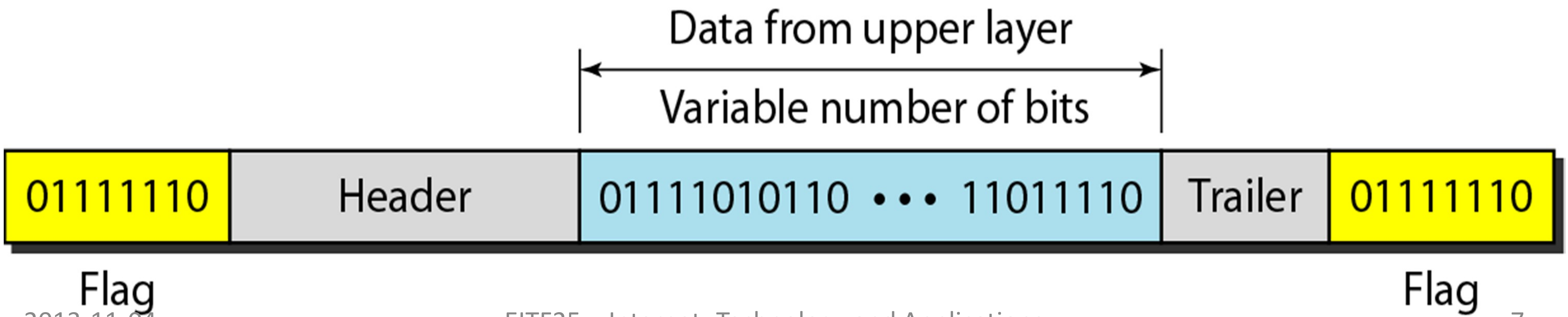
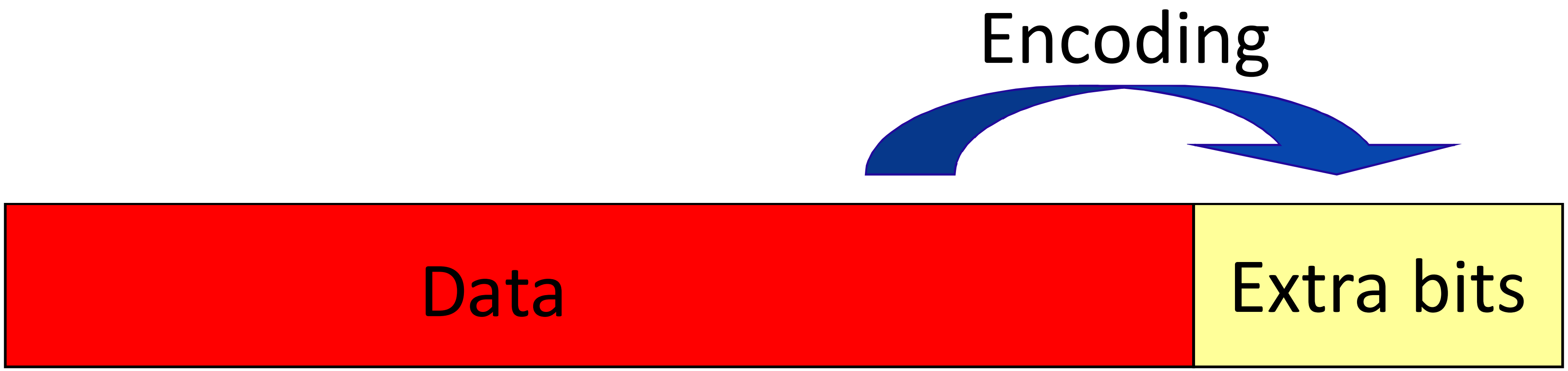
Framing

- Physical layer → bitstream
- Link layer → frames
- We need logical transmission units
  - Synchronisation points
  - Switching between users
  - Error handling



