

Vorlesung Internet of Everything

Kapitel 4.1 – Kommunikation

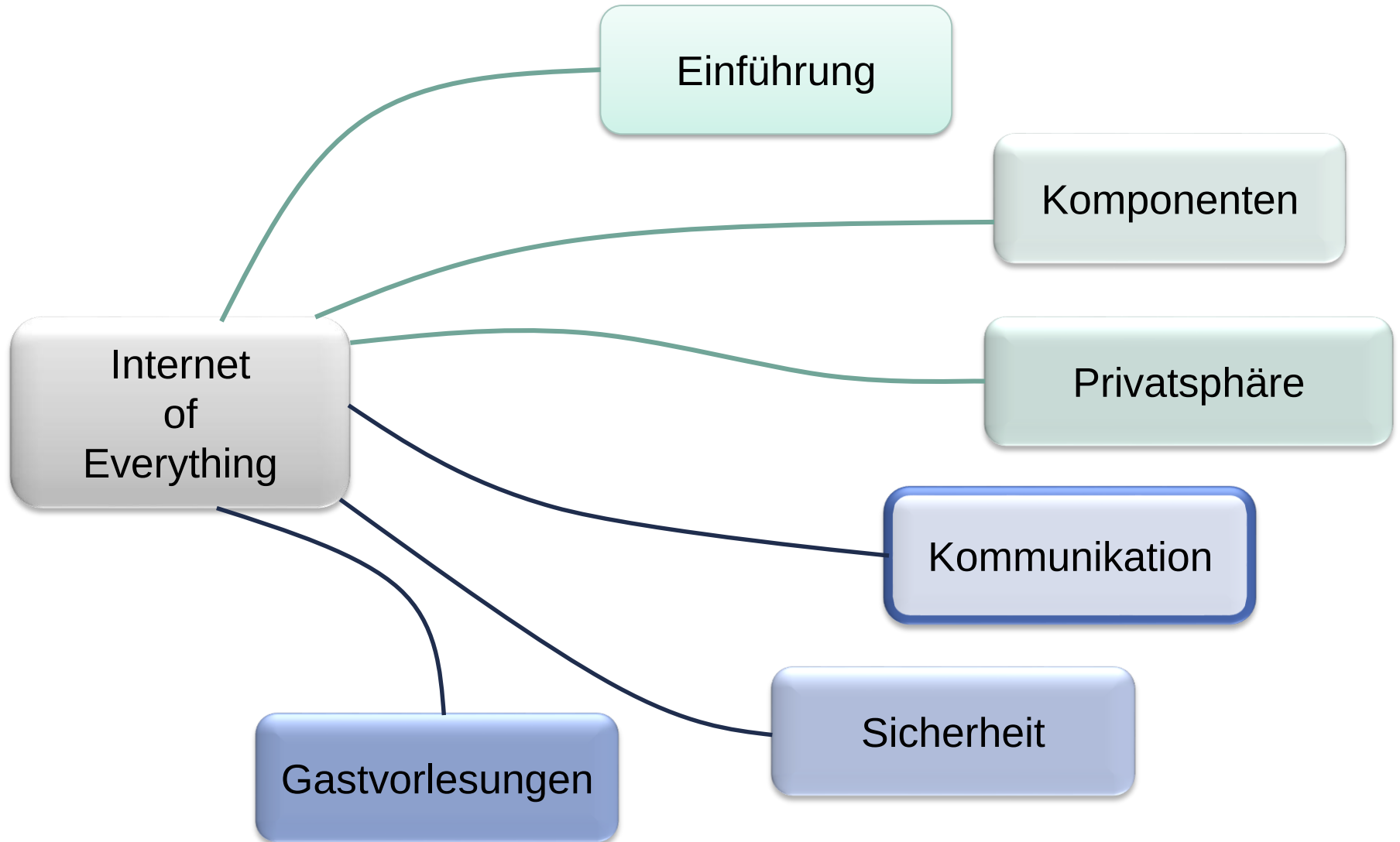
Prof. Dr. Martina Zitterbart, Martin Florian, Markus Jung
[zitterbart, florian, m.jung]@kit.edu

Institut für Telematik, Prof. Zitterbart



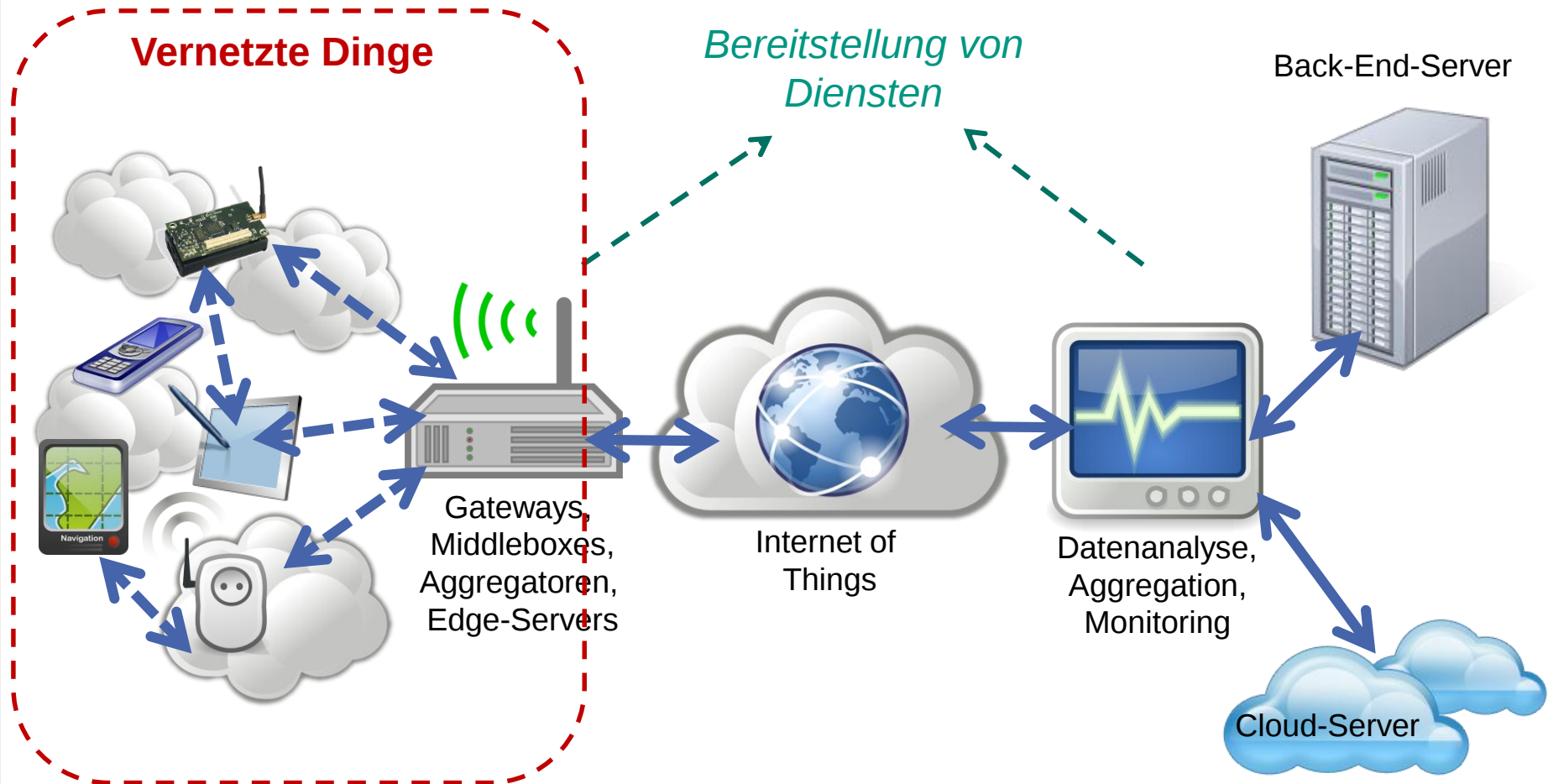
© Peter Baumung

Inhalte der Vorlesung



High-level Kommunikationsarchitektur

■ ... für ein Internet of Everything

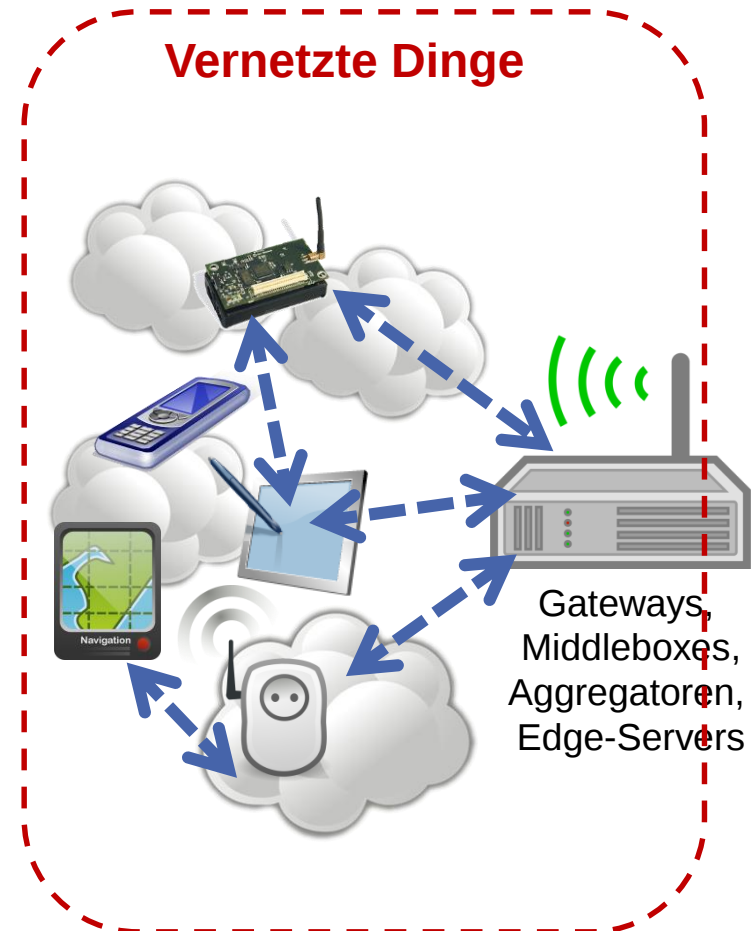


Vernetzte Dinge

- Dinge sind häufig batteriebetrieben
 - Energieverbrauch ist kritisch

- Drahtlose Netze zur Vernetzung der Dinge
 - Kommunikation über mehrere Hops möglich
 - Dinge sind oftmals gleichzeitig Datenquelle/-senke und weiterleitender Knoten
 - Protokolle für den Medienzugriff
 - Routingprotokolle
 - Zuverlässiger Datentransport?
 - Protokolle für den Datentransport

- Anschluss an das Internet
 - „Kompatibilität“ mit Internet-Protokollen

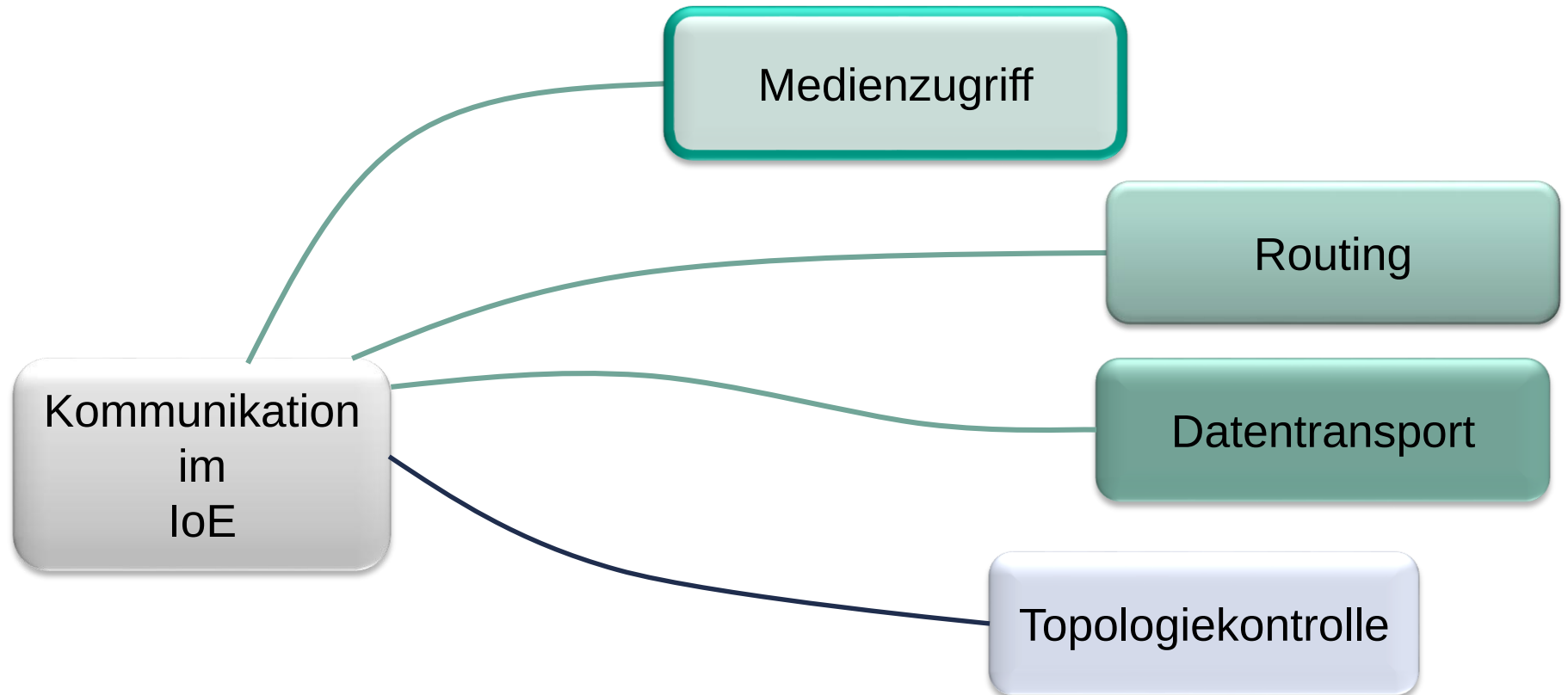


Verschiedene Blickwinkel

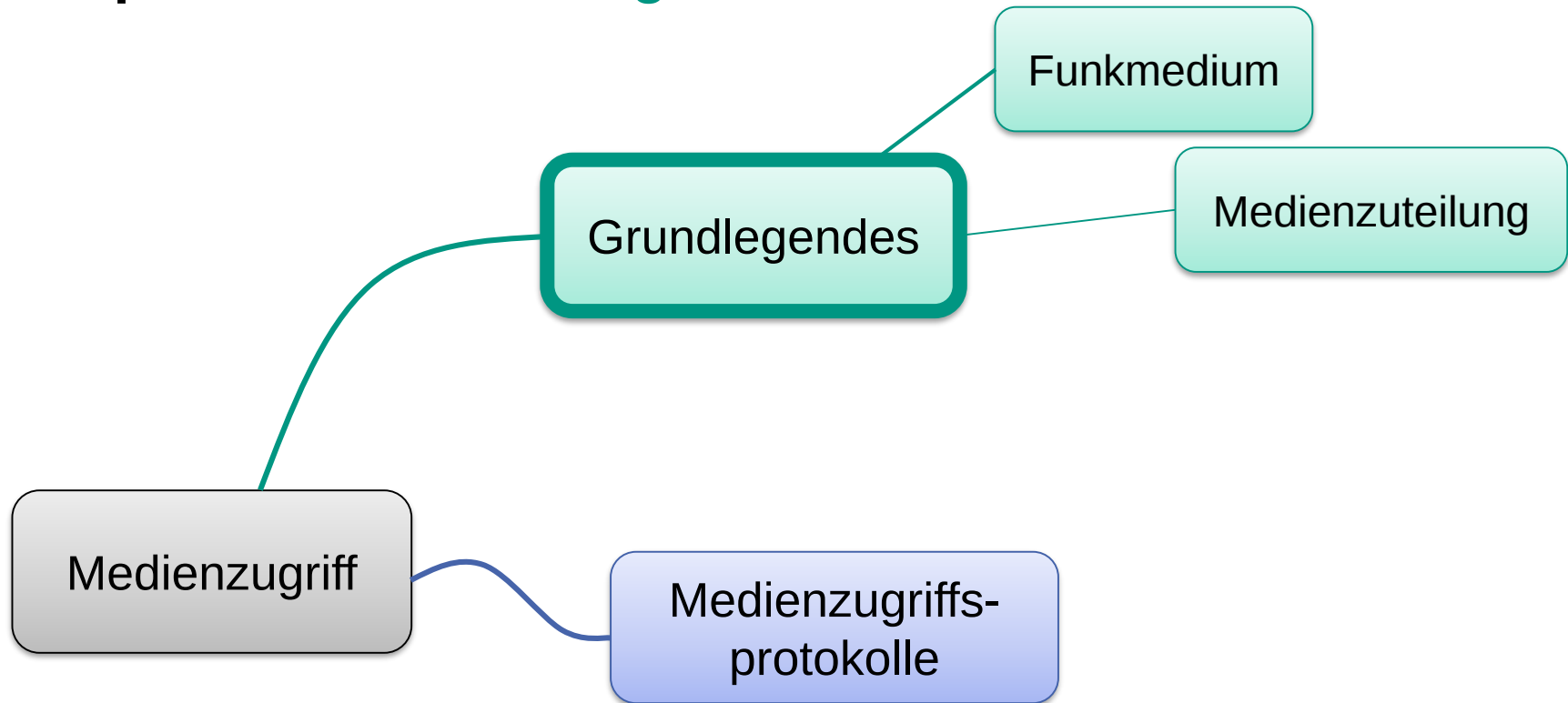
- Viele, viele Forschungsarbeiten
 - ... einige grundlegende Arbeiten werden vorgestellt
- Erste Produkte und Standards
 - ZigBee
 - Von der ZigBee Alliance getrieben 
 - „ZigBee is the language that allows you to control the everyday devices around you“
 - Anwendungsbereiche
 - Smart Homes, Connected Lighting, Utility Industry, Retail Services
 - 6LoWPAN
 - Von der IETF getrieben
 - Anbindung an das Internet steht im Mittelpunkt, basierend auf IPv6
 - ... werden beide in der Vorlesung betrachtet

Cyber Physical Systems (CPS)

- Vernetzte Komponenten mit Sensoren und Aktuatoren die physikalische Prozesse steuern
 - „vernetzte Regelschleifen“
 - Regelung der Heizung basierend auf erfassten Temperaturwerten im Raum
 - Steuerung einer Fabrikanlage
 - Steuerung des Smart Grid
- CPS sehen wir als Bestandteile des Internet of Everything
 - Haben teilweise hohe Anforderungen an Safety
 - Reibungsloses Zusammenspiel zwischen Regelkreis und Kommunikation ist wichtig
 - Geringe Latenzen, Daten in richtiger Reihenfolge ...
- Time Sensitive Networking
 - Aktuelle Standardisierung in der IEEE als Teil von 802.1
 - *Time-synchronized low latency streaming services*
 - Medienzugriff, Ressourcenreservierung, Scheduling ...



Kapitel 4.1: Medienzugriff



Drahtlose Kommunikation

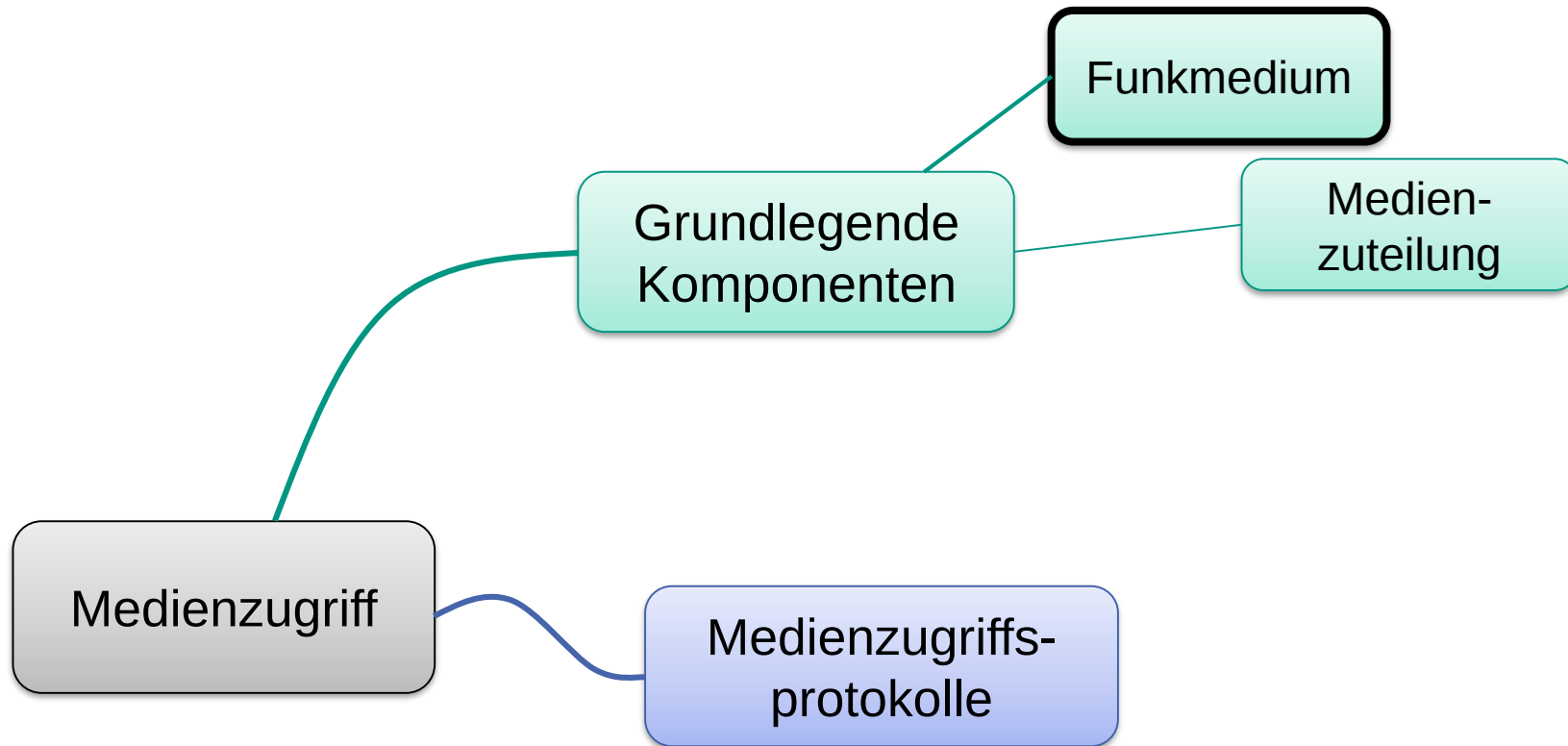
■ Betrachtetes Medium: Funk

- Relativ hohe Unzuverlässigkeit des drahtlosen Mediums
 - Hohe Fehlerraten
 - Geteiltes Medium
 - Zugriff muss geregelt werden – geeignete Verfahren erforderlich
- ... siehe auch Vorlesung Mobilkommunikation



■ Anforderungen

- Hohe Lebenszeit der vernetzten Dinge
 - ... möglichst wenig Energie verbrauchen
- Robustheit
 - Z.B. häufige Topologieänderungen durch „schlafende“ Systeme
- Skalierbarkeit
 - Vernetzung von hunderten, tausenden Dingen
- Selbstorganisation
- Sicherheit und Schutz der Privatsphäre

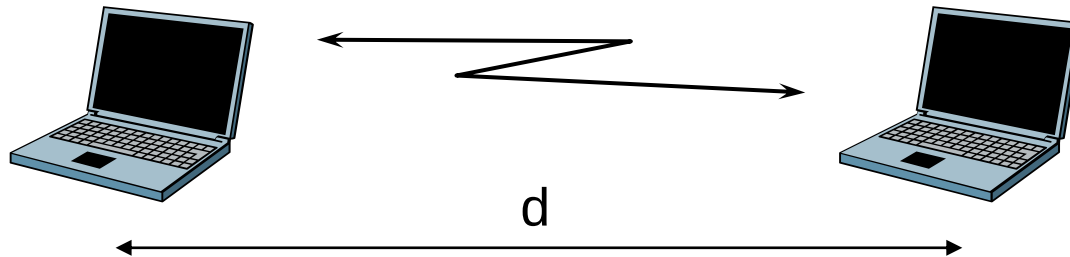


Funkmedium

- Höhere Fehlerraten als drahtgebundene Medien
 - Z.B. Streuung, Beugung, Reflexion, Abschattung
- Niedrigere Datenraten als drahtgebundene Medien
 - Z.B. IEEE 802.15.4 ~250 kbit/s
- Höhere Verzögerungen, größere Schwankungen
 - Übertragungsqualität unterliegt hohen Schwankungen
 - Verbindungsabbrüche
- Geringere Sicherheit gegenüber Abhören, aktive Attacken
 - Luftschnittstelle ist für jeden einfach zugänglich, Basisstationen können vorgetäuscht werden
 - Systeme können korrumpiert, entfernt und zugefügt werden
- Regulierung der Frequenzbereiche
 - Frequenzen müssen koordiniert werden (national/international)
 - Viele sinnvoll nutzbare Frequenzen sind schon vergeben
 - WSAWs nutzen üblicher Weise das ISM Band (~ 2.4 GHz)
- Unidirektionale Links

