Memory Map mit Wandervorschlägen (Teil 16)

Heute beschäftigen wir uns mit Speicherzellen, die ausnahmslos etwas mit der im C 64 beziehungsweise VC 20 integrierten RS232-Schnittstelle zu tun haben.

Wir haben schon früher Speicherzellen besprochen, die mit der RS232-Schnittstelle zu tun haben, ohne die letztere dabei genauer zu betrachten. Da der heutige Teil des Kurses jedoch die zehn wichtigsten Speicherzellen behandelt, welche diese Schnittstelle betreffen, komme ich nicht umhin, näher auf sie einzugehen. Ich tue das mit voller Absicht, obwohl dadurch die erklärenden Texteinschübe mehr Platz einnehmen als die Besprechung der Speicherstellen selbst. Falls Sie mit Schnittstellen des Computers nicht vertraut sind, sollten Sie zuerst den Texteinschub »Schnittstelle und Port« anschauen, mit dem ich die gängige Begriffsverwirrung klären möchte.

Adresse 659 ($293)

RS232-Steuerregister

Jeder OPEN-Befehl, mit dem bekanntlich eine Datei (File) eröffnet wird, kann neben File-Nummer und Geräte-Nummer auch einen File-Namen haben. Der File-Namen einer RS232-Schnittstelle hat maximal nur vier Zeichen. Das erste Zeichen wird in diese Speicherzelle 659 gebracht und steuert dort Übertragungsgeschwindigkeit, die Wortlänge und die Anzahl der Stopp-Bits. Die nähere Bedeutung dieser Fachwörter können Sie dem Texteinschub »Die Elemente der RS232-Schnittstelle« entnehmen. Tabelle 1 zeigt die Bedeutung jedes einzelnen Bits dieser Speicherzelle.

Die praktische Anwendung dieser Bit-Werte innerhalb eines OPEN-Befehls ist ausführlich im Texteinschub »Die Programmierung der RS232-Schnittstelle« beschrieben.

Adresse 660 ($294)

RS232-Befehlsregister

In diese Speicherzelle wird, ähnlich wie bei der Zelle 659, das zweite Zeichen des File-Namens gebracht. Die einzelnen Bits steuern das Handshake-Protokoll (3-/X-Leitung), den Duplex-Modus (Halb-/Voll-Duplex) und die Parity-Prüfung (keine, gerade, ungerade). Die nähere Bedeutung dieser Fachwörter können Sie dem Texteinschub »Die Elemente der RS232-Schnittstelle« entnehmen. Tabelle 2 zeigt die Bedeutung jedes einzelnen Bits dieser Speicherzelle.

Wenn Sie sich für die praktische Anwendung dieser Bit-Werte innerhalb eines OPEN-Befehls interessieren, dann verweise ich auf den Texteinschub »Die Programmierung der RS232-Schnittstelle«.

Adresse 661 bis 662 ($295 bis $296)

RS232 frei wählbare Übertragungsgeschwindigkeit

Es war ursprünglich vorgesehen, durch entsprechende Wahl des dritten und vierten Zeichens im File-Namen beliebige Übertragungsgeschwindigkeiten einzustellen. Die jeweiligen Werte sollten die Speicherzellen 661 und 662 enthalten. Diese Möglichkeit wurde aber nicht eingebaut. Der Grund dafür dürfte wohl der sein, daß die wählbaren Übertragungsgeschwindigkeiten aller Geräte auf bestimmte Werte normiert sind.

Adresse 663 ($297)

RS232-Statusregister

Genauso wie in der Speicherzelle 144 der Status aller Ein- und Ausgabe-Operationen angezeigt wird, werden alle Fehler der RS232-Schnittstelle in der Speicherzelle 663 angezeigt. Die Bedeutung der einzelnen Bits, wenn sie auf 1 gesetzt sind, zeigt Tabelle 3.

Der Status wird nicht automatisch angezeigt, sondern muß vom Programm abgefragt werden. Abfragen können Sie sowohl durch PEEKen der Speicherzelle 663 als auch durch Aufrufen der Statusvariablen ST. Die Variable ST, die normaler-

BINÄR	BITWERT	STEUERFUNKTION
Bit 0 bis 3 steuern die Übertragungsgeschwindigkeit		
xxxx0000	0	(siehe 1)
xxxx0001	1	50 bit/s
xxxx0010	2	75 bit/s
xxxx0011	3	110 bit/s
xxxx0100	4	134,5 bit/s
xxxx0101	5	150 bit/s
xxxx0110	6	300 bit/s
xxxx0111	7	600 bit/s
xxxx1000	8	1200 bit/s
xxxx1001	9	1800 bit/s
xxxx1010	10	2400 bit/s
die Werte 11 bis 15 sind nicht belegt		
Bit 4 ist nicht belegt		
Bit 5 und 6 steuern die Wortlänge		
x00xxxxx	0	8 Bit-Wort
x01xxxxx	32	7 Bit-Wort
x10xxxxx	64	6 Bit-Wort
x11xxxxx	96	5 Bit-Wort
Bit 7 steuert die Stopp-Bits		
0xxxxxxx	0	1 Stopp-Bit
1xxxxxxx	128	2 Stopp-Bit

