CoNe Freiburg

IIF INSTITUT FÜR INFORMATIK FREIBURG

Systeme II

4./5. Woche Sicherungsschicht

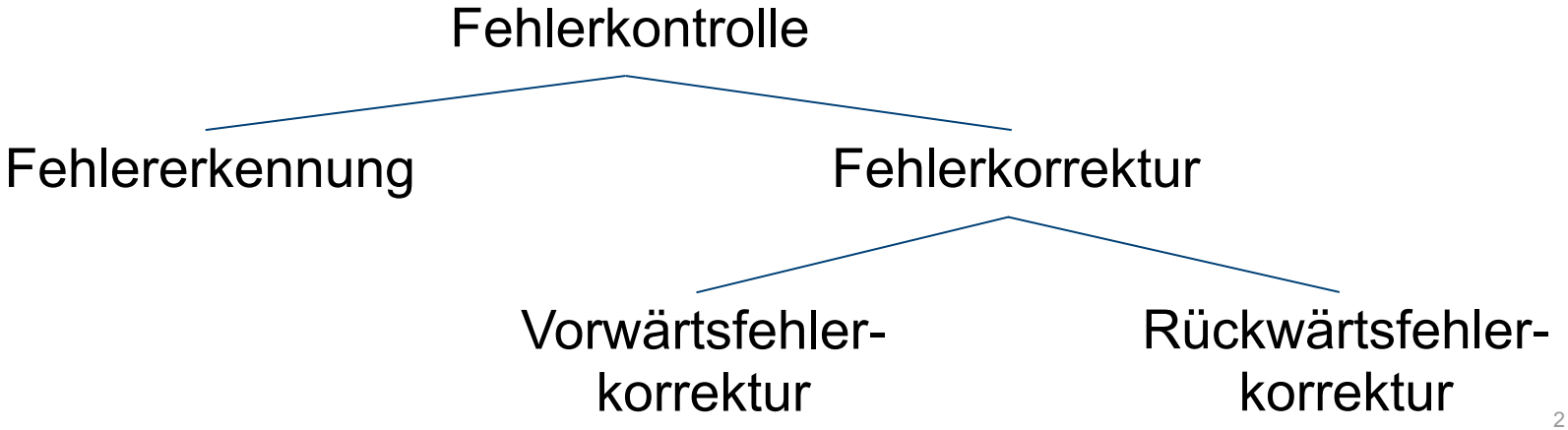
Christian Schindelhauer
Technische Fakultät
Rechnernetze und Telematik
Albert-Ludwigs-Universität Freiburg

UNI FREIBURG

Fehlerkontrolle

CoNe Freiburg

- Zumeist gefordert von der Vermittlungsschicht
  - Mit Hilfe der Frames
- Fehlererkennung
  - Gibt es fehlerhaft übertragene Bits?
- Fehlerkorrektur
  - Behebung von Bitfehlern
  - Vorwärtsfehlerkorrektur (Forward Error Correction)
    - Verwendung von redundanter Kodierung, die es ermöglicht Fehler ohne zusätzliche Übertragungen zu beheben
  - Rückwärtsfehlerkorretur (Backward Error Correction)
    - Nach Erkennen eines Fehlers, wird durch weitere Kommunikation der Fehler behoben

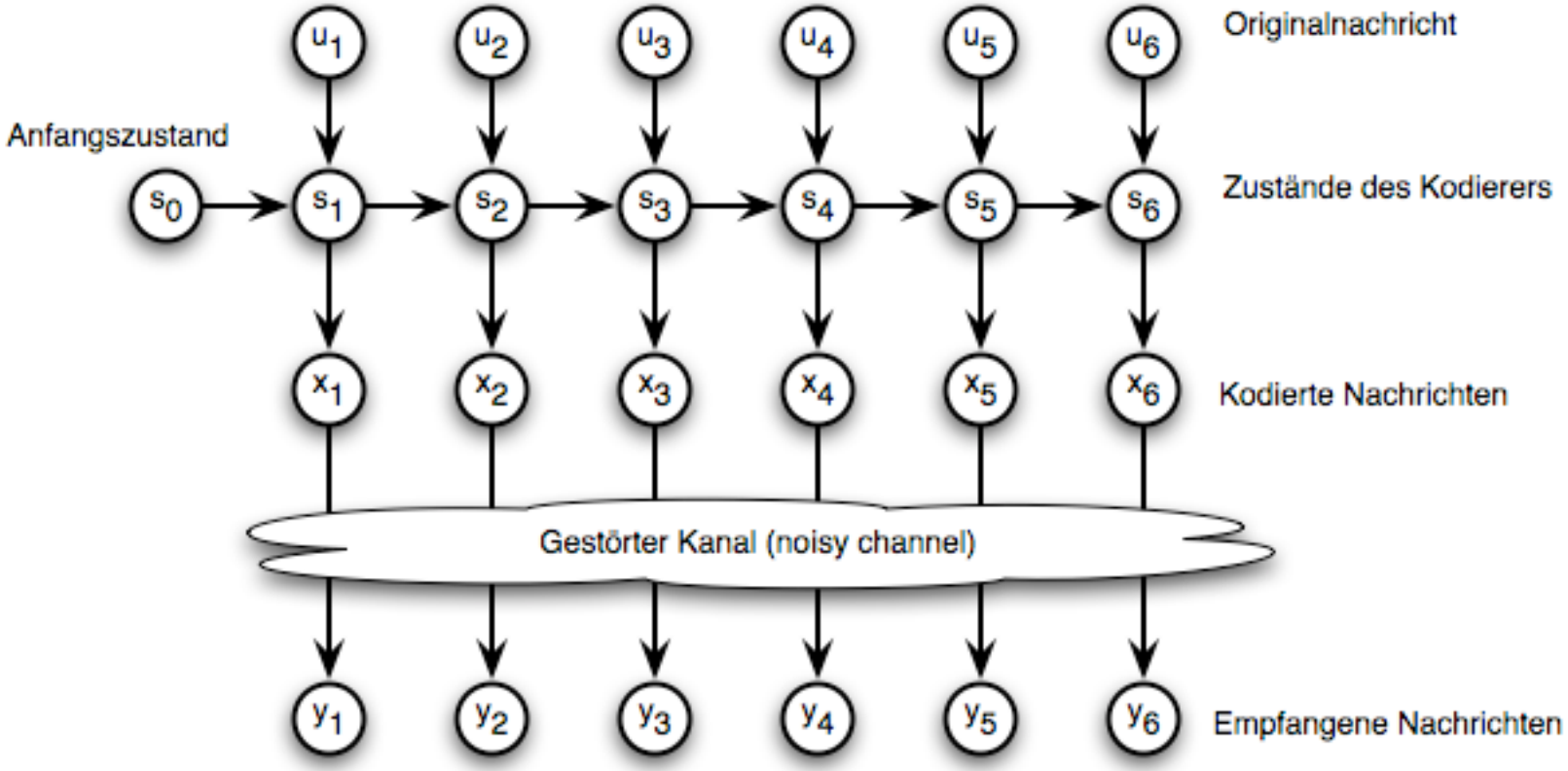


UNI FREIBURG

CoNe Freiburg

Faltungs-Codes

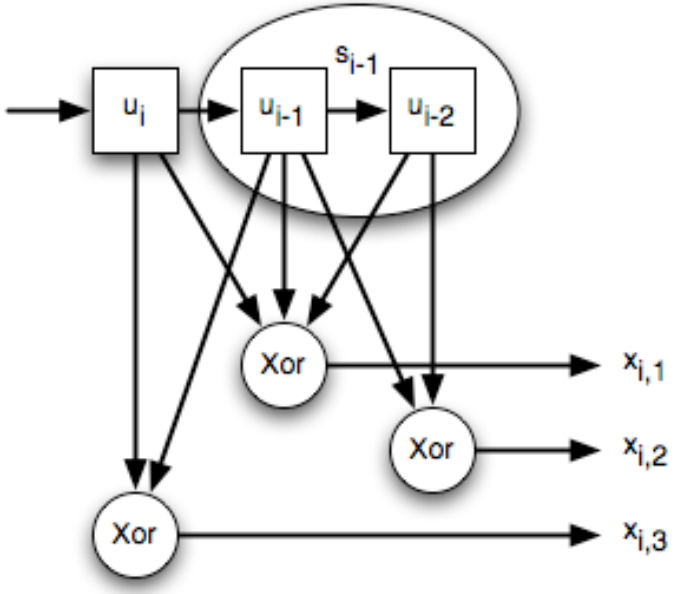
- Faltungs-Codes (Convolutional Codes)
  - Daten und Fehlerredundanz werden vermischt.
  - k Bits werden auf n Bits abgebildet
  - Die Ausgabe hängt von den k letzten Bits und dem internen Zustand ab.



