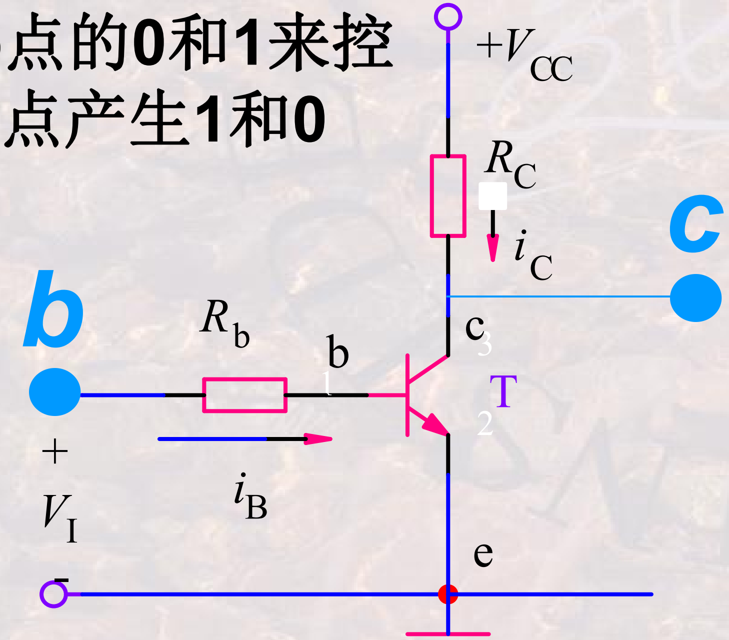
实现0和1的基本元器件: 三极管

- 三极管的基本特性:
  - 开关和放大
  - 以较小的b极电流信号可控制较大的e极流过的电流--放大。


"The first transistor ever assembled, invented in Bell Labs in 1947." Photo and text from Porticus.org, www.porticus.org/bell/belllabs_transistor.html. (Follow that link to see more historical documents and images about Bell Labs and the transistor.)

第一个三极管试验装置

用b点的0和1来控制c点产生1和0

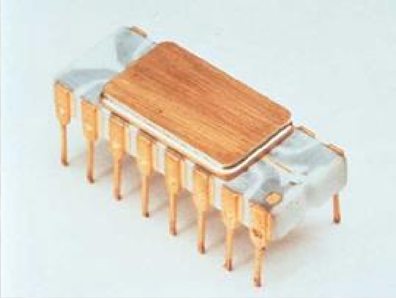


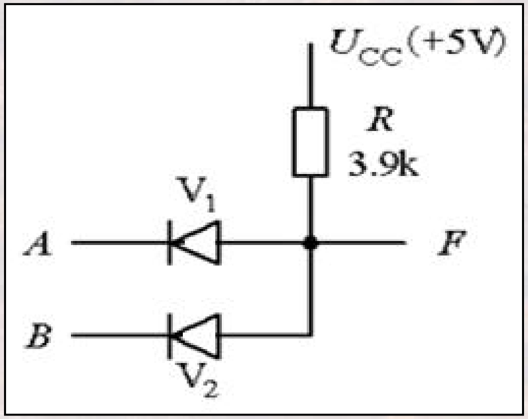
典型的三极管电路

2.1.5 0和1与电子元器件
(3) 如何用基本电子元件实现基本逻辑运算?

