NTNU

Kunnskap for en bedre verden

Wireless Transmission

Elissar Khloussy

The slides of this lecture are based on :

- Wireless Communication Networks and Systems, chapter 9, by C. Beard & W. Stallings
- Computer Networking, a top-down approach, chapter 6, by J. Kurose and K. Ross

September 2022

Wireless Channel Impairments

- Wireless channel impairments include noise, path loss, shadowing, fading, …
- Wireless communication systems should be designed to overcome those impairments
- Several techniques might be used:
    - Adaptive Modulation and Coding
    - Diversity techniques
    - Data encoding
    - …

Addressing Wireless Channel Impairments

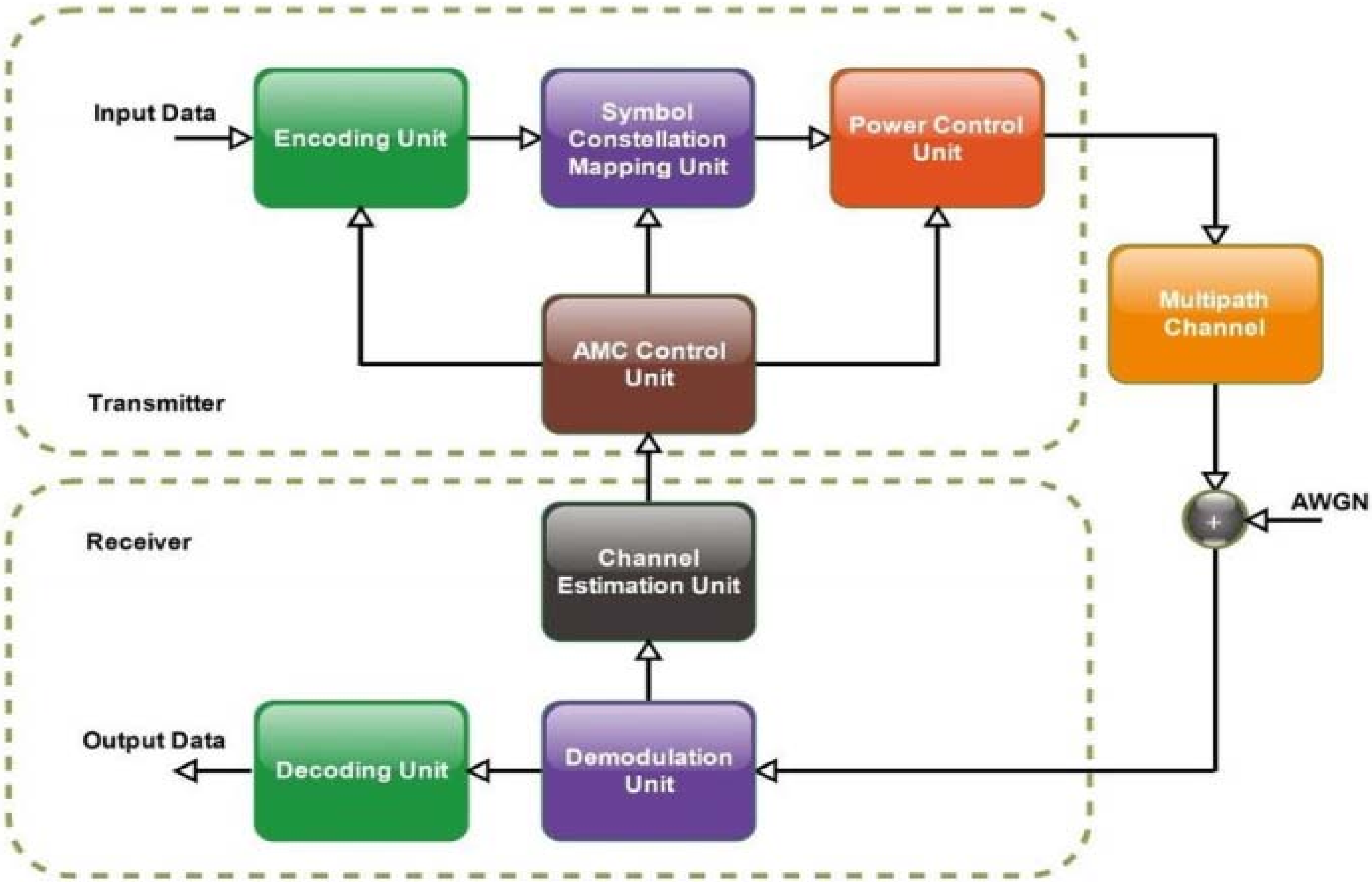
- Adaptive modulation and coding

- Diversity techniques
  - Multiple Input Multiple Output (MIMO) antennas

- Spread spectrum

- Error detection and correction

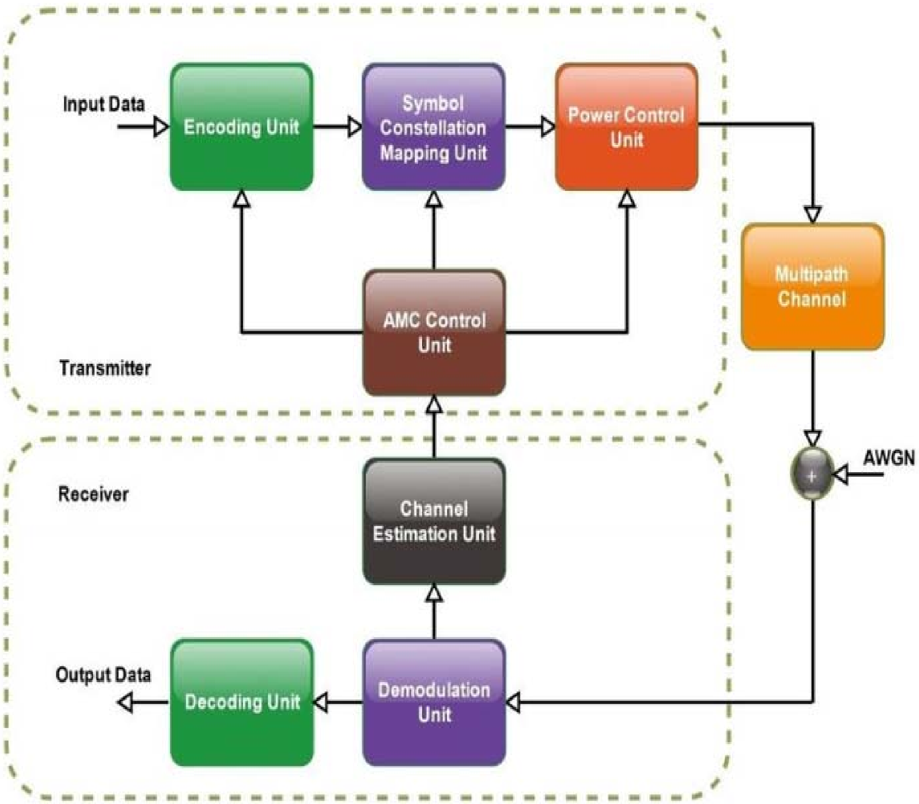
Adaptive Modulation and Coding (AMC)



AMC: Adapting the modulation, coding and other signal parameters to the conditions on the radio link

Adaptive Modulation and Coding (AMC)

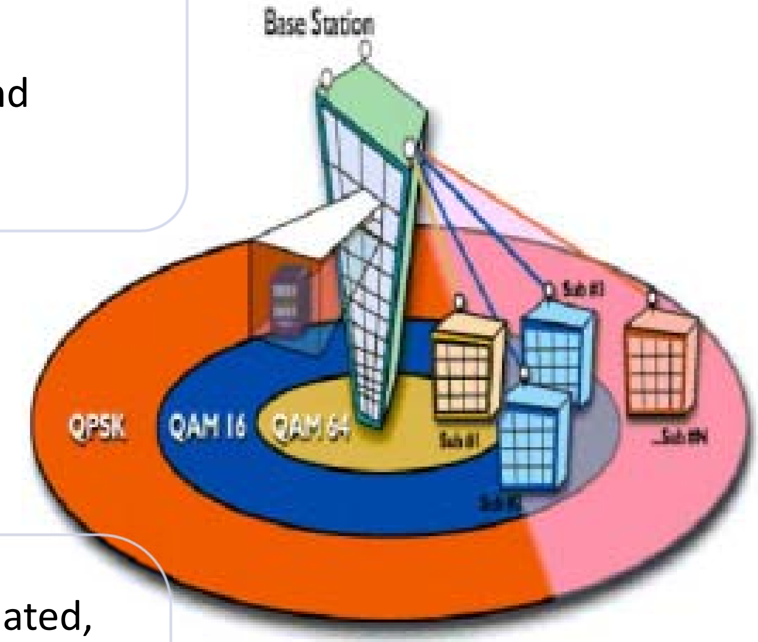
- This process is dynamic
- Requires some channel state information at the transmitter
- Transmitter and receiver coordinate the changes
- Many radio communication systems use AMC
  - Based on the channel conditions, they adapt the modulation scheme to obtain the highest data rate for the given conditions
  - Example: When SNR decreases, they revert to a lower modulation scheme to make the link more reliable with fewer data errors and re-sends



Adaptive Modulation and Coding (AMC)

The goal of AMC is the efficient utilization of resources

- Enhanced throughput
- Allows the system to overcome fading and interference



Challenges:

- if the channel changes faster than estimated, AMC will perform poorly
- Error in the channel estimate will result in selecting the wrong data rate, transmitting at a higher power or lower power than needed
- High complexity

As the range increases, lower modulation is selected. For the closer users, higher order modulations might be chosen for increased throughput

Addressing Wireless Channel Impairments

- Adaptive modulation and coding
- Diversity techniques
  - Multiple Input Multiple Output (MIMO) antennas
- Spread spectrum
- Error detection and correction

Diversity techniques

Individual channels experience independent fading events!

- The idea is to send multiple copies of the same signal with different characteristics
    - Space diversity – using multiple antennas
    - Frequency diversity – techniques where the signal is spread out over a larger frequency bandwidth or carried on multiple frequency carriers
    - Time diversity – techniques aimed at spreading the data out over time (using different time slots)
- Diversity techniques are used to improve the system performance over a fading channel

Multiple Input Multiple Output (MIMO) Antennas

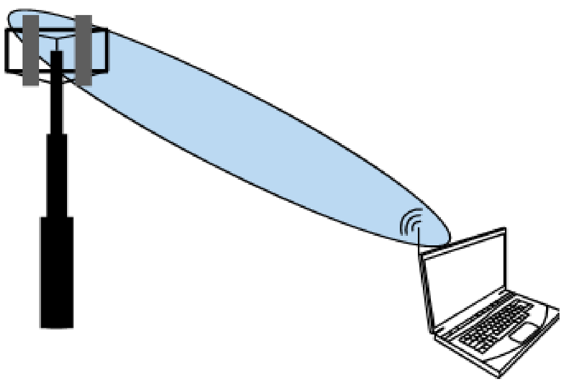
- The transmitter and/or receiver have multiple antennas (antenna array)

- Allows to send and receive more than one signal on different transmit and receive antennas

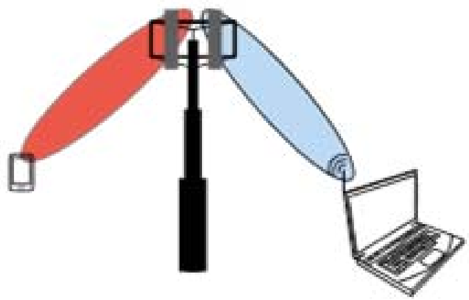
- Increased data rates and transmission range without additional transmit power or bandwidth
- Examples
  - IEEE 802.11n, LTE, …

MIMO - Functions

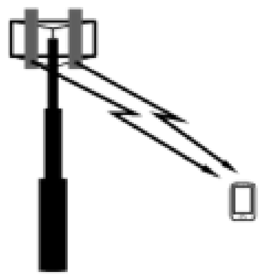
- Spatial diversity: Send or receive redundant streams of information in parallel along multiple spatial paths
  - Increases reliability and range (unlikely that all paths will be degraded simultaneously)
- Spatial multiplexing: split high-rate signal into multiple lower rate streams and transmit over different antennas
- Beaforming: emit the same signal from all antennas to **maximize signal power** at receiver antenna
- Multi-user MIMO (MU-MIMO): directional antenna beams established to multiple users simultaneously



