

第一章：BCD码补码，逻辑电路与或非，地址总线（了解）

1 数和数制

◆ 十进制：逢十进一 0,1,...,9

例：(892)₁₀ 或 892D

◆ 二进制：逢二进一 0,1

例：(10010)₂ 或 10010B

◆ 八进制：逢八进一 0,1,...,7

例：(71)₈ 或 71Q

◆ 十六进制：逢十六进一 0,1,...,9,A,B,C,D,E,F

例：(3A)₁₆ 或 3AH

十进制数：D(Decimal)来表示；

二进制数：B(Binary)来表示；

八进制数：O(Octal)或Q来表示；

十六进制数：H(Hexadecimal)来表示。

1. 进位计数法与数制

2	3	4	.	1	3
10^2	10^1	10^0		10^{-1}	10^{-2}

位权是指某个固定位置上的计数单位

图1.1 十进制数的位权

■ 二进制： $(110.11)_2=1\times 2^2+1\times 2^1+0\times 2^0+1\times 2^{-1}+1\times 2^{-2}$

$$N_2=\pm \sum_{i=-m}^{n-1} x_i2^i=\pm \left(\sum_{i=0}^{n-1} x_i2^i+\sum_{i=-1}^{-m} x_i2^i\right)$$

■ 八进制： $(123.45)_8=1\times 8^2+2\times 8^1+3\times 8^0+4\times 8^{-1}+5\times 8^{-2}$

■ 十六进制： $(1B.E5)_{16}=1\times 16^1+B\times 16^0+E\times 16^{-1}+5\times 16^{-2}$

n位整数、m位小数的任意r进制数N的通式：

$$N=\pm \sum_{i=-m}^{n-1} x_ir^i=\pm \left(\sum_{i=0}^{n-1} x_ir^i+\sum_{i=-1}^{-m} x_ir^i\right)$$

十进制	二进制	八进制	十六进制
0	0000	0	0
1	0001	1	1
2	0010	2	2
3	0011	3	3
4	0100	4	4
5	0101	5	5
6	0110	6	6
7	0111	7	7
8	1000	10	8
9	1001	11	9
10	1010	12	A
11	1011	13	B
12	1100	14	C
13	1101	15	D
14	1110	16	E
15	1111	17	F

十进制转换为r进制

整数部分

- 十进制数转换为r进制数

- 整数部分: 除以 2 (8,16) 取余
- 小数部分: 乘以 2 (8,16) 取整

■ 例1.3 将十进制数25转换成二进制数。

解 $(25)_{10} = (11001)_2$

2	25	取余数	
2	12	1	←最低位
2	6	0	
2	3	0	
2	1	1	
	0	1	←最高位

■ 例1.4 将十进制数97转换成十六进制数。

解 $(97)_{10} = (61)_{16}$

16	97	取余数	
16	6	1	←最低位
	0	6	←最高位

■ 例1.5 将十进制小数0.8125转换成二进制小数。

解 $(0.8125)_{10}=(0.1101)_2$

	0.8125	积的整数部分
×	2	
—	1.6250	1 ←最高位
	0.6250	
×	2	
—	1.2500	1
	0.2500	
×	2	
—	0.5000	0
	0.5000	
×	2	
—	1.0000	1 ←最低位
	0.0000	余下的小数部分为 0,结束

2. 数制转换

二进制 \rightarrow $\left\{ \begin{array}{l} \text{八进制: 3位合并成1位} \\ \text{十六进制: 4位合并成1位} \end{array} \right.$

$\left. \begin{array}{l} \text{八进制} \\ \text{十六进制} \end{array} \right\} \rightarrow$ 二进制 $\begin{array}{l} \text{1位拆分为3位} \\ \text{1位拆分为4位} \end{array}$

■ 例1.7 将二进制数 $(11010110.11)_2$ 转换为八进制数。

解 $(11010110.11)_2 = (326.6)_8$

$$\frac{011}{3} \quad \frac{010}{2} \quad \frac{110}{6} \quad . \quad \frac{110}{6}$$

2 数据格式

- 机器码：一个数在机器（计算机）中的表示形式。
- 真值：一般书写的实际数值数据。

- 表示的数的范围受计算机字长的限制；

例：计算机字长为8位时, 无符号数的范围是
0000 0000——1111 1111 (0~255)

