

微机原理与接口技术

控制科学与工程系
李新波



什么是微机

➤ 计算机

- 1946年
- 一定的运算能力
- 灵活的人机界面
- 可自主完成某种功能的设备

↔ 自动化

➤ 微型计算机 vs 大型计算机



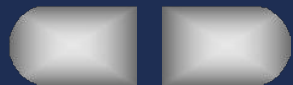
什么是微机原理与接口技术

- 内部结构
- 指令系统
- 接口与总线



开课意义

- 培养方案的要求
- 后续课程的基础
- 应试
- 竞赛与动手能力培养



课程目标

掌握:

- 微型计算机的基本工作原理
- 汇编语言程序设计方法
- 微型计算机接口技术
- 建立微型计算机系统的整体概念，形成微机系统软硬件开发的初步能力



教材及实验指导书

- 教材:

- 《微机原理与接口技术》（第4版）. 吴宁主编. 清华大学出版社

- 参考书:

- 《微型计算机原理与接口技术》（第2版）. 尹建华编. 高教出版社
- 《微机计算机原理与接口技术》. 邹逢兴主编. 清华大学出版社

- 网络资源:

- 国家级精品课《微机原理》. 浙江大学、西北工业大学



考核及考试

- 考核方式:

- 课堂测验**10%**
- 课后作业**10%**
- 实 验**10%**

- 考试方式:

- 闭卷 -- 类似计算机二级、三级



微型计算机的发展

一、史前时代(1623—1895)

1623年，德国科学家契克卡德制造了人类有史以来第一台机械计算机，这台机器能够进行六位数的加减乘除运算。



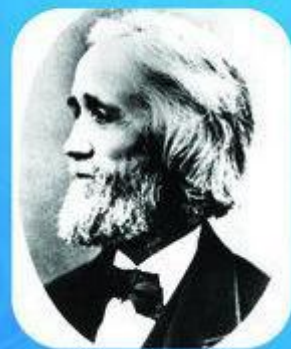
1674年，莱布尼茨对计算机进行了改进，使之成为一种能够进行连续运算的机器，并且提出了“二进制”数的概念。



1834年，巴贝奇提出了分析机的概念，机器共分为三个部分：堆栈、运算器、控制器。他的助手，阿达·奥古斯塔为分析机编制了人类历史上第一批计算机程序。



1868年，美国新闻工作者克里斯托夫·肖尔斯发明了沿用至今的QWERTY键盘。



微型计算机的发展

二、电子管时代(1911——1946)

1895年,英国青年工程师弗莱明通过“爱迪生效应”发明了人类第一只电子管。



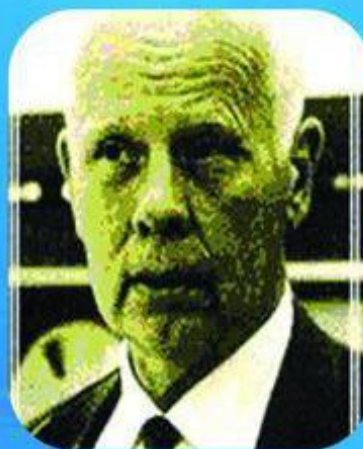
1935年,IBM制造了IBM601穿孔卡片式计算机,该计算机能够在几秒钟内计算出乘法运算。20多岁的德国工程师楚泽研制出了机械可编程计算机Z1,并采用了二进制形式,其理论基础即来源于布尔代数。



1946年2月14日,美国宾西法尼亚大学摩尔学院教授莫契利和埃克特共同研制成功了ENIAC (Electronic Numerical Integrator And Computer) 计算机。



1937年11月,美国AT&T贝尔实验室研究人员斯蒂比兹制造了电磁式数字计算机“Model-K”。



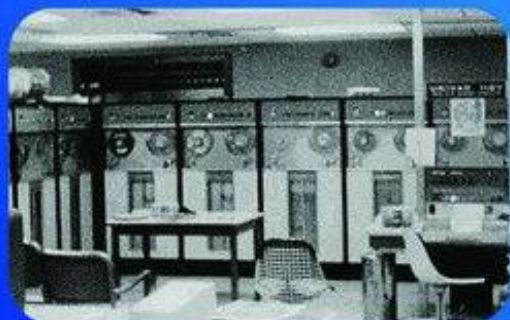
微型计算机的发展

三、晶体管时代(1947——1958)

1947年12月23号, 贝尔实验室的肖克利、布拉顿、巴丁创造出了世界上第一只半导体放大器件, 他们将这种器件重新命名为“晶体管”。



1951年6月14日, 莫契利和埃克特联袂制造的UNIVAC计算机正式移交美国人口普查局使用, 从而使电脑走出了实验室, 开始为人类社会服务, 从此人类社会进入了计算机时代。



1952年1月, 由计算机之父, 冯·诺伊曼 (Von Neumann) 设计的IAS电子计算机EDVAC问世。这台IAS计算机总共采用了2300个电子管, 运算速度却比拥有18000个电子管的“埃尼阿克”提高了10倍。



同年贝尔实验室使用800只晶体管组装了世界上第一台晶体管计算机TRADIC。



微型计算机的发展

四、微处理器时代 (1971—2003)

1974年4月1日，Intel推出了自己的第一款8位微处理芯片8080。12月，电脑爱好者爱德华·罗伯茨发布了自己制作的装配有8080处理器的计算机“牛郎星”，这也是世界上第一台装配有微处理器的计算机。



1975年，克雷完成了自己的第一个超级计算机“克雷一号” (CARY-1)，实现了每秒一亿次的运算速度。该机占地不到7平方米，重量不超过5吨，共安装了约35万块集成电路，同时这也标志着巨型机跨进了第三代电脑的行列。



1977年2月，Intel发布80286处理器。时钟频率提高到20MHz，并增加了保护模式，



可访问16M内存。支持1GB以上的虚拟内存。每秒执行270万条指令，集成了134000个晶体管。



1983年5月8日，IBM推出了IBM PC的改进型号IBM PC/XT，并为其内置了硬盘。

微型计算机的发展

五、智能化时代（2003-至今）

智能计算机已经成为一个动态的发展的概念，它始终处于不断向前推进的计算机技术的前沿。

智能计算机



平板电脑



云计算（cloud computing）是基于互联网的相关服务的增加、使用和交付模式，通常涉及通过互联网来提供动态易扩展且经常是虚拟化的资源。它意味着计算能力也可作为一种商品通过互联网进行流通。云计算是继1980年代大型计算机到客户端-服务器的大转变之后的又一种巨变。



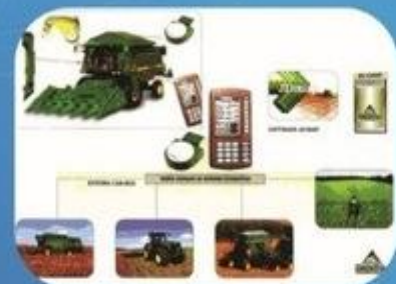
物联网是新一代信息技术的重要组成部分。其英文名称是“The Internet of things”，物联网就是物物相连的互联网。按约定的协议，把任何物品与互联网相连接，进行信息交换和通信，以实现对物品的智能化识别、定位、跟踪、监控和管理的一种网络。