Beamforming



Spatial multiplexing



Diversity

MIMO -Summary

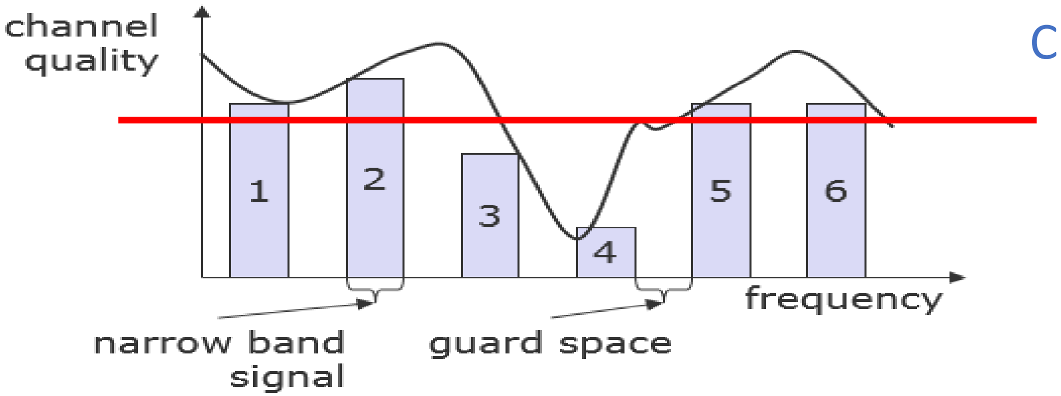
- Spatial diversity and multiplexing are the building blocks of MIMO systems
- Spatial diversity is a technique in MIMO that reduces signal fading by sending multiple copies of the same radio signal through multiple antennas
- Spatial multiplexing is a technique in MIMO that boosts data rates by sending the data payload in separate streams through spatially separated antennas.
- The MIMO antenna technology has been a key part of mobile communications since the 3G era.
- The more recent mobile networks, including 4G LTE and 5G networks rely heavily on the enhanced variants of MIMO technology, including Massive MIMO

Addressing Wireless Channel Impairments

- Adaptive modulation and coding
- Diversity techniques
    - Multiple Input Multiple Output (MIMO) antennas
- Spread spectrum
- Error detection and correction

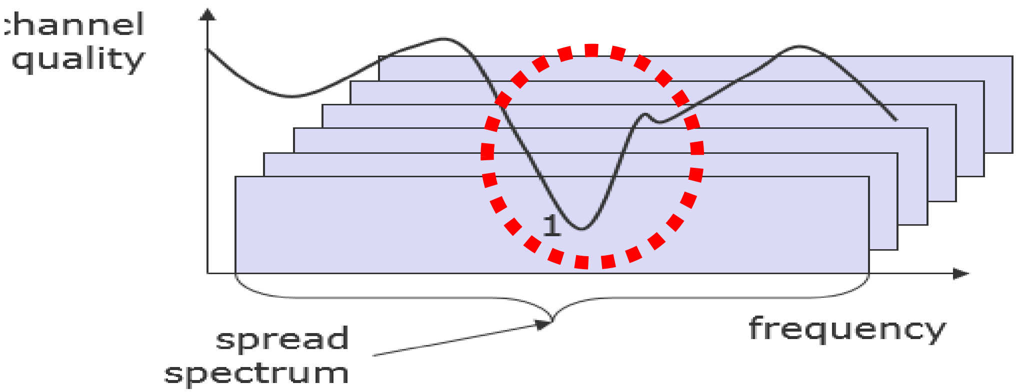
Spread Spectrum

- Narrowband signal → can be wiped up by frequency-dependent fading
- Approach: spread the narrowband signals into broadband signals using a special code



Channel quality changes over time!

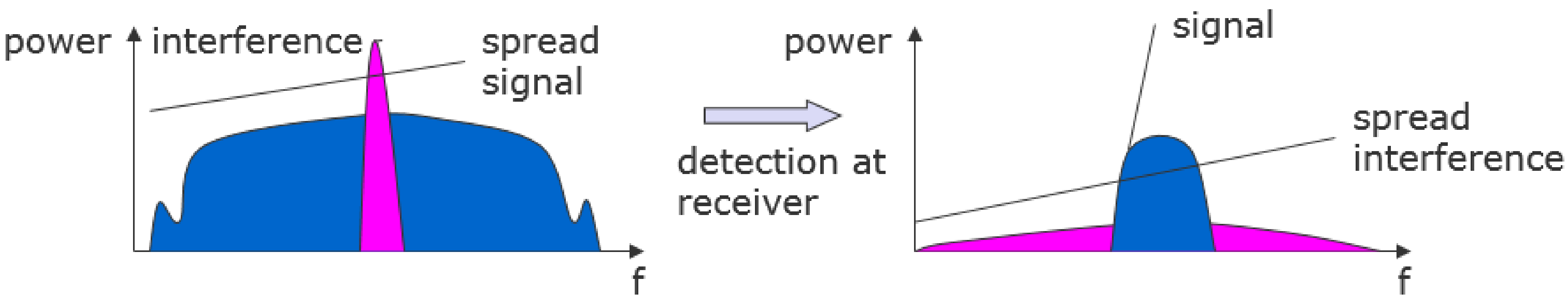
narrowband channels



spread spectrum channels

Spread Spectrum (ctn’d)

- The sender spreads the narrowband signal
- If the spread signal has enough energy despite the interference, then the signal still can be reconstructed
- The receiver despreads the received signal
- Result: spread interference and narrowband signal



Spread Spectrum (ctn’d)

Advantages:

- Resistant to narrowband interference
- Signals are hard to detect, better security
- Efficient multiplexing: coexistence of several signals without the need of dynamic coordination

Disadvantages:

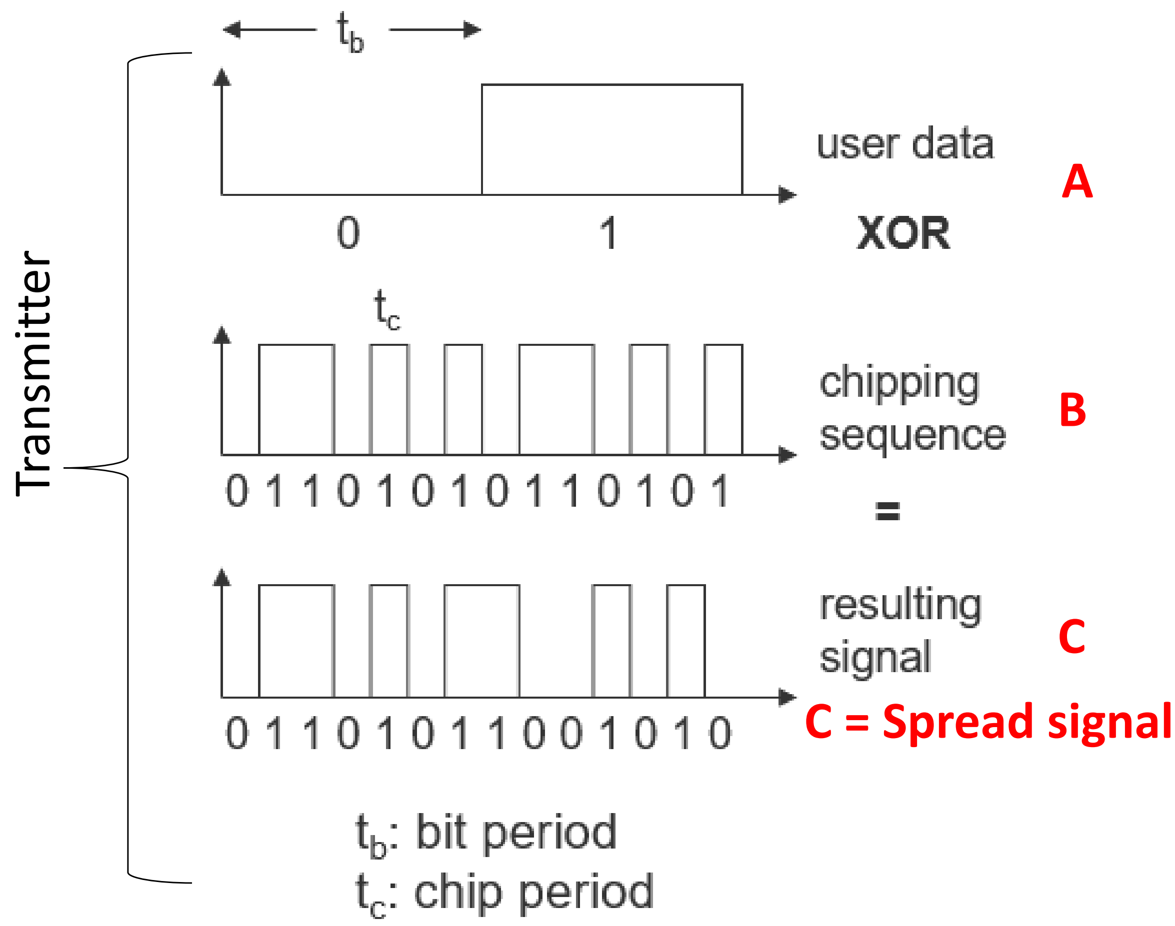
- Increased complexity at the receiver
- Requires wide frequency band to spread the signal

How is the spreading done?

- Direct Sequence (DSSS)
- Frequency Hopping (FHSS)

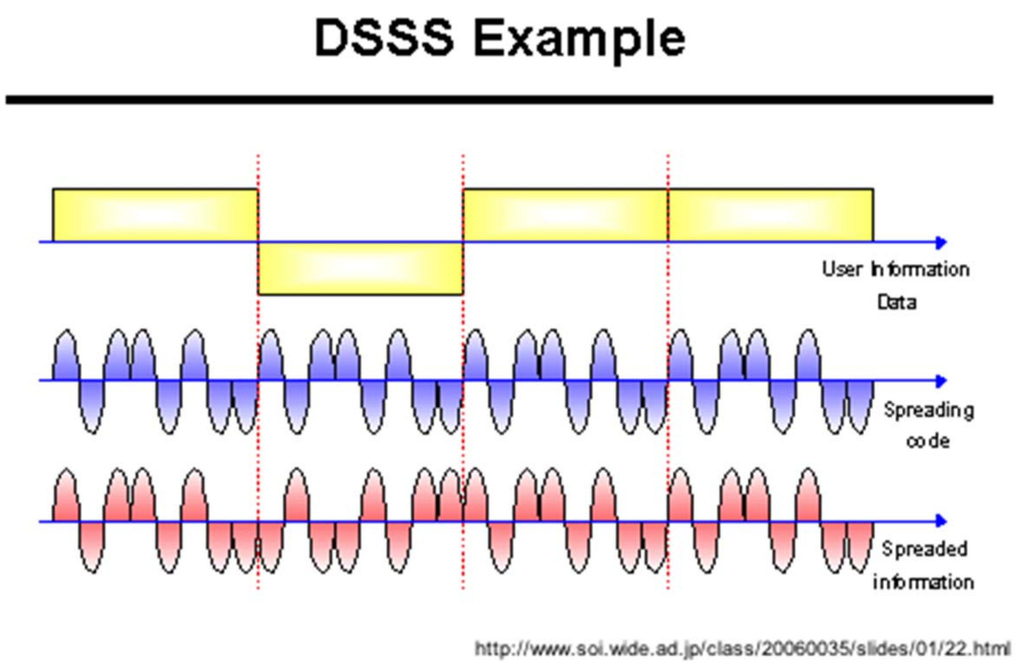
Direct Sequence Spread Spectrum (DSSS)

- Uses pseudo-random code (chipping sequence)
- XOR the signal with the chipping sequence

