H.-P. Helfert, DL6MAA, Gustav-Mueller-Str. 8 , D-8948 Mindelheim
U. Strate, DF4KV, Lommerwiese 18, D-5330 Koenigswinter 1

PACTOR - Einfuehrende Protokollbeschreibung (Level 1)

(Stand: 5. November 1990)

Zielsetzungen

PACTOR (PT, lat.: "der Vermittler") wurde von Funkamateuren fuer Funkamateure im Rahmen des Experimentierfunkdienstes entwickelt; eine kommerzielle Nutzung ist nicht vorgesehen.

PACTOR soll den derzeit bestehenden Mangel an einer zuverlaessigen und schnellen Fernschreibbetriebsart auf Kurzwelle, besonders fuer die Bereiche unterhalb 10 MHz, beseitigen.

Auf folgende Punkte wurde beim PT-Design besonderer Wert gelegt:

>>> Fehlerfreie Datenuebertragung
(praktisch: Fehlerquote kleiner 10 ** -5)

>>> Echte binaere Datenuebertragung
(z. B. kompletter ASCII-Zeichensatz)

>>> Gute Ausnutzung der aktuellen Kanalkapazitaet
(automatische Anpassung der Schrittgeschwindigkeit, vollstaendig bitsynchrone Uebertragung, Memory-ARQ, keine speziellen Requestbloecke)

>>> Sehr kleiner noetiger Stoerabstand fuer Aufrechterhaltung einer Verbindung

>>> Einfache Betriebsabwicklung, optimierter Halbduplexbetrieb mit schnellem Richtungswechsel und beidseitig bestaetigtem QRT

>>> Shiftlagenneutralitaet (Mark- und Spacedefinition entfaellt)

>>> Realisierbarkeit mit einfacher Hardware
(Level 1 auch fuer Portabel- und Notfunk verwendbar an uebli-
chen Amateurfunkgeraeten)

>>> Noetige Bandbreite maximal 600 Hz

>>> Vollstaendige Transparenz fuer Dritte (z. B. behoerdliche Ueber-
wachung)

>>> Kompatibilitaet zu Erweiterungen durch Level-Definition und das Supervisorkonzept.

Folgende Erweiterungen sollten bei Einfuehrung leistungsfaehigerer Hardware erprobt werden:

* Effizientere Quellencodierung (z. B. Markow-Codierung)
* Adaptive fehlerkorrigierende Kanalkodierung
* Modifiziertes Memory-ARQ
* Modemschaltungen mit Signalprozessoren zur bitsynchronen Demodulation und automatischen Frequenzgangkorrektur bzw. zur Erprobung neuer Modulationsverfahren
* Hoehere Schrittgeschwindigkeiten fuer QRG's > 10 MHz
(automatische Umschaltmoeglichkeit auf Modus 200/400 Bd)

Protokoll-Grundzuege

Das PACTOR-Grundschema aehnelt der AMTOR-Struktur, die sehr gut an den auf Kurzwelle ueblichen Dialogbetrieb angepasst ist. Es handelt sich um Halbduplex-ARQ: Datenpakete (Bloecke), die die Nutzinformation tragen, werden durch kurze Quittungssignale (Controls) der empfangenden Station bestaetigt. Im Fehlerfall kann der Blockempfaenger durch entsprechende Controls Wiederholungen eines Datenpaketes anfordern.

* Synchronbetrieb

Um die verfuegbare Kanalkapazitaet gut auszunutzen, bedarf es einer bitsynchronen Uebertragung. Der Synchronismus muss dabei waehrend der gesamten Verbindungsdauer aufrechterhalten werden.

