mehrere Hilfsprogramme den Kassettenpuffer teilen) vor jedem Aufruf neu generieren. Die Schwierigkeiten, die dadurch entstehen, daß der Basic-Datenzeiger per RESTORE nur immer an den Datenanfang gesetzt werden kann, lassen sich beispielsweise durch ein in Computer persönlich, Ausgabe 10/84, Seite 52, beschriebenes Hilfsprogramm beseitigen, mit welchem der Datenzeiger an beliebige Stellen gesetzt werden kann. Das nun noch zu beschreibende Programm nach Listing 4 vermeidet die Schwierigkeiten durch ausschließliche Verwendung von POKE-Anweisungen anstelle von DATA-Zeilen. Um den Aufwand so gering wie möglich zu halten, wurde angestrebt, mit möglichst weniger POKE-Anweisungen auszukommen. Das wurde mit einem Trick erreicht: Das für die Rückübersetzung der Token beim Ausdrucken nach dem LIST-Befehl zuständige Maschinenunterprogramm des Betriebssystems enthält fast alle Befehlssequenzen, die wir für unsere Zwecke benötigen. Das Programm in Listing 4 lädt den Maschinenprogrammabschnitt 50929 bis 51008 (C6F1 — C740), der zugegebenermaßen viel für unsere Zwecke überflüssigen Ballast enthält, mit einer einfachen FOR-NEXT-Schleife in den Kassettenpuffer, (Es wird mit GOSUB700 angesprungen). Jeder zweite und weitere Ansprung kann mit GOSUB730 erfolgen und wird dann in kaum wahrnehmbaren Bruchteilen einer Sekunde ausgeführt.

(Fred Behringer/ev)

```
      LDY #$FF       ZEICHENZAEHLER
NEXT  INY            NAECHSTES ZEICHEN
      LDA $C09E,Y    IN AKKU
      BPL NORM       WORTENDE ?
      AND #$7F       DANN NORMALISIERT
      JSR $CB47      ZEICHEN AUSGEBEB
      LDA #$2E       WORTTRENNZEICHEN
NORM  JSR $CB47      ZEICHEN AUSGEBEB
      BNE NEXT       ZEICHEN IN AKKU = 0 ?
ENDE  RTS            DANN ZURUECK ZU BASIC
```

Listing 3. Assemblerdarstellung des nach Listing 2 erzeugten Maschinenprogramms

```
700 REM:HELP
710 FORI=0TO80:POKE828+I,PEEK(50929+I):N
EXT
720 POKE836,55:POKE892,169:POKE893,166:P
OKE897,6:POKE903,180:POKE909,96
730 POKE782,255:SYS898:RETURN
READY.
```

Listing 4. Simulation von HELP als Maschinenprogramm für Kassettenpuffer, ohne DATA-Zeilen. Länge 106 Bytes, Einlesen 1,5 s, Ausführung »augenblicklich«.

Genau betrachtet: RS232/V.24-Schnittstelle

Eine kurze und bündige Beschreibung der RS232-Schnittstelle Ihres C 64. Was machen die Signale, wie sind die Pin-Belegungen?

Bei der RS232-Schnittstelle werden die Daten Bit für Bit übertragen, im Gegensatz zur Centronics- oder IEEE-488-Norm, bei der ganze Bytes übergeben werden. Die Bits werden als eine Folge von Spannungsimpulsen mit einer bestimmten Dauer übertragen. In der Praxis werden dabei Pakete von 5 bis 8 Datenbit übertragen, die von einem Startbit und 1 bis 2 Stop-Bit eingerahmt sind (Bild 1). Das Startbit hat grundsätzlich logischen Low- und die Stop-Bits High-Pegel. Vor dem Stop-Bit kann ein sogenanntes Paritäts-Bit vereinbart werden, das die Anzahl der High-Zustände im Datenwort immer gerade oder ungerade macht.

Beispiel: Sind in einer 8-Bit-Übertragung 5 Bit gesetzt, wird das Paritäts-Bit ebenfalls gesetzt, wenn gerade Parität vereinbart wurde.

