Lehrstuhl für Netzarchitekturen und Netzdienste
Fakultät für Informatik
Technische Universität München

Grundlagen Rechnernetze und Verteilte Systeme

IN0010, SoSe 2019

Übungsblatt 5

27. Mai – 31. Mai 2019

Hinweis: Mit * gekennzeichnete Teilaufgaben sind ohne Lösung vorhergehender Teilaufgaben lösbar.

Aufgabe 1 Medienzugriffsverfahren

a)* Erläutern Sie kurz das Prinzip von *ALOHA*.

b) Wie werden Kollisionen in *ALOHA* erkannt?

c) Erläutern Sie kurz das Prinzip von ***Slotted** ALOHA*.

d) Worin besteht der Vorteil von *Slotted ALOHA* gegenüber normalem *ALOHA*?

e)* Erläutern Sie kurz das Prinzip von *CSMA*.

f) Erläutern Sie kurz, welche Ergänzungen *CSMA/**CD*** gegenüber reinem *CSMA* hat.

g) Wie werden erfolgreiche Übertragungen bei *CSMA/CD* bei Ethernet erkannt?

h) Erläutern Sie kurz, welche Ergänzungen *CSMA/**CA*** gegenüber reinem *CSMA* hat.

i)* Was versteht man unter *Binary Exponential Backoff*?

Aufgabe 2 ALOHA und CSMA/CD

Gegeben sei ein Netzwerk (s. Abbildung 1) bestehend aus drei Computern, welche über ein Hub miteinander verbunden sind. Die Distanzen zwischen den Computern betragen näherungsweise $d_{12} = 1\,\text{km}$ bzw. $d_{23} = 500\,\text{m}$. Etwaige indirekte Kabelführung darf vernachlässigt werden. Die Übertragungsrate betrage $r = 100\,\text{Mbit/s}$. Die relative Ausbreitungsgeschwindigkeit betrage wie üblich $\nu = 2/3$. Die Lichtgeschwindigkeit sei mit $c_0 = 3 \cdot 10^8\,\text{m/s}$ gegeben.

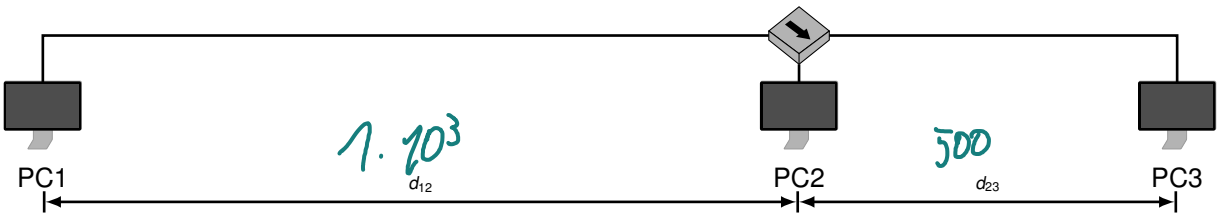


Abbildung 1

Zum Zeitpunkt

- $t_0 = 0\,\text{s}$ findet keine Übertragung statt und keiner der Rechner hat Daten zu versenden,
- $t_1 = 5\,\mu\text{s}$ beginnt PC1,
- $t_2 = 15\,\mu\text{s}$ beginnt PC2 und
- $t_3 = 10\,\mu\text{s}$ beginn PC3

jeweils einen Rahmen der Länge 94 B zu senden.

a)* Berechnen Sie die Serialisierungszeit t_s für eine Nachricht.

b)* Berechnen Sie die Ausbreitungsverzögerungen $t_p(1,2)$ und $t_p(2,3)$ auf den beiden Streckenabschnitten.

c) Zeichnen Sie für ALOHA und 1-persistentes CSMA/CD jeweils ein Weg-Zeit-Diagramm, das den Sendevorgang im Zeitintervall $t \in [t_0, t_0 + 30\,\mu\text{s})$ darstellt. Maßstab: $100\,\text{m} \triangleq 5\,\text{mm}$ bzw. $2{,}5\,\mu\text{s} \triangleq 5\,\text{mm}$, Slotzeit:

Prof. Dr.-Ing. Georg Carle carle@tum.de
Dr.-Ing. Stephan Günther, Henning Stubbe grnvs@net.in.tum.de

1a) Es gibt eine zentrale Station, an welche alle Sender. Korrekte Sendungen, werden auf einem separaten Kanal quittiert.

b) Wenn nur von dem Empfänger, der Basisstation, welche nicht quittiert.

c) Slotted bedeutet, dass nur zu bestimmten Zeiten man das Senden starten darf.

d) Weniger Kollisionen

e) Sender merken, wenn die Leitung schon belegt ist und senden dann nicht.

f) Man hört auf zu senden, wenn man feststellt, dass auch jemand anderes gesendet hat. Nachrichten müssen bestimmte Länge haben.

g) Wenn keine Kollision festgestellt wurde gilt sie als erfolgreich übertragen.

h) CA = Collision Avoidance ist p-Persistentes CSMA.

i)

Binary Exponential Backoff

Beim k-ten Sendeversuch einer Nachricht

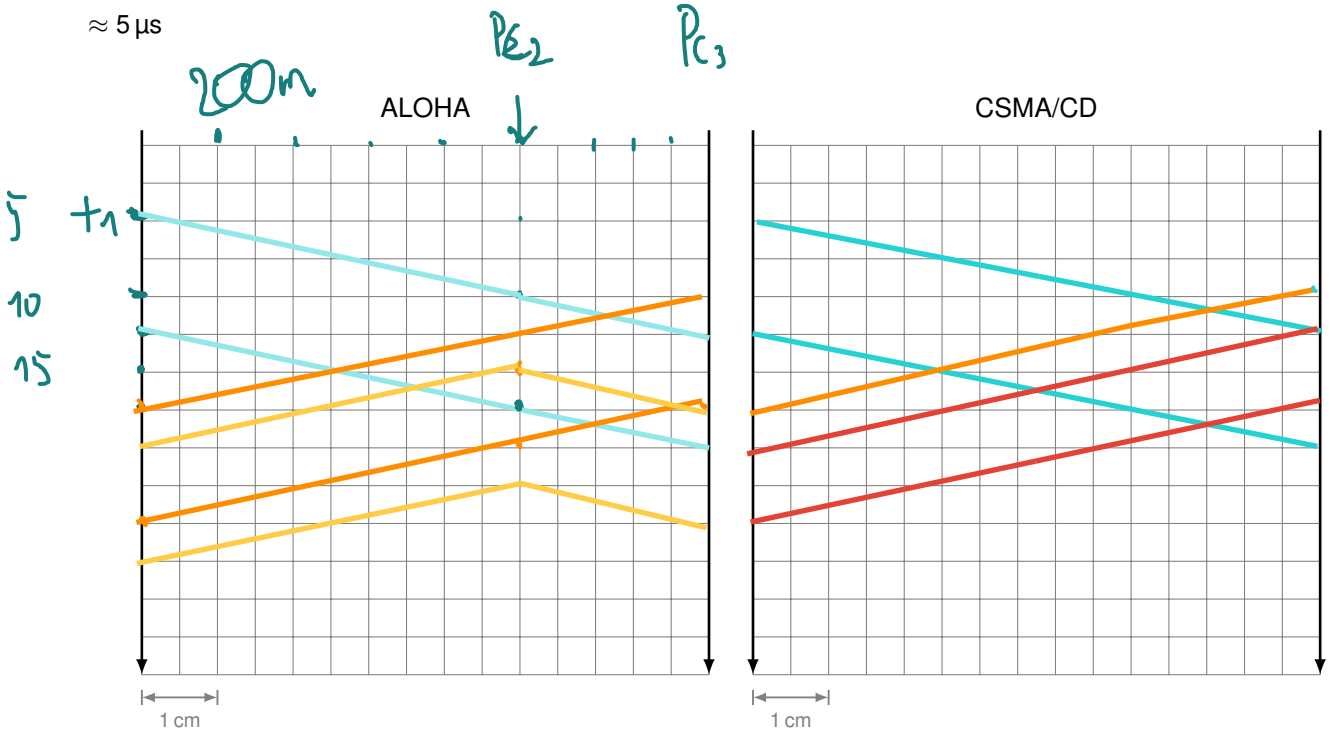
- wählt der Sender zufällig $n \in \{0, \dots, \min\{2^{k-1} - 1, 1023\}\}$ aus und
- wartet n Slotzeiten vor einem erneuten Sendeversuch.

