Übung 6 zur Vorlesung "Vernetzte Systeme" WS 1999/2000

Prof. Dr. F. Mattern

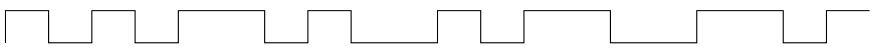
Ausgabedatum: 24. Nov. 1999
Abgabedatum: 1. Dez. 1999

23. Aufgabe (Kodierungen)

Wichtige Anforderungen an Kodierungen für die Übertragung von Bitströmen sind die *Gleichspannungsfreiheit*, der *Bandbreitenbedarf* und die Möglichkeit der *Taktrückgewinnung*.

a) (2 Punkte) Beurteilen Sie die vier aus der Vorlesung bekannten Kodierungsverfahren bezüglich der genannten Anforderungen. Hinweis: Betrachten Sie für den Bandbreitenbedarf die Anzahl der Spannungswechsel, die für die Übertragung eines Bits benötigt werden.

b) (1 Punkt) Folgende Signalform gehört zu einem Datenstrom, der nach der Manchester-Methode kodiert wurde. Bestimmen Sie Anfang und Ende der Bit-Intervalle, indem Sie die Taktinformationen aus dem Signal extrahieren und geben Sie dann die Bitfolge an, die durch diese Signalform repräsentiert wird.



c) (2 Punkte) Zeichnen Sie für den Binärwert 11100000000001001100001 das kodierte Signal für die vier unter a) genannten Codierungen.

24. Aufgabe (Signal-Rausch-Abstand)

Nach Shannon gibt es einen eindeutigen Zusammenhang zwischen der verfügbaren Bandbreite, dem Signal-Rausch-Abstand und der maximal möglichen Anzahl übertragbarer Bit pro Sekunde[1].

[1] Bitte beachten Sie: b = bit, kb = kilobit = 10^3bit

a) (2 Punkt) Der Signal-Rausch-Abstand einer Satellitenverbindung betrage 20 dB, die Bandbreite eines Videokanals 10MHz. Wie hoch ist die maximale Bitrate in b/s?

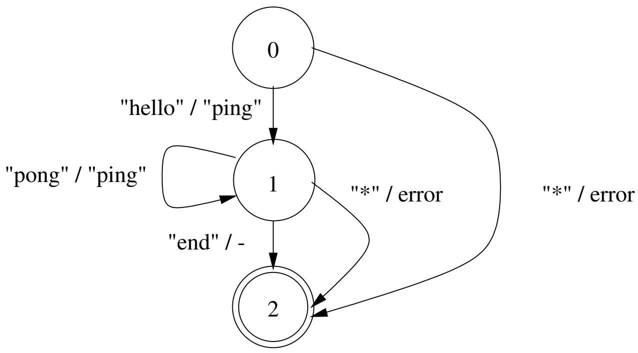
b) (2 Punkt) In einem digitalen Telefonnetz möchte man bei einer Bandbreite von 3100 Hz eine Datenrate von 56 kb/s erreichen. Wieviel dB muss der Signal-Rausch-Abstand mindestens betragen?

c) (3 Punkt) Nach dem informationstheoretischen Theorem von Shannon ist es nicht möglich, mehr als ca. 30 kb/s über eine typische analoge Telefonleitung zu übertragen. Erklären Sie mit eigenen Worten, wie es möglich ist, dass mit V.90-Modems bis 54 kb/s übertragen werden können.

25. Aufgabe (FTP-Client)

In Aufgabe 22 haben Sie einen einfachen HTTP-Client in Java entwickelt. Bei HTTP handelt es sich um ein relativ einfaches Protokoll: auf eine Anfrage vom Client folgt immer eine Antwort vom Server. Ausserdem ist der Client zustandslos, d.h. die Verarbeitung der Antworten vom Server erfolgt unabhängig vom Zustand des Clients immer gleich.

Komplexere Protokolle, bei denen die Bearbeitung von Eingaben zusätzlich vom Zustand des Clients oder Servers abhängt, beschreibt man oft mittels endlicher Automaten. Folgende Abbildung zeigt den Zustandsautomaten des Clients für ein einfaches "Ping-Pong-Protokoll".

