Karlsruhe Institute of Technology

Institut fuer Technik der Informationsverarbeitung

Dr.-Ing. Jens Becker *(jens.becker@kit.edu)* **M. Sc. Matthias Stammler** *(stammler@kit.edu)*

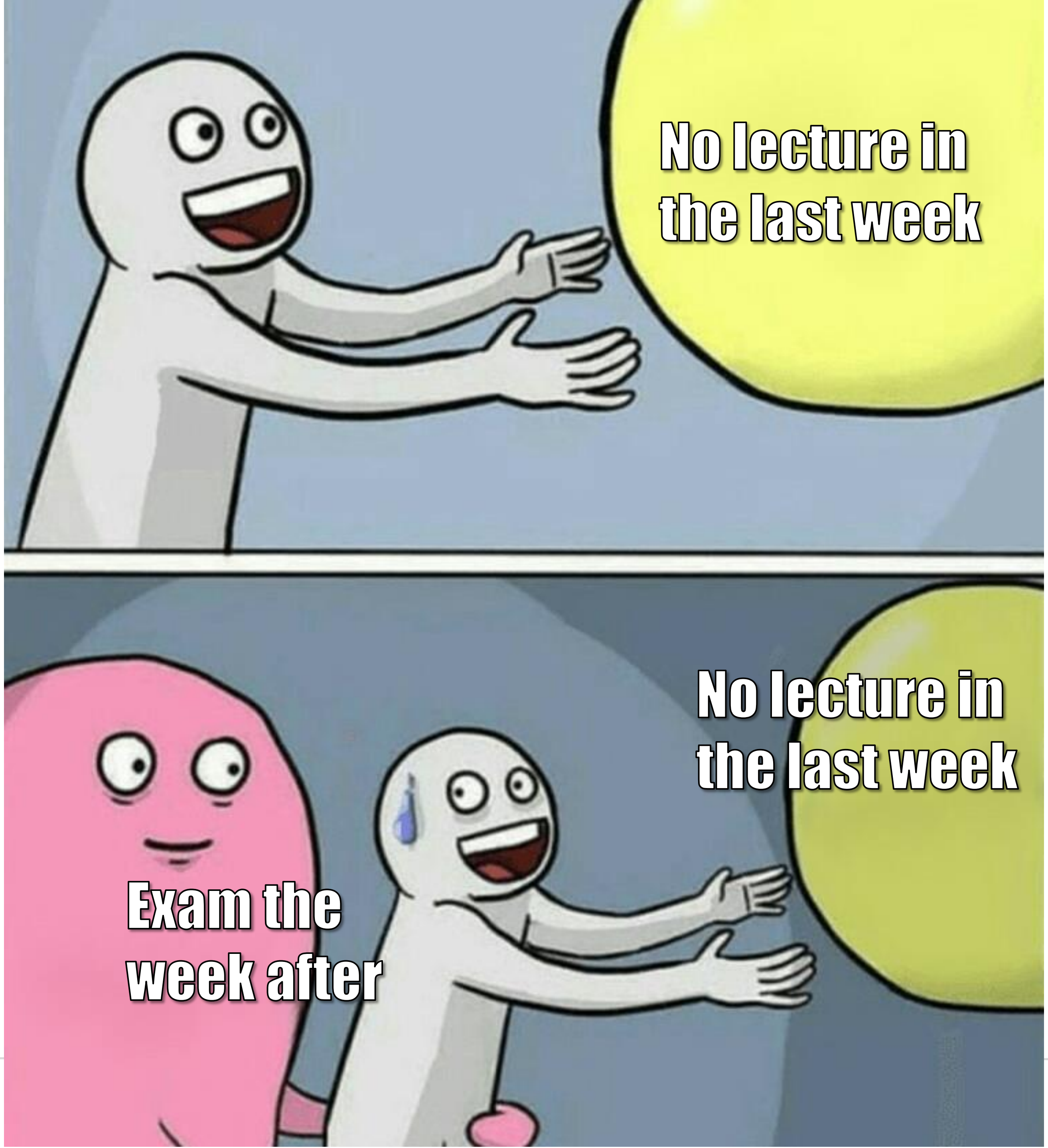
Communication Systems and Protocols

Exercise 6

General Information
Communication Systems and Protocols (Exercises)