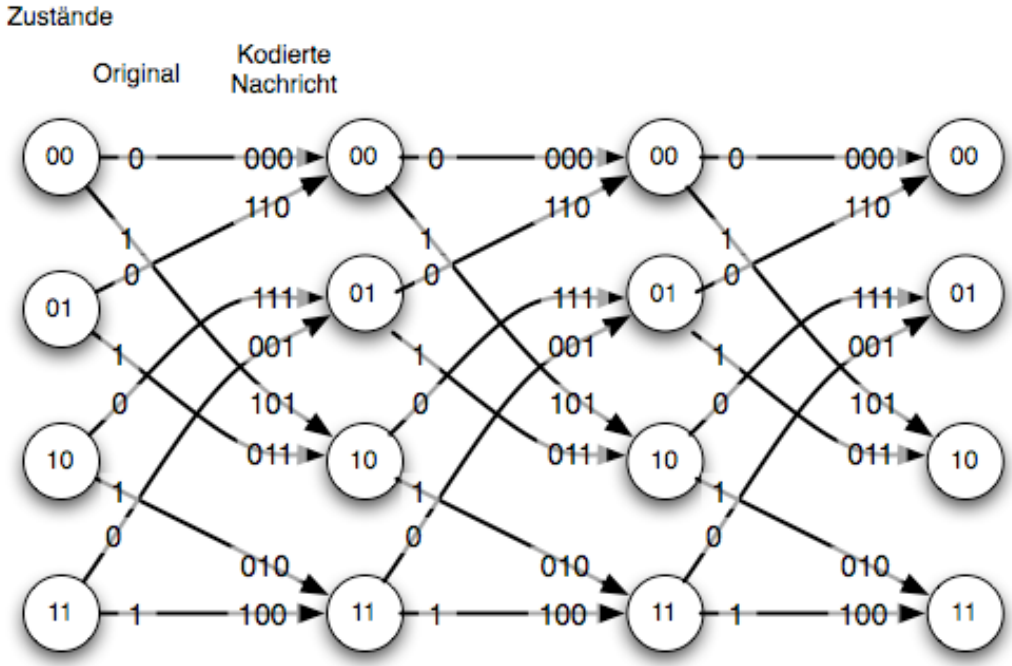
CoNe Freiburg

Beispiel

Faltungs-Kodierer



Trellis-Diagramm



CoNe Freiburg

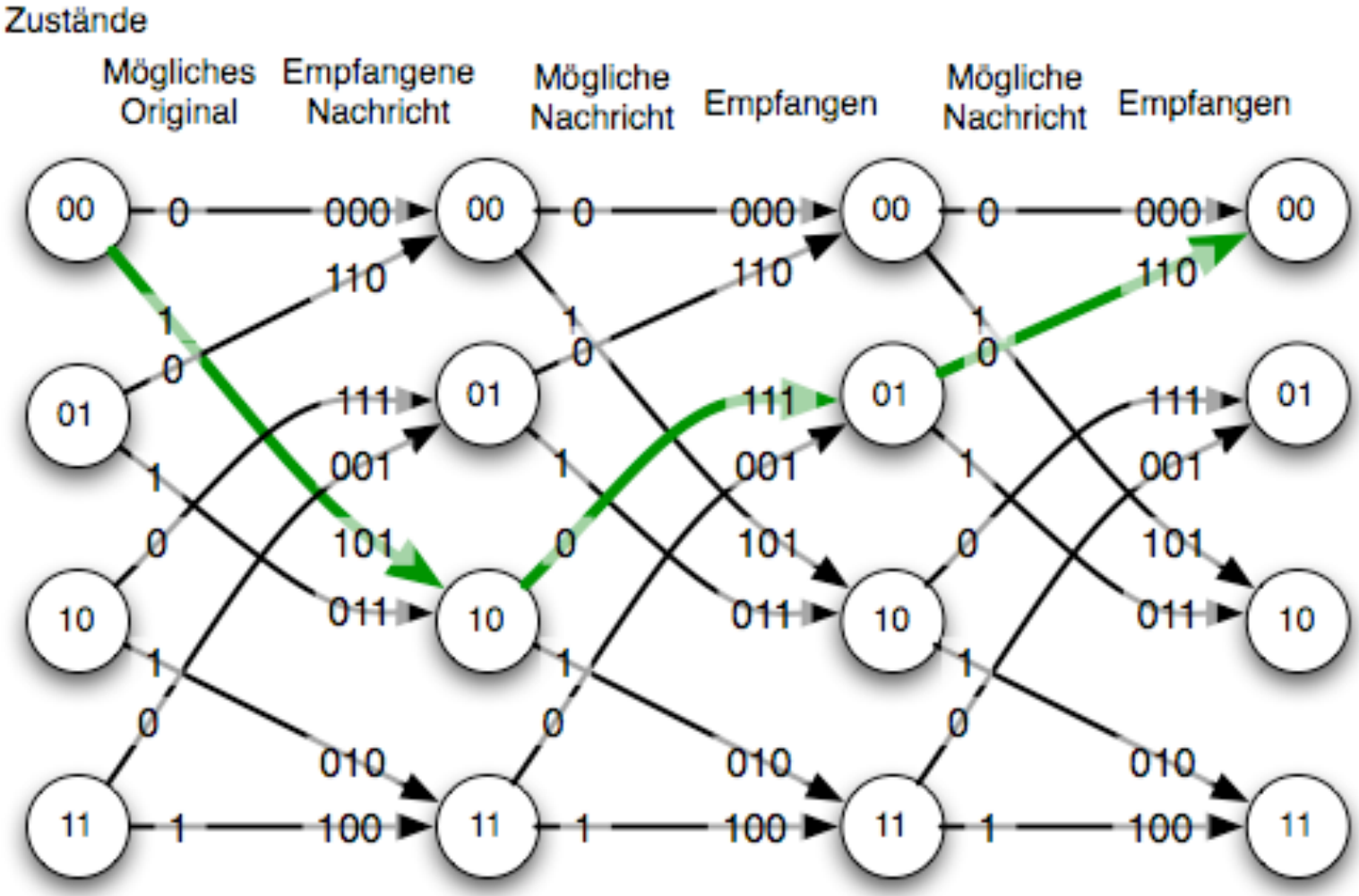
Dekodierung der Faltungs-Codes: Algorithmus von Viterbi

- Dynamische Programmierung
- Zwei notwendige Voraussetzungen für Dekodierung
  - (für den Empfänger) unbekannte Folge von Zuständen
  - beobachtete Folge von empfangenen Bits (möglicherweise mit Fehler)
- Der Algorithmus von Viterbi bestimmt die warscheinlichste Folge von Zuständen, welches die empfangenen Bits erklärt
  - Hardware-Implementation möglich

UNI FREIBURG

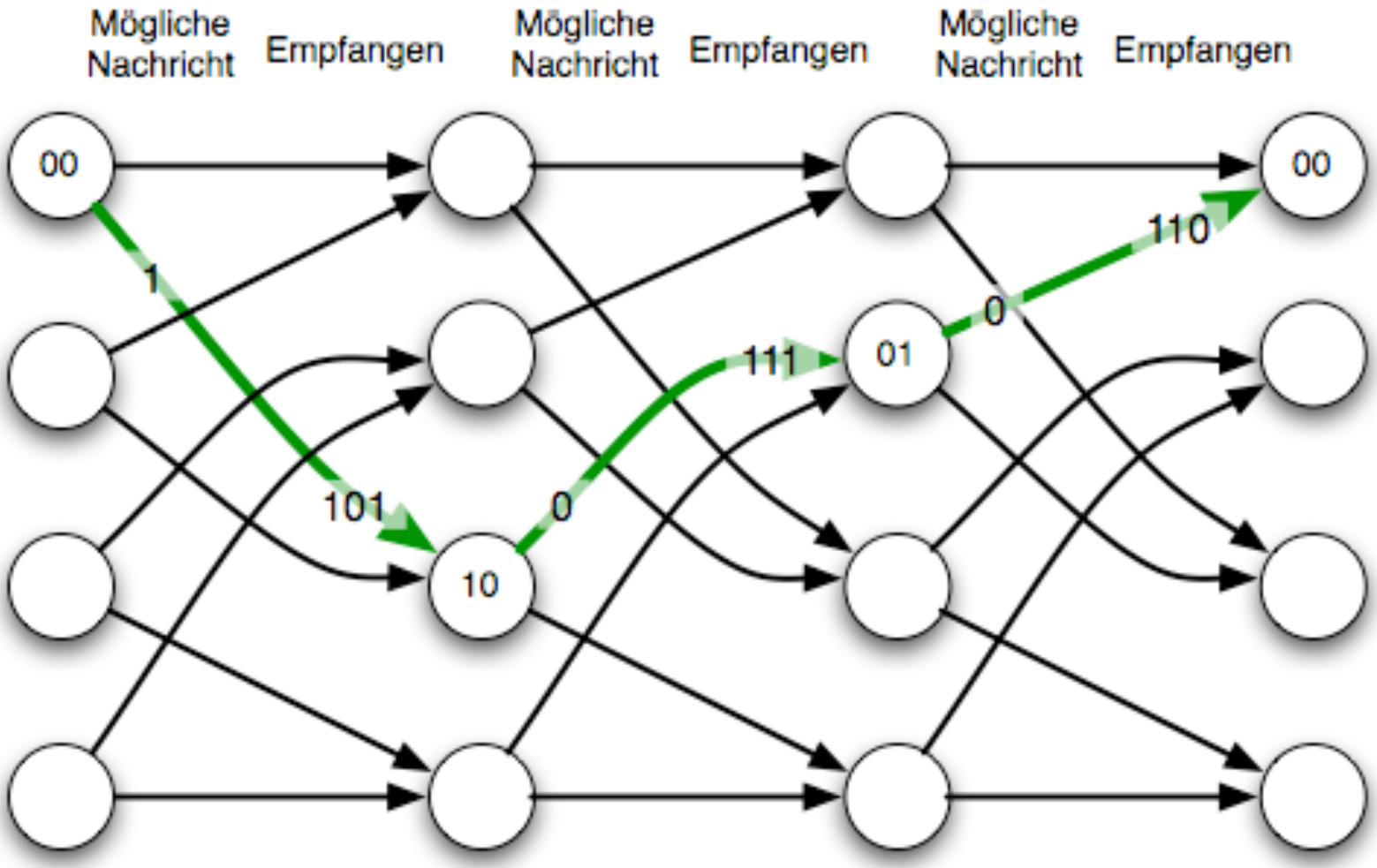
CoNe Freiburg

Dekodierung (I)



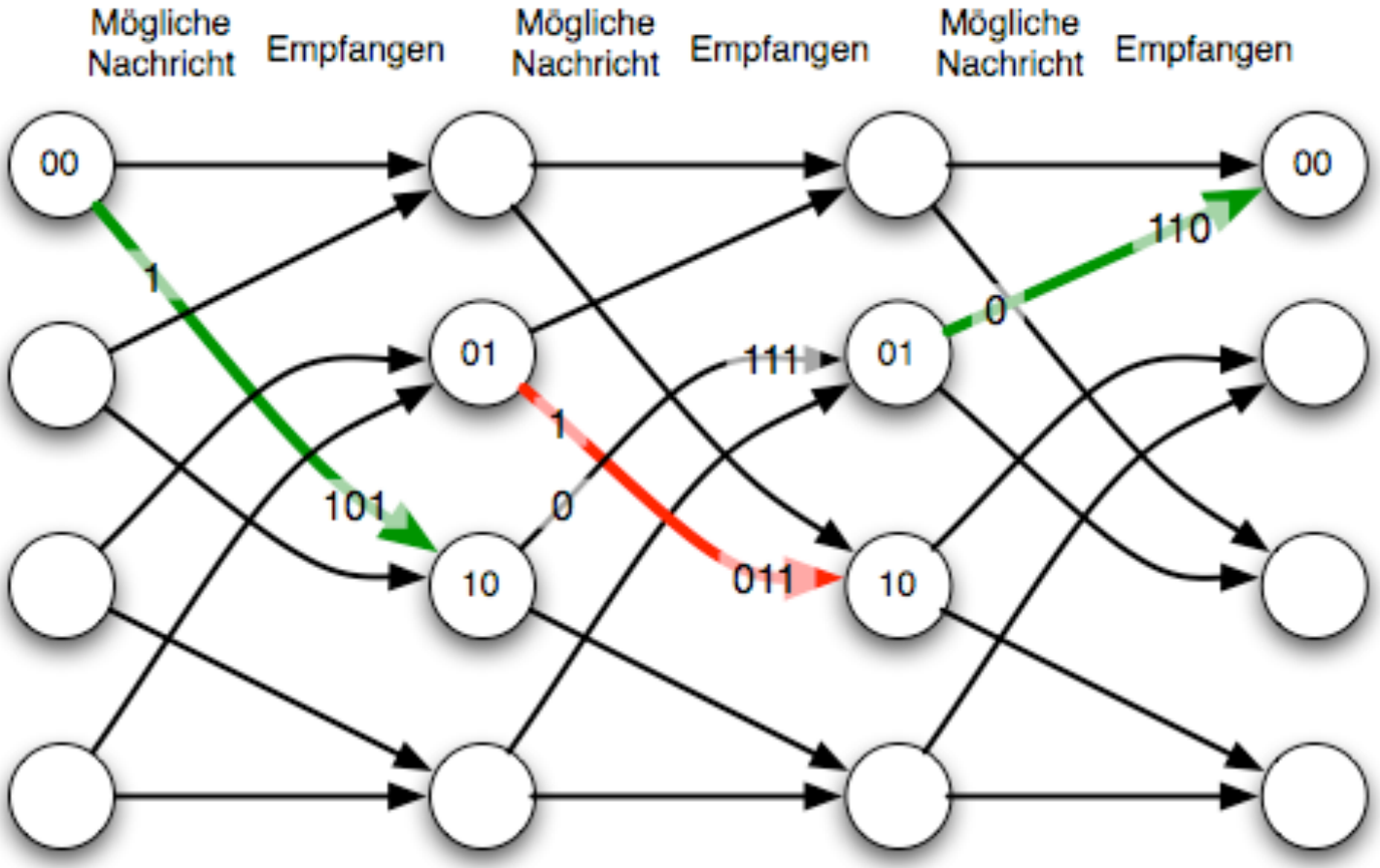
CoNe Freiburg

Dekodierung (II)



