计算机组成原理

第二章 数据表示

2.5 CRC校验及其实现

2.5 CRC校验及其实现

1 CRC校验的基本原理

•增加冗余码（校验位）

$N=k+r \le 2^r - 1$

•生成多项式G(x)

收发双方约定的一个(r + 1)位二进制数，发送方利用G(X)对信息多项式做模2除运算,生成校验码。接收方利用G(X)对收到的编码多项式做模2除运算检测差错及错误定位。

•G(x)应满足的条件

A、最高位和最低位必须为1；

B、当被传送信息（CRC码）任何一位发生错误时，被生成多项式做除后应该使余数不为0；

C、不同位发生错误时，模2除运算后余数不同；

D、对不为0余数继续进行模2除运算应使余数循环。

1 CRC校验的基本原理

•常见生成多项式G(x)

N	K	码距d	G(x)多项式	G(x)
7	4	3	x^3+x+1	1011
7	4	3	x^3+x^2+1	1101
7	3	4	$x^4+x^3+x^2+1$	11101
7	3	4	x^4+x^2+x+1	10111
15	11	3	x^4+x+1	10011
15	7	5	$x^8+x^7+x^6+x^4+1$	111010001
31	26	3	x^5+x^2+1	100101
31	21	5	$x^{10}+x^9+x^8+x^6+x^5+x^3+1$	11101101001
63	57	3	x^6+x+1	1000011
63	51	5	$x^{12}+x^{10}+x^5+x^4+x^2+1$	1010000110101

2 模2除运算

•模2运算规则

a)加/减运算 (异或运算，加不进位，减不借位)

0±0=0，0±1=1，1±0=1，1±1=0

b)模2除法

按模2减，求部分余数，不借位。

2 模2除运算

c)上商原则

①部分余数首位为1时，商为1，减除数；

②部分余数首位为0时，商为0，减0；

③当部分余数的位数小于除数的位数时，该余数即为最后余数。

```
                1 0 1          商
          ┌──────────
1 0 1     │ 1 0 0 1 0        首位为 1
          │ 1 0 1            模 2 减，商 1
          ├──────
            0 1 1            首位为 0
            0 0 0            减 0，    商 0
            ─────
              1 1 0          首位为 1
              1 0 1          模 2 减，商 1
              ─────
                1 1          余数位数少于除数，为最后余数
```

3 CRC 编码方法

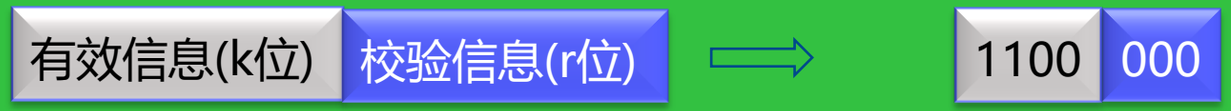
(1)根据待校验信息的长度k，按照 $k+r \le 2^r - 1$ 确定校验位r的位数

如对4位信息 **1100** 进行CRC编码，根据 $4+r \le 2^r - 1$

得 $r_{min} = 3$

(2)根据r 和生成多项式的选择原则，选择位数为 **r +1** 的生成多项式**G(X)= 1011**

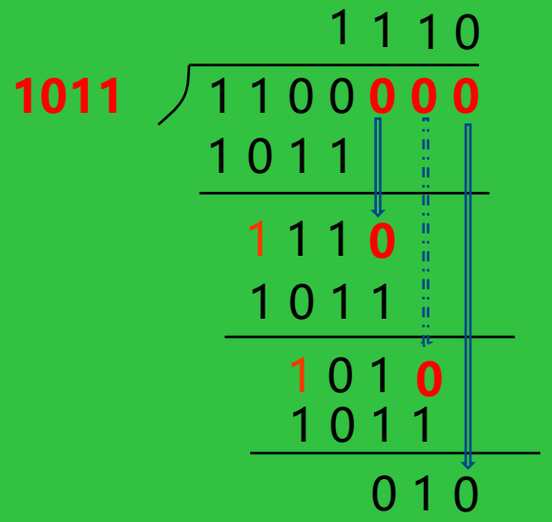
(3)进行下列变化



即：将待校验的二进制信息Q(X)逻辑左移 r 位,得到Q(X)'