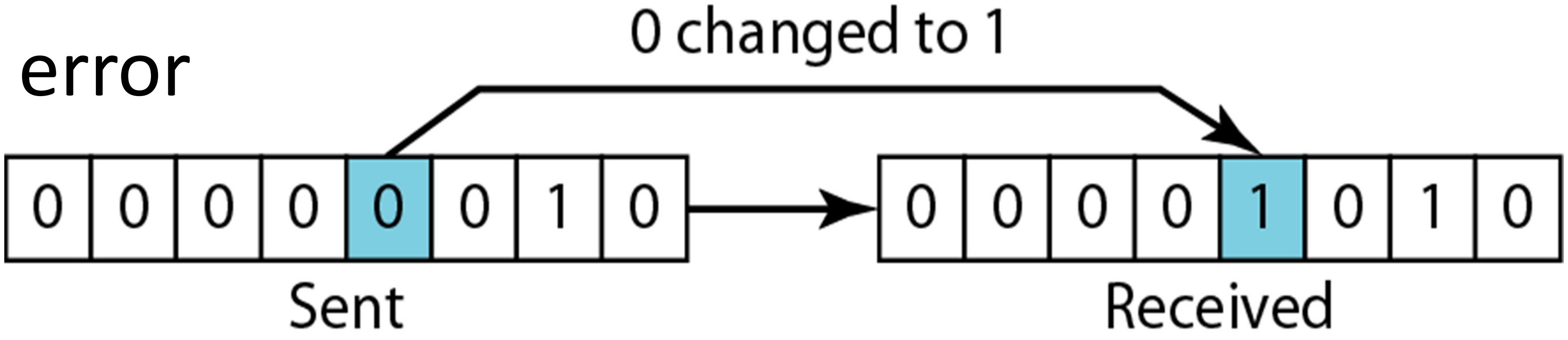
Error control

- Data assumed error-free by higher layers
  - Errors occur at lower layers (physical)
  - Job for LLC layer
- Extra (redundant) bits added to data
  - Generated by an encoding scheme from data

