KLEINCOMPUTER

KC 85/5

System - Handbuch

KLEINCOMPUTER

KC 85/4
KC 85/5
KC 85/5+

System - Handbuch

CAOS 4.6 bis 4.8

veb mikroelektronik
'wilhelm pieck'
mühlhausen

KC-CLUB
Mario Leubner

Impressum der Originalausgabe für den KC 85/4:

Gesamtherstellung: Druckerei August Bebel Gotha
Ri 1/89 WV/6/1-10 84297

veb mikroelektronik „wilhelm pieck“ mühlhausen

Der Vertrieb dieser Druckschrift erfolgt ausschließlich durch den Herausgeber. Nachfragen bei der Druckerei sind zwecklos.

Hinweise, die zur Verbesserung dieser Dokumentation führen, werden gern entgegengenommen.

Redaktionsschluss: Juli 1988

Überarbeitet und erweitert für den KC 85/5 von Mario Leubner,

Dank für die Digitalisierung des KC 85/4-Handbuchs und für das Kontrolllesen des Manuskripts an Elmar Klinder, Götz Hupe und Ralf Kästner.
Dank an René Nitzsche für das Kontrolllesen des KC 85/5-Handbuchs und für die Umsetzung der Idee, einen dünnen Zeichensatz für den KC 85 zu integrieren, sowie die Erstellung der zugehörigen Abbildungen für das Handbuch.

Redaktionsschluss

Ausgabe für CAOS 2.2:	September 1985
Ausgabe für CAOS 3.1:	1987
Ausgabe für CAOS 4.2:	Juli 1988
Ausgabe für CAOS 4.4:	April 2010 / August 2010
Ausgabe für CAOS 4.5:	April 2015
Ausgabe für CAOS 4.6:	Dezember 2015 (nicht veröffentlicht)
Ausgabe für CAOS 4.6-4.8:	April 2023

Bitte Korrekturen oder Ergänzungen dem Autor mitteilen an: maleuma@web.de

Für die Erstellung dieses Handbuchs wurde unter anderem folgende Software benutzt:

- *LibreOffice für das Gesamtlayout und die Texte*
- *paint.net für die Zeichenbilder*
- *sPLAN für Grafiken*
- *JKCEMU für die Screenshots*
- *PapDesigner für die Programmablaufpläne*

Inhaltsverzeichnis

Inhaltsverzeichnis

Inhaltsverzeichnis

Inhaltsverzeichnis

EINLEITUNG

Das vorliegende Handbuch ist die überarbeitete und stark erweiterte Fassung des Systemhandbuchs vom KC 85/4. Der Kleincomputer **KC 85/4** ist ein vielseitig einsetzbarer Kleinrechner aus der KC 85-Reihe des VEB Mikroelektronik „Wilhelm Pieck“ Mühlhausen und wurde vom KC-Club zum **KC 85/5** weiterentwickelt. Da diese Hardwareerweiterung bereits auf der Hauptplatine des KC 85/4 vorbereitet war, beschränkt sich der Umbau auf den Austausch weniger Bauteile:

Bauteil	Originaltyp	Austauschtyp	Funktion
D15-D22	64k dRAM U2164	256k dRAM 41256	RAM0, RAM4, RAM8
D10	8k EPROM U2764	8k EPROM 27(C)64	CAOS-ROM-E
D11	8k ROM U2364D BM600 (Basic-Kern)	32K EPROM 27(C)256	USER-ROM mit Basic, Debugger, Assembler, Editor
D12	4k EPROM U2732	8k EPROM 27(C)64	CAOS-ROM-C

Die drei neu eingesetzten EPROM-Schaltkreise müssen mit dem aktuellen CAOS programmiert sein.

In einer weiteren Ausbaustufe zum **KC 85/5+** werden die drei EPROMs durch einen 512 KByte Flash-ROM ersetzt. Dazu wurde eine Adapterplatine entwickelt, welche in die drei EPROM-Fassungen gesteckt wird [68].

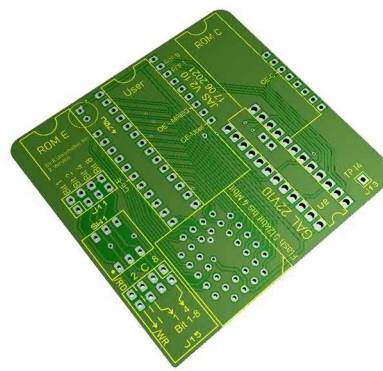


Bild 1: Ersatzleiterplatte für die 3 EPROMs mit einem 512K Flash-ROM

Damit wird eine Umschaltung zwischen 8 voneinander unabhängigen CAOS-Versionen möglich. Pro System stehen dann 64 KByte ROM zur Verfügung, je 2x 16K für den CAOS-ROM-C und CAOS-ROM-E sowie 32K für den USER-ROM.

Dies ermöglicht, Programme auszuführen, welche eine spezielle CAOS-Version erfordern oder mit neueren CAOS-Versionen nicht kompatibel sind. Mit der Adapterplatine sind drei Varianten möglich:

1. Variante: Umschaltung mit Drehschalter in der Gehäusefront
2. Variante: softwaregesteuerte Umschaltung über den JUMP-Befehl von CAOS 4.8, vier zusätzliche Verbindungen zu D009 (DL374) und D013 (DL003) sind nötig.
3. Variante: wie Variante 2 mit der Möglichkeit den FLASH zu beschreiben, zwei weitere Verbindungen zu den Signalen /RD und /WR sind nötig.

Falls ein M052 verwendet wird, empfiehlt sich außerdem die USB-Software ab Version 3.0 mit DEVICE-Treiber.

Das **modulare Konzept** der KC 85-Entwicklungsreihe ermöglicht eine überaus große Anpassungsfähigkeit des Rechners an die verschiedenen Aufgaben und Einsatzgebiete.

Die modularen Baugruppen ermöglichen z. B. eine Speichererweiterung bis zu 4 MByte, die Verwendung verschiedener Programmiersprachen (z. B. BASIC, Assembler, FORTH u. a.), den Anschluss verschiedener Peripheriegeräte (Drucker, Schreibmaschine, X-Y-Schreiber u. a.) sowie den Aufbau von Netzen. Dadurch wird der KC 85/5 zu einem wirksamen und effektiven Arbeitsmittel. Mithilfe entsprechender Programme ist der KC 85/5 für verschiedenste Anwendungen einsetzbar, z. B. für Lagerverwaltung, Aufbau und Nutzung von Datenbanken, für Ausbildungszwecke, in der Textverarbeitung, zur Prozessüberwachung und -steuerung sowie als Auswerteeinheit für Labormessungen oder als intelligentes Terminal in der Kopplung mit BC- bzw. PC-Geräten oder größeren Rechnern. Der KC 85/5 bildet das Grundgerät für ein ausbaufähiges Computersystem. Die Ergänzungs- und Erweiterungseinheiten werden in Form von steckbaren Modulen und Erweiterungsaufsätzen (zur Aufnahme dieser Module) angeboten. Sie sind unkompliziert vom Anwender in die jeweiligen Modulschächte einzusetzen und zu kontaktieren. In Verbindung mit entsprechenden Programmen resultieren hieraus die genannten vielseitigen Einsatzmöglichkeiten des Kleincomputers, u. a. auch im Heimbereich.

Das Sortiment von Erweiterungsmodulen und -aufsätzen sowie die breite Palette von Anwenderprogrammen wird von Mitgliedern des KC-Clubs ständig ergänzt. Dadurch wird es möglich, sowohl neue Erkenntnisse, als auch den ständigen Fortschritt in der Bauelemententwicklung sowie neu entstehende Anwendungsbereiche und Rechnerperipherien für den KC 85/5 zugänglich zu machen. Damit ist der KC 85/5 ein Kleinrechner, der immer auf dem aktuellen Entwicklungsstand und Einsatzspektrum gehalten werden kann.

Zum Lieferumfang der Grundausstattung des KC 85/4 gehörten:

- das KC 85/4-Grundgerät,
- die KC 85/4-Tastatur und
- die KC 85/4-Dokumentation

Die Anwendersoftware wurde in Form von Magnetbandkassetten angeboten und ist zur Nutzung mithilfe eines Kassettenrecorders in den Computer zu laden. Außerdem wurden Softwaremodule angeboten. Selbst erstellte Programme können auf einer Magnetbandkassette gespeichert und von dort ebenfalls in den Computer geladen werden. Deshalb müssen Sie über einen handelsüblichen Kassettenrecorder, wie z. B. GERACORD, LCR-C, ANETT, BABETT, LCR oder SONETT als Computerspeichereinheit verfügen.

Weiterhin benötigen Sie ein Fernsehgerät oder einen Monitor als Anzeigeeinheit.

Diese KC 85/5-Dokumentation beinhaltet:

- das Systemhandbuch,
- das BASIC-Handbuch mit den BASIC-Übersichten,
- das Handbuch für den Editor, Assembler, Reassembler und Debugger

Das Systemhandbuch beschreibt die Inbetriebnahme des Computers und die Menüanweisungen des Betriebssystems. Außerdem finden Sie hier eine kurze, aber umfassende Beschreibung der Betriebssystem-Software (Programme und Daten) und der Hardware (alles gegenständlich „Anfassbare“) des Computers.

Der Inhalt des BASIC-Kapitels 4.1 wurde aus dem BASIC-Handbuch des KC 85/4 übernommen und mit den erfolgten Anpassungen bis CAOS 4.8 aktualisiert. Damit können Sie die Programmiersprache BASIC leicht erlernen. Da alle HC-BASIC-Interpreter auf demselben BASIC-Interpreterkern aufbauen, sind die Ausführungen für den KC 85/5 im wesentlichen auch für die KC 85/3 und KC 85/4 gültig. Auf Besonderheiten wird an den entsprechenden Stellen hingewiesen. Die BASIC-Übersichten sind eine Zusammenfassung der wichtigsten Informationen für die Arbeit in BASIC am KC 85/5.

Bevor Sie jedoch das Gerät in Betrieb nehmen, bitten wir Sie, die ersten Kapitel des Systemhandbuches eingehend zu studieren und die allgemeinen Hinweise zu beachten.

Das vorliegende Handbuch soll als universelles Handbuch auch für ältere CAOS-Versionen mit gelten, deshalb werden ergänzende Angaben als Fußnoten *[1],

Unterschiede zwischen den CAOS-Versionen in roter Schrift und

Besonderheiten der DEVICE-Schnittstelle in blauer Schrift dargestellt.

Verweise auf externe Literatur stehen in eckigen Klammern [0] und können im Anhang 5.6. ab Seite 453 nachgeschlagen werden.

*1 Fußnoten stehen am Ende der Seite, mit einer Linie vom normalen Text getrennt.

USER-ROM

Der USER-ROM des KC 85/5 enthält vier unabhängig voneinander nutzbare Speicherebenen, welche mit Einschränkungen auch mit eigenständigen Programmen belegt werden können. Beschrieben wird in diesem Handbuch die Standard-Belegung mit BASIC, Assembler, Debugger und Editor.

Mit ASM 2.1 steht dem KC 85/5 ein leistungsfähiger Assembler für die Erstellung von Maschinenprogrammen in CAOS 4.8 zur Verfügung. Der neu geschriebene 80-Zeichen-Editor im Stil von „WordPro“ wird zum Editieren der Assemblerquelltexte genutzt, kann aber auch normale Textdateien erzeugen und bearbeiten. Ein Testmonitor „KC-Debugger“ und ein Reassembler komplettieren die Entwicklungsumgebung. Es handelt sich hier um die Weiterentwicklung der Software TEMO, die bereits mit dem Modul M027 für den KC 85/3 und KC 85/4 zur Verfügung stand.

Das von CAOS 4.3 bis CAOS 4.6 im USER-ROM des KC 85/5 integrierte FORTH ist zugunsten des neuen Editors ab CAOS 4.8 nicht mehr enthalten. Da die Programmiersprache FORTH gegenüber der Version im ROM-Modul M026 nicht verändert wurde, stellt dies keinen großen Nachteil dar. FORTH kann jederzeit als Modul genutzt werden. Für CAOS 4.7 steht noch ein alternativer USER-ROM mit FORTH zur Verfügung, dann fehlt aber der Texteditor und statt ASM 2.0 ist EDAS 1.7 als Editor/Assembler enthalten.

Konzept der DEVICE-Verwaltung

Das Betriebssystem CAOS wurde ursprünglich auf Kassettenarbeit ausgelegt, was auch am Namen „Cassette Aided Operating System“ erkennbar ist. Die Entwicklung des D004 eröffnete dem KC 85 die Diskette als zusätzliches Speichermedium. Mit der Entwicklung des Moduls M052 sind USB-Sticks als Speichermedium am KC 85 üblich geworden. Diese Erweiterungen brachten jede für sich neue Menüworte zum Laden und Speichern der Dateien mit. So gab es neben LOAD und SAVE noch ein FLOAD und FSAVE für Diskette und ein ULOAD bzw. USAVE für USB. Jedoch kamen nur die Programme in den Genuss der neuen Speichermedien, die genau dafür programmiert worden sind. Für BASIC gab es deshalb noch eine Erweiterung BASEX bzw. UBASEX und für EDAS ein DEVEX bzw. UDEVEX. Das CAOS-Menü wurde überfrachtet mit den vielen neuen Menüworten und die kleinen Hilfsprogramme belegten kostbaren Speicherplatz im RAM. Mit CAOS 4.6 wurde deshalb eine universelle DEVICE-Umschaltung eingeführt, die von allen Programmen genutzt wird. Fest integriert waren dabei zunächst nur Kassette und Diskette. Für andere Speichergeräte konnten Treiber nachgeladen werden. Ab CAOS 4.7 werden DEVICE-Treiber automatisch auch von USB-Modulen in das System integriert. Ziel des Konzepts:

- Kassettenbetriebsart 100% kompatibel zu früheren CAOS-Versionen.
- Anwendungsprogramme sollen möglichst nichts davon merken, mit welchem Speichermedium gerade gearbeitet wird.
- Möglichst hohe Kompatibilität zu existierender Software.

1. AUFBAU UND BEDIENUNG

1. AUFBAU UND BEDIENUNG

1.1. Inbetriebnahme

1.1.1. Bedienungselemente und Anschlüsse

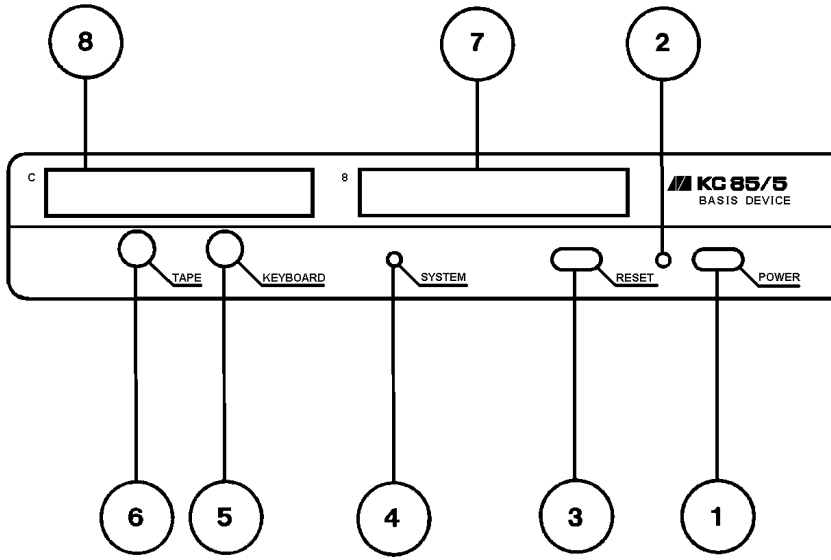


Bild 2: Vorderansicht des KC 85/5-Grundgerätes

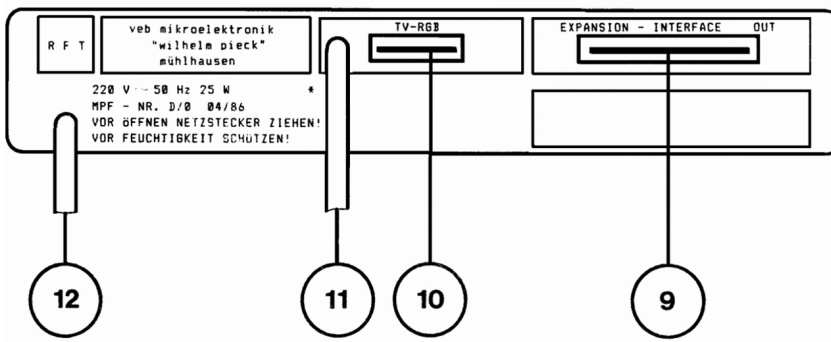


Bild 3: Rückansicht des KC 85/5-Grundgerätes

1. AUFBAU UND BEDIENUNG

① Netzschalter (POWER ON/OFF)

Mit diesem Schalter wird der Computer ein- und ausgeschaltet. Im eingeschalteten Zustand leuchtet die Netzkontrollanzeige.

② Netzkontrollanzeige

Diese Anzeige leuchtet mit vorhandener 5V-Spannung am Netzteil.

③ RESET-Taste

Durch Betätigen dieser Taste wird das Betriebssystem neu initialisiert. Es erscheinen das Menü des Betriebssystems und die Namen der zusätzlich geladenen Maschinenprogramme.

④ Systemkontrollanzeige

Diese LED-Anzeige besitzt zwei Funktionen:

1. TAPE-Kontrolle bei der LOAD- und der SAVE-Routine.
2. Durch die Verwendung der Programmverteiler V und VI im BASIC-Interpreter wird die System-LED zu- und abgeschaltet. Dies ist während der Arbeit in BASIC-Programmen am Flackern der LED zu erkennen.

⑤ Tastaturanschluss (KEYBOARD)

Die Diodenbuchse dient zum Anschluss der Tastatur.

⑥ Tonbandanschluss (TAPE)

Über die Diodenbuchse können ein Magnetbandkassettengerät oder ein entsprechendes Spulenmagnetbandgerät als Speichereinheit für Programme und Daten angeschlossen werden. Diese Buchse ist weiterhin zur Tonausgabe über entsprechenden NF-Verstärker verwendbar. Im weiteren Text wird sich auf das Kassettenmagnetbandgerät (Kassettengerät) bezogen.

⑦ Modulsteckplatz 08

Der Steckplatz dient zum Anschluss von Erweiterungsmodulen.

⑧ Modulsteckplatz 0C

Der Steckplatz dient zum Anschluss von Erweiterungsmodulen.

⑨ EXPANSION-INTERFACE

Hier können Erweiterungsaufsätze angeschlossen werden.

⑩ TV-RGB-Anschluss

Besitzt das verwendete Fernsehgerät (TV-Gerät) einen RGB- oder AV- (FBAS-) Eingang (SCART-Buchse), so können die Bildsignale durch ein Spezialkabel zu diesem Anschluss übertragen werden. Hierbei ist die Tonausgabe des Computers über den Lautsprecher des Fernsehgerätes möglich.

⑪ **UHF-Anschluss**

Sollen die Bildsignale über den Antenneneingang (UHF, Kanal 36, Band IV) in das Fernsehgerät eingespeist werden, so ist die an der Computerrückseite herausgeführte HF-Anschlussleitung in den UHF-Antenneneingang des Fernsehgerätes zu stecken. Dabei ist eine Tonausgabe über das Fernsehgerät jedoch nicht möglich (Lautstärkeregler zurückdrehen).

⑫ **Netzanschlussleitung**

1.1.2. Schritte zur Inbetriebnahme eines KC-Systems

Möchten Sie das Computersystem nun in Betrieb nehmen, so benötigen Sie, wie bereits erwähnt, ein Fernsehgerät oder einen Monitor. Darüber hinaus ist es zur Speicherung von Daten notwendig, einen Kassettenrecorder oder ein entsprechendes Spulenmagnetbandgerät durch ein handelsübliches Diodenkabel anzuschließen.

Sind diese Grundelemente vorhanden, kann das Computersystem wie folgt aufgebaut werden:

1. Stecken Sie den Diodenstecker der Tastatur in die mit KEYBOARD bezeichnete Buchse.
2. Schließen Sie den Recorder mit einem Diodenkabel an den Tonbandeingang (TAPE) des Computers an. An dieser Buchse befinden sich, neben den üblichen Anschlüssen für ein Monokassettengerät (Aufnahme und Wiedergabe), auch der Anschluss für Zweikanalton und ein Steuersignal (TTL-Pegel), mit dem der Kassettenantrieb des Recorders beim Laden und Retten betätigt werden kann.
   Das Diodenkabel zum Recorder gehört nicht zum Lieferumfang des KC 85/4. Beachten Sie, dass sich Dioden- und Überspielkabel in ihrer Anschlussbelegung unterscheiden! Verwenden Sie den richtigen Kabeltyp zum Anschluss des Kassettenrecorders.
   Es kann jeder handelsübliche Kassettenrecorder verwendet werden, der folgende Bedingungen erfüllt:
   a) Die Ausgangsspannung Ua bei Wiedergabe muss größer als 200 mVss sein (nach TGL 28200/13) bei einer Belastung von Ra = 20 kOhm.
   b) Die Eingangsspannung Ue bei Aufnahme darf kleiner sein als 20 mVss bei einer Belastung von Re = 5 kOhm.
   c) Der zu übertragende Frequenzbereich des Kassettenrecorders muss mindestens die Frequenzen 400 Hz…8 kHz umfassen (nach TGL 27616/2). Die Recorder GERACORD, ANETT, LCR, BABETT und SONETT erfüllen diese Forderungen. Nicht geeignet sind z. B. Geräte wie STERN-RECORDER bis R4100 und der Typ SK900.

3. Schließen Sie das an der Rückseite befindliche Antennenkabel an den UHF-Antenneneingang des Fernsehgerätes an.
   Bei einem Monitor mit AV- oder RGB-Eingang wird der TV-RGB-Anschluss des Computers über eine entsprechende Spezialleitung mit dem Monitor verbunden. (Nähere Ausführungen zur Anschlussbelegung finden Sie im Kapitel 2.2.6. auf Seite 101.)
   Die Verbindungsleitung vom TV-RGB-Anschluss des Computers zum Fernsehgerät bzw. Monitor gehört nicht zum Lieferumfang des KC 85/4.
4. Alle drei Geräte sind nun an das Stromnetz (230V/50Hz) anzuschließen.

Falls ein Mono-Kassettenrecorder in Verbindung mit einem Stereo-Diodenkabel verwendet wird, bei dem die Kontakte für Stereoaufnahme und -wiedergabe verbunden sind, kann die Schaltspannung das ordnungsgemäße Laden von Programmen verhindern. Dann ist diese Brücke im Diodenkabel oder im Kassettenrecorder durch einen Fachmann zu entfernen.

Hinweise:

Bei der Verbindung des Kleincomputers mit peripheren Geräten ist darauf zu achten, dass von der Gerätekonfiguration keine unzulässigen Funkstörungen abgestrahlt werden (siehe [14] im Literaturverzeichnis). Für die Grundkonfiguration gemäß Bild 4 bedeutet das insbesondere, dass der HF-Ausgang des Computers nicht gleichzeitig mit der Antenne am Fernsehgerät angeschlossen sein darf und dass für die Verbindung der Geräte untereinander ordnungsgemäß abgeschirmte Kabel verwendet werden. Jede missbräuchliche Anwendung in einer anderen Konfiguration wird entsprechend dem Gesetz über das Post- und Fernmeldewesen geahndet /14/.
Das Gerät wurde vom Ministerium für Post- und Fernmeldewesen abgenommen und für den Betrieb freigegeben.
Stellen Sie den Kanalwähler des Fernsehgerätes auf Kanal 36 (UHF-Bereich, Band IV) ein.
Kontrollieren Sie, ob die Anschlüsse für Tastatur und Recorder in der zugehörigen Buchse stecken.
Schalten Sie nun nacheinander den Recorder, das Fernsehgerät und den Computer (Netzschalter) ein.
Durch Zu- oder Abschalten der Netzspannung des Kassettenrecorders entstehen Störimpulse. Deshalb ist keine Schaltung der Netzspannung des Recorders vorzunehmen, wenn die Verbindung Recorder-Computer über Diodenkabel besteht. Beachten Sie, dass sich Dioden- und Überspielkabel in ihrer Anschlussbelegung unterscheiden! Verwenden Sie den richtigen Kabeltyp zum Anschluss des Kassettengerätes!

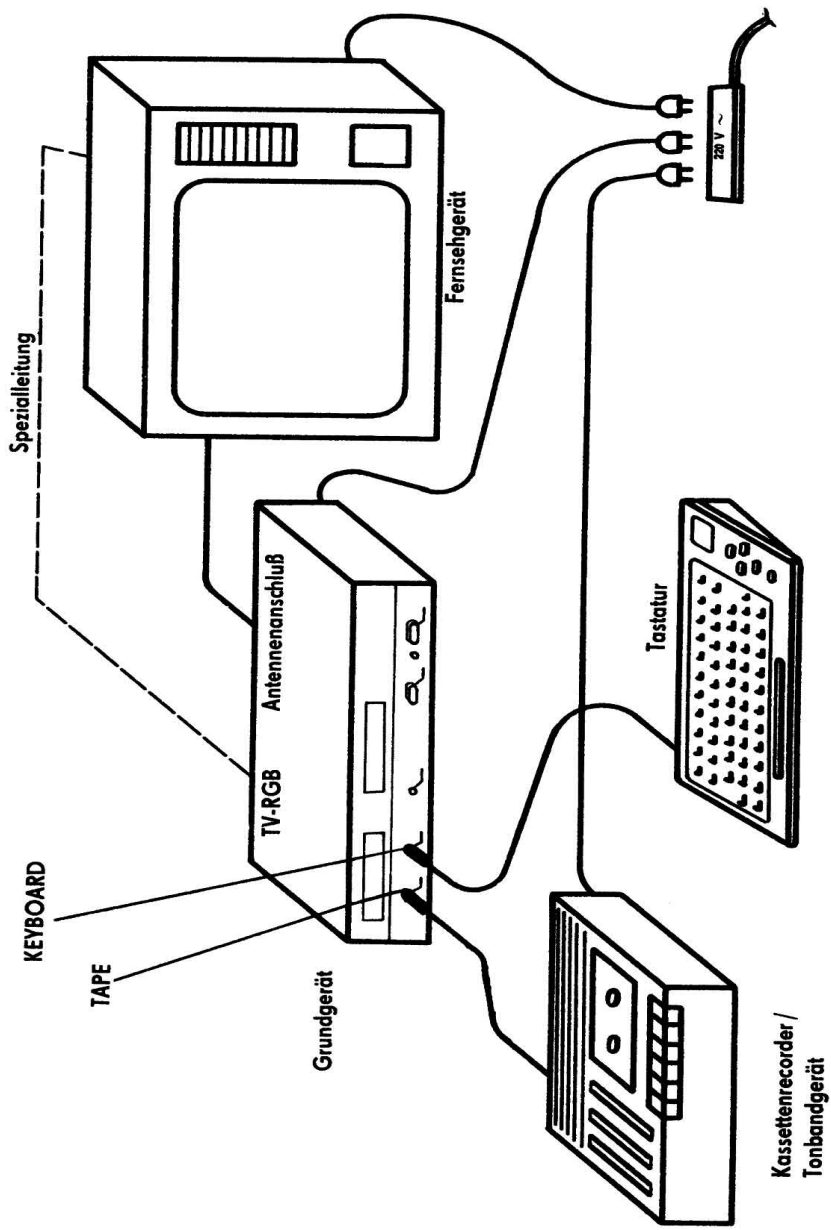


Bild 4: Anschlussschema des Kleincomputersystems

1.1.3. Das erste Computerbild

Nach dem Einschalten des Computers leuchtet die Netzkontrollanzeige auf und der KC 85/5 meldet sich mit folgendem Menü auf dem Fernsehbildschirm arbeitsbereit:

Bild 5: CAOS-Grundmenü

Ist Ihr Fernsehbild unscharf oder verzerrt, regeln Sie es durch die Feineinstellung am Kanalwähler nach.
In der obersten Zeile des Menüs steht der Name des Betriebssystems: im Beispiel KC-CAOS 4.8 sowie das eingestellte Speichergerät, hier TAPE für Kassette. Das CAOS-Betriebssystem ist das Verbindungselement zwischen der Hardware und dem Anwender. Es enthält Programme (siehe Menü), die nach dem Einschalten des Computers den Anwendern zur Verfügung stehen. Beim KC 85/5+ kann zwischen 8 internen Versionen des Betriebssystems umgeschaltet werden. Das interne Betriebssystem kann auch komplett weg- und ein anderes, auf einem Modul befindliches Betriebssystem eingeschaltet werden.
Unter dem Namen des Betriebssystems des KC 85/5 folgt eine Auswahl der angebotenen Kommandoworte des Systems. Sie können angewählt oder eingegeben werden. Vor dem Kommandowort steht das Promptzeichen, das Sie hier als Prozentzeichen erkennen. Im Vergleich dazu besitzt der BASIC-Interpreter dieses Zeichen '>' als Promptzeichen. In der Zeile unter den Menüworten sehen Sie auf dem Bildschirm den Cursor blinken. Er besteht aus 1*7 Bildpunkten und ist somit als kleiner Strich zu erkennen. Auf die jeweilige Cursorposition wird das nächste von der Tastatur bzw. vom Programm ausgegebene Zeichen platziert. Dabei rückt der Cursor selbst nach jedem eingegebenen Zeichen um eine Position nach rechts bzw. vom Zeilenende zum Anfang der nächsten Zeile. Kommt der Cursor auf eine Stelle, auf der sich schon ein Zeichen befindet, blinkt das gesamte Cursorfeld, bestehend aus 8*8 Bildpunkten.

1.1.4. Einschaltfehler

In der folgenden Tabelle werden Ihnen einige Hinweise gegeben zum Erkennen von Einschaltfehlern und deren Beseitigung. Sollten Sie trotzdem kein erkennbares Bild erhalten, ist das Gerät in eine Vertragswerkstatt zu geben.

Tabelle 1: Fehlertabelle für Einschalten KC 85/5

Fehler	Ursache	Beseitigung
- kein Bild, Power-LED dunkel	- Netzstecker nicht in Steckdose	- Netzstecker stecken
	- Gerätesicherung defekt	- Vor dem Öffnen des Gerätes Netzstecker ziehen! Die zwei Gerätesicherungen sind nach dem Entfernen der oberen Abdeckung des Grundgerätes (4 Kreuzschlitz-Schrauben lösen) im Bereich des Netzteils zugänglich. Es dürfen keine Sicherungen mit anderen als den angegebenen Werten eingesetzt werden. Bei erneutem Durchschlagen der Sicherungen ist das Gerät zur Reparatur zu bringen.
- kein Bild, Power-LED leuchtet	- Antennenanschluss an Fernsehgerät nicht angeschlossen	- Anschluss herstellen
	- Kanal 36 (UHF-Bereich, Band IV) nicht eingestellt	- Bei Fernsehgeräten mit AFC-Kanaleinstellung darf diese Taste nicht gedrückt sein. Erst nach dem richtigen Einstellen des Fernsehbildes ist die <AFC>-Taste wieder zu betätigen.
- Bild vorhanden, aber nur waagerechte Streifen bzw. Farbmuster	- Modul oder Aufsatz defekt	- <RESET>-Taste betätigen - erneutes Aus- und Einschalten - ausschalten, Module (und Aufsatz) vom Grundgerät trennen und wieder einschalten.

Fehler	Ursache	Beseitigung
- Bild nur schemenhaft oder unscharf	- Kanal ungenau eingestellt	- Erneutes Einstellen des Kanalwählers
	- falscher Bereich eingestellt	- Band IV Kanal 36 einstellen
	- Antennenleitung im VHF-Bereich gesteckt	- UHF-Anschluss benutzen
- Bild vorhanden, keine Eingabe über Tastatur möglich	- Tastaturanschluss verwechselt	

1.1.5. Hinweise

– Reinigen Sie die Gehäuseoberfläche des Grundgerätes und der Tastatur nur mit einem weichen Tuch, das - sofern nötig - leicht anzufeuchten ist.

– Es kann ein Netzmittel (z. B. Geschirrspülmittel) zugesetzt werden. Verwenden Sie bitte keine schnell verdunstenden Flüssigkeiten (Alkohole, Farbverdünner, Benzin und ähnliches).
Für die Reinigung der Kontakte, z. B. des Steckverbinders TV-RGB, ist Alkohol erlaubt.

– Beim Betrieb ist unbedingt darauf zu achten, dass die Lüftungsschlitze an der Ober- und Unterseite nicht abgedeckt werden (z. B. durch Arbeitsunterlagen, Stellen auf weiche Unterlage usw.)

– Defekte Sicherungen (G-Schmelzeinsätze) können Sie durch die entsprechenden neuen ersetzen. Bei einem häufigen Ausfall der Sicherungen ist es erforderlich, sich an die Vertragswerkstatt zu wenden.

1.2. Tastatur

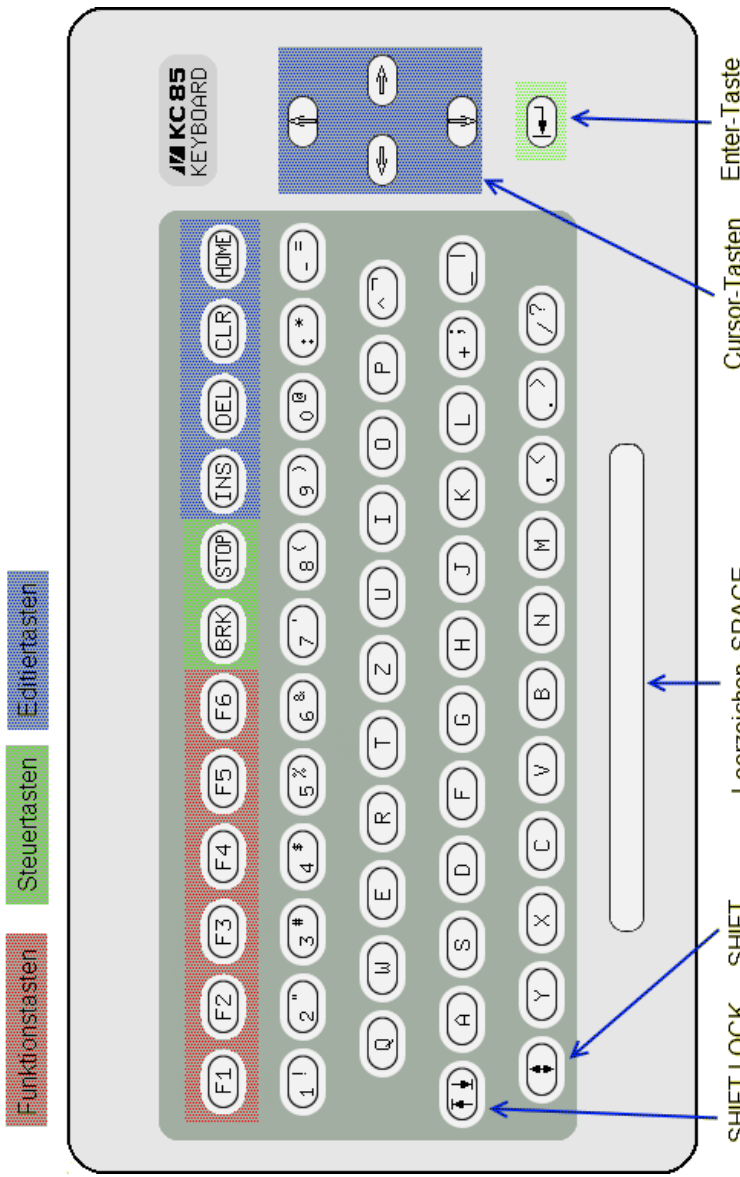


Bild 6: Tastatur mit Editier-, Steuer- und Funktionstasten

1.2.1. Funktion und Tastaturebenen

Nachdem Sie die Verbindungen nach Bild 4 gesteckt, Fernsehgerät, Recorder und Computer wie in Kapitel 1.1.2. beschrieben eingeschaltet haben, liegt vor Ihnen die Tastatur zur Eingabe bereit.

Es ist eine Tastatur zur manuellen Eingabe von Buchstaben, Ziffern, Sonder- und Steuerzeichen. Mit 64 Tasten können Sie dem Computer Informationen eingeben, wie z. B. der Aufruf des BASIC-Interpreters, Programme laden und retten usw. Auf dem Bildschirm werden Ihnen die Reaktion des Computers bzw. die Bestätigung der Ausführung des eingegebenen Kommandos bzw. Befehls angezeigt.

Zur Erweiterung der Kommunikationsmöglichkeiten besitzt die Tastatur verschiedene Belegungsarten: 3 Tastaturebenen und je eine Erstbelegung und eine Zweitbelegung (für 1. und 2. Ebene)

1. Tastaturebene: Programmeingabemodus
   zu erreichen: Grundzustand nach Einschalten des Computers
   beinhaltet: *Erstbelegung*: Großbuchstaben, Ziffern, Sonderzeichen
   Zweitbelegung: Kleinbuchstaben, Sonderzeichen

2. Tastaturebene: Texteingabemodus
   zu erreichen: aus dem Grundzustand und über 
   beinhaltet: *Erstbelegung*: Kleinbuchstaben, Buchstabe ß, Ziffern, Sonderzeichen
   Zweitbelegung: Großbuchstaben, kleine Umlaute, Sonderzeichen

3. Tastaturebene: Steuermodus
   zu erreichen: aus dem Grundzustand über  und 
   rücksetzen: automatisch nach jeder Eingabe
   beinhaltet: Steuerfunktionen

Der Übergang von der Erst- in die Zweitbelegung erfolgt durch Drücken der Taste  . Die Zweitbelegung ist nur so lange wirksam, wie diese Taste gedrückt bleibt! Achtung: Das Benutzen der Taste  bedingt den Wechsel der Tastaturebene – ist also nicht identisch mit  (siehe Tabelle 2 auf Seite 29)

1. AUFBAU UND BEDIENUNG

In der **ersten Tastaturebene** (Programmeingabemodus) gelten für die Erstbelegung die auf den Tasten unten stehenden Zeichen, in der Zweitbelegung die auf den Tasten oben abgebildeten Zeichen. Der Begriff „Programmeingabemodus“ resultiert aus der Tatsache, dass Anweisungen und Kommandos in vielen Programmiersprachen, z. B. auch in BASIC des KC 85/5, mit Großbuchstaben geschrieben werden. Die Großbuchstaben sind in diesem Modus über die Erstbelegung der Tasten erreichbar.

In der **zweiten Tastaturebene** (Texteingabemodus) funktionieren die Buchstaben wie bei einer Schreibmaschine, also in der Erstbelegung erscheinen Kleinbuchstaben, Ziffern und Sonderzeichen. In der Erstbelegung ist weiterhin der Buchstabe ß zugänglich. In der Zweitbelegung sind Großbuchstaben und Sonderzeichen zu erreichen (vgl. Tabelle 35: Umcodierungstabelle der KC-Tastatur ab Seite 222).
Außerdem sind in der Zweitbelegung dieses Modus die Umlaute des deutschen Schriftsatzes zugänglich (siehe Tabelle 2 auf Seite 29). Der Texteingabemodus ist sehr vorteilhaft für die Eingabe von Texten, worauf sich auch die Modusbezeichnung begründet. Allerdings ist die Lage der Tasten für die neu zugänglichen Zeichen (Buchstabe 'ß' und Umlaute) auf der KC-Tastatur nicht identisch mit ihrer Anordnung auf einer Schreibmaschinentastatur.

In der **dritten Tastaturebene** (Steuermodus) wird das nachfolgend über die Tastatur eingegebene Zeichen als Steuerzeichen erkannt und die zugehörige Funktion ausgeführt (vgl. Tabelle 3: ESC-Steuerfunktionen von CAOS 4.x auf Seite 34). Danach wird die 3. Tastaturebene automatisch verlassen und der Computer befindet sich wieder in der 1. bzw. 2. Tastaturebene.

Alle Tasten besitzen die Autorepeat-Funktion. Darunter ist das wiederholte Einlesen des Tastencodes bei längerem Tastendruck zu verstehen.

Die vier, mit Pfeilen gekennzeichneten Tasten, die sich ganz rechts auf der Tastatur befinden, sind die Cursortasten. Mit diesen lässt sich der Cursor in der Erstbelegung der Tastatur beliebig nach oben, rechts, links oder unten über den Bildschirm verschieben (außer in BASIC). Auf diese Art und Weise können Sie auch festlegen, an welcher Stelle des Bildschirms Sie eine Eingabe vornehmen.
Die verschiedenen Tastenarten der Tastatur sind im Bild 6 auf Seite 26 dargestellt, sie werden nachfolgend erläutert.

1.2.2. Buchstaben, Ziffern

Zur Verständigung zwischen dem Anwender und dem Computer müssen spezielle Codes von der Tastatur an den Computer gesendet werden. Drücken Sie z. B. auf die Taste 

, so erscheint auf dem Bildschirm der Buchstabe A. Das ist möglich, weil der Buchstabe von der Tastatur als Impulsfolge an den Computer gesandt wird. Dort wird aus dem empfangenen Code das entsprechende Zeichen generiert und auf dem Bildschirm dargestellt.
Die Codierung der Zeichen wird im Computer gemäß ASCII (American Standard Code for Information Interchange) vorgenommen.
Der Zeichenvorrat des Computers ist im Kapitel 3.10.1. ab Seite 203 aufgeführt.
Den Anforderungen der Computerarbeit entsprechend, besitzt die Tastatur im Vergleich zur Schreibmaschine noch zusätzlich einige Sondertasten. Zur Eingabe von Buchstaben, Zahlen und Sonderzeichen dienen die alphanumerischen Eingabetasten. Diese sind wie auf einer Schreibmaschine angeordnet.
Folgende Unterschiede zwischen Programmeingabe- und Texteingabemodus müssen beachtet werden.

Tabelle 2: Geänderte Tastenbelegungen im Texteingabemodus

Taste	Programmeingabemodus		Texteingabemodus	
	Erst-	Zweit-	Erst-	Zweit-
	belegung		belegung	
	^	¬	ß	ü
	_	|	□	ö
SPACE		■		ä
	0	@	0	©

1.2.3. Steuertasten

Eine weitere Funktionsgruppe bilden am Computer die Steuerfunktionen. Diese können Sie in allen 3 Tastaturebenen benutzen. Folgende Tasten gehören zu dieser Gruppe:

SHIFT: Umschaltung der Tastaturbelegung

Mit dieser Taste wird auf die Zweitbelegung der Tasten für die Dauer der Betätigung umgeschaltet.

SHIFT LOCK / Tabulator

Erstbelegung:

Beim ersten Betätigen dieser Taste erfolgt ein Umschalten auf die Erstbelegung der 2. Tastaturebene. Mit der SHIFT-Taste kann von dieser Ebene aus wieder die Zweitbelegung (dieser Ebene) erreicht werden, die in einigen Fällen mit der Zweitbelegung der ersten Tastaturebene übereinstimmt, mit dieser aber nicht identisch ist (vgl. Tabelle 2 auf Seite 29).

Zweitbelegung:

Tabulatorschritt mit Schrittweite 8.

ENTER: Beenden und Ausführen einer Eingabe

Erstbelegung:

Durch Drücken dieser Taste wird die Eingabe einer Befehls- oder Datenzeile beendet. Dabei wird die Eingabe gleichzeitig bearbeitet, z. B. gespeichert oder das Kommando ausgeführt. Der Cursor wird auf den Beginn der nächsten Bildschirmzeile gesetzt.

Zweitbelegung:

Bis CAOS 4.2 identisch mit Erstbelegung, ab CAOS 4.3 keine Funktion, kann vom Anwenderprogramm separat abgefragt werden.

BREAK: verschiedene modispezifische Steuerfunktionen

Erstbelegung:

Die Taste wird in bestimmten Programmen zur Steuerung, meist zur Programmunterbrechung (z. B. in BASIC) benutzt.
Im CAOS-Menü wird eine leere Eingabezeile erzeugt.

Zweitbelegung:

Bis CAOS 4.2 identisch mit Erstbelegung, ab CAOS 4.3 keine Funktion, kann vom Anwenderprogramm separat abgefragt werden.

STOP: verschiedene modispezifische Steuerfunktionen

Erstbelegung:
Die Taste wird in bestimmten Programmen (z. B. in BASIC für Programmhalt) zur Steuerung benutzt.

Zweitbelegung:
Einschalten der 3. Tastaturebene (ESCape). Mit <SHIFT>-<STOP> wird die 3. Tastaturebene eingeschaltet. Das nachfolgend eingegebene Zeichen wird als Steuerzeichen interpretiert. Es sind nur Ziffern und Buchstaben als Steuerzeichen zulässig.

1.2.4. Editiertasten

Die Editiertasten unterstützen die Bildschirmarbeit am Computer.
Zu ihnen gehören:

INSERT: Zeichen einfügen / Tastenklick

Erstbelegung:
Mit der Taste ist es möglich, in schon vorhandene Schriftzeilen weitere Buchstaben, Ziffern oder Zeichen einzufügen. Dabei werden das auf der Cursorposition befindliche und die rechts davon befindlichen Zeichen insgesamt um eine Stelle nach rechts verschoben. Das dadurch entstandene Leerzeichen kann zur Einfügung genutzt werden.

Zweitbelegung:
Ein- und Ausschalten der akustischen Tastenquittierung (Tastenklick)

DELETE: Zeichen löschen / Zeile löschen

Erstbelegung:
Das Zeichen, auf dem sich der Cursor befindet, wird gelöscht und die Zeile verdichtet, d. h. die Zeichen rechts der Cursorposition bis zum Zeilenende werden um eine Stelle nach links verschoben.

Zweitbelegung:
Die gesamte Zeile, in welcher sich der Cursor befindet, wird gelöscht. Der Cursor befindet sich nach dem Löschen am Anfang der Zeile.

CLEAR: Zeichen löschen / Aufruf Sonderprogramm

Erstbelegung:
Das Zeichen, das sich vor dem Cursor befindet, wird gelöscht und der Cursor bewegt sich eine Position nach links.

Zweitbelegung:
Aufruf eines Sonderprogramms (z. B. HCOPY).

CURSOR HOME: Cursor nach links oben / Fenster löschen

Erstbelegung:
Der Cursor wird in der oberen, linken Ecke des Bildfensters platziert.

Zweitbelegung:
Das Bildfenster wird gelöscht (CLEAR SCREEN). Der Cursor erscheint in der oberen, linken Ecke des Fensters.

Cursor nach unten / SCROLL-Modus

Erstbelegung:
Der Cursor bewegt sich um eine Zeile nach unten.

Zweitbelegung:
Der SCROLL-Modus wird eingeschaltet. Bei Bildüberlauf (d. h. der Bildschirm ist bis auf die unterste Zeile im aktuellen Fenster beschrieben) verschiebt sich der gesamte Bildschirminhalt um eine Zeile nach oben. Dabei verschwindet die oberste Zeile und es entsteht am unteren Bildschirmrand eine freie Zeile, die neu beschrieben werden kann. Der SCROLL-Modus wird nach dem Einschalten des Computers bzw. nach RESET automatisch eingestellt.

Cursor nach rechts / Cursor auf Zeilenende

Erstbelegung:
Der Cursor bewegt sich um ein Zeichen nach rechts.

Zweitbelegung:
Der Cursor geht zum Zeilenende.

Cursor nach links / Cursor auf Zeilenanfang

Erstbelegung:
Der Cursor bewegt sich um ein Zeichen nach links.

Zweitbelegung:
Der Cursor wird auf den Zeilenanfang gesetzt.

Cursor nach oben / PAGE-Modus

Erstbelegung:
Der Cursor bewegt sich um eine Zeile nach oben.

Zweitbelegung:
Der PAGE-Modus wird eingeschaltet. Dieser bewirkt bei Bildüberlauf (aktuelles Fenster) das Rücksetzen des Cursors in die obere linke Ecke des Bildschirmes, sodass dieser erneut überschrieben werden kann. Im PAGE-Modus können Fehler bei der Abarbeitung von Kommandos auf der letzten Zeile des Bildschirmes auftreten. Im Normalfall deshalb den SCROLL-Modus nutzen.

1.2.5. Funktionstasten

Tasten  bis 

Die Funktionen dieser Tasten können durch den Anwender selbst festgelegt werden. Für die Belegung dieser Tasten gibt es das Betriebssystemkommando KEY (Seite 47). Mit der Erst- und Zweitbelegung können insgesamt 12 Tastenfunktionen (F1, …, F9, FA, FB, FC) programmiert werden.

1.2.6. „ESC-Taste" und Arbeit mit der 3. Tastaturebene

Mit der Tastenfolge  gefolgt von  wird der Code 1BH (27 dezimal) an den Computer gesendet und damit die 3. Tastaturebene aufgerufen. Diese Codierung ist bei vielen Computern und Druckern als ESCape-Funktion definiert. In der Computersprache wird der Steuercode ESCape-Code = ESC als Umschaltcode definiert. Mit dem ESC-Code wird dem Computer angezeigt, dass es sich beim nächsten Tastendruck bzw. von der Tastatur gesendeten Zeichen um ein Steuerzeichen handelt. Als Steuerzeichen sind nur Ziffern von 0 bis 9 und alle Buchstaben von A bis Z zugelassen. Dabei wird die Groß- und Kleinschreibung nicht unterschieden.

Beginnend mit CAOS 4.1 sind die Ziffern von 0 bis 9 und der Buchstabe A mit Steuerfunktionen belegt. Weitere Funktionen wurden mit nachfolgenden CAOS-Versionen eingeführt. Diese Steuerfunktionen werden durch Maschinenprogramme realisiert, die unter der jeweiligen Ziffer oder dem jeweiligen Buchstaben abgelegt sind.
In der folgenden Tabelle sind die Steuerfunktionen von CAOS 4.8 zusammengefasst.

Tabelle 3: ESC-Steuerfunktionen von CAOS 4.x

Taste	Funktion	ab CAOS
ESC-0	Tabulatorschritt (Schrittweite 8)	4.1
ESC-1	Anzeigen und Beschreiben von Bild 0	4.1
ESC-2	Anzeigen und Beschreiben von Bild 1	4.1
ESC-3	Anzeigen von Bild 0 und Beschreiben von Bild 1	4.1
ESC-4	Anzeigen von Bild 1 und Beschreiben von Bild 0	4.1
ESC-5	Modulkontrollanzeige	4.1
ESC-6	Systemcheck (interne Module)	4.1
ESC-7	Inverses Schreiben Aus-/Einschalten	4.1
ESC-8	Farbe komplementieren (Vordergrund- wird Hintergrundfarbe und umgekehrt)	4.1
ESC-9	Ein-/Abschalten der Farbebene	4.1
ESC-A	Ein-/Abschalten der Pixelfarbauflösung (wie beim KC 85/4)	4.1
ESC-B	Ein-/Abschalten HRG-Modus (High-Resolution-Grafik)	4.3
ESC-C	Ein-/Abschalten des erweiterten IBM-Zeichensatzes	4.3
ESC-D	Anzeige und Auswahl des aktuellen Speichergerätes	4.6
ESC-E	Sprung zu Adresse E000H (Soft-RESET)	4.8
ESC-F	Sprung zu Adresse F000H (Soft-POWER-ON)	4.8
ESC-G	2-Monitor-Modus EIN/AUS	4.8
ESC-H	Hilfefunktion: Auflisten aller Menüworte	4.8

Die aufgeführten Steuerfunktionen lassen sich beliebig ändern und erweitern. Für das Erweitern stehen Ihnen die Buchstaben I bis Z zur Verfügung (siehe Kapitel 3.10.4. Seite 227).

1.2.7. Möglichkeiten zur Änderung der Tastenfunktionen

Vom CAOS-Betriebssystem werden dem Anwender bestimmte Tastenfunktionen angeboten. Diese können von Anwenderprogrammen verändert werden, z. B. ist die Funktion der Taste STOP in BASIC der Programmhalt und im Betriebssystemkommando DISPLAY ist sie die Umschalttaste in den MODIFY-Modus. Auch reagieren die Cursortasten unter BASIC anders. Es ist nicht möglich, den Cursor über eine BASIC-Zeile hinauszubewegen. Vom Betriebssystem aus kann jedoch der Cursor in alle Richtungen bewegt werden.
Im Textverarbeitungssystem TEXOR befindet sich z. B. auf der Funktionstaste F5 das Zeichen '\' und in der Zweitbelegung '◇'. Dadurch sind beide Zeichen über Tastendruck auf dem Bildschirm darstellbar. Im Normalfall sind diese Zeichen nicht im Zeichenvorrat des Computers enthalten.
Sie können im Computer auf einem freien Speicherbereich (z. B. BC00H) neu erstellte Zeichen ablegen und diese über die Tastatur ansprechen. Außerdem sind das Verlegen und Löschen von Tastenfunktionen möglich.
Mit den 6 Funktionstasten stehen Ihnen 12 (mit Zweitbelegung) weitere Tastenbelegungsmöglichkeiten zur Verfügung, ohne die vorhandenen zu beeinflussen.
Ebenfalls wird durch die Umschalttasten ⇕ und ⇕ die Funktion einer Taste geändert, wie z. B. das Schalten in die verschiedenen Tastaturebenen.
Zum Einbinden weiterer Tastenfunktionen wurde die Umschaltfunktion (ESC) für die 3. Tastaturebene gewählt.
Mit diesem Schritt stehen insgesamt 36 mögliche Steuerfunktionen zur Verfügung. Diese können auch komplett neu belegt werden, wovon z. B. der Editor Gebrauch macht. Im Texteingabemodus werden beispielsweise mit ESC-0 bis ESC-9 spezielle Druckersteuerzeichen eingegeben.

Eine komplette Neubelegung aller Tastencodes ist außerdem durch die Tastaturtabelle KTAB möglich, siehe Kapitel 3.10.2 ab Seite 221.

! Dies gilt jedoch nicht für Tastaturen, welche an Modulen (z. B. M052, M053) betrieben werden. Diese Tastaturen senden den Tastencode direkt zum Computer und verwenden nicht die Tabelle KTAB zur Codierung.

1.3. Ein notwendiger Blick hinter die Kulissen

1.3.1. Vom Bit zur Hexadezimalzahl

Bevor wir uns weiter mit dem Computer vertraut machen, werden Sie in diesem Abschnitt einige Grundbegriffe der Computertechnik kennenlernen.
Die kleinste Informationseinheit, die der Computer kennt, ist ein Bit. Ein Bit kann nur eine von zwei möglichen Informationen tragen. Diese Informationen können Sie auch als Zahlen ansehen, also 1 oder 0. Zwei Bit können demnach 4 Zahlen darstellen, nämlich: 00, 01, 10, 11. Überlegen wir uns anhand des folgenden Schemas wie diese Entwicklung weitergeht:

1 Bit	kann	2^1 = 2^1	= 2 Zahlen darstellen.
2 Bit	können	2^2 = 2^2	= 4 Zahlen darstellen.
3 Bit	können	2^3 = 2^3	= 8 Zahlen darstellen.
4 Bit	können	2^4 = 2^4	= 16 Zahlen darstellen.
⋮			
8 Bit	können	2^8 = 2^8	= 256 Zahlen darstellen.
⋮			
16 Bit	können	2^16 = 2^{16}	= 65.536 Zahlen darstellen.

Die Darstellung der Zahlen in der Form 2^8 ist die Exponentendarstellung in der Computerschreibweise (siehe BASIC-Handbuch).
Dabei kommen den 4 Bits und 8 Bits als Einheit eine besondere Bedeutung zu. Um die Arbeit mit dem Computer übersichtlicher zu gestalten, fasst man z. B. jeweils vier Bits (man sagt dazu auch Tetrade) zu einer hexadezimalen Ziffer zusammen.

Bits	hexadezimal	dezimal
0000	0	0
0001	1	1
0010	2	2
0011	3	3
0100	4	4
0101	5	5
0110	6	6
0111	7	7
1000	8	8
1001	9	9
1010	A	10
1011	B	11
1100	C	12
1101	D	13
1110	E	14
1111	F	15

Wie Sie sehen, ergeben sich mit 4 Bit 16 verschiedene Kombinationen.
Da die gebräuchlichen 10 arabischen Ziffern zur Darstellung dieser 4 Bit langen dualen Werte nicht mehr ausreichen, wird das Ziffernrepertoire, wie dargestellt, um die Ziffern A, B, C, D, E und F ergänzt. Diese Ziffern sind die Grundbausteine für ein Stellenwertsystem zur Basis 16 (Hexadezimalsystem).
Zweimal 4 Bit oder zwei hexadezimale Ziffern sind 8 Bit oder ein Byte. Unser Computerspeicher ist in jeweils 8 Bit, also in Bytes aufgeteilt. Jedes Byte ist durch eine Adresse ansprechbar. Die Adressen können ebenfalls mit hexadezimalen Ziffern ausgedrückt werden. Der Prozessor kann direkt 2^{16} Byte = 65.536 Byte oder 64 KByte zu je 8 Bit adressieren (1 KByte = 1024 Byte). Die Adressen laufen dabei von 0000H bis FFFFH.

1.3.2. Logische Funktionen

Ein logischer Ausdruck besteht aus Vergleichsaussagen, die durch logische Operatoren miteinander verbunden sein können. Vergleichsoperatoren sind z. B. =, <, >. Ein logischer Ausdruck kann wahr (TRUE) oder falsch (FALSE) sein. Um das Ergebnis eines Vergleiches auszudrücken, werden Zahlenwerte eingesetzt. Das sind für falsch = 0 und für wahr = 1.
Außerdem gibt es die Booleschen Operatoren AND = Konjunktion (logisch UND), OR = Disjunktion (logisch ODER) und NOT = Negation (Verneinung), die z. B. in BASIC enthalten sind.
Eine weitere logische Funktion (nicht im KC-BASIC enthalten) ist XOR = Antivalenz (log. ENTWEDER ODER).
Der Überlagerungsmode der Grafikgrundbefehle wird durch die XOR-Verknüpfung vom Pixel-RAM mit der zu zeichnenden Grafik realisiert.

Beispiel:

Für die logische Funktion XOR (exklusiv ODER) wird das Ergebnis wahr = 1, wenn von 2 Variablen A und B eine wahr ist.

A	B	A OR B	A AND B	A XOR B
0	0	0	0	0
1	0	1	0	1
0	1	1	0	1
1	1	1	1	0

Bei einer XOR-Verknüpfung im Pixel-RAM wird ein Punkt gesetzt, wo vorher keiner und ein Punkt gelöscht, wo vorher einer war.
Durch mehrmaliges Aufrufen von Grafikbefehlen können so mit denselben Parametern Punkte gesetzt und gelöscht werden.

1.3.3. Steuerbyte

Ein Byte kann verschiedene Informationen besitzen. Diese sind codiert und können Daten, Befehle oder Steuerinformationen sein. Daten z. B. enthalten den Code von Buchstaben, Ziffern usw. Bei Steuerinformationen enthält jedes Bit in einem Byte eine bestimmte Wertigkeit, die ebenfalls, wie bei allen anderen Informationen, 0 oder 1 sein kann. Jedoch erfolgt hier durch die Zuordnung für 1 = einschalten und für 0 = ausschalten eine Steuerung wie bei einem Schalter. Demzufolge kann mit einem Bit eine Schalterfunktion realisiert werden. Auf dem gleichen Prinzip basiert der Aufbau des Steuerbytes kk.
Das Steuerbyte kk ist ein Parameter des SWITCH-Kommandos bzw. der BASIC-Anweisung SWITCH. In Verbindung mit der Steckplatzadresse mm werden dem Computer Informationen gegeben, ob es sich um das Ein- oder Ausschalten eines Moduls oder eines RAM-Segmentes handelt.
Das Schalten kann über Menüwort oder vom Programm aus erfolgen. Dabei sind die jeweiligen Bedingungen der Module und Speichersegmente zu beachten.

1.3.4. Speicher für Programme und Daten

Im KC 85/5 sind 2 Speicherarten (ROM und RAM) eingebaut. Aus dem ROM, dem Festwertspeicher, können wir nur Informationen auslesen, aber nichts hineinschreiben. Er enthält die Grundprogramme, die nach dem Einschalten des Computers sofort selbsttätig starten (z. B. Programme zur Tastaturabfrage, zum Aussenden des Kontrollbildes u. ä.). Dieser Speicher hat im KC 85/5 einen Umfang von 48 KByte (Betriebssystem, BASIC und Anwendungsprogramme). Mit der Flash-Erweiterung des KC 85/5+ stehen jedem der 8 Betriebssysteme 64 KByte ROM zur Verfügung – der CAOS-ROM ist dabei zweimal vorhanden.

Im RAM, dem Arbeitsspeicher, befinden sich alle Programme und Daten, die Sie von einem Speichergerät einlesen oder mit der Tastatur eingeben. Aus diesem RAM können Sie sowohl lesen als auch Informationen hineinschreiben. Beim Ausschalten des Computers gehen jedoch sämtliche Daten dieses Speicherbereiches verloren. Der RAM hat einen Umfang von 256 KByte. Diese 256 KByte stehen jedoch nicht uneingeschränkt zur Verfügung, da auch RAM-Speicherplätze für die Grundprogramme des Betriebssystems CAOS benötigt werden.
Die Speicher sind in Segmente eingeteilt, die z. B. über das Betriebssystemkommando SWITCH mit Steuerbytes ein- oder ausgeschaltet werden können. Eine Übersicht der Speicheraufteilung des KC 85/5 befindet sich im Bild 24 auf Seite 122.

1.3.5. Farb- und Grafikspeicher

Der Bildwiederholspeicher (IRM) des KC 85/5 besitzt einen Speicherumfang von 64 KByte. Zur Darstellung auf dem Bildschirm enthält er 2 Bildspeicher (Bild 0, Bild 1). Zu jedem Bild gehören ein Farb-, Pixel- und ASCII-Speicher.

Für die Farbinformation von 8 nebeneinander liegenden Bildpunkten ist im Farbspeicher ein Byte reserviert. Es enthält die Vorder- und Hintergrundfarbwerte.
Sie können für die Farbgestaltung zwischen 16 Vordergrundfarben und 8 Hintergrundfarben wählen. Im Betriebssystemmenü ist es bereits möglich, die Farbauswahl über COLOR zu treffen.
Mit der ESC-Steuerfunktion und den Buchstaben 'A' oder 'B' kann eine punktweise Farbauflösung eingestellt werden (hochauflösende Farbgrafik). Dabei sind jedoch nur 4 Farben zur Bildschirmgestaltung möglich. Das Blinken kann ebenfalls nicht verwendet werden.

Der IRM ist unterteilt in 4 Ebenen zu je 16 KByte und wird im Adressbereich ab 8000H eingeblendet (siehe Bild 24 auf Seite 122). Der Pixel- oder Farbspeicher umfasst den Adressbereich von 8000H bis A7FFH für 320*256 Bildpunkte. Im Pixel-RAM von Bild 0 liegen ab Adresse A800H Arbeitszellen, die unabhängig vom Schaltzustand der IRM-Ebenen sichtbar sind, damit die Programme immer auf diese Arbeitszellen Zugriff haben.

Die Speicherbereiche der anderen drei IRM-Ebenen werden als „versteckte“ IRM-Bereiche bezeichnet, auf diese kann nur unter speziellen Bedingungen zugegriffen werden. Details dazu finden Sie in Kapitel 3.3.2 (Verwalten des KC-internen Speichers) auf Seite 126.

1.4. Joystick und Joysticktreiber

Mit dem Einsatz eines Joysticks kann der Bedienkomfort für viele Spiel- und Anwenderprogramme wesentlich erhöht werden. Dazu ist eines der Module M008 oder M021 erforderlich. Diese Module ermöglichen den Anschluss eines Joysticks an den KC 85/5. Der Betrieb von zwei Joysticks bzw. zwei Joystick-Modulen ist nicht möglich!

Ab CAOS 4.5 ist ein Joysticktreiber enthalten, welcher die Module M008 bzw. M021 direkt unterstützt. Ist eines dieser Modultypen im KC-System vorhanden, dann wird der Joysticktreiber automatisch initialisiert. Die Joystickabfrage arbeitet interruptgesteuert. Da Zeitgeber-Interrupts nur während der Benutzung des Joysticks freigeschaltet werden, wird die normale Programmabarbeitung nicht ausgebremst. Betätigungen des Joysticks werden vom Treiber in Tastencodes umgesetzt. Ab CAOS 4.6 ist außerdem ein komfortabler Editor für die Joystickfunktionen im Betriebssystem integriert.

Der in CAOS enthaltene Joysticktreiber erzeugt dabei Tastencodes, die wie eine Tastatureingabe behandelt werden. Die Funktion der Tastencodes wird durch eine 12 Byte große Tabelle gesteuert, deren Beginn in der Arbeitszelle JOYTAB (B7F0H) im IRM abgelegt ist. Diese Joysticktabelle selbst hat folgenden Aufbau:

Tabelle 4: Joysticktabelle

Adresse	Vorbelegung	Funktion	
(JOYTAB)+0	02H	Wartezyklen für Tastenwiederholung (Autorepeat) – 02H entspricht etwa Tastaturverhalten	
(JOYTAB)+1	0BH	Up	nach oben
(JOYTAB)+2	0AH	Down	nach unten
(JOYTAB)+3	0DH	Fire+Fire2	beide Feuertasten gedrückt
(JOYTAB)+4	08H	Left	nach links
(JOYTAB)+5	00H	Up+Left	nach links-oben
(JOYTAB)+6	00H	Down+Left	nach links-unten
(JOYTAB)+7	0DH	Fire	Feuertaste 1
(JOYTAB)+8	09H	Right	nach rechts
(JOYTAB)+9	00H	Up+Right	nach rechts-oben
(JOYTAB)+10	00H	Down+Right	nach rechts-unten
(JOYTAB)+11	20H	Fire2	Feuertaste 2

Hinweise:

- Fire2 ist die primäre, meist auch die einzige Feuertaste.
- Die diagonalen Bewegungsrichtungen sind nicht mit einem Tastencode vorbelegt.
- Zum Anschluss des Joysticks siehe Seite 112.
- Zum Editieren der Joystick-Einstellungen steht ein komfortabler Editor bereit, siehe Seite 49.
- Zur Nutzung des Joysticks unter BASIC siehe Seite 340.

1.5. Kommandos des Betriebssystems KC-CAOS

Erst durch das Betriebssystems CAOS ist ein Arbeiten mit dem Kleincomputer möglich. Es enthält Programme zur Steuerung der angeschlossenen Geräte.

1.5.1. Das Menü

Die Arbeit mit dem CAOS-Betriebssystem erfolgt über die Tastatur mit der auf dem Fernsehgerät bzw. Monitor dargestellten Menütabelle. Ein Menüwort repräsentiert jeweils ein Programm bzw. Kommando. Die im Grundmenü dargestellten Kommandos sind im Betriebssystem enthalten. Als Anwender können Sie das Menü durch eigene Kommandos (Maschinenprogramme) erweitern, siehe Kapitel 3.4. Seite 128.

Das auszuführende Kommando kann mit dem Cursor angewählt oder nochmals unter dem Menü eingegeben werden. Es ist darauf zu achten, dass sich in der Zeile, in der jetzt der Cursor steht, kein anderes Zeichen außer dem Promptzeichen '%', gefolgt von dem Kommando befindet. Die Betätigung der Taste 

erzeugt bei Bedarf eine leere Eingabezeile in der Zeile, wo sich der Cursor gerade befindet. Nicht sichtbare Menüworte können nur eingegeben werden.

Jedem Programm können nach seinem Menüwort bis zu 10 Parameter übergeben werden. Vom Menüwort und untereinander werden die Parameter durch Leerzeichen getrennt.

Bei der Eingabe eines Menüwortes am KC 85/5 ist es nicht erforderlich, den vollständigen Namen auszuschreiben. Es müssen nur so viele Zeichen eingegeben werden, wie zur eindeutigen Identifizierung des Wortes notwendig sind (auch bei Parameterübergabe). Menüworte können wahlweise in Klein- oder Großbuchstaben oder gemischt eingegeben werden. Um klein geschriebene (nicht sichtbare) Menüworte zu erreichen, muss jedoch das Kommando zwingend in Kleinbuchstaben eingegeben werden.
Bei mehrfach vorkommenden und damit nicht eindeutigen Menüworten wird das im Menü zuerst stehende Wort erkannt. Das muss besonders beim Erstellen eigener Programme mit Menüworten beachtet werden!

Beispiel:

%L <ENTER>	entspricht	%LOAD <ENTER>
%K 1 <ENTER>	entspricht	%KEY 1 <ENTER>

Mit Betätigung der Taste  wird das Kommando ausgeführt. Bei einer falschen Eingabe, z. B. eines Menüwortes, das nicht im Betriebssystem enthalten ist, erscheint eine Fehleranzeige – siehe Kapitel 3.4.3. Seite 130.

1. AUFBAU UND BEDIENUNG

In der folgenden Übersicht werden die im Grundmenü enthaltenen Unterprogramme mit dem Hinweis auf eine ausführliche Beschreibung kurz erläutert. In **grüner Schrift** ist dabei die kürzeste noch eindeutige Schreibweise dargestellt. Menüworte in Kleinbuchstaben oder mit angehängtem Leerzeichen sind nicht sichtbar.

Menüworte im Grundmenü	Bedeutung	Handbuch
%**B**ASIC	Kaltstart des BASIC-Interpreters	Seite 254
%**R**EBASIC	Warmstart des BASIC-Interpreters	Seite 254
%**S**WITCH	Module oder Speicherbereiche schalten, gesteckte Module und Schaltzustände anzeigen	Seite 71
%**J**UMP	Sprung in ein anderes Betriebssystem	Seite 76
%**M**ENU	Aufruf des aktuellen Menüs	Seite 45
%**S**AVE	Ausgabe von Programmdateien	Seite 57
%**V**ERIFY	Kontrolllesen von Programmdateien (bei Magnetbandaufzeichnungen)	Seite 58
%**L**OAD	Laden von Programmdateien	Seite 53
%**C**OLOR	Anzeigen und Festlegung der Vorder- und Hintergrundfarbe	Seite 63
%**D**ISPLAY	Anzeige von Speicherbereichen	Seite 78
%**MO**DIFY	Speicheranzeige und Veränderung	Seite 77
%**W**INDOW	Einstellen eines Fensters	Seite 62
%**K**EY	Funktionstastenbelegung programmieren und auflisten	Seite 47
%**LS**TDEV	Druckertreiber initialisieren	Seite 80
%**V2**4DUP	V.24-Duplexroutine initialisieren zur Datenübertragung	Seite 84
%**DE**VICE	Anzeige und Auswahl der Speichergeräte	Seite 51

Weitere Menüworte von Kommandos sind in Speicherebenen enthalten, die nicht ständig eingeblendet sind (CAOS-ROM-C und die 4 Ebenen des USER-ROM). Diese Menüworte werden erst sichtbar, wenn die entsprechende Speicherebene mit SWITCH eingeschaltet wird. Die Menüworte können aber trotzdem eingegeben und gestartet werden. CAOS durchsucht diese Systemebenen automatisch, falls das eingegebene Menüwort im sichtbaren Speicher nicht enthalten ist. Wird ein gleich lautendes Menüwort allerdings im gerade sichtbaren Speicher gefunden, dann wird dieses gestartet. Die **grün** geschriebenen Abkürzungen gelten hier also nur, falls kein anderes Menüwort dazu passt.

Zusätzliche Menüworte	Bedeutung	Handbuch
Im CAOS-ROM-E:		
%**G**O	Start eines MC-Programms	Seite 79

Zusätzliche Menüworte	Bedeutung	Handbuch
%**DIR**	Verzeichnis anzeigen	Seite 59
%**CD**	Laufwerk wählen	Seite 52
%**E**RA	Datei löschen *2	Seite 60
%**REN**	Datei umbenennen *2	Seite 60
Im CAOS-ROM-C:		
%**vi**ew	Speicherschnellansicht	Seite 78
%**T**YPE	Anzeige einer Textdatei *2	Seite 61
%**DU**MP	HEX/ASCII-Dump einer Datei *2	Seite 61
%**I**NIT	Abarbeitung Datei INITIAL.UUU *2	Seite 59
%HELP	Anzeige CAOS-Versionsdatum und Auflistung aller verfügbaren Kommandos	Seite 46
Im USER-ROM:	**(Debugger-Ebene)**	
%**DISA**SS	Speicherinhalt direkt reassemblieren	Seite 412
%**Q**MR	Quelltext aus Speicherinhalt erzeugen	Seite 413
%**TE**MO	Kaltstart Testmonitor	Seite 420
%**RET**EMO	Warmstart Testmonitor	Seite 420
%**ST**ACK	Anzeige des System-Stacks	Seite 85
%**JE**DIT	Joystick-Editor	Seite 49
%**?**c	Taschenrechner CALC	Seite 85
%**?i**	CALC auf Drucktaste installieren	Seite 85
%**BS**AVE	BASIC-Programm als KCB-Datei speichern	Seite 362
Im USER-ROM:	**(Assembler-Ebene)**	
%**A**SM	Kaltstart Assembler	Seite 376
%**REA**SM	Warmstart Assembler	Seite 376
%**SETRO**	Schreibschutz einer Datei setzen *3	Seite 61
%**SETW**R	Schreibschutz einer Datei aufheben *3	Seite 61
%**T**IME	Anzeige von Datum und Uhrzeit *3	Seite 85
%**P**RINT	Zeichenausgabe zu Drucker	Seite 83
Im USER-ROM:	**(Editor-Ebene)**	
%**E**DIT	Kaltstart Editor	Seite 428
%**REE**DIT	Warmstart Editor	Seite 428

*2 Kommando mit Kassette nicht nutzbar
*3 Kommando nur für D004 oder D008

CAOS-Kommando MENU

%MENU [n]

Die Ausführung des Kommandos MENU bewirkt das Löschen des Bildschirmes und das Auflisten des aktuellen Menüs. Die Ausgabe kann mit der Taste BRK abgebrochen werden. MENU kann mit einem optionalen Parameter zusätzliche Informationen anzeigen, möglich sind folgende Werte:

%MENU 1	zusätzliche Anzeige der versteckten Menüworte
%MENU 2	zusätzliche Anzeige der Startadresse zu jedem Menüwort
%MENU 3	Anzeige mit versteckten Menüworten und Startadressen

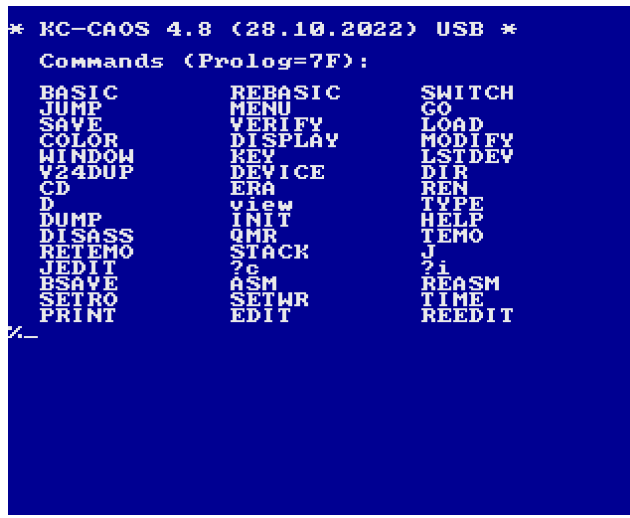
Bild 7: CAOS-Menü mit Parameter 3 aufgerufen

CAOS-Kommando HELP

%HELP

Seit CAOS 4.4 kann durch das Kommando %help (in Kleinbuchstaben und deshalb im Menü nicht sichtbar) das Erstellungsdatum der CAOS-Version angezeigt werden, z. B.:

Ab CAOS 4.8 kann %HELP auch in Großbuchstaben eingegeben werden. Es werden zusätzlich alle Menüworte aufgelistet, die sich im aktuell eingeschalteten Speicherbereich sowie den CAOS-ROM-Ebenen befinden. So erhält man einen schnellen Überblick über alle Menüworte, die man ohne vorherigen SWITCH-Befehl direkt eingeben und ausführen kann.



1.5.2. Funktionstasten und Joystick

CAOS-Kommando KEY

Das Kommando KEY kann in drei Formaten benutzt werden:

1. Format: Anzeige der belegten Funktionstasten

%KEY

Wird KEY ohne Parameter aufgerufen, dann werden alle belegten Funktionstasten aufgelistet. Das bis CAOS 4.2 vorhandene Menüwort KEYLIST wurde damit ersetzt. Ist keine Funktionstaste belegt, erhält man die Anzeige:

Beispielanzeige von mit BASIC-Funktionen belegten Funktionstasten:

2. Format: Löschen aller belegten Funktionstasten

%KEY 0

Der Parameter 0 veranlasst das Löschen aller belegten Funktionstasten.

3. Format: Belegen bzw. Bearbeiten einer Funktionstaste

%KEY n (n = 1…F)

Diese Kommandoform von KEY dient zur Belegung bzw. Bearbeitung der Funktionstasten F1 bis F6, deren Zweitbelegung F7 bis FC und der zusätzlichen Tastencodes FD bis FF. Mit dem hexadezimal anzugebenden Parameter n wird die Nummer der zu bearbeitenden Funktionstaste 1 bis 15 ausgewählt. Ist die Funktionstaste bereits belegt, ist deren Belegung änderbar.

Die Tastencodes FD bis FF sind mit der Standard-KC-Tastatur nicht erreichbar, können aber von einer externen Tastatur verwendet werden, indem diese Zeichencodes von der Tastatur gesendet werden.

Als Tastenbelegung können alle Zeichen, auch die Steuerzeichen (mit Ausnahme der Taste (STOP)) programmiert werden. Die Summe aller Tastenbelegungen darf 143 Zeichen (bzw. 140 Zeichen bei Nutzung der 3 zusätzlichen Funktionstasten FD bis FF) nicht übersteigen.

Bei der Belegung der F-Tasten gibt es zwei Betriebsarten: den Editiermodus und den Interpretiermodus, zu Beginn des Kommandos befindet man sich im Interpretiermodus.

– Beim Interpretiermodus werden die Steuertasten mit ihren Symbolen dargestellt und der Funktionstaste zugefügt.
– Beim Editiermodus sind die Steuertasten normal wirksam und können zum Editieren der Eingabe verwendet werden.

Mit (STOP) wird der Modus gewechselt. Im Editiermodus wird die Eingabe mit (↵) beendet oder mit (BRK) abgebrochen, ohne den neuen F-Tasteninhalt zu aktivieren! Eine Besonderheit gibt es zu beachten: Der Bildschirm darf nicht scrollen, da so die Eingabezeile aus dem Video-RAM nicht mehr zurückgelesen werden kann!

Beispiel:

Die Taste F2 soll mit dem Befehl RUN und Enter belegt werden, so ist einzugeben:

Eingabe	**Bildschirmanzeige**	**Bemerkung**
1.) KEY	%KEY_	
2.) <Leerzeichen>	%KEY _	
3.) 2	%KEY 2_	
4.) <Enter>	F2 :_	Kommandoabschluss
5.) RUN	F2 :RUN_	
6.) <Enter>	F2 :RUN↵_	
7.) <Stop>	F2 :RUN↵_	Wechsel zu Editiermodus
8.) <Enter>	%_	

CAOS-Kommando JEDIT

Eine der Neuerungen ab CAOS 4.5 ist der integrierte Joystick-Treiber. Mit JEDIT steht ab CAOS 4.6 ein Joystick-Editor für diesen Treiber zur Verfügung.

Aufruf:

%JEDIT [Argumentliste | 0]

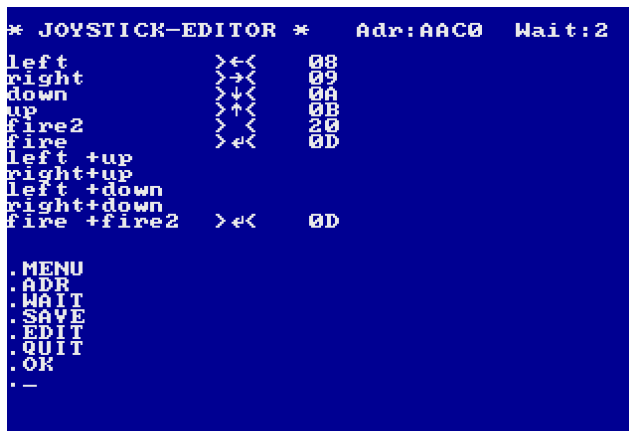
JEDIT zeigt die Tastenbelegung an und springt in ein eigenes Menü mit dem Punkt als Promptzeichen. Beim Aufruf können die Tasten als Parameter übergeben werden. Den Tastencode kann man als 8Bit-Hexzahl oder als Zeichen mit vorangestelltem Komma angeben. JEDIT übernimmt die Codes von oben nach unten in die eigene Liste, also in der Reihenfolge:

links, rechts, unten, oben, Feuer2, Feuer, links-oben, rechts-oben, links-unten, rechts-unten, Feuer+Feuer2

Es müssen nicht alle 11 Codes angegeben werden. Tasten bzw. Tastenkombinationen, deren Code nicht angegeben wurde, werden deaktiviert (auf Null gesetzt).

Eine Besonderheit ist der Aufruf %JEDIT 0, JEDIT lädt dann die Standardwerte.

Das Menü von JEDIT



Alle in JEDIT getroffenen Änderungen werden zunächst temporär gespeichert. Erst nach Aufruf von **.OK** werden die Änderungen übernommen und man gelangt zurück ins CAOS. Durch Aufruf von **.QUIT** lassen sich die Änderungen verwerfen und JEDIT wird abgebrochen.

.EDIT

Aufruf:

.EDIT [Argumentliste | 0]

Wird .EDIT ohne Parameter aufgerufen, gelangt man in den Joystick-Editor. Dieser fragt der Reihe nach alle Tastencodes ab. Mit Ausnahme von STOP können alle Tasten verwendet werden, auch die Steuertasten und Funktionstasten.
Nach Drücken von STOP oder nach Eingabe des 11. Codes (Fire+Fire2) wird der Editor beendet.
Die Eingabe bestimmter Tasten kann übersprungen bzw. wiederholt werden, indem man am Joystick den Steuerknüppel bewegt oder eine Feuertaste drückt. Der Editor springt dann zur entsprechenden Position.

Wird .EDIT mit Parameter aufgerufen, werden genau wie bei %JEDIT die Tastencodes übernommen oder die Standardwerte geladen.

.SAVE

.SAVE speichert die Joysticktabelle ab. Die erzeugte Datei ist selbststartend. Diese kopiert beim Laden die Tabelle an die in der Arbeitszelle JOYTAB (B7F0H) vorgegebene Adresse.
Die Datei wird aus den temporären Daten von JEDIT erzeugt. Somit ist ein späterer Aufruf von .QUIT problemlos möglich.

.WAIT

Aufruf:

.WAIT nn

.WAIT ändert die Anzahl der Wartezyklen, bis der Autorepeat der Tastencodes einsetzt. Diese müssen als Dezimalzahl im Bereich 0 bis 63 angeben werden. Standardwert ist 2 und entspricht in etwa der Repeat-Folge der KC-Tastatur.

.ADR

Aufruf:

.ADR nnnn

.ADR ändert die Adresse der Joysticktabelle.
.ADR prüft die Adresse auf Gültigkeit. So wird verhindert, dass die Joysticktabelle im Stack, dem IX-Arbeitsbereich, der SUTAB, der DEVICE-Treiber-Tabelle, dem PIXEL-, Monitor-RAM oder im ROM-Bereich angelegt wird.

.OK

Beim Aufruf von .OK wird die Tabelle an dieser Adresse erzeugt und in JOYTAB (B7F0H) eingetragen.

Beispiele:	**Joystick-Tasten einiger Spiele**
Gourmand	10 11 13 12 0D 0D
Pursuit	10 11 13 12 0D 0D
Schiessbude	59 2F 00 00 0D 20
Pegasus5	2E 2F 00 00 59 59
Revolution	08 09 0A 0B 59 58
Flammendes Inferno	59 2F 0A
Kiste	59 2F
Zahn (angreifende Gebisse)	,J ,K ,M ,I 0D 0D

1.5.3. Arbeit mit Speichergeräten (Kassette, Diskette, USB)

CAOS-Kommando DEVICE

Bis CAOS 4.5 war die Magnetbandkassette das einzige direkt unterstützte Speichermedium. Für die Arbeit mit Diskette waren Hilfsprogramme wie FLOAD, BASEX, DEVEX usw. erforderlich. Mit CAOS 4.6 wurde ein Treibersystem eingeführt, welches bis zu 8 verschiedene Speichergeräte verwalten und umschalten kann. Bei CAOS 4.6 war standardmäßig TAPE für Kassette und DISK für D004 integriert. Seit CAOS 4.7 werden nun zusätzlich noch Treiber aus externen Modulen wie M052-USB eingebunden. Das aktuelle Speichergerät wird im CAOS-Menü in der Titelzeile angezeigt. Wird das Kommando:

%DEVICE [n]

ohne zusätzlichen Parameter n aufgerufen, dann listet es alle verfügbaren Speichergeräte und deren zugeordnete Nummern auf. DEVICE 0=TAPE ist immer vorhanden und entspricht dem Kassettenbetrieb der früheren CAOS-Versionen – das ist sozusagen der Kompatibilitätsmodus. Ist ein D004 bzw. D008 im KC 85-System vorhanden und befindet sich dieses in der CAOS-Betriebsart, dann gibt es ein weiteres DEVICE mit der Bezeichnung DISK. CAOS-Betriebsart heißt, im D004 läuft das Programm DEP (Disketten-Erweiterungs-Programm) und steuert die Prozesse im D004. Für jedes gesteckte Modul M052 mit USB-Software Version 2.5 (besser 3.0) oder höher, gibt es dann noch ein DEVICE mit der Bezeichnung USB.
Die Reihenfolge der Speichergeräte ist:

0 = TAPE
DISK, wenn D004/D008 in CAOS-Betriebsart
USB-Module in der Reihenfolge der Steckplätze

Zur Ausführung der DEVICE-Funktionen wird das jeweilige Modul von CAOS eigenständig zu- und auch wieder abgeschaltet.

! ACHTUNG! Der Modulzugriff funktioniert nur, wenn kein höher priorisiertes Speichermodul auf Adresse C000H eingeschaltet ist. Deshalb empfiehlt es sich, das USB-Modul vor anderen Speichermodulen anzuordnen!

Gibt es außer TAPE noch weitere Speichergeräte im KC 85-System, dann wird das erste dieser Speichergeräte beim Systemstart automatisch aktiviert.

Das mit dem Kommando DEVICE eingestellte Speichergerät wirkt auf alle CAOS-Unterprogramme, die bisher für die Kassettenein- und -ausgabe zuständig waren. Somit gilt das DEVICE nicht nur für die CAOS-Kommandos wie LOAD und SAVE, sondern auch für Anwenderprogramme, einschließlich BASIC, ASM und EDIT.

CAOS-Kommando CD

%CD [verzeichnis]

Mit dem Kommando CD kann das Verzeichnis eines Speichergerätes gewählt werden. Bei Kassette bewirkt CD abwechselnd einen Start bzw. Stopp des Recorders.

Bei Diskette können Laufwerk und USER-Bereich gewählt werden. Wurde in der Kommandozeile nichts angegeben, dann erhält man das zurzeit aktive Laufwerk angezeigt. Mit (BRK) kann hier abgebrochen werden. Ansonsten ist jetzt die Eingabe des neuen Laufwerkes (A-P), gefolgt von einem USER-Bereich (0-15) möglich. Zwischen Laufwerk und User darf kein Leerzeichen stehen. Die Funktion arbeitet nur, wenn im D004 eine DEP-Version ab 2.0 läuft, ab DEP-Version 3.5 sind auch USER-Bereiche bis 31 zugelassen. Ab CAOS 4.8 erfolgt die Anzeige und Eingabe des USER-Bereichs als Dezimalzahl, in älteren CAOS-Versionen als einstellige Hex-Zahl 0-F. Ab CAOS 4.8 kann auch nur der USER-Bereich gewechselt werden, ohne das Laufwerk zu ändern.

Beispiele:

%CD	Anzeige des aktuellen Laufwerks/User-Bereichs
%CD C	wechselt zu Laufwerk C, der User-Bereich bleibt erhalten
%CD A15	wechselt zu User-Bereich 15 in Laufwerk A
%CD 5	wechselt zu User-Bereich 5 des aktuellen Laufwerks

Bei USB wird in das Verzeichnis gewechselt, welches direkt nach CD in der Kommandozeile angegeben wurde. Ohne Angabe des Verzeichnisses erhält man eine Auflistung der Unterverzeichnisse und danach eine Eingabeaufforderung.
Ab USB-Treiber-Version 3.0 kann auch ein kompletter Pfad angegeben werden.

Beispiele:

%CD CAOS	wechselt in das Unterverzeichnis CAOS
%CD ..	wechselt vom Unterverzeichnis eine Ebene höher
%CD /	wechselt in das Hauptverzeichnis
%CD /BASIC/SPIELE	absolute Pfadangabe ab dem Hauptverzeichnis
%CD CAOS/SUBDIR	relativer Pfad ab dem aktuellen Verzeichnis

CAOS-Kommando LOAD

%LOAD [nnnn [s]]

Möchten Sie auf Ihrem Kleincomputer ein Maschinenprogramm nutzen, das auf Magnetbandkassette oder einem anderen unterstützten Speichergerät gespeichert ist, so ist dieses in den Computer zu laden. Dazu wird das Kommando LOAD genutzt. Mit LOAD können Maschinenprogramme bis zu einer Größe von 31,5 KByte in den Computer eingelesen werden. Das entspricht dem Speicherbereich von 200H bis 7FFFH. Ab CAOS 4.8 (12.09.2021) können auch größere Dateien eingelesen werden, zum Beispiel bis Adresse BFFFH, wenn der IRM aus- und der RAM8 eingeschaltet ist.

Falls Sie das Programm von Kassette einlesen möchten, stellen sie zunächst das DEVICE TAPE ein. Spulen Sie nun das Magnetband an den Anfang des Programms, welches Sie nutzen möchten. Der in der vom Hersteller mitgelieferten Programmbeschreibung angegebene Zählerstand des Programmanfangs ist ein Richtwert. Den exakten Zählerstand müssen Sie für Ihr Gerät selbst ermitteln, da die Recorderzählwerke von Gerät zu Gerät differieren. Den Programmanfang erkennen Sie am Programmvorton. Dieser ist ein deutlicher Pfeifton.
Schalten Sie nun Ihren Recorder zur Wiedergabe ein und drücken Sie während des Pfeiftons die Taste  (rechts unten auf der Tastatur). Mit Betätigen dieser Taste wird das Kommando, auf das der Cursor weist (in unserem Fall LOAD), ausgeführt. Der Computer entschlüsselt und speichert die am Tonbandanschluss (TAPE-Buchse) ankommenden Signale als Computerdaten.

Hinweis:

BASIC-Programme werden nicht mit dem CAOS-Kommando LOAD, sondern mit den entsprechenden BASIC-Anweisungen geladen, siehe Kapitel 4.1. Bei selbststartenden BASIC-Programmen vom KC 85/3 kann es beim Start Probleme geben. In diesem Fall kann über Zuschalten des BASIC-Interpreters (SWITCH 2 1) vor dem Laden versucht werden, das Programm zu starten.
Kommen nach dem Vorton die Daten, so könnte auf dem Bildschirm z. B. folgendes Bild entstehen:

```
%LOAD
TEST 0200 0400
02>
```

Anhand dieses Bildes können Sie den Ladevorgang auf dem Bildschirm verfolgen. Nachdem das LOAD-Kommando zur Ausführung gebracht wurde, erscheint als Erstes der eingelesene Programmname (im Beispiel TEST). Ihm folgen die Anfangs- und die Endadresse des Programms (im Beispiel 200 bzw. 400) als hexadezimale Zahlen. Bei selbst startenden Programmen wird seit CAOS 4.3 noch als dritter Wert die Startadresse angezeigt. Nun werden die Blocknummern der eingelesenen Blöcke des Programms angezeigt. Ein Block besteht aus 128 Byte. Der erste Block 01 enthält den Programmnamen und wird als einziger nicht angezeigt. Der letzte Block hat, unabhängig von der Länge des Programms, stets die Blocknummer FF. Der Winkel hinter jeder Blocknummer zeigt als Kontrollzeichen die fehlerfreie Übernahme des eingelesenen Blockes an. Taucht nach den Blocknummern der Cursor wieder auf dem Bildschirm auf, so ist der Ladevorgang beendet oder das Programm startet.
Befindet sich ein Datenfehler im eingelesenen Block, erscheint als Kontrollzeichen ein '?' anstelle des Winkels hinter der entsprechenden Blocknummer. Hinter dem '?' erscheint der Cursor und wartet auf eine Eingabe. Mit beliebiger Tastenbetätigung (außer BRK und ⇩) kann der fehlerhafte Block erneut gelesen werden. Dazu spulen Sie das Magnetband um mindestens einen Block zurück und starten den Lesevorgang neu. Falls ein anderer Block, als der erwartete gelesen wird, so zeigt der Computer die Blocknummer mit einem nachfolgenden '*' an. Dies erleichtert das Finden des fehlerhaften Blocks. Sobald der fehlerhafte Block richtig (fehlerfrei) eingelesen wurde, erscheint hinter der Blocknummer wieder der Winkel.
Der Ladevorgang wird nun normal fortgesetzt. Kann der Block nach einem oder mehreren Versuchen nicht gelesen werden, ist es möglich, nach der Fehleranzeige und Drücken von ⇩ den Block fehlerhaft in den Speicher zu übernehmen. Mit dem Kommando MODIFY können Sie die Fehler nach dem Einlesen beseitigen, wenn Ihnen der Inhalt des Programms bekannt ist. Der Ladevorgang kann jederzeit mit BRK abgebrochen werden.

Soll ein Programm nicht auf die Anfangsadresse geladen werden, mit der es gespeichert wurde, so besteht die Möglichkeit, die Anfangsadresse durch den Parameter nnnn zu verändern. Dabei ergibt sich nnnn als Differenz bzw. Offset aus der Anfangsadresse, auf die das Programm geladen werden soll, und der gespeicherten Anfangsadresse.
Ist ein Programm z. B. mit der Anfangsadresse 0200 gespeichert worden und soll auf die Anfangsadresse 0900 geladen werden, so ist der Parameter nnnn mit 700 anzugeben:

```
%LOAD 700
```

Vergessen Sie das Leerzeichen zwischen Anweisung und Parameter nicht! Befehlsausführung erfolgt wie üblich erst durch die Betätigung von ↵.

Der zweite mögliche Parameter s dient dazu, den Start eines selbst startenden Programms zu unterbinden. Dabei ist der Wert des Parameters unerheblich. Soll das Programm ohne Ladeoffset geladen werden, dann muss der erste Parameter 0 sein:

```
%LOAD 0 0
```

Die Darstellung der Blocknummern untereinander (siehe folgende Tabelle) ist nur hier in der Beschreibung so gewählt, auf dem Bildschirm sind die fehlerhaften Blöcke (Ausschrift Blocknummer Stern) auf einer Stelle. Nachfolgende Tabelle hilft, Ladefehler zu erkennen und zu beseitigen.

Tabelle 5: Mögliche Ursachen für Lesefehler

Fehler	**Ursache**	**Beseitigung**
1. Es erscheint kein Programmname, aber unregelmäßige Blocknummern 01* 02* 03* 00* 05*	1.1. Pegel vom Recorder zu gering	- vgl. technische Daten des Recorders - anderen Recorder verwenden - Recorder überprüfen lassen, siehe Tabelle 1, Einschaltfehler Seite 24
	1.2. Brücke im Diodenkabel oder im Recorder	- Verbindungskabel zum Recorder prüfen
2. Es erscheint kein Programmname, aber Blocknummern wie im Beispiel 02* 03* 04*	2.1. erster Block nicht gefunden.	- zurückspulen - bei wiederholtem Fehler ist die Aufzeichnung fehlerhaft
3. Bei LOAD erscheint der Programmname und dahinter die Ausschrift „Keine MC-Datei!“ (bis CAOS 4.2 drei Fragezeichen hinter dem Dateinamen) z. B. SSS SPIEL ???	3.1. Es wurde versucht, eine Datei zu laden, welche kein Maschinenprogramm ist. (z. B. BASIC-Quellprogramm)	- Suchen des richtigen Programms

Fehler	Ursache	Beseitigung
4. Hinter einer Blocknummer erscheint ein Fragezeichen z. B. 07?	4.1. Datenfehler im mit Fragezeichen gekennzeichneten Block (Knitterstellen, Drop Out) 4.2. Zufälliger Fehler (Schalten eines elektrischen Gerätes; Gleichlaufschwankungen oder schlechter Band-Kopf-Kontakt durch mechanische Erschütterungen)	- zurückspulen des Bandes und betätigen einer beliebigen Taste (außer BRK und ⇩ und Wiederholung des Einlesens des fehlerhaften Blockes - Bei wiederholtem Fehler liegt ein Fehler in der Aufzeichnung vor. In diesem Fall kann der Block durch Drücken von ⇩ ggf. fehlerhaft übernommen und im MODIFY-Modus korrigiert werden
5. Es lassen sich nur Programme einlesen, welche auf demselben Recorder aufgezeichnet wurden	5.1. Falsche Tonkopfeinstellung am Recorder	- Recorder überprüfen lassen

Falls Sie das Programm von einem anderen Speichergerät als Kassette einlesen möchten, stellen sie zunächst das DEVICE entsprechend ein. Nach Eingabe des Kommandos LOAD folgt jetzt die Abfrage eines Dateinamens. Geben Sie hier bitte den Namen der Datei an, den Sie einlesen möchten. Der Dateiname kann bis zu 8 Zeichen lang sein, werden weitere Zeichen angegeben, dann werden die nächsten 3 als Dateityp gewertet. Zwischen Dateiname und Dateityp kann als Trennzeichen sowohl ein Punkt als auch ein Leerzeichen stehen.
Ab CAOS 4.8 und Treiber-Version 3.0 kann bei USB dem Dateinamen auch eine Pfadangabe vorangestellt werden. Dann wird zunächst der Pfad eingestellt und danach die Datei geladen.

Beispiele für Dateinamen:

%LOAD

Name: SPIEL	lädt die Datei SPIEL.KCC
Name: SPIEL.KCB	lädt die Datei SPIEL.KCB
Name: SPIEL KCC	lädt die Datei SPIEL.KCC
Name: 12345678KCC	lädt die Datei 12345678.KCC
Name: /SPIEL	lädt die Datei SPIEL.KCC aus dem Hauptverzeichnis
Name: /SPIELE/GO	lädt die Datei GO.KCC aus dem Verzeichnis SPIELE

Name: SUBDIR/GAME lädt die Datei GAME.KCC aus dem Unterverzeichnis SUBDIR
Name: /CAOS/SPIELE/KC4/HIBI.KCC
auch die Angabe des kompletten Pfades ist möglich

Im Fehlerfall, wenn also beispielsweise die Datei auf dem Speichergerät nicht vorhanden ist, wird nach einer Fehlermeldung sofort abgebrochen.

Alles andere gilt gleichermaßen wie bei Kassette.

CAOS-Kommando SAVE

%SAVE aaaa eeee [ssss [v]]

Mit diesem Kommando kann man Programme und Daten aus dem Computer auf ein externes Speichergerät, zum Beispiel Magnetbandspeicher retten (abspeichern). Stellen Sie mit dem Kommando DEVICE zunächst das gewünschte Speichergerät ein. Beim Kommando SAVE sind die Anfangsadresse aaaa und die Endadresse eeee (beinhaltet die nachfolgende Adresse der vom Programm oder der Datei belegten Speicherzelle) des zu rettenden Speicherbereiches als Parameter anzugeben. Soll das abzuspeichernde Programm selbst startend sein, so muss eine Startadresse ssss als dritter Parameter angegeben werden.