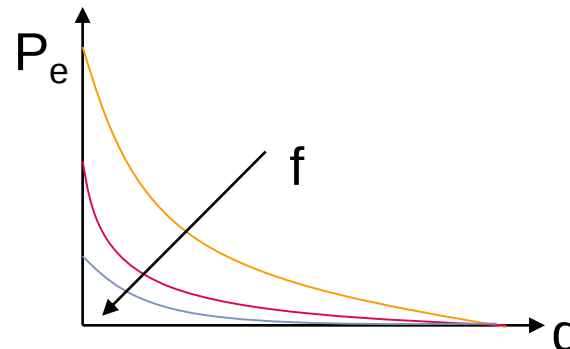
Signalausbreitung

- Signalausbreitung im idealen Fall
 - Signalausbreitung wird nicht durch Hindernisse beeinflusst



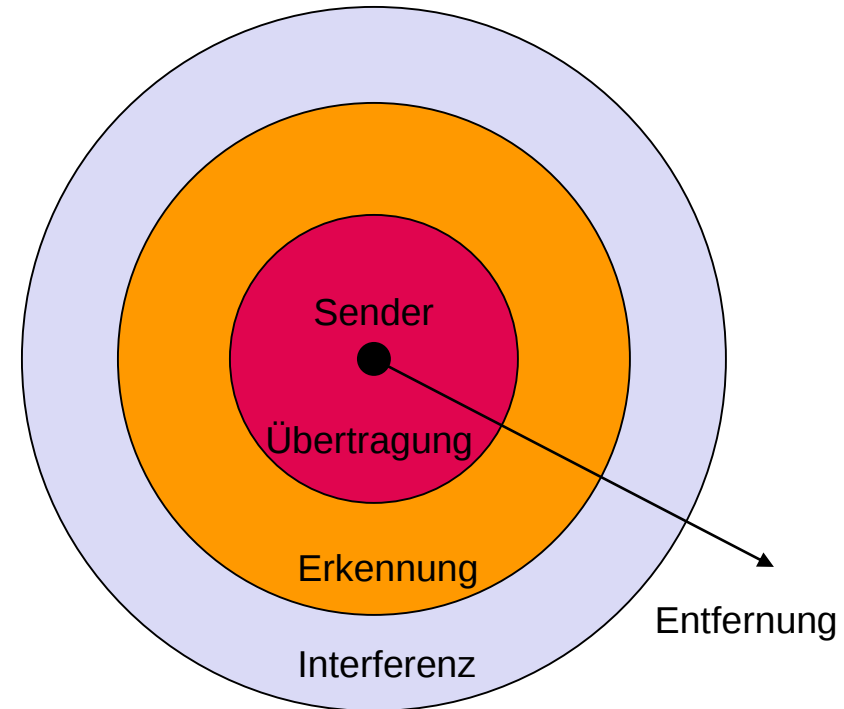
- Mittlere Empfangsleistung nimmt mit $1/d^2$ ab
 - d = Entfernung zwischen Sender und Empfänger
 - P_s, P_e : Sende- bzw. Mittlere Empfangsleistung
- Proportionalitätsfaktor hängt von Frequenz ab

$$P_e \sim \frac{1}{d^2} P_s$$



■ Unterschiedliche Bereiche der Signalausbreitung

- Übertragungsbereich
 - Kommunikation möglich
 - Niedrige Fehlerrate
- Erkennungsbereich
 - Signalerkennung möglich
 - Keine Kommunikation möglich
- Interferenzbereich
 - Signal kann nicht detektiert werden
 - Signal trägt zum Hintergrundrauschen bei
- Bereiche sind nicht scharf gegeneinander abgegrenzt!
 - Unit-Disc Modelle sind idealisiert, aber nicht unbedingt realistisch
 - Umwelteinflüsse (z.B. Wetter, Personen) beeinflussen Bereiche



Semi-Broadcast-Medium

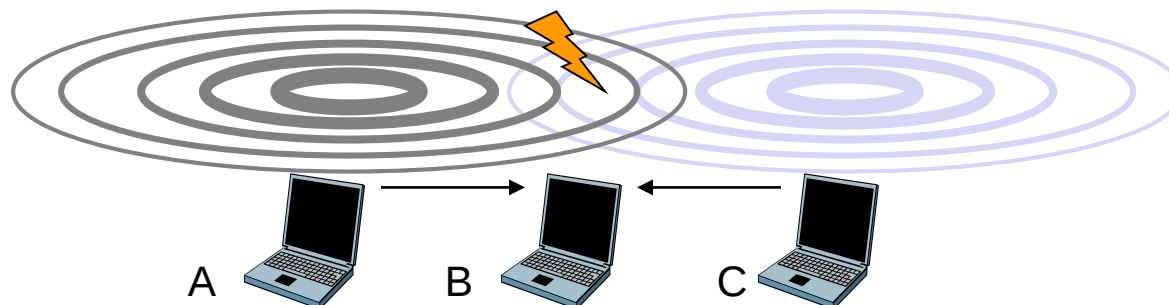
- Drahtgebundene Netze (z.B. Ethernet)
 - Systeme sind über **Broadcast-Medium** verbunden
 - Systeme hören gesendete Dateneinheiten aller Systeme im Netz
 - Kollisionserkennung durch Sender möglich
 - CSMA/CD-basierte Verfahren für Medienzuteilung möglich

- Drahtlose Netze
 - Systeme sind über **Semi-Broadcast-Medium** verbunden
 - Systeme hören nur Dateneinheiten von Systemen in Reichweite
 - Sender kann Kollision nicht erkennen, Kollision tritt beim Empfänger auf
 - Problem der versteckten und ausgelieferten Endsysteme
 - CSMA/CD-basierte Verfahren für Medienzuteilung nicht einsetzbar

Versteckte Endsysteme

■ Beispielsituation

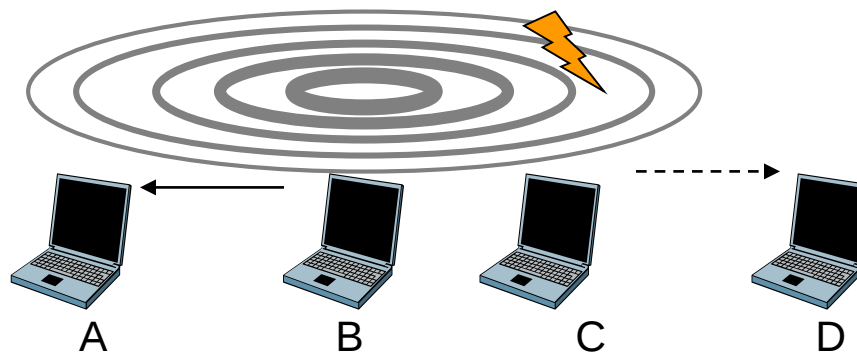
- System A sendet zu System B
- System C kann Signale von System A nicht mehr empfangen
- System C will zu System B senden
 - Medium ist für System C frei (Carrier Sense versagt)
- Kollision bei System B
 - System A kann dies nicht erkennen (Collision Detection versagt)
- System A ist „versteckt“ für System C



Ausgelieferte Endsysteme

■ Beispielsituation

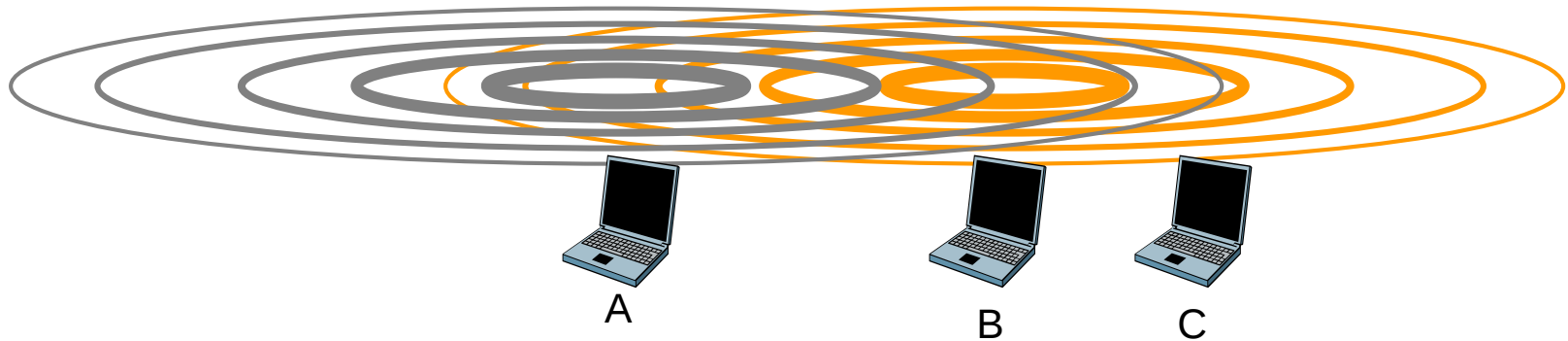
- System B sendet zu System A
- System C will zu System D senden
- System C muss warten
 - Carrier Sense signalisiert ein „besetztes“ Medium
- da System A aber außerhalb der Reichweite von System C ist, ist dies unnötig
- System C ist System B „ausgeliefert“



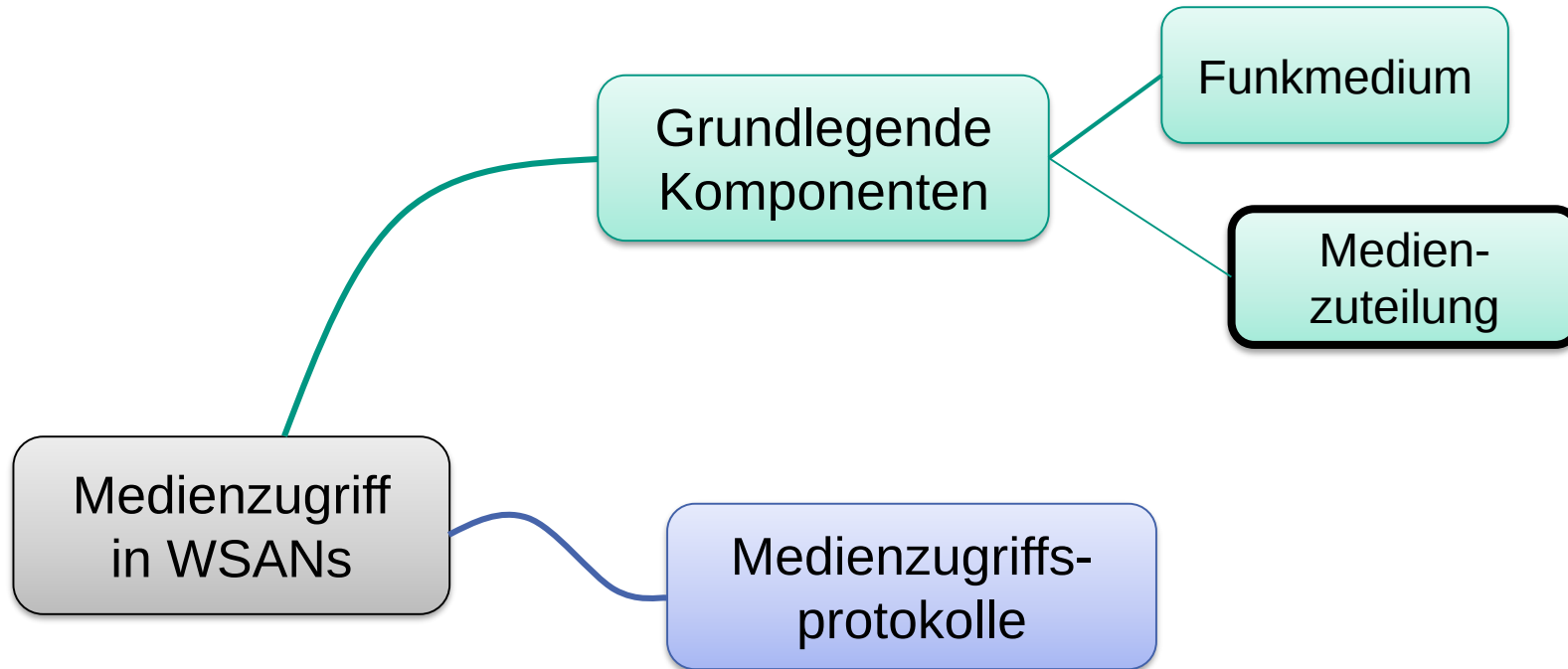
Nahe und ferne Endsysteme

■ Beispielsituation

- Systeme A und B senden, System C soll empfangen
 - Signalstärke nimmt quadratisch mit der Entfernung ab
 - Signal von System B „übertönt“ das von System A
 - System C kann System A nicht hören



- Problem muss bei der Konzipierung eines drahtlosen Netzes beachtet werden
 - Je nach Art der verwendeten Mechanismen ist exakte Leistungskontrolle erforderlich, um „übertönen“ zu vermeiden.



Medienzuteilung

- Grundlegende Einordnung der hier betrachteten Verfahren
 - Verwendung von **Zeitmultiplex**
 - **Konkurrierende** Verfahren
 - Evtl. in Kombination mit fest zugeteilten Zeitschlitz

- Besonders zu beachten
 - Energieverbrauch
 - Latenz der Kommunikation

Energieverbrauch

■ Ausgangssituation

- Sensorknoten sind häufig Batteriebetrieben
- **Energie** ist wichtige, aber sehr beschränkte Ressource

■ Energie sparen

- ... den **Funktransceiver** möglichst häufig deaktivieren
 - „schlafen legen“
- Problem
 - Wann muss ein System empfangsbereit sein? Wenn es schläft kann es keine Daten empfangen

■ Beschränkung

- **Kein Duplex-Betrieb** unterstützt
 - Gleichzeitiges Senden und Empfangen i.d.R. nicht möglich

Beispiele für unnötigen Energieverbrauch

■ Kollisionen beim Medienzugriff

- Mehrere Systeme senden gleichzeitig
- Überlagerung der Dateneinheiten
- Dateneinheiten vom Empfänger nicht dekodierbar
- Sendewiederholungen erforderlich

→ Kollisionsvermeidung

■ Unnötiges Lauschen (Idle Listening)

- System wartet auf den Empfang einer Dateneinheit obwohl niemand an dieses System sendet

■ Mithören (Overhearing)

- System empfängt Dateneinheit, die nicht an es gerichtet ist

→ Duty-Cycling

Kollisionsvermeidung

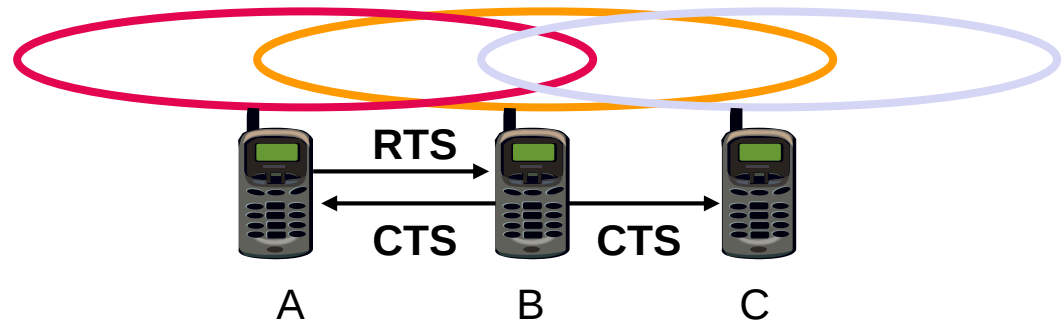
- Separate Signalisierung
 - Out-of-Band Signalisierung
 - Auf separatem Kanal
 - Reservierung fester Zeitschlitz
 - In-Band-Signalisierung
 - Auf gleichem Kanal

- Beispiel In-Band-Signalisierung
 - Multiple access with collision avoidance (MACA)
 - Sender sendet kurze Request to Send (RTS) Dateneinheit
 - Empfänger antwortet mit Clear to Send (CTS) Dateneinheit
 - Sender sendet Daten
 - Während Handshake-Phase können noch Kollisionen auftreten!
 - Aber mit geringerer Wahrscheinlichkeit, da RTS/CTS sehr klein sind

RTS-CTS Handshake

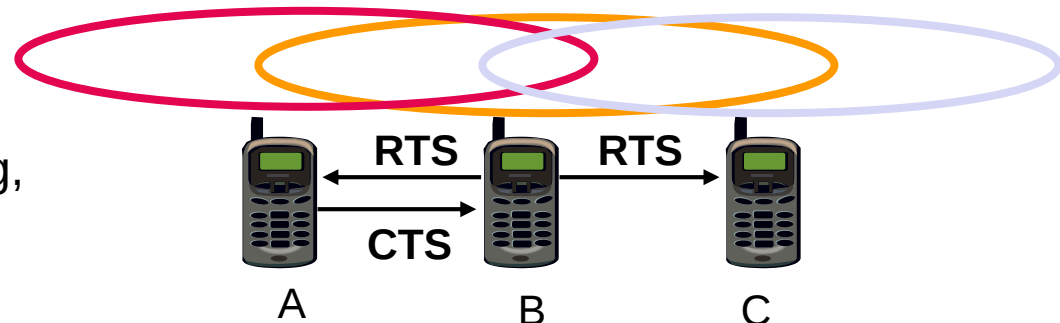
■ Vermeidung des Problems „versteckter“ Endsysteme

- A und C wollen zu B senden
- A sendet zuerst RTS
- B antwortet mit
 - CTS teilt Dauer der Belegung mit
- C wartet, da es das CTS von B hört



■ Vermeidung des Problems „ausgelieferter“ Endsysteme

- B will zu A, C irgendwohin senden
- C wartet nicht mehr unnötig, da es das CTS von A nicht empfängt



Duty-Cycling

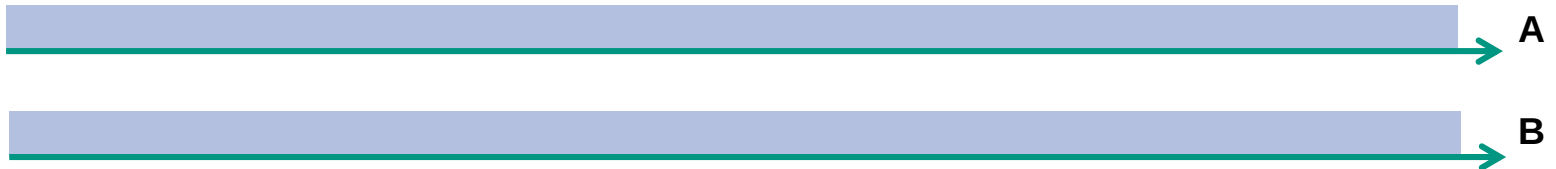
- Zielsetzung
 - Energieverbrauch reduzieren

- Idee: Funktransceiver so oft wie möglich deaktivieren
 - Reduziert **Overhearing** und **Idle Listening**
 - Erhöht **Latenz** der Übertragung!

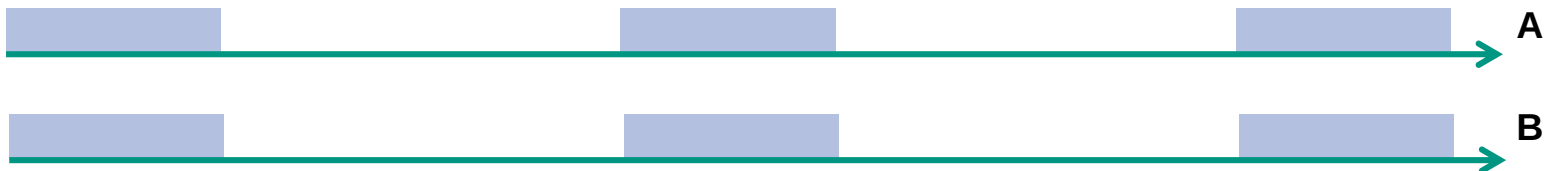
- Funktransceiver bei Bedarf aktivieren um drahtloses Medium auf Aktivität zu prüfen / eigene Übertragungen durchzuführen
 - Sensorknoten aktivieren Funktransceiver koordiniert (gleichzeitig)
→ **synchrones** Duty-Cycling
 - Erfordert Zeitsynchronisation zwischen den Knoten
 - Sensorknoten aktivieren Funktransceiver ohne Koordination
→ **asynchrones** Duty-Cycling

Beispiel: Keine Aktivität

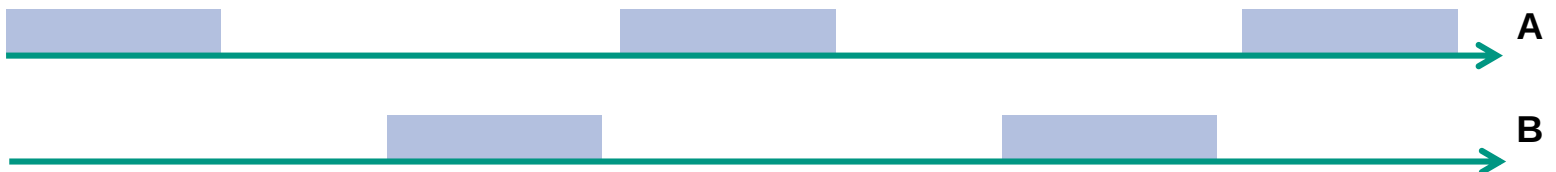
Ohne
Duty-Cycling



Synchrones
Duty-Cycling



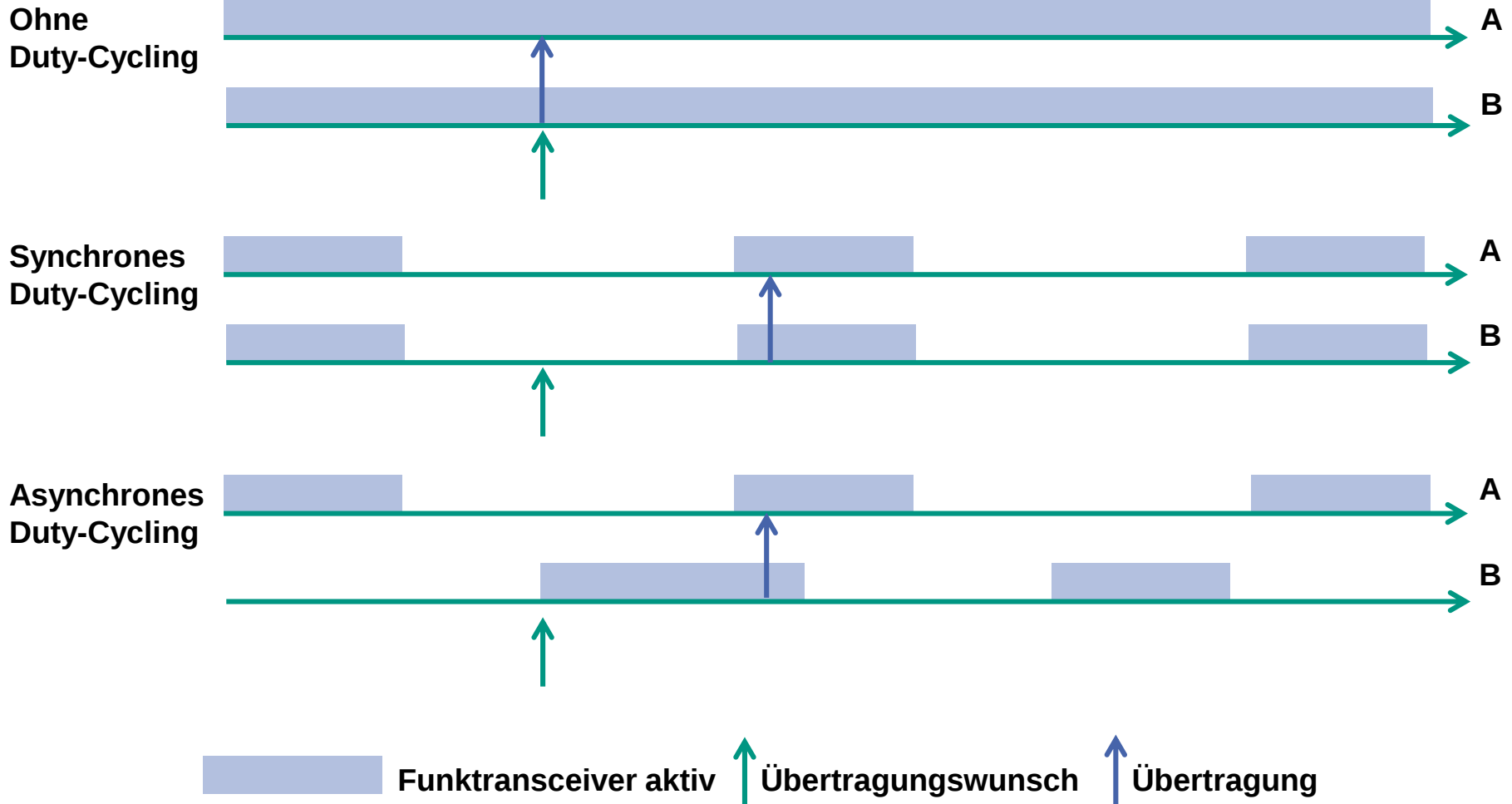
Asynchrones
Duty-Cycling



 Funktransceiver aktiv

Beispiel: Übertragung B → A

Zu beachten: Duty-Cycling verursacht zusätzliche Latenz



WLAN und Bluetooth im Internet of Everything?