Die maximale Wartezeit ergibt sich bei $k = 11$ (also bei 12 Sendeversuchen) und beträgt 1023 Slotzeiten.

Ähnlich zu CSMA/CA man vergrößert nur das Zeitintervall um das Doppelte. Sowohl bei CD als auch CA.

2b) $t_p(1,2) = \frac{1000}{3 \cdot 10^8 \cdot 2/3} = 5\,\mu s$

$t_p(2,3) = \quad = 2{,}5\,\mu s$



d) Aus der vorhergehenden Teilaufgabe ist zu erkennen, dass bei beiden Verfahren Kollisionen auftreten. Im Gegensatz zu ALOHA funktioniert CSMA/CD aber unter den gegebenen Umständen nicht. Warum?

e) Wie lautet für CSMA/CD die Bedingung, dass ein Knoten eine Kollision rechtzeitig erkennen kann?

f) Berechnen Sie für CSMA/CD die maximale Entfernung zweier Rechner innerhalb einer Kollisionsdomäne in Abhängigkeit der minimalen Rahmenlänge. Setzen Sie die Werte für FastEthernet ein ($r = 100\,\text{Mbit/s}$, $l_{\text{min}} = 64\,\text{B}$).

g)* Welchen Einfluss haben Hubs, Brücken und Switches auf die Kollisionsdomäne?

Aufgabe 3 Cyclic Redundancy Check (CRC)

Die Nachricht `10101100` werde mittels CRC, wie in der Vorlesung eingeführt, gesichert. Als Reduktionspolynom sei $r(x) = x^3 + 1$ gegeben.

a)* Wie lang ist die Checksumme?

b) Bestimmen Sie die Checksumme für die gegebene Nachricht.

c) Geben Sie die übertragene Bitfolge an.

Bei der Übertragung trete nun das Fehlermuster `00100000000` auf.

d)* Wie lautet die empfangene Bitfolge?

e) Zeigen Sie, dass der Übertragungsfehler erkannt wird.

f)* Geben Sie ein Fehlermuster an, welches nicht erkannt werden kann.

g) CRC wurde in der Vorlesung ausdrücklich als fehlererkennender, nicht aber als fehlerkorrigierender Code eingeführt. Zeigen Sie, dass mittels CRC selbst 1 bit-Fehler im konkreten Beispiel dieser Aufgabe nicht korrigierbar sind.

Prof. Dr.-Ing. Georg Carle
carle@tum.de
Dr.-Ing. Stephan Günther, Henning Stubbe
grnvs@net.in.tum.de

e) die Serialisierungszeit muss länger sein, als 2x die Ausbreitungsdauer.

d) Bei ALOHA wird der Verlust eines Rahmens dadurch erkannt, dass der Absender keine Bestätigung erhält. Ein derartiges Quittungsverfahren gibt es bei CSMA/CD nicht. Stattdessen nimmt ein Sender bei CSMA/CD an, dass ein Rahmen erfolgreich übertragen wurde, falls während der Übertragung keine Kollision aufgetreten ist. Im konkreten Fall hat PC1 allerdings die Übertragung abgeschlossen, bevor ihn die Übertragung bzw. das JAM-Signal von PC3 erreicht. PC1 erkennt daher die Kollision nicht und geht fälschlicherweise von einer erfolgreichen Übertragung aus.