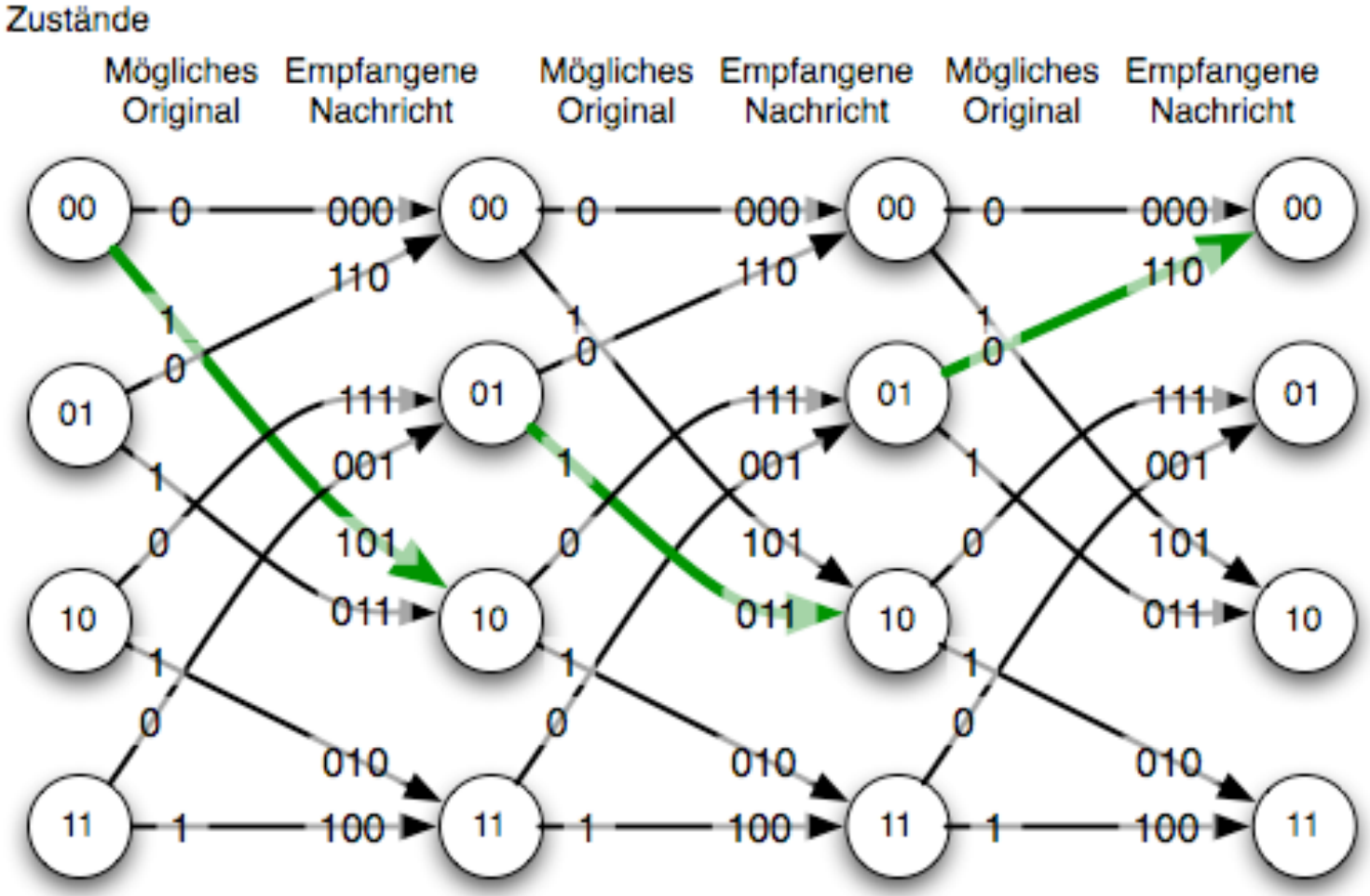
CoNe Freiburg

Dekodierung (III)



CoNe Freiburg

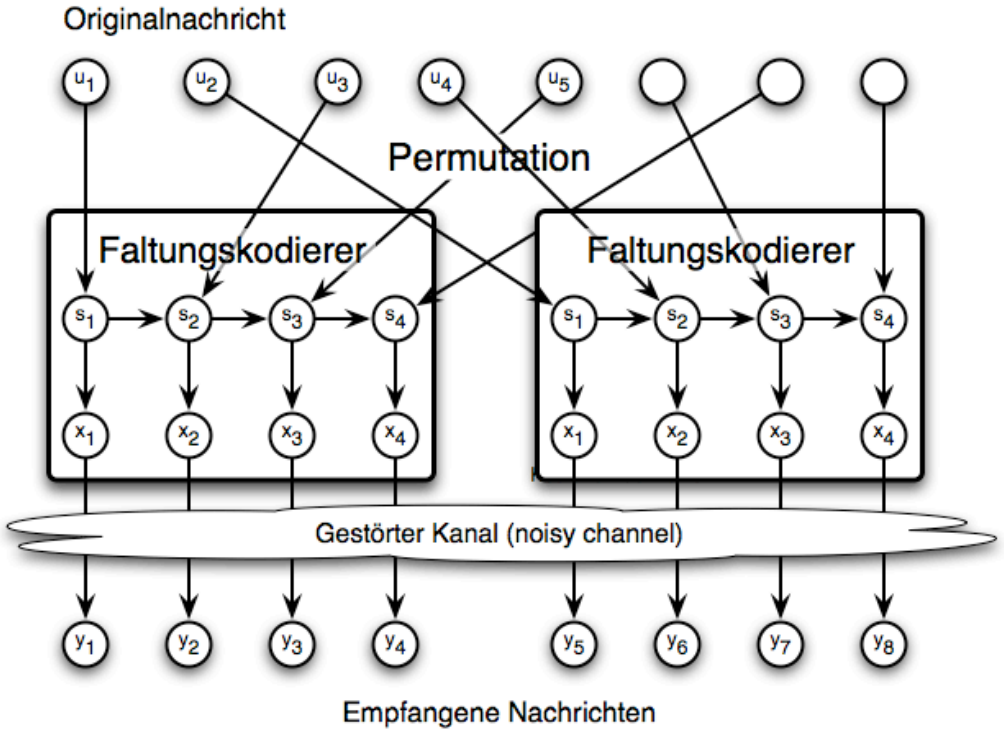
Dekodierung (IV)



CoNe Freiburg

Turbo-Codes

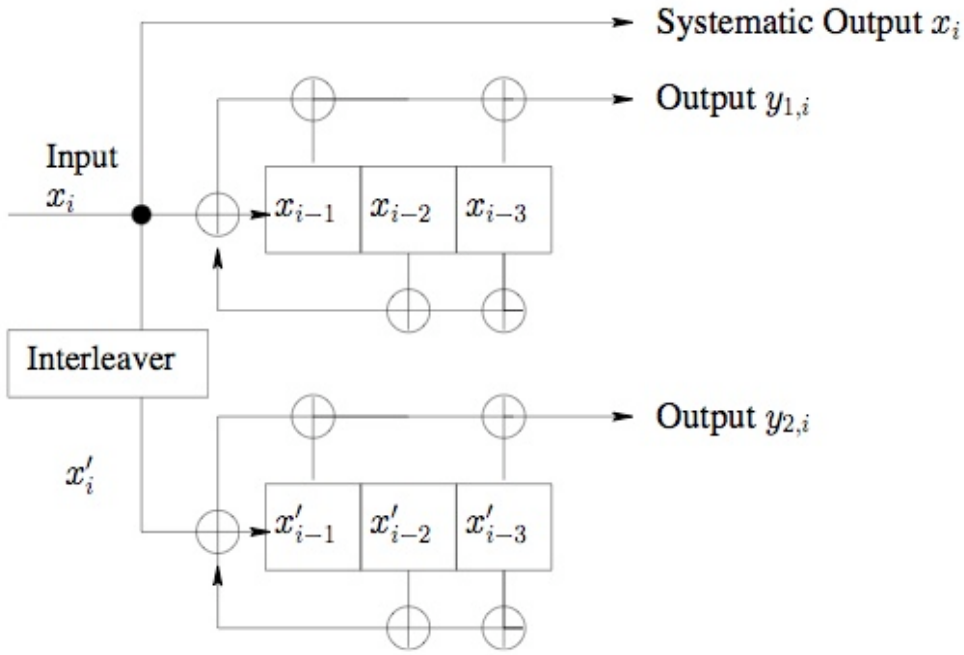
- Turbo-Codes sind wesentlich effizienter als Faltungs-Codes
  - bestehen aus zwei Faltungs-Codes welche abwechselnd mit der Eingabe versorgt werden.
  - Die Eingabe wird durch eine Permutation (Interleaver) im zweiten Faltungs-Code umsortiert



CoNe Freiburg

Turbo-Codes

- Beispiel:
  - UMTS Turbo-Kodierer
- Dekodierung von Turbo-Codes ist effizienter möglich als bei Faltungscodes
- Kompensation von Bursts