3 CRC编码

(4)对Q(X)’ 按模2运算法则除G(x)，求CRC编码中的r位校验信息

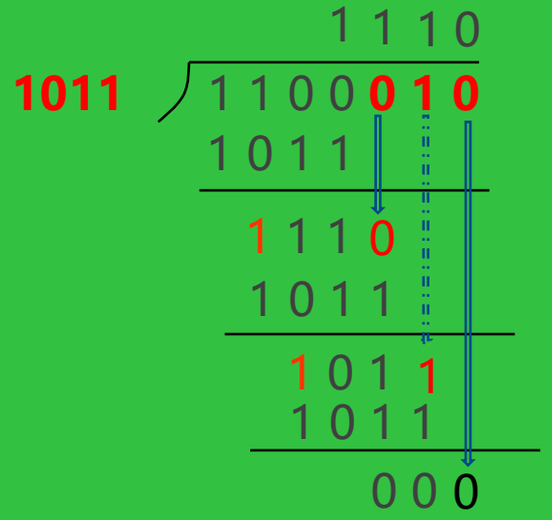


(5)用得到的余数替换Q(X)’ 的最后r位即可得到对应的CRC编码

即为1100 的CRC 编码

4 CRC的检错与纠错

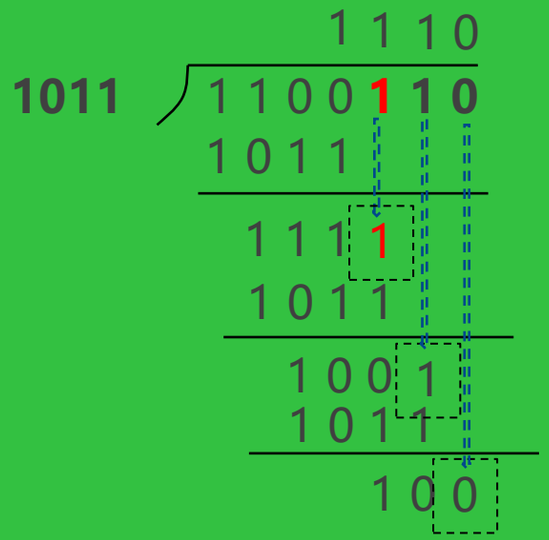
接收方利用G(X)对收到的编码多项式做模2除运算

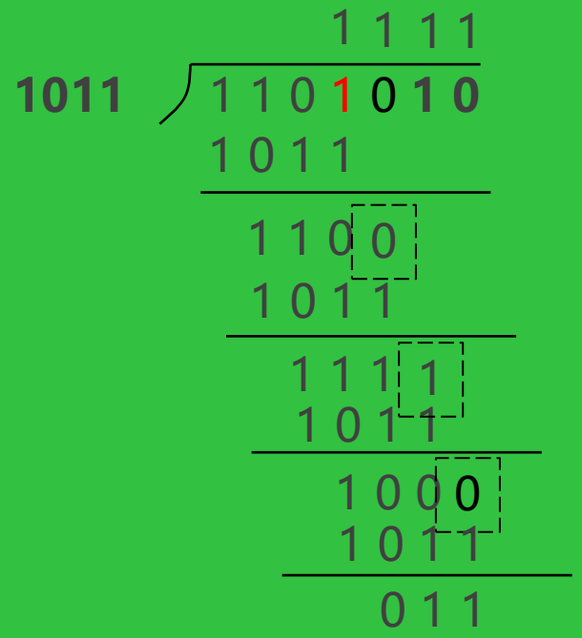


余数为0说明传输没有错误

4 CRC的检错与纠错

接收方利用G(X)对收到的**有错**编码多项式做模2除运算





余数不为0说明传输有错

4 CRC的检错与纠错

- （7,4）编码不同数位出错对应的余数

A_1~A_7	余数	出错位
1100010	000	无
1100011	001	7
1100000	010	6
1100110	100	5
1101010	011	4
1110010	110	3
1000010	111	2
0100010	101	1