■ WLAN (IEEE 802.11)

- Hoher Energiebedarf durch
 - Kollisionen
 - Idle Listening
 - Overhearing

■ Bluetooth (IEEE 802.15.1)

- Langsamer, aufwändiger Synchronisationsprozess zum Verbindungsaufbau (Slow Frequency Hopping)
 - Verbindungsverwaltung kompliziert und teuer
- Synchronisation mehrerer Piconetze problematisch (Scatternetze)
 - Häufige Datenverluste beim Überbrücken von Piconetzgrenzen
 - Durch mehrere Radio-Interfaces behebbar
- Darüber hinaus
 - Bluetooth erfordert Zertifizierung bei Einsatz in kommerziellem Umfeld



[LeDB03]

Bluetooth Smart (Bluetooth 4.0)

- Enthält die folgenden Protokolle
 - Bluetooth classic, Bluetooth high speed und **Bluetooth low energy**

- Einige Eigenschaften von Bluetooth low energy
 - Reichweite: ca. 150 m
 - Datenrate: 1 Mbit/s, optimiert für kleine Portionen von Daten (8-27 Byte)
 - Kein Streaming, nicht für Filetransfer optimiert
 - Gut für Übertragung von Zustandsinformation (Zeit, Temperatur ...)
 - Senden von Daten nur in sehr kurzen Zeitspannen, sonst Funktransceiver aus
 - Bei klassischem Bluetooth war Audio Übertragung zentraler Use-Case → dauerhafte Übertragung
 - Dateneinheiten im Vergleich zu klassischem Bluetooth deutlich kleiner
 - Adaptives Frequenzsprungverfahren im 2,4 GHz-Band
 - Relativ viel Intelligenz im Controller, z.B. Duty-Cycling
 - Lebenszeit der Geräte im Vergleich zu “normalem” Bluetooth deutlich erhöht

Bluetooth Smart

- Weitere Eigenschaften von Bluetooth low energy
 - Low energy Gerät versucht nicht dauerhaft neue Geräte zu finden bzw. Beacons zu senden
 - Übernimmt der Master → Mehr Energiebedarf beim Master
 - Spezielle Low Energy (LE) Profile für energie-effizienteren Betrieb
 - Z.B. HRP – Heart Rate Profile
 - Eingebaute Sicherheitsfunktionalität, AES-128-CCM Verschlüsselung
 - Lizenzfreier Betrieb
 - Optimierte Hardware gegenüber klassischem Bluetooth
 - Deutlich geringere Leistungsaufnahme (0,01 – 0,05W vs. 1W)
 - Kürzere Übertragungsdauer pro Dateneinheit (3ms vs. 100ms)
 - Geringere Latenz (6ms vs. 100ms)
- ... Konkurrenz für ZigBee

→ Insgesamt energie-effizienterer Betrieb möglich

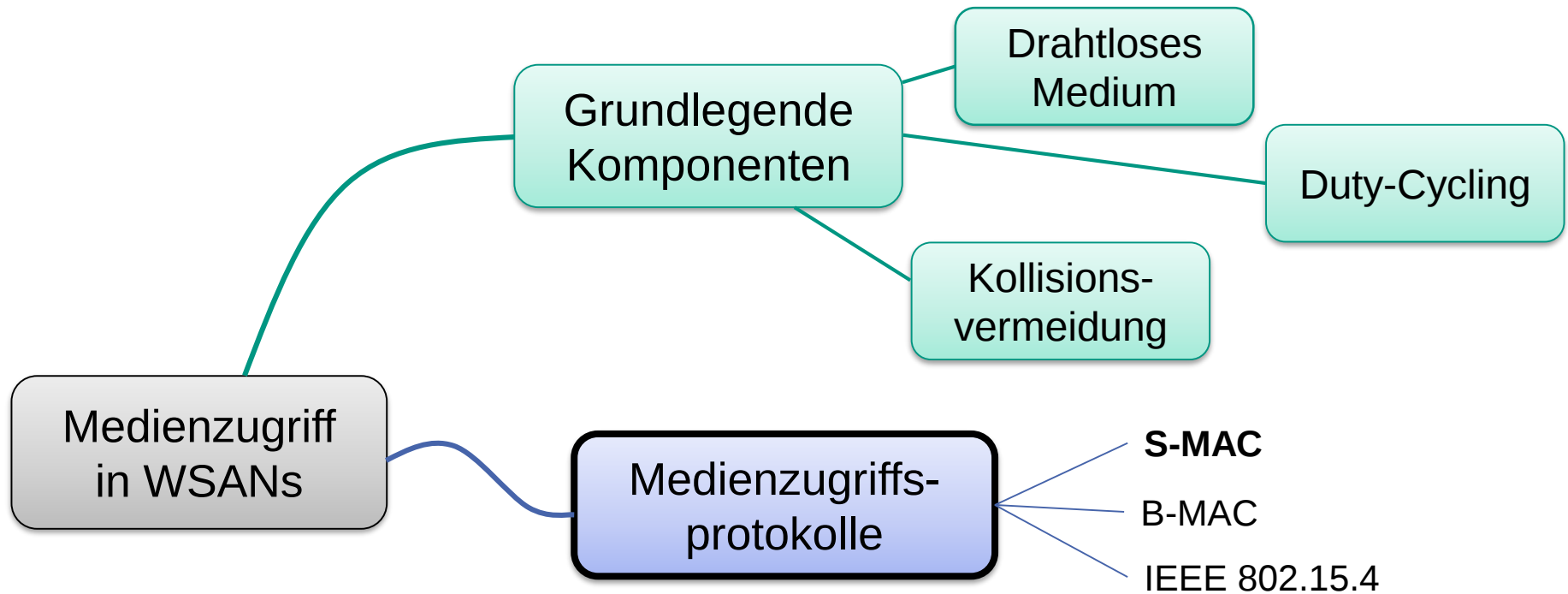
Medienzugriffsprotokolle aus dem akademischen Bereich

Beispiele „klassischer“ MAC-Protokolle

- Entwickelt für drahtlose Sensornetze

- **S-MAC** (Sensor MAC)
 - Entwickelt 2002 von Wei Ye, John Heidemann und Deborah Estrin an der University of Southern California
 - Inspirierte viele der späteren MAC-Protokolle für drahtlose Sensornetze
 - Bisher nicht im praktischen Einsatz, Interesse eher akademisch
 - Zeitliche Synchronisation der Knoten

- **B-MAC** (Berkley Media Access Control)
 - Entwickelt 2004 von Joseph Polastre, Jason Hill & David Culler an der University of California, Berkeley
 - Periodische Prüfung des Kanals
 - Keine zeitliche Synchronisation der Knoten



■ Zielsetzung

- Energiebedarf gering halten, hierzu
 - Vermeiden von unnötigem Lauschen (idle listening)
 - Vermeiden von Kollisionen
 - Vermeiden von Overhearing
- Gute Skalierbarkeit
- Autokonfiguration

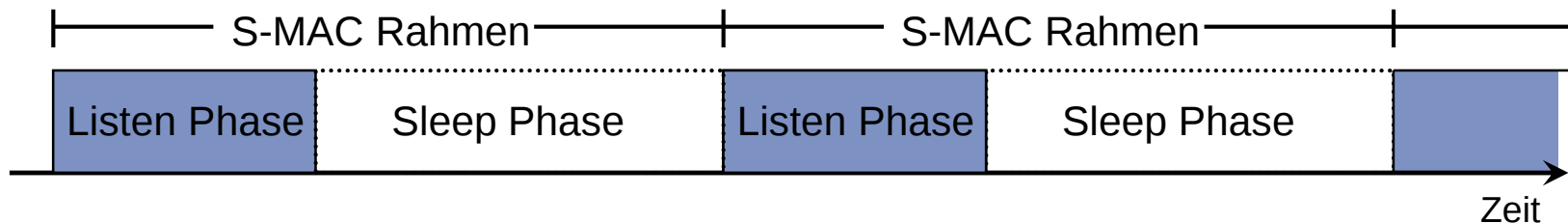
■ Beim Entwurf weniger berücksichtigt

- Fairness pro System
- Latenz

Grundlegendes Prinzip

- Idee
 - Koordiniertes Schlafen vermeidet „idle listening“
 - Systeme erwachen gleichzeitig zur Kommunikation (rendezvous-point)
- Erforderlich
 - Zeitliche Synchronisation der Systeme
 - Hierzu Diskretisierung der Zeitachse in Rahmen fester Länge
- Jeder Rahmen besteht aus 2 Phasen fester Länge
 - Listen Phase: Synchronisation + Datenaustausch anstoßen
 - Sleep Phase: System schläft oder Datenaustausch
 - Ermöglicht Energie zu sparen

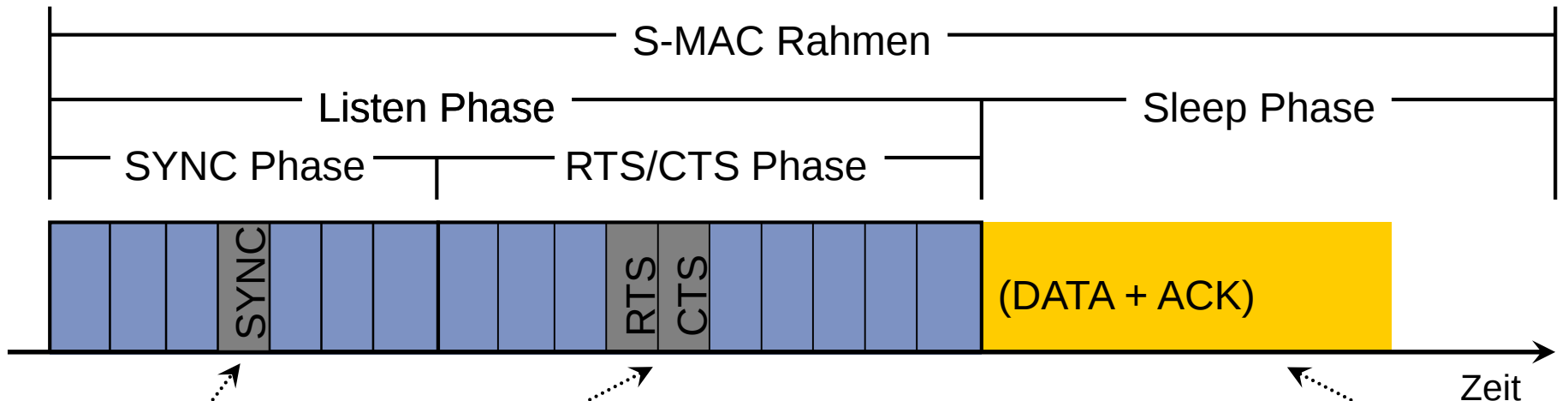
■ Grundlegendes Schema



Listen- & Sleep Phase

- Strukturierung der Phasen
 - Listen Phase in Zeitschlitz gegliedert
 - Pro Zeitschlitz maximal eine Dateneinheit (SYNC, RTS, CTS)
 - Konkurrierender Zugriff auf Zeitschlitz (=: contention)
 - Sleep Phase ohne Gliederung
- Listen Phase: System ist im Wachzustand
 - Sync Phase: Synchronisation durch SYNC Dateneinheit
 - Sende SYNC Dateneinheit in zufällig gewähltem Zeitschlitz
 - SYNC Dateneinheit enthält Zeitspanne bis zum Beginn der nächsten Sleep Phase
 - RTS/CTS Phase: Datenaustausch anstoßen
 - Wähle Zeitschlitz für RTS zufällig, dann Carrier Sense + RTS/CTS
 - Maximal eine Dateneinheit pro Rahmen kann übertragen werden
- Sleep Phase: Datenaustausch oder System schläft
 - Entfällt, falls Datenaustausch angestoßen wurde
 - Sonst: Funkschnittstelle temporär deaktiviert

S-MAC Rahmen



Dateneinheit wird während der Sleep Phase übertragen
Sleep Phase verkürzt sich entsprechend

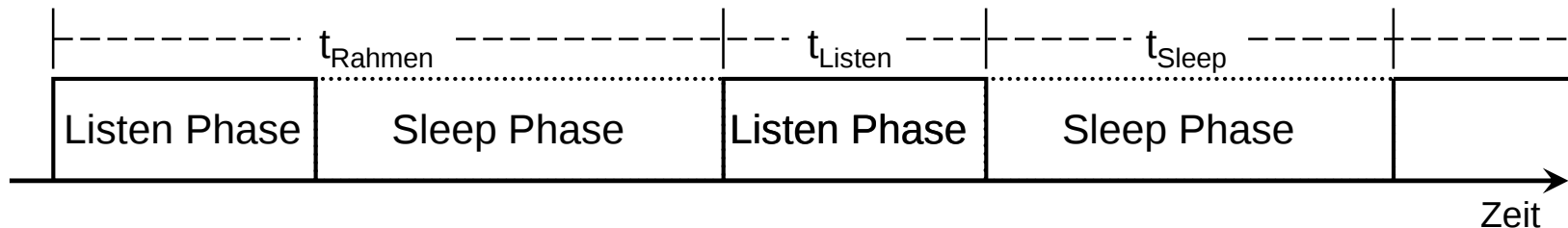
RTS in zufälligem Zeitschlitz gesendet
Empfänger antwortet mit CTS

SYNC in zufälligem Zeitschlitz gesendet
enthält Zeitspanne bis zum Beginn der nächsten Sleep Phase

Wichtige Protokollparameter

- Listen Phase: Zeitdauer der Listen Phase (t_{Listen})
 - Muss ausreichend lang sein, um SYNC, RTS, CTS innerhalb geeigneter Contention Phasen übertragen zu können
 - Länge ergibt sich aus Parametern der MAC und PHY Schicht z.B. Datenrate, etc.
 - Ist also nicht frei wählbar

- Duty Cycle: Zeitanteil, den ein System im Wachzustand verbringen soll
 - Duty Cycle := t_{Listen} / t_{Rahmen}
 - Frei wählbar



Zusammenhang wichtiger Parameter

■ Ableitbare Parameter: Rahmenlänge & Sleep Phase

■ Werden aus Duty Cycle und Listen Phase berechnet

■ Rahmenlänge: Zeitdauer eines Rahmens (t_{Rahmen})

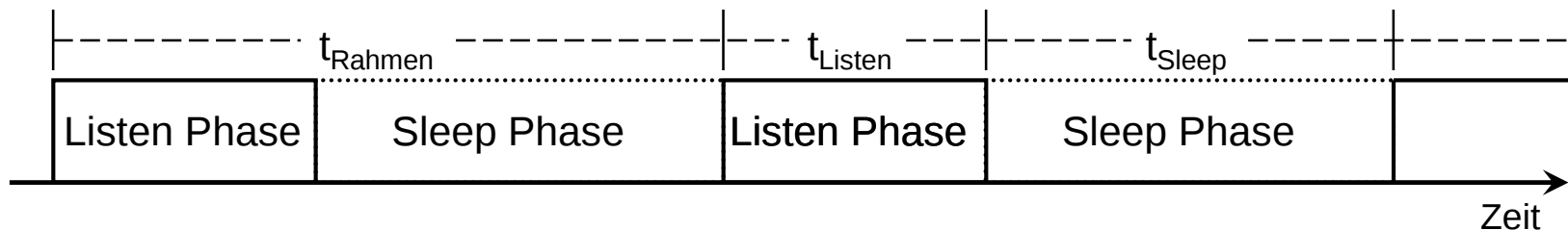
■ $t_{Rahmen} = t_{Listen} + t_{Sleep}$

■ Zeitdauer der Sleep Phase (t_{Sleep})

■ Beispiel

■ Vorgabe: Duty Cycle = 10%, $t_{Listen} = 130$ ms

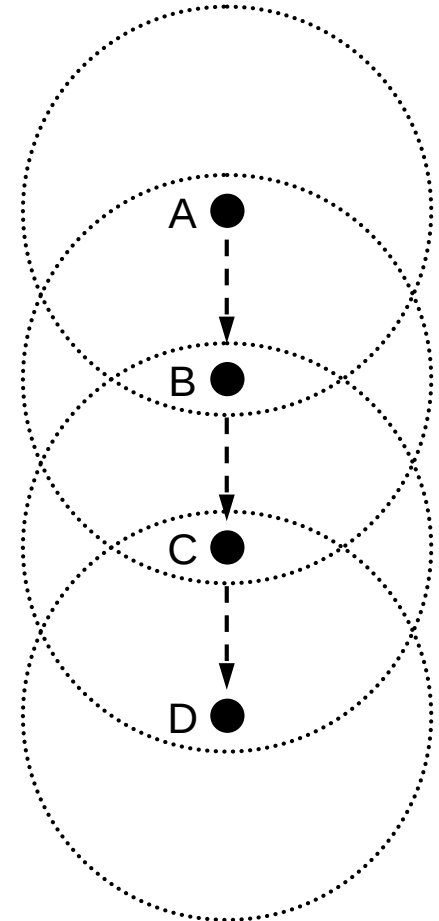
■ Berechnet: $t_{Rahmen} = 1300$ ms, $t_{Sleep} = 1170$ ms



Medienzugriff: Beispiel

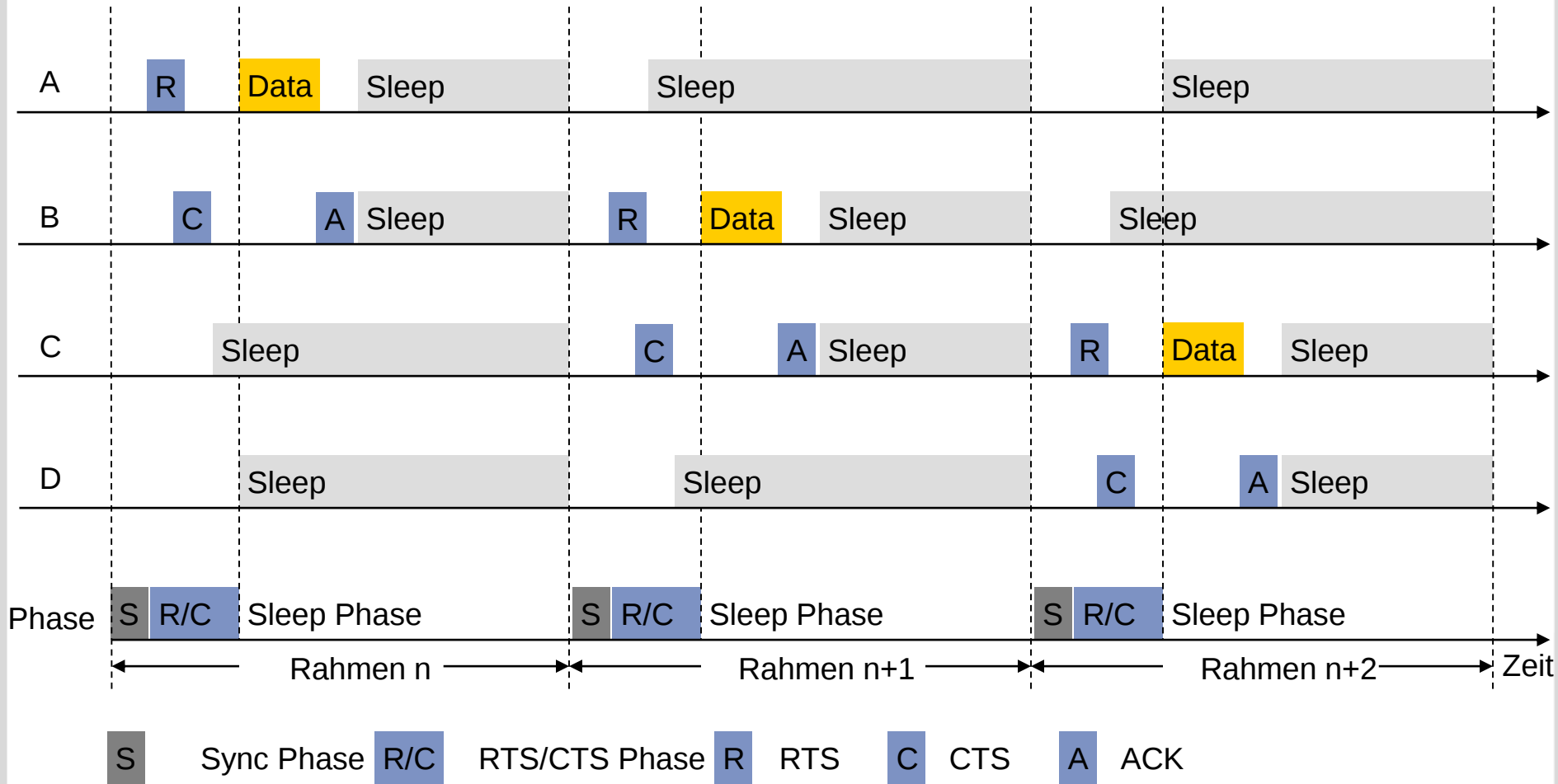
■ Multihop-Szenario

- 4 Systeme in einer Reihe
- Nur benachbarte Systeme sind in Kommunikationsreichweite
- System A hat Dateneinheit für D
 - $A \rightarrow B \rightarrow C \rightarrow D$
- Ende-zu-Ende Übertragung
 - Maximal eine Dateneinheit pro S-MAC Rahmen
 - Entfernung: 3 Hops
 - Dauer: 3 Rahmen (im Idealfall)



Beispiel

■ Multihop: $A \rightarrow B \rightarrow C \rightarrow D$



Erläuterungen zum Beispiel

- Annahme
 - Systeme sind bereits synchronisiert

- Rahmen n, RTC/CTS Phase
 - A möchte Dateneinheit an B übertragen
 - A wartet Beginn der nächsten RTS/CTS Phase ab und wählt zufälligen Zeitschlitz für RTS
 - Medium ist bis zum gewählten Zeitschlitz frei → A sendet RTS
 - B antwortet mit CTS und bleibt empfangsbereit
 - C empfängt CTS und geht schlafen (overhearing avoidance)

- Rahmen n, Sleep Phase
 - D geht schlafen (da bisher unbeteiligt)
 - A überträgt Dateneinheit
 - B antwortet mit ACK und geht schlafen
 - A empfängt ACK und geht schlafen

- Folgende Rahmen analog

Synchronisation

■ Aufgabe

- System muss Anfang der nächsten Sleep Phase kennen lernen

■ Vorgehensweise

- SYNC-Dateneinheit enthält verbleibende Zeitspanne bis zum Beginn der nächsten Sleep Phase
- System lernt Zeitplan von Nachbarn durch regelmäßigen Austausch von SYNC Dateneinheiten
- Falls noch kein Nachbar bekannt ist
 - System wählt eigenen Zeitplan
 - Sonst: folge gelerntem Zeitplan
 - → es entstehen zeitlich synchronisierte „Inseln“ mit jeweils eigenem Zeitplan
- Systeme an „Insel“-Grenzen lernen und befolgen zusätzlich Zeitpläne benachbarter Inseln
 - Grenze der Insel wird so überbrückt
 - System muss mehr Energie aufwenden

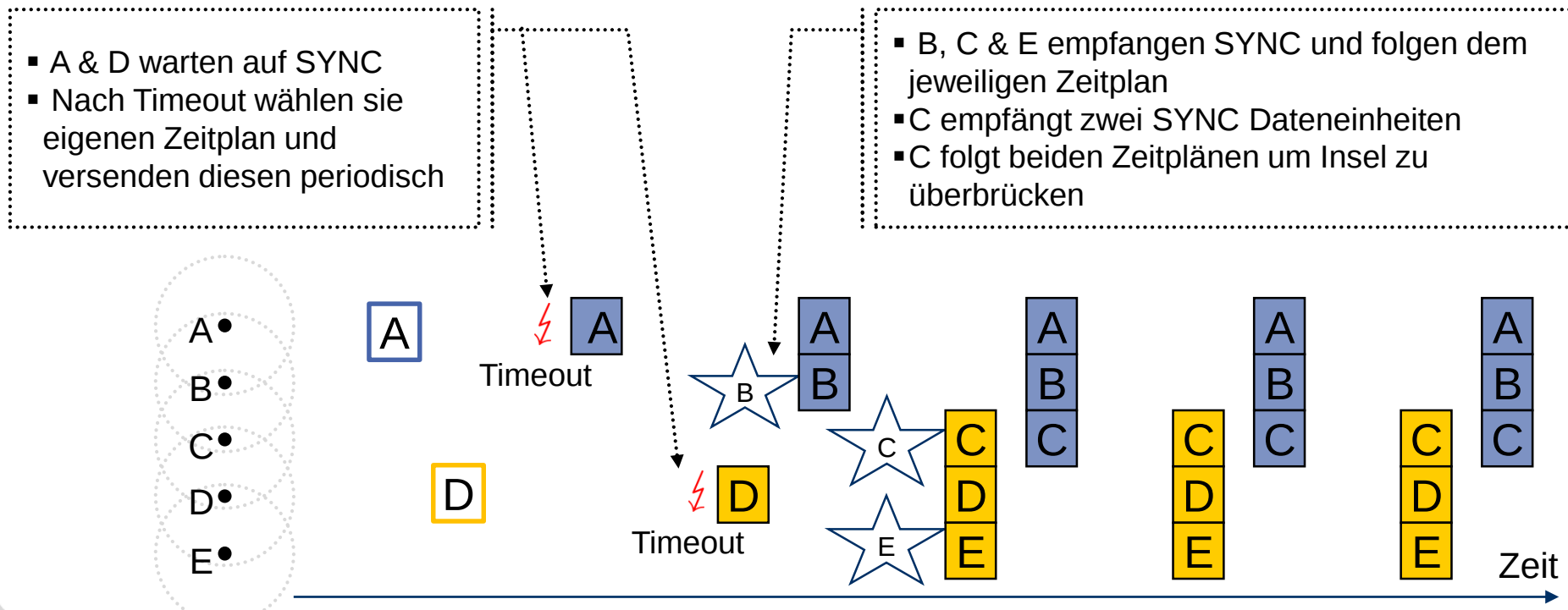
Synchronisation: Beispiel

■ Beispielszenario

- Nur direkt benachbarte Systeme können sich gegenseitig hören

■ Beispiel

- Um A und D bilden sich zwei Synchronisationsinseln
- C überbrückt zwischen As und Ds Zeitplan-Insel (blau bzw. gelb)