Matthias Stammler, M. Sc.
Institut fuer Technik der Informationsverarbeitung (ITIV)
Building 30.10, Room 218
Email: stammler@kit.edu, csp@itiv.kit.edu
Website: https://www.itiv.kit.edu/21_7686.php

Karlsruhe Institute of Technology



Controller Area Network (CAN)

Karlsruhe Institute of Technology

- 1981 developed by Bosch and Intel
- Goal: Cross-linkage of complex controllers and control units
- International dissemination in the automobile area, the sector of home appliance, in textile machines, in medical engineering devices, …
- Low priced bus transceivers through high volume number

CAN Characteristics

- Multi master architecture
- High transmission rate (max. 1 Mbit/s)
- Message distribution system
- Message prioritization
- Network wide data consistency
- Priority controlled and collision free bus access
- Very high failure recognition capability
- Powerful failure treatment
- Node monitoring
- NRZ coding with bit stuffing

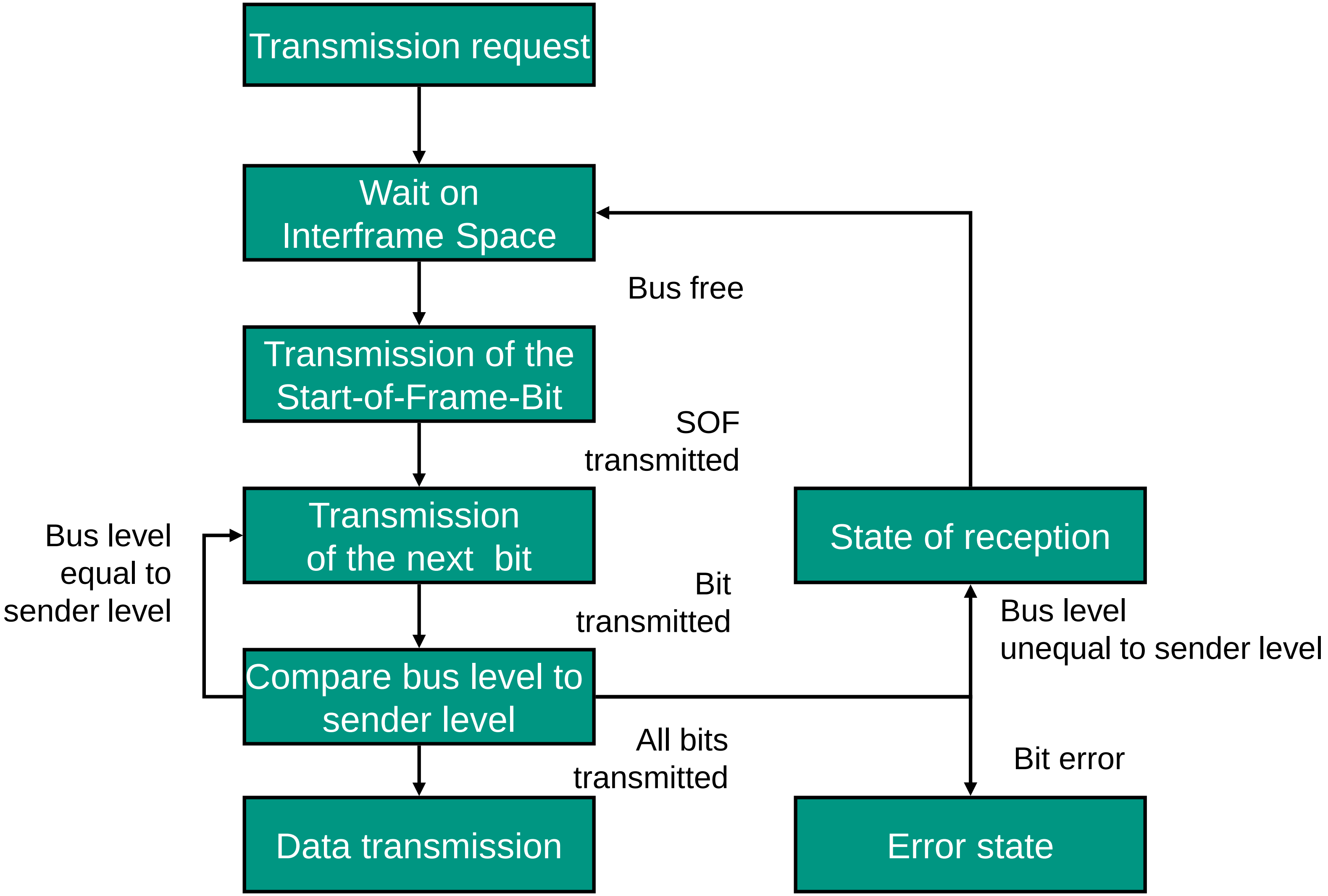
Bus Arbitration

