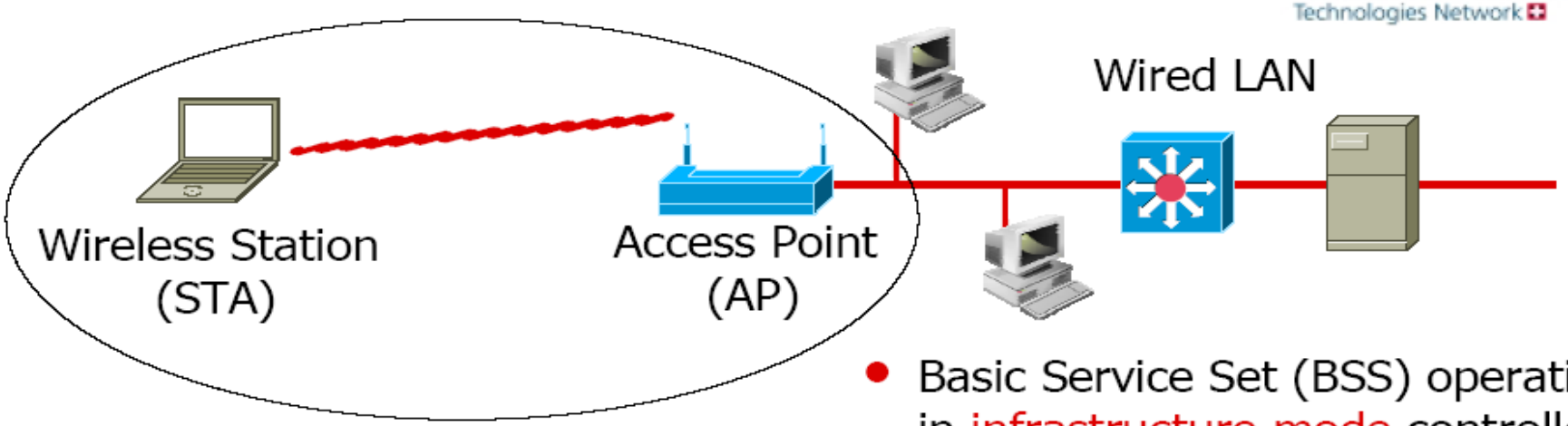
Computing Principles for Embedded Systems

Prof. Yingying (Jennifer) Chen

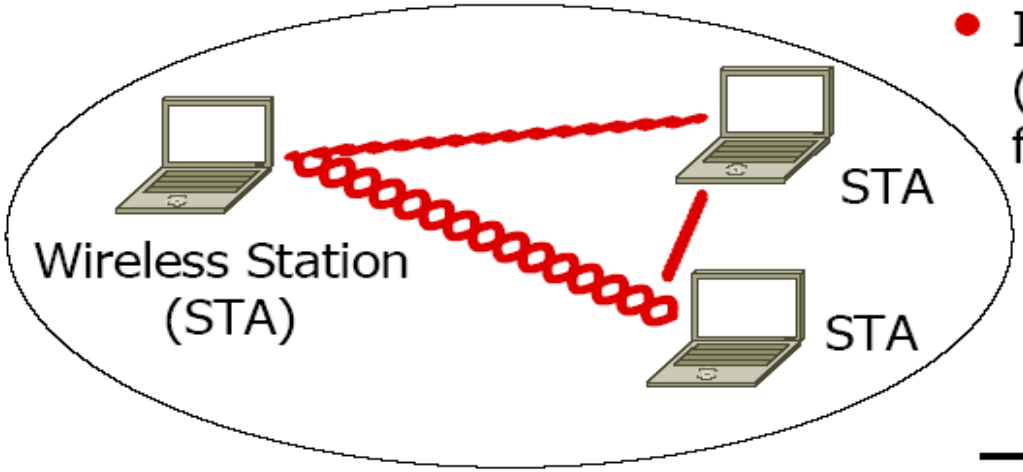
Department of Electrical and Computer Engineering
Rutgers Technology

Wireless Networks

IEEE 802.11 WLAN Architecture



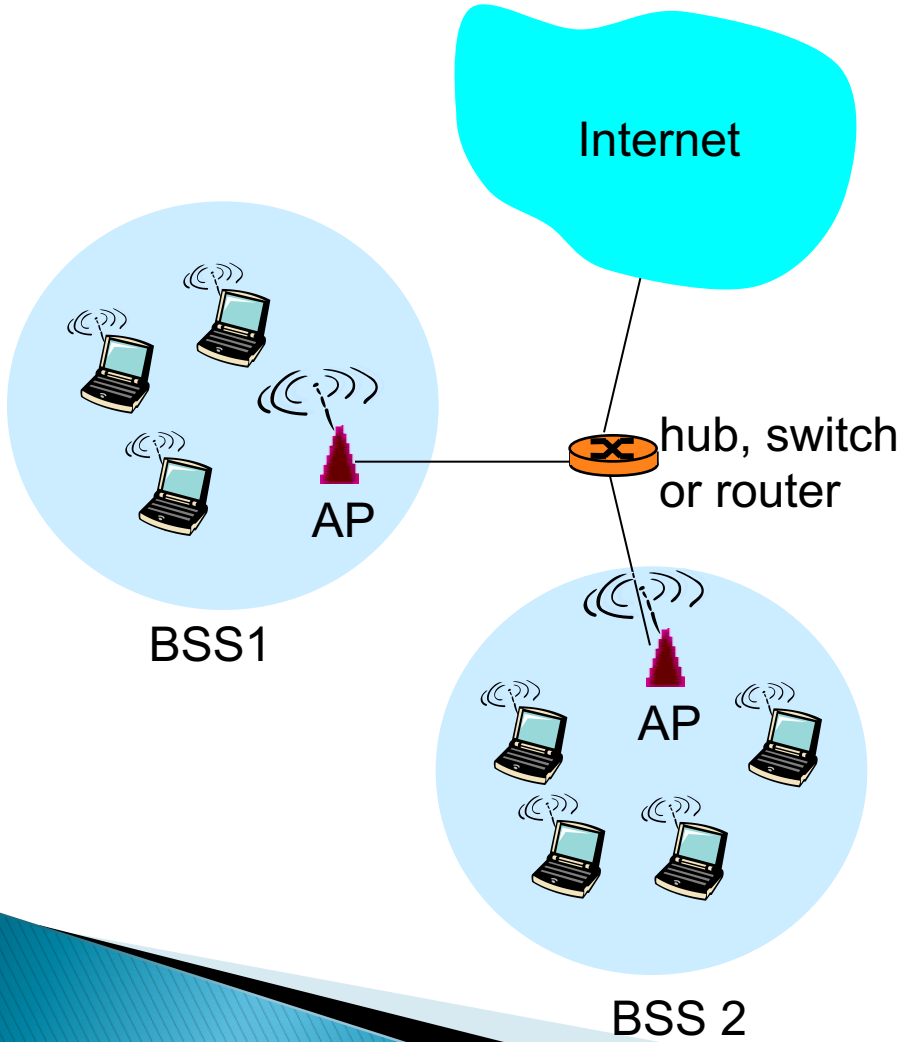
- Basic Service Set (BSS) operating in infrastructure mode controlled by access point
- Independent Basic Service Set (IBSS) operating in ad hoc mode forms a peer-to-peer network



IEEE 802.11 Wireless LAN

- 802.11b
  - 2.4-5 GHz unlicensed radio spectrum
  - up to 11 Mbps
  - direct sequence spread spectrum (DSSS) in physical layer
    - all hosts use same chipping code
  - widely deployed, using base stations
- 802.11a
  - 5-6 GHz range
  - up to 54 Mbps
- 802.11g
  - 2.4-5 GHz range
  - up to 54 Mbps
- All use CSMA/CA for multiple access
- All have base-station and ad-hoc network versions

802.11 LAN architecture



- wireless host communicates with base station
  - base station = access point (AP)
- Basic Service Set (BSS) (aka "cell") in infrastructure mode contains:
  - wireless hosts
  - access point (AP): base station
  - ad hoc mode: hosts only

Wireless Link Characteristics

Differences from wired link ….

