

Computer Networks

Exercise Session 07

Prof. Dr. Oliver Hahm

Frankfurt University of Applied Sciences
Faculty 2: Computer Science and Engineering
oliver.hahm@fb2.fra-uas.de
<https://teaching.dahahm.de>

December 18, 2024

General Schedule

All exercises will follow this general schedule

- Identify potential understanding problems
 - Ask your questions
 - Recap of the lecture
- Address the understanding problems
 - Answer your questions
 - Repeat certain topics
- Walk through the exercises/solutions → Some hints and guidance
 - Work time or presentation of results

Contention-based Medium Access

You have seen ...

- that participants must **compete for medium access** in contention-based MAC protocols
- **collisions** reduce the **performance** of the network
- they should be **detected** and **avoided**
- the trade-off between **throughput** and **latency**

Contention-free Medium Access

You have seen ...

- how **resources** like time or frequencies can be **allocated** in advance for contention-free medium access
- that (particularly static) contention-free MAC protocols provide less throughput compared to contention-based protocols on low utilization of the network
- that combination of MAC protocols is feasible

Data Link Layer: Error Control

You have seen . . .

- that errors may occur on the Physical Layer and it is one of the services provided by the Data Link Layer to **handle these errors**
- what **checksums** are and how they can be built with **parity bits** or **CRCs**
- what a **Hamming distance is** and what needs to be fulfilled to allow for errors to be **detected** or **corrected**

Exercise 1: Error Control

- In order to **detect** or even **correct errors redundancy** is required
 - increased frame sizes
 - ⇒ less goodput
- If no error control is in place on the data link layer, upper layers need to verify the validity of the data and request retransmissions

Exercise 1: Error Control

- In order to **detect** or even **correct errors redundancy** is required
 - increased frame sizes
 - ⇒ less goodput
- If no error control is in place on the data link layer, upper layers need to verify the validity of the data and request retransmissions
- **Hamming distance**
 - What is the hamming distance between **1111 0000** and **0000 0000**?

Exercise 1: Error Control

- In order to **detect** or even **correct errors redundancy** is required
 - increased frame sizes
 - ⇒ less goodput
- If no error control is in place on the data link layer, upper layers need to verify the validity of the data and request retransmissions
- **Hamming distance**
 - What is the hamming distance between **1111 0000** and **0000 0000**?
 - 4 → Four bits need to be changed to get from the first to the second word

Exercise 1: Error Control

- In order to **detect** or even **correct errors redundancy** is required
 - increased frame sizes
 - ⇒ less goodput
- If no error control is in place on the data link layer, upper layers need to verify the validity of the data and request retransmissions
- **Hamming distance**
 - What is the hamming distance between **1111 0000** and **0000 0000**?
 - 4 → Four bits need to be changed to get from the first to the second word
 - What is the hamming distance between **1010 1010** and **0101 0101**?

Exercise 1: Error Control

- In order to **detect** or even **correct errors redundancy** is required
 - increased frame sizes
 - ⇒ less goodput
- If no error control is in place on the data link layer, upper layers need to verify the validity of the data and request retransmissions
- **Hamming distance**
 - What is the hamming distance between **1111 0000** and **0000 0000**?
 - 4 → Four bits need to be changed to get from the first to the second word
 - What is the hamming distance between **1010 1010** and **0101 0101**?
 - 8 → All eight bits need to be changed to get from the first to the second word