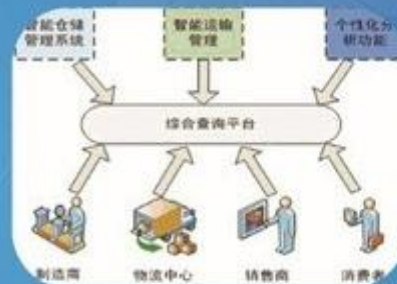
智能交通



智能楼宇

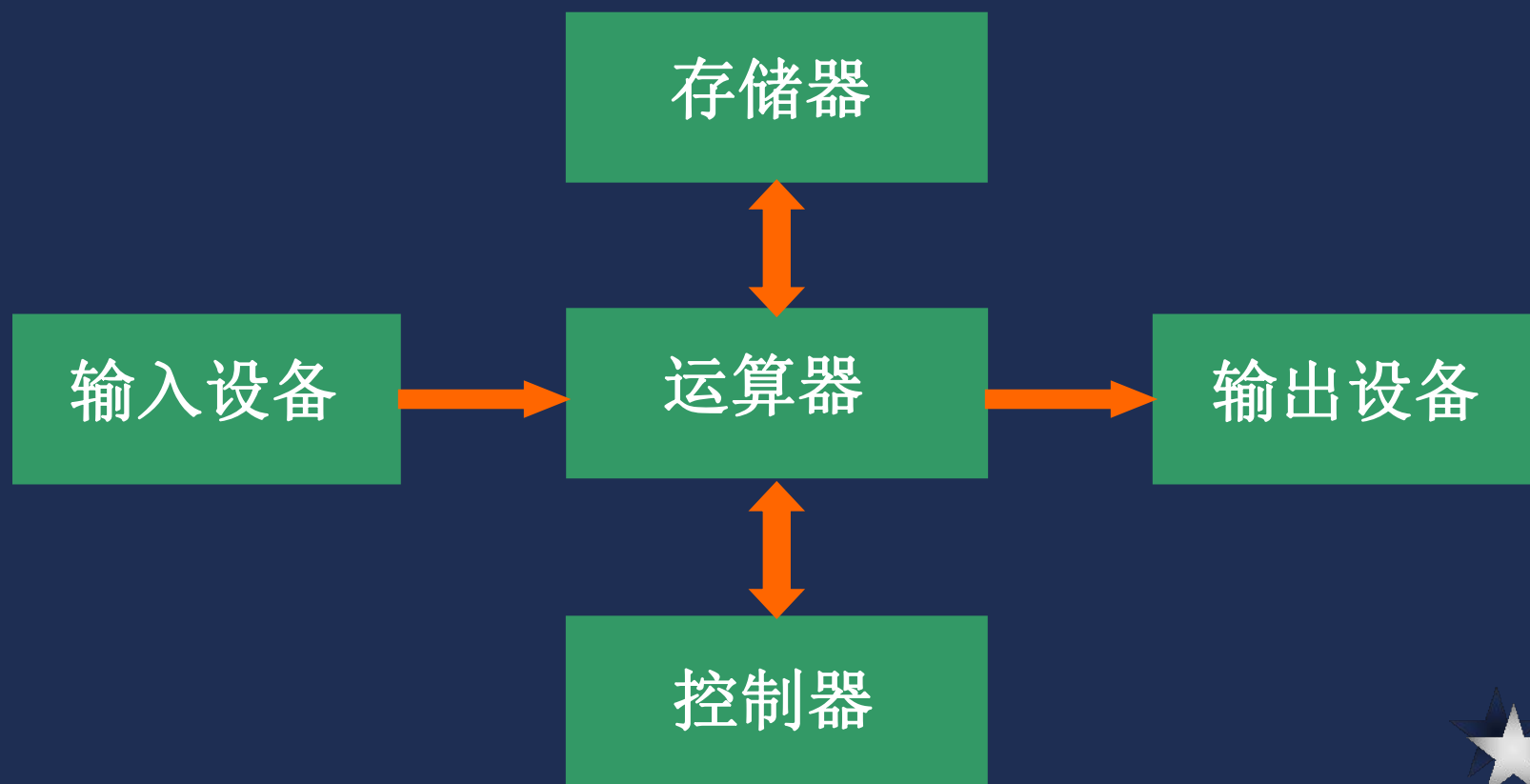


智能农业

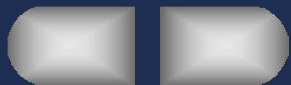


智能物流

冯·诺依曼计算机体系结构

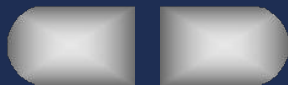


二进制思想、程序存储思想



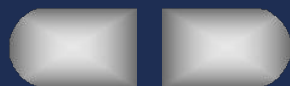
冯 • 诺依曼机的特点:

- 将计算过程描述为由许多条指令按一定顺序组成的程序，以**二进制形式**放入**存储器**保存
- 由**控制器**控制整个程序 and 数据的存取以及程序的执行；**根据指令流驱动原理**，指令按其在存储器中存放的顺序执行；
- 以**运算器**为核心，所有的执行都经过运算器。

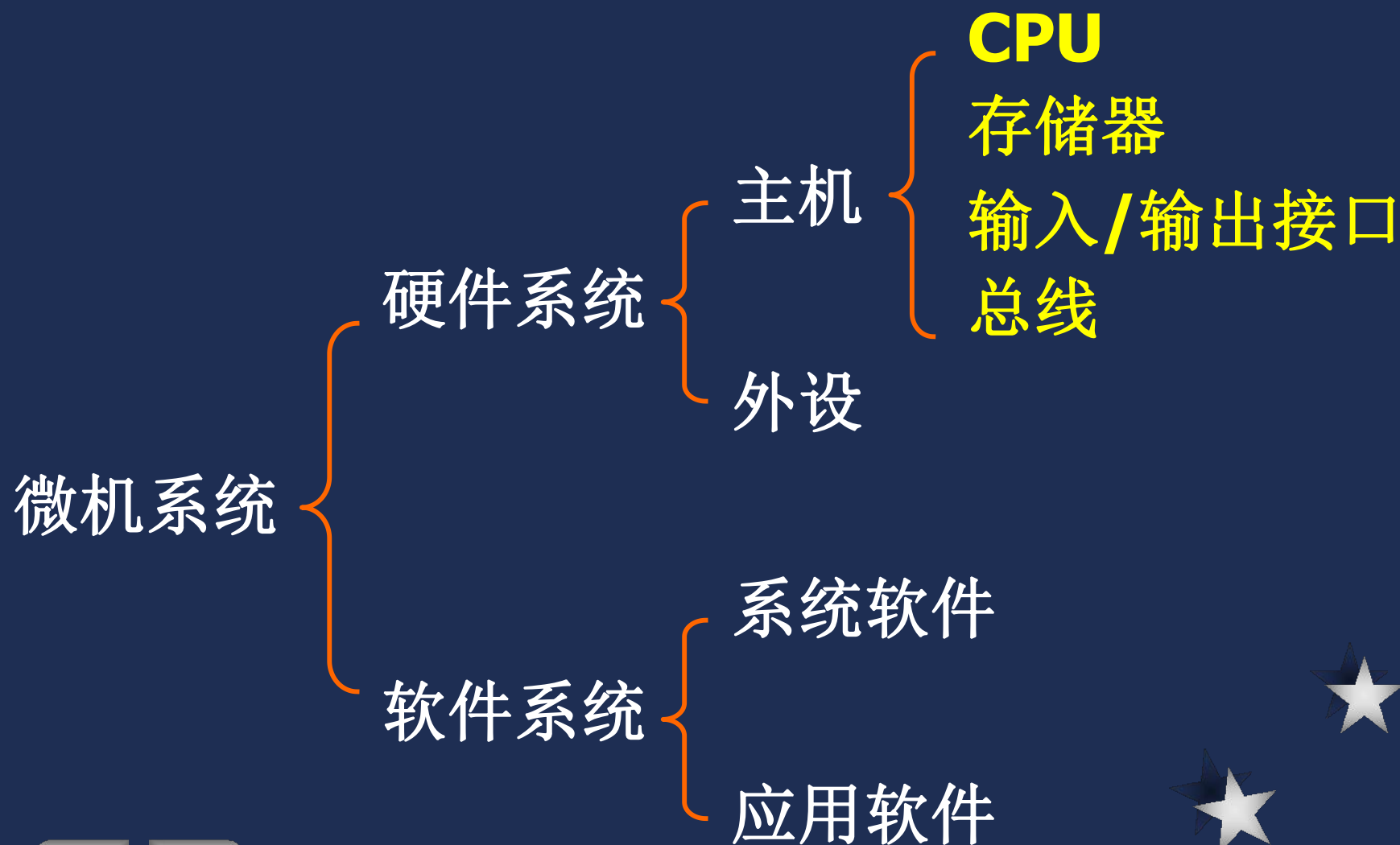


主要介绍80x86 80x88系列

- 1 兼容性
- 2 多代共存
- 3 基本结构相同



系统组成

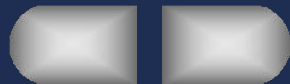
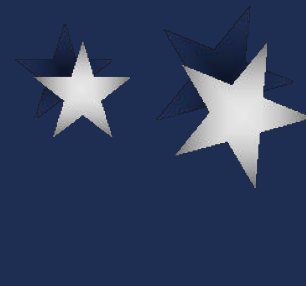


微处理器

- 微处理器简称**CPU** (Central Processor Unit)，是计算机的核心。主要包括：

{ 运算器 ALU (Arithmetic and Logic Unit)
控制器 CU
寄存器组 - 内部RAM

解释指令、执行指令、与外界交换数据



存储器

- 用于存放计算机工作过程中需要操作的数据和程序。

按位置可
分为

内部存储器

速度快

外部存储器

信息量大

硬盘

联机存储
脱机存储

按工作方
式可分为

随机存取存储器 (RAM)

随机读写, 断电丢失

只读存储器 (ROM)