- Participants are connected to the bus via open collector drivers
  - Dominant LOW, logical ‚0'
  - Recessive HIGH, logical ‚1'
    → „Wired-AND"

- Arbitration scheme: CSMA/CR

- Arbitration is done with the help of the identifiers of a message
  - Identifier with the lowest value has got highest priority (dominant ‚0')
  - Every identifier can exist only once within the whole system

Procedure of bit wise bus arbitration

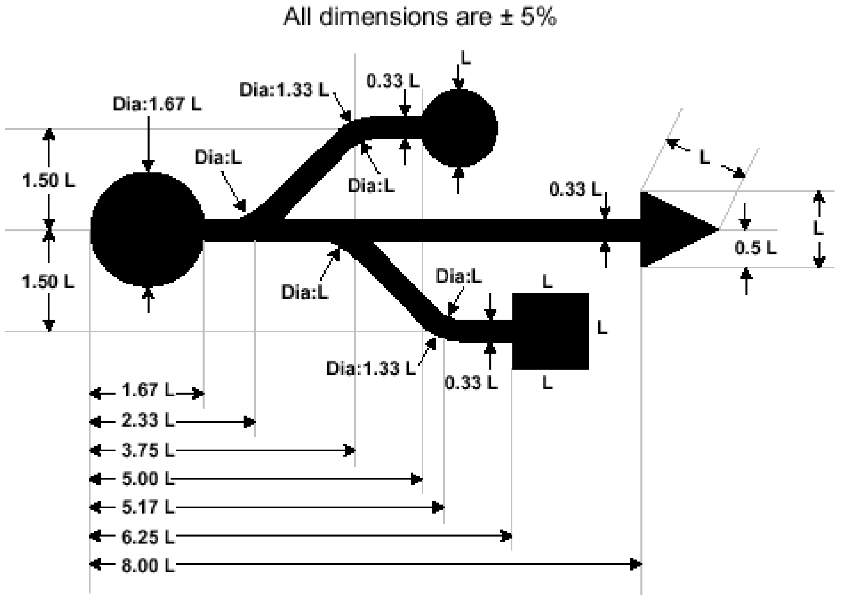


Task 1: CAN bus

Time Allocated **10 min**

Universal Serial Bus (USB) - Overview

- Bus system used to connect peripheral devices like mouse, keyboard and printer to the PC
- Serial bus
  - Hot-plugging, auto configuration
  - At most 127 attached devices possible
  - Tree topology, branching is done in hubs
  - Point-to-point connections between individual devices
  - Central control through one single host (PC)
  - Single Master / multiple slaves
  - Polling is done by host, no arbitration required
  - Different speeds dependent on version