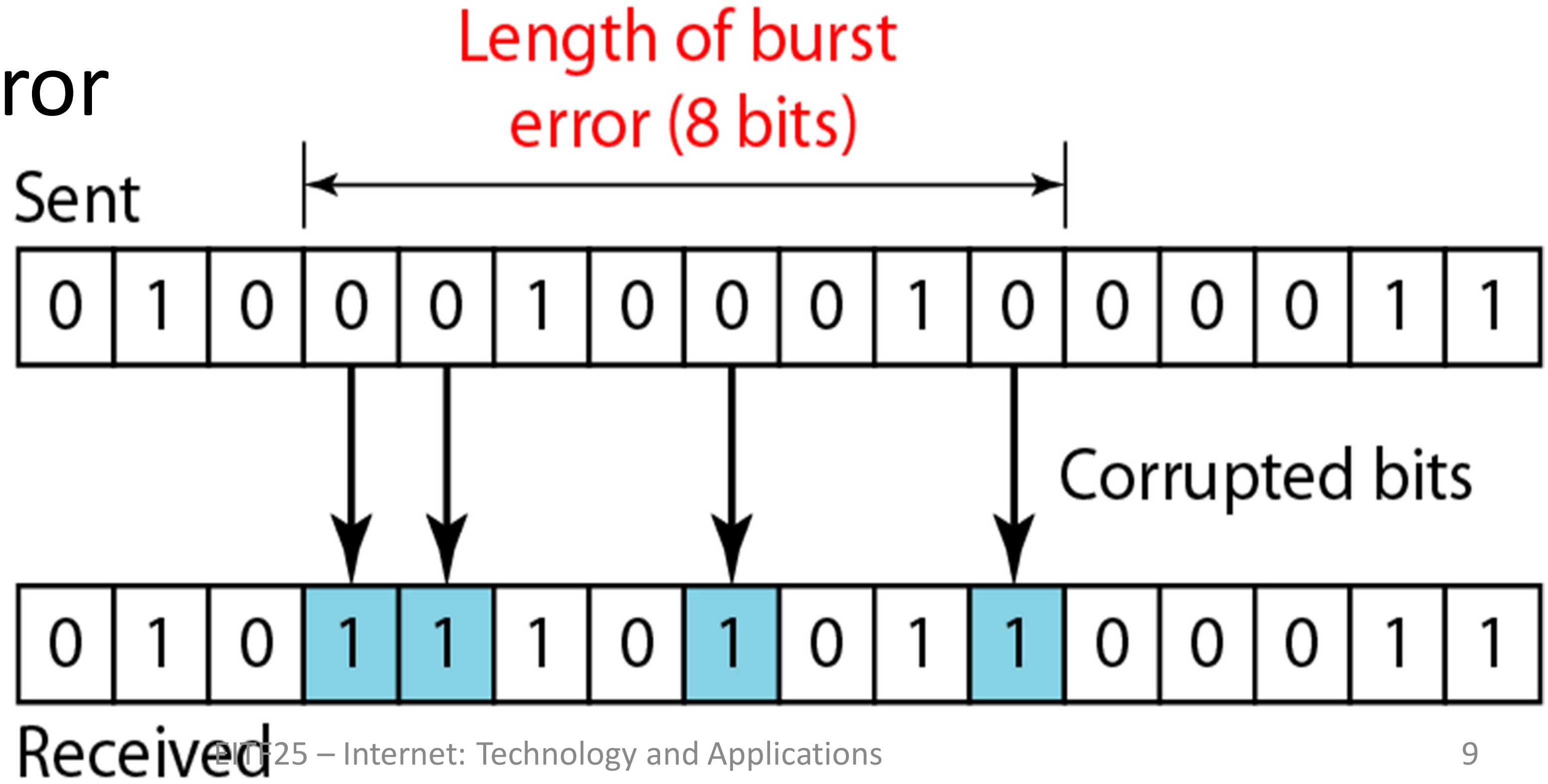


Error types

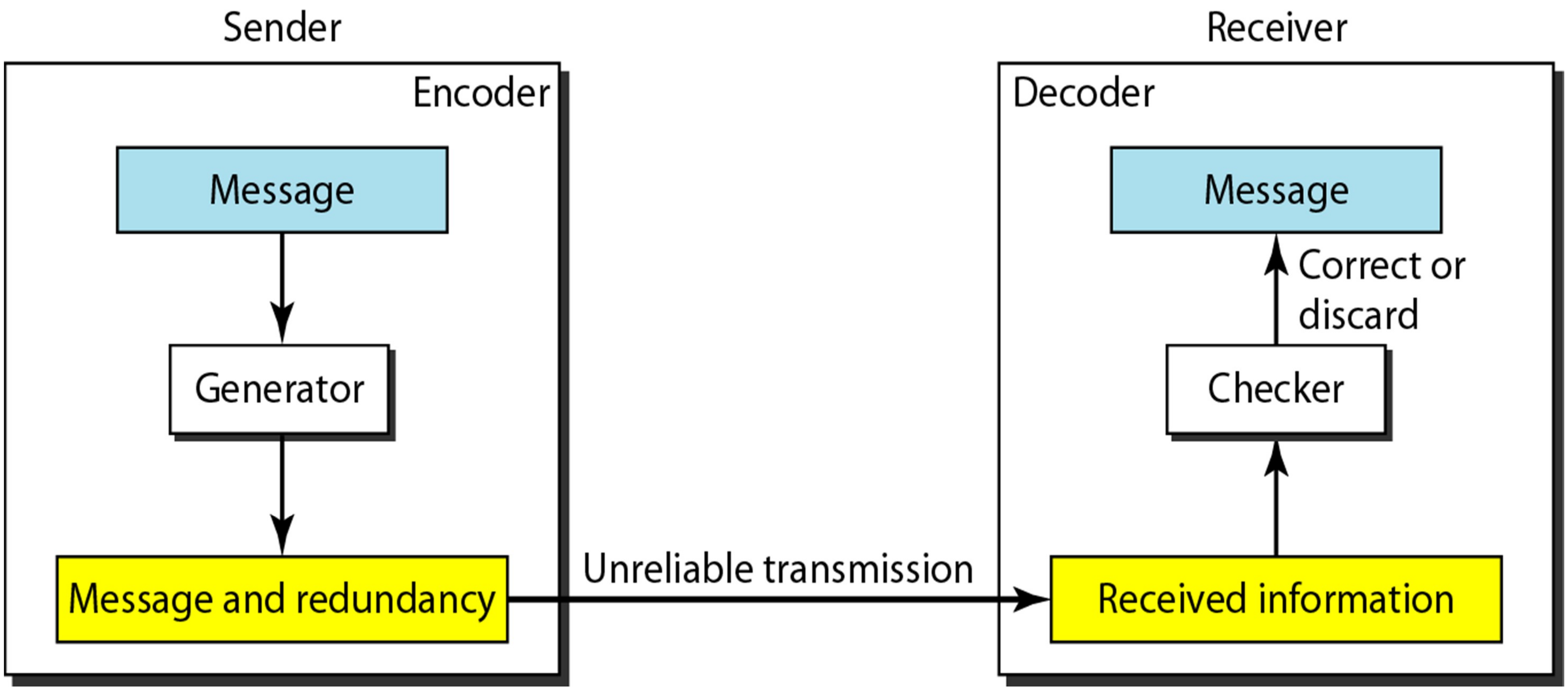
- Bit error



- Burst error



Error detection process



Error detection schemes

- Simple parity-check code
- Cyclic Redundancy Check (CRC)
- Checksum

Simple Parity-Check Code

- Extra bit added to make the total number of 1s in the codeword
  - Even → even parity
  - Odd → odd parity