Päckchenweise Übertragung

Um die Störungs-Anfälligkeit der Übertragung zu mindern, wird logisch »Eins« (gesetztes Bit) nicht durch +5V (TTL-Pegel) realisiert, sondern mit einer Spannung von —3 bis —12V und logisch »Null« mit +3 bis +12V (RS232 nach DIN 66020). Eine andere Norm ist die RS232/TTY, die gegen äußere Störungen recht unempfindlich ist. Bei dieser Norm werden die logischen Zustände durch das Fließen oder Fehlen eines Stromes (20mA) dargestellt. Der C 64 hat zwar die nötige Software für eine RS232-Schnittstelle im Betriebssystem integriert, verfügt aber nicht über die entsprechenden Spannungspegel. Im C 64 gibt es nur zwei Spannungen: +5V (TTL) und 9V Wechselspannung. Es ist also ein Interface zur Spannungskonvertierung nötig. Links in Bild 2 finden Sie den Schaltplan eines solchen Interfaces (Bauanleitung in Ausgabe 3/85). Rechts im Bild die diskrete Lösung, für die Konvertierung von 0/5V auf ±12V (oben) und von ±12V auf 0/5V (unten). Beachten Sie, daß jede Sende- und Empfangsleitung die entsprechende Transistorschaltung braucht.

Mit einer Masse- und einer Datenleitung könnte schon eine Übertragung von Texten an einen Drucker erfolgen. Was ist aber, wenn die Datenübertragung schneller ist, als der

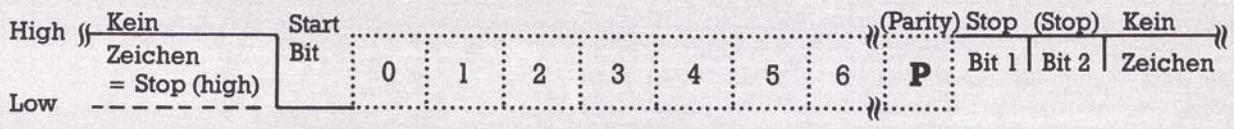

Bild 1. So sieht eine RS232-Übertragung schematisch aus

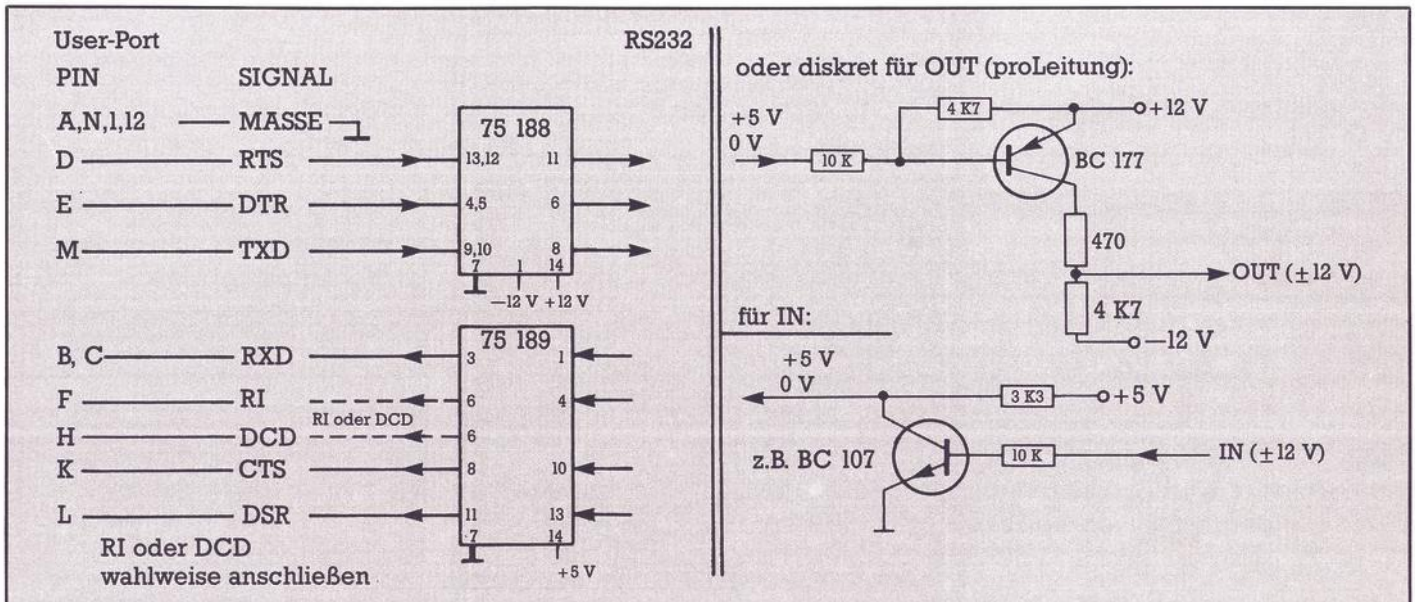

Bild 2. Schaltpläne für ein RS232/V.24-Interface. Links die Lösung mit IC, rechts die diskrete mit Einzelelementen