- decreased signal strength: radio signal attenuates as it propagates through matter (path loss)
- interference from other sources: standardized wireless network frequencies (e.g., 2.4 GHz) shared by other devices (e.g., phone); devices (motors) interfere as well
- multipath propagation: radio signal reflects off objects ground, arriving ad destination at slightly different times

…. make communication across (even a point to point) wireless link much more "difficult"

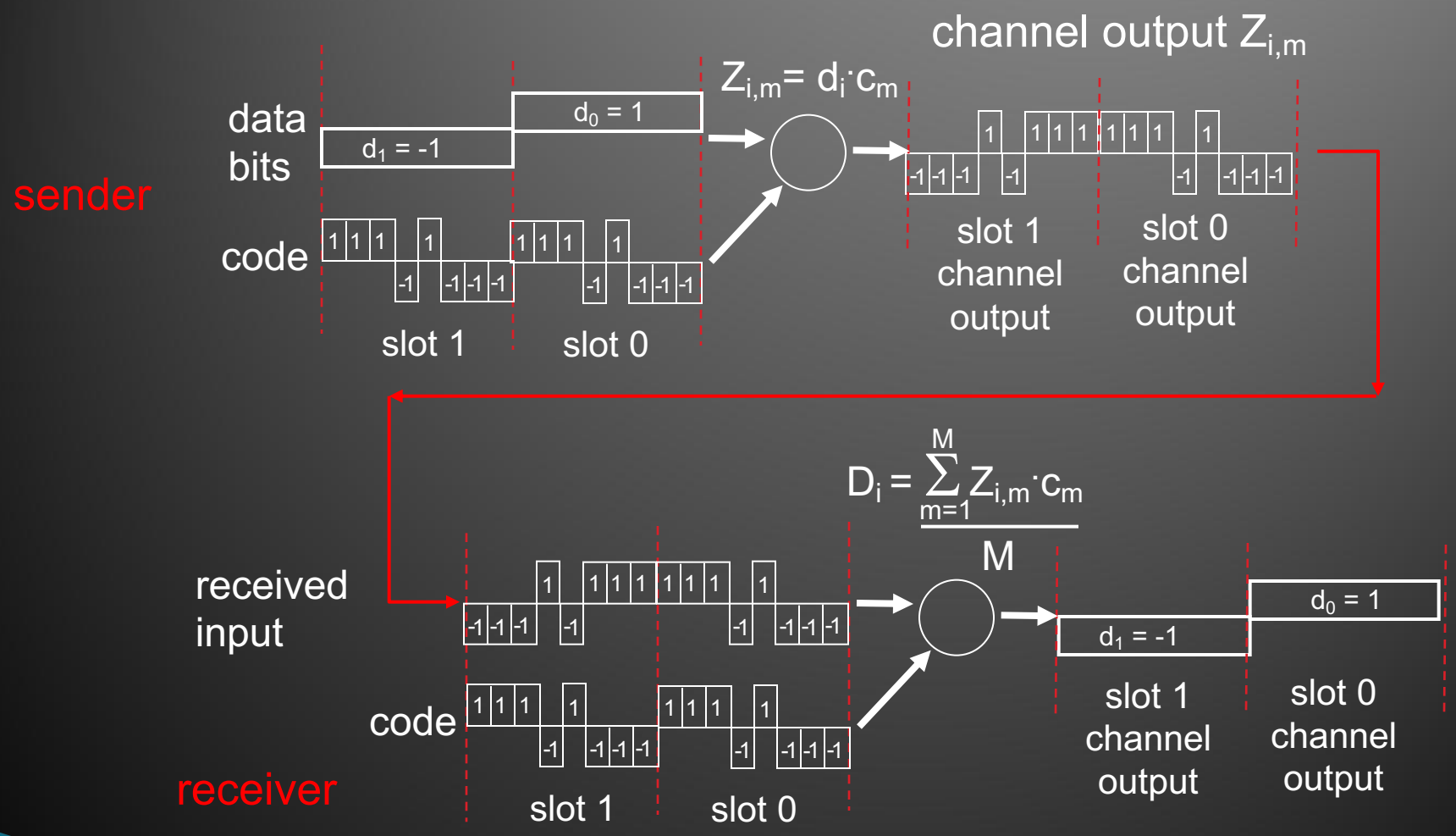
802.11: Channels, association

- 802.11b: 2.4GHz–2.485GHz spectrum divided into 11 channels at different frequencies
  - AP admin chooses frequency for AP
  - interference possible: channel can be the same as that chosen by neighboring AP!
- host: must *associate* with an AP
  - scans channels, listening for *beacon frames* containing AP's name (SSID) and MAC address
  - selects AP to associate with
  - may perform authentication
  - will typically run DHCP to get IP address in AP's subnet

Code Division Multiple Access (CDMA)

- used in several wireless broadcast channels (cellular, satellite, etc) standards
- unique "code" assigned to each user; i.e., code set partitioning
- all users share same frequency, but each user has own "chipping" sequence (i.e., code) to encode data
- *encoded signal* = (original data) X (chipping sequence)
- *decoding:* inner-product of encoded signal and chipping sequence
- allows multiple users to "coexist" and transmit simultaneously with minimal interference (if codes are "orthogonal")