Tabelle 1. Die Bedeutung der einzelnen Bit im RS232-Steuerregister

BINÄR	BITWERT	STEUERFUNKTION
Bit 0 steuert den Handshake-Typ		
xxxxxxx0	0	3-Leitung
xxxxxxx1	1	X-Leitung
Bit 1 bis 3 sind nicht belegt		
Bit 4 steuert den Duplex-Modus		
xxx0xxxx	0	Voll-Duplex
xxx1xxxx	16	Halb-Duplex
Bit 5 bis 7 steuern den Paritäts-Test		
000xxxxx	0	keine Parität
001xxxxx	32	ungerade (1)
010xxxxx	64	keine Parität
011xxxxx	96	gerade (1)
100xxxxx	128	keine Parität
101xxxxx	160	Parität = 1 (2)
110xxxxx	192	keine Parität
111xxxxx	224	Parität = 0 (2)

Tabelle 2. Das RS232-Befehlsregister im einzelnen

BIT	BIT-WERT	BEDEUTUNG
0	1	Fehler bei Paritäts-Test
1	2	Fehler in der Bit-Folge
2	4	Überlauf des Eingabepufferspeichers
3	8	Eingabepufferspeicher ist leer
4	16	das Clear-To-Send-(CTS-)Signal (Handshake) fehlt
5	32	nicht belegt
6	64	das Data-Set-Ready-(DSR-)Signal (Handshake) fehlt
7	128	Übertragung ist unterbrochen

Tabelle 3. Das RS232-Status-Register

weise den Inhalt der Zelle 144 wiedergibt, schaltet nach dem Eröffnen eines RS232-Kanals durch OPEN 1,2 auf die Speicherzelle 663 um. Jedoch ist Vorsicht geboten, da durch Aufruf von ST der Inhalt von 663 gelöscht wird. Es ist ratsam, den Wert von ST erst einer anderen Variablen zuzuordnen, wenn sie mehrfach verwendet werden soll. Falls das Status-Register einen Fehler anzeigt, muß das Programm entsprechende Konsequenzen ziehen. Wenn zum Beispiel Bit 0 oder Bit 1 gesetzt sind, ist es angebracht, das letzte Daten-Byte noch einmal zu übertragen. Wenn Bit 2 gesetzt ist, heißt dies, daß der GET#-Befehl den Eingabepufferspeicher nicht schnell genug entleert. Falls die Übertragungsgeschwindigkeit von 300 bit/s, die maximal mit einem Basic-Programm erreichbar ist, nicht ausreicht, muß entweder der Sender langsamer eingestellt werden, oder Sie schreiben das Programm in Maschinensprache.

Adresse 664 ($298)

Anzahl der zu übertragenden Bits

Diese Speicherzelle wird verwendet, um festzustellen, mit wievielen Nullen das zu übertragende Zeichen aufgefüllt werden muß, um die in Speicherzelle 659 (Bit 5 und 6) ausgewählte Wortlänge herzustellen (siehe auch Speicherzellen 168 und 180).

Adresse 665 bis 666 ($299 bis $29A)

Zeit, die zum Übertragen eines Bit gebraucht wird

Sobald ein RS232-Kanal eröffnet worden ist, berechnet das Betriebssystem einen Wert, der die Zeitdauer eines Bits festlegt. Da die Übertragungsrate in Speicherzelle 659 einstellbar ist, hängt diese Bit-Dauer von der gewählten Übertragungsgeschwindigkeit ab. Die Bit-Dauer errechnet sich aus der Systemfrequenz (985,25 KHz) geteilt durch die Übertragungsgeschwindigkeit. Dieser Wert steht in Low/High-Byte-Darstellung in diesen beiden Speicherzellen, von wo aus er vom Betriebssytem abgerufen wird.

Adresse 667 ($29B)

Index auf das Ende des Eingabepufferspeichers

Dieser Index wird verwendet, um Daten in den Eingabepufferspeicher zu schreiben. Wenn man ihn nämlich zum Inhalt der Speicherzelle 247/248 addiert, erhält man die Adresse des zuletzt in den Eingabepufferspeicher eingegebenen Bytes.

Adresse 668 ($29C)

Index auf den Anfang des Eingabepufferspeichers

Dieser Index wird verwendet, um Daten aus dem Eingabepufferspeicher auszulesen. Wenn man ihn nämlich zum Inhalt der Speicherzelle 247/248 addiert, erhält man die Adresse des ersten in den Eingabepufferspeicher eingegebenen Bytes.

Adresse 669 ($29D)

Index auf den Anfang des Ausgabepufferspeichers

Dieser Index wird verwendet, um Daten aus dem Ausgabepufferspeicher auszulesen. Wenn man ihn nämlich zum Inhalt der Speicherzelle 249/250 addiert, erhält man die Adresse des ersten in den Ausgabepufferspeicher eingegebenen Bytes.