Drucker die Zeichen zu Papier bringen kann? Der Drucker muß dem Computer sagen, wenn er zuviel Arbeit bekommt. Er kann dies auf drei verschiedene Arten tun:

— Software-Protokoll mit XON/XOFF

Es wird eine zusätzliche Leitung zwischen Computer und Peripherie eingerichtet, über die das empfangende Gerät den Code $13 (XOFF) sendet, wenn es keine Daten mehr annehmen kann. Dieses Signal hat die gleiche Aufgabe wie die Busy-Leitung einer Centronics-Schnittstelle; es stoppt die Datenübertragung. Die Freigabe erfolgt mit dem Code $11 (XON). Die Codes $11 und $13 entsprechen den ASCII-Codes DC1 und DC2.

Die neue Leitung kann natürlich auch zur Übertragung von mehr Informationen verwendet werden. Sende- und Empfangsgerät müssen dann allerdings in der Lage sein, zwei Leitungen (XON/XOFF und die normale Datenleitung) gleichzeitig zu verwalten. Mit dieser zusätzlichen Leitung wird auch der sogenannte Vollduplex-Betrieb möglich. Vollduplex heißt, daß beide Geräte gleichzeitig senden oder empfangen können. Im Gegensatz zum Halbduplex-Betrieb, bei dem zur gleichen Zeit nur in eine Richtung übertragen werden kann.

— Software-Protokoll mit ETX/ACK

Auch bei dieser Lösung kommt man nicht ohne eine zusätzliche Leitung aus. Sie heißt DTR (Data Terminal Ready). Ist zum Beispiel der angeschlossene Drucker bereit, Daten anzunehmen, aktiviert er die DTR-Leitung und sendet $06 (ACKnowledge). Der Computer schickt nun eine Reihe Datenworte über die Sendeleitung und schließt die Übertragung mit $03 zwischendurch immer wieder ab. Den nächsten Datenblock sendet er erst dann, wenn der Drucker sein ACK gegeben hat. Damit die Übertragung nicht in einem Chaos entartet, muß der Sende-Computer über das Puffervermögen des Empfängers informiert sein, um rechtzeitig ein ETX (End Of Text) senden zu können. Nur so kann ein Überlauf des Puffers und der damit einhergehende Datenverlust verhindert werden.

Hat das Empfangsgerät ein ETX festgestellt, werden die empfangenen Daten bearbeitet. Kann der Empfänger neue Daten aufnehmen, sendet er ein ACK an den Computer und die Übertragung beginnt von Neuem.

— Hardware-Protokoll

Spätestens hier wird es unübersichtlich. Es hilft nur noch Tabelle 1 zur Erklärung der ganzen Signale. Als üblicher Stecker

Bit (dez. Wert)	Bedeutung
	1.) Kontrollregister (8 Bit)
Bit 7 (128)	**0** = 1 Stop-Bit **1** = 2 Stop-Bits
Bit 6 (64)	**0** } 8 Daten-Bits **0** } 7 Daten-Bits **1** } 6 Daten-Bits **1** } 5 Daten-Bits
Bit 5 (32)	**0** **1** **0** **1**
Bit 4 (16)	nicht benutzt
	Baudraten (Bit/sec)
Bit 3 (8)	**0** } nicht impl. **0** } 50 **0** } 75 **0** } 110 **0** } 134,5 **0** } 150 **0** } 300 **0** } 600 **1** } 1200 **1** } 1800 **1** } 2400
Bit 2 (4)	**0** **0** **0** **0** **0** **1** **1** **0** **1** **1** **0** **1** **1** **0** **0** **0** **1** **0** **1** **0**
Bit 1 (2)	**0** **0** **1** **0** **1** **0** **1** **0** **1** **0** **1** **0** **1** **0** **1** **0** **1** **0**
Bit 0 (1)	**0** **1** **0** **1** **0** **1** **0** **1** **0** **1** **0** **1** **0** **1** **0** **1** **0**
	2.) Kommandoregister (8 Bit)
Bit 7 (128)	**0** **0** **1** **1** keine Paritäts- **0** } ung. Par. **0** } ger. Par. **1** Bit 8:1 **1** Bit 8:0
Bit 6 (64)	**0** **1** **0** **1** überprüfung (alle **0** **1** **1** **0** ohne **1** ohne
Bit 5 (32)	**0** **0** **0** **0** vier Komb.) **1** **1** **1** Par. **1** Par.
Bit 4 (16)	**0** Vollduplex **1** Halbduplex
Bit 3 (8)	nicht benutzt
Bit 2 (4)	nicht benutzt
Bit 1 (2)	nicht benutzt
Bit 0 (1)	**0** Freilaufmodus (3-Draht) **1** Hardwarehandshake (X-Draht)

