# CRC

一、CRC校验原理

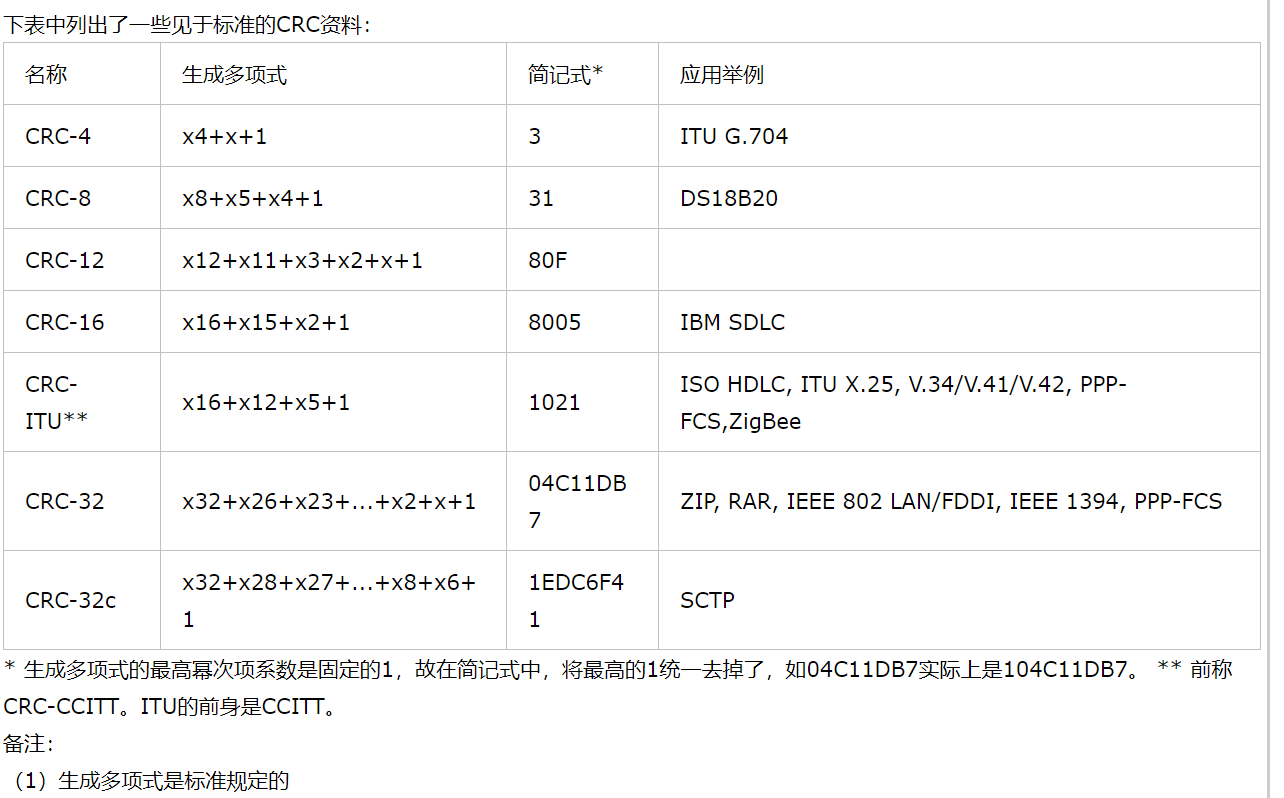
CRC是cyclic redundancy check的缩写，循环冗余校验。一种根据网络数据包或电脑文件等数据产生简短固定位数校验码的一种散列函数，主要用来检测或校验数据传输或者保存后可能出现的错误。生成的数字在传输或者存储之前计算出来并且附加到数据后面，然后接收方进行检验确定数据是否发生变化。

CRC是校验和的一种。

 CRC校验原理看起来比较复杂，好难懂，因为大多数书上基本上是以二进制的多项式形式来说明的。其实很简单的问题，其根本思想就是先在要发送的帧后面附加一个数（这个就是用来校验的校验码，但要注意，这里的数也是二进制序列的，下同），生成一个新帧发送给接收端。当然，这个附加的数不是随意的，它要使所生成的新帧能与发送端和接收端共同选定的某个特定数整除（注意，这里不是直接采用二进制除法，而是采用一种称之为“**模2除法**”）。到达接收端后，再把接收到的新帧除以（同样采用“**模2除法**”）这个选定的除数。因为在发送端发送数据帧之前就已通过附加一个数，做了“去余”处理（也就已经能整除了），所以结果应该是没有余数。如果有余数，则表明该帧在传输过程中出现了差错。

二、标准

不同的生成多项式，对应不同的标准



2）CRC校验码是基于将位串看作是系数为0或1的多项式，一个k位的数据流可以看作是关于x的从k-1阶到0阶的k-1次多项式的系数序列。采用此编码，发送方和接收方必须事先商定一个生成多项式G(x)，其高位和低位必须是1。要计算m位的帧M(x)的校验和，基本思想是将校验和加在帧的末尾，使这个带校验和的帧的多项式能被G(x)除尽。当接收方收到加有校验和的帧时，用G(x)去除它，如果有余数，则CRC校验错误，只有没有余数的校验才是正确的。

(3) 名称 生成多项式 简记式\* 标准引用

CRC-4 x4+x+1 3                 ITU G.704

CRC-8 x8+x5+x4+1 0x31

CRC-8 x8+x2+x1+1 0x07

CRC-8 x8+x6+x4+x3+x2+x1 0x5E

CRC-12 x12+x11+x3+x2+x+1              80F

CRC-16 x16+x15+x2+1 8005 IBM SDLC

CRC16-CCITT x16+x12+x5+1 1021 ISO HDLC,ITU X.25,V.34/V.41/V.42, PPP-FCS

CRC-32 x32+x26+x23+...+x2+x+1 04C11DB7 ZIP, RAR, IEEE 802 LAN/FDDI, IEEE 1394, PPP-FCS

CRC-32c x32+x28+x27+...+x8+x6+1    1EDC6F41     SCTP。

其中CRC-4  0x3=11 最高次幂省略，其余位次对应幂次，即对应多项式有0、1、4次幂。

依次类推：CRC-8      0x31=110001  即对应多项式有0、4、5、8次幂。

CRC-16    0x8005= 1000000000000101    即对应多项式有0、2、15、16次幂。

三、CRC CCITT

<https://www.lammertbies.nl/comm/info/crc-calculation>

* Width = 16 bits
* Truncated polynomial = 0x1021
* Initial value = 0xFFFF
* Input data is NOT reflected
* Output CRC is NOT reflected
* No XOR is performed on the output CRC

分析其计算过程，在需要计算的数值前面加上16bit1，在后面加上16个0。作为被除数。除数为1,0001,0000,0010,0001。

若采用串行方式实现，可以将输出初始化为0xFFFF。由于每次异或（除法中的减法）的最高位都为0，所以只需要将每次异或的最高位去掉，在最低位补充一个输入值即可。

<http://srecord.sourceforge.net/crc16-ccitt.html>