

+	0	1	x	$x+1$
0	0	1	x	$x+1$
1		0	$x+1$	x
x			0	0
$x+1$				

$F_4[x]$  上的加法，等同于多项式在后域上的加法。  
 $\Downarrow$

当两个多项式相加时，实际上是在它们对应的比特位进行逐位  $XOR$ 。

对于  $F_4[x]$  来说， $\{0, 1, x, x+1\}$ ，两个元素相乘时，结果次数可能超过一次，如  $x \cdot x = x^2$ ，而  $x^2$  并不在  $F_4[x]$  中，使用  $r(x) = x^2 + x + 1$  进行取模。

因为在模运算世界里，模数 Modulus 本身就是除数，即  $0$  (12点，也是 0 点)

所以，在以  $r(x) = x^2 + x + 1$  为模的运算中，只要出现这个完满多项式，它就被消掉。

所以  $x^2 + x + 1 \equiv 0$        $x^2 \equiv -1(x+1)$       又因为  $x+a = x-a \therefore x^2 \equiv (x+1)$   
 得到高次项，除  $r(x)$ ，加减号变一下，得余数  
 $x^2 : (x^2 + x + 1) = 1$        $\therefore 1 \cdot (x^2 + x + 1) - (x+1) = x^2$   
 $\sim$   
 $= + (x+1)$        $\star$  减号加号互换。

余数就是乘积的结果。

$$d(x) = (a(x) \cdot b(x)) \cdot \text{Mod } r(x)$$



也可以用降次公式快速求。

·	0	1	x	$x+1$
0	0			
1	0	①1		
x	0	x	② $x^2 = x+1$	
$x+1$	0	$x+1$	③ $x^2 + 2x + 1$	2x 直接删掉， $(+1)x = 0 \cdot x = 0$
			$x^2 + x + 1$	$\frac{x^2 + 1}{x^2 + x + 1} = 1$ 余 $x$

在实际的计算如以上图中，我们通常不选择下面的多项式。

$r(x) = p(x) \cdot (x+1)$ ，它可以分解为两个多项式，一定有  $(x+1)$

$e(x)$  实际上记录了错误发生在哪一位， $e(1)$  是总中共的错误数，

$e(1)$  在 mod 2 运算下为 1，说明 错误数是奇数 odd.

如果  $e(x)$  能被  $r(x)$  整除，则  $e(x) = f(x) \cdot r(x)$ .  $x=1$  时  $e(1) = f(1) \cdot 1$

又： $r(1) = 0 \therefore e(1) = 0$  才能整除，但有错，奇数时， $e(1) = 1$ ，所以下整除。

只要错误位数是奇数，接收方算出的余数绝对不是 0，错误会被发现。

- eg: Reduction Polynomial  $r(x) = x^3 + x^2 + 1$ , data  $m(x) = x^7 + x^5 + x^2 + 1$
- ① determine coefficients:  $r(x) \hat{=} 1101$  and  $m(x) \hat{=} 10100101$
  - ②  $\deg(r(x)) = 3$   $\Rightarrow m(x)$  multiply by  $x^3$ ,  $m'(x) \hat{=} 10100101000$
  - ③  $c(x) = m'(x) \bmod r(x)$

degree 就是最高位所在的索引值

$$\begin{array}{r}
 \overline{10100101000} \\
 1101 \\
 \hline
 01110 \\
 1101 \\
 \hline
 001110 \\
 1101 \\
 \hline
 001110 \\
 1101 \\
 \hline
 001100 \\
 1101 \\
 \hline
 0001
 \end{array}$$

- CRC32 能够检测出高概率 (约 100%) 不检测到的错误
- ① Long burst burst error: 宽发错误长度大于 32 位
  - ② 偶数个多个比特错误
  - ③ 有多个 burst 但成的复合错误, 且  $L > n$

注:  $C(x) = 001$ , 取  $\deg(r(x))$  为

接收方收到比错了  $S'(x)$ ,  $C'(x) = S'(x) \bmod r(x)$ , 看余数是否为 0

$C'(x) \neq 0 \Rightarrow$  一定有错,  $C'(x) = 0$  大概率无错.

CRC 理论上可以纠错, 但实际上平常这么做

eg: ATM: Asynchronous Transfer Mode, use a 1B checksum to detect 1, 2, 3 bit, can correct single bit errors in the 4B ATM Header

Bluetooth: secure 10bit data blocks with 95bit checksum, correct single bit

不支持纠错: 以太网

- ① 帧的帧长可达 1500 字节. ( $1500 \times 8 = 12000$  bits) 下面称 jumbo frames.
- ② Code rate 极高, 0.997, 全是数据, 错误极少

## Error Detection:

尽管物理层已使用了 Channel Coding 如 4B5B 或 Manchester，但在实际应用中，由于噪声、干扰或信号衰减，Bit errors 仍然是不可避免的 (Transmission Error)。

信道编码 L<sub>1</sub>: 它会纠正错误 Error-correcting codes，尝试直接修复比特链路层校验和：仅用于 Error detection，不进行纠错。

↳ 通常直接丢弃该帧，至于是否重传，交给更高层级如 TCP 处理。

这个它本身被 Discard，永不至 L<sub>3</sub>、L<sub>4</sub>。TCP 会确认是否丢了包。

## CRC:

目标：① Detect large number of errors: Single-bit Multibit errors

CRC 强项：突发错误：Burst errors，短时间出现集中的连续错误位。

② Small amount of added redundancy 见余量极小，增加额外 bits

③ Detect, not correct

比特串的多项式表示：(一个 nT<sub>2</sub> 的比特串，其最高次幂是 n-1)

$$a(x) = \sum_{i=0}^{n-1} a_i x^i \text{ with } a_i \in F_2 \text{ with } F_2 = \{0, 1\}$$