Wird ein vierter Parameter v (Wert beliebig) angegeben, so wird die Startadresse beim Offset-Einlesen nicht umgerechnet. Die Parameter aaaa und eeee sind in jedem Fall, die Parameter ssss und v nur bei Bedarf anzugeben.
Soll z. B. ein Programm mit der Startadresse 2100H, welches im Arbeitsspeicher den Adressbereich 2000H bis 2300H belegt, auf Magnetband gespeichert werden, so sind folgende Eingaben direkt hintereinander auszuführen:

Eingabe	Bildschirmanzeige
1.) SAVE	%SAVE_
2.) Leerzeichen	%SAVE _
3.) 2000	%SAVE 2000_
4.) Leerzeichen	%SAVE 2000 _
5.) 2300	%SAVE 2000 2300_
6.) Leerzeichen	%SAVE 2000 2300 _
7.) 2100	%SAVE 2000 2300 2100_
8.) <ENTER>-Taste	NAME: _

Das so auf dem Bildschirm entstehende, syntaktisch fehlerfreie Kommando 'SAVE 2000 2300 2100' wird durch die Betätigung der Taste  ausgeführt. Dabei erscheint vorerst nur das Wort 'Name:' auf dem Bildschirm. Sie können nun dem auszugebenden Programm einen Namen mit maximal 11 Zeichen geben. Dieser Name wird mit abgespeichert und sowohl beim Kontrolllesen einer Magnetbandaufzeichnung (VERIFY) als auch beim Laden (LOAD) wieder zur

Anzeige gebracht. Die Ausgabe des Speicherinhaltes wird auf dem Bildschirm durch Anzeige der Blocknummern (Blöcke zu 128 Byte) protokolliert. Die Blocknummern sind Hexadezimalzahlen. Es ist ratsam, den Programmanfang auf Magnetband vor der Aufnahme durch den Zählerstand oder akustisch zu kennzeichnen.
Sind diese Vorbereitungen alle getroffen, so werden zur Ausgabe des Programms der Recorder auf Aufnahme geschaltet und die Taste  betätigt.

Falls Sie das Programm auf ein anderes Speichergerät als Kassette ausgeben, wird eine Datei mit dem eingegebenen Namen angelegt. Dabei stehen 8 Zeichen für den Dateinamen und 3 für den Dateityp zur Verfügung. Ab CAOS 4.8 und Treiber-Version 3.0 kann bei USB dem Dateinamen auch hier eine Pfadangabe vorangestellt werden. Dann wird zunächst der Pfad eingestellt und danach die Datei dort abgespeichert. Die bei LOAD angegebenen Beispiele für Dateinamen gelten auch für SAVE. Wenn auf dem Speichergerät bereits eine Datei unter dem Namen existiert, erhalten Sie eine Abfrage, ob diese überschrieben werden soll.
Im Fehlerfall, wenn also beispielsweise der Datenträger voll ist oder ein Schreibfehler erkannt worden ist, wird nach einer Fehlermeldung sofort abgebrochen.

Alles andere gilt gleichermaßen wie bei Kassette.

! Die Kommandos FLOAD und FSAVE zum Laden und Speichern auf Diskette sind seit CAOS 4.6 nicht mehr enthalten, stattdessen kann jetzt SAVE und LOAD mit DEVICE=DISK benutzt werden.

CAOS-Kommando VERIFY

%VERIFY

Magnetbandaufzeichnungen (Maschinenprogramme, Daten, BASIC-Programme usw.) können mit dem Kommando VERIFY überprüft werden. Dazu wird das Magnetband an den Programmanfang zurückgespult, danach der Recorder auf Wiedergabe eingeschaltet und die Anweisung VERIFY durch Betätigung der Taste  ausgeführt.
Auf der Anzeige erscheinen der Programmname, die Blocknummern der verglichenen Blöcke und die dazugehörigen Kontrollzeichen (> , ?, * vgl. LOAD).
Bei fehlerlosem Einlesen der Aufzeichnung erscheint nach jeder Blocknummer das Zeichen '>'. Die eventuell auftretenden anderen Kontrollzeichen und die entsprechende Fehlerursache sind in der Tabelle 5 zum Kommando LOAD zusammengefasst.

Das Kommando VERIFY kann nur bei DEVICE=TAPE aufgerufen werden, bei allen anderen Speichergeräten hat es keine Wirkung.

CAOS-Kommando DIR

%DIR [maske]

Mit dem Kommando DIR können Sie sich das Inhaltsverzeichnis eines Speichergerätes anzeigen lassen.

Ist als Speichergerät Kassette eingestellt, dann kann der Inhalt einer gesamten Kassette angezeigt werden. Dazu ist die Kassette nach Eingabe des DIR-Kommandos komplett abzuspielen. Es wird dazu immer nur der Inhalt des ersten Blockes aufgelistet mit Dateiname sowie den Adressen, falls es sich um einen gültigen CAOS-Vorblock handelt. Eine eventuell angegebene Maske hat bei Kassette keine Wirkung auf das Kommando. Das Einlesen kann mit der Taste BRK abgebrochen werden, dazu muss ein Signal vom Kassettenrecorder anliegen.
Tipp: Noch einmal zur letzten Datei zurückspulen und dann BRK drücken.

Ist ein anderes Speichergerät (z. B. Diskette) eingestellt, dann kann durch Angabe einer Maske eine Auswahl der gewünschten Dateien getroffen werden. Möglich sind dabei die Sonderzeichen '*' für einen beliebigen Teil des Dateinamens und '?' für ein beliebiges Zeichen. Der Punkt zur Trennung von Name und Typ ist nicht anzugeben, da der Dateiname wie eine Zeichenkette mit 11 Byte Länge behandelt wird. Schreibgeschützte Dateien auf Diskette werden mit einem Stern nach dem Dateinamen gekennzeichnet.
Ab CAOS 4.8 und Treiber-Version 3.0 kann bei USB der Maske noch eine Pfadangabe vorangestellt werden. Dann wird zunächst der Pfad eingestellt und danach die Dateien aufgelistet.

Beispiele:

%DIR *	alle Dateien (Angabe des * kann entfallen)
%DIR *KCC	Dateien mit Dateityp 'KCC'
%DIR a*	Dateien, die mit 'A' beginnen
%DIR *EX*	alle Dateinamen, die die Zeichenfolge 'EX' enthalten
%DIR /BASIC/*SSS	alle BASIC-Programme im Verzeichnis BASIC

CAOS-Kommando INIT

%INIT [dateiname]

Einlesen und starten einer CAOS-Kommandodatei vom aktuell eingestellten Speichergerät (außer Kassette). Ohne Angabe eines Dateinamen wird die Standard-Datei INITIAL.UUU vom aktuell eingestellten Speichergerät/Laufwerk/Pfad geladen. Wird ein Dateiname angegeben, dann kann ab CAOS 4.8 und Treiber-Version 3.0 bei USB dem Dateinamen optional eine Pfadangabe vorangestellt werden. Dann wird zunächst der Pfad eingestellt und danach die Datei geladen. Die eingelesenen Daten werden vom Kassettenpuffer temporär nach A880H-A8FFH ko-

piert und dort wie eine Funktionstaste abgearbeitet. Die Datei darf maximal einen Block, also 128 Zeichen groß sein.

! Hinweis: Der IRM-Bereich von B700H bis B732H (Kassettenpuffer) wird vom DIR-Kommando als Arbeitsspeicher benutzt. Deshalb konnte es bei CAOS 4.3 bis 4.5 zu Problemen kommen, wenn innerhalb der Datei INITIAL.UUU ein DIR ausgeführt wurde, da die Kommandos aus der INITIAL.UUU direkt aus dem Kassettenpuffer abgearbeitet wurden.
In CAOS 4.6 werden die INIT-Kommandos auf Adresse 0100H, ab CAOS 4.7 im IRM ab Adresse A880H abgearbeitet, um den System-RAM davon zu entlasten. %INIT funktioniert bei CAOS 4.7 nur mit DEVICE=DISK

CAOS-Kommando REN

%REN [oldname newname]

Mit diesem Kommando kann man eine Datei umbenennen, der alte und neue Dateiname wird angefordert, falls dieser nicht direkt in der Kommandozeile angegeben wird. Die Dateinamen sind vollständig (ohne Jokerzeichen) einzugeben.
Ab CAOS 4.8 und Treiber-Version 3.0 kann bei USB dem ersten Dateinamen optional eine Pfadangabe vorangestellt werden. Dann wird zunächst der Pfad eingestellt und danach die Datei umbenannt.
Bei Kassette als Speichergerät hat REN keine Wirkung und führt zu einer Fehlermeldung.

CAOS-Kommando ERA

%ERA [name]

Das Kommando ERA gestattet Dateien auf einem Speichergerät zu löschen.
Bei Diskette darf der eingegebene Dateiname '?' als Joker enthalten, dann werden alle Dateien gelöscht, die der Maske entsprechen!
Bei USB muss der Dateiname eindeutig sein, es kann immer nur eine Datei gelöscht werden. Ab CAOS 4.8 und Treiber-Version 3.0 kann dem Dateinamen optional eine Pfadangabe vorangestellt werden. Dann wird zunächst der Pfad eingestellt und danach die Datei gelöscht.
Bei Kassette als Speichergerät hat ERA keine Wirkung und führt zu einer Fehlermeldung.

Für das Kommando ERA gilt wie auch für SETRO, SETWR, TYPE und DUMP:

Der Dateiname kann im Anschluss an das Menüwort in der Kommandozeile angegeben werden. Erfolgte keine Angabe, dann wird der Dateiname angefordert.

CAOS-Kommando SETRO

%SETRO [name]

Das Kommando SETRO ist nur für Diskette verfügbar und wirkt auch auf die Diskette, wenn ein anderes Speichergerät eingestellt ist. Mit SETRO wird der Schreibschutz einer Datei gesetzt. Schreibgeschützte Dateien lassen sich nicht überschreiben oder löschen.

CAOS-Kommando SETWR

%SETWR [name]

Das Kommando SETWR ist nur für Diskette verfügbar und wirkt auch auf die Diskette, wenn ein anderes Speichergerät eingestellt ist. Das Kommando SETWR hebt den Schreibschutz einer Datei wieder auf.

CAOS-Kommando TYPE

%TYPE [name]

TYPE dient der Anzeige einer Textdatei. Das Programm wartet jeweils am Bildschirmende, fortgesetzt wird mit jeder beliebigen Taste außer BRK (Abbruch) und Shift-CLR (HARDCOPY). Bei Erkennung eines Ende-Codes 03H oder 1AH wird der Anzeigevorgang beendet.
Ab CAOS 4.8 und Treiber-Version 3.0 kann dem Dateinamen optional eine Pfadangabe vorangestellt werden. Dann wird zunächst der Pfad eingestellt und danach die Datei angezeigt.
TYPE ist für Kassette nicht sinnvoll und führt deshalb zu einer Fehlermeldung.

CAOS-Kommando DUMP

%DUMP [name]

DUMP dient der hexadezimalen Anzeige einer beliebigen Datei als HEX/ASCII-Dump. Am Dateiende wird mit der Meldung „Dateiende überschritten“ abgebrochen, falls nicht vorher mit BRK der Anzeigevorgang beendet wurde.
Ab CAOS 4.8 und Treiber-Version 3.0 kann dem Dateinamen optional eine Pfadangabe vorangestellt werden. Dann wird zunächst der Pfad eingestellt und danach die Datei angezeigt.
DUMP ist für Kassette nicht sinnvoll und führt deshalb zu einer Fehlermeldung.

1.5.4. Beeinflussen der Bildschirmausgabe

CAOS-Kommando WINDOW

Das Kommando WINDOW kann in drei Formaten benutzt werden:

1. Format: Definieren eines Fensters

%WINDOW ersteZeile letzteZeile ersteSpalte letzteSpalte [Nummer]

Durch WINDOW ist es möglich, vom CAOS-Menü aus ein Fenster einzustellen. Anzugeben sind ab CAOS 4.8 jeweils die erste und letzte Zeile (Bereich 0 bis 31) sowie die erste und letzte Spalte (Bereich 0 bis 39) als Dezimalzahlen (analog dem gleichnamigen BASIC-Befehl). Das optionale fünfte Argument definiert die Fensternummer (0 bis 9). Ohne Angabe wird Fenster Nr. 0 definiert.
Bis CAOS 4.7 sind dem Kommando WINDOW die Argumente als Hex-Zahlen mit der Bedeutung za (1. Zeile), zn (Zeilenanzahl), sa (1. Spalte) und sn (Spaltenanzahl) anzugeben.

2. Format: Wiederaufrufen eines definierten Fensters

%WINDOW nr

Mit der Eingabe von WINDOW und der Fensternummer wird das Fenster mit der angegebenen Nummer aufgerufen.

3. Format: Einstellung der Standardwerte

%WINDOW

Wird WINDOW ganz ohne Argumente aufgerufen, dann werden seit CAOS 4.5 die CAOS-Standardwerte für das Fenster aktiviert, das sind: Fenster Nr. 0, volle Bildschirmgröße mit 40*32 Zeichen, Schriftfarbe weiß auf blau und Scroll-Mode.
Bei CAOS 3.4 bis 4.4 wird ohne Argumente das Fenster Nr. 0 aktiviert.

Beispiel:

%WINDOW 0 31 0 39 1 (bis CAOS 4.7: %WINDOW 0 20 0 28 1)

Im Beispiel wird das Fenster Nr. 1 mit folgenden Parametern eingestellt:
Von Zeile 0 bis 31 und von Spalte 0 bis 39 – also das gesamte Bild.

%WINDOW 5 8 3 11 (bis CAOS 4.7: %WINDOW 5 4 3 9)

Im Beispiel wird das Fenster Nr. 0 mit folgenden Parametern eingestellt:
Von Zeile 5 bis 8 (vier Zeilen lang), von Spalte 3 bis 11 (neun Zeichen breit).

CAOS-Kommando COLOR

%COLOR [fv [fh]]

Das Kommando COLOR beeinflusst die Farbe der Zeichenausgabe. Bis CAOS 4.7 sind die Farbcodes 10 bis 15 hexadezimal einzugeben als A bis F, ab CAOS 4.8 dezimal von 0 bis 15. Weiterhin ist zwischen LORES- und HRG-Modus zu unterscheiden. Es gibt drei Kommandoformen:

– ohne Parameter werden die Zahlenwerte der aktuellen Einstellung angezeigt.
– bei nur einem Parameter wird nur die Vordergrundfarbe fv festgelegt.
– werden beide Parameter angegeben, dann legt der erste Parameter fv die Vordergrundfarbe und der zweite Parameter fh die Hintergrundfarbe fest.

Im LORES-Modus sind die 16 Vordergrund- und 8 Hintergrundfarben codiert.

Tabelle 6: Vorder- und Hintergrundfarben im LoRes-Modus

Farbe	Vordergrund	Hintergrund	Farbe
schwarz	0	0	schwarz
blau	1	1	blau
rot	2	2	rot
purpur	3	3	purpur
grün	4	4	grün
türkis	5	5	türkis
gelb	6	6	gelb
weiß	7	7	grau
schwarz	8		
violett	9		
orange	10		
purpurrot	11		
grünblau	12		
blaugrün	13		
gelbgrün	14		
weiß	15		

Die Hintergrundfarben erscheinen eine Nuance dunkler als die Vordergrundfarben. Die mit dem Kommando COLOR festgelegte Farbkombination bezieht sich immer auf das jeweils eingestellte Fenster und Bild.

Im LORES-Modus besteht die Möglichkeit, Vordergrundfarben auf dem Bildschirm blinkend darzustellen. Dazu wird der entsprechende Farbcode mit 16 addiert. Möchten Sie z. B. die Farbkombination gelb blinkende Vordergrundfarbe auf rotem Hintergrund realisieren, so geben Sie direkt hintereinander ein:

Eingabe	Bildschirmausgabe
1.) COLOR	%COLOR_
2.) Leerzeichen	%COLOR _
3.) 22	%COLOR 22_
4.) Leerzeichen	%COLOR 22 _
5.) 2	%COLOR 22 2_

Durch Drücken der Taste  wird der Farbcode gespeichert und alle folgenden Bildschirmausgaben erscheinen in der gewünschten Farbkombination auf dem Bildschirm. Im obigen Beispiel finden Sie den Vordergrundparameter 22 und den Hintergrundparameter 2. Der Hintergrundparameter 2 (für rot) ist direkt der Tabelle 6 zu entnehmen. Der Vordergrundparameter setzt sich zusammen aus der Farbfestlegung 6 (für gelb) und dem addierten Blink-Code 16 (6+16=22). Soll der Vordergrund nicht blinken, so gilt einfach: COLOR 6 2

Durch CLEAR SCREEN (gleichzeitige Betätigung von  und 

) wird der Bildschirm im eingestellten Fenster gelöscht und erscheint in der aktuell eingestellten Hintergrundfarbe. Nachfolgende Texte werden in der aktuell eingestellten Vordergrundfarbe dargestellt.

Wird mittels ESC-B (siehe Tabelle 3 auf Seite 34) der HRG-Modus eingestellt, dann stehen insgesamt nur noch 4 Farben zur Verfügung. Jede dieser 4 Farben kann dabei Vordergrund- oder Hintergrundfarbe sein. Blinken ist nicht möglich.

Tabelle 7: Die Codierung der 4 Farben im HRG-Modus

Farbe	Vorder- oder Hintergrund
schwarz	0
rot	1
türkis	2
weiß	3

1.5.5. Verwalten und Schalten der Speichersegmente und Module

Eine wesentliche Grundeigenschaft des KC 85/5 ist die hohe Ausbaufähigkeit und Flexibilität des Systems. So können Sie Speicher bis 4 MByte verwalten. Dabei werden mithilfe des Kommandos SWITCH die einzelnen Speicherbereiche und Module zugeschaltet oder vom Prozessor getrennt. Das Kommando SWITCH gibt dem Anwender außerdem einen Überblick des momentanen Zustandes und der Struktur aller im Computersystem befindlichen Speicher und Module.

Das Kommando JUMP erlaubt das Wegschalten des fest installierten und die Nutzung eines anderen Betriebssystems.

Modulsteckplätze, Strukturbytes und Steuerbytes

Zur Unterscheidung besitzt jedes Modul ein charakteristisches Strukturkennbyte, welches vom Hersteller festgelegt und von der Software ausgelesen werden kann.
Angesprochen wird ein Modul über seinen Steckplatz mm, da auch mehrere gleiche Module in einem KC 85-System vorkommen können. Die Steckplätze 0 bis 7 sind den internen „Modulen“, also dem im Grundgerät eingebauten Speicherbereichen zugeordnet.
Zur Steuerung der Module dient ein Steuerbyte kk, welches im Modul selbst und im Modulsteuerbytespeicher abgespeichert wird.

Die Modulsteckplätze sind mit dem Parameter mm so zugeordnet:

Tabelle 8: Übersicht der Steckplätze im KC-System

mm	Speicher im Grundgerät oder Steckplatz
00	RAM auf Adresse 0000H
01	IRM Bildspeicher auf Adresse 8000H
02	USER-ROM-Blöcke auf Adresse C000H
03	RAM-Blöcke auf Adresse 8000H
04	RAM-Blöcke auf Adresse 4000H
05	CAOS-ROM auf Adresse C000H
06 *4	reserviert für interne RAM-Erweiterung
07 *4	reserviert für internes V.24-Modul
08-0B	Modulsteckplatz 8 (Grundgerät rechts) Grundmodul einschließlich 3 weiterer möglicher Submodule
0C-0F	Modulsteckplatz C (Grundgerät links) Grundmodul einschließlich 3 weiterer möglicher Submodule
10 - EF	Die Zuordnung des Parameters mm zu den Steckplätzen im Erweiterungsaufsatz ist der dem Aufsatz beiliegenden Bedienungsanleitung zu entnehmen.
F0-F3	Modulsteckplatz F0 (D004/D008 rechts) Grundmodul einschließlich 3 weiterer möglicher Submodule
F4-F7	Modulsteckplatz F4 (D004/D008 links) Grundmodul einschließlich 3 weiterer möglicher Submodule
FC	D004/D008: Interner ROM-Bereich

*4 Die Steckplatzadressen 06 und 07 sind für geplante Erweiterungen reserviert.

Modul- und Interruptprioritätskette

Werden mehrere Module mit gleicher Speicher- oder E/A-Adressen eingeschaltet, so darf beim Zugriff des Prozessors nur das Speicher- oder E/A-Modul auf der niedrigsten Modulsteckplatzadresse aktiv sein.

Zu diesem Zweck sind die Module in einer Modulprioritätskette zusammengeschaltet, welche über die Signalleitungen MEI (Modul Enable Input) und MEO (Modul Enable Output) gesteuert wird und im Grundgerät beginnt. Im Grundgerät besitzt der Modulsteckplatz 8 (rechter Steckplatz) gegenüber dem Modulsteckplatz C (linker Steckplatz) die höhere Priorität. Ein logisches 1 auf der MEI-Leitung signalisiert Modulfreigabe. Ein angesprochenes Modul höchster Priorität schaltet seine MEO-Leitung auf logisch 0 und sperrt damit alle nachfolgenden Module im Falle einer gleichen Adressierung.

Parallel zur Modulprioritätskette verfügt das KC-System über eine Interruptprioritätskette für die E/A-Module. Diese Kette wird durch die Signalleitungen IEI (Interrupt Enable Input) und IEO (Interrupt Enable Output) gebildet. Auch diese Prioritätskette beginnt im Grundgerät und wird über die Modulsteckplätze im Grundgerät an die Modulsteckplätze in den Erweiterungsaufsätzen weitergegeben. Die Kette darf nicht unterbrochen sein. Bei Speichermodulen sind die Signalleitungen IEI und IEO deshalb gebrückt.

Die Interruptprioritätskette ermöglicht hardwareseitig die Priorisierung von Unterbrechungsanforderungen, welche durch E/A-Module über die Signalleitung /INT an den Prozessor gerichtet sind. Mit IEI = 1 ist die Kette freigegeben. Bei IEI = 0 befindet sich ein höherpriorisiertes E/A-Modul in der Interruptbearbeitung. Tritt in einem E/A-Modul ein Interruptereignis ein, das durch den Prozessor quittiert wurde, und liegt die IEI-Leitung auf logisch 1, so schaltet das Modul seine IEO-Leitung auf logisch 0. Damit werden alle nachfolgend im Interrupt befindlichen E/A-Module blockiert bzw. auftretende Unterbrechungsereignisse zeitweise ignoriert.

! Bei der Bestückung des KC 85-Systems mit Erweiterungsmodulen ist auf eine lückenlose Steckfolge vom Steckplatz 8 aufsteigend zu achten, damit die Modulprioritätskette und die Interruptprioritätskette geschlossen sind, ansonsten können Fehlfunktionen auftreten!

Siehe auch die noch etwas ausführlicheren Ausführungen in der Beschreibung zum Modul M005 [64].

Interne Module

Ist die Steckplatzadresse mm kleiner als acht, so handelt es sich um Speichersegmente im Grundgerät. Diese können ebenfalls zu- und abgeschaltet werden. Besteht ein Speicher aus mehreren Segmenten, dann haben die einzelnen Bits des Steuerbytes kk ebenfalls noch eine Bedeutung.

Siehe dazu auch die Speicherübersicht in Bild 24 auf Seite 122.

RAM0 (mm = 00)

x	x	x	x	x	x	wr	on

Im RAM0 liegen die IX-Arbeitszellen von CAOS, der Systemstack und die Interrupttabelle. Das Abschalten des RAM0 darf nur erfolgen, wenn die Arbeitszellen mit dem Systemprogramm SIXD vorher in einen anderen Bereich verlagert wurden. Siehe dazu Kapitel 3.6.4 Seite 183.

IRM (mm = 01)

x	x	x	x	x	x	x	on

Das Schalten des Bildspeichers mit dem Menüwort SWITCH nimmt eine Sonderstellung ein. Die Arbeitszellen (Monitor-RAM) von CAOS liegen im IRM, dieser muss deshalb ständig eingeschaltet sein.
Der mit dem Kommando SWITCH vorgegebene Schaltzustand wird nur im Modulsteuerbytespeicher eingetragen und nicht sofort wirksam. Erst wenn ein Speicherzugriff erfolgt, z. B. bei LOAD, SAVE, MODIFY oder DISPLAY wird der Schaltzustand des IRM kurz wirksam. Das ermöglicht die Nutzung der RAM8-Ebenen, so als ob der IRM tatsächlich ausgeschaltet wäre.
Der IRM hat 4 Segmente, diese werden jedoch nicht im Steuerbyte angegeben, da Bild, Farb- und Pixelebene automatisch bzw. mittels der ESC-Sequenzen gesteuert werden. Nur Bit 0 ist zum Ein- bzw. Ausschalten des IRM definiert.

! ACHTUNG! Das Schalten des IRM mit dem Unterprogramm 26H (MODU) wird im Gegensatz zum CAOS-Kommando SWITCH sofort wirksam!

USER-ROM (mm = 02)

1	1	s	s	x	x	x	on

Der USER-ROM enthält insgesamt 4 Segmente, die im Steuerbyte anzugeben sind. Die Segmentnummer muss dual verschlüsselt in Bit 4 und Bit 5 eingetragen werden. Folgende ROM-Inhalte sind in CAOS 4.8 enthalten:

- Segment 0 (SWITCH 2 C1): BASIC-Interpreter
- Segment 1 (SWITCH 2 D1): KC Debugger (Testmonitor, Reassembler)
- Segment 2 (SWITCH 2 E1): Assembler ASM *5
- Segment 3 (SWITCH 2 F1): Editor EDIT *5

Die CAOS-Menüworte BASIC und REBASIC schalten die richtige Ebene automatisch ein. Menüworte aus den anderen 3 Ebenen können ebenfalls aufgerufen werden, ohne dass das entsprechende Segment vorher zugeschaltet sein muss. ASM kann also z. B. sofort aufgerufen werden, wenn vorher gerade mit dem Reassembler gearbeitet wurde.
Beim KC 85/3 und KC 85/4 gab es lediglich eine Ebene für den BASIC-ROM, welche mit SWTCH 2 0/1 geschaltet wird. Der KC 85/2 besitzt keinen ROM an dieser Stelle.

RAM8-Blöcke (mm=03)

x	x	s	s	s	s	wr	on

Der RAM8 besitzt beim KC 85/5 insgesamt 14 Ebenen. Um 14 Ebenen kodieren zu können, werden 4 Bit benötigt. Die Segmentnummer muss dual verschlüsselt von Bit 2 bis Bit 5 eingetragen werden. Außer den Ebenen 0 bis 13 können auch noch die Ebenen 14 und 15 angegeben werden, wobei die Ebene 14 dem RAM0 und die Ebene 15 dem RAM4 entspricht, welche dann gleichzeitig noch im RAM8 sichtbar sind! Dies kann für besondere Anwendungen recht nützlich sein, ist aber im Falle des RAM0 mit Vorsicht zu genießen, da dort meist die Systemarbeitszellen liegen.
Der KC 85/4 hat nur 2 RAM8-Ebenen, welche mit Bit 2 geschaltet werden.

Beispiel:

Soll das RAM8-Segment 1 schreibgeschützt sein, muss für kk folgendes Bitmuster eingegeben werden: 0000 0101. Die erforderliche Eingabe ist also:

SWITCH 03 05

---

*5 Für CAOS 4.7 stehen zwei Varianten des USER-ROM zur Verfügung:
- Standard: mit 80-Zeichen-Editor EDIT und Assembler ASM 2.0
- Alternativ mit EDAS 1.7 (Editor+Assembler) und FORTH 3.1

RAM4 (mm=04)

x	x	x	x	x	s	wr	on

Seit CAOS 4.3 besitzt der RAM4 2 Ebenen. Die Segmentnummer muss in Bit 2 eingetragen werden. Die zweite Ebene ist eine „virtuelle“ Ebene und wird aus dem normalerweise nicht zugänglichen versteckten Bereichen des IRM gebildet. Beim Wechsel der RAM4-Ebene wird der Inhalt des RAM4 mit den entsprechenden IRM-Bereichen ausgetauscht, sodass dieser sich wie eine zweite echte RAM-Ebene verhält. Der Umschaltvorgang dauert aber wesentlich länger! Beim Einschalten des Rechners wird Ebene 0 aktiviert, bei RESET bleibt die gerade eingestellte Ebene erhalten.

Folgende Zuordnung gilt zwischen den Adressen des RAM4 und des IRM:

RAM4 Adressbereich	IRM Adressbereich	Ebene
4000-4FFF	A800-B7FF	Bild 0 Color-IRM
5000-67FF	A800-BFFF	Bild 1 Pixel-IRM
6800-7FFF	A800-BFFF	Bild 1 Color-IRM

Werden die versteckten IRM-Bereiche von Anwenderprogrammen benutzt, dann sollte die zweite RAM4-Ebene nicht genutzt werden, um Konflikte zu vermeiden.

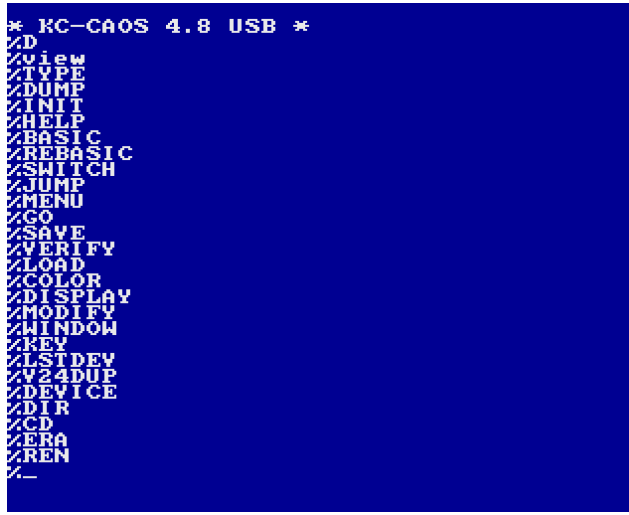
Hinweis: Der Bereich von B800H bis BFFFH im Color-IRM von Bild 0 wird seit CAOS 4.6 vom Rechner CALC genutzt.

CAOS-ROM-C (mm = 05)

x	x	x	s	x	x	x	on

Der CAOS-ROM auf der Adresse C000H bis DFFFH wird von CAOS nur bei Bedarf ein- und danach automatisch wieder ausgeschaltet. Das ist der Grundzustand beim Einschalten des KC 85/5. Soll der CAOS-ROM-C nicht ausgeblendet werden, dann kann das durch SWITCH 5 1 eingestellt werden. Dadurch werden alle CAOS-Menüworte sichtbar, die sich in diesem Speicherbereich befinden.

Der CAOS-ROM-C hat die höchste Priorität, ist er eingeblendet, dann sind keine anderen Speicherbereiche auf dem Adressbereich C000H-DFFFH mehr sichtbar. Im folgenden Bild sind mit %MENU 1 auch die sonst versteckten Menüworte des CAOS-ROM-C mit angezeigt worden.



Die Funktionalität zum dauerhaften Einschalten des CAOS-ROM-C steht seit CAOS 4.3 zur Verfügung.

Beim KC 85/5+ mit FLASH-Erweiterung kann zwischen zwei Ebenen des gesamten CAOS-ROM umgeschaltet werden. Die Inhalte dieser beiden ROM-Ebenen sind identisch bis auf den Zeichensatz. In CAOS 4.8 wurde im Steuerbyte für den CAOS-ROM das Bit 4 = „s“ neu definiert. Damit kann zwischen den beiden ROM-Ebenen gewechselt werden und man erhält wahlweise den dünnen oder den normalen Zeichensatz. Siehe dazu die abgebildeten Zeichensätze ab Seite 203. Ohne die Flash-Erweiterung hat das Bit 4 keine Wirkung.

%SWITCH 5 10	schaltet den alternativen Zeichensatz ein
%SWITCH 5 0	schaltet in den ersten Zeichensatz zurück

CAOS-Kommando SWITCH

Das Kommando SWITCH kann in drei Formaten benutzt werden.

1. Modulübersicht
2. Schaltzustand eines Moduls anzeigen
3. Modul schalten

1. Format: Modulübersicht

Ohne weitere Parameter erhält man eine Übersicht über alle vorhandenen Module und deren Schaltzustände.

Beispiel:

```
%SWITCH
00 FF 03 RAM0 ON
10 F4 C3 16K RAM
FC A7 04 FLOPPY
```

Modulbezeichnung
Steuerbyte
Strukturbyte
Steckplatz

Es werden für alle Module der Steckplatz, das Strukturbyte, das Steuerbyte und falls bekannt, die Modulbezeichnung ausgegeben. Bei den internen Modulen wird zusätzlich noch der Schaltzustand angezeigt, dabei bedeuten:

ON	Modul ist eingeschaltet
ON*	Modul ist schreibgeschützt eingeschaltet
OFF	Modul ist ausgeschaltet

SWITCH ohne Parameter ersetzt das bei CAOS 4.2 vorhandene Menüwort MODUL und das bis CAOS 4.5 vorhandene Kommando SYSTEM.

Für die internen Module gilt folgende Zuordnung:

```
- -   CAOSE     ON/OFF   . . . Betriebssystem ROM auf Adresse E000H
00    RAM0      ON/OFF   . . . RAM auf Adresse 0000H
01    Bild 0/1  ON/OFF   . . . Bild 0 oder 1 ein- bzw. ausgeschaltet
02    USER n    ON/OFF   . . . USER-ROM-Segment n auf Adresse C000H
                                 n = 0 ist BASIC
                                 n = 1 ist Debugger (REAS, TEMO, …)
                                 n = 2 ist Assembler ASM
                                 n = 3 ist Editor EDIT
03    RAM8 n    ON/OFF   . . . RAM-Segment n auf Adresse 8000H
                                 n = 0 ist RAM8-Block 0
                                 . . .
                                 n = D ist RAM8-Block 13
04    RAM4 n    ON/OFF   . . . RAM auf Adresse 4000H
                                 n = 0 ist RAM4-Block 0
                                 n = 1 ist RAM4-Block 1
05    CAOSC     ON/OFF   . . . Betriebssystem ROM auf Adresse C000H
```

Folgende Erweiterungsmodule werden von CAOS 4.8 am Strukturbyte erkannt:

Tabelle 9: bekannte Erweiterungsmodule

Strukturbyte	Bezeichnung (Anzeige)	Bemerkung zum Modultyp
- - *6	JOY/CEN	M008 oder M021
00H	VARIABEL	z. B. M042 Menü/Treiber
01H	START-ROM	z. B. M033 Typestar oder M041
70H	32K ROM	M045
71H	64K ROM	M046
72H	128K ROM	M047
73H	256K ROM	M048 *7
74H	512K ROM	M049 (A. Schlechte)
76H	4M ROM	M044 (in Entwicklung)
77H	64K RAM	M032 (mit 64K-Bestückung) *7
78H	128K RAM	M036
79H	256K RAM	M032 (mit 256K-Bestückung)
7AH	512K RAM	M034
7BH	1M RAM	M035 oder M035x4 (KC-Club)
A7H	FLOPPY	D004 oder D008
BFH	MT02	Prüfmodul (MT01-Nachbau)

*6 Modul ohne Strukturbyte, Erkennung an PIO auf Adresse 91H

Strukturbyte	Bezeichnung (Anzeige)	Bemerkung zum Modultyp
C0H…D7H	USER	Anwendermodule
D9H	EPROMER 32K	M030 (für 2K-32K EPROMs)
DAH	PIO-3 (bis CAOS 4.7) WEATHER (ab CAOS 4.8)	M002 (PIO-3) *7 M004 (Wetter / SRN)
DBH	EPROMER 64K	M030 (KC-Club, 1K-64K EPROMs)
DCH	SOUND	M066 (KC-Club)
DFH	MP3	M366 (SRN)
E3H	DAU1	M029
E7H	ADU1	M010
ECH	SCANNER	M051 (KC-Club)
EEH	V.24	M003 oder M053
EFH	DIGITAL I/O	M001
F0H	8K CMOS-RAM	M120 *7
F1H	16K CMOS-RAM	M122 *7 oder M041
F2H	32K CMOS-RAM	M124 *7
F3H	4*8K RAM 8*8K ROM	M062 mit RAM (E. Mueller) M062 mit ROM (E. Mueller)
F4H	16K RAM	M022
F5H	32K RAM	M024 *7
F6H	64K RAM	M011
F7H	8K ROM	M025 oder M040
F8H	16K ROM	M028, M040 oder M041
F9H	GIDE	M064 (in Planung)
FBH	8K SOFTWARE	M012 TEXOR *8 M026 FORTH *8 M027 EDAS (Development) *8
FCH	BASIC	M006 (für KC 85/2) oder M041
FDH	NET+USB	M052 mit Netzwerk (KC-Club)
FDH	USB	M052 ohne Netzwerk (KC-Club)
FFH	RAM, IRM, ROM	Internes Modul

*7 Diese Module kamen offenbar nicht in den Handel

*8 Bei diesen Modulen wird das erste im ROM gefundene Menüwort angezeigt. Wird kein Menüwort gefunden, dann ist die Anzeige „SOFTWARE“.

2. Format: Schaltzustand eines Moduls anzeigen

%SWITCH mm

Wird SWITCH nur mit dem Parameter mm aufgerufen, erfolgt die Anzeige des momentanen Zustands des sich im Steckplatz mm befindlichen Moduls bzw. internen Speichers. Das aktuelle Steuerbyte und damit der Schaltzustand des Moduls bleiben erhalten.

Beispiel:

Eingabe	Bildschirmausgabe
1.) SWITCH	%SWITCH_
2.) Leerzeichen	%SWITCH _
3.) 8	%SWITCH 8_
4.) <ENTER>-Taste	08 FB C1
	%_

08 – Steckplatzadresse

FB – Strukturbyte: gibt den Modultyp an (z. B. FB = 8K Softwaremodul) Jedes Modul besitzt zur Kennung ein bestimmtes Strukturbyte, das vom Hersteller festgelegt wurde.

C1 – Steuerbyte kk: gibt den Speicherbereich C000H und den Schaltzustand des Moduls an.

Die Ausschrift auf dem Bildschirm gibt an, dass sich im Steckplatz 8 ein Softwaremodul, z. B. TEXOR (Kennung FB), befindet. Dabei belegt dieses den Adressbereich ab C000H und ist schreibgeschützt (1) eingeschaltet.

3. Format: Modul schalten

%SWITCH mm kk

Mithilfe des Parameters kk können verschiedene Zustände für die Module eingestellt werden. Die Bedeutung der einzelnen Bits des Parameters kk ist je nach Modultyp unterschiedlich festgelegt und in der jeweiligen Bedienungsanleitung des Moduls beschrieben. Für die internen Module kann das ab Seite 67 nachgelesen werden.

Beispiel:

Eingabe	Bildschirmausgabe
1.) SWITCH	%SWITCH_
2.) Leerzeichen	%SWITCH _
3.) C	%SWITCH C_
4.) Leerzeichen	%SWITCH C _
5.) C3	%SWITCH C C3_
4.) <ENTER>-Taste	10 F4 00 → C3 %_

10 – Steckplatzadresse: hier der erste Steckplatz in einem Erweiterungsaufsatz

F4 – Strukturbyte: gibt den Modultyp an, hier ein 16K-RAM-Modul M022

00 – Steuerbyte alt: Anzeige des bisherigen Schaltzustandes, hier: AUS

C3 – Steuerbyte kk: Anzeige des neuen Schaltzustandes, hier der Adressbereich C000H mit Schreib-/Lesezugriff

Tabelle 10: Übersicht einiger möglicher Zustände für Speichermodule

Speicherzustand	**kk**
Ausgeschaltet	00
Eingeschaltet und schreibgeschützt (für RAM)	01
Ausgeschaltet / reserviert für eine Anwendung	02
Das Folgende gilt nur für bestimmte Speichermodule:	
Eingeschaltet und nicht schreibgeschützt (Normalbetrieb für RAM)	03
Speicherbereich 4000H eingeschaltet für Schreib-/Lesezugriff	43
Speicherbereich C000H schreibgeschützt eingeschaltet	C1
Speicherbereich C000H eingeschaltet für Schreib-/Lesezugriff	C3

Die erste Zahl des Steuerbytes kk gibt also meist den Adressbereich und die zweite Ziffer den Schaltzustand des Moduls an. Die genaue Bedeutung des Steuerbytes kk ist im Handbuch des jeweiligen Moduls nachzulesen.

CAOS-Kommando JUMP

%JUMP mm

Mit diesem Kommando ist der Sprung in ein anderes Betriebssystem möglich. Der Parameter mm stellt dabei das Sprungziel dar, wo sich das zu startende Betriebssystem befindet. Das kann ein ROM-Modul sein, ein RAM-Modul auf welches das System vorher geladen wurde oder eine der 8 Bänke des Flash-ROM im KC 85/5+. Die Startadresse eines solchen Betriebssystems ist immer die Adresse F012H.

Für mm = 0 gilt bei CAOS 4.7:

Das Kommando JUMP 0 entspricht einem Sprung zu RESET, kann also zum Warmstart des eingebauten CAOS benutzt werden. Auch CAOS 3.4i kennt ein JUMP 0, um in ein parallel eingebautes CAOS 3.1 zurückzuspringen. Dazu muss beim KC 85/3 das Bit 4 von PIO-Port A vom Signal /NMI abgetrennt und zur EPROM-Umschaltung neu verdrahtet werden.

Für mm = [0..7] gilt ab CAOS 4.8:

Sprung in eines der 8 internen Betriebssysteme eines KC 85/5+ mit Flash-ROM-Erweiterung und Softwaresteuerung. Die Belegung der 8 System-Bänke kann frei gewählt werden. Die empfohlene Belegung ist:

0 CAOS 4.8 (Einschaltzustand, auch bei Hardware-RESET)
1 CAOS 4.1 oder CAOS 5.0
2 CAOS 4.2
3 CAOS 4.3
4 CAOS 4.4
5 CAOS 4.5
6 CAOS 4.6
7 CAOS 4.7

Ohne die softwaregesteuerte Flash-ROM-Erweiterung hat JUMP 0-7 bei CAOS 4.8 die gleiche Wirkung wie JUMP 0 unter CAOS 4.7.

Für mm = [8..FC] gilt:

Das Betriebssystem befindet sich auf einem Modul im Modulschacht mm. Hierbei wird der ROM des Grundgerätes abgeschaltet. Das Steuerbyte der JUMP-Anweisung ist FFH (FDH bei CAOS 4.7, um ein RAM-Modul schreibgeschützt einzuschalten).
JUMP schaltet seit CAOS 4.3 alle anderen Speichermodule vorher ab, damit nicht versehentlich in ein höher priorisiertes RAM- oder ROM-Modul gesprungen wird.

! Falls sich auf dem Steckplatz mm kein Modul befindet, erscheint eine Fehlermeldung. Es wird aber nicht überprüft, ob es sich um ein Speichermodul mit gültigem Inhalt handelt.

1.5.6. Gezielter Speicherzugriff

CAOS-Kommando MODIFY

%MODIFY aaaa [n]

Dieses Kommando ermöglicht ein Überprüfen und Verändern des Speicherinhaltes ab der als Parameter einzugebenden Speicheradresse aaaa. Der zweite Parameter n erlaubt das gleichzeitige Anzeigen mehrerer Bytes, wird n weggelassen, dann gilt n=1. Es werden die Adresse und der Speicherinhalt angezeigt. Durch einen Druck auf die Taste  erscheint die jeweils folgende Speicheradresse mit deren Inhalt. Sowohl die Adresse als auch der Inhalt können mit der Tastatur verändert werden.

Durch Betätigung von  wird der angezeigte Wert gespeichert.

Unabhängig von der gewählten Spaltenbreite n ist es möglich, mehr oder weniger Daten in einer Zeile einzugeben. Normalerweise wird der Speicherinhalt als hexadezimaler Maschinencode geschrieben. Darüber hinaus können aber auch direkt ASCII-Zeichen eingegeben werden. Dazu muss vor das entsprechende Zeichen jeweils ein ',' (Komma) gesetzt werden. Sollen ganze Zeichenketten eingegeben werden, sind diese in " ' " (Hochkomma) einzuschließen.

Beispiel:

%MODIFY 200
0200 7F 7F 'TEST' 01 usw.

Um zur vorhergehenden Adresse zurückzugelangen, ist ein ':' einzugeben, die Anzeige geht dann um n Adressen zurück. Soll der MODIFY-Modus ab einer bestimmten Adresse fortgesetzt werden, sind hinter der angezeigten Adresse ein '/' und die neue Adresse einzugeben. Treten Eingabefehler auf, so wird der MODIFY-Modus automatisch mit der Adresse wiederholt, an welcher der Fehler aufgetreten ist.

Die MODIFY-Betriebsart wird durch die Eingabe des Punktes und Drücken der Taste  oder durch Drücken von  beendet.

In der folgenden Tabelle finden Sie die Aktionsmöglichkeiten der MODIFY-Betriebsart noch einmal zusammenfassend dargestellt.

Zeichen / Taste	Funktion
:	Aktuelle Speicheradresse um n verringern
/aaaa	Adresse der gewünschten Speicherzelle
.	Verlassen des MODIFY-Modus
,Z	Code des Zeichens Z (5AH) eingeben
'Zeichenkette'	Die Zeichencodes der Zeichenkette werden übernommen
↵	Übernahme der aktuellen Zeile
BRK	Verlassen des MODIFY-Modus
↓	Übergang zur nächsten Adresse ohne Übernahme einer aktuellen Änderung

CAOS-Kommando DISPLAY

%DISPLAY aaaa [ss [n]]

Das Kommando DISPLAY bewirkt die Ausgabe des Speicherinhaltes ab Adresse aaaa. Dabei werden n Byte bzw., beim Fehlen von n, 8 Byte in einer Zeile als hexadezimale Codes und als ASCII-Zeichen nebeneinander aufgelistet. Es gelangt jeweils die durch den Parameter ss festgelegte Anzahl von Zeilen zur Anzeige. Wird der Parameter ss nicht eingegeben, so werden jeweils vier Zeilen angezeigt. Die Anzeige kann durch Betätigen einer beliebigen Taste, mit Ausnahme der Tasten BRK und STOP fortgesetzt werden.

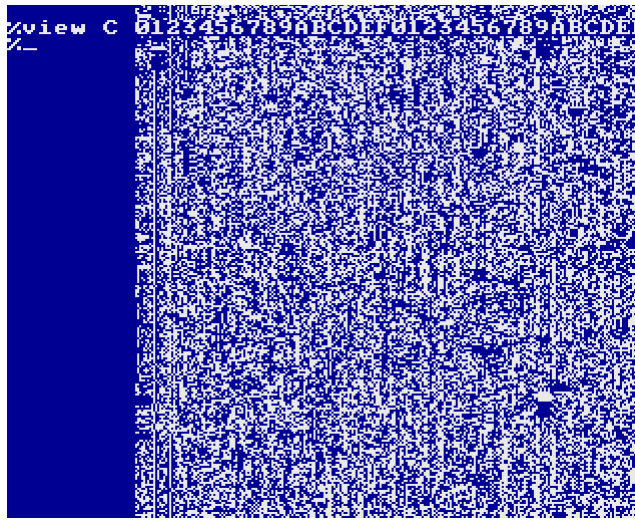
Durch BRK wird das Auflisten beendet. STOP bewirkt den Übergang in den MODIFY-Modus, wobei hier Speicherinhalte nur im Hexadezimalteil geändert werden können und der ASCII-Teil nicht aktualisiert wird. Ansonsten gelten alle Zeichenvereinbarungen wie bei MODIFY beschrieben.

CAOS-Kommando VIEW

%view aaaa | a

Das Kommando view gibt es seit CAOS 4.3, es ist wegen des in Kleinbuchstaben gehaltenen Menüwortes standardmäßig nicht im Menü sichtbar und muss deshalb direkt eingegeben werden. Es ermöglicht eine Art „Speicherschnellansicht“.

Dazu werden 4 KByte Speicherinhalt ab der angegebenen Adresse in den Pixel-RAM des aktuellen Bildes kopiert, dabei wird ein per SWITCH-Befehl ausgeblendeter IRM berücksichtigt, um auch die RAM8-Bereiche darstellen zu können. Die Speicheradresse kann wahlweise komplett im Format 'aaaa' oder auch einstellig 'a' angegeben werden, dann gilt z. B. C für C000:



Das Beispiel zeigt den Inhalt des BASIC-ROM an. Direkt hinter dem Menüwort wird als Orientierungshilfe noch ein Adresslineal „0123…F“ im Bild dargestellt.

CAOS-Kommando GO

%GO ssss [HL [DE [BC [A]]]]

Mit GO kann direkt zur angegebenen Speicheradresse ssss gesprungen werden. Das Kommando ist standardmäßig nicht im Menü sichtbar und kann deshalb nur direkt eingegeben werden. Das Kommando go gibt es ab CAOS 4.3. Erst seit CAOS 4.6 können als weitere Parameter die Registerwerte für HL, DE, BC und A angegeben werden, welche vor Sprung zur Adresse ssss geladen werden. Bis CAOS 4.7 musste GO in Kleinbuchstaben eingegeben werden.

Beispiele:

%go F000	wie POWER on (mit Speicher löschen)
%go E000	wie RESET
%go ssss	Start von MC-Programmen, deren Selbststart beim Laden verhindert wurde, ssss ist hierbei die Startadresse.

1.5.7. V.24-Software und Druckertreiber für V.24 und Centronics

CAOS-Kommando LSTDEV

Das Menüwort LSTDEV vereint seit CAOS 4.5 die beiden bisherigen Menüworte V24OUT (V.24-Druckertreiber) und CEN (Centronics-Druckertreiber in CAOS 4.3 und 4.4). Unterstützt werden die Module M001 (Digital-In-Out), M003/M053 (V.24 bzw. RS232) und M021 (Joy+Centronics). Die Aufruf-Parameter lehnen sich an das bis CAOS 4.4 vorhandene Kommando V24OUT an:

%LSTDEV [mm [k [n [p [d]]]]]

Dabei bedeuten:

- mm - Modulschacht des verwendeten Moduls (8, C,…)
  mm = 0 um ein Modul M021 zu verwenden
- k - Kanal des V.24-Moduls (1 oder 2)
- n - USER-Ausgabekanal (2 oder 3)
- p - Reaktion auf SHIFT-CLEAR
  p = 0 keine Reaktion
  p = 1 Ein- bzw. Ausschalten der Protokollfunktion
  p = 2 HARDCOPY für die Matrixdrucker:
  K 6311/ 12/ 13/ 14/ 27/ 28 bzw.
  SCREENCOPY für die Schreibmaschinen
  S 3004, S 6005/ 09/ 10, S 6120/ 30
- d - Druckertype (siehe Tabelle 11 auf Seite 82)
  Parameter von 0 bis C sind zugelassen.

Steckt auf dem angegebenen Steckplatz keines der unterstützten Module oder steckt kein V.24-Modul im System ohne Angabe eines Steckplatzes, meldet sich der KC mit ERROR.
Nach Aufruf von LSTDEV meldet sich der Cursor bei gestecktem Druckermodul nur dann wieder, wenn:

– ein Drucker am Modul angeschlossen ist und
– das Druckgerät angeschaltet ist und gesendete Zeichen empfängt.

Ist eine Bedingung nicht erfüllt, gelangt man nur durch das Betätigen der <RESET>-Taste am Grundgerät in das Menü zurück.

Wird bei LSTDEV kein Parameter angegeben, wird das KC-System so initialisiert:

– Aktivierung des ersten gefundenen V.24-Moduls
– Zeichenausgabe über Kanal 1 des V.24-Moduls
– Initialisierung des USER-Ausgabekanals 2
– Reaktion auf SHIFT CLEAR: keine Reaktion
  (Protokollfunktion bei CAOS 4.1 bis 4.3)

Diese Standardwerte können durch Angabe der Parameter verändert werden. Dabei gelten folgende Besonderheiten:

- Der Aufruf des Menüwortes LSTDEV ganz ohne Parameter mm sucht das erste verfügbare V.24-Modul (M003 oder M053) ab Steckplatz 7 und stellt den Kanal 1 auf Druckerausgabe ein. Die Einstellung erfolgt auf 9.600 Baud, 1 Stoppbit, 8 Bit pro Zeichen und keine Paritätsprüfung und passt zum Drucker K 6313 und andere.
- Soll das M021 zur Druckerausgabe angesprochen werden, dann muss der Steckplatz mm=0 angegeben werden – unabhängig davon, wo das Modul wirklich steckt.
- Für mm=7…FB wird das Modul des angegebenen Steckplatzes verwendet, sofern das ein M001 oder M003/M053 ist.
- Als zweiter Parameter folgt der SIO-Kanal k, im Falle des M001 oder M021 muss hier ein beliebiger Dummy-Parameter angegeben werden.
- Der dritte Parameter n steht für den USER-Kanal 2 oder 3.
- Der vierte Parameter p steht für die Protokoll- oder Hardcopy-Funktion. Ohne Angabe von p erfolgt keine Reaktion auf Shift-CLR.

Beispiele:

%LSTDEV C 1 2 1 0 - Im Schacht C steckt das V.24-Modul, Kanal 1 des Moduls, USER-Ausgabekanal 2 und Protokollfunktion sind eingestellt. Hier wurde der Drucker K6313 mit dem letzten Parameter festgelegt. Werden nur 4 Parameter angegeben, erfolgt die Festlegung K6313 oder des zuletzt eingestellten Druckgerätes.

%LSTDEV 0 1 2 1 0 - Es befindet sich ein M021 im System auf einem beliebigen Steckplatz. USER-Ausgabekanal 2 und Protokollfunktion sind eingestellt für Drucker K6313.

Protokollfunktion

Nach der Drucker-Initialisierung mit p = 1 kann durch gleichzeitige Betätigung der Tasten  und 

die Protokollfunktion aufgerufen und durch die gleiche Tastenkombination wieder abgeschaltet werden. Die Protokollfunktion bewirkt, dass alle Zeichen, die an den Bildschirm ausgegeben werden, auch an den Drucker ausgegeben werden. Für das Steuerzeichen 09H und für das Zeichen 7FH wird ein Leerzeichen ausgegeben. Gleiches gilt bei der Zeichenausgabe vom Anwenderprogramm aus. Der Parameter d ist nur bei den Schreibmaschinen für einige notwendige Codewandlungen von Bedeutung (bei S6010 werden z. B. die deutschen Umlaute konvertiert.)

HARDCOPY und SCREENCOPY

Nach der Drucker-Initialisierung mit p = 2 kann über die Tastenkombination 

bei den Matrixdruckern die Funktion HARDCOPY und bei den Schreibmaschinen die Funktion SCREENCOPY aufgerufen werden. Damit wird der Bildschirminhalt auf dem Drucker ausgegeben.

SCREENCOPY bewirkt hier die Ausgabe aller ASCII-Zeichen des aktuellen Bildschirminhalts an die Schreibmaschine, wobei auf dem Bildschirm vorhandene Grafiken nicht mit ausgedruckt werden können.

HARDCOPY bewirkt bei den Matrixdruckern die punktweise Ausgabe des Bildschirminhalts. Damit werden
– alle ASCII-Zeichen,
– alle selbst definierten Pseudografikzeichen und
– die Grafiken

auf den Drucker ausgegeben.

Die Funktionen HARDCOPY bzw. SCREENCOPY werden nur erreicht durch
– Initialisierung des KC-Systems über die CAOS-Anweisung
  %LSTDEV mm k n p d mit p = 2 oder
– Setzen des Bit 0 des Inhalts der Speicherzelle 0B7E1H (HCPZ) wenn vorher die Protokollfunktion aktiv war, z. B. von BASIC aus über:
  VPOKE 14305, VPEEK (14305) OR 1

Ausgelöst werden können die Funktionen HARDCOPY bzw. SCREENCOPY auch aus einem Anwenderprogramm heraus durch die normale Ausgabe des Steuercodes 0FH. Von BASIC aus kann das z. B. mit PRINT CHR$(15); geschehen.
Je nach Druckertyp (Parameter d) werden unterschiedliche Treiber aufgerufen:

Tabelle 11: Festlegung des Drucker-Parameters d

Drucker	d	Hardcopy	Druckgeräte	d	Screencopy
K6313/K6327	0	ESC,*,5,320	S3004	8	
K6314/K6328	1	ESC,*,5,640	S6005/ S6009	9	
K6311	2	ESC,K,320	S6010	A	
K6312	3	ESC,K,640	S6120	B	
K6303/K6304	4	ESC,K,320	S6130	C	
ESC/P2	5	ESC,*,39,320	-	D	
ESC/P2	6	ESC,*,0,320	-	E	
ESC/P2	7	ESC,*,39,960	-	F	

Treiber 0 bis 4 und 8 bis C sind bereits seit CAOS 4.1 enthalten. Die Treiber 5 bis 7 gibt es ab CAOS 4.3, sie sind für den Anschluss von 24-Nadel-Druckern mit

ESC/P2 vorgesehen (z. B. Epson LQ100). Das erzeugte Hardcopy unterscheidet sich in der Anzahl und Dichte der gedruckten Punkte:

d=5 - Dichte 180 dpi, es wird mit allen 24 Nadeln gedruckt, jeder Druckpunkt entspricht einem Pixel, dadurch erscheint das gedruckte Bild entsprechend klein.

d=6 - Dichte 60 dpi, es wird nur mit 8 Nadeln (jede dritte Nadel) gedruckt, jeder Druckpunkt entspricht einem Pixel, das Druckbild erscheint in normaler Größe ist aber sehr hell (vergleichbar mit der DRAFT-Schriftart).

d=7 - Dichte 180 dpi, es wird mit 24 Nadeln gedruckt, um die selbe Größe wie bei d=6 zu erreichen wird jedes Pixel auf 3x3 Druckpunkte vergrößert, das Bild erscheint in gutem Kontrast (vergleichbar mit der LQ-Schriftart).

Entfällt der Parameter d, wird für den Matrixdrucker K6313 bzw. für Drucker mit gleichen Übertragungsbedingungen initialisiert. Die Parameter p und d können entfallen, wenn p = 0 ist und ein K6313 oder ein kompatibles Druckgerät angeschlossen ist. Der Parameter d ist nur für die Hardcopy-Routinen bzw. bei den Schreibmaschinen für einige notwendige Codewandlungen von Bedeutung (bei S6010 werden z. B. die deutschen Umlaute konvertiert.).

Die Einstellung der Übertragungsbedingungen der V.24-Schnittstelle ist im Kapitel 3.12.3 ab Seite 239 nachzulesen.

CAOS-Kommando PRINT

%PRINT Argumentliste

Das Kommando PRINT gibt es seit CAOS 4.3, damit können Ausgaben direkt an den Drucker gesendet werden. Voraussetzung ist ein aktivierter Druckertreiber, welcher mit LSTDEV vorher eingestellt wurde. Es wird das installierte Modul benutzt.
In der Argumentliste von PRINT sind zulässig:
– 8-Bit-Hexzahlen
– einzelne Zeichen, mit vorangestelltem Komma
– in Hochkommas eingeschlossene Zeichenketten

Beispiel:

%PRINT 1B ,R 2 'deutscher Zeichensatz' d a

Damit wird die Steuersequenz ESC,R,2 zum Drucker geschickt um diesen auf den deutschen Zeichensatz umzuschalten. Die Zeichenkette 'deutscher Zeichen-

satz' und die beiden abschließenden Codes d (0DH = CR) und a (0AH = LF) bewirken den Ausdruck der Textzeile.
Verwendet wird das Kommando PRINT hauptsächlich, um Druckereinstellungen zu verändern. Mögliche Steuersequenzen sind dem Handbuch des verwendeten Druckers zu entnehmen.

CAOS-Kommando V24DUP

%V24DUP [mm k n]

Das Betriebssystem enthält neben der Software für die Datenausgabe über V.24 (z. B. zu einem Drucker) auch Software für den Datenaustausch zwischen zwei Computern. Der Datenaustausch erfolgt in beiden Richtungen (Senden und Empfangen). Beim Start des KC-Systems wird der Kanal 2 des ersten vorhandenen V.24-Moduls bereits auf eine interruptgesteuerte Duplexroutine eingestellt. Mit dem Menüwort V24DUP und der Eingabe der Parameter erfolgt die Initialisierung einer Duplexroutine für den Pollingbetrieb.

Die Parameter bedeuten:

mm	- Modulschacht des V.24-Moduls (8, C, …)
k	- Kanal des V.24-Moduls (1 oder 2)
n	- USER-Aus/Eingabekanal (2 oder 3)

Fehlen die Parameter, wird das erste gefundene V.24-Modul mit den zuletzt eingestellten Werten initialisiert. Beim Systemstart werden die Werte für k = 2 und für n = 3 eingesetzt.
CAOS 4.5 bis 4.7 nimmt bei fehlenden Werten immer den V.24-Kanal 1 und USER-Kanal 2 an.

Beispiel:

%V24DUP 8 2 2 - Das Modul steckt im Schacht 8, V.24-Kanal 2 und USER-Kanal 2 werden benutzt.

Nach dem Aufruf von V24DUP wird der entsprechende Ausgabe- und Eingabekanal umgestellt. Danach kann z. B. in BASIC mit LIST#3(#2), PRINT#3(#2), INPUT#3(#2) und LOAD#3(#2) gearbeitet werden.
Nun kann von BASIC (z. B. über INPUT # 2 (#3)) eine Eingabe der Daten von einem Peripheriegerät erfolgen.

1.5.8. Sonstiges

CAOS-Kommando STACK

%STACK

Dieses Kommando kommt aus dem Debugger im USER-ROM, es zeigt den Wert des Stackpointers und den ersten auf dem Stack liegenden Wert an. Dies kann zur Fehlersuche hilfreich sein.

Beispiel:

%STACK
01C2 F26F

Der Stackpointer zeigt auf 01C2H und dort ist die Adresse oder der Wert F26FH abgelegt. Im Beispiel ist F26FH die LOOP-Adresse der Kommandoschleife von CAOS 4.8.

CAOS-Kommando TIME

%TIME

Das Kommando ‘time‘ gibt es seit CAOS 4.5, bei CAOS 4.5 ist das Menüwort in Kleinbuchstaben gehalten und deshalb standardmäßig nicht im Menü sichtbar. Ab CAOS 4.7 befindet sich das Kommando TIME in Großbuchstaben im Assembler-Segment des USER-ROM. Das Menüwort ist ebenfalls unsichtbar, solange die zugehörige Ebene nicht eingeblendet ist.

Mit TIME wird Datum und Uhrzeit vom D004 (D008) ausgelesen und am Bildschirm angezeigt. Voraussetzung dafür ist ein laufendes Programm DEP.COM ab Version 3.0. Der Datumstempel für USB auf der Adresse 000C wird vom Menükommando TIME nicht aktualisiert!

Taschenrechner CALC

CALC ist ein BASIC-basierender Rechner für CAOS, welcher von Frank Ludwig entwickelt worden ist. CALC wurde für den KC 85/5 und die Nutzung im USER-ROM optimiert. Er befindet sich seit CAOS 4.6 im Segment des KC-Debuggers. CALC wird mit %? und dem zu berechnenden Ausdruck aufgerufen.

%? Ausdruck

In der nächsten Zeile wird dann das Ergebnis angezeigt. Handelt es sich dabei um eine ganze Zahl zwischen -999999 und 999999, wird sie sowohl dezimal, als auch hexadezimal angezeigt.

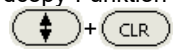
Intern benutzt CALC die Arithmetik des Basic-Interpreters. Dadurch können auch komplexe Berechnungen durchgeführt werden. Es sind nahezu alle BASIC-Funktionen nutzbar. Operationen mit kurzen Zeichenketten sind ebenfalls möglich.
Für den Ausdruck gilt die gewohnte BASIC-Syntax. Konstanten, Operatoren und Funktionen können wie gewohnt verwendet werden. Zusätzlich kann man in CALC Hexadezimalzahlen bis 24 Bit angeben. Diese müssen mit der Kennzeichnung H enden. Variablen kann man nicht verwenden. Sie verursachen in CALC einen SN-ERROR.

Beispiel:

```
%? B99CH+5*10
 B9CEH  47566
```

Es wird die Anfangsadresse für Fenster 5 im Speicher berechnet.

Als zweite Möglichkeit lässt sich CALC mit der Tastenkombination  +  aufrufen. Das laufende Programm wird unterbrochen und es erscheint am oberen Bildschirmrand ein Fenster. Darin kann man seine Berechnungen durchführen, solange bis man in einer leeren Zeile  drückt.

Drückt man in dem Fenster  , dann wird die berechnete Dezimalzahl an das unterbrochene Programm übergeben wie eine Tastatureingabe.
Ein installierter Druckertreiber wird durch CALC nicht beeinträchtigt.
Seine Screencopy- oder Hardcopy-Funktion wird aufgerufen, wenn man im CALC-Fenster ein zweites Mal  drückt.

Mit dem Menüwort

```
%?i
```

wird die „Drucktaste“ für CALC initialisiert. Dabei wird im RAM ab Adresse 108H ein 40 Byte großes Maschinenprogramm installiert.

Verwendung des Speichers für CALC

Der Speicher zwischen 0300H und 09EFH wird temporär als Arbeitsbereich von CALC und für den Basic-Interpreter benötigt. Um den Inhalt dieses Bereiches wiederherstellen zu können, nutzt CALC einen Teil des versteckten IRM als Auslagerungsspeicher. Verwendet wird der Speicher ab B800H im Color-IRM von Bild 0. Dieser Bereich wird bis jetzt von keiner CAOS-Version genutzt.

Die Übergabe der Dezimalzahl an das unterbrochene Programm erfolgt wie eine Funktionstasten-Zeichenfolge. Dazu kopiert CALC die Zahl an das obere Ende des Funktionstastenspeichers (Adressbereich B98E-B99BH).

2. HARDWARE

2. HARDWARE

2.1. Elemente des Blockschaltbildes

Als KC 85/5 gilt ein aufgerüsteter KC 85/4, bei dem die 64K-dRAMs gegen 256K-Typen, die CAOS-EPROMs gegen zwei 8K-EPROMs vom Typ 2764 und der BASIC-ROM gegen einen 32K-EPROM vom Typ 27256 ausgetauscht wurden. Die Leiterplatte ist bereits für diese Speichertypen vorbereitet, sodass darüber hinaus keine Arbeiten erforderlich sind.
Mithilfe des Blockschaltbildes des KC 85/5 lassen sich alle Baugruppen des Grundgerätes sowie die möglichen Erweiterungsbaugruppen übersichtlich darstellen (Bild 8, Seite 89). Im Folgenden sollen die einzelnen Funktionsgruppen näher beschrieben werden.

2.1.1. Zentrale Recheneinheit (ZRE)

Die ZRE besteht aus dem Mikroprozessor (CPU) U880D, dem Arbeitsspeicher (RAM) (256 KByte, Adressbereich 0000H-BFFFH), dem Bildwiederholspeicher (IRM) (64 KByte, Adressbereich 8000H-BFFFH) und dem Festwertspeicher (ROM) (48 KByte, Adressbereich C000H-FFFFH). Der ROM enthält das Betriebssystem, den BASIC-Interpreter, Assembler ASM, Debugger und Editor EDIT. RAM, IRM und ROM sind blockweise abschaltbar (vgl. Anweisung SWITCH).

2.1.2. Bildwiederholspeicher (IRM)

Der IRM (Image Repetition Memory) ist so konzipiert, dass jeder Bildpunkt (Pixel) auf dem Fernsehgerät im Pixel-RAM gespeichert ist (Bildschirmgröße 320*256 Punkte). In einem Feld von 8*8 Bildpunkten wird jeweils ein Zeichen abgebildet. Somit ist es möglich, maximal 40 Zeichen pro Zeile und 32 Zeilen pro Bild darzustellen. Im Lores-Modus ist jedem Bildfeld von 8*1 Bildpunkten ein Farbbyte zugeordnet. Im Hires-Modus kann jeder Bildpunkt eine von 4 Farben annehmen, welche sich aus dem entsprechenden Bit des Pixel- und Farbbytes ergibt. Der KC 85/5 besitzt 2 Bilder (0 und 1), die unabhängig voneinander beschrieben und angezeigt werden können.

2.1.3. Videointerface (VIF)

Das Videointerface hat die Aufgabe, die im IRM gespeicherten Informationen so aufzubereiten, dass diese auf dem Fernsehbildschirm dargestellt werden können. Es ist so ausgelegt, dass das Fernsehgerät direkt über den TV-RGB-Eingang (SCART- oder PERI-Buchse), über den FBAS-Eingang (AV-Buchse) oder über den Antenneneingang angeschlossen werden kann. Die beiden zuerst genannten Anschlüsse sind als gemeinsamer direkter Steckverbinder an der Rückwand des Grundgerätes herausgeführt. Zum Anschluss an den Antenneneingang ist die am Computer herausgeführte HF-Leitung zu verwenden.

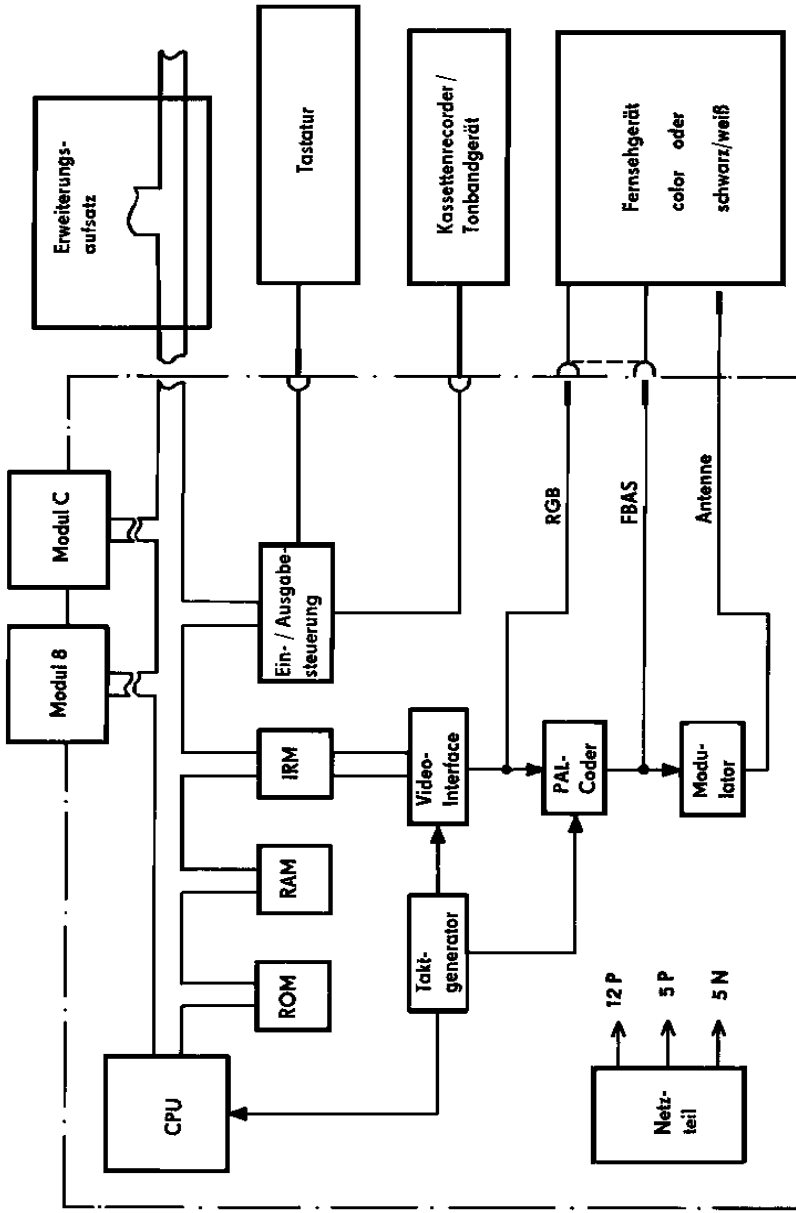


Bild 8: Blockschaltbild KC 85/5-System

Die Bildqualität verbessert sich gegenüber dem Antenneneingang bei einer Verbindung mit der AV-Buchse und wird beim Anschluss an den RGB-Eingang optimal. Schließen Sie ein Farbfernsehgerät am Antenneneingang oder an der AV-Buchse an, können Sie nur dann farbige Bilder vom KC 85/5 „empfangen“, wenn Ihr Gerät einen PAL-Decoder enthält.

2.1.4. Ein- und Ausgabesteuerung (EAS)

Die EAS hat die Aufgabe, die von der Tastatur und/oder vom Kassettengerät ankommenden seriellen Signale so aufzubereiten, dass sie vom Computer weiterverarbeitet werden können. Weiterhin werden die vom Computer erzeugten seriellen Signale für das Kassettengerät aufbereitet sowie die Tonausgabe gesteuert.

2.1.5. Tonerzeugung und Tonausgabe

Für die Tonerzeugung werden im KC 85 zwei CTC-Kanäle verwendet, welche in der Betriebsart Zeitgeber arbeiten und den halben Systemtakt entsprechend der programmierten Werte für Vorteiler v und Zeitkonstante z teilen. Die resultierende Tonfrequenz ist 1,77 MHz / 2 / (16 * z), bei Vorteiler v=0,
oder 1,77 MHz / 2 / (256 * z), bei Vorteiler v=1.

Die Tonausgabe erfolgt:
– am Steckverbinder „TV-RGB“ (vgl. Bild 13 Seite 101) über das Fernsehgerät mit RGB- oder FBAS-Eingang, einkanalig (rechter und linker Kanal gemixt) in 16 Lautstärkestufen,
– an der Diodenbuchse „TAPE“ (vgl. Bild 9 Seite 92), zweikanalig mit konstantem Pegel über einen Mono- oder Stereoverstärker oder auch über das Kassettengerät in Stellung „Aufnahme“ mit betätigter Pausen- oder Schnellstopptaste (falls Ihr Kassettengerät eine Tonwiedergabe während der Aufnahme ermöglicht),
– über einen internen Piezosummer, einkanalig (nur rechter Tonkanal).

2.1.6. Tastatur

In der Tastatur ist ein Fernbedienungsschaltkreis U807D zur Serialisierung der Tasteninformationen eingesetzt. Die Verbindung zum Computer erfolgt über eine abgeschirmte Leitung, über die sowohl die Stromversorgung zur Tastatur als auch der Datentransport erfolgen.

2.1.7. Netzteil

Aus der Netzspannung von 230V/50Hz werden Gleichspannungen von +12V, +5V und -5V abgeleitet.

2.2. Externe Anschlüsse

Das KC 85/5-Grundgerät verfügt über folgende externe Anschlussmöglichkeiten:

– Diodenbuchse TAPE
– Diodenbuchse KEYBOARD
– Modulsteckplatz 08
– Modulsteckplatz 0C
– Steckverbinder EXPANSION-INTERFACE
– Steckverbinder TV-RGB

Im Folgenden finden Sie eine detaillierte technische Beschreibung dieser Anschlüsse.

2.2.1. Diodenbuchse TAPE

Über diesen, an der Frontseite des Computers befindlichen Anschluss, wird die Speichereinheit Kassettengerät mit dem Computersystem durch ein handelsübliches Diodenkabel (Mono) verbunden.
Hier sind neben den Anschlüssen für ein Mono-Kassettengerät (Aufnahme und Wiedergabe) auch ein Computerausgang für den Stereo-Ton und eine Schaltspannung für den Motor des Kassettengerätes (TTL-Pegel) herausgeführt. Damit ist es möglich, über eine Stereo-Anlage, die vom Computer erzeugten Töne zweikanalig wiederzugeben. Der Antrieb eines Kassettengerätes, das dafür geeignet sein muss, wie z. B. der LCR-Data, kann gesteuert werden (entsprechend geschaltetes Stereo-Kabel).
Da die Aufzeichnungsdichte der Programme und Daten sehr hoch ist, ist darauf zu achten, dass sich das Kassettengerät in einem einwandfreien technischen Zustand befindet und dass nur Magnetbandkassetten ohne Klebe- oder Knitterstellen verwendet werden.

Signalbeschreibung der Diodenbuchse TAPE:

Signal-name	Signalbedeutung	An-schluss	Sonstige Bedingungen
00	Bezugspotential Masse	2	
WRITE SOUND-L	Schreibsignal bzw. Tonsignal 1 vom Computer	1	0,2V Uss an 100kOhm 0,4V Uss unbelastet
SOUND-R	Tonsignal 2 vom Computer	4	wie Tonsignal 1
TAPE ON	Einschaltsignal für Kassettenrecorder	5	Ausgang, TTL-Pegel schaltet bei Ein- und Ausgabeoperationen auf high
READ	Lesesignal vom Kassettenrecorder	3	Eingang

Bitte beachten Sie bei Anschluss von Stereo-Kassettenrecordern, dass das Einschaltsignal TAPE ON auf dem Anschluss 5 herausgeführt ist!

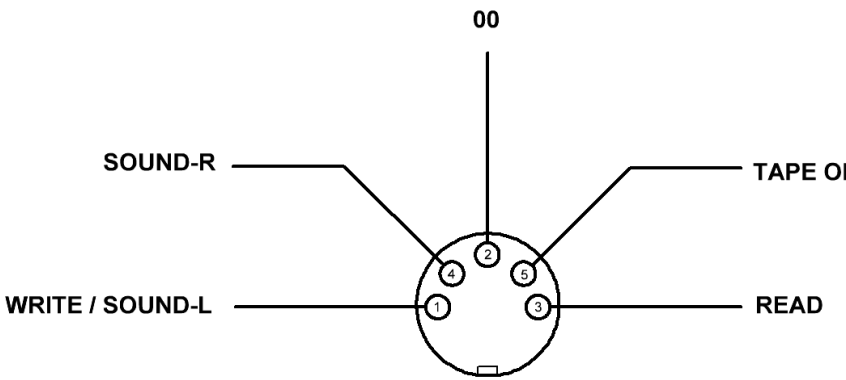


Bild 9: Anschlussbelegung der Diodenbuchse TAPE

2.2.2. Diodenbuchse KEYBOARD

An der Frontseite befindet sich neben der TAPE-Buchse der KEYBOARD-Anschluss. Diese Diodenbuchse dient zum Anschluss der Standard- oder der D005-Komfort-Tastatur des KC.

Signalbeschreibung der Diodenbuchse KEYBOARD:

Signal-name	Signalbedeutung	An-schluss	Sonstige Bedingungen
Schirm	Intern mit Masse verbunden	2	
kin	Eingangssignal von Tastatur	1	Gleichzeitig Stromversorgung der Standardtastatur
+5V	Stromversorgung D005-Tastatur	4	Nur, wenn Brücke RB01 auf der Hauptplatine entsprechend gesetzt ist.
KOUT	Ausgangssignal zu Tastatur	5	Rücksetzen D005-Tastatur in den CAOS-Modus *9
00	Bezugspotential Masse	3	

Im folgenden Bild ist die Anschlussbelegung der Buchse dargestellt.

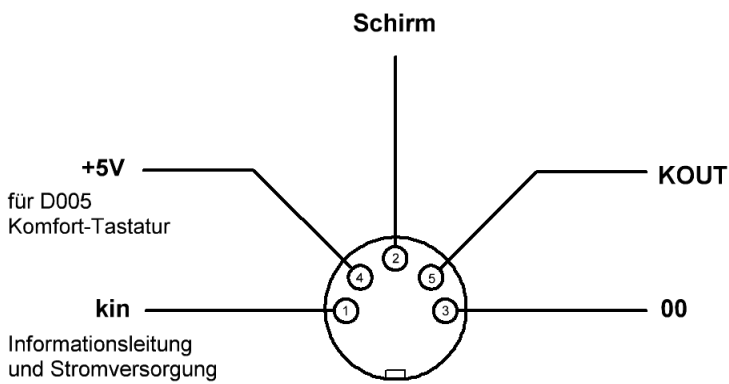


Bild 10: Anschlussbelegung der Diodenbuchse KEYBOARD

*9 Das Rücksetzen der D005-Tastatur funktioniert nur mit dem modifizierten EPROM der D005-Tastatur, neue Version vom 14.01.1995

2.2.3. Modulsteckplätze 08 und 0C

Diese Steckplätze dienen ausschließlich der Aufnahme der vom Hersteller angebotenen bzw. vom KC-Club neu entwickelten Zusatzmodule. Die maximal zulässigen Spitzenströme je Modul betragen:

300 mA bei + 5 V ± 5 %
100 mA bei + 12 V ± 10 %
5 mA bei - 5 V ± 10 %.

Jedes Modul, mit Ausnahme von M005 (Leermodul), M007 (Adapter), M008 (Joystick) und M021 (Joystick+Centronics), besitzt eine Prioritätssteuerung, die seine Einordnung in das KC-System ermöglicht. Dadurch können mehrere Module, auch vom gleichen Typ, im KC-System vorhanden sein. Ist das der Fall, gilt folgende Rangordnung:

Falls alle Module gleichen Typs eingeschaltet sind, besitzt dasjenige Modul, das sich auf dem Modulsteckplatz mit der niedrigsten Steckplatzadresse mm befindet, (gegenüber den anderen Modulen gleichen Typs) die höchste Priorität. Demzufolge besitzt der niedrigste Modulsteckplatz (mm=8) die höchste Priorität.

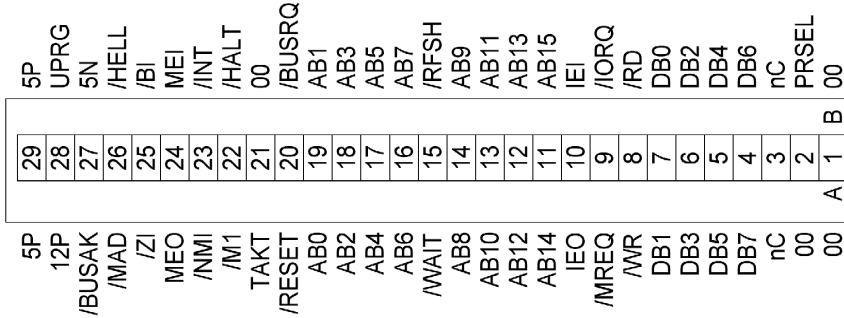


Bild 11: Anschlussbelegung des Modulsteckverbinders (Blick auf Frontseite)

2.2.4. Steckverbinder EXPANSION-INTERFACE

Eine ähnliche Signalbelegung wie der Modulsteckverbinder hat auch der Steckverbinder am EXPANSION-INTERFACE. Dieser Steckverbinder ist zum Anschluss von Erweiterungsaufsätzen vorgesehen. Im Vergleich zum Modulsteckverbinder sind dort allerdings einige Signale nicht belegt, das betrifft:

Anschluss	Modulsteckverbinder	EXPANSION-INTERFACE
10B	IEI	nicht belegt
26A	MAD	nicht belegt
24B	MEI	nicht belegt
27B	5N	nicht belegt
28B	UPRG	nicht belegt
29A	5P	nicht belegt
29B	5P	Nicht belegt

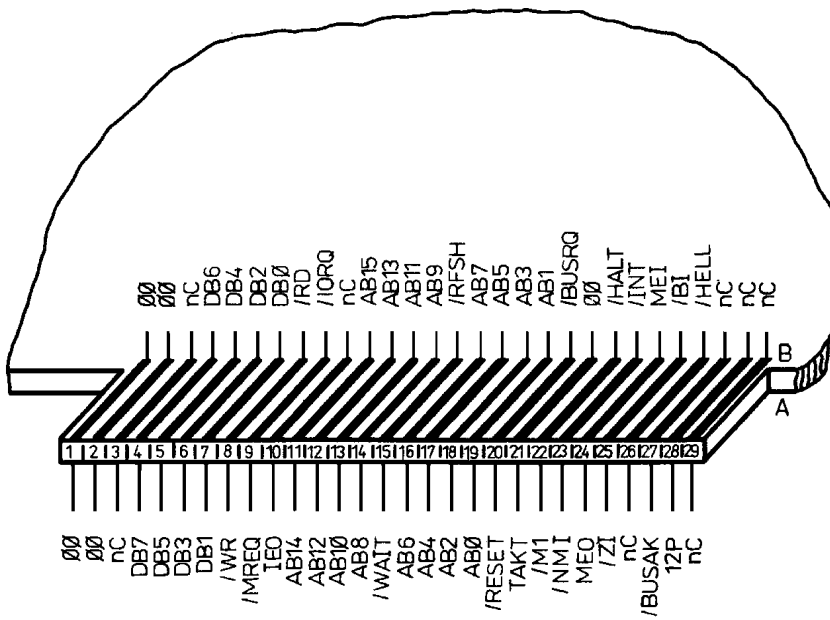


Bild 12: Anschlussbelegung EXPANSION-INTERFACE

2.2.5. Signalbeschreibung Modul- und EXPANSION-INTERFACE

Im Folgenden finden Sie eine zusammenfassende Beschreibung der Signale, welche am Modulsteckverbinder und am EXPANSION-INTERFACE anliegen. Die Anschlussbelegung des EXPANSION-INTERFACE ist im Systemhandbuch des KC 85/4 [48] abgedruckt. Die Anschlussbelegung am Modulsteckverbinder kann auch in [64] und [65] nachgelesen werden. Auf Unterschiede wird separat hingewiesen.

Tabelle 12: Signalbeschreibung am Modul- und Expansion-Steckverbinder

An-schluss	Signal-name	Signalbedeutung	Aktiv-Pegel	Sonstige Bedingungen
1A	00	Masse, Bezugspotential	-	
1B	00	Masse, Bezugspotential	-	
2A	00	Masse, Bezugspotential	-	
2B	PRSEL	Programmier-Select	-	Nur in Programmieraufsatz verwendet, mit 00 (Masse) belegt
3A	-	Nicht belegt		
3B	-	Nicht belegt		
4A	DB7	Datenbus Bit 7	High	Bidirektional *11, angeschlossene Sender müssen 3-state-Ausgänge besitzen
4B	DB6	Datenbus Bit 6	High	
5A	DB5	Datenbus Bit 5	High	
5B	DB4	Datenbus Bit 4	High	
6A	DB3	Datenbus Bit 3	High	
6B	DB2	Datenbus Bit 2	High	
7A	DB1	Datenbus Bit 1	High	
7B	DB0	Datenbus Bit 0	High	
8A	/WR	Schreiben: Signal zeigt an, dass durch den Prozessor Daten zum Speicher bzw. zu den E/A-Ports transportiert werden	Low	Unidirektional *11
8B	/RD	Lesen: Signal zeigt an, dass durch den Prozessor Daten oder Befehle vom Speicher bzw. von den E/A-Ports gelesen werden	Low	Unidirektional *11

An-schluss	Signal-name	Signalbedeutung	Aktiv-Pegel	Sonstige Bedingungen
9A	/MREQ	Speicheranforderung: Signal zeigt eine gültige Adresse für eine Speicherlese- oder -schreiboperation an	Low	Unidirektional *11
9B	/IORQ	Ein-/Ausgabeanforderung: Signal zeigt eine gültige Ein-/Ausgabeadresse an bzw. zusammen mit M1, dass ein Interruptgesuch von der CPU akzeptiert wurde	Low	Unidirektional *11
10A *10	IEO	Interrupt-Freigabe-Ausgang: Signal zeigt an, dass sich keine E/A-Ports mit höherer Priorität im Interrupt-Behandlungszustand befinden	High	Unidirektional, Prioritätssteuerung der E/A-Ports; Signal zum IEI des nachfolgenden E/A-Ports
10B *10	IEI	Interrupt-Freigabe-Eingang: Signal zeigt an, dass sich keine E/A-Ports mit höherer Priorität im Interrupt-Behandlungszustand befinden	High	Unidirektional, Prioritätssteuerung der E/A-Ports; Signal vom IEO des vorherigen E/A-Ports
11A	AB14	Adressbus Bit 14	High	Unidirektional *11, angeschlossene Sender müssen 3-state-Ausgänge besitzen
11B	AB15	Adressbus Bit 15	High	
12A	AB12	Adressbus Bit 12	High	
12B	AB13	Adressbus Bit 13	High	
13A	AB10	Adressbus Bit 10	High	
13B	AB11	Adressbus Bit 11	High	
14A	AB8	Adressbus Bit 8	High	
14B	AB9	Adressbus Bit 9	High	
15A	/WAIT	Warten: Signal zeigt dem Prozessor an, dass der adressierte Speicher bzw. E/A-Port nicht für einen Datenaustausch bereit ist	Low	Sammelleitung, angeschlossene Sender müssen Open-Kollektor-Stufen besitzen

*10 Das Signal IEI liegt beim D002 am EXPANSION-INTERFACE-IN an 10A und 10B an und das Signal IEO am EXPANSION-INTERFACE-OUT an 10A und 10B

An-schluss	Signal-name	Signalbedeutung	Aktiv-Pegel	Sonstige Bedingungen
15B	/RFSH	Auffrischen: Signal zeigt an, dass die Adressleitungen AB0…AB6 eine Adresse zum Auffrischen von dynamischen RAMs führen	Low	Unidirektional
16A	AB6	Adressbus Bit 6	High	Unidirektional *11 angeschlossene Sender müssen 3-state-Ausgänge besitzen; Adressierung von Speicher und E/A-Ports, AB0…AB6 sind mit /RFSH als Refresh-Adresse für dynamische RAMs gültig
16B	AB7	Adressbus Bit 7	High	
17A	AB4	Adressbus Bit 4	High	
17B	AB5	Adressbus Bit 5	High	
18A	AB2	Adressbus Bit 2	High	
18B	AB3	Adressbus Bit 3	High	
19A	AB0	Adressbus Bit 0	High	
19B	AB1	Adressbus Bit 1	High	
20A	/RESET	Rücksetzen: Zentrales Rücksetzsignal	Low	Sammelleitung, angeschlossene Sender müssen Open-Kollektor-Stufen besitzen
20B	/BUSRQ	Busanforderung: Signal zeigt dem Prozessor an, dass er die Busherrschaft abgeben soll	Low	Sammelleitung, angeschlossene Sender müssen Open-Kollektor-Stufen besitzen
21A	TAKT	Systemtakt	-	Unidirektional, nur Grundgerät als Sender zulässig
21B	00	Masse, Bezugspotential	-	
22A	/M1	Befehlslesezyklus: Signal zeigt an, dass der Prozessor einen Befehlslesezyklus durchführt bzw. zusammen mit /IORQ, dass ein Interruptgesuch von der CPU akzeptiert wurde	Low	Unidirektional
22B	/HALT	Prozessor-HALT: Signal zeigt den HALT-Zustand des Prozessors an	Low	Unidirektional

*11 Signalleitung direkt mit der CPU verbunden, kann bei DMA-Zugriff von extern gesteuert werden, das gilt jedoch nur für die beiden Modulsteckplätze 8 und C und den Expansionsport im Grundgerät – nicht bei den Erweiterungsgeräten!

An-schluss	Signal-name	Signalbedeutung	Aktiv-Pegel	Sonstige Bedingungen
23A	/NMI	Nichtmaskierbares Unterbrechungsgesuch	Low	Sammelleitung, angeschlossene Sender müssen Open-Kollektor-Stufen besitzen
23B	/INT	Maskierbares Unterbrechungsgesuch: Signal zeigt eine Bedienungsanforderung durch einen E/A-Port an	Low	Sammelleitung, angeschlossene Sender müssen Open-Kollektor-Stufen besitzen
24A *12	MEO	Modul-Freigabe-Ausgang: Signal zeigt an, dass sich kein Modul mit höherer Priorität im Datenaustausch mit dem Prozessor befindet	High	Unidirektional, Prioritätssteuerung der Module; Signal zum nachfolgenden Modul
24B *12	MEI	Modul-Freigabe-Eingang: Signal zeigt an, dass sich kein Modul mit höherer Priorität im Datenaustausch mit dem Prozessor befindet	High	Unidirektional, Prioritätssteuerung der Module; Signal vom vorherigen Modul
25A	/ZI	Zeileninhalt: Signal zeigt den Informationsbereich innerhalb einer Fernsehzeile an	Low	Unidirektional, nur Grundgerät ist als Sender zulässig
25B	/BI	Bildinhalt: Signal zeigt den Informationsbereich innerhalb eines Fernsehbildes an	Low	Unidirektional, nur Grundgerät ist als Sender zulässig
26A	/MAD	Moduladress-Signal: Die Adressleitungen AB10 bis AB15 werden zur Modulsteckplatzauswahl decodiert und den jeweiligen Steckplätzen als Steuersignal zugeordnet	Low	Unidirektional, Signal wird für jeden Steckplatz separat gebildet *13
26B	/HELL	Auftastsignal: Signal bewirkt ein Einschalten der höchsten Intensität des Elektronenstrahls der Bildröhre (Weißpegel)	Low	Unidirektional, Grundgerät ist Empfänger (nur bei RGB)

*12 Das Signal MEI liegt beim D002 am EXPANSION-INTERFACE-IN an 24A und 24B an und das Signal MEO am EXPANSION-INTERFACE-OUT an 24A und 24B

An-schluss	Signal-name	Signalbedeutung	Aktiv-Pegel	Sonstige Bedingungen
27A	/BUSAK	Busfreigabe: Signal zeigt an, dass der Prozessor den Bus freige-geben hat, alle Ausgänge (außer BUSAK und RFSH) befinden sich im hochohmi-gen Zustand	Low	Unidirektional
27B	5N	Spannung 5V negativ	-	max. 5mA belastbar *13
28A	12P	Spannung 12 V positiv	-	Modulsteckplatz: max. 100mA belastbar Expansion-Interface: max. 20mA belastbar
28B	UPRG	EPROM-Programmierspan-nung	-	Nur in Programmierauf-satz verwendet, mit 00 (Masse) belegt
29A	5P	Spannung 5V positiv	-	max. 300mA belastbar *13
29B	5P	Spannung 5V positiv	-	

*13 Signal liegt am Expansion-Interface nicht an

2.2.6. Steckverbinder TV-RGB

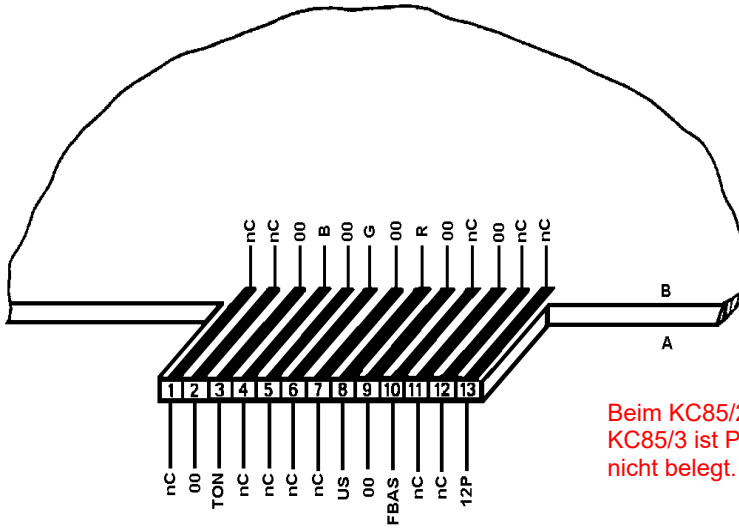


Bild 13: Anschlussbelegung des Steckverbinders TV-RGB

Der Steckverbinder TV-RGB dient zur Ausgabe des Bild- und Tonsignals wahlweise als RGB- oder FBAS-Signal.

Signal-name	Signalbedeutung	An-schluss	Ltg.-Anz.	Sonstige Bedingungen
00	Bezugspotential, Masse	Siehe Bild 13	6	
TON	Audio-Ausgang	3A	1	2 V Uss an Rl > 10 kOhm
R	Rot-Signal	8B	1	Differenzspannung 0,7 V eff. Last-Impedanz 75 Ohm überlagerte Gleichspannung 0V bis 2V
G	Grün-Signal	6B	1	wie R-Signal
B	Blau-Signal	4B	1	wie R-Signal
FBAS	Videoausgang, Videosignalgemisch	10A	1	1V Differenz zwischen Spitzen-Weiß-Pegel und Synchronisationspegel
US	RGB-Umschaltsignal	8A	1	1V an 75 Ohm
12P	AV-Umschaltsignal	13A	-	+12V

Die beste Bildqualität wird mit dem RGB-Signal erreicht. Bei TV-Geräten und Monitoren mit SCART- bzw. Euro-AV-Anschluss wird dies durch die Verwendung eines SCART-Kabels erreicht.

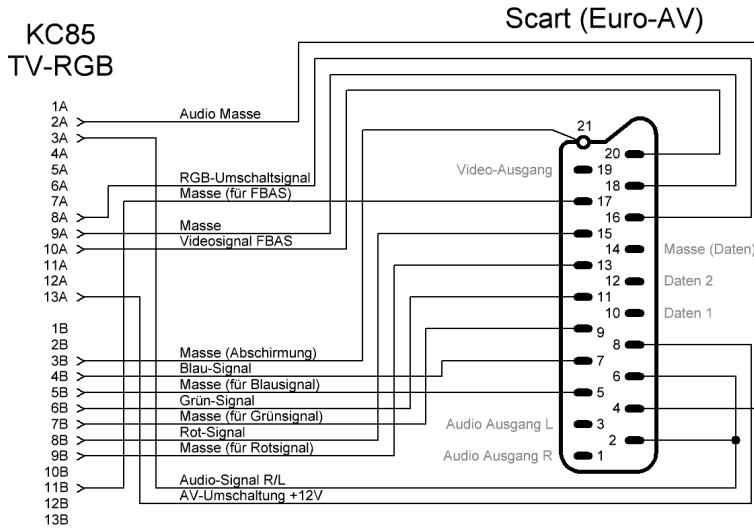


Bild 14: Anschlussbelegung SCART-Kabel

Es sind folgende Kabelverbindungen erforderlich:

TV-RGB	SCART-Stecker	Funktion
2A	4	Masse (Audio)
3A	2, 6	Audiosignal R / L
8A	16	RGB-Umschaltsignal
9A	18	Masse (für RGB-Umschaltsignal)
10A	20	FBAS-Videoeingang und Synchronsignal
13A	8	AV-Umschaltung +12V
3B	21	Masse (Abschirmung)
4B	7	Bildsignal blau
5B	5	Masse (für Blausignal)
6B	11	Bildsignal grün
7B	9	Masse (für Grünsignal)
8B	15	Bildsignal rot
9B	13	Masse (für Rotsignal)
11B	17	Masse (für FBAS-Signal)

Es ist auch möglich, den Monitor Commodore 1084-S an den KC 85/5 anzuschließen. Dieser Monitor besitzt einen echten RGB-Eingang. Das »S« in der Typenbezeichnung weist auf das Monitor-Modell mit Stereo-Lautsprechern hin.

Bei den Anschlüssen des 1084-S gibt es baujahrbedingte Unterschiede. Neuere Geräte besitzen neben einer kombinierten linearen RGB/TTL-RGB-Buchse noch den genormten Euro-AV- bzw. SCART-Buchse sowie Cinch-Buchsen für den linken und rechten Ton und das zusammengesetzte Videosignal CVBS. Ältere Modelle haben statt der SCART-Buchse zwei getrennte DIN-Buchsen für TTL bzw. lineares RGB. Einige Modelle besitzen auch eine 9polige D-Sub-Buchse. Für diese muss man sich dann ein passendes Kabel mit D-Sub-Stecker anfertigen. Wollen Sie noch die Tonausgabe verwenden, müssen Sie zusätzlich die Cinch-Buchsen mit dem Audiosignal des KC 85 verbinden. Am Steckverbinder TV-RGB liegt allerdings nur ein Monosignal an, sodass die beiden Cinch-Stecker parallel geschaltet werden müssen.