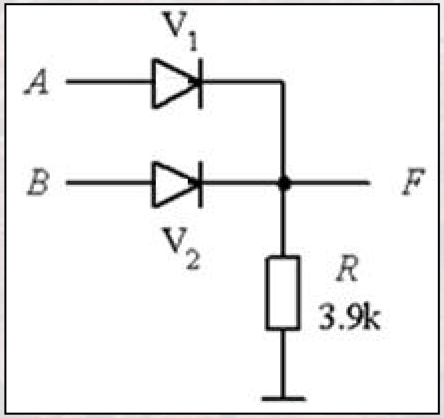
用二极管、三极管可实现基本的集成电路: 与门、或门和非门

◆这些电路被封装成集成电路(芯片)，即所谓的门电路。

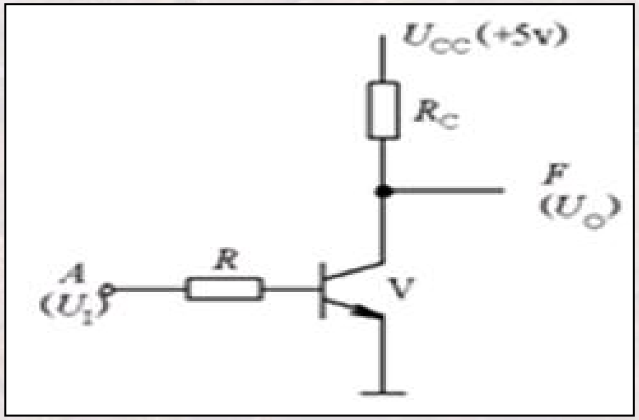




“与”门电路



“或”门电路



“非”门电路

(4) 如何用电信号及电子元件实现基本逻辑运算？

基本门电路的符号表示及其特性

◆**与门电路**：是实现逻辑与运算的集成电路，即：只有当两个输入端为高电平(1)时，则输出端为高电平(1)；否则，输出端为低电平(0)。

◆**或门电路**：是实现逻辑或运算的集成电路，即：只有当两个输入端为低电平(0)时，则输出端为低电平(0)；否则，输出端为高电平(1)。

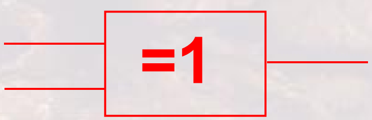
◆**非门电路**：是实现逻辑非运算的集成电路，即：当输入端为高电平(1)时，则输出端为低电平(0)；输入端为低电平(0)时，则输出端为高电平(1)。

◆**异或门电路**：是实现逻辑异或运算的集成电路，即：当两个输入端同为高电平(1)或同为低电平(0)时，则输出端为低电平(0)；否则，输出端为高电平(1)。

与门电路符号

或门电路符号

非门电路符号



异或门电路符号

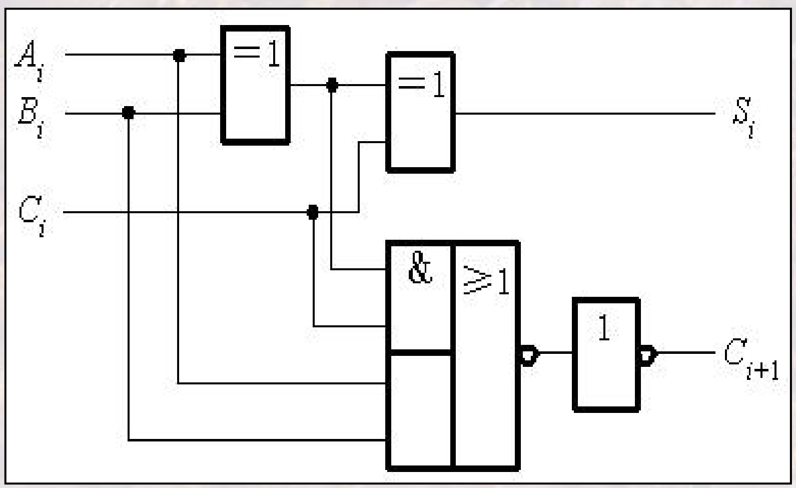
(5) 如何用已实现的基本逻辑运算(门电路)来实现更复杂的运算？

基于门电路的复杂组合逻辑电路

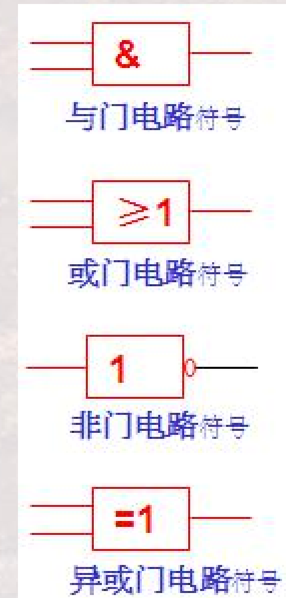
◆示例1：一位加法器的示例。

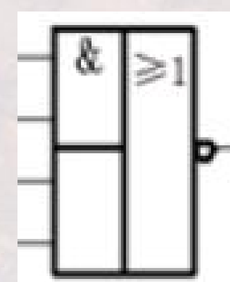
不考虑进位
$$\begin{cases} S_i = A_i \text{ XOR } B_i \\ C_{i+1} = A_i \text{ AND } B_i \end{cases}$$