Adresse 670 ($29E)

Index auf das Ende des Ausgabepufferspeichers

Dieser Index wird verwendet, um Daten in den Ausgabepufferspeicher zu schreiben. Wenn man ihn nämlich zum Inhalt der Speicherzelle 249/250 addiert, erhält man die Adresse des zuletzt in den Ausgabepufferspeicher eingegebenen Bytes.

Adresse 671 bis 672 ($29F bis $2A0)

Zwischenspeicher für den IRQ-Vektor während Kassetten-Ein/Ausgabe

Die Routinen des Betriebssystems, die Daten auf, beziehungsweise von Kassette ein- und ausgeben, werden durch die Interrupt-Routine gesteuert. Diese Routine unterbricht normalerweise 60mal in der Sekunde alle Aktivitäten des Computers, um diverse »Hausaufgaben« (Uhr weiterschalten, STOP-Taste abfragen und so weiter) auszuführen. Bei Kassetten-Ein-/Ausgaben ist diese Interrupt-Routine jedoch abgeschaltet. Dies wird dadurch erreicht, daß der Vektor in Speicherzelle 788/789, der auf die Anfangsadresse der Interrupt-Routine zeigt, auf eine Adresse der Kassetten-Routine gesetzt wird. Um nach der Kassettenoperation weitermachen zu können, wird der »alte« Interrupt-Vektor in dieser Speicherzelle 671/672 gespeichert.

Adresse 673 ($2A1)

bei C 64: Flagge für RS232-Interrupt
bei VC 20: frei verfügbar

Diese Speicherzelle enthält den Wert des Interrupt-Steuerregisters 56589, das die RS232-Schnittstelle steuert. Die Bedeutung der einzelnen Bits, wenn sie auf 1 gesetzt sind, zeigt Tabelle 4. Diese Flagge kann zu Steuerzwecken abgefragt werden. Um beispielsweise ein Programm warten zu lassen, bis der Ausgabepufferspeicher geleert ist, gibt man die Anweisung

```
100 IF (PEEK(673) AND 1) THEN 100
```

die das Programm so lange aufhält, bis die Übertragung abgeschlossen und Bit 0 der Flagge gelöscht ist. Damit sind alle Speicherzellen, welche die RS232-Schnittstelle steuern, behandelt. Das nächste Mal kommen wir zuerst zu einem großen, frei benutzbaren Speicherbereich, danach zu einer Reihe von Sprung-Vektoren. (Dr.H.Hauck/ah)

Bit 0	(Bitwert 1)	Daten werden gesendet
Bit 1	(Bitwert 2)	Daten werden empfangen
Bit 4	(Bitwert 16)	Schnittstelle wartet auf Daten vom Sender

Tabelle 4. Flagge für den RS232-Interrupt

SCHNITT-STELLE	Erweit ①	Seriell ④	Kasset ⑤	User ⑥	Spiel ⑧	PORT
RS232				X		
IEC/IEEE	X	X				
Centronics		X		X		

Tabelle 5. Realisierbarkeit der IEC-, IEEE- und Centronics-Schnittstelle am C 64 und VC 20