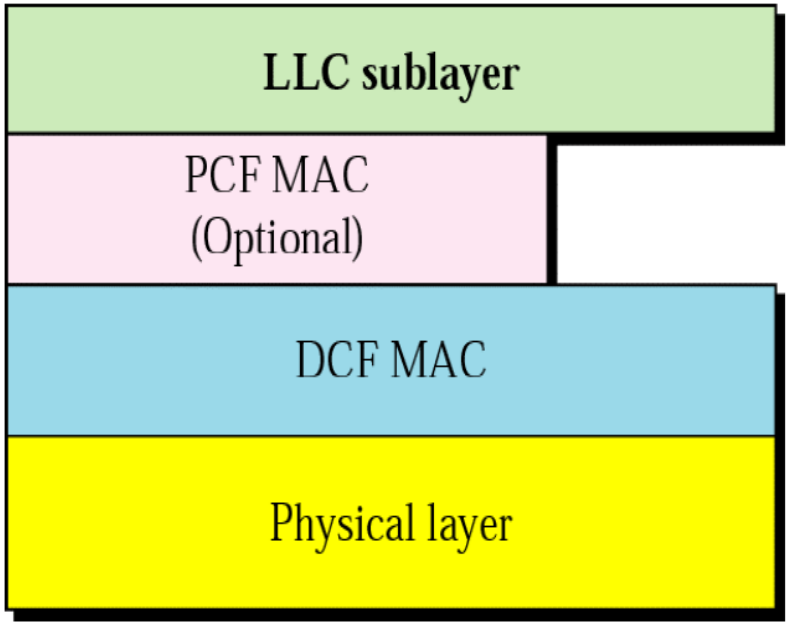
CDMA Encode/Decode



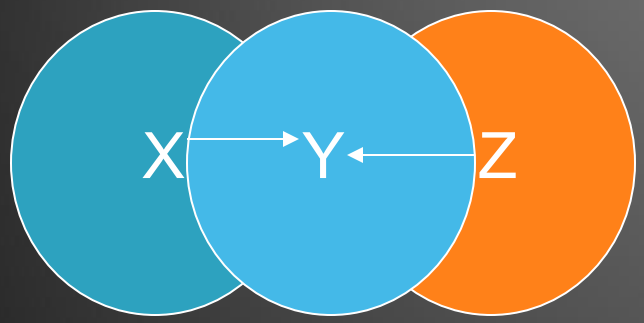
Wireless Medium Access

Two modes

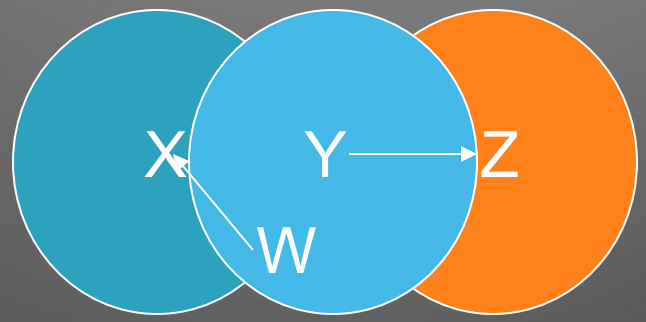
1) CSMA/CA– A contention based protocol. In 802.11 this mode is known as Distributed Coordination Function (DCF)
2) Priority-based access – A contention free access protocol usable on the infrastructure mode. Known as Point Coordination Function (PCF)



Problems with Carrier Sensing



Hidden Terminal Problem



Exposed Terminal Problem

Hidden Terminal problem:

- Z can't sense X; Tx to Y and collision with X
- No carrier does not always imply safe to send

Exposed terminal problem:

- W senses Y but can send to X
- Carrier may not imply unsafe to send

Problems of Wireless MAC

- Carrier sense
  - in Ethernet, we use carrier sense to avoid and detect potential collision
  - for wireless networks, the hidden-terminal, and the exposed-terminal problems make carrier sense (i.e., listen before talk) neither necessary nor sufficient
    - not detected transmission at the sender does not imply no current transmission to the receiver
    - detected transmission at the sender does not imply transmission will cause collision

IEEE 802.11 MAC Protocol: CSMA/CA

802.11 sender

1 if sense channel idle for **DIFS** then

transmit entire frame (no CD – collision detection)

2 if sense channel busy then

start random backoff time

timer counts down while channel idle

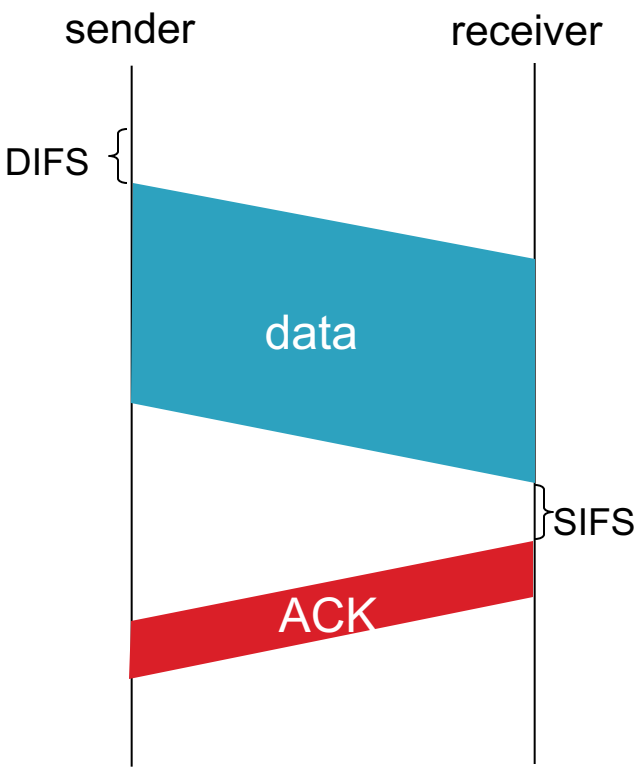
transmit when timer expires

if no ACK, increase random backoff interval, repeat 2

802.11 receiver

– if frame received OK

return ACK after **SIFS** (ACK needed due to hidden terminal problem)



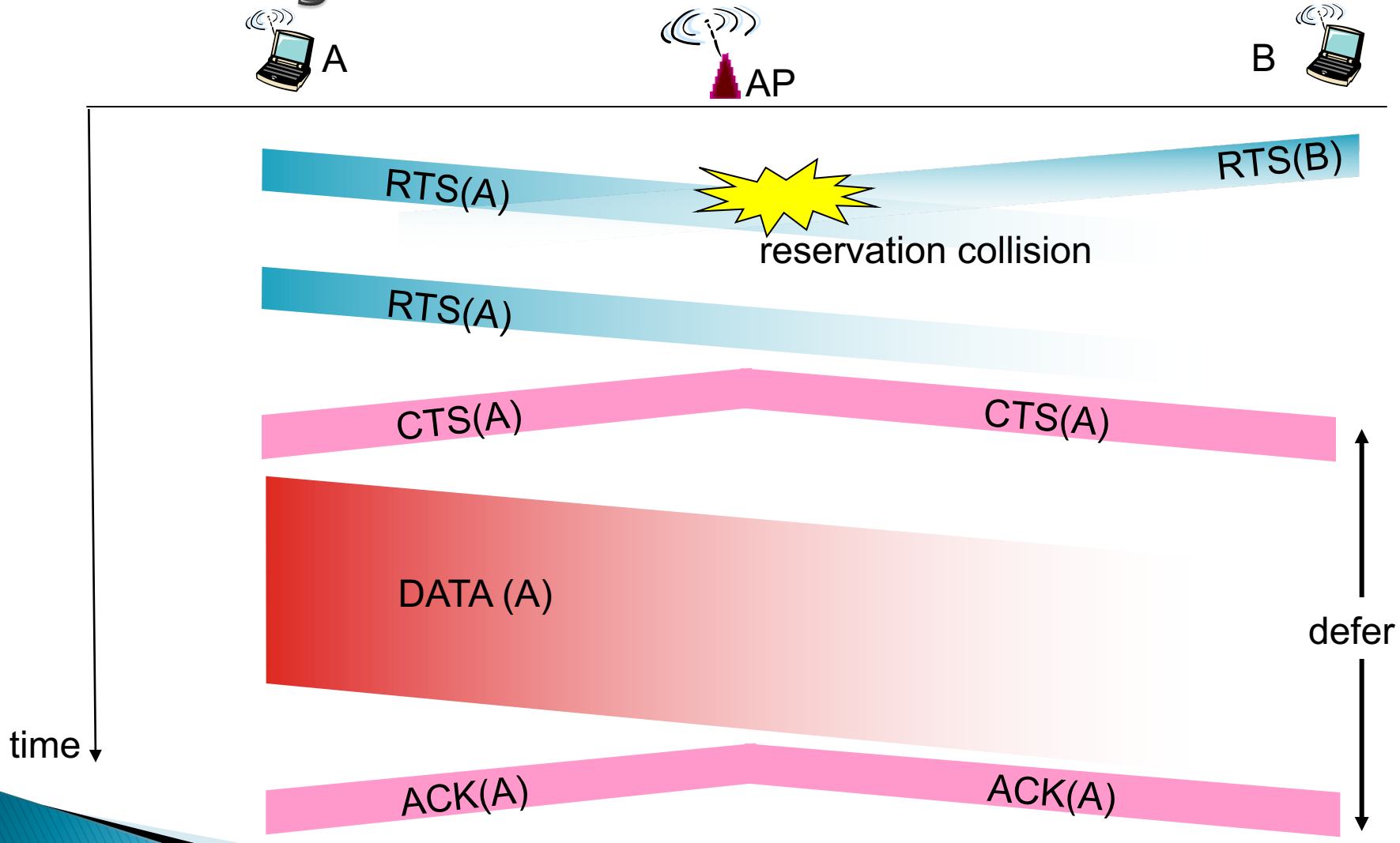
Avoiding collisions (more)

idea: allow sender to "reserve" channel rather than random access of data frames: avoid collisions of long data frames

