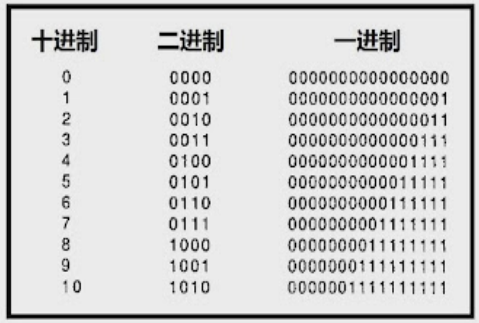

| 十进制 | 二进制 | 一进制 |
| --- | --- | --- |
| 0 | 0000 | 0000000000000000 |
| 1 | 0001 | 0000000000000001 |
| 2 | 0010 | 0000000000000011 |
| 3 | 0011 | 0000000000000111 |
| 4 | 0100 | 0000000000001111 |
| 5 | 0101 | 0000000000011111 |
| 6 | 0110 | 0000000000111111 |
| 7 | 0111 | 0000000001111111 |
| 8 | 1000 | 0000000011111111 |
| 9 | 1001 | 0000000111111111 |
| 10 | 1010 | 0000001111111111 |

二进制与十进制

# 第1.2节 整数的表示和算法

Section 1.2: Integer Representations and Algorithms

# 知识要点

1. 整数n进制展开式
2. 整数运算算法
3. 模指数运算

# 1.2.1 整数表示

- 在日常生活中，我们都用**十进制**(以10为基数)记号来表示整数. 例如765, 我们表示的是$7\cdot10^2 + 6\cdot10^1 + 5\cdot10^0$.

- 有时候, 用10以外的数为基数更方便.
  - 计算机中通常用**二进制**记号(以2为基数)来做算术运算.
  - **八进制**(以8为基数)记号或者**十六进制**(以16为基数)记号来表示字符，如字母和数字.
  - 再比如, 古玛雅人用20为基数的**二十进制**.
  - 再比如, 古巴比伦人用60为基数的**六十进制**.

- 实际上, 我们可以用任何大于1的整数为基数来表示整数.

# 1.2.1 整数表示

- 【定理】：令$b$是一个大于1的整数，则如果$n$是一个正整数，就可以唯一的表示为如下形式$n = a_k b^k + a_{k-1} b^{k-1} + \cdots + a_1 b^1 + a_0 b^0$，其中$k$是非负整数，$a_0$，$a_1$，$a_2$，…，$a_k$是小于$b$的非负整数，且$a_k \neq 0$. 这给出的$n$的表示称为**$n$的$b$进制展开式**. 可记为$(a_k a_{k-1 \ldots} a_1 a_0)_b$

- 例如$(245)_8$表示 $2 \cdot 8^2 + 4 \cdot 8 + 5 = (165)_{10} = 165$，典型地，整数的十进制展开式的下标10可以省略.

# 1.2.1 整数表示

- 在计算机中采用**二进制展开式**来表示整数. 那么每位数字要么是0要么是1, 一个整数的二进制展开式就是一个位串.

- 例: 以 $(101011111)_2$, $(11011)_2$为二进制展开式的整数分别是什么?
- 解:
  - $(101011111)_2 = 1\cdot 2^8 + 0\cdot 2^7 + 1\cdot 2^6 + 0\cdot 2^5 + 1\cdot 2^4 + 1\cdot 2^3 + 1\cdot 2^2 + 1\cdot 2^1 + 1\cdot 2^0 = 351.$
  - $(11011)_2 = 1\cdot 2^4 + 1\cdot 2^3 + 0\cdot 2^2 + 1\cdot 2^1 + 1\cdot 2^0 = 27.$

# 1.2.1 整数表示

- **八进制展开式**是以8为基数的展开式，因此展开式中只有以下数字{0,1,2,3,4,5,6,7}.

- 例:以$(7016)_8$, $(111)_8$为八进制展开式的十进制整数分别是多少?
- 解:
  - $(7016)_8 = 7\cdot 8^3 + 0\cdot 8^2 + 1\cdot 8^1 + 6\cdot 8^0 = 3598$
  - $(111)_8 = 1\cdot 8^2 + 1\cdot 8^1 + 1\cdot 8^0 = 64 + 8 + 1 = 73$

# 1.2.1 整数表示

- **十六进制展开式**需要用到16个不同的数字，通常使用的是{0,1,2,3,4,5,6,7,8,9,A,B,C,D,E,F}, 其中A到F字母表示数字10到15.