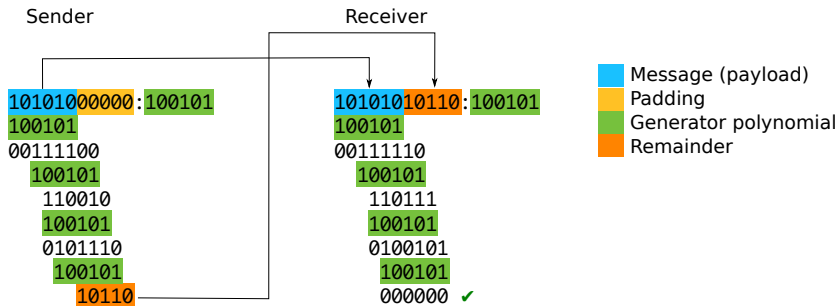
Exercise 1: Error Control

- In order to **detect** or even **correct errors redundancy** is required
 - increased frame sizes
 - ⇒ less goodput
- If no error control is in place on the data link layer, upper layers need to verify the validity of the data and request retransmissions
- **Hamming distance**
 - What is the hamming distance between **1111 0000** and **0000 0000**?
 - **4** → Four bits need to be changed to get from the first to the second word
 - What is the hamming distance between **1010 1010** and **0101 0101**?
 - **8** → All eight bits need to be changed to get from the first to the second word
 - If the allowed code words comprise only **10000** and **01111**, the minimum hamming distance is 5. Hence, up to two bit errors can be corrected. E.g., which word has been sent, if the receiver gets **11100**?

Exercise 1: Error Control

- In order to **detect** or even **correct errors redundancy** is required
 - increased frame sizes
 - ⇒ less goodput
- If no error control is in place on the data link layer, upper layers need to verify the validity of the data and request retransmissions
- **Hamming distance**
 - What is the hamming distance between **1111 0000** and **0000 0000**?
 - **4** → Four bits need to be changed to get from the first to the second word
 - What is the hamming distance between **1010 1010** and **0101 0101**?
 - **8** → All eight bits need to be changed to get from the first to the second word
 - If the allowed code words comprise only **10000** and **01111**, the minimum hamming distance is 5. Hence, up to two bit errors can be corrected. E.g., which word has been sent, if the receiver gets **11100**?
 - **10000**

Exercise 2: CRC



The CRC checksum is the remainder of the division of the message itself by the generator polynomial. The same calculation for the message plus appended remainder results to 0 if no transmission error has occurred.

Error-correction: Hamming Code

Error correction can be realized via **Hamming code**

- The bits of a data block are **numbered** from left to right, starting with 1
 - Bits, which are powers of 2 (1, 2, 4, 8, 16, etc.) are **parity** (or check) bits
 - The remaining bits are the **payload**
- **Example:**
 - 8 bits payload: 01001100

Position:	1	2	3	4	5	6	7	8
Payload:	0	1	0	0	1	1	0	0

Position:	1	2	3	4	5	6	7	8	9	10	11	12
Data to be transmitted:	?	?	0	?	1	0	0	?	1	1	0	0

Hamming Code – Parity Bits

- Each **position** in the data block can be expressed by the **same amount** of digits that we have as parity bits
- → In our example, we have four parity bits and each position can be expressed by four binary digits
- Examples:

Position: 1	⇒	Value: 0001
Position: 2	⇒	Value: 0010
Position: 3	⇒	Value: 0011
Position: 4	⇒	Value: 0100
...		
Position: 12	⇒	Value: 1100

Hamming Code – Sender Procedure

- The sender calculates the parity bits values
- → it performs an **XOR operation** for those positions that contain a **1**
 - In the example it is position 5, position 9 and position 10

Position:	1	2	3	4	5	6	7	8	9	10	11	12
Data to be transmitted:	?	?	0	?	1	0	0	?	1	1	0	0

0101	Position 5
1001	Position 9
XOR 1010	Position 10

=	0110

- The result are the values of the parity bits
 - These are inserted into the transmission

Position:	1	2	3	4	5	6	7	8	9	10	11	12
Data to be transmitted:	0	1	0	1	1	0	0	0	1	1	0	0

Hamming Code – Receiver Procedure (error-free)

- The receiver can **verify** if a bit sequence is correct
 - It performs the same operation as the sender to calculate the parity bits
 - Then, it performs another **XOR operation** of the **calculated** and **received parity bits**

```
Received data: 1  2  3  4  5  6  7  8  9 10 11 12
                0  1  0  1  1  0  0  0  1  1  0  0
```

```

0101   Position 5
1001   Position 9
XOR 1010   Position 10
-----
0110   Parity bits calculated
XOR 0110   Parity bits received
-----
= 0000   => Correct transmission
```