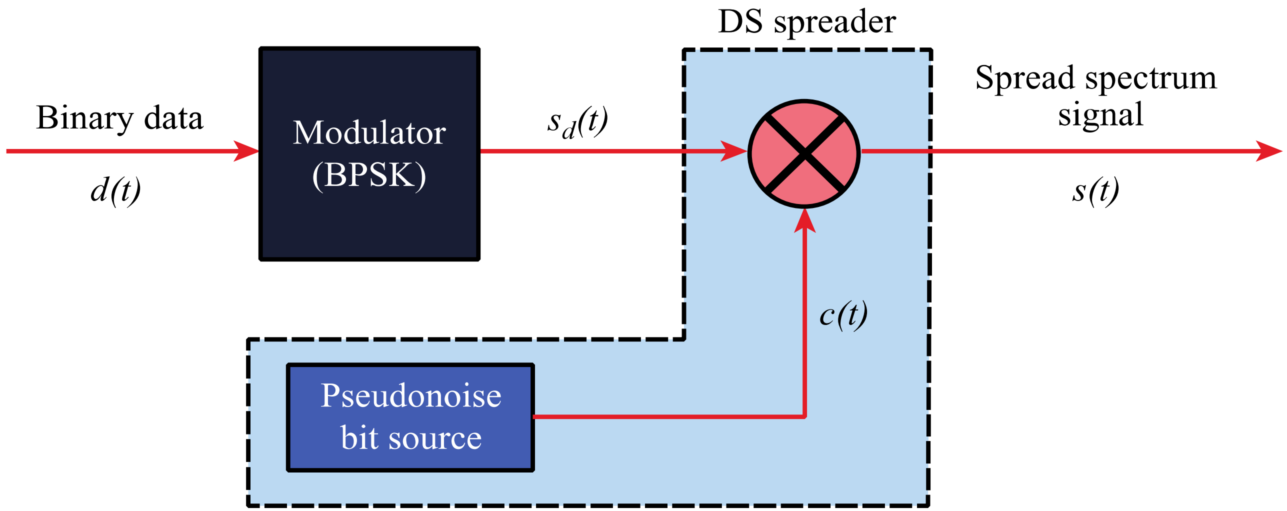


Direct Sequence Spread Spectrum (DSSS)

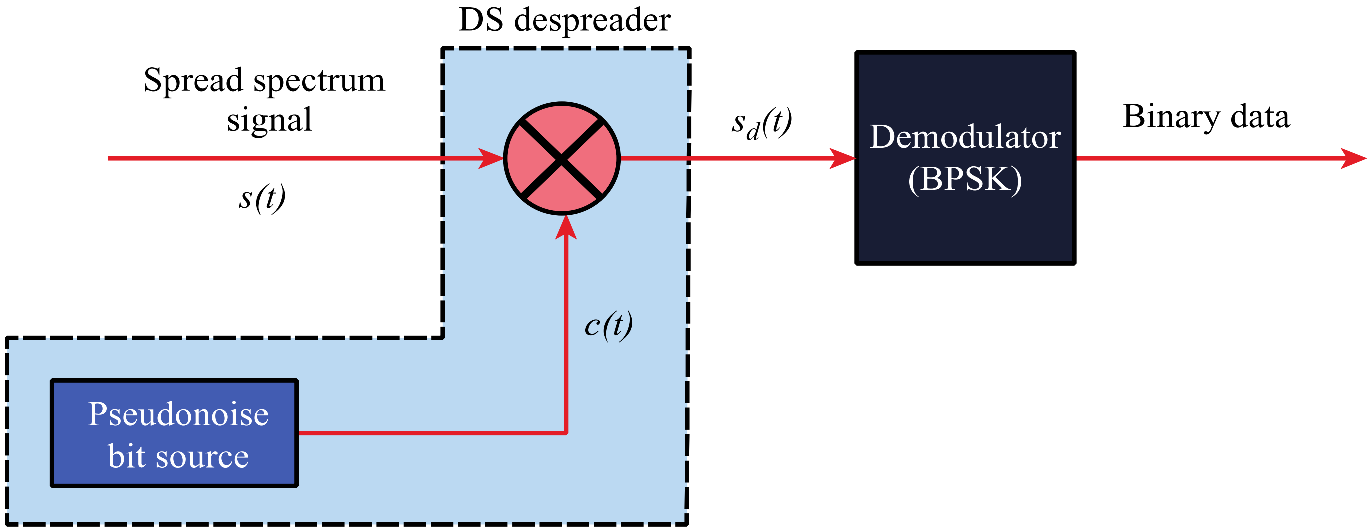
- Advantages:
  - Reduces the effect of frequency-selective fading
  - Cellular networks:
    - Possible to reuse same frequency but with different chipping sequences
- Disadvantages:
  - Precise power control is required
    - signals that arrive simultaneously should reach the receiver at about the same power level



DSSS implementation



(a) Transmitter

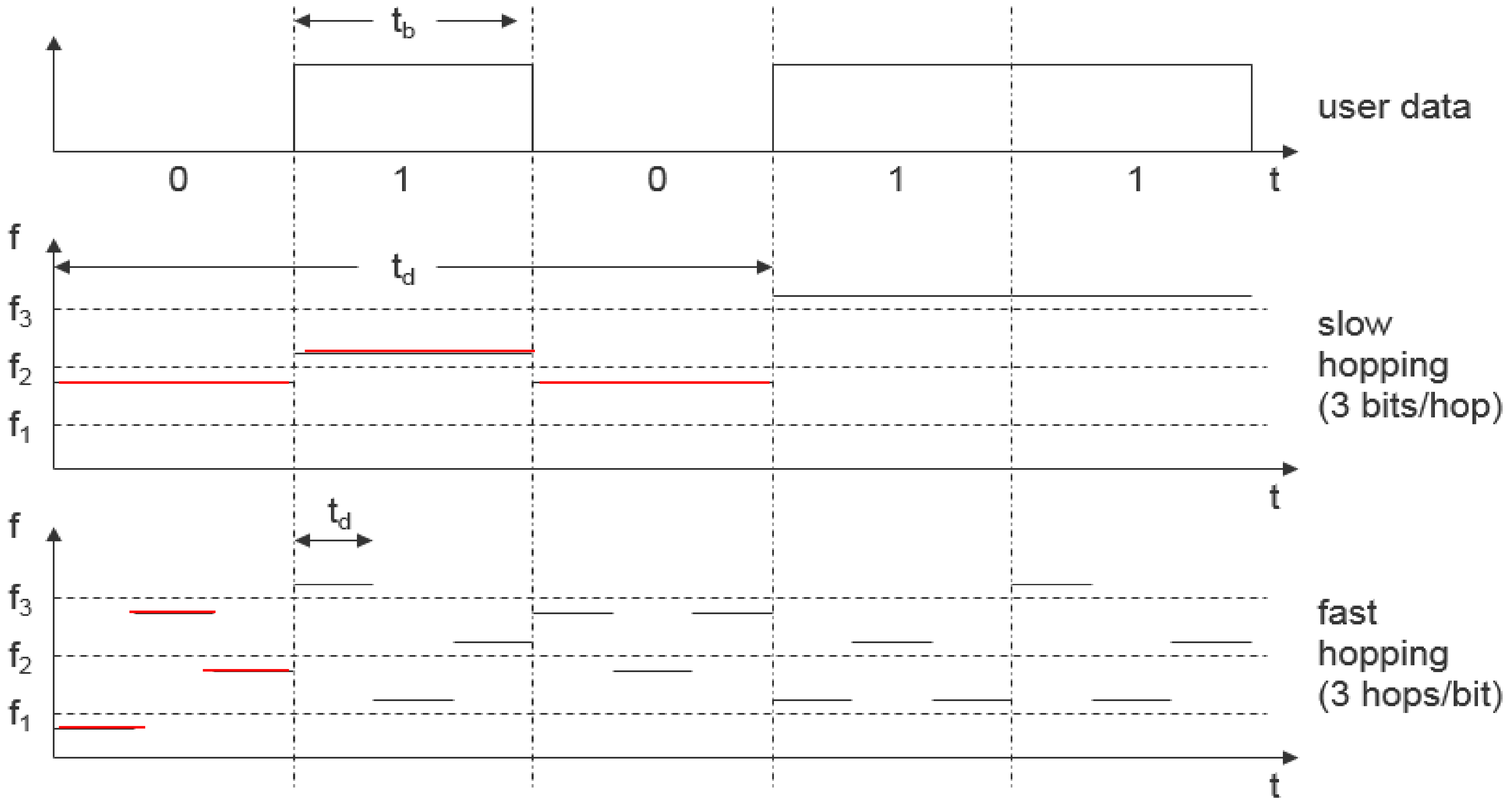


(b) Receiver

Frequency Hopping Spread Spectrum (FHSS)

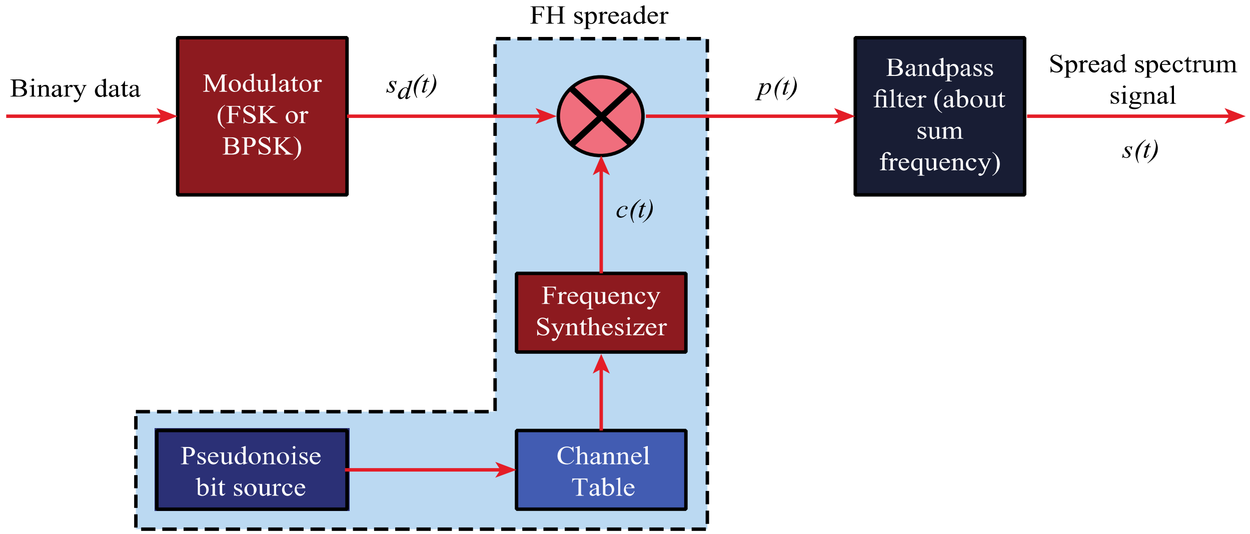
- Signal is broadcast over seemingly random series of radio frequencies
  - sequence of frequency changes determined via pseudo random number sequence
- Signal hops from frequency to frequency at fixed intervals
- **Slow frequency hopping**: multiple symbols are transmitted in one frequency hop
- **Fast frequency hopping**: multiple hops are required to transmit one symbol.
- Advantages
  - simple implementation
  - Limits frequency selective fading and interference to short period
  - uses only small portion of spectrum at any time
- Disadvantages
  - Less robust than DSSS
  - Easier to detect

Frequency Hopping Spread Spectrum (FHSS)

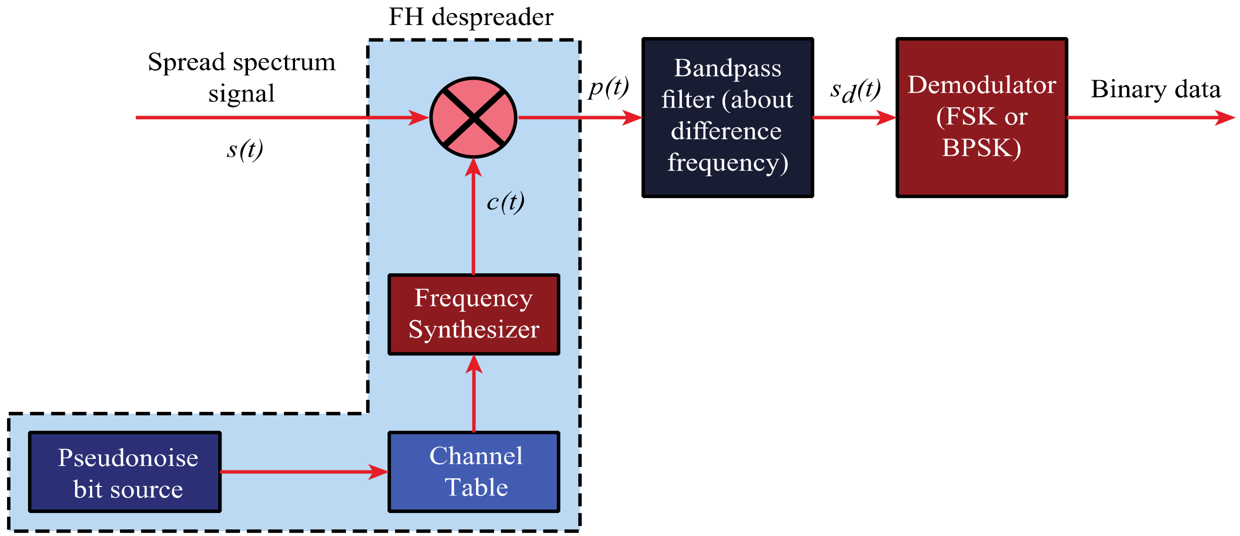


t_b: bit period t_d: dwell time

Frequency Hopping Spread Spectrum (FHSS)