只读不写, 断电保留

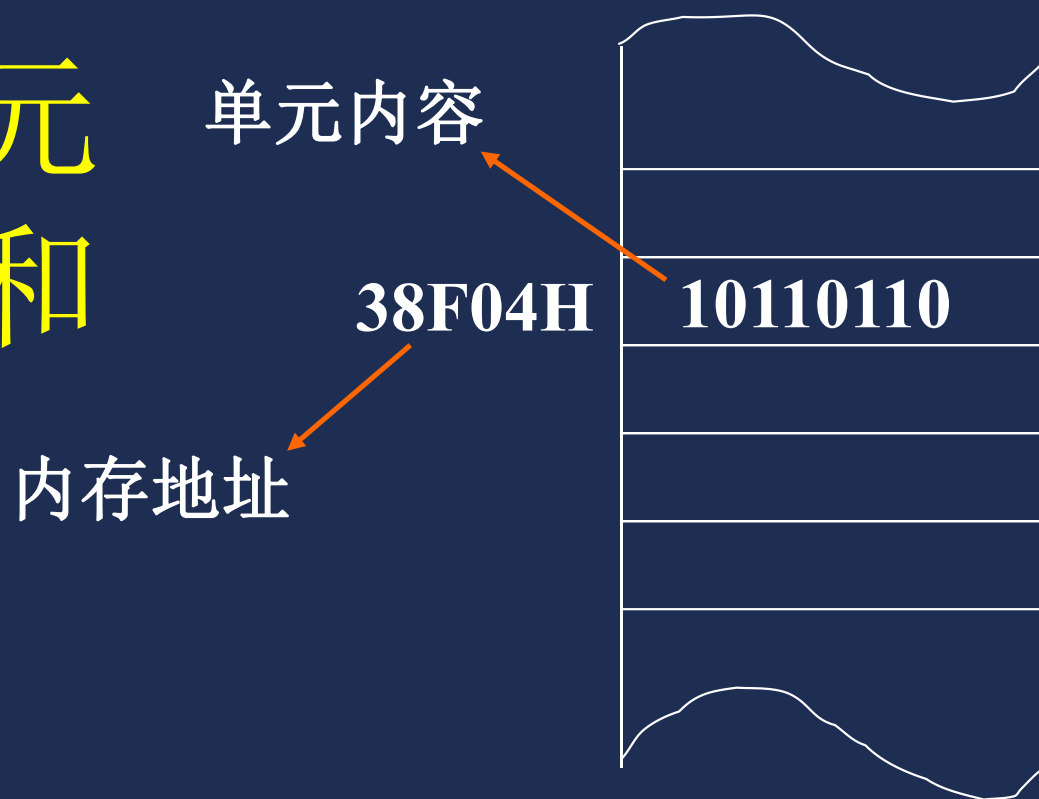


内存单元

- 内存有许多单元组成，每个单元可存放一个**8位**的二进制数，即一个**字节**（Byte, B）的二进制信息。

- 每个单元都对应一个地址，以实现单元内容的寻址。

内存单元 的地址和 内容



内存单元



内存容量

- 内存所含存储单元的个数，反应存储信息量的大小
- 以字节为单位

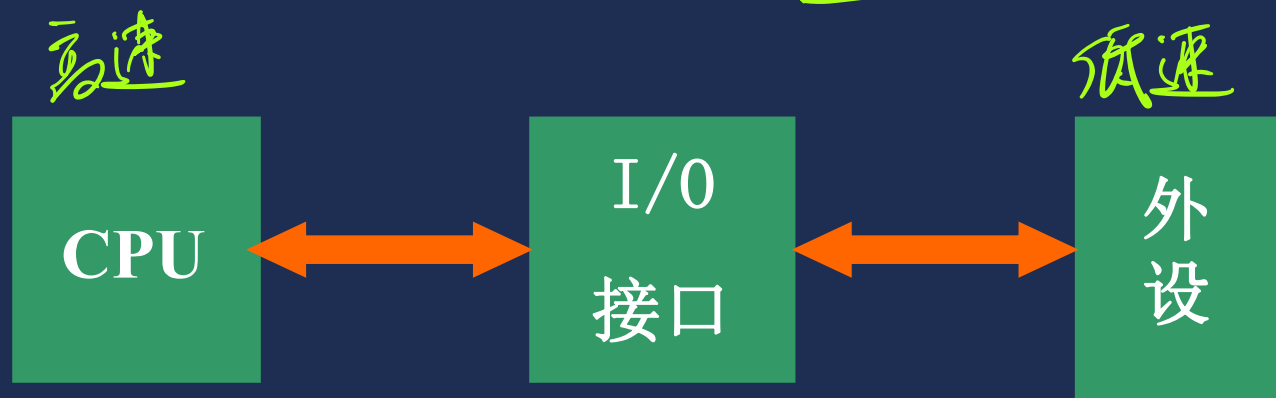
$$1\text{KB} = 2^{10}\text{B} = 1024\text{B}$$

$$1\text{MB} = 1024\text{KB} = 2^{20}\text{B}$$



输入/输出接口

- 接口是CPU与外部设备间的桥梁



- 数据缓冲寄存;
- 信号电平或类型的转换;
- 实现主机与外设间的运行匹配。

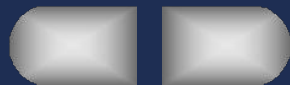


接口的分类

{ 串行接口
并行接口

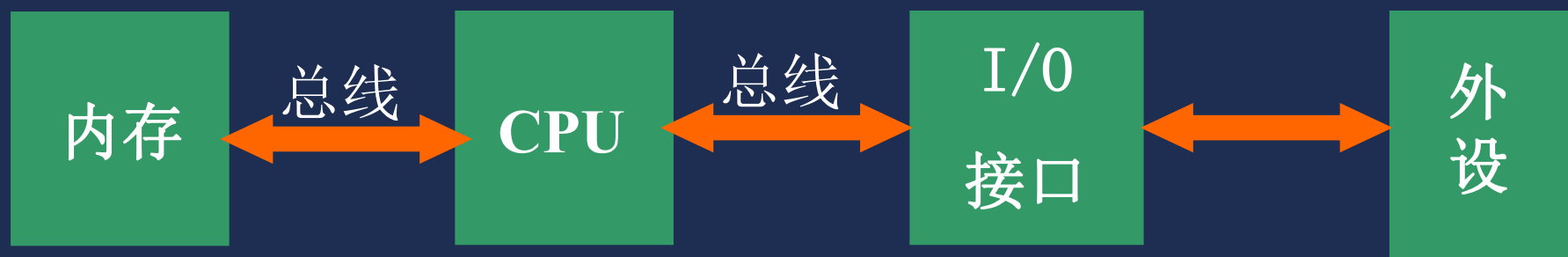
{ 输入接口
输出接口

{ 数字接口
模拟接口



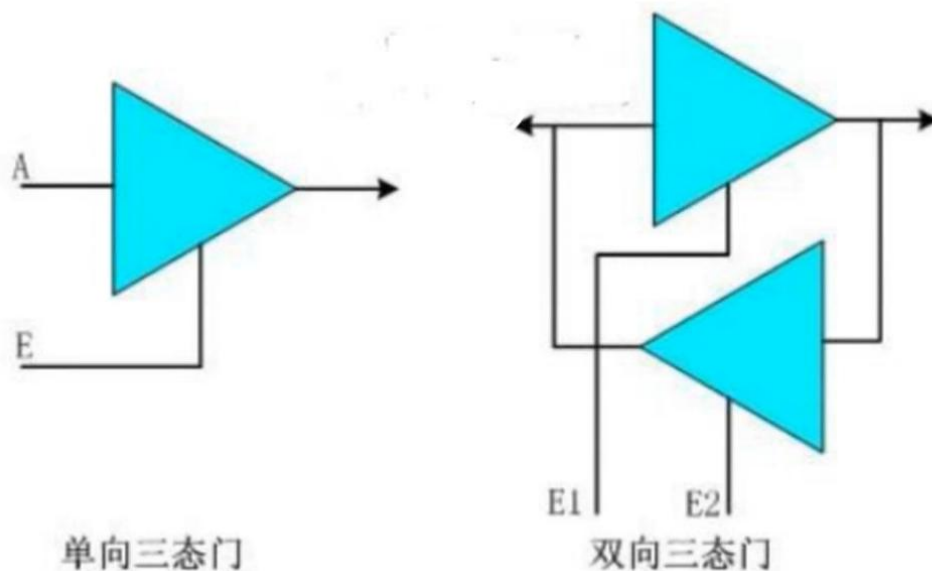
硬件系统的连接结构--总线

- 分类—^{AB}地址总线、^{DB}数据总线、^{CB}控制总线
- 以CPU为核心，其他部件通过三态门全部“挂接”在系统总线上



三态门

三态门是一种重要的总线接口电路。三态门都有一个EO控制使能端，来控制门电路的通断。可以具备这三种状态的器件就叫做三态器件。当EO有效时，三态电路呈现正常的“0”或“1”的输出；当EO无效时，三态电路给出高阻态输出。



总线



- **1. 地址总线AB**
 - 单向输出、三态控制；输出地址信号
- **2. 数据总线DB**
 - 双向、三态控制；传输数据
- **3. 控制总线CB**
 - 三态控制，每根线有固定作用、传输方向单一；传送控制命令、时序信息和状态信息



软件系统

- 软件：为运行、管理和维护计算机系统或为实现某一功能而编写的各种程序的总和及其相关资料。



二、计算机中数的表示方式

- 1. 无论数值还是数的符号，都只能用0和1表示
- 2. 通常（有符号数）用一个数的最高位作为符号位，0表示正数，1表示负数
- 3. 机器数：计算机中使用的、连同符号位也编码化的二进制数
- 4. 机器数的真值：机器数所代表的数值大小