- 例: 十六进制展开式$(2\mathrm{AE0B})_{16}$, $(\mathrm{E5})_{16}$的十进制展开式分别是多少?
- 解:
  - $(2\mathrm{AE0B})_{16}=2\cdot16^4+10\cdot16^3+14\cdot16^2+0\cdot16^1+11\cdot16^0=175627$;
  - $(\mathrm{E5})_{16}=14\cdot16^1+5\cdot16^0=224+5=229$

# 1.2.1 整数表示

- 其中十六进制和二进制，每一个十六进制的数字都可以用4位二进制的数字(0或者1)来表示．比如$(1110\ 0101)_2 = (\mathrm{E5})_{16}$．因为$(1110)_2=(\mathrm{E})_{16}, (0101)_2=(5)_{16}$

- 那么任何两种进制之间的转换应该如何计算呢？这儿重点考虑十进制到其他任意进制的转换.

# 1.2.1 整数表示

- **进制的转换**: 构造一个整数$n$(十进制的整数$n$)的$b$进制展开式,
  - 首先, 用$b$除以$n$得到商和余数, 即$n = bq_0 + a_0$ , $0 \le a_0 \le b$. 余数$a_0$就是$n$的$b$进制展开式的最右边的数字.
  - 接下来, 用$b$除以上一步的商$q_0$, 得$q_0 = bq_1 + a_1$, $0 \le a_1 \le b$. 其中$a_1$就是$n$的$b$进制展开式中从右边第二位的数字.
  - 继续以上过程, 连续用商除以$b$并以余数为新的$b$进制数.
  - 这一个过程在商为0时终止. 这个过程中从右向左产生$n$的$b$进制数字.
- 例如, $(5)_{10}=(101)_2$

# 1.2.1 整数表示

- 例: 求$(12345)_{10}$ 的八进制展开式.
- 解:首先用8除以12345得到:
  - $12345 = 8 \cdot 1543 + 1$
  - $1543 = 8 \cdot 192 + 7$(继续用8除以上一次的商)
  - $192 = 8 \cdot 24 + 0$(继续用8除以上一次的商)
  - $24 = 8 \cdot 3 + 0$(继续用8除以上一次的商)
  - $3 = 8 \cdot 0 + 3$(继续用8除以上一次的商)
  - 商为0, 终止.

以上过程中产生了一连串的余数1,7,0,0,3. 它们按照从右向左排列的数字30071就是对应的八进制展开式. 记作$(12345)_{10} = (30071)_8$ .

# 1.2.1 整数表示

- 二进制与八进制、十六进制展开式之间的转换其实是非常容易的.
- 每个八进制数字对应一组三位的二进制数字，而每个十六进制数字对应一组四位的二进制数字.
- 这种对应关系如下表所示, 其中未表示开头的0:

TABLE 1 Hexadecimal, Octal, and Binary Representation of the Integers 0 through 15.

| 十进制: | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 十六进制: | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | A | B | C | D | E | F |
| 八进制: | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 |
| 二进制: | 0 | 1 | 10 | 11 | 100 | 101 | 110 | 111 | 1000 | 1001 | 1010 | 1011 | 1100 | 1101 | 1110 | 1111 |

3位二进制数字可转换为1位八进制数字

4位二进制数字可转换为1位十六进制数字

# 1.2.1 整数表示

- 例: 求$(11\ 1110\ 1011\ 1100)_2$ 的八进制和十六进制展开式.
- 解:
  - 为了求八进制, 每三个为一组, 必要时在最左一组的开头加0. 这样我们得到从左到右为011,111,010,111和100. 对应的八进制数字则为3,7,2,7,4. 那么, 八进制展开式为$(37274)_8$ .
  - 类似地, 为了求十六进制, 每四个为一组, 必要时在最左一组的开头加0. 得到0011,1110,1011,1100. 对应的十六进制数字3,E,B,C. 那么十六进制展开式为 $(3EBC)_{16}$.