- 符号位被数字化 (正号：0 负号：1)

例：计算机字长为8位时, 有符号数的范围是
1 111 1111——0 111 1111 (-127~+127)

3 二进制编码和运算

原码

原码的形式为：

$$[x]_{\text{原}} = \begin{cases} 0x_{n-2}x_{n-3}\cdots x_1x_0 & x \geq 0 \\ 1x_{n-2}x_{n-3}\cdots x_1x_0 & x \leq 0 \end{cases}$$

■ 例如，当机器字长 $n=8$ 时，

$+1 = +0000001\text{B}$ ，则 $[+1]_{\text{原}} = 0\ 0000001\text{B}$

$+127 = +1111111\text{B}$ ，则 $[+127]_{\text{原}} = 0\ 1111111\text{B}$

$-1 = -0000001\text{B}$ ，则 $[-1]_{\text{原}} = 1\ 0000001\text{B}$

$-127 = -1111111\text{B}$ ，则 $[-127]_{\text{原}} = 1\ 1111111\text{B}$

对于二进制数，正数的原码就是它本身，负数的原码符号位取1，数值部分是真值的绝对值。

对于二进制数，正数的原码就是本身，负数的原码符号位取。

重点：0有两种原码表示方式，+0和-0。

$[+0]_{\text{原}} = 0000\ 0000\text{B}$

$[-0]_{\text{原}} = 1000\ 0000\text{B}$

反码

$X > 0$ 时， $[X]_{\text{反}} = [X]_{\text{原}}$

$X < 0$ 时， $[X]_{\text{反}} = [X]_{\text{原}}$ 除符号位外，各位取反。

例如：

$+1 = +0000001B$, 则 $[+1]_{\text{反}} = 0\ 0000001B$

$+127 = +1111111B$, 则 $[+127]_{\text{反}} = 0\ 1111111B$

$-1 = -0000001B$, 则 $[-1]_{\text{反}} = 1\ 1111110B$

$-127 = -1111111B$, 则 $[-127]_{\text{反}} = 1\ 0000000B$

在反码表示中, $+0$ 和 -0 的反码不同, 即 0 有两种反码表示形式:

$+0 = +0000000B$, 则 $[+0]_{\text{反}} = 0\ 0000000B$

$-0 = -0000000B$, 则 $[-0]_{\text{反}} = 1\ 1111111B$

结论: 二进制正数的反码和原码相同, 负数的反码是原码除符号位外各位取反。

补码

$X > 0$ 时, $[X]_{\text{补}} = [X]_{\text{原}}$

$X < 0$ 时, $[X]_{\text{补}} = [X]_{\text{原}}$ 除符号位外, 各位取反, 最后加1。也就是反码加1。

例如:

钟表的形式为:

$$-3 = +9 \pmod{12}$$

用补码表示时, 可以把负数转化为正数, 减法转化为加法。

在补码表示中, $+0$ 和 -0 的补码形式相同, 即 0 只有一种补码表示形式:

$+0 = +0000000B$, 则 $[+0]_{\text{补}} = 0\ 0000000B$

$-0 = -0000000B$, 则 $[-0]_{\text{补}} = 1\ 1111111 + 1 = 0\ 0000000B$

结论: 二进制正数的补码和原码相同, 负数的补码是原码除符号位外各位取反, 最后加1。

$+1$ 的补码是 $0000\ 0001B$

-1 的补码是 $1111\ 1111B$

$+127$ 的补码是 $0111\ 1111B$

-127 的补码是 $1000\ 0001B$

-128 的补码是 $1000\ 0000B$

机器字长为n位，那么最大的正数是 $2^{(n-1)} - 1$ ，最小的负数是 $-2^{(n-1)}$ 。

补码运算

机器数最高位是1的时候，表示负数，求真值的时候，其余n-1位取反加1。

比如：-1的补码是1111 1111B，求真值的时候，减1取反，得到1000 0001B，也就是-1。

二进制补码运算

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

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

例1.15 补码进行下列运算：

① $(+33) + (+15)$; ② $(-33) + (+15)$; ③ $(+33) + (-15)$; ④ $(-33) + (-15)$ 。

解： $+33 = +0100001\text{B}$, $[+33]_{\text{补}} = 0\ 0100001\text{B}$

$+15 = +0001111\text{B}$, $[+15]_{\text{补}} = 0\ 0001111\text{B}$

$-33 = -0100001\text{B}$, $[-33]_{\text{补}} = 1\ 1011111\text{B}$

$-15 = -0001111\text{B}$, $[-15]_{\text{补}} = 1\ 1110001\text{B}$

$$\begin{array}{r} 0\ 0100001\ [+33]_{\text{补}} \\ +\ 0\ 0001111\ [+15]_{\text{补}} \\ \hline 0\ 0110000\ [+48]_{\text{补}} \end{array}$$

$$\begin{array}{r} 1\ 1011111\ [-33]_{\text{补}} \\ +\ 0\ 0001111\ [+15]_{\text{补}} \\ \hline 1\ 1101110\ [-18]_{\text{补}} \end{array}$$

$$\begin{array}{r} 0\ 0100001\ [+33]_{\text{补}} \\ +\ 1\ 1110001\ [-15]_{\text{补}} \\ \hline (1)\ 0\ 0010010\ [+18]_{\text{补}} \end{array}$$

$$\begin{array}{r} 1\ 1011111\ [-33]_{\text{补}} \\ +\ 1\ 1110001\ [-15]_{\text{补}} \\ \hline (1)\ 1\ 1010000\ [-48]_{\text{补}} \end{array}$$

↑
进位，丢掉

↑
进位，丢掉

计算结果不能超过补码表示范围，否则溢出

如果机器字长为8位，计算 $(+64) + (+65) = ?$

$$(+64) + (+65) = +129$$

$$\begin{array}{r} 0\ 1000000 \\ +\ 0\ 1000001 \\ \hline 1\ 0000001 \end{array} \longrightarrow -127$$

正溢出！

十进制数编码运算

BCD码：二进制编码的十进制数，每个十进制数用4位二进制数表示。

- **BCD码(Binary Coded Decimal)**：是二进制编码的十进制数。

十进制数	8421码	十进制数	8421码
0	0000	5	0101
1	0001	6	0110
2	0010	7	0111
3	0011	8	1000
4	0100	9	1001

BCD码的两种格式：

- 压缩BCD码（组合BCD码）：1个字节中存放2位十进制数的BCD码；
- 非压缩BCD码（非组合BCD码）：1个字节中仅存放1位十进制数的BCD码；

例1.18 求十进制数57.3的BCD码。

5 7 . 3

0101 0111 . 0011

所以, $(57.3)_{10} = (01010111.0011)_{\text{BCD}}$

例1.19 求BCD码10000011.0111所对应的十进制数。

10000011 . 0111

8 3 . 7

所以, $(10000011.0111)_{\text{BCD}} = (83.7)_{10}$

十进制数43用**压缩的BCD码**表示为01000011。


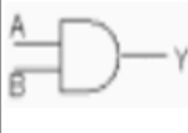

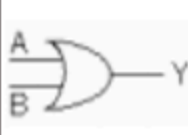



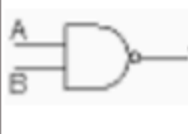
十进制数43用**非压缩的BCD码**表示为××××0100××××0011。

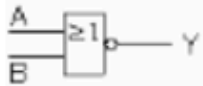






BCD码计算的时候, 如果结果大于9, 需要加6修正

② $(5)_{10} = (0101)_{\text{BCD}}$, $(7)_{10} = (0111)_{\text{BCD}}$

	0	1	0	1	5
+	0	1	1	1	7
<hr/>					
	1	1	0	0	结果大于9
+	0	1	1	0	加6修正
<hr/>					
	1	0	0	1	0
					12