Tabelle 2. Funktion des Kontroll- und Kommandoregisters

für V.24-Signale hat sich ein 25-poliger D-Sub-Stecker (im Laborslang Cannon gennant) durchgesetzt. Die Bezeichnung der Kontakte ist gleich dreimal genormt: DIN 66020, CCITT V.24 (Comité Consultatif International Télégraphique et Téléphonique) und EIA RS232C (Electronic Industries Association). Die Bedeutung der Signale ist bei allen Normen gleich, nur die Signalpegel differieren. Die deutsche Norm verlangt, im Gegensatz zu den anderen, negative Logik.

Zum Anschluß einer RS232/V.24-Schnittstelle ist es in den allermeisten Fällen nicht nötig, alle Leitungen zu benutzen. So werden nur wenige unter Ihnen eine synchrone Datenübertragung mit zusätzlichem Clock-Signal realisieren, wie beim seriellen IEC-Bus des C 64. Normalerweise reichen die folgenden Leitungen aus:

1. eine Masseleitung
2. je Richtung eine Datenleitung
3. je Richtung eine Busyleitung

Punkt 1 und 2 dürften klar sein. Punkt 3 kann auf vielfältige Art realisiert werden. In aller Regel werden die Kontakte S2 (RTS) und M2 (CTS) benutzt. Die Erklärung erfolgt am besten an einem Beispiel: Der Drucker zeigt seine Empfangsbereitschaft an, indem er M2 aktiviert. Dieses Signal fragt der Computer ständig am Anschluß S2 ab. Ist M2 inaktiv, stoppt der Computer die Datenübertragung. Nehmen wir an, anstelle des Druckers sei ein Meßgerät angeschlossen, das nur ab und zu Anweisungen vom Computer bekommt und ansonsten sich um interne Aufgaben kümmert. Dann wäre es wenig vorteilhaft, wenn das Meßgerät ständig Befehle vom Computer erwartet; für die eigentlichen Meßaufgaben bliebe zu wenig Zeit. In diesem Fall gestattet das einfache Abfragen des Kontaktes S1 (per Interrupttechnik) eine fast ungestörte Bearbeitung eines Programmes. S1 ist mit RTS (Request To Send) identisch. Aktiviert der Computer S1, »spitzt das Meßgerät die Ohren« und das Meßprogramm verzweigt in die Datenempfangsroutine.

Nicht auf die Norm verlassen

Die Hersteller von V.24-Schnittstellen scheinen sich nicht immer völlig einig zu sein, wie die Belegung und Bedeutung der einzelnen Stecker-Pins ist. So sind diesem Beitrag hauptsächlich Praxiserfahrungen zugrunde gelegt. Besser als Normblätter ist die Überprüfung der Schnittstelle mit einem Speicheroszilloskop oder einem Digital-Analyzer. Zumal die Steuerleitungen ab und zu mit Fantasienamen belegt oder nicht eindeutig als Sende- oder Empfangsleitung gekennzeichnet werden. Zum störungsfreien Betrieb einer Schnittstelle sollten nichtbenutzte Leitungen auf ein festes Potential gelegt werden. Man verhindert dadurch, daß die Übertragung bei einer eventuellen Abfrage einer solchen Leitung, mit undefiniertem logischen Zustand, nicht unterbrochen wird.

V.24 beim C 64

Beim C 64 kann diese Schnittstelle als Gerät der Nummer 2 angesteuert werden. Zur Bestimmung der Kontrollparameter sind zwei Register vorhanden, die auch von Basic aus erreicht werden können.

Einstellparameter

Mit Tabelle 2 können Sie die Werte bestimmen, die Sie in Kommando- und Kontrollregister schreiben müssen, um ein bestimmtes Übertragungsprotokoll zu bewerkstelligen. Eine »1« bedeutet ein gesetztes Bit.