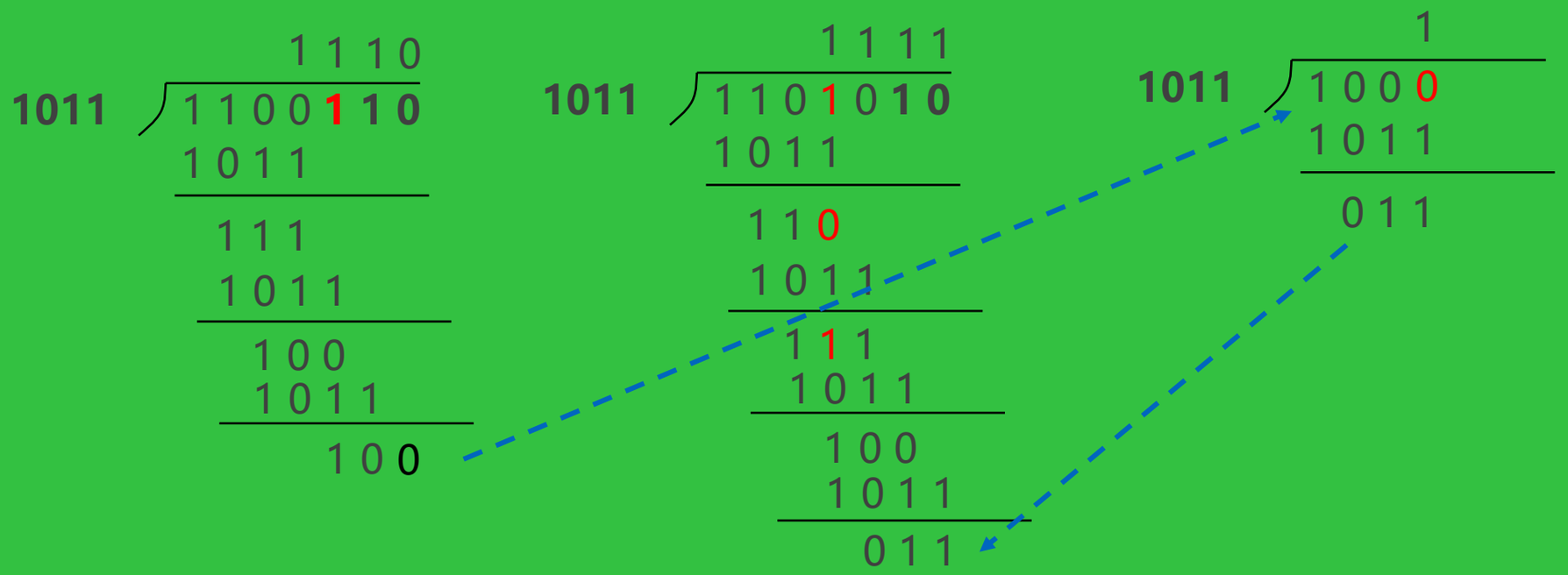
G(X) = 1011

A_1~A_7	余数	出错位
1100101	000	无
1100100	001	7
1100111	010	6
1100001	100	5
1101101	101	4
1110101	111	3
1000101	011	2
0100101	110	1

G(X) = 1101

4 CRC的检错与纠错

- 一位出错情况下余数的循环特性



4 CRC的检错与纠错

- 利用出错情况下余数的循环特性进行纠错

A_1~A_7	余数	出错位
1100010	000	无
1100011	001	7
1100000	010	6
1100110	100	5
1101010	011	4
1110010	110	3
1000010	111	2
0100010	101	1

若余数不为0，一边对余数补0继续做模2除，同时让被检测的校验码循环左移，当余数为101时，出错位也移到A1位置。通过异运算纠正后继续循环左移和执行余数模2除法，直到修改后的出错位回原位。不需对每一位提供纠正电路。