# 1.2.1 整数表示

- 例: 求$(765)_8$和 $(\mathrm{A8}D)_{16}$的二进制展开式.
- 解:
  - 为了求二进制，把每个八进制数字转换为一组3个二进制数字. 其中7表示为111, 6表示为110, 5表示为101，于是$(765)_8$=$(1\ 1111\ 0101)_2$.
  - 类似地，把每个十六进制数字转换为一组4个二进制数字. 其中A表示为1010, 8表示为1000, D表示为1101. 因此，$(\mathrm{A8}D)_{16}$=$(1010\ 1000\ 1101)_2$.

# 1.2.2 整数运算算法

- 以前我们学过十进制的加减乘除法则, 比如11+20=31,11*3=33.
- 在计算机中我们是基于二进制来表示整数，那么二进制下如何进行整数运算？这儿重点考虑的是二进制数的加法、乘法、除法中求商、除法中求余.

# 1.2.2 整数运算算法

- 两个二进制展开式表示的整数的**加法算法**:

  其中最基本的2个只有一位的二进制数字相加有以下四种情况:

  0+0=0;

  0+1=1;

  1+0=1;

  1+1=0, 其中进位为1

- 总结:”逢二进一“的核心思想

# 1.2.2 整数运算算法

- 两个二进制展开式表示的整数的**加法算法**:
  - 给定两个整数$a$和$b$，其$n$位的二进制展开式分别为$a=(a_{n-1}a_{n-2}\ldots a_0)_2$ 和$b=(b_{n-1}b_{n-2}\ldots b_0)_2$
  - 要把$a$和$b$相加，首先把最右边的位相加，可得$a_0+b_0=c_0*2+s_0$，其中$s_0$是$a+b$的二进制展开式中最右边数字，$c_0$是进位，其中$c_0$可能为0，也可能为1.
  - 然后把下一对二进制位以及进位相加，可得$a_1+b_1+c_0=c_1*2+s_1$，其中$s_1$是$a+b$的二进制展开式中的从右开始的第二位数字，$c_1$是进位.
  - 循环以上过程，得到$s_0$，$s_1\ldots s_n$. 因此，$a+b=(s_n s_{n-1}\ldots s_1 s_0)_2$.

# 1.2.2 整数运算算法

- 例:把$a=(1110)_2$和$b=(1011)_2$相加.
- 解:
  - 根据算法的步骤, $a_0+b_0$=0+1=$c_0*2+s_0$=0*2+1, 所以$s_0$为1，进位$c_0$为0;
  - 然后, $a_1+b_1+c_0=1+1+0=c_1*2+s_1=1*2+0$, 所以$s_1$为0, 进位$c_1$为1;
  - 然后，$a_2+b_2+c_1=1+0+1=c_2*2+s_2=1*2+0$，所以$s_2$为0，进位$c_2$为1;
  - 然后，$a_3+b_3+c_2=1+1+1=c_3*2+s_3=1*2+1$，所以$s_3$为1，进位$c_3$为1;
  - 因此最后结果为11001. $(1110)_2$+$(1011)_2=(11001)_2$

# 1.2.2 整数运算算法

- 例:把$a=(1110)_2$和$b=(1011)_2$相加.
- 解法2:

$$
\begin{array}{r}
1110 \\
+\ 1011 \\
\hline
11001
\end{array}
\qquad
\begin{array}{l}
\\
\\
1\ 1\ 1
\end{array}
$$

其中斜体表示的是进位

# 1.2.2 整数运算算法

- 两个二进制展开式表示的整数的**乘法算法**:

  其中最基本的2个只有一位的二进制数字相乘有以下四种情况:

  0*0=0;

  0*1=0;

  1*0=0;