* Paketlaenge

Eine laengere Versuchsreihe hat gezeigt, dass es fuer den Betrieb in stark fluktuierenden Kanaelen nicht sinnvoll ist, die Datenpaketlaenge an die aktuellen Bedingungen automatisch anzupassen.
Eine Aenderung des Zeitrasters fuehrt besonders in schlechten Kanaelen zu Ueberschneidungen von Datenbloecken und Controls, so dass diese Methode als unwirtschaftlich verworfen wurde.
Die Frage nach der guenstigsten Paketlaenge wurde durch Ausmesssen verschiedener Kurzwellenkanaele mit einer FSK-Uebertragung beantwortet. Die gewonnenen Daten wurden durch eine Computersimulation auf ARQ-Systeme uebertragen. Als im Langzeitmittel optimale Kurzwellen-Paketlaenge ergab sich dabei eine Dauer von etwa einer Sekunde.
Die Zyklusdauer (Paketlaenge + Empfangspause) sollte in jedem Fall deutlich kleiner als die haeufigste Fading-Periodendauer sein.
Bei der Computersimulation gehen auch kurze Stoerimpulse ein; ohne Beruecksichtigung dieser "Bursts" koennte die Paketlaenge erheblich ver-
groesert werden.
Alternativ koennte das Problem kurzer Fehlerbuendel aehnlich wie in der digitalen Audiotechnik durch Anwendung fehlerkorrigierender Faltungscodes geloest werden. Deren Vorteile machen sich im allgemeinen aber erst bei bekannter, annaehernd konstanter Kanalqualitaet bemerkbar.
Eine Verkuerzung der Paketlaengen kommt im uebrigen der Betriebstechnik (kurze Break-In-Zeiten) entgegen.

* Paketsicherung

Die Datenbloecke werden durch CRC gesichert. Die Anwendung fehlerkorrigie-
render Codes ist zunaechst nicht vorgesehen, kann jedoch in spaeteren aufwaertskompatiblen Versionen in Verbindung mit leistungsfaehigerer Hard-
ware realisiert werden.

* Nutzung fehlerhafter Pakete

Um die in fehlerhaft empfangenen Paketen enthaltene Information zu nutzen, bedient sich PACTOR eines Memory-ARQ-Verfahrens (s. u.), das eine deutliche Steigerung der Uebertragungsrate bei geringen Stoerabstaenden bewirkt.

* Anpassung der Schrittgeschwindigkeit

Die Informationsgeschwindigkeit im realen Kanal ist einerseits durch die verfuegbare Bandbreite (bei gegebenem Stoerabstand) und andererseits bei binaerem FSK auch durch auftretende Phasenverzerrungen infolge Mehrwegeausbreitung begrenzt.

Auf Kurzwelle konnten mit 200 Baud besonders waehrend der Tagesstunden gute Ergebnisse erzielt werden. Am Abend treten verstaerkt Laufzeitverzerrungen auf, die eine Reduzierung der Schrittgeschwindigkeit auf 100 Baud erforderlich machen.
PACTOR verfuegt ueber eine automatische Steuerung zur Umschaltung auf die jeweils guenstigste Schrittgeschwindigkeit.

Die Protokolldetails

Grundsaetzliche Begriffe:

```
MASTER = Station, die den Verbindungsaufbau initiiert.
SLAVE  = die angerufene Station
TX     = Sender von Informationspaketen
RX     = Empfaenger von Informationspaketen
BK     = Break-In , vom RX ausgeloester Senderichtungswechsel
```

Im folgenden wird von einer byteweisen Datenorganisation ausgegangen, da dies der Realisierung auf Mikrorechnern entgegenkommt. Das niederwertigste Bit wird zuerst ausgesendet.
Die Shiftlage der FSK-Aussendung wird einmalig beim Verbindungsaufbau fixiert. Mit jedem neuen Paket oder Kontrollsignal wird die Shiftlage invertiert.

* Timing

PACTOR arbeitet als bitsynchrones System mit einem festen Zeitraster.

```
Gesamt-Zyklusdauer                 : 1.25 sec
Paketdauer                         : 0.96 sec
Fenster fuer Kontrollsignalempfang : 0.29 sec
Kontrollsignaldauer                : 0.12 sec
```

Es verbleibt eine Restzeit von 170 ms fuer Umschaltvorgaenge und die Signallaufzeit, so dass sich wie bei AMTOR eine maximale DX-Entfernung von ca. 20000 km ergibt (geringfuegige Korrekturen gegenueber cq-DL 11/90).