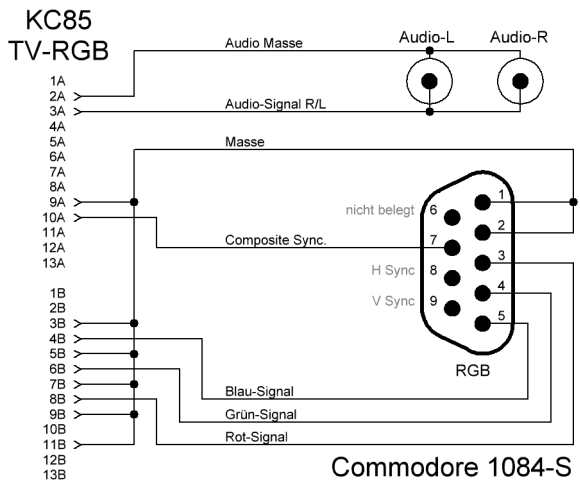


Bild 15: Anschlussbelegung RGB-Kabel für Monitor 1084-S

Es sind folgende Kabelverbindungen erforderlich:

TV-RGB	D-Sub-Stecker	Funktion
9A, 3B, 5B, 7B, 9B, 11B	1, 2	Masse (Bildsignale)
10A	7	Synchronsignal
4B	5	Bildsignal blau
6B	4	Bildsignal grün
8B	3	Bildsignal rot
2A	Masse (Cinch)	Masse (Audio)
3A	Audio (Cinch)	Audiosignal R und L

2.3. Systemausbau

2.3.1. Module und Erweiterungsaufsätze

Der KC 85/5 ist eine Weiterentwicklung der Reihe KC 85/2-4. Das Grundgerät erlaubt den Anschluss von 2 Erweiterungsmodulen und Erweiterungsaufsätzen (vgl. Bild 8: Blockschaltbild KC 85/5-System auf Seite 89).
Für die Module befinden sich an der Vorderseite des Grundgerätes zwei Modulschächte, in die die Module eingesteckt und mit dem Rechnerbus kontaktiert werden.

Zum KC 85/5-System stehen derzeit folgende Module und Zusatzgeräte zur Verfügung:

Tabelle 13: Übersicht verfügbarer Erweiterungsmodule und Geräte

Name	Bezeichnung	Strukturbyte	Beschreibung	Portadressen
M001	DIGITAL IN/ OUT	EF	Peripheriemodul zum Anschluss von anwenderspezifischen Schaltungen oder Geräten mit Parallelschnittstellen [37]	CTC: 00..03H PIO: 04..07H
M002	PIO PORT 3	DA	6 PIO-Ports mit je 8 Leitungen (vermutlich nicht zum Einsatz gekommen, Portadresse des PIO3 kollidiert mit M030!)	PIO1: B0..B3H, PIO2: B4..B7H, PIO3: B8..BBH
M003	V.24	EE	V.24-Interface zum Anschluss von Peripherieeinheiten, wie z. B. Drucker [1], [35]	SIO: 08..0BH, CTC: 0C..0FH
M004	WEATHER	DA	Neuentwicklung René Nitzsche: Wetter-Anzeigemodul für I²C-Sensoren, 1x PIO, 1x CTC, 32K (E)EPROM und Atmega32	CTC: B0..B3H, PIO: B4..B7H
M005	USER		Leermodul zur Ankopplung eigener Schaltungen an den KC 85/2-5 [32]	
M006	BASIC	FC	BASIC + CAOS 3.1 für den KC 85/2 *[14]	

Name	Bezeichnung	Struktur-byte	Beschreibung	Port-adressen
M007	ADAPTER		Adaptermodul zum Heraus-führen des Systembusses aus dem Modulschacht [32], [33]	
M008	JOY-MODUL	-	Modul zum Anschluss eines Spielhebels / Joystick.	PIO: 90..93H
M010	ADU1	E7	Analog-Digital-Umsetzer mit 4 Kanälen; wandelt analoge in digitale Signale um [40]	PIO: 40..43H
M011	64 KBYTE RAM	F6	Speichererweiterung um 64 KByte dynamischer RAM [36]	
M012	TEXOR	FB	40-Zeichen-Textverarbei-tungs- sowie ein Sortierpro-gramm und Druckertreiber-routinen [41]	
M021	JOYSTICK+ Centronics	-	Modul zum Anschluss eines Spielhebels und eines Dru-ckers mit Parallelschnittstelle (Erweiterung des M008)	PIO: 90..93H
M022	EXPANDER RAM	F4	Speichererweiterung um 16 KByte dynamischen RAM	
M025	USER EPROM 8K	F7	Speichererweiterung um 8 KByte EPROM, selbst zu pro-grammieren	
M026	FORTH	FB	FORTH 3.1	
M027	DEVELOP-MENT	FB	EDAS 1.4 für KC 85/2-4 *[14]	
M028	USER EPROM 16K	F8	Speichererweiterung um 16 KByte EPROM, selbst zu pro-grammieren	
M029	DAU1	E3	Der Digital-Analog-Umsetzer wandelt digitale in analoge Signale um.	OUT: 44..47H

*14 Dieses Modul ist für den KC85/5 nicht erforderlich, da die Software bereits im USER-ROM des Grundgerätes enthalten ist

Name	Bezeichnung	Strukturbyte	Beschreibung	Portadressen
M030	EPROMMER	D9 DB	EPROM-Brenner 2-32K mit Software auf 8K-EPROM EPROM-Brenner 1-64K mit Software auf 16K-EPROM (Neuentwicklung KC-Club)	PIO1: B8..BBH, PIO2: BC..BFH
M032	256K segmented RAM	79	Speichererweiterung um 256 KByte dynamischen RAM in 16 Blöcken zu je 16 KByte	
M033	TYPESTAR	01	Dieses 16K-ROM-Modul enthält das 80-Zeichen-Textverarbeitungsprogramm Type-Star sowie das RAM-Floppy-System RAMDOS für CAOS.	
M034	512K segmented RAM	7A	Speichererweiterung um 512 KByte dynamischen RAM in 32 Blöcken zu je 16 KByte	
M035	1MB segmented RAM	7B	Speichererweiterung um 1024 KByte dynamischen RAM in 64 Blöcken zu je 16 KByte	
M035*4	4*1MB segmented RAM	7B	Neuentwicklung KC-Club: Modul verhält sich wie 4 einzelne 1MB-Module M035 auf dem Steckplatz und dessen Submodulsteckplätzen	
M036	128 segmented RAM	78	Speichererweiterung um 128 KByte dynamischen RAM in 8 Blöcken zu je 16 KByte	
M037	segmented ROM 32K/64K/128K	70 – 73	ROM-Modul zur variablen Bestückung von 32K bis 128K bzw. 256K	
M040	USER PROM	F7 / F8	ROM-Modul zur variablen Bestückung mit 8K oder 16K	
M041	EEPROM 2x 16K	01 / F1 / F8 / FC	Neuentwicklung René Nitzsche: 2 Submodule zu je 16K, in 8K-Blöcken rotierbar	

Name	Bezeichnung	Struktur-byte	Beschreibung	Port-adressen
M045	32K segmented ROM	70	Speichererweiterung um 32 KByte EPROM in 4 Blöcken zu je 8 KByte, selbst zu programmieren	
M046	64K segmented ROM	71	Speichererweiterung um 64 KByte EPROM in 8 Blöcken zu je 8 KByte, selbst zu programmieren	
M047	128K segmented ROM	72	Speichererweiterung um 128 KByte EPROM in 16 Blöcken zu je 8 KByte, selbst zu programmieren	
M048	256K segmented ROM	73	Speichererweiterung um 256 KByte EPROM in 16 Blöcken zu je 16 KByte, selbst zu programmieren	
M049	215K segmented ROM	74	Speichererweiterung um 512 KByte ROM in 32 Blöcken zu je 16 KByte, selbst zu programmieren	
M051	Scanner	EC	Neuentwicklung KC-Club: Interface für Handscanner, I²C- und serielle Schnittstelle	PIO: 20..27H
M052	USB+Netzwerk	FD	Neuentwicklung KC-Club: USB- und Netzwerkschnittstelle sowie Software dafür in 32 KByte EEPROM (4 Blöcke zu je 8 KByte)	Netzwerk-PIO: 28..2BH, USB-PIO: 2C..2FH
M052	USB	FD	Neuentwicklung KC-Club: Nur USB-Schnittstelle sowie Software dafür in 8 KByte EEPROM	PIO: 2C..2FH
M053	RS232	EE	wie M003, jedoch hat Kanal 2 TTL-Pegel zum Anschluss externer Tastaturen	SIO: 08..0BH, CTC: 0C..0FH
M062	RAM/ROM	F3	Neuentwicklung E. Mueller: 64K ROM oder 32K RAM	

Name	Bezeichnung	Struktur-byte	Beschreibung	Port-adressen
M066	SOUND	DC	Neuentwicklung René Nitzsche: Soundmodul mit Soundchip AY-3-8910, CTC	38..3FH
M120	8K CMOS RAM	F0	8K CMOS-RAM, batteriegestützt	
MT01 MT02	Prüfmodul	- BF	Prüfmodul für Modulsteckplätze und D002	C0..C9H
D002	BUSDRIVER		Aufsatzgerät mit 4, für den Anwender frei verfügbaren Modulsteckplätzen [39]	80H
D004	FLOPPY DISK	A7	Aus zwei Aufsatzgeräten bestehende Erweiterung die den KC 85/5 um ein zweites Prozessorsystem mit 4MHz, 64K System-RAM und ein Diskettenlaufwerk für die Ausführung CP/M-kompatibler Software unter dem Betriebssystem KC-Micro-DOS ergänzt.	F0..F4H
D005	Komfort-Tastatur		Ist eine Tastatur, die an Stelle der Originaltastatur an die KEYBOARD-Buchse angeschlossen wird	

Name	Bezeichnung	Struktur-byte	Beschreibung	Port-adressen
D008	MULTI DISK	A7	Ist ein D004-kompatibles Aufsatzgerät, das den KC 85/5 um ein zweites Prozessorsystem mit 4 bis 16MHz, 64K bzw. 128K System-RAM, 2MB-RAM-Floppy und IDE-Festplatteninterface für die Ausführung CP/M-kompatibler Software unter dem Betriebssystem ML-DOS ergänzt. Es können sowohl 5,25“-Diskettenlaufwerke vom D004 als auch 3,5“-HD-Diskettenlaufwerke angeschlossen werden. Es wurden mehrere Prototypen erfolgreich getestet. Vom KC-Club wurde 2018 die ursprüngliche Version 1.0 mit Modifikationen als V100 in Serie aufgelegt.	F0..F4H

2.3.2. Systemausbau über V.24-Schnittstelle

Durch serielle oder parallele Schnittstellen in einem Computersystem können Daten an externe Geräte (Drucker, Plotter, Computer usw.) gesendet oder von ihnen empfangen werden. Diese Möglichkeiten bestehen auch am KC-System. Im Betriebssystem CAOS 4.x ist bereits Software für die serielle Schnittstelle enthalten. Die Schnittstellenmodule heißen M003 V.24 oder M053 RS232 und unterscheiden sich nur im Signalpegel des 2. Kanals. Um Verwechslungen auszuschließen, besitzt dieser Kanal des M053 mit TTL-Pegel eine 6polige Buchse. Im Auslieferzustand ist Pin 6 nicht belegt, kann aber nachträglich mit +5V beschaltet werden, um z. B. eine externe Tastatur mit Spannung zu versorgen.

Je Modul können zwei externe Geräte an das KC-System angeschlossen werden. Hier werden nur einige Varianten genannt. Denkbar wären z. B. folgende Zusammenstellungen:

– Drucker und Plotter,
– Drucker und 2. Computer,
– Schreibmaschine (z. B. S3004) als Druckgerät und Eingabetastatur
– Computerkopplung usw.

Durch die Verwendung mehrerer M003- bzw. M053-Module ergibt sich die Möglichkeit, mehr als nur zwei serielle Geräte anzuschließen. Dabei ist zu beachten, sind mehrere V.24-Module eingeschaltet, ist das Modul im niedrigsten Steckplatz am höchsten priorisiert. Das heißt, das Modul im Modulschacht 8 ist am höchsten priorisiert. Durch Abschalten eines höher priorisierten V.24-Moduls werden auch die niedriger priorisierten V.24-Module nutzbar (z. B. Modul M003 im Modulschacht C wird nutzbar, wenn das M053 im Schacht 8 abgeschaltet wurde).

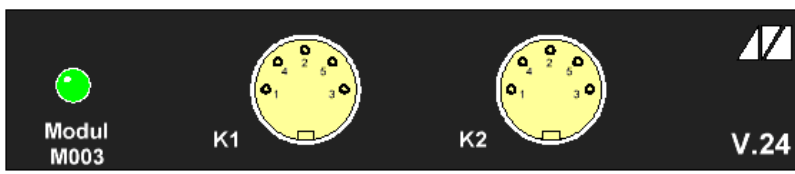
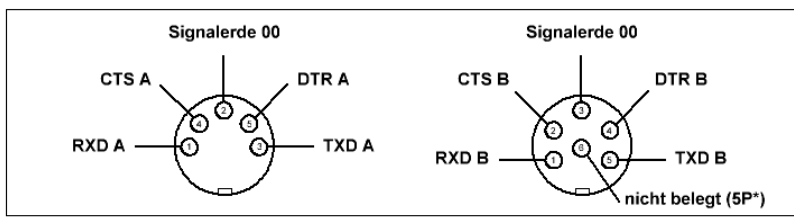


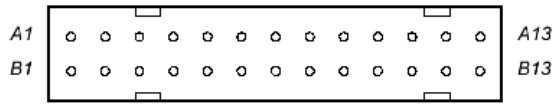
Bild 16: Frontansicht Modul M003 und Signalbelegung M053



CAOS 4.x verwendet standardmäßig immer nur das erste gefundene V.24-Modul, siehe dazu auch Kapitel 3.12. Seite 235 im Abschnitt Software.

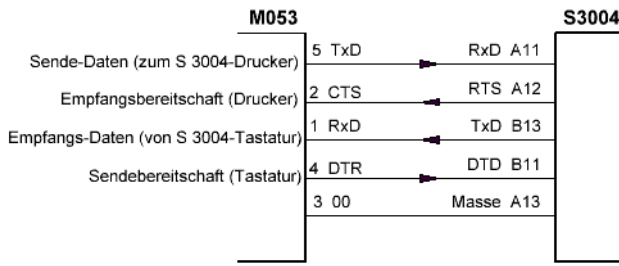
Benutzung einer Schreibmaschine Erika S3004 am KC 85/5:

Die Schreibmaschine S3004 war in der DDR weit verbreitet und kann dank seriellem Interface als Drucker und als Eingabegerät (Tastatur) für den KC 85 genutzt werden. Die Voraussetzung dazu ist eine Interfacebox IF6000 und ein Modul M003. Bei der Nutzung am Kanal 2 des M053 ist keine Interfacebox nötig. Die Anschlussbelegung der Interfacebuchse der S3004 sowie die Beschaltung des Kabels zur rechten Buchse des M053 zeigen die folgenden beiden Bilder:



(Blick auf die Buchse)

A11 Datenempfänger RxD	*B11 Empfänger bereit DTD*
A12 Sendegsuch RTS	*B12 Uss +5V*
A13 Masse	*B13 Datensender TxD*



Im stromlosen Zustand ist die Schreibmaschine S3004 mit dem IF6000 (V.24) und dem M003 (rechte oder linke Buchse) bzw. direkt mit dem M053 (rechte Buchse) zu verbinden.

! Bei einer direkten Verbindung unbedingt die rechte Buchse des M053 zur Arbeit mit der S 3004 benutzen! Die linke Buchse des M053 (Kanal 1) arbeitet mit Normalpegel und führt zur Zerstörung der SIO in der Schreibmaschine!

Die Grundeinstellung der S3004 entspricht der Voreinstellung des KC 85/5, siehe Kapitel V.24-Software ab Seite 235.
Der Druckertreiber ist mit dem Kommando LSTDEV (Parameter siehe Seite 80) zu initialisieren.

Zu beachten ist eine Grundeinstellung der Schreibmaschine S3004 von 72 Zeilen pro Blatt. Die S3004 geht automatisch auf Spalte 10 nach dem Einschalten und kann maximal 65 Zeichen pro Zeile drucken. Deshalb empfiehlt es sich im Editor den linken Rand auf 0 und den rechten Rand auf Spalte 64 zu setzen.

Die Tastatur der S3004 ist nach Betätigung der Return-Taste an der Schreibmaschine sofort einsatzbereit.

2.3.3. Anschluss eines Joysticks

Mit dem Einsatz eines Joysticks kann der Bedienkomfort für viele Spiel- und Anwenderprogramme wesentlich erhöht werden. Die Module M008 bzw. M021 ermöglichen den Anschluss eines Joysticks an den KC 85/5. Dazu kann jeder digitale Joystick verwendet werden, der als Anschluss eine 9-polige D-Sub-Buchse besitzt. Das Joystick-Modul verfügt über einen 9-poligen D-Sub-Stecker, die Anschlussbelegung ist in Tabelle 14 ersichtlich. Ein Joystick-Modul kann prinzipiell in jedem Modulsteckplatz betrieben werden, die Modulprioritätskette muss jedoch geschlossen bleiben. Der Betrieb von zwei Joystick-Modulen ist nicht möglich!

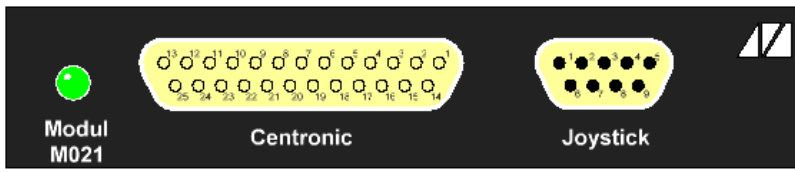


Bild 17: Frontansicht Modul M021

Tabelle 14: Belegung der Anschlüsse am Joystick-Modul

Anschluss	Funktion
1	UP
2	DOWN
3	LEFT
4	RIGHT
5	
6	FIRE2 *15
7	+5V
8	GND (COM)
9	FIRE

! ACHTUNG! Das Stecken bzw. Entfernen von Modulen darf nur bei ausgeschaltetem Computer bzw. Aufsatz erfolgen! Ebenso sollte der Joystick nur im stromlosen Zustand des Gerätes kontaktiert werden.

*15 FIRE2 ist bei den meisten Joysticks die erste bzw. einzige Feuer-Taste

2.3.4. Anschluss eines Druckers mit Parallelschnittstelle

Ab CAOS 4.3 wird der Anschluss von Druckern mit Parallelschnittstelle (Centronics) direkt unterstützt. Dazu ist entweder ein Modul M021 (erweitertes M008) oder ein M001 erforderlich. Beim M021 kann ein handelsübliches Druckerkabel mit 25poligem D-Sub-Stecker verwendet werden. Soll der Drucker an einem M001 angeschlossen werden, dann gilt die Anschlussbelegung wie im M001-Handbuch. Tabelle 15 zeigt die beiden Varianten der Schnittstellenbelegung im Vergleich.

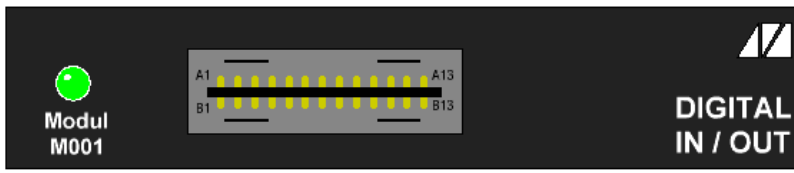


Bild 18: Frontansicht Modul M001

Tabelle 15: Belegung Parallelschnittstelle

Signalname	Anschluss M001	Bezeichnung M001	Anschluss M021	Bezeichnung M021
Masse	1A, 1B		18 - 28	
DATA1	2A	PIO A0	2	PIO B0
DATA2	3A	PIO A1	3	PIO B1
DATA3	4A	PIO A2	4	PIO B2
DATA4	5A	PIO A3	5	PIO B3
DATA5	6A	PIO A4	6	PIO B4
DATA6	7A	PIO A5	7	PIO B5
DATA7	8A	PIO A6	8	PIO B6
DATA8	9A	PIO A7	9	PIO B7
/STROBE	2B	PIO B0	1	PIO A6
BUSY	4B	PIO B2	11	PIO A7

2.3.5. Anschluss eines Plotters XY 4231 am Modul M001

Eine weitere interessante Möglichkeit ist der Anschluss eines Plotters zur grafischen Ausgabe am KC 85/5. Der tschechische Kleinplotter XY 4131 kann Zeichnungen im Format A4 ausgeben. Zwei Schrittmotoren bewegen dazu das Papier des Plotters in X-Richtung und den Stift in Y-Richtung jeweils in Schritten zu 0,1mm. Ein weiteres Signal dient dem Absenken bzw. Anheben des Stiftes. In der folgenden Tabelle sowie dem Bild ist die von Mühlhausen empfohlene Anschlussvariante am IN/OUT-Modul M001 nach [59] und [60] dargestellt. Als Ansteuersoftware eignet sich die Erweiterung Plotter BASIC 2.0 [69].

Tabelle 16: Empfohlene Anschlussbelegung Plotter XY 4131 am M001

Signalname	Anschluss M001	Bezeichnung M001	Bedeutung	Anschluss Plotter
GND	1A, 1B	Masse	Masse	6
STEP	2B	PIO B0	Zählschritt	2
X-Y	3B	PIO B1	Richtung X oder Y	3
V-R	4B	PIO B2	vorwärts / rückwärts	4
/READY	7B 8B (Diode zu 7B)	PIO B5 PIO B6	Bereitschaftsmeldung	5
PEN	9B	PIO B7	Stift heben / senken	1

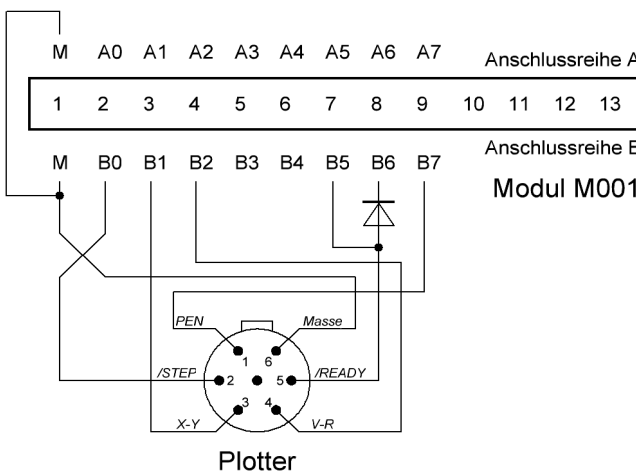


Bild 19: Empfohlenes Anschlussbild Plotter XY 4131 am M001

2.3.6. Anschluss einer Tastatur am Modul M052

Neben der Standard- oder D005-Komfort-Tastatur können am KC 85/5 auch Tastaturen am V.24-Modul angeschlossen werden. Eine weitere interessante Möglichkeit ergibt sich bei Verwendung des Moduls M052 in Verbindung mit einem VNC2-Modul mit spezieller Firmware und dazu angepasster Software im EEPROM des Moduls. Die Module M052 besitzen jeweils zwei USB-Buchsen. Die Tastatur kann dabei an einer beliebigen USB-Buchse angeschlossen werden. Die andere USB-Buchse wird dann für den USB-Stick genutzt.

Ab CAOS 4.7 wird ein Modul M052 mit enthaltenem USB-Treiber beim Einschalten automatisch initialisiert. Die VNC2-Software ab Version 2.7 im EEPROM des M052 installiert dabei einen Tastatur-Interrupt, sodass sofort nach dem Einschalten des KC 85/5 eine Tastatur am M052 funktionsfähig ist. Somit können USB-Tastaturen sowie auch PS/2-Tastaturen bei Verwendung eines M052 mit zusätzlicher PS/2-Buchse (z. B. M052 mit Netzwerk) am KC 85/5 verwendet werden.

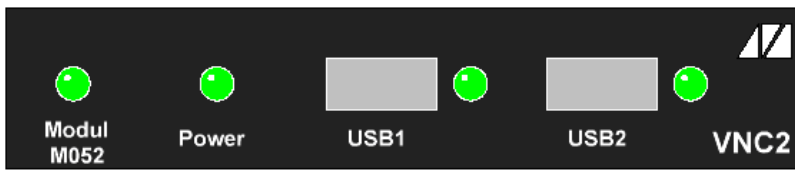


Bild 20: Frontansicht Modul M052 ohne Netzwerk

Bild 21: Frontansicht Modul M052 mit Netzwerk und PS/2-Buchse

Hinweis!
Mit den VDIP1 bzw. VDIP2 der ersten Auflagen der Module M052 ist ein Anschluss von PS/2- oder USB-Tastaturen nicht möglich. Für den VNC1 existiert keine Firmware mit PS/2-Unterstützung und eine Firmware-Anpassung ist beim VNC2 möglich.

3. SOFTWARE

3. SOFTWARE

3.1. Systemkonzept

3.1.1. Merkmale des Betriebssystems

Der Aufbau des CAOS-Betriebssystems ist im Bild 22 auf Seite 119 als Schema veranschaulicht.
Der KC 85/5 enthält einen RAM von 256 KByte, einen IRM (Bildwiederholspeicher) von 64 KByte und einen ROM von 48 KByte. Der Festwertspeicher (ROM) enthält das Betriebssystem, d. h. die wichtigsten Programme zur Bedienung der Peripherie sowie BASIC-Interpreter, Debugger, Assembler ASM und Editor EDIT.
Das Betriebssystem KC-CAOS (CASSETTE AIDED OPERATING SYSTEM) verwaltet die Gerätetreiber-Routinen mittels Menütechnik.
In den folgenden Kapiteln sollen die einzelnen Software-Baugruppen näher beschrieben werden. Voraussetzung zur Anwendung dieser Softwarebeschreibung sind Kenntnisse in der Assemblerprogrammierung.

Das Betriebssystem KC-CAOS 4.8 ist, um vielen Anwendungsbereichen gerecht zu werden, sehr flexibel ausgelegt. Es ermöglicht dem Anwender:

– den Arbeitsspeicher für das Betriebssystem, den Kellerspeicher (STACK) und die Interrupt-Tabellen an beliebigen Stellen im RAM anzuordnen,
– leicht eigene Maschinenprogramme durch Menütechnik in das System einzubinden,
– den eigenen Programmen beim Aufruf über Menü bis zu maximal 10 Parameter zu übergeben,
– die Systemressourcen durch eine große Anzahl von Systemunterprogrammen vollständig zu nutzen,
– Erweiterungsbaugruppen (Module) zu verwalten, d. h., es können somit mehrere Module quasi gleichzeitig betrieben werden,
– die im Grundgerät enthaltenen Speicher (RAM, IRM, ROM) ein- und auszuschalten,
– das im Grundgerät enthaltene Betriebssystem umzuschalten oder komplett abzuschalten und mit einem anderen, in einem Modul enthaltenen, zu arbeiten,
– RAM-Speicherblöcke mit einem Schreibschutz zu versehen,
– die 6 auf der Tastatur enthaltenen Funktionstasten (F1…F6) in beiden möglichen Belegungen mit beliebigen Codes oder Zeichenketten, z. B. Menü- oder BASIC-Schlüsselwörtern oder Abarbeitungstastenfolgen (JOBS) zu belegen,
– für die Darstellung von Zeichen auf dem Bildschirm beliebige Zeichenbildtabellen (Zeichengeneratoren) zu verwenden, d. h. man kann sich Zeichenbilder frei definieren (z. B. kyrillische Buchstaben, Grafikzeichen) und diese z. B. auf Magnetband abspeichern und
– die Zeichencodes der Tastatur beliebig zuzuordnen.

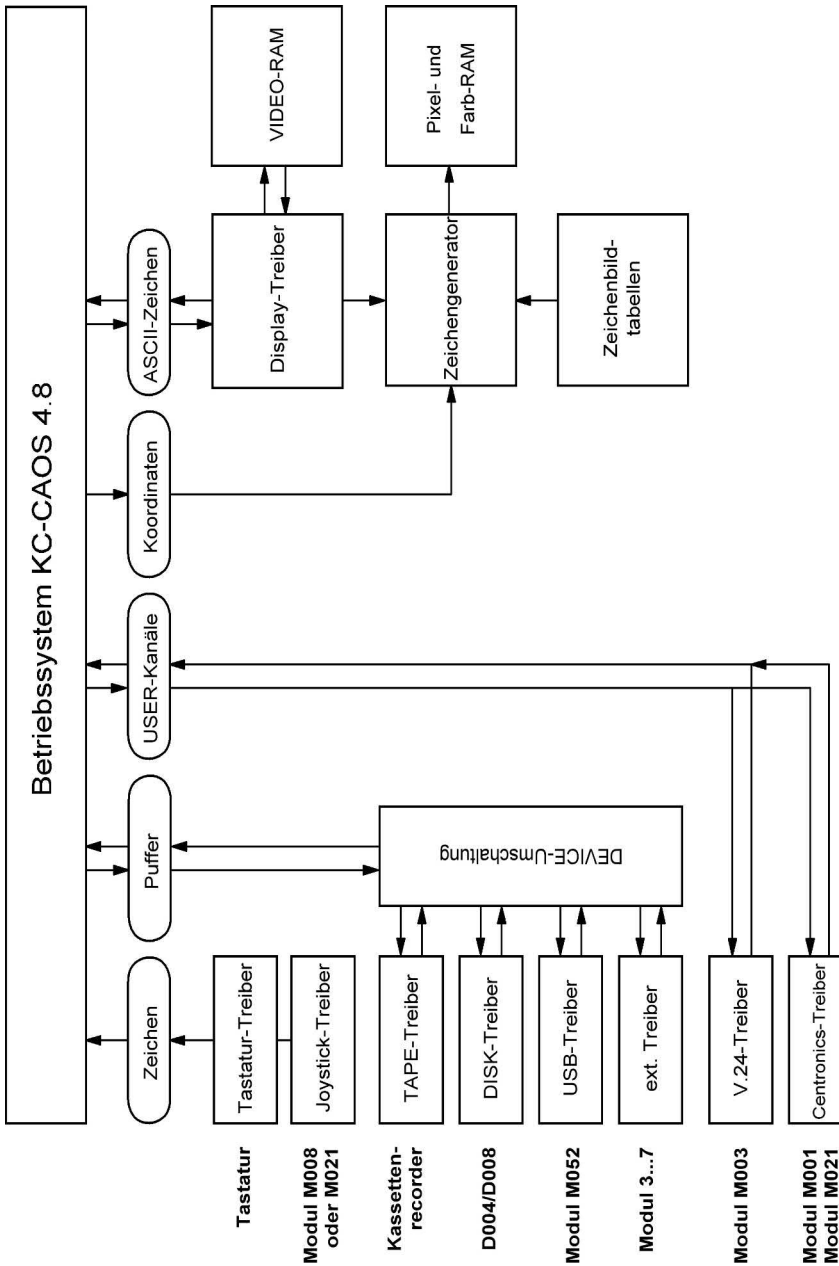


Bild 22: Schematischer Aufbau des Betriebssystems CAOS

3.1.2. Die zentrale Steuerschleife

Die im Bild 23 dargestellte zentrale Steuerschleife verdeutlicht die Steuerung der Funktionen des Betriebssystems KC-CAOS.

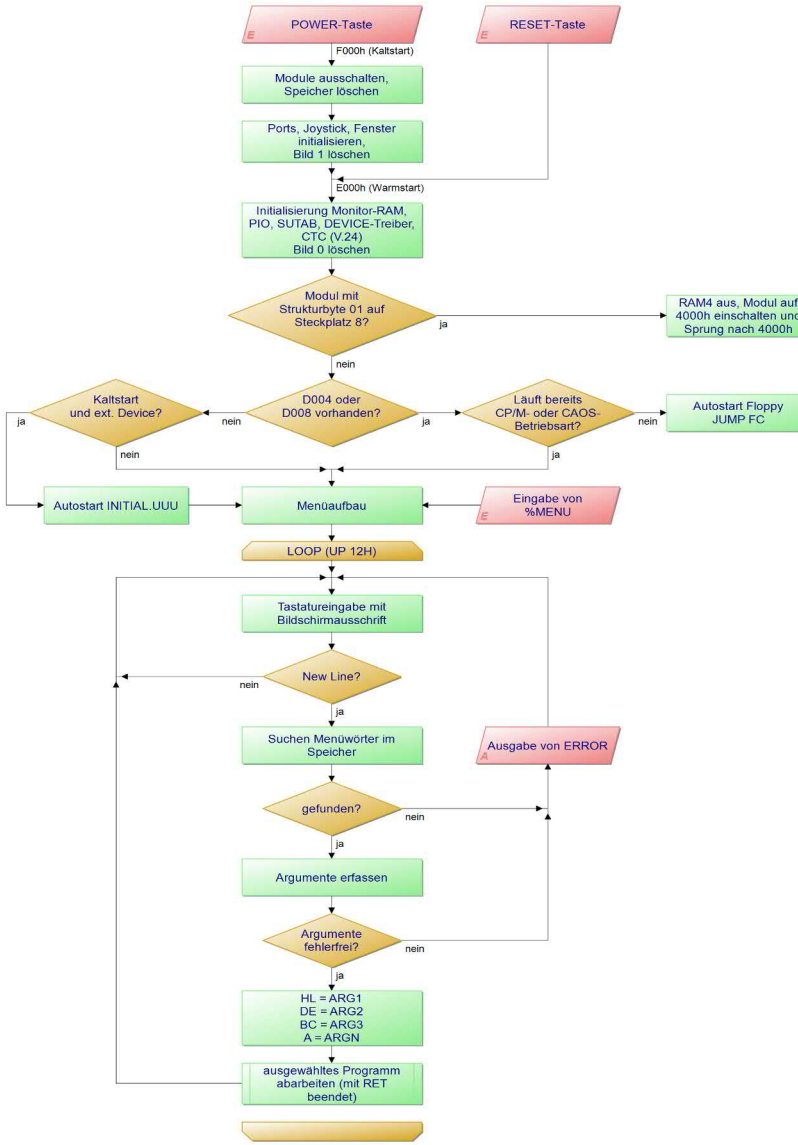


Bild 23: Zentrale Steuerschleife des Betriebssystems CAOS

3.2. Speicheraufteilung

Eine Übersicht der Speicheraufteilung des KC 85/5 vermittelt Bild 24, Seite 122.

Tabelle 17: Speicherübersicht (interne Module)

Adressen	vorhandener Speicher	Nutzung
0000H-BFFFH	256 KByte dynamischer RAM, dabei liegen vierzehn Blöcke zu je 16K hintereinander ab Adresse 8000H	Anwenderspeicher RAM
8000H-BFFFH	64 KByte dynamischer RAM (IRM), dabei liegen vier Blöcke zu je 16K hintereinander ab Adresse 8000H	Bildwiederholspeicher IRM
C000H-DFFFH	32 KByte ROM, dabei liegen vier Blöcke zu je 8K hintereinander ab Adresse C000H	USER-ROM (BASIC-Interpreter, Debugger, Assembler, Editor)
C000H-FFFFH	16 KByte ROM	Betriebssystem CAOS

Tabelle 18: Grobe Gliederung RAM und ROM

Adressbereich hexadezimal	dezimal	Bemerkungen
0000H-01FFH	00000-00511	System-RAM mit Interrupttabelle und Stack Details siehe Kapitel 3.6.1 ab Seite 178
0200H-BFFFH	00512-49151	frei für Anwender (ab 8000H RAM8-Ebenen)
8000H-A7FFH	32768-43007 00000-10239 *16	Pixel-RAM und Color-RAM in 4 Ebenen (Bildpunktspeicher)
A800H-ACFFH	43008-44287 10240-11519 *16	IRM-Arbeitszellen, Teil 1 Details, siehe Kapitel 3.6.8 ab Seite 186
AD00H-B6FFH	44288-46847 11520-14079 *16	Video-RAM Bild 1 und 0 (ASCII-Speicher)
B700H-BFFFH	46848-49151 14080-16383 *16	IRM-Arbeitszellen, Teil 2 Details, siehe Kapitel 3.6.8 ab Seite 186

*16 in den BASIC-Anweisungen VPEEK und VPOKE zu verwendende Speicheradressen (vgl. BASIC-Handbuch)

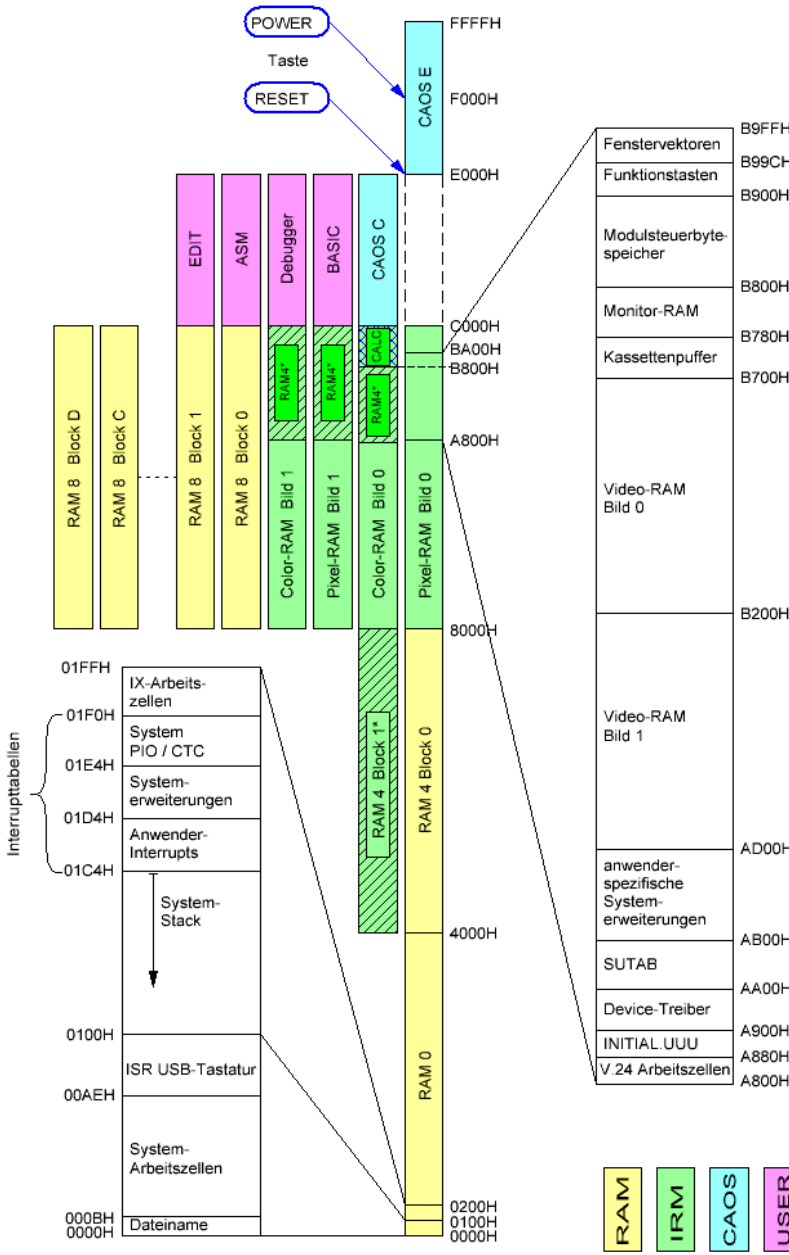


Bild 24: Übersicht der Speicheraufteilung des KC 85/5 mit CAOS 4.8

3.2.1. Fenstervektorspeicher

Das Betriebssystem gestattet es, 10 verschiedene Bildschirmfenster zu definieren und jederzeit wieder aufzurufen, wobei die Parameter des aktuellen Fensters gerettet werden. Als Fensternummern sind 0 bis 9 zugelassen.

Der Fenstervektor ist wie folgt aufgebaut:

Anfangsadresse WNDFN = 0B99CH + n * 0AH; n = Fensternummer

WNDFN - Fensteranfang Spalte
+1 - Fensteranfang Zeile
+2 - Fenstergröße Spaltenanzahl
+3 - Fenstergröße Zeilenanzahl
+4 - Cursorposition Spalte
+5 - Cursorposition Zeile
+6 - Steuerbyte STBT (vgl. Kapitel 3.6.8 Seite 186)
+7 - Farbe
+8 - Adresse des Reaktionsprogramms bei Fensterende
+9 (SCROLL- oder PAGE-Modus)

Die Initialisierung und der Aufruf eines Fensters erfolgen über Systemunterprogramme 3CH und 3DH (vgl. Kapitel 3.5.6.) bzw. über das CAOS-Kommando WINDOW.

Bei der Einschaltinitialisierung des Systems werden alle 10 Fenster auf maximale Größe, SCROLL-Modus, Farbe weiß/blau und Cursor in HOME-Position eingestellt. Nach dem Einschalten ist Fenster Nr. 0 aktiv (Fenster Nr. 9 bei CAOS 4.1 bis 4.7).

3.2.2. Modulsteuerbytespeicher

Im Modulsteuerbytespeicher sind die Steuerbytes sowohl für die internen Speicherbereiche (Moduladressen 0 bis 7) als auch für jedes Modul (Moduladressen 8 bis FF) enthalten.
Zur Unterstützung der softwaregesteuerten Modulverwaltung ist für jede mögliche Moduladresse ein Speicherplatz für die durch die SWITCH- oder JUMP-Anweisung über Systemrufe ausgegebenen Steuerbytes vorhanden.
Die Adresse berechnet sich wie folgt:

B800H + Moduladresse

Alle Modulsteuerungsausgaben sollten durch Systemaufrufe (UP-Nr. 26H) und nicht durch direkte Ausgaben über die Ausgabeadresse 80H erfolgen. Bei der Initialisierung des Systems erfolgen ein Löschen des gesamten Modulsteuerbytespeichers und ein Eintrag für die internen Speicherblöcke.

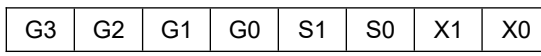
3.3. Modulverwaltung

3.3.1. Verwalten der KC-Module und Zusatzgeräte

Der KC 85/5 ermöglicht es, durch eine spezielle Steuerung, mehrere Module vom gleichen Typ quasi gleichzeitig zu betreiben. Diese können im Grundgerät oder in einem Aufsatz stecken. Mit dem Kommando SWITCH mm kk werden die Module vom Menü, von BASIC oder vom Programm aus geschaltet.
Das Ansprechen der Module erfolgt über die vom Steckplatz abhängigen Modul-adressen.

Die Moduladressen mm sind folgendermaßen definiert:

Modulsteckplatz mm (mm ≥ 8)



Nr. des Submoduls

	X1	X0
einfaches Modul	0	0
Submodul 1	0	1
Submodul 2	1	0
Submodul 3	1	1

Steckplatz im Gerät

	S1	S0	S1	S0
oben	1	1	1	0
unten	0	1	0	0

Gerätenummer

Gerät	G3	G2	G1	G0
Grundgerät	0	0	0	0
1. Aufsatz	0	0	0	1
. . .	.	.	.	.
D004/D008	1	1	1	1

mm . . . . . zweistellige hexadezimale Steckplatzadresse
G3-G0, S1, S0, X1, X0 . . . . . Bits

Die Gerätenummern der Aufsätze entnehmen Sie bitte der jeweiligen Anleitung.

3. SOFTWARE

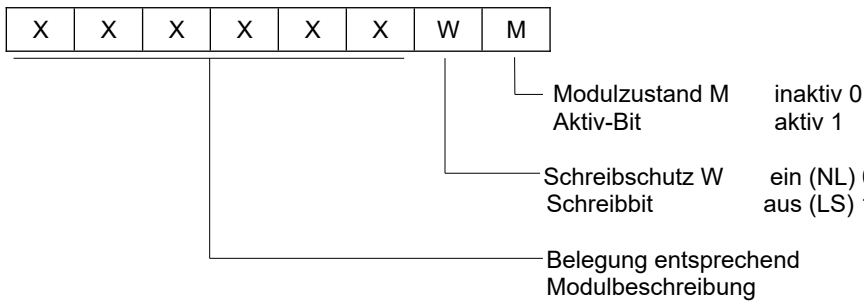
Die Moduladressierung erfolgt über 16-Bit-I/O-Adressen:

Adressbus	High (Register B)	Low (Register C)
	GGGGSSXX	80H

Register B enthält die Modulsteckplatzadresse. Beim Lesen der entsprechenden Adresse sendet jedes Modul ein spezielles Strukturbyte auf den Datenbus. Die Kennungen der Module sind den Modulbeschreibungen zu entnehmen.

Das Schalten der Module erfolgt über Ausgabe eines Steuerbytes kk an die Moduladresse.

Steuerbyte kk



Diese Bits werden mit dem Einschalten von entsprechenden Modulen in einer bestimmten Form belegt. Die Belegung ist den jeweiligen Modulbeschreibungen zu entnehmen. Ab Bit 7 werden bei bestimmten Modulen die Anfangsadresse (Basisadresse) und ab Bit 2 die Speichersegmente eingetragen.

kk . . . . . zweistelliges hexadezimales Steuerbyte
X, W, M . . . . . Bits des Steuerbytes besitzen der Reihenfolge nach sinkende Stellenwertigkeit.

Zum Beispiel wird beim Modul M025 USER PROM 8K ab Bit 7 die absolute Adresse dem Speicherbereich des Moduls zugeordnet. Werden mehrere Module mit gleichen Speicher- oder E/A-Adressen ein geschaltet, so ist beim Zugriff des Prozessors nur das Modul auf der niedrigsten Moduladresse wirksam (Hardware-Prioritätskette).

3.3.2. Verwalten des KC-internen Speichers

Im Bild 24 auf Seite 122 ist die Speicheraufteilung des KC 85/5 dargestellt. Den im Grundgerät enthaltenen Speicherblöcken sind folgende „Modul“-Adressen zugeordnet:

– RAM auf Adresse 0000H - 00H
– IRM-Blöcke auf Adresse 8000H - 01H
– ROM-Blöcke auf Adresse C000H - 02H
– RAM-Blöcke auf Adresse 8000H - 03H
– RAM-Blöcke auf Adresse 4000H - 04H
– CAOS-ROM auf Adresse C000H - 05H
– (reserviert für RAM-Erweiterung - 06H)
– (reserviert für internes V.24-Modul - 07H)

Diese Blöcke werden über die Datenleitungen der Ports A und B des internen PIO-Bausteins sowie über die Ausgabeports 84H und 86H gesteuert. Bei den RAM-Blöcken kann ein Schreibschutz gesetzt werden.

Die internen Speicher (RAM, IRM, ROM) enthalten keine Modulsteuerung, die aktuellen Steuerbytes werden aber in den Modulsteuerbytespeicher eingetragen. Die Bedeutung der einzelnen Bits entnehmen Sie bitte dem Kapitel „Interne Module“ ab Seite 67.

Der IRM ist unterteilt in 4 Ebenen zu je 16 KByte, wobei der Adressbereich von 8000H bis A7FFH als Pixel- und Farbspeicher für die beiden Bilder dient. Im Pixel-RAM von Bild 0 liegen ab Adresse A800H Arbeitszellen, welche unabhängig vom Schaltzustand der IRM-Ebenen sichtbar sind, damit die Programme immer auf diese Arbeitszellen Zugriff haben. Die Speicherbereiche der anderen drei IRM-Ebenen werden als „versteckte“ IRM-Bereiche bezeichnet. Auf diese kann nur unter speziellen Bedingungen zugegriffen werden:

CAOS-ROM C = ein	(Ausgabeport 86H:	Bit 7 = 1)
CAOS-ROM E = aus	(PIO-Port A:	Bit 0 = 0)

Da alle Interruptroutinen von CAOS in den ROM E führen, darf der CAOS-ROM E nur bei gesperrtem Interrupt (DI) abgeschaltet werden.
Von CAOS werden die versteckten IRM-Bereiche für 2 Anwendungsfälle genutzt:

– ab CAOS 4.3 für eine zweite virtuelle RAM4-Ebene von 16 KByte, vgl. dazu die Ausführungen zum RAM4 auf Seite 69.

– ab CAOS 4.6 als Zwischenspeicher für den CAOS-Taschenrechner CALC, vgl. dazu die Ausführungen auf Seite 85.

Es gibt folgende Zuordnungen zu den internen E/A-Ports:

PIO Port A (Daten: Adresse 88H, Steuerwort: 8AH):

Bit 0 - CAOS-ROM E
Bit 1 - RAM0
Bit 2 - IRM
Bit 3 - Schreibschutz RAM0 (1 = Schreibschutz aus)
Bit 4 - K OUT (Ausgang zu Keyboard)
(/NMI bei KC 85/2 und KC 85/3)
Bit 5 - LED „SYSTEM“ („TAPE“ bei KC 85/2 und KC 85/3)
Bit 6 - Motorschaltspannung (Schnellstopp) des Recorders
Bit 7 - USER-ROM C (BASIC-ROM bei KC 85/3 und /4,
CAOS-ROM F beim KC 85/2)

PIO Port B (Daten: Adresse 89H, Steuerwort: 8BH):

Bit 0 - Rücksetzen der Symmetrie-Flip-Flops für Tonausgabe (bei KC 85/4+5)
Bit 1 -
Bit 2 - } Lautstärkeregelung für Tonausgabe (4 Bits, Low = aktiv)
Bit 3 - Bei KC 85/2+3 wird die Lautstärke mit 5 Bits (Bit 0-4) geregelt
Bit 4 -
Bit 5 - RAM8 CAOS 3.4: Bit 5+6 USER-ROM
Bit 6 - Schreibschutz RAM8 (1 = Schreibschutz aus)
Bit 7 - blinken der Vordergrundfarbe ein/aus

Ausgabe Port 84H bzw. (IX + 1)

Bit 0 - Anzeige Bild 0 oder 1
Bit 1 - Zugriff auf Pixel- oder Farbebene (Pixel = 0 oder Farbe = 1)
Bit 2 - Zugriff auf Bild 0 oder 1
Bit 3 - hohe Farbauflösung ein/aus (0 = hohe oder 1 = niedrige)
Bit 4 -
Bit 5 - } Auswahl der RAM8-Ebene (4 Bit = 16 Ebenen)
Bit 6 -
Bit 7 -

Ausgabe Port 86H bzw. (IX + 4)

Bit 0 - RAM4
Bit 1 - Schreibschutz RAM4 (1 = Schreibschutz aus)
Bit 2 - Systemumschaltung, Reset-Impuls *17
Bit 3 - Systemumschaltung, Zähl-Impuls *17 } Bank bei 2MB RAM-Erweiterung
Bit 4 - Umschaltung Zeichensatz *17
Bit 5 - } Auswahl der USER-ROM Ebene
Bit 6 - } 0 = EDIT, 1 = ASM, 2 = TEMO, 3 = BASIC
Bit 7 - CAOS-ROM C (im Normalfall abgeschaltet)

*17 Bei CAOS 4.8 für softwaregesteuerte Flash-ROM-Erweiterung

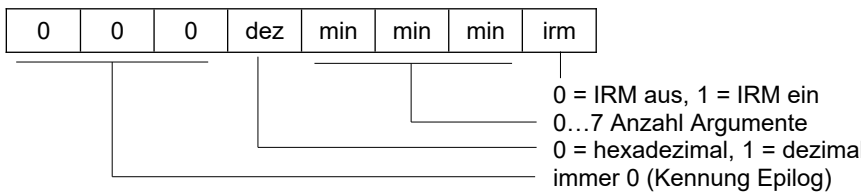
3.4. Menükonzept

3.4.1. Erweiterung des CAOS-Menüs

Das verwendete Menükonzept ist unabhängig von bestimmten Speicherplätzen, das heißt, jedes Programm auf beliebigen Speicherplätzen kann mit entsprechendem „Vorspann“ ins Menü eingetragen und über dieses gestartet werden.

Vorspann:	7FH	
	7FH	Prolog
	NN	Neues Menüwort:
	:	Beliebig lange Zeichenkette
	:	aus Großbuchstaben, Kleinbuchstaben, Ziffern und
	MM	Doppelpunkt (ASCII)
	00 bis 1F	Epilogbyte
Programm:	PP	1. Befehlsbyte des Programms
	.	
	.	
	C9	Letztes Befehlsbyte des Programms (RET- Rückkehr ins Menü)

Der Epilog kann Werte von 0 bis 1FH annehmen. Dies ermöglicht die Erkennung, wie viele und welche Argumente für ein Kommando erforderlich sind. Dabei bedeuten die einzelnen Bits:



Bis CAOS 4.2 (mit Ausnahme von CAOS 2.5) waren nur die beiden Epilogbytes 0 und 1 zulässig, diese sind uneingeschränkt weiter gültig. Durch geeignete Kombination kann der Service der Argumentkontrolle auch für eigene Programme genutzt werden. Programme mit Epilogbytes > 1 können aber nur unter CAOS 2.5 und ab CAOS 4.3 gestartet werden.

*Bei CAOS 2.5 und 4.3 bis 4.7 wurde mit Bit 4=1 festgelegt, dass genau so viele Argumente einzugeben sind wie durch die Bits 1 bis 3 festgelegt ist. Ab CAOS 4.8 bedeutet ein gesetztes Bit 4, dass die Argumente als Dezimalzahl anzugeben sind, wie beispielsweise beim Kommando WINDOW auf Seite 62.

Eine Besonderheit bildet das Epilogbyte 1FH: Es steht nicht für „mindestens 7 dezimale Argumente“, sondern ermöglicht die Übergabe von nicht-numerischen Argumenten an das aufrufende Programm. Die Argumentliste wird dabei vom

Kommandointerpreter nicht getestet und dem aufgerufenen Programm als einziger (!) Parameter im Registerpaar DE die VRAM-Adresse nach dem Kommandowort übergeben. Ein Beispiel dafür ist das DIR-Kommando ab CAOS 4.5.

Die mittels Menütechnik zu startenden Maschinenprogramme müssen als Unterprogramme definiert sein, d. h., sie müssen mit RETURN abgeschlossen sein. Die Unterprogramme werden bei Anwahl durch das Menüwort auf dem ersten Byte nach dem Epilog gestartet.
Für die Zeichenkette des Programmnamens gilt, dass bei Großbuchstaben, Ziffern und dem Doppelpunkt (Codes 30H bis 3AH und 41H bis 5AH) der Eintrag ins Menü auf dem Bildschirm erfolgt.
Bei Sonderzeichen, Groß- und Kleinbuchstaben sowie Ziffern (Codes 20H bis 7FH) ist der Aufruf über den Namen möglich. Es kann z. B. der Name eines Programms aus Groß- und Kleinbuchstaben bestehen. Dabei erfolgt keine Ausschrift im Menü.

Die **Menüwortsuche** erfolgt im aktuell zugeschalteten Speicher von C000H bis BFFFH. Seit CAOS 4.3 werden danach noch der CAOS-ROM C und schließlich die drei zusätzlichen Ebenen des USER-ROM C durchsucht.

- Steht das Menüwort im CAOS-ROM C, so wird dieser für die Abarbeitung des Kommandos eingeblendet, der Schaltzustand des USER-ROM C bleibt dabei erhalten.
- Wurde das Menüwort in einer Ebene des USER-ROM C gefunden, dann wird der CAOS-ROM C automatisch abgeschaltet, falls er zugeschaltet war. Der ursprüngliche Schaltzustand des USER-ROM wird ab CAOS 4.6 in der Speicherzelle ARGC abgelegt. Damit kann das dort laufende Programm bei Programmende den Schaltzustand regenerieren. Das Programm hat selbst dafür zu sorgen, dass ARGC nicht verändert wird bzw. dessen Inhalt anderweitig zu sichern, falls diese Funktionalität genutzt werden soll.

3.4.2. Übernahme von Parametern

Für Menüworte mit Epilogbytes von 0 bis 1EH können beim Programmaufruf auf der gleichen Bildschirmzeile bis zu 10 Argumente übergeben werden. Die Anzahl der Argumente wird im Speicherplatz ARGN abgelegt. Die Werte der Argumente sind als 16-Bit-Zahlen auf den Speicherplätzen ARG1 bis ARG10 abgelegt.
Gleichzeitig werden die ersten drei Argumente in diese Register geladen:

A = ARGN*,
HL = ARG1,
DE = ARG2,
BC = ARG3

und können von den jeweiligen Unterprogrammen verwendet werden.
*Die Anzahl der Argumente mit A = ARGN wird seit CAOS 3.3 übergeben.

3.4.3. Fehlermeldungen

Seit CAOS 4.3 gibt die Menükommandoroutine Fehlermeldungen im Klartext aus, dabei bedeuten:

falsches Kommando	Menüwort weder im aktiven Speicher, noch in den ROM C-Segmenten gefunden.
fehlerhafte Argumente	Formatfehler in den dezimalen bzw. hexadezimalen Argumenten.
zu wenig Argumente	für das Menüwort sind mehr Argumente anzugeben.
zu viele Argumente	für das Menüwort ist eine feste Anzahl von Argumenten vorgeschrieben, die überschritten wurde. (nur bei CAOS 4.3 bis CAOS 4.7)
kein D004!	Es wurde ein Diskettenkommando aufgerufen, obwohl kein D004 oder D008 an das KC-Grundgerät angeschlossen ist.
CAOS-Disk starten!	Vor der Arbeit mit den Menüworten des Disk-Treibers muss die CAOS-Betriebsart mit DEP gestartet werden!
kein DEP2!	die DRIVE-Funktion kann erst ab DEP in der Version 2.0 benutzt werden.
keine MC-Datei!	mit LOAD wurde versucht, eine Datei zu laden, welche keinen gültigen Vorblock für MC-Programme enthält.
kein Modul!	Der im JUMP-Kommando angegebene Steckplatz enthält kein Modul!

Die Menüworte des DISK-Treibers geben die Fehlermeldungen des DEP aus. Wenn DEP ab der Version 2.0 benutzt wird im Klartext, ansonsten mit den üblichen Nummern.

3.4.4. Programmbeispiele

Für das Umspeichern von Speicherbereichen an eine andere Stelle ist das Programm „COPY“ in das Menü aufzunehmen.

Folgendes Assemblerprotokoll nimmt das Umspeichern vor:

ADR.	MC	Anweisung	Bemerkung
0000	7F 7F	DEFW 7F7FH	; Prolog
0002	43 4F 50 59	DEFM 'COPY'	; Menüwort
0006	01	DEFB 1	; Epilog
0007	ED B0	LDIR	; Umspeichern
0009	C9	RET	; Rücksprung ins CAOS

Der Maschinencode (MC) ist mit dem Kommando MODIFY, z. B. ab Adresse 0 hexadezimal einzugeben und anschließend das Menü durch MENU aufzurufen. Im Menü erscheint nun auch das Kommandowort COPY. Es kann z. B. wie folgt verwendet werden:

Kopieren der Zeichenbildtabelle 1 aus dem ROM (Anfangsadresse 0EE00H) in den RAM ab Adresse 2000H mit einer Länge von 512 Byte (200H).

```
COPY EE00 2000 200
     (HL) (DE) (BC)
```

Für Anwenderprogramme mit eigenem Menüfeld ist es möglich, den Standardprolog 7FH durch Eintragen eines neuen Prologbytes in (IX+9) zu ändern. Dadurch werden nur die Kommandos gezeigt, welche den neuen Prolog enthalten. Es können auch nur diese Programme über Kommando gestartet werden. Als Prolog sollten Bytes genutzt werden, welche möglichst nicht mehrfach hintereinander in Programmen, Texten oder Bitmaps auftreten (z. B. B0H bis B7H = OR n, n … Register). Diese Befehle werden kaum zweimal hintereinander im Programm auftreten und sind auch nicht mit ASCII-Zeichen zu verwechseln.

Die folgende Übersicht zeigt einige bereits verwendete Prologbytes:

Tabelle 19: Beispiele bekannter Prologbytes

Prologbyte	**Verwendung**
3DH	M052, USB-Terminal bis CAOS 4.2
3EH	M052, USB-Terminal ab CAOS 4.3
7FH	CAOS Systemprolog
A1H	JEDIT
BBH	Spiel HOUSE
BCH	Spiel PENGO
DDH	EDAS, ASM, EDIT, SPS-Grafik
DFH	Systemprolog bei CAOS 2.4
EDH	RAMDOS, COPY
FDH	Debugger TEMO

Beispiel:

```
        DEFW    7F7FH           ; Standardprolog
        DEFM    'NEWMENU'
        DEFB    1
        LD      (IX+9),0B0H     ; neuen Prolog eintragen
        JR      MENU            ; Menüanzeige
        ;
        DEFW    0B0B0H          ; Alternativprolog
        DEFM    'BYE'           ; Rückstellen auf
        DEFB    1               ; Standardprolog
        LD      (IX+9), 7FH     ; Standardprolog eintragen
        JR      MENU            ; Menüanzeige
        ;
        DEFW    0B0B0H          ; Alternativprolog
        DEFM    'MENU'
        DEFB    1               ; Menüanzeige
        ;
MENU:   POP     HL              ; Stack bereinigen
        LD      A,12            ; CLS
        CALL    0F003H          ; PV 1
        DEFB    0               ; UP 0          (CRT)
        CALL    0F003H          ; PV 1
        DEFB    46H             ; UP 46H        (MENU)
```

Hinweis: Der Aufruf des CAOS-Unterprogramms 46H führt zur Menükommandoroutine also zur Eingabeschleife für neue Kommandos. Eine Rückkehr zum aufrufenden Programm erfolgt nicht. Deshalb ist nach diesem Aufruf kein RET-Befehl erforderlich.

3.5. Systemschnittstellen und nutzbare CAOS-Adressen

3.5.1. Einsprungadressen für Systemstart

Um den Anwendern des Kleincomputers KC 85/5 die Arbeit zu erleichtern, stellt das Betriebssystem CAOS spezielle Systemunterprogramme zur Verfügung. Der Aufruf dieser Systemunterprogramme (UP) wird über Programmverteiler gesteuert.
Das Betriebssystem enthält eine Liste, in der alle UP nummeriert sind. Dem Programmverteiler muss als Parameter diese UP-Nummer übertragen werden, damit wird das entsprechende UP gestartet. Für den Anwender sind im wesentlichen 16 Adressen des Betriebssystems interessant:

E000H: RESET-Adresse: Diese Adresse wird beim Tasten-RESET des KC 85/5 angesprungen. Der Systemspeicher wird neu initialisiert. Der Anwenderspeicher bleibt erhalten.

F000H: POWER ON RESET: Diese Adresse wird beim Einschalten des KC 85/5 angesprungen. Der komplette RAM-Speicher wird gelöscht, alle Module werden abgeschaltet und das System wird initialisiert.

F012H: Einsprungadresse des Systems bei JUMP (wie E000H, jedoch ohne Initialisierung des Grundgeräte-PIO-Bausteins).

3.5.2. Spezielle CAOS-Adressen

E011H: BASIC-Menüwort: Auf dieser Adresse steht seit dem KC 85/4 das Prologbyte 7FH vom Menüwort BASIC. Da der BASIC-ROM bei den Vorgängern KC 85/2 und KC 85/3 standardmäßig eingeschaltet war, ist bei diesen Rechnern kein zusätzliches Menüwort im ROM-E erforderlich. Der Inhalt der Adresse E011H wird deshalb häufig zur Unterscheidung des Rechnertyps herangezogen.

EDFFH: Ab CAOS 4.1 steht auf dieser Speicherzelle die BCD-codierte Versionsnummer von CAOS, der Wert 42H entspricht z. B. CAOS 4.2.

FDF8H: Ab CAOS 3.4 steht auf dieser Adresse eine allgemein nutzbare Bittabelle mit den Werten 80H, 40H, 20H, 10H, 8, 4, 2, 1.

Beispiel: Abfrage der CAOS-Version

Zur Versionsabfrage können vom Anwenderprogramm die folgenden beiden Speicheradressen ausgewertet werden:

```
VERS:   LD      A,(0E011H)      ; ab KC 85/4 steht hier Menüwort „BASIC“
        CP      7Fh             ; KC 85/4 ?
        LD      A,0             ; KC 85/2 und KC 85/3 = Version 0.0
        RET     NZ
        LD      A,(0EDFFH)      ; CAOS-Versionsnummer
        RET
```

3.5.3. Schalter für IRM und STACK

Diese Gruppe der Programme schaltet den IRM und verändert den STACK.

F018H: Einschalten des IRM und Setzen des Stackpointers auf (SYSP).
Der Standardwert von (SYSP) ist 01C4H (bis CAOS 3.3 = 01D4H).
Programm darf nur zusammen mit F01BH verwendet werden.

F01BH: Abschalten des IRM und Rückstellen des Stackpointers.
Diese beiden Programme werden von BASIC genutzt.

Für die Programme F018H und F01BH gilt: Der Registerinhalt von BC und F geht verloren. Der Stackpointer SP wird im Register IY zwischengespeichert, das heißt während der IRM eingeschaltet ist, darf das Register IY vom Anwenderprogramm nicht verändert werden. Diese Programme werden vor allem von BASIC benutzt und stehen ab CAOS 3.1 zur Verfügung!

3.5.4. Programmverteiler (PV)

F003H: Programmverteiler PV1
Nur bei diesem Programmverteiler erfolgt die Parameterübergabe vom Unterprogramm an das Hauptprogramm für die Register BC, DE, HL und AF. Die Unterprogrammnummer muss im rufenden Programm unmittelbar nach dem CALL-Befehl notiert werden.

Beispiel: CALL 0F003H
DEFB UP-Nr. (Unterprogrammnummer)

Die Parameter für die UP werden in den Registern übergeben. Die Register werden entsprechend den Unterprogrammen verändert.
Der IRM muss bei Aufruf von PV1 eingeschaltet sein!

F006H: Programmverteiler PV2
Dieser Programmverteiler entspricht dem PV1. Die UP-Nr. wird jedoch im IRM auf einer festgelegten Adresse übergeben (ARGC vgl. Kapitel 3.6.8.). Die Register BC, DE, HL werden gerettet. Es werden keine Parameter in den Registern BC, DE, HL vom UP zurückgegeben.
Der IRM muss bei Aufruf von PV2 eingeschaltet sein!

F009H: Programmverteiler PV3
Funktion und Register wie bei PV2. Die UP-Nr. wird im Register E übergeben. Damit entfällt allerdings das Register E für die Parameterübergabe.
Der IRM muss bei Aufruf von PV3 eingeschaltet sein!

F00CH: Programmverteiler PV4
Funktion und Register wie bei PV3, jedoch mit Einschalten des IRM beim Aufruf und Abschalten des IRM beim Rücksprung. Das Ein- und Abschalten des IRM erfolgt dabei nicht durch die Programme auf Adresse F018H/F01BH. Das heißt, sowohl Stackpointer als auch das IY-Register werden vom PV4 nicht verändert.

F00FH: Relativer Unterprogrammaufruf (für verschiebliche Programme).
Mit UP-Abstand unmittelbar nach Aufruf.

z. B.	RCALL UP	
entspricht:	CALL 0F00FH	
	DEFW UP-NEXT	Differenz zwischen Unterprogrammadresse und Adresse des nächsten Befehls, wird vom Assembler eingetragen.
NEXT:	(nächster Befehl)	

Es werden keine Register verändert.
Bis CAOS 4.2 wird das DE-Doppelregister verändert und damit nicht übergeben!

F015H: Programmverteiler PV5:
Aufruf des Programmverteilers PV3 mit Einschalten des IRM und Setzen des Stackpointers auf (SYSP) vor UP-Aufruf und Ausschalten des IRM sowie Rückstellen des Stackpointers nach dem UP-Aufruf. PV5 wird von BASIC benutzt.
Programmverteiler PV5 steht ab CAOS 3.1 zur Verfügung!

F01EH: Programmverteiler PV6:
Wie Programmverteiler PV5, jedoch UP-Nr.-Übergabe im IRM (ARGC). Beim Eintragen der UP-Nr. muss der IRM ebenfalls eingeschaltet werden!
Programmverteiler PV6 steht ab CAOS 3.1 zur Verfügung!

F021H: Programmverteiler PV7:
Aufruf einer DEVICE-Treiber-Funktion. Die Parameterübergabe erfolgt wie bei PV1 mit dem Byte, welches dem CALL-Befehl folgt: Es gilt jedoch eine andere Nummerierung der Unterprogramme! Aufgerufen werden über diesen PV7 ausschließlich Device-Routinen in Abhängigkeit vom aktuell eingestellten DEVICE in Bit 2–4 von (IX+8). Die gleich lautenden DEVICE-Routinen können auch über PV1 bis PV6 aufgerufen werden, sind darüber jedoch etwas langsamer, da eine zusätzliche Adressberechnung erforderlich ist. Für neue Software sollte bevorzugt der PV7 benutzt werden, für kompatible Software die PV1-6.
Der IRM muss bei Aufruf von PV7 eingeschaltet sein!

Beispiel: CALL 0F021H
DEFB UP-Nr. (Treiber-Programmnummer)

Diese Funktion steht ab CAOS 4.7 zur Verfügung.
In CAOS 4.3 bis 4.5 war an dieser Stelle ein zu SERVICE.KCC kompatibler Floppy-Treiber enthalten. In CAOS 4.6 ist auf dieser Adresse keine Funktion verfügbar!

Das aktuelle DEVICE wird in Bit 2–4 von (IX+8) gespeichert, siehe Seite 185. Um eine Dateiauswahl und Fehlerauswertung bei „Nicht-TAPE-Geräten“ realisieren zu können, wurden einige Unterprogramme bezüglich der Parameter ergänzt.

3.5.5. Veränderung der Unterprogrammtabelle SUTAB

Die Programmverteiler 1 bis 6 realisieren den Unterprogrammaufruf über eine Tabelle der Anfangsadressen dieser Unterprogramme. Die Anfangsadresse der Tabelle steht in der Speicherzelle SUTAB (Adresse B7B0H im IRM). Die Unterprogrammtabelle befindet sich bis CAOS 4.5 im ROM, ab CAOS 4.6 im IRM. Soll diese Tabelle verändert oder erweitert werden, ist wie folgt vorzugehen:

1. Bestimmen der Anfangsadresse der aktuellen Unterprogrammtabelle aus der Speicherzelle SUTAB.
2. Test, ob die Tabelle im RAM oder ROM steht. Falls sich die Tabelle bereits im RAM befindet, können Adressen direkt geändert und die Punkte 3 und 4 übersprungen werden.
3. Umspeichern der Tabelle in den RAM in der Länge 2 * Anzahl der UP-Nr. (Länge bei CAOS 4.7: 2*4AH=94H). Die Länge der UP-Tabelle steht seit CAOS 4.8 in der Adresse unmittelbar vor der SUTAB, also auf Adresse AA00H.
4. Eintragen der neuen Anfangsadresse der Unterprogrammtabelle in die Speicherzelle SUTAB.
5. Ergänzen/Ändern von Unterprogrammen.

Die zur Verfügung stehenden CAOS-Unterprogramme werden im folgenden Kapitel beschrieben. Für die Erweiterung der SUTAB sind ab CAOS 4.8 einige Bytes unmittelbar nach der SUTAB im IRM freigehalten, also reserviert für diesen Zweck.

3.5.6. Liste der nutzbaren Unterprogramme für PV1-PV6

Legende

Name:	**Name des Unterprogramms (UP) für PV1-6**
UP-Nr.:	**Nummer des UP (hexadezimal)**
FKT.:	Beschreibung der Funktion
PE:	Parameterübergabe vom Hauptprogramm an UP, vor Aufruf
PA:	Parameterübergabe nach RETURN des UP, nur bei PV1
VR:	Veränderte Register
Bemerkung:	Hinweise zu verschiedenen CAOS-Versionen Hinweise zu DEVICE-Funktionen

Name: . . . CRT . . . . . UP-Nr. 00H

FKT.:	Zeichenausgabe auf Bildschirm	
PE:	Register A	- Zeichencode (ASCII)
PA:	-	
VR:	-	
Bemerkung:	vgl. auch UP-Nr. 24H	

Name: . . . MBO . . . . . UP-Nr. 01H

FKT.:	Ausgabe Datenblock von Puffer auf Datenträger (128 Byte)	
PE:	Register BC	- Länge Vorton
	(IX+2)	- Blocknummer -1
	(IX+5)	- L (Pufferadresse)
	(IX+6)	- H (Pufferadresse)
PA:	Register HL	= Pufferende + 1
	Register DE	= Pufferende + 1 bei CY=0 Fehlercode, bei CY=1 *
	(IX+2)	= Block-Nr.
	CY=1	= Fehler *
VR:	AF, BC, DE, HL	
Bemerkung:	* Bis CAOS 4.5 ist CY unbestimmt. Fehlermeldungen werden auf dem Bildschirm angezeigt. Ab USB-Treiber-Version 3.0 wird im Fehlerfall in Register DE ein Fehlercode zurückgegeben.	
Ab CAOS 4.7:	Diese Device-Funktion führt zu PV7, UP-Nr. 0 und ist darüber auch direkt aufrufbar.	

Name: . . . UOT1 . . . . . UP-Nr. 02H

FKT.: Ausgabe auf Anwenderkanal 1 (z. B. Druckerausgabe)
PE: Register A - Zeichencode
PA/VR: entsprechend der Routine
Bemerkung: Adresse der selbst zu erstellenden Routine muss auf UOUT1 (Speicherzellen B7BEH und B7BFH) eingetragen werden.

Name: . . . UOT2 . . . . . UP-Nr. 03H

FKT.: Ausgabe auf Anwenderkanal 2 (z. B. V.24-Ausgabe)
PE: Register A - Zeichencode
PA/VR: entsprechend der Routine
Bemerkung: Adresse der selbst zu erstellenden Routine muss auf UOUT2 (Speicherzellen B7C4H und B7C5H) eingetragen werden.

Name: . . . KBD . . . . . UP-Nr. 04H

FKT.: Tasteneingabe mit Einblendung des Cursors, wartet, bis Taste gedrückt bzw. liefert die Codefolge von vorher betätigter F-Taste
PE: -
PA: Register A = Zeichencode (ASCII)
VR: AF, HL
Bemerkung: vgl. auch UP-Nr. 16H

Name: . . . MBI . . . . . UP-Nr. 05H

FKT.: Einlesen Datenblock von Datenträger in den Puffer (128 Byte)
PE: (IX+5) - L (Pufferanfang)
(IX+6) - H (Pufferanfang)
PA: (IX+2) = Block-Nr.
CY = 1 = Block fehlerhaft
Register DE = Fehlercode, bei CY=1 *
VR: AF, BC, (DE im Fehlerfall)
Bemerkung: * Fehlermeldungen werden auf dem Bildschirm angezeigt.
Ab USB-Treiber-Version 3.0 wird im Fehlerfall in Register DE ein Fehlercode zurückgegeben.
Ab CAOS 4.7: Diese Device-Funktion führt zu PV7, UP-Nr. 1 und ist darüber auch direkt aufrufbar.

Name: . . . USIN1 . . . . . UP-Nr. 06H

FKT.: Eingabe von Anwenderkanal 1 (z. B. V.24-Eingabe)
PA: Register A - Zeichencode
PE/VR: entsprechend der Routine
Bemerkung: Adresse des selbst zu erstellenden Programms muss in UIN1 (Speicherzellen B7C1H und B7C2H) eingetragen werden.

Name: . . . USIN2 . . . . . UP-Nr. 07H

FKT.: Eingabe von Anwenderkanal 2 (z. B. V.24-Eingabe)
PA: Register A - Zeichencode
PE/VR: entsprechend der Routine
PE/PA/VR: entsprechend der Routine
Bemerkung: Adresse des selbst zu erstellenden Programms muss in UIN2 (Speicherzellen B7C7H und B7C8H) eingetragen werden.

Name: . . . ISRO . . . . . UP-Nr. 08H

FKT.: Initialisierung der Dateiausgabe, Ausgabe des ersten Blockes (Block-Nr. 01H) mit langem Vorton *
PE: Register HL - Zeiger auf Dateinamen/Pfad *
(IX+5) - L (Pufferadresse)
(IX+6) - H (Pufferadresse)
PA: Register HL = Pufferende + 1
Register DE = Pufferende + 1 bei CY=0
Fehlercode, bei CY=1 *
(IX+2) = Block-Nr.
CY=1 = Fehler *
VR: AF, BC, DE, HL
Bemerkung: * Vorton bei CAOS 2.2 und 3.1: 8.192 Schwingungen
bei CAOS 3.3, 4.1-4.8: 4.096 Schwingungen
CAOS 3.4, OS pi/88 und OS pi/90: 2.560 Schwingungen
* Bis CAOS 4.5 ist CY unbestimmt.
In Register HL muss ab CAOS 4.7 der Dateiname übergeben werden, falls DEVICE nicht TAPE ist. Bei CAOS 4.6 wurde der Dateiname aus dem Vorblock entnommen, das war aber nicht immer sichergestellt. Ab USB-Treiber-Version 3.0 kann dem Dateinamen ein Pfad vorangestellt sein. Im Fehlerfall wird in Register DE ein Fehlercode zurückgegeben. Fehlermeldungen werden auf dem Bildschirm angezeigt.
Ab CAOS 4.7: Diese Device-Funktion führt zu PV7, UP-Nr. 2 und ist darüber auch direkt aufrufbar.

Name: . . . CSRO . . . . . UP-Nr. 09H

FKT.: Abschluss-(Close-)Routine für Dateiausgabe, Ausgabe des letzten Blockes (Block-Nr.: FFH)

PE:
- Register BC - Länge Vorton
- (IX+5) - L (Pufferadresse)
- (IX+6) - H (Pufferadresse)

PA:
- Register HL = Pufferende + 1
- Register DE = Pufferende + 1 bei CY=0, Fehlercode, bei CY=1 *
- (IX+2) = Block-Nr.
- CY=1 = Fehler *

VR: AF, BC, DE, HL

Bemerkung: * Bis CAOS 4.5 ist CY unbestimmt.
Bis CAOS 4.5 wurde innerhalb von CSRO noch ein Zeilenvorschub CR+LF auf dem Bildschirm ausgegeben.
Fehlermeldungen werden auf dem Bildschirm angezeigt.
Ab USB-Treiber-Version 3.0 wird im Fehlerfall in Register DE ein Fehlercode zurückgegeben.

Ab CAOS 4.7: Diese Device-Funktion führt zu PV7, UP-Nr. 3 und ist darüber auch direkt aufrufbar.

Name: . . . ISRI . . . . . UP-Nr. 0AH

FKT.: Initialisierung Magnetbandeingabe / Datei öffnen und Einlesen des 1. Blockes

PE:
- Register HL - Zeiger auf Dateiname/Pfad *
- (IX+5) - L (Pufferadresse)
- (IX+6) - H (Pufferadresse)

PA:
- (IX+2) = Block-Nr.
- CY = 1 = Block fehlerhaft
- Register DE = Fehlercode, bei CY=1 *

VR: AF, BC, (DE im Fehlerfall) vgl. Unterprogramm MBI

Bemerkung: * Register HL muss ab CAOS 4.6 übergeben werden, falls DEVICE nicht TAPE ist. Ab USB-Treiber-Version 3.0 kann dem Dateinamen ein Pfad vorangestellt sein. Im Fehlerfall wird in Register DE ein Fehlercode zurückgegeben. Fehlermeldungen werden auf dem Bildschirm angezeigt.

Ab CAOS 4.7: Diese Device-Funktion führt zu PV7, UP-Nr. 4 und ist darüber auch direkt aufrufbar.

Name: . . . CSRI . . . . . UP-Nr. 0BH

FKT.: Abschluss der Magnetbandeingabe / Datei schließen
PE: -
PA: CY = 1 = Fehler *
Register DE = Fehlercode, bei CY=1 *
VR: AF, HL, (DE im Fehlerfall)
Bemerkung: * Fehlerauswertung nur bei DEVICE > 0 und ab CAOS 4.7, sonst CY unbestimmt bei Rückkehr. Fehlermeldungen werden auf dem Bildschirm angezeigt.
Ab USB-Treiber-Version 3.0 wird im Fehlerfall in Register DE ein Fehlercode zurückgegeben.
Ab CAOS 4.7: Diese Device-Funktion führt zu PV7, UP-Nr. 5 und ist darüber auch direkt aufrufbar.

Name: . . . KBDS . . . . . UP-Nr. 0CH

FKT.: Tastenstatusabfrage ohne Quittierung der Taste
PE: -
PA: CY = 0 => keine Taste gedrückt, dann Register A unverändert
CY = 1 => Taste gedrückt, dann A = Zeichencode (ASCII)
VR: AF
Bemerkung: Funktionstasten liefern Codes F1H – FCH.

Name: . . . BYE . . . . . UP-Nr. 0DH

FKT.: Sprung auf RESET (Warmstart des Systems)
PE/PA/VR: -
Bemerkung: entspricht Sprung zu Adresse E000H

Name: . . . KBDZ . . . . . UP-Nr. 0EH

FKT.: Tastenstatusabfrage mit Quittierung der Taste (Autorepeat)
PE: -
PA: CY = 0 => keine Taste gedrückt, dann Register A unverändert
CY = 1 => Taste gedrückt, dann A = Zeichencode (ASCII)
VR: AF
Bemerkung: Funktionstasten liefern die Codes F1H - FCH.

Name: . . . COLORUP . . . . UP-Nr. 0FH

FKT.:	Farbe einstellen	
PE:	Register E	- Hintergrundfarbe (0…7)
	Register L	- Vordergrundfarbe (0…1FH)
	(ARGN) = 1	- nur Vordergrundfarbe
	= 2	- Vorder- und Hintergrundfarbe
PA:	-	
VR:	AF, L	

Name: . . . LOAD . . . . UP-Nr. 10H

FKT.:	Einlesen von Maschinenprogrammen vom eingestellten DEVICE. Dateiname und Adressen sowie Blocknummern werden während des Einlesens angezeigt.	
PE:	(ARGN) = 0	- LOAD ohne Offset
	= 1	- LOAD mit Offset
	= 2	- Autostart unterdrücken *
	(ARG1)	- Ladeoffset
	Register HL	- Zeiger auf Dateiname/Pfad * (8 Zeichen für Namen, 3 Zeichen für Typ)
PA:	CY = 1	= Fehler * (Lesefehler, BRK, ungültiger Vorblock, Datei nicht vorhanden)
VR:	AF, BC, DE, HL	
Bemerkung:	* Autostart unterdrücken mit (ARGN)=2 ab CAOS 4.3, * Fehler werden ab CAOS 4.7 und ohne Autostart mit CY=1 zurückgemeldet, vorher war CY unbestimmt. * ab CAOS 4.6 ist HL=Dateiname zum Laden bei allen Devices außer Kassette erforderlich. Ab Treiber-Version 3.0 kann bei USB dem Dateinamen ein Pfad vorangestellt sein. Im Fehlerfall wird in Register DE ein Fehlercode (siehe Seite 169) zurückgegeben.	

Name: . . . VERIF . . . . UP-Nr. 11H

FKT.:	Überprüfung von Kassettenaufzeichnungen auf Übereinstimmung der Prüfsumme über die Datenblöcke und aufgezeichnete Prüfsumme
PE/PA:	-
VR:	AF, BC, DE, HL

Name: . . . LOOP . . . . UP-Nr. 12H

FKT.:	Rückgabe der Steuerung an CAOS ohne Speicherinitialisierung. Dieses Programm kann bei Menüprogrammen genutzt werden, wenn ein RET-Befehl nicht mehr möglich ist.
PE/PA/VR:	-

Name: . . . NORM . . . . . UP-Nr. 13H

FKT.: Rückschalten des Ein- und Ausgabekanals auf CRT und KBD
PE: -
PA: Register HL = alter Ausgabezeiger
VR: HL

Name: . . . WAIT . . . . . UP-Nr. 14H

FKT.: Warteschleife
PE: Register A t = (A) * 6 ms
PA: -
VR: AF, B
Bemerkung: Programmschleife arbeitet ohne Interrupt, bei A=0 wird eine maximale Wartezeit von ca. 1,5 Sekunden erreicht (256 * 6ms)

Name: . . . LARG . . . . . UP-Nr. 15H

FKT.: Lade Register mit Argumenten
PE: -
PA: Register HL = (ARG1)
Register DE = (ARG2)
Register BC = (ARG3)
Register A = (ARGN) *
VR: A, BC, DE, HL
Bemerkung: * Register A wird erst ab CAOS 3.1 geladen

Name: . . . INTB . . . . . UP-Nr. 16H

FKT.: Eingabe eines Zeichens vom aktuellen Eingabekanal (über INTAB definiert)
PE: -
PA: Register A = Zeichencode (ASCII)
VR: -

Name: . . . INLIN . . . . . UP-Nr. 17H

FKT.: Eingabe einer Zeile mit Funktion aller Cursortasten, Abschluss mit <ENTER> oder Abbruch <BRK>

PE: -

PA:
- Register DE = Adresse des Zeilenanfangs des eingestellten Fensters im Video-RAM
- CY=0 = Eingabe mit Enter abgeschlossen
- CY=1 = Abbruch mit BRK *

VR: AF, DE

Bemerkung: * Unterscheidung BRK/Enter ist bei CAOS 3.4 und ab CAOS 4.3 verfügbar, bei den anderen CAOS-Versionen wird INLIN nur mit <ENTER> abgeschlossen, CY ist dabei unbestimmt.

Name: . . . RHEX . . . . . UP-Nr. 18H

FKT.: Umwandlung einer Zeichenkette (Hexadezimalzahl) in interne Darstellung

PE: Register DE - Anfangsadresse der Zeichenkette

PA:
- Register DE = Ende der Zeichenkette
- (NUMNX) = Länge der erfassten Zeichenkette
- (NUMVX) = Umgewandelte Zahl
- CY = 1 = Fehler, Zeichenkette enthält Zeichen, die keine Hexziffern sind, Länge zu groß usw.

VR: AF, DE, HL

Name: . . . ERRM . . . . . UP-Nr. 19H

FKT.: Ausschrift des Textes „ERROR“ mit Zeilenvorschub CR,LF

PE/PA: -

VR: AF

Name: . . . HLHX . . . . . UP-Nr. 1AH

FKT.: Ausgabe des Wertes des Registers HL als Hexzahl und danach ein Leerzeichen

PE: Register HL

PA: -

VR: AF

Name: . . . HLDE . . . . . UP-Nr. 1BH

FKT.: Ausgabe der Register HL und DE als Hexzahlen und danach ein Leerzeichen

PE: Register HL, Register DE

PA: -

VR: AF

Name: . . . AHEX . . . . UP-Nr. 1CH

FKT.: Ausgabe Register A als Hexzahl
PE: Register A
PA: -
VR: AF

Name: . . . ZSUCH . . . . UP-Nr. 1DH

FKT.: Suche nach Zeichenkette (Menüwort)
PE: Register A - Prolog (für CAOS-Menü: 7FH)
Register BC - Länge des Suchbereiches
Register DE - Anfang der Vergleichszeichenkette
Register HL - Anfang des Suchbereichs
PA: Register DE = Ende+1 Vergleichskette (wenn gefunden)
Register HL = Ende+1 gefundene Kette
CY = 1 = Kette gefunden
VR: AF, BC, DE, HL
Bemerkung: Die Vergleichszeichenkette muss mit einem Steuerzeichen oder einem Leerzeichen abgeschlossen sein (Codes 00H…20H).

Name: . . . SOUT . . . . UP-Nr. 1EH

FKT.: Setze neuen Zeiger auf Ausgabetabelle: auf Adresse (HL) steht neue UP-Nr.
PE: Register HL - neuer Zeiger auf OUTAB
PA: Register HL = alter Zeiger
VR: HL

Name: . . . SIN . . . . UP-Nr. 1FH

FKT.: Setze neuen Zeiger auf Eingabetabelle: auf Adresse (HL) steht neue UP-Nr.
PE: Register HL - neuer Zeiger auf INTAB
PA: Register HL = alter Zeiger
VR: HL

Name: . . . NOUT . . . . UP-Nr. 20H

FKT.: Setze Zeiger für Ausgabe auf Standard (CRT)
PE: -
PA: Register HL = alter Zeiger
VR: HL

Name: . . . NIN . . . . . UP-Nr. 21H

FKT.: Setze Zeiger für Eingabe auf Standard (KBD)
PE: -
PA: Register HL = alter Zeiger
VR: HL

Name: . . . GARG . . . . . UP-Nr. 22H

FKT.: Erfassen von maximal 10 Hexzahlen aus Zeichenkette und Wandlung in die interne Darstellung
PE: Register DE - Adresse des ersten Zeichens
PA: Register DE = Adresse des letzten Zeichens + 1
(ARGN) = Anzahl der erfassten Zahlen
(ARG1)…(ARG10) = Werte der Zahlen
CY = 1 = Fehler
VR: AF, BC, DE, HL
Bemerkung: Zulässige Ziffern in Zeichenkette: 0…9, A…F; Leerzeichen; Ende 00H

Name: . . . OSTR . . . . . UP-Nr. 23H

FKT.: Ausgabe einer Zeichenkette, die nach UP-Aufruf steht, Abschluss mit 00H
PE/PA: -
VR: AF

Bsp.:
```
CALL   0F003H      ; PV1
DEFB   23H         ; UP-Nr.: OSTR
DEFM   'Fehler'    ; Ausgabe „Fehler“
DEFW   0D0AH       ; Neue Zeile
DEFW   707H        ; 2 * BEEP
DEFB   0           ; Ende
```

Bemerkung: Dieses Unterprogramm ist nur über PV1 nutzbar, für alle anderen PV ist das Unterprogramm ZKOUT zu verwenden!

Name: . . . OCHR . . . . . UP-Nr. 24H

FKT.: Zeichenausgabe an Gerät, das über Ausgabetabelle eingestellt werden kann (vgl. UP-Nr. 1EH, 20H)
PE: Register A - Zeichencode (ASCII)
PA: -
VR: AF

Name: . . . CUCP . . . . . UP-Nr. 25H

FKT.: Komplementiere Cursor
PE: (CURSO) - Cursorposition
PA/VR: -
Bemerkung: Befindet sich auf der Cursorposition ein ASCII-Zeichen, dann wird das Cursormuster (Speicherzelle B7EEH) verwendet - ansonsten nur die vorletzte Zeile des Zeichens (Strichcursor).

Name: . . . MODU . . . . . UP-Nr. 26H

FKT.: Modulsteuerung
- Lesen des Modultyps (Register A < 2)
- Aussenden des Steuercodes (Register A ≥ 2)

PE: Register A - Anzahl der Parameter
- 1 = Register L
- 2 = Register D und L

Register L - Modulsteckplatz
Register D - Modulsteuerbyte
PA: Register H = Modultyp (Strukturbyte)
Register D = Modulsteuerbyte neu *
Register A = Modulsteuerbyte alt *
VR: AF, H, BC, DE *
Bemerkung: Steuerbyte wird im Modulsteuerbytespeicher eingetragen.
* Bis CAOS 3.1 wird das Modulsteuerbyte in Register E zurückgegeben, ab CAOS 3.3 in Register D.
Ab CAOS 4.7 enthält Register A das vorherige Steuerbyte.

Name: . . . JUMP . . . . . UP-Nr. 27H

FKT.: Abschalten von CAOS-ROM, USER-ROM und Speichermodulen*, danach Sprung in ein neues Betriebssystem.
PE: Register A - Modulsteckplatz
PA/VR: Fehlermeldung, wenn sich auf dem Steckplatz kein Modul befindet.
Bemerkung: Die Startadresse des neuen Betriebssystems liegt auf 0F012H, in den Modulsteuerbytespeicher wird FFH eingetragen, um das Modul schreibgeschützt auf C000H einzuschalten.
* Ab CAOS 4.3 werden Speichermodule mit Kennbyte 7xh oder Fxh vor der Ausführung von JUMP ausgeschaltet.
Bei CAOS 2.3, 3.3 und 4.7 wird das Modul mit FDH eingeschaltet, sonst mit FFH.
Bis CAOS 3.1 wird der neue Schaltzustand im Modulsteuerbytespeicher teilweise nicht bzw. nicht korrekt eingetragen.
Bei CAOS 4.8 kann mit diesem UP auch in eine andere System-Bank gesprungen werden, falls die Flash-ROM-Erweiterung eingebaut ist.

Name: . . . LDMA . . . . . UP-Nr. 28H

FKT.:	LD (HL),A	
PE:	Register A	- Byte
	Register HL	- Adresse
PA/VR:	-	
Bemerkung:	Nur sinnvoll über PV4 - PV6, um in den IRM zu schreiben.	

Name: . . . LDAM . . . . . UP-Nr. 29H

FKT.:	LD A,(HL)	
PE:	Register HL	- Adresse
PA:	Register A	= Byte auf Adresse (HL)
VR:	-	
Bemerkung:	Nur sinnvoll über PV4 - PV6, um Inhalt aus dem IRM zu lesen	

Name: . . . BRKT . . . . . UP-Nr. 2AH

FKT.:	Test auf Unterbrechungsanforderung (Drücken der <BRK>-Taste)	
PE:	-	
PA:	CY = 1	= Taste <BRK> gedrückt
	Register A	Tastencode falls Taste gedrückt, bei BRK = Tastencode 03H
VR:	AF	

Name: . . . SPACE . . . . . UP-Nr. 2BH

FKT.:	Ausgabe eines Leerzeichens über UP-Nr. 24H
PE/PA:	-
VR:	AF

Name: . . . CRLF . . . . . UP-Nr. 2CH

FKT.:	Ausgabe von „NEWLINE“ (Codes 0DH=CR und 0AH=LF)
PE/PA:	-
VR:	AF

Name: . . . HOME . . . . . UP-Nr. 2DH

FKT.:	Ausgabe des Steuerzeichens „HOME“ (Code 10H)
PE/PA:	-
VR:	AF

Name: . . . MODI . . . . . UP-Nr. 2EH

FKT.: Aufruf des Systemkommandos MODIFY
PE: Register A - Anzahl der Argumente *
wenn A < 2, dann ein Zeichen pro Zeile
Register HL - Anfangsadresse
Register E - Zeichen pro Zeile *
PA: -
VR: AF, BC, DE, HL
Bemerkung: * Die Register A und E werden erst ab CAOS 4.3 ausgewertet, vorher galt immer ein Byte pro Zeile.

Name: . . . PUDE . . . . . UP-Nr. 2FH

FKT.: Löschen bzw. Testen eines Bildpunktes
PE: (HOR) - Horizontalkoordinate (0 … 13FH)
(VERT) - Vertikalkoordinate (0 … FFH)
PA: CY = 1 = Punkt außerhalb (Fehler)
Lores: Register A = Farbbyte
Z = 1 = Punkt hatte bereits Hintergrundfarbe
Hires: Register A = Farbe des Bildpunktes (0 … 3) *
Z = 0 (immer) = alle 4 Farben sind Vordergrundfarben
VR: AF
Bemerkung: (HOR) = (VERT) = 0 entspricht linker unterer Ecke
* ab CAOS 4.7 wird im Hires-Modus die Farbe des Bildpunktes ermittelt und in Register A zurückgegeben. Der Bildpunkt wird dabei nicht verändert.

Name: . . . PUSE . . . . . UP-Nr. 30H

FKT.: Setzen eines Bildpunktes
PE: (HOR) - Horizontalkoordinate (0 … 13FH)
(VERT) - Vertikalkoordinate (0 … FFH)
(FARB) - Bildpunktfarbe
Lores: Bit 0 = 1 XOR Funktion
Bit 1 = 1 Punkt löschen
Bit 2 = 1 Farbe ignorieren *
3 - 7 = Farbe (Vordergrund)
Hires: 3, 4 = Farbe
PA: CY = 1 = Punkt außerhalb (Fehler)
VR: AF
Bemerkung: * Farbe ignorieren nur bei CAOS 3.4 und ab CAOS 4.7

Name: . . . SIXD . . . . . .UP-Nr. 31H

FKT.: Verlagerung des IX-Arbeitsbereiches von CAOS

- Initialisierung der Interrupttabelle, *
- Initialisierung der Tastaturtabelle,
- Initialisierung des IX-Registers,
- Setzen IM2,
- Initialisierung der PIO, CTC,
- Initialisierung des Kassettenpuffers,
- Initialisierung des Menü-Prologbytes 7FH
- Neuaufbau Device-Treibertabelle *
- Neuaufbau der SUTAB im IRM *

PE: Register A - Höherwertiger Adressteil
A=FFH SUTAB und Device-Treiber initialisieren *

PA: (MIXIT) = Höherwertiger Adressteil

VR: AF, BC, DE, HL, IX

Bemerkung: Durch dieses UP werden alle internen Speicherblöcke in den Grundzustand zurückgesetzt. Ein V.24-Tastaturinterrupt wird dabei zurückgesetzt.
* Bei CAOS 4.6 und 4.7 werden SUTAB und Device-Treiber immer neu initialisiert.
* Ab CAOS 4.8 werden SUTAB und Device-Treiber nur mit Aufruf A=FFH neu initialisiert, der Arbeitsbereich wird dann auf 01H gesetzt.
Die Anwender-Interrupts werden bei der Verlagerung des Arbeitsbereiches mit an den neuen Adressbereich übernommen.

Name: . . . DABR . . . . . .UP-Nr. 32H

FKT.: Berechnung der VRAM-Adresse der Cursorposition im gerade eingestellten Fenster und Bild

PE: Register D - Zeile auf Bildschirm
Register E - Spalte auf Bildschirm

PA: Register HL = Adresse im Speicher
CY = 1 = Cursorposition außerhalb (Fehler)

VR: F, HL

Bemerkung: Dieses Programm ermöglicht das Zurücklesen von ASCII-Zeichen aus dem Bildschirmspeicher (VRAM).

Name: . . . TCIF . . . . . UP-Nr. 33H

FKT.:	Test, ob Cursorposition im definierten Fenster ist	
PE:	Register D	- Zeile der Cursorposition
	Register E	- Spalte der Cursorposition
PA:	CY = 1	= Cursor außerhalb (Fehler)
VR:	AF	
Bemerkung:	* Ab CAOS 4.8 wird bei Fenstergröße 0 Zeilen oder 0 Spalten immer CY=1 zurückgemeldet.	

Name: . . . PADR . . . . . UP-Nr. 34H

FKT.:	Berechnung von Pixel- und Farbadresse aus Zeichenposition	
PE:	Register H	- Vertikalposition (0 … FFH)
	Register L	- Horizontalposition (0 … 27H)
PA:	Register HL	= Zeichen- und Farbadresse *
	CY = 1	= Position außerhalb
VR:	F, HL, DE *	
Bemerkung:	HL = 00 entspricht linker oberer Ecke.	
	* Beim KC 85/2 und KC 85/3 wird im Register DE die Farbadresse übergeben, beim KC 85/4 und KC 85/5 ist DE unverändert. Diese Tatsache kann zur Unterscheidung des Rechnertyps genutzt werden.	

Name: . . . TON . . . . . UP-Nr. 35H

FKT.:	Tonausgabe	
PE:	(ARG1)	- Tonhöhe 1, rechts (Zeitkonstante CTC 0)
	(ARG1+1)	- Vorteiler 1 (0, 1) 0≙16 und 1≙256
	(ARG2)	- Tonhöhe 2, links (Zeitkonstante CTC 1)
	(ARG2+1)	- Vorteiler 2 (0, 1) 0≙16 und 1≙256
	(ARG3)	- Lautstärke (0 … 1FH) in Zweierschritten
	(ARG3+1)	- Tondauer (0 … FFH) in 20 ms-Schritten bzw. 0 = Dauerton
PA:	-	
VR:	AF, BC, DE, HL	
Bemerkung:	Tonhöhe=0 entspricht Ton ausgeschaltet, Tondauer wird über Interrupt CTC Kanal 2 realisiert	

Name: . . . SAVE . . . . . UP-Nr. 36H

FKT.:	Ausgabe von Maschinenprogrammen auf das eingestellte DEVICE	
PE:	Register HL	- Zeiger auf Dateiname/Pfad * (8 Zeichen für Namen, 3 Zeichen für Typ)
	(ARG1)	- Anfangsadresse des Programms
	(ARG2)	- Endadresse des Programms
	(ARG3)	- Startadresse des Programms
	(ARGN)	- Anzahl der Parameter (2 = ARG1, ARG2) (3 = ARG1 … ARG3 bei selbststartenden Programmen)
PA:	CY = 1	= BRK oder Fehler *
VR:	AF, BC, DE, HL	

Bemerkung: * Ab CAOS 4.8 wird mit CY=1 ein Fehler bei der Dateiausgabe oder Abbruch mit der <BRK>-Taste gemeldet.
Vorher war CY unbestimmt.
* Ab Treiber-Version 3.0 kann bei USB dem Dateinamen ein Pfad vorangestellt sein.