Erläuterungen zum Beispiel

A

System A kommt zum Netz hinzu

A

System A sendet SYNC Dateneinheit

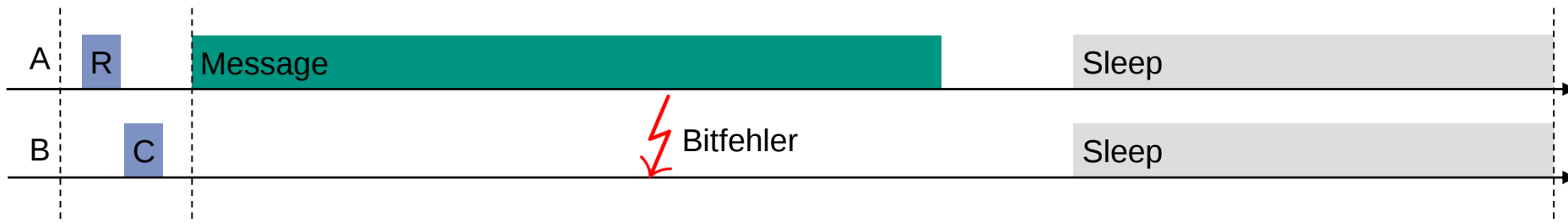
- System A kommt zum Netz hinzu
 - Nach Ablauf eines Timers ist noch kein Nachbar bekannt
 - A wählt eigenen Zeitplan (blau) und broadcastet diesen periodisch
 - D & A können sich nicht hören, D verhält sich analog
- System B kommt zum Netz hinzu
 - Lernt vor Ablauf des Timers den ‚blauen‘ Zeitplan und übernimmt diesen
→ B broadcastet periodisch den ‚blauen‘ Zeitplan
 - E verhält sich analog zu B, lernt und broadcastet den ‚gelben‘ Zeitplan
- System C kommt zum Netz hinzu
 - C lernt zunächst den ‚gelben‘ Zeitplan
 - C lernt zusätzlich den ‚blauen‘ Zeitplan
 - C befolgt und broadcastet beide Zeitpläne

Message Passing

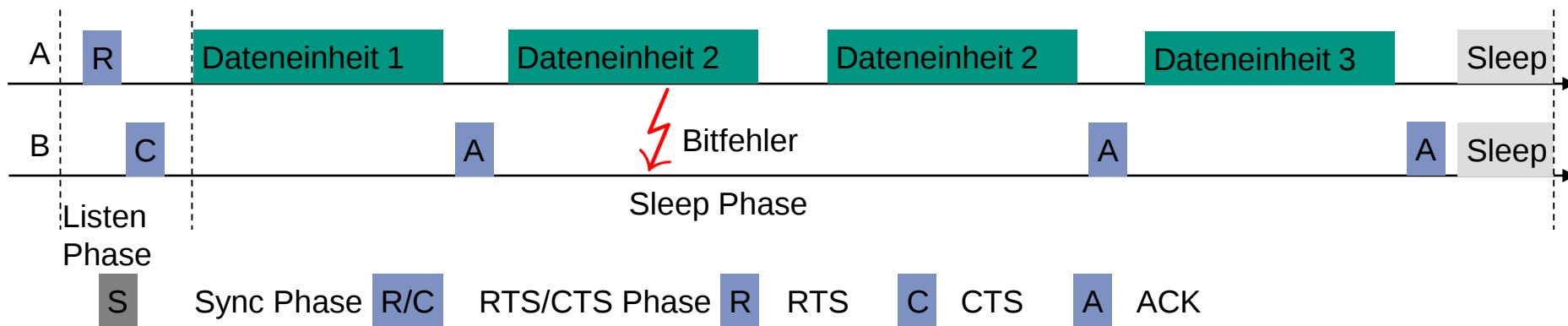
- Ziel
 - Übertragung großer Einheiten von Nutzdaten
- Problem
 - Bitfehlerwahrscheinlichkeit steigt mit Länge der Dateneinheit
- Message:= Menge von zusammengehörigen Nutzdaten
 - Message kann in einer großen Dateneinheit übertragen werden
 - Hohe Wahrscheinlichkeit, dass die gesamte Dateneinheit neu übertragen werden muss
 - Message kann in mehrere kleine Dateneinheiten geteilt werden
 - Kleinere Wahrscheinlichkeit für Neuübertragung der ganzen Message
 - Durch jede Dateneinheit entsteht Overhead durch RTS/CTS
 - Es kann max. eine Dateneinheit pro Rahmen übertragen werden
- Message Passing
 - Fragmentiere Message in mehrere Dateneinheiten
 - Übertrage alle Dateneinheiten als Burst nach einem einzigen RTS/CTS Handshake
 - Jede Dateneinheit wird einzeln bestätigt
 - Bei einem Bitfehler muss nur die fehlerhafte Dateneinheit neu übertragen werden

Beispiel: Message Passing

- Übertragung einer Message, 1 Bit fehlerhaft
- Ohne Message Passing
 - Neuübertragung im nächsten Rahmen notwendig



- Mit Message Passing



Erweiterung: Adaptive Listening

■ Beobachtung

- Periodische Listen/Sleep Phasen sparen Energie aber vergrößern die Verzögerung in Multihop-Szenarien
- In jedem Rahmen kann maximal eine Dateneinheit weitergereicht werden (pro SYNC-Insel)

■ Idee

- Führe nach Übertragung einer Dateneinheit eine zusätzliche Phase ein um einen neuen Datenaustausch anzustoßen
- → Adaptive Listening Phase (ALP)

■ Problem

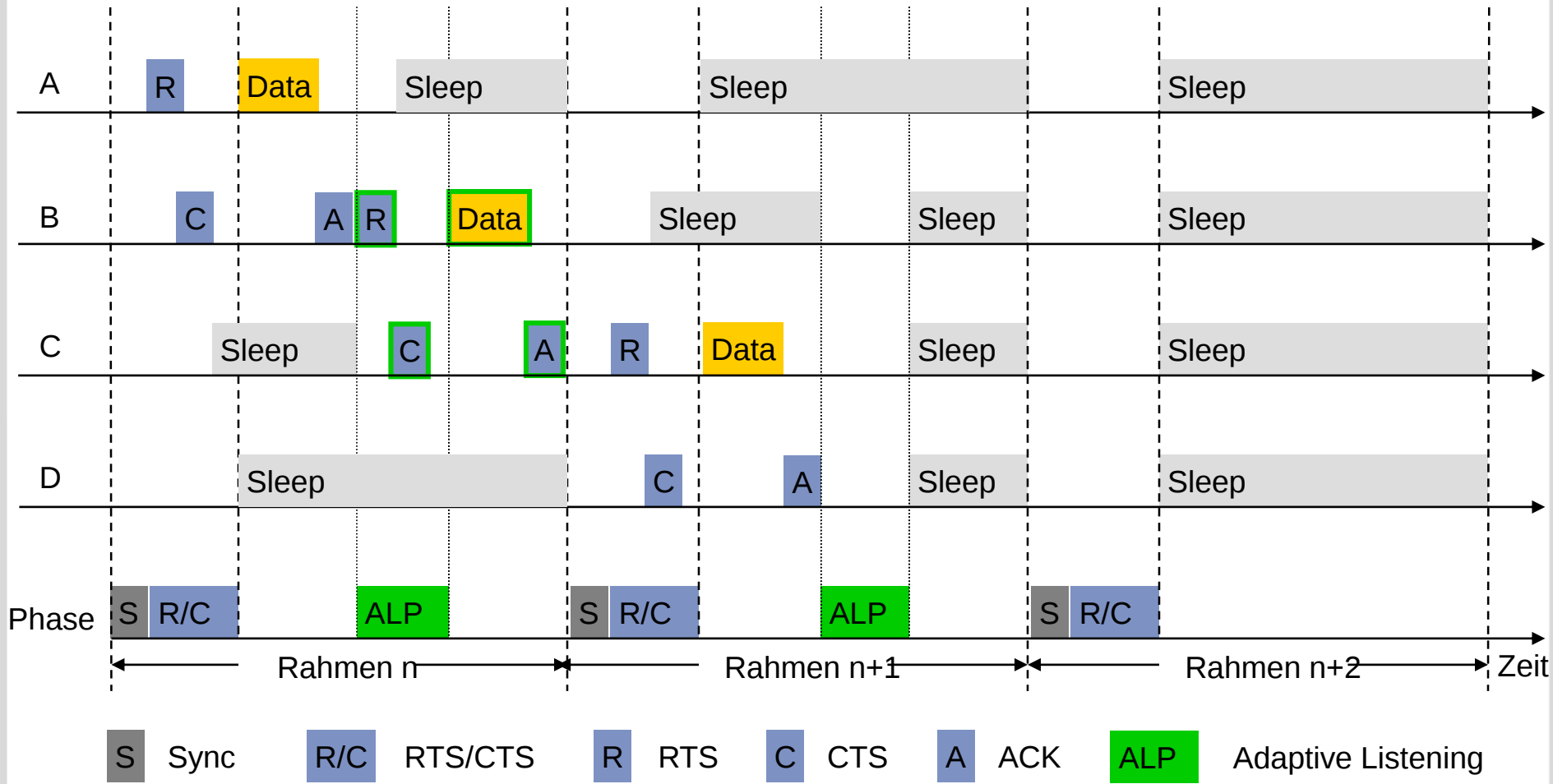
- Woher weiß ein schlafendes System, dass die Datenübertragung abgeschlossen ist?



[YeHE04]

Beispiel: Adaptive Listening

■ Multihop: $A \rightarrow B \rightarrow C \rightarrow D$ (wie vorne)



Erläuterungen zum Beispiel

- Annahme
 - Systeme sind bereits synchronisiert
- Rahmen n, RTS/CTS Phase
 - A sendet RTS, B sendet CTS
 - RTS/CTS enthalten die Zeitdauer der Datenübertragung inkl. ACK
 - C empfängt CTS und geht für im CTS stehende Zeitdauer schlafen (overhearing avoidance)
- Rahmen n, Sleep Phase
 - D geht schlafen (ganze Sleep Phase, D hat CTS nicht empfangen)
 - A überträgt Dateneinheit
 - B antwortet mit ACK, A empfängt ACK
- Nach Ablauf der in RTS/CTS angekündigten Zeitspanne
 - Zusätzliche RTS/CTS Phase wird eingefügt (sog. Adaptive Listening Phase)
 - B kann noch während des selben Rahmens die Dateneinheit an C weiterleiten
- Folgende Rahmen analog

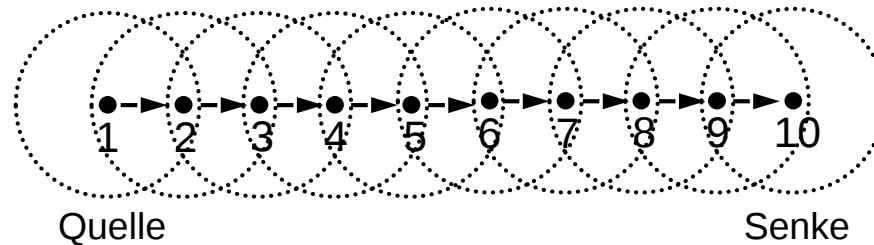
Leistungsbewertung

■ Ziel

- Bewertung und Vergleich der Leistungsfähigkeit unterschiedlicher Protokollvarianten anhand objektiver Kenngrößen

■ Versuchsaufbau

- 10 Systeme in einer Reihe, multihop



- Simulator: ns-2
- Verkehrsgenerator: Continuous Bitrate (CBR)
- Dateneinheiten fester Länge zu je 100 Byte
- Eingangspuffer: 50 Dateneinheiten je System

■ Untersuchte Protokollvarianten

- 1. Ohne periodisches Schlafen
- 2. S-MAC ohne Adaptive Listening, 10% Duty Cycle
- 3. S-MAC mit Adaptive Listening, 10% Duty Cycle

Kenngrößen zur Leistungsbewertung

■ Durchschnittlicher Energiebedarf pro Byte

- Gesamtenergiebedarf aller Systeme geteilt durch die Anzahl von der Senke empfangener Bytes

$$\bar{E} = \frac{\sum_i \text{Energieverbrauch}(\text{System}_i)}{\sum_{d \in S} \text{LängeInBytes}(d)}$$

■ Durchschnittliche Ende-zu-Ende Verzögerung

- Summe aller Ende-zu-Ende Verzögerungen geteilt durch die Anzahl der Dateneinheiten

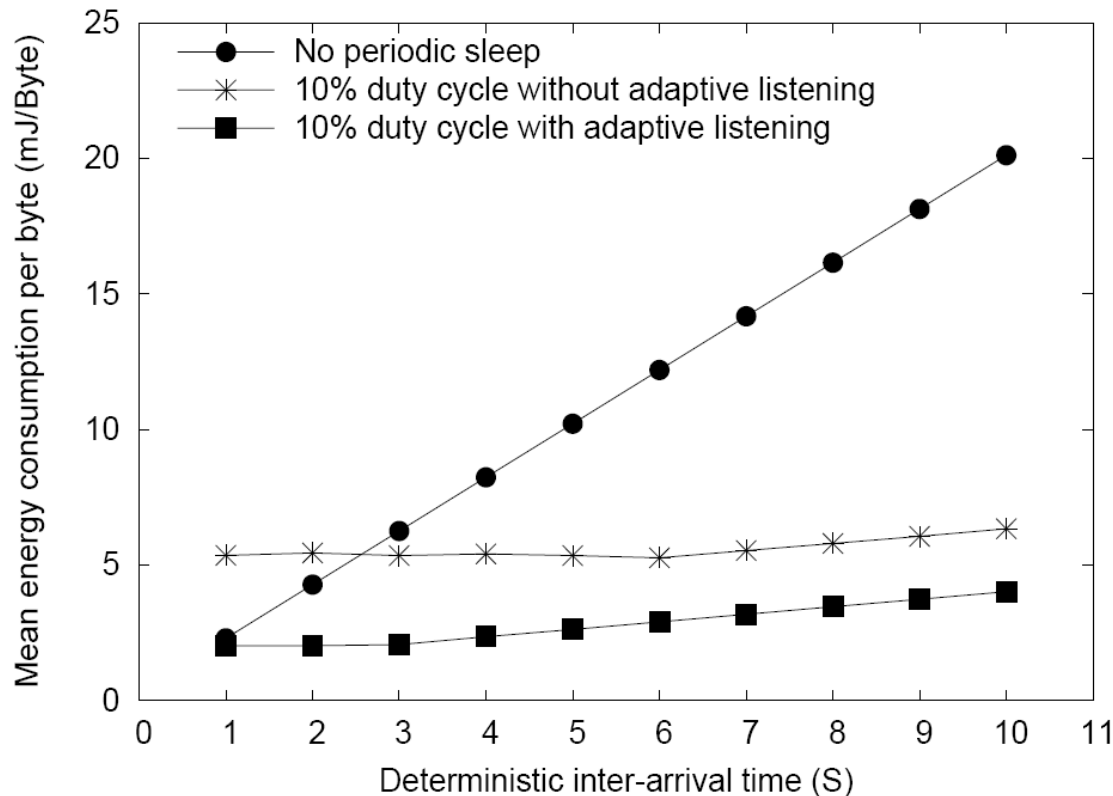
$$\bar{V} = \frac{\sum_{d \in A} \text{EmpfangszeitPkt}(d) - \text{SendezeitPkt}(d)}{k}$$

■ Ende-zu-Ende Goodput

- Gesamtzahl von der Senke empfangener Bytes geteilt durch die Zeitspanne zwischen Versenden der ersten Dateneinheit bis zum Empfang der letzten Dateneinheit an der Senke

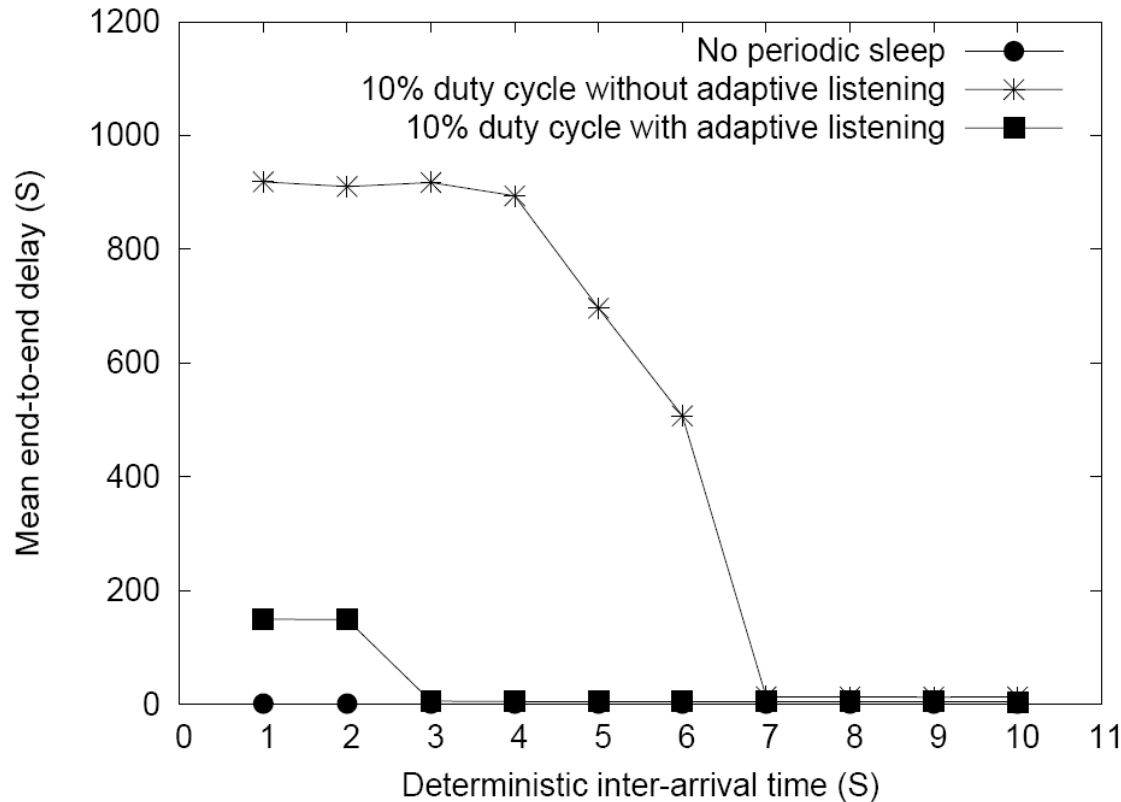
$$\bar{G} = \frac{\sum_{d \in A} \text{LängeInBytes}(d)}{\text{EmpfangszeitPkt}(d_k) - \text{SendezeitPkt}(d_0)}$$

A := Menge aller von der Senke empfangenen Dateneinheiten, $k := |A|$, $d \in A$



- Beide S-MAC Varianten reduzieren den Energiebedarf (durch Idle Listening), insbesondere bei geringer Verkehrslast (=lange Zwischenankunftszeiten)
- Bei hoher Last kann Adaptive Listening den Energiebedarf nochmals halbieren

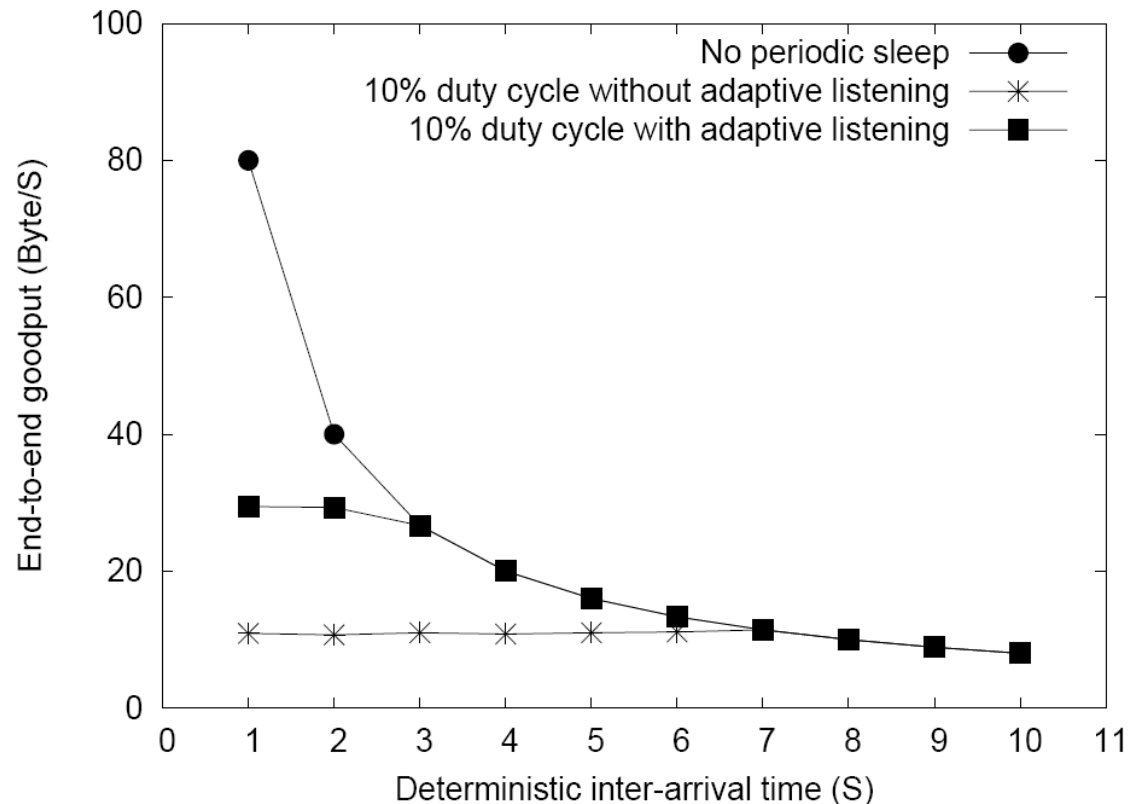
Durchschnittliche Ende-zu-Ende Verzögerung



- 10 Hop Netz und große Warteschlangen sorgen bei S-MAC für eine sehr große Ende-zu-Ende Verzögerung bei hoher Verkehrslast (Sättigungskurve)
- S-MAC ohne ALP verlässt die Sättigung bei 4s Zwischenankunftszeit
- S-MAC mit ALP verlässt die Sättigung bei 2s Zwischenankunftszeit
- S-MAC mit ALP hat wesentlich geringere Verzögerung während Sättigung

Ende-zu-Ende Goodput

- S-MAC reduziert wesentlich den erreichbaren Goodput
- Adaptive Listening verbessert die Ausnutzung der verfügbaren Datenrate (durch verminderte der Verzögerung)



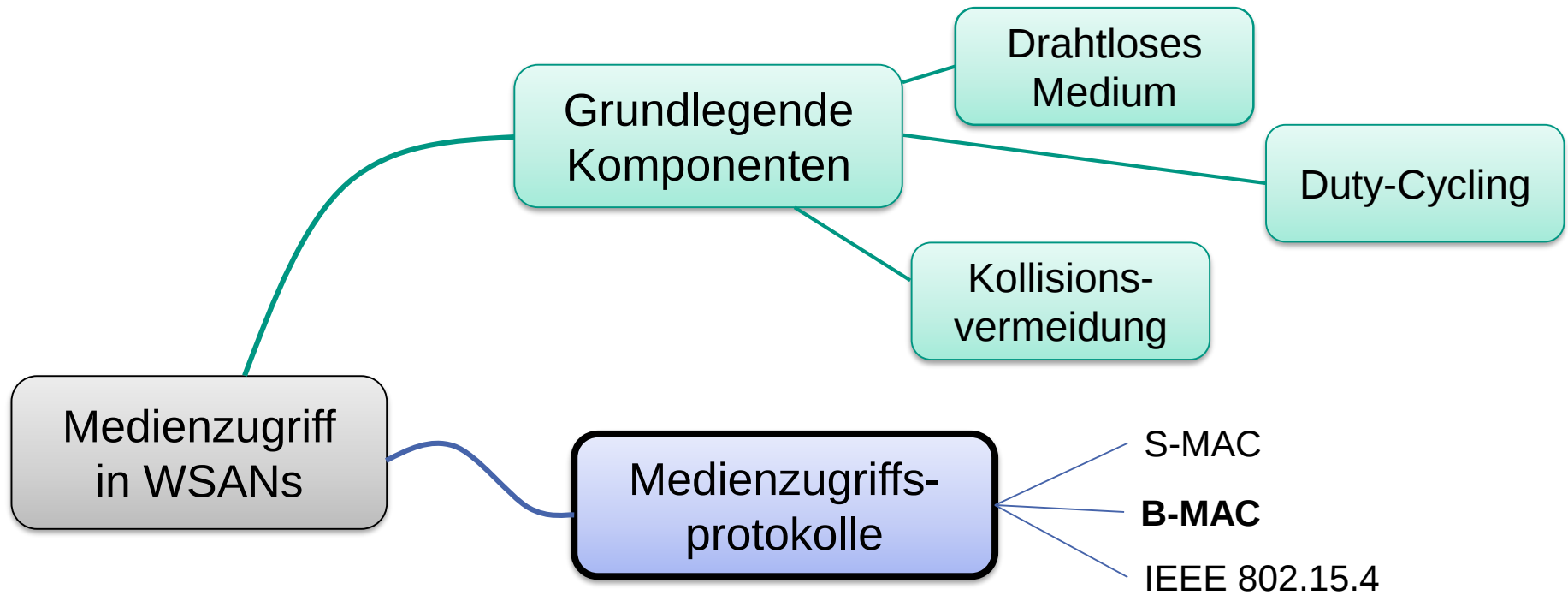
Zusammenfassung S-MAC

■ Ansatz

- Energie sparen durch periodisches Schlafen und koordiniertes Aufwachen
- RTS/CTS zur
 - Aushandlung des Zugriffsrechts auf den Kanal (Arbitrierung)
 - Vermeidung des „Versteckte Endgeräte“ Problems
- Fragmentierung großer Nachrichten in kleine Dateneinheiten zur effizienten Übertragung

