



2. Übung zur Vorlesung Rechnernetze (Musterlösung)

Abgabe: 16.11.2004

Aufgabe 1: Leitungscodierung (2 Punkte)

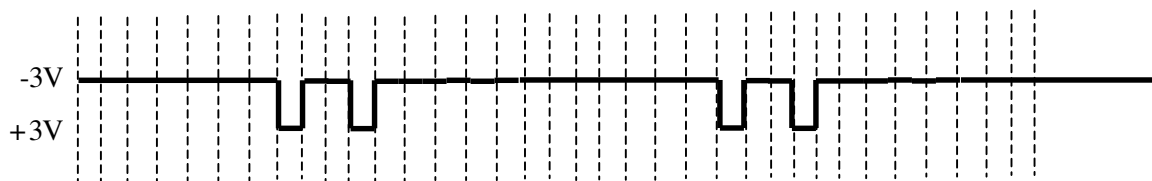
Erklären Sie, was man im Zusammenhang mit Leitungscodierung unter einer Codeverletzung versteht und welchem nützlichen Zweck eine solche Codeverletzung dienen kann! Nennen Sie ein Beispiel, wo und warum Codeverletzungen absichtlich erzeugt werden!

Lösungsvorschlag:

Die Leitungscodierung definiert, welche Symbole über eine Leitung übertragen werden, um die Werte von Bits oder Bitgruppen darzustellen. In der digitalen Datenübertragung wird ein Symbol z.B. durch eine bestimmte Spannung oder eine Spannungsveränderung erzeugt. Die Leitungscodierung legt außerdem fest, welche Sequenzen von Symbolen zur Darstellung von Datenbits zulässig sind, und welche Sequenzen unzulässig sind (z.B. sind aufeinanderfolgende positive Impulse bei AMI nicht zulässig). Eine unzulässige Sequenz von Symbolen wird auch als Codeverletzung bezeichnet. Diese kann absichtlich erzeugt werden, um dem Empfänger zusätzliche Informationen zu übermitteln – wie z.B. bei HDB3. Während bei der reinen AMI-Codierung keine sichere Takterkennung bei langen Null-Sequenzen möglich ist, werden bei HDB3 vier aufeinander folgende Nullen durch vier Symbole ersetzt, die eine Codeverletzung beinhalten. Die Ersetzung an sich dient dazu, sicherzustellen, dass in bestimmten Zeitabständen Signalübergänge vorhanden sind, um so eine sichere Takterkennung zu ermöglichen. Dadurch, dass diese Ersetzung auch eine Codeverletzung beinhaltet, kann der Empfänger erkennen, dass es sich um eine Ersetzung gehandelt hat, und somit kann er aus den übertragenen Symbolen die Originalnachricht wieder rekonstruieren.

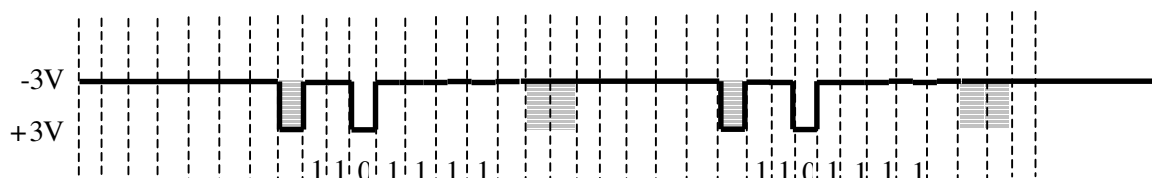
Aufgabe 2: Asynchrone Datenübertragung über die V.24 Schnittstelle (3 Punkte, 1+1+1)

- In der Zeichnung sehen Sie Signale, die über die V.24 Schnittstelle übertragen werden. Identifizieren Sie die übertragenen ASCII-Zeichen! Gehen Sie davon aus, dass 1 Startbit, 7 Datenbits, 1 Parity-Bit (odd parity) und 2 Stoppbits verwendet wurden. Beachten Sie, dass das niederwertigste Datenbit zuerst auf der Leitung erscheint!
- Was würde passieren, wenn durch ein kurzzeitige Störung auf der Leitung das Startbit des ersten Zeichens nicht erkannt wurde (d.h. zu dem Zeitpunkt, an dem das Startbit erscheinen sollte, wird fälschlicherweise eine Spannung von $-3V$ gemessen)?
- Nehmen Sie nun an, dass zwischen der Übertragung der beiden ASCII-Zeichen keine Pause stattgefunden hätte (d.h. das Startbit des zweiten Zeichens folgt unmittelbar auf das letzte Stoppbit des ersten Zeichens). Erläutern Sie, ob eine wie in Teilaufgabe b) beschriebene Störung auf der Leitung dann immer noch dieselben Auswirkungen hätte!