Systemtakt-Betriebsmodi:

1) Intern (Standardmodus): Quarzzeitbasis mit Genauigkeit 15 ppm oder besser. Der SLAVE-Takt wird auf den MASTER-Takt synchronisiert (Auswertung der Flankenwechsel).
2) Extern (vorgesehen fuer Spezialanwendungen): Beide Stationen leiten ihren Systemtakt von einer Zentralquelle (z. B. 50-Hz-Verbund, TV-Zeilenfrequenz, DCF77) ab.

* Kontrollsignal- und Paketaufbau

Die Kontrollsignale (CS) werden vom RX als Empfangsbestaetigung ausgesendet. Wiederholung des gleichen CS bedeutet 'REQUEST', d. h. die Anforderung einer Blockwiederholung. CS3 bildet den Kopf des BK-Paketes, CS4 fordert einen Geschwindigkeitswechsel an (s. u.).
Die CS haben eine einheitliche Laenge von 12 Bit und werden stets mit einer Schrittgeschwindigkeit von 100 Baud ausgesendet. Die Bitmuster sind so gewaehlt, dass sich optimale Hammingdistanz (8 Bit) und guenstige Symmetrieeigenschaften ergeben.

Bezeichnung	Funktion	Bitmuster(HEX)
CS1	Bestaetigung	4D5
CS2	Bestaetigung	AB2
CS3	Break-In	34B
CS4	Speedchange	D2C

Alle Pakete, ausgenommen Spezialfaelle, bestehen aus den drei Teilen Kopf (Header), Datenbereich und Verwaltungsteil (Status + CRC).

Aufbau der Datenpakete

100 Baud :

/Header/ Datenbereich 64 Bit /Statusbyte/CRC1/CRC2/

200 Baud :

/Header/ Datenbereich 160 Bit /Statusbyte/CRC1/CRC2/

Erlaeuterungen:

1) Header : Bitmuster 55(HEX)

Das Kopfbyte liefert Zusatzinformation, die zur Synchronisation, zur REQUEST-Erkennung bei Memory-ARQ sowie beim Monitor-Modus genutzt wird. Bei jedem Paket, das neue Information enthaelt, wird das Bitmuster invertiert.

2) Datenbereich:

Der Datenbereich kann beliebige Binaerinformation enthalten; die Art der Codierung wird im Statusbyte angegeben. Gegenwaertig kann zwischen 8-Bit-ASCII und 7-Bit-ASCII (Huffman-komprimiert) gewaehlt werden. Das ASCII-Zeichen RS (1E HEX) dient in beiden Faellen als IDLE-Symbol.
Der Huffman-Code eignet sich besonders zur Uebertragung von deutschem und englischen Klartext. Die resultierende mittlere Symbollaenge betraegt ca. 4,5 Bit. Jedes Paket stellt eine Informationseinheit dar: Passt ein Symbol nicht mehr komplett in ein Paket, wird es im naechsten Datenpaket gesendet. Eine Code-Tabelle findet sich im Anhang.

3) Statusbyte-Belegung:

```
Bit 0:  Paketzaehler (LSB)
Bit 1:  Paketzaehler (MSB)         Datenmodus    Bit 3  Bit 2
Bit 2:  Datenmodus   (LSB) -->     -----------------------------
Bit 3:  Datenmodus   (MSB) -->     ASCII-8-Bit    0       0
Bit 4:  noch nicht belegt          Huffman-Code   0       1
Bit 5:  noch nicht belegt          nicht bel.     1       0
Bit 6:  BK-Anforderung             nicht bel.     1       1
Bit 7:  QRT-Bit
```

Die Bits 0 und 1 dienen als Paketzaehler; aufeinanderfolgende Pakete mit gleichem Zaehlerwert werden vom RX als REQUEST-Pakete erkannt. Die modulo-4-Zaehlung ermoeglicht die Behandlung des (in der Praxis sehr unwahrscheinlichen) Falles nicht erkannter Kontrollsignalfehler.

4) CRC-Bytes:

Der CRC wird nach CCITT-Norm ueber das gesamte Paket, beginnend mit dem Datenbereich (d. h. ohne Header bzw. CS3 bei BK-Paketen) berechnet.