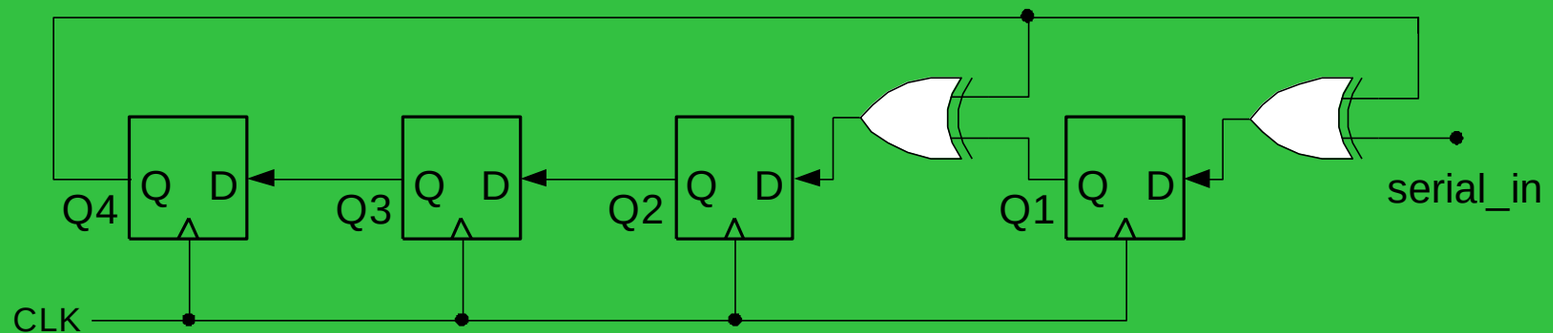
当位数增多时，循环码校验能有效地降低硬件代价，这是它得以广泛应用的主要原因。

N	K	码距d	G(x)多项式	G(x)
7	4	3	x^3+x+1	1011
7	4	3	x^3+x^2+1	1101
7	3	4	$x^4+x^3+x^2+1$	11101
7	3	4	x^4+x^2+x+1	10111
15	11	3	x^4+x+1	10011
15	7	5	$x^8+x^7+x^6+x^4+1$	111010001
31	26	3	x^5+x^2+1	100101
31	21	5	$x^{10}+x^9+x^8+x^6+x^5+x^3+1$	11101101001
63	57	3	x^6+x+1	1000011
63	51	5	$x^{12}+x^{10}+x^5+x^4+x^2+1$	1010000110101

5 CRC 编码的实现

- 硬件实现方式

```
          1 1 0 1
      ┌──────────
10011 ) 1 1 0 0 0 0 0 0
        1 0 0 1 1
        ---------
          1 0 1 1 0
          1 0 0 1 1
          ---------
            1 0 1 0 0
            1 0 0 1 1
            ---------
              0 1 1 1
```