  1*1=1

- 总结:二进制数乘法过程可仿照十进制数乘法进行

# 1.2.2 整数运算算法

- **乘法算法**:给定两个$n$位的整数$a$和$b$，其$n$位的二进制展开式分别为$a=(a_{n-1}a_{n-2}\ldots a_0)_2$ 和$b=(b_{n-1}b_{n-2}\ldots b_0)_2$
  - 要把$a$和$b$相乘，利用分配律可以看出$ab = a(b_0 \cdot 2^0 + b_1 \cdot 2^1 + \cdots + b_{n-1} \cdot 2^{n-1}) = a(b_0 \cdot 2^0) + a(b_1 \cdot 2^1) + \cdots + a(b_{n-1} \cdot 2^{n-1})$.
  - 要把$a$和$b$相乘, 首先注意当$b_j$=1时, $ab_j = a$. 当$b_j$=0时, $ab_j = 0$.
  - 当用2乘以一项时, 可以把该项的二进制展开式向左移一位并在尾部加上0. 因而可以通过把$ab_j$的二进制展开式向左移$j$位，再在尾部加上$j$个0来获得$a(b_j \cdot 2^j)$.
  - 最后, 把$n$个整数$a(b_j \cdot 2^j)$, $j$=0,1,2,3,...,$n-1$, 相加就得到$ab$.

# 1.2.2 整数运算算法

- 例:把$a=(110)_2$ 和$b=(101)_2$相乘.
- 解:

$$
\begin{array}{r}
110 \\
\times\ 101 \\
\hline
110 \\
000 \\
110 \\
\hline
11110
\end{array}
$$

首先注意到

- $a(b_0 \cdot 2^0) = (110)_2 * 1 * 2^0 = (110)_2$
- $a(b_1 \cdot 2^1) = (110)_2 * 0 * 2^1 = (0000)_2$
- $a(b_2 \cdot 2^2) = (110)_2 * 1 * 2^2 = (11000)_2$

因此, 把$(110)_2$,$(0000)_2$,$(11000)_2$相加, 可得

$ab=(11110)_2$

# 1.2.2 整数运算算法

- **$div$和$mod$算法**:给定整数$a$和$d$, $d>0$, 计算$q=a$ **$div$** $d$ 和$r=a$ **$mod$** $d$.
  - 当$a$为正时, 就从$a$中尽可能多次减去$d$, 直到剩下的值小于$d$为止. 所做减法的次数就是商, 而最后减剩下的值就是余数.
  - 当$a$为负时, 先求出$|a|$除以$d$的商和余数. 然后当$a<0$, 且$r>0$时, 就用这结果来计算当$a$除以$d$的商为**$-(q+1)$**, 余数**$d-r$**.
- 前面提及-11=3×(-4)+1, 那么$q=-4$, $r=1$. 而不是-11=3×(-3)-2(不满足$r$大于等于0的要求)
- 备注:这儿$a$和$b$没有要求是二进制下的$n$位整数; 此外还有一些其他更快的计算方式, 有兴趣的同学自行阅读课外书籍.

# 1.2.2 整数运算算法

- 例:$a=(11)_2$和$d=(10)_2$, 求$a$ $\boldsymbol{div}$ $d$和$a$ $\boldsymbol{mod}$ $d$的结果
- 解:$(11)_2-(10)_2=(1)_2$, 因为$(1)_2$小于$(10)_2$, 所以$a$ $\boldsymbol{div}$ $d$=减法的次数为1. $a$ $\boldsymbol{mod}$ $d$=多次减法后的剩余值为1.

```
  01   该结果为商, a div d
10)11
   10
   --
    1   该结果为余数, a mod d
```

# 1.2.2 整数运算算法

- 例: $a=(-11)_{10}$ ,$d=(3)_{10}$时求$a$ $\boldsymbol{div}$ $d$和$a$ $\boldsymbol{mod}$ $d$的结果
- 解:
  - 因为$a$小于0, 那么我们首先求出$|a|$=11除以$d$的商和余数. 对应的结果分别为$q$=3和$r$=2.
  - 然后, 当$a$<0且$r$>0时, 就用这结果来计算当$a$除以$d$的商$a$ $\boldsymbol{div}$ $d$= −($q$+1)=-4, 余数$a$ $\boldsymbol{mod}$ $d$ $= d - r = 1$.

# 1.2.3 模指数运算

- 在密码学中，能够有效地计算 $b^n \bmod m$ 很重要，这其中$b, n, m$都是大整数. 采用传统的先计算$b^n$，再求**mod** 的方法不可行，因为$b^n$是一个非常大的数.
- 例如:$123^{1001} \bmod 101$，这儿都是默认的十进制数，当使用计算器还能算出结果.