UNI FREIBURG

CoNe Freiburg

Interleavers

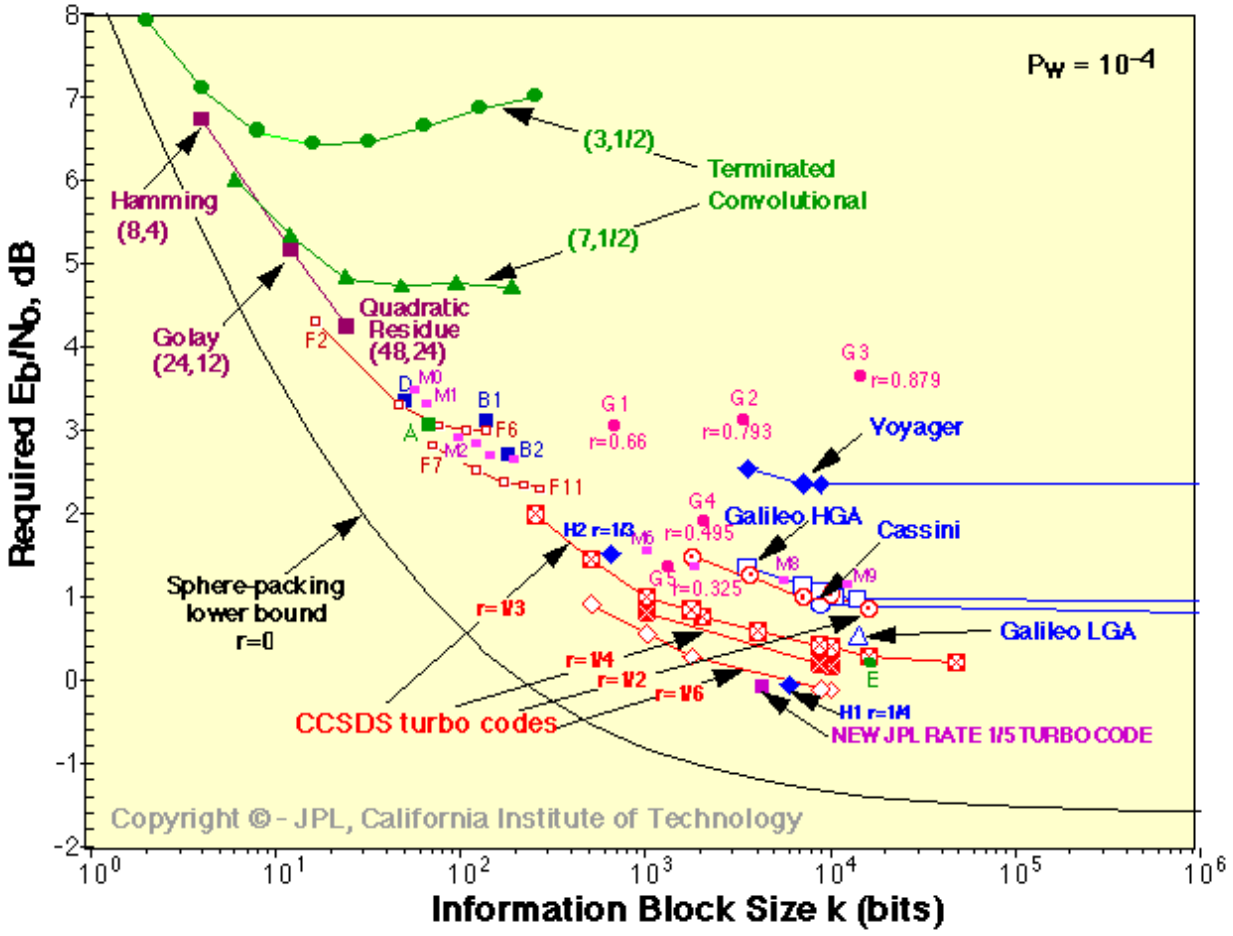
- Fehler treten oftmals gehäuft auf (Bursts)
  - z.B.: Daten: 0 1 2 3 4 5 6 7 8 9 A B C D E F
  - mit Fehler: 0 1 2 3 **?** **?** **?** **?** **?** 9 A B C D E F
- Dann scheitern klassische Kodierer ohne Interleavers
  - Nach Fehlerkorrektur (zwei Zeichen in Folge reparierbar):
    0 1 2 3 4 5 **?** 7 8 9 A B C D E F
- Interleaver:
  - Permutation der Eingabekodierung:
    0 1 2 3
    4 5 6 7
    8 9 A B
    C D E F
  - z.B. Row-column Interleaver:
    0 4 8 C 1 5 9 D 2 6 A E 3 7 B F
  - mit Fehler: 0 4 8 C **?** **?** **?** **?** **?** 6 A E 3 7 B F
  - Rückpermutiert: 0 **?** **?** 3 4 **?** 6 7 8 **?** A B C D **?** F
  - nach FEC: 0 1 2 3 4 5 6 7 8 9 A B C D E F

UNI FREIBURG

CoNe Freiburg

Codes im Vergleich

- Code-Rate versus Signal-Rausch-Verhältnis
  - Stand 1998: (www331.jpl.nasa.gov/public/AllCodesVsSize.GIF)



CoNe Freiburg

Fehlererkennung: CRC

- Effiziente Fehlererkennung: Cyclic Redundancy Check (CRC)
- Praktisch häufig verwendeter Code
  - Hoher Fehlererkennungsrate
  - Effizient in Hardware umsetzbar
- Beruht auf Polynomarithmetik im Restklassenring Z_2
  - Zeichenketten sind Polynome
  - Bits sind Koeffizienten des Polynoms

UNI FREIBURG

CoNe Freiburg

Rechnen in Z_2

- Rechnen modulo 2:
- Regeln:
  - Addition modulo 2 = Xor = Subtraktion modulo 2
  - Multiplikation modulo 2 = And

A	B	A + B
0	0	0
0	1	1
1	0	1
1	1	0

A	B	A - B
0	0	0
0	1	1
1	0	1
1	1	0

A	B	A · B
0	0	0
0	1	0
1	0	0
1	1	1

  - Beispiel: $0 + (1 \cdot 0) + 1 + (1 \cdot 1) =$

UNI FREIBURG

CoNe Freiburg

Polynomarithmetik modulo 2

- Betrachte Polynome über den Restklassenring Z_2
  - $p(x) = a_n x^n + \ldots + a_1 x^1 + a_0$
  - Koeffizienten a_i und Variable x sind aus $\in \{0,1\}$
  - Berechnung erfolgt modulo 2
- Addition, Subtraktion, Multiplikation, Division von Polynomen wie gehabt

UNI FREIBURG

CoNe Freiburg

Zeichenketten und Polynomarithmetik

- Idee:
  - Betrachte Bitstring der Länge n als Variablen eines Polynoms
- Bit string: $b_n b_{n-1} \ldots b_1 b_0$

  Polynom: $b_n x^n + \ldots + b_1 x^1 + b_0$
  - Bitstring mit (n+1) Bits entspricht Polynom des Grads n
- Beispiel
  - A xor B = A(x) + B(x)
  - Wenn man A um k Stellen nach links verschiebt, entspricht das
    - $B(x) = A(x)\, x^k$
- Mit diesem Isomorphismus kann man Bitstrings dividieren

UNI FREIBURG

CoNe Freiburg

Polynome zur Erzeugung von Redundanz: CRC

- Definiere ein Generatorpolynom *G(x)* von Grad *g*
  - Dem Empfänger und Sender bekannt
  - Wir erzeugen g redundante Bits
- Gegeben:
  - Frame (Nachricht) M, als Polynom M(x)
- Sender
  - Berechne den Rest der Division $r(x) = x^g M(x) \bmod G(x)$
  - Übertrage $T(x) = x^g M(x) + r(x)$
    - Beachte: $x^g M(x) + r(x)$ ist ein Vielfaches von G(x)
- Empfänger
  - Empfängt m(x)
  - Berechnet den Rest: m(x) mod G(x)

UNI FREIBURG

CoNe Freiburg

CRC Übertragung und Empfang

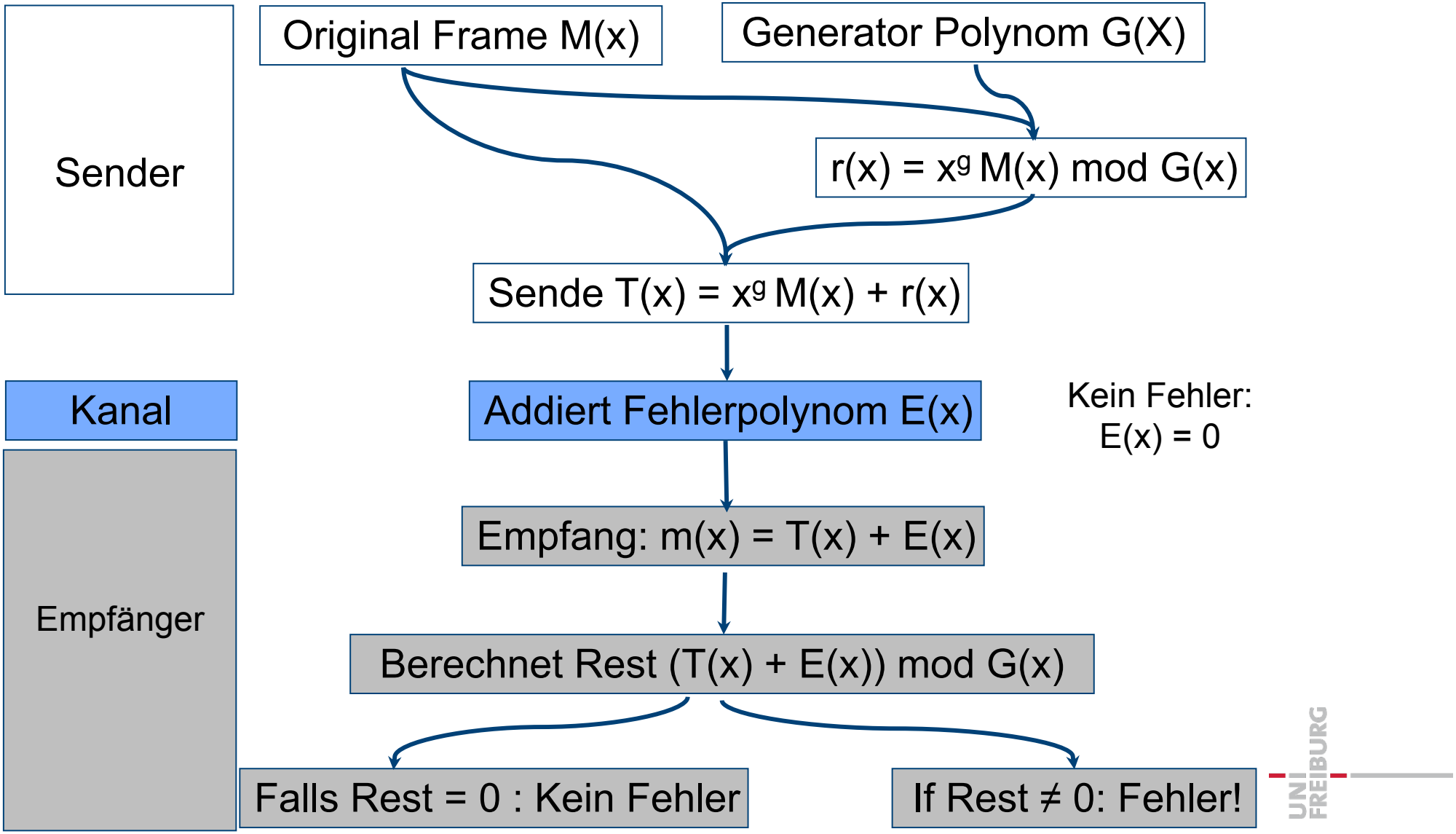
- Keine Fehler:
  - T(x) wird korrekt empfangen
- Bitfehler: T(x) hat veränderte Bits
  - Äquivalent zur Addition eines Fehlerpolynoms E(x)
  - Beim Empfänger kommt T(x) + E(x) an
- Empfänger
  - Empfangen: m(x)
  - Berechnet Rest m(x) mod G(x)
  - Kein Fehler: m(x) = T(x),
    - dann ist der Rest 0
  - Bit errors: m(x) mod G(x) = (T(x)+ E(x)) mod G(x)
    = T(x) mod G(x) + E(x) mod G(x)