4 逻辑电路

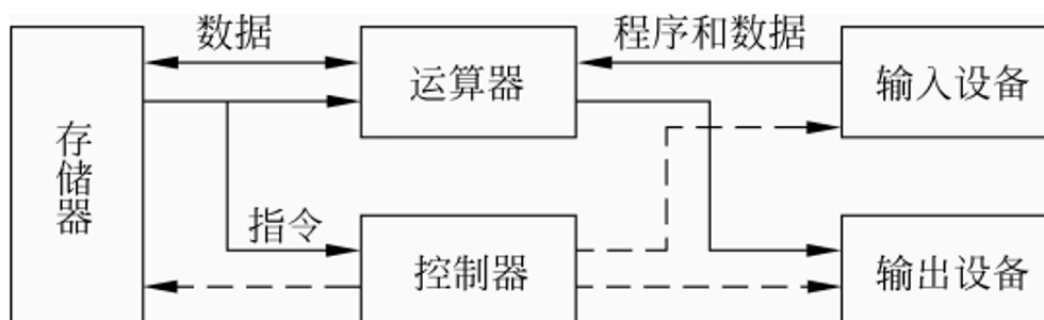
序号	名称	国标图形符号	常用图形符号	逻辑表达式	真值表		
					A	B	F
1	与门			$Y = A \cdot B$	0	0	0
					0	1	0
					1	0	0
					1	1	1
2	或门			$Y = A + B$	0	0	0
					0	1	1
					1	0	1
					1	1	1
3	非门			$Y = \overline{A}$	0		1
					1		0
4	与非门			$Y = \overline{A \cdot B}$	0	0	1
					0	1	1
					1	0	1
					1	1	0

5	或非门			$Y = \overline{A + B}$	0 0 1 0 1 0 1 0 0 1 1 0
6	异或门			$Y = A \oplus B$	0 0 0 0 1 1 1 0 1 1 1 0
7	同或门 (异或非门)			$Y = A \odot B$ $Y = \overline{A \oplus B}$	0 0 1 0 1 0 1 0 0 1 1 1
8	与或非门			$Y = \overline{AB + CD}$	略

计算机系统组成

冯·诺伊曼计算机

- **存储程序控制**的基本思想：将编好的**程序和原始数据**事先存入**存储器**中，然后再启动计算机工作，使计算机在不需要人工干预的情况下，**自动、高速地**从**存储器**中**取出指令**加以**执行**。



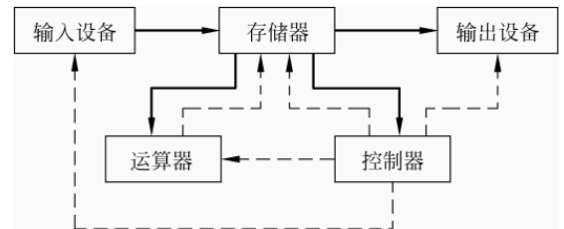
冯·诺依曼结构的计算机是**以运算器为中心的**。

此时是以运算器为中心，存储器和控制器分开的。

之后的现代计算机是以存储器为中心，运算器和控制器合并在一起。

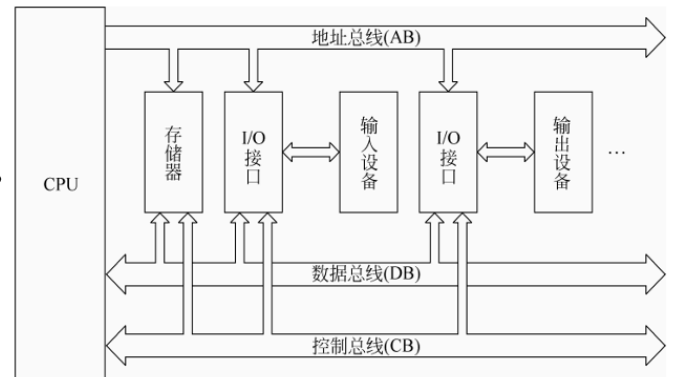
① 以存储器为中心的计算机系统

- 提高了批量数据交换效率



② 总线

- 连接CPU与I / O，提供外设访问内存和CPU资源的通道
- 数据总线(DB): 传送程序或数据
- 地址总线(AB): 传送内存地址
- 控制总线(CB): 传送各种控制信息



总线

总线就是一组共享的通信线路，用于连接计算机各个部件，传送数据和控制信息。

I/O系统

又称作适配器，存在于CPU和外设之间，是信息交换的中转站。用于解决CPU和外设之间的速度不匹配问题。

主要性能指标

字长

位 bit，是计算机进行数据处理的最小单位。字长是CPU一次能处理的二进制数据位数，也是CPU内部寄存器的位数。

字节 byte，是计算机存储数据的基本单位，1字节=8位。