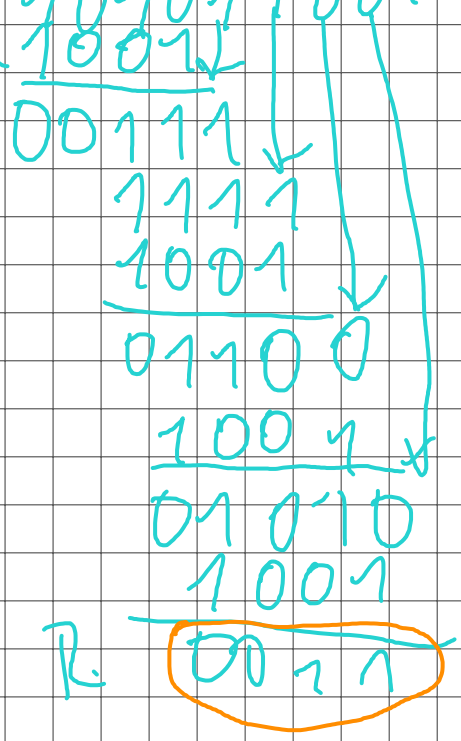
f) $\frac{I_{min}}{r} = 2 \cdot \frac{d_{max}}{v_{CO}}$

$\frac{I_{min} \cdot v_{CO}}{2 \cdot r} = d_{max} = 512m$

g) Hubs vergrößern die Kollisionsdomäne. Switches und Bridges unterbrechen sie.

3a) grad(n(x)) = 3 Bit

b) 10101100 : 1001 = 1011 1011



d) Die Empfangene Bitfolge ist das XOR aus der Übertragenen Bitfolge und dem Fehlermuster

XOR
$$\begin{array}{r} 10101110001 \\ 00100000000 \\ \hline 10001110001 \end{array}$$

e) Bei der Division bleibt ein Rest übrig, welcher klar macht, dass ein Fehler aufgetreten ist

Aufgabe 4 Bit-Stuffing (Hausaufgabe)

In der Vorlesung wurden Verfahren zum Erkennen von Rahmengrenzen vorgestellt. Bei FastEthernet beispielsweise werden Rahmengrenzen mit Hilfe von Steuerzeichen erkannt, was durch die Verwendung der 4B5B-Kodierung möglich wird. Dabei stellt der 4B5B-Code bereits sicher, dass diese Steuerzeichen nie zufällig durch die Konkatenation von Codewörtern auftreten können.

Im Folgenden wollen wir untersuchen, wie das *High Level Data Link Control (HDLC) Protocol* diesem Problem begegnet. Bei HDLC handelt es sich um ein alternatives Schicht-2 Protokoll, welches beispielsweise auf seriellen Schnittstellen zwischen Cisco-Routern eingesetzt wird. Ein Beispiel für einen HDLC-Rahmen ist in Abbildung 2 dargestellt (vgl. Ethernet-Rahmen in den Vorlesungsfolien).

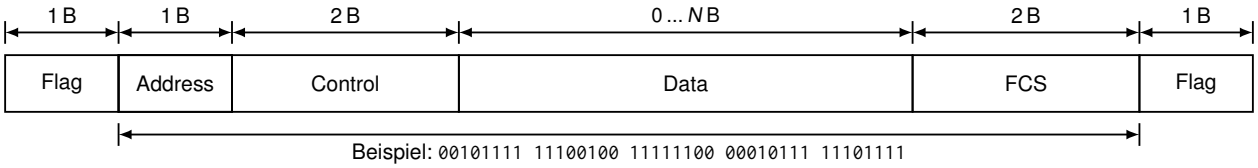


Abbildung 2: Aufbau eines HDLC-Rahmens

HDLC sieht am Anfang und Ende eines jeden Rahmens ein spezielles Begrenzungsfeld (Flag) mit dem Wert `0x7E` vor. Es ist also nicht auf die Verwendung zusätzlicher Codes zur Erkennung von Rahmengrenzen angewiesen. Allerdings muss nun sichergestellt werden, dass das Begrenzungsfeld nicht zufällig in einem Rahmen auftaucht. Alle Headerfelder seien von konstanter[1] Länge. Die Länge der Nutzdaten sei stets ein ganzzahliges[2] Vielfaches von 8 bit.