* Aufbau einer PACTOR-Verbindung

Vom MASTER werden Synchronisationspakete mit folgendem Aufbau ausgesendet:

/Header/S L A V C A L L/SLAVCA/   (Header= 55 HEX)

/<------100 Baud------>/200Bd /

Diese Pakete enthalten das SLAVE-Rufzeichen (max. 8 Byte, ASCII-Codiert) als 100 Baud- und 200-Baud-Bitmuster. Bei kuerzeren Rufzeichen werden Leerbytes mit 0F(HEX) aufgefuellt. Die SLAVE-Station sucht den einlaufenden Bitstrom nach Synchronisationspaketen mit ihrem Rufzeichen ab (Beide moeglichen Shiftlagen werden geprueft) und antwortet nach erfolgreicher Synchronisation mit der Aussendung von CS. Die eigentliche Synchronisation wird mit dem Head-Byte und einer waehlbaren Anzahl Bytes des ersten Rufzeichenteils vorgenommen. Das 200 Bd-Muster dient lediglich zur Ueberpruefung der Kanalqualitaet. Wird es fehlerfrei empfangen, wird mit CS1 geantwortet, ansonsten mit CS4, was zur sofortigen Geschwindigkeitsreduzierung auf 100 Baud fuehrt.
Analog dazu sucht der MASTER im Empfangsfenster zwischen zwei Paketen nach gueltigen CS in beiden Shiftlagen. Sobald das erste gueltige CS empfangen und synchronisiert ist, wird das erste normale Datenpaket mit Head=AA (HEX) und Paketzaehler=1 ausgesendet.

* Levelfestlegung beim Verbindungsaufbau

Beim Verbindungsaufbau ist der maximal moegliche Systemlevel der Gegenstelle unbekannt, daher wird jede Verbindung zunaechst auf dem kleinsten Level (1) gestartet.
Die ersten uebertragenen Zeichen enthalten Informationen ueber maximal verfuegbare Software-Levelnummer, gefolgt vom MASTER-Rufzeichen, abgeschlossen mit CR (Beispiel : 1DF4KV   <CR>). Stellt die SLAVE-Station eine Ueberforderung fest (kleinerer eigener Systemlevel), teilt sie dies dem MASTER mittels BK und Supervisorinformation (s. u.) sofort mit, ansonsten wird die Verbindung auf dem angegebenen MASTER-Level fortgesetzt.
Die User-BK-Moeglichkeit ist erst mit Abschluss der MASTER-Systeminformation (<CR>) freigegeben.

* Senderichtungswechsel

Nach einem korrekt empfangenen Paket kann die RX-Station mittels BK-Kontrollsignalen (CS3) eine Umkehrung der Senderichtung anfordern. Das CS3 bildet bereits den Kopf des ersten, vom neuen TX ausgesendeten Datenpaketes. Bei 200 Baud belegt das CS3 die ersten drei bei 100 Baud, die ersten zwei Bytes des Paketes, wobei in diesem Fall 4 Bit unbenutzt bleiben. Der CRC wird wie beim normalen Datenpaket erst vom Beginn des Datenbereichs an berechnet, der Paketzaehler wird auf 0 gesetzt. Das naechste Datenpaket nach einem BK-Paket erhaelt als Header 55(HEX).
Empfaengt der TX ein CS3, schaltet er sofort in den RX-Modus und liest das Restpaket vollstaendig ein. Bei fehlerfrei empfangenem Paket antwortet er mit CS1, bzw. CS3, falls ein sofortiges "Rebreak" gewuenscht wird. Eine Wiederholung des BK-Paketes wird mittels CS2 angefordert. Die Antwort-CS auf ein BK-Paket beginnen im Zeitraster eine CS-Laenge (0.12 sec) vor Ende des alten eigenen TX-Blocks.
Durch Setzen von Bit 6 im Statusbyte kann der TX auch selbst ein BK anfordern, da dieser Status den RX zur Bestaetigung mit CS3 veranlasst.