- sender first transmits *small* request-to-send (RTS) packets to BS using CSMA
  - RTSs may still collide with each other (but they're short)
- BS broadcasts clear-to-send CTS in response to RTS
- CTS heard by all nodes
  - sender transmits data frame
  - other stations defer transmissions

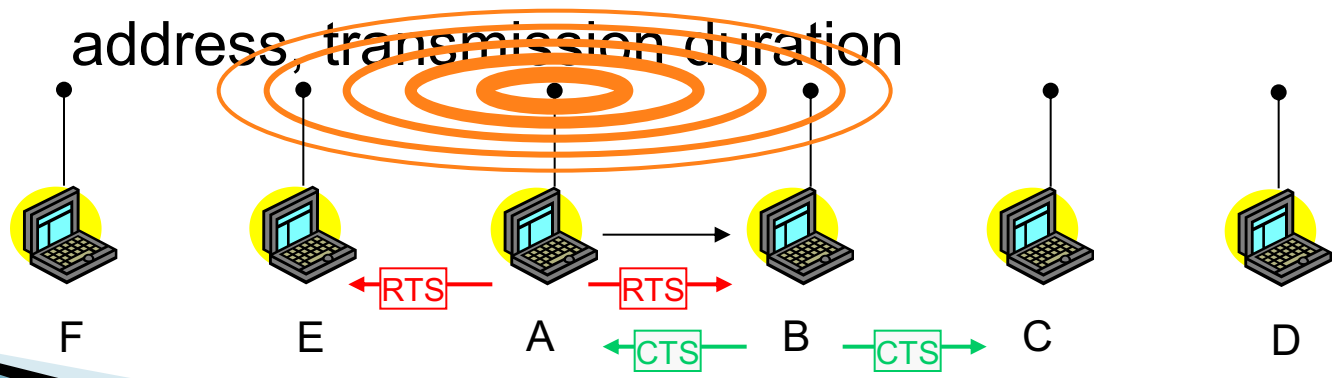
Avoid data frame collisions completely using small reservation packets!

Collision Avoidance: RTS-CTS exchange



Basic Solution: Using RTS/CTS to Address the Carrier Sense Problem

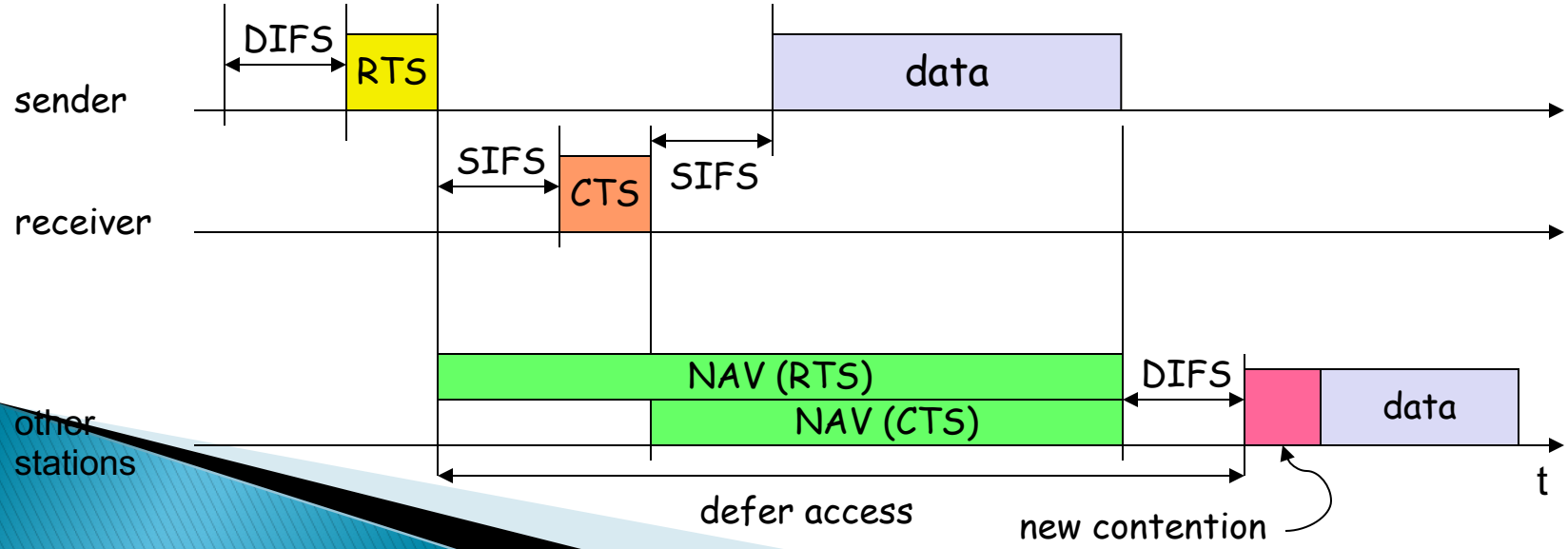
- Short signaling packets---virtual carrier sense
  - RTS (request to send) and CTS (clear to send)
    - to avoid collision at the receiver, any station who hears a CTS should not transmit
    - frames need to contain sender address, receiver address, transmission duration



Example: A sends to B

Basic Control Flow of RTS/CTS

- Sender sends RTS with NAV (**Network allocation Vector**, i.e. reservation parameter that determines amount of time the data packet needs the medium) after waiting for DIFS
- Receiver acknowledges via CTS after SIFS (if ready to receive)
  - CTS reserves channel for sender, notifying possibly hidden stations;
  - any station hearing CTS should be silent for NAV
- Sender can now send data at once



DIFS and SIFS

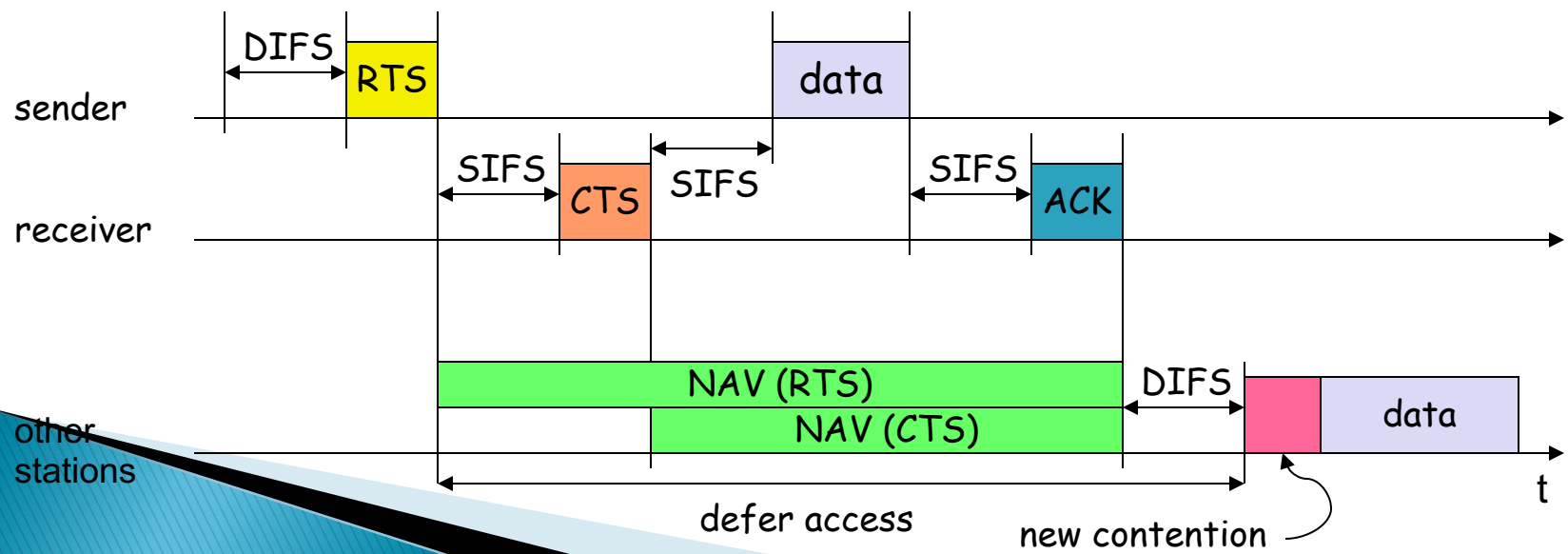
- DCF Interframe Space DIFS
  - If it finds that the medium is continuously idle for DCF Interframe Space (DIFS) duration, it is then permitted to transmit a frame. If the channel is found busy during the DIFS interval, the station should defer its transmission.