0 Fehlerindikator

UNI FREIBURG

CoNe Freiburg

CRC – Überblick



CoNe Freiburg

Der Generator bestimmt die CRC-Eigenschaften

- Bit-Fehler werden nur übersehen, falls E(x) ein Vielfaches von G(x) ist
- Die Wahl von G(x) ist trickreich:
- Einzel-Bit-Fehler: $E(x) = x^i$ für Fehler an Position i
  - G(x) hat mindestens zwei Summenterme, dann ist E(x) kein Vielfaches
- Zwei-Bit-Fehler: $E(x) = x^i + x^j = x^j (x^{i-j} + 1)$ für i>j
  - G(x) darf nicht $(x^k + 1)$ teilen für alle k bis zur maximalen Frame-Länge
- Ungerade Anzahl von Fehlern:
  - E(x) hat nicht (x+1) als Faktor
  - Gute Idee: Wähle (x+1) als Faktor von G(x)
    - Dann ist E(x) kein Vielfaches von G(x)
- Bei guter Wahl von G(x):
  - kann jede Folge von r Fehlern erfolgreich erkannt werden

UNI FREIBURG

CoNe Freiburg

CRC in der Praxis

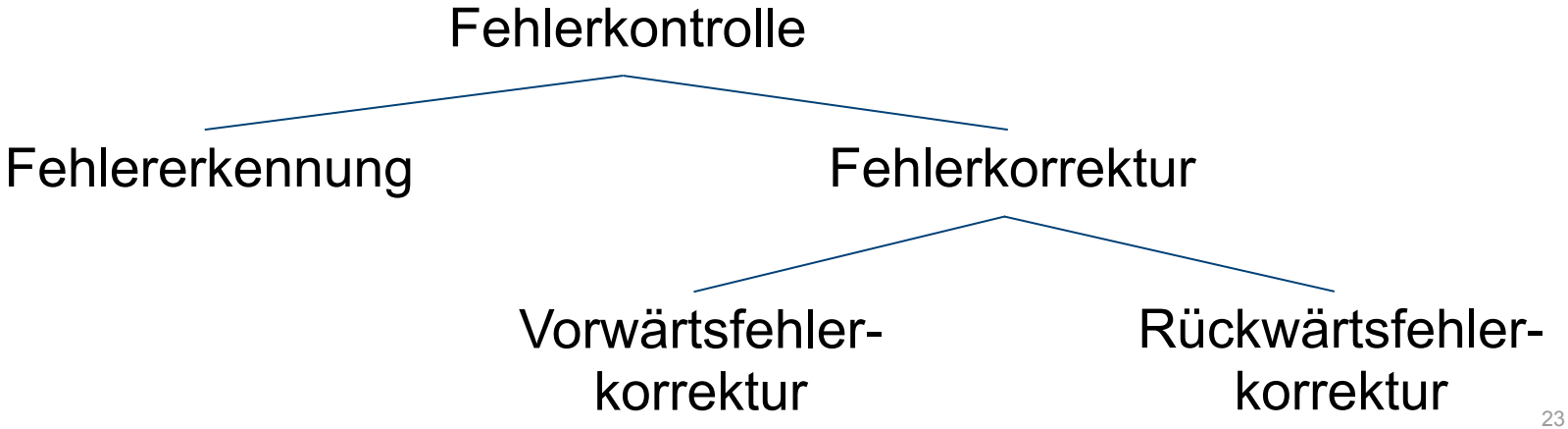
- Verwendetes irreduzibles Polynom gemäß IEEE 802:
  - $x^{32} + x^{23} + x^{16} + x^{12} + x^{11} + x^{10} + x^8 + x^7 + x^5 + x^4 + x^2 + x + 1$
- Achtung:
  - Fehler sind immer noch möglich
  - Insbesondere wenn der Bitfehler ein Vielfaches von G(x) ist.
- Implementation:
  - Für jedes Polynom x^i wird $r(x,i)= x^i \bmod G(x)$ berechnet
  - Ergebnis von B(x) mod G(x) ergibt sich aus
  - $b_0\, r(x,0) + b_1\, r(x,1) + b_2\, r(x,2) + \ldots + b_{k-1}\, r(x,k-1)$
  - Einfache Xor-Operation

UNI FREIBURG

Fehlerkontrolle

CoNe Freiburg

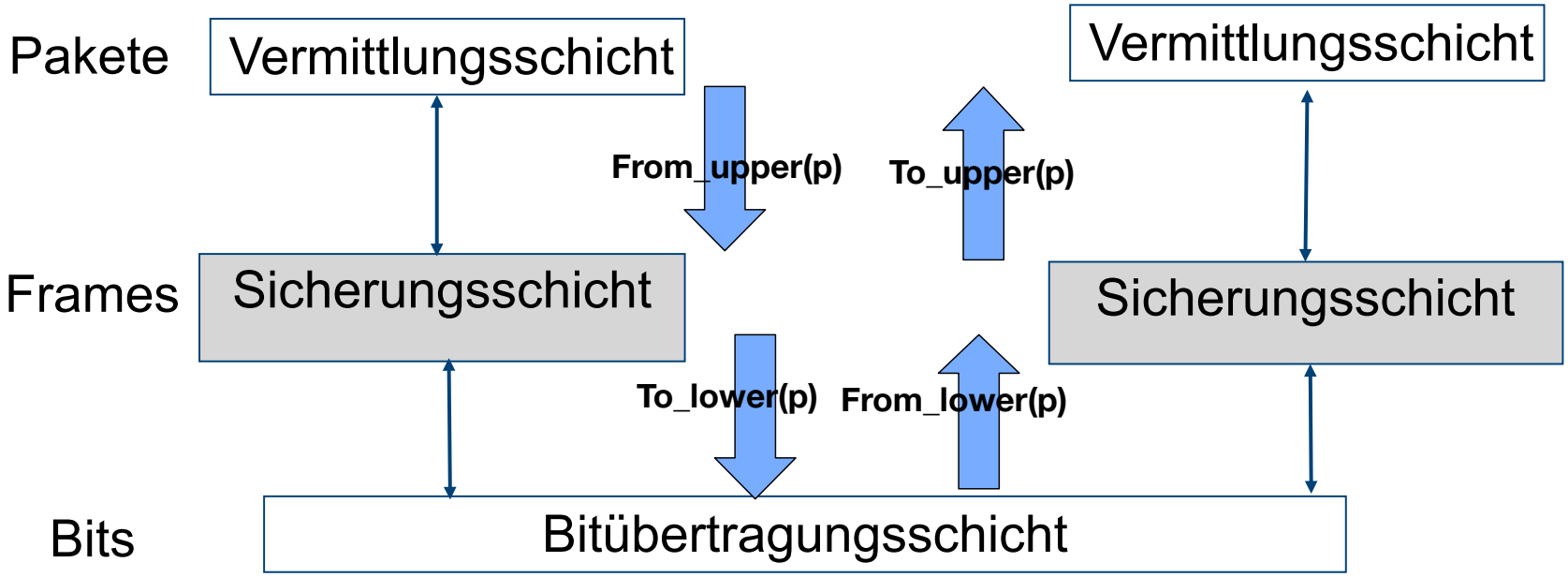
- Zumeist gefordert von der Vermittlungsschicht
  - Mit Hilfe der Frames
- Fehlererkennung
  - Gibt es fehlerhaft übertragene Bits?
- Fehlerkorrektur
  - Behebung von Bitfehlern
  - Vorwärtsfehlerkorrektur (Forward Error Correction)
    - Verwendung von redundanter Kodierung, die es ermöglicht Fehler ohne zusätzliche Übertragungen zu beheben
  - Rückwärtsfehlerkorretur (Backward Error Correction)
    - Nach Erkennen eines Fehlers, wird durch weitere Kommunikation der Fehler behoben



UNI FREIBURG

Rückwärtsfehlerkorrektur

- Bei Fehlererkennung muss der Frame nochmal geschickt werden
- Wie ist das Zusammenspiel zwischen Sender und Empfänger?

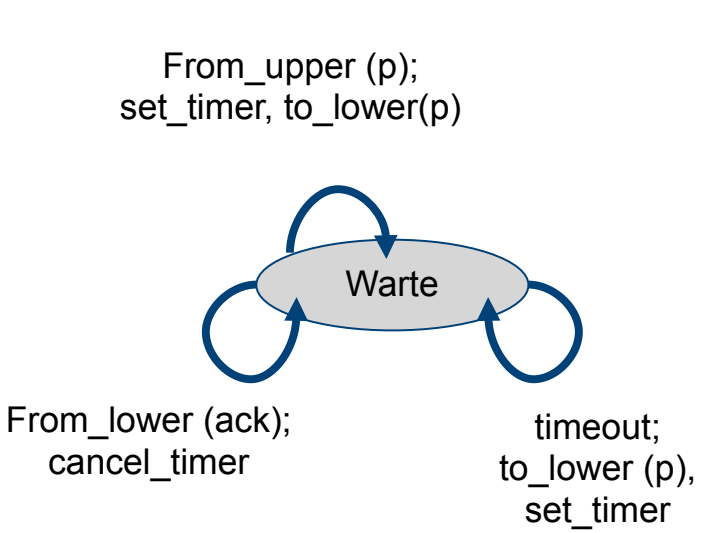


to_lower, from_lower beinhalten CRC
oder (bei Bedarf) Vorwärtsfehlerkorrektur

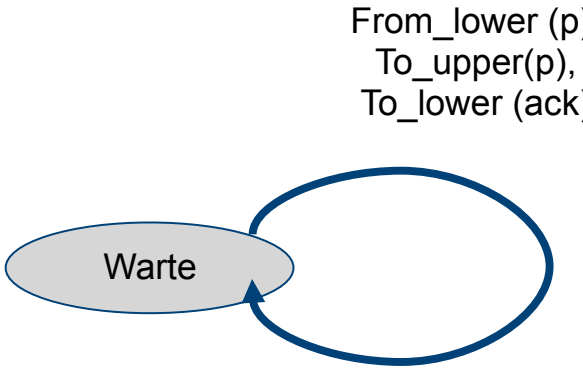
Einfaches Simplex-Protokoll mit Bestätigungen