Name: . . . MBIN . . . . . UP-Nr. 37H

FKT.: Byteweise Eingabe vom eingestellten DEVICE mit Namensvergleich beim ersten Block

PE:
- Register D - Steuerbyte
  - Bit 3 = 1 INIT (Eingabe öffnen)
  - Bit 6 = 1 Close (Eingabe schließen)
- Register HL - Name nur bei INIT (Adresszeiger 11 Byte)
- Bit 5,(IX+7) - 1 = Blocknummernanzeige unterdrücken

PA:
- Register A = Datenbyte (nicht bei Close)
- CY = 1 = Fehler *
  - (Datei nicht gefunden oder Lesefehler)

VR: AF, DE, HL

Bemerkung: Dateiname: es werden immer 11 Zeichen ausgewertet, die ersten 3 Zeichen stellen den Dateityp, die folgenden 8 Zeichen den Dateinamen dar, ohne Punkt als Trennzeichen und ohne 0 als Endekennung.

Der Namensvergleich erfolgt bei TAPE anhand der ersten 11 Byte von Block Nr. 01. Die eigentlichen Daten werden dann ab dem 12. Byte dem Datenblock entnommen. Bei allen anderen Speichergeräten beginnt der erste Block sofort mit den Nutzdaten.

Nach dem blockweisen Einlesen werden die Daten byteweise dem Puffer entnommen. Während des Einlesens werden die Blocknummern angezeigt. Bei BASIC-Listings *.UUU werden die Blocknummern automatisch unterdrückt, für andere Dateitypen kann das mit Bit 5,(IX+7) ebenfalls erreicht werden. Bei einem nachfolgendem Close wird das Bit 5,(IX+7) automatisch wieder zurückgesetzt.

Fehlermeldungen werden am Bildschirm angezeigt. Kehrt die Funktion MBIN mit einem Fehler zurück, muss dennoch ein weiterer Aufruf mit gesetztem Bit 6 von Register D erfolgen, um die geöffnete Eingabe zu schließen.

Diese Funktion steht ab CAOS 3.1 zur Verfügung und wird von BASIC genutzt!

* Bis CAOS 4.7 wird im Fehlerfall zu der Adresse gesprungen, welche in der IRM-Speicherzelle IOERR (B7C9H) abgelegt ist. Dabei wird der IRM abgeschaltet und der Stackpointer auf den in IY gespeicherten Wert gesetzt. Voreinstellung für IOERR ist die Fehlerausgabe des BASIC-Interpreters.

Ab CAOS 4.8 kehrt die Routine im Fehlerfall mit gesetztem CY-Flag zurück, bei vorherigen CAOS-Versionen ist CY unbestimmt.

Name: . . . MBOUT . . . . . UP-Nr. 38H

FKT.: Byteweise Ausgabe auf das eingestellte DEVICE

PE:
- Register A - Daten
- Register D - Steuerbyte
  - Bit 3 = 1 INIT (Ausgabe öffnen)
  - Bit 6 = 1 Close (Ausgabe schließen)
- Register HL - Name nur bei INIT (Adresszeiger 11 Byte)
- Bit 5,(IX+7) - 1 = Blocknummernanzeige unterdrücken *

PA:
- CY = 1 = Fehler *
  (mit BRK abgebrochen oder Fehler beim Schreiben der Datei)

VR: AF, DE, HL

Bemerkung:

Dateiname: es werden immer 11 Zeichen ausgewertet, die ersten 3 Zeichen stellen den Dateityp, die folgenden 8 Zeichen den Dateinamen dar, ohne Punkt als Trennzeichen und ohne 0 als Endekennung.

Mit den auszugebenden Bytes wird der Kassettenpuffer gefüllt und dann blockweise ausgegeben, sobald sich 128 Byte im Puffer befinden oder das Close-Bit gesetzt ist. Bei TAPE wird der Dateiname in die ersten 11 Byte des Puffers eingetragen. Die eigentlichen Daten werden dann ab dem 12. Byte geschrieben. Bei allen anderen Speichergeräten beginnt der erste Block sofort mit den Nutzdaten.

Während der Blockausgabe wird die Blocknummer angezeigt.

* Ab CAOS 4.8 kann durch Setzen des Bit 5,(IX+7) die Blocknummernanzeige unterdrückt werden wie bei MBIN. Bei einem nachfolgendem Close wird das Bit 5,(IX+7) automatisch wieder zurückgesetzt.

Fehlermeldungen werden am Bildschirm angezeigt. Kehrt die Funktion MBOUT mit einem Fehler zurück, muss dennoch ein weiterer Aufruf mit gesetztem Bit 6 von Register D erfolgen, um die geöffnete Ausgabe zu schließen.

Diese Funktion steht ab CAOS 3.1 zur Verfügung und wird von BASIC genutzt!

* Bis CAOS 4.7 wird im Fehlerfall zu der Adresse gesprungen, welche in der IRM-Speicherzelle IOERR (B7C9H) abgelegt ist. Dabei wird der IRM abgeschaltet und der Stackpointer auf den in IY gespeicherten Wert gesetzt. Voreinstellung für IOERR ist die Fehlerausgabe des BASIC-Interpreters.

Ab CAOS 4.8 kehrt die Routine im Fehlerfall mit gesetztem CY-Flag zurück, bei vorherigen CAOS-Versionen ist CY unbestimmt.

Ab CAOS 4.8 wird der Kassettenpuffer mit 00H initialisiert, bevor Daten eingetragen werden.

Name: . . . KEY . . . . . UP-Nr. 39H

FKT.: Belegen einer F-Taste (Aufruf der Menükommandoroutine)

PE: Register A = Nr. - Nr. der Taste (1-0FH),
bei unzulässiger Nr. sofortige Rückkehr.
A = 0 - alle F-Tasten löschen

PA: -

VR: AF, BC, DE, HL

Bemerkung: Dieses Programm fordert Tastatureingaben.
Funktion steht für A=1 bis 0CH ab CAOS 3.1 zur Verfügung, ab CAOS 4.3 ist zusätzlich A=0 und A=0DH bis 0FH möglich!

Name: . . . KEYLI . . . . . UP-Nr. 3AH

FKT.: Anzeige der Belegung der F-Tasten (Aufruf der Menükommandoroutine KEYLIST = KEY ohne Parameter)

PE/PA: -

VR: AF, BC, HL

Bemerkung: Diese Funktion steht ab CAOS 3.1 zur Verfügung!

Name: . . . DISP . . . . . UP-Nr. 3BH

FKT.: HEX-/ASCII-Dump (Aufruf der DISPLAY-Menükommandoroutine)

PE: Register A - Anzahl der Argumente
A < 2 4 Zeilen
A < 3 8 Zeichen pro Zeile
Register HL - Anfangsadresse
Register E - Zeilenanzahl
Register C - Zeichen pro Zeile

PA: -

VR: AF, BC, DE, HL

Bemerkung: Taste BRK = Abbruch
Taste STOP = Übergang in MODIFY-Modus
Diese Funktion steht ab CAOS 3.1 zur Verfügung!

Name: . . . WININ . . . . . .UP-Nr. 3CH

FKT.: Initialisierung eines neuen Fensters
PE: Register A - Fensternummer (0-9)
Register HL - Fensteranfang
Register DE - Fenstergröße
PA: CY = 1 = Fehler (Nr., Anfang oder Größe)
VR: AF, BC, DE, HL
Bemerkung: Wenn die Parameter ohne Fehler sind, wird das aktuelle Fenster gespeichert und anschließend das neue Fenster initialisiert. Die Cursorposition des neuen Fensters wird auf die linke obere Ecke (0,0) gesetzt. Die Werte von STBT, COLOR und SCROLL/PAGE-Mode werden vom vorher aktiven Fenster übernommen. Im Fehlerfall bleibt das Fenster unverändert.
Diese Funktion steht ab CAOS 3.1 zur Verfügung, für CAOS 4.1 muss PE: C=L sein!

Name: . . . WINAK . . . . . UP-Nr. 3DH

FKT.: Aufruf eines Fensters über seine Nummer mit Abspeicherung des aktuellen Fenstervektors
PE: Register A - Fensternummer (0-9)
PA: CY = 1 * = falsche Nummer
VR: AF, BC, DE, HL
Bemerkung: Diese Funktion steht ab CAOS 3.1 zur Verfügung!
* Bis CAOS 3.3 wird der Fehler mit CY=0 statt CY=1 gemeldet!

Name: . . . LINE . . . . . UP-Nr. 3EH

FKT.: Zeichnen einer Linie mit dem eingestellten Linientyp auf dem Bildschirm von X0/Y0 nach X1/ Y1
PE: (ARG1) - X0 - X-Koordinate-Anfang
(ARG2) - Y0 - Y-Koordinate-Anfang
(ARG3) - X1 - X-Koordinate-Ende
(ARG4) - Y1 - Y-Koordinate-Ende
(FARB) Bit 0 = 1 XOR-Funktion
Bit 1 = 1 Linie löschen
Bit 2 = 1 Farbe ignorieren *
Bit 3 - 7 Farbe (Vordergrund)
PA: -
VR: AF, BC, DE, HL, AF', BC', DE', HL'
Bemerkung: Diese Funktion steht ab CAOS 3.1 zur Verfügung!
* Farbe ignorieren nur bei CAOS 3.4 und ab CAOS 4.7

Name: . . . CIRCLE . . . . . UP-Nr. 3FH

FKT.: Zeichnen eines Kreises mit dem eingestellten Linientyp auf dem Bildschirm mit Mittelpunkt XM/YM und Radius R

PE:
- (ARG1) - XM - X-Koordinate-Mittelpunkt
- (ARG2) - YM - Y-Koordinate-Mittelpunkt
- (ARG3) - R - Radius
- (FARB) Bit 0 = 1 XOR-Funktion
  Bit 1 = 1 Kreis löschen
  Bit 2 = 1 Farbe ignorieren *
  Bit 3 - 7 Farbe (Vordergrund)

PA: -

VR: AF, BC, DE, HL, BC', DE', HL'

Bemerkung: Diese Funktion steht ab CAOS 3.1 zur Verfügung!
* Farbe ignorieren nur bei CAOS 3.4 und ab CAOS 4.7

Name: . . . SQR . . . . . UP-Nr. 40H

FKT.: Berechnen der Quadratwurzel

PE: Register HL - 16 Bit-Zahl (vorzeichenlos)

PA: Register A = Ergebnis 8 Bit

VR: AF, HL, DE

Bemerkung: Diese Funktion steht ab CAOS 3.1 zur Verfügung!

Name: . . . MULT . . . . . UP-Nr. 41H

FKT.: Berechnung des Produktes zweier 8-Bit-Zahlen

PE: Register D, C - Faktoren (8 Bit)

PA: Register BA = Produkt (16 Bit)

VR: AF, HL, DE, B

Bemerkung: Diese Funktion steht ab CAOS 3.1 zur Verfügung!

Name: . . . CSTBT . . . . . UP-Nr. 42H

FKT.: Zeichenausgabe mit Negation des Bits 3 des Steuerbytes (STBT) des Bildschirmprogramms (Ausführung der Steuerzeichen / Abbildung der Steuerzeichen)

PE: Register A - Zeichencode (ASCII)

PA: -

VR: -

Bemerkung: Dieses Programm dient der Ausgabe der Steuerzeichensymbole auf dem Bildschirm.
Diese Funktion steht ab CAOS 3.3 zur Verfügung. Bei CAOS 3.1 wurde mit diesem Unterprogramm nur das Steuerbyte STBT umgeschaltet ohne Zeichenausgabe!

Name: . . . INIEA . . . . . UP-Nr. 43H

FKT.: Initialisierung eines E/A-Ports über Tabelle

PE: Register HL - Anfangsadresse der Tabelle

Tabellenaufbau

1. Byte = E/A-Adresse
2. Byte = Anzahl der Initialisierungsbytes (n)
3. Byte = } Initialisierungsbytes
. .
. .
. .
n. Byte = }

PA: Register HL = 1. Byte nach der Tabelle

VR: F, HL

Bemerkung: Die globale CPU-Interruptfreigabe wird nicht beeinflusst.
Diese Funktion steht ab CAOS 3.1 zur Verfügung!

Name: . . . INIME . . . . . UP-Nr. 44H

FKT.: Initialisierung mehrerer E/A-Ports über Tabelle(n)

PE: Register HL - Anfangsadresse der Tabelle
Register D - Anzahl der Ports

PA: Register HL = 1. Byte nach der Tabelle

VR: F, D, HL

Bemerkung: Die E/A-Tabelle besteht aus (D) Tabellen analog UP-Nr. 43H (INIEA). Die globale CPU-Interruptfreigabe wird am Beginn der Routine gesperrt und nach der Initialisierung wieder freigegeben.
Diese Funktion steht ab CAOS 3.1 zur Verfügung!

Name: . . . ZKOUT . . . . . UP-Nr. 45H

FKT.: Ausgabe einer über Register HL adressierten Zeichenkette
PE: Register HL - Anfang der Zeichenkette
PA: Register HL = Ende der Zeichenkette+1
(Adresse nach 0-Byte)
VR: AF, HL
Bemerkung: Die auszugebende Zeichenkette besteht aus ASCII-Zeichen und wird mit 00H abgeschlossen (vgl. UP-Nr. 23 OSTR). Das Programm wird vorrangig bei Programmverteilern PV5 und PV6 eingesetzt.
Diese Funktion steht ab CAOS 3.1 zur Verfügung!

Beispiel:

```
        LD      HL,TXT
        LD      E,45H
        CALL    PV5
        .
        .
TXT:    DEFB    0CH                     ; CLS
        DEFB    0AH                     ; CUD
        DEFM    '===Testprogramm==='
        DEFW    0A0DH                   ; neue Zeile
        DEFB    0
```

Name: . . . MENU . . . . . UP-Nr. 46H

FKT.: Ausschreiben des aktuellen Menüs und Übergang in die Kommandoeingabe

PE:

HL	-	Beginn Suchbereich	(C000H) *
BC	-	Länge Suchbereich	(0000H) *
(IX+9)	-	Prologbyte	
(ARGN) *	-	nur bei (ARGN)=1 wird ARG1 ausgewertet	
(ARG1) = 0	-	MENU normal anzeigen	
= 1	-	versteckte Menüworte mit anzeigen	
= 2	-	Adressen mit anzeigen	
= 3	-	versteckte Menüworte und Adressen	

PA/VR: -

Bemerkung: Das Programm dient zur Anzeige und Eingabe des aktuellen Menüs unter Berücksichtigung des eingestellten Prologbytes. Es erfolgt kein Löschen des Bildschirms und keine Generierung der Titelzeile des Systems. Prologbyte des Systems ist 7FH, weitere benutzte Prologbytes siehe Seite 131. Das Suchen des Prologbytes beginnt seit CAOS 3.4 immer ab Adresse C000H. Es wird der gesamte Adressbereich bis BFFFH durchsucht.

* Diese Funktion steht ab CAOS 3.1 zur Verfügung, dort noch mit variablem Suchbereich entsprechend Register HL und BC. Ab CAOS 3.3 wird immer von C000H bis BFFFH gesucht.
Ab CAOS 4.5 wird ARGN/ARG1 als Parameter ausgewertet.

Name: . . . LSTOUT . . . . . UP-Nr. 47H

FKT.: Initialisieren Druckerausgabe

PE:

(ARG1)	-	Modulschacht des zu nutzenden Moduls M001, M003, M053 (0 für M021)
(ARG2)	-	Kanal des V.24-Moduls (1/2)
(ARG3)	-	USER-Ausgabekanal (2/3)
(ARG4) = 0	-	keine Reaktion auf SHIFT CLEAR
= 1	-	Protokoll ein/aus bei SHIFT CLEAR
= 2	-	bei SHIFT CLEAR HARDCOPY/ SCREENCOPY
(ARG5)	-	Druckertyp (siehe Tabelle 11 auf Seite 82)
(ARGN)	-	Anzahl der Argumente (0 bis 5)
(INTV1)	-	Anfangsadresse der V.24-Initialisierungstabelle (SIO, CTC)
(INTV1L)	-	Länge der V.24-Initialisierungstabelle

PA: -

VR: AF, BC, DE, HL

Bemerkung: Die V.24-Initialisierungstabelle gilt nur für M003/M053, die ersten 2 Byte in der Tabelle gelten immer der CTC-Initialisierung. CR+LF wird zum Drucker gesendet.
Diese Funktion steht ab CAOS 4.1 zur Verfügung, ab CAOS 4.5 kann damit auch ein M001 oder M021 initialisiert werden.

Name: . . . V24DUP . . . . . UP-Nr. 48H

FKT: Initialisierung V.24-Duplexroutine

PE:

(ARG1)	-	Modulschacht V.24-Moduls (8, C …)
(ARG2)	-	Kanal des V.24-Moduls (1/2)
(ARG3)	-	USER-Aus/Eingabekanal (2/3)
(ARGN)	-	Anzahl der Argumente (0 oder 3)
(INTV2)	-	Anfangsadresse der Initialisierungstabelle
(INTV2L)	-	Länge der Initialisierungstabelle

PA: -

VR: AF, BC, DE, HL

Bemerkung: Die ersten 2 Byte in der Tabelle gelten immer der CTC-Initialisierung.
Diese Funktion steht ab CAOS 4.1 zur Verfügung!
CAOS 4.5 bis 4.7 niehmen bei fehlenden Werten immer den V.24-Kanal 1 und USER-Kanal 2 an, ansonsten die letzten verwendeten Werte.

Name: . . . SETDEV . . . . . UP-Nr. 49H

FKT:	Gerätetreiber auswählen, abfragen oder anzeigen		
PE:	Register A	0…7	- Auswahl Gerätetreiber Nr. 0-7
		0FDH	- aktuellen Treiber abfragen *
		0FEH	- aktuellen Treibername anzeigen
		0FFH	- Auflisten aller Gerätetreiber
PA:	CY=1	= ausgewählter Treiber nicht aktiv	
	bei CY=0	A	- aktuelle Treiber-Nr. 0-7 *
		HL	- Zeiger auf Treibername *
		Z=1	- Kassettentreiber *
VR:	AF, BC, DE, HL		

Bemerkung: Die Nr. des Gerätetreibers wird in Bit 2-4 (IX+8) gespeichert
Diese Funktion steht ab CAOS 4.6 zur Verfügung
* Abfrage des aktuellen Treibers bei CAOS 4.6 nicht möglich, bei CAOS 4.7 mit A=8
* Rückmeldung HL und Z ab CAOS 4.7
* Rückmeldung Treiber-Nr. 0-7 in Register A ab CAOS 4.8

Name: . . . HLDEZ . . . . . UP-Nr. 4AH

FKT:	Ausgabe des Wertes des Registers HL als Dezimalzahl	
PE:	Register HL	Wert
PA:	-	
VR:	AF, BC, HL	

Bemerkung: Die Anzeige erfolgt ein- bis fünfstellig ohne führende Nullen
Diese Funktion steht ab CAOS 4.8 zur Verfügung

Name: . . . RDEZ . . . . . UP-Nr. 4BH

FKT.:	Umwandlung einer Zeichenkette (Dezimalzahl) in interne Darstellung	
PE:	Register DE	- Anfangsadresse der Zeichenkette
PA:	Register DE	= Ende der Zeichenkette
	(NUMNX)	= Länge der erfassten Zeichenkette
	(NUMVX)	= Umgewandelte Zahl
	CY = 1	= Fehler, Zeichenkette enthält Zeichen, die keine Dezimalziffer sind oder Zahlenwert zu groß
VR:	AF, C, DE, HL	

Bemerkung: Diese Funktion steht ab CAOS 4.8 zur Verfügung

Name: . . . GARGC . . . . . UP-Nr. 4CH

FKT.:	Erfassen von maximal 10 Argumenten mit wählbarer Zahlenbasis aus Zeichenkette und Wandlung in die interne Darstellung	
PE:	Register C	- Zahlenbasis (C= 2 bis 16) (C=10 für Dezimalzahlen C=16 für Hexadezimalzahlen)
	Register DE	- Adresse des ersten Zeichens
PA:	Register DE	= Adresse des letzten Zeichens + 1
	(ARGN)	= Anzahl der erfassten Zahlen
	(ARG1)…(ARG10)	= Werte der Zahlen
	CY = 1	= Fehler
VR:	AF, BC, DE, HL	
Bemerkung:	vergleiche UP-Nr. 22H. Zulässige Ziffern je nach Zahlenbasis 0 … 9, A … F; Leerzeichen; Ende 00H	
	Diese Funktion steht ab CAOS 4.8 zur Verfügung	

3.5.7. Liste der nutzbaren Unterprogramme für PV7

Über Programmverteiler PV7 (auf Adresse 0F021H) werden ab CAOS 4.7 die DEVICE-Funktionen aufgerufen. Die Parameterübergabe erfolgt wie beim PV1 mit dem Byte, welches dem CALL-Befehl folgt. Es gilt jedoch eine andere Numerierung der Unterprogramme!

Beispiel: CALL 0F021H
DEFB UP-Nr. (Treiber-Programmnummer)

Die über PV7 aufgerufenen Routinen werden an den aktiven Device-Treiber weitergeleitet. Das aktuelle DEVICE wird in Bit 2–4 von (IX+8) gespeichert, siehe Seite 185. Gleichnamige DEVICE-Routinen können auch über PV1 bis PV6 aufgerufen werden, sind darüber jedoch etwas langsamer, da eine zusätzliche Adressberechnung erforderlich ist. Für neue Software sollte bevorzugt der PV7 benutzt werden, für kompatible Software die PV1-6. Der IRM muss bei Aufruf von PV7 eingeschaltet sein!
Um eine Dateiauswahl und Fehlerauswertung bei „Nicht-TAPE-Geräten“ realisieren zu können, wurden einige Unterprogramme bezüglich der Parameter ergänzt.

Name: . . . MBO . . . . . PV7, UP-Nr. 0

FKT.:	Ausgabe Datenblock von Puffer auf Datenträger (128 Byte)	
PE:	Register BC	- Länge Vorton
	(IX+2)	- Blocknummer -1
	(IX+5)	- L (Pufferadresse)
	(IX+6)	- H (Pufferadresse)
PA:	Register HL	= Pufferende + 1
	Register DE	= Pufferende + 1 bei CY=0 Fehlercode, bei CY=1 *
	(IX+2)	= Block-Nr.
	CY=1	= Fehler *
VR:	AF, BC, DE, HL	
Bemerkung:	* Bis CAOS 4.5 ist CY unbestimmt. Fehlermeldungen werden auf dem Bildschirm angezeigt. Ab USB-Treiber-Version 3.0 wird im Fehlerfall in Register DE ein Fehlercode zurückgegeben. Diese Funktion ist auch über PV1-6, UP-Nr. 01H aufrufbar.	

Name: . . . MBI . . . . . PV7, UP-Nr. 1

FKT.:	Einlesen Datenblock von Datenträger in den Puffer (128 Byte)	
PE:	(IX+5)	- L (Pufferanfang)
	(IX+6)	- H (Pufferanfang)
PA:	(IX+2)	= Block-Nr.
	CY = 1	= Block fehlerhaft
	Register DE	= Fehlercode, bei CY=1 *
VR:	AF, BC, (DE im Fehlerfall)	
Bemerkung:	* Fehlermeldungen werden auf dem Bildschirm angezeigt.	
	Ab USB-Treiber-Version 3.0 wird im Fehlerfall in Register DE ein Fehlercode zurückgegeben.	
	Diese Funktion ist auch über PV1-6, UP-Nr. 05H aufrufbar.	

Name: . . . ISRO . . . . . PV7, UP-Nr. 2

FKT.:	Initialisierung der Dateiausgabe, Ausgabe des ersten Blockes (Block-Nr. 01H) mit langem Vorton *	
PE:	Register HL	- Zeiger auf Dateinamen/Pfad *
	(IX+5)	- L (Pufferadresse)
	(IX+6)	- H (Pufferadresse)
PA:	Register HL	= Pufferende + 1
	Register DE	= Pufferende + 1 bei CY=0 Fehlercode, bei CY=1 *
	(IX+2)	= Block-Nr.
	CY=1	= Fehler *
VR:	AF, BC, DE, HL	
Bemerkung:	* Vorton bei CAOS 2.2 und 3.1: 8.192 Schwingungen bei CAOS 3.3, 4.1-4.8: 4.096 Schwingungen CAOS 3.4, OS pi/88 und OS pi/90: 2.560 Schwingungen	
	* Bis CAOS 4.5 ist CY unbestimmt.	
	In Register HL muss ab CAOS 4.7 der Dateiname übergeben werden, falls DEVICE nicht TAPE ist. Bei CAOS 4.6 wurde der Dateiname aus dem Vorblock entnommen, das war aber nicht immer sichergestellt. Ab USB-Treiber-Version 3.0 kann dem Dateinamen ein Pfad vorangestellt sein. Im Fehlerfall wird in Register DE ein Fehlercode zurückgegeben. Fehlermeldungen werden auf dem Bildschirm angezeigt.	
	Diese Funktion ist auch über PV1-6, UP-Nr. 08H aufrufbar.	

Name: . . . CSRO . . . . . PV7, UP-Nr. 3

FKT.: Abschluss-(Close-)Routine für Dateiausgabe, Ausgabe des letzten Blockes (Block-Nr.: FFH)

PE:
- Register BC - Länge Vorton
- (IX+5) - L (Pufferadresse)
- (IX+6) - H (Pufferadresse)

PA:
- Register HL = Pufferende + 1
- Register DE = Pufferende + 1 bei CY=0, Fehlercode, bei CY=1 *
- (IX+2) = Block-Nr.
- CY=1 = Fehler *

VR: AF, BC, DE, HL

Bemerkung:
* Bis CAOS 4.5 ist CY unbestimmt.
Bis CAOS 4.5 wurde innerhalb von CSRO noch ein Zeilenvorschub CR+LF auf dem Bildschirm ausgegeben.
Fehlermeldungen werden auf dem Bildschirm angezeigt.
Ab USB-Treiber-Version 3.0 wird im Fehlerfall in Register DE ein Fehlercode zurückgegeben.
Diese Funktion ist auch über PV1-6, UP-Nr. 09H aufrufbar.

Name: . . . ISRI . . . . . PV7, UP-Nr. 4

FKT.: Initialisierung Magnetbandeingabe / Datei öffnen und Einlesen des 1. Blockes

PE:
- Register HL - Zeiger auf Dateiname/Pfad *
- (IX+5) - L (Pufferadresse)
- (IX+6) - H (Pufferadresse)

PA:
- (IX+2) = Block-Nr.
- CY = 1 = Block fehlerhaft
- Register DE = Fehlercode, bei CY=1 *

VR: AF, BC, (DE im Fehlerfall) vgl. Unterprogramm MBI

Bemerkung:
* Register HL muss ab CAOS 4.6 übergeben werden, falls DEVICE nicht TAPE ist. Ab USB-Treiber-Version 3.0 kann dem Dateinamen ein Pfad vorangestellt sein. Im Fehlerfall wird in Register DE ein Fehlercode zurückgegeben. Fehlermeldungen werden auf dem Bildschirm angezeigt.
Diese Funktion ist auch über PV1-6, UP-Nr. 0AH aufrufbar.

Name: . . . CSRI . . . . . PV7, UP-Nr. 5

FKT.:	Abschluss der Magnetbandeingabe / Datei schließen	
PE:	-	
PA:	CY = 1	= Fehler *
	Register DE	= Fehlercode, bei CY=1 *
VR:	AF, HL, (DE im Fehlerfall)	
Bemerkung:	* Fehlerauswertung nur bei DEVICE > 0 und ab CAOS 4.7, sonst CY unbestimmt bei Rückkehr. Fehlermeldungen werden auf dem Bildschirm angezeigt.	
	Ab USB-Treiber-Version 3.0 wird im Fehlerfall in Register DE ein Fehlercode zurückgegeben.	
	Diese Funktion ist auch über PV1-6, UP-Nr. 0BH aufrufbar.	

Name: . . . DRVER . . . . . PV7, UP-Nr. 6

FKT:	Abfrage der DEVICE-Treiber-Version	
PE:	-	
PA:	Register A	Versionsnummer BCD
	Register HL	Zeiger auf Textstring zu Treiber-Version
PA/VR:	AF, HL	
Bemerkung:	Diese Funktion steht ab CAOS 4.8 zur Verfügung, ursprünglich war sie bei CAOS 4.6 für MBIN vorgesehen. In CAOS 4.7 ist diese Funktion unbenutzt.	
	Voraussetzung ist ein DEVICE-Treiber ab Version 3.0.	

Name: . . . DRV:USR . . . . . PV7, UP-Nr. 7

FKT:	Treiberspezifische Funktionen je nach DEVICE	
PE:	Register E	Nummer der Unterfunktion
PA:	CY=0	OK
	CY=1	Funktion nicht vorhanden oder Fehler
	Register DE	Fehlercode falls CY=1
PA/VR:	andere Register je nach Unterfunktion	
Bemerkung:	Diese Funktionen stehen ab CAOS 4.8 zur Verfügung, ursprünglich war das UP-Nr. 7 bei CAOS 4.6 für MBOUT vorgesehen. In CAOS 4.7 ist diese Funktion unbenutzt. Vor Verwendung in CAOS 4.8 sollte mit PV7, UP-Nr. 6 überprüft werden, dass eine Treiber-Version 3.0 oder höher vorliegt.	

3. SOFTWARE

Der USB-Treiber 3.0 im M052-ROM stellt die nachfolgend beschriebenen Unterfunktionen innerhalb des PV7 UP-Nr. 7 zur Verfügung. Dabei gilt allgemein:

Dateiname/Pfad ist ein Zeiger auf einen Null-terminierten String auf einen Dateinamen mit optional vorangestelltem Pfad. Elemente des Dateipfades sind durch Slash „/“ oder Backslash „\“ voneinander zu trennen. Beginnt der String mit einem Pfad-Trennzeichen, dann wird dies als absoluter Pfad vom Hauptverzeichnis interpretiert. Ansonsten wird der Pfad ab dem aktuell eingestellten Verzeichnis angenommen. Hier einige Beispiele:

NAME.TYP	ohne Pfad
/NAME.TYP	Datei im Hauptverzeichnis
/PFAD1/PFAD2/NAME.TYP	Datei im angegebenen Pfad
\PATH\NAME.TYP	Datei im angegebenen Pfad
PFAD3/NAME.TYP	Datei im relativen Pfad

Fehlercode Ein Fehler wird durch CY=1 bei Rückkehr von allen Funktionen angezeigt. Dann enthält das Registerpaar DE einen Fehlercode für weitere Details. Dabei steht das erste Zeichen des Fehlercodes in Register D und das zweite Zeichen in Register E. Keine der Funktionen zeigen Fehlertexte an. Eine Auswertung muss im Anwenderprogramm erfolgen. Die Fehlercodes entsprechen den VNC-Codes zuzüglich einiger Erweiterungen:

BC	Bad Command
CF	Command Failed
DE	Dir End (Verzeichnis zu Ende)
DF	Disk Full
FI	File Invalid
FN	Filename Invalid
FO	File Open
MD	Mode (falscher Modus, Treibermeldung)
MO	Missing Operand (kein Dateiname angegeben)
ND	No Disk
NE	Dir Not Empty
NF	No Function (Unterfunktion nicht vorhanden)
NU	No Upgrade
PI	Path Invalid
RO	Read Only
TO	Time Out (Treibermeldung)

Name: . . . USB:USER-Mode . . . . . PV7, UP-Nr. 7.0

FKT:	direkten Zugriff auf VNC durch Anwenderprogramm erlauben	
PE:	Register E	0
PA:	CY=0	OK
VR:	HL, DE, AF	

Bemerkung: Der USB-PIO des M052 wird von dieser Funktion zugeschaltet. Der M052-Interrupt holt nur die Daten von der USB-Tastatur ab. Die Daten von der FIFO-Schnittstelle sind vom Anwenderprogramm abzuholen. Die Nutzung der Funktionen RAWIN und RWAOUT ist gestattet.
Die anderen CAOS-Funktionen des PV7 bzw. die äquivalenten CAOS-Systemprogramme dürfen nicht genutzt werden, solange der USB-USER-Mode aktiv ist!

Name: . . . USB:CAOS-Mode . . . . . PV7, UP-Nr. 7.1

FKT:	direkten Zugriff auf VNC durch Anwenderprogramm sperren	
PE:	Register E	1
PA:	CY=0	OK
VR:	HL, DE, AF	

Bemerkung: Der USB-PIO des M052 wird von dieser Funktion abgeschaltet. Der M052-Interrupt verarbeitet alle Daten vom VNC (USB-Tastatur und FIFO-Schnittstelle).
Die Nutzung der Funktionen RAWIN und RWAOUT ist verboten. Die CAOS-Funktionen des PV7 bzw. die äquivalenten CAOS-Systemprogramme dürfen genutzt werden, solange der USB-CAOS-Mode aktiv ist! Der CAOS-Mode ist der Einschaltzustand des KC-Systems mit M052.

Name: . . . USB:RAWOUT . . . . . PV7, UP-Nr. 7.2

FKT:	Schreiben eines Bytes in den VNC inkl. Time-Out-Prüfung	
PE:	Register E	2
	Register A	zu schreibendes Byte
PA:	CY=0	OK, Byte wurde geschrieben
	CY=1	Fehler, dann
	Register DE	Fehlercode „TO“ oder „MD“
VR:	AF, (DE bei Fehler)	

Bemerkung: Diese Funktion darf von einem Anwendungsprogramm nur im USB-USER-Mode aufgerufen werden!

Name: . . . USB:RAWIN . . . . . PV7, UP-Nr. 7.3

FKT:	Lesen eines Bytes vom VNC inkl. Time-Out-Prüfung	
PE:	Register E	3
PA:	CY=0	OK, Byte wurde gelesen, dann
	Register A	gelesenes Datenbyte
	CY=1	Fehler, dann
	Register DE	Fehlercode „TO“ oder „MD“
VR:	AF, (DE bei Fehler)	
Bemerkung:	Diese Funktion darf von einem Anwendungsprogramm nur im USB-USER-Mode aufgerufen werden!	

Name: . . . USB:TEST . . . . . PV7, UP-Nr. 7.4

FKT:	Test, ob eine Datei auf dem USB-Stick vorhanden ist	
PE:	Register E	4
	Register HL	Zeiger auf Pfad/Dateiname
PA:	Register HL	Zeiger auf Dateiname (ohne Pfad)
	CY=0	Datei vorhanden, dann
	Register DEBC	Größe der Datei in Byte (32 Bit-Zahl)
		(10000H ≙ 64KByte, FFFFFFFFH ≙ 4GByte)
	CY=1	Fehler, dann
	Register DE	Fehlercode
VR:	AF, BC, DE, HL	
Bemerkung:	Falls im Dateinamen ein Pfad enthalten ist, wird dieser Pfad eingestellt. Falls diese Datei im Anschluss geöffnet werden soll, kann der in HL zurückgegebene Dateiname mit abgetrenntem Pfad benutzt werden.	

Name: . . . USB:OPEN . . . . . PV7, UP-Nr. 7.5

FKT:	Datei zum Lesen öffnen	
PE:	Register E	5
	Register HL	Zeiger auf Pfad/Dateiname
PA:	CY=0	OK, dann
	Register DEBC	Größe der Datei in Byte (32 Bit-Zahl)
	CY=1	Fehler, dann
	Register DE	Fehlercode
VR:	AF, BC, DE, HL	
Bemerkung:	Falls im Dateinamen ein Pfad enthalten ist, wird dieser Pfad eingestellt, auch wenn die Datei nicht vorhanden ist.	

Name: . . . USB:READ1 . . . . . PV7, UP-Nr. 7.6

FKT:	Ein Datenbyte aus Datei lesen	
PE:	Register E	6
PA:	CY=0	OK, dann
	Register A	gelesenes Datenbyte
	CY=1	Fehler, dann
	Register DE	Fehlercode
VR:	AF, BC, (DE bei Fehler)	
Bemerkung:	Diese Funktion liest das nächste Datenbyte aus der aktuell geöffneten Datei ein. READ1 kann im Wechsel mit READN benutzt werden.	

Name: . . . USB:READN . . . . . PV7, UP-Nr. 7.7

FKT:	Datenbytes aus Datei in den Speicher einlesen	
PE:	Register E	7
	Register HL	Adresse, wo Daten abzulegen sind
	Register BC	Anzahl der zu lesenden Datenbytes
PA:	CY=0	OK, dann
	Register HL	Adresse des letzten Datenbytes
	CY=1	Fehler, dann
	Register DE	Fehlercode
VR:	AF, BC, DE, HL	
Bemerkung:	Diese Funktion liest eine variable Anzahl Datenbytes aus der aktuell geöffneten Datei in den Speicher ein. READN kann im Wechsel mit READ1 benutzt werden. Hat die Datei weniger Bytes als zum Lesen angefordert werden, dann wird ein Fehlercode erzeugt und keine Daten gelesen. Das Anwenderprogramm sollte die Anzahl der gelesenen Datenbytes selbst überwachen anhand der Dateigröße, welche von UP 7.4 und 7.5 zurückgemeldet wird.	

Name: . . . USB:CREATE . . . . . PV7, UP-Nr. 7.8

FKT:	neue Datei zum Schreiben öffnen	
PE:	Register E	8
	Register HL	Zeiger auf Pfad/Dateiname
PA:	CY=0	OK
	CY=1	Fehler, dann
	Register DE	Fehlercode
VR:	AF, BC, DE, HL	
Bemerkung:	Falls im Dateinamen ein Pfad enthalten ist, wird dieser Pfad vor dem Öffnen der Datei eingestellt. Ist die Datei bereits vorhanden, dann wird der Dateizeiger auf den Anfang gestellt und die Datei damit überschrieben.	

Name: . . . USB:APPEND . . . . . PV7, UP-Nr. 7.9

FKT:	vorhandene Datei zum Schreiben öffnen	
PE:	Register E	9
	Register HL	Zeiger auf Pfad/Dateiname
PA:	CY=0	OK
	CY=1	Fehler, dann
	Register DE	Fehlercode
VR:	AF, BC, DE, HL	
Bemerkung:	Falls im Dateinamen ein Pfad enthalten ist, wird dieser Pfad vor dem Öffnen der Datei eingestellt. Nach dem Öffnen der Datei steht der Dateizeiger am Dateiende, weitere Daten können am Dateiende angefügt werden. Ist die angegebene Datei nicht vorhanden, dann wird eine neue Datei angelegt.	

Name: . . . USB:WRITE1 . . . . . PV7, UP-Nr. 7.10

FKT:	Ein Datenbyte in Datei schreiben	
PE:	Register E	10
	Register A	zu schreibendes Datenbyte
PA:	CY=0	OK
	CY=1	Fehler, dann
	Register DE	Fehlercode
VR:	AF, BC, (DE bei Fehler)	
Bemerkung:	Diese Funktion schreibt ein Datenbyte in die aktuell geöffnete Datei. WRITE1 kann im Wechsel mit WRITEN benutzt werden.	

Name: . . . USB:WRITEN . . . . . PV7, UP-Nr. 7.11

FKT:	Datenbytes aus dem Speicher in Datei schreiben	
PE:	Register E	11
	Register HL	Adresse der zu schreibenden Datenbytes
	Register BC	Anzahl der zu schreibenden Datenbytes
PA:	CY=0	OK, dann
	Register HL	Adresse des letzten Datenbytes
	CY=1	Fehler, dann
	Register DE	Fehlercode
VR:	AF, BC, DE, HL	
Bemerkung:	Diese Funktion schreibt eine variable Anzahl Datenbytes aus dem Speicher in die aktuell geöffnete Datei. WRITEN kann im Wechsel mit WRITE1 benutzt werden.	

Name: . . . USB:CLOSE . . . . . PV7, UP-Nr. 7.12

FKT:	Datei schließen	
PE:	Register E	12
PA:	CY=0	OK
	CY=1	Fehler, dann
	Register DE	Fehlercode
VR:	AF, BC, DE, HL	
Bemerkung:	Diese Funktion schließt eine zum Lesen oder Schreiben geöffnete Datei. Der Dateiname der zuletzt geöffneten Datei wird ab der Adresse 0000-000Ah gespeichert und von dieser Funktion dort ausgelesen.	

Name: . . . USB:FIRST . . . . . PV7, UP-Nr. 7.13

FKT:	Ersten Verzeichniseintrag suchen	
PE:	Register E	13
	Register HL	Zeiger auf Maske (optional mit Pfad)
	(IX+5)	L (Ablagepuffer)
	(IX+6)	H (Ablagepuffer)
PA:	CY=0	Eintrag gefunden, dann
	(NUMVX)	Anzahl = 1
	CY=1	Eintrag nicht gefunden oder Fehler, dann
	Register DE	Fehlercode
VR:	AF, BC, DE, HL	
Bemerkung:	Die Maske wird ab Adresse B773H (oberes Ende des Kassettenpuffers) abgelegt. Der Dateiname wird ab der Adresse abgelegt, welche in (IX+5) und (IX+6) steht, standardmäßig ab B700H. Bei CY=0 muss anschließend UP-Nr. 7.14 solange aufgerufen werden, bis das Verzeichnis komplett eingelesen ist.	

Name: . . . USB:NEXT . . . . . PV7, UP-Nr. 7.14

FKT:	Nächsten Verzeichniseintrag suchen	
PE:	Register E	14
	(IX+5)	L (Ablagepuffer)
	(IX+6)	H (Ablagepuffer)
PA:	CY=0	Eintrag gefunden, dann
	(NUMVX)	Anzahl der gefundenen Einträge
	CY=1	Eintrag nicht gefunden oder Fehler, dann
	Register DE	Fehlercode
VR:	AF, BC, DE, HL	
Bemerkung:	Die mit USB:FIRST abgelegte Maske wird weiter benutzt. Der Dateiname wird ab der Adresse abgelegt, welche in (IX+5) und (IX+6) steht, standardmäßig ab B700H. Das UP 7.14 muss so lange aufgerufen werden, bis mit CY=1 und DE='DE' das Ende des Verzeichnisses angezeigt wird.	

Name: . . . USB:STICK . . . . . PV7, UP-Nr. 7.15

FKT:	Test, ob USB-Stick vorhanden ist	
PE:	Register E	15
PA:	CY=0	USB-Stick ist vorhanden
	CY=1	kein USB-Stick vorhanden, dann
	DE	Fehlercode
VR:	AF, BC, DE, HL	
Bemerkung:	Diese Funktion prüft nur, ob ein Stick am USB-Anschluss angesteckt ist.	

Name: . . . DIRANZ . . . . . PV7, UP-Nr. 8

FKT:	Verzeichnisinhalt anzeigen	
PE:	(IX+8)	- Bit 2-4: Device-Nummer
	Register DE	- Zeiger auf Dateimaske, Ende mit 00H
PA:	Register HL	- Anzahl Dateien
VR:	AF, BC, DE, HL	
Bemerkung:	Der Verzeichnisinhalt wird am Bildschirm angezeigt, bei voller Bildschirmseite wird eine Tastatureingabe abgewartet. Am Ende wird dabei noch eine Zusammenfassung (USB: Anzahl Dateien, DISK: freier Speicherplatz) angezeigt. Bei Device=TAPE wird das Verzeichnis einer Kassette angezeigt, wenn diese komplett abgespielt wird. Bei TAPE wird die Dateimaske nicht ausgewertet, es wird auch nicht auf eine Tastatureingabe bei voller Bildschirmseite gewartet. Mit BRK kann die Anzeige abgebrochen werden. Diese Funktion steht ab CAOS 4.7 zur Verfügung!	

Name: . . . CD . . . . . PV7, UP-Nr. 9

FKT:	Verzeichnis bzw. Pfad wechseln	
PE:	(IX+8)	- Bit 2-4: Device-Nummer
	Register DE	- Zeiger auf Verzeichnisname, Ende mit 00H
PA:	CY=1	Verzeichnis/Pfad nicht vorhanden
VR:	AF, BC, DE, HL	

Bemerkung: Bei TAPE wird mit dieser Funktion der Motor des Kassettenrecorders aus/ein geschaltet. Zeigt DE auf eine leere Zeichenkette, wird das Verzeichnis abgefragt. Fehlermeldungen werden auf dem Bildschirm angezeigt.
Bei DISK wird ein Laufwerkswechsel im D004/D008 ausgeführt, anzugeben ist dabei der Laufwerksbuchstabe und/oder der USER-Bereich in dezimaler Schreibweise also 0 bis 31.
Bei USB ist der Name des Verzeichnisses anzugeben in welches gewechselt werden soll. Ab USB-Treiber-Version 3.0 kann ein kompletter Pfad angegeben werden, siehe Erläuterungen auf Seite 169. Im Fehlerfall wird in Register DE ein Fehlercode zurückgegeben.
Diese Funktion steht ab CAOS 4.7 zur Verfügung! Ab CAOS 4.8 erfolgt die Anzeige und Eingabe des USER-Bereichs als Dezimalzahl, in älteren CAOS-Versionen als einstellige Hex-Zahl 0-F. Ab CAOS 4.8 kann auch nur der USER-Bereich ohne Laufwerksangabe gewechselt werden.

Name: . . . ERA . . . . . PV7, UP-Nr. 10

FKT:	Datei löschen	
PE:	(IX+8)	- Bit 2-4: Device-Nummer
	Register DE	- Zeiger auf Dateiname, Ende mit 00H
PA:	CY=1	Fehler
VR:	AF, BC, DE, HL	

Bemerkung: Falls DE auf eine leere Zeichenkette zeigt, wird der Dateiname abgefragt. Fehlermeldungen werden auf dem Bildschirm angezeigt.
Diese Funktion steht ab CAOS 4.7 zur Verfügung!

Name: . . . REN . . . . . PV7, UP-Nr. 11

FKT:	Datei umbenennen	
PE:	(IX+8)	- Bit 2-4: Device-Nummer
	Register DE	- Zeiger auf 2 Dateinamen, als erstes der alte Dateiname, als zweites der neue Dateiname, getrennt mit Leerzeichen, Ende mit 00H
PA:	CY=1	Fehler
VR:	AF, BC, DE, HL	

Bemerkung: Falls DE auf eine leere Zeichenkette zeigt, wird der alte und der neue Dateiname abgefragt. Fehlermeldungen werden auf dem Bildschirm angezeigt.
Diese Funktion steht ab CAOS 4.7 zur Verfügung!

3.6. Arbeitszellen des Betriebssystems

3.6.1. Arbeitszellen im System-RAM

Der Adressbereich von Adresse 0000H bis 01FFH im RAM0 wird von CAOS für systemnahe Arbeitszellen, die Interrupttabelle und den Systemstack genutzt. Bei Nutzung durch den Anwender sind Überschneidungen mit Arbeitszellen anderer Programme zu vermeiden. Die nachfolgende Übersicht soll dabei eine Hilfestellung sein, kann aber keinen Anspruch auf Vollständigkeit erheben.

Tabelle 20: Verwendung von Arbeitszellen im RAM0

Adresse	In Benutzung für	Bemerkung
0000H - 000AH	Dateiname und -typ für Dateiroutinen	EDAS, ASM, EDIT, BASIC…
000BH	Interrupt-Mode für USB-Modul M052 *ACHTUNG, Inhalt nicht manuell verändern!*	Ab USB-Version 2.7
000CH - 000FH	Datumstempel für Dateiarbeit (USB)	Siehe Kapitel 3.6.3.
0000H - 004DH	Hilfsroutine beim Start von ZAS für CP/M-Mode (JUMP FC)	D004 bis V3.3.1 D008 bis V3.4.1 *18
0000H - 0078H	Ausgabe von Fehlermeldungen beim Booten von D004/D008	D004 bis V3.3.1 D008 bis V3.4.1 *18
0000H - 00D5H	FLOAD2.KCC ein erweitertes FLOAD mit Anzeige von Fehlern in Klartext ab DEP 2.1 und Möglichkeit der Autostart-Unterdrückung (inklusive CALL*12 für BASIC)	Ab CAOS 4.7 nicht mehr erforderlich (wird ab DEP 3.4 nicht mit geladen) *18
0000H - 00E4H	FLOAD.KCC Originalversion von Mühlhausen (inklusive CALL*12 für BASIC)	Ab CAOS 4.7 nicht mehr erforderlich *18
0000H - 00FFH	Warmstart der D004/D008-CAOS-Betriebsart (JUMP FC 0), kopiert FLOAD ab Adresse 0	Ab CAOS 4.7 nicht mehr erforderlich *18
0010H - 006AH	Arbeitszellen M052-Terminal für VNC1 (einschließlich Interruptroutine)	Ab USB-Version 2.6
0010H - 001BH	Arbeitszellen M052-Terminal für VNC2 (ohne Interruptroutine)	Ab USB-Version 2.6
0010H - 0093H	Arbeitszellen Vinculum-Terminalprogramm	Bis USB-Version 2.5
0010H - 009FH	Arbeitszellen von Assembler/Editor	Ab CAOS 4.7
0010H - 00A9H	Arbeitszellen vom KC-Debugger 2.1	

*18 Nicht relevant für ein aktuelles System (CAOS 4.8, D004/D008 ROM ab 3.5, DEP 3.5)

Adresse	In Benutzung für	Bemerkung
0040H - 00C8H	Arbeitszellen EDAS 1.4-1.6	Ab CAOS 4.7 nicht mehr erforderlich *18
00AEH - 00E0H	VNC2-Interrupt M052-USB ab CAOS 4.7 (auch für PS/2- und USB-Tastatur)	Ab USB-Version 2.7
00D8H - 0107H	Einsprungtabelle der BASIC-Diskettenfunktionen bei SERVICE.KCC in der Originalversion von Mühlhausen bzw. CAOS 4.3 bis 4.5	Ab CAOS 4.7 nicht mehr erforderlich *18
0108H - 012FH	Tasten-UP von CALC, falls dieses mit %?i aktiviert worden ist.	ab CAOS 4.5/4.6
0108H - 013AH	VNC2-Interrupt M052-USB bis CAOS 4.7	Ab USB-Version 2.7
0150H - 016FH	DISK/TAPE-Umschaltroutine aus BASEX.KCC bzw. CAOS 4.3 bis 4.5	Ab CAOS 4.7 nicht mehr erforderlich *18
0180H - 019CH	Temporäre Ladeadresse der Datei *.JOY	ab CAOS 4.6
…….H - 01C3H	Systemstack siehe Kapitel 3.6.5	*19
01C4H - 01EFH	Interrupttabelle siehe Kapitel 3.6.6	*19
01F0H - 01FFH	IX-Arbeitszellen siehe Kapitel 3.6.7	*19

3.6.2. Verwendung von Restart-Befehlen

Der Z80 bzw. U880 bietet 8 Restart-Befehle. Restart-Befehle führen zu festgelegten Adressen im RAM0, das sind:

0000H RST 0
0008H RST 8
0010H RST 10H
0018H RST 18H
0020H RST 20H
0028H RST 28H
0030H RST 30H
0038H RST 38H

*19 Systemstack, Interrupttabelle und IX-Arbeitszellen können mithilfe des Unterprogramms SIXD auf einen anderen Adressbereich umgelegt werden, um den Speicherbereich von 0100H bis 01FFH anderweitig zu nutzen. (siehe Kapitel 3.6.4 „Verlagern von Arbeitszellen des Betriebssystems“). Die anderen in der Tabelle 20 aufgeführten Arbeitszellen werden damit aber nicht verlagert.

Die Wirkung ist identisch mit einem CALL, jedoch ist der Restart-Befehl nur ein Byte lang anstatt der drei Byte bei einem CALL. Die Benutzung von Restart-Befehlen ist also geeignet um den Speicherbedarf von Programmen zu verringern und wird gern für häufig aufgerufene Unterprogramme eingesetzt. Beim Einschalten des KC 85 sind auf diesen Adressen zunächst keine Funktionen aktiviert. Das heißt, ein Anwenderprogramm muss seine gewünschten Routinen zunächst eintragen.
Bei der Verwendung der Restart-Befehle ist noch zu beachten, dass dieser Speicherbereich im System-RAM auch von anderen Programmen bzw. Programmteilen mit verwendet wird, siehe Kapitel 3.6.1.
Zum Beispiel verwendet das Hilfsprogramm FLOAD oder auch die USB-Routinen der M052-Software den Bereich von 0 bis 000AH zur Ablage des Dateinamens.
Soll also Dateiarbeit stattfinden, dann sind die Restart-Befehle RST 0 und RST 8 tabu.
Von Adresse 000BH bis 000FH stehen im System-RAM wichtige Daten für den Betrieb des M052 und einer eventuell angeschlossenen USB-Tastatur.
Der Bereich ab Adresse 0010H wird von einigen Programmen als Zwischenspeicher für Arbeitszellen oder ähnliches benutzt. Beispiele hierfür sind der Editor, Assembler, Testmonitor, aber auch das USB-Terminalprogramm des Moduls M052. Da der KC 85 aber immer nur ein Programm abarbeiten kann, ist es beispielsweise nicht relevant, ob der Editor diese Arbeitszellen benutzt – solange man den Editor nicht startet.
Ein Anwendungsprogramm kann also durchaus einen Befehl RST 10H benutzen, muss allerdings dafür sorgen, dass ab Adresse 0010H der entsprechende Programmcode eingetragen wird, der bei RST 10H abgearbeitet werden soll.
In der Regel wird dann dort nur ein Sprungbefehl eingetragen.

Beispiel:

Für die Ausgabe von Zeichenketten existiert das Unterprogramm 23H mit der Bezeichnung OSTR. Dieses soll für ein Anwenderprogramm durch einen Restart-Befehl realisiert werden.

```
LD    DE,10H          ; hier wird der RESTART-Befehl eingetragen
LD    A,0C3H          ; Befehlscode für JUMP-Befehl
LD    (DE),A          ; Befehlscode eintragen
INC   DE              ; nächste Speicherzelle für Sprungziel
LD    HL,(0B7B0H)     ; SUTAB-Adresse ermitteln
LD    BC,2*23H        ; Offset zu Routine OSTR in der SUTAB
ADD   HL,BC           ; HL zeigt jetzt auf die Adresse der Routine OSTR
LDI                   ; niederwertiger Teil der Adresse
LDI                   ; höherwertiger Teil der Adresse
…
RST   10H             ; Restart 10H ≙ OSTR
DB    'Hallo KC85',0DH,0AH,0
RET
```

In ähnlicher Weise können auch die anderen Restart-Befehle bis 38H belegt werden. Kritisch sind nur RST 0 und RST 8 wegen der Dateinamen und M052-Arbeitszellen. Und falls für CAOS-Versionen vor 4.6 ein FLOAD benötigt wird, dann sollte die Variante von SERVICE.KCC benutzt werden, welche nicht im RAM0 steht.

3.6.3. Systemzeit

Der KC 85 besitzt von Haus aus keine Systemzeit, deshalb unterstützt auch kein CAOS-Unterprogramm Funktionen zu Datum und Uhrzeit.

Bei Erweiterung des KC-Systems durch ein D004 läuft in dessen Prozessorsystem automatisch eine CTC-gesteuerte Uhr. Falls ein D008 (oder ein D004 mit GIDE-Interface) genutzt wird, dann gibt es sogar eine batteriegestützte Echtzeituhr (RTC = real time clock). Das Diskettenerweiterungsprogramm DEP, welches in der CAOS-Betriebsart im D004/D008 läuft, stellt ab der Version 3.0 zyklisch diese Systemzeit im BCD-Format im Koppel-RAM zur Verfügung. Damit kann von CAOS aus auf eine Systemzeit zugegriffen werden. Das Kommando TIME macht davon Gebrauch und zeigt Datum und Uhrzeit an, siehe Seite 85.

Bei Erweiterung des KC-Systems durch ein USB-Modul M052 können Dateien von einem USB-Stick gelesen und auf diesen geschrieben werden. Ohne Systemzeit erhalten alle mit dem KC auf USB geschriebenen Dateien ein fest vom VNC-Chip vorgegebenes Datum (20.12.2004 00:00:00 Uhr).

Gibt es im KC 85-System ein M052, und kein D004 oder D008 mit laufendem DEP 3.x, dann wird beim Systemstart das Versionsdatum der M052-Software als festes Datum abgelegt. Man kann auch das USB-Terminalprogramm starten, wobei dann das aktuelle Datum abgefragt und ebenfalls mit der entsprechenden Codierung abgelegt vom SAVE-Kommando genutzt wird. Geschriebene Dateien erhalten dann dieses Datum und die Uhrzeit 00:00:00 Uhr.

Befindet sich im KC 85-System sowohl ein M052 als auch ein D004/D008 mit laufendem DEP.COM ab Version 3.0, dann nutzt der USB-Treiber im EEPROM des Moduls die Systemzeit vom D004/D008 beim Schreiben von USB-Dateien. Die geschriebenen Dateien erhalten somit das aktuelle Datum.

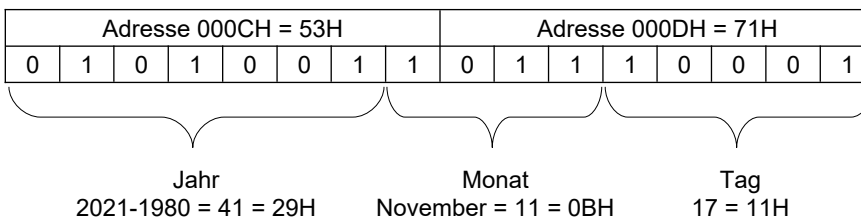
Die USB-Kommandos benötigen die Systemzeit aber nicht im BCD-Format, deshalb werden die Daten vorher vom Treiber konvertiert und im System-RAM ab Adresse 000CH abgelegt. Tabelle 21 zeigt, in welchem Format die Systemzeit dort eingetragen ist. Das gilt für die M052-Software ab der Version 2.8.

Tabelle 21: Format der Systemzeit für USB-Kommandos

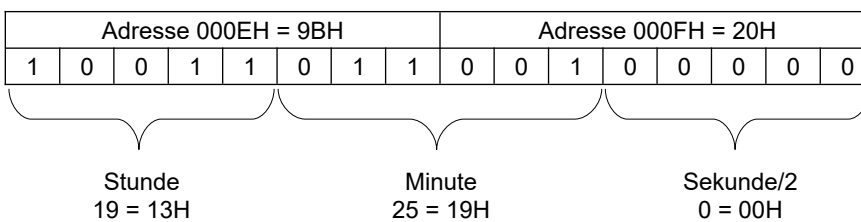
Adresse	Bit-Positionen	Beschreibung	Wertebereich	Bedeutung
000CH - 000DH	9-15	Jahr	0-127	0 = 1980 127 = 2107
	5-8	Monat	1-12	1 = Januar 12 = Dezember
	0-4	Tag	1-31	1 = erster des Monats
000EH - 000FH	11-15	Stunde	0-23	24-Stunden-Format
	5-10	Minute	0-59	
	0-4	Sekunde/2	0-29	0 = 0 Sekunden 29 = 58 Sekunden

Gibt es beispielsweise durch andere Module weitere Zeit-Quellen im KC-System, dann kann jedes Anwenderprogramm die Systemzeit entsprechend Tabelle 21 dort ablegen. Ein darauf folgendes USB-SAVE-Kommando wird dann die dort abgelegte Systemzeit verwenden. Ist aber ein D004/D008 mit DEP.COM ab Version 3.0 vorhanden, dann hat diese Systemzeit Vorrang und der Wert in den Systemzellen wird bei jedem SAVE mit der aktuellen Systemzeit überschrieben.

Beispiel: 17. November 2021



Beispiel: 19:25 Uhr



Nachteil/Kompromiss dieser Variante:

Befindet sich kein D004/D008 im System oder läuft kein DEP ab Version 3.0, dann wird beim Schreiben auf den USB-Stick der Wert aus den Systemzellen 000CH-000FH benutzt. Wurden nun durch ein anderes Programm die Daten in den Speicherzellen überschrieben, so erhalten die auf den USB-Stick geschriebenen Dateien ein falsches oder ungültiges Datum.

3.6.4. Verlagern von Arbeitszellen des Betriebssystems

Das I-Register der CPU wird beim Einschalten oder bei RESET auf 01, das IX-Register auf 01F0H gesetzt. Damit ist der System-RAM im Adressbereich 0100H bis 01FFH eingestellt. Falls dieser Speicherbereich anderweitig benötigt wird, kann der Arbeitsbereich einschließlich der beiden Register umgeladen werden, wobei der niederwertige Teil des Registers IX erhalten bleiben muss.
Die festen Systemzellen von 0000H bis 019CH (siehe Tabelle 20) werden dabei nicht mit verlagert!

Im Folgenden soll an einem Beispiel erläutert werden, wie der Arbeitsspeicherbereich (Stack, Interrupttabellen und IX-Bereich) vom RAM0 auf das Ende des RAM4 verlagert werden kann.

```
DI                  ; Sperren Interrupt
LD      SP,7FC4H    ; 32 Byte freihalten
                    ; für USER-Interrupttabelle
LD      (0B7AEH),SP ; Merken Stackanfang
LD      A,7FH       ; Höherwertiger Teil IX, I-Register
LD      E,31H       ; UP-Nr. 31H SIXD
CALL    F009H       ; PV3
EI                  ; Freigabe Interrupt
```

! Eine Verlagerung in den IRM ist nur bedingt möglich, der IRM darf in diesem Fall nicht mehr ausgeblendet werden und der RAM8 oder andere Speichermodule können nicht mehr auf der Adresse 8000H genutzt werden. Den Bereich ab Adresse 0C000H sollte der Anwender nicht benutzen, da dieser Bereich vom CAOS-ROM im Bedarfsfall belegt wird. Beim Nichtbeachten dieser Hinweise kann es zu undefinierten Zuständen des Systems kommen.

3.6.5. Kellerspeicher (Stack)

Der Stackpointer (SP) wird beim RESET/Einschalten auf 01C4H (01D4H bei CAOS 2.2 und CAOS 3.3) gesetzt, kann aber auf jeden anderen freien Speicher gelegt werden. Der Speicherplatz SYSP (B7AEH) dient als Merkzelle für den Initialisierungswert des SP.

3.6.6. Interrupttabelle

Das I-Register der CPU wird beim RESET/Einschalten auf 01H gesetzt, solange die Systemarbeitszellen nicht umgelagert werden, ergibt sich folgende Lage der Interrupttabelle:

Tabelle 22: Belegung Interrupttabelle

Adresse	Beschreibung		ab CAOS	
01C4H-01CFH	Frei für Anwender-Interrupts			
01D0H	Interrupt USB-PIO von Modul M052		4.7	*20,21
01D2H	-			*20,21
01D4H	Empfehlung für CTC Kanal 2 des 2. M003	(COM 3/4)	–	*22,23
01D6H	Empfehlung für CTC Kanal 3 des 2. M003	(COM 3/4)	–	*22,23
01D8H	Empfehlung für SIO des 2. M003	(COM 3/4)	–	*22,23
01DAH	Empfehlung für SIO des 1. M003	(COM 1/2)	–	*24,23
01DCH	Empfehlung für CTC Kanal 2 des 1. M003	(COM 1/2)	–	*24,23
01DEH	Empfehlung für CTC Kanal 3 des 1. M003	(COM 1/2)	–	*24,23
01E0H	Interrupt PIO Kanal A des Joystick-Moduls (M008/M021)		4.5	*25
01E2H	Interrupt SIO Kanal B des 1. M003		4.1	*24
01E4H	Interrupt PIO Kanal A - Kassetteneingabe		2.2	
01E6H	Interrupt PIO Kanal B - Tastatur		2.2	
01E8H	Interrupt CTC Kanal 0 - kein Interrupt, Tonhöhe 1 (rechts)		2.2	
01EAH	Interrupt CTC Kanal 1 - Kassettenausgabe, Tonhöhe 2 (links)		2.2	
01ECH	Interrupt CTC Kanal 2 - Tondauer, Blinkfrequenz		2.2	
01EEH	Interrupt CTC Kanal 3 - Tastatur		2.2	

Die Unterstützung von zwei V.24-Modulen mit 4 COM-Ports COM1 bis COM4 ist nicht in CAOS enthalten. Die angegebenen Adressen stellen eine Empfehlung für externe Programme dar, wie z. B. den Mauspfeiltreiber von UNIPIC, können aber auch anderweitig genutzt werden.

*20 Nur wenn ein USB-Modul M052 im System vorhanden ist
*21 01D0 und 01D2 wird von KOP-Treiber der M052-Netzwerk-PIO benutzt
*22 Nur wenn zwei V.24-Module im System vorhanden sind
*23 Derzeit nur benutzt, wenn ein externer Mauspfeiltreiber aktiv ist.
*24 Nur wenn ein V.24-Modul M003/M053 im System vorhanden ist
*25 Nur wenn ein Joystick-Modul M008/M021 im System vorhanden ist

3.6.7. Arbeitszellen im IX-Bereich

Das IX-Register wird beim RESET/Einschalten auf 01F0H geladen, im IX-Bereich liegen folgende Arbeitszellen von CAOS:

Tabelle 23: Belegung der IX-Arbeitszellen

Adresse	Bedeutung
IX + 0:	Zeitkonstanten-Speicher für Kassettenroutinen
IX + 1:	Merkzelle für Ausgabe Port 84H (bis CAOS 3.3: Prüfsummenberechnung innerhalb LOAD/SAVE)
IX + 2:	aktuelle Blocknummer bei Kassetten-Ein-/Ausgabe
IX + 3:	erwartete Blocknummer bei Kassetten-Ein-/Ausgabe
IX + 4:	Merkzelle für Ausgabe Port 86H (bis CAOS 3.3: Merkzelle für Autostart bei LOAD)
IX + 5: IX + 6:	Pufferadresse für Kassetten-Ein-/Ausgabe (Normal: B700H)
IX + 7:	Bit-Parameter für Kassetten-Ein-/Ausgabe Bit 0 0 = VERIFY 1 = READ Bit 1 1 = Autostart unterdrücken Bit 2-4 Anzahl Argumente Bit 5 1 = Blocknummernausgabe unterdrücken bei MBIN/MBOUT Bit 6 RESET-Schutz für BASIC-Programme, nur bei CAOS 3.1 Bit 7 RESET-Schutz für MC-Programme, bis CAOS 3.1
IX + 8:	Bit-Parameter für Tastatureingabe Bit 0 1 = Tastencode steht zur Verfügung Übernahmequittierung mit RES 0, (IX+8) Bit 1 1 = Tonausgabe läuft Bit 2-4 Gerätenummer ab CAOS 4.6 (Bit 3 und 4 werden bis CAOS 3.4 vom Tastaturtreiber genutzt) Bit 5 1 = Tastenklick EIN Bit 6 1 = Tastencodes werden aus Zeichenkette entnommen (F-Taste), Zeiger ist FTAST = B7D1H Bit 7 0 = SHIFT LOCK
IX + 9:	Prologbyte für Menü ab CAOS 3.1 (CAOS-Standard = 7FH, weitere siehe Seite 131)
IX +10:	Zähler für Autorepeat der Tastatur
IX +11:	Zwischenspeicher für Register A bei IRMON / IRMOFF (Prompt-Zeichen bei OS/PI'88 und OS/PI'90)
IX +12:	Scancode von Tastatur
IX +13:	Tastaturcode (ASCII)
IX +14: IX +15:	Low Tastaturcodetabelle KTAB High Tastaturcodetabelle KTAB

3.6.8. Arbeitszellen im IRM

Tabelle 24: IRM-Arbeitszellen

Adresse	Name	Länge (Byte)	Inhalt / Beschreibung
A800H - A81FH	V24PL	32	V.24-Arbeitszellen und Initialisierungstabellen ab CAOS 4.1 (beim KC 85/2 und KC 85/3 liegt von A800H bis B1FFH der COLOR-RAM)
A820H - A87FH		96	reserviert für Maustreiber oder die Verwaltung mehrerer V.24-Module
A880H - A8FFH	INITU	128	Kommandoablage von INITIAL.UUU als Tastencode ab CAOS 4.7 Arbeitszellen von Assembler und Reassembler
A900H - A9FFH	DEVTAB	256	Devicetreibertabellen für 8 Geräte (8*32 Byte) ab CAOS 4.7
AA00H	SULEN	1	Länge der SUTAB (erste unbenutzte UP-Nr.) ab CAOS 4.8
AA01H - AA9AH	SUTAB	154	Arbeitskopie der Unterprogrammtabelle im IRM (UP-Nr. 00H bis 4CH) ab AA00H bei CAOS 4.6 und 4.7
AA9BH - AABFH		36	reserviert für Erweiterung der SUTAB (UP-Nr. 4DH bis 5EH)
AAC0H - AADCH	JOYIRM	29	Joystick-Tabelle und JEDIT-Arbeitszellen ab A894H bei CAOS 4.5 und 4.6 ab AA94H bei CAOS 4.7
AADDH - AAA6H	SHADOW	11	Schattenspeicher für Fensterdaten inaktives Bild ab CAOS 4.8
AAE8H - AAFFH	WIN-TEMO	18	Fenster-Daten bei Debugger-Betrieb, je 2x WINNR, WINON, WINLG, CURSO, STBT, COLOR, (IX+1) und CCTL0
AB00H - ACFFH		512	reserviert für anwenderspezifische Systemerweiterungen (z. B. Treiber für UOUT1/2, UIN1/2)
AD00H - B1FFH	VRAM1	1280	Video-RAM für Bild 1 (ASCII-Speicher) ab CAOS 4.1
B200H - B6FFH	VRAM0	1280	Video-RAM für Bild 0 (ASCII-Speicher)

Adresse	Name	Länge (Byte)	Inhalt / Beschreibung
B700H - B77FH	CASS	128	Sektorpuffer für Dateioperationen (Kassette, Diskette, USB …)
B780H	ARGC	1	UP-Nr. bei PV2 und PV6 ab CAOS 4.6: zusätzlich USER-ROM-Zustand bei Menüwortaufruf aus USER-ROM
B781H	ARGN	1	Anzahl der Argumente bei Kommandoeingabe
B782H	ARG1	2	1. Argument
B784H	ARG2	2	2. Argument
B786H	ARG3	2	3. Argument
B788H	ARG4-9	12	4.-9. Argument
B794H	ARG10	2	10. Argument
B796H	NUMNX	1	Zeichenanzahl der erfassten Zahl
B797H	NUMVX	2	Wert der erfassten Zahl
B799H	HCADR	2	Adresse für Kanalumschaltung der System-Ein/Ausgabe. Aufruf über Tastatur; Code 0FH DE enthält Cursorposition.
B79BH	WINNR	1	Nr. des aktuellen Bildschirmfensters
B79CH	WINON	2	Fensteranfang L: Spalte (0 … 39 bzw. 0H … 27H) H: Zeile (0 … 31 bzw. 0H … 1FH)
B79EH	WINLG	2	Fenstergröße L: 1 … 40 bzw. 28H Spalten H: 1 … 32 bzw. 20H Zeilen (je nach WINON)
B7A0H	CURSO	2	Relative Cursor-Position im Fenster L: Spalte H: Zeile
B7A2H	STBT	1	Steuerbyte für Bildschirmprogramm Bit 0 = 1 Schreiben Pixel-IRM AUS Bit 1 = 1 Schreiben Color-IRM AUS Bit 2 = 1 Inversdarstellung EIN (ab CAOS 3.4) Bit 3 = 1 Steuercode als Zeichen darstellen Bit 4 = 1 ESC aktiv (ab CAOS 4.1) Bit 5 = 1 IBM-Zeichensatz EIN (ab CAOS 4.3) Bit 6 = 1 HRG-Modus EIN (ab CAOS 4.3) Bit 7 = 1 2-Monitor-Modus EIN (ab CAOS 4.8)

Adresse	Name	Länge (Byte)	Inhalt / Beschreibung
B7A3H	COLOR	1	Farbbyte für Bildschirmprogramm B(7) B(6) B(5) B(4) B(3) B(2) B(1) B(0) A(V) X(V) G(V) R(V) B(V) G(H) R(H) B(H) Index: V: Vordergrund (Farbe für Bit im Pixel-IRM=1) H: Hintergrund (Farbe für Bit im Pixel-IRM=0) B: Farbe blau R: Farbe rot G: Farbe grün X: Verschiebung im Farbkreis um 30° A: Alternierende Zeichendarstellung (Blinken der Vordergrundfarbe) Kombination der Bits ergeben Mischfarben.
B7A4H	WEND	2	Anfangsadresse des Reaktionsprogramms auf Erreichen des Fensterendes (Page-/Scrollmode usw.)
B7A6H	CCTL0	2	Adresse der Zeichenbildtabelle für Codes 20H-5FH normal: EE00H
B7A8H	CCTL1	2	Adresse der Zeichenbildtabelle für 00-1FH und 60-7FH normal: FE00H
B7AAH	CCTL2	2	Adresse der Zeichenbildtabelle für Codes A0-DFH normal: EE00H
B7ACH	CCTL3	2	Adresse der Zeichenbildtabelle für 80-9FH und E0-FFH normal: FE00H
B7AEH	SYSP	2	INIT-Adresse des System-Stackpointers normal: 01C4H (bei CAOS 2.2 und 3.3 ist SYSP = 01D4H)
B7B0H	SUTAB	2	Adresse der Unterprogrammtabelle
B7B2H	CTAB	2	Tabelle der Stammcodes für das Bildschirmprogramm
B7B4H	NCAOS	5	Einsprung in neues System über IRM
B7B9H	OUTAB	2	Adresse für Zeiger auf UP-Nr. für Ausgabe-Kanal (normal: 00=CRT - Bildschirm)
B7BBH	INTAB	2	Adresse für Zeiger auf UP-Nr. für Eingabe-Kanal (normal: 04=KBD - Tastatur)
B7BDH	UOUT1	3	Sprung in USER-Ausgabekanal 2 (z. B. BASIC PRINT #2 …)
B7C0H	UIN1	3	Sprung in USER-Eingabekanal 2 (z. B. BASIC INPUT #2 …)

Adresse	Name	Länge (Byte)	Inhalt / Beschreibung
B7C3H	UOUT2	3	Sprung in USER-Ausgabekanal 3 (z. B. BASIC PRINT #3 …)
B7C6H	UIN2	3	Sprung in USER-Eingabekanal 3 (z. B. BASIC INPUT #3 …)
B7C9H	IOERR	2	Sprungadresse zur Fehlerausgabe bei UP 37H MBIN und UP 38H MBOUT (Voreinstellung BASIC: Sprung zu ?IO-ERROR) ab CAOS 4.8 nicht mehr genutzt
B7CBH	VRAM	2	Beginn Video-RAM (ASCII-Speicher) ab CAOS 4.1 (bei CAOS 3.4, OS PI'88 und OS/PI'90 Merkzelle für BASIC-SP)
B7CDH	ZOTAB	2	Zwischenspeicher für OUTAB während TAPE Ein- und Ausgabe ab CAOS 4.1
B7CFH	ZWEND	2	Zwischenspeicher für WEND während TAPE Ein- und Ausgabe
B7D1H	FTAST	2	Zeiger für Abarbeitung der Zeichenfolge von F-Tasten
B7D3H	HOR	2	X-Wert für Grafikprogramm (0-319)
B7D5H	VERT	1	Y-Wert für Grafikprogramm (0-255)
B7D6H	FARB	1	Vordergrundfarbe / Blinken für Grafikprogramm, vgl. COLOR Bit 0 = 1 XOR-Funktion Bit 1 = 1 Punkte löschen Bit 2 = 1 Farbebene nicht verändern* Bit 3-7 Vordergrundfarbe + Blinken * Bit 2 nur unter CAOS 3.4 und ab 4.7 benutzt
B7D7H	MIXIT	1	Höherwertiger Teil von IX und der Interrupttabelle (vgl. folgender Abschnitt)
			Ab hier Arbeitszellen, die sich je nach CAOS-Version unterscheiden!
B7D8H	VORTN	2	Zwischenspeicher für Vortonlänge bei byteweiser TAPE Ausgabe ab CAOS 4.1 (bei CAOS 3.1 Vortonlänge auf Adresse B7DAH)
B7DAH	DTPTR	1	Zeiger in Kassettenpuffer für Byte-Routinen TAPE Ein- und Ausgabe (CAOS 4.2 benutzt dafür 3 Byte von B7D8H bis B7DBH)
B7DBH	CTCMD	1	Modus von CTC-Kanal 2 normal: 47H ab CAOS 4.3

Adresse	Name	Länge (Byte)	Inhalt / Beschreibung
B7DCH	BLINK	1	Zeitkonstante für CTC 2 (Blinkfrequenz) ab CAOS 4.3 normal: 0CH
B7DDH	L3TAB	2	Adresse der ESC-Steuerfunktionstabelle ab CAOS 4.1
B7DFH	L3SIZ	1	Anzahl der Steuerfunktionen ab CAOS 4.1
B7E0H	COUNT	1	Zeiteinheiten bis 1. Autorepeat (Tastatureingabe) ab CAOS 3.4 normal: 05H
B7E1H	HCPZ	1	Steuerbyte für Druckerausgabe Bit 0 0=Protokoll-, 1=Hardcopyfunktion Bit 1 0=USER-Kanal 1, 1=USER-Kanal 2 Bit 2 0=SIO-Kanal A, 1=SIO-Kanal B Bit 3 - Bit 4…7 Druckertyp (0 bis F) Bit 7 0=Matrixdrucker, 1=Schreibmaschine
B7E2H	INTV1	2	Anfangsadresse der Initialisierungstabelle für die V.24-Druckerausgabe
B7E4H	INTV1L	1	Länge der Initialisierungstabelle für die V.24-Druckerausgabe
B7E5H	INTV2	2	Anfangsadresse der Initialisierungstabelle für die V.24-Duplex-Funktion
B7E7H	INTV2L	1	Länge der Initialisierungstabelle für die V.24-Duplex-Funktion
B7E8H	HCPZ2	1	Steuerbyte für V.24-Duplexroutine Bit 0 - Bit 1 0=USER-Kanal 1, 1=USER-Kanal 2 Bit 2 0=SIO-Kanal A, 1=SIO-Kanal B Bit 3 WR5 Bit 3: Senden ein/aus Bit 4 - Bit 5,6 WR5: Bit pro Zeichen (Senden) Bit 7 WR3: Bit pro Zeichen (Empfang)
B7E9H	OFILT	3	Sprung zur Druckerausgaberoutine (V.24 oder Centronics)
B7ECH	PROMPT	1	System-Promptzeichen ab CAOS 4.3 CAOS-Standard 24H % Assembler 25H * Testmonitor TEMO 2BH + Editor 2DH - Joystick-Editor JEDIT 2EH .

Adresse	Name	Länge (Byte)	Inhalt / Beschreibung
B7EDH	LINTYP	1	Linientyp, Standardwert FFH (volle Linie) Strichlinie 0FH Strich-Punkt-Linie 5FH ab CAOS 4.3
B7EEH	CUMUST	2	Zeiger auf Cursormuster ab CAOS 4.3
B7F0H	JOYTAB	2	Zeiger auf Tabelle der Joystick-Tastencodes ab CAOS 4.5
B7F2H - B7F4H		3	reserviert
B7F5H - B7FFH	FNAME	11	Zwischenspeicher für Dateiname, z. B. bei ASM, EDIT MBIN und MBOUT ab CAOS 4.7
B800H - B8FFH	MODST	256	Modulsteuerbytespeicher ab CAOS 3.4 für interne Module initialisiert
B900H - B99BH	FNDEF	156	Funktionstastenspeicher, siehe Kapitel 3.7 Seite 192
B99CH - B9FFH	WNDFN	100	Fenstervektorspeicher für 10 Fenster, siehe Kapitel 3.2.1 Seite 123 (ab CAOS 3.1)
BA00H - BFFFH		1536	Frei für Anwender mit erhöhter Zugriffszeit *[26]

*26 Die Zugriffszeit auf den gesamten IRM-Speicherbereich von 8000H bis BFFFH kann sich auf 2,4µs erhöhen, da der Speicherzugriff mit dem Grafiksystem synchronisiert werden muss. Dies verzögert die Programmabarbeitung in diesem Bereich.

3.7. Funktionstasten

Die Funktionstasten liefern von den Tastaturprogrammen KBD, KBDZ die in der folgenden Tabelle aufgeführten folgende Codes.

Tabelle 25: Codes der Funktionstasten

Taste	Code: 1. Belegung	Code 2. Belegung
F1	F1H	F7H
F2	F2H	F8H
F3	F3H	F9H
F4	F4H	FAH
F5	F5H	FBH
F6	F6H	FCH

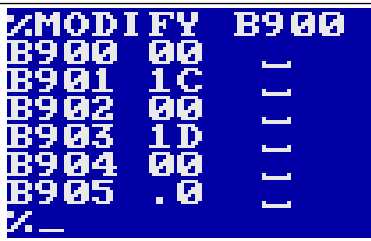
Beim Betätigen einer Funktionstaste wird vom Tastaturprogramm KBDS die Zeichenübergabe auf Zeichen aus dem zugehörigen Puffer (ab B900H) umgeschaltet, der Pufferaufbau ist dynamisch. Das heißt, die Zeichenanzahl zu den einzelnen Funktionstasten liegt nicht fest, sondern wird nur von der Puffergröße begrenzt. Der Puffer muss mit 00 beginnen und mit 00 abgeschlossen werden. Die Zeichenketten für die einzelnen F-Tasten werden ebenfalls durch ein 00-Byte getrennt. Es sind als Codes alle Codierungen zugelassen. Normalerweise erfolgt die Belegung der F-Tasten durch die CAOS-Anweisung KEY oder durch die gleichnamige BASIC-Anweisung.
Dabei ist es möglich, auf den F-Tasten „JOBS“ abzulegen, deren Abarbeitung mittels <BRK>-Taste abgebrochen werden kann.

3.7.1. Speicher für Funktionstastenbelegung

Sollen auf den Funktionstasten Codes abgelegt werden, die nicht auf der Tastatur vorhanden sind, kann dies durch das MODIFY-Kommando im Betriebssystem, durch die VPOKE-Anweisung vom BASIC-Interpreter aus oder direkt über ein Maschinenprogramm erfolgen.

Beispiel:

Es sollen nicht auf der Tastatur befindliche Codes über die F-Tasten erzeugt werden.

Eingabe	Erläuterung
	Kommando mit <ENTER> abschließen erstes Trennzeichen F1: LIST - Kommando zweites Trennzeichen F2: RUN - Kommando drittes Trennzeichen <Punkt><ENTER> Verlassen von MODIFY

Eine Veränderung bzw. Anzeige der somit eingegebenen Codes ist mit KEY bzw. KEYLIST möglich. Die F-Tastenpuffergröße (0B900H-0B99BH) muss bei MODIFY-Eingabe vom Anwender selbst überwacht werden!

3.7.2. Belegen der Funktionstasten mit Steuerzeichen

Eine Belegung der Funktionstasten mit ESC-Funktionen ist über die Funktion KEY nicht möglich, weil diese Funktionen im Eingabemodus sofort ausgeführt werden. Diese Belegung ist aber über das Systemkommando MODIFY möglich.

Beispiel:

Belegung der Funktionstasten <F1> und <F2> mit „Bild 0 anzeigen und schreiben“ (ESC '1') bzw. „Bild 1 anzeigen und schreiben“ (ESC '2'). Mit der Eingabe von:

%MODIFY B900 <ENTER>-Taste
B900 00 1B 31 00 1B 32 00 . <ENTER>-Taste
(Abschluss mit Punkt und <ENTER>-Taste)

werden die Funktionen auf die beiden Funktionstasten gelegt.

3.8. Magnetbandaufzeichnung

3.8.1. Verfahren

Die Aufzeichnung auf Kassette erfolgt nach einem Verfahren, welches Vorteile bezüglich Übertragungsrate und Synchronisation gegenüber anderen bekannten Verfahren bietet. Es wird nicht nur vom KC 85/2…5, sondern auch vom KC 85/1 und KC87 von Robotron verwendet. Dadurch ist ein Datenaustausch zwischen beiden Systemen möglich. Zur Aufzeichnung dienen drei verschiedene Frequenzen, wobei jeweils eine komplette Schwingung eine logische Einheit umfasst:

Nullbit:	f = 2400Hz	(KC 85/2-5 ca. 1950 Hz)
Einsbit:	f = 1200Hz	(KC 85/2-5 ca. 1050 Hz)
Trennzeichen:	f = 600Hz	(KC 85/2-5 ca. 557 Hz)

Jedes Byte wird wie folgt auf Magnetband abgespeichert:

Byteaufbau:	8 Datenbits beginnend mit Bit 0 (1200Hz-Schwingung je 1-Bit bzw. 2400Hz-Schwingung je 0-Bit), danach ein Trennzeichen (600Hz-Schwingung)

3.8.2. Dateiaufbau

Die Datenbytes werden auf dem Magnetband sequentiell in Blöcken zu je 128 Bytes abgespeichert. Jeder Block besitzt diesen Aufbau:

- Vorton: bestehend aus Schwingungen mit 1200Hz (Einsbit)
  - erster Block: langer Vorton, etwa 4000 Schwingungen
  - folgende Blöcke: je nach Verarbeitungszeit (für MC-Programm 160 Schwingungen)
- Ein Trennzeichen
- Ein Byte Blocknummer (erster Block Nr. 01H; folgende Blöcke aufsteigend nummeriert; letzter Block Nr. FFH)
- 128 Datenbytes
- Ein Byte Prüfsumme über die 128 Datenbytes

Hinweis: Beim Speichern identischer Dateien auf anderen Datenträgern werden nur die 128 Byte Nutzdaten abgespeichert. Blocknummern und Prüfsummen sind nur bei der Magnetbandaufzeichnung notwendig.

Jede Datei besteht aus einem Vorblock (Block Nr. 01H) und nachfolgenden Datenblöcken, ist also mindestens zwei Blöcke lang. Der letzte Block erhält immer die Blocknummer FFH, um das Dateiende anzuzeigen.

Der Vorblock einer MC-Datei ist wie folgt aufgebaut:

Tabelle 26: Aufbau CAOS-Vorblock (MC-Datei)

Byte-Nr.	Adresse in Puffer	Bedeutung
1. Byte	(IX+2)	Blocknummer = 01H
2. - 9. Byte	B700H	Name, bestehend aus alphanumerischen Zeichen
10. - 12. Byte	B708H	Dateityp, vgl. folgende Abschnitte
13. - 17. Byte	B70BH	Reservierte Bytes für den Hersteller Für Anwenderprogramme müssen diese 00H enthalten.
18. Byte	B710H	Anzahl der nachfolgenden 2-Byte-Argumente. Für ladbare Maschinenprogramme und Speicherabzüge (DUMP) muss dieses Byte einen Wert zwischen 02H und 0AH enthalten. Dabei gilt: 02H: Programm wird geladen, danach Rückkehr in das rufende Programm. 03H: Programm wird geladen, danach Start des Programms bei angegebener Startadresse. Wird das Programm relativ geladen, so erfolgt der Start bei umgerechneter Startadresse. 04H…07H: wie bei 03H, jedoch ohne Umrechnung der Startadresse beim relativen Laden.
19. - 20. Byte	B711H	Ladeadresse
21. - 22. Byte	B713H	Endadresse + 1
23. - 24. Byte	B715H	Startadresse
25. - 129. Byte	B717H - B77FH	Diese Datenbytes können Parameter zur genauen Definition der Datei enthalten. Standardmäßig sind alle Bytes 00H.
130. Byte	-	Prüfsumme über die 128 Datenbytes

Anmerkungen:

1. Ab CAOS 4.8 wird der Dateiname mit bis zu 12 Zeichen in der Form 8.3 in den Vorblock eingetragen, also mit bis zu 8 Zeichen für den Namen, einem Punkt als Trennzeichen und bis zu 3 Zeichen für den Dateityp.
2. Bei BASIC-Dateien gilt ein anderer Aufbau: Hier enthält der erste Block nach der Blocknummer 01H eine 3-Byte-Typ- und eine 8-Byte-Namensinformation. Ab dem 13. Byte des ersten Blockes sind bereits Daten enthalten.
3. Auch Textdateien vom CAOS-Editor haben einen anderen Aufbau. Hier enthält der Vorblock, welcher ausschließlich bei der Magnetbandausgabe zusätzlich vorhanden ist, nur den Dateinamen. Alle restlichen Bytes sind 00H.

3.8.3. Dateitypen

Da jedes Dateiformat einen anderen Aufbau hat, werden zur Unterscheidung Dateitypen aus 3 Zeichen benutzt. Diese sind im Vorblock in den Bytes 10 bis 12 (bei BASIC in den Bytes 2 bis 4) anzugeben. Es gelten folgende Festlegungen:

Tabelle 27: Auswahl an Dateitypen

Dateityp	Beschreibung
(F)	FORTH – Quellprogramm
ASM	Quelltextdateien für Assemblerprogramme, z. B. ASM, EDAS
BAK	Sicherungskopie, wird teilweise automatisch beim Speichern angelegt (z. B. vom Assembler oder Editor)
COM	Maschinenprogramm COM-Dateien von CAOS werden beim Abspeichern auf Diskette vom Dienstprogramm DEP automatisch in KCC umbenannt um Verwechslungen mit CP/M-Programmen zu vermeiden.
DUM	Speicherabzüge
JOY	Joystick-Tastencodes abgespeichert mit JEDIT
KCB	BASIC-Programm, abgespeichert als Maschinenprogramm
KCC	CAOS-Maschinenprogramm
KEY	abgespeicherte F-Tasten-Belegungen
PAS	PASCAL-Quelltext
PIC	Bilddatei, Speicherabzug vom KC 85/3 *27
PIF	Bilddatei, Speicherabzug KC 85/4 Farbebene *27
PIP	Bilddatei, Speicherabzug KC 85/4 Pixelebene *27
SSS	BASIC-Programm (Sonderfall: Dateityp in den Bytes 2 bis 4)
TTT	BASIC Daten, Felder (Sonderfall: Dateityp in den Bytes 2 bis 4)
TXT	Textdatei, meist im ASCII-Zeichensatz
TXW	Textdatei von WordPro, meist IBM-Zeichensatz
UUU	BASIC-Programmlisting im Textformat oder Kommandofolgen (z. B. INITIAL.UUU)
WP9	Textdateien von WordPro9, Speicherabzug mit CAOS-Vorblock *27
WPR	Textdateien von WordPro'86, Speicherabzug mit CAOS-Vorblock *27

*27 Kann auch komprimiert sein.

3.9. DEVICE-Schnittstelle

3.9.1. Funktion der DEVICE-Umschaltung

Das CAOS-Betriebssystem beinhaltet Systemunterprogramme zur Ein- und Ausgabe auf Magnetband. Um verschiedene Speichergeräte nutzen zu können, müssen diese Unterprogramme je nach Speichergerät auf die zugehörigen Treiber zugreifen. Zur Speicherung der DEVICE-Nummer werden die drei bisher ungenutzten Bits 2–4 der Speicherzelle (IX+8) verwendet. CAOS merkt sich dort, welches DEVICE gerade aktiv ist. Damit können bis zu 8 Speichergeräte verwaltet werden.

In CAOS 4.6 wurden bei Umschaltung des Speichergerätes die Adressen in der Unterprogrammtabelle SUTAB umgeschrieben, weswegen die SUTAB nun generell im IRM liegt.

Ab CAOS 4.7 stehen in der SUTAB Unterprogramme, welche in Abhängigkeit der Bits 2–4 von (IX+8) in den jeweiligen Treiber verzweigen.

DEVICE 0 ist immer TAPE und damit kompatibel zu den bisherigen CAOS-Versionen. Befindet sich ein D004 oder D008 mit laufender CAOS-Betriebsart im KC 85-System, dann wird dieses das nächste Device und mit DISK bezeichnet. Beim Starten sucht CAOS außerdem nach Modulen M052 und M064. Falls sich in dessen (E)EPROM ein DEVICE-Treiber befindet, dann wird dieser automatisch eingebunden.

Im IRM-Adressbereich von A900H bis A9FFH befindet sich eine Device-Treiber-Tabelle. Diese enthält 8 Einträge zu je 32 Byte und wird beim Systemstart automatisch erzeugt. Die Basisadressen der Treiber sind:

A900H Treiber Nr. 0 (TAPE)
A920H Treiber Nr. 1
A940H Treiber Nr. 2
A960H Treiber Nr. 3
A980H Treiber Nr. 4
A9A0H Treiber Nr. 5
A9C0H Treiber Nr. 6
A9E0H Treiber Nr. 7

Tabelle 28: Aufbau der DEVICE-Treiber

Adresse	Größe	Funktion
+00H	Byte	Aktivierungs-Kennbyte. Bei einem aktiven Treiber entspricht dieses Byte der Nummer des Treibers. Also 00H beim TAPE-Treiber auf Adresse A900H, 01H beim Treiber Nr. 1 auf Adresse A920H usw. CAOS initialisiert inaktive Treiber mit Kennbyte FFH.
+01H	Byte	Modulsteckplatz, wo sich der Treiber befindet. CAOS schaltet das Modul auf diesem Steckplatz bei Aufruf einer Treiber-Funktion automatisch ein und anschließend wieder aus.
+02H	Byte	Steuerbyte zum Ausschalten des Moduls. Hier trägt CAOS vor Aufruf jeder Treiber-Funktion den aktuellen Modulzustand anhand der Modulsteuerbyte-Tabelle ein, um diesen Schaltzustand nach Rückkehr wieder zu regenerieren.
+03H	Byte	Steuerbyte zum Einschalten des Moduls. Hier steht das Steuerbyte mit dem das Modul mit der Treibersoftware auf den Adressbereich C000H eingeblendet werden kann.
+04H	4 Byte	Name des Treibers in ASCII. Dieser Name wird sowohl im CAOS-Menü als auch bei der Device-Umschaltung angezeigt.
+08H	Adresse	MBO Ausgabe eines 128-Byte-Datenblocks zur geöffneten Ausgabedatei
+0AH	Adresse	MBI Eingabe eines 128-Byte-Datenblocks von der geöffneten Eingabedatei
+0CH	Adresse	ISRO Ausgabedatei öffnen und ersten Block 01 ausgeben
+0EH	Adresse	CSRO Ausgabedatei schließen und Endeblock FF ausgeben
+10H	Adresse	ISRI Eingabedatei öffnen und Block 01 einlesen
+12H	Adresse	CSRI Eingabedatei schließen
+14H	Adresse	Abfrage Treiber-Version ab CAOS 4.8 (MBIN bei CAOS 4.6)

Adresse	Größe	Funktion
+16H	Adresse	Treiber-spezifische Funktionen ab CAOS 4.8 (MBOUT bei CAOS 4.6)
+18H	Adresse	DIR = Verzeichnisanzeige mit Maske Der Verzeichniseintrag wird am Bildschirm angezeigt, nach einer gefüllten Bildschirmseite wird eine Tastatureingabe abgewartet.
+1AH	Adresse	CD = Laufwerk- bzw. Verzeichniswechsel
+1CH	Adresse	ERA = Datei löschen
+1EH	Adresse	REN = Datei umbenennen

3.9.2. DEVICE nach Systemstart oder RESET

Beim Systemstart oder Betätigung der RESET-Taste wird immer der Treiber 0 für TAPE angelegt, welcher die Funktionen der früheren CAOS-Versionen mit Kassettenein- und -ausgabe bereitstellt.

Befindet sich im KC-System ein D004/D008 in der CAOS-Betriebsart, dann wird als Treiber Nr. 1 der DISK-Treiber angelegt, welcher die Funktionen der nachladbaren Programme FLOAD und FSAVE sowie der wichtigsten Dienstprogramme bereitstellt.

Weitere Treiber werden in allen vorhandenen Modulen mit Strukturkennbyte FDH (M052) oder F9H (M064) in der Reihenfolge der Steckplätze gesucht. Wird im (E)EPROM dieser Module ein DEVICE-Treiber gefunden, dann kopiert CAOS die ersten 32 Byte in die DEVICE-Treiber-Tabelle. Der Treiber im Speicher des Moduls hat fast den gleichen Aufbau wie in Tabelle 28 beschrieben, nur mit einem kleinen Unterschied: Auf den Adressen +00H und +01H muss das Kennbyte 46H stehen. Daran erkennt CAOS, dass es sich um einen DEVICE-Treiber handelt. In der DEVICE-Treiber-Tabelle werden die beiden ersten Bytes dann durch das passende Aktivierungskennbyte und den Modulsteckplatz ersetzt.

Befindet sich im KC-System ein D004/D008, dann wird beim Systemstart automatisch der DISK-Treiber aktiviert. Ansonsten der Treiber des ersten Moduls mit einem kompatiblen DEVICE-Treiber. Befindet sich kein anderes Speichergerät im KC 85/5, dann gibt es nur den TAPE-Modus, also wie bei den vorherigen CAOS-Versionen.

3.9.3. Erweiterung mit eigenen Treibern

CAOS kann bis zu 8 DEVICE-Treiber verwalten. Zusätzlich zu den Standard-Treibern für TAPE und DISK sowie den Treibern aus den Modulen, können auch eigene Treiber aus nachladbaren Programmen ergänzt werden. Dazu sollte das Programm nach dem ersten freien Treiber in der DEVICE-Treiber-Tabelle suchen und dann einen 32 Byte großen Datenblock entsprechend der Tabelle 28 dorthin kopieren. Muss zum Einspringen kein Modul geschaltet werden, dann kann für den Steckplatz 0 in der Tabelle stehen.

Beispiel:

```
        LD      HL,0A900H       ; DEVICE-Treiber-Tabelle im IRM
        LD      DE,32           ; Größe eines Treibers
        LD      B,8             ; 8 Treiber möglich
        XOR     A               ; A=0
M1:     CP      (HL)            ; Treiber aktiv?
        JR      NZ,M2           ; nein
        ADD     HL,DE           ; Zeiger auf nächste Treiber-Adresse
        INC     A               ; nächster Treiber
        DJNZ    M1
        ; alle 8 Treiber sind schon benutzt
        …

M2:     LD      (HL),A          ; Treiber aktivieren
        INC     HL
        EX      DE,HL
        LD      DE,DRV+1        ; eigener Treiber
        LD      BC,31           ; restliche Länge
        LDIR                    ; kopieren
        ; neuer Treiber ist kopiert und kann jetzt verwendet werden
        …

DRV:    DB      0               ; Aktivierungsbyte
        DB      0               ; Modulsteckplatz
        DB      1               ; Steuerbyte
        DB      'DRV'           ; Treiber-Name
        ; jetzt folgen die Einsprungadressen der 12 Routinen
        …
```

Um einen reibungslosen Betrieb zu gewährleisten, müssen sich eigene Treiber bezüglich der Parameter genau an den im Handbuch beschriebenen Routinen orientieren.

3.9.4. Treiberspezifische USB-Funktionen ab Version 3.0

Der mit CAOS 4.7 eingeführte Sprungverteiler PV7 gestattet über den DEVICE-Treiber den Aufruf von insgesamt 12 Funktionen. Die Funktionen 6 und 7 waren dabei ursprünglich (in CAOS 4.6) für die byteweisen Routinen der BASIC-Schnittstelle vorgesehen, stellten sich aber bei der Kassettenausgabe als zu langsam heraus. Ab CAOS 4.7 laufen MBIN und MBOUT deshalb über die blockweisen Routinen und die Funktionen 6 und 7 waren nicht mehr erforderlich. Deshalb ist eine anderweitige Nutzung möglich geworden.