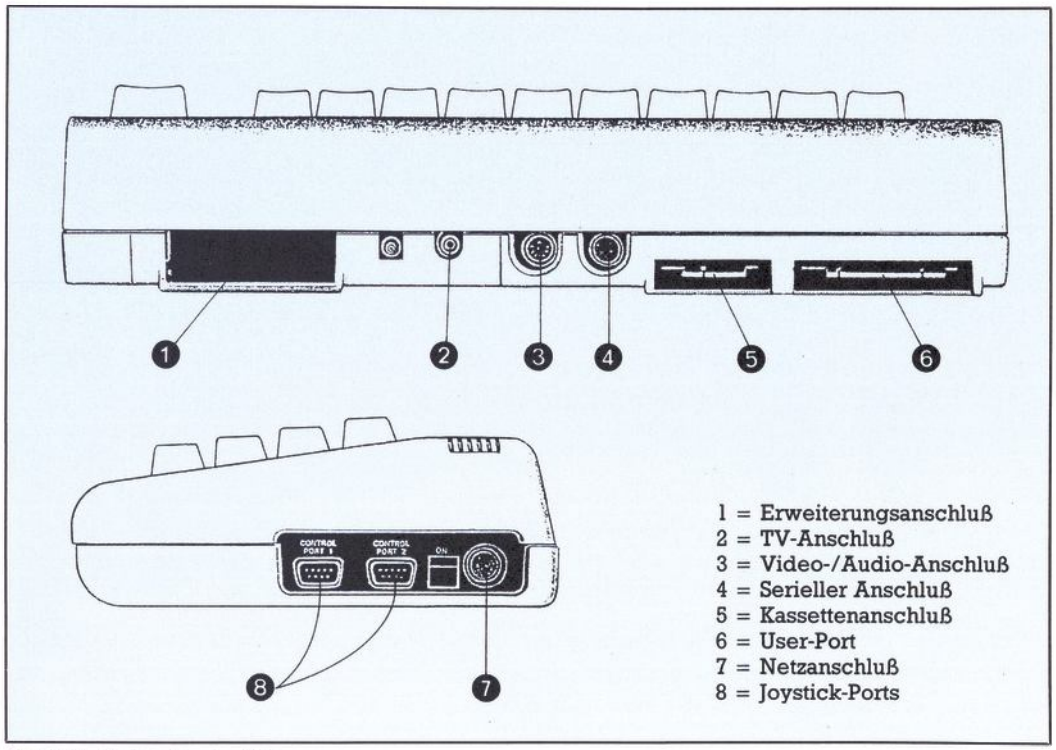

Bild 1. Die Schnittstellen des C 64

Texteinschub #1
Schnittstelle und Port

Immer wenn die Rede davon ist, den Computer mit irgendwelchen Geräten zu verbinden, tauchen Fachwörter auf, wie Interface, Schnittstelle, Port, Eingang, Ausgang und Stecker. Da im Kurs gerade die Speicherzellen behandelt werden, die für die RS232-Schnittstelle zuständig sind, möchte ich die Gelegenheit nutzen, ein wenig Klarheit in dieses Begriffswirrwar zu bringen. Zuerst sollen die Begriffe erklärt werden:

Interface: ist das englische Wort für *Schnittstelle*.

In einer Schnittstelle sind die Regeln und Vorschriften festgelegt, wie Daten zwischen zwei Geräten (zum Beispiel Computer und Floppy) ausgetauscht werden. Festgelegt ist hauptsächlich:

— ob ein Datenwort auf einen Schlag (parallel) oder jedes Bit einzeln (seriell) übertragen wird
— die Geschwindigkeit der Übertragung
— die Signale, mit denen die beteiligten Geräte den Ablauf der Übertragung steuern
— mit welchen Spannungs- oder Stromwerten die binäre 1 beziehungsweise 0 dargestellt wird
— die elektrischen Spannungen und Ströme, die bei der Übertragung maximal auftreten dürfen

Sie sehen, eine *Schnittstelle* ist in erster Linie eine Anzahl von Regeln. Manchmal allerdings werden auch die Module und Spezialkabel, welche die Regeln technisch in die Tat umsetzen, Schnittstellen genannt.

Über einen *»Ausgang«* kann der Computer (oder ein anderes Gerät) Daten abgeben, über einen *»Eingang«* erhält er Daten. Ein *»Port«* ist beides, Ein- und Ausgang. Ein *»Stecker«* schließlich ist die technische Ausführung der Verbindung.

So, nach dieser Begriffserklärung wollen wir uns anschauen, welche Ports, Ein- und Ausgänge, der Computer hat. Die Zeichnung dieser Anschlüsse (Bild 1) habe ich dem Commodore-Handbuch entnommen, nicht aber ihre Bezeichnungen, denn diese gehen bereits wild durcheinander.

Der »Erweiterungsanschluß« (1) wird hauptsächlich als Eingang für Spielmodule verwendet. Er ist aber ein echter Port, nicht zuletzt zur Speichererweiterung beim VC 20 und kann für Schnittstellen über entsprechende Routinen des Betriebssystems programmiert werden.

Der »TV-Anschluß« (2) ist ein reiner Ausgang des im Computer eingebauten Fernsehmodulators, der beim VC 20 fehlt.

Der »Video/Audio-Anschluß« (3) ist ebenfalls ein reiner Ausgang der Ton- und Bildsignale für einen Monitor oder für den externen Fernsehmodulator des VC 20.

Der »Serielle Anschluß« (4) ist ein Port, über den das Diskettenlaufwerk und Drucker angeschlossen werden. Er ist für Schnittstellen programmierbar.

Der »Kassettenanschluß« (5) ist ebenfalls ein Port, der speziell für die Datasette eingerichtet ist. Bastler und Tüftler, die Schaltpläne lesen können und das Betriebssystem des Computers kennen, müßten in der Lage sein, mit diesem Port auch andere externe Geräte zu steuern. Für die genormten Schnittstellen kommt er meines Wissens nicht in Frage.

Der »User-Port« (6) ist das, was sein Name sagt, nämlich ein Port für verschiedene Anwendungen und Schnittstellen. Er ist über 16 Register des 6526 Complex Interface-Adapters (CIA) mit den Adressen 56320 bis 563215 frei programmierbar. Der VC 20 hat dafür einen 6522 Versatile Interface-Adapter (VIA), dessen 16 Register die Adressen 37136 bis 37151 haben.

Der »Netzanschluß« (7) ist ein reiner Eingang, aber nicht für Daten.