# 1.2.3 模指数运算

- 例如:$123^{10000} \bmod 101$


# 1.2.3 模指数运算

- 为了解决以上难题$b^n \bmod m$，我们可以采用**模指数运算**. 常用的模指数算法有二进制法, 二进制NAF算法, 滑动窗口算法等. 这儿我们介绍**二进制法**(或称逐次平方法, successive squaring method), 其他方法有兴趣的同学可自行课外学习.
- 令$n$的二进制展开式为$n = (a_{k-1}a_{k-2} \ldots a_1 a_0)_2$. 我们注意到
  - $b^n = b^{(a_{k-1}\cdot 2^{k-1}+\cdots+a_1\cdot 2^1+a_0)}$
  - $= b^{a_{k-1}\cdot 2^{k-1}} * \cdots * b^{a_2\cdot 2^2} * b^{a_1\cdot 2^1} * b^{a_0}$
  - $= (b^{a_{k-1}})^{2^{k-1}} * \cdots * (b^{a_2})^4 * (b^{a_1})^2 * (b^{a_0})^1$, 其中$a_{k-1}, a_{k-2}, \ldots a_1, a_0$要么为1, 要么为0
- 为了计算$b^n$, 只需要计算$b$, $b^2$, $(b^2)^2 = b^4$, $(b^4)^2 = b^8$, ..., $b^{2^{k-1}}$的值. 然后有了这些值, 把其中$a_j = 1$的那些项$b^{2^j}$相乘就可以得到$b^n$的值.

# 1.2.3 模指数运算

- 例:计算$3^{11}$.
- 解:
  - 首先注意11=$(1011)_2$, 因此$3^{11}=3^{1*2^3} * 3^{1*2^1} * 3^{1*2^0} = 3^8 * 3^2 * 3^1$.
  - 通过连续取平方可以得到$3^2$ =9, $3^4 = 9^2$ =81, $3^8 = 81^2$ =6561. 因此, $3^{11}=3^8 * 3^2 * 3^1=6561 * 9 * 3$=177147.

# 1.2.3 模指数运算

- 例:计算$3^{644}$.
- 解:
  - 首先注意$644 = (10\,1000\,0100)_2$，因此$3^{644} = 3^{1*2^9} * 3^{1*2^7} * 3^{1*2^2} = 3^{512} * 3^{128} * 3^4$.
  - 通过连续取平方可以得到$3^2$ =9, $3^4 = 9^2$ =81, $3^8 = 81^2$ =6561, $3^{16}=6561^2$ =43046721, $3^{32}=43046721^2$ =...

- 因此，从这个例子我们看出以上方法中仍然计算量十分庞大．为了提高效率，$b^n \bmod m$中，我们每乘以一项，都做一次**mod** $m$运算，以缩小结果值.

# 1.2.3 模指数运算

- 为了计算$b^n \bmod m$, 依次求出 $b \bmod m$, $b^2 \bmod m$, $b^4 \bmod m$,.... $b^{2^{k-1}} \bmod m$, 并把其中$a_j = 1$的那些项$b^{2^j} \bmod m$相乘, 在每次乘法后求乘积除以$m$所得的余数. 具体伪代码如下所示:

# 1.2.3 模指数运算

- **模指数运算** $b^n \bmod m$ 伪代码

**procedure** modular exponentiation ($b$整数;$n = (a_{k-1}a_{k-2} \ldots a_1a_0)_2$; $m$正整数)
$x$ := 1
power := $b$ **mod** $m$
**for** $i$:=0 **to** $k$-1
    **if** $a_i$ = 1 **then** $x$ := ($x$ * power) **mod** $m$
    power := (power * power) **mod** $m$
**return** $x$ {$x$等于 $b^n \bmod m$}