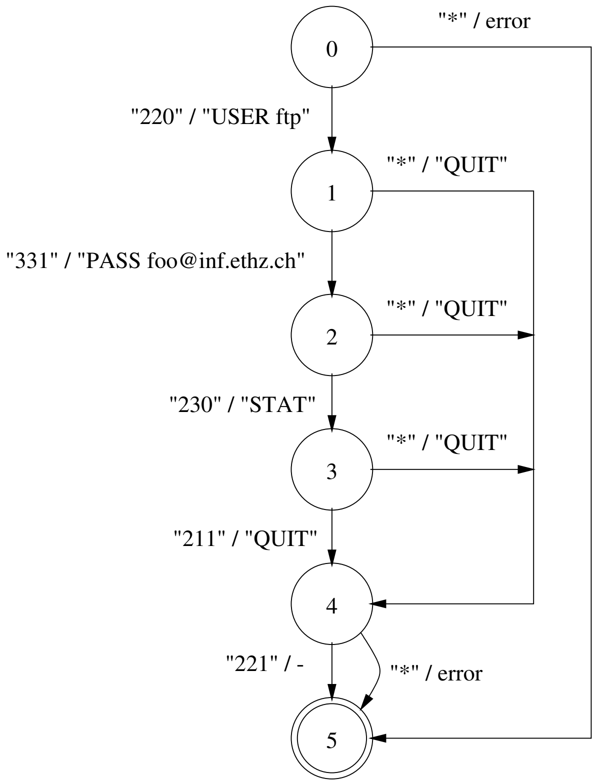


Der Client kann sich in drei möglichen Zuständen (0, 1, 2) befinden. Zustand 0 ist der Startzustand, in dem sich der Client zu Beginn befindet. Die Zustandsübergänge sind jeweils mit `Eingabe/Aktion` beschriftet, wobei `Eingabe` die vom Server erhaltene Eingabe und `Aktion` die daraufhin ausgelöste Aktion (meistens eine Ausgabe an den Server) ist.

Empfängt der Client im Zustand 0 die Eingabe "hello", so sendet er "ping" an den Server und begibt sich in Zustand 1. Alle anderen Eingaben ("*") sind ein Fehler und der Client begibt sich in Zustand 2 (den Endzustand). Empfängt der Client in Zustand 1 ein "pong" vom Server, so antwortet er mit einem erneuten "ping" und bleibt in Zustand 1, solange, bis er ein "end" oder eine unzulässige Eingabe empfängt und sich in den Endzustand begibt. Eine beispielhafte Implementierung dieses Protokolls in Java finden Sie unter `http:`

//www.inf.ethz.ch/department/IS/vs/education/WS9900/VS/. Machen Sie sich insbesondere mit der Methode `handleInput()` des Clients vertraut, die den obigen Zustandsautomaten implementiert.

Ihre Aufgabe ist es nun, einen einfachen Client für FTP (File Transfer Protocol, als Standarddienst im Internet unter Port 21 erreichbar) zu entwickeln. Der Client soll sich bei einem FTP-Server als anonymer Benutzer anmelden, den Status des Servers abfragen und sich dann abmelden. Dazu müssen bestimmte Kommandos an den Server geschickt und die Antworten des Servers ausgewertet werden. Die Antworten beginnen mit einem dreistelligen Statuscode, gefolgt von einem Leerzeichen (Statuscodes, die von einem '-' gefolgt werden, sollten ebenso wie Zeilen, die nicht mit einem Statuscode beginnen, ignoriert werden[2]) und weiterem Text. Der Client kann durch folgenden Zustandsautomaten beschrieben werden:



Machen Sie sich zunächst mit dem Protokoll vertraut, indem Sie den Zustandsautomaten von Hand mit `telnet` nachvollziehen oder nachfolgende Beispielsitzung durcharbeiten.

(8 Punkte) Implementieren Sie diesen Zustandsautomaten (am einfachsten, indem Sie den Ping-Pong-Client entsprechend abändern). Testen Sie Ihre Implementierung, indem sie folgende FTP-Server ausprobieren: `vs2.inf.ethz.ch, www.inf.ethz.ch` und `ftp.redhat.com`. Schicken Sie den Quelltext per E-mail (als Attachement) an Ihren Tutor.

[2] Mittels der Methods `charAt(int pos)` können Sie auf einzelne Zeichen eines Strings zugreifen.

In der folgenden Beispielsitzung sind Eingaben mit '>' markiert. Überlegen Sie sich, welche Zeilen (bzw. Statuscodes) Sie ignorieren müssen.

```
220 FTP server ready.
> USER ftp
331 Guest login ok, send your complete e-mail address as password.
> PASS foo@inf.ethz.ch
230-     THE SOFTWARE AVAILABLE FROM THIS SITE IS PROVIDED AND LICENSED
230-     "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR
230-     IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES
230-     OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE.
230-
230-     Priority FTP access (via priority.redhat.com) is for the use
230-     of registered users only.  Red Hat reserves the right to cancel
230-     access at its discretion.
230-
230 Guest login ok, access restrictions apply.
> STAT
211-pub.iad.redhat.com FTP server status:
     Version wu-2.6.0(1) Thu Oct 21 12:27:00 EDT 1999
     Connected to vs2.inf.ethz.ch (129.132.13.71)
     Logged in anonymously
     TYPE: ASCII, FORM: Nonprint; STRUcture: File; transfer MODE: Stream
     No data connection
     0 data bytes received in 0 files
     0 data bytes transmitted in 0 files
     0 data bytes total in 0 files
     35 traffic bytes received in 0 transfers
     1009 traffic bytes transmitted in 0 transfers
     1094 traffic bytes total in 0 transfers
211 End of status
> QUIT
221-You have transferred 0 bytes in 0 files.
221-Total traffic for this session was 1204 bytes in 0 transfers.
221-Thank you for using the FTP service on pub.iad.redhat.com.
221 Goodbye.
```