■ Vorteile / Nachteile

- Geringer Energiebedarf
 - Reduzierung von Kollisionen durch RTS/CTS
 - Vermeidung von Idle Listening durch periodisches Schlafen
 - Vermeidung von Overhearing durch RTS/CTS + Schlafen
- Geringer Durchsatz, hohe Latenzen durch lange Wartezeit auf nächsten freien Rahmen



B-MAC

■ Zielsetzung

- Energieeffizienter Betrieb durch
 - Kollisionsvermeidung
 - Effiziente Kanalnutzung bei hohen und niedrigen Datenraten
- Skalierbarkeit bezüglich der Anzahl der Systeme
- Toleranz gegenüber sich ändernden Funkbedingungen
- Rekonfigurierbarkeit durch das Protokoll der Vermittlungsschicht
- Einfache Implementierbarkeit, geringe Codegröße, geringer Speicherverbrauch

■ Besonderheiten

- Periodisches Prüfen des Kanals, statt zeitlicher Synchronisation
- Problem der Versteckten Endgeräte nicht behandelt
- Keine Fragmentierung großer Nachrichten
- Wird zuweilen als Standard MAC für drahtlose Sensornetze angenommen, da TinyOS-Implementierung vorhanden

Grundlegendes Prinzip

■ Ruhezustand / empfangsbereit

- System schläft meist
- Erwacht gelegentlich (kurz!), um Kanal auf Daten zu prüfen
- Wach bleiben, falls Daten zum Empfang anstehen

} = *Low Power Listening (LPL)*

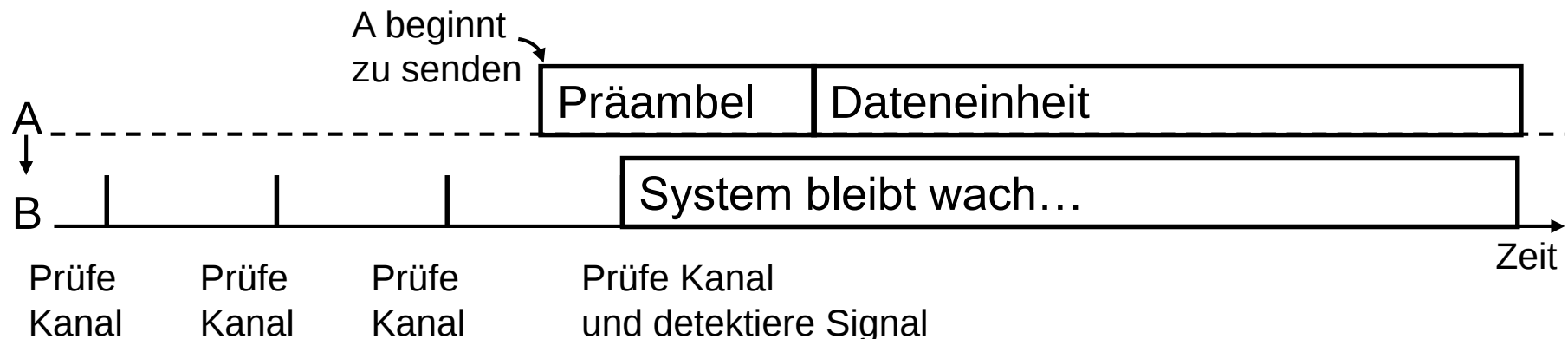
■ Senden

- Prüfe ob Kanal frei ist
- Übertrage Präambel + Daten

} = *Clear Channel Assessment (CCA)*

Low Power Listening (LPL)

- Potentieller Empfänger schläft und prüft nur periodisch den Kanal
- Präambel pro Dateneinheit stellt sicher, dass der Empfänger die Übertragung nicht verpasst
 - Intervall zwischen aufeinanderfolgenden Kanalüberprüfungen legt Präambel-Mindestdauer fest
- Beispiel
 - B prüft periodisch den Kanal
 - A sendet Präambel + Dateneinheit an B
 - Sobald B ein Signal detektiert, bleibt B wach



B-MAC: Leistungsbewertung

- Vergleich B-MAC vs. S-MAC
 - B-MAC „Erweiterungen“
 - RTS/CTS
 - ACKs
- Plattform
 - TinyOS / Mica2
- Platzierung der Systeme
 - 15 cm über dem Boden
 - Sichtverbindung
 - Experiment in störungsfreier Umgebung
- Multihop Experimente
 - Sendereichweite reduziert auf Minimum
 - 1 m Abstand zwischen den Systemen



[PoHC04]

Messmethodik & Parametrisierung

■ Durchsatzmessung

- Nur Nutzdaten berücksichtigen
- Protokoll-Overhead ausgeklammert
- $n = 1..20$ Systeme im selben Abstand zum Empfänger
- Jedes System sendet so schnell wie möglich

■ Verzögerungsmessung

- System an Oszilloskop angeschlossen
- Signal an Pin bei Senden/Empfang einer Dateneinheit

■ Messung der Leistungsaufnahme

- Indirekt: Anzahl der Aufrufe pro Operation & Messung der Leistungsaufnahme pro Operation (siehe Tabelle)

Length (bytes)	B-MAC	S-MAC
Preamble	8	18
Synchronization	2	2
Header	5	9
Footer (CRC)	2	2
Data Length	29	29
Total	46	60

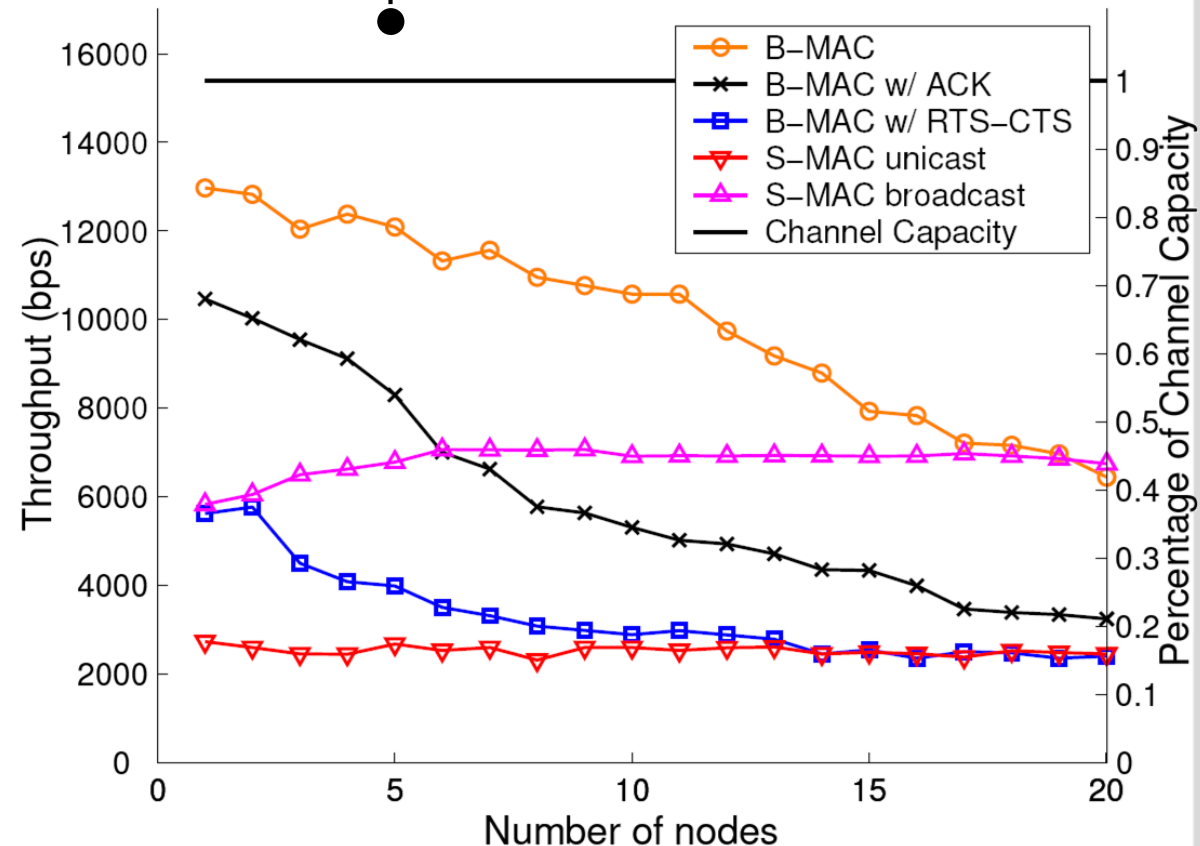
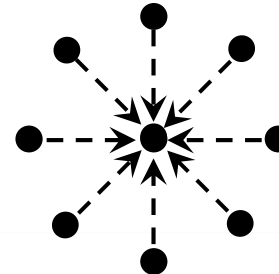
Operation	I (mA)
Initialize radio (b)	6
Turn on radio (c)	1
Switch to RX/TX (d)	15
Time to sample radio (e)	15
Evaluate radio sample (f)	6
Receive 1 byte	15
Transmit 1 byte	20
Sample sensors	20



[PoHC04]

Durchsatz

- Je weniger Systeme um den Kanal konkurrieren, desto höher ist der Durchsatz
- Bei nur einem System hat B-MAC höheren Durchsatz als S-MAC Broadcast
- B-MAC erreicht fast das 4,5 fache des Durchsatzes von S-MAC Unicast durch
 - Geringeren Rechenaufwand pro Dateneinheit
 - Effektivität des Clear Channel Assessments



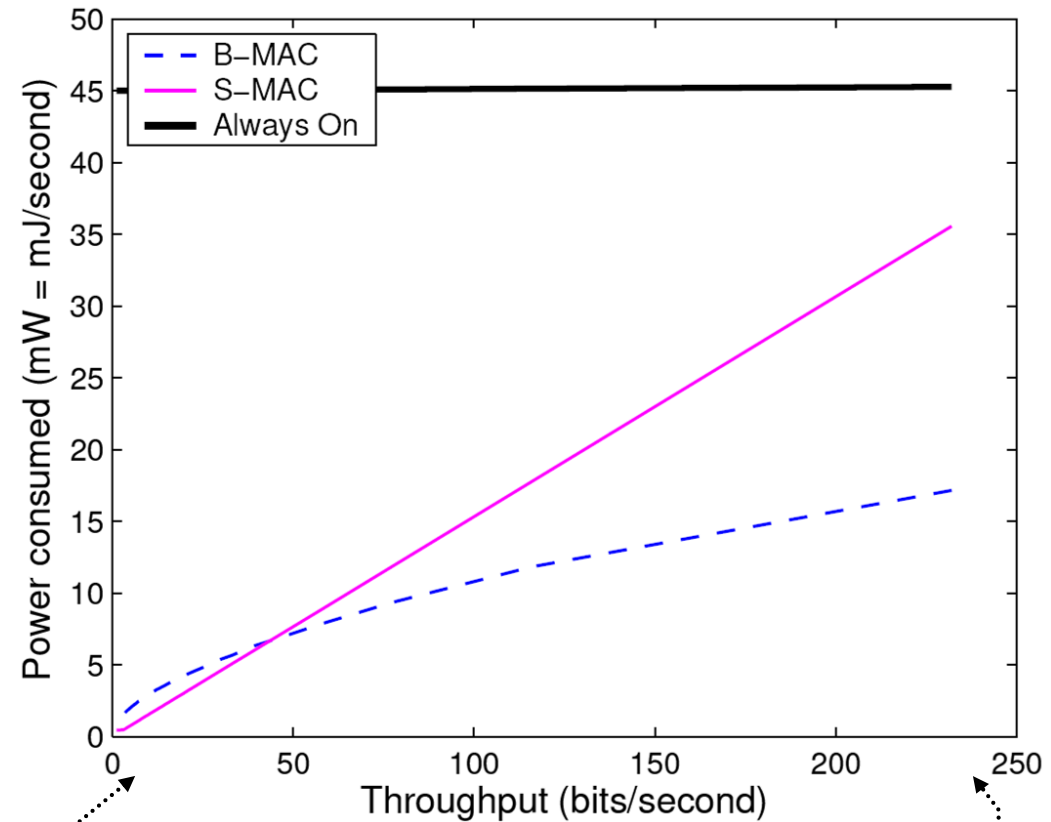
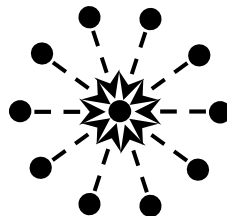
Zur Durchsatzmessung untersuchte Protokollvarianten

- B-MAC
 - B-MAC wie beschrieben
- B-MAC w/ ACK
 - Übertragung einer Empfangsbestätigung nach erfolgreichem Empfang einer Dateneinheit
- B-MAC w/ RTS-CTS
 - B-MAC mit RTS-CTS Handshake vor eigentlicher Datenübertragung
 - Bei zur Übertragung anstehender Dateneinheit versende RTS
 - Adressat antwortet mit CTS, falls dieser empfangsbereit ist
 - Übertrage RTS und CTS mittels LPL
 - Zur Übertragung der Dateneinheit werden LPL und CCA nicht verwendet
 - Übertragung einer Empfangsbestätigung (ACK) nach erfolgreichem Empfang der Dateneinheit
- S-MAC unicast
 - S-MAC bei Unicast Adressierung (insb: RTS-CTS verwendet)
- S-MAC broadcast
 - S-MAC bei Übertragung an Broadcast Adresse (hier kein RTS-CTS, Dateneinheit wird stattdessen direkt übertragen)

Energiebedarf und Durchsatz

- MAC Stellschrauben zur Anpassung des gewünschten Durchsatzes
 - Durchsatz indirekt über Verzögerung zwischen zwei übertragenen Paketen gesteuert
 - B-MAC: CCA Sample Intervall
 - S-MAC: Duty Cycle

- Höherer Durchsatz bedeutet
 - S-MAC: Energiebedarf wächst linear
 - Overhead der SYNC-Periode
 - B-MAC: Energiebedarf wächst sublinear



B-MAC: kurzes Sample Intervall
S-MAC: hoher Duty Cycle

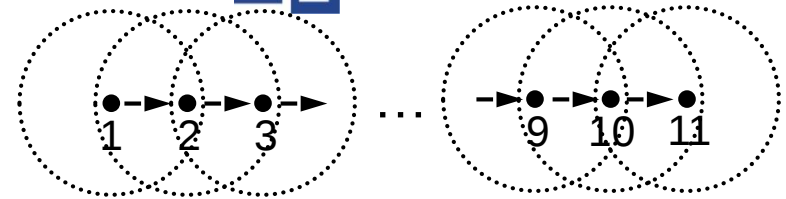
B-MAC: langes Sample Intervall
S-MAC: geringer Duty Cycle
(Listen Phase fix => lange Rahmen)



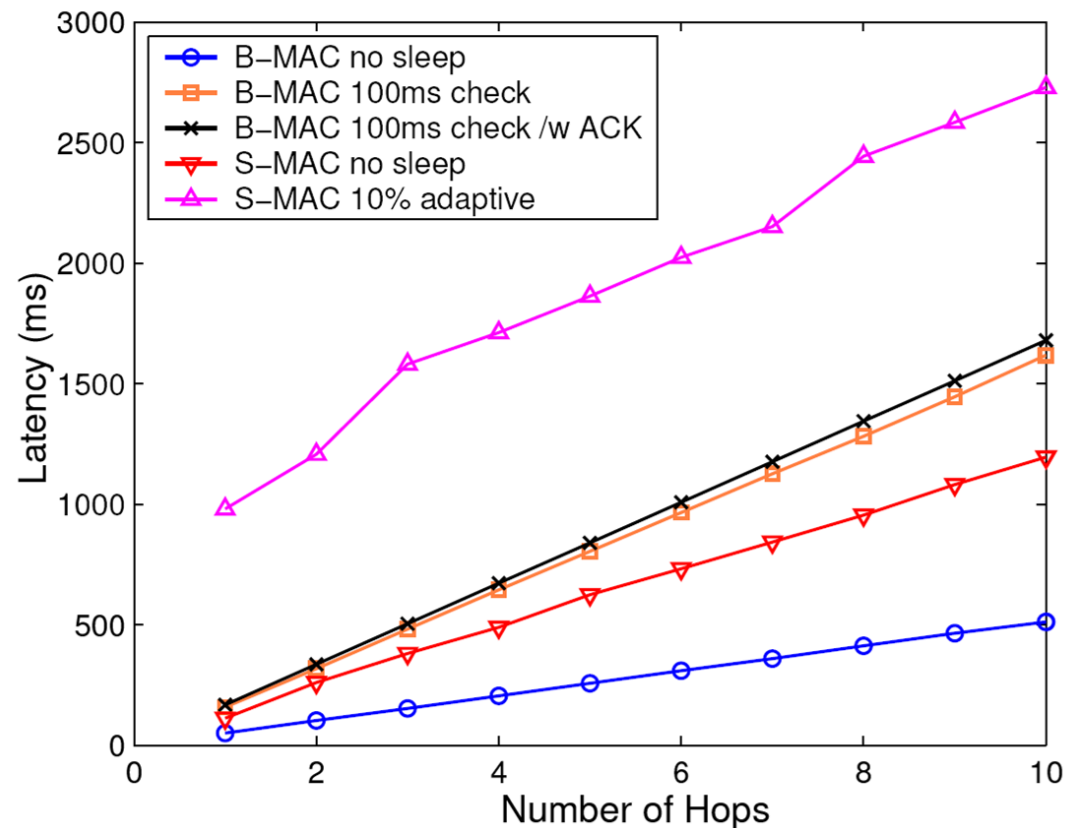
[PoHC04]

Ende-zu-Ende Verzögerung

[PoHC04]



- Ende-zu-Ende Verzögerung steigt linear mit der Anzahl Hops
- Die Steigung wächst mit zunehmendem Protokolloverhead
- B-MAC pro Hop Verzögerung bleibt konstant
- S-MAC mit 10% Duty Cycle und Adaptive Listening
 - 1150ms initiales Warten auf nächste RTS/CTS Phase
 - Danach geringere pro Hop Verzögerung durch Adaptive Listening



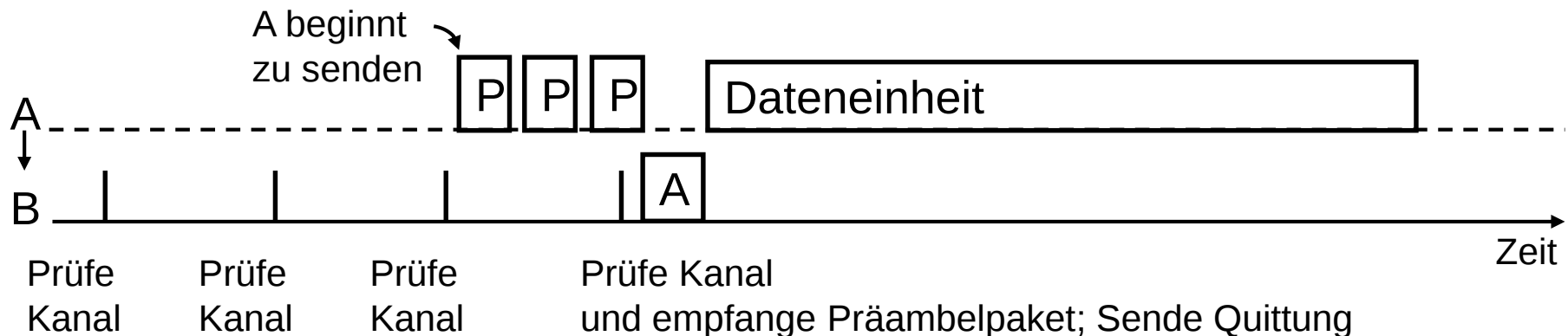
Erweiterungen: X-MAC

■ Beobachtung

- Lange Präambel in vielen Fällen ungünstig
 - Verursacht hohe Kosten beim Sender insbesondere bei langen Schlafzeiten
- Idle-Listening Problem für nicht beteiligte Knoten nicht gelöst
 - Auch eigentlich nicht beteiligte Empfänger wachen auf, wenn Präambel empfangen wird

■ Idee

- Statt einer großen Präambel, sende viele kleine Präambelpakete  [Buettner06]
- Für Unicast: Integriere Zieladresse in kurze Präambelpakete
 - Empfangender Knoten kann Empfang der Präambelpakete quittieren
 - Übertragung kann daraufhin starten
 - Nichtbeteiligte Knoten können weiter schlafen



B-MAC: Zusammenfassung

■ Ansatz

- Clear Channel Assessment (CCA) zur Kanalarbitrierung
- Low Power Listening (LPL) zum Energie Sparen

■ Vorteile / Nachteile

- Keine Synchronisation der Systeme notwendig
- Einfaches Protokoll, kompakte Implementierung
- Geringe Ende-zu-Ende Latenz
- LPL erfordert Präambel
- Keine Behandlung von versteckten Endgeräten

Implementierungsdetails: 802.15.4 + B-MAC/X-MAC

- In vielen Knoten 802.15.4 auf Layer 1 und 2 vorzufinden
- Wie implementiert man B-MAC ?
 - B-MAC benötigt dauerhaft gesendete Präambel (Bitstrom)
 - CCA Check weckt Knoten auf, dann wachbleiben im Empfangsmodus
 - Senden von 100 Byte Nutzdaten belegt Kanal ca. 5ms, dann ca. 5ms Pause aufgrund Wechsel Empfangen/Senden
 - Was passiert wenn CCA genau in dieser Pause stattfindet ?
 - Wie geht man mit automatisierten Bestandteilen in 802.15.4 um ?
 - CRC Erzeugung, automatische Präambel, Framing, ...
- Wie implementiert man X-MAC ?
 - X-MAC benötigt auch Schicht 2 Informationen → Zieladresse
 - Wie kann diese in Präambel eingebettet werden ?