* Geschwindigkeitsverminderung 200 --> 100 Baud

Der RX kann immer nach einem fehlerhaften Paket ein CS4 senden, was auf der TX-Seite immer als "REJECT" interpretiert wird: Das gerade gesendete, nicht bestaetigte Paket wird verworfen und die in ihm enthaltene Information erneut im 100-Baud-Modus ausgesendet. Der Paketzaehlerwert des ersten 100-Bd-Paketes wird vom verworfenen 200-Baud-Paket uebernommen. Empfaengt der RX einen 100-Bd-Block mit gleichem Paketzaehler wie im vorangegangenen 200-Bd-Paket, so bedeutet dies doppelt aufgenommene Information: Es muessen soviele Zeichen (ausgenommen IDLE) von der Ausgabe ferngehalten werden, wie im letzten 200-Bd-Paket ankamen. Nach einem "speeddwn"-CS4 stellen RX und TX den Header auf 55(HEX), um korrekte Memory-ARQ-Funktion (s. u.) sicherzustellen.

Sonderfaelle:

'REJECT' eines QRT-Paketes: --> Aussenden eines 100-Bd-QRT-Paketes
'REJECT' eines BK-Paketes : --> Aussenden eines normalen 100-Bd-Paketes mit Head= 55(HEX) statt des CS3-Kopfes.
CS4 dient als "REQUEST"-CS fuer den ersten 100-Bd-Block, Bestaetigung erfolgt mittels CS1 oder CS3, danach kann bereits wieder ein CS4 als "speedup"-Signal gesendet werden (s. u.).

* Geschwindigkeitserhoehung 100 --> 200 Baud

Der RX kann nach jedem richtig empfangenen 100-Bd-Paket (ausgenommen "REQUEST"-Pakete) mit CS4 bestaetigen, was den TX zur Umschaltung auf 200 Baud zwingt. Der RX schaltet sofort nach CS4-Aussendung auf 200-Baud-Empfang. (Sollte der TX das CS4 nicht erhalten, so ist dies nicht mit zusaetzlichem Informationsverlust verbunden, obwohl der RX bereits die "falsche" Geschwindigkeit eingestellt hat. Das neue Paket waere ein "Request"-Paket, also irrelevant.)
CS4 in Folge (ohne zwischenzeitliches richtiges CS) werden als "Request" interpretiert, also ignoriert.
Nach erfolgreicher Geschwindigkeitserhoehung muss mindestens noch ein zweites 200-Bd-Paket vor dem naechsten CS4 ("speeddown) empfangen werden.
Ablehnungsmoeglichkeit:
Erhaelt der RX nach dem CS4 nicht spaetestens nach einer vorgewaehlten Anzahl Zyklen ein fehlerfreies 200-Bd-Paket, so antwortet er mit einem OK-CS (verschieden vom letzten CS vor dem CS4) und schaltet danach zurueck auf 100 Baud. Der TX muss seinerseits das auf CS4 folgende CS auf OK testen, in diesem Fall unverzueglich auf 100 Bd zurueckschalten und die 200 Bd-Information wiederholen (bzw. im QRT-Fall einen 100-Baud-QRT-Block aufbauen).

* Supervisorkonzept

Um Kompatibilitaet zu kommenden Software-Versionen zu gewaehrleisten, muss bereits im Level-1-Protokoll die Uebertragung von Systeminformationen (SI) moeglich sein. Zu diesem Zweck wird ein Supervisor-Zeichen (SB) definiert, das die normale Textausgabe auf die Systemebene umlenkt. Das auf das SB folgende Zeichen enthaelt den Supervisor-Funktionscode (SIC), gefolgt von Parametern. Den SI-Abschluss bildet ein <SPACE>.

Vorlaeufige SI-Definitionen:

SB: ASCII <FS> = 1C (HEX)

SIC:

A: Levelnummer folgt ('1' = Level 1)
B: Mycall folgt (kann z. B. automatisch in IDLE-Pakete eingefuegt werden, um mitlesenden Dritten die Identifikation zu ermoeglichen)
a .. <DEL> : Controlzeichen ausgeben 01 bis 1F (HEX) ; ermoeglicht volle Datentransparenz

Die gesamte SI laeuft wie gewoehnlicher Text ueber das PT-System, um Komplikationen beim Geschwindigkeitswechsel zu vermeiden.

Sofort benoetigte SI muss in den laufenden Text eingeschoben werden, fruehestens nach dem letzten bereits (mindestens einmal) bearbeiteten Zeichen, um im Wiederholungsfalle Fehler zu vermeiden.