1 1 0 0 0

1 0 1 1 0

1 0 1 1 0

5 CRC 的应用

- 软件实现方式

```
/******该文件使用查表法计算CCITT 标准的CRC-16检验码，并附测试代码********/
#include
#define CRC_INIT 0xffff   //CCITT初始CRC为全1
#define GOOD_CRC 0xf0b8   //校验时计算出的固定结果值
/****下表是常用ccitt 16,生成式1021反转成8408后的查询表格****/
unsigned short crc16_ccitt_table[256] =
{      0x0000, 0x1189, 0x2312, 0x329b, 0x4624, 0x57ad, 0x6536, 0x74bf,0x8c48, 0x9dc1, 0xaf5a, 0xbed3, 0xca6c, 0xdbe5, 0xe97e, 0xf8f7,0x1081, 0x0108, 0x3393,
0x221a, 0x56a5, 0x472c, 0x75b7, 0x643e,0x9cc9, 0x8d40, 0xbfdb, 0xae52, 0xdaed, 0xcb64, 0xf9ff, 0xe876,0x2102, 0x308b, 0x0210, 0x1399, 0x6726, 0x76af, 0x4434,
0x55bd,0xad4a, 0xbcc3, 0x8e58, 0x9fd1, 0xeb6e, 0xfae7, 0xc87c, 0xd9f5,0x3183, 0x200a, 0x1291, 0x0318, 0x77a7, 0x662e, 0x54b5, 0x453c,0xbdcb, 0xac42, 0x9ed9,
0x8f50, 0xfbef, 0xea66, 0xd8fd, 0xc974,0x4204, 0x538d, 0x6116, 0x709f, 0x0420, 0x15a9, 0x2732, 0x36bb,0xce4c, 0xdfc5, 0xed5e, 0xfcd7, 0x8868, 0x99e1, 0xab7a,
0xbaf3,0x5285, 0x430c, 0x7197, 0x601e, 0x14a1, 0x0528, 0x37b3, 0x263a,0xdecd, 0xcf44, 0xfddf, 0xec56, 0x98e9, 0x8960, 0xbbfb, 0xaa72,0x6306, 0x728f, 0x4014,
0x519d, 0x2522, 0x34ab, 0x0630, 0x17b9,0xef4e, 0xfec7, 0xcc5c, 0xddd5, 0xa96a, 0xb8e3, 0x8a78, 0x9bf1,0x7387, 0x620e, 0x5095, 0x411c, 0x35a3, 0x242a, 0x16b1,
0x0738,0xffcf, 0xee46, 0xdcdd, 0xcd54, 0xb9eb, 0xa862, 0x9af9, 0x8b70,0x8408, 0x9581, 0xa71a, 0xb693, 0xc22c, 0xd3a5, 0xe13e, 0xf0b7,0x0840, 0x19c9, 0x2b52,
0x3adb, 0x4e64, 0x5fed, 0x6d76, 0x7cff,0x9489, 0x8500, 0xb79b, 0xa612, 0xd2ad, 0xc324, 0xf1bf, 0xe036,0x18c1, 0x0948, 0x3bd3, 0x2a5a, 0x5ee5, 0x4f6c, 0x7df7,
0x6c7e,0xa50a, 0xb483, 0x8618, 0x9791, 0xe32e, 0xf2a7, 0xc03c, 0xd1b5,0x2942, 0x38cb, 0x0a50, 0x1bd9, 0x6f66, 0x7eef, 0x4c74, 0x5dfd,0xb58b, 0xa402, 0x9699,
0x8710, 0xf3af, 0xe226, 0xd0bd, 0xc134,0x39c3, 0x284a, 0x1ad1, 0x0b58, 0x7fe7, 0x6e6e, 0x5cf5, 0x4d7c,0xc60c, 0xd785, 0xe51e, 0xf497, 0x8028, 0x91a1, 0xa33a,
0xb2b3,0x4a44, 0x5bcd, 0x6956, 0x78df, 0x0c60, 0x1de9, 0x2f72, 0x3efb,0xd68d, 0xc704, 0xf59f, 0xe416, 0x90a9, 0x8120, 0xb3bb, 0xa232,0x5ac5, 0x4b4c, 0x79d7,
0x685e, 0x1ce1, 0x0d68, 0x3ff3, 0x2e7a,0xe70e, 0xf687, 0xc41c, 0xd595, 0xa12a, 0xb0a3, 0x8238, 0x93b1,0x6b46, 0x7acf, 0x4854, 0x59dd, 0x2d62, 0x3ceb, 0x0e70,
0x1ff9,0xf78f, 0xe606, 0xd49d, 0xc514, 0xb1ab, 0xa022, 0x92b9, 0x8330,0x7bc7, 0x6a4e, 0x58d5, 0x495c, 0x3de3, 0x2c6a, 0x1ef1, 0x0f78 };
unsigned short do_crc(unsigned short reg_init, unsigned char *message, unsigned int len)
{
    unsigned short crc_reg = reg_init;
      while (len--)
        crc_reg = (crc_reg >> 8) ^ crc16_ccitt_table[(crc_reg ^ *message++) & 0xff];
      return crc_reg; }
```

https://blog.csdn.net/yazhouren/article/details/50186497

5 CRC 的应用

- 关于CRC的国际标准(节选)

标准名称	生成多项式	表示法(省略了最高位1)
CRC-1	X+1 (用途：硬件，也称为奇偶校验位)	0x1 （11）
CRC-5-CCITT	X^5+X^3+X+1 (ITU G.704标准)	0xB （101011）
CRC-5-USB	X^5+X^2+1 (USB信令包)	0x5
CRC-7	X^7+X^3+X+1 (用途：通信系统)	0x9
CRC-8-ATM	X^8+X^2+X+1 (用途：ATM HEC)	0x7
CRC-12	$X^{12}+X^{11}+X^3+X^2+X+1$ (通信系统)	0x80F
CRC-16-CCITT	$X^{16}+X^{12}+X^5+1$ (X25,V.41,Bluetooth,PPP, IrDA)	0x1021
CRC-32-MPEG2	IEEE 802.3 以太网协议	IEEE 802.3