Die »Spielanschlüsse« (8) (nur einer beim VC 20) werden eigentlich nur als Eingang für Joysticks, Lichtgriffel und Paddles (Drehregler) verwendet, obwohl sie vom Prinzip her programmierbare Ports sind. Ihre universelle Verwendung ist sicher nur Spezialisten vorbehalten. Zuletzt sollen auch die verbreitetsten Schnittstellen noch erwähnt werden. International haben sich besonders die folgenden drei Schnittstellen durchgesetzt:

— die RS232-Schnittstelle
— die IEC/IEEE-488-Schnittstelle
— die Centronics-Schnittstelle

Die »RS232-Schnittstelle« ist eine serielle Schnittstelle. Sie wurde schon vor 100 Jahren als »TTY-Version« zur Textübertragung mit Fernschreibern eingerichtet, bei der die logische 0 durch einen Strom von 20 Milliampere und die 1 durch keinen Strom dargestellt wurde. Heute wird fast nur noch die »V.24-Version« zur Datenfernübertragung verwendet, bei der die 0 durch eine positive Spannung zwischen 3 und 15 Volt, die 1 aber durch eine entsprechende negative Spannung dargestellt wird. Beim C 64 und VC 20 ist die RS232-/V.24-Schnittstelle am User-Port verfügbar, allerdings nicht mit den oben genannten Spannungswerten für die 0 und 1. Dieses für Commodore typische Sparverfahren macht eine zusätzliche Signalumsetzung erforderlich. Die RS232-Schnittstelle wird hauptsächlich für Datenübertragung per Modem oder Akustikkoppler eingesetzt. Ihr Arbeitsprinzip ist im Texteinschub »Die Elemente der RS232-Schnittstelle« beschrieben.

Zur parallelen Datenübertragung entstand in Europa die »IEC-625-Schnittstelle«, in USA die »IEEE-488-Schnittstelle«. Beide sind praktisch identisch. Sie unterscheiden sich nur in der Verwendung des Steckers (was natürlich idiotisch ist). Bei den Commodore-Computern ist diese Schnittstelle sowohl am Erweiterungs-Port (1) als auch über den seriellen Port (4) einrichtbar. Der serielle Port allerdings enthält wiederum eine für Commodore typische Einschränkung. Er erlaubt, wie sein Name andeutet, nur eine serielle Datenübertragung. Das heißt, statt — wie bei der IEC-/IEEE-Schnittstelle festgelegt — alle 8 Bits eines Wortes über acht Leitungen gleichzeitig, werden hier die Bits hintereinander auf nur einer Leitung übertragen. Auch das erfordert eine zusätzliche Anpassung. Das Prinzip der IEC-/IEEE-Schnittstelle wurde bereits im Ausgabe 3/85 ab Seite 24 von Arnd Wängler genau beschrieben.

Die »Centronics-Schnittstelle« ist aus der harten Realität des Geschäftslebens entstanden. Während sich noch die Normstellen in Europa und USA herumstritten, hat der damalige Marktführer unter den Druckerherstellern, die Firma Centronics, eine eigene Schnittstelle geschaffen, die sich schlicht und einfach durch die weite Verbreitung der Centronics-Drucker durchgesetzt hat. Sie ist eine parallele Schnittstelle, die sich beim C 64/VC 20 sowohl am User-Port (6) als auch am seriellen Port (4) einrichten läßt. Zur Beschreibung der Schnittstelle habe ich in meiner Literatursammlung nur zwei Aufsätze gefunden, die eine von Georg Werner in c't, Ausgabe 4/84, Seite 92, die andere von Peter Bonsch in Computer persönlich, Ausgabe 11/83 ab Seite 152. Tabelle 5 gibt eine Zusammenfassung über die Realisierbarkeit der drei Schnittstellen an den Ports von C 64 und VC 20.

Texteinschub #2
Die Elemente der RS232-Schnittstelle

Da meine Texteinschübe kurz sein sollen, beschränke ich mich auf Erklärungen der Vorgänge, die mit den im Kurs behandelten Speicherzellen 659 bis 670 zu tun haben. Weitere Erläuterungen können Sie dem Aufsatz von Jens Maßmann der Ausgabe 5/85, Seite 80 entnehmen.

Wortlänge: Die Schnittstelle ermöglicht die Übertragung von Datenwörtern (Bytes), deren Länge vor der Übertragung eingestellt werden kann. Es sind Wortlängen von 5, 6, 7 oder 8 Bit erlaubt. Die Wortlänge wird durch Bit 5 und 6 der Speicherzelle 659 eingestellt.

Übertragungsgeschwindigkeit: Daten werden seriell übertragen, das heißt alle Bits eines Datenwortes (Byte) laufen hintereinander über eine Leitung zum Empfänger. Dabei ist wesentlich, daß Sender und Empfänger sich einig sind, mit welcher Geschwindigkeit die Bit-Kette übertragen wird. Diese Übertragungsgeschwindigkeit wird in Bit pro Sekunde angegeben. Die Übertragungsgeschwindigkeit wird durch Bit 0 bis 3 der Speicherzelle 659 eingestellt und reicht von 110 bis 2400 bit/s.

Stopp-Bits: Die Übertragungsgeschwindigkeit muß sowohl im Sender als auch im Empfänger der Daten fest eingestellt werden. Da die beiden Geräte dies unabhängig voneinander tun, besteht die Gefahr, daß diese Einstellungen nicht ganz genau gleich sind. Das könnte zur Folge haben, daß sie nach vielleicht 2000 Bit um 1 Bit auseinanderliegen. Alle nachfolgenden Übertragungen wären dann völlig falsch.