- Short Interframe Space SIFS (if ready to receive)
  - is the amount of time in microseconds required for a wireless interface to process a received frame and to respond with a response frame.

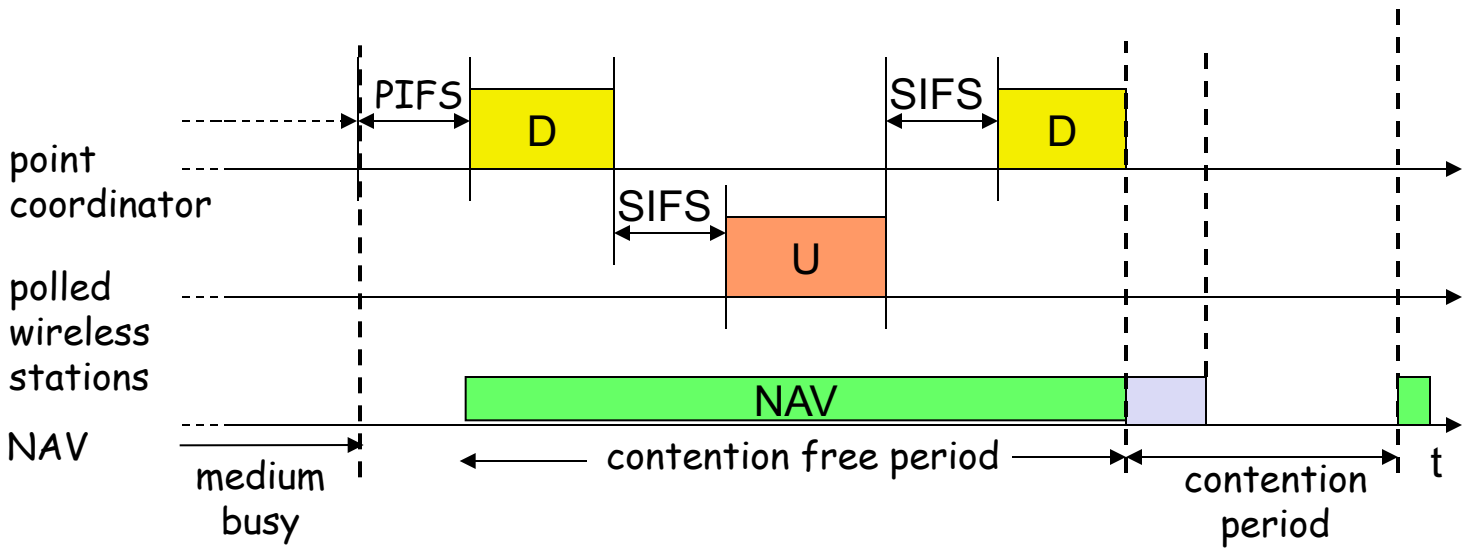
802.11: RTS/CTS + ACK, the Final Version

- 802.11 adds ACK in the signaling to improve reliability
  - implication: to avoid conflict with ACK, any station hearing RTS should not send for NAV
  - thus a station should not send for NAV if it hears either RTS and CTS

Note: RTS/CTS is optional in 802.11, and thus may not be always turned on---some network interface cards turn it on only when the length of a frame exceeds a given threshold



802.11: PCF for Polling



D: downstream poll, or data from point coordinator
U: data from polled wireless station

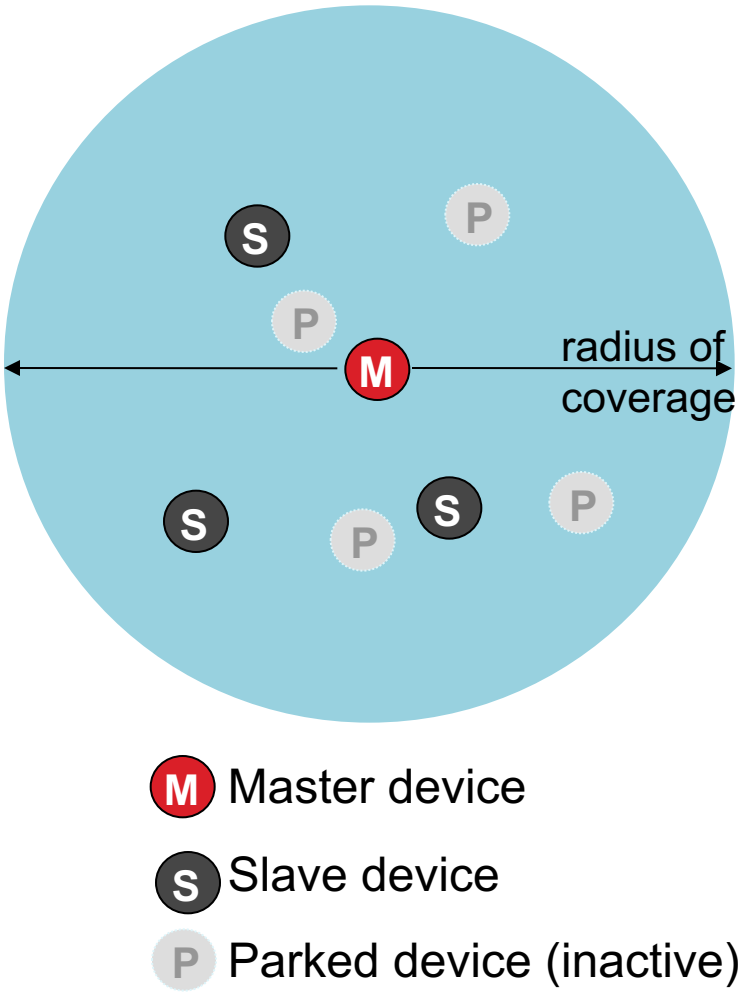
Bluetooth (802.15)