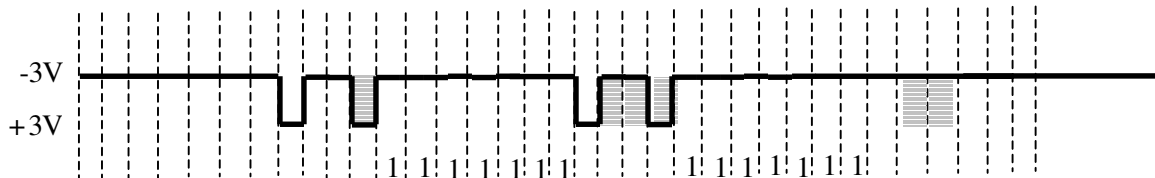


Lösungsvorschlag:

- Die gesendeten Bitsequenzen (ohne Parity) sind: 1101111 und noch mal 1101111. Da das niederwertigste Bit zuerst gesendet wurde, handelt es sich also um das Zeichen: 1111011 (binär) = $0x7B$ (hexadezimal). Das entspricht dem ASCII-Zeichen „{“.



- b) In diesem Fall würde das dritte Datenbit des ersten Zeichens fälschlicherweise für ein Startbit gehalten werden, und es würde daher angenommen werden, dass die gesendete Bitsequenz 1111111 war. Dann müsste das Parity-Bit allerdings eine 0 sein. Da dies aber nicht der Fall ist, würde erkannt werden, dass ein Fehler vorlag.
- c) In diesem Fall sieht das Signal wie unten angegeben aus. Würde nun das Startbit des ersten Zeichens nicht erkannt, dann würde wie in Teilaufgabe b) das dritte Datenbit des ersten Zeichens fälschlicherweise als Startbit interpretiert werden, und es würde angenommen werden, dass die übertragene Bitsequenz 1111111 war. Darauf müsste nun eine 0 im Parity-Bit folgen und dann zwei Stopbits. Unglücklicherweise folgen nun aber tatsächlich eine 0 und zweimal die 1, sodass die Störung nicht bemerkt wird. Die nachfolgende 0 wird nun erneut fälschlicherweise als Startbit des nächsten Zeichens interpretiert. Es wird erneut angenommen, dass die Bitsequenz 1111111 übertragen wurde. Dieses Mal wird nun aber erkannt, dass das Parity-Bit nicht korrekt ist. Das zweite Zeichen wird also als fehlerhaft erkannt.



Aufgabe 3: Dateitransfer (5 Punkte, 1+2+2)

- a) Wie lange dauert die asynchrone Übertragung (Even Parity, zwei Stopbits, acht Datenbits) des Inhalts einer 500Kbyte großen ASCII-Datei über ein serielles Nullmodemkabel bei einer Übertragungsrate von 19.200 Baud?
- b) In Teilaufgabe a) sind wir davon ausgegangen, dass nur Nutzdatenbits übertragen werden. In der ersten Übung haben Sie nun aber das Kermit-Protokoll zur Übertragung von Dateien kennen gelernt, und wie sie nun wissen, werden bei einem solchen Dateitransfer nicht nur die reinen Nutzdaten übertragen. In dieser Aufgabe soll nun eine 500KByte Datei über das oben beschriebene Nullmodemkabel mittels des Kermit-Protokolls übertragen werden. Berechnen Sie, wie lange die Übertragung der Datei dauert! (Berücksichtigen Sie bei Ihrer Berechnung nur Datenpakete und Ack-Pakete! Gehen Sie davon aus, dass keine Pakete verloren gehen!)
- c) Sie wollen das Kermit-Protokoll nun verändern, um den durch das Protokoll verursachten Overhead zu verringern. Überlegen Sie sich, wie sie dies erreichen können (es sind viele Möglichkeiten denkbar), und erläutern sie auch, ob derartige Veränderungen auch negative Folgen hätten!