- 1、$n$用二进制表示
- 2、初始时，$x$置为1, power置为$b$ **mod** $m$

# 1.2.3 模指数运算

- **模指数运算** $b^n \bmod m$ 伪代码

**procedure** modular exponentiation ($b$整数;$n = (a_{k-1}a_{k-2} \ldots a_1a_0)_2$; $m$正整数)
$x$ := 1
power := $b$ **mod** $m$
**for** $i$:=0 **to** $k$-1
    **if** $a_i$ = 1 **then** $x$ := ($x$ * power) **mod** $m$
    power := (power * power) **mod** $m$
**return** $x$ {$x$等于 $b^n \bmod m$}

- 3、for循环内:如果二进制中某一项$a_i$为1, 那么$x$=($x$ *power) **mod** $m$. 同时power=(power*power) **mod** $m$($a_i$为0时, for循环内只执行这个)
- 4、循环结束后, $x$的值为 $b^n \bmod m$的结果

# 1.2.3 模指数运算

- 例:计算$3^{644}$ **mod** 645.
- 解:
  - 644的二进制展开式为$(10\ 1000\ 0100)_2$. 因此$k-1$=9.
  - 首先令$x$=1, power=3 **mod** 645=3.
  - 然后通过连续地取平方并**mod** 645来减少结果值. 计算$3^{2^j}\ \mathbf{mod}\ 645, j = 1,2,\ldots,9$. $a_j$是645的二进制展开式的第$j$位. 如果$a_j$=1, 就在$x$当前值乘以$3^{2^j}\ \mathbf{mod}\ 645$来减少结果值. 具体过程如下:
  - $i$=0:因为$a_0$=0, 所以有$x$=1, power=$3^2$ **mod** 645=9
  - $i$=1:因为$a_1$=0, 所以有$x$=1, power=$9^2$ **mod** 645=81
  - $i$=2:因为$a_2$=1, 所以有$x$=1∗81, power=$81^2$ **mod** 645=111

# 1.2.3 模指数运算

- 解(续):
  - $i$=3:因为$a_3$=0, 所以有$x$= 81, power=$111^2$ **mod** 645=66
  - $i$=4:因为$a_4$=0, 所以有$x$= 81, power=$66^2$ **mod** 645=486
  - $i$=5:因为$a_5$=0, 所以有$x$= 81, power=$486^2$ **mod** 645=126
  - $i$=6:因为$a_6$=0, 所以有$x$=81, power=$126^2$**mod** 645=396
  - $i$=7:因为$a_7$=1, 所以有$x$=(81*396) **mod** 645=471, power=$396^2$ **mod** 645=81
  - $i$=8:因为$a_8$=0, 所以有$x$=471, power=$81^2$**mod** 645=111
  - $i$=9:因为$a_9$=1, 所以有$x$=(471*111) **mod** 645=36
  - 根据以上步骤, 可以得出结果为$3^{644}$ **mod** 645=36.

# 1.2.3 模指数运算

- 例:计算$3^{644}$ **mod** 645.
- 解法2(简化的写法):
  - 644的二进制展开式为$(10\ 1000\ 0100)_2$.
  - $3^{644}=3^4 * 3^{128} * 3^{512}$
  - 然后通过连续地取平方并**mod** 645来减少结果值. 计算$3^2$ **mod** 645=9; $3^4$ **mod** 645=81; $3^8$ **mod** 645=111; $3^{16}$ **mod** 645=66; $3^{32}$ **mod** 645=486; $3^{64}$ **mod** 645=126; $3^{128}$ **mod** 645=396; $3^{256}$ **mod** 645=81; $3^{512}$ **mod** 645=111.
  - 因此, $3^{644}$ **mod** 645=81*396*111 **mod** 645=36

# 1.2.3 模指数运算

- 例:计算$54^{13}$ **mod** 55.
- 解:
  - 13的二进制展开式为$(1101)_2$. 因此$54^{13} = 54^1 * 54^4 * 54^8$.
  - 通过连续地取平方并**mod** 55来减少结果值. $54^2$ **mod** 55=1; $54^4$ **mod** 55=1; $54^8$ **mod** 55=1.
  - 因此, $54^{13}$**mod** 55=54*1*1 **mod** 55=54.

# 第1.2节 整数的表示和算法小结

- 整数n(默认的10进制)的b进制展开式
- b进制下的加法, 乘法, **div**, **mod**运算算法
- $b^n \bmod m$, 模指数运算