二进制数 { 无符号数
有符号数



1. 无符号数的表示方法

十进制 Decimal, D

二进制 Binary, B

十六进制 Hexadecimal, H

BCD码

ASCII码

- 234.98D或 (234.98)_D

- 1101.11B或 (1101.11)_B

- ABCD.BFH或 (ABCD.BF)_H

十进制

特点:

- 以十为底，逢十进一；有0-9十个数字符号。
用D表示。

权值表达式:

$$\begin{aligned} D &= D_{n-1} \times 10^{n-1} + D_{n-2} \times 10^{n-2} + \cdots + D_0 \times 10^0 + D_{-1} \times 10^{-1} + \cdots + D_{-m} \times 10^{-m} \\ &= \sum_{i=-m}^{n-1} D_i \times 10^i \end{aligned}$$



二进制

特点:

- 以2为底，逢2进位；只有0和1两个符号。用**B**表示。

权值表达式:

$$\begin{aligned}(B)_2 &= B_{n-1} \times 2^{n-1} + B_{n-2} \times 2^{n-2} + \cdots + B_0 \times 2^0 + B_{-1} \times 2^{-1} + \cdots + B_{-m} \times 2^{-m} \\ &= \sum_{i=-m}^{n-1} B_i \times 2^i\end{aligned}$$



十六进制

特点:

- 有0--9及A--F共16个数字符号，逢16进位。用H表示。

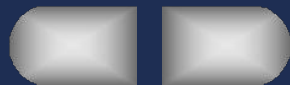
权值表达式:

$$\begin{aligned}(H)_{16} &= H_{n-1} \times 16^{n-1} + H_{n-2} \times 16^{n-2} + \cdots + H_0 \times 16^0 + H_{-1} \times 16^{-1} + \cdots + H_{-m} \times 16^{-m} \\ &= \sum_{i=-m}^{n-1} H_i \times 16^i\end{aligned}$$



2. 各种进制数间的转换

- 非十进制数到十进制数的转换
- 十进制到非十进制数的转换
- 二进制与十六进制数之间的转换



非十进制数到十进制数的转换

- 按相应的权值表达式展开

- 例:

- $1011.11B = 1 \times 2^3 + 0 \times 2^2 + 1 \times 2^1 + 1 \times 2^0 + 1 \times 2^{-1} + 1 \times 2^{-2}$
 $= 8 + 2 + 1 + 0.5 + 0.25$
 $= 11.75$

- $5B.8H = 5 \times 16^1 + 11 \times 16^0 + 8 \times 16^{-1}$
 $= 80 + 11 + 0.5$
 $= 91.5$



十进制到非十进制数的转换

- 到二进制的转换：

对整数：除2取余；

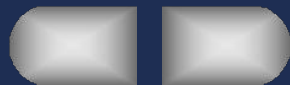
对小数：乘2取整。

- 到十六进制的转换：

对整数：除16取余；

对小数：乘16取整。

317.125D=100111101.001B



二进制与十六进制间的转换

- 用4位二进制数表示1位十六进制数

- 例:

- 25.5

- $= 1\ 1001.1\text{B} = 19.8\text{H}$

- 1100 1010. 0110 101B

- $= \text{CA.6AH}$



3. 计算机中的编码

- ASCII码

- 西文字符编码

- 字符的编码，一般用7位或8位二进制码表示。

- 熟悉0---F的ASCII码

有效位7位
最高位默认为0

- BCD码

- 用二进制编码表示的十进制数

- 压缩BCD码

- 用4位二进制码表示一位十进制数

- 扩展BCD码

- 用8位二进制码表示一位十进制数



Hex	Char	Hex	Char	Hex	Char	Hex	Char	Hex	Char	Hex	Char	Hex	Char	Hex	Char
00		20		40	@	60	'	80	Ç	A0	á	C0	Ł	E0	α
01		21	!	41	A	61	a	81	ü	A1	â	C1	ł	E1	β
02		22	..	42	B	62	b	82	ë	A2	í	C2	Ł	E2	Γ
03		23	#	43	C	63	c	83	ä	A3	ó	C3	ł	E3	Π
04		24	\$	44	D	64	d	84	ä	A4	ú	C4	ł	E4	Σ
05		25	%	45	E	65	e	85	ä	A5	ñ	C5	ł	E5	σ
06		26	&	46	F	66	f	86	ä	A6	Ñ	C6	ł	E6	μ
07		27	,	47	G	67	g	87	ç	A7	o	C7	ł	E7	Υ
08		28	(48	H	68	h	88	è	A8	ô	C8	ł	E8	ϕ
09		29)	49	I	69	i	89	ë	A9	ç	C9	ł	E9	Θ
0A		2A	*	4A	J	6A	j	8A	è	AA	ı	CA	ł	EA	Ϸ
0B		2B	+	4B	K	6B	k	8B	ï	AB	½	CB	ł	EB	δ
0C		2C	,	4C	L	6C	l	8C	î	AC	¼	CC	ł	EC	ø
0D		2D	-	4D	M	6D	m	8D	ì	AD	ı	CD	ł	ED	Φ
0E		2E	.	4E	N	6E	n	8E	Ä	AE	«	CE	ł	EE	€
0F		2F	/	4F	O	6F	o	8F	Ä	AF	»	CF	ł	EF	∩
10		30	0	50	P	70	p	90	æ	B0	☼	D0	ł	F0	≡
11		31	1	51	Q	71	q	91	Æ	B1	▬	D1	ł	F1	≡
12		32	2	52	R	72	r	92	ø	B2	■	D2	ł	F2	∠
13		33	3	53	S	73	s	93	ø	B3	—	D3	ł	F3	∠
14		34	4	54	T	74	t	94	ö	B4	ł	D4	ł	F4	┐
15		35	5	55	U	75	u	95	ö	B5	ł	D5	ł	F5	┐
16		36	6	56	V	76	v	96	ü	B6	ł	D6	ł	F6	÷
17		37	7	57	W	77	w	97	ü	B7	ł	D7	ł	F7	»
18		38	8	58	X	78	x	98	ü	B8	ł	D8	ł	F8	•
19		39	9	59	Y	79	y	99	ö	B9	ł	D9	ł	F9	•
1A		3A	:	5A	Z	7A	z	9A	Ü	BA	ł	DA	ł	FA	.
1B		3B	:	5B	[7B	{	9B	€	BB	ł	DB	▬	FB	Ł
1C		3C	<	5C	\	7C		9C	£	BC	ł	DC	▬	FC	=
1D		3D	=	5D]	7D	}	9D	¥	BD	ł	DD	▬	FD	z
1E	▲	3E	>	5E	^	7E	~	9E	ƒ	BE	ł	DE	▬	FE	■
1F	▼	3F	?	5F	_	7F	ˆ	9F		BF	ł	DF	▬	FF	

BCD码与二进制数之间的转换

- 先转换为十进制数，再转换二进制数；反之同样。
- 例： $(0001\ 0001\ .0010\ 0101)_{\text{BCD}}$
= 11 .25 D
= $(1011\ .01)_{\text{B}}$



三、无符号数的运算



- 无符号二进制数的算术运算
- 无符号数的表达范围
- 运算中的溢出问题
- 无符号数的逻辑运算

1. 无符号数的算术运算

- 加法运算 ($1+1=0$ (有进位))
- 减法运算 ($0-1=1$ (有借位))
- 乘法运算 (注意乘数为2时的规律)
- 除法运算 (注意除数为2时的规律)

$$00001011 \times 0100 = 00101100B$$

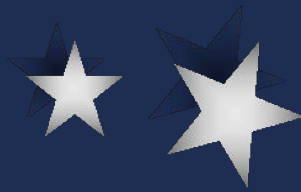
- 除法运算 (注意除数为2时的规律)

$$00001011 : 0100 = 00000010B$$

—— 余数 11B

每乘以2, 相对于被乘数向左移动1位

每除以2, 相对于被除数向右移动1位



4 2 1
2 0 0

乘除运算例

- 00001011×0100
 $= 00101100B$

- $00001011 \div 0100 = 00000010B$

即：商 = $00000010B$

余数 = $11B$

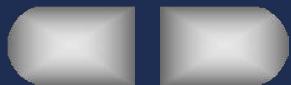


2. 无符号数的表示范围:


$$0 \leq X \leq 2^n - 1$$

若运算结果超出这个范围，则产生溢出。

对无符号数：运算时，当最高位向更高位
有进位（或借位）时则产生
溢出。



[例]:

$$\begin{array}{r} 11111111 \\ + 00000001 \\ \hline 1\ 00000000 \end{array}$$


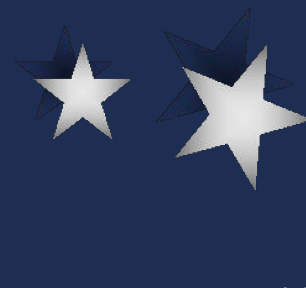
最高位向前有进位，产生溢出



3. 逻辑运算

与
或
非
异或

- 任何数和“0”相“与”，结果为0。
- 任何数和“1”相“或”，结果为1。
- “非”运算即按位求反
- 两个二进制数相“异或”：
相同则为0，相异则为1



四、有符号数的运算

计算机中符号数的表示

符号位 + 数值

机器数

符号位

“0” → 表示正

“1” → 表示负

$$+52 = +0110100 = \underline{0} \quad \underline{0110100}$$

真值

符号位

数值

$$-52 = -0110100 = \underline{1} \quad \underline{0110100}$$

真值

符号位

数值

1. 符号数的表示 --原码、反码、补码

原码

- 最高位为符号位，用“0”表示正，用“1”表示负；其余为该数的绝对值。
- 优点： 真值和其原码表示之间的对应关系简单，容易理解；
- 缺点： 计算机中用原码进行加减运算比较困难，0的表示不唯一。