Tabelle 29: Übersicht der DEVICE-Funktionen

Beschreibung	UP-Nr.:	PV1-PV6	PV7
MBO: Ausgabe Datenblock 128 Byte		01H	0
MBI: Eingabe Datenblock 128 Byte		05H	1
ISRO: Dateiausgabe öffnen (erster Block)		08H	2
CSRO: Dateiausgabe schließen (letzter Block)		09H	3
ISRI: Dateieingabe öffnen (erster Block)		0AH	4
CSRI: Dateieingabe schließen (letzter Block)		0BH	5
Abfrage der Treiberversion *28		-	6
Treiberspezifische Funktionen *28		-	7
DIR: Verzeichnisanzeige auf Bildschirm		-	8
CD: Verzeichnis wechseln		-	9
ERA: Datei löschen		-	10
REN: Datei umbenennen		-	11

Ab CAOS 4.8 wird mit der Funktion 6 eine Versionsabfrage der Treiber-Software realisiert. Der Rückgabewert in Register A stellt die Versionsnummer BCD dar. Der Wert 30H entspricht also Version 3.0.

Die Funktion 7 dient dem Aufruf von bis zu 256 Treiber-spezifischen Routinen. Die Unterfunktionsnummer ist dazu in Register E zu übergeben. Vor Verwendung der treiberspezifischen Funktionen wird der Aufruf der Funktion 6 zur Kontrolle der Treiberversion 3.0 oder höher empfohlen.

*28 Voraussetzung für die Verwendung der neuen Funktionen ist eine Treiberversion 3.0 oder höher, wie sie bei der USB-Software verfügbar ist. Ab CAOS 4.8 wird eine Treiberversion 3.0 oder höher zwingend vorgeschrieben. Somit kann davon ausgegangen werden, dass bei einem vorliegenden CAOS 4.8 die nachfolgend beschriebenen Funktionen zur Verfügung stehen.

3. SOFTWARE

Möchte ein Anwenderprogramm unter CAOS 4.7 prüfen, ob eine passende Treiberversion vorliegt, dann kann nachfolgender Programmcode genutzt werden, welcher auch bei älteren Versionen einen definierten Wert liefert:

```
LD      HL,0B700H       ; Kassettenpuffer
LD      D,L             ; D=0, kein INIT/CLOSE bei MBIN
LD      (HL),D          ; 0 in Kassettenpuffer eintragen
LD      L,0DAH          ; B7DAH ist der Zeiger im Kassettenpuffer
LD      (HL),D          ; 0 eintragen, Zeiger auf Pufferanfang
CALL    0F021H          ; PV7
DB      6               ; Versionsabfrage oder MBIN
CP      30H             ; Version 3.0 vorhanden?
JP      C,ERROR         ; ältere Software erkannt
```

3.10. Tastatur, Zeichenvorrat, Steuercodes

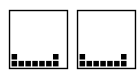
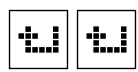
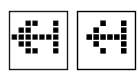
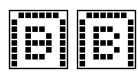
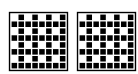
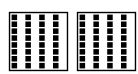
3.10.1. Zeichenvorrat des KC 85/5 und Zuordnung zur Tastatur

Der KC 85/5 hat zwei Darstellungsmodi: den CAOS-Zeichensatz und den erweiterten IBM-Zeichensatz. Umgeschaltet wird der Darstellungsmodus mit Bit 5 des Steuerbytes STBT (Adresse B7A2H).
Ist der CAOS-Zeichensatz aktiv, dann wiederholen sich die Zeichen (nicht die Funktionen) der Codes 0 bis 127 auf den Codes 128 bis 255, wenn keine anderen Zeichenbildtabellen vereinbart wurden.
Bei aktivem IBM-Zeichensatz werden die vereinbarten Zeichenbildtabellen ignoriert.
Ab CAOS 4.8 gibt es neben dem klassischen Zeichensatz noch einen alternativen Zeichensatz mit schmalen/dünnen Zeichen. Dieser basiert auf einem für den LLC2 erstellten Zeichensatz und wurde von René Nitzsche für CAOS 4.8 ergänzt und aufbereitet. Der Zeichensatz steht als alternativer EPROM-Inhalt zur Verfügung. Die Wahl des EPROMs entscheidet über den Zeichensatz. Beim KC 85/5+ mit 512K-Flash-ROM-Erweiterung kann der Zeichensatz per SWITCH-Befehl des CAOS-ROM C umgeschaltet werden, siehe Seite 70.

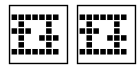
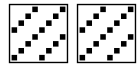
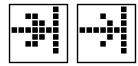
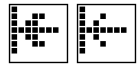
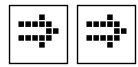
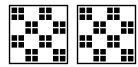
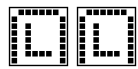
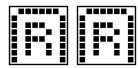
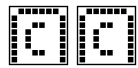
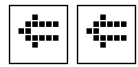
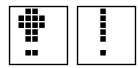
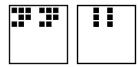
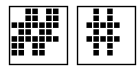
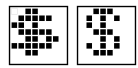
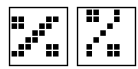
CAOS-Zeichensatz

In der folgenden Tabelle ist der Zeichenvorrat des CAOS-Zeichensatzes in einer Übersicht dargestellt – jeweils für beide alternativen Zeichensätze.

Tabelle 30: CAOS-Zeichensatz

Code Dezimal	Code Hex	Zeichen normal schmal	Funktion
0	00		Dummy-Zeichen (Leerfunktion)
1	01		Backspace CLR (ein Zeichen löschen)
2	02		Zeile löschen
3	03		BREAK
4	04		Shift-BREAK (keine Funktion)
5	05		Tabulatorschritt (gleiche Wirkung wie ESC-0)

Code Dezimal	Hex	Zeichen normal	schmal	Funktion
6	06			nicht benutzt
7	07			BEEP
8	08			Cursor nach links
9	09			Cursor nach rechts
10	0A			Cursor nach unten
11	0B			Cursor nach oben
12	0C			Bildschirm löschen
13	0D			Enter
14	0E			Shift-Enter (nicht benutzt)
15	0F			Aufruf Sonderprogramm (z. B. Drucker)
16	10			Cursor in linke, obere Ecke setzen
17	11			PAGE-Modus
18	12			SCROLL-Modus
19	13			STOP
20	14			Ein- oder Abschalten des Tastenklicks
21	15			nicht benutzt

Code Dezimal	Hex	Zeichen normal	schmal	Funktion
22	16			SHIFT LOCK
23	17			nicht benutzt
24	18			setzt den Cursor an das Ende der Zeile
25	19			setzt den Cursor an den Anfang der Zeile
26	1A			INS (Zeichen einfügen)
27	1B			ESC (danach wird ein Steuerzeichen erwartet)
28	1C			LIST (nur in BASIC)
29	1D			RUN (nur in BASIC)
30	1E			CONT (nur in BASIC)
31	1F			DEL (Zeichen löschen)
32	20			SPC (Leerzeichen)
33	21			REM (nur in BASIC)
34	22			Anführungszeichen
35	23			
36	24			Kennzeichnung von Stringvariablen (nur in BASIC)
37	25			

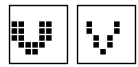
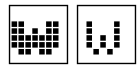
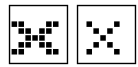
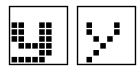
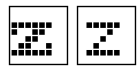
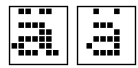
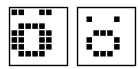
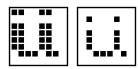
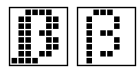
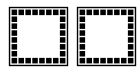
Code Dezimal	Hex	Zeichen normal	schmal	Funktion	
38	26				
39	27				
40	28				
41	29				
42	2A			Multiplikation	(in BASIC und weiteren höheren Programmiersprachen)
43	2B			Addition	(in BASIC und weiteren höheren Programmiersprachen)
44	2C			tabellierte Ausgabe (nur in BASIC)	
45	2D			Subtraktion	(in BASIC und weiteren höheren Programmiersprachen)
46	2E			Dezimalpunkt	(in BASIC und weiteren höheren Programmiersprachen)
47	2F			Division	(in BASIC und weiteren höheren Programmiersprachen)
48	30				
49	31				
50	32				
51	33				
52	34				
53	35				

Code Dezimal	Hex	Zeichen normal	schmal	Funktion
54	36			
55	37			
56	38			
57	39			
58	3A			Trennzeichen zwischen mehreren Anweisungen (nur in BASIC)
59	3B			Ausgabe auf Ausgabe (ohne Zwischenraum) (nur in BASIC)
60	3C			
61	3D			Wertzuweisung (LET) (nur in BASIC)
62	3E			
63	3F			
64	40			
65	41			
66	42			
67	43			
68	44			
69	45			Exponentendarstellung *10^X (in BASIC und weiteren Programmiersprachen)

Code Dezimal	Hex	Zeichen normal	schmal	Funktion
70	46			
71	47			
72	48			
73	49			
74	4A			
75	4B			
76	4C			
77	4D			
78	4E			
79	4F			
80	50			
81	51			
82	52			
83	53			
84	54			
85	55			

Code Dezimal	Hex	Zeichen normal	schmal	Funktion	
86	56				
87	57				
88	58				
89	59				
90	5A				
91	5B			Vollzeichen	
92	5C				
93	5D			Negationszeichen	
94	5E			Exponent	(nur in BASIC)
95	5F				
96	60				
97	61				
98	62				
99	63				
100	64				
101	65				

Code Dezimal	Hex	Zeichen normal	schmal	Funktion
102	66			
103	67			
104	68			
105	69			
106	6A			
107	6B			
108	6C			
109	6D			
110	6E			
111	6F			
112	70			
113	71			
114	72			
115	73			
116	74			
117	75			

Code Dezimal	Hex	Zeichen normal schmal	Funktion
118	76		
119	77		
120	78		
121	79		
122	7A		
123	7B		
124	7C		
125	7D		
126	7E		
127	7F		
241 bis 246	F1 bis F6		Erstbelegung der Funktionstasten F1 bis F6
247 bis 252	F7 bis FC		Zweitbelegung der Funktionstasten F1 bis F6

IBM-Zeichensatz

Ab CAOS 4.3 besitzt der KC noch einen weiteren Zeichensatz, welcher alle Hex-Codes von 0 bis FFh umfasst. Dieser IBM-Zeichensatz ist an den Original-Zeichensatz des IBM-PC ab 1981 (Codepage 437) angelehnt. Im folgenden Bild ist diese Codepage von MS-DOS zum Vergleich zu sehen.

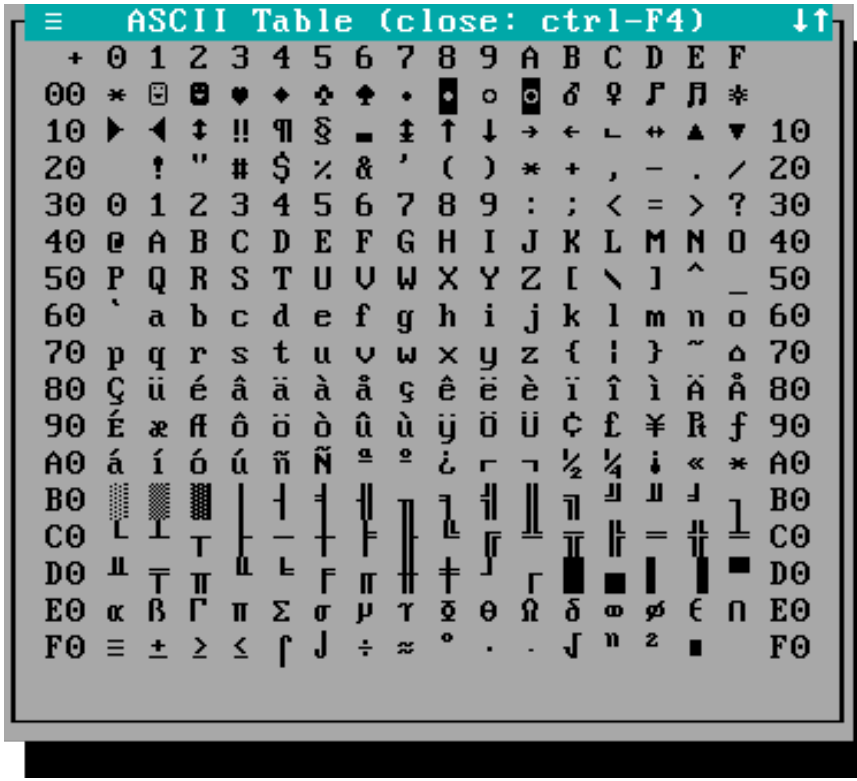


Bild 25: MSDOS-Zeichensatz Codepage 437

Der IBM-Zeichensatz von CAOS hat gegenüber dem normalen CAOS-Zeichensatz einen teilweise anderen Zeichenvorrat, er enthält internationale Sonderzeichen, Grafikzeichen und Symbole. In der folgenden Tabelle ist dieser Zeichenvorrat dargestellt. Nicht abgedruckte Zeichen sind dabei identisch mit den entsprechenden CAOS-Zeichen nach Tabelle 30.

Tabelle 31: IBM-Zeichensatz

Code Dezimal	Code Hex	Zeichen normal	Zeichen schmal
0	00		
1	01		
2	02		
3	03		
4	04		
5	05		
6	06		
7	07		
8	08		
9	09		
10	0A		
11	0B		
12	0C		
13	0D		
14	0E		
15	0F		
16	10		
17	11		
18	12		
19	13		
20	14		
21	15		
22	16		
23	17		
24	18		
25	19		
26	1A		
27	1B		
28	1C		
29	1D		
30	1E		
31	1F		

Code Dezimal	Code Hex	Zeichen normal	Zeichen schmal
128	80		
129	81		
130	82		
131	83		
132	84		
133	85		
134	86		
135	87		
136	88		
137	89		
138	8A		
139	8B		
140	8C		
141	8D		
142	8E		
143	8F		
144	90		
145	91		
146	92		
147	93		
148	94		
149	95		
150	96		
151	97		
152	98		
153	99		
154	9A		
155	9B		
156	9C		
167	9D		
158	9E		
159	9F		

Code Dezimal	Hex	Zeichen normal	schmal
160	A0		
161	A1		
162	A2		
163	A3		
164	A4		
165	A5		
166	A6		
167	A7		
168	A8		
169	A9		
170	AA		
171	AB		
172	AC		
173	AD		
174	AE		
175	AF		

Code Dezimal	Hex	Zeichen normal	schmal
176	B0		
177	B1		
178	B2		
179	B3		
180	B4		
181	B5		
182	B6		
183	B7		
184	B8		
185	B9		
186	BA		
187	BB		
188	BC		
189	BD		
190	BE		
191	BF		

Code Dezimal	Code Hex	Zeichen normal	Zeichen schmal
192	C0		
193	C1		
194	C2		
195	C3		
196	C4		
197	C5		
198	C6		
199	C7		
200	C8		
201	C9		
202	CA		
203	CB		
204	CC		
205	CD		
206	CE		
207	CF		
208	D0		
209	D1		
210	D2		
211	D3		
212	D4		
213	D5		
214	D6		
215	D7		
216	D8		
217	D9		
218	DA		
219	DB		
220	DC		
221	DD		
222	DE		
223	DF		

Code Dezimal	Hex	Zeichen normal	schmal	Code Dezimal	Hex	Zeichen normal	schmal
224	E0			240	F0		
225	E1			241	F1		
226	E2			242	F2		
227	E3			243	F3		
228	E4			244	F4		
229	E5			245	F5		
230	E6			246	F6		
231	E7			247	F7		
232	E8			248	F8		
233	E9			249	F9		
234	EA			250	FA		
235	EB			251	FB		
236	EC			252	FC		
237	ED			253	FD		
238	EE			254	FE		
239	EF			255	FF		

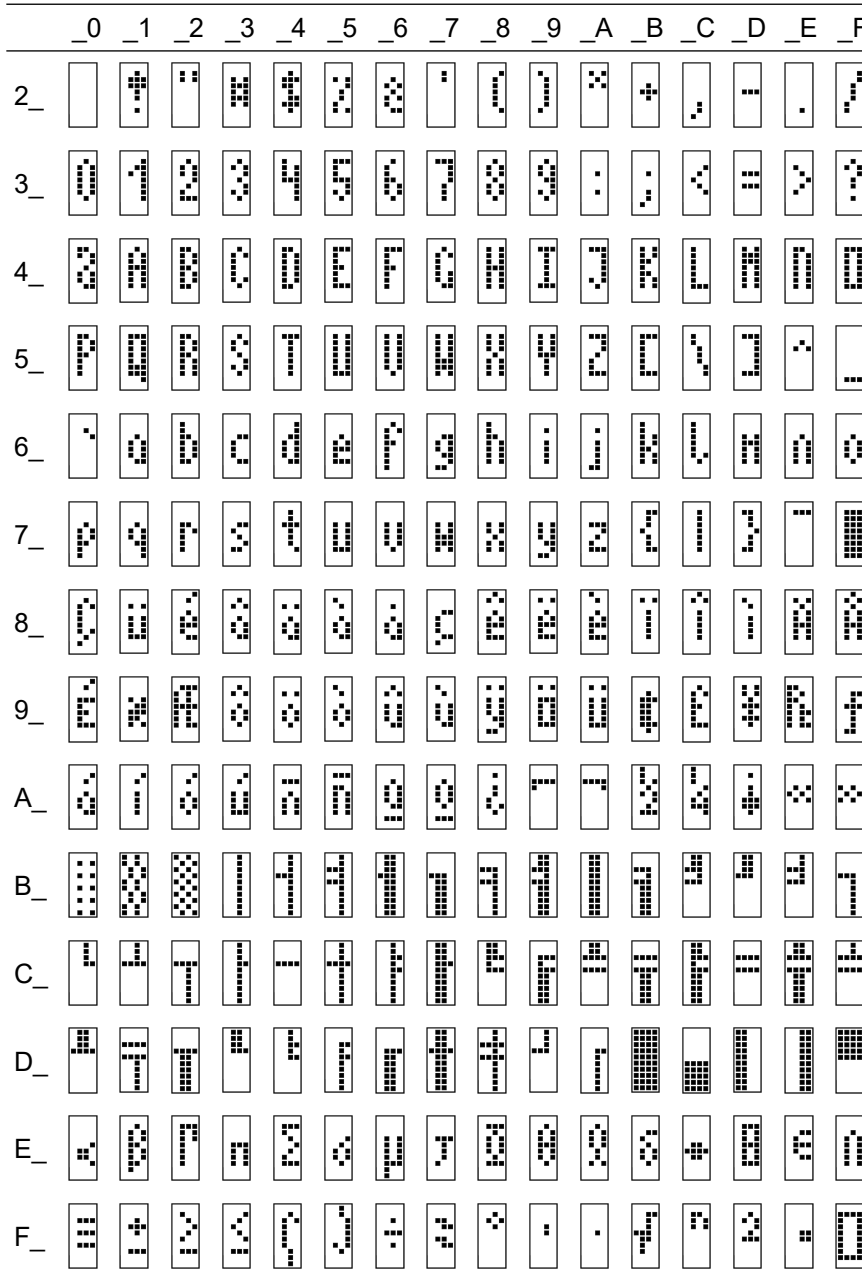
Code Dezimal	Code Hex	Zeichen normal	Zeichen schmal	Code Dezimal	Code Hex	Zeichen normal	Zeichen schmal
91	5B			123	7B		
92	5C			124	7C		
93	5D			125	7D		
				126	7E		
96	60			127	7F		

80-Zeichensatz

Ab CAOS 4.7 besitzt der KC noch einen dritten Zeichensatz, welcher sich in der Editor-Ebene des USER-ROM befindet und ausschließlich vom Editor benutzt wird. Im Gegensatz zum CAOS- bzw. IBM-Zeichensatz, bei denen die Zeichen eine Größe von 8x8 Pixel umfassen, sind die Zeichen beim 80-Zeichensatz nur 4 Pixel breit, dafür aber 10 Pixel hoch. Damit lassen sich auf dem Bildschirm 25 Zeilen zu je 80 Zeichen darstellen. Der 80-Zeichensatz umfasst alle Hex-Codes von 0 bis FFH wie der IBM-Zeichensatz. Dabei sind die Codes von 00H bis 1FH für den internen Gebrauch bestimmt, ab 20H sind die Zeichen angelehnt an den IBM-Zeichensatz und kompatibel zu WordPro6. Für die Umlaute gibt es im 80-Zeichensatz noch Austauschzeichen für die Varianten „CAOS“ und „deutsch“.

In den folgenden 3 Tabellen ist dieser Zeichenvorrat dargestellt.

Tabelle 32: 80-Zeichensatz


_0 _1 _2 _3 _4 _5 _6 _7 _8 _9 _A _B _C _D _E _F
2_
3_
4_
5_
6_
7_
8_
9_
A_
B_
C_
D_
E_
F_

Die internen Zeichen des 80-Zeichensatzes werden z. B. für die Statuszeile, die Druckersteuerzeichen und die Blockmarkierungen genutzt.

02H	linke Randmarke für Lineal der Statuszeile
03H	rechte Randmarke für Lineal der Statuszeile
05H	Glöckchensymbol Statuszeile (Anzeige Tastenklick ein)
06H	Tabulator-Positionen für Lineal der Statuszeile
0EH	Symbol für „Zeile binden“
0FH	Symbol für Einleitung HEX-Sequenz
1CH	Blockanfangsmarke
1DH	Anzeige CAPS-Funktion in Statuszeile (Großbuchstaben)
1EH	Blockendemarke
1FH	Anzeige CAPS-Funktion in Statuszeile (Kleinbuchstaben)

Tabelle 33: Interne Zeichen im 80-Zeichensatz

	_0	_1	_2	_3	_4	_5	_6	_7	_8	_9	_A	_B	_C	_D	_E	_F
0_																
1_																

Die Austauschzeichen für CAOS und Deutsch werden im Texteditor benutzt.

Tabelle 34: Austauschzeichen im 80-Zeichensatz für CAOS und Deutsch

	40	5B	5C	5D	60	7B	7C	7D	7E
IBM- oder ASCII-Zeichensatz									
CAOS-Zeichensatz									
Deutscher Zeichensatz									

3.10.2. Zuordnung Tastennummer zu Tastencode

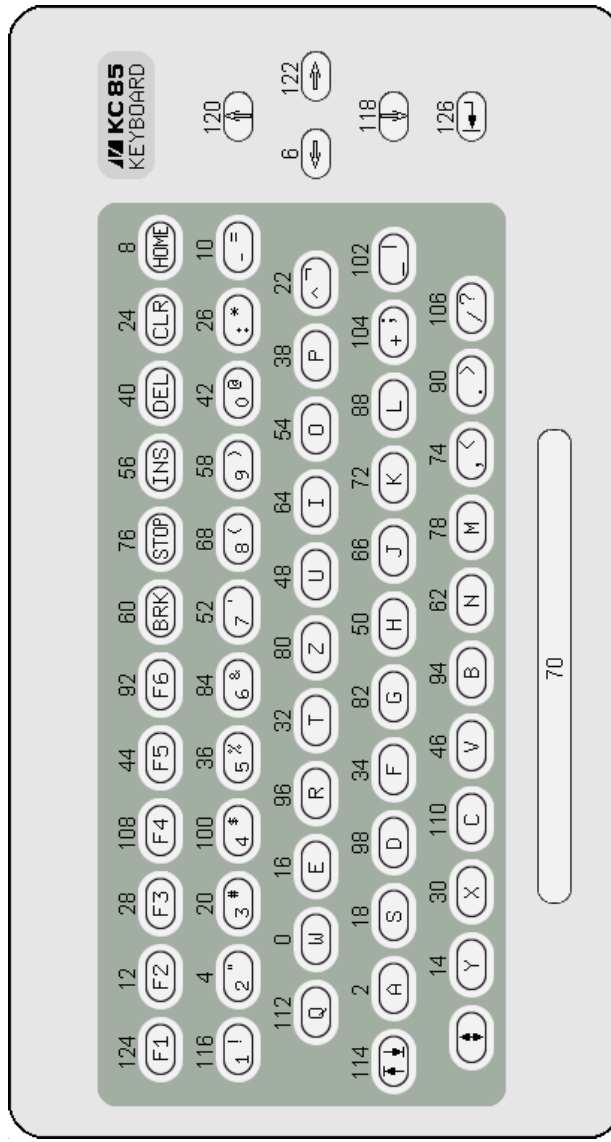


Bild 26: Ansicht der Tastatur und Reihenfolge in der Umcodierungstabelle

Der Tastencode wird über eine Tabelle (KTAB vgl. Kapitel „Arbeitszellen im IRM“, ab Seite 186) aus den seriellen Impulsfolgen des Fernsteuerschaltkreises U807D gewonnen. Eine Änderung der Codes zu den einzelnen Tasten ist durch Aufbau einer neuen Umcodierungstabelle und Eintragen deren Anfangsadresse in KTAB möglich. Diese Tabelle umfasst 128 Byte, wobei jeder Taste zwei Bytes zugeordnet sind.

Erstes Byte	Erstbelegung der Taste
Zweites Byte	Zweitbelegung (bei gleichzeitiger Betätigung von <SHIFT>)

Im Bild 26 auf Seite 221 sind die Ansicht der Tastatur des KC 85/5 und die Reihenfolge der Tasten in der folgenden Umcodierungstabelle dargestellt.

Tabelle 35: Umcodierungstabelle der KC-Tastatur

Erstbelegung ohne <SHIFT>-Taste					**Zweitbelegung mit <SHIFT>-Taste**				
			SHIFT-LOCK					SHIFT-LOCK	
Sende-wort	Bez.	ASCII-Code	Bez.	ASCII-Code	Sende-wort	Bez.	ASCII-Code	Bez.	ASCII-Code
0	W	57	w	77	1	w	77	W	57
2	A	41	a	61	3	a	61	A	41
4	2	32			5	"	22		
6	CUL	08			7	CCR	19		
8	HOME	10			9	CLS	0C		
10	-	2D			11	=	3D		
12	F2	F2			13	F8	F8		
14	Y	59	y	79	15	y	79	Y	59
16	E	45	e	65	17	e	65	E	45
18	S	53	s	73	19	s	73	S	53
20	3	33			21	#	23		
22	^	5E	ß	7E	23	¬	5D	ü	7D
24	CLR	01			25	HCOPY	0F		
26	:	3A			27	*	2A		
28	F3	F3			29	F9	F9		
30	X	58	x	78	31	x	78	X	58
32	T	54	t	74	33	t	74	T	54

Erstbelegung ohne <SHIFT>-Taste					Zweitbelegung mit <SHIFT>-Taste				
			SHIFT-LOCK					SHIFT-LOCK	
Sende-wort	Bez.	ASCII-Code	Bez.	ASCII-Code	Sende-wort	Bez.	ASCII-Code	Bez.	ASCII-Code
34	F	46	f	66	35	f	66	F	46
36	5	35			37	%	35		
38	P	50	p	70	39	p	70	P	50
40	DEL	1F			41	CLLN	02		
42	0	30			43	@	40	©	60
44	F5	F5			45	FB	FB		
46	V	56	v	76	47	v	76	V	56
48	U	55	u	75	49	u	75	U	55
50	H	48	h	68	51	h	68	H	48
52	7	37			53	'	27		
54	O	4F	o	6F	55	o	6F	O	4F
56	INS	1A			57	CLICK	14		
58	9	39			59	)	29		
60	BRK	03			61	Sh-BRK	04		
62	N	4E	n	6E	63	n	6E	N	4E
64	I	49	i	69	65	i	69	I	49
66	J	4A	j	6A	67	j	6A	J	4A
68	8	38			69	(	28		
70	SPC	20			71		5B	ä	7B
72	K	4B	k	6B	73	k	6B	K	4B
74	,	2C			75	<	3C		
76	STOP	13			77	ESC	1B		
78	M	4D	m	6D	79	m	6D	M	4D
80	Z	5A	z	7A	81	z	7A	Z	5A
82	G	47	g	67	83	g	67	G	47
84	6	36			85	&	26		
86	LIST	1C *29			87	RUN	1D *29		

Erstbelegung ohne <SHIFT>-Taste					Zweitbelegung mit <SHIFT>-Taste				
			SHIFT-LOCK					SHIFT-LOCK	
Sende-wort	Bez.	ASCII-Code	Bez.	ASCII-Code	Sende-wort	Bez.	ASCII-Code	Bez.	ASCII-Code
88	L	4C	l	6C	89	l	6C	L	4C
90	.	2E			91	>	3E		
92	F6	F6			93	FC	FC		
94	B	42	b	62	95	b	62	B	42
96	R	52	r	72	97	r	72	R	52
98	D	44	d	64	99	d	64	D	44
100	4	34			101	$	24		
102	_	5F	□	7F	103	|	5C	ö	7C
104	+	2B			105	;	3B		
106	/	2F			107	?	3F		
108	F4	F4			109	FA	FA		
110	C	43	c	63	111	c	63	C	43
112	Q	51	q	71	113	q	71	Q	51
114	CAPS	16			115	TAB	05		
116	1	31			117	!	21		
118	CUD	0A			119	SCROL	12		
120	CUU	0B			121	PAGE	11		
122	CUR	09			123	CEL	18		
124	F1	F1			125	F7	F7		
126	ENTER	0D			127	Shift-ENTER	0E		

Rot: Tastaturcodes modifiziert ab CAOS 4.3

Enthalten die <SHIFT LOCK>-Tasten keine Eintragungen, so entsprechen diese den Eintragungen in den Spalten ohne <SHIFT LOCK>.

*29 Taste 86/87 ist bei der Standard-Tastatur nicht vorhanden, bei der D005-Komfort-Tastatur mit ROM-Version vom 14.01.1995: CTR-L = LIST und CTRL-R = RUN

3.10.3. Steuercodes des KC 85/5

In der Speicherzelle CTAB (siehe Kapitel 3.6.8.) ist ein Zeiger auf eine Programmverteiler-Tabelle abgelegt, die die Zuordnung der Steuercodes zu den einzelnen Bildschirmprogrammfunktionen organisiert. In ihr sind die Anfangsadressen der zugeordneten Unterprogramme enthalten. Sollen Steuerprogramme geändert werden, müssen diese Tabelle in den RAM kopiert und die entsprechenden neuen Anfangsadressen in der Speicherzelle CTAB verändert werden. Die Stelle in der Tabelle errechnet sich aus dem ASCII-Code * 2.
In der folgenden Tabelle sind die Steuercodes des KC 85/5 mit Namen und Funktionen enthalten.

Tabelle 36: Steuercodes des KC 85/5

Code	Name	Funktion (speziell für CRT)
00	DUMMY	Füllzeichen; keine Funktion
01	CLEAR	Löschen eines Zeichens; auf aktueller Position werden ein SPACE eingetragen und der Cursor um eine Position nach links verschoben (nicht in BASIC).
02	CLL	CLEAR A LINE - Löschen einer Zeile; die aktuelle Bildschirmzeile wird mit '00' gefüllt und der Cursor wird an den Anfang dieser Zeile gestellt.
03	BREAK	Programmende; keine Funktion in der CRT-Routine, Abbruch der Zeichenübergabe durch eine F-Taste.
04	-	Shift-Break; keine Funktion
05	ESC0	Tabulator; setzt den Cursor auf die nächste Tabulatorposition (Schrittweite 8), gleiche Funktion wie ESC-0
06	-	nicht belegt
07	BEEP	Signaltonausgabe; Ausgabe eines kurzen Tones, z. B. zur Fehlersignalisierung (Tondauer ist nicht interruptgesteuert).
08	CUL	Cursor Left; Cursor um eine Position innerhalb des Fensters nach links verschieben bis max. auf HOME-Position.
09	CUR	Cursor Right; Cursor um eine Position innerhalb des Fensters nach rechts verschieben, ggf. rollen des Fensters nach oben.
0A	CUD	Cursor Down; Cursor um eine Zeile nach unten verschieben, bei Fensterende ggf. rollen des Fensters.

Code	Name	Funktion (speziell für CRT)
0B	CUU	Cursor Up; Cursor um eine Zeile nach oben bis max. in die Zeile 0 des Fensters verschieben.
0C	CLS	Clear Screen; Löschen des Fensters und eintragen des Codes 00 in das Fenster des Video-RAM.
0D	CR	New line; Funktion wie CCR (Code 19H)
0E	-	Shift-Enter; keine Funktion
0F	HCOPY	Aufruf Sonderprogramm (z. B. Hardcopy), Anfangsadresse des Sonderprogramms auf B799H
10	HOME	Cursor Home; Cursor auf Fensteranfang (Zeile 0, Spalte 0), Fensterinhalt unverändert
11	PAGE	Umschaltung auf PAGE-Modus; Modus bewirkt, dass nach Erreichen des Fensterendes der Cursor bei unverändertem Fensterinhalt auf HOME-Position gestellt wird (In diesem Modus ist im CAOS keine Kommandoeingabe auf der untersten Zeile möglich!).
12	SCROLL	Umschalten auf SCROLL-Modus; Bewirkt, dass nach Erreichen des Fensterendes alle Zeilen des Fensters um eine Zeile nach oben verschoben werden, wobei die oberste Zeile für die Anzeige verloren geht. Als unterste Zeile wird eine mit Code 00H gefüllte Leerzeile eingefügt und der Cursor auf deren Anfang positioniert (dieser Modus entspricht der Grundeinstellung).
13	STOP	keine Funktion in der CRT-Routine.
14	CLICK	Ein- und Ausschalten des Tastenklicks
15	-	nicht belegt
16	CAPS	Dauerumschaltung (SHIFT LOCK) Ein/ Aus
17	-	nicht belegt
18	CEL	setzt den Cursor an das Ende der Zeile
19	CCR	Cursor Carriage Return; Cursor auf den Anfang der aktuellen Zeile setzen, ohne diese zu verändern.

Code	Name	Funktion (speziell für CRT)
1A	INS	Insert; Einfügen eines Leerzeichens (Code 20H) und Rechtsverschieben aller danach stehenden Zeichen innerhalb einer Textzeile (nicht unbedingt identisch mit Bildschirmzeile). Das heißt, es werden so viele Zeichen verschoben, bis der Code 00 erkannt wird, auch über die Bildschirmzeile hinaus. Solange mehr als ein Dummyzeichen vorhanden ist, gehen diese dabei verloren.
1B	ESC	Einschalten der 3. Tastaturebene für das nächste Zeichen.
1C	LIST	In der CRT-Routine nicht benutzt, nur in BASIC.
1D	RUN	
1E	CONT	
1F	DEL	Delete; Löschen des Zeichens auf der Cursorposition und Verdichten des Textes durch Linksverschieben aller Zeichen bis zu einem Dummyzeichen und Einfügen eines weiteren Dummyzeichens (vgl. INS).

3.10.4. ESC-Steuercodes

Die Steuerfunktionen der Tasten von 0 bis 9, und A bis Z können vom Anwender beliebig umbelegt und erweitert werden. Groß- und Kleinbuchstaben werden nicht unterschieden. Die Anzahl der Steuerfunktionen muss in die Arbeitszelle L3SIZ (0B7DFH) eingetragen werden. Die Anfangsadressen der neuen Steuerfunktionen sind dabei in einer Tabelle bereitzustellen, wobei die Anfangsadresse dieser Tabelle in die Zelle L3TAB (0B7DDH und 0B7DEH) eingetragen werden muss. Soll diese Tabelle erweitert werden und vorhandene Funktionen erhalten bleiben, muss sie von L3TAB zuvor aus dem ROM- in den RAM-Bereich kopiert werden.
Beim Erstellen neuer Steuerfunktionen ist zu beachten, dass das Register DE nicht zerstört wird. DE kann aber gezielt verändert werden, da in ihm die neue bzw. alte Cursorposition übergeben wird.

An zwei kurzen Beispielen in BASIC soll die Anwendung der ESC-Funktion gezeigt werden. Im Beispiel 1 wird die Umschaltung zwischen den Bildern 0 und 1 demonstriert.

Beispiel 1:

```
 10 COLOR6,1:CLS
 20 PRINTAT(15,11);CHR$(27);"1";"HIER IST BILD 0!"
 30 PAUSE20
 40 PRINTCHR$(27);"2";:COLOR4,0:CLS
 50 PRINTAT(15,11);"HIER IST BILD 1!";CHR$(27);"2";
 60 PAUSE20
 70 PRINTAT(20,4);"ICH SCHREIBE JETZT AUF BILD 0!";CHR$(27);"4";
 80 COLOR1,5:CLS:FORI=0TO100STEP5:CIRCLE159,127,I,0:NEXT
 90 PRINTAT(1,9);"HIER IST WIEDER BILD 0!";CHR$(27);"1";:PAUSE20
100 PRINTCHR$(27);"2";:COLOR6,1:CLS
110 FORI=0TO2*PISTEP0.03
120 X=159+65*SIN(I*3)
130 Y=127+50*SIN(I*4)
140 PSETX,Y,7:NEXT:PRINTAT(1,9);"HIER IST WIEDER BILD 1!"
150 PRINTCHR$(27);"2";:PAUSE20
160 PRINTCHR$(27);"1";:PAUSE20
170 GOTO150
```

Das Beispiel 2 zeigt verschiedene Möglichkeiten, bei der hohen Farbauflösung (pixelweise), den Bildschirm mit einer der vier Farben, quasi als Hintergrundfarbe, einzufärben.

Beispiel 2:

```
 10 PRINTCHR$(27);"A";
 20 !HINTERGRUND SCHWARZ
 30 COLOR0,0:CLS:GOSUB130:PAUSE20
 40 !HINTERGRUND TUERKIS
 50 COLOR31,7:CLS:GOSUB130:PAUSE20
 60 !HINTERGRUND ROT
 70 COLOR31,7:PRINTCHR$(27);"9";CHR$(12);CHR$(27);"9";:GOSUB130
 80 PAUSE20
 90 !HINTERGRUND WEISZ
100 COLOR 31,7:CLS:PRINT CHR$ (27);"9";:VPOKE14242,
    VPEEK(14242) OR1:CLS
110 VPOKE14242,VPEEK(14242) AND 254:PRINT CHR$(27);"9";:
    GOSUB130:PAUSE20
120 GOTO20
130 CIRCLE129,97,50,0:CIRCLE189,97,50,1
140 CIRCLE129,157,50,2:CIRCLE189,157,50,3:RETURN
```

3.11. Bildschirmausgaben, Zeichen, Pseudozeichen, Grafik

3.11.1. Zeichenbildtabellen und deren Verwaltung

Zur Ergänzung des internen Zeichenbildvorrates und der Groß- und Kleinbuchstaben, Ziffern, Sonderzeichen (Codes 00 - 7FH) können eigene Zeichenbildtabellen erstellt werden. Pro Zeichen werden 8 Byte benötigt.

Bildpunkte = Bits: seitenrichtig, nicht negiert, oberste Bildpunktzeile = niedrigste Adresse. Die Anfangsadressen der Tabellenzeiger müssen entsprechend den zugehörigen Codes in die Speicherzellen CCTL0 bis CCTL3 eingetragen werden. Für die Codes 20H bis 5FH und A0H bis DFH (CCTL0 und CCTL2) ist die Zeichenbildtabelle für Großbuchstaben und Ziffern eingetragen.
Die Codes 00H - 1FH dienen im Normalfall als ausführbare Steuerzeichen (vgl. Arbeitszelle STBT). Wird Bit 3 (STBT) gesetzt, werden die Zeichen aus CCTL1 zum Bildschirm gesendet. Dies sind Symbole für die Steuercodes bzw. spezielle Grafiksymbole. Die Codes 60H bis 7FH und E0H bis FFH erzeugen die Kleinbuchstaben.
Die Zeichenbildtabellen CCTL1 und CCTL3 sind nach dem Einschalten und nach jedem RESET auf die Adresse 0FE00H und die Zeichenbildtabellen CCTL0 und CCTL2 auf die Adresse 0EE00H gesetzt.

ACHTUNG! Ist der IBM-Zeichensatz aktiv – entweder durch Setzen von Bit 4 der Speicherzelle STBT oder durch Verwendung der Funktion ESC-C, dann werden die in CCTL0 bis CCTL3 eingestellten Zeichenbildtabellen ignoriert und der im ROM hinterlegte Zeichensatz direkt genutzt.

3.11.2. Erweiterung des Zeichenvorrates

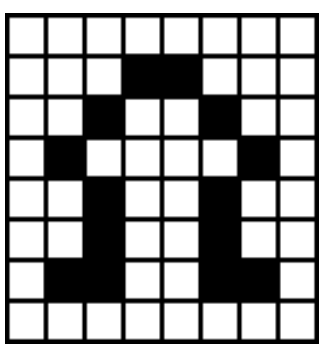
Anhand eines Beispieles soll die Erweiterung des Zeichenvorrates erläutert werden.

Für die Zeichen mit den Codes 0A0H bis 0DFH, die in der Zeichenbildtabelle CCTL2 liegen, sollen spezielle Zeichen definiert werden. Die neue Zeichenbildtabelle wird im Speicherbereich 0BC00 bis 0BDFFH abgelegt.

1. Umschalten des Zeigers auf die neue Zeichenbildtabelle.

   MODIFY B7AA
   B7AA 00
   B7AB EE - ändern in BC

2. Generieren eines neuen Zeichens mit dem Code 0A0H



Zeichenbild	Bild-Code	Hex-Code	
	0000 0000	00	
	0001 1000	18	
	0010 0100	24	
	0100 0010	42	8 Byte
	0010 0100	24	
	0010 0100	24	
	0110 0110	66	
	0000 0000	00	

Wenn der Hex-Code ab Adresse 0BC00H mit MODIFY abgelegt ist, wird das Zeichen 0A0H mit diesem Bild so auf dem Bildschirm dargestellt. Vom BASIC aus kann dieses Zeichen nun über die Anweisung PRINT CHR$(160) zur Anzeige gebracht werden (0A0H = 160 im Dezimalzahlensystem).

3.11.3. Adresszuordnung im IRM (Grafik- und Video-RAM)

Mithilfe der folgenden Formeln kann man die Speicherzellen, die die Informationen zur Darstellung eines beliebigen Bildpunktes enthalten, ermitteln. Die Bildinformationen sind im IRM nach folgendem Prinzip abgelegt:
Je 8 horizontal nebeneinander liegende Bildpunkte sind im Pixel-RAM als ein Byte abgespeichert. Dieses Byte enthält nur die Vordergrund-Hintergrund-Information der Bildpunkte. Die Farbinformation ist für jeweils eine Reihe von 8 Bildpunkten zu einem Byte im COLOR-RAM zusammengefasst. Dieses Byte legt also für 8 Bildpunkte eine Vorder- und eine Hintergrundfarbe fest.

Bei hoher Farbauflösung (siehe ESCape-Funktionen, Tabelle 3 auf Seite 34) wird auch das Pixelbyte für die Farbinformation verwendet. Hier gibt es keine Vorder- und Hintergrundfarben. In diesem Modus sind also nur vier Farben möglich, wobei eine Farbe als Hintergrundfarbe des Bildschirmes verwendet werden kann. Nun stehen noch drei Farben für Grafiken zur Verfügung.

Darüber hinaus enthält der IRM zwei Video-RAM-Bereiche für Bild 0 und Bild 1, die auch als ASCII-Puffer bezeichnet werden. Sie speichern die Codes der auf dem Bildschirm dargestellten Zeichen ab.

Um das Farb- und das Pixelbyte eines Bildpunktes zu bestimmen, werden die Pixelzeilennummer und die Zeichenspaltennummer, in der sich der Punkt befindet, hexadezimal verwendet. Mit der folgenden Formel kann man die Pixelbyte- bzw. Farbbyteadresse errechnen:

Adresse = 8000H + Zeichenspalte * 100H + Pixelzeile
0 ≤ Zeichenspalte ≤ 27H
0 ≤ Pixelzeile ≤ 0FFH

Der Farb- und der Pixelspeicher befinden sich im gleichen Adressbereich. Will man direkt auf den Farbspeicher zugreifen, muss die Farbebene erst zugeschaltet werden (ESC '9').

Hinweis:
Die Adresszuordnung im IRM des KC 85/4 und KC 85/5 ist gegenüber den Vorgängertypen (KC 85/2, KC 85/3) verändert worden. Aus diesem Grund kann es bei Programmen der Vorgängertypen, wenn sie am KC 85/5 abgearbeitet werden, zu fehlerhafter Bilddarstellung kommen, sofern die Programme unmittelbar Informationen in den Pixel- oder Farbspeicher einschreiben. Programme, die zwischen den verschiedenen KC-Typen austauschbar sein sollen, müssen deshalb zur Bildschirmausgabe konsequent die entsprechenden Unterprogramme des Betriebssystems nutzen (z. B. UP-Nr.: 00H, 23H, 30H, 31H, 34H). Ein unmittelbares Beschreiben der „sichtbaren“ IRM-Bereiche mit einer im Anwenderprogramm enthaltenen Adressrechnung ist nur bei typspezifischen Anwenderprogrammen möglich.

Die Adresse im Video-RAM lässt sich durch folgende Berechnung ermitteln:

Adresse im Video-RAM (ASCII-Puffer) für Bild 0:
= B200H + Zeichenspalte + 40 * Zeichenzeile
= B200H + Zeichenspalte + 5 * Pixelzeile

Adresse im Video-RAM (ASCII-Puffer) für Bild 1:
= AD00H + Zeichenspalte + 40 * Zeichenzeile
= AD00H + Zeichenspalte + 5 * Pixelzeile

3.11.4. Von der Cursor- zur Pixelposition

Die Beziehung zwischen Zeichen- und Pixelposition für Vollgrafik (jeder Punkt auf dem Bildschirm ist ansprechbar) ist wie folgt:

1. Horizontal (X-Wert)
   X = 8 * Zeichenspalte + Position im Byte

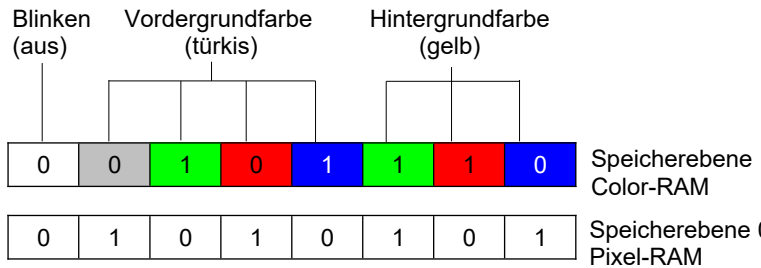
2. Vertikal (Y-Wert)
   Y = 255 - Punktzeile = 255 - 8 * Zeichenzeile - Position im Zeichen

3.11.5. Bit- und Bytemodus der Farbauflösung

Im KC 85/4 und KC 85/5 sind jedem Bild zwei Speicherebenen (Pixel- und Color-RAM) zugeordnet.
Die Bit-Informationen in den Ebenen werden vom Videointerface (VIF) verarbeitet und auf dem Bildschirm dargestellt. Das VIF ist von Bild 0 auf Bild 1 und umgekehrt über Software umschaltbar. Es kann in 2 verschiedenen Modi arbeiten:

1. Byteweise Farbauflösung (Bytemodus = LORES)

Die Vorder- und Hintergrundfarbinformationen gelten wie beim KC 85/3, jedoch nur für 1 * 8 Bildpunkte horizontal in einer Linie ist ein Farbbyte reserviert. Mit diesem Farbbyte können 16 Vorder- und 8 Hintergrundfarben sowie Blinken für die Vordergrundfarbe eingestellt werden.
Das folgende Bild soll den Zusammenhang zwischen den 2 Speicherebenen im Bytemodus verdeutlichen.



Damit ergibt sich türkis/hell als Vordergrundfarbe und gelb/dunkel als Hintergrundfarbe. Und so liegen die Bildpunkte tatsächlich mit der sichtbaren Farbe auf dem Bildschirm:

Bild 27: Beispiel zur Darstellung des Bytemodus der Farbauflösung

Aus diesem Bild wird ersichtlich, dass in 8 Bildpunkten jeweils nur 2 Farben darstellbar sind. Die Farben werden in der Speicherebene 1 (Color-RAM) eingestellt. Ob es Vorder- oder Hintergrundfarbe ist, wird in der Speicherebene 0 (Pixel-RAM) festgelegt. Dabei erfolgt die Zuordnung 0 = Hintergrundfarbe und 1 = Vordergrundfarbe.

2. Pixelweise Farbauflösung (Bitmodus = HIRES)

Ein Bit aus jeder Ebene (Color- und Pixel-RAM) wird als Farbinformation eines Bildpunktes gewertet. Mit den zwei zur Verfügung stehenden Bits können vier

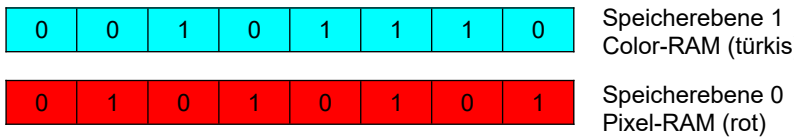
Farben dargestellt werden. Dem Pixelspeicher ist dabei die Farbe Rot und dem Farbspeicher die Farbe Türkis zugewiesen.

Es ergeben sich folgende Farben:

Farbziffer	Farbe	Bit im Farb-RAM	Bit im Pixel-RAM
0	schwarz	0	0
1	rot	0	1
2	türkis	1	0
3	weiß	1	1

Farbtafel: Farben für die pixelweise Farbauflösung

Das folgende Bild zeigt die bitweise Farbauflösung.



Die tatsächliche Bildpunktfarbe auf dem Bildschirm ergibt sich aus den im Beispiel aufgeführten Bitkombinationen. Und so liegen die Bildpunkte hier tatsächlich mit der sichtbaren Farbe auf dem Bildschirm:

Bild 28: Beispiel zur Darstellung des Bitmodus der Farbauflösung

Ein kleines BASIC-Programm zeichnet horizontale und vertikale Linien, die den Modus ein- und ausschalten.

```
10 WINDOW 0,31,0,39: COLOR 0,0: CLS
20 PRINT CHR$ (27); "A";
30 FOR Y = 10 TO 250 STEP 10
40 LINE 0,Y,319,Y,1:NEXT
50 FOR X = 10 TO 310 STEP 10
60 LINE X,0,X,255,2: NEXT
70 PAUSE 10: GOTO 10
```

In diesem Programm wird in der 1. Zeile der Bildschirm schwarz gelöscht. Zeile 20 schaltet die ESC-Steuerfunktion 'A' ein. Damit ist der Bitmodus eingestellt. In den Zeilen 30 bis 60 werden Linien gezeichnet und nach einer Pause geht das Programm zum Anfang zurück und nun wird in Zeile 20 der Bytemodus eingestellt.

3.11.6. Verwendung der zwei Bilder

Die Hardware des KC 85/4 enthält zwei voneinander unabhängige Bildspeicher für Pixel und Farbe. Über zwei Bit des Ausgabeport 84H wird festgelegt, auf welches der beiden Bildspeicher die CPU zugreift und welches der beiden Bilder angezeigt wird, siehe Seite 127.

Das Betriebssystem CAOS unterstützt die Umschaltung der Bilder durch die ESC-Funktionen 1 bis 4, siehe Tabelle 3 auf Seite 34. Umgeschaltet wird mit diesen ESC-Funktionen aber nicht nur die Zuordnung der beiden Steuerbits. Jedem Bild wird auch der entsprechende ASCII-Speicher (Video-RAM) auf den Adressen AD00H bzw. B200H zugeordnet.

Ab CAOS 4.8 werden für jedes der beiden Bilder noch weitere Parameter separat gespeichert. Dazu werden die Parameter des gerade nicht im Zugriff befindlichen Bildes in einem Schattenspeicher im IRM abgelegt und beim Bildwechsel mit dem aktiven Bereich ausgetauscht.

Bildparameter		IRM-Adresse	Schattenspeicher
Nummer des Fensters	WINNR	B79B	AADD
Fensteranfang	WINON	B79C	AADE
Fenstergröße	WINLG	B79E	AAE0
Cursorposition	CURSO	B7A0	AAE2
Steuerbyte	STBT	B7A2	AAE4
Farbe	COLOR	B7A3	AAE5
Page- oder Scrollmode	WEND	B7A4	AAE6

Dieser „2-Monitor-Modus“ kann unter bestimmten Umständen zu Darstellungsproblemen bei älteren Anwendungsprogrammen führen, wenn diese die beiden Bilder nutzen, aber das neue Verhalten nicht kennen. Deshalb kann mit der Funktion ESC-G temporär das alte Verhalten zu- und auch wieder abgeschaltet werden. Der Grundzustand mit aktiviertem „2-Monitor-Modus“ wird bei POWER-ON oder RESET wiederhergestellt.

Gespeichert wird der Zustand des „2-Monitor-Modus“ durch Setzen des Bit 7 des Steuerbytes STBT auf Adresse B7A2H. Software, welche das Steuerbyte auf einen bestimmten Wert setzt, beeinflusst damit auch den Status des „2-Monitor-Modus“.

3.12. V.24-Software

Das Betriebssystem KC-CAOS enthält eine universelle Druckertreiber- und Koppelroutine. Mit ihr lassen sich alle Druckgeräte mit V.24-Schnittstelle bedienen. Dazu ist ein Modul M003 V.24 bzw. M053 RS232 im KC-System erforderlich. Die Anschlussbedingungen und die Bedienungsanleitung für die Module sind aus den Modul-Beschreibungen zu entnehmen. Das M003 und das M053 unterscheidet sich nur im Pegel der Schnittstelle, da aus Sicht der Software die Module identisch sind, wird im weiteren Text stellvertretend nur noch vom V.24-Modul gesprochen.

3.12.1. Systeminitialisierung

Nach dem Einschalten des KC 85/5 (Kaltstart) oder bei RESET (Warmstart) wird im System nach einem V.24-Modul gesucht und dessen Steckplatz auf der Adresse A800H im IRM abgelegt. Befindet sich ein V.24-Modul im System, wird es aktiviert und initialisiert. Dabei werden jeweils der Kanal 1 als Drucker- und der Kanal 2 als interruptgesteuerte Duplexroutine initialisiert. Steckt kein V.24-Modul im KC-System, dann wird die Druckerausgabe nicht initialisiert.

Bei einem vorhandenem V.24-Modul werden außerdem zwei Initialisierungstabellen für die Übertragungsbedingungen von Druckertreiber und Duplexroutine im RAM (IRM) abgelegt. Die Anfangsadresse und die Längen der Tabellen sind in den Arbeitszellen INTV1 und INTV1L (Kanal 1 = Drucker) bzw. INTV2 und INTV2L (Kanal 2 = Duplex) eingetragen. Die Initialisierungstabellen und deren Längen können jederzeit geändert bzw. die Zeiger können auf andere Tabellen umgestellt werden. Dabei ist zu beachten, dass die ersten zwei Byte der Tabellen immer der CTC-Initialisierung dienen.

Der USER-Ausgabekanal 2 (z. B. in BASIC PRINT#2) wird bei vorhandenem V.24-Modul auf Druckerausgabe eingestellt. Bei eigenen Treiberprogrammen muss beachtet werden, dass nach jedem Warmstart (RESET) des Systems die Sprungadresse in UOUT1 wieder auf die interne Druckerroutine gestellt wird.
Der USER-Ausgabekanal 3 (z. B. in BASIC PRINT#3) wird auf die Duplexausgaberoutine eingestellt.

Die USER-Eingabekanäle 2 und 3 sind nach Kalt- bzw. Warmstart nicht initialisiert. Sie werden erst nach Aufruf des Systemunterprogramms V24DUP initialisiert. Damit wird aber auch ein eventuell benutzter V.24-Interupt deaktiviert.

3.12.2. Duplexroutine (mit Empfangsinterrupt)

Wie oben bereits beschrieben, wird der Kanal 2 eines vorhandenen V.24-Moduls beim Kalt- oder Warmstart eingeschaltet und auf Duplex initialisiert. Dabei ist die Empfangsroutine interruptgesteuert. Wird also ein Zeichen von außen an diesen V.24-Kanal gesendet, wird ein Interrupt ausgelöst und das Zeichen ausgewertet. Reagiert wird in der Interruptroutine prinzipiell nur auf zwei Zeichen bzw. ASCII-Codes, und zwar sind das 0DH (ENTER) und 1BH (ESC). Mit einem 0DH kann dem KC mitgeteilt werden, dass mit einer anderen Tastatur gearbeitet werden soll (über V.24). Nach einem 0DH-Empfang wird eine neue Interruptempfangsroutine (für Tastatur) initialisiert. Die Zeichen, die danach über diese Schnittstelle empfangen werden, sind wie bei der normalen Tastaturroutine in der Zelle (IX+13) abgelegt. Die V.24-Interuptroutinen schalten bei Bedarf das erste V.24-Modul ein. So kann ein weiteres V.24-Modul zum Senden/Empfangen verwendet werden, indem das erste V.24-Modul ausgeschaltet wird. Das betrifft vor allem die Anwendung als externer Tastatur-Eingang. CAOS merkt sich dazu den Steckplatz des verwendeten Moduls auf der Adresse A800H.

Bei Empfang von ESC (1BH) wird vom Interruptmodus in den Pollingmodus übergegangen, wobei das z. B. laufende Programm unterbrochen wird. Nach dem Senden von ESC muss vom Sender eine kurze Sendepause eingeschoben werden, da sonst eventuell bereits empfangene Zeichen durch die Uminitialisierung verloren gehen. In BASIC kann das mit Pause 1 erfolgen. Weiterhin ist in BASIC darauf zu achten, dass alle PRINT-Anweisungen mit einem ';' abgeschlossen sein müssen, da sonst nach jedem PRINT zusätzlich ein 0DH und 0AH gesendet wird. Mit der Anweisung NULL 0 muss die Ausgabe von Dummy-Zeichen abgeschaltet werden.

Das Zeichen nach ESC wird als Steuerzeichen interpretiert. Zulässig sind 'T' und 'U'. Bei allen anderen Codes wird wieder in den Interruptmodus übergegangen.

Mit ESC 'T' kann direkt in den Speicher geschrieben werden. Dazu sind nach dem 'T' die Anfangsadresse aaaa, die Anzahl der zu schreibenden Bytes nnnn und die nnnn Bytes selbst an den KC zu senden:

```
ESC 'T'   aa aa   nn  nn  (nnnn * Bytes)
1BH 54H low high low high …
```

Um z. B. den Pixel-RAM zu beschreiben, ist folgende Codefolge zu senden:

```
1B 54 00 80 00 28 … (Pixelbytes) …
```

Mit ESC 'U' können Programme im KC gestartet werden. Nach 'U' ist die Startadresse ssss zu senden:

```
ESC 'U'   ss ss
1BH 55H low high
```

Es könnte z. B. ein Programm gestartet werden, das vorher mit ESC 'T' gesendet wurde. Die gestarteten Programme können mit RETURN (RET) zum unterbrochenen Programm zurückkehren.
Sowohl bei ESC 'T' als auch bei ESC 'U' wird auf folgende Speicherebenen zugegriffen:

0000H … 03FFH	RAM0
4000H … 07FFH	aktive RAM4-Ebene bzw. Modulebene falls RAM4=aus
8000H … BFFFH	IRM falls dieser eingeschaltet ist oder die erste sichtbare RAM-Ebene falls des IRM ausgeschaltet ist
C000H … DFFFH	sichtbares RAM-Modul in diesem Speicherbereich
E000H … FFFFH	CAOS-ROM – Schreiben nicht möglich!

Beispiel:

Der Pixel-RAM eines KC 85/5 soll an einen anderen KC 85/5 gesendet werden. Dazu ist folgendes Sendeprogramm möglich:

```
10 NULL 0
20 PRINT#3 CHR$ (27);: PAUSE 1
30 PRINT#3 "T";CHR$ (0); CHR$ (128);
40 PRINT#3 CHR$ (0); CHR$ (40);
50 FOR I = 0 TO 40 * 256 -1
60 PRINT#3 CHR$ (VPEEK (I));
70 NEXT
```

Die Duplexroutine enthält auch eine Senderoutine. Der USER-Ausgabekanal 3 wird nach jedem Kalt- bzw. Warmstart auf diese Senderoutine initialisiert.

Die Übertragungsbedingungen für Senden und Empfangen sind:

Übertragungsgeschwindigkeit	: 1.200 Baud
Bits pro Zeichen	: 8
Stoppbits	: 1
Paritätsprüfung	: keine

Die Übertragungsbedingungen der interruptgesteuerten Duplexroutine können vom Anwender nicht geändert werden.
Mit der interruptgesteuerten Duplexroutine ist es z. B. möglich, eine Schreibmaschine (z. B. S3004) zur Eingabe am Computer und als Druckgerät zu verwenden. Dazu muss aber die Schreibmaschine mit den gleichen Übertragungsbedingungen senden und empfangen wie das KC-System.

Werden andere Sendebedingungen gewünscht, muss mit der anderen Duplexroutine im Pollingbetrieb ohne Interrupt gearbeitet werden, siehe Kommando V24DUP auf Seite 84.

Im folgenden Bild sind die verschiedenen Duplexroutinen von CAOS dargestellt und dabei angegeben, unter welchen Bedingungen zwischen diesen Routinen gewechselt wird.

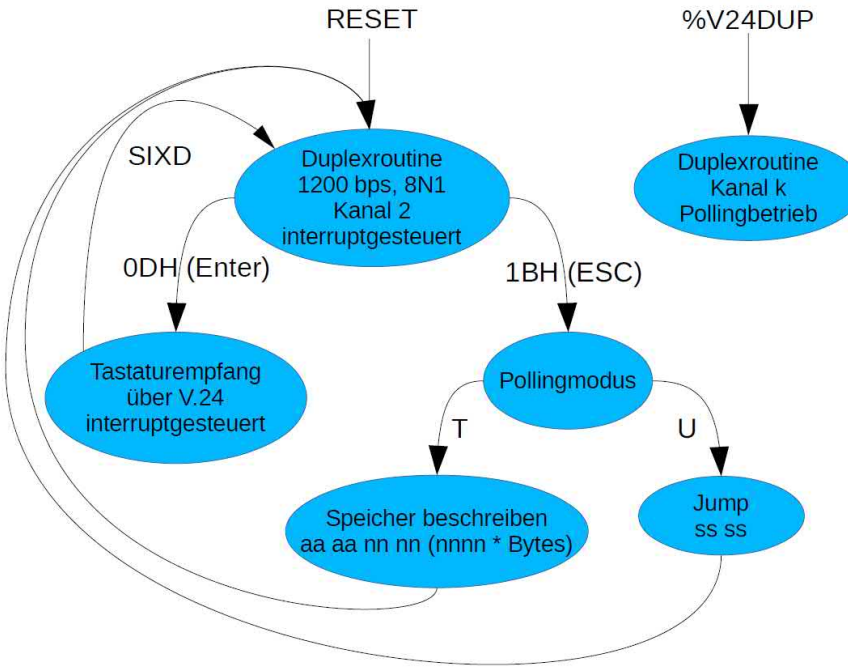


Bild 29: Duplexroutinen von CAOS

Das CAOS-Systemunterprogramm 31H (SIXD) setzt einen V.24-Tastaturinterrupt auf die interruptgesteuerte Duplexroutine zurück. Das heißt, nach der Ausführung dieses Unterprogramms muss bei einer am V.24-Modul angeschlossenen Tastatur erneut die Entertaste betätigt werden, bevor wieder Tastatureingaben möglich sind.

! Wurde der Kanal 2 des V.24-Moduls mit LSTOUT (V24OUT) oder V24DUP uminitialisiert, kann der interruptgesteuerte Duplexmodus für den Kanal 2 nur über einen Systemneustart erreicht werden (z. B. mit RESET).

3.12.3. Übertragungsbedingungen für Druckerbetrieb

Für die Druckerausgabe wird der Kanal 1 des V.24-Moduls auf Standardwerte eingestellt, welche für Matrixdrucker K6313 u. a. gültig sind. Mit der für die Druckerinitialisierung zuständigen Tabelle werden Übertragungsbedingungen mit folgenden Standardwerten festgelegt (blaue Werte in Tabelle 38):

Übertragungsrate : 9.600 Baud
Stoppbits : 1
Bits pro Zeichen : 8
Paritätsprüfung : keine

Werden andere Parameter gewünscht, sind die entsprechenden Bytes in der Initialisierungstabelle zu ändern. In der Tabelle 38 sind für verschiedene Übertragungsbedingungen jeweils die 8 Initialisierungsbytes aufgeführt.

Die Anfangsadresse der Initialisierungstabelle für Kanal 1 (Drucker) ist in der Arbeitszelle INTV1 und die Länge der Tabelle in INTV1L eingetragen (vgl. Kapitel 3.6.8. ab Seite 186).

Die Initialisierungstabellen und deren Längen können jederzeit geändert bzw. die Zeiger können auf andere Tabellen umgestellt werden. Dabei ist zu beachten, dass die ersten zwei Byte der Tabellen immer der CTC-Initialisierung dienen.

Ist z. B. eine Übertragungsrate von 1.200 Baud, einem Stoppbit und 8 Bits pro Zeichen gewünscht, ist entsprechend der Tabelle 38 nur das erste Byte zu ändern (47 in 07). Das kann z. B. mit dem Systemprogramm MODIFY gemacht werden.

Die Initialisierungstabelle INTV1 hat bei den Standardwerten folgenden Aufbau:

Tabelle 37: Beschreibung der V.24-Initialisierungstabelle für Druckerbetrieb

Adresse	Inhalt	Bedeutung
A801H	47H	CTC: Steuerwort hier: Zählermode, Zeitkonstante folgt
A802H	5BH	CTC: Zeitkonstante = 91
A803H	04H	SIO: Auswahl Schreibregister WR4
A804H	04H	SIO: Daten für WR4 (Empfänger/Sender-Steuerung) hier: Vorteiler=1, ein Stopp-Bit, keine Parität
A805H	03H	SIO: Auswahl Schreibregister WR3
A806H	20H	SIO: Daten für WR3 (Empfänger-Steuerung) hier: Empfänger aus, CTS+DCD-Steuerung
A807H	05H	SIO: Auswahl Schreibregister WR5
A808H	6AH	SIO: Daten für WR5 (Sender-Steuerung) hier: Sender ein, 8 Bit, /DTR=1, RTS=1

Tabelle 38: Initialisierungstabelle für Druckerausgabe über V.24-Modul

Übertragungs-rate in Bit/s	Anzahl Stoppbits	Bit/Zei-chen	Initialisierungsbytes (hex) CTC:		WR4:		WR3:		WR5:	
9.600	1	7	47	5B	04	04	03	20	05	2A
		8	47	5B	04	04	03	20	05	6A
	2	7	47	5B	04	0C	03	20	05	2A
		8	47	5B	04	0C	03	20	05	6A
4.800	1	7	47	B6	04	04	03	20	05	2A
		8	47	B6	04	04	03	20	05	6A
	2	7	47	B6	04	0C	03	20	05	2A
		8	47	B6	04	0C	03	20	05	6A
2.400	1	7	07	2E	04	04	03	20	05	2A
		8	07	2E	04	04	03	20	05	6A
	2	7	07	2E	04	0C	03	20	05	2A
		8	07	2E	04	0C	03	20	05	6A
1.200	1	7	07	5B	04	04	03	20	05	2A
		8	07	5B	04	04	03	20	05	6A
	2	7	07	5B	04	0C	03	20	05	2A
		8	07	5B	04	0C	03	20	05	6A
600	1	7	07	B7	04	04	03	20	05	2A
		8	07	B7	04	04	03	20	05	6A
	2	7	07	B7	04	0C	03	20	05	2A
		8	07	B7	04	0C	03	20	05	6A
300	1	7	47	5B	04	84	03	20	05	2A
		8	47	5B	04	84	03	20	05	6A
	2	7	47	5B	04	8C	03	20	05	2A
		8	47	5B	04	8C	03	20	05	6A
150	1	7	47	5B	04	C4	03	20	05	2A
		8	47	5B	04	C4	03	20	05	6A
	2	7	47	5B	04	CC	03	20	05	2A
		8	47	5B	04	CC	03	20	05	6A
110	1	7	47	7C	04	C4	03	20	05	2A
		8	47	7C	04	C4	03	20	05	6A
	2	7	47	7C	04	CC	03	20	05	2A
		8	47	7C	04	CC	03	20	05	6A

3.12.4. Übertragungsbedingungen für Duplexbetrieb

Bei der Systeminitialisierung werden die Übertragungsbedingungen der Duplexroutine mit diesen Standardwerten festgelegt:

Übertragungsrate : 1.200 Baud
Stoppbits : 1
Bits pro Zeichen : 8
Paritätsprüfung : keine

Auch für die Duplexroutine kann die Initialisierungstabelle geändert werden. Die Anfangsadresse dieser Initialisierungstabelle für Kanal 2 steht in der Arbeitszelle INTV2 und deren Länge in der Zelle INTV2L (vgl. Kapitel 3.6.8. ab Seite 186). In Tabelle 40 sind Initialisierungstabellen für verschiedene Übertragungsbedingungen für den Duplexbetrieb aufgeführt (die blauen Werte zeigen wieder den Einschaltzustand).

Werden andere Parameter gewünscht, sind wieder die entsprechenden Bytes in der Initialisierungstabelle zu ändern. Auch in dieser Tabelle dienen die ersten zwei Byte wieder der CTC-Initialisierung.

Die Initialisierungstabelle INTV2 hat bei den Standardwerten folgenden Aufbau:

Tabelle 39: Beschreibung der V.24-Initialisierungstabelle für Duplexbetrieb

Adresse	Inhalt	Bedeutung
A809H	47H	CTC: Steuerwort hier: Zählermode, Zeitkonstante folgt
A80AH	2EH	CTC: Zeitkonstante = 46
A80BH	18H	SIO: Rücksetzen Port
A80CH	04H	SIO: Auswahl Schreibregister WR4
A80DH	44H	SIO: Daten für WR4 (Empfänger/Sender-Steuerung) hier: Vorteiler=16, ein Stopp-Bit, keine Parität
A80EH	03H	SIO: Auswahl Schreibregister WR3
A80FH	E1H	SIO: Daten für WR3 (Empfänger-Steuerung) hier: Empfänger ein, 8 Bit, CTS+DCD-Steuerung
A810H	05H	SIO: Auswahl Schreibregister WR5
A811H	6AH	SIO: Daten für WR5 (Sender-Steuerung) hier: Sender ein, 8 Bit, /DTR=1, RTS=1

Die genaue Bedeutung der einzelnen Steuerworte kann zum Beispiel in [15] oder [21] bei der CTC- bzw. SIO-Programmierung nachgelesen werden.

Tabelle 40: Initialisierungstabelle für Duplexbetrieb V.24-Modul

Übertragungs-rate in Bit/s	Anzahl Stoppbits	Bit/Zei-chen	Initialisierungsbytes (hex) CTC:			WR4:		WR3:		WR5:	
54.748 *30	1	7	47	01	18	04	44	03	61	05	2A
		8	47	01	18	04	44	03	E1	05	6A
	2	7	47	01	18	04	4C	03	61	05	2A
		8	47	01	18	04	4C	03	E1	05	6A
4.800 -	1	7	47	**	18	04	44	03	61	05	2A
56.000 *31		8	47	**	18	04	44	03	E1	05	6A
	2	7	47	**	18	04	4C	03	61	05	2A
		8	47	**	18	04	4C	03	E1	05	6A
2.400	1	7	47	17	18	04	44	03	61	05	2A
		8	47	17	18	04	44	03	E1	05	6A
	2	7	47	17	18	04	4C	03	61	05	2A
		8	47	17	18	04	4C	03	E1	05	6A
1.200	1	7	47	2E	18	04	44	03	61	05	2A
		8	47	2E	18	04	44	03	E1	05	6A
	2	7	47	2E	18	04	4C	03	61	05	2A
		8	47	2E	18	04	4C	03	E1	05	6A
600	1	7	47	5B	18	04	44	03	61	05	2A
		8	47	5B	18	04	44	03	E1	05	6A
	2	7	47	5B	18	04	4C	03	61	05	2A
		8	47	5B	18	04	4C	03	E1	05	6A
300	1	7	47	5B	18	04	84	03	61	05	2A
		8	47	5B	18	04	84	03	E1	05	6A
	2	7	47	5B	18	04	8C	03	61	05	2A
		8	47	5B	18	04	8C	03	E1	05	6A
150	1	7	47	5B	18	04	C4	03	61	05	2A
		8	47	5B	18	04	C4	03	E1	05	6A
	2	7	47	5B	18	04	CC	03	61	05	2A
		8	47	5B	18	04	CC	03	E1	05	6A

*30 nicht genormte, maximale Übertragungsrate (einstellbar bei Kopplung zweier KC 85 aufgrund gleicher Taktfrequenzen)

*31 bei Tolerierung einer bis zu 5%igen Abweichung der Übertragungsrate können für ** die Werte 0B≙4.800, 06≙9.600, 04≙14.400, 03≙19.200, 02≙28.800 und 01≙56.000 eingesetzt werden

3.13. Programmierung der Tonausgabe

Mit dem KC 85 können Sie Töne monophon über das Fernsehgerät bei FBAS oder RGB-Anschluss zu Gehör bringen. Dabei ist die Lautstärke in 16 Stufen mit einer Schrittweite von 2 regelbar. Darüber hinaus erfolgt die Tonausgabe auch stereophon mittels einer Stereoanlage oder eines anderen geeigneten Musikwiedergabegerätes. Hierbei wird der Computeranschluss TAPE über das Diodenkabel mit dem Wiedergabegerät verbunden. Siehe auch Hardware-Kapitel 2.1.5 auf Seite 90.
Im einfachsten Fall drücken Sie die Schnellstopp-Taste Ihres Recorders und schalten diesen auf Aufnahme. Nun funktioniert der Recorder nicht mehr als Speichereinheit, sondern als Musikwiedergabeeinheit des Computers. Achten Sie jedoch bitte in Ihrem eigenen Interesse darauf, dass keine auf dem Band gespeicherten Programme gelöscht werden!

Töne

Der KC 85 verfügt über zwei Kanäle und einen Tonhöhenumfang von je 7 Oktaven. Soll ein bestimmter Ton erzeugt werden, so müssen Sie angeben:

– auf welchem Kanal
– wie lange
– in welcher Tonhöhe
– in welcher Lautstärke

die Tonausgabe erfolgen soll.

Diese Angaben erfolgen entweder mithilfe der BASIC-Anweisung SOUND oder dem System-UP 35H (TON). Anzugeben sind jeweils folgende Parameter:

Zeitkonstante linker Kanal	ZK1	(ARG1)
Vorteiler linker Kanal	VT1	(ARG1+1)
Zeitkonstante rechter Kanal	ZK2	(ARG2)
Vorteiler rechter Kanal	VT2	(ARG2+1)
Lautstärkestufe	LS	(ARG3)
Tondauer	TD	(ARG3+1)

Dabei legen die Parameter ZK1 und VT1 die Tonhöhe des Kanals 1 und die Parameter ZK2 und VT2 die Tonhöhe des Kanals 2 entsprechend der folgenden Tonwerttabelle fest. Mit dem Parameter LS wird die Lautstärke ($0 \leq LS \leq 31$) und mit Parameter TD die Tondauer ($0 \leq TD \leq 255$) bestimmt. Dabei gilt:

– ist die Lautstärke = 0, so erfolgt keine Tonausgabe über das Fernsehgerät, jedoch weiterhin über die Diodenbuchse
– sind die Zeitkonstanten ZK1 = 0 oder ZK2 = 0, so wird kein Ton ausgegeben

– ist die Tondauer = 0, so wird bis zum nächsten Aufruf ein Dauerton abgegeben.

Die Zuordnung der Töne zu den Zeitkonstanten bzw. Vorteilerstufen kann nach folgender Tabelle erhalten werden:

Tabelle 41: Tonwerttabelle für Vorteiler und Zeitkonstanten

Oktave	0		1		2		3		4		5		6	
Ton	VT	ZK	VT	ZK	VT	ZK	VT	ZK	VT	ZK	VT	ZK	VT	ZK
c	1	204	1	102	1	54	1	27	0	216	0	108	0	54
cis	1	199	1	97	1	50	1	25	0	204	0	102	0	51
d	1	196	1	95	1	48	1	24	0	192	0	96	0	48
dis	1	181	1	90	1	45	1	23	0	182	0	91	0	45
e	1	173	1	86	1	43	1	21	0	171	0	86	0	43
f	1	160	1	81	1	40	1	20	0	162	0	81	0	40
fis	1	152	1	75	1	38	1	19	0	153	0	76	0	38
g	1	144	1	72	1	36	1	18	0	144	0	72	0	36
gis	1	137	1	68	1	34	1	17	0	136	0	68	0	34
a	1	127	1	64	1	32	1	16	0	128	0	64	0	32
ais	1	118	1	60	1	30	1	15	0	120	0	60	0	30
h	1	116	1	58	1	29	0	229	0	114	0	57	0	29
c	1	102	1	54	1	27	0	216	0	108	0	54	0	27

Anmerkungen:

– Die Bezeichnung der Oktaven mit 0 bis 6 entsprechen der englischen Kennzeichnung.
– Die 4. Oktave der international üblichen eingestrichenen Oktave geht dabei von c' bzw. C4 (261,6 Hz) bis h' bzw. H4 (493,9 Hz) und beinhaltet auch den in der Tabelle gelb markierten Kammerton A mit 440 Hz.
– Nicht jeder Ton wird nach dieser Tabelle die exakte Frequenz erhalten. Sie werden selbst feststellen, dass damit eigene Kompositionen auch ganz gut klingen.

3.14. Spezielle Systembedingungen

Bei der Arbeit mit dem KC 85/5 sind folgende systemspezifische Bedingungen zu beachten.

3.14.1. Interrupt

- Es ist Interrupt Modus IM2 vorgeschrieben.
- Interrupt-Serviceroutinen dürfen nur in einen Speicherbereich starten, der immer eingeschaltet ist. Neben dem CAOS-ROM-E ist das vor allem der RAM0 bzw. der 16K-RAM-Block, der vom Indexregister IX adressiert wird. Für Anwender-Interrupts ist auch der RAM4 zulässig, nicht jedoch der Adressbereich von 8000H bis DFFFH.
- Der globale Interrupt darf nur kurzzeitig gesperrt werden, damit die seriell übertragenen Tastaturcodes von der KC-Tastatur korrekt erkannt werden. Anwender-Interruptroutinen sollten möglichst am Anfang des Programmcodes den Interrupt mit dem Befehl EI wieder freigeben.
- Falls Interruptroutinen Speicherebenen umschalten, müssen sie bei Rückkehr den vorherigen Schaltzustand wiederherstellen.
- Viele Systemprogramme beeinflussen die globale Interruptfreigabe durch die Befehle DI und EI während der Abarbeitung, dies sind:
  - Programmverteiler PV1 und PV7
  - Unterprogramme IRMON und IRMOFF
  - relativer UP-Aufruf auf Adresse F00FH ab CAOS 4.3
  - alle Unterprogramme mit Zeichenausgabe bei Zugriff auf Farbebene (z. B. CRT, OCHR, AHEX, OSTR usw.) bei CAOS 4.2 bis 4.7
  - ESC-Funktionen zur Bildschirm-Steuerung bei CAOS 4.3 bis 4.7
  - alle Grafikroutinen bei Zugriff auf Farbebene des IRM (z. B. PUSE, PUDE, LINE, CIRCLE) bei CAOS 4.2 bis 4.7
  - alle Unterprogramme zur Magnetbandein- und -ausgabe (z. B. MBI, MBO, SAVE)
  - Unterprogramm Nr. 26H (MODU) beim Schalten interner Module
  - Unterprogramm Nr. 27H (JUMP)
  - Unterprogramm Nr. 31H (SIXD)
  - Unterprogramm Nr. 35H (TON)
  - Unterprogramm Nr. 44H (INIME)
  - Unterprogramm Nr. 47H (LSTOUT) und 48H (V24DUP)

3.14.2. Index-Register IX und IY

– Das IX-Register wird für die Adressierung der Tastatur/Kassetten-Interruptprogramme sowie als Zeiger für die Merkzellen der Ports 84H und 86H benötigt. Es darf bei freigegebenem Interrupt nicht verändert werden. Der Aufruf von System-Unterprogrammen mit einem veränderten IX-Register kann auch bei gesperrtem Interrupt zu undefinierten Systemzuständen führen.

– IX-Register und Interrupttabelle können über das Unterprogramm Nr. 31H (SIXD) auf andere Speicherbereiche gelegt werden.

– Das IY-Register wird von den Routinen IRMON (F018H) und IRMOFF (F01BH) als Zwischenspeicher für den Anwenderstackpointer verwendet. Das Register IY darf nicht verändert werden, wenn diese Programme genutzt werden (z. B. in BASIC).

3.14.3. I/O-Adressen

– Interne I/O-Adressen: 80H bis 8FH sind reserviert, belegt sind zurzeit:

80H	Modulsteuerung
84H und 86H	interne Ausgabeports
88H–8BH	PIO
8CH–8FH	CTC

Wie aus der Speicherübersicht (Bild 24, Seite 122) zu entnehmen ist, bestehen der 256 KByte Arbeitsspeicher aus 16 mal und der 64 KByte IRM aus 4 mal 16 KByte Blöcken. Um diese verwalten zu können, stehen dem Anwender die Ausgabeadressen 84H und 86H zur Verfügung. Damit kann z. B. das Umschalten von Bild 0 (COLOR- und PIXEL-Block) auf Bild 1 erfolgen. Da der Inhalt der beiden Ausgabeports nicht zurückgelesen werden kann, ist jedem Port eine Speicherzelle zugeordnet, siehe Seite 127 bzw. 185. Das Anwenderprogramm muss sicherstellen, dass die Inhalte der Speicherzellen identisch mit den Ausgabeports sind. Dies hat so zu erfolgen, dass immer als Erstes die Speicherzelle und erst danach der Ausgabeport beschrieben wird.

– Für den Anwender stehen die I/O-Adressen 0C0H bis 0CFH und 0D8H bis 0EFH zur Verfügung. Die anderen I/O-Adressen sind für Module bzw. Aufsätze des Herstellers reserviert.

3.14.4. Stack

– Bei Anwenderprogrammen, welche mit eigenem Stackbereich arbeiten und mit Zusatzmodulen im Speicherbereich 8000H - BFFFH (bei abgeschaltetem IRM) arbeiten, ist es notwendig, entweder den STACK in den Bereich unter 8000H zu legen oder vor Aufruf des Betriebssystems den Stackpointer in diesen Bereich zu verlegen und den IRM einzuschalten (z. B. über Programmverteiler PV5 oder PV6).

3.14.5. Varianten des Systemstarts

CAOS kann durch verschiedene Bedienhandlungen gestartet werden:

1. Einschalten mit der Taste <POWER> = ***Kaltstart***
   das entspricht einem Sprung zur Adresse 0F000H. Beim Einschalten des Computers wird der gesamte Speicher gelöscht, alle Module ausgeschaltet und das Betriebssystem mit allen verfügbaren Device-Treibern initialisiert.
2. Rücksetzen mit der Taste <RESET> = ***Warmstart***
   das entspricht einem Sprung zur Adresse 0E000H. Dabei wird zum CAOS-Menü gesprungen, nur der Systemspeicher und die Device-Treiber werden neu initialisiert. Der restliche RAM-Speicher wird nicht gelöscht, unverändert bleiben weiterhin die Schaltzustände der Module, der Joysticktreiber, die eingestellten Fenster und das Bild 1.
3. Aufruf über Betriebssystem UP-Nr. 12H, Name LOOP
   Der augenblickliche Zustand der Systemarbeitszellen bleibt erhalten.

Siehe Bild 23: Zentrale Steuerschleife des Betriebssystems CAOS auf Seite 120.

Bei Systemstart über die Tasten <POWER> oder <RESET> laufen folgende Prozesse automatisch ab:

– Befinden sich im KC 85/5 Module mit Device-Treibern (z. B. M052, M064), werden diese initialisiert und die Treiber in die Device-Tabelle eingetragen.
  Diese Funktion ist seit CAOS 4.7 enthalten.

– Steckt ein V.24-Modul im KC-System, wird es zugeschaltet und initialisiert. Der Kanal 1 wird auf Druckerausgabe und der Kanal 2 auf Duplexbetrieb mit Empfangsinterrupt eingestellt. So kann das V.24-Modul Interrupts, z. B. von einer externen Tastatur entgegennehmen.
  CAOS 4.2 suchte das V.24-Modul ab Steckplatz 8 in 4er-Schritten, ab CAOS 4.3 in 1er-Schritten und ab CAOS 4.4 beginnend ab Steckplatz 7.

– Steckt ein Modul mit Strukturkennbyte 01H im Modulsteckplatz 8 (z. B. M033 TYPESTAR), dann wird beim Systemstart der RAM4 ausgeblendet, das Modul auf Adresse 4000H schreibgeschützt eingeschaltet und auf 4000H gestartet. Das CAOS-Menü erscheint in diesem Fall nicht.
  Bis CAOS 4.6 wurde das Modul R/W eingeschaltet, in CAOS 3.4 ist diese Autostart-Funktion nicht enthalten.

– Ist ein D004 oder D008 am KC 85/5 angeschlossen, welches noch nicht gestartet ist, dann führt CAOS einen Autostart mit JUMP FC aus. Dadurch wird nicht das CAOS-Menü gestartet, sondern das Betriebssystem des Erweiterungsgerätes.
  Diese Funktion ist seit CAOS 4.3 enthalten.

– Ist kein D004 oder D008 am KC 85/5 angeschlossen, jedoch ein Modul mit Device-Treiber (z. B. M052 / USB), dann wird beim Kaltstart von CAOS auto-

matisch das Kommando %INIT vorbereitet. Mit Initialisierung der Treibersoftware das M052 wird im Hauptverzeichnis des USB-Sticks nach der Datei mit dem Namen INITIAL.UUU gesucht und falls vorhanden, das vorbereitete INIT-Kommando aktiviert. Dadurch wird es möglich, beim Einschalten des KC 85/5 mit M052 einen oder mehrere Befehle aus einer Datei INITIAL.UUU automatisch abzuarbeiten.
Diese Funktion ist seit CAOS 4.8 enthalten und erfordert eine USB-Software ab Version 2.9.

3.15. Übersicht der Systemunterprogramme

Tabelle 42: Übersicht Systemunterprogramme PV1-6

UP-Nr. PV1	Name	Funktion	Seite
00H	CRT	Zeichenausgabe auf Bildschirm	138
01H *32	MBO	Datenblock auf Speichergerät ausgeben	138
02H	UOT1	Ausgabe auf Anwenderkanal 1	139
03H	UOT2	Ausgabe auf Anwenderkanal 2	139
04H	KBD	Tasteneingabe mit Cursor-Einblendung	139
05H	MBI	Datenblockes von Speichergerät einlesen	139
06H	USIN1	Eingabe von Anwenderkanal 1	140
07H	USIN2	Eingabe von Anwenderkanal 2	140
08H *32	ISRO	Initialisierung Dateiausgabe (Schreiben öffnen)	140
09H *32	CSRO	Abschluss Dateiausgabe (Datei schließen)	141
0AH	ISRI	Initialisierung Dateieingabe (Lesen öffnen)	141
0BH	CSRI	Abschluss Dateieingabe (Datei schließen)	142
0CH	KBDS	Tastenstatusabfrage ohne Quittierung	142
0DH	BYE	Sprung auf RESET	142
0EH	KBDZ	Tastenstatusabfrage mit Quittierung	142
0FH	COLORUP	Farbe einstellen	143
10H	LOAD	Laden eines Maschinenprogrammes	143
11H	VERIF	Kontrolllesen von Kassettenaufzeichnungen	143
12H	LOOP	Übergeben Steuerung an CAOS	143
13H *32	NORM	Rückschalten E/A-Kanäle auf CRT und KBD	144
14H	WAIT	Warteschleife	144
15H *32	LARG	Register mit Argumenten laden	144

UP-Nr. PV1	Name	Funktion	Seite
36H	SAVE	Maschinenprogramm auf Speichergerät ausgeben	153
37H	MBIN	Byteweise Eingabe vom Speichergerät (BASIC)	154
38H	MBOUT	Byteweise Ausgabe auf Speichergerät (BASIC)	155
39H	KEY	Belegung einer Funktionstaste	156
3AH	KEYLI	Anzeige der Funktionstastenbelegung	156
3BH	DISP	HEX/ASCII-Dump von Speicherinhalten	156
3CH	WININ	Fenster initialisieren	157
3DH	WINAK	Aufruf Fenster über Fensternummer	157
3EH	LINE	Zeichnen einer Linie	157
3FH	CIRCLE	Zeichnen eines Kreises	158
40H	SQR	Quadratwurzelberechnung	158
41H *32	MULT	Multiplikation zweier 8-Bit-Zahlen	158
42H	CSTBT	Zeichenausgabe mit Negation des Steuerbytes	158
43H *32	INIEA	Initialisierung eines E/A-Ports	159
44H *32	INIME	Initialisierung mehrerer E/A-Ports	159
45H *32	ZKOUT	Ausgabe einer Zeichenkette, Zeiger in HL	160
46H	MENU	Anzeige des aktuellen Menüs, Kommandoeingabe	161
47H	LSTOUT	Druckerinitialisierung (bis CAOS 4.4: V24OUT)	162
48H	V24DUP	Initialisierung V.24-Duplexroutine	162
49H	SETDEV	Speichergerät einstellen (ab CAOS 4.6)	163
4AH	HLDEZ	Ausgabe HL als Dezimalzahl (ab CAOS 4.8)	163
4BH	RDEZ	Erfassung einer Dezimalzahl (ab CAOS 4.8)	163
4CH	GARGC	Erfassung von 10 Argumenten mit wählbarer Zahlenbasis (ab CAOS 4.8)	164

*32 Unterprogramme, welche Parameter in den Registern BC, DE, HL an das Hauptprogramm zurückgeben, benötigen Programmverteiler PV1 oder PV7. Bei allen anderen Programmverteilern werden die Register BC, DE, HL vor der Abarbeitung des gewünschten Unterprogramms gerettet und danach wieder mit den vorherigen Werten geladen.

Tabelle 43: Übersicht Systemunterprogramme PV7 (ab CAOS 4.7)

UP-Nr. PV7	Name	DEVICE-Funktion (nur über PV7 erreichbar)	Seite
UP 0	MBO	Datenblock auf Speichergerät ausgeben	165
UP 1	MBI	Datenblockes von Speichergerät einlesen	166
UP 2	ISRO	Initialisierung Dateiausgabe (Schreiben öffnen)	166
UP 3	CSRO	Abschluss Dateiausgabe (Datei schließen)	167
UP 4	ISRI	Initialisierung Dateieingabe (Lesen öffnen)	167
UP 5	CSRI	Abschluss Dateieingabe (Datei schließen)	168
UP 6	DRVER	Treiber-Version abfragen (ab CAOS 4.8)	168
UP 7.x	DRV:USR	Treiber-spezifische Zusatzfunktionen, siehe Tabelle 44 (ab CAOS 4.8)	168
UP 8	DIRANZ	Verzeichnisinhalt anzeigen	175
UP 9	CD	Verzeichnis wechseln	176
UP 10	ERA	Datei löschen	176
UP 11	REN	Datei umbenennen	177

Tabelle 44: Übersicht USB-Zusatzfunktionen PV7-UP7 (ab CAOS 4.8)

UP-Nr. PV7-7	Name	Treiber-spezifische USB-Funktion (ab Treiber V3.0 über PV7, UP7 erreichbar)	Seite
UP 7.0	USER-MD	USB: USER-Mode für VNC einstellen	170
UP 7.1	CAOS-MD	USB: CAOS-Mode für VNC einstellen	170
UP 7.2	RAWOUT	USB: direktes Schreiben VNC	170
UP 7.3	RAWIN	USB: direktes Lesen VNC	171
UP 7.4	TEST	USB: Test, ob Datei vorhanden ist	171
UP 7.5	OPEN	USB: Datei zum Lesen öffnen	171
UP 7.6	READ1	USB: Ein Byte lesen	172
UP 7.7	READN	USB: mehrere Bytes lesen	172
UP 7.8	CREATE	USB: neue Datei zum Schreiben öffnen	172
UP 7.9	APPEND	USB: Datei zum Schreiben öffnen	173
UP 7.10	WRITE1	USB: Ein Byte schreiben	173
UP 7.11	WRITEN	USB: mehrere Bytes schreiben	173
UP 7.12	CLOSE	USB: Datei schließen	174
UP 7.13	FIRST	USB: ersten Verzeichniseintrag suchen	174
UP 7.14	NEXT	USB: nächsten Verzeichniseintrag suchen	174
UP 7.15	STICK	USB: Test, ob USB-Stick vorhanden ist	175

4. ERWEITERUNGEN

BASIC

Assembler

Reassembler

Testmonitor/Debugger

Editor

4.1. BASIC

Einführung

Als Vorlage für die folgenden Kapitel diente das BASIC-Handbuch des KC 85/4. Der Inhalt wurde für die Gegebenheiten bis CAOS 4.8 ergänzt. Das BASIC-Handbuch des KC 85/4 ist wie ein Lehrbuch aufgebaut und ermöglicht das Erlernen der Programmiersprache durch Beispiele und Übungen. Das Prinzip wurde hier beibehalten, teilweise werden jedoch grundlegende Dinge neu erläutert, welche bereits in vorhergehenden Kapiteln behandelt worden sind, so z. B. das hexadezimale Zahlensystems. Um den Lehrbuch-Charakter zu erhalten, wurden diese Erläuterungen jedoch beibehalten.

Der BASIC-Interpreter ermöglicht es Ihnen, sich mit Ihrem KC 85 fast umgangssprachlich zu unterhalten. Er realisiert also ein Dolmetscher-Programm zwischen der Maschinensprache, die der Computer versteht, und BASIC, einer Sprache, welche sich stark an Englisch anlehnt.

Warum programmieren wir nicht gleich in Maschinensprache? Die Antwort ist einfach. Wir programmieren in einer höheren Programmiersprache, nämlich konkret in BASIC, weil diese für uns viel leichter verständlich ist. BASIC-Anweisungen, wie z. B. PRINT, INPUT, NEXT oder GOTO, sind doch entschieden einprägsamer als Maschinencodes, wie z. B. E5, C9, 7F oder 88.

BASIC beherrschen über 70% der in der Welt vorhandenen Mikrocomputer, d. h. sie können alle in BASIC geschriebenen Programme in Ihrem Computer mit nur geringen Veränderungen eingeben und nutzen. Der Name BASIC ist die Abkürzung für "Beginners All Purpose Symbolic Instruction Code", übersetzt eine "Universelle Programmiersprache für Anfänger". Das "Anfänger" bezieht sich jedoch nur auf eine leichte Erlernbarkeit der Sprache, denn mit den komfortablen BASIC-Anweisungen können nicht nur Anfänger etwas anfangen.

Mithilfe des BASIC-Interpreters erlernt man die Sprache sehr leicht. Die Übersichten ab Seite 367 enthalten nochmals eine Zusammenfassung der BASIC-Anweisungen und dienen als Nachschlagewerk im täglichen Umgang mit dem Computer.

Mit CAOS 4.7 sind BASIC-Anweisungen verändert und erweitert worden. Hier werden auch die Anweisungen beschrieben, die sich gegenüber früheren Versionen geändert haben.

Beim Betrieb mit einem D004/D008 ist zu beachten, dass die seit CAOS 4.6 enthaltene DEVICE-Umschaltung auch unter BASIC wirksam ist. BASEX und ähnliche Hilfsprogramme sind nicht mehr erforderlich. Die CALL-Befehle aus BASEX und FLOAD sind ab CAOS 4.6 nicht mehr gültig! Für die wichtigsten Aufrufe gibt es jetzt fest eingebaute Anweisungen:

CALL-Anweisung bis CAOS 4.5	Ersatz-Anweisung ab CAOS 4.7	Funktion
CALL*12 CALL*D8	BLOAD“NAME“	MC-Datei nachladen
CALL*DE	FILES“MASKE“	Verzeichnis anzeigen
CALL*F0	CHDIR“NAME“	Verzeichnis wechseln
CALL*150	DEVICE	Speichergerät wechseln

Mit der Anweisung

```
PRINT DEEK (513) >0 = Magnetbandzugriff
PRINT DEEK (513) <0 = Diskettenzugriff
```

konnte bei Verwendung von BASEX abgefragt werden, ob der Zugriff gerade auf DISK oder TAPE eingestellt ist. Das funktioniert seit CAOS 4.6 nicht mehr. Als Ersatz bietet sich die Abfrage des gerade eingestellten DEVICE an, welches mit der Anweisung

```
PRINT (PEEK(504) AND 28)/4
```

die DEVICE-Nummer aus den Bit 2-4 von (IX+8) anzeigt. Der Wert 0=TAPE zeigt Magnetbandzugriff an, weitere DEVICE-Nummern ergeben sich anhand der vorhandenen Hardware, vgl. Kapitel 3.9.1 auf Seite 197.

4.1.1. BASIC Kaltstart/Warmstart und Anweisung BYE

Wie Sie bereits erfahren haben, wird uns der BASIC-Interpreter das Programmieren stark erleichtern. Um den Interpreter zu nutzen, müssen wir diesen starten. Es stehen zwei Möglichkeiten zur Verfügung.

Der BASIC-Interpreter wird im CAOS-Menü mit diesen Kommandos gestartet:

%BASIC	Kaltstart des BASIC-Interpreters
%REBASIC	Warmstart des BASIC-Interpreters

Das Betriebssystem wird verlassen und der Interpreter meldet sich beim Kaltstart mit folgendem Bild:

```
HC-BASIC
MEMORY END?:
```

Nun können Sie den zur Verfügung stehenden Speicherbereich durch Vorgabe der dezimalen Speicherendadresse begrenzen. Wird die Speicheradresse nicht vorgegeben, so wird der größtmögliche Speicherbereich genutzt. Durch Druck auf die ENTER-Taste wird die Größe des vom Anwender begrenzten oder nicht begrenzten Arbeitsspeichers für BASIC-Programme in der Form von

```
47854 BYTES FREE
```

angezeigt und der Interpreter meldet sich mit den Zeilen

```
OK
>_
```

betriebsbereit.

47854 BYTES FREE bedeutet, dass uns ein Arbeitsspeicher, der RAM also, von 47854 Byte zur Verfügung steht. Bei diesem Kaltstart werden alle BASIC-Programme und -Daten gelöscht.

Der Warmstart des BASIC-Interpreters kann in gleicher Weise mithilfe des CAOS-Kommandos REBASIC ausgeführt werden. Hierbei bleiben jedoch die BASIC-Programme und -Daten im Speicher erhalten. Diese Anweisung sollte also nur benutzt werden, wenn bereits ein BASIC-Programm im Arbeitsspeicher ist, welches nicht gelöscht werden soll. Beim Warmstart meldet sich der Interpreter sofort mit:

```
OK
>_
```

Tastatur unter BASIC und die Anweisung BYE

Die wichtigste Taste ist auch beim Betrieb des BASIC-Interpreters die ENTER-Taste ganz rechts unten auf der Tastatur. Mit dieser Taste werden eingegebene Kommandos ausgeführt und Programmanweisungen gespeichert.

Die Tasten INS , DEL , ← , → und ↓ sind nur bei der Anweisungseingabe und mithilfe des Befehls EDIT funktionsfähig. In Kapitel 4.1.8. werden wir ausführlich auf diese Problematik eingehen.