- Empfänger bestätigt Pakete dem Sender
  - Der Sender wartet für eine bestimmte Zeit auf die Bestätigung (acknowledgment)
  - Falls die Zeit abgelaufen ist, wird das Paket wieder versendet
- Erster Lösungsansatz

–Sender



Empfänger



CoNe Freiburg

Diskussion

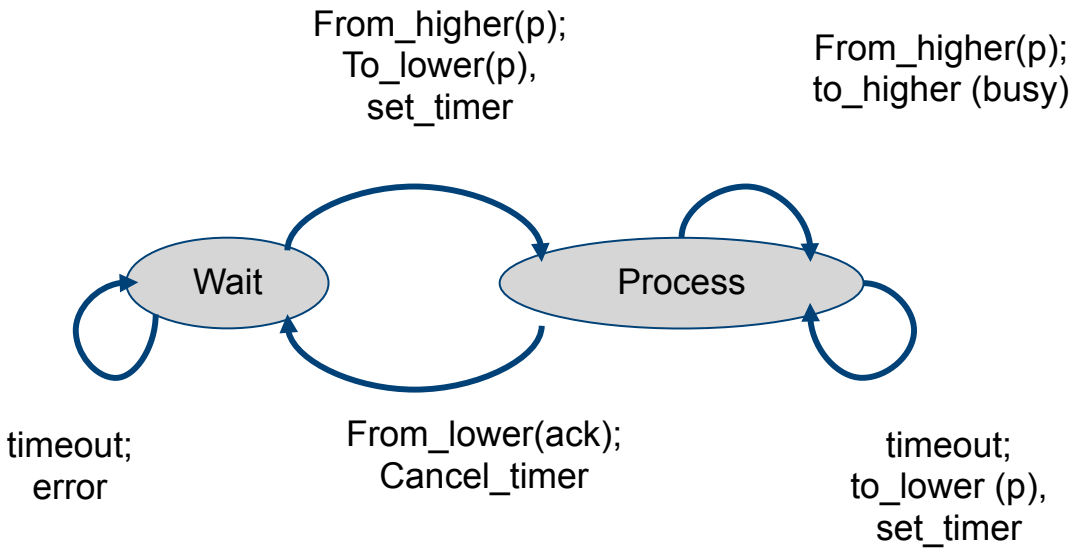
- Probleme
  - Sender ist schneller als Empfänger
  - Was passiert, wenn Bestätigungen verloren gehen?