数0的原码

- 8位数0的原码： $+0 = 0\ 0000000\text{ B}$
 $-0 = 1\ 0000000\text{ B}$

即：数0的原码不唯一。



反码

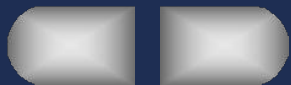
对一个机器数X:

- 若 $X > 0$, 则 $[X]_{\text{反}} = [X]_{\text{原}}$
- 若 $X < 0$, 则 $[X]_{\text{反}} =$ 对应原码的符号位不变,
数值部分按位求反

● 例: $X = -52_{\text{D}} = -0110100_{\text{B}}$

$[X]_{\text{原}} = \underline{1} 0110100_{\text{B}}$

$[X]_{\text{反}} = \underline{1} 1001011_{\text{B}}$

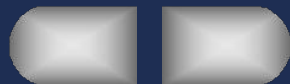


0的反码:

$$[+0]_{\text{反}} = 00000000_{\text{B}}$$

$$[-0]_{\text{反}} = 11111111_{\text{B}}$$

即: 数0的反码也不是唯一的。



补码

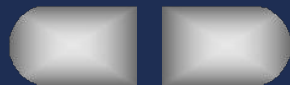
- 若 $X > 0$, 则 $[X]_{\text{补}} = [X]_{\text{反}} = [X]_{\text{原}}$
- 若 $X < 0$, 则 $[X]_{\text{补}} = [X]_{\text{反}} + 1$

● 例: $X = (-52)_D = -0110100_B$

$$[X]_{\text{原}} = 10110100_B$$

$$[X]_{\text{反}} = 11001011_B$$

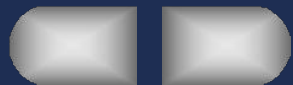
$$[X]_{\text{补}} = [X]_{\text{反}} + 1 = 11001100_B$$





0的补码:

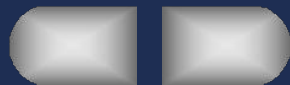
- $[+0]_{\text{补}} = [+0]_{\text{原}} = 00000000_{\text{B}}$
- $[-0]_{\text{补}} = [-0]_{\text{反}} + 1 = 11111111 + 1$
 $= \underline{1} 00000000_{\text{B}}$

对8位字长，进位被舍掉



特殊数10000000

- 对无符号数: $(10000000)_B = 128$
- 在原码中定义为: -0
- 在反码中定义为: -127 
- 在补码中定义为: -128 



符号数的表示范围

对8位二进制数：

- 原码： $-127 \sim +127$
- 反码： $-127 \sim +127$
- 补码： $-128 \sim +127$



2. 有符号二进制数与十进制的转换

对用补码表示的二进制数：1) 求出真值 2) 进行转换

将一个用补码表示的二进制数转换为十进制数

- $[X]_{\text{补}} = \underline{0} \ 0101110B$ 正数

所以：真值 = $0101110B$

$X = +46$

- $[X]_{\text{补}} = \underline{1} \ 1010010B$ 负数

所以：真值不等于 $-1010010B$

而是： $X = [[X]_{\text{补}}]_{\text{补}} = [11010010]_{\text{补}} = -0101110 = -46$

3. 符号数的算术运算——以补码形式

- 通过引进补码，可将减法运算转换为加法运算。

- 即： $[X+Y]_{\text{补}} = [X]_{\text{补}} + [Y]_{\text{补}}$

$$[X-Y]_{\text{补}} = [X+(-Y)]_{\text{补}} = [X]_{\text{补}} + [-Y]_{\text{补}}$$

$X=-0110100$, $Y=+1110100$, 求 $X+Y=?$

- $[X]_{\text{原}} = 10110100$

- $[X]_{\text{补}} = [X]_{\text{反}} + 1 = 11001100$

- $[Y]_{\text{补}} = [Y]_{\text{原}} = 01110100$

- 所以： $[X+Y]_{\text{补}} = [X]_{\text{补}} + [Y]_{\text{补}} = 11001100 + 01110100 = 01000000$

$X+Y = +1000000$

注：运算时符号位须对齐

4.有符号数运算中的溢出问题

- 两个同符号二进制数相加或异符号二进制数相减时，若最高位进位位 \oplus 次高位进位位=1，则结果产生溢出。

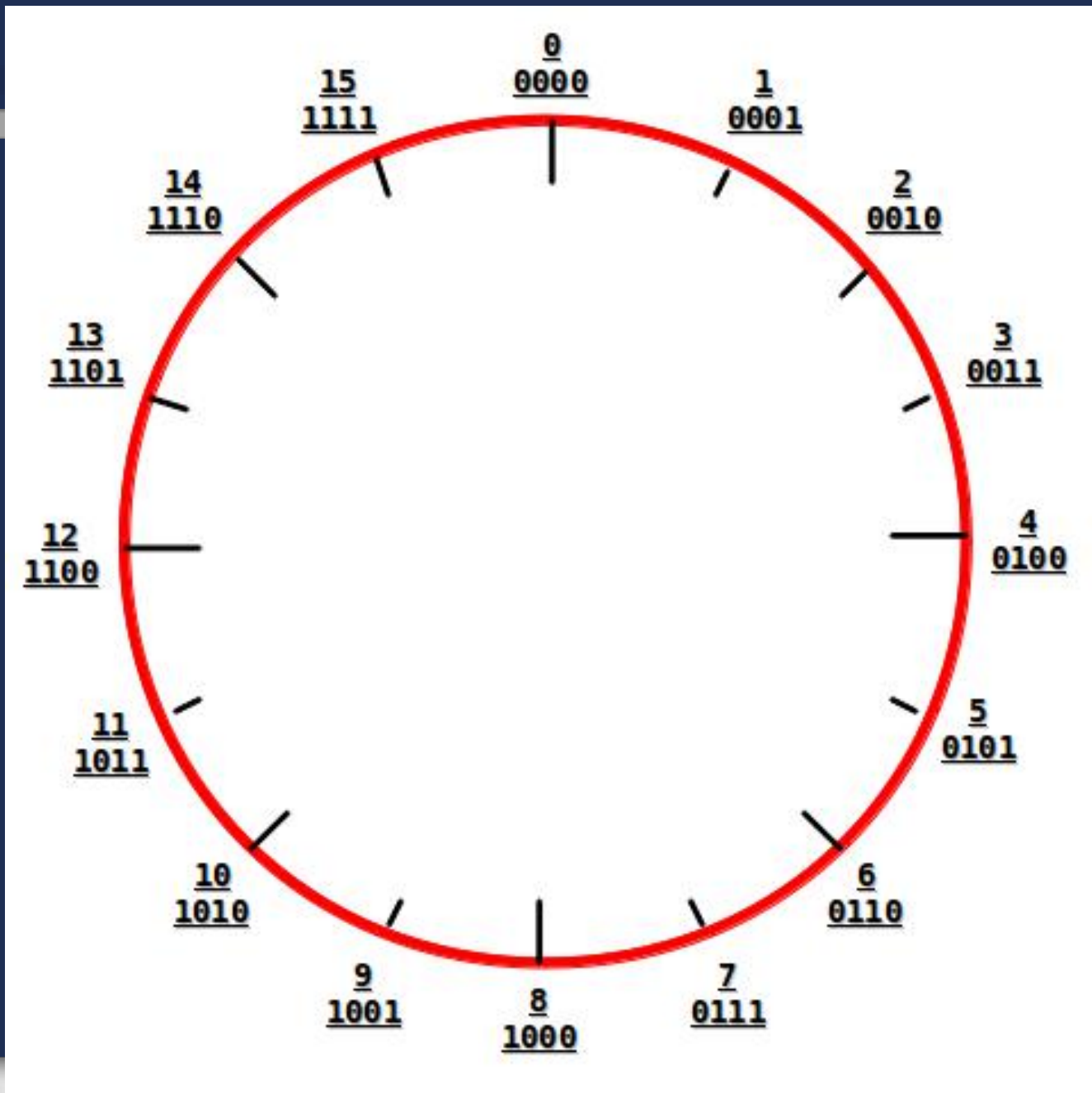
- 若：X=01111000， Y=01101001

$$\begin{array}{r} \text{则：X+Y=} \quad 01111000 \\ \quad \quad \quad + \quad 01101001 \\ \hline \quad \quad \quad 11100001 \end{array}$$