Tabelle 45: Editiermöglichkeiten des BASIC-Interpreters

Tastenbetätigung	Funktion
←	Cursor nach links
→	Cursor nach rechts
↕ + ←	Cursor auf den Zeilenanfang
↕ + →	Cursor an das Zeilenende
↕ + DEL	Zeile löschen
↕ + HOME	Bildschirm löschen
DEL	Zeichen löschen
INS	Zeichen einfügen

Nachdem wir den Interpreter gestartet haben, wollen wir uns der Vollständigkeit halber gleich darüber informieren, wie wir diesen gegebenenfalls auch wieder verlassen können. Mithilfe der BASIC-Anweisung BYE verabschieden wir unseren Interpreter und befinden uns nach Ausführung des Kommandos wieder im Betriebssystem CAOS. Geben Sie ein:

```
BYE
```

Betätigen Sie nun die ENTER-Taste ↵, damit der Befehl ausgeführt wird. Es meldet sich die Kommandoeingabe des Betriebssystems.

Nun machen Sie bitte zur Übung einen "Warmstart" aus dem Betriebssystem in den BASIC-Interpreter, wie oben beschrieben.

Sollten Sie einmal eine fehlerhafte Eingabe mit dem BASIC-Interpreter des KC 85 zur Ausführung gebracht haben, so erscheint auf dem Bildschirm eine Fehlermeldung (z. B. ?SN ERROR) und es wird vom eingebauten Piezosummer sowie über FBAS oder RGB angeschlossene Fernsehgeräte ein akustisches Signal ausgegeben. Die Erklärung zu den Fehlermeldungen, welche Sie in der BASIC-Übersicht auf Seite 374 finden, wird Ihnen eine große Hilfe sein.

Was machen wir, wenn wir uns verschrieben haben oder „es einfach nicht weitergeht“?

1. Wir setzen den Cursor zurück und überschreiben bzw. korrigieren mit den bereits erwähnen Editiermöglichkeiten die falsche Stelle oder
2. wir drücken die ENTER-Taste.
   Führt das nicht zum gewünschten Erfolg, d. h. der Computer reagiert nicht auf Eingabe, so können wir
3. die RESET-Taste drücken (nicht auf der Tastatur, direkt am Computer) und mit einem "Warmstart" (Kommando "REBASIC") aus dem Betriebssystem in den BASIC-Interpreter zurückkehren. Dabei wird das BASIC-Programm nicht zerstört. Führt das auch nicht zum Erfolg, so werden wir
4. nach nochmaliger Betätigung der RESET-Taste den BASIC-Interpreter mit dem Kommando "BASIC" erneut starten. Hierbei wird jedoch das eventuell vorhandene BASIC-Programm zerstört. Arbeitet der BASIC-Interpreter immer noch nicht einwandfrei, so ist es ratsam,
5. das Gerät aus- und einzuschalten.

Merke:

- "Kaltstart" des BASIC-Interpreters mit dem Kommando "BASIC"
- "Warmstart" des BASIC-Interpreters mit dem Kommando "REBASIC" (Hier bleiben bereits eingegebene Programme erhalten.)
- **Anweisung: BYE**
  Format: BYE
  Bemerkung: BYE gibt die Steuerung an das Betriebssystem zurück.
  Beispiel: BYE
- Es ist unmöglich, den KC 85 durch falsche Eingaben zu beschädigen. Schlimmstenfalls müssen Sie den Computer erneut einschalten.

4.1.2. BASIC-Anweisungen PRINT, LET, CLEAR, CLS

Unser KC 85 kann mehr als wir uns im ersten Moment vorstellen können. Er wartet nur darauf, das auszuführen, was wir ihm anweisen. Das erste und wichtigste Kommando, das wir lernen wollen, ist die Anweisung etwas auf dem Bildschirm darzustellen. Geben Sie ein:

```
PRINT 2
```

Nun hat der Computer die Anweisung bekommen, eine "2" auf den Bildschirm zu schreiben und wartet auf den Hinweis, dass er die Anweisung ausführen soll. Diese, wie auch jede andere Anweisung wird durch Betätigen der ENTER-Taste ausgeführt. Überzeugen Sie sich mit einem Druck auf die ENTER-Taste. Sie sehen, der Befehl wird ausgeführt. Es erscheint die Zahl 2. Geben Sie ein:

```
PRINT 7+9
```

(Anweisungsausführung durch ENTER-Taste nicht vergessen!)
Sie erhalten das Ergebnis. Probieren Sie nun selbstständig weiter:

```
PRINT 7-58
```

und

```
PRINT 355+48-66
```

Vergessen Sie bitte nicht, nach jeder Anweisung die ENTER-Taste zu drücken! Unser KC 85 als Taschenrechner besitzt jedoch auch einige Besonderheiten gegenüber der üblichen mathematischen Schreibweise, auf die wir jetzt näher eingehen wollen.

Wir müssen stets den Buchstaben O und die Ziffer 0 (NULL) unterscheiden. Aus diesem Grund wird die Null durchgestrichen.

Führende Nullen vor dem Komma können weggelassen werden.

Da BASIC sehr viele englische Begriffe enthält, wird auch bei der Zahlendarstellung die englische Schreibweise gewählt. Wir schreiben also einen Dezimalpunkt statt eines Dezimalkommas. Der KC 85 arbeitet mit ganzen und reellen Zahlen. Hierbei müssen wir jedoch beachten, dass wir bei der Exponentendarstellung statt "mal zehn hoch" einfach E schreiben.

Nun einige Beispiele:

Mathematische Schreibweise	Computerschreibweise
2,7	2.7
0,16	.16
$3{,}84 * 10^{-20}$	3.84E–20
$9{,}02 * 10^{13}$	9.02E13

Variablen und Konstanten

Das Verständnis dieser beiden Begriffe ist die wichtigste Voraussetzung zum Erlernen einer Programmiersprache. Deshalb werden wir uns jetzt anschaulich mit diesen Begriffen vertraut machen.

Konstanten sind festgelegte, unveränderliche Werte. Der KC 85-BASIC-Interpreterverarbeitet Zahlen und Zeichenketten als Konstanten. Als Zahlen (numerische Konstanten) stehen ganze Zahlen von -999999 bis +999999 sowie reelle Zahlen mit einem Betrag zwischen 9.40396E-39 bis 1.70141E+38 und Null zur Verfügung. Die Zeichenketten-Konstanten, eine Aneinanderreihung beliebiger Zeichen, nennen wir Strings. Sie werden stets von Anführungszeichen eingeschlossen und können höchstens 255 Zeichen lang sein (z. B. "LUTZ").

Variablen, das sagt schon der Name, sind veränderliche Größen. Zum besseren Verständnis stellen wir uns vor, unser Computer besitzt eine Menge von Fächern, die Zahlen oder Worte enthalten können. Den Fächern können wir Namen geben. Diese Namen kennzeichnen unsere Variablen. Der Inhalt eines Faches entspricht dem wert dieser Variablen. Die Namen der Variablen stehen also stellvertretend für den momentan in dem Fach vorhandenem Wert. Veranschaulichen wir uns dies im folgenden Bild:

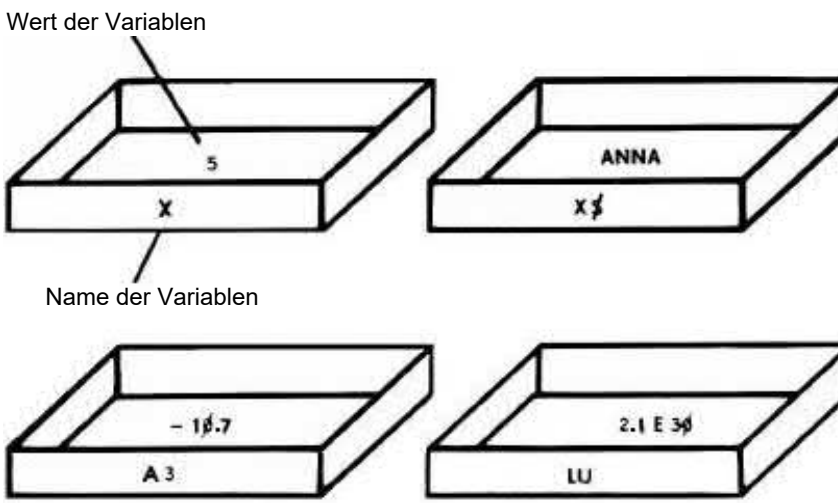


Bild 30: Veranschaulichung des "Gedächtnisses" unseres Computers

Zum betrachteten Zeitpunkt besitzen

- die Variable X den Wert 5
- die Variable X$ den Wert "ANNA"
- die Variable A3 den Wert -10.7
- die Variable LU den Wert 2.1E30

Wie wir sehen, können die Werte einer Variablen sowohl Zahlen als auch Zeichenketten (Strings) sein. Deshalb unterscheiden wir numerische Variablen und Stringvariablen.

Der Variablenname ist unter Berücksichtigung folgender Regeln frei wählbar:

1. Ein Variablenname muss immer mit einem Buchstaben beginnen, z. B. ANNA, LUTZ, WERT, X, A, B7, Typ
2. Am Ende des Namens einer Stringvariablen steht das $-Zeichen, z. B. X$, TYP$
3. Variablennamen dürfen keine Worte enthalten, die schon mit einer festen Bedeutung in BASIC benutzt werden. Der Variablenname LETTER ist z. B. nicht erlaubt, da er die BASIC-Anweisung LET enthält.
4. Obwohl eine Variable beliebig lang sein kann, unterscheidet der Computer nur die ersten beiden Zeichen einer Variablen. So kann er z. B. die Variablen ROHR und ROSE nicht unterscheiden.

Wertzuweisung mit LET und die Anweisung CLEAR

Eine Variable können wir mit einem Speicher in einem Taschenrechner vergleichen. Unser Computer bietet jedoch den Vorteil, dass wir sehr viele solcher Speicher anlegen können. Wir geben Ihnen, wie oben beschrieben, Variablennamen. Mithilfe der Anweisung LET schaffen wir uns eine solche Variable und weisen ihr einen Wert zu. Geben Sie ein:

```
LET WURST=3.57
```

(ENTER-Taste nicht vergessen!)

Der Computer hat sich nun ein Fach (besser eine Variable) angelegt, die den Namen WU trägt und dieser den Wert 3.57 zugeordnet. Wir können uns vorstellen, dass dies in der Praxis z. B. die Abspeicherung des Preises für ein Kilogramm Wurst sein kann. Nun löschen wir den Bildschirm mit der Anweisung CLS.

Geben Sie ein:

```
CLS
```

(ENTER-Taste!)

Jetzt ist der Wurstpreis nicht mehr zu sehen, aber wir möchten ihn wieder erfahren. Geben Sie ein:

```
PRINT WURST
```

Die Ausführung dieser Anweisung bewirkt, dass wir den unter der Variablen WU abgelegten Wert, also den Wurstpreis, angezeigt bekommen. Wenn Sie z. B. den Preis von 3,5 Kg Wurst erfahren wollen, so geben Sie ein:

PRINT 3,5 * WURST

Wie sie bemerken, ist das Zeichen " * " das Multiplikationszeichen. Das Zeichen für die Division ist der Schrägstrich " / ". So ermitteln wir den Preis für 1/2 kg Wurst mit den Anweisungen:

PRINT WURST/2

bzw.

PRINT .5*WURST

Nun probieren Sie das auch mit anderen Mengen Wurst!

Angenommen, diese Wurst ist ausverkauft und dafür wird eine andere Wurstsorte angeboten. Dann geben wir z. B. ein:

LET WURST=3.10

Wenn Sie nun schreiben:

PRINT WURST

So erfahren Sie den neuen Preis.

Da Wertzuweisungen sehr häufig vorkommen, kann die Anweisung LET auch weggelassen werden. Wir schreiben:

WURST=3.10

Darüber hinaus können wir, wie oben bereits erwähn, auch andere Variablen festlegen, z. B.

W2=2*WURST

BROT=0.93

F6=3.20

U4=111 * F6

(Nach jeder Anweisung die ENTER-Taste drücken!)

Das in den Anweisungen auftretende Gleichheitszeichen ist jedoch inhaltlich nicht dem uns aus der Mathematik bekannten identisch. Schreiben wir z. B. F6=3.20, so bedeutet das: Die links stehende festzulegende Variable F6 bekommt den rechts stehenden Wert von 3.20 zugewiesen.

Diese Wertzuweisung darf nie seitenvertauscht erfolgen.

Beim Beginn von Rechnungen ist es üblich, alle nachfolgend verwendeten Variablen zu löschen, d. h. ihnen wird der Wert 0 zugewiesen. An Ihrem Taschenrechner können Sie alle Speicher löschen, indem Sie die C-Taste drücken. Selbstverständlich bietet uns der KC 85 den gleichen Komfort. Wir verwenden den Befehl CLEAR. Geben Sie ein:

CLEAR

Nun überzeugen Sie sich von der Wirkungsweise der Anweisung, indem Sie sich die Werte der vorhin festgelegten Variablen anzeigen lassen.

```
PRINT BROT
```

usw.

Sie werden feststellen, dass die Variablen alle den Wert 0 enthalten.

Die Grundrechenarten

Mit Zahlen und den von Ihnen definierten Variablen können Sie nun nach Belieben rechnen. Geben Sie z. B. ein:

```
PRINT 5*F6+BROT+WURST/3+15.76
```

Falls Sie die Werte für F6, BROT und WURST noch nicht wieder neu festgelegt haben, werden Sie als Ergebnis 15.76 erhalten haben. Dies ist auch ganz klar, da wir die Variablen mit der Anweisung CLEAR gelöscht haben.

Weisen Sie nun den Variablen neue Werte zu und berechnen Sie die Aufgabe nochmals. Sie erhalten das richtige Ergebnis. Bei allen Rechnungen ist zu beachten, dass der BASIC-Interpreter ihm übertragene Aufgaben in einer genau festgelegten Reihenfolge verarbeitet, d. h. er gibt bestimmte Operationen Vorrang vor anderen. So weiß er natürlich auch, dass Punkt- vor Strichrechnung geht. Das bedeutet, dass er z. B. in der letzten Aufgabe, erst die Multiplikation 5*F6 und die Division WURST/3 ausführt. Danach wird von links nach rechts addiert bzw. subtrahiert.

Natürlich können Sie bestimmte Operationen durch Klammern der Vorrang geben, z. B.:

```
PRINT 5*F6+(BROT+WURST)/3+15.76
```

Nun werden Sie ein anderes Ergebnis als vorhin erhalten, da wir Klammern gesetzt haben. Unsere Aufgabe wurde nämlich wie folgt abgearbeitet:

1. Klammerinhalt berechnen
2. Multiplikation und Division ausführen
3. Von links nach rechts addieren bzw. subtrahieren

Um sich zu überzeugen, geben Sie noch folgendes Beispiel ein:

```
A=4+3*2:B=(4+3)*2: PRINT A, B
```

Bei diesem Beispiel sind zwei neue Möglichkeiten angewendet worden: Mit dem Doppelpunkt können mehrere Anweisungen getrennt werden, d. h. es können somit mehrere Anweisungen in einer Zeile geschrieben werden. Durch die Angabe von mehreren Variablen in der PRINT-Anweisung können mehrere Werte mit einer Anweisung dargestellt werden; es entsteht damit eine sogenannte PRINT-Liste.

Beachten Sie, dass das Multiplikationszeichen " * " auch bei Klammerrechnungen und bei Variablen angegeben werden muss.

Beispiele:	C=3*(5+7*A)	richtig
	C=3(5+7*A)	falsch
	C=3*(5+7A)	falsch

Merke:

- **Anweisung: PRINT**
- Besonderheiten der Computerschreibweise
- Variablen, Konstanten, Strings
- Trennen von Anweisungen auf einer Zeile durch Doppelpunkt
- Mit der PRINT-Anweisung können mehrere Werte dargestellt werden
- **Anweisung: LET**
  Format: LET Variable = Ausdruck
  Bemerkung: Die Anweisung weist einer Variablen einen Wert zu. Die Anweisung kann auch weggelassen werden.
  Beispiel: LET X=3*7+5*A
  oder
  Y=3*(7+5*A)
  oder
  X$="ABC 123"
- **Anweisung: CLEAR**
  Format: CLEAR
  Bemerkung: CLEAR löscht den Variablenspeicher. Weitere Funktionen der Anweisung werden später erläutert.
  Beispiel: LET A=4
  PRINT A
  CLEAR
  PRINT A
- **Anweisung: CLS**
  Format: CLS
  Bemerkung: CLS löscht den Bildschirm
  Beispiel: CLS

Übungen

1. Geben Sie ein:

```
PRINT 4.8E0
PRINT 4.8E1
PRINT 4.8E2
⋮
PRINT 4.6E9
```

Wir erkennen, dass Zahlen, die länger als sechs Ziffern sind, in wissenschaftlicher Notation (mit Exponenten) dargestellt werden. Probieren Sie gleiches mit negativen Zahlen!

2. Geben Sie ein:

```
CLS
S51=3210
PRINT S51
CLEAR
PRINT S51
```

Beantworten Sie anhand der Übung folgende Fragen:

Was bewirken die Anweisungen CLS und LET ?
Worin unterscheidet sich das Gleichheitszeichen einer LET-Anweisung vom mathematischen "=" ?
Was bewirkt die Anweisung CLEAR ?

3. Warum werden die beiden Variablen S51 und S52 in BASIC nicht unterschieden?

4. Geben Sie ein

```
PRINT 6*4+5
PRINT 5+4*6
```

Warum gelangt der Computer (im Gegensatz zu vielen Taschenrechnern) bei beiden Aufgaben zum gleichen Ergebnis?

4.1.3. BASIC-Anweisungen RUN, GOTO, LIST, CONT

Text schreiben

Ein großer Vorteil unseres KC 85 gegenüber einem Taschenrechner ist die Möglichkeit der Textverarbeitung.

Geben Sie ein:

```
PRINT "KLEINCOMPUTER AUS MUEHLHAUSEN"
```

ENTER-Taste drücken!

Sie sehen, die Zeichenkette innerhalb der Anführungszeichen wird auf dem Bildschirm geschrieben. So können wir beliebige Zeichenketten mit Hilfe der PRINT-Anweisung in Verbindung mit den Anführungszeichen ausdrucken lassen.

Das Programm

Wollen wir den obigen Text wiederholt ausgeben lassen, so müssen wir nach eben beschriebenen Verfahren die Anweisung jeweils neu eingeben. Hier kommt uns die Möglichkeit der BASIC-Programmierung entgegen. Schreiben wir also ein Programm, das obigen Text ausdruckt. Geben Sie ein:

```
10 PRINT "KLEINCOMPUTER AUS MUEHLHAUSEN"
```

Drücken Sie die ENTER-Taste!

Der Computer hat die Anweisung offensichtlich nicht ausgeführt. Das ist richtig, denn mit Betätigen der ENTER-Taste hat er die Anweisung als eine Programmzeile unter der Zeilennummer 10 abgespeichert. Das ist die zweite wesentliche Funktion der ENTER-Taste (Kommandos werden mithilfe der ENTER-Taste ausgeführt; Programmzeilen jedoch nur abgespeichert).

Nun wollen wir unser Programm, das aus einer Programmzeile mit einer Anweisung besteht, abarbeiten. Geben Sie ein:

```
RUN
```

Vergessen Sie nicht, die ENTER-Taste zu drücken!

Es erscheint der Text auf dem Bildschirm; der Computer meldet sich mit OK und das Programm ist abgearbeitet. Dieses Programm können Sie nun beliebig oft mit der Anweisung RUN starten und ablaufen lassen. Probieren Sie!

Die Anweisung RUN bewirkt also die Abarbeitung eines Programms. Beginnend mit der niedrigsten Zeilennummer werden die Programmzeilen in aufsteigender Nummerierung abgearbeitet. Hierbei gibt es jedoch Ausnahmen. Eine solche ist z. B. die Anweisung GOTO. Geben Sie ein:

```
20 GOTO 10                (ENTER-Taste !)
```

Und nun lassen wir unser um die Zeile 20 erweitertes Programm mit der Anweisung RUN ablaufen. Der Bildschirm wird vollgeschrieben und danach der Bildschirminhalt von unten nach oben geschoben, wenn Sie den Scroll-Modus gewählt haben. Haben Sie genug davon, drücken Sie die BRK-Taste. Dadurch wird der Programmablauf unterbrochen und der BASIC-Interpreter meldet sich mit

```
BREAK IN 10
OK
>_
```

(oder BREAK IN 20, je nach dem in welcher Programmzeile die Abarbeitung unterbrochen wurde).
Nun löschen wir den Bildschirm mit dem Kommando CLS
Wir möchten uns das Programm wieder ansehen. Geben Sie ein:

```
LIST
```

(ENTER-Taste drücken!)

Wir schauen uns anhand unseres Programms noch einmal an, was eben auf dem Bildschirm passierte. Anweisungszeile 10 wurde wie gehabt ausgeführt. Danach erfolgte die Abarbeitung der Zeile 20. Die Anweisung auf dieser Zeile (GOTO 10) ist eine Sprunganweisung zur Zeile 10. Die Ausführung dieser Anweisung bewirkt, dass der Computer mit der Abarbeitung der Programmzeile 10 fortfährt. Nun wird wieder unser Text ausgedruckt und die nächste Anweisung auf Programmzeile 20, wie eben geschildert, abgearbeitet usw.

Wir wollen unser Programm weiter komplettieren, indem wird den Bildschirm zuerst löschen lassen. Wir geben ein:

```
5 CLS
```

Starten Sie das Programm mit RUN und unterbrechen Sie es nach einer gewissen Zeit durch Druck auf die BRK-Taste. Danach können Sie das Programm fortsetzen. Geben Sie ein:

```
CONT
```

(continue, übersetzt fortsetzen)

Sie merken, das Programm arbeitet weiter. Eine andere Möglichkeit den Programmablauf einmal kurz anzuhalten, um sich die Kontrolle der Aufschrift auf dem Bildschirm anzusehen, ist durch das Drücken der STOP-Taste gegeben. Wollen Sie das Programm wieder starten, so betätigen Sie die Taste  und das Programm wird an gleicher Stelle fortgesetzt, an der es unterbrochen wurde; oder. Sie betätigen die BRK-Taste und der BASIC-Interpreter meldet sich mit:

```
BREAK IN …
OK

>_
```

Nun lassen Sie sich das Programm mithilfe des Kommandos LIST wieder anzeigen. Sie sehen, dass die Programmzeile 5 entsprechend der Zeilennummer in Programm eingeordnet wurde. Wir können also nachträglich Programmzeilen einfügen. Es wird deshalb empfohlen, die Programmzeilen in Zehnerschritten zu nummerieren. Sie können dann, wenn es sich später als erforderlich erweist, auch nachträglich noch jeweils bis zu neun Programmzeilen einfügen.

Bei der Anweisung RUN sucht der Computer die Programmzeile mit der niedrigsten Zeilennummer und beginnt dort mit der Abarbeitung des Programms. Es gibt aber auch die Möglichkeit, das Programm auf einer anderen als der niedrigsten Zeilennummer zu starten. Angenommen, wir möchten unser Programm nicht auf der Programmzeile 5, sondern erst auf der Programmzeile 20 starten, so geben wir ein:

```
RUN 20
```

Wir sehen, der Bildschirm wird nicht gelöscht, denn der BASIC-Interpreter arbeitet niemals die Programmzeile 5 ab.

Verschaffen wir uns erst einmal einen Überblick über unsere neuen Erkenntnisse:

- Ein Programm besteht aus Programmzeilen und Zeilennummern.
- Eine Programmzeile wird durch Betätigen der ENTER-Taste gespeichert.
- Die Programmzeilen werden vom Computer entsprechend ihrer Zeilennummerierung aufgelistet und abgearbeitet.
- Die Programmzeilen können wir korrigieren, indem wir die entsprechenden Stellen überschreiben oder die Zeile neu eingeben.
- Programmzeilen können wir löschen, indem wir nur die Zeilennummer eingeben und die ENTER-Taste drücken.
- Ein laufendes Programm können wir durch Betätigen der BRK-Taste unterbrechen, um z. B. irgendwelche Rechnungen durchzuführen, und, sofern wir nichts in unserem Programm geändert haben, mit der Anweisung CONT fortsetzen.
- Ein laufendes Programm können wir durch Betätigen der STOP-Taste anhalten, um z. B. einen Ausdruck länger anzusehen zu können, und durch Betätigen der Taste  fortsetzen.

Beachten Sie bitte noch folgende Hinweise:

Die Zeilennummer einer Programmzeile können Sie zwischen einschließlich 0 und 65529 frei wählen. Es besteht auch die Möglichkeit, mehrere durch Doppelpunkt getrennte Anweisungen auf eine Zeile zu schreiben. Eine Zeile darf dabei nur maximal 72 Zeichen lang sein. Da aber nur 40 Zeichen einschließlich der Zeilennummer auf einer Bildschirmzeile passen, wird automatisch ein Überlauf in die nächste Bildschirmzeile realisiert.

Solche Überläufe stören aber das Schriftbild erheblich und insbesondere dann, wenn man in einem Programm nach Fehlern suchen muss. Es ist daher besser, wenn wir eine BASIC-Zeile einer Bildschirmzeile anpassen und nicht mehr als 40 Zeichen pro Zeile verwenden.

Merke:

- **Anweisung: PRINT** in Verbindung mit einem String bewirkt die Ausgabe des Strings
- Ein Programm besteht aus Programmzeilen
- Eine Programmzeile besteht aus Zeilennummern (n zwischen 0 und 65529), der Anweisung (spezielle BASIC-Anweisung) und aus eventuell noch weiteren Anweisungen, die durch einen Doppelpunkt voneinander getrennt sind.
- Mit Betätigen der ENTER-Taste werden die Anweisungen der Programmzeilen nicht ausgeführt, sondern die Programmzeilen gespeichert.
- Mit Betätigen der STOP-Taste werden BASIC-Programme angehalten. Sie können an gleicher Stelle mit der Taste  fortgesetzt werden.
- Ein Druck auf die BRK-Taste unterbricht jedes laufende Programm. Die Programmzeile, in der die Unterbrechung stattfand, wird angezeigt.
  Beispiel: BREAK IN 20
- **Anweisung: CONT**
  Format: CONT
  Bemerkung: CONT setzt ein mit BRK-Taste unterbrochenes Programm an gleicher Stelle fort.

  Beispiel:
  ```
  BRK-Taste betätigen
   ⋮
  BREAK IN 20
  OK
  >_
  CONT
   ⋮
  ```
- **Anweisung: RUN**
  Format: RUN [Zeilennummer]
  Bemerkung: RUN startet die Programmausführung bei der niedrigsten oder angegebenen Zeile

  Beispiel: RUN oder RUN 20

- **Anweisung: LIST**
  Format: LIST [Zeilennummer]
  Bemerkung: Das im Speicher befindliche Programm wird beginnend mit der niedrigsten oder der angegebenen Zeilennummer aufgelistet. Die Anzahl der mit einer LIST- Anweisung ausgegebenen Zeilen beträgt vorläufig 10, sie kann auch mit dem LINES-Kommando verändert werden. Das Listen kann man durch Drücken der BRK-Taste unterbrechen.
  Beispiel: LIST oder LIST 100

- **Anweisung: GOTO**
  Format: GOTO Zeilennummer
  Bemerkung: GOTO ist eine unbedingte Sprunganweisung zur angegebenen Zeilennummer.
  Beispiel: 10 PRINT “HALLO“
  20 GOTO 10

Übungen

1. Welche zwei wesentlichen Aufgaben der ENTER-Taste sind zu unterscheiden?
2. Wie viele Anweisungen kann man in eine Programmzeile schreiben?
3. Wie kann man ein BASIC-Programm unterbrechen und wieder fortsetzen?

4.1.4. BASIC-Anweisungen REM, NEW, INPUT, LINES

Die Anweisungen REM, NEW, INPUT und LINES

In diesem Kapitel lernen wir weitere Anweisungen kennen. Danach sind Sie in der Lage, kleine zuverlässige Rechenprogramme selbst zu erstellen. Die ersten drei Anweisungen sind schnell erklärt.

Die Anweisung REM ist abgeleitet vom englischen Wort "remark", das übersetzt Bemerkung heißt. Hinter der Anweisung REM können wir für uns zur Information einen beliebigen Kommentar schreiben. Dieser wird gespeichert und mit der Anweisung LIST auch immer ausgegeben. Im Programmablauf werden die Kommentare jedoch ignoriert, d. h. sie haben nicht den geringsten Einfluss auf die Programmausführung. Die Anweisung ist also nur eine Gedächtnisstütze für den Programmierer.

Ein Beispiel:

```
10 REM DIESES PROGRAMM MACHTS NICHTS
20 PRINT "NICHTS"
```

Insbesondere in größeren Programmen ist das Einfügen von Kommentaren sehr nützlich und wird oft angewendet. Sie können für REM auch ein Ausrufezeichen benutzen:

```
10 ! KOMMENTAR
```

Lassen Sie nun das Programm mit dem Kommando RUN ablaufen und sehen Sie es sich mithilfe des LIST-Kommandos wieder an!
Geben Sie ein:

```
NEW
```

NEW löscht alle im Speicher befindlichen Programme und Variablen. Der Computer befindet sich nun im gleichen Zustand wie nach dem BASIC-Kaltstart. Überzeugen Sie sich mithilfe des Kommandos LIST. Sehen Sie:
Keine einzige Programmzeile ist mehr im Speicher zu finden.

Rechenprogramme

Geben Sie bitte ein:

```
10 REM DAS PROGRAMM BERECHNET DAS QUADRAT EINER ZAHL
20 INPUT ZAHL
30 PRINT ZAHL*ZAHL
```

Starten Sie das Programm mit RUN!

Kommt ein Fragezeichen auf Ihrem Bildschirm zum Vorschein, so ist dies keine Fehlermeldung, sondern der Computer ist, nachdem er die Zeile 10 ignoriert hat, zur Abarbeitung der Anweisung INPUT gelangt.

Die Anweisung INPUT unterbricht den Ablauf des Programms und wartet auf eine Eingabe. Geben wir z. B. ein:

```
3
```

Mit dem Druck auf die ENTER-Taste wird die Eingabe beendet und der Computer legt den eingegebenen Wert (3) unter der hinter INPUT stehenden Variablen (ZAHL) ab. Von nun kann man den eingegebenen Wert im Programm nach Belieben durch Benutzen der Variablen zu Rechnungen verwenden. So wird in unserem Beispiel in der Zeile 30 das Quadrat des eingegebenen Wertes berechnet und ausgegeben.

Schauen wir uns nun ein Programm zur Kreisberechnung an, das nach Eingabe des Radius R den Umfang U und die Fläche F eines Kreises ausgibt. Dazu müssen wir noch wissen, dass in unserem BASIC-Interpreter die Konstante PI=3.14159 bereits abgespeichert ist.

Löschen Sie vorher das alte Programm mit NEW und dem Bildschirm mit CLS.

```
10 REM KREISBERECHNUNG
20 CLS
30 INPUT R
40 U=2*PI*R
50 F=PI*R*R
60 PRINT U:PRINT F
```

Probieren Sie das Programm aus!

Nun lassen Sie sich bitte das Programm mit der Anweisung LIST auflisten. Sie können aber auch anders listen. Geben Sie ein:

```
LINES 3
```

Nun werden Sie bei der Anweisung LIST nur drei Zeilen erhalten.

Leider können wir im Programmablauf nicht feststellen, für welche Variable wir einen Wert eingeben. Günstiger ist es, wenn sich unser Computer während des Programmablaufes beider INPUT-Anweisung mit einer Eingabeaufforderung meldet. Geben Sie die Zeile 30 neu ein:

```
30 INPUT "R=";R
```

Lassen Sie nun das Programm ablaufen. Sie sehen, ein in Anführungszeichen gesetzter String direkt nach der INPUT-Anweisung wird im Programmablauf entsprechend ausgedruckt. Dieser nach Belieben einzufügender String ist von der folgenden Variablen in jedem Fall durch ein " ; " abzutrennen. Weiterhin ist es auch möglich, mit einer INPUT-Anweisung mehrere, durch Komma getrennte Variablen einzulesen. Auch unsere Eingabeanzeige können wir noch komfortabler gestalten. Dazu sollten wir wissen, dass mit einer PRINT-Anweisung mehrere, durch Semikolon getrennte Variablen, Zahlen und Strings ausgegeben werden können.

```
60 PRINT "U=";U:PRINT "F=";F
```

Probieren Sie das verbesserte Programm aus!

Analysieren wir wiederholend die Zeile 60.

Der in Anführungszeichen stehende String U= wird unverändert ausgegeben. Das darauf folgende Semikolon bewirkt in Verbindung mit der PRINT-Anweisung, dass die nächste PRINT-Anweisung auf dem Bildschirm unmittelbar folgt. U ist eine Variable, also wird der Wert von U ausgegeben. Der Doppelpunkt trennt die erste von der zweiten PRINT-Anweisung.

Merke:

- Werden die hinter PRINT stehenden Variablen, Zahlen und Strings durch ";" voneinander getrennt, so erfolgt Ausgabe auf Ausgabe ohne Zwischenraum (bzw. bei Zahlen mit je einem Leerzeichen Abstand hinter der Zahl und einem Vorzeichenfeld vor der Zahl) auf dem Bildschirm.
- Eine Trennung durch Komma bewirkt dagegen ein tabellarisches Ausgabeformat.
- **Anweisung: REM**
  Format: REM Kommentar
  Bemerkung: REM ist eine Kommentarkennzeichnung und wird im Programmablauf ignoriert. Statt REM kann auch "!" verwendet werden.
  Beispiel: 10 ! /COM
- **Anweisung: NEW**
  Format: NEW
  Bemerkung: NEW löscht alle im Speicher befindlichen Programme und Variablen
  Beispiel: NEW

- **Anweisung: INPUT**
  Format: INPUT ["String";] Variable [,Variable . . .]
  Bemerkung: INPUT bewirkt eine Anforderung von Daten für das Programm. Es ist möglich, Werte mehreren, durch Komma voneinander getrennten Variablen mit einer INPUT-Anweisung zuzuordnen. Wird in die INPUT-Anweisung eine Stringkonstante aufgenommen, so wird diese bei der Eingabeaufforderung mit ausgegeben; ansonsten erscheint ein Fragezeichen. Erfolgen nicht die vereinbarten Eingaben (z. B. String anstatt numerischer Werte), so meldet der Computer "? REDO FROM START" und die Eingabe muss wiederholt werden. Werden zu wenige Daten eingegeben, erfolgt eine wiederholte Eingabeaufforderung "??". Werden zu viele Daten eingegeben, erfolgt die Ausgabe "EXTRA IGNORED" und die Programmabarbeitung wird mit den benötigten Werten fortgesetzt.

  Beispiel:
  ```
  10 REM UMFANG EINES RECHTECKES
  20 INPUT "SEITENLAENGEN"; A,B
  30 U=2*(A+B)
  40 PRINT U
  ```

- **Anweisung: LINES**
  Format: LINES Zahl
  Bemerkung: LINES legt die Anzahl der auf einmal auszugebenden Zeilen beim Listen fest

  Beispiel: LINES 25

Übungen

1. Übernehmen Sie das Beispielprogramm aus der Zusammenfassung zur Anweisung INPUT. Verbessern Sie die Ausgabefähigkeit der Ausgabe, indem Sie die Möglichkeiten einer Stringvariable in Verbindung mit der PRINT-Anweisung nutzen.
2. Erweitern Sie das Programm von Übung 1 so, dass auch die Fläche des Rechteckes ermittelt und ausgegeben wird.
3. Erstellen Sie ein Programm, das aus dem Grundwert und dem Prozentsatz den Prozentwert ermittelt.
   (Prozentwert / Prozentsatz = Grundwert / 100)
4. Erweitern Sie das Programm von Übung 3 so, dass auch die Summe vom Grundwert und Prozentwert ausgegeben wird.

4.1.5. For-Schleifen (BASIC-Anweisungen FOR...TO...STEP...NEXT)

Die FOR-NEXT-Schleife und ein neuer PRINT-Kniff

Es besteht in der Praxis oft die Notwendigkeit, Programmteile mehrfach, z. B. für die Ermittlung mehrerer Funktionswerte, abzuarbeiten. Das würde sich auf Grund der uns bisher bekannten Anweisungen sehr mühsam gestalten. Wir müßten mit der INPUT-Anweisung Wert für Wert eingeben und das Programm immer neu ablaufen lassen. Für solche Probleme können wir jedoch die FOR-NEXT-Schleife benutzen.

Angenommen, wir möchten uns die Werte der Quadratzahlen von 1 bis 25 ausgeben lassen. Wir schreiben folgendes Programm:

```
10 FOR I=1 TO 25
20 PRINT I*I
30 NEXT I
```

Im einzelnen bedeutet das:

In der Zeile 10 wird eine Variable I vereinbart, die als „Laufvariable" dient und Werte von 1 bis 25 annimmt.

In der Zeile 20 steht die Druckanweisung für die Größe I^2.

Der Anweisungsteil in der Zeile 30 schließt den Zyklus ab. Falls I kleiner ist als 25, so wird die Laufvariable I jeweils um 1 erhöht. Für I=25 bewirkt dieser Befehl ein Verlassen der FOR- NEXT-Schleife.

Die Abarbeitung des Programms wird in der nächsten Zeile fortgesetzt. Lassen Sie das Programm ablaufen!

Das "I" in der Zeile 30 können wir auch weglassen. Mit der Anweisung NEXT sucht sich der Computer immer die letzte FOR-Anweisung. Die FOR-NEXT-Schleife erweist sich also als recht praktisch. Wie ist das aber, wenn wir nicht nur ganze Zahlen berechnen wollen, sondern z. B. auch das Quadrat von 1.5, 2.5 usw. benötigen?

Wir verändern die Zeile 10 wie folgt:

```
10 FOR I=1 TO 25 STEP 0.5
```

Hier wurde zusätzlich eine Schrittweite vereinbart. Das heißt, die Variable I wird nun pro Schleifendurchlauf nicht mehr um 1, sondern um die angegebene Schrittweite 0.5 erhöht. Probieren Sie das Programm mit der veränderten Zeile 10 aus!

Sie werden enttäuscht feststellen, dass die Werte nicht alle auf den Bildschirm passen. Hier gibt es eine kleine Hilfe. Verändern Sie die Zeile 20 wie folgt:

```
20 PRINT I * I,
```

Starten Sie das Programm! Sie sehen, das Komma tabelliert die Ausgaben in drei Bereiche zu je 13 Zeichen.

Wir können auch zwei oder mehrere Schleifen ineinander verschachteln. Probieren Sie folgendes Beispiel aus:

```
10 FOR A=1 TO 3
20 FOR I=1 TO 4
30 PRINT "AEUSSERE SCHLEIFE";A; "INNERE SCHLEIFE";I
40 NEXT I
50 NEXT A
```

Die Befehlszeilen 40 und 50 kann man auch zusammenfassen zu:

```
40 NEXT I,A
```

Nach dieser Vereinbarung ist jedoch die Zeile 50 zu löschen! Achten Sie unbedingt auf die Reihenfolge der Variablen – die Variable der innersten Schleife steht als erste hinter NEXT usw.

Merke:

- **Anweisung: PRINT**
  Format: PRINT [Ausdruck] [Trennzeichen] …
  Bemerkung: PRINT ist eine Ausgabeanweisung. PRINT allein bewirkt die Ausgabe einer Leerzeile. Als Ausdruck können der Anweisung folgen:
  1. Konstanten
  2. Variablen
  3. Ausdrücke (Verknüpfungen von Konstanten und Variablen)

PRINT bewirkt die Ausgabe dieser Konstanten, Variablen, Ergebnisse der Ausdrücke und Strings. Man kann mehrere Ausdrücke hinter eine PRINT-Anweisung schreiben. Diese müssen, mit Ausnahme der String-Konstanten, durch Trennzeichen getrennt sein. Die Trennzeichen Komma und Semikolon haben eine Formatierungsfunktion. Gleiches gilt auch für die Platzierung einer Ausgabe einer PRINT-Anweisung, wenn die vorhergehende mit einem Komma bzw. Semikolon abgeschlossen wurde. Steht am Ende der zuletzt abgearbeiteten PRINT-Anweisung kein Trennzeichen, so erfolgt die Ausgabe der aktuellen PRINT-Anweisung in der nächsten Zeile. Steht zwischen zwei Ausdrücken ein " , ", so erfolgt die Ausgabe tabelliert in 3 Bereichen mit je 13 Zeichen.

Sind die Ausdrücke durch ein " ; " voneinander getrennt, so folgt Ausgabe auf Ausgabe hintereinander. Handelt es sich bei der Ausgabe um eine Zahl, so wird hinter dieser ein Leerzeichen Abstand zur nächsten Ausgabe gelassen. Vor jeder Zahl finden wir das Vorzeichenfeld. Ist die Zahl positiv, so wird das Leerzeichen wie gewohnt weggelassen und es erscheint somit ein weiteres Leerzeichen vor der Zahl. Handelt es sich bei den durch Semikolon getrennten Ausdrücken um Strings, so werden diese ohne Abstand hintereinander ausgegeben.

Das Anweisungswort PRINT kann auch durch ein "?" eingegeben werden. Dieses wird beim nächsten Auflisten des Programms durch "PRINT" ersetzt.

Beispiel: PRINT 1; -2; "D"; "3"

```
10 INPUT X
20 PRINT "DAS QUADRAT VON";X; "IST"; X*X
```

- **Anweisung: FOR … TO … STEP … NEXT …**
  Format: FOR Variable = Anfangswert TO Endwert STEP Schrittweite
  Bemerkung: Die Anweisung FOR … TO … NEXT erzeugt eine Programmschleife, in der die zwischen diesen Anweisungsteilen stehenden Programmzeilen mehrfach abgearbeitet werden. Beim ersten Durchlauf erhält die Laufvariable den Anfangswert. Nach Abarbeitung des NEXT-Befehls wird der Wert der Laufvariablen um die Schrittweite erhöht. Das wiederholt sich so lange, bis der Endwert erreicht ist. Die Schrittweite kann sowohl positiv als auch negativ sein. Wird keine Schrittweite angegeben, so wird diese automatisch zu +1 gesetzt. In der NEXT-Anweiung kann der Variablenname auch weggelassen werden, da sich diese Anweisung immer auf die davor liegende FOR-Schleife bezieht.Der NEXT-Abschluss mehrerer ineinandergeschachtelter FOR-Schleifen kann in einer NEXT-Anweiung zusammengefasst werden. Dabei folgen der Anweisung NEXT von links nach rechts die durch Komma voneinander getrennten Variablen der inneren bis zur äußeren Schleife.
  Beispiel:

```
10 FOR A=8 TO 3 STEP -0.5
20 PRINT A
30 NEXT
```

Übungen

1. Geben Sie ein:

```
10 FOR I=0 TO 25
20 PRINT I
30 NEXT
```

Wie würde sich die Ausgabe verändern, wenn Sie die Programmzeile 20 mit einem "," oder einem ";" beenden würden?

Überprüfen Sie Ihre Antwort am KC 85.

2. Verändern Sie die Zeile 20 von Übung 1 wie folgt:

```
20 PRINT I; "    " ;
```

(Geben Sie zwischen den beiden Anführungszeichen vier Leerzeichen ein.) Erklären Sie die veränderte Ausgabeformatierung!

3. Verändern Sie das Programm der Übung 2 so, dass es nicht mehr in Einerschritten, sondern in ¼ Schritten von 0 bis 25 zählt!

4. Probieren Sie folgendes Programm aus:

```
10 FOR I=20 TO 40
20 PRINT "I=";I, "I * I="; I*I
30 NEXT
```

Was würde passieren, wenn Sie das Komma in der Zeile 20 durch ein Semikolon ersetzen?

5. Verändern Sie das in Übung 4 aufgelistete Programm so, dass es I^3 in der gleichen Weise ausgibt wie I und I^2 !

4.1.6. Vergleiche und BASIC-Anweisungen IF, THEN, ELSE, END

Die IF-Anweisung und die logischen Operatoren

Der KC 85 kann für Sie auch Entscheidungen treffen. Selbstverständlich müssen Sie ihm dazu erst die Entscheidungskriterien, d. h. in welchem Fall er was zu tun hat, mitteilen. Dies machen Sie im Programm mithilfe der IF-Anweisung. Schauen wir uns dazu ein keines Beispiel an. Hier soll der Computer eine erteilte Note, die in das Programm eingegeben wird, mit gut oder schlecht bewerten. Dabei werden die Noten 1 und 2 pauschal mit gut und die anderen mit schlecht bewertet. Geben Sie ein:

```
10 INPUT A
20 IF A=1 OR A=2 GOTO 50
30 PRINT "SCHLECHT"
40 END
50 PRINT "GUT"
60 END
```

Mit folgendem Kommentar ist das Programm sofort verständlich.

Zeile 20 IF A=1 OR A=2 GOTO 50

Übersetzung: Wenn A=1 oder A=2 ist, so gehe zur Zeile 50, ansonsten arbeite die nächste Zeile ab.

Testen Sie das Programm!

In diesem Programm wird erstmals die Anweisung END verwendet. Sie schließt den Programmlauf ab und kann prinzipiell an jeder Stelle des Programms stehen. END am Ende eines Programms ist nicht erforderlich, da sich nach Abarbeitung der letzten Programmzeile BASIC automatisch mit "OK" meldet. Die IF-GOTO-Anweisung funktioniert verallgemeinert wie folgt:

```
IF x GOTO n
```

Wenn der Ausdruck x wahr ist, so gehe zur Programmzeile n, ansonsten arbeite die nächste Zeile ab.

Nebenbei haben Sie eben Bekanntschaft mit einem der Booleschen Operatoren, dem logischen ODER (Anweisung OR) gemacht. Darüber hinaus verfügt unser BASIC-Interpreter über zwei weitere logische Operatoren, das UND und die Negation (Anweisung AND und NOT). Durch geeignete Kombinationen dieser Operatoren können wir weitestgehend alle Probleme der Booleschen Algebra bewältigen. Wir können diese Operatoren nach logischem Ermessen analog der Umgangssprache einsetzen. Hierzu merken wir uns folgende einfache Regel:

Ein Ausdruck der aus einer logischen Verknüpfung (in unserem Beispiel OR) zweier Bedingungen (in unserem Beispiel A=1, A=2) besteht, ist in folgenden Fällen wahr:

Tabelle 46: Logische Operatoren in BASIC

Operator	Übersetzung/Wahrheitskriterium
AND	UND / beide Bedingungen treffen gleichzeitig zu.
OR	ODER / mindestens eine von beiden Bedingungen trifft zu.
NOT	NOT bezieht sich nur auf einen Ausdruck bzw. auf eine Bedingung, die Negation NOT ist genau dann wahr, wenn dieser Ausdruck falsch ist bzw. Die Bedingung nicht zutrifft.

Der Boolesche Zustand für „falsch“ wird dabei durch den Wert 0 ausgedrückt, der Zustand für Wahr mit dem Wert -1.

Es gibt noch eine zweite Variante der IF-Anweisung. Diese sieht verallgemeinert wie folgt aus:

```
IF x THEN Anweisung
```

Das bedeutet inhaltlich: Wenn der Ausdruck x wahr ist, dann führe die hinter THEN stehende Anweisung aus, ansonsten ignoriere die Anweisung und fahre mit der Abarbeitung der nächsten Zeile fort. Statt einer Anweisung kann auch eine Zeilennummer stehen, zu der bei erfüllter Bedingung gesprungen wird. GOTO kann hierbei entfallen. Der Anweisung nach THEN können weitere, durch Doppelpunkt getrennte Anweisungen folgen. Diese werden jedoch nur dann ausgeführt, wenn die vorstehende Bedingung erfüllt wurde.

Wir schreiben nun unser Beispielprogramm so um, dass die zweite Variante der IF Anweisung zum Einsatz kommt. Geben Sie ein:

```
10 INPUT A
20 IF A=1 OR A=2 THEN PRINT "GUT"
30 IF NOT (A=1 OR A=2) THEN PRINT "SCHLECHT"
```

Beachten Sie am Beispiel den logisch einfachen, für die Programmierung sehr wichtigen Einsatz der Booleschen Operatoren. Weiterhin haben wir noch die Möglichkeit, eine alternative Anweisung in die IF-Anweisung einzubauen. Dies geschieht mithilfe des Anweisungsteils ELSE (übersetzt: ansonsten), der an die uns schon bekannte Formen der IF- Anweisung angehängt wird.

Damit können wir unser Beispielprogramm wie folgt weiter vereinfachen:

```
10 INPUT A
20 IF A=1 OR A=2 THEN PRINT "GUT": ELSE PRINT "SCHLECHT"
```

Beachten Sie, dass vor dem Anweisungsteil ELSE ein Doppelpunkt zu setzen ist.

Die Vergleichsoperatoren

Der Interpreter verfügt über folgende Vergleichsoperatoren:

= gleich

< kleiner als

> größer als

<= kleiner oder gleich

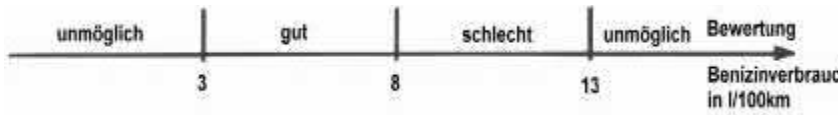
>= größer oder gleich

<> ungleich

Die einfache Handhabung dieser Vergleiche veranschaulichen wir uns am besten an einem Beispielprogramm.

Dieses Programm bewertet den Benzinverbrauch eines PKW wie folgt:

Benzinverbrauch / 100 km	Bewertung
Weniger als 3l oder mehr als 13 l	unmöglich
3l und mehr, aber weniger als 8l	gut
Von 8l bis 13l	schlecht

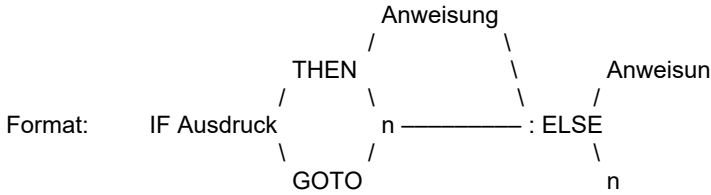


Geben Sie nun das Programm ein:

```
10 INPUT "BEZINVERBRAUCH IN L/100 KM =";A
20 PRINT "BEWERTUNG:";
30 IF A>=3 AND A<8 GOTO 100
40 IF A>=8 AND A<=13 GOTO 200
50 PRINT "UNMÖGLICH": END
100 PRINT "GUT":END
200 PRINT"SCHLECHT"
```

Merke:

- **Anweisung: IF, THEN**
  oder IF, GOTO



  Bemerkung: IF ist eine bedingte Anweisung. Ist der Ausdruck wahr bzw. das Resultat des Ausdrucks nicht Null, so werden die Anweisungen THEN oder GOTO ausgeführt. Steht hinter THEN eine Anweisung, so wird diese ausgeführt. Stehen hinter dem Ausdruck THEN n oder GOTO n, so wird das Programm bei der angegebenen Zeilennummer n fortgesetzt.
  Ist das Resultat jedoch falsch bzw. das Resultat des Ausdrucks Null, so werden die mit GOTO oder THEN verbundenen Anweisungen ignoriert und die ELSE-Anweisung, falls angegeben, ausgeführt. Andernfalls fährt das Programm mit der Abarbeitung der nächsten Zeile fort. Der Ausdruck kann eine Zeichenkette, eine Variable oder ein arithmetischer Ausdruck sein.

  Beispiel:
```
⋮
110 INPUT A$
120 IF A$= "JA" THEN P=P+2
⋮
```

- **Anweisung: END**
  Format: END
  Bemerkung: Die Anweisung kann an beliebigen Stellen des Programms abgespeichert werden und beendet die Programmabarbeitung.

- Für Vergleiche stehen folgende Operatoren zur Verfügung:

=	gleich
<	kleiner als
>	größer als
<= bzw. =<	kleiner oder gleich
>= bzw. =>	größer oder gleich
<>	ungleich

- Die logischen oder Booleschen Operatoren AND, OR und NOT:

  Bei Verknüpfung der Ausdrücke A und B mit den Booleschen Operatoren ergeben sich neue Ausdrücke mit folgenden Wahrheitswerten (wahr = 1; falsch = 0):

A	B	NOT A	A OR B	A AND B
0	0	1	0	0
0	1	1	1	0
1	0	0	1	0
1	1	0	1	1

Übungen

1. Formen Sie das Benzinverbrauchsprogramm analog dem Noten-Programm so um, dass die zweite Variante der IF-Anweisung (IF . . .THEN Anweisung) zur Anwendung kommt! (Lösung Kapitel 4.1.24)
2. Wieso ergeben sich für die Ausdrücke A und NOT NOT A die gleichen Wahrheitswerte?
3. Wieso ergeben sich für die Ausdrücke NOT (A OR B) und (NOT A) AND (NOT B) die gleichen Wahrheitswerte?
4. Die IF-Anweisung wird bei iterativen Näherungsverfahren zum Test der Abbruchbedingungen benutzt.

   Schreiben Sie ein Programm zur Berechnung der Quadratwurzel nach Newton. Hierbei ergibt sich

$$x_{i+1} = x_i - \frac{x^2_i - a}{2x_i}$$

   Die Abbruchbedingung lautet:

$$| x_{i+1} - x_i | < 10^{-3}$$

   (Lösung Kapitel 4.1.24)

4.1.7. Mathematische Funktionen und DEF, RND, RANDOMIZE

Die Operatoren in ihrer Hierarchie

Als letzte Operation sei die Potenzierung a = b^e, in der Computerschreibweise A = A ^ C, genannt. Damit haben wir alle Operatoren unseres BASIC-Interpreters kennengelernt. Beim Umgang mit diesen müssen wir die Reihenfolge ihrer Abarbeitung, d. h. die Hierarchie, in der die Operatoren geordnet sind, beachten. In diese hierarchische Ordnung sind nicht nur, wie in Kapitel 4.1.2 beschrieben, die vier Grundrechenarten, sondern alle Operatoren einbezogen. Stehen mehrere Operatoren in einem Ausdruck, so ist die Reihenfolge ihrer Abarbeitung wie folgt hierarchisch geordnet:

1. Klammern
2. Exponenten ^
3. Negative Vorzeichen
4. * , /
5. + , –
6. Vergleichsoperatoren: = , < , > , <= , >= , <>
7. NOT
8. AND
9. OR

Gleichartige Operatoren werden von links nach rechts abgearbeitet. Da wir diese hierarchische Abarbeitung von außen nicht verfolgen können, versetzen wir uns gedanklich in den Computer und verfolgen an einer Beispielaufgabe die einzelnen Teilschritte der Bearbeitung im Zeitlupentempo.

Aufgabe:								Arbeitsschritte:
3.1	E2	* 2 +	(6 – 5)	* 7	–	12/3		Klammerinhalt ermitteln
3.1	E2	* 2 +	1	* 7	–	12/3		Exponenten berechnen
310		* 2 +	1	* 7	–	12/3		
620		+	1	* 7	–	12/3		Multiplikation und Division
620		+		7	–	12/3		werden von links nach rechts
620		+		7	–	4		ausgeführt
627					–	4		Addition und Subtraktion
623								ebenfalls von links nach rechts

Die mathematischen Funktionen

Wie bereits im Kapitel 4.1.2 erwähnt, besitzen wir durch den BASIC-Interpreter mit unserem KC 85 einen wissenschaftlichen Rechner, der über alle einschlägigen Standardfunktionen verfügt. Bevor wir zu den konkreten mathematischen Funktionen des Computers kommen, werden wir kurz, aber anschaulich, den Begriff der mathematischen Funktion wiederholen.

Aufbau als Gleichung:

y	=	f	(x)
Funktionswert		Funktion	Argument

Eine Funktion f ist die Zuordnungsvorschrift, die dem Argument x eindeutig einen Funktionswert y zuordnet.

Am Beispiel: $y = f(x) = \sqrt{x}$

$f(4) = \sqrt{4} = 2$

Argument → f → Funktionswert

Dabei sind insbesondere immer Beschränkungen des Definitionsbereiches (Bereich der Argumente) und des Wertebereichs (Bereich der Funktionswerte) zu beachten. Diese Beschränkungen resultieren einmal aus der Definition der Funktion selber, zum anderen aus rechentechnischen Gegebenheiten des Computers. Die computerspezifischen Einschränkungen sind in folgender Vorstellung der mathematischen Funktionen des KC 85 vermerkt.

Tabelle 47: Mathematische Funktionen im KC-BASIC

Mathematische Standardfunktionen des KC 85	**Entsprechende mathematische Funktion**	**Bemerkung**
ABS(X)	|x|	Betrag von x
ATN(X)	arctan x	Resultat im Bogenmaß
COS(X)	cos x	X im Bogenmaß
EXP(X)	e^x	X <= 87.3365
INT(X)	ganzer Teil von x	
LN(X)	ln x	X > 0
SGN(X)	Signumfunktion	SGN (X)= { -1 für X < 0; 0 für X = 0; 1 für X > 0
SIN(X)	sin x	X im Bogenmaß
SQR(X)	$\sqrt{x}$	X >= 0
TAN(X)	tan x	X im Bogenmaß

Das Argument der Funktion muss stets in Klammern geschrieben werden. Testen Sie nun die uns zur Verfügung stehenden Funktionen! Beginnen Sie mit folgenden einfachen Beispielen:

```
PRINT SQR(81)
```

oder

```
10 INPUT A
20 PRINT SQR(A)
```

etc.

Durch geeignete Kombinationen der Standardfunktionen können Sie alle geläufigen mathematischen Funktionen realisieren. So ergibt sich z. B. der Sekans aus:

sec x = 1/cos x

In der BASIC-Übersicht ab Seite 367 finden Sie eine Liste solcher hergeleiteten mathematischen Funktionen.

Der BASIC-Interpreter verfügt über die Konstante π mit dem Wert 3.14159. Die Konstante kann durch die Eingabe "PI" genutzt werden:

```
10 INPUT E
20 PRINT "A="; PI*R*R
```

Funktionen definieren

Mit der Anweisung DEF FN können wir umfangreiche Formeln (z. B. die eben beschriebenen hergeleiteten Funktionen) als Funktion definieren und unter einem selbst gewählten Funktionsnamen wieder aufrufen. Dies erleichtert die Schreibarbeit insbesondere bei längeren Formeln, die häufig im Programm vorkommen. Der Name der zu definierenden Funktion ist FN, gefolgt von einer zulässigen Variablenbezeichnung. Machen Sie sich anhand des folgenden kleinen Programms mit der Wirkungsweise der Anweisung vertraut:

```
10 DEF FNQ(X) = 2* X * X – X/2
20 INPUT X
30 PRINT FNQ(X)
40 END
```

Hier haben wir die Funktion FNQ (x) =2x² – x/2 definiert. Das Programm berechnet den Funktionswert des einzelnen Argumentes.

Zufallszahlenberechnung

Die Funktion RND (X) liefert eine Zufallszahl im Bereich 0 ≤ RND (X) < 1

Für X > 0 erscheinen völlig zufällige Zahlen.

Für X = 0 wird der Wert der letzten Zufallszahl noch einmal ausgegeben.

Für X > 0 gibt es für jedes X eine bestimmte Zufallszahl.

Außerdem wird der Zufallsgenerator neu initialisiert. Überzeugen wir uns am Beispiel.

Das folgende Programm gibt nach den letzten Erläuterungen also "völlig" zufällige Zahlen aus. Lassen Sie es mehrfach ablaufen!

```
10 FOR I = 1 TO 5
20 PRINT RND(1)
30 NEXT
```

Diese Funktion können wir zum "Würfeln" benutzen. Probieren Sie das Programm einmal mit folgender Veränderung aus:

```
20 PRINT 6 * RND(1)
```

Die fünf Würfelergebnisse haben nun aber alle die unschöne Eigenschaft, Dezimalstellen hinter dem Komma zu besitzen. Diese können wir beseitigen, indem wir nur den ganzen Teil der Zahl verwenden. Dazu bedienen wir uns der Funktion INT:

```
20 PRINT INT(6 * RND(1))
```

Nach einigen Versuchen ist ihnen sicher aufgefallen, dass bei unseren Würfelergebnissen nie eine 6 "gewürfelt" wurde. Verändern Sie deshalb die Zeile 20 ein letztes Mal wie folgt:

```
20 PRINT INT(6 * RND(1))+1
```

Spielen Sie dieses oder anderes Zufallsspiel öfter, werden Sie bemerken, dass nach dem Start des BASIC-Interpreters jeweils die gleiche Folge von Zufallszahlen erscheint. Dies liegt in der Funktionsweise des Zufallsgenerators begründetet. Beim Start des BASIC-Interpreters erfolgt die Initialisierung des Zufallsgenerators immer mit dem gleichen Wert. Die folgenden Zufallswerte werden ausgehend von diesem Wert stets in gleicher Folge berechnet. Die Wiederholung der stets gleichen Folge von Zufallszahlen können wir durch Einfügen der Programmzeile

```
5 RANDOMIZE
```

beheben. Durch die Anweisung RANDOMIZE erfolgt eine erneute Initialisierung des Zufallsgenerators mit einem zufälligen Anfangswert. Entsprechend den verschiedenen Anfangswerten liefert die RND-Funktion nun auch verschiedene Folgen zufälliger Zahlen.

Merke:

- Stehen mehrere Operatoren in einem Ausdruck, so werden diese der Reihenfolge nach, entsprechend der im Kapitel beschriebenen Hierarchie abgearbeitet.
- Das BASIC des KC 85 verfügt über folgende mathematische Standardfunktionen: ABS, ATN, COS, EXP, INT, LN, SGN, SQR, TAN. Das Argument mit einer Funktion muss stets in Klammern stehen.
- Die Konstante π ist mit PI=3.1459 gespeichert.
- Die Funktion RND erzeugt Zufallszahlen: $0 \leq x < 1$
- **Anweisung: DEF FN**
  Format: DEF FN Name (Verwendetes Argument) = Ausdruck
  Bemerkung: DEF FN definiert eine vom Anwender bestimmte Funktion. Der Funktionsname besteht aus FN, gefolgt von einem zulässigen Variablennamen. Dem Funktionsnamen folgt in Klammern das Argument der Funktion.

  Beispiel:
  ```
  10 DEF FNA(Z) = Z*Z – SIN(Z)
  20 INPUT Z
  30 PRINT FNA (Z)
  ```
- **Anweisung: RANDOMIZE**
  Format: RANDOMIZE
  Bemerkung: Mithilfe der RANDOMIZE-Anweisung wird der Zufallsgenerator neu initialisiert. Wiederholungen von Zufallszahlenfolgen sind damit weitestgehend ausgeschlossen.

  Beispiel:
  ```
  10 RANDOMIZE
  20 FOR I=1 TO 5
  30 PRINT INT(36* RND(1))+1
  40 NEXT I
  ```

Übungen

1. Warum hat unser Würfelspiel nach der zweiten Änderung keine 6 "gewürfelt"?
2. Schreiben Sie ein Programm zur Berechnung der längeren Seite c aus den beiden kürzeren Seiten a und b eines rechtwinkligen Dreiecks ABC. (Es gilt der Satz des Pythagoras $c^2 = a^2 + b^2$.) (Lösung im Kapitel 4.1.24)
3. Schreiben Sie ein Programm zur Berechnung der Lösungen x1 und x2 der gegebenen quadratischen Gleichung $x^2 + px + q = 0$.

   Lösungsvorschrift: $$x12 = \frac{p}{2} \pm \sqrt{\frac{p^2}{4} - q}$$

   (Lösung Kapitel 4.1.24)

4.1.8. EDIT, DELETE, KEY, KEYLIST, AUTO, RENUMBER)

EDIT

Die Anweisung EDIT wird uns in Zukunft eine große Hilfe bei der Korrektur von Programmen sein. Sie haben in Kapitel 4.1.3 erfahren, wie man fehlerhafte Programmzeilen überschreiben kann bzw. durch Eingabe der Zeilennummer löscht.

Mithilfe der Anweisung EDIT können wir nicht nur die gerade in der Eingabe befindliche Programmzeile, sondern auch bereits abgespeicherte Programmzeilen mithilfe der Tasten ⇦ , ⇨ sowie INS und DEL DEL-Taste korrigieren. Probieren Sie das erst einmal an folgendem Beispiel aus:

```
10 INPUT X
20 PRINT "X/2=";X/2
30 PRINT RND(2)
40 PRINT "ENDE"
```

Nachdem Sie das Programm ausprobiert und sicherlich schnell verstanden haben, geben Sie ein:

```
EDIT 20
```

Nun können Sie, wie vorhin beschrieben, innerhalb der Zeile 20 beliebig mit der dem Cursor nach rechts und links fahren. Mithilfe der Zweitbelegung der Tasten ⇦ und ⇨ lässt sich der Cursor am Zeilenanfang bzw. am Zeilenende platzieren. Wir werden die Ausgabe von Zeile 20 noch etwas aussagekräftiger machen, d. h. wir werden die Zeile korrigieren.

… "X/2=" … wollen wir ändern zu … "DIE HAELFTE VON X BETRAEGT" …

Dazu fahren wir mit dem Cursor auf das erste X der Zeile 20 und betätigen 4 mal die DEL- Taste. Wir sehen, das X und die drei nach X befindlichen Zeichen sind gelöscht. Nun geben wir mithilfe der Taste INS so viele Leerzeichen ein, wie Sie zur angegebenen Korrektur Zeichen benötigen. Danach tippen Sie den neuen Text ein. Mit einem Druck auf die ENTER- Taste ist die korrigierte Zeile gespeichert. Betätigen wir dagegen die Taste ⇩ , so wird die Korrektur nicht übernommen.

In beiden Fällen befinden wir uns nun jedoch in der nächsten Programmzeile, die wir ebenfalls, wie eben beschrieben, korrigieren können. Hier könnten wir z. B. RND(2) in RND(-1) ändern. Der EDIT-Modus wird durch Betätigung der BRK-Taste oder nach Übernahme der letzten Programmzeile verlassen.

Weiterhin können wir mit der Eingabe

```
DELETE x,y
```

Alle Programmzeilen eines Programms von Zeile X bis Zeile Y löschen. Überzeugen Sie sich, indem Sie eingeben:

DELETE 20,30

Mit LIST und RUN können Sie nun die Wirkung der Anweisung DELETE 20,30 kontrollieren.

Funktionstastenbelegung

Mithilfe der Funktionstasten können wir einzelne Anweisungen oder eine ganze Folge von Anweisungen mit einem Tastendruck eingeben und auch ausführen. Dazu müssen die Funktionstasten jedoch erst mithilfe der Anweisung KEY wie folgt belegt werden:

1. Eingabe
   KEY Funktionstastennummer
2. ENTER-Taste drücken
3. Eingabe
   Tastenfolge, mit welcher die Funktionstasten belegt werden soll
4. STOP-Taste drücken
5. ENTER-Taste drücken

Als Beispiel werden wir die Taste F1 mit der Anweisung RUN und belegen, sodass ein im Computer befindliches Programm durch Betätigen dieser Taste gestartet werden kann. Dazu sind folgende Eingaben erforderlich:

KEY 1

(ENTER-Taste drücken)

RUN 

(STOP-Taste und danach ENTER-Taste drücken)

Das Steuerzeichen der ENTER-Taste ist im KEY-Eingabemodus einfach durch Drücken der ENTER-Taste einzugeben. Mithilfe der eben programmierten Funktionstaste F1 können wir jetzt ein im Speicher befindliches BASIC-Programm mit einem Druck auf dieselbe starten. Damit erreichen wir im Programmierbetrieb eine Arbeitserleichterung von vorher vier (RUN ) zu jetzt einer F1 Tastenbetätigung. Mit Ausnahme der Umschaltfunktion SHIFT  und der Steuerfunktion STOP können wir alle auf der Tastatur in BASIC verfügbaren Steuer- und Editier-Funktionen, wie am Beispiel ENTER-Funktion beschrieben, durch die Funktionstasten ausführen.

Das heißt, wir können diese durch Zeichen im KEY-Eingabemodus als Tastenbelegung speichern und später durch Druck auf die entsprechende Funktionstaste ausführen. Die Summe aller Tastenbelegungen darf 143 Zeichen nicht übersteigen.

Mit der STOP-Taste wird während der Funktionstasteneingabe zwischen dem Editier- und dem Interpretiermodus umgeschaltet. Im Editiermodus sind die Steuerzeichen wir z. B. die Cursortasten normal wirksam und können zur Korrektur der Eingabe benutzt werden. ENTER beendet die Eingabe.

Vergleiche dazu auch die Erläuterungen zum gleichnamigen CAOS-Kommando auf Seite 47.

Um die Funktionstastenbelegung nicht nach jedem Einschalten des Gerätes erneut eingeben zu müssen, können wir diese abspeichern. (SAVE B900 B99B; siehe Seite 57). Damit steht uns die Funktionstastenbelegung jederzeit nachladbar als Maschinenprogramm zur Verfügung.

Möchten wir uns einen Überblick über alle möglichen Funktionstastenbelegungen verschaffen, so geben wir einfach die Anweisung

```
KEYLIST
```

ein und führen sie aus.

Der daraufhin auf dem Bildschirm erscheinenden Liste der belegten Funktionstasten. Es gibt 12 mögliche Funktionstastenbelegungen. Nach der oben beschriebenen Eingabe ist die Taste F1 mit "RUN  " belegt und die Tasten F2 bis FC sind noch nicht belegt. Die Nummerierung der letzten drei Funktionstastenbelegungen erfolgt hexadezimal (A, B, C hexadezimal entsprechen 10, 11, 12 dezimal).

Sollen diese Tasten jedoch belegt werden, so sind sie dezimal, also mit "KEY10", "KEY11" bzw. "KEY12" aufzurufen.

Die Ausgabe bzw. Ausführung der Funktionstastenbelegung erfolgt von F1 bis F6 durch Betätigen der entsprechenden Taste bzw. von F7 bis F12 durch die Zweitbelegung der Tasten F1 bis F6.

Zeilennummerierung

Beim Programmieren ist es notwendig, vor jede Programmzeile eine Zeilennummer zu schreiben. Mit dem Kommando AUTO kann man den Computer dazu bringen, die Zeilennummerierung selbst durchzuführen, sodass nur noch die Anweisungen selbst eingegeben werden müssen. Der AUTO-Modus wird durch Betätigen der BRK-Taste beendet. Wir können das Kommando AUTO auch durch die Form

AUTO i, j

konkretisieren. Hierbei ist i die erste zu erzeugende Zeilennummer und j legt den Zeilennummernabstand fest.

Testen Sie diese Programmierhilfe an einem kleinen Programm mit mehreren Zeilen.

Nachdem Sie sich nun über die Funktionsweise des Kommandos AUTO am Beispiel informiert haben, geben Sie bitte in das bestehende Programm weitere Programmzeilen ein, sodass möglichst eine "bunte, ungeordnete" Nummerierung entsteht. Eine solche "ungeordnete" Zeilennummerierung ergibt sich in der Praxis oft nach einer notwendigen Korrektur. Hier können wir leicht mit dem Kommando RENUMBER Ordnung schaffen. Geben Sie bitte ein:

RENUMBER

Sie sehen, dieses Kommando bewirkt eine neue Numerierung der Zeilen des Programms in 10er Schritten. Weitere Anwendungsmöglichkeiten des Kommandos RENUMBER entnehmen Sie bitte dem Abschnitt "Merke" dieses Kapitels.

Merke:

- **Anweisung: EDIT**
  Format: EDIT Zeilennummer
  Bemerkung: EDIT bringt die im Anweisungsaufruf benannte Zeile zu Anzeige und versetzt den Computer in den Korrekturmodus. In diesem ist es möglich, innerhalb der aufgerufenen Zeile mithilfe der Tasten  und  den Cursor beliebig zu bewegen. Weiterhin kann man durch die Tasten  bzw.  in der Programmzeile Zeichen lückenlos löschen bzw. Leerzeichen, welche man anschließend neu überschreiben kann, einfügen. Mithilfe der Taste  gelangt der Cursor in die folgende Programmzeile und man kann nun in dieser, wie eben beschrieben, korrigieren. Die zu korrigierende Zeilen werden durch Betätigen der ENTER-Taste übernommen. Den EDIT-Modus verlässt man durch die BRK-Taste.
  Beispiel: EDIT 20

- **Anweisung: DELETE**
  Format: DELETE Anfangszeile [, Endzeile]
  Bemerkung: Löschen aller Zeilen zwischen einschließlich Anfangs- und Endzeile
  Beispiel 1: DELETE 30
  Beispiel 2: DELETE 30, 100

- **Anweisung: AUTO**
  Format: AUTO [Zeilennummer [, Schrittgröße]]
  Bemerkung: AUTO erzeugt automatisch nach jedem Betätigen der ENTER-Taste eine Zeilennummer. Es wird mit der angegebenen Zeilennummer begonnen und diese jeweils um die angegebene Schrittweite erhöht.
  Werden keine Schrittgröße und keine Zeilennummer angegeben, so beginnt AUTO mit der Zeilennummer 10 und nummeriert im 10er Abstand. Stimmt eine erzeugte Zeilennummer mit einer schon im Programm vorhandenen überein, so wird dies durch ein *-Zeichen hinter der Zeilennummer gekennzeichnet.
  AUTO wird durch Betätigen der BRK-Taste beendet.

  Beispiel: AUTO 30, 5 erzeugt die Zeilennummern 30, 35, 40, 45, 50 usw.

- **Anweisung: KEY**
  Format: KEY Funktionstastennummer
  Bemerkung: Bei der Ausführung der KEY-Anweisung im angegebenen Format (Funktionsnummern von 1 bis 12 möglich) wird der Computer in einen Eingabemodus versetzt, in welchem der einzugebende String für die in der KEY-Anweisung benannte Funktionstaste definiert wird. In diesem Eingabemodus können, neben den üblichen alphanumerischen und Sonderzeichen, auch Zeichen für die auf der Tastatur verfügbaren Editier- und Steuerfunktionen durch Druck auf die jeweiligen Editier- und Steuertaste eingegeben werden. Zum Editieren der Eingabe kann durch Betätigung der STOP-Taste zwischen dem Interpretier- und dem Editiermodus umgeschaltet werden. Der Eingabemodus wird durch Betätigen der ENTER-Taste im Editiermodus beendet.
  Mit einem Druck auf die betreffende Funktionstaste werden nun stets der im KEY-Eingabemodus festgelegte String ausgegeben bzw. die darin als Steuer- und Editierzeichen enthaltenen Funktionen ausgeführt.

  Beispiel: KEY 2 (ENTER-Taste drücken)
  WINDOW:COLOR7,1:CLS:LIST (STOP + ENTER Taste)
  OK
  >_

- **Anweisung: KEYLIST**
  Format: KEYLIST
  Bemerkung: Mithilfe dieser Anweisung kann man die Funktionstastenbelegung auflisten.

  Beispiel:
  ```
  KEYLIST
  F1 : RUN↵
  F2 : WINDOW:COLOR7,1:CLS:LIST↵
  F3 : EDIT
  F4 : INPUT
  F5 : FOR I=
  F6 : NEXT
  F7 : CLS
  F8 : GOTO
  F9 : GOSUB
  FA : RETURN
  FB : INKEY$
  FC : ?CHR$
  ```

- **Anweisung: RENUMBER**
  Format: RENUMBER [Ab-alte-Zn , Bis-alte-Zn [,Ab-neue-Zn [, Schrittgröße]]
  Bemerkung: RENUMBER numeriert die in einem Programm vorkommenden Zeilen neu. Wird RENUMBER ohne Parameterangabe ausgeführt, so erfolgt die Zeilennummerierung beginnend mit der kleinsten Programmzeilennummer in 10er Schritten. RENUMBER darf nur in Verbindung mit einem BASIC-Programm verwendet werden, da es sonst zu einem modifizierten Zutand des Computers führt. Neustart erfolgt über RESET

  Beispiel: RENUMBER 12, 110, 20, 5
  Nummeriert Zeilen 12 bis 110 des alten Programms neu. Dabei wird die Zeile 12 zu angegebenen niedrigsten Zeile 20 der Neunummerierung und die weiteren Zeilennummern folgen in der angegebenen Schrittgröße 5.

Übungen

Keine, stattdessen ein Hinweis:

Nutzen Sie diese Programmierhilfen im Weiteren konsequent bei jeder sich bietenden Gelegenheit. Nach einer kurzen Eingewöhnungszeit möchten Sie den dadurch gebotenen Komfort bestimmt nicht mehr missen.

4.1.9. Retten und Laden (DEVICE, CLOAD, CSAVE, BLOAD)

Speichermedien (Tonband, Diskette, USB)

So, wie wir in Maschinensprache geschriebene Programme vom Recorder in den Computer laden, können wir auch in BASIC geschriebene Programme mithilfe des Interpreters durch die BASIC-Anweisung CLOAD in den Computer laden.

Da wir nicht nur daran interessiert sind, im Handel erworbene Programme, sondern auch selbst erstellte wiederholt zu nutzen, müssen wir diese vor dem Ausschalten des Gerätes auf unseren Speicher, das Tonband, "retten" (Englisch: save). Die Anweisung, die diesen Vorgang auslöst, heißt deshalb auch CSAVE.

Das Standard-Speichermedium des KC 85 ist das Tonband bzw. die Magnetbandkassette. Mit dem D004 kommt als weiteres Speichermedium die Diskette hinzu. Bei Verwendung eines Moduls M052 kann auch auf USB gespeichert werden. Ab CAOS 4.7 stehen im BASIC-Interpreter deshalb zusätzlich die Anweisungen DEVICE zur Umschaltung des Speichermediums, CHDIR zum Wechsel des Verzeichnisses und FILES zur Anzeige des Inhaltes auf dem Speichergerät zur Verfügung.

Wir retten ein Programm

Die Funktionsweise der beiden Anweisungen CLOAD und CSAVE verdeutlichen wir uns am besten an einem Beispiel. Schreiben Sie ein kleines Programm, das wir dann als Testprogramm retten und laden wollen. Schauen wir uns diesen Vorgang zunächst für das Medium Magnetband an. Das Programm legen wir unter einem selbst gewählten Namen ab. Der Computer unterscheidet die letzten 8 Buchstaben des Namens. Nennen wir unser Programm der Einfachheit halber TEST.

Bevor wir mit der eigentlichen Aufnahme beginnen, ist es ratsam, sich den Zählerstand des Recorders aufzuschreiben oder den Programmnamen mit Mikrofon auf die Kassette zu sprechen. Dies erleichtert das Suchen des Programms erheblich.

Geben Sie nun ein: (Aber noch nicht die ENTER-Taste drücken!)

```
CSAVE "TEST"
```

Schalten Sie den Recorder auf Aufnahme.

Drücken Sie nun die ENTER-Taste.

Nun wird das im Computer gespeicherte BASIC-Programm auf Magnetband aufgezeichnet. Das Auslesen des Programms erfolgt in Blöcken zu je 128 Bytes. Die angezeigten zweistelligen Zahlen stellen die Blocknummer der ausgegebenen Blöcke dar. Damit ergibt sich die Länge der Aufnahme, also die Anzahl der eingelesen Blöcke, aus der Länge des BASIC- Programms.

Nach dem Retten besteht die Möglichkeit, die Magnetbandaufzeichnung zu prüfen. Dazu erfolgt nach Übernahme des letzten Blockes die Ausschrift:

VERIFY ?(Y)

Mit dieser Ausgabe ist die Aufnahme beendet und Sie können den Recorder ausschalten. Wenn Sie die Überprüfung durchführen möchten, spulen Sie die Kassette auf den Programmanfang zurück, schalten den Recorder zur Wiedergabe ein und betätigen während des Pfeiftones die Taste "Y". Die eingelesenen Datenblöcke werden nun überprüft. Dabei werden der Programmname und die Nummer der Datenblöcke ausgegeben. Hinter jeder Datenblocknummer erscheint ein ">". Dieses Zeichen bestätigt die fehlerfreie Übernahme des Blockes.

Erscheint jedoch hinter der Blocknummer ein "?", so ist dieses als Fehlermeldung zu werten. Es trat also ein Fehler bei der Datenübernahme auf. Bei einem solchen Fehler können Sie nach dem Zurückspulen des Bandes den fehlerhaft gelesenen Block erneut einlesen. Dabei wird der Computer unverändert im VERIFY-Modus belassen. Liest man beim erneuten Lesen Blöcke ein, die vor dem fehlerhaften eingelesenen Block bereits richtig eingelesen wurden, so wird dies durch ein " * " verdeutlicht. Dabei entsteht jedoch kein Schaden.

Wir ein Datenblock wiederholt falsch eingelesen, so ist ein Datenaufzeichnungsfehler zu vermuten, der eventuell durch stellenweise unzureichende Qualität des Magnetbandes verursacht wurde. In einem solchen Fall wiederholen Sie die Aufzeichnung auf einem anderen Magnetbandabschnitt.

Hinweis:

Magnetbänder mit schadhaften Stellen, wie z. B. Knitter- oder Klebestellen, sind nicht zu verwenden!

Der Vergleich ist mit der Ausschrift OK abgeschlossen und der Recorder kann ausgeschaltet werden. Möchten Sie das gerettete Programm nicht überprüfen, wird nach dem Erscheinen der Abfrage VERIFY?(Y) die ENTER-Taste betätigt, wobei als Enderkennung OK ausgeschrieben wird. Nun hören Sie sich die Aufnahme am besten einmal über Lautsprecher an. Spulen Sie das Band zurück und spielen Sie es ab. Am Anfang werden Sie gegebenenfalls den von Ihnen gesprochenen Programmnamen hören. Danach hören Sie einen Pfeifton. Das ist der Vorton. Spätestens jetzt müssten Sie, wenn Sie das Programm wieder vom Recorder in den Computer laden wollen, die ENTER-Taste drücken. An den nachfolgenden schrillen Tönen erkennen Sie das eigentliche Programm.

Wir laden ein Programm

Löschen Sie nun den Arbeitsspeicher mit NEW und überzeugen Sie sich mit LIST, dass unser Testprogramm nicht mehr im Computer gespeichert ist. Spulen Sie die Kassette an den Programmanfang zurück und geben Sie in den Computer ein: (ENTER-Taste noch nicht drücken !)

CLOAD "TEST"

Schalten Sie nun den Recorder zur Wiedergabe ein. Sobald die Ansage beendet ist oder der pfeifende Vorton beginnt, drücken Sie auf die ENTER-Taste. Nun wird das Programm von der Kassette in den Computer geladen. Angezeigt werden wieder die Blocknummern. Bei fehlerhaftem Lesen eines Blockes erscheint ein " ? " hinter der Blocknummer. Dann ist ein erneutes Einlesen des zuvor fehlerhaften Blockes erforderlich. Nach Beendigung des Ladevorganges meldet sich der Computer mit:

FILE FOUND

Nun ist das Programm wieder vollständig im Arbeitsspeicher des Computers vorhanden. Überzeugen Sie sich!

Sollte sich beim Einlesen ein Fehler eingeschlichen haben, so wiederholen Sie bitte den Ladevorgang. Beachten Sie auch, dass bei einigen Recordertypen ein einwandfreies Abspeichern und Laden von der richtigen Einstellung des Aufnahme- und Lautstärkereglers abhängig ist.

Haben wir an unserem KC 85-System ein D004 und/oder ein USB-Modul M052 angeschlossen, können wir die Programme auch auf Diskette oder USB-Stick speichern. Die Benutzung der Anweisungen CLOAD und CSAVE unterscheidet sich kaum von der Kassetten-Variante. Das manuelle Umspulen des Magnetbandes entfällt und nach dem CSAVE erscheint auch nicht die Frage VERIFY?

Zur Ergänzung der Dateiarbeit mit Diskette und USB-Stick gibt es seit CAOS 4.7 noch die BASIC-Anweisung FILES um den Verzeichnisinhalt des aktuell eingestellten Speichergerätes anzuzeigen sowie die Anweisung CHDIR zum Wechsel des Verzeichnisses auf dem Speichergerät.

Merke:

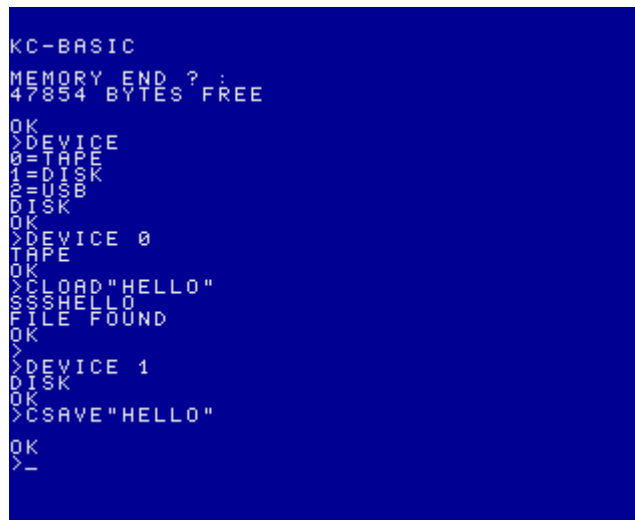
- **Anweisung: CSAVE**
  Format: CSAVE "NAME"
  Bemerkung: CSAVE "NAME" speichert das im Arbeitsspeicher befindliche Programm unter dem angegebenen Namen auf dem eingestellten Speichergerät. Es sind maximal die letzten 8 Zeichen des Namens signifikant.

  Beispiel: CSAVE "TEST"

- **Anweisung: CLOAD**
  Format: CLOAD "NAME"
  Bemerkung: Die Anweisung lädt das angegebene Programm vom eingestellten Speichergerät und legt es hinter dem schon im Speicher befindlichen Programm ab.

  Beispiel: CLOAD "TEST"

- **Anweisung: DEVICE**
  Format: DEVICE [Nummer]
  Bemerkung: Wird keine Nummer angegeben, dann dient diese Anweisung zur Anzeige der möglichen und des aktuell eingestellten Devices. Mit Angabe einer gültigen Nummer, dann erfolgt der Wechsel zu diesem Device.
  Beispiel: Es soll ein BASIC-Programm von Kassette auf Diskette kopiert werden. Als Erstes lässt man sich mit DEVICE die zur Verfügung stehenden Speichergeräte anzeigen. Dann schaltet man mit DEVICE 0 auf TAPE um und liest das Programm von Kassette ein. Nun kann man mit DEVICE 1 auf DISK wechseln und das Programm abspeichern.



Hinweis:

In CAOS 4.5 gab es zwischenzeitlich eine ähnliche Anweisung DRIVE mit dem Format DRIVE d [,u] um Laufwerk / USER-Bereich der Diskette wechseln zu können. Diese Anweisung wurde mit CAOS 4.7 durch die Anweisungen DEVICE und CHDIR ersetzt.

- **Anweisung: FILES**
  Format: FILES ["Maske"]
  Bemerkung: Diese Anweisung zeigt das Verzeichnis des aktuell eingestellten Laufwerks an. Durch Angabe einer geeigneten Maske kann die Auswahl der gelisteten Dateien eingeschränkt werden. Die Anweisung FILES ist bei DEVICE=TAPE nicht sinnvoll.
  Beispiel: FILES "*TTT"

Hinweis:

Die FILES-Anweisung ist bei Kassette im BASIC-Programm wenig sinnvoll, da hierzu die gesamte Kassette abgespielt werden muss. Bei DISK oder USB kann man sich mit dem Kommando FILES die vorhandenen Dateien anzeigen lassen. Die Auflistung vorhandener BASIC-Datenfelder *.TTT sollte also bei Kassette übersprungen werden. Innerhalb des BASIC-Programms sollten deshalb Vorkehrungen getroffen werden, damit das Kommando bei TAPE nicht ausgeführt wird. Dazu ist als in BASIC die DEVICE-Nummer zu ermitteln. Die DEVICE-Nummer ist in den Bits 2-4 von (IX+8) gespeichert. (IX+8) entspricht der Speicheradresse 01F8H, dezimal 504. Dabei muss man noch die gewünschten 3 Bit ausfiltern und das Ergebnis durch 4 teilen.

Im Diektmode kann man das entsprechend ausprobieren:

```
PRINT (PEEK(504) AND 28)/4
```

Diese Befehlszeile zeigt die DEVICE-Nummer an, also 0 für TAPE usw. Um jetzt innerhalb eines BASIC-Programms die Verzeichnisanzeige bei TAPE zu unterbinden, könnte folgende Programmzeile benutzt werden:

```
IF (PEEK(504)AND28)/4 > 0 THEN FILES "*TTT"
```

- **Anweisung: CHDIR**
  Format: CHDIR "Verzeichnis"
  Bemerkung: Diese Anweisung wechselt in das angegebene Verzeichnis je nach eingestelltem Device. Bei DEVICE=DISK kann das ein Disketten- oder Festplattenlaufwerk mit USER-Bereich (0 bis 31) sein, bei DEVICE=USB ein Unterverzeichnis. Bei Device=TAPE schaltet die Anweisung CHDIR die Schaltspannung für den Motor am Kassettenrecorder wechselweise ein bzw. aus, um an eine gewünschte Stelle des Bandes spulen zu können.

Beispiele:		
	CHDIR "A0"	wechselt zu Laufwerk A0: falls DISK
	CHDIR "/"	wechselt zum Wurzelverzeichnis bei USB

Einbinden von Maschinencode-Programmen

Mit der Anweisung BLOAD haben wir darüber hinaus die Möglichkeit, Maschinenprogramme (z. B. Zeichenbildtabellen) vom BASIC-Interpreter aus zu laden. Die Anweisung ist ähnlich der Systemanweisung LOAD zu handhaben (vergleiche dazu Seite 53).

- **Anweisung: BLOAD**
  Format: BLOAD "NAME"
  Bemerkung: Die Anweisung ruft das Betriebssystemprogramm LOAD zum Einlesen von Maschinenprogrammen während der Arbeit des BASIC-Interpreters auf. Bei DEVICE=TAPE ist (wie bei früheren BASIC-Versionen) der Dateiname nicht erforderlich, es wird die Datei eingelesen, welche vom Magnetband abgespielt wird. Für alle anderen Devices muss durch die Angabe des Dateinamens mitgeteilt werden, welche Datei einzulesen ist. Der Dateiname kann bis zu 8 Zeichen lang sein. Ohne Angabe wird als Dateityp .KCC verwendet, andere Dateitypen können jedoch mit angegeben werden.
  Beispiel: Spielprogramm BREAKOUT

Das Programm BREAKOUT besteht aus zwei Teilen. Dem eigentlichen BASIC-Programm und einem Maschinenprogramm, welches sich unmittelbar nach dem BASIC-Programm auf der Kassette befinden sollte. Beim Programmstart erscheint als Erstes die Frage:

HABEN SIE SCHON DAS MASCHINENPROGRAMM NACHGELADEN (J/N)?

Wenn Sie jetzt mit Nein antworten, erscheint ein Hinweis zum Starten des Bandes und das Maschinenprogramm wird von Kassette eingelesen.

Laden Sie das BASIC-Programm von einem anderen Speichergerät als TAPE, also z. B. von einem USB-Stick und beantworten die Frage nach dem Maschinenprogramm mit Nein, dann erhalten Sie diese Fehlermeldung:

```
?MO ERROR IN 31060
```

Schauen wir uns die Zeile 31060an:

```
31060 IFA$="N" THEN PRINTAT(20,10);"BAND STARTEN!" :
BLOAD : RETURN
```

Woher soll BASIC wissen, welche Datei es laden soll? In der Programmzeile 31060 steht nur ein BLOAD. Ergänzen Sie nach dem Befehl BLOAD den Dateinamen wie folgt:

```
31060 IFA$="N" THEN PRINTAT(20,10);"BAND STARTEN!" :
BLOAD"BREAK-ZG" : RETURN
```

Am besten, Sie speichern das geänderte BASIC-Programm gleich noch ab mit CSAVE"BREAKOUT". Nun können Sie das Programm mit RUN erneut starten und das Einlesen des Maschinenprogramms sollte funktionieren – vorausgesetzt, die Datei BREAK-ZG.KCC befindet sich auf dem Datenträger.

Merke:

- **Retten eines BASIC-Programms:**

  1. Programmanfang durch Zählerstand am Recorder merken oder durch akustisches Signal kennzeichnen
  2. Geben Sie ein:
     CSAVE "NAME"
  3. Schalten Sie den Recorder zur Aufnahme ein
  4. Drücken Sie die ENTER-Taste

  Bei DEVICE=DISK entfällt Schritt 1 und 3.

- Laden eines BASIC- bzw. eines Maschinenprogramms:

  1. Spulen Sie die Kassette vor den Anfang des zu ladenden Programms
  2. Geben Sie ein:
     CLOAD "NAME"
     bzw. für Maschinenprogramme
     BLOAD "NAME"
  3. Schalten Sie den Recorder zur Wiedergabe ein
  4. Drücken Sie vor oder spätestens während des pfeifenden Vortons die ENTER-Taste

  Bei DEVICE=DISK entfällt auch hier Schritt 1 und 3. Die Angabe des Dateinamens ist bei DEVICE=TAPE optional, bei allen anderen Speichergeräten erforderlich.

Übungen

Schreiben Sie noch ein oder zwei weitere Programme, die Sie retten und wieder laden. Wechseln Sie dabei auch die Speichergeräte, wenn ihnen weitere zur Verfügung stehen. Testen Sie mit ihrer so entstehenden Mini-Programmbibliothek alle im Kapitel behandelten Anweisungen.

4.1.10. Farbe (Anweisungen PAPER, INK, COLOR)

Mit dem KC 85 können Sie auf Ihrem Farbfernsehgerät 24 Farben darstellen. Das sind im einzelnen 8 Hintergrundfarben und 16 Vordergrundfarben. Unter den Vordergrundfarben verstehen wir die Farbe der Buchstaben bzw. bei der Grafik die Farbe der gezeichneten Linien und Punkte.

Jeder Vordergrund- und Hintergrundfarbe wird zur Programmierung ein Farbcode in Form einer Zahl zugeordnet. Diese ist abhängig vom eingestellten Grafikmodus. Die konkrete Zuordnung ist in den folgenden beiden Farbcode-Tabellen zu entnehmen.

Tabelle 48: Vorder- und Hintergrundfarben im LoRes-Modus

Farbe	Vordergrund		Hintergrund		Farbe
schwarz	0		0		schwarz
blau	1		1		blau
rot	2		2		rot
purpur	3		3		purpur
grün	4		4		grün
türkis	5		5		türkis
gelb	6		6		gelb
weiß	7		7		grau
schwarz	8				
violett	9				
orange	10				
purpurrot	11				
grünblau	12				
blaugrün	13				
gelbgrün	14				
weiß	15				

Tabelle 49: Die Codierung der 4 Farben im HRG-Modus

Farbe	Vorder- oder Hintergrund	
schwarz	0	
rot	1	
türkis	2	
weiß	3	

Im Lores-Modus können wir die Vordergrundfarbe auch blinkend darstellen, indem wir zum entsprechenden Vordergrundfarbcode die Zahl 16 addieren. Dadurch ergeben sich insgesamt 8 Hintergrundfarbcodes (0 bis 7) und 32 Vorder-

grundfarbcodes (0 bis 31). Dabei sind die Vordergrundfarben deutlich heller als die Hintergrundfarben.

Im HRG-Modus gibt es dagegen nur 4 Vordergrund- und 4 Hintergrundfarben.

Mit den drei Anweisungen PAPER (für die Hintergrundfarbe), INK (für die Vordergrundfarbe) und COLOR (für Vorder- und Hintergrundfarbe) sowie den Farbcodes lassen sich umfangreiche Farbmöglichkeiten des KC 85 unkompliziert nutzen. Die genaue Handhabung der eben genannten Befehle entnehmen Sie bitte dem folgenden Abschnitt.

Merke:

- **Anweisung: PAPER**
  Format: PAPER h
  Bemerkung: Einstellen der Hintergrundfarbe; h … Hintergrundfarbcode entsprechend Tabelle 48 bzw. 49
  Beispiel: PAPER 3

- **Anweisung: INK**
  Format: INK v
  Bemerkung: Einstellen der Vordergrundfarbe; v … Vordergrundfarbcode entsprechend Tabelle 48 bzw. 49
  Beispiel: INK 0

- **Anweisung: COLOR**
  Format: COLOR v, h
  Bemerkung: Einstellen der Vorder- und Hintergrundfarbe
  Beispiel: 250 VF=7: HF=2
  260 COLOR VR, HF

- **Farbanweisung innerhalb einer PRINT-Anweisung**

  Werden die Farbanweisung in eine PRINT-Anweisung eingefügt, so bezieht sich die Farbeinstellung nur auf die Ausgabe derselben PRINT-Anweisung. Danach ist der vorherige Farbzustand wieder hergestellt.

  Das Einbinden einer Farbanweisung in eine PRINT-Anweisung geschieht in folgender Form:

  ```
  PRINT Farbanweisung; Ausdruck
  ```

  Beispiel:

  ```
  20 INPUT A$
  30 PRINT COLOR 0,4;A$
  ```

Übung

Erstellen Sie ein Programm, welches alle möglichen Farbkombinationen von den Hintergrundfarben und den blinkenden Vordergrundfarben im Lores-Modus ausgibt.

4.1.11. Grafik (PSET, PRESET, CIRCLE, LINE, PTEST)

Punkte

Der KC 85 bietet Ihnen einen für einen Kleincomputer beispielhaften Grafik-Komfort. Wir sind in der Lage, das Fernsehbild in 320 Punkte (in der Waagerechten) mal 256 Punkten (in der Senkrechten) aufzulösen. Damit stehen uns insgesamt 81.920 Graphikpunkte zur Verfügung. Etwas anschaulicher bedeutet das z. B., dass wir jedes Buchstabefeld in 64 (8x8) Punkte auflösen können. Dadurch sind Sie in der Lage, mathematische Funktionen oder vom Computer überwachte Prozesse grafisch exakt in geschlossenen Kurvenzügen darzustellen. Des Weiteren können Sie Bilder zur Unterstützung selbsterstellter Programme, Diagramme, Muster und Figuren entwerfen.

Was zeichnen wir nun konkret?

Jeder einzelne Punkt ist ansprechbar durch seine Koordinaten. Die X-Koordinate gibt den Abstand vom linken Bildschirmrand (von links nach rechts von 0 bis 319) und die Y-Koordinate den Abstand vom unteren Bildschirmrand (von unten nach oben von 0 bis 255) des jeweiligen Punktes an. Vergleichen Sie hierzu bitte Bild 31 auf der nächsten Seite bzw. die Bilder 41 bis 43 im Anhang ab Seite 440.

Wollen wir nun einen Punkt mit der Koordinate X und Y in einer bestimmten Vordergrundfarbe zeichnen, so geben wir dies in der Form

```
PSET x-Koordinate, y-Koordinate, Farbcode
```

ein. Als Farbcode wird der im letzten Kapitel beschriebene Vordergrundfarbcode verwendet.

Zeichnen Sie einen weißen Punkt mit der den Koordinaten x=20 und y=15, indem Sie eingeben:

```
PSET 20, 15, 7
```

Zeichnen Sie nach Belieben noch weitere Punkte ein.

So, wie wir einen bestimmten Punkt setzen können, ist es auch möglich, diesen mit der Anweisung PRESET zu löschen. Geben Sie ein:

```
PRESET 20, 15
```

Und der von Ihnen vorhin gesetzte Punkt ist wieder gelöscht. Probieren Sie beide Anweisungen ruhig eine Weile aus!