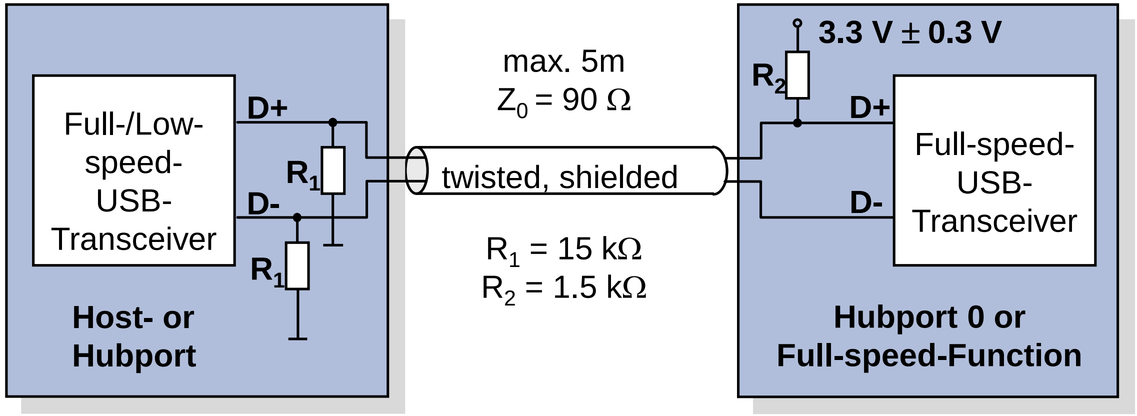


Different Standards

Mode Name	Old Name	Standard	Encoding	Data pairs	Nominal Speed	USB-IF Marketing Name
Low-Speed		USB 1.0	NRZI	1, HDx	1.5 Mbit/s	Basic-Speed USB
Full-Speed					12 Mbit/s	
High-Speed		USB 2.0			480 Mbit/s	Hi-Speed USB
USB 3.2 Gen 1×1	USB 3.0, USB 3.1 Gen 1	USB 3.0	8b/10b	2, FDx	5 Gbit/s	SuperSpeed USB 5Gbps
USB 3.2 Gen 2×1	USB 3.1 Gen 2	USB 3.1	128b/132b	2, FDx	10 Gbit/s	SuperSpeed USB 10Gbps
USB 3.2 Gen 1×2		USB 3.2	8b/10b	4, FDx ×2	10 Gbit/s	N/A
USB 3.2 Gen 2×2			128b/132b	4, FDx ×2	20 Gbit/s	SuperSpeed USB 20Gbps
USB4 Gen 2×1		USB4	64b/66b	2, FDx	10 Gbit/s	N/A
USB4 Gen 2×2			64b/66b	4, FDx ×2	20 Gbit/s	USB4 20Gbps
USB4 Gen 3×1			128b/132b	2, FDx	20 Gbit/s	N/A
USB4 Gen 3×2			128b/132b	4, FDx ×2	40 Gbit/s	USB4 40Gbps

Transmission speed detection

- Devices announce their transmission speed using different pull-up resistors at the data wires
- Example for resistor configuration at 12Mbit/s