A cable replacement

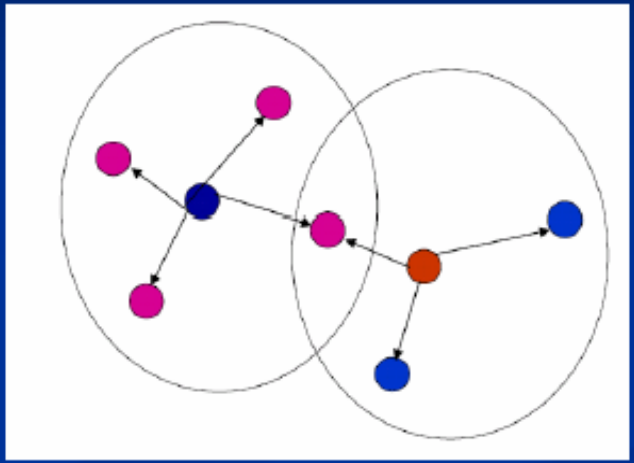
- Operates in the ISM band (2.4Ghz to 2.8 Ghz)
- Range is 10 cm to 10 meters can be extended to 100 meters by use of power control
- Data rates up to 1 Mbps (721Kbps)
- Supposed to be low cost, single chip radio
- Ideal for connecting devices in close proximity (piconet)
  - Phone and earpiece
  - Computer and printer
  - Camera and printer/fax etc

802.15: personal area network

- less than 10 m diameter
- replacement for cables (mouse, keyboard, headphones)
- ad hoc: no infrastructure
- master/slaves:
  - slaves request permission to send (to master)
  - master grants requests
- 802.15: evolved from Bluetooth specification
  - 2.4–2.5 GHz radio band
  - up to 721 kbps



Personal area networks

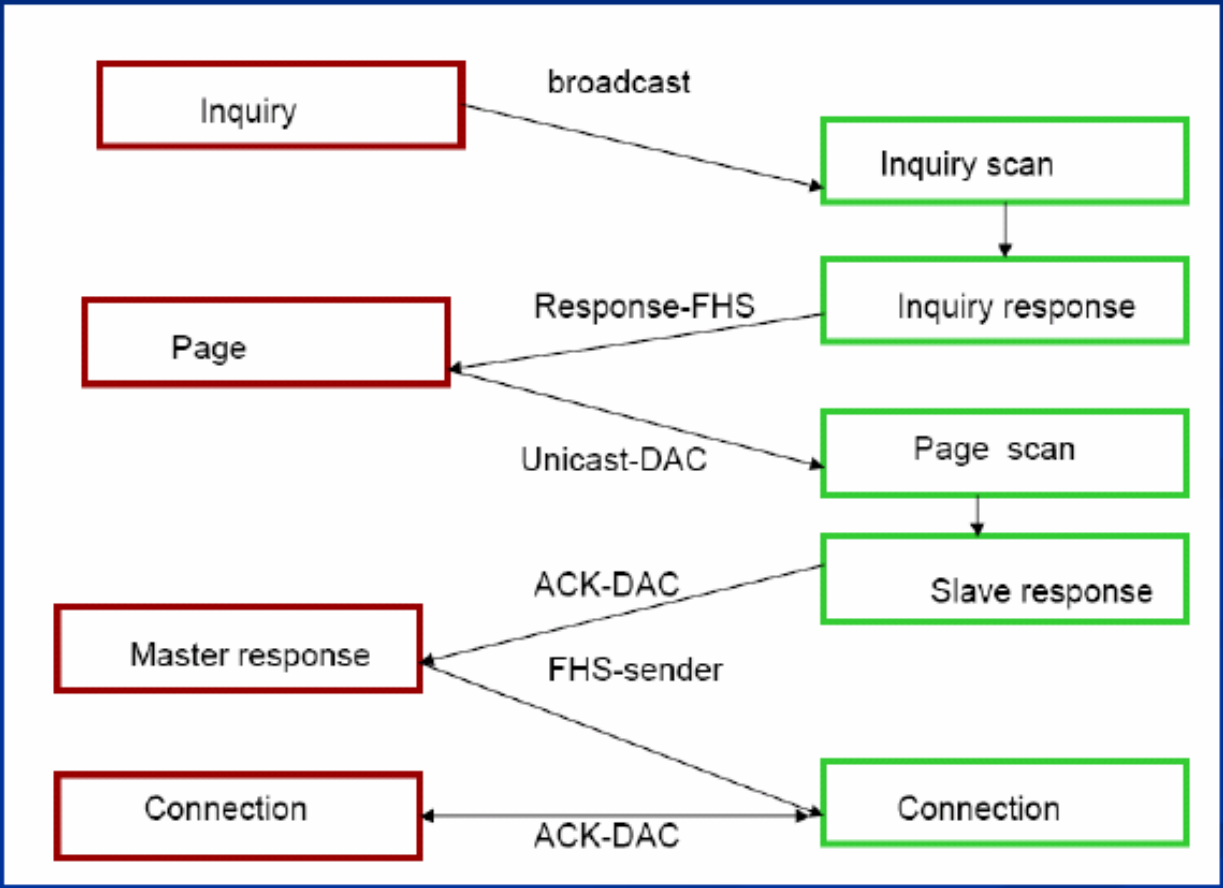


- Piconet
  - Master/slave nodes
  - Master and up to 7 slaves
  - Master allocates channels
- Scatternet
  - Node may be master in one network and slave in another network
  - Allows devices to be shared in different networks

Bluetooth Radio Link

- IEEE 802.11 operates in the same band using DSSS
- Bluetooth uses Frequency Hopping
- 83.5 MHz channel is divided into 79 1-MHz channels
  - 1600 hops per second (stays at one frequency for 625 microsecs)
  - Hopping sequence is 16 or 32
  - Selected by the master based on its MAC address
  - Master can connect up to seven slaves to form a piconet
- All members of the piconet use the same hopping sequence
- Maters starts to transmit in the even slots and the slaves start to transmit in the odd slots

Connection establishment



Network Data Link Layer

- Error Detection and Correction
- Medium Access Control

Error Detection and Correction

- Used on many link types
  - Can be used anytime you don't trust the media
    - Caches, memories, disks
- Major Algorithmic Strategies:
  - Parity
    - 1 and 2 dimensional
  - Hamming codes (Interleaved Parity Codes)
  - Checksums
  - Cyclic Redundancy Codes (CRC)

CRC Polynomial Codes

- Can detect errors on large chunks of data
- Has low overhead
- More robust than parity bit
- Requires the use of a “code polynomial”
  - Example: $x^2 + 1$

Cyclic Redundancy Check

- Procedure:
  1. Let r be the degree of the code polynomial. Append r zero bits to the end of the transmitted bit string. Call the entire bit string $S(x)$
  2. Divide $S(x)$ by the code polynomial using modulo 2 division.
  3. Subtract the remainder from $S(x)$ using modulo 2 subtraction.
- The result is the checksummed message for transmission.

Generating a CRC
Example

Message: 1011 $\longrightarrow$ $1 \times x^3 + 0 \times x^2 + 1 \times x + 1$
$= x^3 + x + 1$

Code Polynomial: $x^2 + 1$ (101)

Step 1: Compute $S(x)$

$r = 2$

$S(x)$ = 101100

Modulo 2 Arithmetic

- Addition/Subtraction:
  - Using an exclusive OR (xor) operation on the corresponding binary digits

A	B	A xor B
0	0	0
0	1	1
1	0	1
1	1	0

  - Modulo 2 addition and subtraction has the same result

Addition:

```
(X) 10110100
(Y)  00101010  +
     -------------
    -
(Z)   10011110
```

Subtraction:

```
(X) 10110100
(Z)   10011110  -
     -------------
    -
(Y)   00101010
```

Generating a CRC

Example (cont'd)

Step 2: Modulo 2 divide

```
                1001
        ┌──────────────────
101     │   101100
            101
            ---
             001
             000
             ---
              010
              000
              ---
               100
               101
               ---
                01   ←———— Remainder
```

Generating a CRC

Example (cont'd)

Step 3: Modulo 2 subtract the remainder from $S(x)$

```
101100
-    01
------
101101
```

Checksummed Message

Decoding a CRC

- Procedure
  1. Let n be the length of the checksummed message in bits
  2. Divide the checksummed message by the code polynomial using modulo 2 division. If the remainder is zero, there is no error detected.