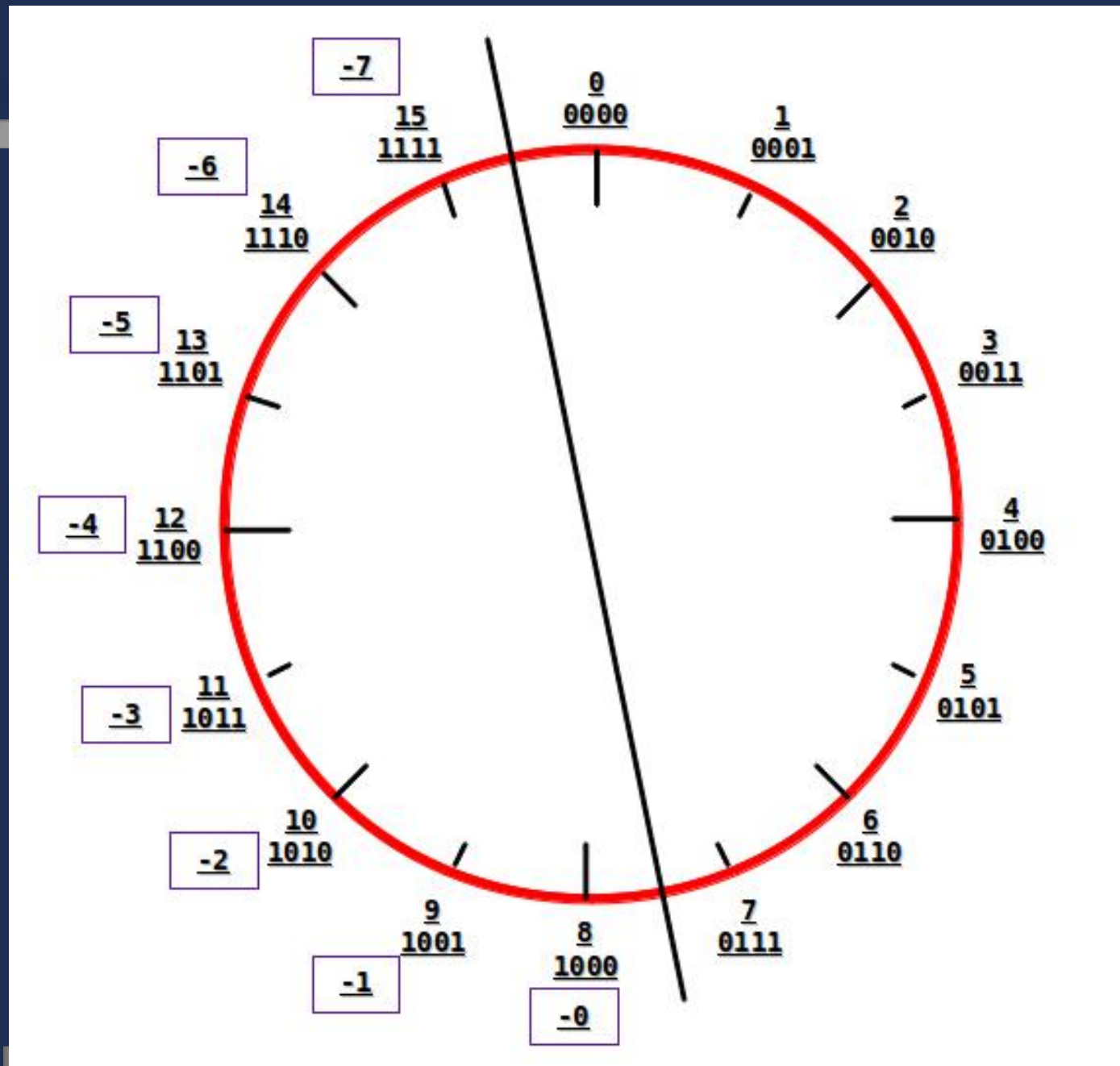
即：次高位向最高位有进位，而最高位向前无进位，产生溢出。

（事实上，两正数相加得出负数，结果出错）

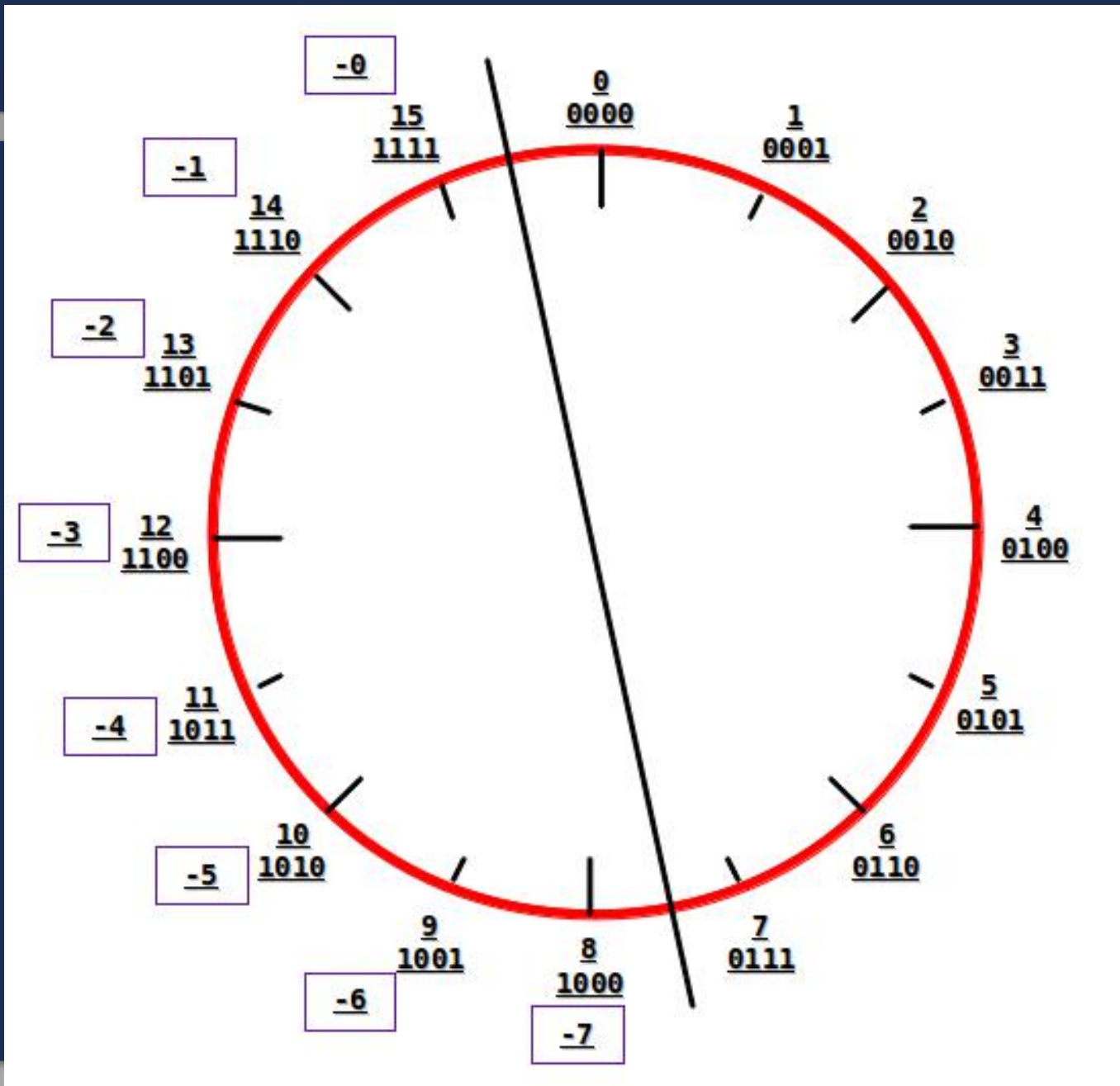
4位二进制无符号数