UNI FREIBURG

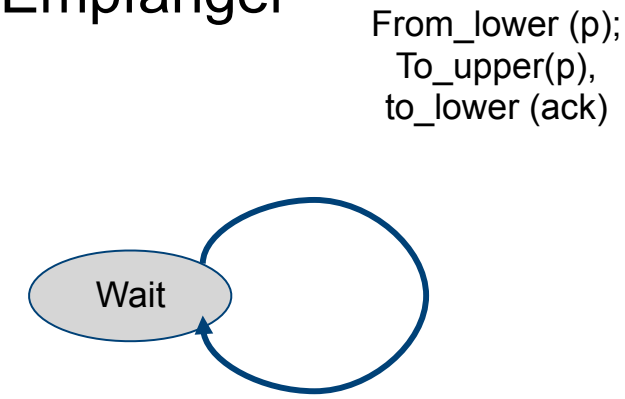
2. Versuch

- Lösung des ersten Problems
  - Ein Paket nach dem anderen

- Sender



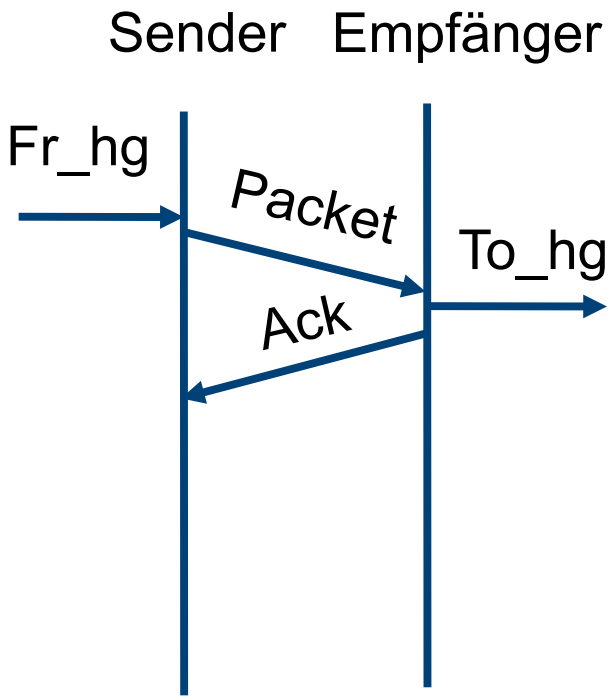
Empfänger

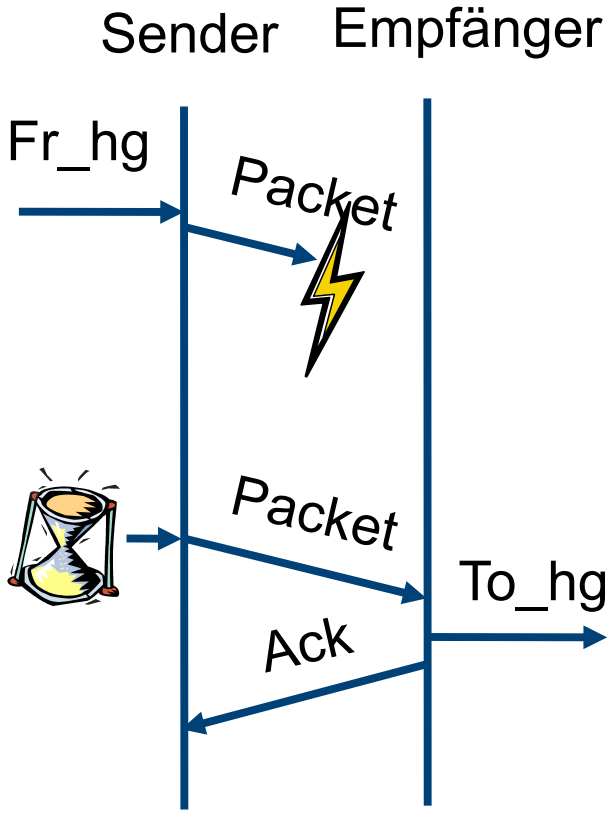


CoNe Freiburg

Diskussion

- Protokoll etabliert elementare Flusskontrolle

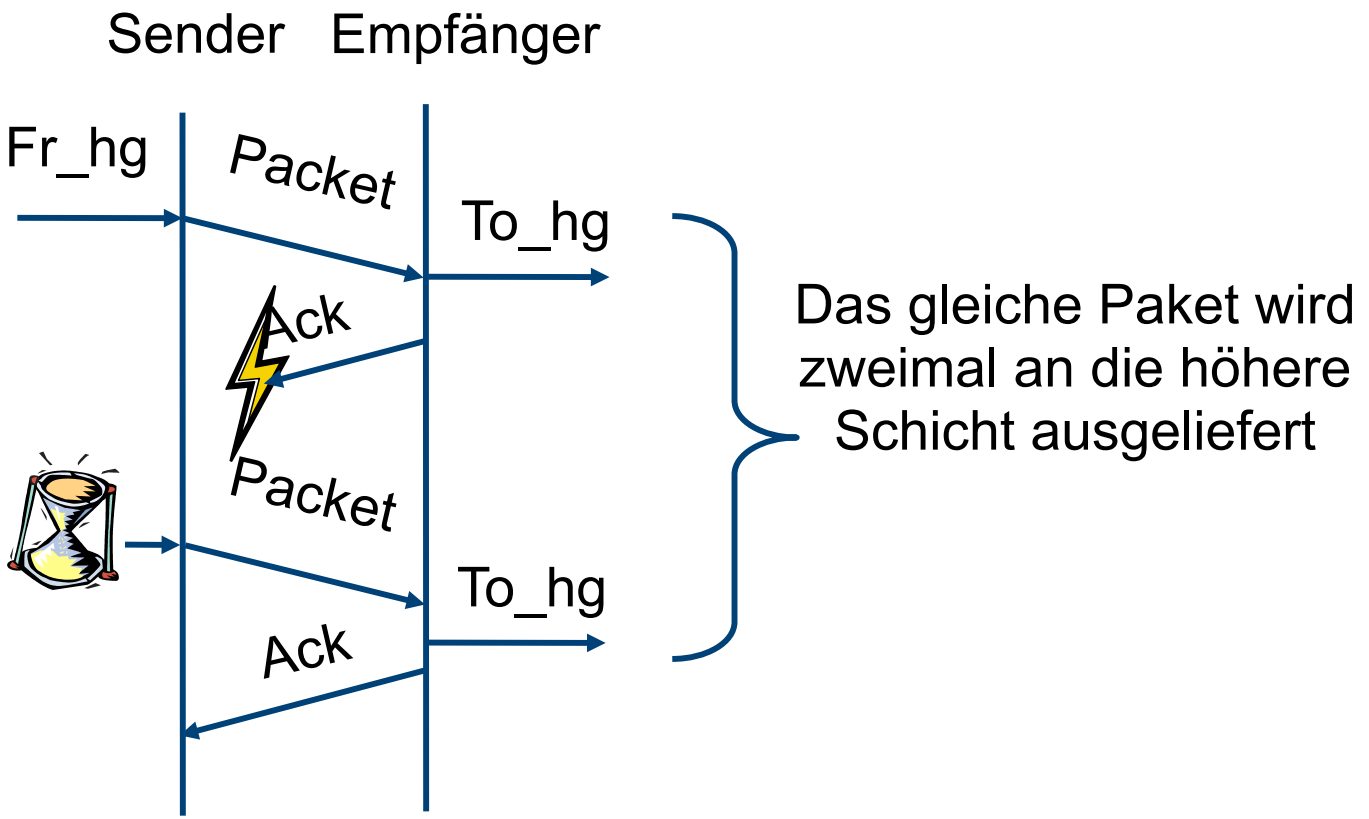




UNI FREIBURG

Diskussion

- 2. Fall: Verlust von Bestätigung



CoNe Freiburg

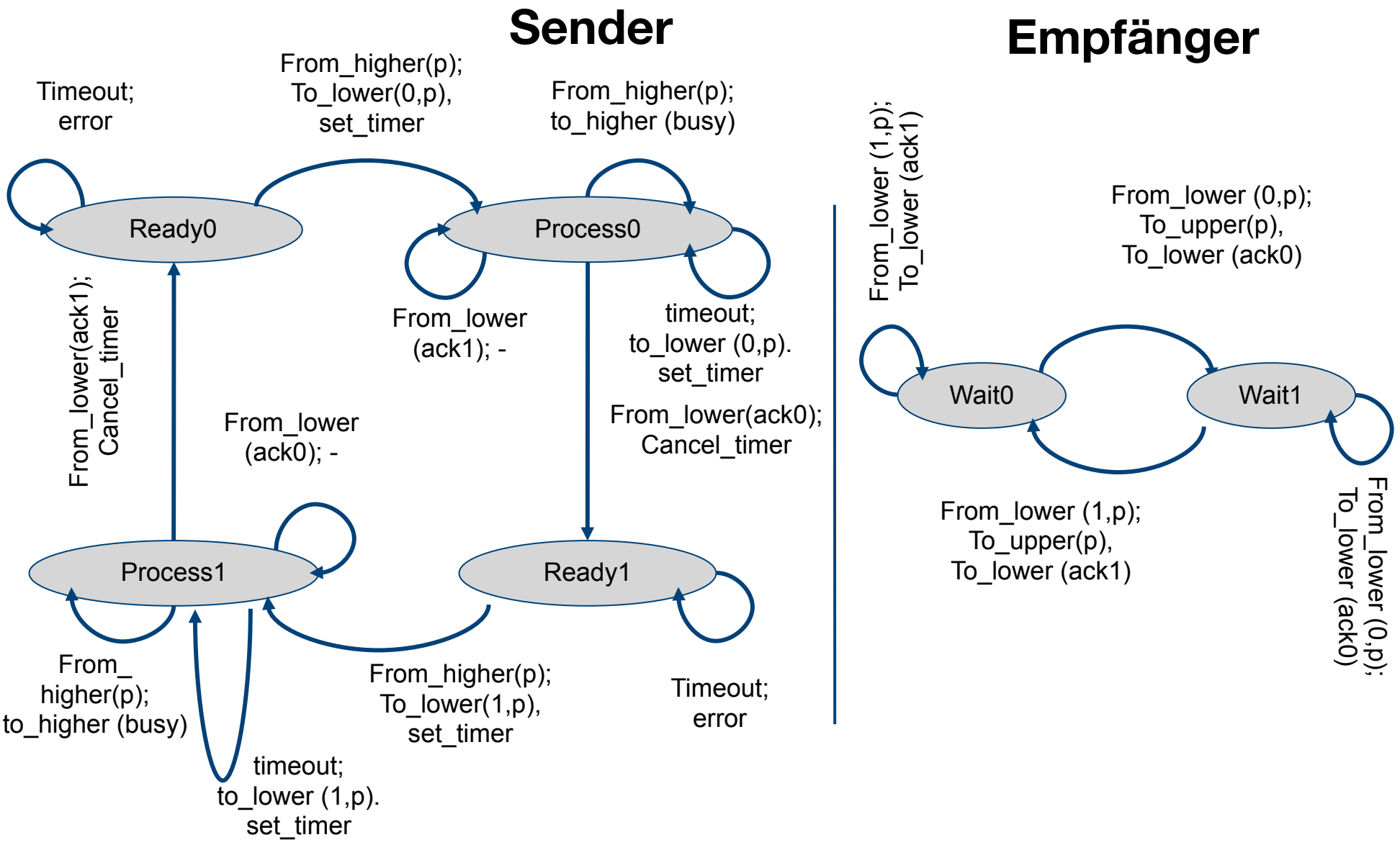
Probleme der 2. Version

- Sender kann nicht zwischen verlorenem Paket und verlorener Bestätigung unterscheiden
  - Paket muss neu versendet werden
- Empfänger kann nicht zwischen Paket und redundanter Kopie eines alten Pakets unterscheiden
  - Zusätzliche Information ist notwendig

- Idee:
  - Einführung einer Sequenznummer in jedes Paket, um den Empfänger Identifikation zu ermöglichen
  - Sequenznummer ist im Header jedes Pakets
  - Hier: nur 0 oder 1
- Notwendig in Paket und Bestätigung
  - In der Bestätigung wird die Sequenznummer des letzten korrekt empfangenen Pakets mitgeteilt
    - (reine Konvention)

UNI FREIBURG

3. Versuch: Bestätigung und Sequenznummern



CoNe Freiburg

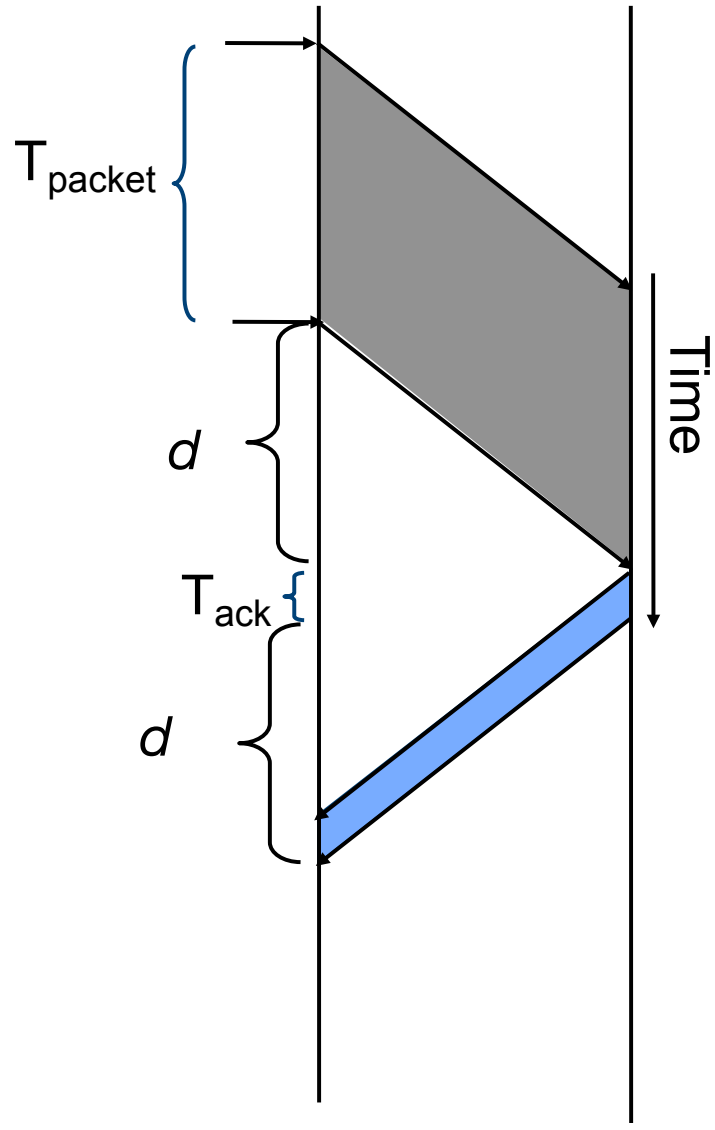
3. Version
Alternating Bit Protocol

- Die 3. Version ist eine korrekte Implementation eines verlässlichen Protokolls über einen gestörten Kanal
  - Alternating Bit Protokoll
  - aus der Klasse der Automatic Repeat reQuest (ARQ) Protokolle
  - beinhaltet auch eine einfache Form der Flusskontrolle
- Zwei Aufgaben einer Bestätigung
  - Bestätigung, dass Paket angekommen ist
  - Erlaubnis ein neues Paket zu schicken

UNI FREIBURG

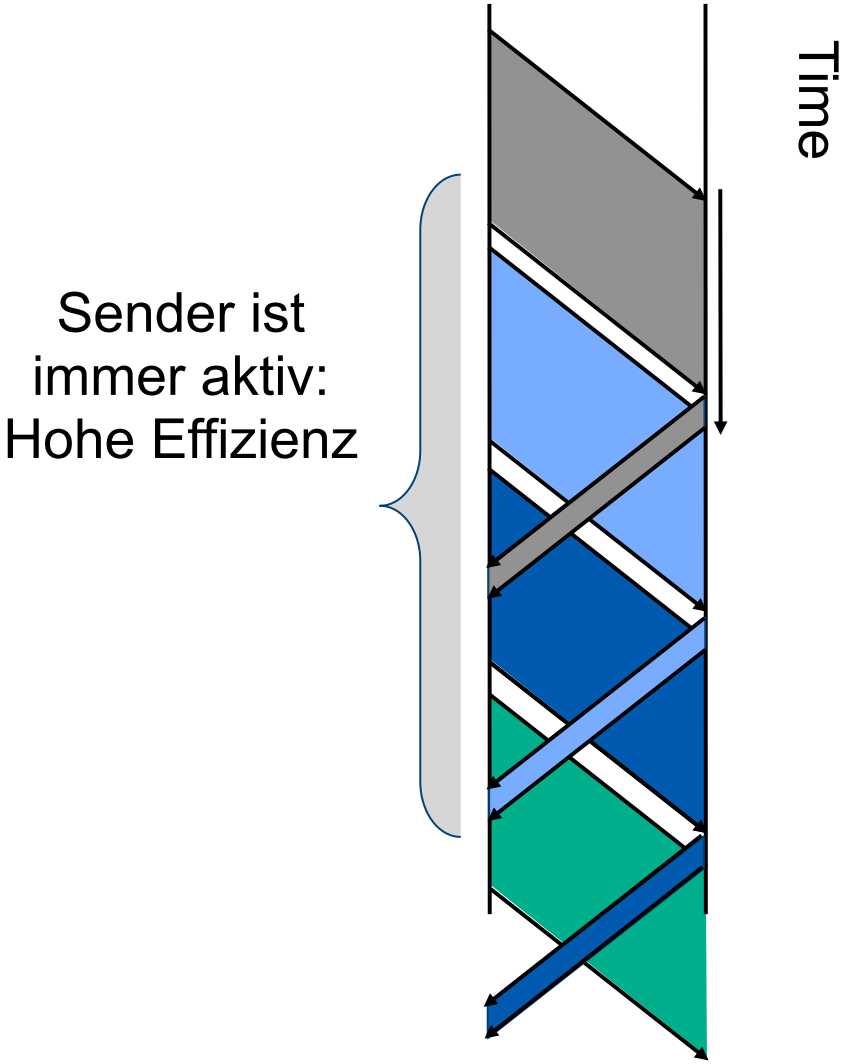
Alternating Bit Protocol – Effizienz

- Effizienz η
  - Definiert als das Verhältnis zwischen
    - der Zeit um zu senden
    - und der Zeit bis neue Information gesendet werden kann
    - (auf fehlerfreien Kanal)
  - $\eta = T_{packet} / (T_{packet} + d + T_{ack} + d)$
- Bei großen Delay ist das Alternating Bit Protocol nicht effizient



Verbesserung der Effizienz

- Durchgehendes Senden von Paketen erhöht Effizienz
  - Mehr “ausstehende” nicht bestätigte Pakete erhöhen die Effizienz
  - “Pipeline” von Paketen
- Nicht mit nur 1-Bit-Sequenznummer möglich