(a) Transmitter



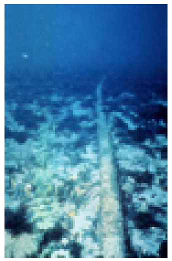
(b) Receiver

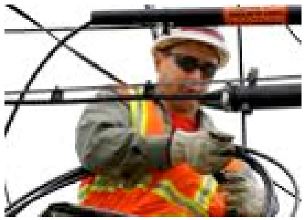
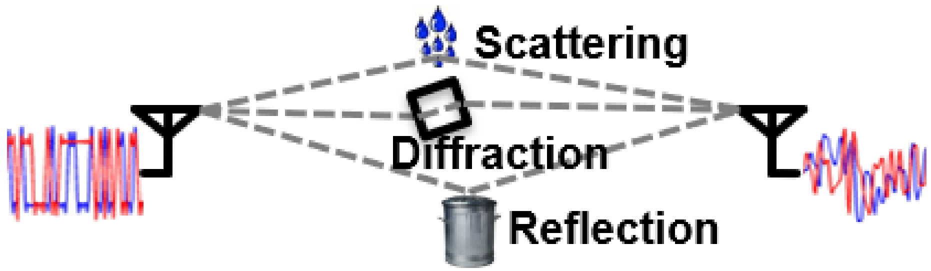
Addressing Wireless Channel Impairments

- Adaptive modulation and coding
- Diversity techniques
    - Multiple Input Multiple Output (MIMO) antennas
- Spread spectrum
- Error detection and correction

Error Correction and Detection

- Links in a network go through hostile environments
  - Both wired, and wireless



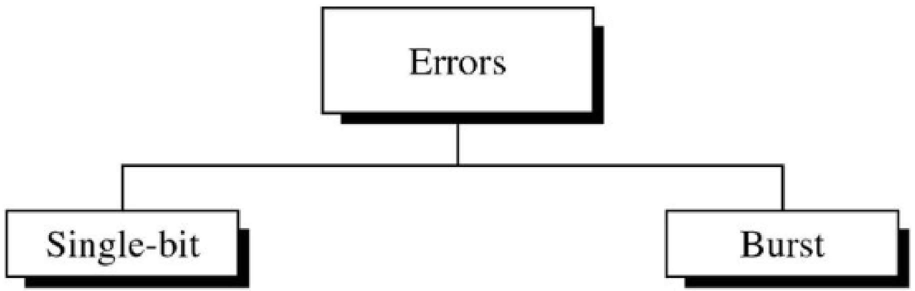



- Consequently, errors will occur on links
  - Some bits might be altered or lost due to noise, attenuation, delay distortion, …
- Errors must be detected and corrected in order to have a reliable communication
- How can we detect and correct these errors?

Error Correction and Detection

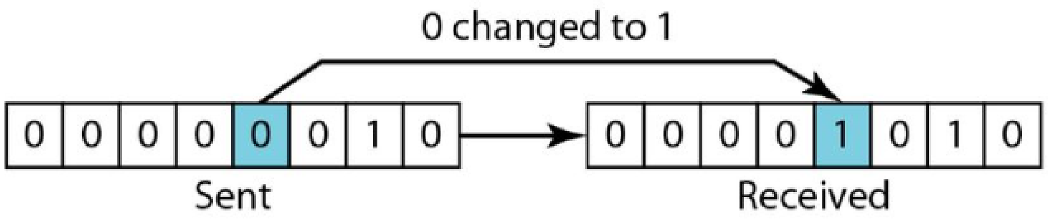
- Error correction and detection are implemented either at the data link layer or the transport layer

- Two types of errors:
  - single-bit error
  - Burst error



Single-bit error

- Only 1 bit in the data unit has changed
- Can happen in parallel transmission where all the data bits are transmitted using separate wires



Example:

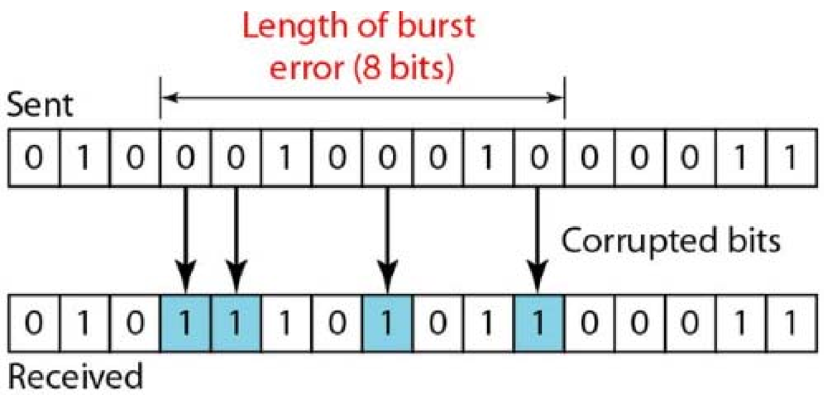
- ➢ If data rate is 1Mbps then each bit lasts only 1/1,000,000 sec. or 1 μs.
- ➢ For a single-bit error to occur, the noise must have a duration of only 1 μs, which is very rare in serial transmission

Burst Error

- 2 or more bits have changed (not necessarily consecutive)
- The duration of noise is normally longer than the duration of 1 bit, therefore several bits might be affected.
- the length of the burst is measured from the first corrupted bit to the last corrupted bit. Some bits in between may not have been corrupted
- Number of corrupted bits depends on the **transmission rate** and the **duration of noise**

Example

- If data is sent at rate = 1Kbps then a noise of 1/100 sec can affect 10 bits (1/100*1000)
- If same data is sent at rate = 1Mbps then a noise of 1/100 sec can affect 10,000 bits ($1/100*10^6$)



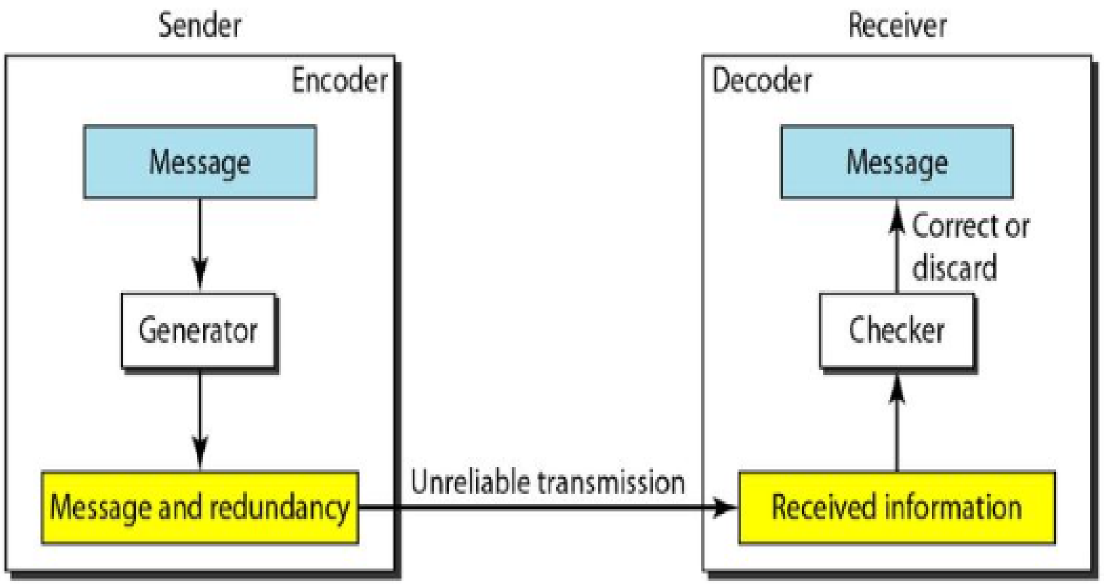
A- Error Detection

What is error detection?

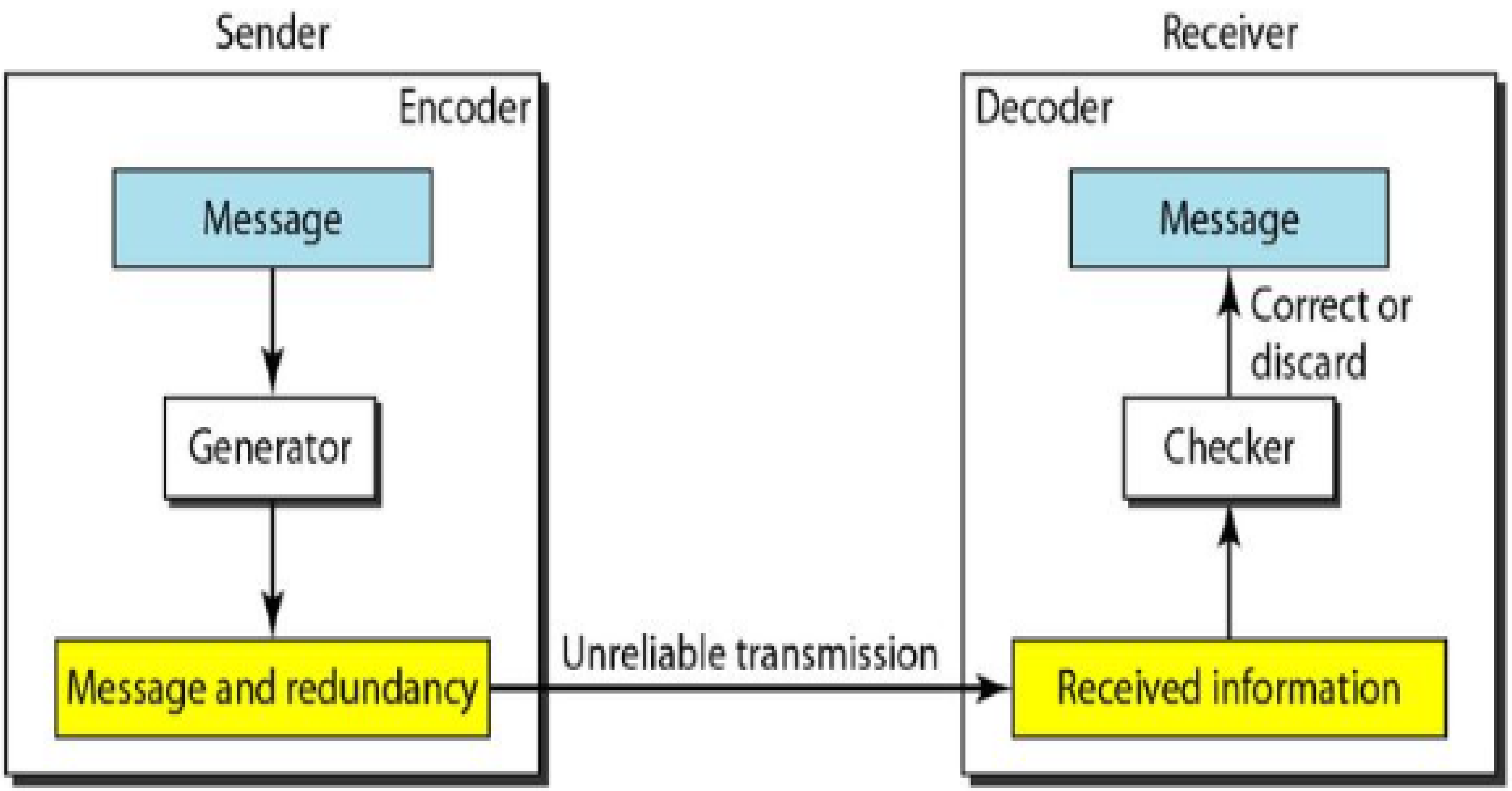
- Error detection is the decision of whether the received data is correct or not <u>without knowing the original message</u>

How?