所有长度为 nT<sub>2</sub> 的比特串构成一个集合  $F_q[x]$

$$F_q[x] = \left\{ a \mid a(x) = \sum_{i=0}^{n-1} a_i x^i, a_i \in F_2 \right\} \quad \text{Finite extension field}$$

所有的系数运算都在  $F_2$  中进行，意味着在运算中系数相加不进位， $1+1=0$ 。

加法 Summation:  $c(x) = a(x) + b(x) = \sum_{i=0}^{n-1} (a_i + b_i) x^i$

$0+0=0$ ,  $0+1=1$ ,  $1+0=1$ ,  $1+1=0$ . 两多项式  $a(x)$ ,  $b(x)$  相加  $\Leftrightarrow$  其系数  $a_i + b_i$  在  $F_2$ 。

数表 A: 1011

B 1100 → 0111, 相同加为 0, 不同为 1,  $a-b=a+b$ .

通过这样下进位保证  $A+B$  在四位，但相乘就一定无法保证在四位。

所以我们要对乘法出来很长的多项式做一个操作，除  $r(x)$  ( $n$  阶) 为了保证乘法运算结果仍在一个固定的集合  $F_q[x]$  中，我们引入 Reduction Polynomial

$$d(x) = (a(x) \cdot b(x)) \bmod r(x)$$

无论原始消息多长，通过模  $r(x)$  运算，产生的校验和长度始终由  $r(x)$  的阶数决定。

E CRC: 为选定的数据块, 如 L2-DPU, 计算出一个固定长度的 Check SUM.

所有的 Code Words 被抽象为有限域  $F_2[x]$  上的多项式.

准备:  $m(x)$ : 原始消息多项式, degree 为  $k$ , 长度为  $k+1$  位.

$r(x)$ : reduction polynomial, degree 为  $n$ .

发送端: ① Append  $n$  zeros: 在原始消息  $m(x)$  末尾加上  $n$  位 0.

$$m'(x) = m(x) \cdot x^n$$

② Determine the remainder

$$c(x) = m'(x) \bmod r(x) \quad \text{木莫二除法: 下进 1, 不借位, 加减 1 行于 } x \oplus R$$

算出的余数  $c(x)$  就是校验和. Check sum

③ 最终发送  $s(x) = m'(x) + c(x)$ ,  $s(x)$  一定能被  $r(x)$  整除.

接收端: 收到  $s'(x) = s(x) + e(x)$ .  $e(x)$  是错误多项式, 可能为 0 或

3,  $e(x)$  上的  $x^{n-1}$  就是 1, 与  $s(x)$  上对应的  $x^{n-1}$  对应, 于是  $s'(x)$

① 余数  $c'(x) = s'(x) \bmod r(x)$

$$\therefore s(x) = s(x) + e(x), s(x) \bmod r(x) = 0,$$

$$\therefore c'(x) = (s(x) + e(x)) \bmod r(x) = \boxed{e(x) \bmod r(x)}$$

接收端在这一生成的余数 本质上就是  $e(x) / r(x)$  的余数.

②  $\Leftrightarrow c'(x) = 0$ . High Probability 没有发生错误, 接收数据并向上层.

但如果错误  $e(x)$  小数是  $r(x)$  的倍数, 余数也是 0.

如  $c'(x) \neq 0$ , ~~not~~ For sure (100%) 发生了错误, 直接丢弃

$r(x)$  特征: 以下三个是 100% 能检测到的错误类型.

①  $r(x)$  只要有两项不能检测到单比特错误: 假设 8 位的  $e(x) = 00000000$ , 第 8 位,

$e(x)$  变成了一个  $x^7$  是 1, 其余位是 0 的:  $e(x) = 0 \cdot x^7 + 0 \cdot x^6 + 1 \cdot x^5 + 0 \cdot x^4 + \dots$

则单比特错误,  $e(x)$  只有一项  $x^i$ ,

$$= \boxed{x^i}$$

$\therefore c'(x) = e(x) \bmod r(x)$ , 只要  $r(x)$  有两项及以上,  $c'(x) \neq 0$

② 只要  $r(x)$  包含因子  $(x+1)$ , 就能检测奇数个错误.

如单发长反显

③ Burst error: 只要 突发长度 小于等于  $r(x)$  的阶数  $n$ , 一定能检测出, 这  $n$  错误就可

$e(x) = 0 \cdot x^7 + 0 \cdot x^6 + 1 \cdot x^5 + 1 \cdot x^{i-1} + \dots + 1 \cdot x^{i-L+1} + 0 \cdot x^L \dots + 0$ . 错误长度为  $L$  能被隐藏!

$= x^i \cdot P_{L-1}(x)$ , 一个阶数为  $L-1$  的多项式, 它的常数项必然为 1, 把  $x^i$  提出来.

突发长度中间也不一定全 1, 可以有 0,  $c'(x) = [x^i \cdot P_{L-1}(x)] \bmod r(x)$

$\because r(x)$  有 2 项以上,  $\therefore x^i$  不整除  $r(x)$ ,  $P_{L-1}(x) \bmod r(x)$ , 只要  $L \leq n$ , 则  $L-1 \leq n$ , 不能整除