* Beenden einer PACTOR-Verbindung

Der Abschluss einer ARQ-Verbindung bring allgemein das Problem mit sich, dass Information ohne echte Bestaetigung gesendet werden muss, da immer wieder eine Bestaetigung der Bestaetigung noetig waere fuer einen definierten Verbindungsabbruch.
PACTOR benutzt spezielle QRT-Pakete, um die Gefahr eines unkontrollierten QRT zu minimieren:

100 Baud:

/Header/ G I S L L A C/Header/Satusbyte/CRC1/CRC2/

200 Baud:

/Header/X/ G I S L L A C/Header/XXX/Statusbyte/CRC1/CRC2/

/<------100 Baud------>/

(maximal 7 signifikante Bytes zulaessig)
(X=beliebig)

Das QRT-Paket wird je nach aktueller Geschwindigkeit nach obigem Schema aufgebaut, wobei das Rufzeichen der Gegenstation ab dem 17. 200-Bd-Datenbyte von rechts nach links stehen muss. Das Rufzeichen wird immer als 100-Bd-Bitmuster gesendet; das QRT-Bit im Statusbyte muss gesetzt sein.

Wird ein QRT-Paket empfangen, sendet der RX noch ein Bestaetigungskontrollzeichen und geht dann in den STBY-Zustand. Das QRT-Paket darf nicht mehr der Dekodierung (Ausdruck) zugefuehrt werden.

Da es bei gestoerten Kanaelen sehr wahrscheinlich ist, dass das erste Bestaetigungs-CS nicht korrekt vom TX empfangen wird, muss der in den STBY-Zustand gegangene RX auch in der normalen Suchphase noch mit dem korrekten CS auf QRT-Pakete antworten.

Der RX sucht dafuer den eingehenden Bitstrom nach dem bytereversen Rufzeichen im QRT-Paket ab und synchronisiert gegebenenfalls darauf. Jedes empfangene QRT-Paket fuehrt zur Aussendung eines Betstaetigungs-CS, wobei die Shiftlage des CS sich nach der Shiftlage des QRT-Paketes richtet. Der RX muss sich dazu natuerlich die alte Relation (vor STBY) von CS-Shiftlage und Blockshiftlage merken; ausserdem muss der CS-Zeitpunkt gespeichert werden.

* Memory-ARQ

Normale ARQ-Systeme verzichten auf die in fehlrhaften Paketen enthaltene Information. Eine solche Vorgehensweise fuehrt bei geringer werdendem Stoerabstand sehr rasch zum Totalausfall einer Verbindung, da es extrem unwahrscheinlich wird, ein Paket fehlerfrei zu empfangen.

Memory-ARQ summiert die Empfangsspannung (Nutz- plus Rauschspannung) gleicher (nocht nicht fehlerfrei aufgenommener) Pakete bitweise auf, wodurch sich der effektive Signal-Rauschabstand im Idealfall um den Faktor n (=Anzahl aufsummierter Pakete) erhoeht.

Beispiel fuer Fall n=2 und konstanter Signal- und Rauschleistung:

(N = Rauschspannung, S = Signalspannung, <..> = Mittelwert)

* Summenspannung: 2*S + N1 + N2;

* Leistung des Summensignales:

<(2*S + N1 + N2)**2> = 4*S**2 + <N1**2> + <N2**2> + <gemischte Terme>;

<gemischte Terme> = 0; <N1**2> = <N2**2> = Rauschleistung;

--> S/N verdoppelt bei n=2, analog fuer beliebige n. (Dies entspricht letztendlich nur dem Satz, dass die Bitenergie in den effektiven Signal-Rauschabstand eingeht --> spezielle Form des Energieerhaltungssatzes. Bitlaenge und Bitleistung einzeln betrachtet sind aussagelos. Memory-ARQ nutzt nutzt also die gesamte Signalenergie aus.
Memory-ARQ senkt den noetigen Mindeststoerabstand fuer eine Verbindung um 1 - 15 dB ab. In realen, also fluktuierenden Kanaelen muss vermieden werden, dass Pakete mit relativ geringem Stoerabstand den Stoerabstand des Summenpaketes wesentlich verschlechtern. Hierzu misst man den Stoerabstand ueber die gesamte Paketlaenge (relativ kleine Unschaerfe) und gewichtet die Summanden mit diesem gemessenen Paketstoerabstand.
Um Quantisierungsrauschen minimal zu halten, wird ein 8-Bit A/D-Wandler zum Einlesen der Empfangsspannung empfohlen.