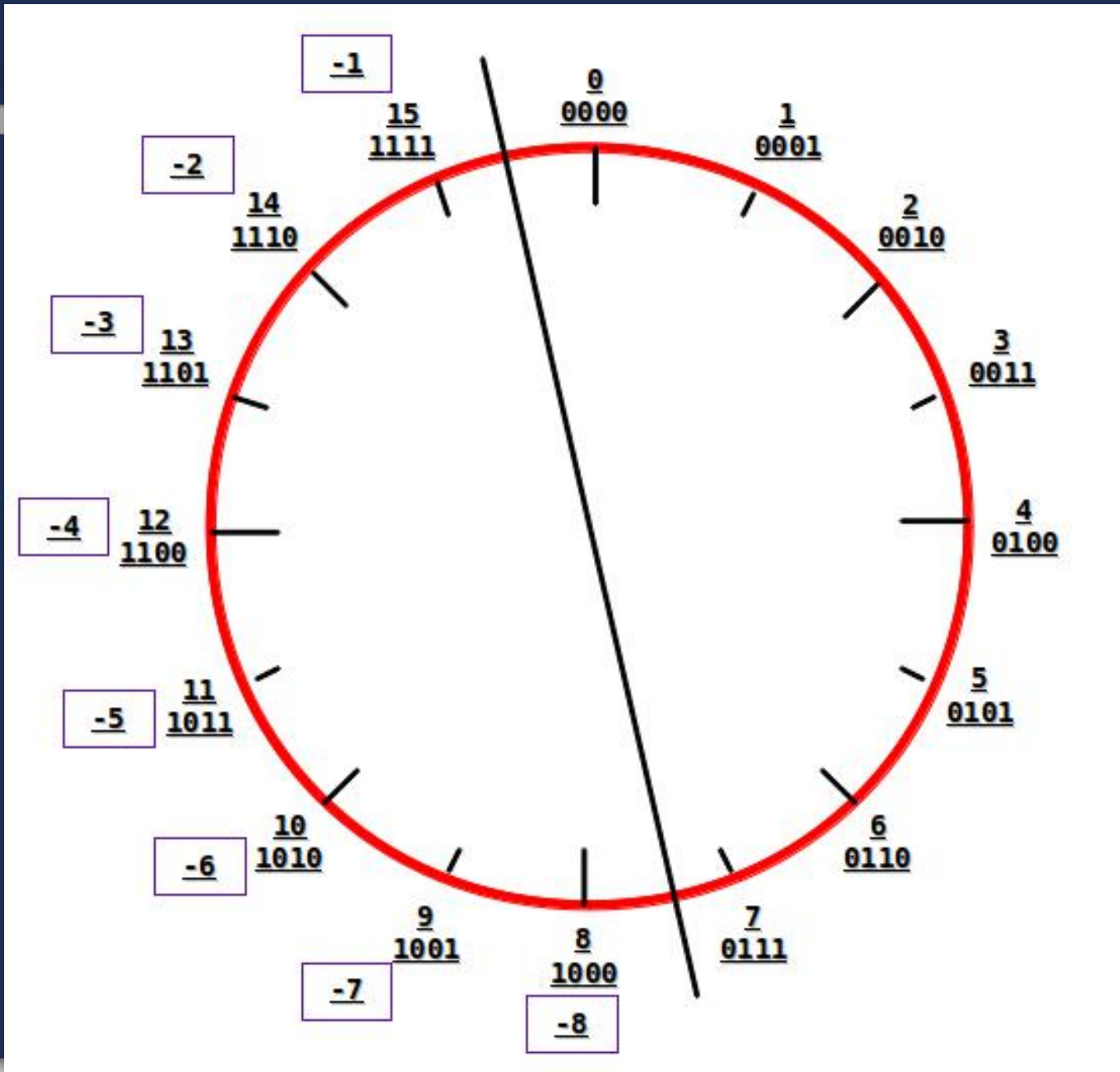
4位二进制有符号数原码表示



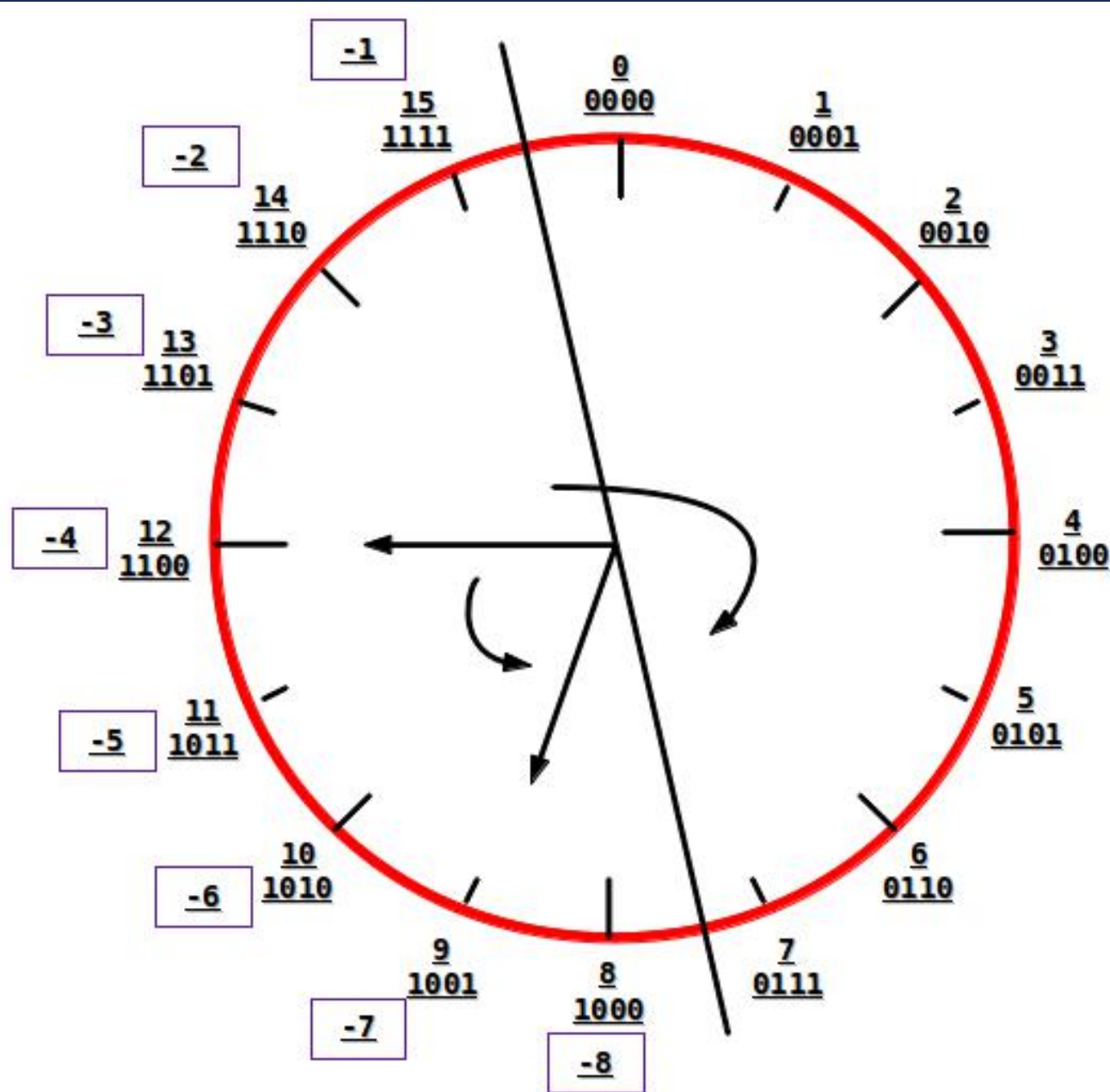
4位二进制有符号数反码表示



4位二进制有符号数补码表示



补码—“共模同余”



