



$r = 5!$

Data --> 1010011010 Polynomial -->  $X^5 + X^4 + X^2 + 1$  (110101)

C4C3C2C1C0	$C4 \oplus C3$	$C4 \oplus C1$	$C4 \oplus C_{in}$	$C_{in}$
00000	0	0	1	1
00001	0	0	0	0
00010	0	1	1	1
00101	0	0	0	0
01010	1	1	0	0
10100	1	1	0	1
11100	0	1	0	1
01100	1	0	0	0
11000	0	1	0	1
00100	0	0	0	0
01000	1	0	0	0
10000	1	1	1	0
10101	1	1	1	0
11111	0	0	1	0
01011	1	1	0	0
10110	1	1	0	0

10110 ← --- remainder = CRC error check bits appended to data

Message = 101001101010110

OR we can use the following technique! ( note that the xor operation is used.)

```

110101 | 101001101000000
      110101
      111001
      110101
      110001
      110101
      100000
      110101
      101010
      110101
      111110
      110101
      10110 <-- remainder!

```

Either way, if the incoming message (including CRC bits) is again divided by the same polynomial and there are no bit errors the remainder will be zero. Verify this to yourself.

*Error-detection performance of cyclic redundancy checksum*

Type of Error	Error Detection Performance
Single-bit errors	100 percent
Double-bit errors	100 percent, as long as the generating polynomial has at least three 1s (they all do)
Odd number of bits in error	100 percent, as long as the generating polynomial contains a factor $x + 1$ (they all do)
An error burst of length $< r + 1$	100 percent
An error burst of length $= r + 1$	probability = $1 - (1/2)^{r-1}$
An error burst of length $> r + 1$	probability = $1 - (1/2)^r$

## Hamming Code Example

$m = \#$  of data bits = 8

$r = \#$  check bits where  $m + r + 1 \leq 2^r$

$$8 + r + 1 \leq 2^r$$

$$9 + r \leq 2^r$$

$$\therefore r \geq 4$$

let  $r = 4$  which means we need 4 check bits to **find and**

**correct a single error.**

Place check bits in powers of 2 positions:

D7D6D5D4P3D3D2D1P2D0P1P0

12 11 10 9 8 7 6 5 4 3 2 1 ← numbered bit positions

Using even parity and a data byte of 01001011 gives:

010011010110

If we receive: (note that bit 9, D4 is an error)

010111010110

P0 – bit 1 covers 1,3,5,7,9,11	X
P1 – bit 2 covers 2,3,6,7,10,11	ok
P2 – bit 4 covers 4,5,6,7,12	ok
P3 – bit 8 covers 8,9,10,11,12	X

(example:  $1 = 1$ ,  $3 = 1+2$ ,  $5 = 1+4$ ,  $7 = 1+2+4$ ,  $9 = 1+8$ ,  $11 = 1+2+8$ )

We see that the two instances where an error is incated both contain 9 and 11, but 11 must be ok because it is covered by P1, therefore the error is bit 9.

OR

Bit 1 plus Bit 8 → Bit 9 in error.