考虑进位
$$\begin{cases} S_i = (A_i \text{ XOR } B_i) \text{ XOR } C_i \\ C_{i+1} = ((A_i \text{ XOR } B_i) \text{ AND } C_i) \text{ OR } (A_i \text{ AND } B_i) \end{cases}$$



$$\begin{array}{rr} & A_i \\ & B_i \\ + & C_i \\ \hline C_{i+1} & S_i \end{array}$$



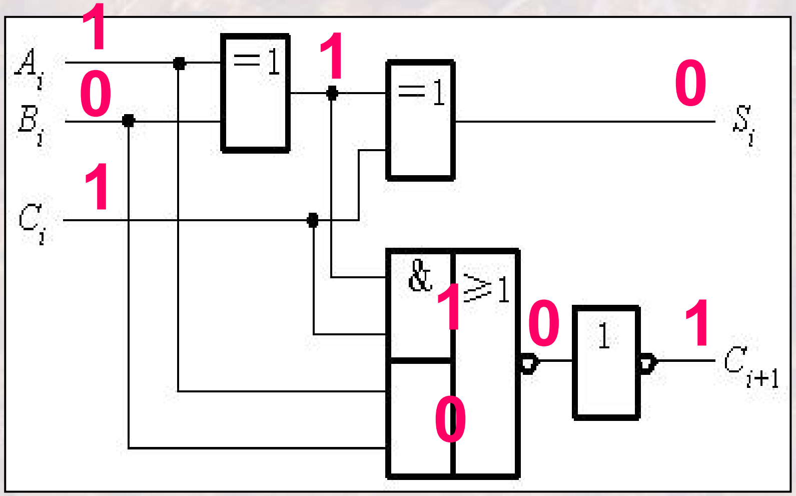


2.1.5 0和1与电子元器件

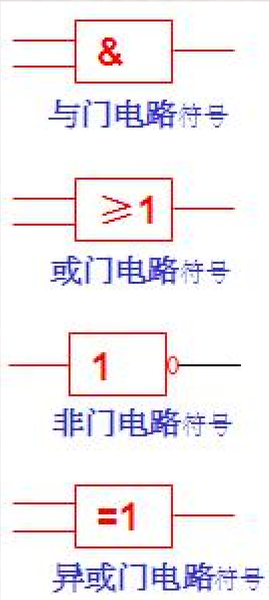
(5) 如何用已实现的基本逻辑运算(门电路)来实现更复杂的运算？

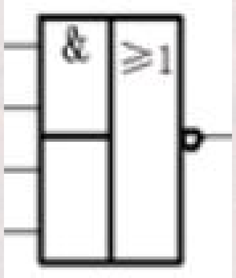