dataword				codeword
10011100	+	0	=	100111000

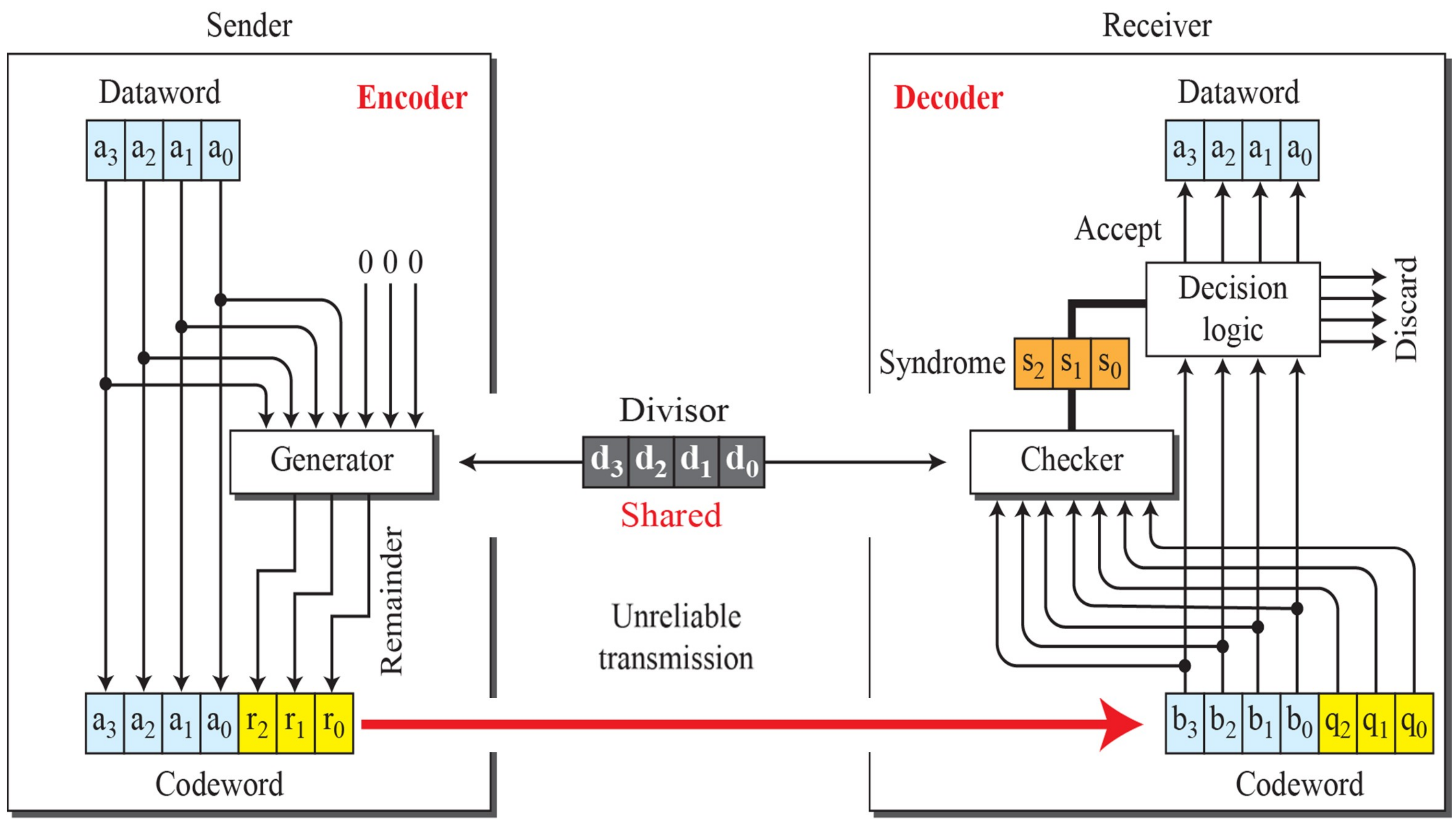
- Can detect an odd number of errors

Block coding

- Divide the message into k-bit blocks, called **datawords**.
- Add r redundant bits to each block. The resulting n-bit blocks ($n=k+r$) are called **codewords**.
- The code rate is $R=k/n$.

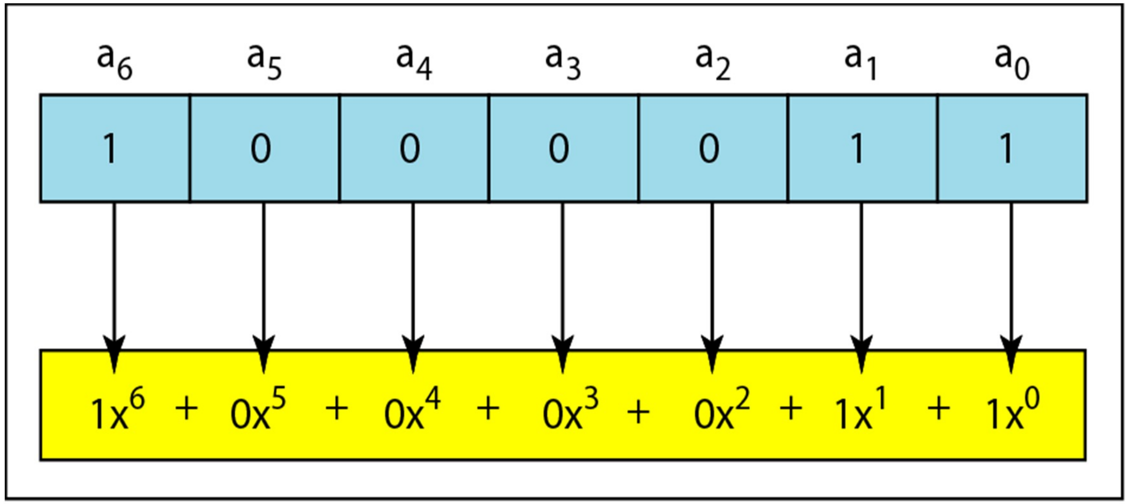
Cyclic Redundancy Check (CRC)