Das Einschalten der RS232-Schnittstelle geschieht beim C 64 mit OPEN filenr.,2,0,CHR$(Kontrollreg.)+CHR$(Kommandoreg.).

Beispiel: OPEN20,2,0,CHR$(64+4+2)+CHR$(32+1).

Mit dieser Anweisung wird für Filenummer 20 vereinbart: 1 Stop-Bit, 6 Datenbit, 300 Baud, ungerade Parität, Vollduplex und Hardwareprotokoll.

Pin/Bedeutung	DIN	CCITT	EIA	User-Port C 64 (VC 20)
1 Masse	E1	101	AA	A-GND (A-GND)
2 Transmit data (TD) Über diese Leitung sendet der C 64 Daten an den Akustikkoppler.	D1	103	BA	M-PA2 (M-CB2) out
3 Received data (RD) Die Empfangsleitung.	D2	104	BB	B-F12 + C-PB0 (B-CB1 + C-PB0)
4 Request to send (RTS) Frage des Computers an das Peripheriegerät, ob es zur Datenübertragung bereit ist.	S2	105	CA	D-PB1 (D-PB1) out
5 Clear to send (CTS) Positive Antwort des Peripheriegerätes auf RTS.	M2	106	CB	K-PB6 (K-PB6)
6 Data set ready (DSR) Akustikkoppler ist betriebsbereit	M1	107	CC	L-PB7 (L-PB7)
7 Signalmasse	E2	102	AB	N-GND (N-GND)
8 Received line signal (DCD)	M5	109	CF	H-PB4 (H-PB4)
9 Testzwecke				
10 Testzwecke				
11 nicht belegt				
12 Secondary carrier detector	HM5	122	SCF	—
13 Secondary clear to send	HM2	121	SCB	—
14 Secondary transmitted Data	HD1	118	SBA	—
15 Transmit clock (TC) from modem	T2	114	DB	—
16 Secondary received data	HD2	119	SBB	—
17 Receiver signal clock (RC)	T4	115	DD	—
18 nicht belegt				
19 Secondary request to send	HS2	120	SCA	—
20 Data terminal ready (DTR) Terminal ist zur Datenübertragung bereit.	S1.x	108.x	CD	E-PB2 (E-PB2) out
21 Signal quality detector	M6	110	CG	—
22 Ring indicator (RI)	M3	125	CE	F-PB3 (F-PB3)
23 Data signal rate det. terminal	S4	111	CH	—
modem	M4	112	CI	—
24 Transmit clock to modem	T1	113	DA	—
25 nicht belegt				

Tabelle 1. Belegung der RS232-Leitungen

Wird eine Filenummer größer als 128 verwendet, sendet der Computer (wie üblich) nach jedem Carriage Return ($0D) ein Linefeed ($0A). Sollten diese Einstellmöglichkeiten nicht ausreichen, können Sie direkt in die RS232-Routine eingreifen. Einige Speicherstellen finden Sie in Tabelle 3.

$0293	Kontrollregister
$0294	Kommandoregister
$0295—$0296	nicht-standard (Bit time 2/100)
$0297	Statusregister
$0298	Anzahl Bits
$0299—$029A	Baudrate
$029B	Zeiger Aufnahme
$029C	Zeiger Eingabe
$029D	Zeiger Senden
$029e	Zeiger Ausgabe

Tabelle 3. Zeropage-Adressen, die von der RS232-Schnittstelle benötigt werden.

Übertragungsraten größer als 2400 Baud, können nur mit eigenen Maschinenprogrammen realisiert werden.

Fehlerabfrage

Das Betriebssystem des C 64 verfügt über eine Fehlerbehandlung der RS232-Schnittstelle. Der Status dazu kann entweder durch die Variable ST abgefragt werden, oder direkt mit Speicherzelle $0297. Die Bedeutung der einzelnen Bits von ST finden Sie in Tabelle 4. (Jens Maßmann/hm)

Bit	dez.	Bedeutung
0	1	Paritätsfehler
1	2	Rahmenfehler
2	4	Empfängerpuffer voll
3	8	unbenutzt
4	16	CTS-Signal fehlt
5	32	unbenutzt
6	64	DSR-Signal fehlt
7	128	Break-Signal empfangen

Tabelle 4. Bedeutung der Statusvariablen