基于门电路的复杂组合逻辑电路

◆可验证一位加法器实现的正确性。



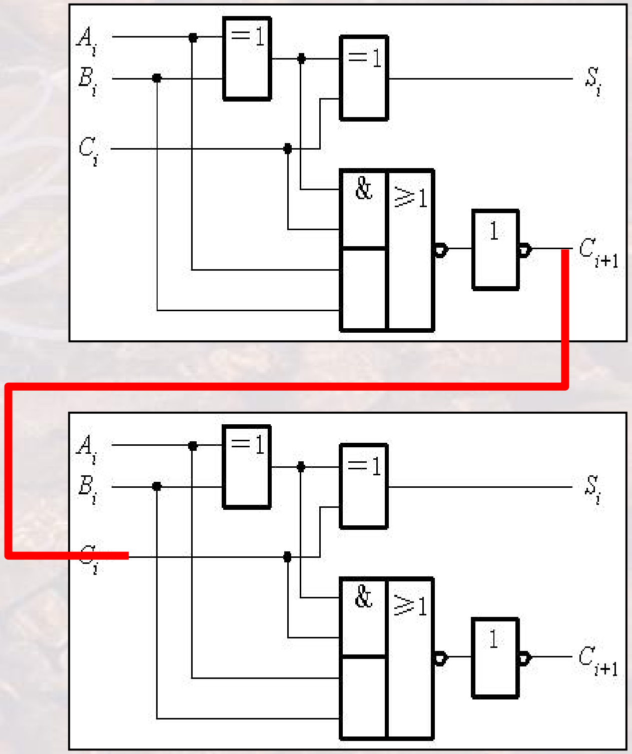
$$
\begin{array}{rcc}
 & & A_i \\
 & & B_i \\
+ & & C_i \\
\hline
 & C_{i+1} & S_i
\end{array}
$$





基于门电路的复杂组合逻辑电路

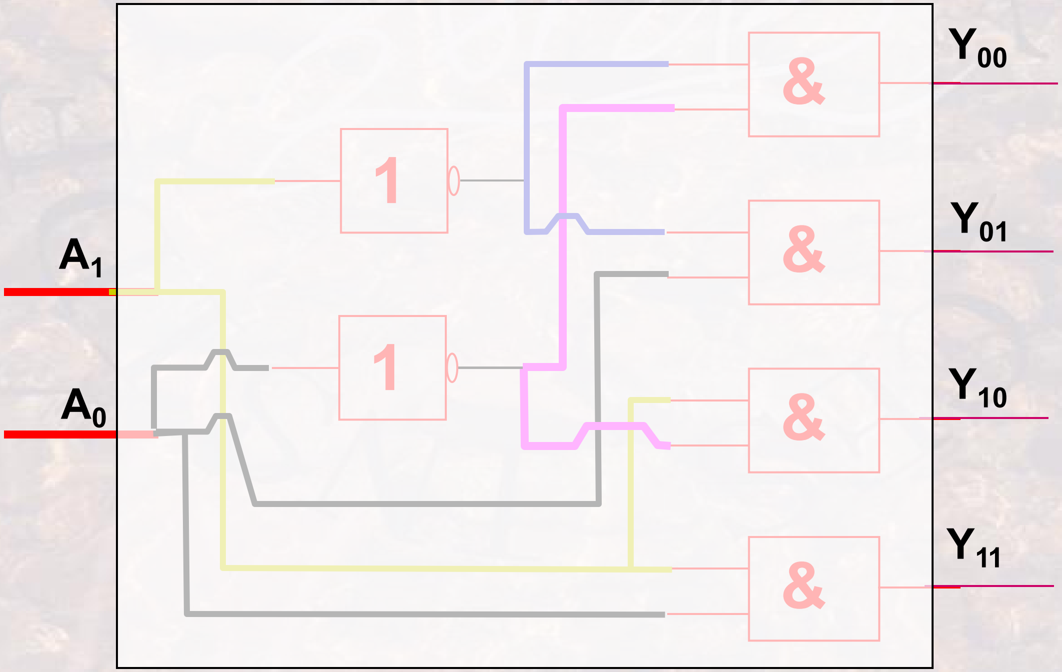
- ◆示例：多位加法器的实现
- ◆用已验证正确的一位加法器，来实现更为复杂的多位加法器
- ◆用已验证正确的多位加法器，来实现更为复杂的乘法器/除法器等(略)
- ◆分层构造：低层电路已验证正确，可被封装起来；用已封装的已验证的低层电路可构造更为复杂的高层电路；如此一层层构造。



2.1.5 0和1与电子元器件
(5) 如何用已实现的基本逻辑运算(门电路)来实现更复杂的运算?