- Predefined shared *divisor* to calculate codeword

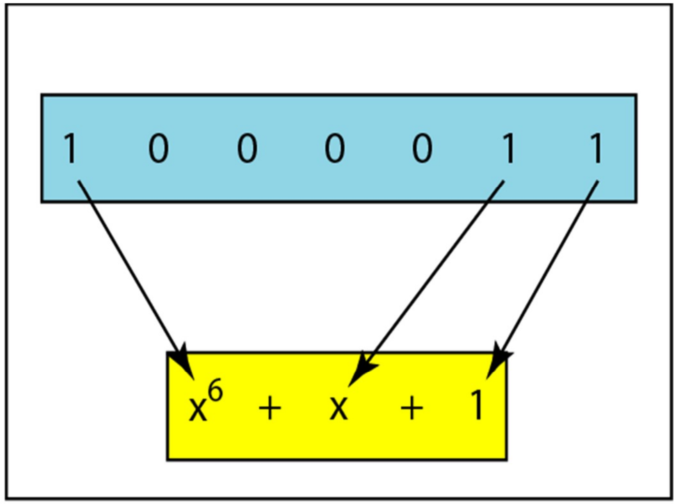


CRC: Polynomial representation

- The dataword of *k* bits is represented by a polynomial, *d(x)*.
- The degree of the polynomial is *k*-1.



a. Binary pattern and polynomial



b. Short form

CRC: The principle

- **Objective**: Send a dataword ***d(x)*** of k bits represented by a polynomial of degree ***k*-1**.
- **Given**: Generator polynomial ***g(x)*** of degree ***m***.
- **Find**: Remainder polynomial ***r(x)*** such that:

  $$c(x) = d(x) \cdot x^m + r(x)$$

  can be divided by g(x) without remainder.
- Codeword ***c(x)*** will then be sent to the receiver.
- ***r(x)*** has degree ***m*-1** or less, and CRC has ***m*** bits.