• 逆时针

• $((-4) - 3) / 16$

• 余数-7

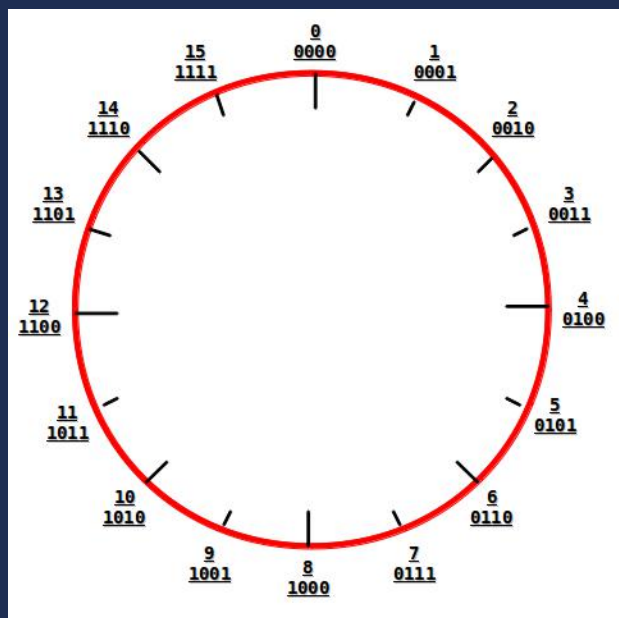
• 顺时针

• $(-4 + 13) / 16$

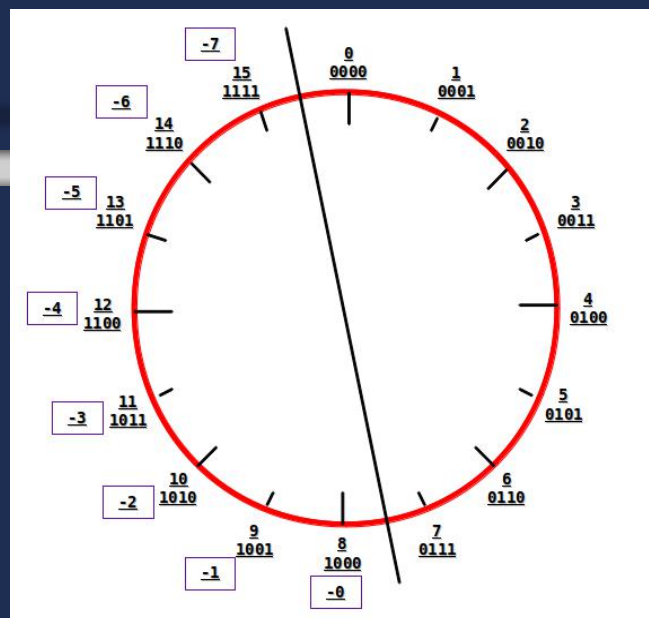
• 余数9

• 在模16情况下，-7和9是一致的

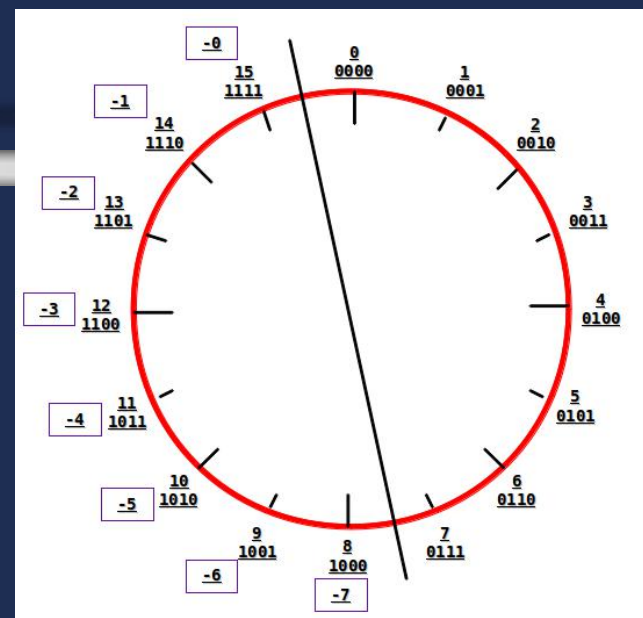
补码—“共模同余”



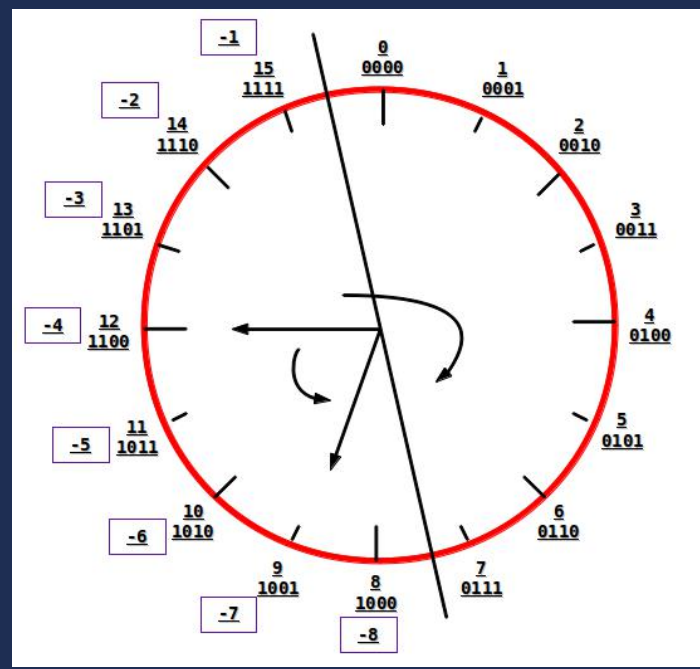
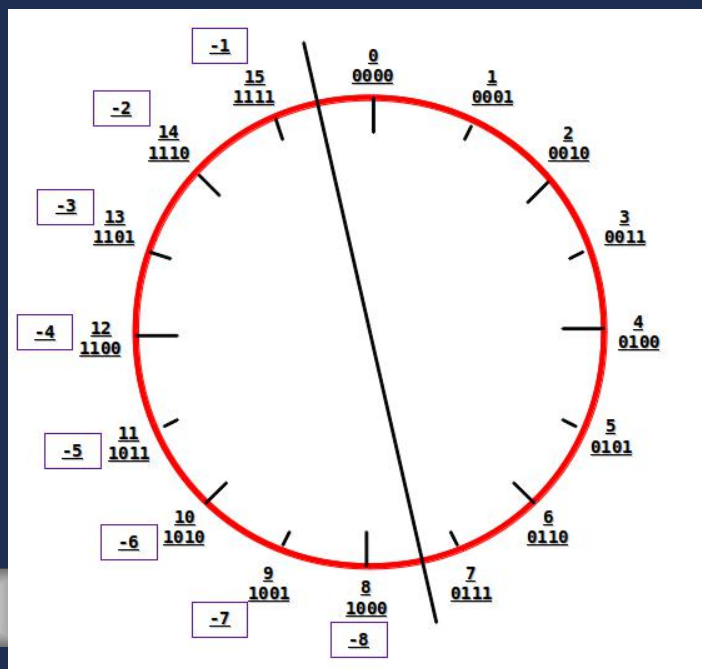
无符号



有符号原码



有符号反码



5.定点数和浮点数

- 定点数：事先约定好小数点的位置，固定不变。
- 浮点数：小数点位置是浮动的。
- 纯小数表示方式。

- $1265.125 = 10^4 \times (0.1265125)$

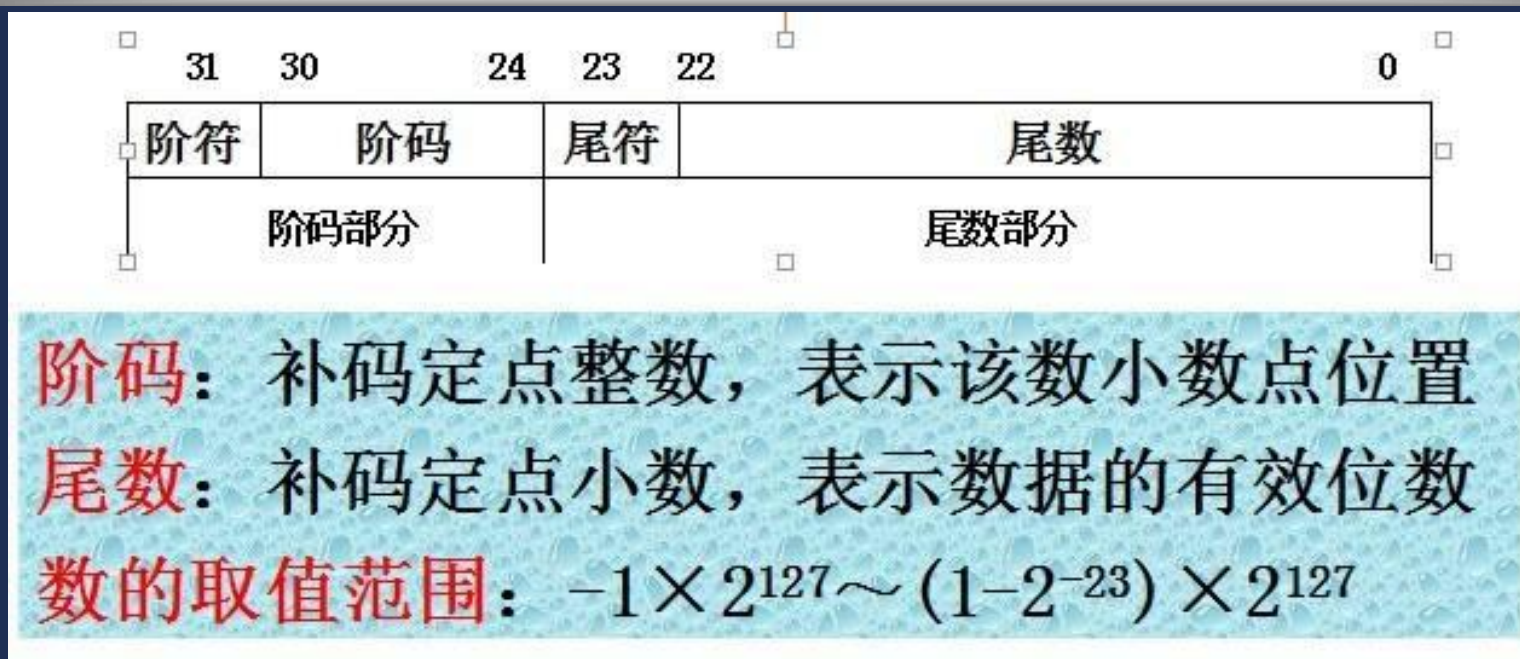
- $-0.06753 = 10^{-1} \times (-0.6753)$

$(11110110.101B)$

$= 2^8 \times (0.11110110101B)$



浮点数的一般表示方式



- $\therefore +01110110.101B = 2^7 \times (0.11110110101B)$
- \therefore 其浮点数表示为
- 0 00000111 1 111101101010000000000000

第1章 微型计算机基础概论

主要内容:

- 微机系统的组成
- 计算机中的常用计数制、编码及其相互间的转换
- 无符号二进制数的算术运算和逻辑运算
- 符号数的表示及补码运算
- 二进制数运算中的溢出
- 难点: 补码的概念及其运算