In das PT-Protokoll sind zum Memory-ARQ folgende Punkte aufgenommen worden:

Summenpakete, deren CRC passt, werden wie normale fehlerfreie Pakete behandelt. (Da der 16-Bit CRC eine hohe Sicherheit gewaehrleistet, bringt Memory-ARQ in der Praxis keinerlei Nachteile mit sich.)

Alte "Request"-Pakete, die durch Stoerung eines OK-CS vom TX ausgesendet werden, duerfen nicht in die Summation gelangen. Hierzu fuehrt man einen "Similarity"-Test durch, der bei PACTOR aus der Ueberpruefung des Headbytes besteht. Bei neuen Paketen (mit neuer Informaton) wird der Header invertiert, so dass der RX auch ohne CRC (Stoerungsfall) erkennen kann, ob es sich um ein "Request"-Paket handelt. Jedes richtige Paket (CRC in Ordnung) fuehrt zur Loeschung der Memory-ARQ-Summen.
Bei CS3-Paketen muss der "Similarity"-Test natuerlich mit dem CS3-Header durchgefuehrt werden.

Das PT-Memory-ARQ ist dank der permanent wechselnden Shiftlage auch faehig, konstante Stoersignale (Traeger) oder andere Unsymmetrien im Kanal auszuumitteln.

Anhang

PT-Huffman-Code

Der Huffman-Code ist relativ unempfindlich gegenueber Abweichungen der realen von der theoretischen Buchstabenhaeufigkeit, so dass fast gleich gute Ergebnisse bei deutschem und englischem Klartext erreicht werden. Der erreichbare Kompressionsfaktor gegenueber ASCII betraegt ca. 1.7 (bei einer Entropie von 4.5 Bit/Zeichen). Wesentliche Verbesserungen koennten bei Beruecksichtigung der statistischen Abhaengigkeiten der einzelnen Zeichen erzielt werden (Markow-Quellencodierung).