CRC: How it works

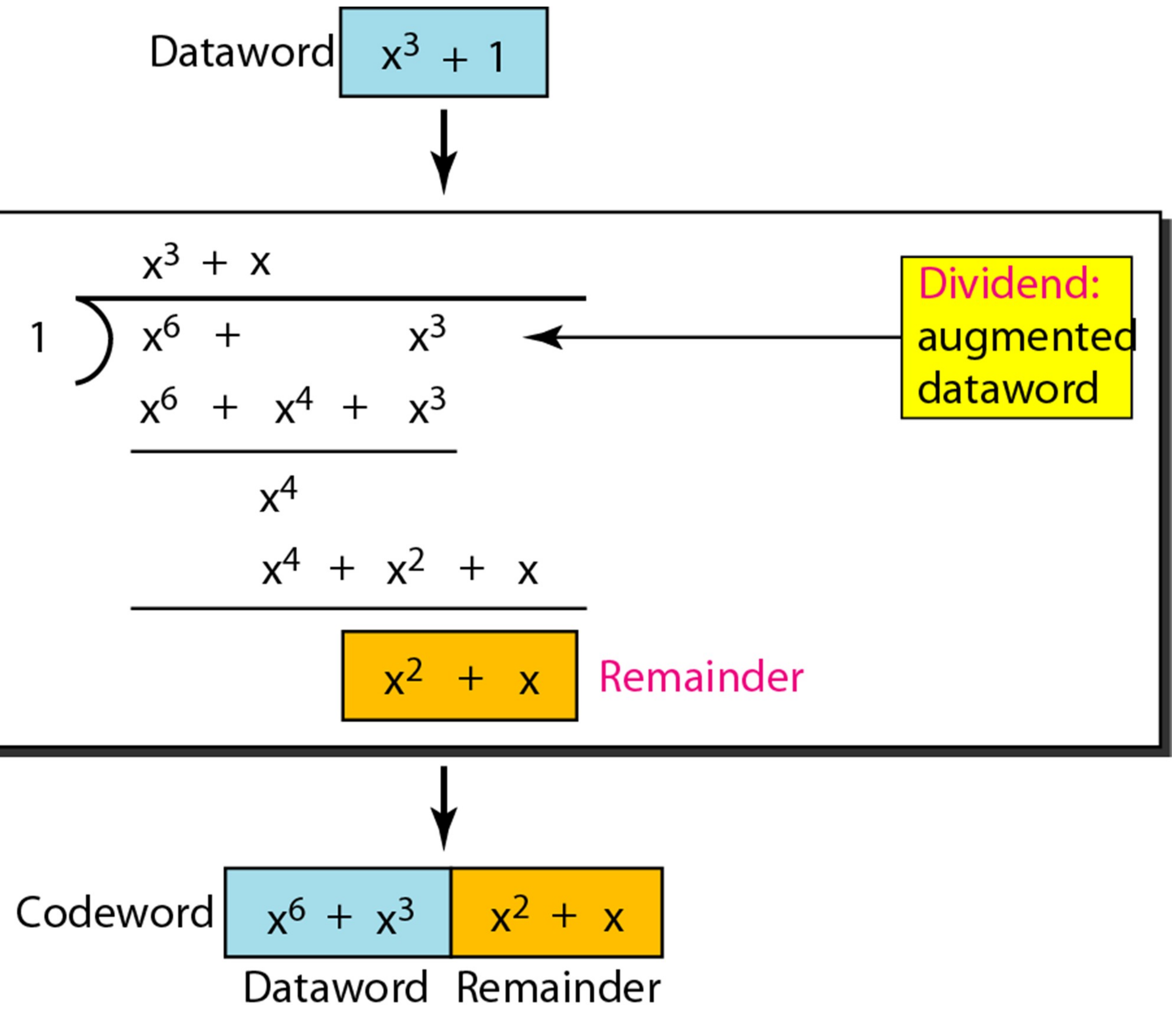
- Sender:
  1. Generate $b(x) = d(x) \cdot x^m$
  2. Divide $b(x)$ by $g(x)$ to find $r(x)$
  3. Send $c(x) = b(x) + r(x)$
- Receiver:
  1. Divide $c'(x) = c(x) + e(x)$ by $g(x)$
  2. Check remainder $r'(x)$ – if 0 data correct, $c(x) = c'(x)$
  3. Remove CRC bits from codeword to get dataword

Example: CRC derivation

- For dataword 1001, derive CRC using generator 1011.

- Data polynomial: $d(x) = x^3+1$
- Generator polynomial: $g(x) = x^3+x+1$
- Dividend: $b(x) = d(x) \cdot x^3 = x^6+x^3$
- Codeword polynomial: $c(x) = d(x) \cdot x^3 + r(x)$
- CRC polynomial: $r(x) = ?$

Example: CRC derivation



Error detection capabilities

- Single errors: $e(x)=x^i$ is not divisable by $g(x)$
- Double errors: $e(x)=x^j+x^i=x^i(x^{j-i}+1)$
  - Use primitive polynomial $p(x)$ with $deg=L$. Then if $n-1<2^L-1$ it is not divisable and all double errors will be detected
- If $x+1 \mid g(x)$ all odd error patterns will be detected
- In practice, set $g(x)=(x+1)\cdot p(x)$

Some standard CRC polynomials

Name	*Polynomial*	*Used in*
CRC-8	$x^8 + x^2 + x + 1$ 100000111	ATM header
CRC-10	$x^{10} + x^9 + x^5 + x^4 + x^2 + 1$ 11000110101	ATM AAL
CRC-16	$x^{16} + x^{12} + x^5 + 1$ 10001000000100001	HDLC
CRC-32	$x^{32} + x^{26} + x^{23} + x^{22} + x^{16} + x^{12} + x^{11} + x^{10} + x^8 + x^7 + x^5 + x^4 + x^2 + x + 1$ 100000100110000010001110110110111	LANs