Folgendes Programm erzeugt einen kleinen Sternenhimmel-Ausschnitt:

```
10 CLS
20 FOR I = 0 TO 30
30 PSET RND(1)*100+100, RND(1)*100+100, 7
40 NEXT
```

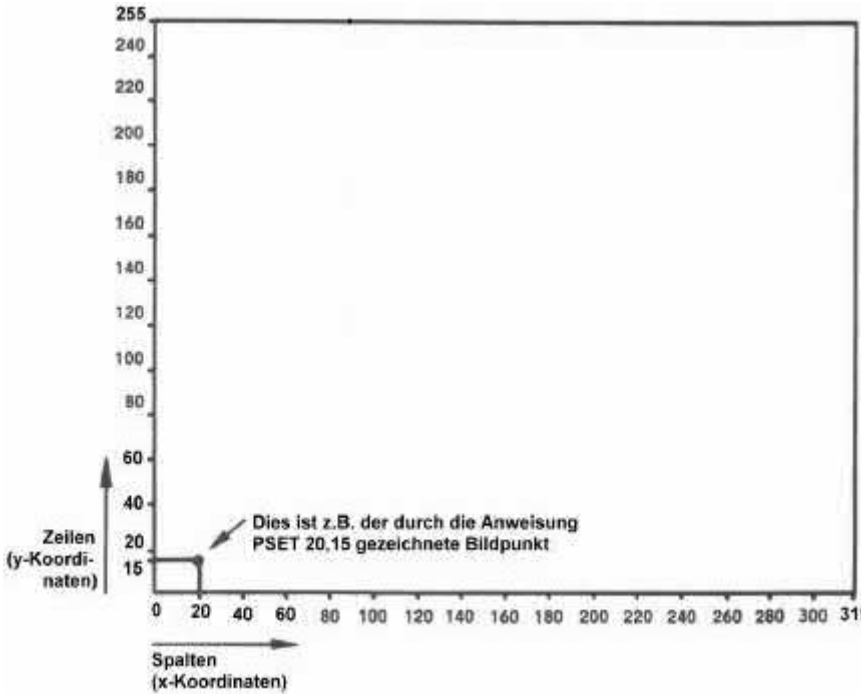
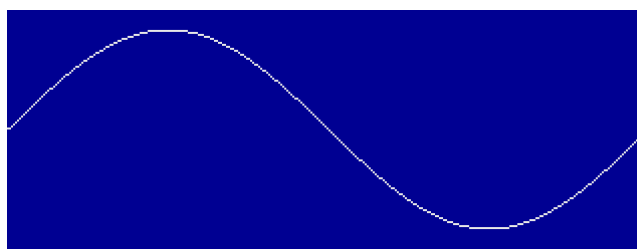


Bild 31: Vollgrafik: Das Anzeigebild besteht aus 320x256 Bildpunkten

Nun schauen wir uns noch ein kleines mathematisches Beispiel an. Folgendes Beispiel zeichnet die Sinusfunktion im Bereich von 0 bis 2PI.

```
10 PAPER 1: CLS
20 FOR X=0 TO 319
30 Y=128+50*SIN(X/159.5*PI)
40 PSET X, Y, 7
50 NEXT
```



Linien und Kreise

Wir können jedoch nicht nur einige Punkte setzten oder löschen. Der BASIC-Interpreter verfügt auch über Grafikanweisungen, die es ermöglichen, unkompliziert geometrische Grundfiguren zu zeichnen.

Die Anweisung CIRCLE in der Form

CIRCLE x-Koordinate, y-Koordinate, Radius, Farbcode

zeichnet auf dem Bildschirm einen Kreis mit dem angegebenen Radius, um den Mittelpunkt der angegebenen Koordinaten x und y in der durch den Farbcode festgelegten Vordergrundfarbe. Möchten wir z. B. einen weißen Kreis mit einem Radius von 50 Bildpunkten in die linke untere Ecke des Bildschirmes zeichnen, so geben wir ein:

```
CIRCLE 49, 49, 50, 7
```

Die Angabe der Mittelpunktkoordinaten orientiert sich ebenfalls an der im Bild 31 beschriebenen Grafikpunkte-Gliederung. Programmieren Sie nach Belieben noch einige weitere Kreise.

Mit der Anweisung LINE in der Form

LINE x_a , y_a , x_e , y_e , Farbcode

können wir eine Linie auf dem Bildschirm zeichnen. Dabei bezeichnen die Koordinaten x_a und y_a den Anfangs- und die Koordinaten x_e und y_e den Endpunkt der Linie. Der letzte Parameter legt wie bei der Anweisung CIRCLE die gewünschte Vordergrundfarbe fest. Die Anweisung

```
LINE 0, 0, 319, 255, 7
```

zeichnet z. B. eine Bildschirmdiagonale, beginnend in der linken unteren Ecke. Mithilfe des folgenden kleinen Programms können Sie die Anweisung LINE unkompliziert probieren:

```
10 INPUT "XA, YA, XE, YE, F"; XA, YA, XE, YE, F
20 LINE XA, YA, XE, YE, F
30 GOTO 10
```

Als letztes lernen wir in diesem Kapitel die Grafik-Funktion PTEST kennen. Mithilfe der Funktion kann man an einem beliebigen Bildpunkt testen, ob dieser die Vorder- oder Hintergrundfarbe trägt. Diese Funktion ist ebenfalls nur in Verbindung mit einer Anweisung sinnvoll zu gebrauchen.

Als Argument wird die x-Koordinate des zu testenden Bildpunktes in Klammern hinter die Funktion geschrieben. Als aktuelle y-Koordinate setzt der Computer die zuletzt in einer der vier Grafik-Anweisungen erwähnte y-Koordinate ein.

Seit CAOS 4.7 kann man alternativ auch die x- und die y-Koordinate durch Komma getrennt in den Klammern angeben und ist kann damit unabhängig die Farbe jedes beliebigen Bildpunktes testen.

Probieren Sie folgendes Beispiel:

```
10 CLS: PSET 50, 100, 7
20 PRINT PTEST (50)
```

Nach Ausführung des Programms werden Sie auf dem Bildschirm, neben dem in Zeile 10 festgelegten Punkt, als Ergebnis der Anweisung in Zeile 20, die Zahl 1 ablesen können.

In unserem Programm wurde der Punkt (50, 100) getestet. Die y-Koordinate 100 ergibt sich, da sie die letzte erwähnte y-Koordinate einer Grafik-Anweisung (in Zeile 10) ist. Ist die Vordergrundfarbe gesetzt, so liefert die Funktion, wie wir am Beispiel sehen können, den Wert 1. Ist jedoch die Hintergrundfarbe gesetzt, so liefert die Funktion den Funktionswert 0.

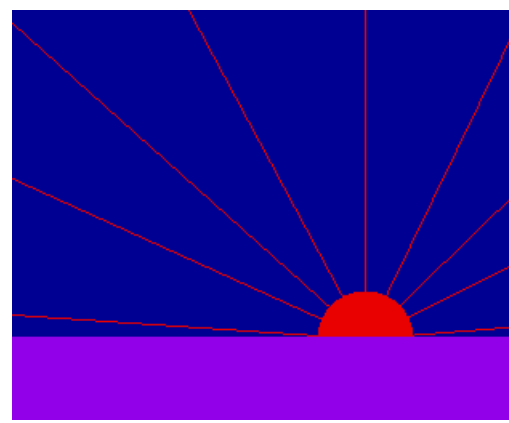
Überzeugen Sie sich an folgendem Beispiel:

```
PRINT PTEST(51)
```

Soeben haben wir den Bildpunkt (51, 100) rechts neben dem gesetzten Punkt getestet. Da dieser in der Hintergrundfarbe erscheint, lautet das Ergebnis der letzten Anweisung 0.

Zeichnen Sie sich nun mit folgendem Programm Ihre eigene aufgehende Computer-Sonne:

```
 10 COLOR7,1:CLS
 20 X=220:Y=50
 30 CIRCLE X,Y,30,2
 40 FOR B=21TO79: PRESET 0,B: A=220
 50 IF PTEST(A)=0 THEN PSET A,B,2: A=A-1:GOTO 50
 60 A=221
 70 IF PTEST(A)=0 THEN PSET A,B,2: A=A+1:GOTO 70
 80 NEXT
 90 LINE X,Y,0,150,2
100 LINE X,Y,0,65
110 LINE X,Y,0,247
120 LINE X,Y,110,255
130 LINE X,Y,220,255
140 LINE X,Y,319,255
150 LINE X,Y,319,146
160 LINE X,Y,319,100
170 LINE X,Y,319,58
180 FOR Y=0TO51
190 LINE 0,Y,319,Y,9
200 NEXT
210 GOTO 210
```



Lassen Sie das Programm mit der Anweisung RUN abarbeiten.

Möchten Sie Ihren Sonnenauf- oder Untergang beenden, so drücken Sie die BRK-Taste ! Das Programm wird unterbrochen und der Computer meldet sich mit dem Cursor eingabebereit. Löschen Sie das Programm bitte noch nicht, da wir es im nächsten Abschnitt noch einmal verwenden werden! Selbstverständlich können Sie es auch auf Magnetband speichern.

Sollten nach der Ausführung der Anweisung CLS noch Reste der zuvor gezeichneten Grafik auf dem oberen und Bildschirmrand verbleiben, so stören Sie sich bitte nicht daran. Im nächsten Kapitel erfahren Sie, wie diese beseitigt werden.

Grafik und Farbe

Lassen Sie das "Sonnenbild" nach Abbruch unseres letzten Programms stehen und geben Sie ein:

```
PSET 50, 25, 6
```

Sicher hatten Sie angenommen, dass sich nun der in unserem "blauen Meer" befindliche Bildpunkt (50, 25) gelb färbt. Stattdessen wurde jedoch, wie Sie sehen können, ein Feld von 8 waagerecht nebeneinander liegenden Punkten gelb gefärbt. Sollten Sie das Programm auf einem KC 85/3 abgearbeitet haben, sind es sogar 8x4 Bildpunkte, die sich gelb gefärbt haben.

Diese Erscheinung ist schnell geklärt. Beim KC 85/3 ist für die Punkte eines Feldes von waagerecht 8 mal senkrecht 4 Punkten im Computer jeweils nur eine Farbcodierung, welche die Vordergrund- und die Hintergrundfarbe festlegt, gespeichert. Damit erscheinen innerhalb dieses Feldes alle Punkte, die eine Vordergrundfarbe tragen in der gleichen Vordergrundfarbe und alle Hintergrundfarbbildpunkte in der gleichen Hintergrundfarbe. Der KC 85/4 hat einen anderen Bildschirmaufbau, hier gilt im LoRes-Modus die gleiche Farbe für ein Feld mit 8 waagerechten Punkten. Siehe auch Bilder 41 bis 43 im Anhang ab Seite 440.

In unserem Beispiel tragen alle Punkte des 8x1-Feldes (beim KC 85/4), in dem der Punkt (50, 25) liegt, die Vordergrundfarbe Violett (Farbcode 9; siehe letzte Programmzeile 190). Da der veränderte Vordergrundfarbcode 6 nicht nur für den Punkt (50, 25) maßgebend ist, nehmen alle 32 Bildpunkte, die ihren Farbwert aus der gleichen Speicherstelle wie der Punkt (50, 25) beziehen, die neu programmierte Vordergrundfarbe gelb an.

Folgende Eingabe auf unser "Sonnenbild" verdeutlicht das eben Gesagte noch einmal am Beispiel:

```
LINE 120, 0, 319, 120, 0
```

Merke:

- Der Bildschirm gliedert sich von links nach rechts in die Spalten 0 bis 319 (x-Koordinate) und von unten nach oben in die Zeilen 0 bis 255 (y-Koordinate). Der Schnittpunkt jeder Zeile und Spalte ist ein Punkt, den wir grafisch darstellen können.

Anweisung: **PSET**

Format: PSET x, y [,f]

Bemerkung: Die Anweisung setzt einen Punkt auf den Bildschirm durch Angabe der Parameter

x = Horizontalkoordinate 0 ≤ x ≤ 319
y = Vertikalkoordinate 0 ≤ y ≤ 255 und
f = Bildpunktfarbe 0 ≤ f ≤ 31 im LoRes-Modus
0 ≤ f ≤ 3 im HRG-Modus

Wird der Parameter f weggelassen, so erscheint der Punkt in der zuletzt gewählten Grafikfarbe.

Beispiel:

```
10 F=7: CLS
20 FOR P=0 TO 2 * PI STEP 0.01
30 X=159 + 100 * SIN(P * 3)
40 Y=127 + 100 * SIN(P * 4)
50 PSET X, Y, F
60 NEXT
```

- **Anweisung: PRESET**

Format: PRESET x, y [,f]

Bemerkung: Die Anweisung löscht einen Punkt auf den Bildschirm durch Angabe der Parameter

x = Horizontalkoordinate 0 ≤ X ≤ 319
y = Vertikalkoordinate 0 ≤ Y ≤ 255 und
f = Bildpunktfarbe (Die Angabe der Bildpunktfarbe hat keine Wirkung, erzeugt aber auch keine Fehlermeldung).

Die Anweisung PRESET ist nur im LoRes-Modus sinnvoll. Im HRG-Modus hat PRESET keine Wirkung, hier sind alle 4 Farben gleichberechtigt und keine der Farben als Hintergrund oder Vordergrund zugeordnet. Ein Bildpunkt kann im HRG-Modus mit der Anweisung PSET in jeder beliebigen Farbe gesetzt werden.

- **Anweisung: LINE**

Format: LINE x_a , y_a , x_e , y_e , [, f]

Bemerkung: Die Anweisung zeichnet eine Linie auf dem Bildschirm, die im Punkt mit dem Koordinaten x_a und y_a beginnt und im Punkt mit den Koordinaten x_e und y_e endet.
Die Farbe der Geraden wird durch den Vordergrundfarbcode f (0 ≤ X ≤ 31) festgelegt. Wird der Parameter f weggelassen, so erscheint die Gerade in der zuletzt gewählten Grafikfarbe.

Beispiel: LINE 10,100,300,100,26

- **Anweisung: CIRCLE**

  Format: CIRCLE x_m , y_m , r [, f]

  Bemerkung: Die Anweisung zeichnet einen Kreis mit den Mittelpunktkoordinaten x_m , y_m und dem Radius r auf dem Bildschirm.
  Die Farbe der Kreislinie wird durch den Vordergrundfarbcode f (0 ≤ X ≤ 31) festgelegt. Wird der Parameter f weggelassen, so erscheint die Kreislinie in der zuletzt gewählten Grafikfarbe.

  Beispiel:
  ```
  10 CLS
  20 FOR R=1 TO 100
  30 CIRCLE 159, 127, R, 16 * RND(1)
  40 NEXT
  ```

Funktion: PTEST

Die Funktion PTEST gibt es ab CAOS 4.7 in zwei Varianten. Format 1 gilt für alle CAOS-Versionen, Format 2 ist ab CAOS 4.7 vorhanden und unterstützt auch den HRG-Modus.

Format 1: PTEST (x)

Bemerkung: Die Funktion testet, ob in einem Bildpunkt die Vordergrundfarbe gesetzt ist. Die x-Koordinate des zu testenden Punktes wird als Parameter in Klammern hinter die Funktion geschrieben. Als y-Koordinate benutzt der Computer die zuletzt genannte Y-Koordinate einer Grafikanweisung. Ist die Vordergrundfarbe gesetzt, liefert die Funktion als Ergebnis den Wert 1; andernfalls den Wert 0. Das funktioniert nur im LoRes-Modus.

Format 2: PTEST (x ,y)

Bemerkung: Die Funktion ermittelt, in welcher Farbe ein Bildpunkt gesetzt ist. Die Parameter
x = Horizontalkoordinate 0 ≤ X ≤ 319 und
y = Vertikalkoordinate 0 ≤ Y ≤ 255
werden in Klammern hinter die Funktion geschrieben. Zu unterscheiden ist nun zwischen LoRes- und HRG-Modus.

Lores:
Ist der Bildpunkt nicht gesetzt (Hintergrundfarbe), liefert die Funktion den Wert 0. Ist der Bildpunkt in Vordergrundfarbe gesetzt, liefert die Funktion als Ergebnis den Wert der Farbe, also 1 für blau, 2 für rot usw. Bei Vordergrundfarbe schwarz liefert die Funktion immer 8, auch wenn als Vordergrundfarbe 0 eingestellt ist. In früheren CAOS-Versionen wurde bei gesetztem Bildpunkt immer der Wert 1 zurückgegeben unabhängig von dessen Farbe.

Hires:
Die Funktion liefert die Funktion als Ergebnis den Wert der Farbe, also 0 für schwarz, 1 für rot, 2 für türkis und 3 für weiß. Eine Unterscheidung zwischen Vordergrund- und Hintergrundfarbe gibt es nicht.

Das folgende Beispielprogramm kann im HRG- und LoRes-Mode abgearbeitet werden. Dabei ist auf die Anzeige zu achten und der ermittelte Wert für die beiden Bildpunkt zu erklären.

Beispiel:

```
10 PAPER 1:CLS
20 FOR R=1 TO 100
30 CIRCLE 159, 127, R, 16 * RND(1)
40 NEXT
50 PRESET 0, 56
55 A=PTEST(70)
60 PRINT"Punkt1=";A,"Punkt2=";PTEST(130,130)
```

Übungen

1. Erstellen Sie ein Programm, das die Sinusfunktion von 0 bis 2PI und das dazugehörige Achsenkreuz des Koordinatensystems zeichnet! (Lösung Kapitel 4.1.24)
2. Erklären Sie das Auftreten farbiger Rechtecke bei der Abarbeitung des Beispielprogramms zur Anweisung CIRCLE!

4.1.12. Bildgestaltung (WINDOW, LOCATE, WIDTH)

Fenster

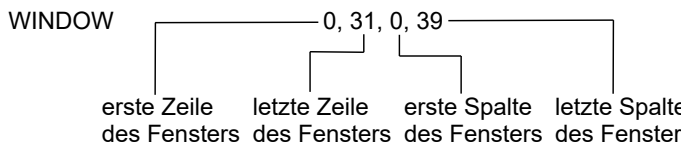
Vielleicht haben Sie es schon einmal ausgezählt; unser Bildschirm gliedert sich in 30 Schriftzeilen mit je 40 Zeichen. Dies ist das "Standard-Format", welches stets nach dem Einschalten des KC 85/3 eingestellt ist.

Darüber hinaus können wir aber noch die verschiedensten Bildausschnitte, wir sagen in Zukunft Fenster, einstellen. Der Wirkungsbereich der Steuerzeichen beschränkt sich auf das jeweils eingestellte Fenster. Nur hier kann geschrieben werden. Der restliche Bildschirm außerhalb des Fensters bleibt bei der Computerarbeit mit zwei Ausnahmen unbeeinflusst.

Die eine Ausnahme ist die PRINT-Anweisung AT. Diese Funktion werden wir im nächsten Abschnitt kennenlernen.

Die andere Ausnahme sind die Grafikanweisungen, welche wir bereits kennengelernt haben.

Als grösstmögliches Fenster lässt sich der Bildschirm auf 32 Zeilen und 40 Spalten einstellen. Dies geschieht in folgender Form:



Beim KC 85/4 sind es nach dem Einschalten bereits 32 Zeilen. Die Nummerierung der einzelnen Zeichenzeilen und Zeichenspalten ersehen Sie aus Bild 32.

Nun sind Ihnen bestimmt auch die nicht gelöschten Bildränder unseres "Sonnenprogramms" aus dem letzten Kapitel erklärlich. Der Effekt tritt jedoch nur beim KC 85/3 auf. Dort ist beim Einschalten das Standardfenster (WINDOW 1, 30, 0 ,39) eingestellt. Die Ausführung der Grafikanweisungen ist unabhängig vom eingestellten Fenster. Dadurch wurde auch auf dem Bildschirm außerhalb des Fensters gezeichnet.

Da die Steuerzeichen jedoch nur innerhalb des Fensters wirken, wurde die Grafik außerhalb des Fensters nicht mit gelöscht. Durch die Ausführung der WINDOW-Anweisung ohne Parameter wird das Standard-Fenster mit 30 Zeilen und 40 Spalten (auch beim KC 85/4) eingestellt. Veranschaulichen Sie sich die Funktionsweise der Anweisung am Beispiel des folgenden Programms!

```
10 WINDOW 0,31,0,39: COLOR 0,6: CLS
20 PRINT "GROESSTMOEGLICHSTES FENSTER"
30 WINDOW 15,25,2,38: COLOR 7,2: CLS
40 PRINT "LISTING IM KLEINEN FENSTER"
50 LIST
```

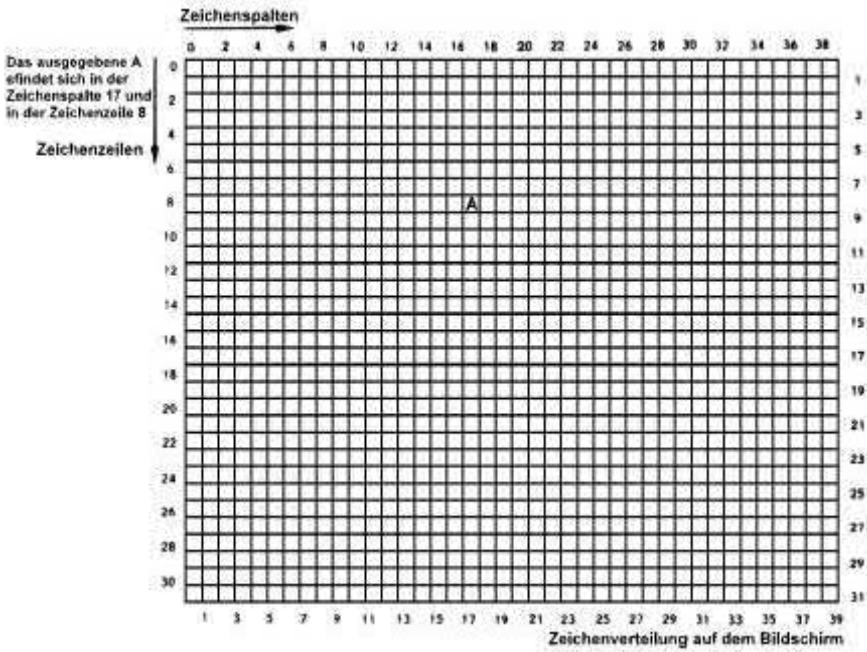


Bild 32: Einteilung des Bildschirms in Zeichenzeilen und Zeichenspalten

Platzieren und Formatieren

Unabhängig vom eingestellten Fenster können wir mit der PRINT-Funktion AT unsere Ausgaben an jeder beliebigen Stelle des Bildschirms platzieren. Dabei ist folgende Form zu beachten:

PRINT AT(Zeile, Spalte);Ausdruck

Möchten wir also z. B. in der Mitte der vorletzten Zeile das Wort ENDE schreiben, so geben wir ein:

PRINT AT(30, 20); "ENDE"

In folgender Weise kann man die PRINT AT-Anweisung eine Farbanweisung einfügen:

PRINT AT(Zeile, Spalte); Farbanweisung;Ausdruck

Verändern wir unser Beispiel und lassen das Wort "ENDE" blau auf gelb blinken:

PRINT AT(30, 20);COLOR 17,6; "ENDE"

Es ist jedoch stets darauf zu achten, dass die eingefügte COLOR-Anweisung beide Parameter enthält. In gleicher Weise können auch an Stelle der COLOR-Anweisung die Farbanweisung INK und PAPER eingefügt werden.

Um unsere Ergebnisse in leicht überschaubarer, tabellierter Form darstellen zu können, stehen uns darüber hinaus die Formatierungsfunktionen TAB und SPC zur Verfügung. Mithilfe dieser Funktionen können wir eine Ausgabe exakt in einer Spalte platzieren. So druckt z. B. die Anweisung

```
PRINT TAB(1); "C"
```

ein C ins 2. Buchstabenfeld vom linken Fensterrand aus. Das Argument von TAB gibt also den Abstand der nach dem Semikolon folgenden Ausgabe zum linken Bildschirmrand in Zeichenfeldern an. Das Argument der Formatierungsfunktion SPC gibt dagegen die Anzahl der zu erzeugenden Leerzeichen zwischen der Ausgabe und dem linken Fensterrand bzw. der letzten bereits in der Zeile erfolgten Ausgabe an. Verdeutlichen Sie sich dies an folgendem Beispiel:

```
10 PRINT TAB(4); "X";TAB(7);"Y"
20 PRINT SPC(4); "X";SPC(7);"Y"
```

Die Angaben der TAB- und der SPC-Funktionen beziehen sich auf das jeweils eingestellte Fenster. Alle drei Formatierungsfunktionen können nur in Verbindung mit einer PRINT- Anweisung wirksam werden!

Mithilfe der Anweisung LOCATE können wir den Cursor an einer beliebigen Stelle innerhalb des Fensters platzieren. Dies erweist sich als sehr vorteilhaft, wenn z. B. auch die Eingabe in eine übersichtliche Bildgestaltung mit einbezogen werden sollen:

```
10 LOCATE 20,30
20 INPUT "A=";A
 ⋮
```

Der erste Parameter hinter LOCATE gibt die Zeile und der zweite die Spalte an, in welcher der Cursor zu platzieren ist.

Um die Zeile zu ermitteln, in welcher sich der Cursor befindet, verfügt der Interpreter über die Funktion CSRLIN(n).

Ist das Argument n gleich null, so erhalten wir die Zeilennummer der Zeile, in der sich der Cursor befindet, bezüglich dem maximalen Bildschirmfenster (32 Zeilen). Im Falle von n > 0, bezieht sich die Zeilenangabe auf das aktuell eingestellte Fenster.

Testen Sie folgendes Beispiel:

```
10 WINDOW10,30,0,39:PAPER3:CLS
20 PRINT CSRLIN(0); CSRLIN(44)
```

Normalerweise beträgt die Länge einer Ausgabezeile auf dem Bildschirm 40 Zeichen. Mit der Anweisung WIDTH können wir diese jedoch verändern. Schauen wir uns die Wirkung der Anweisung an folgendem Beispiel an. Geben Sie ein:

```
10 PRINT "ABCDEFGHIJKLMNOPQRSTUVWXYZ"
```

Lassen sie das Programm abarbeiten. Geben Sie weiter ein:

```
WIDTH 16
```

Beim erneuten Abarbeiten des Programms erkennen Sie nun deutlich, dass die Länge einer Ausgabezeile auf 16 Zeichen begrenzt wurde. Gleiches gilt auch für die Ausgabe auf Drucker.

Merke:

- **Anweisung: WINDOW**
  Format: WINDOW z_a , z_e , s_a , s_e
  Bemerkung: Die Anweisung ermöglicht es, bestimmte Bildausschnitte, sogenannte Fenster, auf dem Bildschirm festzulegen. Dabei besitzen die der Anweisung folgende Parameter folgende Bedeutung:
  z_a - erste Zeile des Fensters
  z_e - letzte Zeile des Fensters
  s_a - erste Spalte des Fensters
  s_e - letzte Spalte des Fensters
  Entsprechend der Zeichenzeilen- und -spalteneinteilung (siehe Bild 32) gilt $0 \le z_a \le z_e \le 31$ und $0 \le s_a \le s_e \le 39$. Bei Ausführung der Anweisung ohne Parameterangabe wird das Standardfenster (WINDOW1,30,0,39) eingestellt.

  Beispiel:
  ```
  10 ZA=2 : ZE=30
  20 SA=1 : SE=38
  30 WINDOW ZA, ZE, SA, SE
  40 CLS
  ```

- **Funktion: AT**
  Format: PRINT AT(z, s); [COLOR-Anweisung;] PRINT-Liste
  Bemerkung: Die Ausgabefunktion AT platziert die nächste Ausgabe beginnend auf dem Zeichenfeld z und in der Spalte s. Anstelle der COLOR-Anweisung können in dem PRINT-Ausdruck in gleicher Weise eine INK- oder PAPER-Anweisung eingefügt werden.

  Beispiel:
  ```
  PRINT AT(158,16); COLOR23,2; "ACHTUNG!"
  ```

- **Funktion:** **TAB**
  und
  SPC

  Format: PRINT TAB(i); Ausdruck
  und
  PRINT SPC(i); Ausdruck

  Bemerkung: Die Ausgabefunktionen TAB und SPC platzieren eine Ausgabe in einer bestimmten Zeichenspalte. Das Argument i von TAB gibt den Abstand der Ausgabe zum linken Fensterrand in Zeichenfeldern an. Das Argument i von SPC gibt die Anzahl der zu erzeugenden Leerzeichen zwischen der Ausgabe und dem linken Fensterrand bzw. der letzten bereits in der Zeile erfolgten Ausgabe an. i ist in beiden Fällen eine ganze Zahl zwischen 0 und 255. Die Funktionen sind nur in Verbindung mit der Anweisung PRINT wirksam.

  Beispiel:
  ```
  5 PRINT "X"; TAB(13); "SINX"; TAB(26); "COSX"
  10 FOR I=0 TO 7 STEP PI/10
  20 PRINT I; TAB(13); SIN(I); TAB(26); COS(I)
  30 NEXT
  ```

- **Anweisung: LOCATE**

  Format: LOCATE z, s

  Bemerkung: LOCATE platziert den Cursor auf die angegebene Spalte der Zeile innerhalb des festgelegten Fensters. Dabei gilt für den die Zeile festlegenden Parameter z: $0 \leq z \leq z_e - z_a$ mit z_a = erste Zeile des Fensters und z_e = letzte Zeile des Fensters.
  Ferner gilt für den Parameter s: $0 \leq s \leq s_e - s_a$ mit s_a = erste Spalte des Fensters und s_e = letzte Spalte des Fensters.

  Beispiel:
  ```
  10 CZ=3: CS=5
  20 LOCATE CZ, CS
  ⋮
  ```

- **Funktion: CSRLIN**
  Format: CSRLIN (n)
  Bemerkung: CSRLIN (n) liefert als Funktionswert die Nummer der Zeile, in welcher sich der Cursor befindet.
  n = 0 - Zeilennummer bezüglich des maximalen Fensters
  n > 0 - Zeilennummer bezüglich des aktuellen Fensters
  n < 0 - nicht definiert

  Beispiel:
```
10 WINDOW 0,31,0, 39 : CLS
15 PRINTCHR$(17)
20 RANDOMIZE
30 FOR I=0 TO 100
40 INK(16*RND(1))
50 PRINTAT(32*RND(1),38*RND(1)); CSRLIN(0)
60 NEXT
70 INK7: PRINTCHR$(18)
```

- **Anweisung: WIDTH**
  Format: WIDTH Zeichenanzahl
  Bemerkung: WIDTH legt die Länge einer Ausgabezeile für Drucker oder Bildschirm fest. Im Standardfall beträgt WIDTH 0 d. h. Die Ausgabezeile ist nicht begrenzt.

  Beispiel: WIDTH 15

Übungen

1. Erläutern Sie den Unterschied der Funktionen TAB und SPC!
2. Welche Werte können die Parameter der Anweisung LOCATE annehmen?

4.1.13. Zeichenvorrat (Funktionen ASC, CHR$)

Die Funktionen ASC und CHR$

Alle Zeichen, wie Buchstaben, Zahlen, Satzzeichen oder Steuerzeichen (z. B. Cursor links), die wir über die Tastatur eingeben können, sind im Computer unter einem bestimmten Code abgelegt. Die Codes sind Zahlen zwischen 0 und 255. Wollen Sie nun wissen, mit welchen Code der Buchstabe Z gespeichert wird, so bedienen Sie sich der Funktion ASC:

```
PRINT ASC("Z")
```

liefert die Zahl 90. Diese Zahl ist der Code des Zeichens Z. umgekehrt können Sie auch mithilfe der Funktion CHR$ zu einem Code das dazu gehörende Zeichen ermitteln. Geben Sie ein:

```
PRINT CHR$(90)
```

Da CHR$ sozusagen die Umkehrfunktion zu ASC ist, erhalten wir nun das Zeichen Z.

Beachten Sie bitte, dass die Argumente der Funktion (bei ASC ein String und bei CHR$ ein Zahlen-Code) in Klammern zu setzen sind.

Ab Seite 203 finden Sie eine Auflistung des gesamten Zeichenvorrates mit den dazugehörenden Codes. Mithilfe dieser Funktion können wir uns ohne viel Aufwand den gesamten Zeichenvorrat unseres Computers ausgeben lassen:

```
10 FOR I=32 TO 255
20 PRINT CHR$(I);
30 NEXT
```

Im Kapitel 4.1.22 werden wir erfahren, wie wir diesen Zeichenvorrat noch ergänzen und verändern können.

Unter den ersten 32 Codes (0-31) finden wir die sogenannten nichtdruckenden Steuerzeichen. Durch diese sind wir nun in der Lage, Steuerfunktionen auszuführen, zu denen man von BASIC aus keinen direkten Zugriff hat.

So können wir zum Beispiel mit:

```
PRINT CHR$(18)
```

den Scrolling-Modus einstellen, der eine Rollen des Bildes bei Überlauf organisiert bzw. mit

```
PRINT CHR$(17)
```

den Page-Modus, der ein Überschreiben des Bildes bewirkt.

Merke:

- **Funktion:** **ASC**
  Format: ASC(String)
  Bemerkung: Die Funktion ASC ermittelt den ASCII-Code des ersten Zeichens des Strings.

  Beispiel: PRINT ASC("ZIGARRE")
  90

- **Funktion:** **CHR$**
  Format: CHR$(Zahl)
  Bemerkung: Die Funktion CHR$ weist der Zahl (ASCII-Code) das entsprechende Zeichen zu.

  Beispiel: PRINT CHR$(77)
  M

Übung

1. Erklären Sie die Wirkung der Anweisung PRINT CHR$(16) !

4.1.14. String-Operationen, String-Funktionen

Zur Wiederholung

Wir wissen, dass unser Computer Zeichenketten, sogenannte Strings, verarbeitet. Diese Strings können wir auch unter Stringvariablen, die am Ende durch ein $ zu kennzeichnen sind, ablegen. Es gibt mehrere Möglichkeiten, Strings in einem Programm festzulegen. Hier einige Beispiele:

```
LET A$ = "BAUM"
INPUT B3$
R$ = INKEY$
```

Die Ausgabe erfolgt mit Hilfe der Anweisung PRINT, z. B. wie folgt:

```
PRINT "ANTWORT:"; G$
```

String-Operationen

Wir können auch mit Strings in gewissem Umfang operieren. So ist es zum Beispiel möglich, Strings durch das Operationszeichen "+" miteinander zu verknüpfen. Geben Sie ein:

```
PRINT "AUTO" + "MOBIL"
```

Selbstverständlich können Sie die Strings auch unter Variablen ablegen und diese dann miteinander oder mit Stringoperationen verknüpfen.

Weiterhin lassen sich die Strings durch alle Kapitel 4.1.6 behandelten Vergleichsoperationen miteinander vergleichen. Ein String ist "kleiner" als ein anderer, wenn er im Alphabet vorher steht; also z. B.:

```
"SCHMIDT" < "SCHMITT"
"BAUM" < "GEDANKE"
"APFEL" < "APFELKUCHEN"
```

String-Funktionen

LEN, VAL, STR$, LEFT$, MID$, RIGHT$, STRING$, VGET$, INSTR

LEN (String) ergibt die Anzahl der Zeichen des Strings. Geben Sie folgende Beispiele ein:

```
PRINT LEN("OTTO")
PRINT LEN("234")
```

VAL (String) ergibt den numerischen Wert des Strings. Ist das erste Zeichen des Strings kein +, -, % und keine Zahl, so ist VAL(String)=0.

Testen Sie folgendes Programm:

```
10 A$="22"
20 B$="33"
40 PRINT A$+B$
50 PRINT VAL(A$) * VAL(B$)
```

STR$ (Zahl) wandelt die Zahl zu einem String (die Ziffernfolge, die die Zahl darstellt) um. Beispiel:

```
PRINT STR$(5672)
```

Die folgenden Stringfunktionen bilden einen neuen String, der im Falle:

LEFT$ (String, x) aus x-Zeichen von links besteht ;

RIGHT$(String, x) aus x-Zeichen von rechts besteht ;

MID$(String, x) aus dem Zeichen ab der x.-Stelle besteht ;

MID$(String, x, y) aus dem y-Zeichen ab der x.-Stelle besteht.

Folgendes Programm demonstriert die Wirkungsweise dieser Funktionen an einem Beispiel:

```
10 A$="PAPAGEI"
20 PRINT LEFT$(A$,4)
30 PRINT RIGHT$(A$,2)
40 PRINT MID$(A$,7)
50 PRINT MID$(A$,3,4)
```

Die Funktion STRING$ in der Form

```
STRING$ (n, String)
```

vervielfältigt den in Klammern stehenden String n-mal.

Beispiel:
```
10 A$="KC85/3": N=5
20 B$=STRING$(N, A$)
30 PRINT B$
```

Die Funktion VGET$ liefert den Inhalt der Cursorposition als String. Beispiel:

```
10 CLS
20 PRINT "TEXT"
30 LOCATE 0, 1: A$=VGET$
40 PRINT AT(10, 10);A$
50 LOCATE 1, 0
```

INSTR (A$, B$) ermittelt die Position, ab welcher A$ in der Zeichenkette B$ enthalten ist. Beispiel:

```
10 A$="PFERDE":B$="BLUMENTOPFERDE"
20 PRINT INSTR(A$, B$)
```

Eingabe mit INKEY$

Zur Vereinfachung der Programmbedienung sowie zur Durchführung von Reaktionstest ist die Funktion INKEY$ gleichermaßen gut geeignet.

INKEY$ holt während des Programmablaufs eine Information von der Tastatur, ohne dass das Programm angehalten oder die ENTER-Taste gedrückt werden muss. Dabei ist es jedoch nur möglich, ein Tastenzeichen als String einzulesen. So kann man mit INKEY$ den Dialogbetrieb einfacher gestalten. Wenn wir mit INPUT eine Eingabe realisieren, so müssen wir nach der Eingabe immer noch die ENTER-Taste drücken. Dies können wir mit der Anweisung INKEY$ wie folgt umgehen:

```
1130 A$=INKEY$ : IF A$="" GOTO 1130
```

Das Programm bleibt so lange in der Zeile 1130 bis der Anwender eine Taste drückt. Dieses wird dann als String unter der Variablen A$ abgelegt und steht somit der weiteren Auswertung zur Verfügung.

Merke:

- Strings können durch das Zeichen "+" miteinander verknüpft werden.
- Strings kann man miteinander vergleichen. Ein String ist "kleiner" als ein anderer, wenn er im Alphabet vorher steht.
- Der BASIC-Interpreter verfügt über folgende Stringfunktionen.
  LEN, VAL, STR$, MID$, LEFT$, RIGHT$, STRING$, VGET$, INSTR

  Das Argument der Funktion muss immer in Klammern stehen.
- **Funktion: INKEY$**
  Format: INKEY$
  Bemerkung: INKEY$ liest von der Tastatur ohne Programmunterbrechung und ohne Betätigung der ENTER-Taste genau ein Zeichen ein.
  Beispiel: 130 LET B$=INKEY$: IF B$=""GOTO 130

Übung

Schreiben Sie ein Programm, das zwei einzugebende Strings alphabetisch ordnet. (Lösung im Kapitel 4.1.24).

4.1.15. Anweisung PAUSE

Die Anweisung PAUSE

Sollen Computerbilder eine bestimmte oder eine beliebige Zeit während des Programmablaufs erhalten bleiben, so können wir die Anweisung PAUSE sehr vorteilhaft einsetzen.

```
 10 CLS: PRINT "GEDAECHTNISTRAINING"
 20 PRINT "BITTE ZIFFERNFOLGE MERKEN UND BEI"
 30 PRINT "ABFRAGE '?' EINGEBEN."
 40 PRINT CHR$(17)
 50 COLOR 0,7:WINDOW 13,13,15,25:CLS
 60 FOR I=1 TO 5
 70 CLS
 80 A=INT(RND(1)*1E6):IF A <111111 GOTO 80
 90 PRINT A: PAUSE25: CLS
100 INPUT B: IF A=B GOTO 120
110 CLS: PRINT"FALSCH": PAUSE30: GOTO 130
120 CLS: PRINT"RICHTIG": PAUSE30
130 NEXT
140 WINDOW: COLOR7,1: CLS
```

Wie Sie beim Test dieses kleinen Beispielprogramms gemerkt haben werden, unterbricht die Anweisung PAUSE die Abarbeitung des Programms für einen bestimmten Zeitraum. Dieser Zeitraum wird mithilfe des Parameters t der Anweisung festgelegt. Für eine PAUSE von einer Sekunde gilt dabei der Parameter t = 10. So bewirkt die Anweisung PAUSE 30 z. B. eine Programmunterbrechung von etwa 3 Sekunden.

Erfolgt keine Parameterangabe, so wird das Programm analog der Tastenfunktion STOP bis zum Betätigen der Taste  unterbrochen. Arbeiten Sie mit einzeiligem Fenster, so vergessen Sie bitte nie, vorher den Page-Modus einzustellen (Zeile 40 des letzten Programms). Ist der Scroll-Modus eingestellt, so wird jede Ausgabe sofort aus dem einzeiligen Fenster "gerollt".

- **Anweisung: PAUSE**
  Format: PAUSE [t]
  Bemerkung: Mit der PAUSE-Anweisung wird eine Programmabarbeitung für die Zeit von t*0,1 Sekunden unterbrochen (1 ≤ t ≤ 255).
  PAUSE ohne Argument entspricht der Taste STOP.

  Beispiel:
  ```
  ⋮
  50 PAUSE 250
  ⋮
  ```

4.1.16. Unterprogrammtechnik (Anweisungen GOSUB, RETURN)

Wird ein bestimmter Programmabschnitt mehrmals im Programm benötigt, so ist es effektiv, diesen als Unterprogramm zu schreiben. Mit der Anweisung GOSUB n wird zu dem auf der Programmzeile n beginnenden Unterprogramm gesprungen und dieses abgearbeitet. Nach der Abarbeitung des Unterprogramms, das immer mit der Anweisung RETURN abzuschließen ist, "springt" der Computer zur Aufrufstelle des Hauptprogramms zurück und führt die Arbeit mit der folgenden Anweisung fort.

Die Funktionsweise dieser beiden Anweisungen werden wir uns an folgendem Programm veranschaulichen:

```
10 PRINT "DAS IST DAS HAUPTPROGRAMM HP"
20 GOSUB 100
30 PRINT "WIEDER IM HP"
40 GOSUB 200
50 PRINT "SCHON WIEDER IM HP"
60 GOSUB 100
70 PRINT "SCHLUSS"
80 END
100 PRINT "HIER IST DAS UNTERPROGRAMM 1"
110 RETURN
200 PRINT "HIER IST DAS UNTERPROGRAMM 2"
210 RETURN
```

Hauptprogramm

Unterprogramm 1

Unterprogramm 2

Das Beispiel unterstreicht die bereits erwähnte Bedeutung der Anweisung END. Fehlt diese Anweisung am Schluss des Programms, so wird der Computer nachfolgende Unterprogramme oder Programme als zum Hauptprogramm gehörend erkennen und weiter abarbeiten. Es kommt dabei jedoch zu Fehlermeldungen (Rücksprung mit RETURN nicht definiert).

Merke:

- **Anweisung: GOSUB**
  Format: GOSUB Zeilennummer

- **Anweisung: RETURN**
  Format: RETURN
  Bemerkung: GOSUB ruft eine BASIC-Unterprogramm auf, welches in der angegebenen Zeilennummer beginnt.
  Jedes BASIC-Unterprogramm ist mit RETURN abzuschließen.

  Beispiel:
  ```
  10 INPUT X
  20 GOSUB 300
  30 PRINT E
  40 END
  ```

```
300 E=(X+365)/52
310 RETURN
```

4.1.17. Mehrfache Programmverzweigungen

ON…GOTO…

Haben Sie in einem Programm mehr als zwei Fälle zu unterscheiden, so können Sie die Anweisung ON GOTO sehr vorteilhaft verwenden. Die Anweisung ist wie folgt aufgebaut:

ON i GOTO Liste von Zeilennummern

Der Sprungbefehl verzweigt im konkreten Fall zu der Zeilennummer, die als i-te in der Liste steht.

Veranschaulichen wir uns das an unserem Benzinverbrauch-Bewertungsprogramm aus Kapitel 4.1.6. Diesmal soll die Bewertung wie folgt sein: t – g

Benzinverbrauch / 100 km	Bewertung
≤ 3 l	unmöglich
3 – 6 l	sehr gut
6 – 9 l	gut
9–12 l	schlecht
> 12 l	unmöglich

```
10 INPUT "BENZINVERBRAUCH IN L/100 KM ="; B
20 A = INT(B/3)+1
30 ON A GOTO 40, 50, 60, 70
40 PRINT "UNMOEGLICH":END
50 PRINT "SEHR GUT":END
60 PRINT "GUT": END
70 PRINT "SCHLECHT":END
```

Ist nun z. B. B=8.5, so errechnet der Computer A=3, für dieses spezielle A bewirkt der Befehl in Zeile 30 einen Sprung zur Zeile 60, weil diese Zeilennummer in der Liste der Zeilennummern in Zeile 30 an dritter Stelle steht.

ON…GOSUB…

Die Anweisung in der Form

ON i GOSUB Liste von Zeilennummern

wirkt wie ON…GOTO… mit der Ausnahme, dass bei der Programmverzweigung Unterprogramme angesprungen werden und nach der Abarbeitung dieser das Programm in der ON…GOSUB…-Anweisung folgenden Zeile fortgesetzt wird.

Merke:

- **Anweisung: ON…GOTO…**
  Format: ON i GOTO Liste von Zeilennummern

- **Anweisung: ON…GOSUB…**
  Format: ON i GOSUB Liste von Zeilennummern

  Bemerkung: Die Anweisungen verzweigen das Programm zu der Zeilennummer, die als i-te in der Liste steht. Im Fall von ON…GOSUB… sind die Zeilennummern die Adresse der aufzurufenden Unterprogramme. Die Zeilennummern werden voneinander durch Komma getrennt. Der Ausdruck i wird stets auf eine ganze Zahl abgerundet.
  Ist i = 0 oder größer als die Anzahl der Listenelemente, wird der Befehl ignoriert und die nächste Zeile ausgeführt. Der abgerundete Wert von i muss im Bereich von 0 bis 255 liegen.

  Beispiel: PROGRAMM SCHERE-STEIN-PAPIER

```
 10 RANDOMIZE
 20 WINDOW0,31,0,39:COLOR7,1:CLS:WINDOW
 30 A$="SCHERE":B$="STEIN":C$="PAPIER"
 40 PRINT A$,B$,C$
 50 PRINT:PRINT
 60 PRINT "SPIELREGELN:":PRINT
 70 PRINT "-SCHERE SCHNEIDET PAPIER":PRINT
 80 PRINT "-STEIN SCHLEIFT SCHERE":PRINT
 90 PRINT "-PAPIER WICKELT STEIN EIN":PRINT:PRINT
100 INPUT "WAS WAEHLST DU?";S$
110 IF S$=A$ THEN S=3:PRINT
120 IF S$=B$ THEN S=6:PRINT
130 IF S$=C$ THEN S=9:PRINT
140 IF S=0 THEN PRINT "FALSCHE EINGABE":GOTO 100
150 WINDOW15,18,0,39
160 COLOR0,5:CLS:PRINT "ICH WAEHLE ";
170 C=INT(3*RND(1))
180 ON C+S GOTO 20,20,190,200,210,220,230,240,250,260,270
190 PRINT A$:GOSUB 460: GOTO 280
200 PRINT B$:GOSUB 410:GOTO 280
210 PRINT C$:GOSUB 360:GOTO 280
220 PRINT A$:GOSUB 360:GOTO 280
230 PRINT B$:GOSUB 460:GOTO 280
240 PRINT C$:GOSUB 410:GOTO 280
250 PRINT A$:GOSUB 410:GOTO 280
260 PRINT B$:GOSUB 360:GOTO 280
270 PRINT C$:GOSUB460
280 COLOR 7,1
290 WINDOW 25,27,0,39
```

```
300 PRINT "EIN NEUES SPIEL? (J/N)"
310 X$=INKEY$: IF X$="" GOTO310
320 IF X$="N" THEN WINDOW:CLS:END
330 IF X$ <>"J" GOTO300
340 C=0:S=0:S$=" ":X$=" "
350 GOTO20
360 PRINT
370 PRINT "GRATULIERE! DU HAST GEWONNEN."
380 SP=1:CP=0
390 GOSUB510
400 RETURN
410 PRINT
420 PRINT "PECH FUER DICH! ICH HABE GEWONNEN."
430 SP=0:CP=1
440 GOSUB510
450 RETURN
460 PRINT
470 PRINT "UNENTSCHIEDEN."
480 SP=0: CP=0
490 GOSUB510
500 RETURN
510 SSP=SSP+SP
520 SCP=SCP+CP
530 COLOR 25,6
540 WINDOW21,23,0,39:CLS
550 PRINT "SPIELER","COMPUTER"
560 PRINT SSP, SCP
570 RETURN
```

4.1.18. Eingabe von mehreren Daten

DATA, READ und RESTORE

Bisher haben wir zwei Möglichkeiten kennen gelernt, einer Variablen einen Wert zuzuweisen. Einmal können wir dies durch eine direkte Wertzuweisung (z. B. V=1.2), zum anderen haben wir die Möglichkeit, die Wertzuweisung mithilfe des INPUT-Befehls zu realisieren. Müssen wir nun größere Datenmengen verarbeiten, so erweisen sich beide Methoden aufgrund der immer wiederkehrenden Eingabe als langwierig. Abhilfe schaffen hier die drei oben genannten Anweisungen. Hinter die Anweisung DATA sind die Daten, also die Werte, die wir verarbeiten wollen, zu schreiben. Diese Werte können sowohl numerische als auch Stringvariablen sein.

Mit der Anweisung

READ Variable (,Variable …)

weisen wir den hinter READ stehenden Variablen die hinter DATA stehenden Werte zu.

Testen Sie dazu folgendes Programm:

```
10 DATA 33,2,3,55,84,-9.5723
15 PRINT"NUN STEHT DER DATA-ZEIGER AUF DEM ERSTEN
   WERT"
20 FOR I=1 TO 6
30 READ A
40 PRINT"A HAT JETZT DEN WERT: ";A: PRINT
45 PRINT"NUN WIRD DER DATA-ZEIGER AUF DEN NAECHSTEN
   WERT GESETZT"
50 NEXT
```

Sicher haben Sie an diesem Beispiel die Funktionsweise der beiden neuen Anweisungen schnell verstanden.

Damit der Computer sich merken kann, welchen der Werte er schon ausgelesen hat und welcher somit der nächste ist, gibt es einen internen DATA-Zeiger, der jeweils auf den nächsten abzuarbeitenden Wert der DATA-Liste zeigt. Würden Sie das Programm verändern und die Daten in der Zeile 10 z. B. noch einmal auslesen wollen, käme dort die Fehlermeldung OD (Out of DATA), was bedeutet, dass nicht genügend Daten zum Auslesen bereitstehen.

Diesen Fehler können Sie beheben, indem Sie die fehlenden Daten in Zeile 10 anfügen oder eine neue Programmzeile (z. B. 12) mit DATA und den Werten hinzufügen.

Benötigen Sie jedoch die gleichen Werte in der gleichen Reihenfolge noch einmal, so brauchen Sie nur den DATA-Zeiger auf den ersten Wert zurücksetzen.

Dies können Sie ganz einfach mit der Anweisung RESTORE bewerkstelligen. In unserem erwähnten Beispiel würden Sie dann folgende Programmzeile einfügen:

```
 95 RESTORE
```

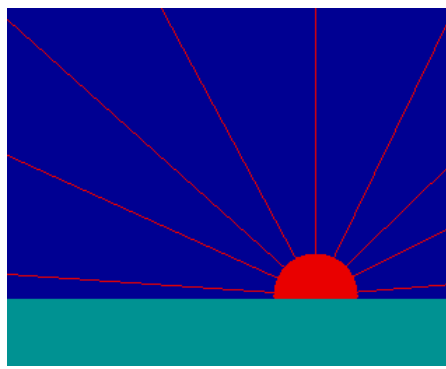
Eine einfache Möglichkeit der geschilderten Programmerweiterung könnte dann wie folgt aussehen:

```
 95 RESTORE
100 FOR I=1 TO 6
110 READ A
120 PRINT "A HALBE HAT JETZT DEN WERT"; A/2
130 NEXT
```

Abschließend soll anhand unseres „Sonnenprogramms“ aus Kapitel 4.1.11 demonstriert werden, wie Sie mithilfe der genannten Anweisungen Programme elegant und wirksam verkürzen können.

Gleichzeitig wurde im Programm die Darstellung der Meeresfläche vereinfacht über die WINDOW-Anweisung abgefahren und die Sonne durch die CIRCLE-Anweisung innerhalb einer FOR-Schleife kürzer und schneller realisiert. Dabei wurden zwei Sonnen um einen Grafikpunkt versetzt gezeichnet, um die Fläche lückenlos mit Vordergrundfarbpunkten "dicht" zu bekommen.

```
 10 COLOR7,1:CLS
 20 X=220:Y=50
 30 FOR R=1 TO 30
 40 CIRCLE X,Y,R,2:CIRCLE X,49,R,2
 50 NEXT
 60 FOR I=1 TO 9
 70 READC,D
 80 LINEX,Y,C,D,2
 90 NEXT
100 WINDOW26,31,0,39:PAPER5:CLS:WINDOW
110 GOTO110
120 DATA 0, 150, 0, 65, 0, 247, 110, 255, 220
130 DATA 255, 319, 255, 319, 146, 319, 100, 319, 58
```



Merke:

- **Anweisung: DATA**
  Format: DATA Konstantenliste
  Bemerkung: DATA kennzeichnet die Liste von numerischen und Stringkonstanten, welche durch READ-Anweisung den einzelnen Variablen zugeordnet werden. Die Listenelemente werden durch Komma voneinander getrennt. Beginnt eine Stringkonstante der Liste mit einem oder mehreren Leerzeichen, so ist diese Konstante in Anführungszeichen zu setzen. Bei mehreren DATA-Listen liest die READ-Anweisung in der Reihenfolge der Zeilennummern aus.

  Beispiel:
  ```
  10 FOR I=1 TO 3
  20 READ X$
  30 PRINT "X$ HAT JETZT DEN WERT"; X$
  40 NEXT
  50 DATA SO, FUNKTIONIERT, DAS
  60 END
  ```

- **Anweisung: READ**
  Format: READ Variablenliste
  Bemerkung: READ liest die mit DATA vereinbarten Werte aus und weist sie den Variablen der Liste in der Reihenfolge der Vereinbarung zu. Die Listenelemente werden durch Komma von einander getrennt.

  Beispiel: siehe Anweisung DATA

- **Anweisung: RESTORE**
  Format: RESTORE [Zeilennummer]
  Bemerkung: RESTORE setzt den DATA-Zeiger auf den ersten Wert der DATA-Liste mit der niedrigsten oder der vereinbarten Zeilennummer, sodass die in der DATA-Anweisung vereinbarten Werte erneut ausgelesen werden können!

  Beispiel: Siehe dieses Kapitel.

4.1.19. Die letzten Tricks bei Programmschwierigkeiten

Sicher ist es Ihnen schon öfter passiert, dass ein Programm bei seinem ersten Lauf Fehler aufwies. Nicht immer waren es nur Schreibfehler. Diese grundlegenden Fehler resultieren meistens aus einer fehlerhaften Problemanalyse oder einer nicht korrekten Umsetzung Ihrer Ideen in BASIC.

Diese Fehler passieren auch Experten immer wieder. Es ist sogar recht ungewöhnlich, dass ein Programm auf Anhieb fehlerfrei läuft. In unseren bisherigen kleinen Programmen sind die Fehler oft, insbesondere auch durch die Fehlermeldungen, recht schnell auszumachen. Je umfangreicher die Programme sind, umso leichter schleichen sich Fehler ein und umso schwieriger sind diese dann zu finden.

Dieses Kapitel möchte Ihnen ein paar Tipps zur Vermeidung und zum schnellen Finden solcher Fehler geben.

Programmerstellung

Die meisten Fehler können wir durch eine gründliche Programmvorbereitung vermeiden. Wie sollte solche Vorbereitung nun konkret aussehen?

Am Anfang eines Programms steht meistens ein Problem aus der Praxis. Dieses Problem gilt es, als erstes genau zu erörtern. Wir überlegen uns also, welche Größen uns bekannt sind, in das Programm eingehen und welche Größen oder Antworten ausgegeben werden sollen. Nun denken wir uns noch eine prinzipielle Lösungsmöglichkeit aus und gliedern diese grob in einzelne Schritte. Diese Schritte übertragen wir dann in einen Programm-Ablauf-Plan, kurz PAP genannt. Ein PAP legt die Struktur eines Programms sehr übersichtlich dar, sodass grundlegende Fehler sofort ins Auge fallen. Als Beispiel wollen wir uns diese und die weiteren Schritte zum Erstellen eines lauffähigen Programms anhand eines PAP anschauen.

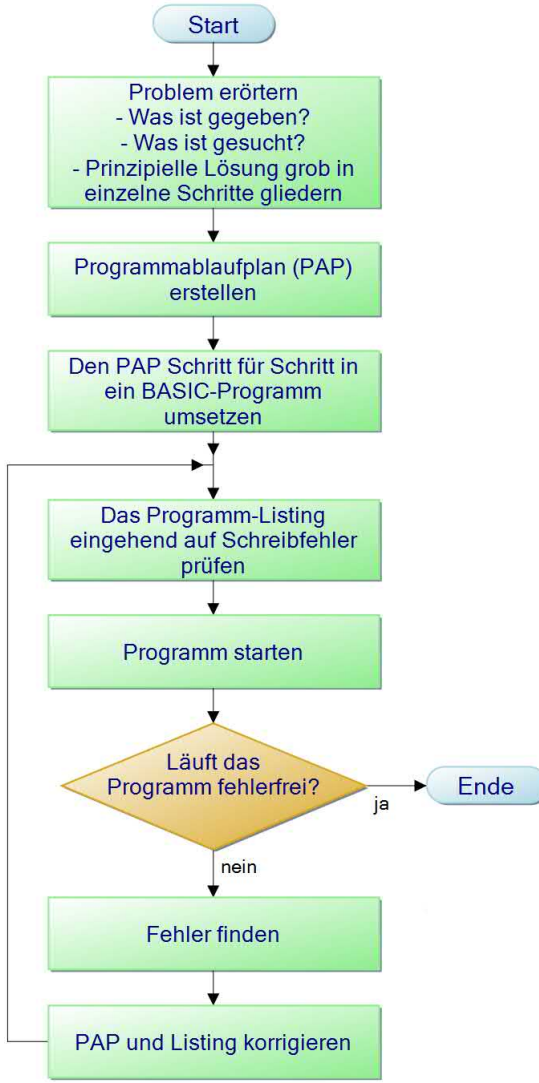
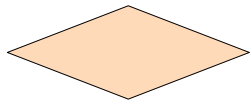


Bild 33: Programmablaufplan zum Erstellen eines lauffähigen Programms

In einem PAP symbolisieren

ein		Marken, z. B. den Anfang oder das Ende eines Programms
ein		eine oder mehrere Anweisungen
ein		eine antwortbedingte Verzweigung des Programms

Anschließend schauen wir uns unser ursprüngliches Noten-Bewertungs-Programm aus Kapitel 4.1.8 im PAP an.

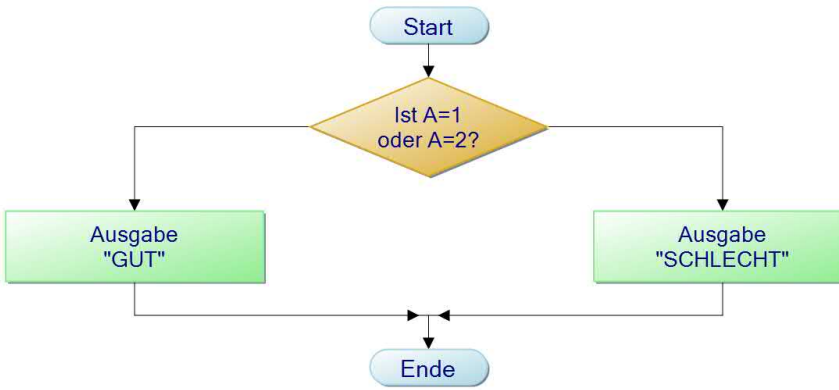


Bild 34: Symbolische Darstellung Noten-Bewertungsprogramm

Das Finden von Fehlern

Trotz aller Bemühungen lassen sich Fehler oft nicht vermeiden. Deshalb wollen wir jetzt auf das kleine Anweisungskästchen unseres ersten PAP, das da "Fehler finden" heißt, eingehen. Den ersten Hinweis auf unseren Fehler bekommen wir durch die Fehlermeldung bei Abbruch des Programms. Mit dieser Information und einer nochmaligen Prüfung finden wir viele Fehler im Listing.

Oft jedoch wird das Programm gar nicht unterbrochen, sondern der Computer macht einfach etwas anderes, uns völlig Unverständliches. In diesem Fall können wir die Anweisung TRON zur besseren Verfolgung des Programmablaufs einsetzen. Die Anweisung bewirkt, dass bei Abarbeitung eines Programms die Nummern der Zeilen in der Reihenfolge ihrer Abarbeitung angezeigt werden. Geben Sie ein:

```
10 FOR I=1 TO 5
20 PRINT I
30 NEXT
TRON
```

Mit dem Kommando TRON haben Sie den eben beschriebenen Anzeigemodus eingestellt. Bei Ablauf des eingegebenen Programms gelangen die Nummern der abgearbeiteten Zeilen zur Anzeige. Der Modus wird durch das Kommando TROFF wieder ausgeschaltet.

Führt auch das nicht zum Erfolg, so können wir die Anweisung STOP in das Programm einfügen und so an jeder beliebigen Stelle anhalten. Nach einer Programmunterbrechung mit STOP meldet sich der Computer wie bei einer Unterbrechung durch die BRK-Taste, z. B. mit

```
BREAK IN 30
```

Durch Eingabe der Anweisung CONT und das Betätigen der ENTER-Taste können Sie das Programm fortsetzen. Testen Sie die Anweisung, indem Sie sie in unser Programm wie folgt einfügen:

```
25 STOP
```

Darüber hinaus können Sie das Programm auch direkt während des Programmablaufs durch drücken der STOP-Taste anhalten. Hier wird das Programm, wie bereits in Kapitel beschrieben, durch Betätigen der  -Taste fortgesetzt.

Nun noch ein letzter Tipp zur Fehlersuche:

Lassen Sie sich durch Einfügen der PRINT-Anweisung die Variablen vor und nach den kritischen Punkten zur Kontrolle ausgeben.

Merke:

- **Anweisung: TRON**
  Format: TRON
  Bemerkung: TRON bewirkt, dass bei Abarbeitung eines Programms die-Nummern der Zeilen in der Reihenfolge ihrer Abarbeitung zur Anzeige gebracht werden.

- **Anweisung: TROFF**
  Format: TRON
  Bemerkung: TROFF schaltet diesen Anzeigemodus wieder aus.
  Beispiel: Siehe dieses Kapitel

- **Anweisung: STOP**
  Format: STOP
  Bemerkung: STOP unterbricht gezielt ein Programm. Der Programmablauf kann durch Ausführung der Anweisung CONT fortgesetzt werden.

  Beispiel:
  ```
  10 X=3
  15 STOP
  20 PRINT X
  30 END
  ```

Übung

Zeichnen Sie zu dem Benzinverbrauch-Bewertungsprogramm von Kapitel 4.1.17 einen PAP.

4.1.20. Musik (Anweisungen SOUND und BEEP)

Tonausgabe

Mit dem KC 85 können Sie Töne monofon über das Fernsehgerät bei FBAS- oder RGB-Anschluss zu Gehör bringen. Dabei ist die Lautstärke in 16 Stufen regelbar (Schrittweite 2).

Darüber hinaus erfolgt die Tonausgabe auch stereofon mittels einer Stereoanlage oder eines anderen geeigneten Musikwiedergabegerätes. Hierbei wird der Computeranschluss TAPE über das Diodenkabel mit dem Wiedergabegerät verbunden.

Im einfachsten Fall drücken Sie die Schnellstopp-Taste Ihres Recorders und schalten diesen auf Aufnahme. Nun funktioniert der Recorder nicht mehr als Speichereinheit, sondern als Musikwiedergabeeinheit des Computers. Achten Sie jedoch bitte in Ihrem eigenen Interesse darauf, dass keine auf dem Band gespeicherten Programme gelöscht werden!

Auch das bereits in Kapitel 4.1.1 erwähnte Fehlersignal kann nun über die eben beschriebenen Wege realisiert werden.

Technische Angaben zur Tonausgabe finden Sie im Kapitel 2.1.5 auf Seite 90.

Töne

Töne sind die Bausteine der Musik und etwas weiter gesehen aller Geräusche. Sie unterscheiden sich über die Tonhöhe, die Tondauer und die Lautstärke. Der KC 85 verfügt über zwei Kanäle, über einen Tonhöhenumfang von 7 Oktaven und ist in 16 Lautstärke-Stufen programmierbar. Wollen wir nun einen bestimmten Ton erzeugen, so müssen wir demzufolge angeben:

- auf welchem Kanal
- wie lange
- in welcher Tonhöhe
- in welcher Lautstärke

die Tonausgabe erfolgen soll.

Diese Angaben erfolgen mithilfe der Anweisung SOUND in der Form:

SOUND z_1 , v_1 , z_2 , v_2 , l_s , t_d

Dabei legen die Parameter z_1 und v_1 die Tonhöhe des Kanals 1 und die Parameter z_2 und v_2 die Tonhöhe des Kanals 2 entsprechend der Tonwerttabelle für Vorteiler und Zeitkonstanten auf Seite 244 fest. Mit den Parametern l_s und t_d werden die Lautstärke ($0 \le l_s \le 31$) und die Tondauer ($0 \le 255$) bestimmt.

Beachten Sie folgende Sonderfälle:

- werden l_s und t_d nicht angegeben, so bleiben die vorherigen Werte erhalten;
- ist die Lautstärke = 0, so erfolgt keine Tonausgabe über das Fernsehgerät, jedoch weiterhin über die Diodenbuchse;
- sind $z_1 = 0$ oder $z_2 = 0$, so wird kein Ton ausgegeben;
- ist die Tondauer = 0, so wird bis zum nächsten Aufruf ein Dauerton abgegeben

Mit dem folgenden Programm soll ein Beispiel aus der Fülle von Anwendungsmöglichkeiten der akustischen Möglichkeiten des Computers demonstriert werden. Im Programm wird die Tastatur als Klaviatur genutzt. Durch das Drücken einer der Zifferntasten 1 bis 8 wird ein Ton der C-Dur-Tonreihe erzeugt. Die Zuordnung der Töne zu den Tasten ist dem Bildschirm zu entnehmen. Die Tonausgabe wird durch Drücken der Leertaste beendet. Das Programm kann nur durch die BRK-Taste beendet werden.

```
 10 WINDOW: COLOR7,1: CLS
 20 PRINTAT (14,12); "c d e f g a h c"
 30 PRINTAT (16,12); COLOR 0,7; "1 2 3 4 5 6 7 8"
 40 A$=INKEY$: IF A$="" GOTO 40
 50 IF ASC(A$)=32 THEN X=0: GOTO 100
 60 Z=VAL(A$):IFZ<1 OR Z>8 GOTO 40
 70 FOR I=1 TO Z
 80 READ X
 90 NEXT
100 SOUND X, 0, 0, 0, 31, 0
110 RESTORE: CLEAR: GOTO 40
120 DATA 216, 192, 171, 162, 144, 128, 114, 108
```

Die Zuordnung der Töne zu den Zeitkonstanten bzw. Vorteilerstufen kann nach Tabelle 41 auf Seite 244 ermittelt werden.

Darüber hinaus können Tonfolgen oder Lieder zur Unterstützung der Aussagekraft fest programmiert werden. Aber es ist z. B. auch möglich, den Computer selbstständig mithilfe der RND-Funktion improvisieren zu lassen. Wie bereits gesagt, es gibt viele Anwendungsmöglichkeiten. Um ein einfaches akustisches Signal zu erzeugen, benutzt man am besten die Anweisung BEEP.
Probieren Sie aus:

```
BEEP
```

Der bei der Ausführung der Anweisung erzeugte Ton entspricht einer SOUND-Anweisung mit dem Vorteiler v = 0 und der Zeitkonstante z = 48. Es ist der gleiche Ton, der auch bei den Fehlermeldungen ausgegeben wird. Mit einem der Anweisung folgenden Parameter können Sie festlegen, wie oft das Signal ausgegeben werden soll. Testen Sie es selbst:

```
BEEP 4
```

Merke:

- **Anweisung: BEEP**
  Format: BEEP [n]
  Bemerkung: Mithilfe der Anweisung BEEP werden akustische Signale mit festgelegter Länge erzeugt. Der Parameter n bestimmt die Anzahl der Signale (0 ≤ n ≤ 255)
  Entfällt der Parameter, so gilt n = 1.

  Beispiel:
  ```
  ⋮
  130 BEEP
  ⋮
  ```

- **Anweisung: SOUND**
  Format: SOUND z_1 , v_1 , z_2 , v_2 [, l_s [, t_d]]
  Bemerkung: Die Anweisung SOUND gestattet die Ausgabe von Tönen mit steuerbarer Tonhöhe, Tonlänge und Lautstärke.
  Parameterfestlegung:
  z_1 - Zeitkonstante für CTC Kanal 1
  v_1 - Vorteiler für CTC Kanal 1
  z_2 - Zeitkonstante für CTC Kanal 2
  v_2 - Vorteiler für CTC Kanal 2
  l_s - Lautstärke
  t_d - Tondauer
  mit $1 \le z_i \le 255$ ($z_i = 0$: Ton ausgeschaltet)

  $$vi = \begin{cases} 0 \\ 1 \end{cases}$$

  $1 \le l_s \le 31$ ($l_s = 0$: Tonausgabe über TV gesperrt)
  $1 \le t_d \le 255$ ($t_d = 0$: Dauerton)

  Beispiel: SOUND 15,1,16,1,31,100

4.1.21. Variablenfelder

Dimensionierung

Bisher haben wir Variablen der Form X, H3, J$ oder R benutzt. Treten jedoch größere Variablenmengen auf, so bedient man sich in der Mathematik indizierter Variablen, also solcher Variablen, die sich durch Indizes (kleine, am Fuß der Variablen befindliche Zahlen, wie z. B. x_1 , x_2) unterscheiden. Diese Möglichkeit bietet selbstverständlich auch unserer Computer.

Eine Gruppe von Variablen, die sich nur durch ihre Indizes unterscheiden, wird dabei ein Variablenfeld genannt. Der Index der Variablen kommt nicht nach unten versetzt „an den Fuß“ der Variablen, sondern wird in Klammern hinter die Variable geschrieben (also X(0), X(1), X(2) usw.).

Dieses Variablenfeld X(i) ist eindimensional, da es nur einen Index enthält. Wir können jedoch auch zweidimensionale oder n-dimensionale Variablenfelder vereinbaren. Diese besitzen dann 2 bzw. n-Indizes. Bei einem zweidimensionalen Variablenfeld wird auch der Name verständlicher. Unser oben angeführtes eindimensionales Feld könnte z. B. eine Reihe von i Pflanzen darstellen. Nehmen wir jetzt jedoch einen zweiten Index für die Anzahl der Reihen dazu, so ergibt sich ein Feld auch anschaulich. Die Feldfestlegung oder besser Dimensionierung der Variablenfelder erfolgt mithilfe der Anweisung DIM. Das mit der Anweisung

```
10 DIM X(2,3)
```

dimensionierte zweidimensionale Variablenfeld besteht konkret aus folgenden Variablen:

```
X(0,0), X(0,1), X(0,2), X(0,3)
X(1,0), X(1,1), X(1,2), X(1,3)
X(2,0), X(2,1), X(2,2), X(2,3)
```

Die durch die Anweisung reservierten Speicherplätze der 12 Variablen sind alle auf 0 gesetzt.

Im weiteren Verlauf des Programms können wir jede einzelne Variable mit einem Wert belegen und mit diesen Variablen wie gewohnt operieren. Dies probieren Sie am besten einmal selbst aus, in dem Sie die obenstehende Programmzeile 10 zu einem Testprogramm ergänzen:

```
20 X(1,2)=8
30 FOR I=0 TO 2
40 FOR J=0 TO 3
50 PRINT X(I; J);
60 NEXT J
70 PRINT
80 NEXT I
```

Sie sehen, lediglich die Variable X(1,2) ist ungleich 0, da ihr in der Zeile 20 unseres Programms der Wert 8 zugewiesen wurde. So wie wir bisher numerische Variablenfelder betrachtet haben, können wir selbstverständlich auch Stringvariablenfelder anlegen: Das nächste Programm demonstriert am Beispiel eines eindimensionalen, aus 12 Elementen bestehenden Stringvariablenfeldes, einfache Möglichkeiten zur effektiven Nutzung von Variablenfeldern.

Das Programm gibt Ihnen über Menü-Technik wahlweise die Möglichkeit, die Elemente des Feldes über die Tastatur einzugeben, sie als Datei vom Recorder einzulesen, die Elemente aufzulisten oder sie als Datei auf Magnetband zu speichern. Dabei lernen wir zwei uns bereits bekannte Anweisungen in einer neuen Form kennen:

```
CSAVE * "Name"; Feldname
CLOAD * "Name"; Feldname
```

Mit diesen Anweisungen kann das bezeichnete Variablenfeld (in unserem Beispiel A$) als Datei gespeichert und wieder in den Computer geladen werden.

Die Vorgänge des Rettens und des Ladens werden, wie im Kapitel 4.1.9 für Programme beschrieben, ausgeführt. Wie das Beispiel zeigt, können diese Anweisungen nicht nur als Kommando, sondern auch geschickt als Programmanweisung eingesetzt werden. Vor dem Laden eines Variablenfeldes ist diese stets mit der Anweisung DIM festzulegen.

```
 10 DIM A$(11)
 20 CLS:PRINT" ARBEIT MIT STRINVARIABLENFELDERN":PRINT
 30 PRINT" 1 VARIABLEN EINGEBEN"
 40 PRINT" 2 VARIABLEN EINLESEN"
 50 PRINT" 3 VARIABLEN LISTEN"
 60 PRINT" 4 VARIABLEN RETTEN"
 70 B$=INKEY$: IF B$=""GOTO 70
 80 B=VAL(B$):PRINT B
 90 IF B=1 OR B=2 OR B=3 OR B=4 OR B=5 GOTO100:ELSEGOTO70
100 ON B GOTO 110,140,170,220
110 FOR I=0 TO 11
120 INPUT A$(I):NEXT
130 GOTO 20
140 PRINT"RECORDER EINSCHALTEN !"
150 CLOAD* "TESTFELD";A$
160 GOTO 20
170 FOR I=0TO11
180 PRINT " A$("I")=";A$(I):NEXT
190 PRINT:PRINT" EINGABE M FUER MENU"
200 M$=INKEY$: IF M$<>"M" GOTO200:ELSE GOTO 20
210 GOTO 20
220 PRINT"RECORDER EINSCHALTEN !":PAUSE 50
230 CSAVE* "TESTFELD";A$
240 GOTO 20
```

Insbesondere bei der Arbeit mit Variablenfeldern wird der vorhandene Speicherplatz schnell überschritten. Dabei wird für numerische Variablen der Speicherbereich nur durch die verfügbare Arbeitsspeicherkapazität begrenzt. Hier hilft dann nur noch eine Systemerweiterung durch RAM-Module.

Für Stringvariablen ist nach dem Start des BASIC-Interpreters ein Speicherbereich von 256 Bytes reserviert. Reicht dieser nicht aus, so können wir den Stringspeicherbereich leicht erweitern. Dazu erfahren Sie jedoch im nächsten Kapitel mehr.

Merke:

- **Anweisung: DIM**
  Format: DIM Variable (Index, … , …)
  Bemerkung: DIM reserviert Speicherplatz für Feldvariablen und setzt diese gleich Null. Die Liste der Indizes kann maximal eine Programmzeile lang sein. Der Wert jedes Ausdrucks gibt den größten zulässigen Index der Dimension an. Der kleinstmögliche ist 0. Es sind beliebig viele Indizes zulässig. Wird keine Dimension vereinbart, wird für jede angegebene Dimension ein größtmöglicher Index von 10 angenommen. Die Indizes sind durch Komma getrennt in Klammern zu setzen.

  Beispiel: 10 DIM X(5,10)

- **Anweisung: CSAVE***
  Format: CSAVE* "Name"; Feldname
  Bemerkung: Die Anweisung speichert das angegebene Feld als Datei ab. Sie kann auch als Programmanweisung verwendet werden.

  Beispiel: CSAVE* "DATEN";X

- **Anweisung: CLOAD***
  Format: CLOAD* "Name"; Feldname
  Bemerkung: Die Anweisung lädt das angegebene Feld aus einer Datei. Sie kann auch als Programmanweisung verwendet werden.

  Beispiel: CLOAD* "DATEN";X

4.1.22. Innenleben des Computers

Wenn wir uns mit dem Innenleben des Computers etwas vertraut machen wollen, müssen wir uns darüber im klaren sein, dass wir bis jetzt bei unserer BASIC-Programmierung noch gar nicht direkt mit dem "Gehirn" des Computers, dem U880D gesprochen haben. Dieser Mikroprozessorschaltkreis versteht nämlich nur seinen U880-Maschinencode. Der BASIC- Interpreter stellt das Bindeglied zwischen uns und dem Maschinencode dar. Für größere Programme und sehr schnelle Programme gibt es in BASIC jedoch einige Anweisungen, deren Verständnis ohne ein Mindestmaß an Kenntnissen über das Computer-Innenleben nicht möglich ist.

Der Computer benutzt intern nicht nur eine andere Sprache, sondern er zählt und rechnet auch anders. Dies liegt in seiner dualen Grundstruktur begründet. Die kleinste Informationseinheit, die der Computer kennt, ist ein Bit. Ein Bit kann nur eine von zwei möglichen Informationen tragen. Diese Informationen können wir auch als Zahlen ansehen, also 1 oder 0. Zwei Bits können demnach vier Zahlen zusammen darstellen, nämlich: 00, 01, 10, 11. Überlegen wir uns anhand des folgenden Schemas, wie diese Entwicklung weiter geht:

1 Bit kann	2^1 = 2 Zahlen darstellen
2 Bit können	2^2 = 4 Zahlen darstellen
3 Bit können	2^3 = 8 Zahlen darstellen
4 Bit können	2^4 = 16 Zahlen darstellen
8 Bit können	2^8 = 256 Zahlen darstellen

Dabei kommt den 4 Bit und 8 Bit als Einheit eine besondere Bedeutung zu. Jeweils 4 Bit fasst der Computer als eine Ziffer auf. Damit stehen dem Computer nicht nur 10 Ziffern wie üblich von 0 bis 9 zur Verfügung, sondern 16. Diese heißen der Reihe nach:

0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, F.

Das interne Zahlensystem des Computers baut also auf die Basis-Zahl 16 auf. Es heißt deswegen, im Gegensatz zu unserem dezimalen oder 10er Zahlensystem, hexadezimales Zahlensystem. Hexadezimale Zahlen werden mit einem H hinter der Zahl gekennzeichnet.

Beispiel:

$23H = 2 * 16^1 + 3 * 16^0 = 32 + 3 = 35$

Zwei mal vier Bit oder zwei hexadezimale Zahlen sind 8 Bit oder ein Byte. Unser Computerspeicher ist in jeweils 8 Bit, also in Bytes aufgeteilt. Jedes Byte besitzt eine eigene Adresse. Die Adressen bestehen aus hexadezimalen Zahlen. Der Prozessor kann direkt 2^{16} Byte = 2^6 mal 2^{10} Byte = 64 KByte zu je 8 Bit adressieren (1 KByte = 1024 Byte). Die Adressen laufen dabei von 0000 bis FFFFH.

Mit folgenden kleinen Programmen können wir dezimale in hexadezimale bzw. hexadezimale in dezimale Zahlen umwandeln.

```
 10 A$="0123456789ABCDEF"
 20 Z$=""
 30 INPUT" DEZIMALZAHL ?";D
 40 R=D-16*INT(D/16)
 50 Z$=MID$(A$,R+1,1)+Z$
 60 IFD=0THEN90
 70 D=INT(D/16)
 80 GOTO 40
 90 PRINT"HEXADEZIMALZAHL :";RIGHT$ (Z$,LEN(Z$)-1)
100 GOTO 20
```

```
 10 INPUT"HEXADEZIMALZAHL ?";Z$
 20 L=LEN(Z$)
 30 D=0
 40 FOR I=1 TO L
 50 T$=MID$(Z$,I,1)
 60 IFT$>="A" AND T$<="F" THEN U=ASC(T$)-55:GOTO90
 70 IF T$>="0" AND T$<="9" THEN U=VAL(T$):GOTO90
 80 PRINT" KEINE HEXADEZIMALZAHL !":GOTO10
 90 D=U*16^(L-I)+D
100 NEXT
110 PRINT“DEZIMALZAHL :“;D
120 GOTO 10
```

Greifen Sie jedoch vom BASIC-Interpreter direkt zum Speicher, müssen Sie nicht unbedingt die hexadezimale Adresse angeben. Die Speicherübersicht des Handbuches enthält sowohl die dezimalen als auch die hexadezimalen Adressen.

Der Speicher selbst besteht aus zwei Arten, dem ROM und dem RAM. Aus dem ROM, dem Festwertspeicher, können wir nur Informationen auslesen, aber nichts hineinschreiben. Er enthält z. B. das Betriebssystem und den BASIC-Interpreter. Unter dem Betriebssystem verstehen wir die Gesamtheit der Grundprogramme, die nach dem Einschalten des Computers sofort selbstständig starten (z. B. Programme zur Tastaturabfrage, Programme zum Aussenden des Kontrollbildes u. ä.) bzw. verfügbar sind (z. B. Programme zur Zusammenarbeit mit dem Magnetbandgerät: SAVE, LOAD, VERIFY). Dieser Festspeicher hat im KC 85/3 einen Umfang von 16 KByte (10 KByte BASIC-Interpreter, 6 KByte Betriebssystem). Im KC 85/5 umfasst der Festspeicher 48 KByte.

Im RAM, dem Arbeitsspeicher, befinden sich alle Programme, die wir mit dem Kassettengerät oder der Tastatur eingeben. Aus diesem RAM können wir sowohl lesen als auch Informationen hineinschreiben. Beim Ausschalten des Computers gehen jedoch sämtliche Programme dieses Speicherbereiches verloren. Der RAM hat im KC 85/3 einen Umfang von 32 KByte. Im KC 85/5 sind es 320 KByte.

Dieser RAM steht uns jedoch nicht uneingeschränkt zur Verfügung, da wir auch RAM-Speicherplatz für den Bildwiederholspeicherplatz und für die Grundprogramme benötigen.

Aber dazu erfahren wir mehr im nächsten Abschnitt.

Speicherverwaltung

Die Speicherverwaltung des Computers wird in diesem Abschnitt der Einfachheit halber am Beispiel des KC 85/3 erläutert (für den KC 85/4-5 siehe Bild 24 auf Seite 122).

Der U880D kann direkt einen Speicherbereich von 64 KByte adressieren. Die 16 KByte ROM und der 32 KByte RAM der Grundausstattung sind dabei wie folgt verteilt:

Speicheradressen	Speicherbelegung
0000H – 3FFFH	RAM (Arbeits-RAM)
4000H – 7FFFH	unbestückt (für RAM-Erweiterung)
8000H – BFFFH	RAM (IRM – Bildwiederholspeicher)
C000H – FFFFH	ROM (Betriebssystem und BASIC-Interpreter)

Der erste 16 KByte RAM-Block, also der Arbeits-RAM, steht Ihnen für Ihre speziellen Programme fast vollständig frei zur Verfügung. Der zweite 16 KByte-RAM-Block ist reserviert für den Bildwiederholspeicher. Dieser enthält die Informationen für das Anzeigebild und ermöglicht die Vollgrafik von 320 x 256 Bildpunkten. Der ROM enthält das Betriebssystem und den BASIC-Interpreter.

Im Bild 35 ist die Speicherverwaltung am Beispiel des KC 85/3 übersichtlich dargestellt.

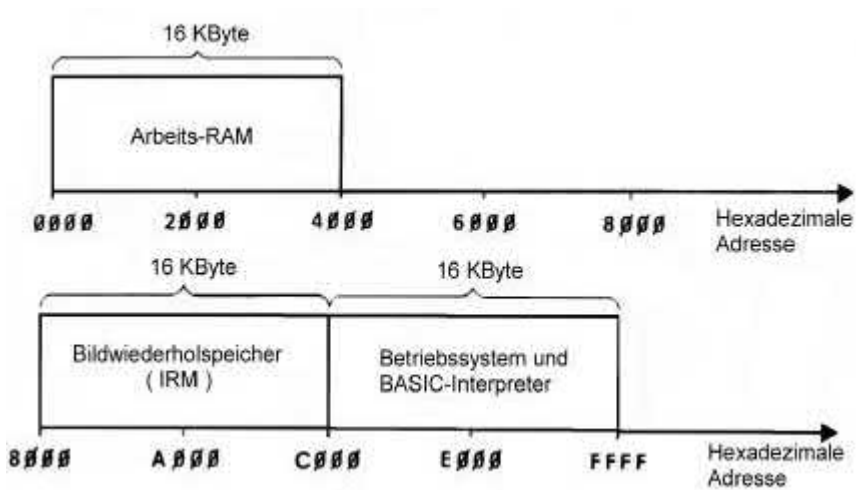


Bild 35: Speicherordnung am Beispiel des KC 85/3

Die nicht belegten Speicheradressen sind für mögliche Speichererweiterungen vorgesehen.

Diese, wie auch die im Grundgeräte bereits enthaltenen Speicherblöcke, können durch die Anweisung

SWITCH m, k

zu- oder abgeschaltet und schreibgeschützt werden. Die Parameter m (Steckplatzadresse) und k (Steuerbyte) sind, wie im Kapitel 1.5.5. ab Seite 64 beschrieben, zu bilden und in dezimaler Form durch Komma voneinander getrennt einzugeben. Eine detaillierte Speicherbeschreibung des KC 85/5 finden Sie im Kapitel 3.2 ab Seite 121.

Wie bereits dargelegt, müssen wir beim Erstellen größerer Programme oder bei der Verarbeitung großer Datenmengen mit dem uns zur Verfügung stehenden Speicherplatz haushalten. Dieses bemühen unterstützt insbesondere die Funktion FRE. Mithilfe der Funktion FRE können wir zu jeder Zeit die Anzahl der noch freien Speicherplätze im RAM oder die Anzahl der freien Bytes im Stringspeicherbereich erfahren. Erweist sich der Stringspeicherbereich als zu klein für ein geplantes Programm, so können wir diesen durch die Anweisung

CLEAR Ausdruck, Ausdruck

neu festlegen. Dabei legt der Wert des ersten Ausdrucks fest, wie viel Bytes für Strings reserviert werden sollen. Mit dem zweiten Ausdruck kann die zugewiesene Grenze des oberen RAM-Bereiches neu bestimmt werden. Die beiden erwähnten Speicheranweisungen werden im letzten Abschnitt des Kapitels ausführlich beschrieben.

Maschinencode und BASIC

Aus der Speicheraufteilung ergeben sich auch die Inhalte. So sind z. B. im Bildwiederholspeicher auf genau bestimmten Speicherplätzen die Farb-, Vordergrund- und Hintergrund-Information abgelegt. Diese Informationen bestehen in der Grundform, wie auch die Informationen der Programme, aus zweistelligen Hexadezimalzahlen. Sind die zu speichernden Werte oder Anweisungen länger als ein Byte bzw. zwei hexadezimalen Zahlen, so werden diese über mehrere Speicheradressen abgelegt.

Eine Einführung in die Arbeit mit dem hexadezimalen Maschinencode würde den Rahmen dieser BASIC-Anleitung Buches sprengen. Dazu verwenden Sie bitte folgende weiterführende Literatur: [21], [60] oder [61] sowie die Ausführungen in den Kapiteln 3 (Software), 4.2 (Assembler ASM) bzw. 4.3 (Disassembler) in diesem Handbuch.

Aber wir können auch ohne diese Spezialkenntnisse direkt aus dem Speicher lesen und einschreiben. Die Werte werden mittels der Anweisungen PEEK und DEEK aus dem Speicher gelesen. Durch die Anweisungen POKE und DOKE werden Werte in die Speicherzellen geschrieben. Sowohl die Speicheradressen

als auch die Speicherinhalte sind dabei in gewohnter dezimaler Schreibweise einzugeben bzw. werden in dieser ausgegeben. Mithilfe dieser Anweisungen kann man somit kleine Maschinenprogramme vom BASIC-Interpreter aus in den dafür geeigneten Speicherbereich schreiben.

Nicht geeignet ist der Bildwiederholspeicherblock, da dieser bei der Arbeit des BASIC-Interpreters abgeschaltet wird. Hier können wir jedoch mithilfe der Anweisungen VPEEK Zellen aus dem Speicherinhalt im Bereich von 8000H bis BFFFH lesen und durch VPOKE Werte in diesem Adressbereich eintragen. Bei Verwendung dieser Anweisungen wird der Adressbereich des Bildwiederholspeichers auf 0000H bis 3FFFH (0 bis 16.383 dezimal) gelegt. Deshalb sind in den Anweisungen nicht die tatsächlichen Adressen, sondern die angesprochenen Adressen abzüglich 8000H anzugeben. Da die Adressangaben dezimal erfolgen, müssen wir also von der dezimalen Adresse 32.768 subtrahieren.

Möchten wir z. B. den Inhalt der Speicherzelle BA00H (47.616 dezimal) erfahren, so müssen wir von der dezimalen Adresse 47.616 die oben genannten 32.768 subtrahieren. Die in der Anweisung PEEK zu verwendende Adresse ergibt sich zu 14.848.

Die Anweisung lautet:

```
PRINT PEEK(14848)
```

Mithilfe der eben genannten sechs Anweisungen sind wir also in der Lage, kleine Maschinenprogramme, Zeichenbildtabellen oder andere Dateien vom BASIC-Interpreter aus zu erstellen.

Solche Maschinenprogramme kann man dann auch innerhalb eines BASIC-Programms durch die Anweisung CALL oder die Funktion USR aufrufen und abarbeiten lassen.

Die Funktion USR(x) ruft ein Maschinenprogramm mit dem Argument x auf. Dabei sind folgende Schritte erforderlich:

1. Speichern der Anfangsadresse des Maschinenprogramms auf Adresse 304H (L-Teil) und 305H (H-Teil)
   (L-Teil…niederwertiger Teil, z. B. 80H bei 3080H; H-Teil…höherwertiger Teil, z. B. 30 bei 3080H)
2. Aufruf der USR-Funktion

Beispiel: Ein Maschinenprogramm wird mit dem MODIFY-Kommando des Betriebssystems CAOS auf eine freie Adresse (z. B. 0H) eingegeben:

Adresse	MC	Anweisung	Bemerkung
0000	CD 6F C9	START: CALL CPRVL3	; BASIC-UP zur Parameter- ; übergabe in Register DE
0003	13	INC DE	; Von BASIC übergebene ; Parameter
0004 0005	7A 43	LD A,D LD B,E	; Reg. A, B –Funktionswert- ; rückgabe
0006	CD B1 D0	CALL FRE3	; BASIC-UP zur Funktions- ; wertzuweisung an ; BASIC- ariable
0009	C9	RET	; Rücksprung in BASIC

Die Eingabe kann aber auch vom BASIC-Programm erfolgen:

```
 10 DATA 205,111,201 ;REM Maschinencode
 12 DATA 19, 122, 67 :REM als Dezimalzahlen
 14 DATA 205,117,208
 16 DATA 201
 20 AD=0 :REM Anfangsadresse des Maschinenprogramms
 30 FOR I=0TO9
 40 READ B:POKE AD + I,B
 50 NEXT
 60 REM Speichern der Anfangsadresse auf 304H
 70 DOKE 772, AD
 80 REM Aufruf des Maschinenprogramms
 90 FOR I=250 TO 260
100 K=USR(I)
110 PRINT "I=";I;"USR(I)=";K
120 NEXT
```

Dieses Beispiel-Maschinencode-Unterprogramm erhöht den eingegebenen Parameter um 1 (INC DE) und gibt diesen Wert als Funktionswert zurück.

Dieses Maschinencode-Unterprogramm funktioniert zunächst unabhängig von der Tatsache, dass der IRM beim Zugriff auf den BASIC-Arbeitsspeicher abgeschaltet wird. Verallgemeinert ergeben sich aus dieser Arbeitsweise des BASIC-Interpreters jedoch folgende Konsequenzen für den Aufruf von Maschinenunterprogrammen vom BASIC aus:

- Liegt ein Maschinenprogramm im IRM, so muss vor seinem Aufruf der IRM zugeschaltet werden. Dies kann durch ein weiteres Maschinenunterprogramm geschehen, das aber im Adressbereich 0000H bis 7FFFH liegen muss.
- Soll von einem Maschinenunterprogramm aus, das im BASIC-Arbeitsspeicher liegt, auf den IRM zugegriffen werden, z. B. zur Realisierung von Bild-

schirmausschriften (Text und/oder Grafik), muss der IRM ebenfalls zugeschaltet werden.

- Vor der Rückkehr ins BASIC muss der IRM wieder abgeschaltet werden.

Die folgende Befehlsfolge realisiert das Ein- und Ausschalten des IRM mithilfe spezieller Unterprogramme des Betriebssystems, die außerdem den STACK des BASIC-Interpreters verlegen und den Inhalt des BC-Registers zerstören.

```
CD 18 F0        CALL    0F018H          ; Einschalten des IRM
                  :
                  :     ; Zugriff auf den IRM ist möglich
                  :     ; z. B. zum Maschinenunterprogramm dort bzw.
                  :     ; zur Realisierung von Bildschirmausschriften
                  :
CD 1B F0        CALL    0F01BH          ; Abschalten des IRM
C9              RET                     ; Rückkehr ins BASIC
```

Beispiel: Das folgende Beispiel ist ein Rahmenprogramm, mit dem es möglich ist, ein Maschinenunterprogramm im IRM aufzurufen. Es wird auf die freien Speicherplätze unterhalb des BASIC-Arbeitsspeichers gelegt. Das Maschinenunterprogramm beginnt auf Adresse 0BE00H im IRM.

Adresse	MC	Anweisung	Bemerkung
0000	CD 6F C9	CALL CPRVL3	; UP zur Parameterübergabe ; von BASIC an das MC- ; Programm
0003	CD 18 F0	CALL 0F018H	; Zuschalten des IRM
0006	CD 00 BE	CALL 0BE00H	; Aufruf des Anwender- ; Maschinenprogramms
0009	CD 1B F0	CALL 0F01BH	; Abschalten des IRM
000C 000D	7A 43	LD A,D LD B,E	; in den Registern A und B ; erfolgt die Parameter- ; Rückgabe
000E	CD B1 D0	CALL FRE3	; UP zur Funktionswert- ; zuweisung an die ; BASIC-Variable
0011	C9	RET	; Rückkehr zu BASIC

Das Maschinenunterprogramm, welches auf der Adresse 0BE00H beginnt, übernimmt den Parameter X der USR(X)-Funktion im DE-Register. Im obigen Beispiel wird der berechnete Funktionswert auch im DE-Register an das Rahmenprogramm zurückgegeben. Hierfür kann jedes Registerpaar, außer dem BC-Register, verwendet werden, da das BC-Register durch das Unterprogramm zum Abschalten des IRM zerstört wird. Das Maschinenunterprogramm ab Adresse 0BE00H realisiert hier die gleiche Funktion wie das bereits besprochen Beispiel:

Es erhöht den eingegeben Parameter um 1 (INC DE) und gibt den Wert als Funktionswert zurück.

Adresse	MC	Anweisung	Bemerkung
BE00	13	INC DE	; vom BASIC übergebener ; Parameter wird um 1 erhöht
BE01	C9	RET	; Rückkehr zum ; Rahmenprogramm

Die Eingabe des Rahmenprogramms sowie des Maschinenunterprogramms kann vom Betriebssystem-Niveau aus mit MODIFY erfolgen. Die Eingabe kann auch per BASIC- Programm geschehen was im folgendem dargestellt wird.

```
 10 REM Maschinencode des Rahmenprogramms
 12 DATA 205,111,201,205,24,240,205,0,190
 14 DATA 205,27,240,122,67,205,177,208,201
 20 REM Maschinencode des Maschinenunterprogramms
 22 DATA 19,201
 30 REM AD = Anfangsadresse
 32 AD = 0
 34 REM MP = Anfangsadresse des Maschinenunterprogramms minus
    8000H
 36 MP=15872
 40 REM Eingabe des Rahmenprogramms mit POKE
 42 FOR I = 0 TO 17
 44 READ B: POKE AD+I, B
 46 NEXT
 50 REM Eingabe des Maschinenunterprogramms mit VPOKE
 52 READ B: VPOKE MP, B: READ B: VPOKE MP+1, B
 60 REM Speichern der Anfangsadresse
 70 DOKE 772,AD
 80 REM Aufruf des Maschinenunterprogramms vom Rahmenprogramm
 90 FOR I = 250 TO 260
100 K = USR(I)
110 PRINT "I"; I; "USR(I)="; K
120 NEXT
```

Statt der einfachen Erhöhung des Parameters I um 1 kann ab Adresse 0BE00H jedes beliebige andere Maschinenprogramm laufen. Dazu müssten die Zeilen 22 und 52 entsprechend geändert werden.

Achtung: Die angegebenen Adressen 304H, 305H, C96FH (UP CPRVL3) und D0B1H (UP FRE3) sind nicht für die Magnetbandversion des BASIC-Interpreters gültig.

Merke:

- **Anweisung: SWITCH**
  Format: SWITCH m, k
  Bemerkung: Die BASIC-Anweisung SWITCH ermöglicht das Schalten von Modulen auch innerhalb eines BASIC-Programms. In folgenden Angaben unterscheidet sich die BASIC-SWITCH-Anweisung von der CAOS-SWITCH-Anweisung (vgl. Seite 71)

	CAOS-Anweisung	BASIC-Anweisung
SWITCH	- (%) Promptzeichen des Systems	- (>) Promptzeichen von BASIC
	- Parameter m und k können entfallen	- Eingabepflicht der Parameter m und k
	- Status lesen und schalten sowie Modulübersicht	- nur schalten
	- Parameter durch Leerzeichen getrennt	- Parameter durch Komma getrennt
	- Parameter hexadezimal	- Parameter dezimal

Beispiel: Einschalten des V.24-Moduls auf Steckplatz C

```
%SWITCH C 1                >SWITCH 12,1
0C EE 00 → 01              OK
%_                         >_
```

Drucken des Namen Emil

```
10 SWITCH 12,1          zuweisen V.24-Modul
20 PRINT #2 "EMIL"      Druckanweisung
30 SWITCH 12,0          abschalten V.24-Modul
```

- **Funktion: FRE**
  Format: FRE (v)
  Bemerkung: Die Funktion FRE gibt die Anzahl der noch freien Speicherplätze im RAM an. Ist v eine Stringvariable, wird die Anzahl der freien Bytes im Stringspeicherbereich angegeben.
  Beispiel: PRINT FRE (A$)

- **Anweisung: CLEAR**
  Format: CLEAR [Ausdruck [, Ausdruck]]
  Bemerkung: CLEAR löscht den Variablenspeicher. Zusätzlich wird mit dem Wert des ersten Ausdrucks festgelegt, wie viel Speicherplätze für Strings reserviert werden sollen. Mit dem zweiten Ausdruck kann die obere Grenze des zugewiesenen RAM-Bereiches neu bestimmt werden. Wird kein Ausdruck angegeben, bleibt die Größe des Stringspeichers unverändert.

  Beispiel: CLEAR 1000

- **Funktion: PEEK**
  Format: PEEK(i)
  Bemerkung: PEEK(i) liest ein Byte von der Speicherstelle i.