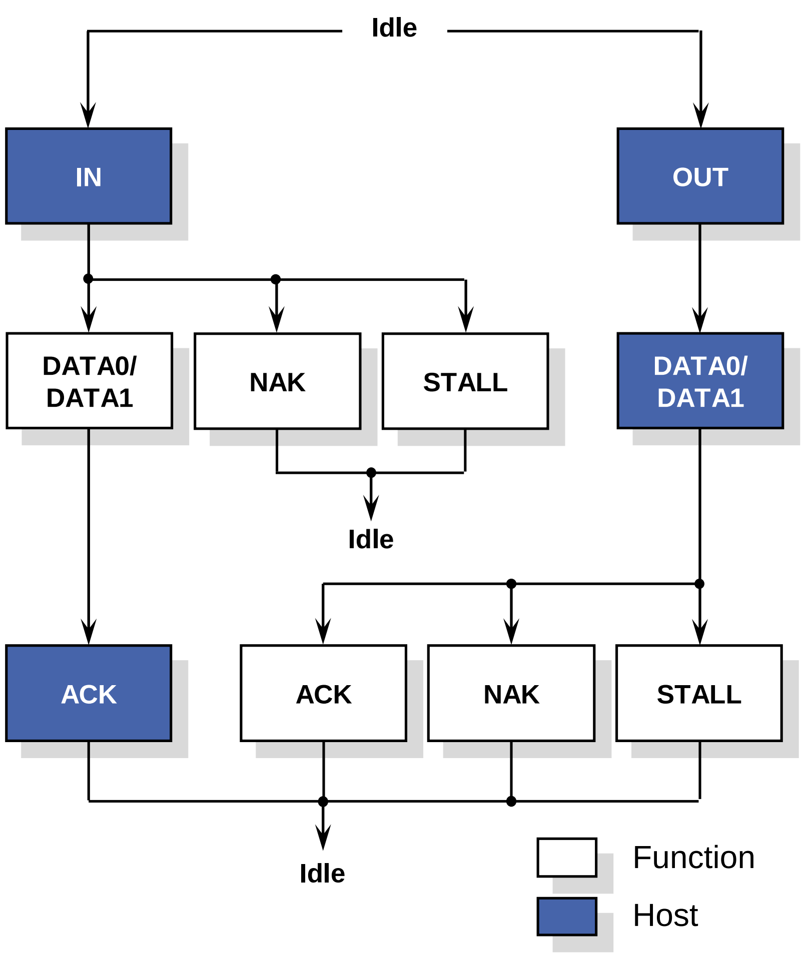


- At 1.5Mbit/s
  - Low-Speed Transceiver
  - R2 at D-
  - Cable does not need to be twisted/shielded
  - Z0 arbitrary
  - Max. 3m wire length
- High-speed devices announce themselves as Full-speed devices (12Mbit/s) first
  - At configuration time, the host is informed about the higher transmission rate possible
  - In normal operation, the resistor is disabled in order to provide symmetrical wire behavior

Transfer process

- The host starts a transfer by sending a corresponding token packet
- Then one ore more data packets are transmitted
- Each packet is acknowledged by a handshake packet
- If errors have occurred during a transmission, no handshake packet is send. The sender has to retransmit the packet
  - For isochronous transfers there is no handshake. Defective data is not transmitted again!

Exemplary Bulk Transfer:



Task 2: Universal Serial Bus (USB)

Time Allocated 10 min

Routing Requirements

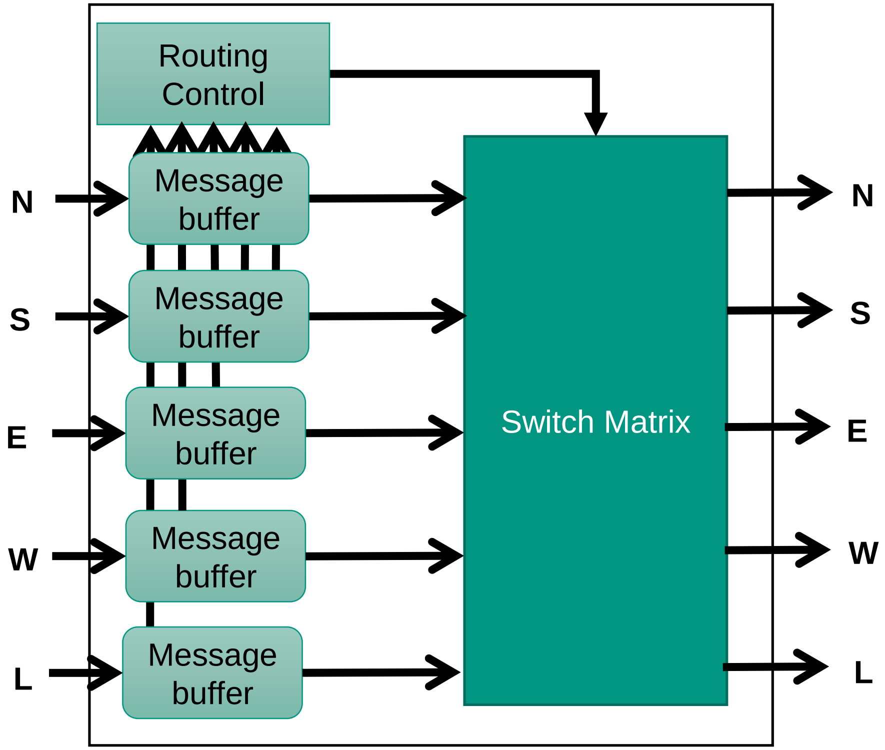
Usually multiple messages m_i are exchanged by multiple node pairs A_j and B_j