Deshalb wird die Übertragung nach jedem Wort neu eingestellt, man nennt das »synchronisieren«. Dazu dient am Anfang eines Wortes ein Start-Bit und am Ende eines Wortes ein oder zwei Stopp-Bit. Die Stopp-Bits definieren den Ruhezustand (logische 1) der Übertragungsleitung, ihre Anzahl die minimale Zeit des Ruhezustandes. Sobald ein neues Wort mit einem Start-Bit (logi-

sche 0) beginnt, übernimmt der Empfänger den Übergang von 0 nach 1 als Startimpuls für die Abfrage der nächsten ankommenden Bits. Die Anzahl der Stopp-Bits werden durch Bit 7 der Speicherzelle 659 eingestellt.

Paritätsprüfung: Zusätzlich zu den Datenbits und den Start/Stopp-Bits können sogenannte Paritätsbit übertragen werden. Sie ermöglichen eine grobe Fehlerkontrolle. Der Sender errechnet die Quersumme aller Datenbits. Bei der sogenannten »geraden« Paritätsprüfung wird das Paritätsbit so gewählt, daß es die Quersumme zu einer geraden Zahl ergänzt, bei der »ungeraden« Paritätsprüfung ist es gerade umgekehrt. Der Empfänger macht dieselbe Rechnung und vergleicht sein Paritätsresultat mit dem empfangenen Paritätsbit des Senders. Sie sollten natürlich gleich sein. Auf diese Weise können einfache Übertragungsfehler erkannt werden. In den Commodore-Computern sind noch zwei weitere Möglichkeiten eingebaut, nämlich das Paritätsbit ohne Quersummenrechnung immer auf 1 oder aber immer auf 0 zu setzen. Mit den Bits 5 bis 7 der Speicherzelle 660 können insgesamt vier verschiedene Paritätsprüfungen eingestellt werden.

Duplex-Modus: Sind zwei Geräte, von denen eines nur empfangen, nicht aber selbst senden kann, über eine Leitung verbunden, nennt man diese Einbahnstraße eine Simplex-Verbindung. Können aber beide Geräte senden und empfangen, spricht man von einer Duplex-Verbindung. Duplex gibt es in zwei Betriebsarten. Der Voll-Duplex-Modus erlaubt ein gleichzeitiges Senden beider Geräte. Im Halb-Duplex-Modus kann immer nur ein Gerät senden, allerdings wechselweise. Der Duplex-Modus wird durch Bit 4 der Speicherzelle 660 eingestellt.

Handshake-Protokoll: Mit Handshake (Händeschütteln) wird ein Verfahren bezeichnet, bei dem zwei Geräte sich gegenseitig durch gesonderte Signale mitteilen, ob sie bereit sind, Daten abzusenden beziehungsweise zu empfangen. Die Festlegung der zeitlichen Reihenfolge dieser Handshake-Signale nennt man Protokoll. Dieses Verfahren hat den Vorteil, daß Sender und Empfänger völlig unabhängig voneinander ihr eigenes Programm ausführen können und nur selten aufeinander warten müssen. Voraussetzung ist allerdings ein Pufferspeicher (siehe unten). Es gibt zwei Arten von Handshakes, den 3-Leitungs-Handshake (auch Rückkanal-Handshake genannt) und den X-Leitungs- oder Voll-Handshake. Der 3-Leitungs-Handshake braucht, wie der Name sagt, nur drei Leitungen: für gemeinsame Erde (Masse), für die gesendeten Daten und für die empfangenen Daten. Der Handshake besteht darin, daß der Empfänger auf der freien Leitung, eben dem Rückkanal, dem Sender durch je ein Zeichen mitteilt, wenn er bereit ist, Daten zu übernehmen oder wenn er keine Daten übernehmen kann. Der X-Leitungs-Handshake stellt viel mehr Leitungen zur Verfügung, nämlich die gleichen drei wie vorher, zusätzlich aber pro Sender und Empfänger mehrere Leitungen für Anmeldung und Rückmeldung der Bereitschaft, sowie für die Ausführung der Übertragung. Es gibt theoretisch insgesamt 25 Leitungen für die RS232-Schnittstelle, beim C 64 beziehungsweise VC 20 sind aber nur zehn ausgeführt. Das Handshake-Protokoll kann durch Bit 0 der Speicherzelle 660 ausgewählt werden.

Pufferspeicher: Immer wenn ein RS232-Kanal geöffnet wird, zwackt das Betriebssystem des Computers dem Programmspeicher am oberen Ende zwei Pufferspeicher ab, mit einer Größe von je 256 Byte für empfangene und zu sendende Daten. Diese First-In-First-Out-Speicher (die als erste eingespeicherten Daten werden auch als erste wieder ausgelesen) sind als dynamische Ringspeicher aufgebaut. Statt zu warten, bis der Empfänger zur Datenübernahme bereit ist, schreibt der Sender die Daten in den Pufferspeicher, aus dem die Schnittstelle sie an den Empfänger weitergibt, sobald dieser bereit ist. Dieses fast ungeordnete Füllen und Leeren des Pufferspeichers hat zur Folge, daß Beginn und Ende des Speichers je nach Datenmenge innerhalb der 256 Byte stets in Bewegung sind. Um jederzeit die Anfangs- und Endadressen feststellen zu können, werden sie in den Speicherzellen 667 bis 670 mitgezählt.