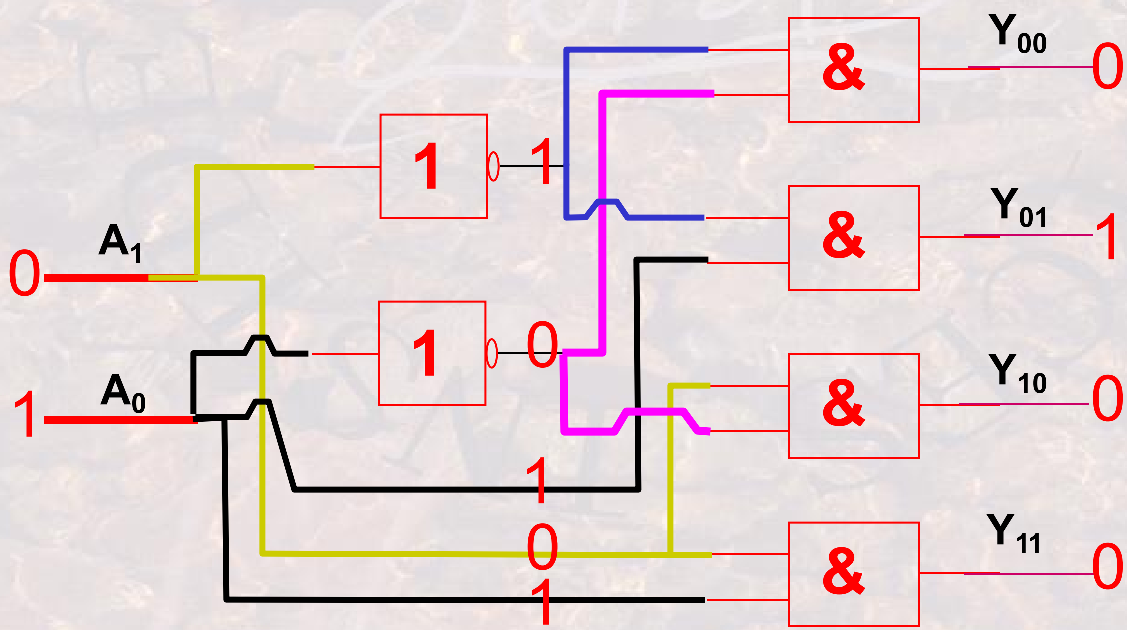
基于门电路的复杂组合逻辑电路

- 另一个示例：2-4译码器及其电路实现。



基于门电路的复杂组合逻辑电路

- 另一个示例：2-4译码器及其电路实现
- 可依据门电路的特性，输入2位二进制数01, 检查是否是第2条线(Y01)为高电平,有效?

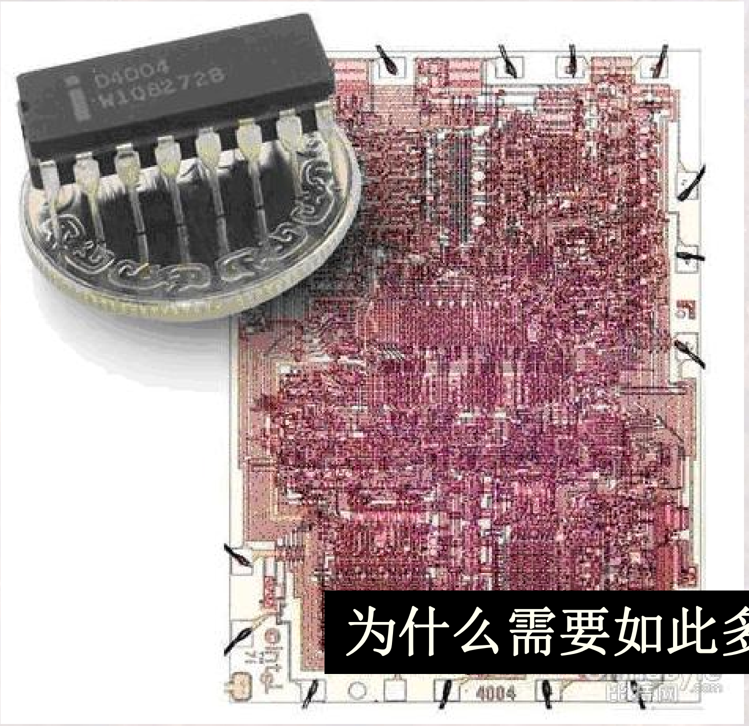


2.1.5 0和1与电子元器件
(5) 如何用已实现的基本逻辑运算(门电路)来实现更复杂的运算?