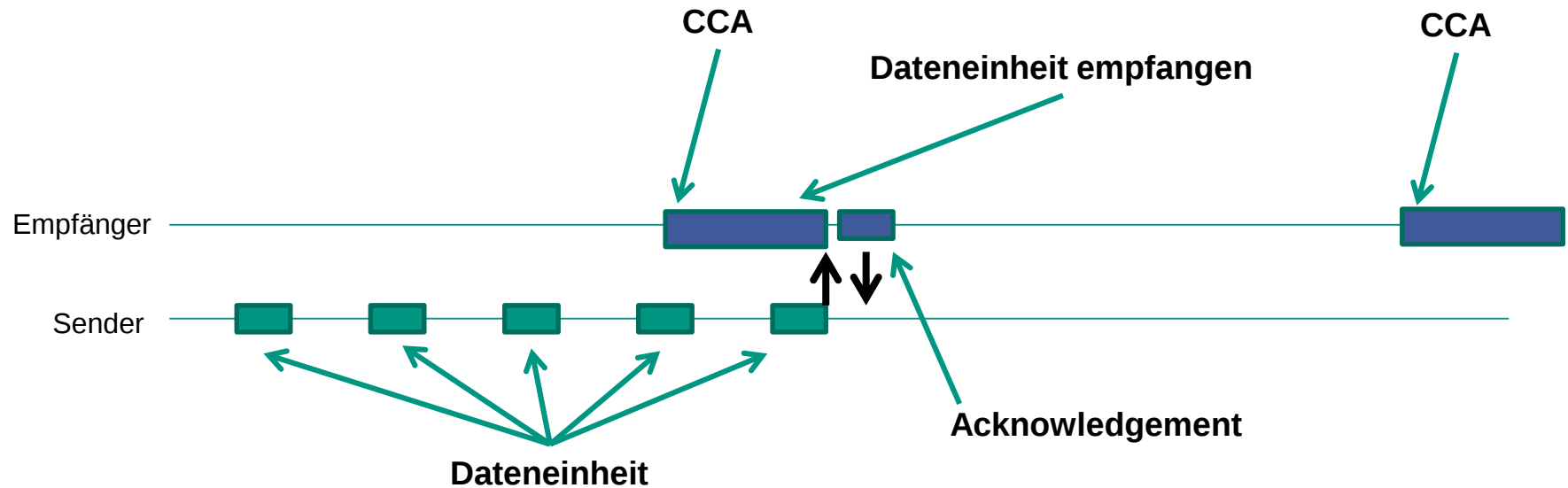
Implementierungsdetails: 802.15.4 + B-MAC/X-MAC (II)

- Erkenntnis: B-MAC und X-MAC nicht auf vorhandenen Funkchips implementierbar
 - 802.15.4 paketbasiert, erlaubt kein dauerhaftes Senden einer Präambel
 - Längste übertragbare Präambel somit 1 Paket der Länge 128 Byte

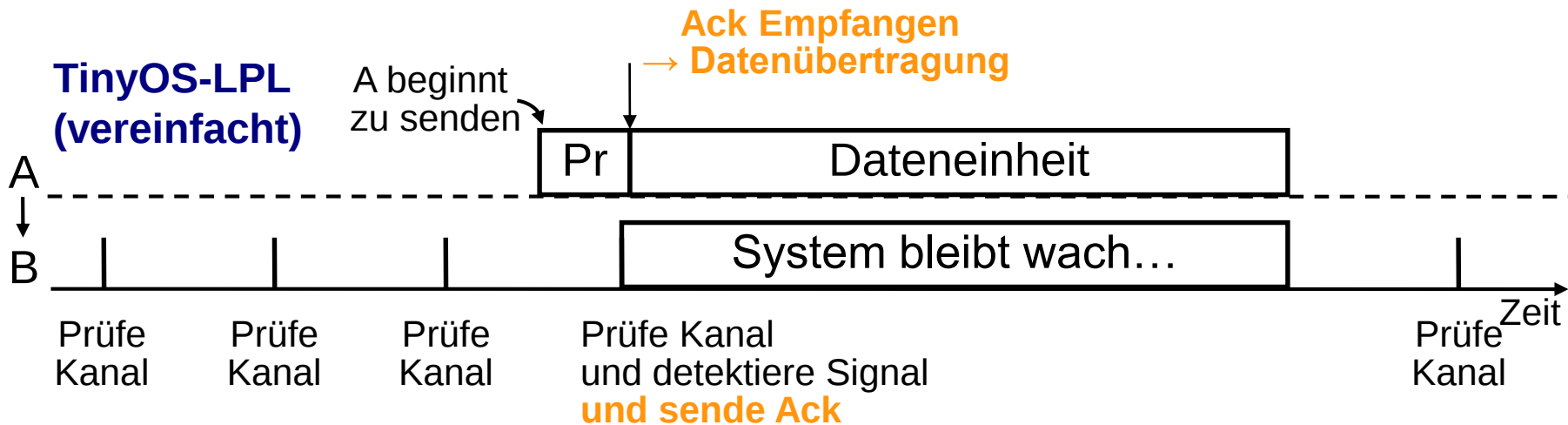
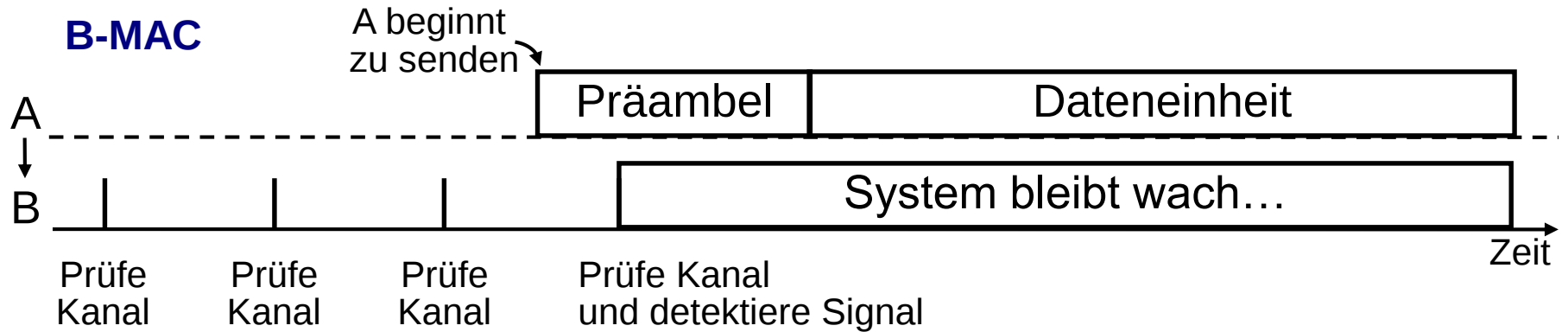
- Anpassung an paketbasierte Funkschnittstellen
 - Statt Präambel, sende Dateneinheit wiederholt
 - Zieladresse enthalten
 - Wenn Knoten aufwacht und Paket empfängt, sende ACK
 - Wachzeit der Knoten muss größer als „Lücke“ zwischen aufeinanderfolgenden Dateneinheiten sein
 - Beispiel BoX-MAC-2
 - Betriebssystemspezifische Implementierungen vorhanden
 - TinyOS-LPL und Contiki-MAC

Implementierungsdetails: 802.15.4 + B-MAC/X-MAC (III)

■ Schematischer Ablauf TinyOS-LPL



Vergleich B-MAC – TinyOS-LPL



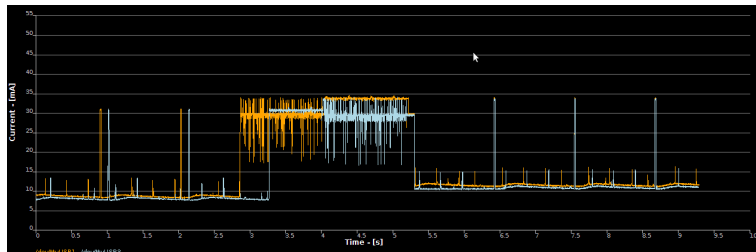
B-MAC/TinyOS-LPL Demo

■ Versuchsaufbau

- Zwei Sensorknoten „spielen Ping-Pong“
 - Tauschen wiederholt eine Dateneinheit aus
- Eingesetzte MAC-Protokolle austauschbar
 - B-MAC
 - TinyOS-LPL
- An beiden Sensorknoten wird die Leistungsaufnahme während der Ausführung gemessen
 - Funktionsweise der MAC-Protokolle anhand der Leistungsaufnahme sichtbar

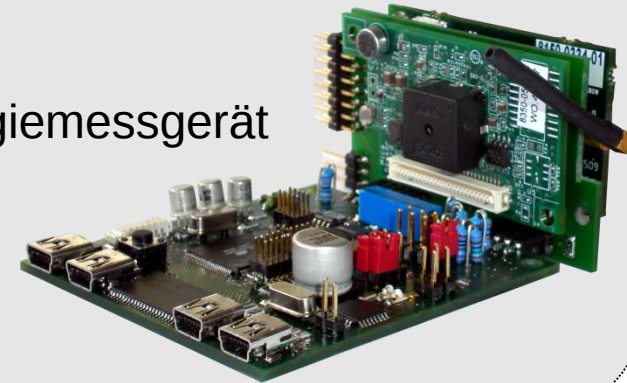
B-MAC/TinyOS-LPL Demo

Anzeige des Energiebedarfs beider
Sensorknoten



Sensorknoten

Energiemessgerät



Sensorknoten

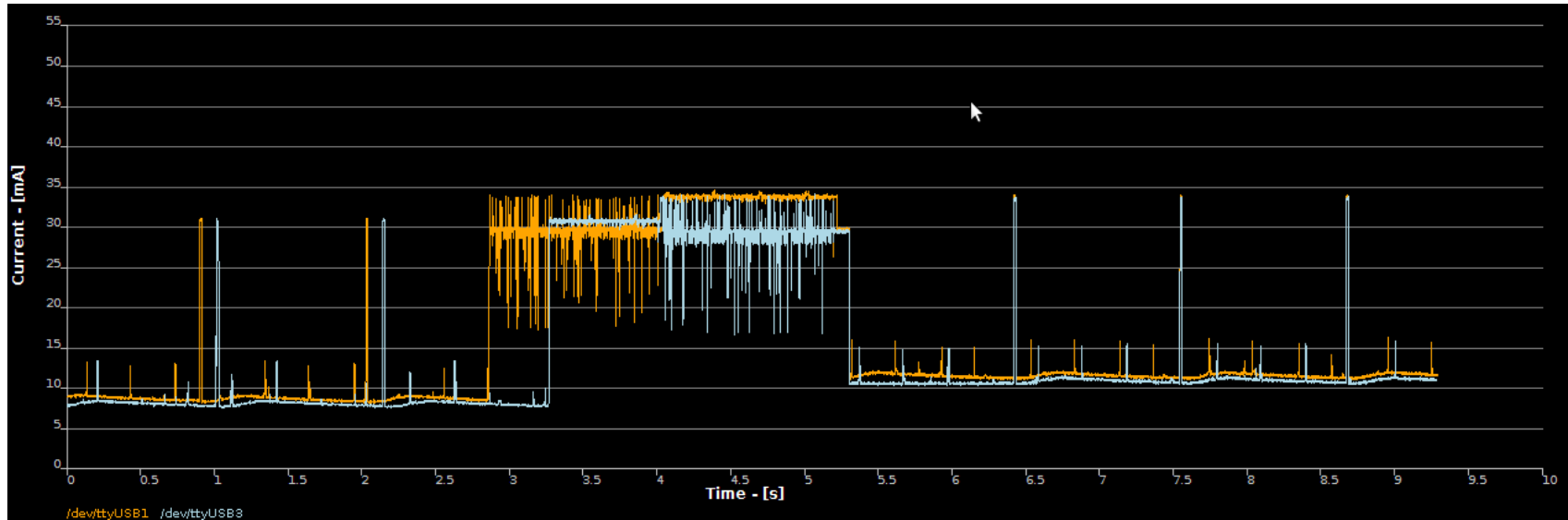


Sensorknoten

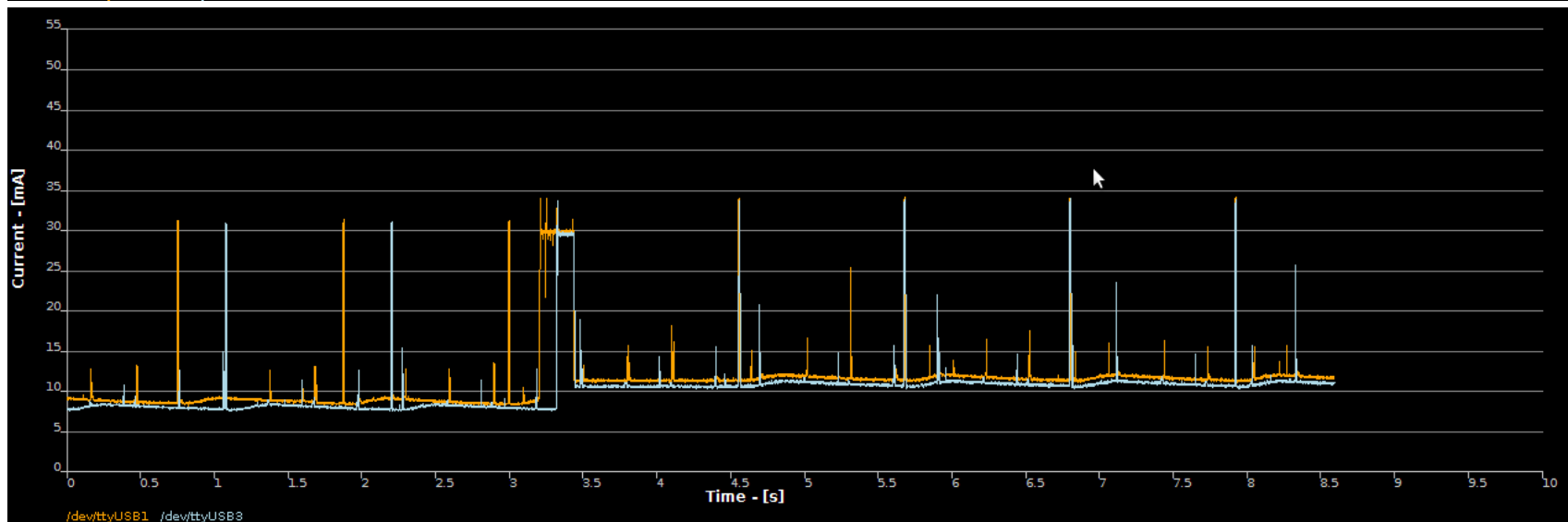
Ping-Pong

B-MAC/TinyOS-LPL: Energiebedarf

B-MAC



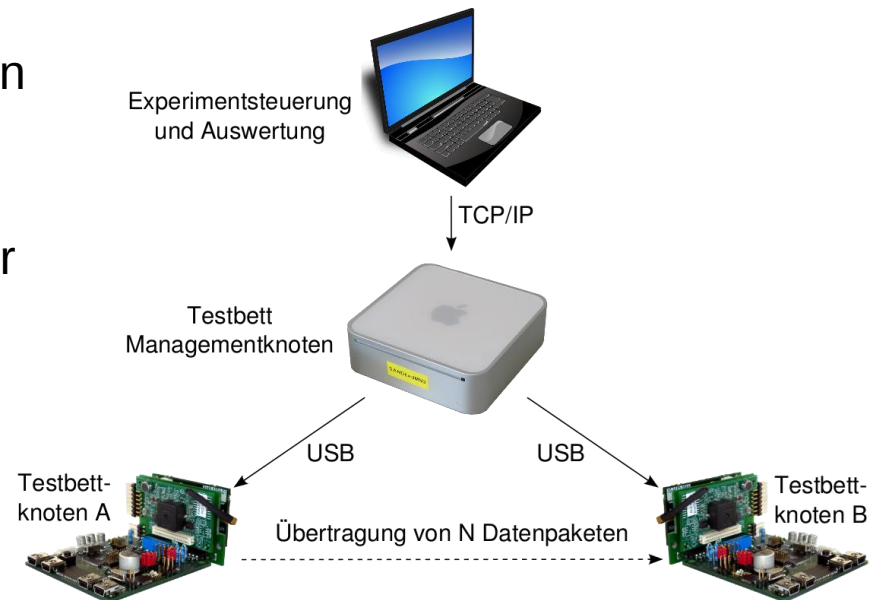
TinyOS-LPL



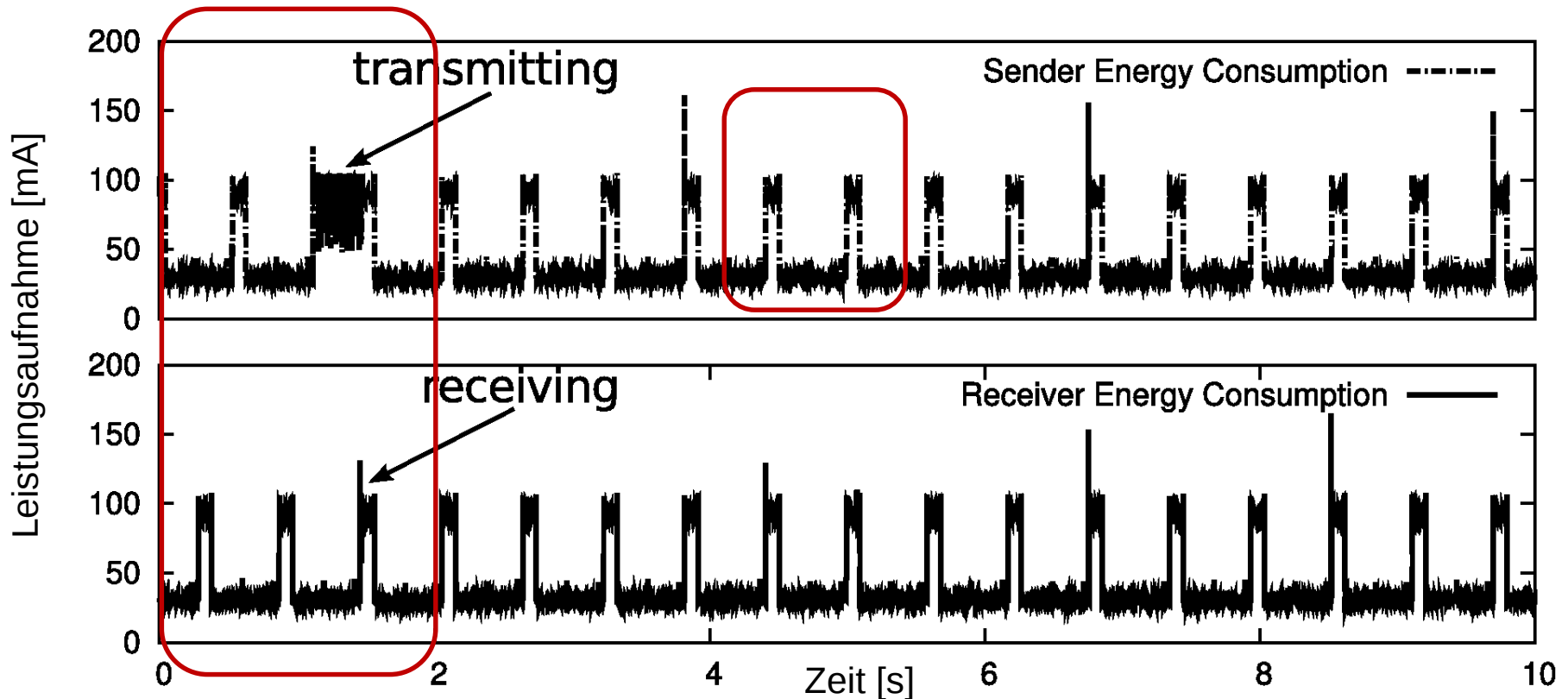
B-MAC/TinyOS-LPL: Energiebedarf

- Einfluss der CCA-Periodendauer auf Energiebedarf von Kommunikationsvorgängen?
 - Längere Schlafperiode → Geringerer Energiebedarf?

- Untersuchung eines einfachen Beispiels
 - Übertragung von N Dateneinheiten mit jeweils 1 Byte Payload
 - Messung des Energiebedarfs an den beteiligten Sensorknoten über einen Messzeitraum von 10 Sekunden

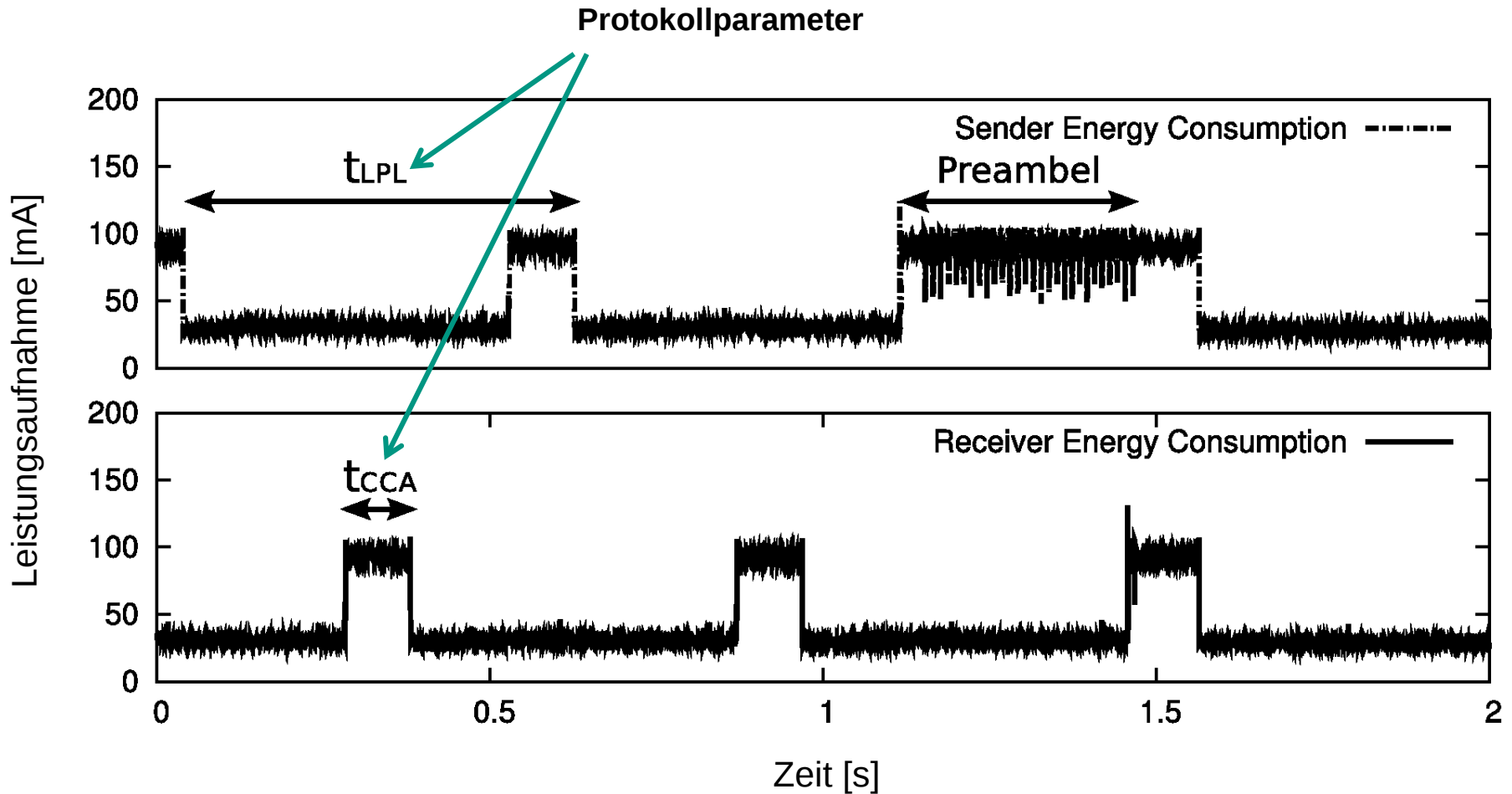


Messung (Beispiel mit TinyOS-LPL)



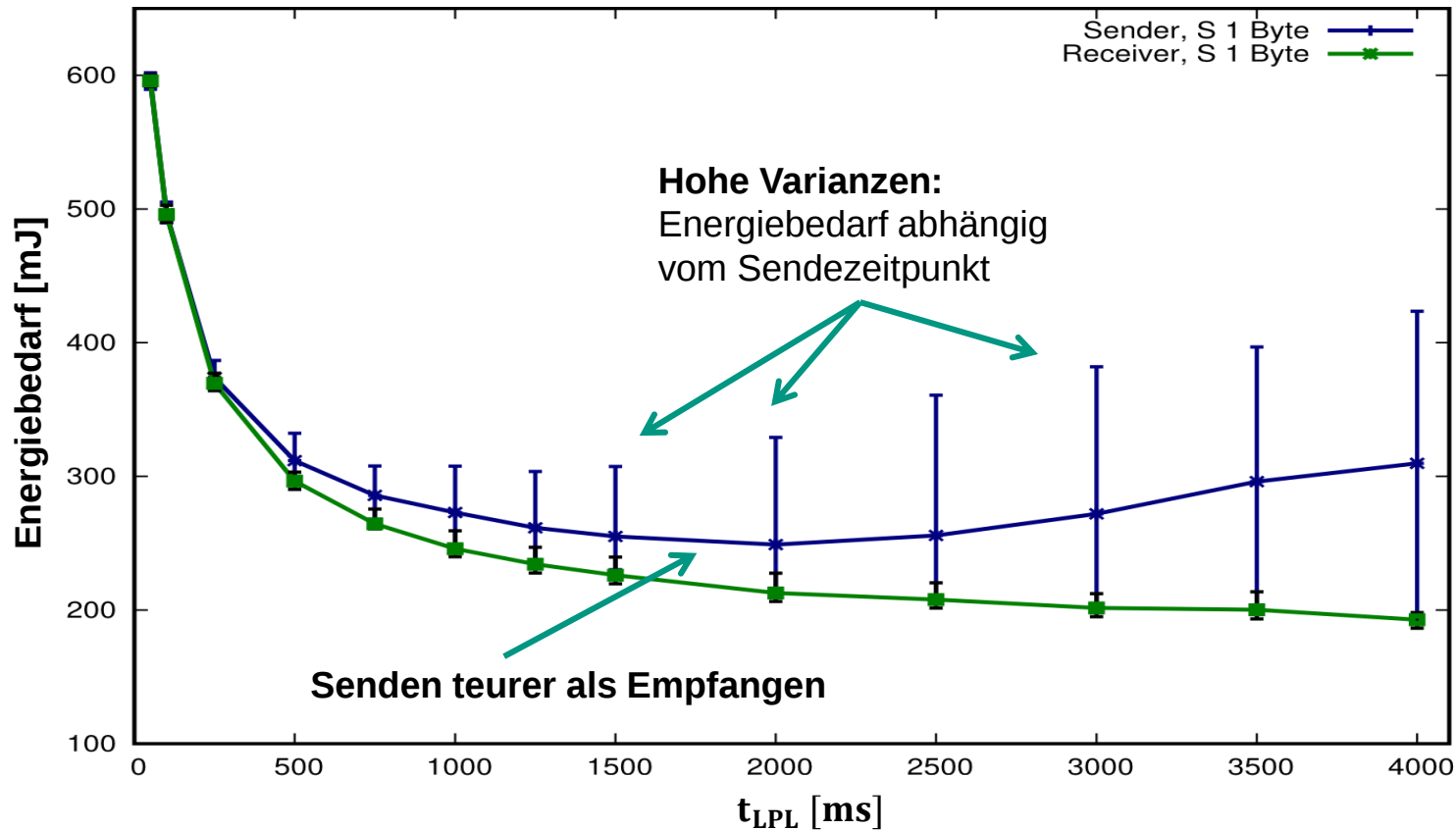
- Übertragung der Dateneinheit anhand der Leistungsaufnahme erkennbar
 - Senden einer Dateneinheit ist teuer (Präambel)
- Auch ohne Übertragung einer Dateneinheit erhöhte Leistungsaufnahme
 - Warum ?