Statusregister: In der Speicherzelle 663 werden alle Fehler einer RS232-Übertragung festgehalten. Jedes Bit hat eine eigene Bedeutung, die in der Tabelle bei der Beschreibung der Speicherzelle 663 angegeben ist. Diese Fehler werden leider nicht, wie im Basic, automatisch angezeigt. Sie müssen vielmehr ausgelesen und identifiziert werden, um dann im Programm mit entsprechenden Maßnahmen korrigiert zu werden.

Texteinschub #3
Die Programmierung der RS232-Schnittstelle

Die Programmierung der RS232-Schnittstelle ist denkbar einfach. Alle dazu notwendigen Routinen sind im Betriebssystem des Computers bereits enthalten. Das genau macht ja die Schnittstelle so attraktiv. Die Schnittstelle verwendet genau dieselben Befehle wie die serielle Schnittstelle, über die der Computer mit Floppy und Drucker verbunden ist, nämlich OPEN, CMD, PRINT#, GET#, INPUT# und CLOSE. Auch die Statusvariable ST wird herangezogen. Wichtig ist jedoch, daß die RS232-Schnittstelle die Gerätenummer 2 hat.

Eröffnung des RS232-Kanals

Wie gewohnt, wird er mit dem OPEN-Befehl geöffnet:
OPEN File-Nr, Geräte-Nr, Datenkanal-Nr, File-Name
— die File-Nummer kann Werte von 0 bis 255 annehmen, wie bei jedem OPEN-Befehl auch
— die Geräte-Nummer ist immer 2
— der Wert der Datenkanal-Nummer ist bedeutungslos, da immer nur ein RS232-Kanal offen sein darf. Wird zusätzlich ein zweiter Kanal geöffnet, werden die Daten des ersten Kanals im Pufferspeicher zerstört.
— der File-Name hat hier eine besondere und entscheidende Bedeutung. Er besteht aus maximal vier Zeichen. Der ASCII-Wert des ersten Zeichens wird in die Speicherzelle 659 übertragen und legt dadurch die Übertragungsgeschwindigkeit, die Wortlänge und die Anzahl der Stopp-Bits fest (siehe Texteinschub Nr.2). Der ASCII-Wert des zweiten Zeichens gelangt in die Speicherzelle 660 und bestimmt dadurch das Handshake-Protokoll, den Duplex-Modus und die Paritätsprüfung (siehe Texteinschub Nr.2). Zeichen 3 und 4 sind nicht festgelegt. Man kann den File-Namen des OPEN-Befehls in zwei Arten schreiben, die natürlich identisch sind:

```
(1) OPEN 1,2,0,CHR$(7+64+128) + CHR$(1+16+32)
(2) OPEN 1,2,0,CHR$(199)+CHR$(49)
```

Theoretisch könnte man noch eine dritte Schreibweise hernehmen, nämlich die Zeichen hinschreiben, die den ASCII-Wert 199 beziehungsweise 49 haben. Dann käme die Schreibweise einem File-Namen noch am nächsten. Ein Blick in die Tabelle der ASCII-Codes belehrt uns aber eines Besseren, da wir Zweideutigkeiten nicht ausschließen können. Also ist die Schreibweise der Zeichen mit ihren ASCII-Werten doch am besten. Ich persönlich ziehe die Schreibweise (1) vor, da wir aus ihr sofort die dadurch definierten Werte ablesen können. Das erste der beiden Zeichen »CHR$(7+64+128)« bedeutet:
Datenrate = 600 bit/s
Wortlänge = 6 Bit
Stopp-Bit = 2

Sie können die Zusammenhänge direkt der Tabelle entnehmen, die bei der Erklärung der Speicherzelle 659 steht.

Entsprechend wird aus der Tabelle der Speicherzelle 660 das zweite Zeichen »CHR$(1+16+32)« zusammengesetzt:
Handshake = X-Leitung
Duplex = Halb-Duplex
Parität = Ungerade

Der OPEN-Befehl mit Gerätenummer 2 hat noch eine Besonderheit, die ich schon bei der Besprechung der Speicherzellen 55/56 erwähnt habe. Sobald er nämlich im Programm auftaucht, wird durch ihn der Zeiger in 55/56, der ja das obere Ende des Programmspeichers angibt, um 512 Byte nach unten geschoben, um Platz für die beiden Pufferspeicher zu schaffen. Wenn das mitten im Programm passsiert und vorher schon Zeichenketten (Strings) definiert worden sind (die bekanntlich vom oberen Ende des Speichers aus angelegt werden), werden diese überschrieben. Also Vorsicht: Wer beabsichtigt, in einem Programm eine RS232-Schnittstelle zu aktivieren, soll diese unbedingt am Anfang des Programms öffnen, damit der Speicherplatz richtig zugeordnet wird.