Decoding a CRC

Example

```
101101  <------ Checksummed message (n = 6)
  |
  v
1011    <------ Original message (if there are
                no errors)

            1001
      ____________
101 |     101101
        101
        ---
         001
         000
         ---
          010
          000
          ---
           101
           101
           ---
            00  <------ Remainder = 0
                        (No error detected)
```

Decoding a CRC
Another Example

When a bit error occurs, there is a large probability that it will produce a polynomial that is not an even multiple of the code polynomial, and thus errors can usually be detected.

```
                1000
        ____________________
101 |   101001
        101
        ---
         000
         000
         ---
          000
          000
          ---
           001
           000
           ---
            01
```

Remainder = 1
(Error detected)

Choosing a CRC polynomial

- The longer the polynomial, the smaller the probability of undetected error
- Common standard polynomials:

  (1) CRC-12: $x^{12} + x^{11} + x^3 + x^2 + x^1 + 1$

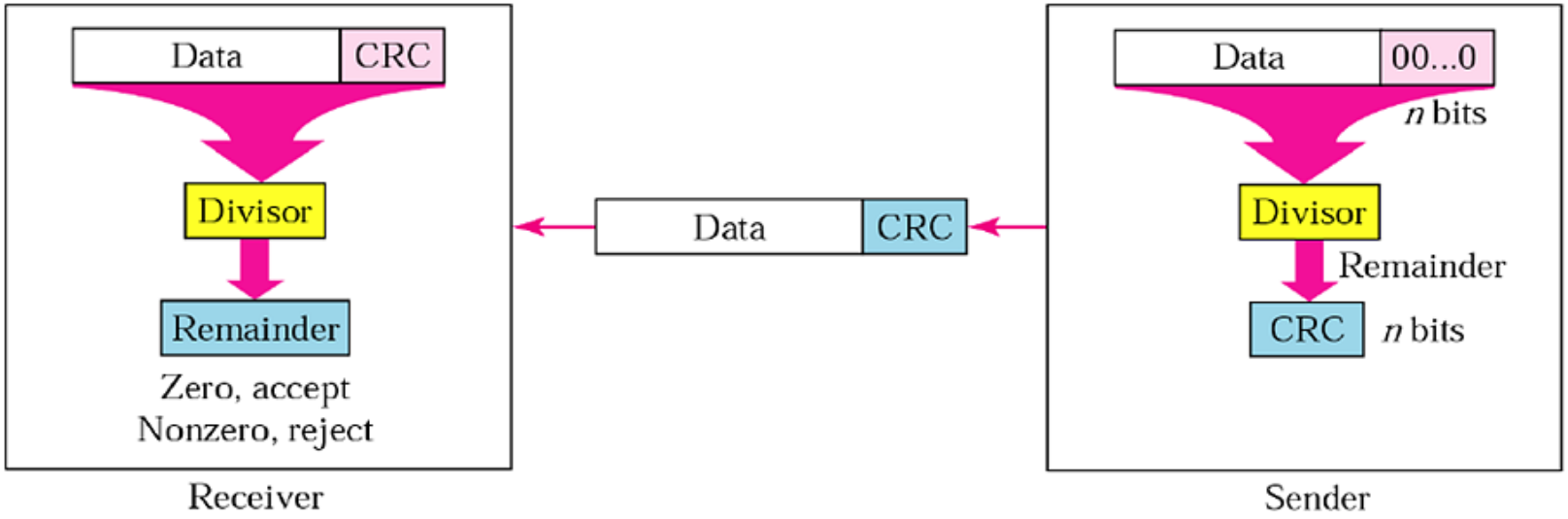
  (2) CRC-16: $x^{16} + x^{15} + x^2 + 1$

  (3) CRC-CCITT: $x^{16} + x^{12} + x^5 + 1$

Summary

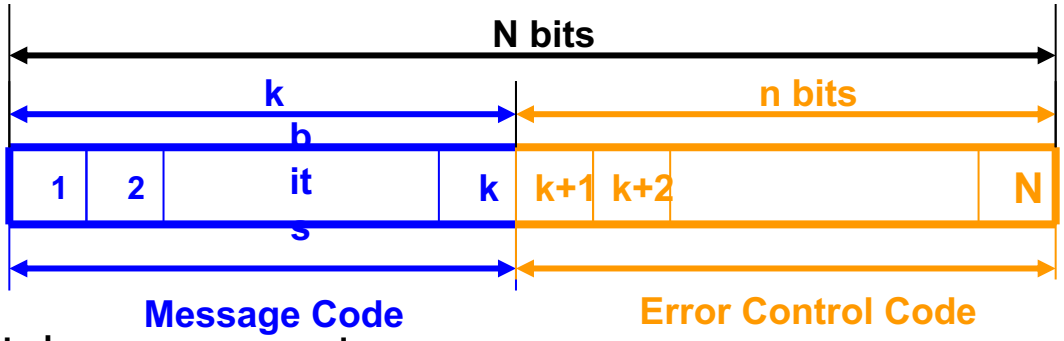
- Cyclic redundancy check (CRC) codes
  - Invented by W. Wesley Peterson, and published in 1961
  - A type of linear block codes
    - Generally, not cyclic, but derived from cyclic codes
  - A systematic error detecting code
    - A group of error control bits (which is the remainder of a polynomial division of a message polynomial by a generator polynomial) is appended to the end of the message block
    - With considerable burst-error detection capability
  - The receiver generally has the ability to send retransmission requests back to the data source through a feedback channel

CRC Generator and Checker



CRC Codes

- Binary (N,k) CRC codes
  - k message or data bits are encoded into N code bits by appending to the message bits a sequence of $n=N-k$ bits



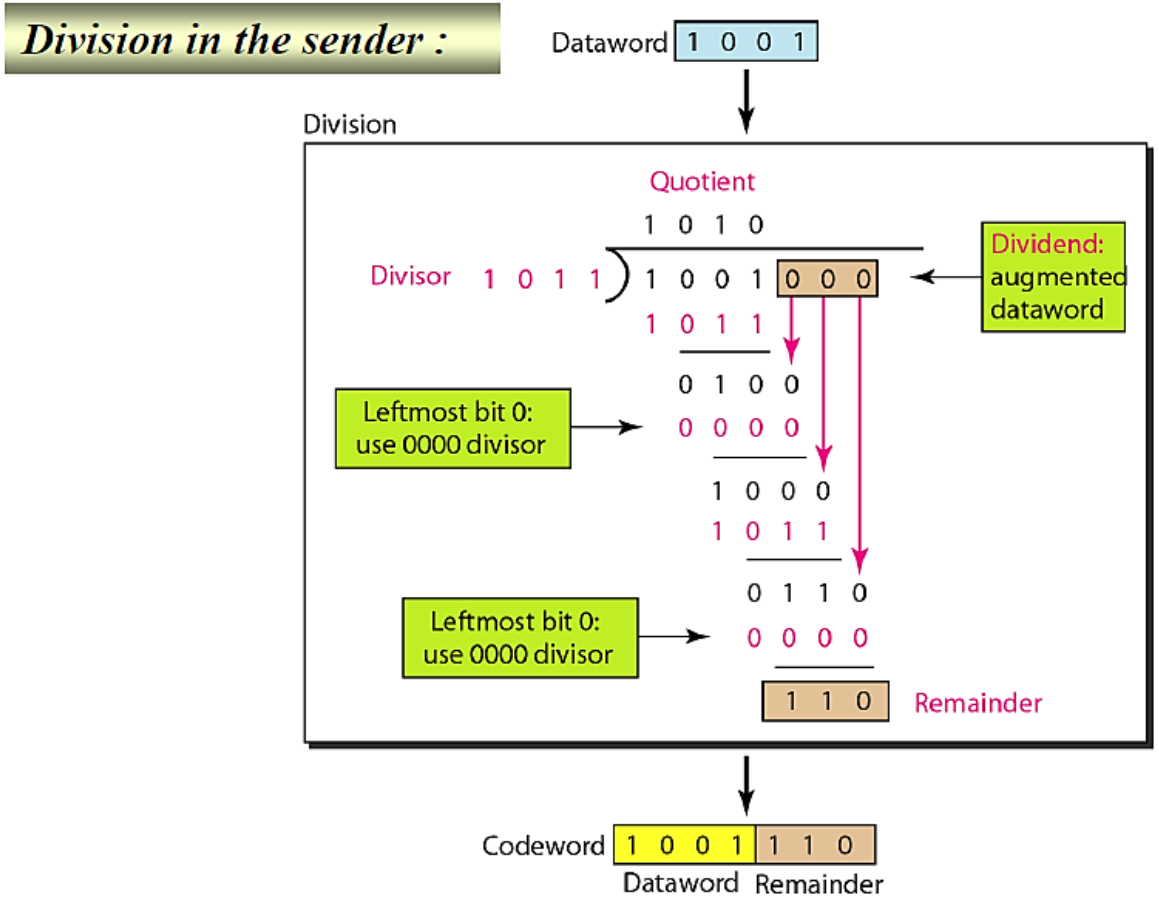
  - Polynomial representation
    - Message code: $m = [m_{k-1}\ m_{k-2} \ldots m_1\ m_0]$