- By using redundancy:
  - adding extra bits (redundant bits)
  - Goal: detect error at the destination

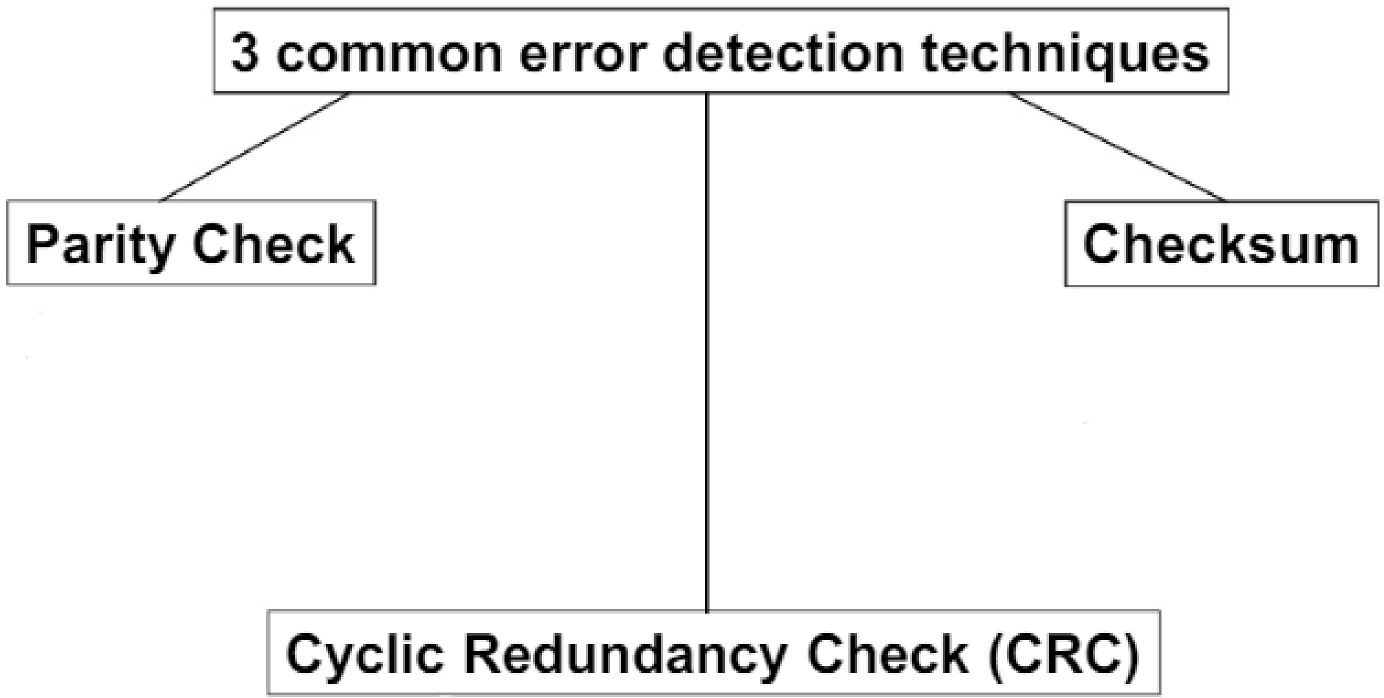


A- Error Detection

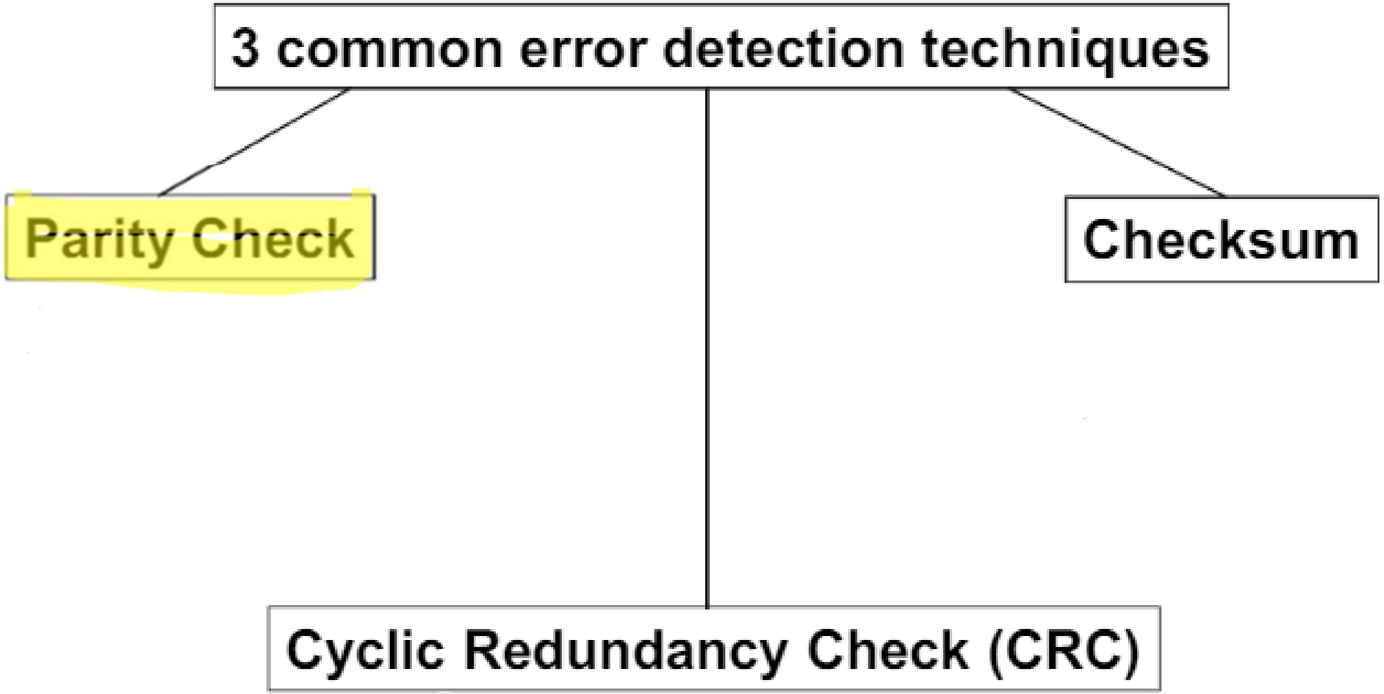


Checking function at the receiver side: calculates the redundant bits and compares them to the received ones to decide whether the data has been corrupted or not

Techniques for error detection

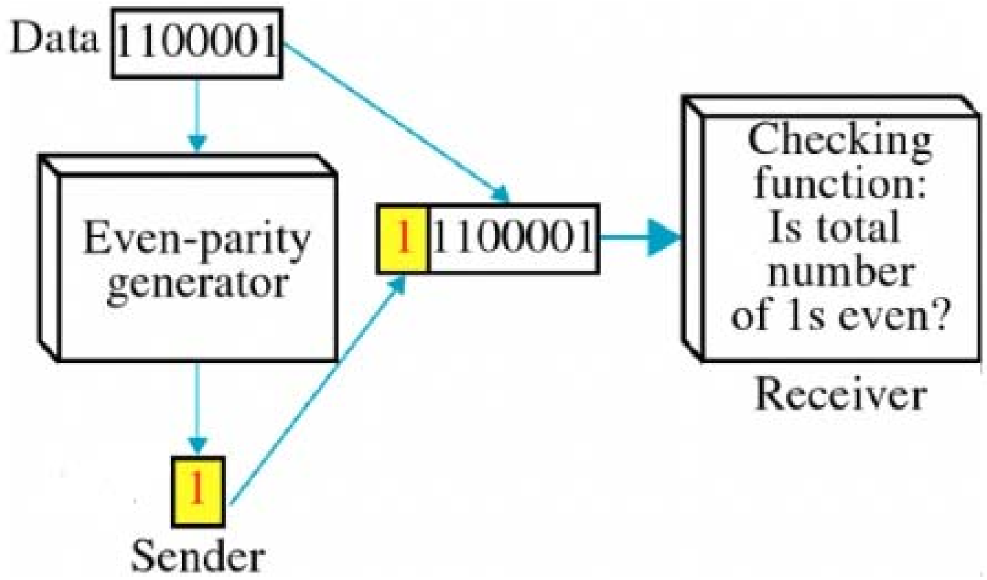


Techniques for error detection



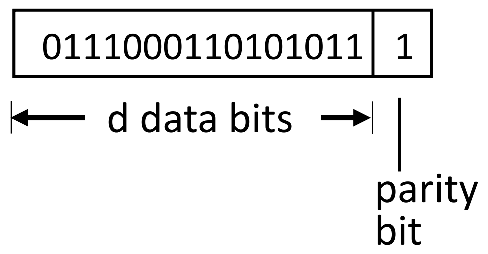
Parity Checks

- Parity bit(s) generated at the sender
- The checking function at the receiver decides whether to accept or reject the received data



Parity Checks – Single bit parity

Even parity: set parity bit so there is an even number of 1's



single bit parity

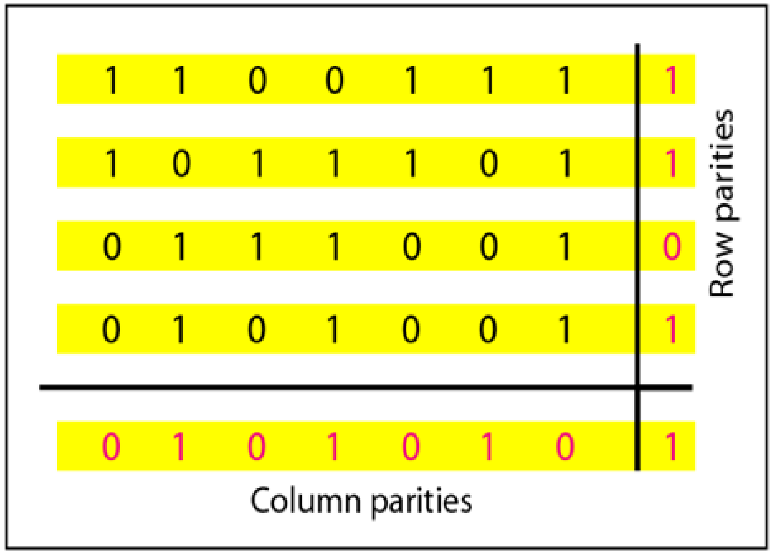
Performance:

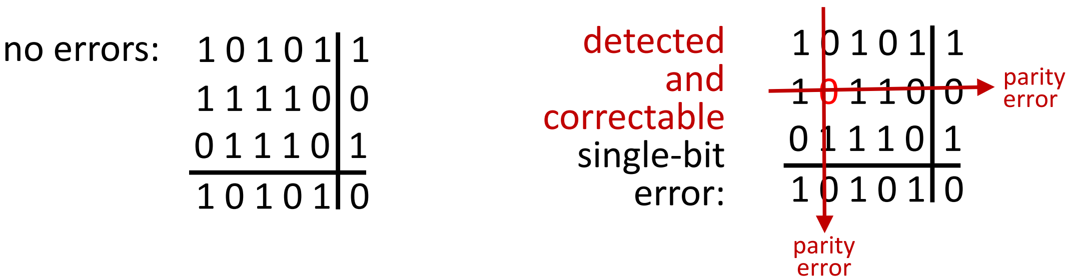
- Detects single-bit error
- Detects burst errors if the number of errors is odd

Parity checks – Two-dimensional bit parity

- detect and correct single bit errors

- Performance
  - If two bits in one data unit are damaged and two bits in the exact same position in another data are corrupted, then the errors will not be detected