复杂部件的硬件实现(芯片、主板)

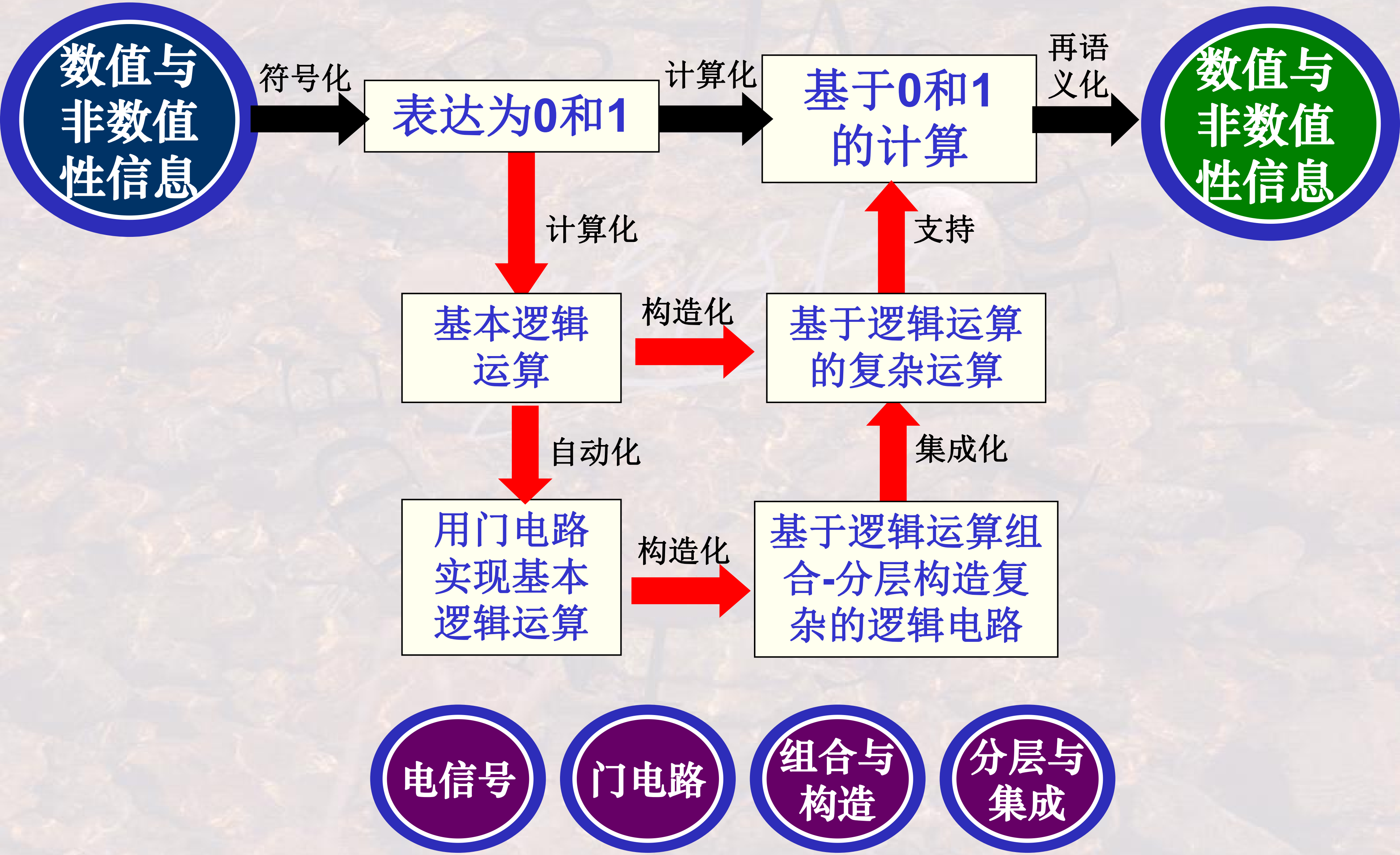
◆ 微处理器芯片即是复杂组合逻辑集成在一块板上并封装而成的电路:

从Intel4004在**12平方毫米**的芯片上集成了**2250颗**晶体管➔到Pentium 4处理器内建了**4200万颗**晶体管，以及采用**0.18微米**的电路➔再到英特尔的**45纳米**Core 2至尊/至强四核处理器上装载了**8.2亿颗**晶体管。



为什么需要如此多的晶体管呢？

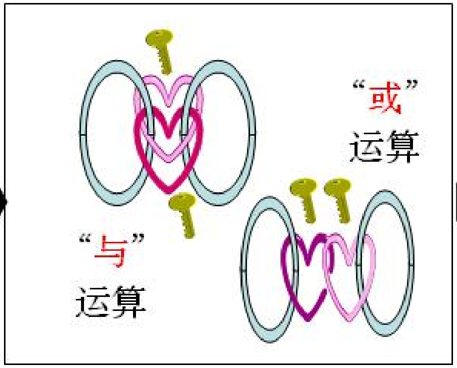
2.1.5 0和1与电子元器件
(6) 小结



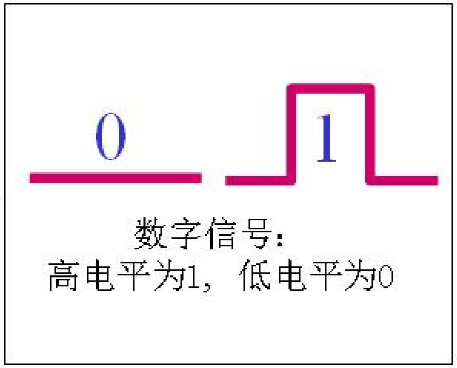
2.1.5 0和1与电子元器件

小结：电子技术实现在“符号化-计算化-自动化”思维中的位置？

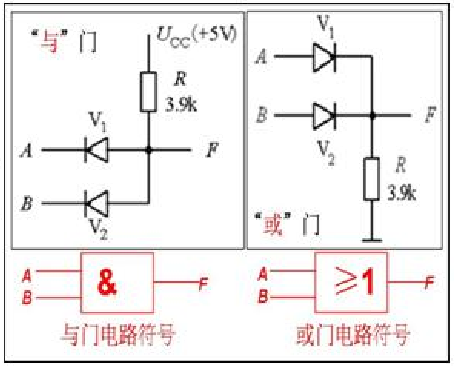
现象和思维均可表达成0和1



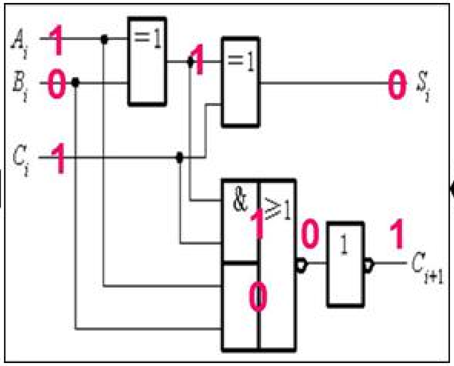
用0和1可进行算术与逻辑运算



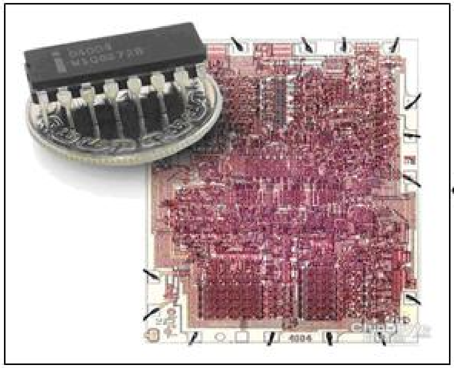
0和1可用电子技术实现



电子技术实现逻辑运算



分层构造实现复杂运算



芯片--集成的复杂组合逻辑电路

语义符号化 ➔ 符号计算化 ➔ 计算0(和)1化

➔ 0(和) 1自动化➔ 分层构造化➔ 构造集成化；