Hamming Code – Receiver Procedure (bit error)

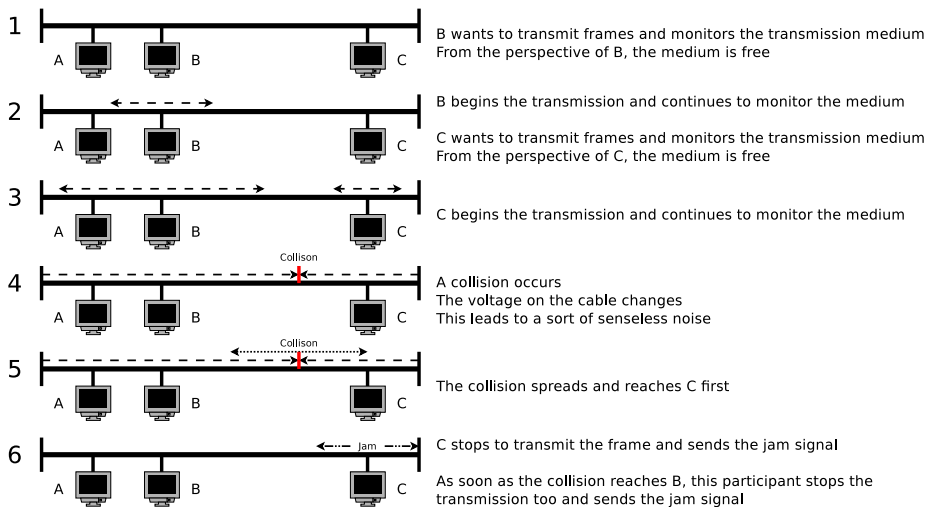
Received data: 1 2 3 4 5 6 7 8 9 10 11 12
 0 1 0 1 1 0 0 0 0 1 0 0

```

    0101   Position 5
XOR 1010   Position 10
-----
    1111   Parity bits calculated
XOR 0110   Parity bits received
-----
= 1001   => Bit 9 is defective!
```

- Possible results of the calculation:
 - Position number of the modified bit
 - 0 if the transmission was correct
- If ≥ 2 bits have been modified, the only statement that can be made is, that bits have been modified at all
 - The positions can not be determined this way

Example of CSMA/CD



Network Size and Collision Detection

- A collision must be detected by the sender
 - It is important that the transmission of a frame is **not completed** when a collision occurs
 - Otherwise, the network device might already be finished with sending the frame and assumes the transmission was successful
- Each frame must have a certain **minimum length**
 - It has to be guaranteed that its **transmission duration** is longer than the **maximum RTT** (round trip time)
 - Remember: The RTT is the time it takes for a frame to travel from one end of the network to the most distant end and return back
 - **This ensures that a collision reaches the sender before its transmission is finished**
 - If a sender detects a collision, it knows that its frame has not arrived correctly at the receiver, and can try the transmission again later

Example: Minimum Frame Length and Collision Detection

- Ethernet specifies a **maximum network size** and a **minimum frame length**
- The **minimum frame length**, where collision detection is still possible, is calculated as follows:

$$P = 2 * U * \frac{D}{V}$$

P = Minimum frame length in bits
 U = Data rate of the transmission medium in bits per second (bps)
 D = Network length in meters
 V = Signal speed on the transmission medium in meters per second)

- Calculation example for 10BASE5 with 10 Mbps and coaxial cables:

- $U = 10 \text{ Mbps} = 10,000,000 \text{ bps}$
- $D = 2,500 \text{ meters}$ (this is the maximum length for 10BASE5)
- $V = \text{speed of light} * \text{velocity factor}$
 - Speed of light = 299,792,458 m/s
 - Velocity factor = 0.77 for coaxial cables
 - $V = 299,792,458 \text{ m/s} * 0.77 \approx 231,000,000 \text{ m/s}$

$$P = 2 * 10 * 10^6 \text{ bps} * \frac{2,500 \text{ m}}{231 * 10^6 \text{ m/s}} \approx 217 \text{ bits} \approx 28 \text{ bytes}$$