Code geordnet nach Haeufigkeit,
LSB (uerst gesendet) links!

```
space      :   32           10
e          :   101          011
n          :   110          0101
i          :   105          1101
r          :   114          1110
t          :   116          00000
s          :   115          00100
d          :   100          00111
a          :   97           01000
u          :   117          11111
l         :   108          000010
h          :   104          000100
g          :   103          000111
m          :   109          001011
           :   13           001100
           :   10           001101
o          :   111          010010
c          :   99           010011
b          :   98           0000110
f          :   102          0000111
w          :   119          0001100
D          :   68           0001101
k          :   107          0010101
z          :   122          1100010
.          :   46           1100100
,          :   44           1100101
S          :   83           1111011
A          :   65           00101001
E          :   69           11000000
p          :   112          11000010
v          :   118          11000011
0          :   48           11000111
F          :   70           11001100
B          :   66           11001111
C          :   67           11110001
I          :   73           11110010
T          :   84           11110100
O          :   79           000101000
P          :   80           000101100
1          :   49           001010000
R          :   82           110000010
(          :   40           110011011
)          :   41           110011100
L          :   76           110011101
N          :   78           111100000
Z          :   90           111100110
M          :   77           111101010
9          :   57           0001010010
W          :   87           0001010100
5          :   53           0001010101
y          :   121          0001010110
2          :   50           0001011010
3          :   51           0001011011
4          :   52           0001011100
6          :   54           0001011101
7          :   55           0001011110
8          :   56           0001011111
H          :   72           0010100010
J          :   74           1100000110
U          :   85           1100000111
V          :   86           1100011000
           :   28           1100011001
x          :   120          1100011010
K          :   75           1100110100
?          :   63           1100110101
=          :   61           1111000010
q          :   113          1111010110
Q          :   81           1111010111
j          :   106          00010100110
G          :   71           00010100111
-          :   45           00010101111
:          :   58           00101000111
!          :   33           11110011101
/          :   47           11110011110
*          :   42           001010001100
"          :   34           110001101100
%          :   37           110001101101
'          :   39           110001101110
_          :   95           111100001100
&          :   38           111100111001
+          :   43           111100111110
>          :   62           111100111111
           :   64           0001010111000
$          :   36           0001010111001
<          :   60           0001010111010
X          :   88           0001010111011
#          :   35           0010100011011
Y          :   89           00101000110101
;          :   59           11110000110100
\          :   92           11110000110101
           :   91           001010001101000
           :   93           001010001101001
           :   127          110001101111000
           :   126          110001101111001
           :   125          110001101111010
           :   124          110001101111011
           :   123          110001101111100
           :   96           110001101111101
           :   94           110001101111110
           :   31           110001101111111
           :   29           111100001101100
           :   27           111100001101101
           :   25           111100001101110
           :   24           111100001101111
           :   23           111100001110000
           :   22           111100001110001
           :   21           111100001110010
           :   20           111100001110011
           :   19           111100001110100
           :   18           111100001110101
           :   17           111100001110110
           :   16           111100001110111
           :   30           111100001111000
           :   15           111100001111001
           :   14           111100001111010
           :   12           111100001111011
           :   11           111100001111100
           :   9            111100001111101
           :   8            111100001111110
           :   7            111100001111111
           :   6            111100111000000
           :   5            111100111000001
           :   4            111100111000010
           :   3            111100111000011
           :   2            111100111000100
```

```
          :   1           111100111000101
          :   0           111100111000110
          :   26          111100111000111


/*------------------------------------------------------------------------
 *          Datendurchsatzberechnung bei PACTOR
 *              ohne/mit Memory-ARQ
 *          Annahme: lineare Superposition
 *                                                          waa 10/90
 *-----------------------------------------------------------------------*/
#include <stdio.h>
#include <mth.h>

double sn,pp,pe,pss,psum,snx,db_min=-20.,db_step,db_max=15.;
double bw=600.,baudr=200.,bwd2br,ln10d10,db,pneu,pa,x;
int n,bits=192;

main()
{
 register int i,k,maxsum=400;

 bwd2br=bw/baudr/2.;     /*   Bandbreite/(2*Baudrate)         */
 ln10d10=log(10.)/10.;

 printf("\nEingabe Anzahl Datenpunkte :");
 scanf ("%d",&n);
 db=db_min;
 db_step=(db_max-db_min)/(n-1);   /* Schrittweite                     */
 printf("\n   Stoerabstand (dB)    Durchsatz normal (%%) \
    Durchsatz Memo-ARQ (%%) ");
 printf("\n----------------------------------------------------------\
------------------");
 for (i=0;i<n;i++){
   printf("\n\t %-8.3lf",db);
   sn=exp(db*ln10d10);            /* Umrechnung dB -> SN              */
   pe=0.5*exp(-sn*bwd2br);        /* FSK-Bitfehlerwahrscheinlichkeit */
   if (pe<0.1)
     pp=exp(log(1.-pe)*bits);     /* Paket-OK-Wahrscheinlichkeit     */
   else
     pp=0.0;
   pss=pp;
   pp*=100.;
   printf("\t    %8.3lf ",pp);
   pa=pss;
   psum=pss;
   for (k=2;k<=maxsum;k++){       /* Iteration fuer M-ARQ            */
     snx=sn*k;                    /* Ann. : lineare Stoerabstands-   */
     if (snx>50.)                 /*   Verbesserung bei Memo-ARQ     */
       pe=0.0;
     else
       pe=0.5*exp(-snx*bwd2br);
     if (pe<0.1)
       pp=exp(log(1.-pe)*bits);
     else
       pp=0.0;
     pneu=(1.-pa)*pss;
     pa+=pneu;
     x=(1.-pa)*pp;
     pa+=x;
     psum+=x/k;
   }
   psum*=100.;
   printf("\t\t%8.3lf",psum);
   db+=db_step;
 }
}

EOF.
```