      $m(X) = m_{k-1}X^{k-1}+m_{k-2}X^{k-2}+\ldots+ m_1X^1+m_0$
    - Appended bits (error control code): $R = [r_{n-1}\ r_{n-2} \ldots r_1\ r_0]$

      $R(X) = r_{n-1}X^{n-1}+r_{n-2}X^{n-2}+\ldots+ r_1X^1+r_0$
    - CRC code: $C = [c_{N-1}\ c_{N-2} \ldots c_1\ c_0] = [m_{k-1}\ m_{k-2} \ldots m_1\ m_0\ r_{n-1}\ r_{n-2} \ldots r_1\ r_0]$

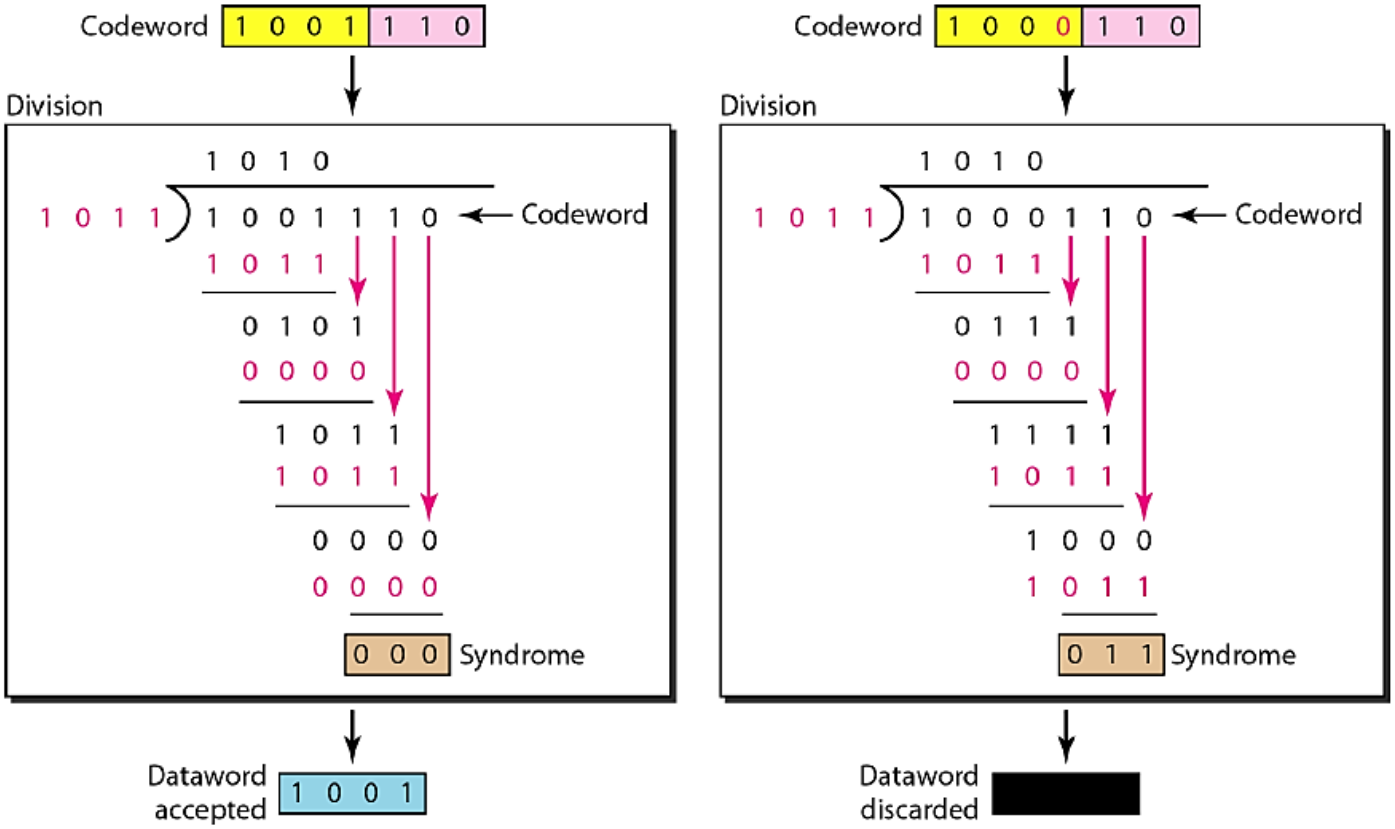
      $C(X) = c_{N-1}X^{N-1}+c_{N-2}X^{N-2}+\ldots+ c_1X^1+c_0 = X^n m(X) + R(X)$

Example of CRC(7,4) – Sender



Example of CRC(7,4) – Receiver

Division in the receiver (two cases) :



More Examples

- Sender:
  - Suppose we want to transmit the information string: 1111101.
  - The receiver and sender decide to use the (arbitrary) polynomial pattern, 1101.
  - The information string is shifted left by one position less than the numbe of positions in the divisor.
  - The remainder is found through modulo 2 division (at right) and added to the information string: 1111101000 + 111 = 1111101111.

```
          1011011
1101)1111101000
     1101
     001010
       1101
       01111
        1101
        001000
          1101
          01010
           1101
           0111
```

More Examples

- Receiver:
  - If no bits are lost or corrupted, dividing the received information string by the agreed upon pattern will give a remainder of zero.
  - We see this is so in the calculation at the right.
  - Real applications use longer polynomials to cover larger information strings.

```
          1011011
1101)1111101111
     1101
     001010
       1101
       01111
        1101
        001011
          1101
          01101
           1101
           0000
```

More Examples

- Receiver:
  - If 1 bits is corrupted (in red), dividing the received information string by the agreed upon pattern will give a remainder of non-zero.
  - We see this is so in the calculation at the right.

```
          1011011
1101)1111110111 0
     1101
     ----
     001010
       1101
       -----
       01111
        1101
        ----
        001011
          1101
          ----
          0110 0
           1101
           ----
           000 1
```