Messung (Beispiel mit TinyOS-LPL)



- Erhöhte Leistungsaufnahme durch periodischen CCA Check
- Aufwachzeitpunkte und Aufwachdauer im MAC-Protokoll parametrisierbar

Evaluierung – Einfluss von t_{LPL}



- Größerer Wert für t_{LPL} (länger Schlafen) reduziert nicht immer Energiebedarf
 - Es gibt einen anwendungsabhängigen, “optimalen” Wert für t_{LPL}

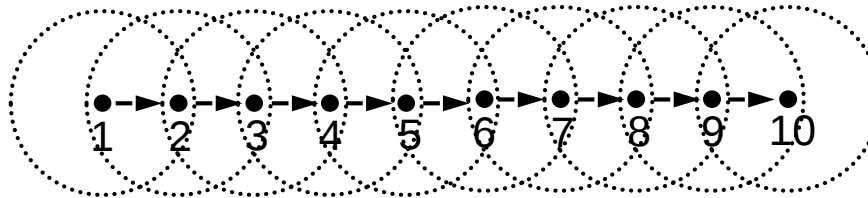
Energiebedarf TinyOS-LPL vs. S-MAC

■ Ziel

- Möglichst realitätsnahe Bewertung und ein Vergleich des Energiebedarfs
- Vergleiche Evaluierung von B-MAC vs. S-MAC

■ Versuchsaufbau

- 10 Systeme in einer Reihe, multihop



Quelle

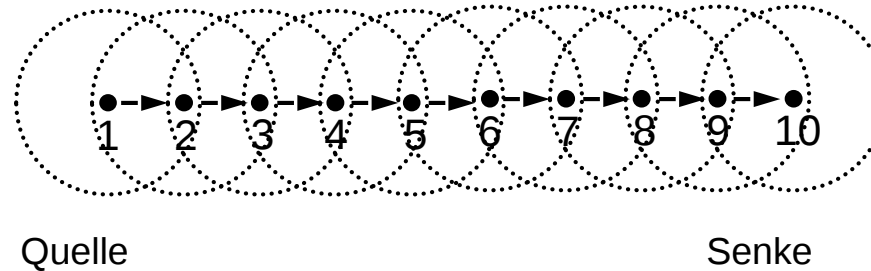
Senke

- Messung des Energiebedarfs aller beteiligten Sensorknoten
- Datenpakete fester Länge zu je 20 Byte
- MicaZ Sensorknoten

■ Untersuchte Protokollvarianten

- TinyOS-LPL mit 100ms und 1000ms Sleep Time
- S-MAC ohne Adaptive Listening, 2% und 10% Duty-Cycle

Ergebnisse TinyOS-LPL vs. S-MAC

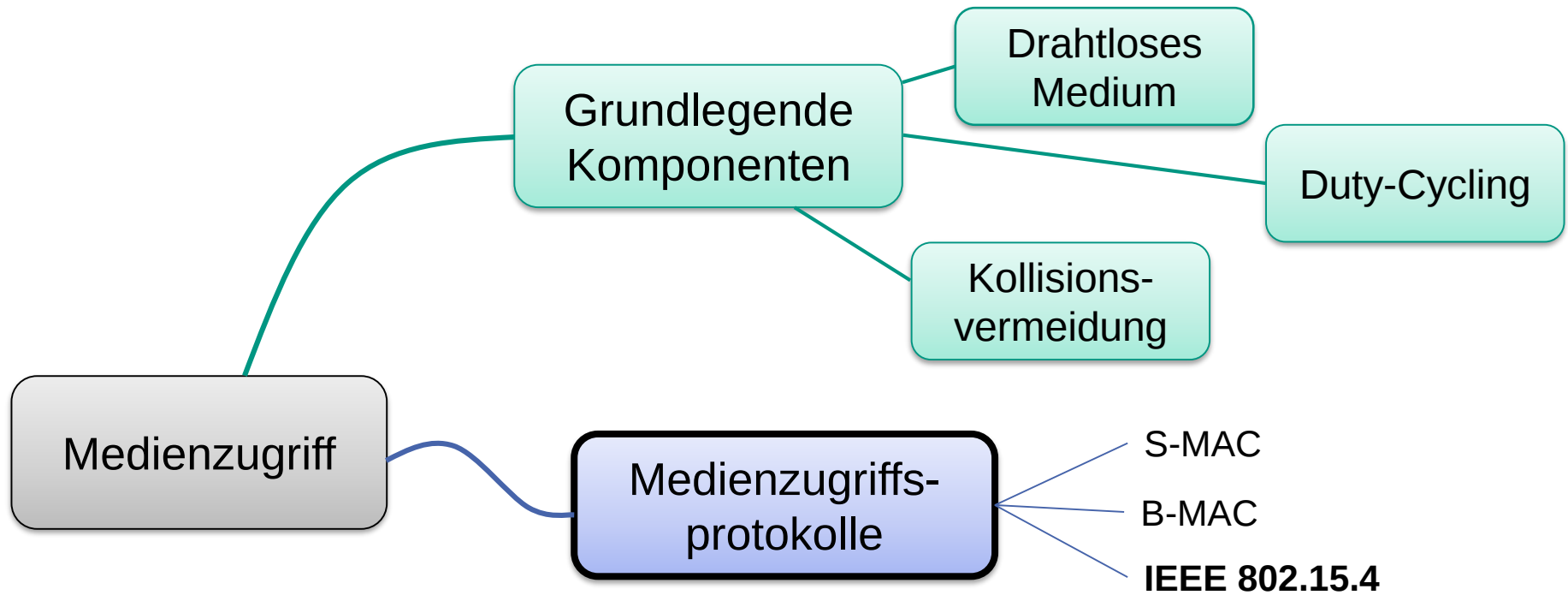


■ Gemessene Größen: Energiebedarf vs. Verzögerung

	TinyOS-LPL 100ms	TinyOS-LPL 1000ms	S-MAC 2%	S-MAC 10%
Energiebedarf [J]	0,0512	0,511	2,8448	1,376
Latenz [s]	0,140	59,041	78,3362	14,257

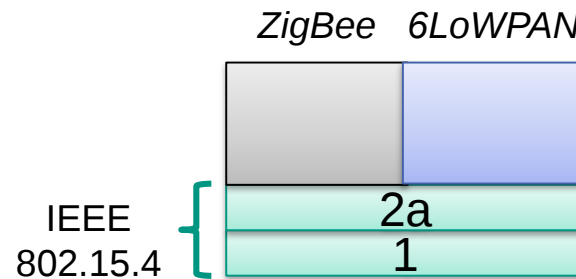
→ In diesem Szenario hat TinyOS-LPL geringeren Energiebedarf und kleinere Latenz!

Durch die IEEE standardisiertes Protokoll



IEEE 802.15.4

- IEEE Standard für Low-Rate WPAN Anwendungen
 - Entwickelt von der IEEE 802.15 Working Group, Task Group 4
 - IEEE 802.15.4 definiert physikalische Schicht und Medienzugriff
- Wird z.B. von ZigBee und 6LoWPAN genutzt



- Ziele
 - Kleine bis mittlere Datenraten
 - Moderate Verzögerungen
 - Geringer Energiebedarf (Batteriebetrieb Monate und Jahre)
 - Geringe Komplexität
- Zielanwendungen
 - Sensoren, interaktive Spielzeuge, Smart badges (Intelligente Etiketten), Fernsteuerungen, Gebäudeautomatisierung

Allgemeines

■ Eigenschaften

- Frequenzbänder: 868/914/2450 MHz
- Datenrate von max. 250 kbit/s
- Reichweite: 10 m
- ... viele unterschiedliche Ausprägungen der physikalischen Schicht definiert

■ MAC-Protokoll

- Größe der Dateneinheiten max. 127 Byte
- Kombiniert Zeitplan-basiertes und Konkurrenz-basiertes Verfahren
 - Realzeitfähigkeit durch garantierte Zeitschlitz
- Asymmetrisch: Systeme können unterschiedliche Rollen annehmen
- Basiert auf CSMA/CA
 - Carrier Sense Multiple Access with Collision Avoidance
 - Auch bei WLAN verwendet

Grundlagen

- Zwei verschiedene Typen von Netzen möglich
 - **Peer-to-Peer-Netze** (=„Mesh“) und
 - **Stern-Netze** (=„Star“)

- Zwei verschiedene Klassen von Systemen
 - Vollfunktionstüchtige Systeme (Full Function Devices, FFD)
 - Eingeschränkte Systeme (Reduced Function Devices, RFD)
 - Nur in Stern-Netzen

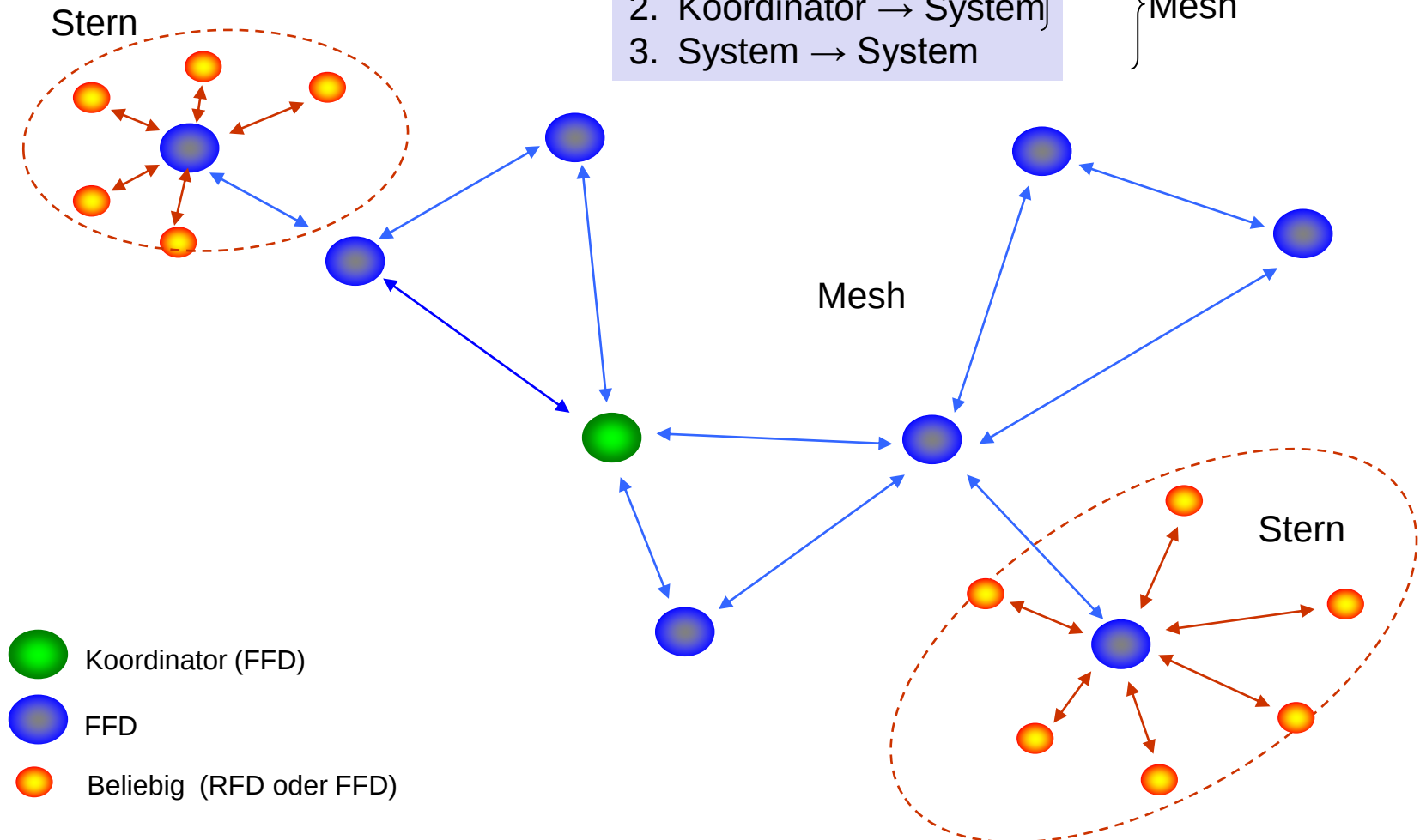
- Jedes Netz besitzt spezielles FFD, den „**Koordinator**“ (Coordinator)
 - Organisiert Netz mittels periodisch versendeter Beacons
 - Synchronisation der Systeme, Aufforderung zum Senden, Identifikation des Netzes ...

- Zwei Betriebsmodi
 - Beacon Modus
 - Non-Beacon Modus

Netzbeispiel

Kommunikationstypen:

1. System $\rightarrow$ Koordinator
 2. Koordinator $\rightarrow$ System
 3. System $\rightarrow$ System
- } Star } Mesh



Beacon Modus

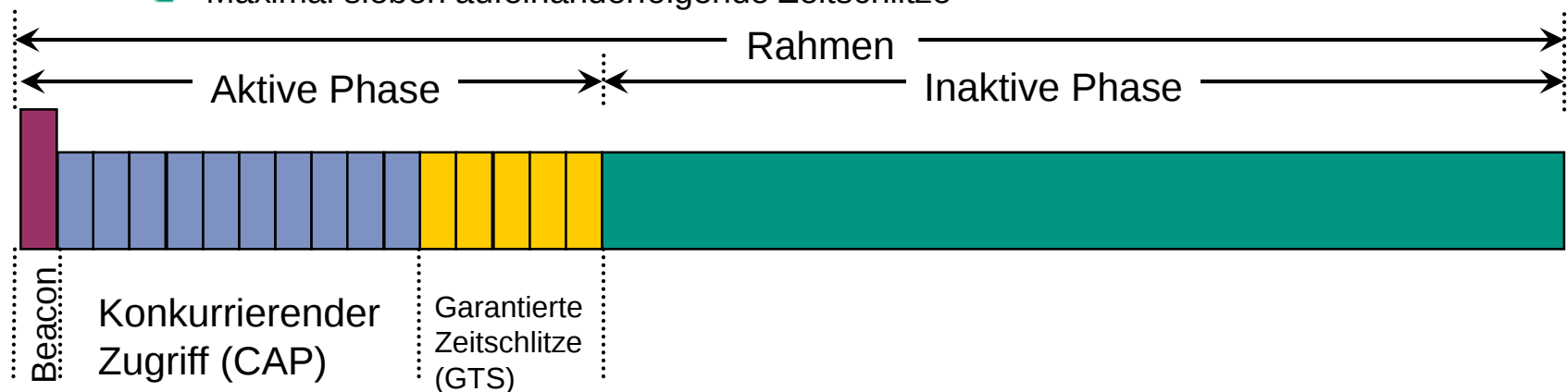
- Stern-Netz
 - Systeme sind Koordinator zugeordnet
 - Formieren ein PAN (Personal Area Network)
 - durch PAN-Identifikator identifiziert

- Koordinator
 - Führt Buch über Systeme und die Vergabe von Adressen
 - Sendet regelmäßig Beacons
 - Z.B. mit PAN-Identifikator
 - Bearbeitet Anforderungen für garantierte Zeitslitze
 - Vermittelt zwischen Systemen und Peer Koordinatoren

- Grundlegende Annahme
 - Für Koordinatoren stellt Energie keine limitierte Ressource da
 - Koordinatoren übernehmen mehr Aufgaben als andere Systeme

Rahmenstruktur

- Rahmen haben eine feste Länge
- Inaktive Phase
 - Alle Systeme können Transceiver ausschalten
- Aktive Phase
 - Dauer von 16 Zeitschlitzten, Koordinator muss aktiv sein
 - CAP: Contention Access Period
 - Zugriff durch Slotted CSMA/CA
 - GTS: Guaranteed Time Slots
 - Zeitschlitzte müssen vom System beim Koordinator angefordert werden
 - Reservierung gilt bis Zeitschlitzte wieder freigegeben werden
 - Maximal sieben aufeinanderfolgende Zeitschlitzte



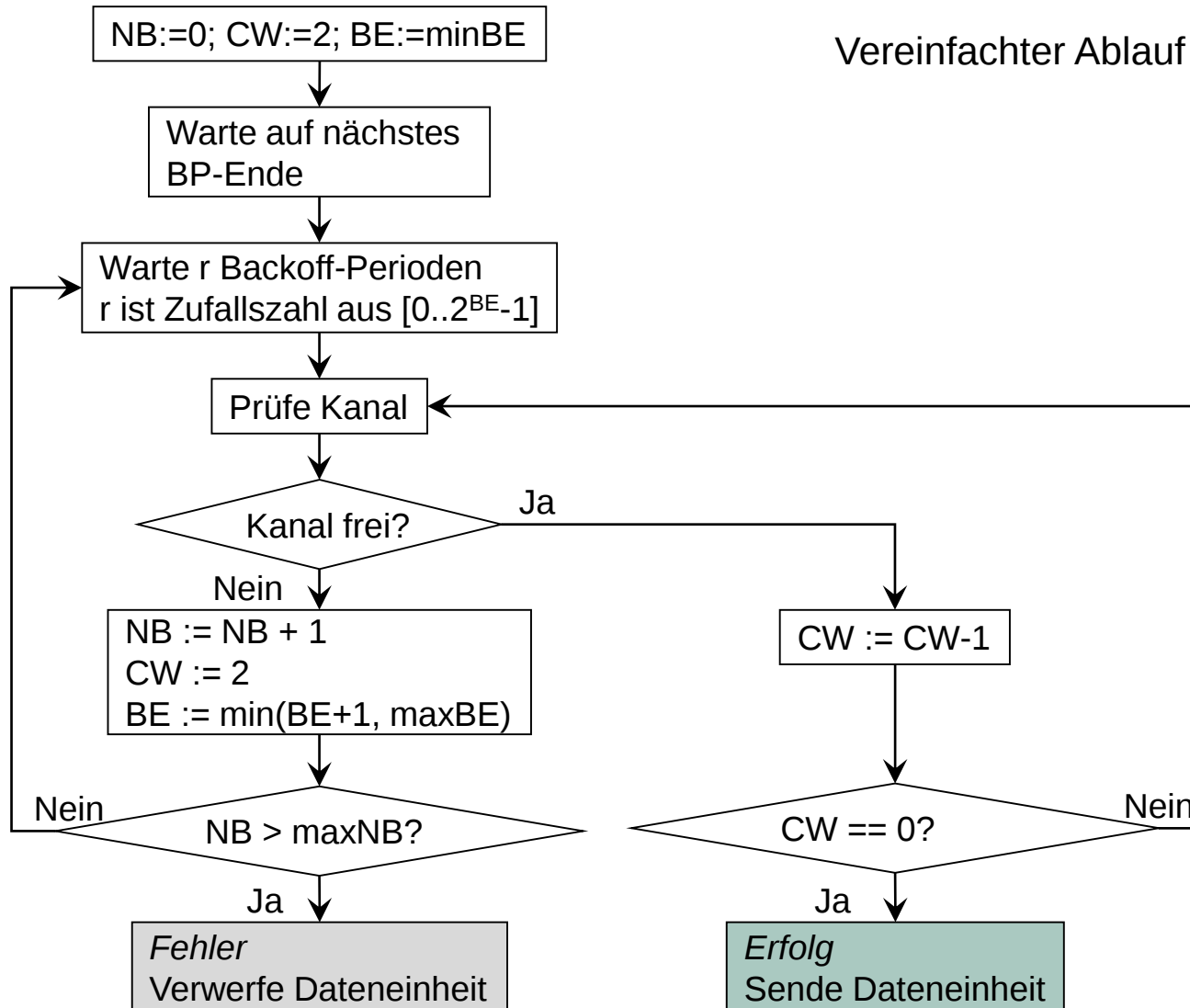
Konkurrierender Zugriff während CAP

- Slotted CSMA/CA regelt Medienzugriff
 - CSMA: Carrier Sense Multiple Access
 - Kanal abhören (CS), bevor auf den Kanal gesendet wird
 - Kein Schutz vor versteckten Endsystemen (kein RTS/CTS)
 - CA: Collision Avoidance
 - Exponentieller Backoff zur Kollisionsvermeidung

- Verwendete Variablen
 - CAP Zeitschlitz in Backoff Perioden (BP) untergliedert
 - NB: Anzahl der bisher durchgeführten Backoffs
 - maxNB: Maximalzahl durchzuführender Backoffs
 - CW: Contention Window
 - BE: Backoff Exponent
 - minBE: mindest-Backoff Exponent
 - maxBE: maximal-Backoff Exponent

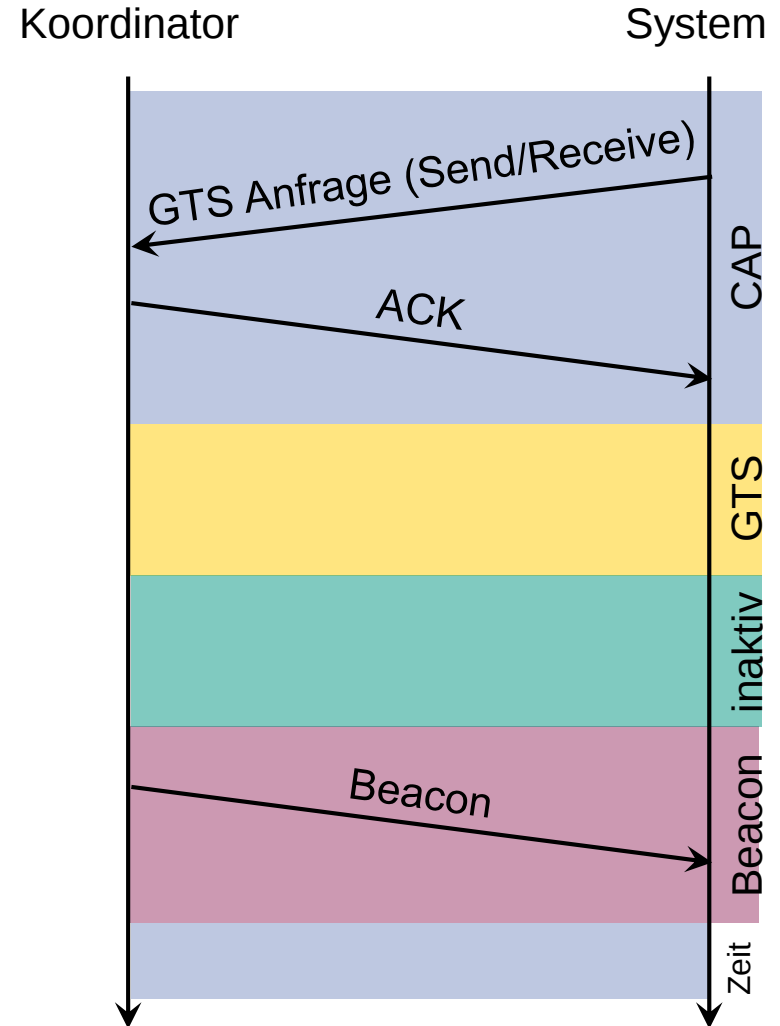
Slotted CSMA/CA Ablauf

Vereinfachter Ablauf aus  [KaWi05]



Garantierte Zeitschlitz (GTS)

- Vom Koordinator verwaltet
- GTS-Anfrage an Koordinator
 - Anforderung von GTS-Zeitschlitz
 - Unterscheidung von Sende- und Empfangs-Zeitschlitz
- Antwort des Koordinators
 - Erfolgt in zwei Schritten
 - Unverzögertes ACK bestätigt Empfang
 - Keine Information über Erfolg der Anfrage
 - Zugeordnete GTS-Zeitschlitz in einem der nächsten Beacon bekannt gegeben
 - Kann sich verzögern, falls Ressourcen nicht ausreichen
- Zeitschlitz des Rahmens bleibt zugeordnet
 - Bis System diesen explizit frei gibt
 - Bis Koordinator feststellt, dass Schlitz eine bestimmte Dauer nicht genutzt wurde



Datenübertragung

■ System zu Koordinator

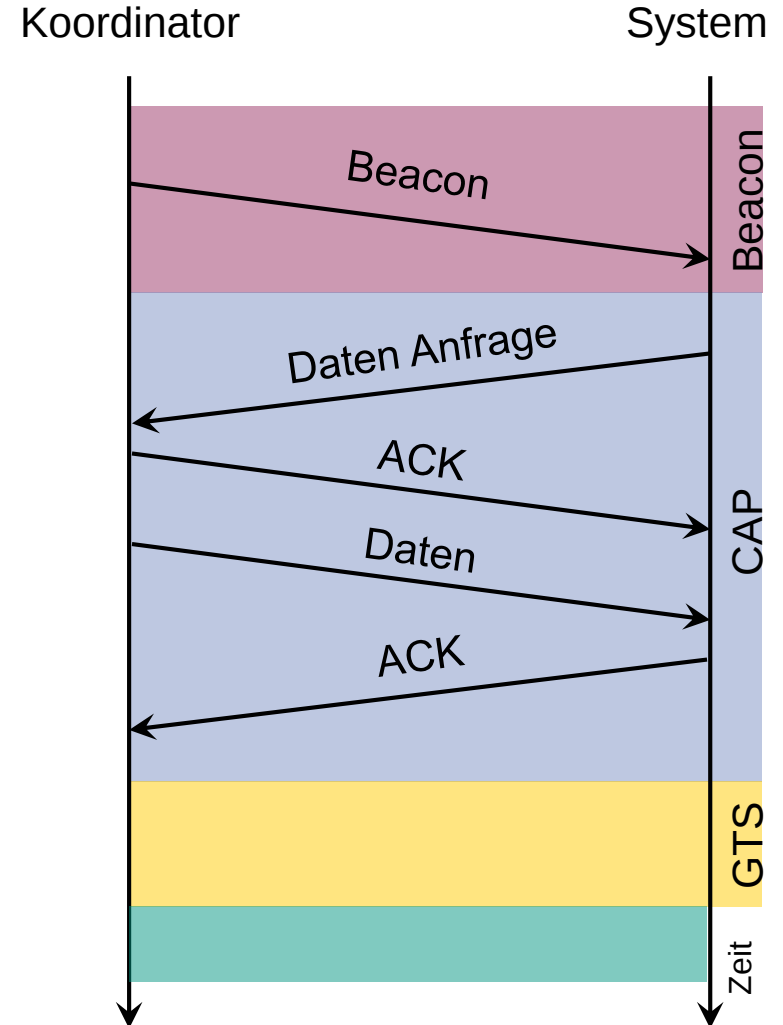
- Falls System GTS-Zeitschlitz (send) reserviert hat
 - Zeitschlitz kann direkt genutzt werden
 - Dateneinheit + IFS + ACK müssen in den Zeitschlitz passen
 - IFS = Inter Frame Space, ACK = Acknowledgement
- Sonst
 - Sende während CAP, mittels Slotted CSMA/CA

■ Koordinator zu System

- Falls System GTS-Zeitschlitz (receive) reserviert hat
 - Koordinator nutzt reservierten Zeitschlitz
 - System antwortet mit ACK (im selben Zeitschlitz)
- Sonst
 - ... (auf der nächsten Folie) ...