Daten an den RS232-Kanal übergeben

Die Daten werden zuerst in den Ausgabepuffer gebracht, von dort gelangen sie, vom Handshake-Protokoll gesteuert, an den Empfänger. Die Befehle dazu sind CMD und PRINT#.

»CMD File-Nr.,Zeichen« schaltet vom Bildschirm auf den RS232-Empfänger um. »PRINT# File-Nr, Zeichen« schreibt die Zeichen in den Ausgabepufferspeicher, von wo sie die Schnittstelle automatisch herausholt. Beide Befehle wirken genauso, wie bei anderen Dateien. Vorsicht ist jedoch geboten, wenn laufend Daten in den Pufferspeicher geschrieben werden, ohne zu wissen, ob die Schnittstelle den Puffer auch wieder entleert hat. Bei Überlauf des Puffers gehen Daten verloren. Es ist ratsam, durch Vergleich der beiden Indizes in den Speicherzellen 669 und 670, die Anfang und Ende des Ausgabepufferspeichers markieren, auf Überlauf zu prüfen.

Daten vom RS232-Kanal übernehmen

Daten, die von der Schnittstelle in den Eingabepufferspeicher gebracht worden sind, werden mit INPUT# oder GET# ausgelesen:
INPUT# File-Nr, Zeichen
GET# File-Nr, Zeichen

Auch hier kann ein Überlaufen des Pufferspeichers auftreten, wenn nämlich die Schnittstelle mehr oder schneller Daten liefert, als mit GET# oder INPUT# ausgelesen werden können. Dieser Zustand kann sowohl durch Überprüfung der Indizes in den Speicherzelle 667 und 668 als auch durch Prüfung von Bit 2 des Statusregisters in 663 erkannt beziehungsweise vermieden werden. Der Speicher kann auch leer sein. Bei Verwendung von INPUT# wartet der Rechner und stürzt bei abgeschalteter Schnittstelle ab. Es ist deshalb empfehlenswert, immer den Befehl GET# zu verwenden, der bei leerem Speicher höchstens einen Nullstring (" ") liefert. Bit 3 des Statusregisters prüft diesen Fall.

Schließen des RS232-Kanals

Der Befehl »CLOSE File-Nr.« schließt den Kanal. Dabei werden die Ein- und Ausgabe-Pufferspeicher aufgelöst, indem der Zeiger in Speicherzelle 55/56 wieder auf das Ende des Programmspeichers zeigt. Alle Handshake-Leitungen werden in den Ruhezustand gesetzt und alle Datenübertragungen unterbunden.

Programm-Beispiel

Für eine echte Demonstration müßten Sie eine RS232-Schnittstelle über den User-Port eingerichtet haben. Da ich das nicht voraussetzen kann, begnüge ich mich damit, das im Programmierhandbuch von Commodore angegebene Beispiel zu bringen und zu kommentieren.

```
10 OPEN 1,2,0,CHR$(6+32)+CHR$(32+128)
20 GET#1,A$
30 GET B$
40 IF B$ " " THEN PRINT#1,B$;:PRINT B$
50 GET#1,C$
60 PRINT C$;
70 PR = PEEK(663)
80 IF PR=0 OR PR=8 THEN 30
100 IF PR AND 1 THEN PRINT "PARITY-FEHLER"
110 IF PR AND 2 THEN PRINT "BITFOLGE-FEHLER"
120 IF PR AND 4 THEN PRINT "EINGABESPEICHER VOLL"
130 IF PR AND 128 THEN PRINT "UNTERBRECHUNG"
```

Zeile 10 öffnet den RS232-Kanal mit 1 Stopp-Bit, 300 bit/s und 7 Bit Wortlänge, außerdem über 3-Leitungs-Handshake, Voll-Duplex und ohne Parität.

Zeile 20 will den Eingabespeicher auslesen, der aber noch leer ist. Der resultierende Nullstring interessiert uns nicht, aber die Schnittstelle signalisiert über Handshake, daß wir bereit sind, Daten zu übernehmen.

Zeile 30 fragt inzwischen die Tastatur ab. Wenn eine Taste gedrückt worden ist, schiebt Zeile 40 das Zeichen in den Ausgabespeicher und druckt es nochmal auf dem Bildschirm aus.

Zeile 50 liest wieder den Eingabespeicher aus. Falls inzwischen Daten über die Schnittstelle gekommen sind, druckt Zeile 60 das erste Zeichen auf den Bildschirm.

Zeile 70 ordnet der Variablen PR(üfung) den Inhalt des Statusregisters 663 zu. Ist kein Fehler aufgetreten (PR = 0) oder ist der Eingabespeicher immer noch leer (PR = 8), dann springt Zeile 80 zurück zur Tastaturabfrage, und der Zyklus läuft weiter. Ist aber ein Fehler aufgetreten, wird dieser ab Zeile 100 geprüft und ausgedruckt.