Routing goals

- Even Load Distribution over the network
- Minimization of communication path or time (latency)
- Prevention of Deadlocks
- Minimization of Contention (delay of messages due to competing requirements)
- Minimization of Congestions (deletion of messages, for example caused by buffer overflows)

Routing

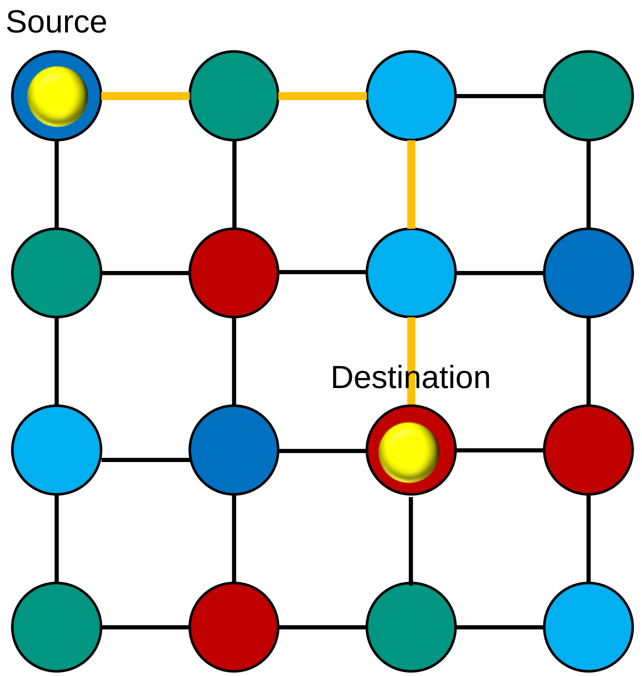
- Routing Unit implements a (pre-)defined behavior
  - connects correct IN/OUT
  - implements the communication paths
- Routing has to be defined somehow
  - Routing algorithms
- Routing Optimization Goals
  - minimal latencies
  - short routing path
  - balanced network load
- Examples of Routing Algorithms
  - X-Y Routing
  - Dijkstra-Algorithm



Routing Schemes

- Routing schemes can be classified into different categories.

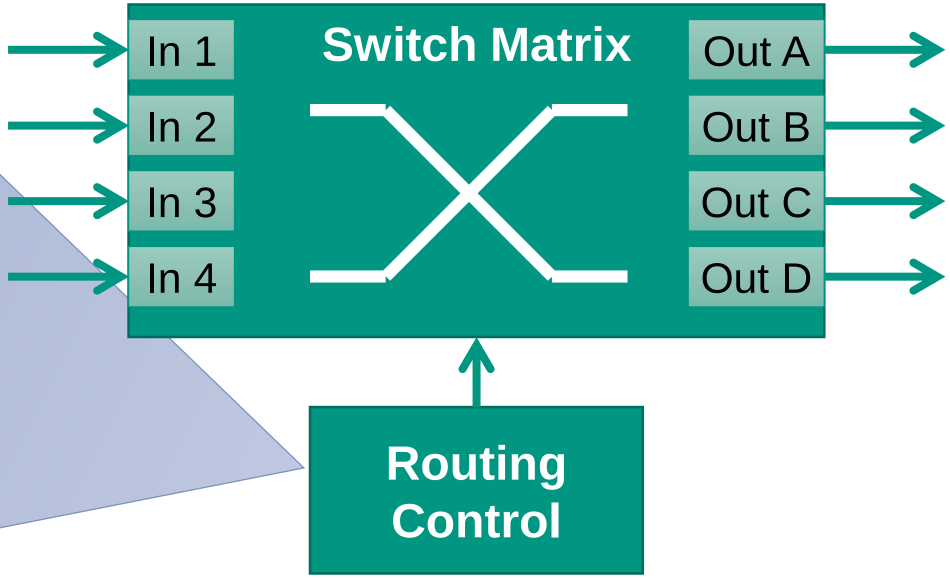
- For example:
  - Static (Deterministic) and Dynamic (Adaptive)
  - Distributed and Source routing
  - Minimal Routing and Non-Minimal Routing