Datenübertragung

- Sonst
 - ... fortgesetzt
 - Koordinator integriert Sendewunsch in Beacon
 - Adresse des Systems im „pending address“ Feld
 - System antwortet mit Daten Anfrage
 - Koordinator quittiert Empfang der Anfrage (ACK)
 - Koordinator sendet Daten
 - System quittiert Empfang der Daten
- Bei Fehlern wiederholt das System die Anfrage während des nächsten Rahmens



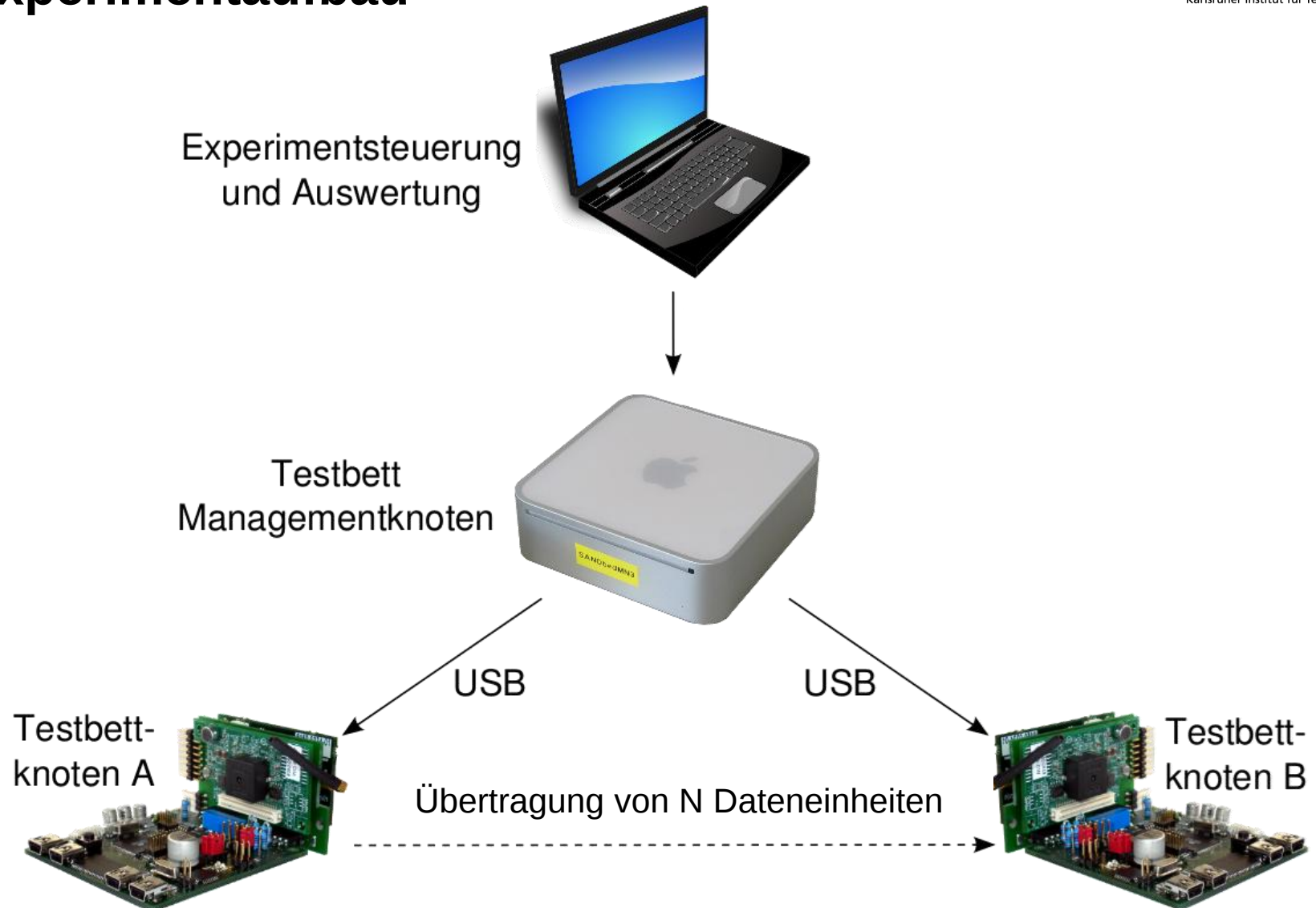
Non-Beacon Modus

- Bisher: Beacon Modus mit garantierten Zeitschlitzten
- 802.15.4 bietet darüber hinaus auch Non-Beacon Modus
- Rahmenstruktur
 - Keine feste Rahmenstruktur
 - Zugriff auf das Medium durch unslotted CSMA/CA
 - Wegen fehlender Zeitsynchronisation keine Zeitschlitzte möglich
 - Koordination oder Synchronisation nicht Teil des Standards
- Peer-to-Peer Netz: Systeme können untereinander frei kommunizieren
 - Formieren ein PAN, durch PAN-Identifikator identifiziert
 - PAN Koordinator wird frei gewählt, etwa der erste kommunizierende Knoten
 - Koordinator muss immer aktiv sein

Energiebedarf von MAC-Protokollen

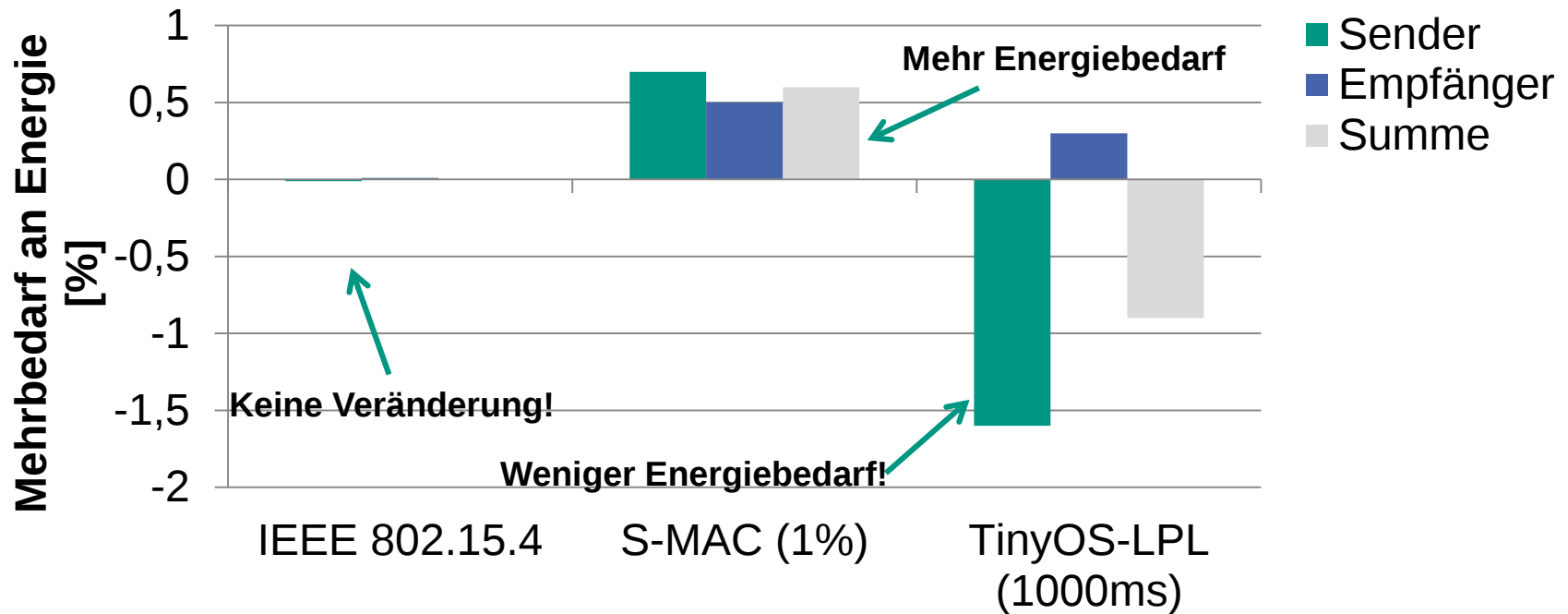
- Welchen Einfluss haben MAC-Protokolle auf den Energiebedarf von Kommunikationsvorgängen?
- Häufige Annahme bei der Evaluation des Energiebedarfs
 - Höhere Anzahl an Datenpaketen, mehr Datenvolumen
→ höherer Energiebedarf
- Ist diese Annahme immer gültig ?
 - Untersuchung eines einfachen Beispiels
 - Übertragung von N Dateneinheiten mit jeweils S Byte Payload
 - Messung des Energiebedarfs an den beteiligten Sensorknoten über einen Messzeitraum von 10 Sekunden
 - Nutzung unterschiedlicher Medienzugriffsverfahren
 - TinyOS LPL, S-MAC und IEEE 802.15.4

Experimentaufbau



Evaluierung – Einfluss der Paketgröße

Szenario: $N=1$ Paket, $S=1$ Byte Payload vs. $S=90$ Byte Payload
Metrik: Energiemehrbedarf bei 10 Sekunden Betrieb

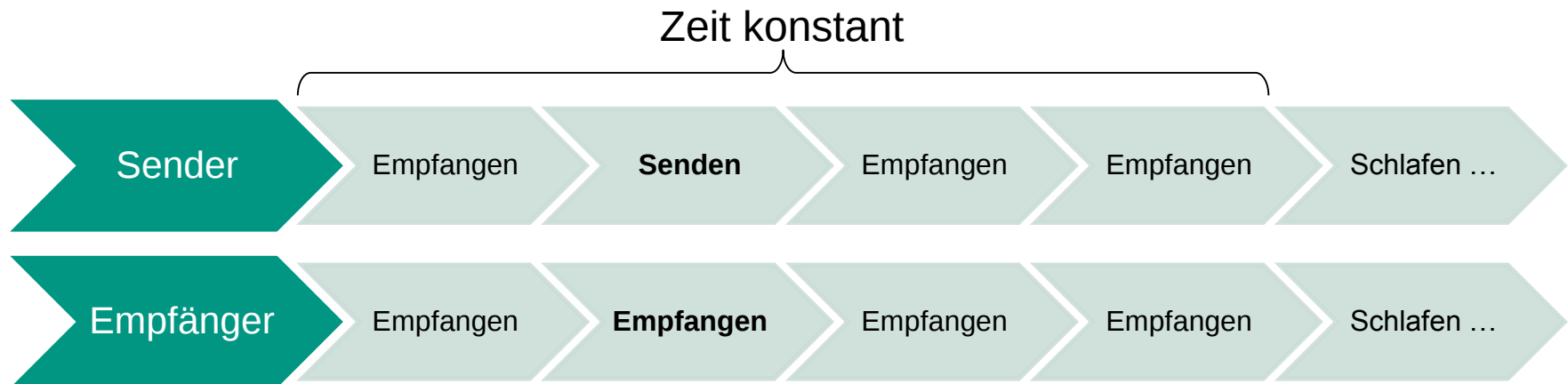
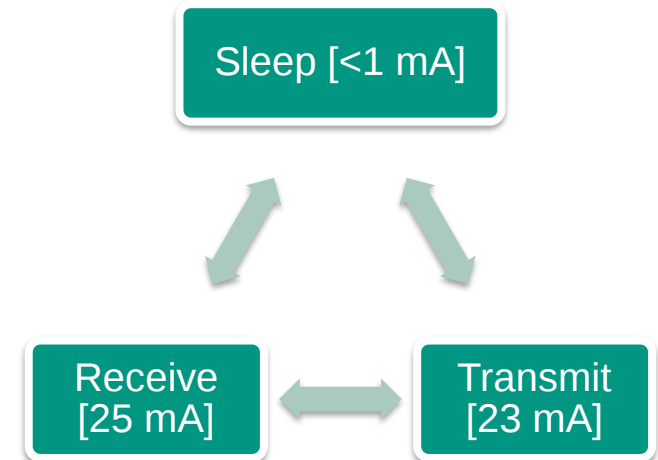


- Je nach MAC-Protokoll mehr Energiebedarf, weniger Energiebedarf oder gleicher Energiebedarf für das Versenden einer größeren Dateneinheit

→ Mehr Datenvolumen führt nicht automatisch zu mehr Energiebedarf

Evaluierung – Einfluss der Paketgröße

- Leistungsaufnahme: IEEE 802.15.4
 - Funktransceiver länger im Zustand „Transmit“, kürzer im Zustand „Receive“
 - Für die längere Dauer der Übertragung, ca. 3 ms
 - Unterschied der Leistungsaufnahme beider Zustände minimal, ca. 2 mA
- Unterschied minimal

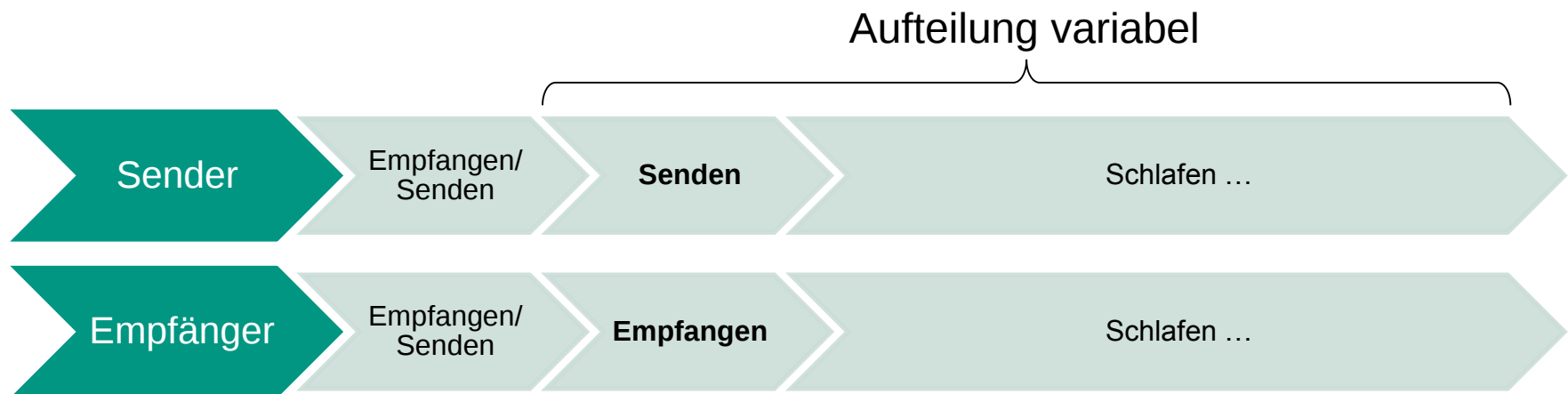
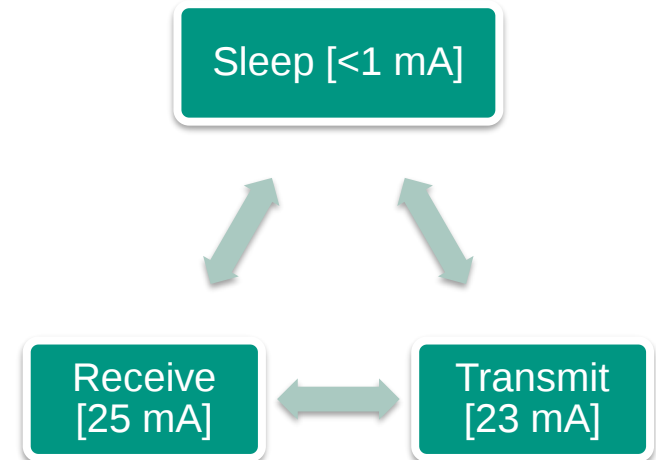


Evaluierung – Einfluss der Paketgröße

■ Leistungsaufnahme: S-MAC

- Funktransceiver länger im Zustand „Transmit“, kürzer im Schlafzustand
 - Unterschied der Leistungsaufnahme recht groß, ca. 25 mA

→ Geringer Mehrbedarf für die größere Dateneinheit

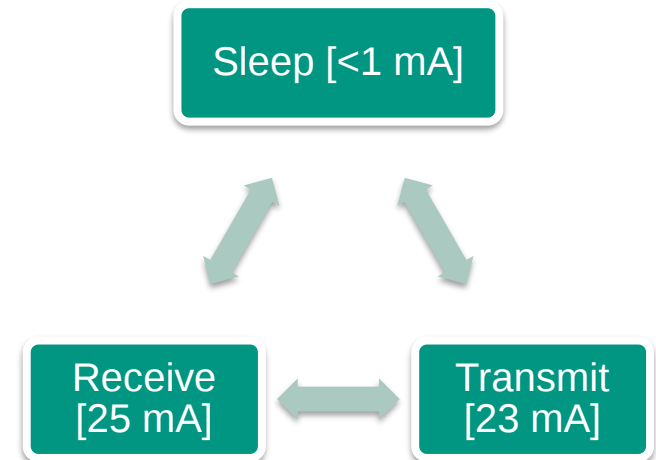


Evaluierung – Einfluss der Paketgröße

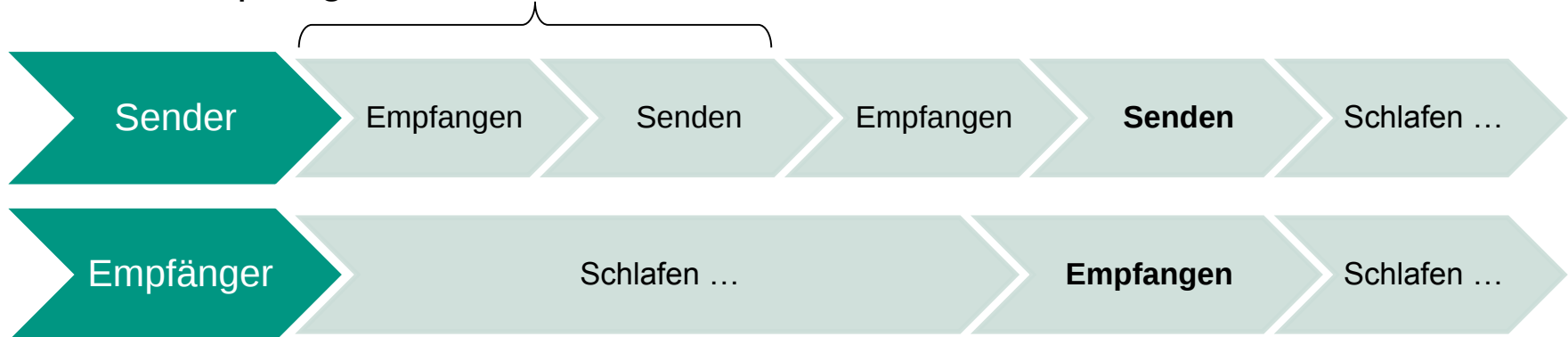
■ Leistungsaufnahme: TinyOS-LPL

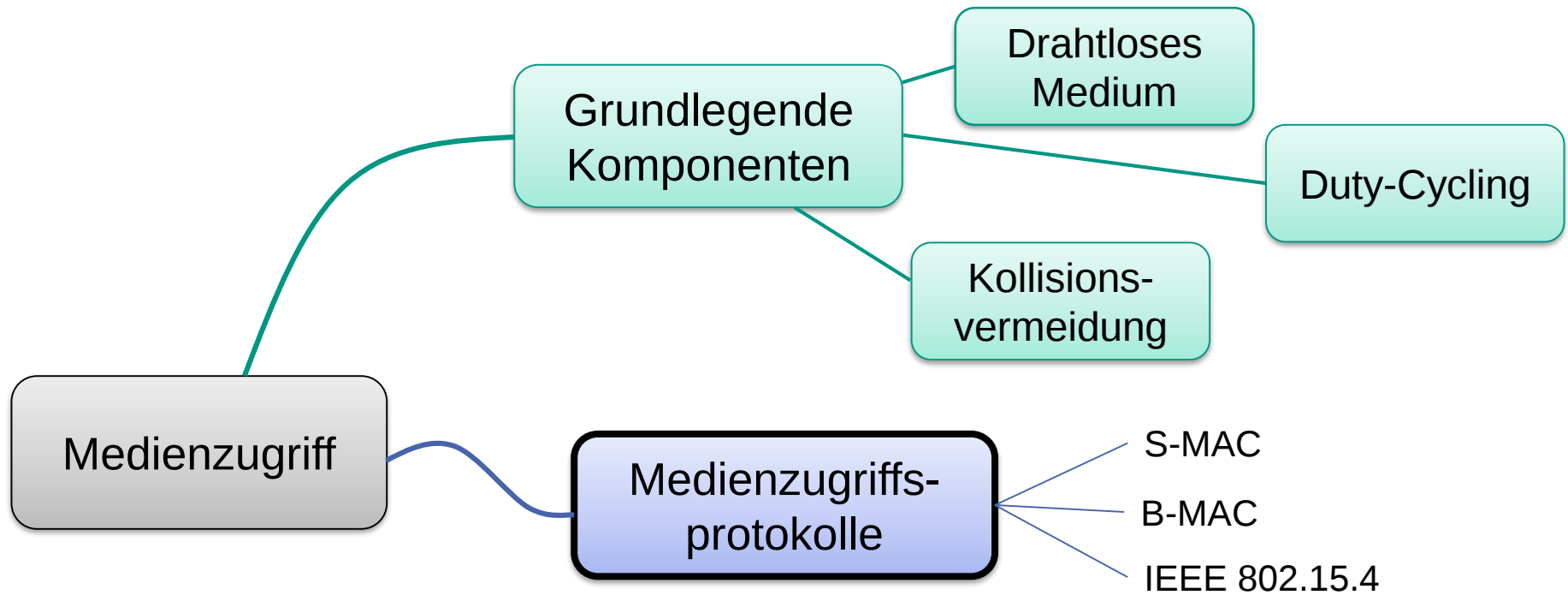
- Funktransceiver deutlich länger im Zustand „Transmit“ (Präambel!), kürzer im Zustand „Receive“
 - Senden ist billiger als nur empfangen, unterschied ca. 2 mA
 - Für längeren Zeitraum!

→ Weniger Energiebedarf für die größere Dateneinheit



Empfangen konstant, Senden variabel





Zusammenfassung Medienzugriff

- Aus der Forschung existiert eine Vielzahl von Ideen zum Medienzugriff
 - Systematische Vergleiche hinsichtlich Leistungsfähigkeit und Implementierbarkeit fehlen häufig
- Einige Protokolle haben sich in der Praxis durchgesetzt
 - **B-MAC** in der Forschung
 - Wird/wurde häufig für prototypische Implementierungen genutzt, da es für diese als Implementierung vorlag
 - **IEEE 802.15.4** ist ein internationaler Standard der im kommerziellen Umfeld genutzt wird
 - Als Basis bei ZigBee und 6LoWPAN verwendet
 - Implementierungen von abgewandeltem B-MAC bzw. X-MAC in TinyOS
 - **Bluetooth Smart**
 - Muss noch abgewartet werden, ob sich das durchsetzt

Literatur I

- [Buettner06] M. Buettner, G. Yee, E. Anderson, R. Han, X-MAC: A Short Preamble MAC Protocol For Duty-Cycled Wireless Sensor Networks, SenSys '06: Proceedings of the 4th international conference on Embedded networked sensor systems, ACM Press, 2006, 307-320
- [ChDH06] Feng Chen, Falko Dressler & Armin Heindl. End-to-end performance characteristics in energy-aware wireless sensor networks PE-WASUN '06: Proceedings of the 3rd ACM international workshop on Performance evaluation of wireless ad hoc, sensor and ubiquitous networks, ACM Press, 2006, 41-47
- [Erge04] Sinem C. Ergen, ZigBee/ IEEE 802.15.4 Summary, Internal Report to Advanced Technology Lab of National Semiconductor, August 2004.
- [KaWi05] Holger Karl & Andreas Willig, Protocols and Architectures for Wireless Sensor Networks, Wiley, 2005, Kap 5
- [KuRD06] Kumar, S.; Raghavan, V. S. & Deng, J. Medium Access Control Protocols for Ad-Hoc Wireless Networks: A Survey, Elsevier Ad Hoc Networks Journal, 2006, 4, 326-358
- [LeDB03] Martin Leopold, Mads Bondo Dydensborg, & Philippe Bonnet. Bluetooth and Sensor Networks: A Reality Check Proceedings of the 1st International Conference on Embedded Networked Sensor Systems, 2003, 103 - 113
- [PoHC04] J. Polastre, J. Hill & D. Culler. Versatile low power media access for wireless sensor networks SenSys '04: Proceedings of the 2nd international conference on Embedded networked sensor systems, ACM, 2004, 95-107

Literatur II

- [Rapp99] T. S. Rappaport, Wireless Communications, Prentice Hall, 1999
- [RaOGLA03] V. Rajendran, K. Obraczka, & J.J. Garcia-Luna-Aceves. Energy-efficient collision-free medium access control for wireless sensor networks SenSys '03: Proceedings of the 1st international conference on Embedded networked sensor systems, ACM Press, 2003, 181-192
- [vDaLa03] van Dam, T. & Langendoen, K. An adaptive energy-efficient MAC protocol for wireless sensor networks SenSys '03: Proceedings of the 1st international conference on Embedded networked sensor systems, ACM Press, 2003, 171-180
- [YeHE02] Wei Ye; John Heidemann & Deborah Estrin. An energy-efficient MAC protocol for wireless sensor networks INFOCOM 2002. Twenty-First Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings. IEEE, 2002, 3, 1567-1576 Vol.3
- [YeHE04] Wei Ye; John Heidemann & Deborah Estrin. Medium access control with coordinated adaptive sleeping for wireless sensor networks IEEE/ACM Transactions on Networking, IEEE Press, 2004, 12, 493-506