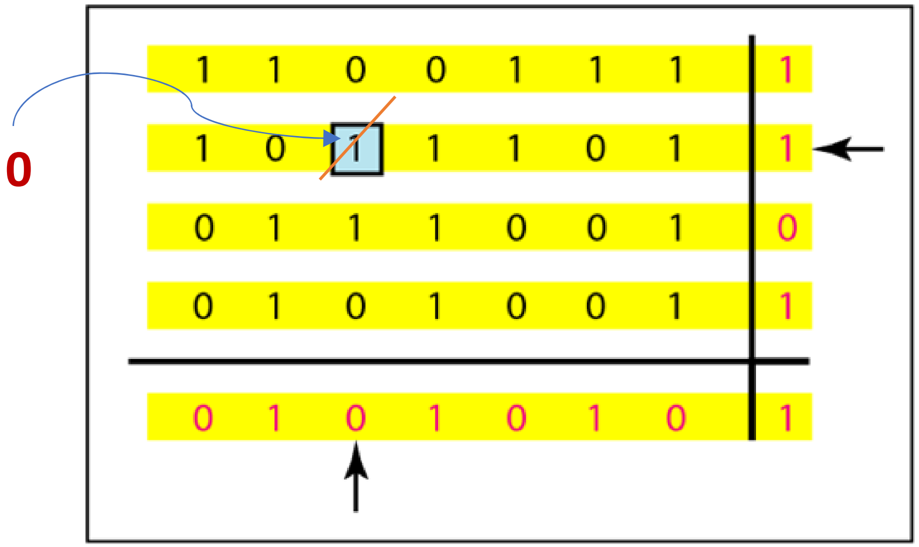




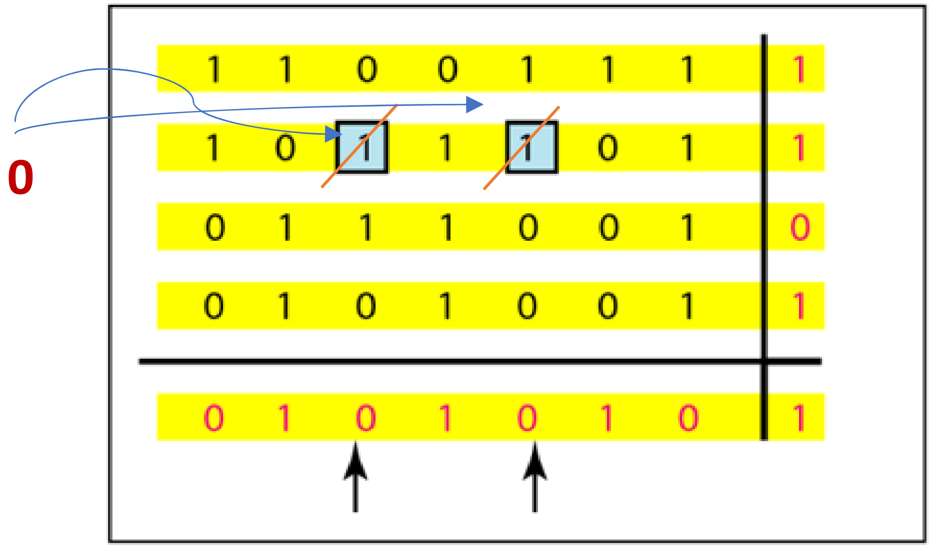
Parity Checks

Can these errors be detected and/or corrected?

Case A - 1 error, 2 parities affected



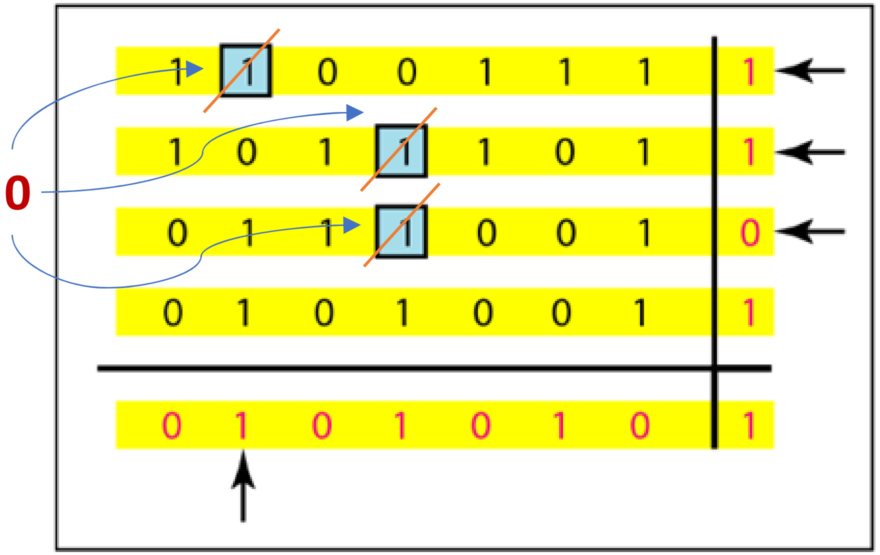
Case B - 2 errors, 2 parities affected



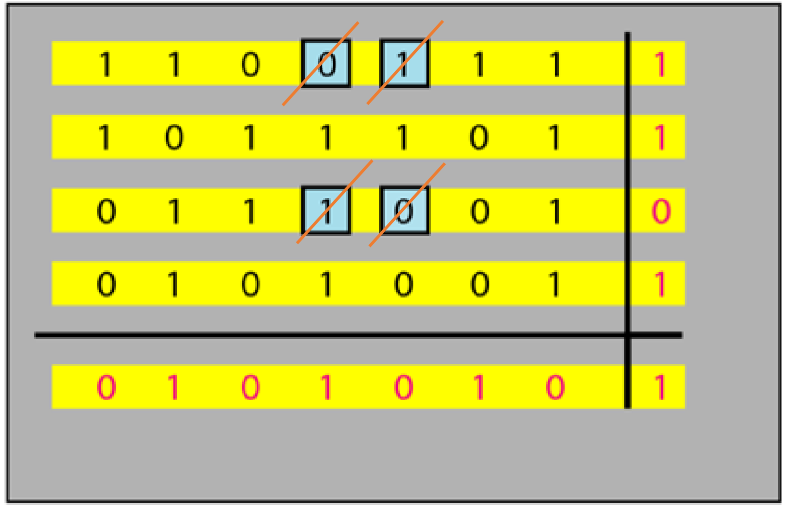
Parity Checks

Can these errors be detected and/or corrected?

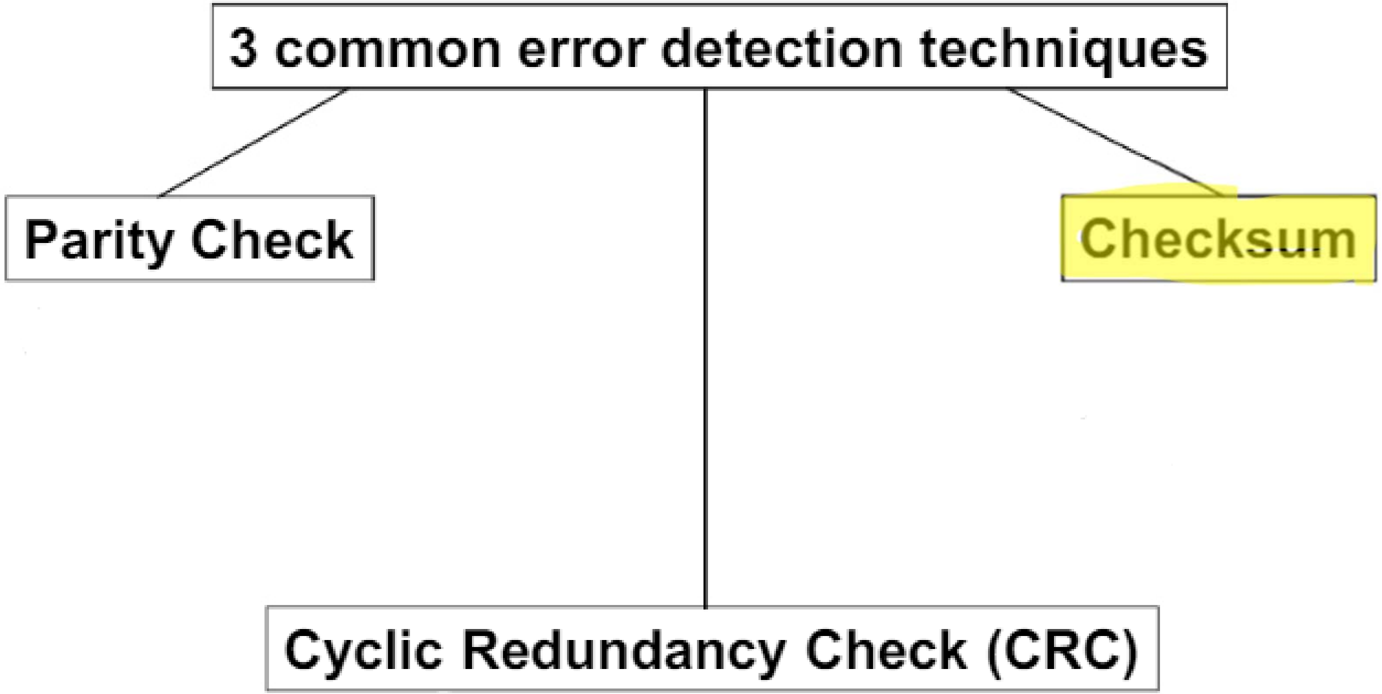
Case C - 3 errors, 4 parities affected



Case D - 4 errors, 0 parity affected



Techniques for error detection

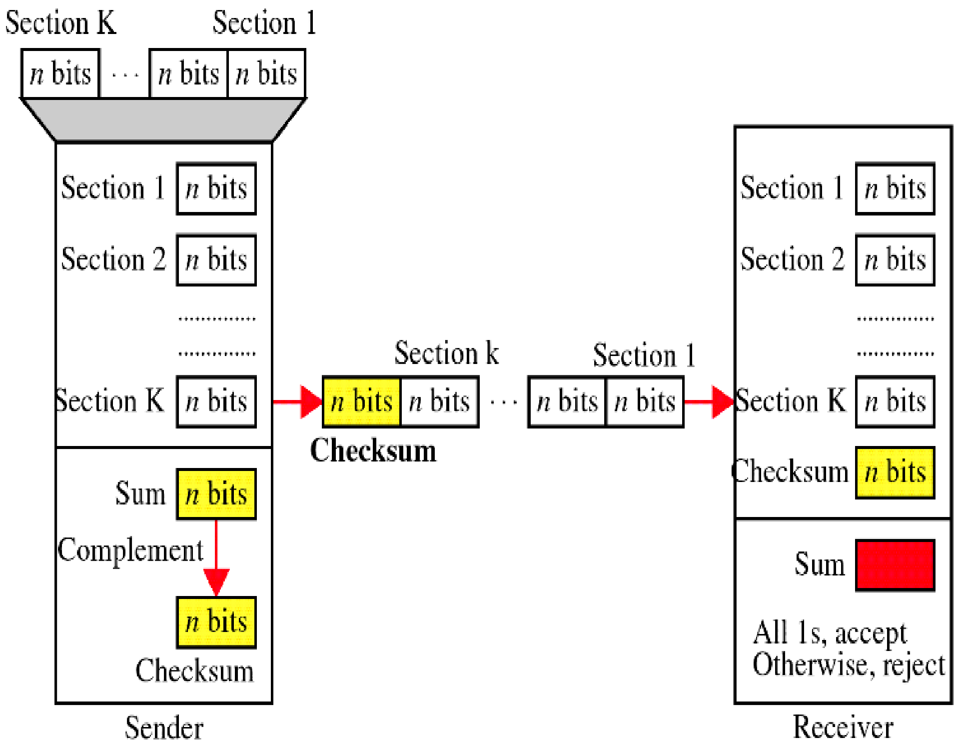


Checksum

- Checksum= addition of the 1's complement sum of a data segment content

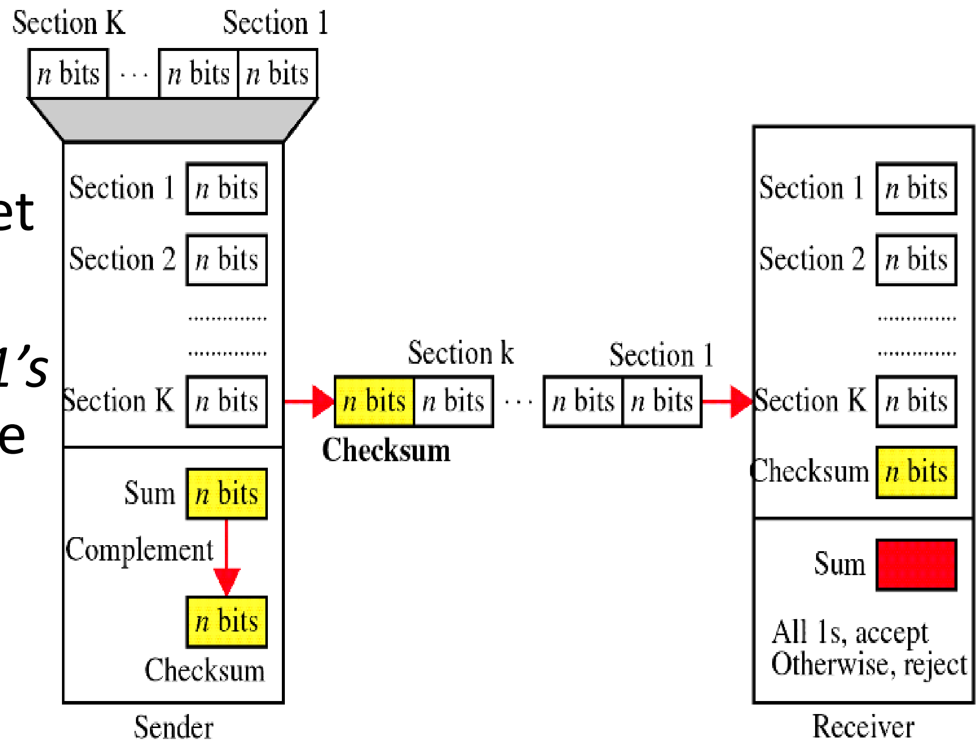
- The sender : Checksum creation
  - calculates the checksum of the data and adds it to the checksum field