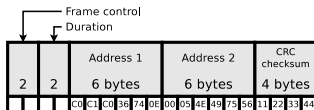
- Outcome: The minimum frame length of 64 bytes for Ethernet is more than enough

WLAN Control Frames (Special Frames) – RTS Frame

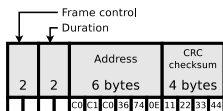
The control frames RTS, CTS and ACK have a different structure compared with data frames

- Length of **RTS frames**: 20 bytes
- With the RTS frame, a station, which wants to transmit frames, **sends a reservation request** for the transmission medium to the Access Point
- 1st address field = MAC address of the Access Point
- 2nd address field = MAC address of the station, which sends the request

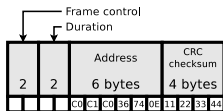
RTS frame



CTS frame



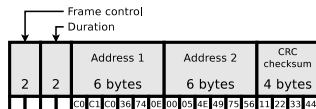
ACK frame



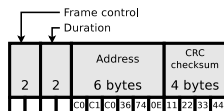
WLAN Control Frames (Special Frames) – CTS Frame

- Length of **CTS frames**: 14 bytes
- With a CTS frame, an Access Point **confirms the reservation request** for the transmission medium
- address = MAC address of the station, which sent the reservation request

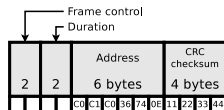
RTS frame



CTS frame



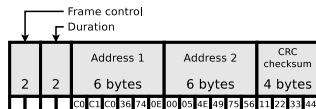
ACK frame



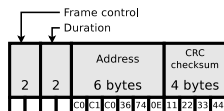
WLAN Control Frames (Special Frames) – ACK Frame

- Length of **ACK frames**: 14 bytes
- With an ACK frame, the receiver **confirms the successful transmission of a frame** at the sender
- address = MAC address of the station, which transmitted the frame successfully

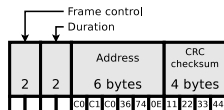
RTS frame



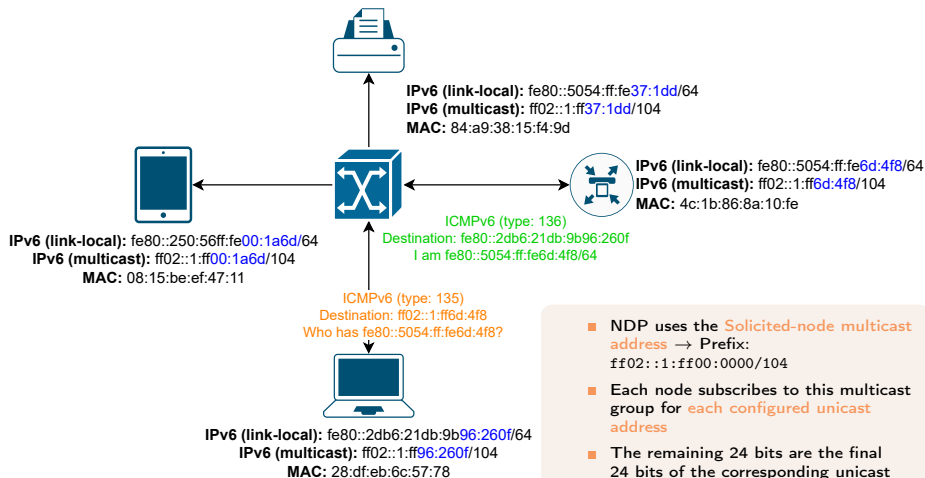
CTS frame



ACK frame



Exercise 5: NDP



- NDP uses the **Solicited-node multicast address** → Prefix:
ff02::1:ff00:0000/104
- Each node subscribes to this multicast group for **each configured unicast address**
- The remaining 24 bits are the final 24 bits of the corresponding unicast address
- Only nodes registered to this address will receive the ICMP message