CoNe Freiburg

Gleitende Fenster

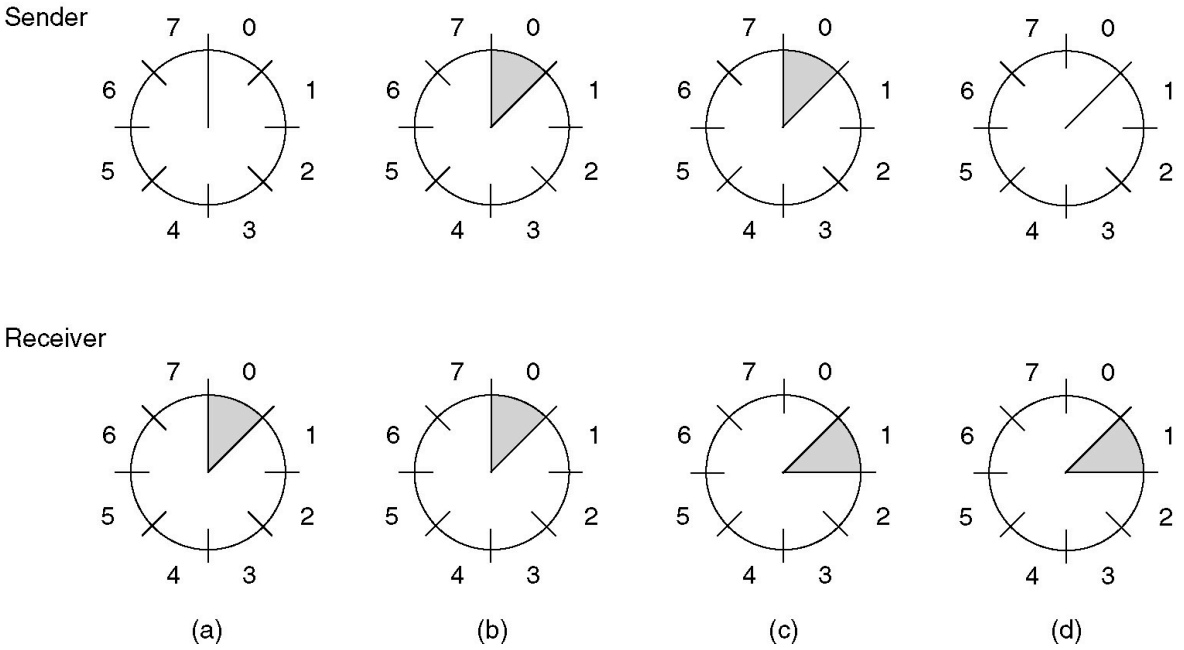
- Der Raum für Sequenznummern wird vergrößert
  - auf n Bits oder 2^n Sequenznummern
- Nicht alle davon können gleichzeitig verwendet werden
  - auch bei Alternating Bit Protocol nicht möglich

- "Gleitende Fenster" (sliding windows) bei Sender und Empfänger behandeln dieses Problem
  - Sender: Sende-Fenster
    - Folge von Sequenznummer, die zu einer bestimmten Zeit gesendet werden können
  - Empfänger: Empfangsfenster
    - Folge von Sequenznummer, die er zu einer bestimmten Zeit zu akzeptieren bereit ist
  - Größe der Fenster können fest sein oder mit der Zeit verändert werden
  - Fenstergröße entspricht Flusskontrolle

UNI FREIBURG

Beispiel

- “Sliding Window”-Beispiel für n=3 und fester Fenstergröße = 1
- Der Sender zeigt die momentan unbestätigten Sequenznummern an
  - Falls die maximale Anzahl nicht bestätigter Frames bekannt ist, dann ist das das Sende-Fenster



a. Initial: Nichts versendet

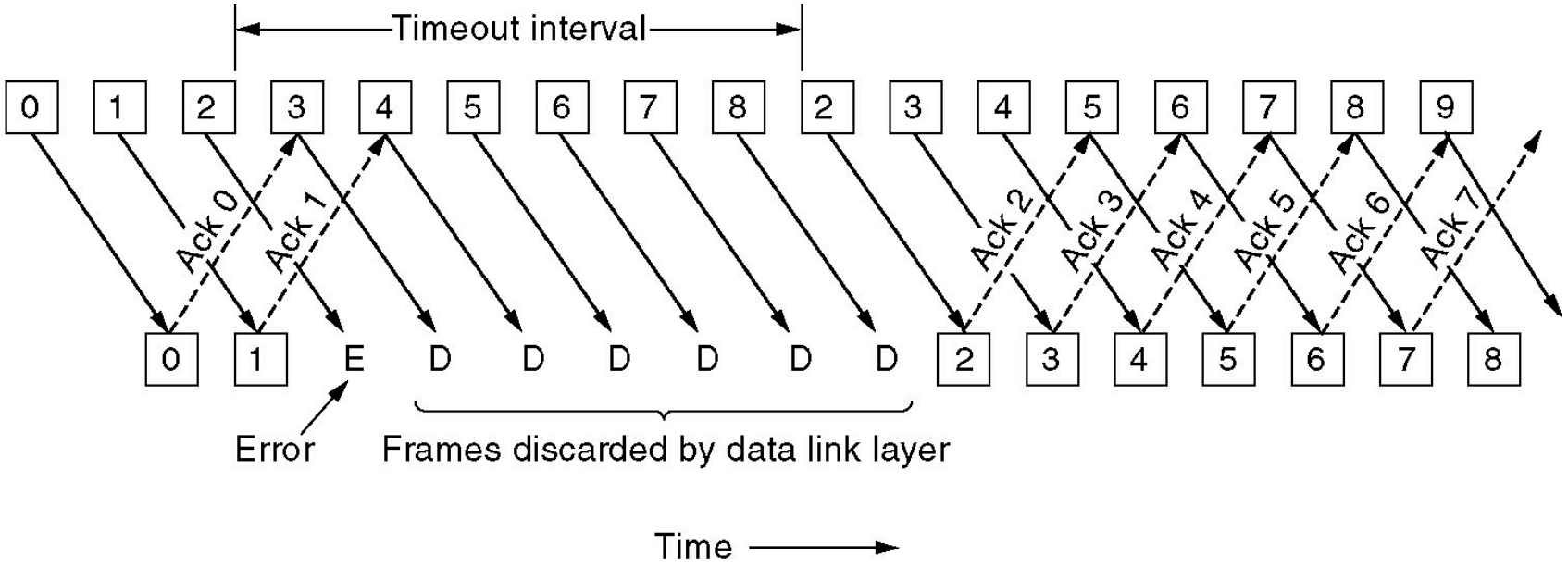
b. Nach Senden des 1. Frames mit Seq.Nr. 0

c. Nach dem Empfang des 1. Frame

d. Nach dem Empfang der Bestätigung

Übertragungsfehler und Empfangsfenster

- Annahme:
  - Sicherungsschicht muss alle Frames korrekt in der richtigen Reihenfolge verschicken
  - Sender “pipelined” Paket zur Erhöhung der Effizienz
- Bei Paketverlust:
  - werden alle folgenden Pakete ebenfalls fallen gelassen



CoNe Freiburg

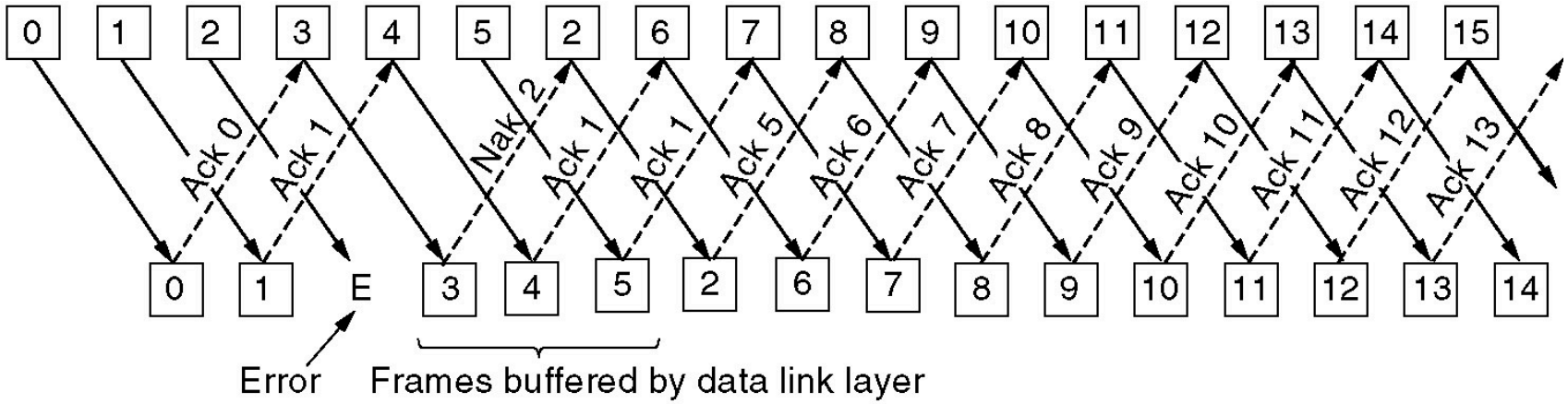
Go-back-N

- Mit Empfangsfenster der Größe 1 können die Frames, die einem verloren Frame folgen, nicht durch den Empfänger bearbeitet werden
  - Sie können einfach nicht bestätigt werden, da nur eine Bestätigung für des letzte korrekt empfangene Paket verschickt wird
- Der Sender wird einen “Time-Out” erhalten
  - Alle in der Zwischenzeit versandten Frames müssen wieder geschickt werden
  - “Go-back N” Frames!
- Kritik
  - Unnötige Verschwendung des Mediums
  - Spart aber Overhead beim Empfänger

UNI FREIBURG

Selektierte Wiederholung

- Angenommen
  - der Empfänger kann die Pakete puffern, welche in der Zwischenzeit angekommen sind
  - d.h. das Empfangsfenster ist größer als 1
- Beispiel



  - Der Empfänger informiert dem Sender fehlende Pakete mit negativer Bestätigung
  - Der Sender verschickt die fehlenden Frames selektiv
  - Sobald der fehlende Frame ankommt, werden alle (in der korrekten Reihenfolge) der Vermittlungsschicht übergeben

CoNe Freiburg

Duplex-Betrieb und Huckepack

- Simplex
  - Senden von Informationen in einer Richtung
- Duplex
  - Senden von Informationen in beide Richtungen
- Bis jetzt:
  - Simplex in der Vermittlungsschicht
  - Duplex in der Sicherungsschicht
- Duplex in den höheren Schichten
  - Nachrichten und Datenpakete separat in jeder Richtung
  - Oder Rucksack-Technik
    - Die Bestätigung wird im Header eines entgegen kommenden Frames gepackt

UNI FREIBURG

CoNe Freiburg

IIF INSTITUT FÜR INFORMATIK FREIBURG

Systeme II

4./5. Woche: Sicherungsschicht

Christian Schindelhauer
Technische Fakultät
Rechnernetze und Telematik
Albert-Ludwigs-Universität Freiburg

UNI FREIBURG