a)* In Abbildung 2 ist eine beispielhafte Bitsequenz für den Rahmen ohne Begrenzungsfelder abgebildet. Diese soll nun mittels HDLC übertragen werden. Um zu verhindern, dass das Begrenzungsfeld `0x7E` in den Daten auftaucht, soll Bit-Stuffing verwendet werden. Überlegen Sie sich ein möglichst effizientes Verfahren. Beschreiben Sie dieses in Worten.

b) Geben Sie die gesamte zu übertragende Bitsequenz gemäß dem in Teilaufgabe a) entwickelten Verfahren an. Markieren Sie die eingefügten Bitstellen.

c)* Modifizieren Sie den in Abbildung 2 dargestellten HDLC-Header, so dass anstelle von Bit-Stuffing Längenangaben verwendet werden können. Nehmen Sie dabei an, dass maximal $N = 255\,\mathrm{B}$ Daten pro Frame übertragen werden können.

[1] Das ist eine Vereinfachung. Bei HDLC können einige Header-Felder unterschiedliche Längen haben.
[2] Obwohl dies eine übliche Konvention ist, handelt es sich auch hier um eine vereinfachende Annahme.

a)* In Abbildung 2 ist eine beispielhafte Bitsequenz für den Rahmen ohne Begrenzungsfelder abgebildet. Diese soll nun mittels HDLC übertragen werden. Um zu verhindern, dass das Begrenzungsfeld `0x7E` in den Daten auftaucht, soll Bit-Stuffing verwendet werden. Überlegen Sie sich ein möglichst effizientes Verfahren. Beschreiben Sie dieses in Worten.

Nach jeweils fünf konsekutiven Einsen wird eine Null eingefügt. Dies kann vom Empfänger leicht rückgängig gemacht werden, indem er nach jeweils fünf konsekutiven Einsen die darauf folgende Null wieder entfernt.

Würde erst nach sechs Einsen eine Null eingefügt, so würde die Bitfolge `0111111` nach dem Stuffing genau das Begrenzungsfeld ergeben. Bereits nach vier oder weniger Einsen zu stopfen erhält zwar die Codetransparenz, ist jedoch weniger effizient.

Anmerkung: Warum nicht nach jeder sechsten Eins eine Siebte stopfen? Aus Sicht der Codetransparenz spricht nichts dagegen. Für HDLC wurde aber die oben beschriebene Variante gewählt. Der einzige Vorteil der HDLC-Version besteht in meinen Augen darin, dass lange Eins-Folgen verhindert werden. Je nach Leitungscode kann dies die Taktrückgewinnung ermöglichen, z. B. bei Einsatz einer „inversen“ MLT-3 Kodierung, bei der eine logische Null durch einen Pegelwechsel und eine logische Eins durch einen ausbleibenden Pegelwechsel dargestellt wird. Bei anderen Leitungscodes, wie z. B. NRZ, wird die Taktrückgewinnung nicht ermöglicht, da nach wie vor beliebig lange Nullfolgen auftreten können.

b) Geben Sie die gesamte zu übertragende Bitsequenz gemäß dem in Teilaufgabe a) entwickelten Verfahren an. Markieren Sie die eingefügten Bitstellen.

`01111110 00101111101100100111111010000010111111010111 01111110`

c)* Modifizieren Sie den in Abbildung 2 dargestellten HDLC-Header, so dass anstelle von Bit-Stuffing Längenangaben verwendet werden können. Nehmen Sie dabei an, dass maximal $N = 255\,\mathrm{B}$ Daten pro Frame übertragen werden können.

Es reicht, zwischen Flag- und Address-Feld ein neues Feld „Length“ einzufügen. Dieses habe eine Länge von 1 B und ermöglicht damit die Angabe der Werte $0, \ldots, 255$. Das Datenfeld trägt stets ein ganzzahliges Vielfaches von 8 bit. Damit werden Daten bis zu 255 B möglich. Die übrigen Headerfelder haben (da nicht anders angegeben) feste Längen.