- The receiver: Checksum validation
  - computes the checksum of the received segment
  - If computed checksum equals checksum field value → Data accepted
  - Otherwise, the data is rejected



Checksum Generation

1. The data is divided into *k* sections, each of *n* bits
2. All sections are added together to get the sum
3. The sum is complemented *(change 1's to 0's and 0's to 1's)* and becomes the checksum
4. The checksum is sent with the data (after section K)



Checksum Example

- Data unit to be transmitted:

  10011001111000100010010010000100

- 8-bit checksum is used
  - Data is divided into 8-bits segments

10011001	11100010	00100100	10000100

- All segments are added, then the 1’s complement is found

Checksum Example (Ctn'd)

- 10011001 + 11100010 + 00100100 + 10000100
- N.B. Extra bits are wrapped around and added to the sum
- The data alongside with the checksum is sent to the receiver

11011010	10011001	11100010	00100100	10000100

```
Carry   1 1 1 1 1
        1 0 0 1 1 0 0 1
        1 1 1 0 0 0 1 0
        0 0 1 0 0 1 0 0
        1 0 0 0 0 1 0 0
      ---------------------
    1 0 0 0 1 0 0 0 1 1
                      1 0
      ---------------------
        0 0 1 0 0 1 0 1
1's complement 1 1 0 1 1 0 1 0
```

Checksum Example (Ctn’d)

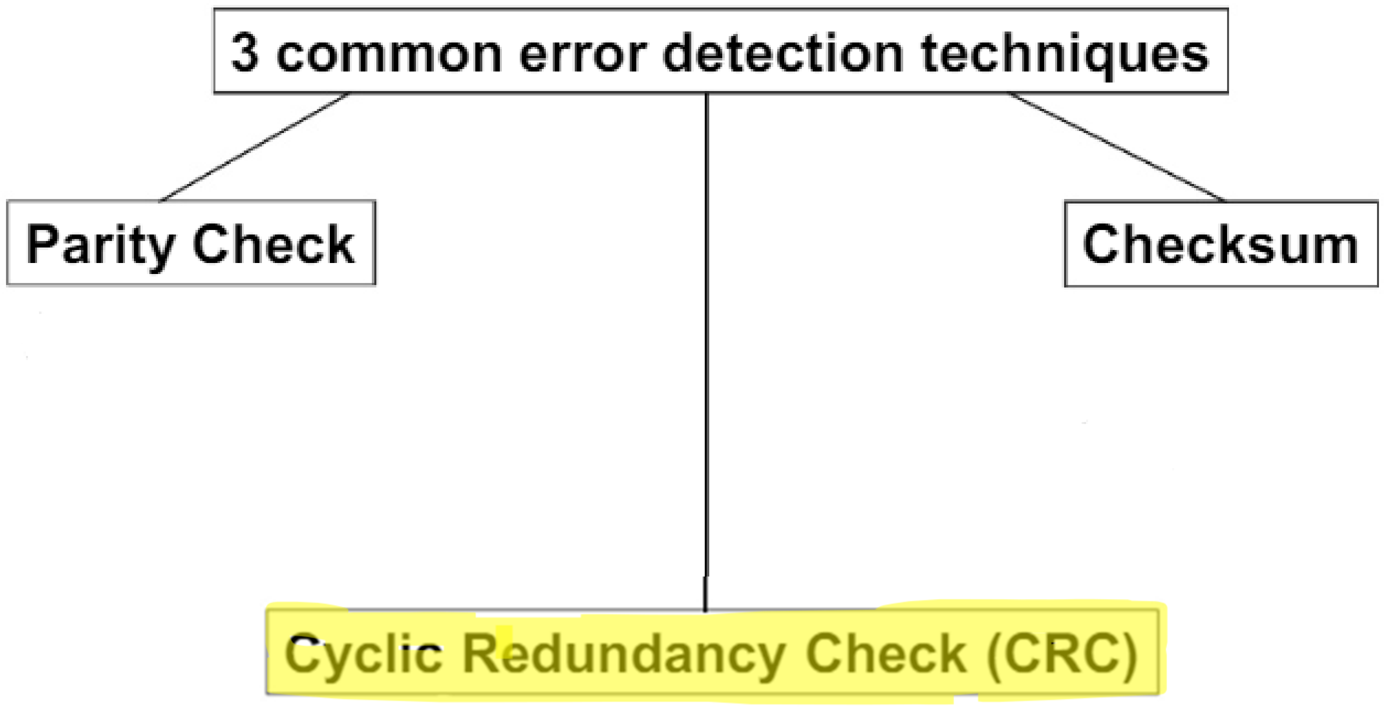
- At the receiver side: The sum of data and checksum is performed.
- If all 1’s then data is accepted

```
Carry     1 1 1 1 1 1
          1 0 0 1 1 0 0 1
          1 1 1 0 0 0 1 0
          0 0 1 0 0 1 0 0
          1 0 0 0 0 1 0 0
          1 1 0 1 1 0 1 0
        -------------------
        1 1 1 1 1 1 0 1
                      1 0
        -------------------
          1 1 1 1 1 1 1 1
```

Checksum performance

- Detects errors when the number of affected bits is odd
- Detects most errors if the number of affected bits is even
  - If bits in same position but with opposite values are changed, then the sum will not change and the receiver will not detect the errors

Techniques for error detection

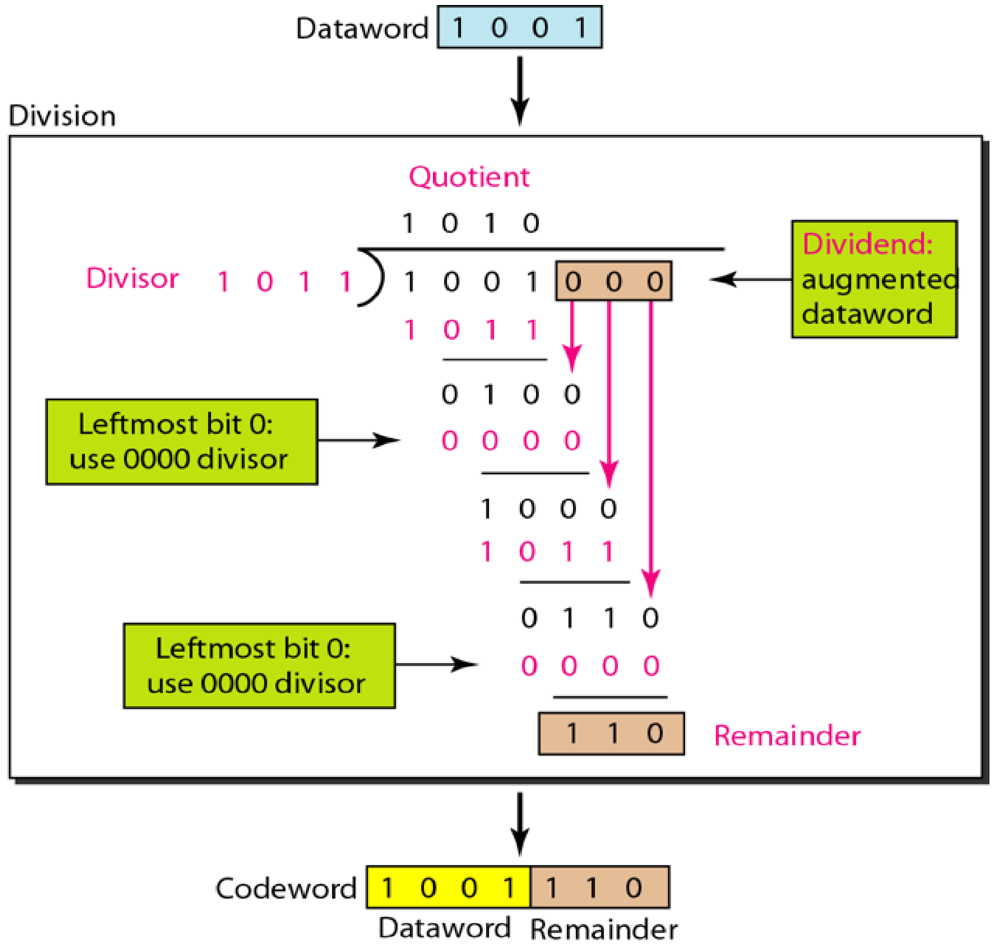


Cyclic Redundancy Check (CRC)

- An error detection method that is based on binary division
- A divisor is known for both the sender and the receiver
- A CRC is generated at the sender side and verified at the receiver side

Cyclic Redundancy Check (CRC) – CRC generation

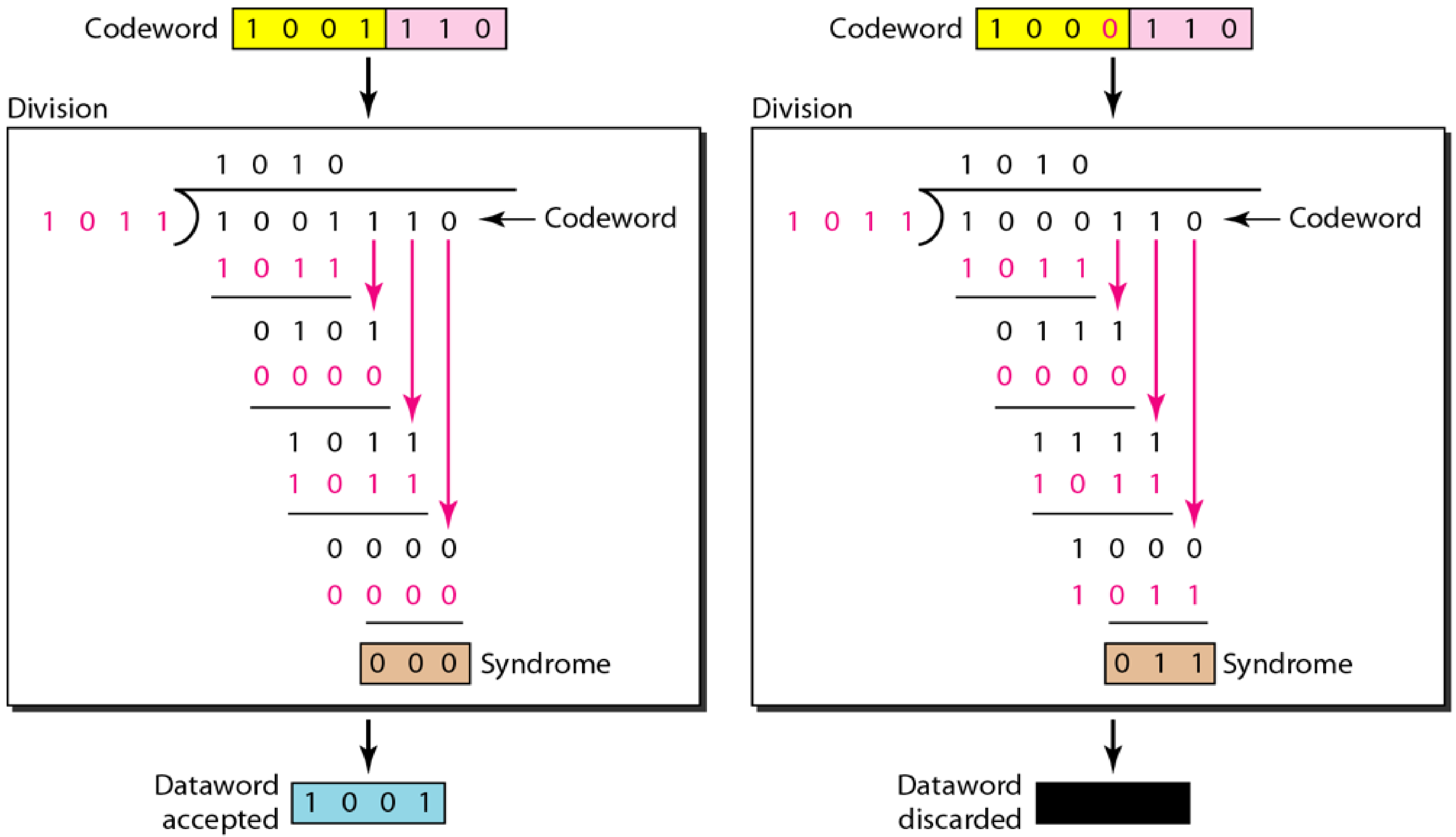
- The length of the divisor is *n*
- Append (*n-1*) 0's to the dataword
- Perform the binary division of augmented dataword by the divisor
- The CRC = remainder of the decision
- Codeword = Dataword +CRC



A	B	A **XOR** B
0	0	0
0	1	1
1	0	1
1	1	0

Example: Divisor = 1011, then three 0's are appended to the dataword
Data transmitted: 1 0 0 1 1 1 0

Cyclic Redundancy Check (CRC) – Receiver side



- The receiver divides the received data by the divisor.
- If the result of the division is all 0’s at the remainder, then no transmission error is assumed.