Lösungsvorschlag:

- a) Ein Start-Bit gibt es immer, also übertragen wir insgesamt 12 Bit pro Nutzdatenbyte (1 Startbit, 2 Stopbits, 1 Paritätsbit, 8 Datenbits).
 19.200 Baud sind in diesem Fall 19.200 bit/s.
 $19.200 \text{ bit/s}, 12 \text{ bit/Nutzdatenbyte} \rightarrow 1.600 \text{ Byte/s (Nutzdaten)}$
 $500 \text{ Kbyte} = 1024 * 500 \text{ Byte} = 512.000 \text{ Byte}$
 $\text{Übertragungszeit} = 512.000 \text{ Byte} / 1.600 \text{ Byte/s} = \mathbf{320 \text{ s}}$
- b) Wie in Aufgabe a) übertragen wir 12Bit / Nutzdatenbyte, und erhalten eine Nutzdatenrate von 1.600 Byte/s
 $500 \text{ Kbyte} = 1024 * 500 \text{ Byte} = 512.000 \text{ Byte}$
 Paketgröße bei Kermit maximal: 96 Byte davon 91 Byte Daten und 5 Byte Header/Checksumme (das EOP Byte ist bei Kermit optional und wurde in dieser Rechnung nicht berücksichtigt. Lösungen, die das EOP Byte ebenfalls berücksichtigen, werden natürlich ebenfalls akzeptiert.)
 $\Rightarrow 5.626 * 91 \text{ Byte} + 1 * 34 \text{ Byte} = 512.000 \text{ Byte}$
 $\Rightarrow 5.626 \text{ Pakete} \rightarrow 96 \text{ Byte und } 1 \text{ Paket} \rightarrow 39 \text{ Byte} = 540.135 \text{ Byte und}$
 $\Rightarrow 5.627 \text{ Ack-Pakete} \rightarrow 5 \text{ Byte} = 28.135 \text{ Byte}$
 $\Rightarrow \Sigma = 568.270 \text{ Byte}$
 $\Rightarrow \text{Übertragungszeit} = 568.270 \text{ Byte} / 1.600 \text{ Byte/s} = \mathbf{355,16875 \text{ s}}$
 Bemerkung: Die tatsächliche Übertragungszeit ist etwas länger, weil auch noch die Pakete Send-Init, File-Header und End-Of-Transmission sowie das optionale File-Attribute Paket übertragen werden. Diese sollten aber laut Aufgabenstellung nicht berücksichtigt werden. Grund: die Größe des File-Header Pakets ist variabel und kann daher nicht allgemeingültig berechnet werden, da hier unter anderem der Dateiname übertragen wird, welcher variable Länge haben kann.
- c) Mehrere Möglichkeiten denkbar, 1 Punkt pro Nennung, die folgende Liste erhebt keinen Anspruch auf Vollständigkeit
- Vergrößerung der Pakete, Nachteil: Wahrscheinlichkeit, dass Paket durch Störungen auf der Leitung kaputt geht, steigt
 - Verkleinerung des Headers. Das Typ-Feld ist z.B. ein ganzes Byte (= 8 Bit) lang, obwohl es nur 8 verschiedene Pakettypen gibt. 8 Werten lassen sich bereits mit nur 3 Bit kodieren. Ebenso könnte das Sequenznummern-Feld oder das Längensfeld kleiner gewählt werden. Es wäre theoretisch auch möglich, bestimmte Felder komplett wegzulassen, allerdings würde sich dadurch die Semantik des Protokolls ändern. Ein Weglassen der Checksumme würde z.B. den Overhead verringern, aber eine Fehlererkennung wäre dann nicht mehr möglich.

- *Anstatt nach jedem einzelnen Paket ein Acknowledge zu schicken, könnte man das Acknowledge seltener (z.B. erst am Ende der Datei schicken). Solange es keine Störungen auf der Leitung gibt, würde dies die Effizienz des Protokolls verbessern. Treten allerdings Störungen auf, müssten alle Pakete noch mal übertragen werden auch wenn nur ein einziges Paket kaputt war!*