Example for Routing Control – Routing Table

- One row per destination address
- Content stores output port for a given destination
- Table can be static or dynamic
  - Example algorithm: Dijkstra

Destination Address	Output
0x0001	Out C
0x0002	Out C
0x0003	Out A
0x0004	Out D
⋮	⋮
0xFFFE	Out B
0xFFFF	Out C



Dijkstra-Algorithm: Properties

Karlsruhe Institute of Technology

- Presented 1959 by Edsger W. Dijkstra (1930-2002)
- Finds the shortest path to every node from a given starting node
- Application for non-mesh-based networks
  - Distance may not be the only criteria
- Djikstra can compute routes that are optimal regarding given constraints (weights) other than pure Manhattan distance
  - Length
  - Bandwidth
  - Cost
  - …

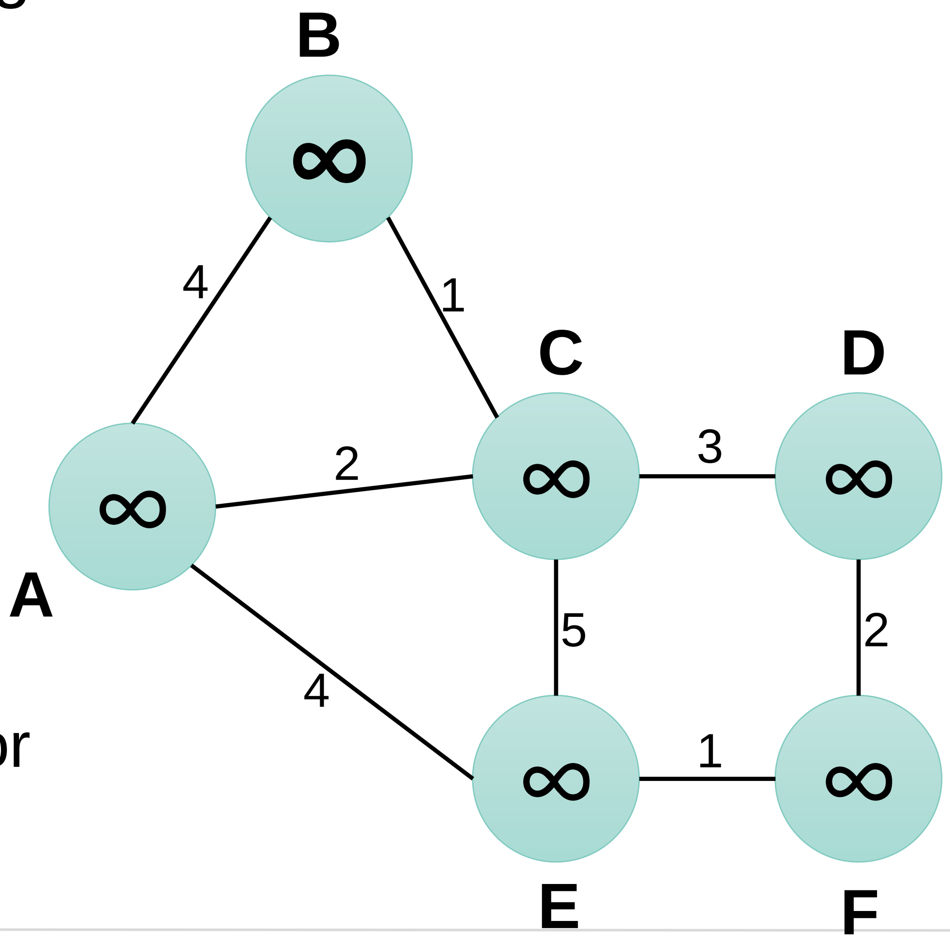
Dijkstra-Algorithm

- Finds the shortest path from a starting node to all other nodes in a known topology

A. Set distance to infinity for all vertices
B. Repeat until all vertices are marked:
   1. Mark the vertex with smallest distance
   2. For all neighbors add current distance and edge weight
   3. If sum is smaller than current distance

      → update distance

      → set marked vertex as predecessor



Task 3: Routing

Time Allocated 10 min

Thank you for your attention