A- Error Detection - Summary

- All of these methods are for error detection
- Generally, they do not provide any way to locate/correct the errors

- Parity check:
  - Extremely simple, can be fooled by certain types of errors (e.g., two bits in error)
  - Useful for small frame sizes when the typical error is a single bit

- Checksum:
  - Pretty simple
  - More powerful than parity check (lower probability of undetected errors)

- CRC:
  - Most complicated and most powerful (extremely low probability of undetected errors for longer CRC checks)
  - IEEE 802 specifies a CRC-32 check

B- Error Correction

- Error detection methods are not enough!

- Error correction may happen in two ways:
  - Ask the sender to re-transmit the entire information (Backward Error Correction)
  - The receiver applies error-correcting algorithms (Forward Error Correction)

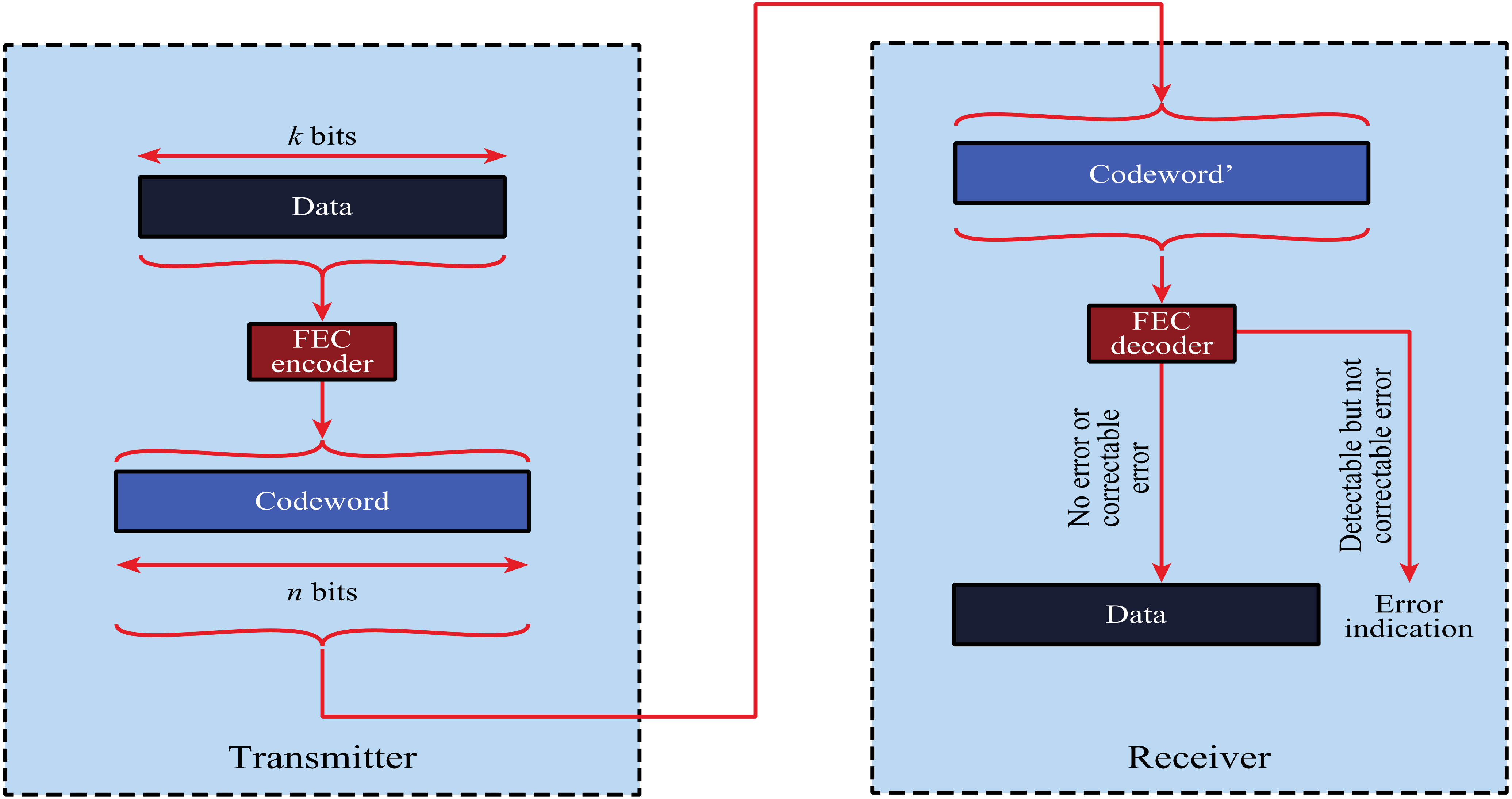
- Problems with re-transmission:
  - Not suitable for multimedia transmission
  - What if link is not bi-directional, e.g., HDTV?
  - What if BER on link is very high?
  - What If the link has high latency, e.g., satellite communications?

- Forward error correction (FEC) is a way of adding redundancy to messages so that the receiver can both detect and correct common errors.

Forward Error Correction (FEC)

- Advantages
  - FEC does not require handshaking between the source and the destination, therefore it can be used for broadcasting of data to many destinations simultaneously from a single source
  - Another advantage is that FEC saves bandwidth required for retransmission making it suitable for real time systems
- Main limitation:
  - if there are too many errors, the frames need to be retransmitted.

Forward Error Correction (FEC)



Transmitter

- Forward error correction (FEC) encoder maps each k-bit block into an n-bit block codeword
- Codeword is transmitted

Receiver

- Block passed through an FEC decoder for error checking

FEC - Codebook

- A codebook is a mapping from *k*-bit data sequences to *n*-bit codewords with n > k.
- The code rate r = $\frac{k}{n}$< 1.
- Example (5,2) code (r = $\frac{2}{5}$):

Data Block	Codeword
00	00000
01	00111
10	11001
11	11110

- The transmitter and receiver both know the codebook.
- The transmitter takes data blocks, maps them to codewords and transmits the codeword
- The receiver receives the codewords (potentially with one or more errors) and maps them back to data blocks

FEC - Hamming Distance

- The error correction capability of a block code is directly related to the “Hamming distance” between each of the codewords.
- The Hamming distance between n-bit codewords v1 and v2 is defined as

$$d(v_1, v_2) = \sum_{\ell=0}^{n-1} \text{XOR}(v_1(\ell), v_2(\ell))$$

- This is simply the number of bits in which v1 and v2 are different
- Example: v1 = 011011 and v2 = 110001. An XOR of these codewords gives XOR(v1, v2) = 101010. Hence the Hamming distance d(v1, v2) = 3
- The **minimum distance** is defined as

$$d_{\min} = \min_{i \neq j} d(v_i, v_j)$$

Hamming Distance vs. Redundancy

The redundancy of an (n, k) code is

$$\text{redundancy} = \frac{n-k}{k}.$$

Our (5,2) example code again:

Data Block	Codeword
00	v_1=00000
01	v_2=00111
10	v_3=11001
11	v_4=11110

The redundancy is $\frac{5-2}{2} = \frac{3}{2}$.

The Hamming distances are

$$\begin{aligned} d(v_1, v_2) &= 3 \\ d(v_1, v_3) &= 3 \\ d(v_1, v_4) &= 4 \\ d(v_2, v_3) &= 4 \\ d(v_2, v_4) &= 3 \\ d(v_3, v_4) &= 3 \end{aligned}$$

hence $d_{\min} = 3$.

In general, we want $d_{\min}$ to be large and the redundancy to be small.

Which Code is Better?

Code 1:

Data Block	Codeword
00	v_1=00000
01	v_2=00111
10	v_3=11001
11	v_4=11110

Code 2:

Data Block	Codeword
0	v_1=000
1	v_2=111

d_{min} = 3

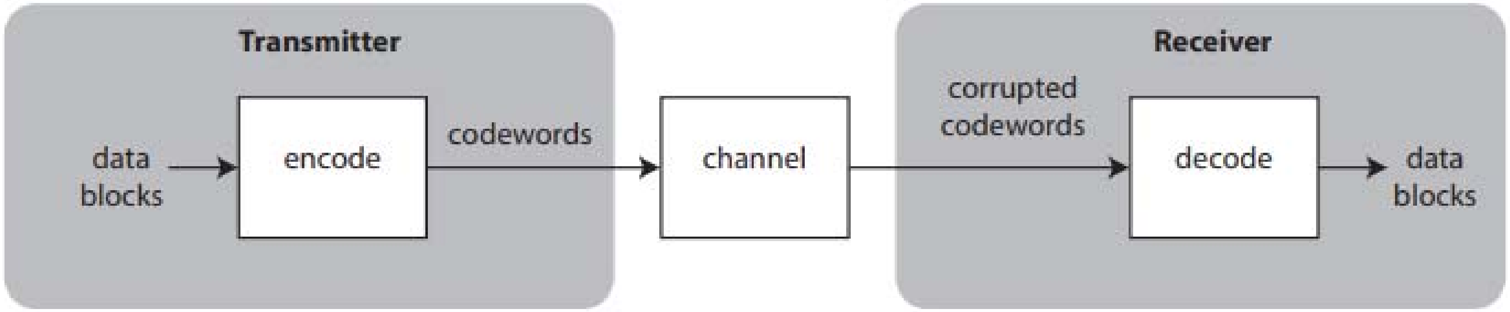
Redundancy = $\frac{5-2}{2} = \frac{3}{2}$

d_{min} = 3

Redundancy = $\frac{3-1}{1}$ = 2

Code 1 is better!

Decoding Invalid Codewords



- Suppose the transmitter wants to send data block 00 using our (5,2) block code.
- This gets encoded as 00000 and sent through the channel.
- Suppose the output of the channel is 00100 (one bit received in error). This is not a valid codeword.
- What should the receiver do?

Minimum Distance Decoding

- When an invalid codeword is received, the receiver should choose the valid codeword with the minimum Hamming distance to the invalid codeword.

Our (5,2) example code again:

Data Block	Codeword
00	v_1=00000
01	v_2=00111
10	v_3=11001
11	v_4=11110

If we receive 00100, the Hamming distances are

$$d(00100, v_1) = 1$$
$$d(00100, v_2) = 2$$
$$d(00100, v_3) = 4$$
$$d(00100, v_4) = 3$$

hence we should pick codeword v_1. The receiver then decodes this codeword as the data block 00 and the data is correctly received.

This (5,2) code can always correct codewords received with one error.

Minimum Distance Decoding (ctn'd)

Data block	Codeword
00	00000
01	00111
10	11001
11	11110

- Received: 01100
  - Two bits from 00000
  - Two bits from 11110
  - No other codes closer
  - Cannot decode!

Remarks on FEC

- Forward error correction is used extensively in wireless and wired communication systems.
- Rather than rejecting and re-requesting erroneous messages, the receiver can automatically correct the most common types of errors.

- Performance of block codes characterized by d_{min} and redundancy
- Inherent tradeoff: increasing d_{min} requires increasing redundancy (lowering the code rate *r*)

- Block coding (as covered here) is just one type of coding
- Modern codes (Turbo codes (mid 1990s), Low density parity check (LDPC) codes) can get very close to the Shannon limit

Error Detection and Correction - Summary

- Even with the use of redundant bits there still may be undetected bit errors; i.e., the receiver may be unaware that the received information contains bit errors
  - ➢ Need to choose error-detection scheme that keeps the probability of such occurrences small

- However, more sophisticated error-detection and-correction techniques incur a larger overhead!
  - ➢ More computation is needed to transmit a larger number of parity bits