  Beispiel: 10 E = PEEK(128)

- **Anweisung: POKE**
  Format: POKE i, j
  Bemerkung: Die Anweisung speichert das Byte j in der Speicherzelle i. Ist i negativ, wird es als 65.536 + i interpretiert.

  Beispiel: POKE 2,15

- **Funktion: DEEK**
  Format: DEEK (i)
  Bemerkung: DEEK (i) liest ein Byte von den Speicherzellen i und i+1.

  Beispiel: 10 PRINT DEEK(218)

- **Anweisung: DOKE**
  Format: DOKE i, j
  Bemerkung: Die Anweisung speichert den Wert j in den Speicherzellen i und i+1. Sind i und j negativ, so werden sie als 65.536 +i bzw. 65.636 + i + 1 interpretiert.

  Beispiel: DOKE 0, 302

- **Anweisung: CALL**
  Format: CALL Dezimaladresse
  oder
  CALL* Hexadezimaladresse
  Bemerkung: CALL i ruft ein Maschinenprogramm mit der Startadresse i auf. Das Maschinenprogramm muss mit RETURN (0C9H) abgeschlossen sein und darf keine Register verändern. CALL* wirkt wie CALL. Die Startadresse ist dabei jedoch hexadezimal anzugeben. Bei Verwendung von CALL muss gegebenenfalls beachtet werden, dass der Bildwiederholspeicher beim Erreichen des Maschinenprogramms abgeschaltet ist.

  Beispiel: CALL * 30A0

- **Funktion:** **VPEEK**
  Format: VPEEK (Adresse)
  Bemerkung: VPEEK gewährleistet das Lesen des Inhaltes einer Speicherzelle im Bildwiederholspeicher

- **Anweisung:** **VPOKE**
  Format: VPOKE Adresse, Wert
  Bemerkung: Die Anweisung realisiert das Beschreiben einer Speicherzelle im Bildwiederholspeicher.
  VPEEK und VPOKE sind erforderlich, um Speicherzellen im Bildwiederholspeicher zu lesen bzw. zu beschreiben, da dieser bei der Arbeit des BASIC-Interpreters abgeschaltet wird. Der Anfang des Bildwiederholspeichers liegt dabei auf Adresse 0 (vgl. PEEK und POKE).

  Beispiel:
  ```
  10 W = VPEEK(12)
  20 PRINT W
  30 VPOKE 25, W
  ```

- **Funktion:** **USR**
  Format: USR (X)
  Bemerkung: Die Funktion ruft ein Maschinenprogramm mit dem Argument X auf. Die Anfangsadresse des Maschinenprogramms ist auf die Speicherplätze 304H und 305H (772 und 773 dezimal), z. B. mit der Anweisung DOKE zu speichern. Die Parameterübergabe vom BASIC-Programm zum Maschinenprogramm wird mit dem Unterprogramm CPRVL3 (Adresse C96FH) und die Parameterrückgabe vom Maschinenprogramm mit dem Unterprogramm FRE3 (Adresse D0B1H) des BASIC-Interpreters realisiert.
  Der Parameter vom BASIC wird im Register DE an das Maschinenprogramm übergeben. Der Rückgabewert ist vom Maschinenprogramm in den Registern A (H-Teil) und B (L-Teil) zur Übergabe an BASIC bereitzustellen. Bei der Verwendung von USR muss gegebenenfalls beachtet werden, dass der Bildwiederholspeicher beim Erreichen des Maschinenprogramms abgeschaltet ist.

  Beispiel:
  ```
  POKE 773,38
  K = USR(X)
  ```

4.1.23. Arbeit mit der Peripherie

Zusatzgeräte und BASIC

Mit BASIC-Programmen können wir mit unserem KC 85 auch Zusatzgeräte, die Peripherie, ansteuern.

Eine Peripherie erhöht die Anwendungsbreite und den Wert des Computersystems enorm. So besitzen Sie z. B. mit einem anschließbaren Drucker und dem KC 85 eine elektronische Schreibmaschine. Zu den vom Hersteller angebotenen Zusatzgeräten wird stets eine ausführliche Beschreibung und Bedienungsanleitung mitgeliefert.

Im Gegensatz zu den im Kapitel 4.1.9 beschriebenen Anweisungen CSAVE und CLOAD, die speziell die periphere Einheit Speichergerät ansprechen und eine rechnerinterne (übersetzte) Zeichendarstellung übermitteln, werden in diesem Kapitel allgemeingültige Anweisungen zum Betreiben von Zusatzgeräten beschrieben.

Mithilfe der Anweisung

```
LIST#n "Name"
```

lassen sich Programmlisten Zeichen für Zeichen auf ein durch die Zahl n bezeichnetes, frei wählbares Peripheriegerät (z. B. Drucker für n=2) ausgeben.

Liegt nun ein auf diese Weise gespeichertes Programm z. B. auf Magnetband vor (LIST#1 "Name"), so können wir dieses mit der Anweisung

```
LOAD#1 "Name"
```

wieder Zeichen für Zeichen in den Computer lesen. Dabei ergibt sich im Gegensatz zur Anweisung CLOAD der Vorteil, dass beim Nachladen von Programmen das zu ladende Programm entsprechend den Zeilennummern richtig bezüglich des bereits gespeicherten Programms einsortiert wird.

Auch Daten können wir auf ein Peripheriegerät ausgeben und von dort wieder laden. Hierzu benötigen wir die Anweisungen OPEN, CLOSE, PRINT# und INPUT#. Durch die Anweisung

```
OPENr#n "Name"
```

wird der Kanal zu dem durch n bezeichneten Peripheriegerät zur Datenausgabe oder zur Dateneingabe geöffnet. Das r ist im konkreten Fall durch ein O zur Datenausgabe bzw. durch ein I zur Dateieingabe zu ersetzen.

Ist der Kanal geöffnet, kann die Datenausgabe durch

```
PRINT#n Daten
```

oder die Dateneingabe durch

```
INPUT#n Daten
```

erfolgen. Ist die Datenübertragung beendet, wird der Kanal mit

CLOSEr#n

wieder geschlossen. Betrachten Sie bitte noch das Beispiel im Abschnitt “Merke“ dieses Kapitels.

Das in den Anweisungen enthaltene, zur Gerätezuweisung bestimmte n ist wie folgt festgelegt:

0	Bildschirm und Tastatur
1	Kassettenrecorder bzw. Speichergerät (DISK, USB)
2	frei für weitere Anwendungen (Drucker, Plotter, Floppy)
3	Lochstreifeneinheit usw.

Zur Synchronisation einer langsamer arbeitenden peripheren Einheit kann vor der Datenausgabe eine NULL-Anweisung z. B.

NULL 60

eingegeben werden. Dadurch werden an jeder Zeile 60 bzw. die angegebene Anzahl "NULL-" oder auch "Dummyzeichen" angehängt. Für die Grundeinstellung sind 10 Dummyzeichen festgelegt. Diese bedeutungslosen Zeichen ermöglichen die synchrone Zusammenarbeit zwischen dem KC 85 und der peripheren Einheit. Die Anzahl der zu vereinbarenden Dummyzeichen ist abhängig vom Peripheriegerät.

Darüber hinaus stehen uns in BASIC zur Ansteuerung und Überwachung peripherer Einheiten die Anweisungen INP, OUT, WAIT, JOYST sowie die Funktion POS zur Verfügung. Die Handhabung dieser Anweisungen entnehmen Sie bitte dem folgenden Abschnitt “Merke“.

Merke:

- **Anweisung: OPEN**
  Format: OPENr#n "Name"
  Bemerkung: OPEN öffnet eine Kanaloperation mit der Namensübergabe auf den Kanal n für dateifähige Geräte (z. B. Floppy) oder Geräte mit Namensverwaltung. Der Parameter r legt fest, ob es sich um eine Aus- oder Eingabe handelt:
  r = I Eingabe } ohne Trenn- oder Leerzeichen
  r = O Ausgabe } an OPEN (oder CLOSE) anfügen!
  Beispiel: siehe Anweisung PRINT#, INPUT#

- **Anweisung: CLOSE**
  Format: CLOSEr#n
  Bemerkung: CLOSE schließt den Kanal n (nach Dateneingabe r=I oder nach Datenausgabe r=O). CLOSE führt der BASIC-Interpreter automatisch bei Programmende oder -abbruch (Taste oder ERROR) aus.

  Beispiel: siehe Anweisungen PRINT#, INPUT#

- **Anweisung: PRINT#, INPUT#**
  Format: PRINT#n Daten
  INPUT#n Daten
  Bemerkung: PRINT# ermöglicht die Ausgabe von Daten auf einen durch n definierten Kanal.
  INPUT# ermöglicht die Eingabe der mit PRINT ausgegebenen Daten von einem durch n wählbaren Peripheriegerät. Beide Anweisungen sind nur in Verbindung mit OPEN und CLOSE (Übergabe des Dateinamens) zu verwenden. Ein Lesen der mit PRINT# ausgegebenen Daten ist nur durch die Anweisungen OPEN und CLOSE möglich. Sollen Daten jedoch nur ausgegeben werden (z. B. auf Drucker), können sie auch mit der Anweisung PRINT# ohne OPEN ausgegeben werden.

  Beispiel 1:
  ```
  10 CLS : DIM A$(4)
  20 FOR I = 0 TO 4 :INPUT A$(I):NEXT
  30 PRINT" RECORDER EINSCHALTEN !"
  40 OPENO#1 "BEISPIEL"
  50 FOR I = 0 TO 4 :PRINT#1 A$(I):NEXT
  60 CLOSEO#1
  ```

  Beispiel 2:
  ```
  10 CLS : DIM A$(4)
  20 PRINT" RECORDER EINSCHALTEN !"
  30 OPENI#! "BEISPIEL"
  40 FOR I = 0 TO 4 :INPUT#1 A$(I):NEXT
  50 CLOSEI#1
  ```

- **Anweisung: LIST#, LOAD#**
  Format: LIST#n "Programmname"
  LOAD#n "Programmname"
  Bemerkung: LIST# ermöglicht die Ausgabe eines Programmlistings (ASCII-Zeichen für ASCII-Zeichen) auf ein durch n wählbares Peripheriegerät.
  Mit LOAD# können die mit LIST# ausgegebenen Programme wieder eingegeben werden. Durch n wird wiederum das Peripheriegerät festgelegt.

  Beispiel:
  LIST#2 "KALKU" (Ausgabe des Programmlistings KALKU auf Drucker)
  LOAD#1 "TEST" (Einlesen des mit LIST#1 "TEST“ auf Magnetband ausgegebenen Programmlistings)

- **Peripheriecodierung n** für die Anweisungen OPEN, CLOSE, PRINT#, INPUT#, LIST# und LOAD#

n	Peripherie
0	Bildschirm und Tastatur
1	Speichergerät (Kassette, Floppy, USB)
2	Frei für Anwender (vorzugsweise Drucker)
3	Frei für Anwender (Vorzugsweise V.24-Duplexschnittstelle)

- **Anweisung: NULL**
  Format: NULL Zahl
  Bemerkung: NULL legt die Anzahl der am Ende einer Zeile auszugebenden ASCII-Nullcodes (so genannte Dummyzeichen, die keinen Einfluss auf das Programm haben) fest. Damit wird eine Synchronisation von Rechner und langsamer Peripherie erreicht.

  Beispiel: NULL40

- **Funktion: INP**
  Format: INP(i)
  Bemerkung: INP(i) liefert das aus dem Port i gelesene Byte
  $(0 \leq i \leq 255)$

  Beispiel: A = INPT(255)

- **Anweisung: OUT**
  Format: OUT i, j
  Bemerkung: Die Anweisung gibt das Byte j aus dem Port i aus.
  $(0 \leq i \leq 255)$

  Beispiel: OUT 32,100

- **Anweisung: WAIT**
  Format: WAIT i, j, (k)
  Bemerkung: WAIT hält den Programmablauf in einer Warteschleife bis am Port i das erwartete Bitmuster erscheint. Der eingelesene Wert wird exklusiv-oder-verknüpft mit k und anschließend und-verknüpft mit j. Ist das Resultat 0, bleibt das Programm in der Warteschleife, ansonsten wird es fortgesetzt. Wird k nicht angegeben, so ist k gleich 0.

  Beispiel: WAIT 32,2

- **Funktion: POS**
  Format: POS (i)
  Bemerkung: Die Funktion gibt die aktuelle Schreibposition in der Zeile an. Die Position links außen ist gleich 0. Die Zahl i ist dabei ein Dummyargument (d. h. der Wert des Arguments hat keinen Einfluss auf die Funktion).

  Beispiel: 1180 IF POS(2) = 30 GOTO 50

- **Anweisung: BORDER**
  Format: BORDER n
  Bemerkung: Mit der Anweisung BORDER wird beim KC87 die Farbe des Bildschirmrandes festgelegt. Die Anweisung ist auch im KC 85-BASIC enthalten, jedoch lässt sich damit der Bildschirmrand nicht beeinflussen. Je nach CAOS-Version hat BORDER unterschiedliche Wirkungen

CAOS 3.1	schaltet Bit 5+6 von PIO-Port B (unbenutzt)
CAOS 3.4	keine Wirkung
CAOS 4.1 – 4.4	schaltet Bit 5+6 von PIO-Port B (RAM8)
CAOS 4.5 – 4.8	keine Wirkung

  Beispiel: BORDER-Befehle sollten in Programmen, welche vom KC87 übernommen wurden, beim KC 85/3-5 möglichst entfernt werden.

Joystick-Abfragen

Der BASIC-Interpreter des KC 85/3 und KC 85/4 war für die Abfrage eines Joysticks bereits vorbereitet, sie führte zur Adresse 02FDH. Dort musste ein nachzuladender Joysticktreiber seine Funktion eintragen. Der seit CAOS 4.5 enthaltene Joysticktreiber unterstützt den Joystick direkt. Die in früheren CAOS-Versionen vorgesehene Adresse 02FDH wird dazu nicht mehr genutzt.

Die BASIC-Funktion JOYST liefert Informationen über die Betätigung des Spielhebels (Joystick) und ist kompatibel zum KC87 (beim Betrieb mit einem Joystick).

Im BASIC-Programm kann die momentane Stellung des Joysticks abgefragt werden. In Abhängigkeit vom Ergebnis dieser Abfrage können Sie dann den weiteren

Ablauf Ihres BASIC-Programms steuern. Damit die Spielhebelbetätigung richtig ausgewertet werden kann, muss der Spielhebel so gehalten werden, dass die flache schmale Taste, die Aktionstaste, vom Körper weg zeigt.

- **Funktion:** **JOYST**
  Format: JOYST(1)
  Bemerkung: Der angegebene Parameter (1) ist dabei ein ganzzahliger Wert für die Nummer des verwendeten Spielhebels. Beim KC 85/5 gibt es nur den Spielhebel 1, beim KC87 können auch zwei Spielhebel angeschlossen werden, welche dann mit 1 bzw. 2 abgefragt werden.
  Als Funktionswert erhält man einen numerischen Wert, der durch die Stellung des Spielhebels bestimmt wird. Bei gleichzeitig gedrückter Aktionstaste wird der Wert 16 zu den Stellungsinformationen addiert. Das lässt sich ganz leicht mit dem folgenden Beispielprogramm 1 kontrollieren.

  Beispiel 1:
```
10 LOCATE 10,10 : PRINT JOYST(1) : GOTO 10
```

  Beispiel 2:
```
10 REM Joystickabfrage unter BASIC
20 IF JOYST(1)=1 THEN PRINT "links"
30 IF JOYST(1)=2 THEN PRINT "rechts"
40 IF JOYST(1)=4 THEN PRINT "runter"
50 IF JOYST(1)=8 THEN PRINT "hoch"
60 GOTO 20
```

Spielhebeldruckpunkt	Funktionswert von JOYST	äquivalente Tastaturbetätigung
Ruhestellung	0	keine Taste gedrückt
	1	[←]
	2	[→]
	4	[↓]
	8	[↑]
	5	[←][↓]
	6	[→][↓]
	9	[←][↑]
	10	[→][↑]
Aktionstaste	16	[ESC]

Bild 36: Joystick-Abfrage unter BASIC

4.1.24. Lösungen zu den BASIC-Übungen

In diesem Kapitel wird jeweils eine Lösung der in den Übungen zur Aufgabe gestellten BASIC-Programmen vorgestellt. Sollten Sie ein anderes Programm als Lösung erarbeitet haben, welches die Aufgabe ebenfalls erfüllt, so ist dieses genauso richtig wie die hier aufgeführten Programme. Denn es gibt fast immer eine Vielzahl von Lösungsmöglichkeiten.

Kapitel 4.1.6 Übung 1

```
10 INPUT"BENZINVERBRAUCH IM L/KM=";A
20 PRINT"BEWERTUNG:";
30 IF A>=3 AND A<8 THEN PRINT "GUT"
40 IF A>=8 AND A<=13 THEN PRINT "SCHLECHT"
50 IF A<3 OR A>13 THEN PRINT "UNMOEGLICH!"
```

Kapitel 4.1.6 Übung 4

```
10 CLS:PRINT"NAEHERUNGSVERFAHREN ZUR BERECHNUNG"
20 PRINT"DER QUADRATWURZEL NACH NEWTON":PRINT
30 INPUT" ZAHL:";A
40 INPUT" GESCHAETZTE NAEHERUNG:" ;X
50 PRINT" NAEHERUNGEN:"
60 N=X-(X*X-A)/(2*X)
70 IF ABS(N-X)<1E-3 THEN 90
80 X=N: PRINT N: GOTO 60
90 PRINT:PRINT" DIE QUADRATWURZEL DER ZAHL"A
100 PRINT" IST"N"."
```

Kapitel 4.1.7 Übung 2

```
10 ! DREIECKSBERECHNUNG
20 INPUT" A,B:";A,B
30 A=A*A:B=B*B
40 C=SQR(A+B)
50 PRINT" C=";C
60 END
```

Kapitel 4.1.7 Übung 3

```
 10 ! QUADRATISCHE GLEICHUNG
 20 INPUT" P,Q:";P,Q
 30 S=-P/2
 40 W=S*S-Q
 50 IF W<0 THEN GOTO 100
 60 W=SQR(W)
 70 X1=S+W: X2=S-W
 80 PRINT" X1=";X1
 90 PRINT" X2=";X2:GOTO 110
100 PRINT"GLEICHUNG IM REELLEN ZAHLENBEREICH NICHT LOES-
    BAR"
110 END
```

Kapitel 4.1.11 Übung 1

```
 10 PAPER1:CLS
 12 LINE0,128,319,128,7
 14 LINE0,0,0,255
 20 FOR X=0 TO 319
 30 Y=128+50*SIN(X/159.5*PI)
 40 PSETX,Y,7
 50 NEXT
```

Kapitel 4.1.14 Übung 1

```
 10 INPUTA$
 20 INPUTB$
 30 IF A$<= B$GOTO 70
 40 W$ = A$
 50 A$ = B$
 60 B$ = W$
 70 PRINT A$
 80 PRINT B$
```

4.1.25. CAOS-Kommando BSAVE

Im CAOS-Menü (USER-ROM) gibt es seit CAOS 4.7 das Menüwort BSAVE. Dieses dient zum Abspeichern eines BASIC-Programmes als selbststartendes Maschinencode-Programm. Voraussetzung dafür ist ein vorher mit BASIC erstelltes oder geladenes BASIC-Programm im Arbeitsspeicher des Computers. Aufruf:

%BSAVE

Zunächst wird geprüft, ob BASIC initialisiert ist und sich ein Programm im Arbeitsspeicher befindet. Ist dies der Fall, dann werden die Adressen der zu erzeugenden Maschinencode-Datei angezeigt und der Dateiname abgefragt. Dabei wird automatisch der Dateityp .KCB vergeben, um die Dateien von reinen Maschinencode-Programmen unterscheiden zu können. Abgespeichert wird die Datei dann mit einer Selbststart-Routine, welche mit allen CAOS-Versionen funktioniert.

Hinweis

Falls ein selbststartendes BASIC-Programm *.KCB zunächst ganz normal startet, aber beim Aufruf von BLOAD oder SAVE abstürzt, kann die Ursache an der enthaltenen Autostartroutine der KCB-Datei liegen. Da dieser Fehler viele *.KCB-Dateien betreffen dürfte, hier ein Hinweis, wie dieses Problem gelöst werden kann:

! Es existieren verschiedene Varianten selbststartender BASIC-Programme. Wenn der BASIC-Selbststart einer *.KCB-Datei nicht mithilfe der F-Tasten-Funktion vom CAOS, sondern mittels Maschinencode erfolgt, kann es passieren, dass BASIC beim Ausführen einer Datei-Anweisung abstürzt. Das passiert zum Beispiel, wenn der BASIC-ROM durch direkte Ein-/Ausgabebefehle aktiviert wird, ohne dabei SWITCH zu benutzen. Dadurch wird der aktuelle Zustand des USER-ROM im Modulsteuerbytespeicher nicht initialisiert. Das führt zu Fehlern bei der Nutzung der DEVICE-Schnittstelle, weil zum Wechsel zwischen BASIC und dem DEVICE-Modul umgeschaltet werden muss. Im Modulsteuerbytespeicher fehlt für den USER-ROM dann aber der Eintrag C1H und BASIC wird nach Abarbeitung der DEVICE-Routinen nicht wieder aktiviert.

Das Problem lässt sich umgehen, indem man vor dem Laden des Programms den BASIC-ROM mit %SWITCH 2 C1 manuell einschaltet, oder indem man mit %LOAD 0 0 das Programm lädt und dann mit %BSAVE vom CAOS 4.8 neu speichert. Dabei wird dann eine funktionierende Autostart-Routine eingetragen.

4.1.26. BASIC-Token

Wenn im BASIC-Interpreter eine Programmzeile eingegeben wird, so existiert sie zunächst im BASIC-Puffer als normaler ASCII-Text. Eine spezielle Routine im BASIC-Interpreter wandelt den Text für die weitere Verarbeitung in die „interne

Darstellung“ um. Jedes im Text gefundene reservierte Wort wird dabei in ein spezielles Byte größer 7FH umgewandelt, das sogenannte Token. Die Bedeutung der Token unterscheidet sich von Rechner zu Rechner. Siehe dazu auch [58].

Für den KC 85/5 gilt die folgende Übersicht.

Tabelle 50: Übersicht der BASIC-Token

	8_	9_	A_	B_	C_	D_	E_	F_
_0	END	OUT	LIST	^	EXP	LOAD	PAPER	RANDOMIZE
_1	FOR	ON	CLEAR	AND	COS	TRON	AT	VGET$
_2	NEXT	NULL	CLOAD	OR	SIN	TROFF	COLOR	LINE
_3	DATA	WAIT	CSAVE	>	TAN	EDIT	SOUND	CIRCLE
_4	INPUT	DEF	NEW	=	ATN	ELSE	PSET	CSRLIN
_5	DIM	POKE	TAB(	<	PEEK	INKEY$	PRESET	DEVICE
_6	READ	DOKE	TO	SGN	DEEK	JOYST	BLOAD	FILES
_7	LET	AUTO	FN	INT	PI	STRING$	VPEEK	CHDIR
_8	GOTO	LINES	SPC(	ABS	LEN	INSTR	VPOKE	ZERO
_9	RUN	CLS	THEN	USR	STR$	RENUMBER	LOCATE	HOME
_A	IF	WIDTH	NOT	FRE	VAL	DELETE	KEYLIST	SCREEN
_B	RESTORE	BYE	STEP	INP	ASC	PAUSE	KEY	SIZE
_C	GOSUB	!	+	POS	CHR$	BEEP	SWITCH	LABEL
_D	RETURN	CALL	-	SQR	LEFT$	WINDOW	PTEST	SCALE
_E	REM	PRINT	*	RND	RIGHT$	BORDER	CLOSE	GCLS
_F	STOP	CONT	/	LN	MID$	INK	OPEN	

Hinweise:

- Die Token von **80H bis D4H** kommen vom BASIC-Kern und sind bei allen KC-Typen identisch, bis E1H auch beim KC87. Die Token von **D5H bis F4H** kommen von der BASIC-Erweiterung und gelten von CAOS 3.1 bis CAOS 4.4
- Die Token **F5H bis F7H** gelten ab CAOS 4.7 (F5H heißt bei CAOS 4.5 DRIVE)
- Die Token **F8H bis FEH** sind nur in der Erweiterung Plotter-BASIC 2.0 vorhanden, siehe das zugehörige Handbuch [69]
- BORDER hat beim KC 85/3-5 keine Funktion (bis CAOS 4.4 Absturz möglich)
- Nur bei TAB(und SPC(gehört die Klammer zum Token, deshalb sind TAB und SPC (ohne Klammer) keine reservierten Wörter
- Die aus zwei Zeichen bestehenden Token: AT, FN, IF, LN, ON, OR, PI und TO sind nicht als Variablen verwendbar

4.1.27. BASIC-Arbeitszellen

In der folgenden Übersicht sind die BASIC-Arbeitszellen (Workspace) und deren Bedeutung beschrieben. Diese Arbeitszellen sind bei allen CAOS-Versionen identisch, da der Inhalt des BASIC-ROM immer unverändert geblieben ist. Die folgende Übersicht ist dem Quelltext des BASIC-Interpreters entnommen und ähnlich auch in [57] enthalten.

Tabelle 51: BASIC-Arbeitszellen (WorkSpace)

Adr.	Länge	Inhalt	Beschreibung
0300H	3	C3 8C C0	Sprungbefehl BASIC-Warmstart (REBASIC)
0303H	3	C3 67 C9	Sprungbefehl in die USR(X)-Funktion, die Startadresse der MC-Routine muss auf 0304H eingetragen werden (DOKE 772,AD)
0306H	1	00	Portadresse Variable
0307H	1	00	E/A-Flag
0308H	1	00	OUT-Index
0309H	1	00	IN-Index
030AH	1	00	TRACE-Flag
030BH	14		Schnelle SBC-Routine für Gleitkommadivision, Variablen auf Adresse 30CH, 310H, 314H, 317H
0319H	3	00 00 00	RND-Variablen
031CH	32		Daten für die RND-Funktion (Pseude-Zufallszahlen) Array mit 8 Konstanten
033CH	4	52 C7 4F 80	Pseudo-Zufallszahl für RND(0) = .811635
0340H	1	0B	Anzahl der Dummy-Zeichen: » NULL X « (0-255)
0341H	1	FF	Ausgabezeilenlänge (255)
0342H	1	1B	Anzahl der darstellbaren Formate pro Zeile -1 Formatlänge (13) + 1 = 27
0343H	1	00	Ausgabeunterdrückung, wenn Zelle <> 0
0344H	2	0A 00	Zeilenanzahl für LIST. Normal = 10 (LINES X)
0346H	2	0A 00	Kopie von 0344
0348H	2	00 00	Prüfsumme für CLOAD* und CSAVE*
034AH	3	C3 AE C5	Eingaberoutine für Daten
034DH	1	00	AUTO-Flag
034EH	2	00 00	Aktuelle Zeilennummer
0350H	2	00 00	Zeilennummer-Abstand
0352H	2	00 00	(NANF)

Adr.	Länge	Inhalt	Beschreibung
0354H	2	00 00	(DISTAN)
0356H	2	65 04	RAM-Mindestgröße (PRAM+100)
0358H	2	FE FF	Aktuelle Zeilennummer (65534)
035AH	3	00 00 C9	WINJP
035DH	1	00	DATBYT (Kopierschutz), wenn Zelle <> 0 DOKE 861,0 hebt Kopierschutz auf
035EH	1	00	DATFLG (List- und Editierschutz) wenn Zelle <> 0
035FH	2	01 04	Programmstart-Adresse (0401H)
0361H	74	00 00 00 00	Eingabepufferanfang; initialisiert wird 0361-0363H
- - - - - - - - - - - - - - - - - - - keine Defaultbelegung - - - - - - - - - - - - - - - - - - -			
03ACH	1		Cursorposition
03ADH	1		Suche Variable (0) oder füge sie ein
03AEH	1		Datentyp
03AFH	1		Data-Statement-Flag
03B0H	2		Memory Size (Speicherende, Anfang Stringraum)
03B2H	2		Adresse der nächsten freien Stelle in der String-Adresstabelle
03B4H	11		String-Adresstabelle
03BFH	1		Ende der Tabelle
03C0H	4		Länge und Adresse des aktuellen Strings
03C4H	2		Zeiger für nächsten String
03C6H	2		Index des zuletzt abgearbeiteten Bytes
03C8H	2		Adresse des Programmzeigers während der Suche nach einer vorherigen FOR-Anweisung
03CAH	2		Zeilennummer des zuletzt gelesenen DATA-Statements
03CCH	1		FOR-Flag
03CDH	1		INPUT-Flag
03CEH	1		Read/Write-Flag
03CFH	2		Zeilennummer, Adresse des aktuellen Statements
03D1H	2		Adresse des nächsten Elements
03D3H	2		Letzte Zeilennummer bei END/STOP
03D5H	2		Adresse des letzten Bytes bei ERROR
03D7H	2		Adresse der Liste der numerischen und Stringvariablen (Programmende + 1)
03D9H	2		Adresse der Liste der Feldvariablen
03DBH	2		Erste freie Adresse nach den Variablenlisten
03DDH	2		Aktuelle Adresse im DATA-Statement während READ

Adr.	Länge	Inhalt	Beschreibung
03DFH	2	Adresse des FN-Statements	
03E1H	4	Arithmetik-Register 0	
03E5H	4	Arithmetik-Register 1	
03E9H	1	Vorzeichen der Operation (SGN)	
03EAH	13	Druckpuffer	
03F7H	4	Arithmetik-Register 2	
03FBH	1	Listen-Flag (Status) 0 = Grundliste; 1 = Erweiterung	
03FCH	1	Erweiterungsflag 0 = Erweiterung nicht vorhanden	
03FDH	1	PRINT-Erweiterungsflag 0 = ohne PRINT-Erweiterung	
03FEH	1	Farbbyte	

4.1.28. BASIC-Übersichten

Die BASIC-Übersichten enthalten eine Kurzbeschreibung des HC-BASIC des KC 85/4. Sie dienen als Nachschlagewerk im täglichen Umgang mit dem Computer und genügen dem Fachmann auch als Einstieg in die Spezifik dieses BASIC.

Konstanten

Der BASIC-Interpreter verarbeitet Integerzahlen (vorzeichenbehaftete ganze Zahlen), reelle Zahlen (Gleitkomma-, Festkommazahlen) und Strings als Konstanten.
Die Konstante PI ist mit PI = 3.14159 gespeichert.
Das Ausgabeformat für eine Zahl ergibt sich wie folgt:

1. Integerzahlen zwischen -999999 und +999999 werden als Integerwert ausgegeben.
2. Ist der Betrag einer Zahl größer oder gleich 0.01 und kleiner oder gleich 999999, wird die Zahl mit Festkomma und ohne Exponenten ausgegeben.
3. Fällt eine Zahl nicht in die Kategorie I oder 2, wird sie als Gleitkommazahl mit Exponent ausgegeben.

Der Interpreter verarbeitet Zahlen mit einem Betrag zwischen 9.40396E-39 und 1.70141E+38, einschließlich Null. Strings sind Zeichenketten von alphanumerischen Zeichen, die 0 bis 255 Zeichen lang sein können. String-Konstanten werden in Hochkomma eingeschlossen.

Variablen

Variablennamen müssen immer mit einem Buchstaben beginnen und können beliebig lang sein. Es werden aber immer nur die beiden ersten Zeichen des Namens verarbeitet. Der Variablenname darf kein reserviertes Wort enthalten. Das sind alle BASIC-Anweisungen. Ein $-Zeichen am Ende des Variablennamens weist sie als Stringvariable aus. Die Feldvariablen können so viele Indizes haben, wie in eine Eingabezeile passen.

Operatoren

Der BASIC-Interpreter des KC 85/4 verfügt über alle üblichen mathematischen und logischen Operatoren.
Die Reihenfolge der Abarbeitung ist wie folgt hierarchisch geordnet:

1. Klammern
2. Exponenten ^
3. Negative Vorzeichen
4. Multiplikation, Division: *, /
5. Addition, Subtraktion: +, -
6. Vergleichsoperatoren: =, <, >, <=, >=, <>
7. NOT
8. AND
9. OR

Die logischen Operatoren wirken bitweise auf 16-Bit-Integerzahlen im Bereich von -32768 bis +32767.

Tabelle 52: Wahrheitswerte der logischen Operatoren

A	B	NOT A	A OR B	A AND B
0	0	1	0	0
0	1	1	1	0
1	0	0	1	0
1	1	0	1	1

Die Vergleichsoperatoren und die Addition als Verknüpfung sind auch auf Strings anwendbar.

Anweisungen

Tabelle 53: Übersicht der BASIC-Anweisungen

HC-BASIC	Bemerkung	Syntax
AUTO	Selbständige Zeilennummerierung	AUTO[Zeilennummer[,Schrittweite]]
BEEP	Erzeugung eines Signaltones	BEEP[Anzahl]
BLOAD	Maschinenprogramm einlesen	BLOAD“name“
BYE	Rückkehr zum Betriebssystem	BYE
CALL*	Aufruf eines Maschinenprogramms	CALL*Startadresse (hex.)
CALL	Aufruf eines Maschinenprogramms	CALL Startadresse (dez.)
CHDIR	Verzeichnis wechseln	CHDIR "name"
CIRCLE	Zeichnen eines Kreises	CIRCLExm,ym,r[,g]
COLOR	Einstellen der Vorder- und Hintergrundfarbe	COLORv,h
CLEAR	Löschen des Variablenspeichers; Begrenzung des String- und BASIC-Speichers	CLEAR[Ausdruck[,Ausdruck]]
CLOAD	Einlesen eines BASIC-Programms	CLOAD"name"
CLOAD*	Einlesen eines Variablenfeldes	CLOAD*"name";feldname
CLOSE	Schließen einer Kanaloperation	CLOSEr#n r=I, 0
CLS	Bildschirm löschen	CLS
CONT	Fortsetzen eines mit BRK unterbrochenen Programms	CONT
CSAVE	BASIC-Programm abspeichern	CSAVE"name"
CSAVE*	Variablenfeld abspeichern	CSAVE*"name";feldname

4.1. BASIC

HC-BASIC	Bemerkung	Syntax
DATA	Datenliste; mit READ zu lesen	DATA Konstante[,Konstante,…]
DEEK	Lesen der Adressen i und i+1	I=DEEK(i) -32768 <= I <= 32767
DEF FN	Definition einer Funktion	DEF FNname(Parameter)=Ausdruck
DELETE	Programmzeilen a bis e löschen	DELETE a,e
DEVICE	Speichergerät einstellen	DEVICE [nr]
DIM	Variablenfeld dimensionieren	DIM feldname(Index[, Index…])
DOKE	Schreibt Wert j auf Adressen i und i+1	DOKEi,j -32763 <= j <= 32767
EDIT	Programmkorrektur	EDIT Zeilennummer
ELSE	Alternative zu IF-THEN	IF…THEN…:ELSE…
END	Programmabschluss	END
FILES	Verzeichnisinhalt von Speichergerät anzeigen	FILES ["maske"]
FOR TO STEP	Festlegen einer Programmschleife (Schleife geht bis NEXT)	FOR Variable = Anfangswert TO Endwert [STEP Schrittweite]
GOSUB	Unterprogrammaufruf	GOSUB Zeilennummer
GOTO	Unbedingte Sprunganweisung	GOTO Zeilennummer
IF	Bedingte Sprunganweisung	IF Ausdruck GOTO Zeilennummer IF Ausdruck THEN Zeilennummer IF Ausdruck THEN Anweisung [:Anweisung…] IF Ausdruck THEN Anweisung: ELSE Anweisung
INK	Einstellen der Vordergrundfarbe	INK v
INP	Liefert das aus dem Port i gelesene Byte	INP i
INPUT	Eingabeanforderung	INPUT["Strings";] Variable
INPUT#	Eingabeanforderung vom wählbaren Peripheriegerät	INPUT#n Variable
JOYST	Spielhebelabfrage	I=JOYST(1)
KEY	Belegen der Funktionstasten	KEY Funktionstastennummer
KEYLIST	Auflisten der Funktionstastenbelegung	KEYLIST
LET	Wertzuweisung LET kann entfallen.	LET Variable = Ausdruck
LINE	Zeichnen einer Linie	LINE xa,ya,xe,ye(,g)
LINES	Festlegen der Anzahl der aufzulistenden Zeilen	LINES [Anzahl]
LIST	Auflisten eines Programms	LIST[Zeilennummer]
LIST#	Ausgeben eines Programms auf wählbares Peripheriegerät	LIST#n "Programmname"

4.1. BASIC

HC-BASIC	Bemerkung	Syntax
LOAD#	Einlesen eines mit LIST# ausgegebenen Programms	LOAD#n "Programmname"
LOCATE	Platzieren des Cursors im aktuellen Fenster	LOCATE z,s
NEW	Löschen des Programm- und Variablenspeichers	NEW
NEXT	Abschluss einer FOR-Programmschleife	NEXT [Variable,Variable,…]
NULL	Legt die Anzahl der auszugebenden Dummyzeichen am Ende einer Zeile fest	NULL Zahl
ON GOTO	Bedingte Mehrfachverzweigung zu verschiedenen Programmteilen	ON Ausdruck GOTO Liste von Zeilennummern
ON GOSUB	Bedingte Mehrfachverzweigung zu verschiedenen Unterprogrammen	ON Ausdruck GOSUB Liste von Zeilennummern
OPEN	Eröffnen einer Kanaloperation	OPENr#n r = I, 0
OUT	Gibt Byte j aus Port i aus	OUT i,j 0 <= I <= 255
PAPER	Einstellen der Hintergrundfarbe	PAPER h
PAUSE	Pause in der Programmabarbeitung	PAUSE[t] Zeitfaktor = t * 0,1 s
PEEK	Lesen der Adresse i	I=PEEK(i) 0 <= I <= 255
POKE	Schreibt Byte j auf Adresse i	POKE i,j 0 <= j <= 255
PRESET	Löschen eines Bildpunktes	PRESET x,y[,g]
PRINT	Ausgabe auf Bildschirm In diese Anweisung können COLOR, INK, PAPER, AT, SPC oder TAB eingebunden werden. PRINT kann durch "?" ersetzt werden.	PRINT [Print-Liste] PRINT Farbanweisung; Print-Liste PRINT Formatierungsfunktion; Print-Liste PRINT AT(z,s); Farbanweisung; Print-Liste
PRINT#	Ausgabe auf wählbares Peripheriegerät (siehe PRINT)	PRINT#n [Print-Liste]
PSET	Setzen eines Bildpunktes	PSET x,y[,g]
PTEST	Testen eines Bildpunktes	PTEST (x[,y])
RANDOMIZE	Zufallsgenerator starten	RANDOMIZE
READ	Zuordnung der in der DATA-Anweisung stehenden Werte zu den angegebenen Variablen	READ Variable[,Variable,…]
REM	Kommentar-Kennzeichnung REM kann durch "!" ersetzt werden.	REM Kommentar
RENUMBER	Neunummerierung der Programmzeilen	RENUMBER[Ab-alte-Zn,Bis-alte-Zn [,Ab-neue-Zn[,Schrittweite]]

HC-BASIC	Bemerkung	Syntax
RESTORE	DATA-Zeiger wird auf die angegebene oder die erste DATA-Zeile gesetzt	RESTORE [Zeilennummer]
RETURN	Ende eines Unterprogramms	RETURN
RUN	Programmstart	RUN [Zeilennummer]
SOUND	Tonausgabe	SOUND z1,v1,z2,v2[,ls][,td]
STOP	Unterbrechen eines Programms	STOP wirkt wie Taste <BRK>
SWITCH	Modul- und Speicherverwaltung; Schreibschutz setzen und löschen	SWITCH m,k
TROFF	Ausschalten des Kontroll-Modus	TROFF
TRON	Einschalten des Kontroll-Modus	TRON
VPEEK	Lesen der IRM-Adresse i+32768	I=VPEEK<i) 0 <= I <= 255
VPOKE	Schreibt Byte j auf IRM-Adresse i+32768	VPOKE i,j 0 <= j <= 255
WAIT	Programm hält an, wenn an Port i anliegendes Bitmuster exklusiv-oder-verknüpft mit k und und-verknüpft mit j gleich Null ist.	WAIT i,j[,k]
WIDTH	Festlegen der Länge einer Ausgabezeile	WIDTH [Zeichenzahl]
WINDOW	Festlegen eines Fensters	WINDOW [za,ze,sa,se]

Funktionen

Tabelle 54: Übersicht der BASIC-Funktionen

HC-BASIC	Bemerkung
----------> MATHEMATISCHE FUNKTIONEN <----------	
ABS(X)	absoluter Betrag von X;
ATN(X)	Arcustangens, Resultat im Bogenmaß;
COS(X)	Cosinus (X im Bogenmaß);
EXP(X)	Exponentialfunktion: ex, X<=87.3365;
INT(X)	Ganzer Teil von X;
LN(X)	Natürlicher Logarithmus von X;
SGN(X)	Signumfunktion;
SIN(X)	Sinus (X im Bogenmaß);
SQR(X)	Quadratwurzel (√ x);
TAN(X)	Tangens (X im Bogenmaß);
----------> STRING-FUNKTIONEN <----------	
INSTR (A$,B$)	Ermittelt die Position, ab welcher A$ in B$ enthalten ist;

HC-BASIC	Bemerkung
LEFT$(A$,X)	Liefert die ersten X Zeichen von A$;
LEN(X$)	Zeichenlänge des X$;
MID$(A$,X,Y)	Y Zeichen von A$, beginnend mit dem X-ten Zeichen;
RIGHT$(A$,X)	Liefert die letzten X Zeichen von A$;
STRING$(N,A$)	Vervielfacht Zeichenkettenausdrücke;
STR$(X)	Formt den Wert X in einen String um;
VAL(A$)	Numerischer Wert von A$;
VGET$	Liefert den Inhalt der Cursorposition;
---------->	**SONSTIGE FUNKTIONEN** <----------
ASC(X$)	Liefert den ASCII-Code des ersten Zeichens von X$;
AT(z,s)	Schreibt PRINT-Anweisung an bestimmte Stelle des Bildschirms;
CHR$(X)	Liefert das Zeichen des ASCII-Codes X;
CSRLIN(N)	Liefert die Nummer der Zeile, in welcher der Cursor steht;
FRE(Variable)	Gibt die Größe des noch freien RAM-Speicherplatzes an;
INKEY$	Tastaturabfrage, Format: Stringvariable = INKEY$;
POS(I)	Liefert die aktuelle Schreibposition in der Zeile;
RND(X)	Erzeugt Zufallszahlen zwischen 0 und 1;
SPC(I)	Formatierungsfunktion;
TAB(I)	Formatierungsfunktion;
USR(X)	Aufruf einer Funktion, die als Maschinenprogramm geschrieben ist, mit Parameterübergabe.

Weitere mathematische Funktionen

Funktionen, über die das HC-BASIC nicht direkt verfügt, können mithilfe der Standardfunktionen berechnet werden. Eine kleine Auswahl von Beispielen gibt dazu folgende Übersicht.

Tabelle 55: Weitere mathematische Funktionen in BASIC

Funktion	Berechnung
SEKANS	SEC(X) = 1/COS(X)
COSEKANS	CSC(X) = 1/SIN(X)
COTANGENS	COT(X) = 1/TAN(X)
ARCUSSINUS	ARSIN(X) = ATN(X/SQR(1-X*X))
ARCUSCOSINUS	ARCOS(X) = -ATN(X/SQR(1-X*X))+PI/2
ARCUSCOTANGENS	ARCOT(X) = ATN(X)+PI/2
ARCUSSEKANS	ARSEC(X) = ATN(X/SQR(X*X-1))
ARKUSCOSECANS	ARCSC(X) = ATN(X/SQR(X*X-1))+(SGN(X)-1)*PI/2

Funktion	Berechnung
SINUS HYPERBOLICUS	SINH(X) = (EXP(X)-EXP(-X))/2
COSINUS HYPERBOLICUS	COSH(X) = (EXP(X)+EXP(-X))/2
TANGENS HYPERBOLICUS	TANH(X) = 1-EXP(-X)/(EXP(X)+EXP(-X))*2
COTANGENS HYPERBOLICUS	COTH(X) = EXP(-X)/(EXP(X)-EXP(-X))*2+1
SEKANS HYPERBOLICUS	SECH(X) = 2/(EXP(X)+EXP(-X))
COSEKANS HYPERBOLICUS	CSCH(X) = 2/(EXP(X)-EXP(-X))
ARCUSSINUS HYPERBOLICUS	ARSINH(X) = LN(X+SQR(X*X+1))
ARCUSCOSINUS HYPERBOLICUS	ARCOSH(X) = LN(X+SQR(X*X-1))
ARCUSTANGENS HYPERBOLICUS	ARTANH(X) = LN((1+X)/(1-X))/2
ARCUSCOTANGENS HYPERBOLICUS	ARCOTH(X) = LN((X+1)/(X-1))/2
ARCUSSEKANS HYPERBOLICUS	ARSECH(X) = LN(SQR(1-X*X)+1)/X)
ARCUSCOSEKANS HYPERBOLICUS	ARCSCH(X) = LN((SQR(1+X*X)+1)/X)*SGN(X)

Farbwerte

Die Farbfestlegung erfolgt durch die Anweisungen
COLOR v, h
INK v
PAPER h

Der Farbcode für Vordergrund errechnet sich wie folgt:
Farbcode v = 16 * b + f
f - Code der Vordergrundfarbe
h - Code der Hintergrundfarbe
b - Code zum Blinken der Vordergrundfarbe
(b = 1 für Blinken; b = 0 für Nicht-Blinken)

Zur Übersicht der zur Verfügung stehenden Farben siehe Tabelle 48, Seite 303.

Die Grafikbefehle PSETx,y,g
CIRCLExm,ym,r,g
LINExa,ya,xe,ye,g
können mit verschiedenen Farbauflösungen verwendet werden. Für die einfache Farbauflösung (LoRes), d. h. ein Farbbyte für 1x8 Punkte, ergibt sich folgende Berechnungsvorschrift für die Vordergrundfarbe: g = 16 * b + f + c
c = 0 Normalfall
c = 64 Überlagerungsfunktion (siehe auch Beschreibung der Speicherzelle
c =128 Löschfunktion FARB = 0B7D6H auf Seite 3.6.8)

Wird mit der hochauflösenden Grafik (pixelweise) gearbeitet, sind nur 4 Farben möglich: 0 schwarz
1 rot
2 türkis
3 weiß
Blinken, Überlagerungs- und Löschfunktion sind nicht möglich.

Fehlermeldungen

Nachdem ein Fehler aufgetreten ist, kehrt der BASIC-Interpreter auf die Kommandoebene zurück.

Das Programm kann nicht mit einem CONT-Befehl fortgesetzt werden. Der Zusammenhang aller GOSUB- und FOR-Anweisungen wird erst bei einem erneuten Programmstart wieder hergestellt.

Die Fehlermeldung hat folgendes Format:
Direkte Betriebsart: ?XX ERROR
Indirekte Betriebsart; ?XX ERROR IN n
(XX – Fehlercode; n – Zeilennummer)

Tabelle 56: Liste der BASIC-Fehlermeldungen

BAD	Falsche Verwendung der Elemente des Variablenfeldes
BAD IN	Fehler beim Laden bzw. Retten eines Variablenfeldes
BS	Subscript out of range (Feldelement außerhalb des dimensionierten Bereiches aufgerufen)
CN	Can't continue (Programm kann nicht mit CONT fortgesetzt werden)
DD	Doubly defined array (Feld mehrfach dimensioniert)
FC	Illegal function call (unzulässiger Funktionsaufruf)
ID	Illegal direct (fehlerhafte Eingabe im Direktbetrieb)
IO	Input-Output-ERROR (falscher Name beim Programmladen)
LS	String too long (String länger als 255 Zeichen)
MO	Missing Operand (Anweisung unvollständig, Operand fehlt)
NF	Next without for (Variablen von NEXT und FOR passen nicht zusammen)
OD	Out of Data (Es wurden durch die Data-Anweisungen zu wenig Daten für eine READ-Anweisung spezifiziert)
OM	Out of memory (vorhandener Speicherplatz im RAM reicht für die Ablage bzw. Abarbeitung eines Programms nicht aus)
OS	Out of String space (vereinbarter Speicherplatz für String reicht nicht aus)
OV	Numeric overflow (Ergebnis einer Berechnung ist größer als 1.70141E38)
RG	Return without GOSUB (RETURN trat vor GOSUB auf)
SN	Syntax ERROR (syntaktischer Fehler)
ST	Literal string pool table full (String zu lang oder zu komplex)
TM	Type mismatch (Variablen einer Gleichung indizieren verschiedene Typen, z. B. Zahl und String, oder einer Funktion wurde anstatt einer Zahl ein String übergeben oder umgekehrt)
UF	Undefined user function (Funktion noch nicht definiert)
UL	Undefined line (Es wurde eine nicht existente Zeilennummer angegeben)
/0	Division by zero (Division durch Null)

4.2. Assembler ASM

EDAS ist ein Editor/Assembler für den KC 85/2-5. Bisher bekannte Versionen sind:

EDAS 1.2	RAM-Version zum Nachladen von Kassette
EDAS 1.3	Modulversion mit HC901-CAOS und EDAS 1.3
EDAS 1.4	Version vom Modul M027 DEVELOPMENT
EDAS 1.5	mit DEVEX erweiterte Version ohne Laufwerkwechsel
EDAS 1.6	mit DEVEX erweiterte Version mit Laufwerkwechsel (D004)
EDAS 2.0	Bestandteil von CAOS 2.5 für den KC 85/2-3

Für den KC 85/5 mit CAOS 4.7 wurde dieser Assembler erneut weiterentwickelt. Er unterstützt jetzt das DEVICE-System von CAOS 4.7 direkt. Veröffentlicht wurden diese Versionen:

EDAS 1.7	Editor/Assembler mit integriertem 40-Zeichen-Editor
ASM 2.0	Assembler mit Zugriff auf externem 80-Zeichen-Editor
ASM 2.1	wie ASM 2.0, zusätzlich Unterstützung binäres Zahlenformat

Im Gegensatz zu EDAS besitzt der Assembler ASM keinen Editor. In ASM ist stattdessen der Aufruf des 80-Zeichen-Texteditors integriert, welcher auch als eigenständiger Texteditor genutzt werden kann. Siehe Kapitel 4.5 ab Seite 428. ASM kann dafür einen größeren Quelltextspeicher verwalten. Da kein Editor mehr enthalten ist, erfolgte die Umbenennung von EDAS in ASM. Im folgenden Text wird der Assembler nur noch ASM genannt, die Aussagen gelten mit Einschränkungen auch für die älteren EDAS-Versionen. Auf Unterschiede wird hingewiesen.

Der Assembler ASM gestattet es dem Anwender, Programme in symbolischer Form auf dem Kleincomputer zu schreiben, d. h. durch Eingabe leicht zu merkender Abkürzungen. Dadurch ist es möglich, ohne Kenntnis des eigentlichen binären Formats des Maschinenbefehls Maschinenprogramme zu erstellen. Darüber hinaus brauchen auch die Werte von Speicher-Adressen nicht berücksichtigt zu werden, sondern es genügt an deren Stelle mit frei wählbaren symbolischen Namen (sog. "Marken" oder "Labels") zu arbeiten, die die Speicherstellen kennzeichnen. Als Operanden können Namen von Speicherstellen, Registernamen und Konstanten auftreten.

Neben den eigentlichen vom Prozessor ausführbaren Anweisungen umfasst die hier beschriebene Assemblersprache auch Anweisungen für den Übersetzungsvorgang, die es dem Programmierer erlauben, die Arbeitsweise des Übersetzungsprogramms zu beeinflussen. Solche nicht ausführbaren Anweisungen nennt man "Pseudoanweisungen". Ein Programm, das aus einer Folge von ausführbaren und nicht ausführbaren Anweisungen besteht, nennt man Quellprogramm oder Quelltext. In jeder Zeile dieses Quelltextes können vor dem eigentlichen

Befehl und den Operanden dieses Befehls auch eine Marke stehen und nach dem Befehl ggf. ein Kommentar zur Erläuterung.

Der gesamte Quelltext wird vom Assembler in die eigentliche Maschinensprache (in binärer Form) des U880- bzw. Z80-Prozessors (CPU) übersetzt, wobei jedem Assemblerbefehl genau ein Maschinenbefehl entspricht. Das auf diese Weise erstellte Maschinenprogramm kann nun auf dem Computer direkt ausgeführt werden (im Gegensatz z. B. zu BASIC-Programmen, die den BASIC-Interpreter während der Ausführung benötigen).

Es sei darauf hingewiesen, dass es verschiedene Assemblersprachen für die U880 bzw. Z80 CPU gibt. Der hier beschriebene Assembler verarbeitet Quellprogramme in ZILOG-Mnemonik (wie z. B. die Assembler der Bürocomputer A5120/30 unter den Betriebssystemen UDOS und SCPX).

4.2.1. CAOS-Kommandos ASM und REASM

Mit diesen CAOS-Menükommandos wird der Assembler gestartet:

%ASM	Kaltstart CAOS-Assembler ASM ab CAOS 4.8
%REASM	Warmstart CAOS-Assembler ASM ab CAOS 4.8

Falls eine ältere Version von EDAS benutzt wird, heißen die Menüworte:

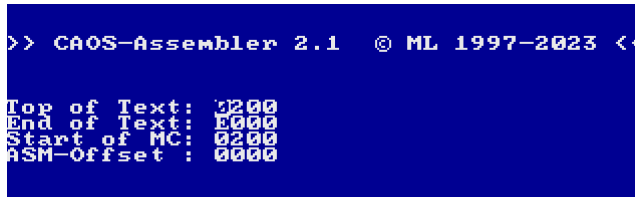
%EDAS	Kaltstart Editor/Assembler EDAS bis CAOS 4.7
%REEDAS	Warmstart Editor/Assembler EDAS bis CAOS 4.7

Ist mit ASM ein Quelltext erstellt worden, kann als Folge mit REASM gestartet werden. Dabei erfolgt keine Initialisierung des Speichers (Warmstart) und der Quelltext bleibt erhalten. Ohne vorherige Initialisierung führt ein Warmstart zu undefinierten Zuständen und kann einen Programmabsturz verursachen.

ASM ist ein 2-Pass-Assembler, welcher sich in der Ebene E1 des USER-ROM befindet. Er belegt den Speicherbereich von C000H-DFFFH. Eine Zuschaltung des Speicherbereiches ist nicht erforderlich, da CAOS den USER-ROM automatisch nach Menüworten durchsucht.
Sie können die RAM-Grenzen für den Quelltext und die Lage des Maschinen-Code-Bereichs in den Grenzen von 0200H bis C000H (bei Vorhandensein eines Moduls M022 oder M011 bis E000H) frei wählen. Für den Bereich ab 8000H wird der IRM und der USER-ROM bei Zugriff weggeschaltet. Der Schaltzustand wird durch die System-LED angezeigt, LED aus entspricht IRM und USER-ROM aus.
Die mit der Option "O" erzeugten MC-Programme werden mit eingeschaltetem IRM geschrieben, sodass man auch Programme für den Speicherbereich von BA00H bis C000H erzeugen kann.

4.2. Assembler ASM

Beim Start von ASM mit dem gleichlautenden Menüwort erscheint unter der Titelzeile eine Abfrage zur Festlegung des zu benutzenden Speichers:



Top of Text:	0200	Festlegen der unteren RAM-Grenze für den Quelltext.
End of Text:	E000	Festlegen der oberen RAM-Grenze+1. Der Vorgabewert ist das Maximum, das mit dem momentanen Schaltzustand der Module bzw. des internen Speichers erreicht werden kann.
Start of MC:	0200	Modifizierung des standardmäßigen Maschinencode-Bereiches. (hat bei Verwendung des ORG-Befehls keine Bedeutung)
ASM-Offset:	0000	Modifizierung des Assembleroffsets für Option "O". Die physische Speicheradresse ergibt sich aus logischer Adresse plus Assembleroffset.

Der Quelltext liegt grundsätzlich geteilt in zwei Teiltexten vor. Trennstelle ist gleichzeitig der Bildschirmanfang des Editors. Die Markentabelle liegt hinter dem Teiltext 2. Der in der Statuszeile angegebene freie Speicher ist der Zwischenraum zwischen den beiden Teiltexten.

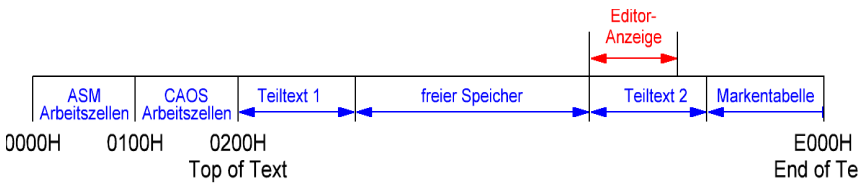


Bild 37: Speicherbelegung im Editor/Assembler

Während des Assemblerlaufes liegt der Text ungeteilt am Speicheranfang, sodass die Markentabelle von der vereinbarten Speicherobergrenze nach unten wachsen kann. Dabei wird überwacht, dass sie nicht in den Text hinein wächst. In diesem Fall würde eine Fehlermeldung ausgegeben und der Assemblerlauf abgebrochen.

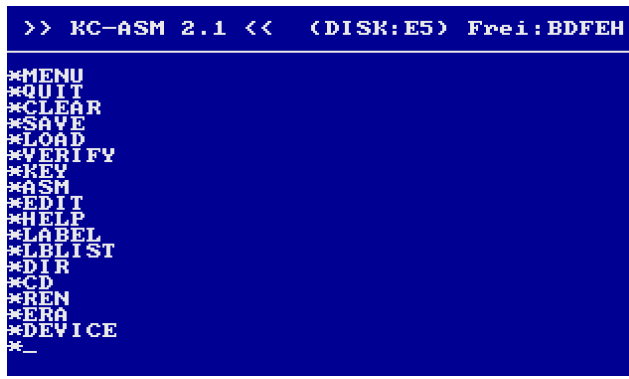
4.2. Assembler ASM

4.2.2. Das Assembler-Menü

Beim Start des Assemblers bzw. bei Rückkehr aus dem Editor erscheint ein kurzes Menü. Das komplette, lange Menü mit weiteren Menüpunkten kann durch das Menüwort ***MENU** angezeigt werden.

In der Titelzeile wird in der Mitte das aktuell eingestellte DEVICE und bei Diskette zusätzlich das eingestellte Laufwerk mit USER-Bereich angezeigt.

Rechts wird außerdem noch in hexadezimaler Schreibweise der freie Speicher angezeigt, der nicht vom Quelltext oder der Markentabelle belegt ist.



Im Einzelnen bedeuten die Menüworte:

ASM-Kommando MENU

*MENU Ausschreiben des vollständigen Menüs.
Befindet sich eine geladene oder bereits abgespeicherte Datei im Speicher, dann zeigt die Titelzeile an Stelle der ASM-Version den Dateinamen an.

ASM-Kommando QUIT

*QUIT verlassen des Assemblers zu CAOS

ASM-Kommando CLEAR

*CLEAR Löschen des Text- und/oder Markenspeichers. Durch zwei unabhängige Fragen kann auch nur der Text- bzw. Markenspeicher gelöscht werden.

4.2. Assembler ASM

ASM-Kommando SAVE

*SAVE Abspeichern des im Speicher befindlichen Quelltextes auf das aktuell eingestellte Speichergerät. Der letzte verwendete Dateiname wird wieder vorgegeben, als Dateityp wird ASM vorgegeben. Dateiname und -Typ können aber überschrieben werden. Abgespeichert wird im CAOS-Format mit Vorblock, der den Dateinamen enthält. Der Quelltext selbst ist ASCII-Text mit CR+LF als Zeilenwechsel. Nach der letzten Quelltextzeile kommt nochmals CR+LF und die Ende-Kennung 03H. Der Rest des letzten Datenblockes bis zu vollen 128 Bytes wird mit 1AH (CP/M-Ende-Zeichen) aufgefüllt.

! Hinweis: Soll der Quelltext ohne CAOS-Vorblock gespeichert werden, dann kann zum Editor gewechselt und die dortige SAVE-Funktion benutzt werden.

ASM-Kommando LOAD

*LOAD Einlesen eines Quelltextes vom aktuell eingestellten Speichergerät. Es können sowohl Dateien im CAOS-, bzw. EDAS-Format als auch solche im Textformat eingelesen werden. Die Unterscheidung erfolgt automatisch, wenn kein CAOS-Vorblock erkannt wird. Als Ende-Kennung wird sowohl der Code 03H (CAOS-Ende-Zeichen) als auch 1AH (CP/M-Ende-Zeichen) ausgewertet.

Falls sich noch ein Quelltext im Speicher befindet, wird der neue Text stets vor die erste Zeile der zuletzt bearbeiteten Textseite, also nach Teiltext 1 eingefügt. Damit kann eine „merge“-Funktion realisiert werden. Für ein völlig neues Programm ist der Speicher zunächst mit CLEAR zu leeren!

Ist als DEVICE Kassette eingestellt, dann wird die nächste Datei vom Recorder eingelesen und dabei der Dateiname aus dem Vorblock angezeigt und abgespeichert. Tritt während des Ladevorgangs ein Lesefehler auf ("*" erscheint), so kann nach Rückspulen der Kassette erneut versucht werden, den fehlerhaften Block zu lesen. Gelingt das trotz mehrmaligen Probierens nicht, so ist der Ladevorgang mit der BREAK-Taste abzubrechen. Bei Abbruch ist der bis dahin eingelesene Text verfügbar. Auf die BREAK-Taste reagiert der Rechner nur, wenn ein Signal vom Kassettenrecorder anliegt.

ASM-Kommando VERIFY

*VERIFY Aufruf der CAOS-Routine VERIFY zur Kontrolle von Kassettenaufzeichnungen. Bei Quelltexten mit mehr als 0FFH Blöcken wird der Vergleich systembedingt bei Block FF> abgebrochen.

4.2. Assembler ASM

ASM-Kommando KEY

*KEY [n] Ohne Parameter: KEYLIST (Aufruf Menükommandoroutine)
n = 0 : Grundbelegung des F-Tastenspeichers herstellen
1 <= n <= F : wie %KEY (Aufruf Menükommandoroutine)
n = D5 : Grundbelegung für D005-Tastatur herstellen

Beim Kaltstart von ASM wird die Grundbelegung der KC-Tastatur initialisiert, dabei ist F1=ESC, F2=TAB und F3=weitersuchen für Find-Funktion des Editors.

ASM-Kommando ASM

*ASM Übersetzen des Quelltextes in Maschinencode. Dabei kann mit folgenden Optionen der Assemblerlauf beeinflusst werden:

- + Eine bereits vorhandene Markentabelle wird vor dem Assemblerlauf nicht gelöscht.
- 1 Es wird nur der Pass 1 (Erzeugung der Markentabelle) ausgeführt
- 2 Die bestehende Symboltabelle wird ohne Aktualisierung wieder verwendet (nur Pass 2 wird ausgeführt). Die Option "+" braucht dabei nicht extra angegeben zu werden.
- B Alle Ausgaben (Listing und Fehleranzeigen) werden auf Bildschirmformat (39 Zeichen) verkürzt.
- L Ausgabe des kompletten Assemblerlistings auf Bildschirm (Drucker)
- O Der erzeugte Maschinencode wird direkt in den Speicher geschrieben. Um ein Überschreiben der vom Assembler benötigten Speicherbereiche zu verhindern, hat der Programmierer geeignete ORG-Adressen und Parameter beim Start von ASM zu wählen. ASM überwacht ab Version 2.1 die Speicherbereiche und erzeugt einen Fehlercode "M", wenn unzulässige Adressen beschrieben werden sollen.
- P Ausgabeumschaltung auf den Drucker. Dazu muss zuvor im CAOS-Menü ein Drucker auf der Schnittstelle UOUT1 initialisiert worden sein.
- S Der generierte Maschinencode wird auf das eingestellte Speichergerät ausgegeben. In diesem Fall darf der Quelltext nur eine ORG-Anweisung am Anfang enthalten.

Bei Verwendung einer Marke mit dem Namen "START", wird diese als Startadresse für selbststartende Programme interpretiert. Diese Adresse wird als dritter Parameter bei der Assembler-Option 'S' mit ausgegeben. Der Pass 3, also die Ausgabe des Maschinencodes mit der Option 'S' wird jedoch nur erreicht, wenn der Pass 2 fehlerfrei gelaufen ist. Zum Abschluss wird neben der Fehlerzahl auch die Endadresse+1 und eventuell eine vorhandene Startadresse ausgegeben. Der Assemblerlauf kann an jeder Stelle mit BRK abgebrochen werden. Bei der Option

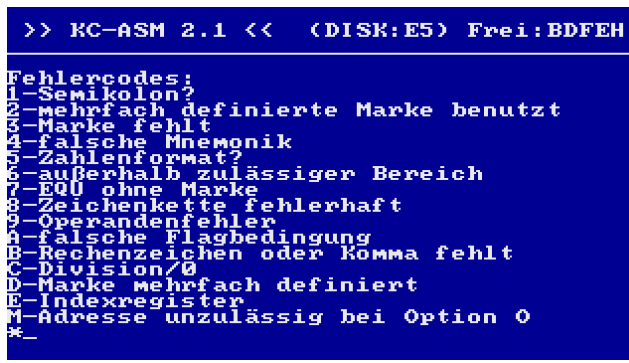
'S' wird der Dateityp KCC erzeugt, auch dieser kann bei Bedarf überschrieben werden.

ASM-Kommando EDIT

*EDIT wechselt direkt zum Editiermodus des 80-Zeichen-Editors. Die Rückkehr mit BRK führt zunächst zum Menü des Editors, von dem weitere Kommandos erreichbar sind. Vom Editor zum Assembler gelangt man mit den Kommandos QUIT oder ASM. QUIT führt dabei zum Menü, während ASM direkt zum gleichnamigen Kommando des Assemblers führt.

ASM-Kommando HELP

*HELP [c]
zeigt eine kurze Erklärung zu den vom Assembler verwendeten Fehlercodes an. Als optionaler Parameter c kann gezielt die Beschreibung für einen Fehlercode abgefragt werden. Ansonsten erfolgt eine Auflistung aller Fehlercodes. Siehe dazu auch Tabelle 60 auf Seite 397.



ASM-Kommando LABEL

*LABEL zeigt den Wert einer Marke an. Der Name der Marke wird abgefragt, Voraussetzung: Assemblerlauf mit Pass 1 wurde durchlaufen.

ASM-Kommando LBLIST

*LBLIST [n]
gibt eine Liste aller verwendeten Marken und deren Werte in n (sonst 3) Druckspalten aus.
Voraussetzung: ein Assemblerlauf mit Pass 1 wurde durchlaufen.

ASM-Kommandos DIR, CD, REN, ERA und DEVICE

Diese 5 Kommandos haben die gleiche Syntax und Wirkung wie die gleichnamigen Kommandos im CAOS-Menü, siehe Seite 51 bis 60.

4.2.3. Arbeitsweise des Prozessors

Mit dem U880-Prozessor stehen dem Programmierer über 600 Operationscodes für arithmetische, logische, Programmorganisations-, Datentransfer- sowie Ein-/Ausgabe-Befehle zur Verfügung.

Sämtliche Befehle eines abzuarbeitenden Maschinenprogramms stehen in externen Speicherbausteinen, dem Hauptspeicher des Rechners. Die Befehlsabläufe ("Befehlszyklen") sehen alle im Prinzip gleich aus (siehe Bild 38).

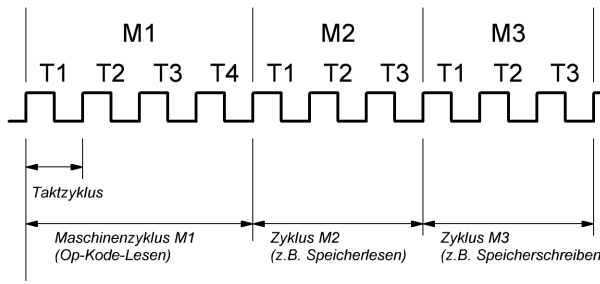


Bild 38: Beispiel eines Befehlszyklus

Man erkennt aus Bild 38, dass folgende Hierarchie besteht:

Die Befehlszyklen bestehen aus mehreren Maschinenzyklen. Die Anzahl der Maschinenzyklen pro Befehl ist unterschiedlich (zwischen 1 und 6). Jeder Maschinenzyklus besteht seinerseits aus mehreren "Taktzyklen" (3 bis 6 Taktzyklen). Die Dauer eines Taktzyklus ist durch die Frequenz des Taktgenerators der CPU gegeben. Beim KC 85/5 beträgt die Taktfrequenz etwa 1,77 MHz, d. h. die Dauer eines Taktzyklus beträgt etwa 565ns.

In Tabelle 61 ab Seite 399 ist für die einzelnen Befehle der CPU die Gesamtanzahl der Taktzyklen T jeweils angegeben. Man kann hieraus die Ausführungszeit eines Befehls ermitteln, was z. B. für Programme mit Zeitschleifen notwendig ist.

Zur Arbeitsweise der CPU und des Interruptsystems sei außerdem auf die einschlägige Literatur verwiesen, z. B. [15], [21] und [54] bis [56].

4.2.4. U880/Z80-Registerstruktur

Der U880 bzw. Z80 besitzt einen zweigeteilten Registersatz mit jeweils 6 allgemeinen Registern HL, DE und BC, die wahlweise als 8-Bit-Register oder als 16-Bit-Register benutzt werden können. Dazu jeweils ein Akkumulator A, in dem die arithmetischen und logischen Operationen ausgeführt werden und ein Flagregister F zur Statusidentifikation.

Die zwei Indexregister IX und IY ermöglichen indizierte Adressierungen. Daneben existiert noch ein 16-Bit-Stackpointer SP, ein 16-Bit Befehlszähler PC, das Interruptvektor-Register und ein 7-Bit-Refreshregister R.

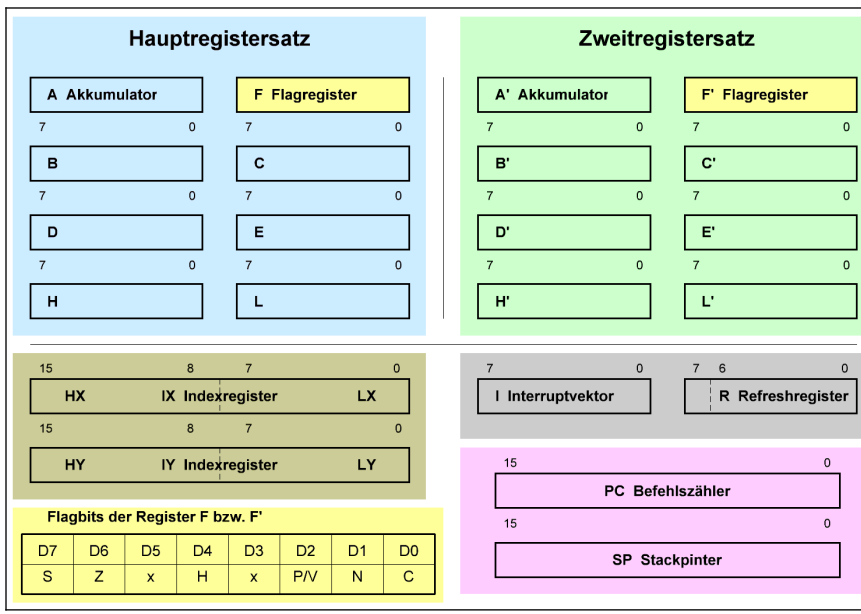


Bild 39: U880/Z80-Registerstruktur

Der Zahlenbereich der 8-Bit-Register geht von 0 bis 255 (bzw. -128 bis +127 bei vorzeichenbehafteten Zahlen) und der Zahlenbereich der 16-Bit-Register von 0 bis 65535 (bzw. -32768 bis +32767).

Nach dem Einschalten des Rechners wird immer der Hauptregistersatz angesprochen. Das Umschalten auf die Hintergrundregister geschieht durch 2 Austauschbefehle getrennt für Akkumulator/Flag (EX AF,AF') und Allgemeinregister (EXX). Danach beziehen sich alle Befehle bis zum erneuten Umschalten auf die Hintergrundregister.

Akkumulator

Das 8-Bit-Register A dient bei arithmetischen und logischen Befehlen zur Aufnahme eines Operanden. Der andere Operand kommt aus einem anderen Register oder aus dem Speicher. Das Ergebnis der Operation wird wieder im Akkumulator abgelegt, und steht dort für eine weitere Verarbeitung zur Verfügung.

Allgemeine Register

Diese können als 8-Bit-Register (B, C, D, E, H, L) oder als 16-Bit-Registerpaare (BC, DE, HL) benutzt werden. Die Register kann man frei als Zwischenspeicher verwenden, jedoch beziehen sich bestimmte Befehle auf einzelne Register. So dient das HL-Registerpaar der indirekten Speicheradressierung. Das DE-Registerpaar kann mit dem Registeraustauschbefehl über HL demselben Zweck dienen. Bei einigen Befehlen werden beide Registerpaare als Adressenspeicher für Quell- und Zieladressen benutzt (z. B. LDIR).
Das Register B bzw. BC wird vorwiegend als Zählregister verwendet.

Befehlszähler

Das 16-Bit-Register PC enthält den aktuellen Befehlszählerstand. Beim Einschalten des Rechners wird der Befehlszähler auf Null gesetzt. Bei Sprung- und Unterprogrammbefehlen wird er mit einem neuen Wert geladen, sonst wird er automatisch jeweils um die Befehlslänge erhöht.

Stackpointer

Der Stackpointer SP enthält die 16-Bit-Adresse der aktuellen Spitze des Kellerspeichers der CPU. Der Kellerspeicher arbeitet nach dem Prinzip, dass die zuletzt gespeicherten Daten wieder als erste ausgegeben werden ("last in – first out"). Er dient vorwiegend zur Aufnahme der Rückkehradressen bei Unterprogrammaufrufen und Interruptroutinen. Außerdem kann er zum Ablegen (PUSH) und Wiedereinlesen (POP) von 16-Bit-Daten aus den Registern verwendet werden. Durch Setzen des Stackpointers im Initialisierungsprogramm des Rechners wird die Lage des für den Kellerspeicher zur Verfügung stehenden Teils des Operativspeichers (RAM-Bereich) festgelegt. Die Größe ist zunächst nicht begrenzt. Beim Programmerstellen ist aber zu beachten, dass für das jeweilige Programm ausreichend Kellerspeicher-Plätze zur Verfügung stehen.

Der Stackpointer wird beim Abspeichern im Keller um 2 Byte verkleinert und beim Einlesen um 2 Byte erhöht. Er zeigt immer auf den zuletzt eingespeicherten Wert.

Ein Unterprogrammaufruf kellert den Befehlszählerstand ein, ein Rückkehrbefehl kellert den PC wieder aus. Außerdem können in den Unterprogrammen Daten mit PUSH und POP zwischengespeichert werden. Damit nicht versehentlich Befehls-

zähler und Daten verwechselt werden, ist bei der Verwendung von PUSH und POP größte Sorgfalt auf ein symmetrisches Ein- und Auskellern zu legen.

Beim KC 85/5 zeigt der Stackpointer nach dem Einschalten und nach RESET auf die Adresse 01C4H.

Indexregister

Die Indexregister IX und IY werden zur indizierten Adressierung (siehe Kapitel 4.2.6) verwendet oder können als 16-Bit-Datenregister verwendet werden.

Bei der indizierten Adressierung kann auf einen Speicherbereich in einer Umgebung von -128 bis +127 um den im Register gespeicherten Adressenwert direkt zugegriffen werden.

Das IX-Register wird beim KC 85/5 als Zeiger zu Systemspeicherzellen genutzt und sowohl in den Systemunterprogrammen als auch in Systeminterruptroutinen verwendet und sollte in Anwenderprogrammen nicht verändert werden. Das IX-Register wird beim Einschalten und nach RESET beim KC 85/5 mit dem Wert 01F0H initialisiert.

Interruptvektorregister

Das I-Register beinhaltet den höherwertigen Adressteil der Tabelle für die Interruptroutinen. Es hat beim KC 85/5 nach Einschalten oder RESET den Wert 1 (siehe dazu Kapitel 4.2.5).

Refreshregister

Dieses 7-Bit-Register R wird bei jeder Befehlsverarbeitung um 1 erhöht. Es dient zum Auffrischen der Inhalte der dynamischen RAM-Speicher und ist für den Programmierer kaum von Bedeutung. Es wird oft für die Berechnung von Zufallszahlen mit einbezogen.

Flag-Bits

Die CPU verfügt über zwei Status-(Flag-)Register (siehe Bild 39). Durch Veränderung einzelner Bits wird über die Art des Ergebnisses der letzten Prozessoroperation Auskunft gegeben. Diese Auskunft wird hauptsächlich dazu benutzt, bedingte Sprünge vorzunehmen, d. h. je nach Ergebnis der Prüfung eines dieser Bedingungsbits die eine oder aber eine andere Aktion durchzuführen.

Im Ergebnis der Befehlsausführung werden je nach Operation verschiedene Statusbits im F-Register gesetzt. Die Anordnung der 8 Bit in diesem Register ist noch einmal in Bild 40 dargestellt.

Flagbits der Register F bzw. F'

D7	D6	D5	D4	D3	D2	D1	D0
S	Z	x	H	x	P/V	N	C

Bild 40: Flag-Bits im F-Register

Hierbei bedeuten:

C – Übertragsbit (Carry-Flag)
N – Additions-/Subtraktionsbit (Add/Subtract-Flag)
P/V – Paritaets-/Überlaufbit (Parity/Overflow-Flag)
H – Halb-Byte-Überlaufbit (Half-Carry-Flag)
Z – Nullbit (Zero-Flag)
S – Vorzeichenbit (Sign-Flag)
X – nicht verwendet

Bei der Programmabarbeitung können 4 dieser Bits (S, Z, P/V, C) direkt als Sprungbedingung ausgewertet werden. Zwei Flags (H und N) dienen der CPU zur Realisierung der BCD-Arithmetik, zwei Bits (D3 und D5) sind nicht gesetzt. Die Beeinflussung der Flagbits durch die CPU-Befehle ist im Kapitel 4.2.10. jeweils angegeben.

Das **Carry-Flag** wird gesetzt (=1), wenn bei der Addition zweier Operanden ein Übertrag von Bit 7 entsteht, sowie wenn bei der Subtraktion ein Bit geborgt werden muss (das Ergebnis negativ wird). Darüber hinaus fungiert das Carry-Flag als Bit-Speicher bei Verschiebe- und Rotationsbefehlen.

Das **Zero-Flag** wird gesetzt, wenn das Ergebnis einer Operation den Wert Null ergibt. Bei Einzelbitbefehlen dient es zur Übergabe ausgelesener Bits.

Die Funktion des **P/V-Flag**s hängt von der verwendeten Operation ab. Bei logischen und Verschiebebefehlen wird die Parität des Ergebnisses angezeigt (gerade Parität: P/V = 1; ungerade Parität P/V = 0). Bei arithmetischen Befehlen wird das P/V-Flag als Vorzeichen-Überlaufkennzeichnung benutzt; es wird z. B. gesetzt, wenn das Ergebnis zweier Vorzeichenzahlen außerhalb des zulässigen Bereiches von -128 bis +127 liegt.

Das **Sign-Flag** zeigt nach Additionen und Subtraktionen an, ob das Ergebnis positiv ist (S = 0) oder negativ (S = 1).

Das **Half-Carry-Flag** wirkt wie das Carry-Flag, jedoch wird der Übertrag von Bit 3 auf Bit 4 angezeigt.

Mit dem **Add/Subtract-Flag** wird gekennzeichnet, ob als letzter Befehl eine Addition (N = 0) oder eine Subtraktion (N = 1) durchgeführt wurde.

Die genaue Reaktion der Flags auf die einzelnen Befehle kann der Befehlsliste (siehe Tabelle 61) entnommen werden.

4.2.5. Interruptsystem

Soll der Rechner auf externe Ereignisse reagieren, so hat man die Möglichkeit, entweder den betreffenden Eingabekanal ständig abzufragen oder das laufende Programm mittels Interrupt zu unterbrechen und nach Reaktion auf das Eingabesignal (Interruptprogramm) das ursprüngliche Programm fortzusetzen. Die Tastaturabfrage und das Kassetteninterface des KC 85 arbeiten z. B. mit Interrupt.

Die U880-CPU hat zwei getrennte Signaleingaenge zur Auslösung von Interrupts:

NMI – nichtmaskierbarer Interrupt höchster Priorität

INT – maskierbarer Interrupt (kann in 3 verschiedenenInterrupt-Modi betrieben werden)

Nichtmaskierbarer Interrupt (NMI)

Die NMI-Signalzuführung kann nicht gesperrt werden. Ein NMI-Signal führt also in jedem Fall zu einer Unterbrechung des laufenden Programms und zu einem erzwungenen Unterprogrammsprung zur Speicheradresse 0066H. An dieser Stelle muss die Interruptbehandlungsroutine vom Programmierer eingetragen sein.

Die Interruptroutine muss am Ende mit einem Rücksprung ins unterbrochene Programm (mit dem Befehl RETN) abgeschlossen werden.

Maskierbarer Interrupt (INT)

Die INT-Signalzuführung kann mithilfe der Befehle EI eingeschaltet (enable interrupt) und mittels DI ausgeschaltet (disable interrupt) werden (maskieren). Ist das Interruptsystem nicht freigegeben (DI), so werden INT-Anforderungen ignoriert.

Die Steuerung des INT-Eingangs der CPU erfolgt über DIE zwei Merker IFF1 und IFF2 (Interruptflipflops). Der Befehl EI setzt beide auf 1 und DI beide auf 0. IFF2 dient als Merker der Stellung von IFF1 bei der NMI-Behandlung (während der NMI-Behandlung, d. h. bis RETN, ist kein INT möglich).

Wird nun ein anstehendes INT-Signal erkannt, so führt die CPU hardwaremäßig eine DI-Operation aus, d. h. IFF1 und IFF2 werden auf 0 gesetzt. Sollen weiterhin Interrupts zugelassen werden, so ist spätestens vor Verlassen der Interruptbehandlungsroutine mit RETI der Befehl EI zu programmieren.

Der maskierbare Interrupt INT kann in 3 Arbeitsweisen betrieben werden, die mit den Befehlen IM 0, IM 1 bzw. IM 2 eingeschaltet werden. Der KC 85 arbeitet stets im Interrupt-Mode IM 2. Eine Veränderung würde zum Absturz des CAOS-Betriebssystems führen. Zur Volständigkeit sind die drei Modi beschrieben.

Interrupt-Mode IM 0

Wird ein INT-Signal akzeptiert, so wird gleichzeitig ein auf dem Datenbus vom Interruptauslöser (peripherer Baustein) bereitzustellender 1-Byte-Befehl eingelesen und anschließend ausgeführt. Im Normalfall werden dazu die Restart-Befehle

RST n (n = 0, 8, 10H, 18H, 20H, 28H, 30H, 38H)

verwendet, die einen Unterprogrammaufruf zur Speicheradresse n bewirken, an der die Interruptbehandlungsroutine beginnen muss.

Interrupt-Mode IM 1

Beim Akzeptieren des INT-Signals wird unabhängig von den anderen Eingängen ein Unterprogrammsprung zur Adresse 0038H durchgeführt (ähnlich wie NMI).

Interrupt-Mode IM 2

Diese Betriebsart der CPU ist die leistungsfähigste und gestattet die individuelle Behandlung unterschiedlicher peripherer Bausteine. Der KC 85 arbeitet in diesem Modus. Die Interruptbehandlung läuft nach folgendem Schema ab:

In der CPU wird aus dem Wert des Interruptregisters I und dem vom peripheren Baustein auf dem Datenbus bereitzustellenden Interruptvektor IV eine 16-Bit-Adresse gebildet. Das Interruptregister I bildet dabei den höherwertigen Adressteil und der Interruptvektor IV den niederwertigen. Von dieser und der folgenden Speicheradresse wird die eigentliche Startadresse der Interruptbehandlungsroutine ausgelesen und ein Unterprogrammsprung dorthin durchgeführt. Die Interruptbehandlungsroutine muss mit RETI beendet werden, wobei ggf. zuvor das Interruptsystem mit EI einzuschalten ist. Mit dem Interruptregister I wird also die Lage der Tabelle der Startadressen für die Interruptbehandlungsroutinen im Speicher festgelegt. Durch den vom peripheren Baustein bereitgestellten Interruptvektor IV, der geradzahlig sein muss, wird eine der 128 möglichen Startadressen ausgewählt, an der die Interruptroutine beginnt.

Beim KC 85/5 hat das Interruptregister nach RESET den Wert 1, d. h. die Interrupttabelle steht im Speicher von 0100H bis 01FFH. Von diesen Adressen stehen für den Anwender im CAOS-System aber nur die Werte von 01C4H bis 01DFH außer 01D0/01D1H zur Verfügung, siehe Tabelle 22.

Die Anwenderbausteine müssen die entsprechenden Interruptvektoren 0C4H, 0C6H, …, 0DEH oder 0DFH liefern.

4.2.6. Adressierungsarten

Der Befehlssatz des Prozessors beinhaltet 6 verschiedene Adressierungsarten zur Bereitstellung von Register-, Speicher- oder Ein/Ausgabe-Adressen für zu spezifizierende Datenwörter.

Direkte Adressierung

Der Operationscode beinhaltet vollständig die entsprechende Adresse.

```
Z. B.:  LD A,B
        LD (0200H),HL
```

Implizite Adressierung

Der Operationscode bezieht sich fest auf bestimmte Speicherplätze oder Register.

```
Z. B.:  EXX
        SCF
```

Unmittelbare Adressierung

Dem Operationscode folgt unmittelbar eine 8- oder 16-Bit-Konstante im Speicher.

```
Z. B.:  LD A,6
        XOR 20H
```

Indirekte Adressierung

Die 16-Bit-Adresse befindet sich in einem Registerpaar der CPU. Der Befehl bezieht sich indirekt auf diese Adresse.

```
Z. B.:  LD A,(HL)
        LDIR
```

Indizierte Adressierung

Der Operationscode beinhaltet ein Datenbyte (zwischen -128 und +127), dass sich zum Inhalt des Doppelregisters IX oder IY addiert die vollständige Adresse ergibt.

```
Z. B.:  LD A,(IX+6)
```

4.2.7. Syntax der Assemblersprache

Ein Assembler-Quelltext besteht aus einer Folge von Anweisungen (sog. Statements), die zusammen das Anwenderprogramm ergeben. Jede der Quelltextzeilen ist aus einem Markenfeld, einem Operationscode- oder Anweisungsfeld (mnemonische Ausdrücke), einem Operandenfeld und einem Kommentarfeld aufgebaut.

Beispiel für eine Quelltextzeile:

Marke	Anweisung	Operand(en)	Kommentar
START:	LD	A,6	; Akku laden

Je nach Art der Befehle können oder müssen einzelne dieser Felder wegfallen. Die einzelnen Felder müssen durch eine beliebige Anzahl von Leerzeichen oder Tabulatoren voneinander getrennt werden. In jeder Zeile des Quelltextes können vor dem eigentlichen Befehl und den Operanden dieses Befehls auch eine Marke stehen und nach dem Befehl ggf. ein Kommentar zur Erläuterung. Im Quelltext wird außer in Zeichenketten nicht zwischen Groß- und Kleinbuchstaben unterschieden.

Marken

Marken sind symbolische Bezugspunkte innerhalb des Programms. Sie werden verwendet, um in einer anderen Anweisung auf den momentanen Befehlszählerstand, auf eine andere Marke oder auf eine Konstante Bezug nehmen zu können. Eine Marke muss in der ersten Spalte der Programmzeile beginnen und sollte 6 Zeichen nicht überschreiten. Das erste Zeichen muss ein Buchstabe sein, als weitere sind Buchstaben, Ziffern, der Punkt und das Unterstreichungszeichen zulässig. Marken können mit Doppelpunkt(en) abgeschlossen werden (müssen aber nicht).

Verboten für Marken sind folgende Zeichenketten:

Registernamen:	A, B, C, D, E, H, L, I, R, AF, BC, DE, HL, IX, IY, SP, HX, HY, LX, LY
Flagbedingungen:	C, NC, Z, NZ, M, P, PE, PO

Eine spezielle Bedeutung hat die Marke mit dem Namen START. Ist eine Marke mit diesem Namen vorhanden, dann wird diese als Startadresse für selbststartende Programme interpretiert.

Operationscodes

Im Anweisungs- oder Operationscodefeld steht eine der ZILOG-Maschinenbefehlsmnemoniken oder eine Assembler-Pseudoanweisung. Das Operationscode-

feld beginnt frühestens in der 2. Spalte der Programmzeile, d. h., wenn das Markenfeld leer ist, muss vor dem Operationscode mindestens ein Trennzeichen (Leerzeichen oder Tabulator) stehen.

Operanden

Je nach Art des Operationscodes muss das Operandenfeld entweder leer sein, oder es enthält einen oder zwei (durch ein Komma getrennte) Operanden, die eine Adresse (Speicher, Register oder Ein/Ausgabe-Kanal), eine Konstante oder eine Flag-Bedingung repräsentieren. Die Operanden ergänzen die jeweiligen Anweisungen durch eine Information darüber, mit welchen Parametern die Operation durchzuführen ist.

Es sind folgende Schlüsselwörter für die Operandenfelder reserviert:

- Die Namen der internen Register der CPU, die jeweils den 8-Bit-Inhalt eines dieser Register ansprechen. Die Registernamen sind: A, B, C, D, E, F, H, L, I und R, sowie HX, HY, LX, LY für die 8-Bit-Hälften der Indexregister.
- Die Namen der Doppelregister und Registerpaare. Doppelregister sind die 16-Bit-Register IX, IY, SP und PC. Über die Registerpaar-Bezeichnungen AF, BC, DE und HL lassen sich die oben bezeichneten 8-Bit-Register der CPU paarweise (als 16-Bit-Wörter) vom Assemblerbefehl ansprechen.
- Die in der CPU integrierten Zweitregister werden in der Assemblersprache mit einem Hochkomma gekennzeichnet. Die Namen sind AF', BC', DE' und HL'. Von diesen Namen ist jedoch lediglich die Kombination AF' als Operand zulässig (in der Anweisung EX AF,AF').
- Der Zustand der 4 vom Assemblerprogramm testbaren Bedingungs-Bits wird in der Flag-Bedingung wie folgt notiert.

Tabelle 57: Flag-Bedingungen

Bezeichnung	Bedingung erfüllt	Bedingung nicht erfüllt
Übertrags-Bit (Carry)	C	NC
Null-Bit (Zero)	Z	NZ
Vorzeichen-Bit (Sign)	M (Minus, negativ)	P (Plus, positiv)
Paritäts-/Überlauf-Bit (Parity / Overflow)	PE (gerade, even)	PO (ungerade, odd)

Arithmetik

Als Konstanten oder Speicheradressen können arithmetische Ausdrücke stehen, die durch eine beliebige Anzahl von Rechenoperationen von Dezimalzahlen, Hexadezimalzahlen, Binärzahlen, Marken, Zeichenkettenkonstanten, $ und # bestehen (ohne Leerzeichen dazwischen !). Die Zeichen $ und # sind dabei synonym zur Kennzeichnung des momentanen Befehlszählerstandes verwendbar.

Hexadezimalzahlen müssen mit einer Ziffer (0 bis 9) beginnen und mit der Kennzeichnung H enden (z. B. 200H, 3FFFH, 0E000H).

Binärzahlen werden ab ASM 2.1 unterstützt und müssen mit der Kennzeichnung B enden (z. B. 1101B, 10110001B).

Zeichenkettenkonstanten sind in Hochkommas (Apostroph) einzuschließen. Sie stehen für den ASCII-Wert (gemäß Tabelle 30 auf Seite 203) der jeweiligen Zeichen (z. B. 'A' steht für 41H, 'NO' für 4E4FH). Tritt bei der Berechnung der Ausdrücke durch den Assembler ein arithmetischer Überlauf auf, so wird dieser nicht berücksichtigt und geht verloren (z. B.: 0E000H+32768 ergibt 6000H).

Zur Verbindung von Ausdrücken in Operanden können folgende Rechenzeichen verwendet werden. Die Reihenfolge der Abarbeitung ist dabei stets von links nach rechts ohne Berücksichtigung von mathematischen Regeln! Die Verwendung von Klammern ist nicht vorgesehen.

Tabelle 58: Rechenoperationen des Assemblers

Rechenzeichen	Bedeutung
+	Addition
-	Subtraktion
*	Multiplikation
/	Ganzzahlige Division ohne Rest
%	Rest bei ganzzahliger Division (Modulo-Division)

Kommentare

Kommentare dienen zu Dokumentationszwecken und zur Erhöhung der Übersichtlichkeit der Quellprogramme. Sie sind kein funktioneller Bestandteil des Programms und werden beim Assembliervorgang übersprungen.

Ein Kommentar darf in jeder Spalte der Programmzeile beginnen und er endet mit dem Zeilenende. Das erste Zeichen eines jeden Kommentars muss ein Semikolon (;) sein.

4.2. Assembler ASM

Besonderheiten

Der Assembler ASM verarbeitet die ZILOG-Z80-Mnemonik mit einigen Besonderheiten. Zum Beispiel kann eine durch HL adressierte Speicherzelle sowohl mit (HL) als auch mit M benannt werden. Außerdem sind zu einigen Befehlen mehrere Syntax-Formen erlaubt.

Tabelle 59: Syntax-Alternativen des Assemblers

Normale Syntax		**Alternative Schreibweise(n)**	
EX	AF,AF'	EX	AF
ADD	A,r	ADD	r
ADC	A,r	ADC	r
SUB	A,r	SUB	r
SBC	r	SBC	A,r
AND	r	AND	A,r
OR	r	OR	A,r
XOR	r	XOR	A,r
CP	r	CP	A,r
ADD	HL,dd	ADD	dd
ADC	HL,dd	ADC	dd *33
SBC	HL,dd	SBC	dd *33
IN	A,(n)	IN	A,n
		IN	(n)
		IN	n
OUT	(n),A	OUT	n,A
		OUT	(n)
		OUT	n

*33 Diese Kurzform funktioniert seit ASM 2.1 fehlerfrei, vorherige Assemblerversionen übersetzten sie mit falschem Code

4.2.8. Pseudoanweisungen

Pseudooperationen sind Anweisungen an den Assembler zur Steuerung der Übersetzung des Quelltextes. Es gibt daher zu Pseudoanweisungen keinen Maschinencode. Sie sind aber wie ausführbare Anweisungen aufgebaut; können also mit einer Marke versehen und mit einem Kommentar beendet werden.

Normalerweise beginnt ein Assemblerprogramm mit einer ORG-Pseudoanweisung. Sie legt fest, auf welche Adresse der Beginn des Maschinenprogramms gelegt wird. Wird sie weggelassen, so legt der Assembler den Anfang standardmäßig auf die für die Speicherablage des Maschinenprogramms beim Start des Assemblers reservierte Speicheradresse.

Für Ausdruck kann im Operandenfeld eine Marke, eine hexadezimale Zahl oder eine Verknüpfung stehen. Der Assembler verarbeitet folgende Pseudoanweisungen:

ORG Ausdruck	Setzt den Adresszähler auf den Wert des Ausdrucks. Üblicherweise wird damit der Speicherbeginn eines Maschinenprogramms definiert. Die Anweisung kann aber auch benutzt werden, um im Programm freie Speicherplätze zu reservieren.
ORG $+Ausdruck	Reserviert die durch den Ausdruck festgelegte Anzahl von Bytes.
ORG Ausdruck1 **ORG** Ausdruck2	Beim Übersetzen mit der Assembler-Option 'O' wird der Code direkt an die angegebenen Adressen in den Speicher geschrieben. Beim Übersetzen mit der Assembler-Option 'S' wird ein fortlaufender Datenstrom ausgegeben, beginnend ab der ersten ORG-Adresse. Jede weitere ORG-Anweisung erzeugt Code für diese Adresse, geladen wird dieser jedoch unmittelbar nach dem vorherigen Programmcode.
Marke **EQU** Ausdruck	Weist der Marke den Wert des Ausdrucks zu. Damit kann im Assemblerprogramm mit symbolischen Bezeichnungen anstelle von Konstanten gearbeitet werden.
DEFB Ausdruck	"Definiere Byte" - legt den durch den Ausdruck festgelegten Byte-Wert auf die nächste Speicherstelle. DEFB erzeugt genau ein Byte. Bytes können auch als "1-Byte-Zeichenkette" angegeben (z. B: DEFB 'X') und mit Rechenzeichen verbunden werden (z. B: DEFB 'y'+80H). Das ist bei DB nicht möglich und führt zur Fehlermeldung "8".

4.2. Assembler ASM

DB Ausdruck, …

Als Operanden können Bytes und Zeichenketten gemischt und mit Komma voneinander getrennt angegeben werden. Durch die Verarbeitung von Zeichenketten in DB kann dieser Befehl auch anstelle von DEFM Anwendung finden.

DEFW Ausdruck, …

"Definiere Wort" - legt den durch den Ausdruck festgelegten 2-Byte-Wert (Wort) auf die nächsten beiden Speicherstellen. Dabei wird zunächst das niederwertige Byte und danach das höherwertige Byte abgelegt. Durch Komma getrennt können mehrere Worte in einer Zeile erzeugt werden.

DW Ausdruck, …

Gleichbedeutend wie DEFW (Kurzform).

DEFM 'Text'

Legt die ASCII-Werte der durch 'Text' definierten Zeichenkette im Speicher ab. Die Zeichenkette muss in Hochkommas eingeschlossen sein.

DEFM 'HALLO' legt die z. B. die Bytefolge 48H, 41H, 4CH, 4CH, 4FH in den Speicher.

DEFS x,y

"x" gibt an, wie viele gleiche Bytes erzeugt werden sollen. Wird "y" weggelassen, so werden Nullbytes, ansonsten Bytes mit dem Wert "y" erzeugt.

DS x,y

Gleichbedeutend wie DEFS (Kurzform).

END

Bewirkt die Beendigung der Übersetzung ab dieser Stelle. Nachfolgender Text wird nicht mehr beachtet. Der "END"-Befehl wurde aus Kompatibilitätsgründen zu anderen Assemblern eingeführt. END ist z. B. beim Testen von Programmen einsetzbar, wenn noch nicht bis zum Ende übersetzt werden soll.

4.2.9. Fehlererkennung

Fehlerhafte Quellprogrammzeilen werden beim Assemblieren in der ersten Spalte mit einer Fehlernummer (gefolgt von einem "*") versehen und über das Ausgabegerät (Bildschirm oder Drucker) ausgedruckt.

Es sei darauf hingewiesen, dass vom Assembler keine vollständige Analyse aller möglichen Fehlerquellen durchgeführt wird. Obwohl die Fehlererkennung des Assemblers gegenüber EDAS 1.4 bereits erweitert wurde, kann jedoch ASM nicht alle auftretenden Fälle und Kombinationen zurückweisen.

Tabelle 60: Assembler-Fehlercodes

Anzeige	Bedeutung
"1"	Es fehlt ein Semikolon in der Kommentarzeile
"2"	Mehrfach definierte Marke im Operand benutzt
"3"	Marke nicht definiert, auch bei relativen Sprüngen an nicht definierte Marken
"4"	Falsche Mnemonik (Befehl unbekannt)
"5"	Falsches Zahlenformat
"6"	Operandenfehler bei JR, IX oder IY - außerhalb des Bereiches -128 … +127 - IM 3 und größer - Bit-Befehle mit Bitnummer > 7
"7"	Keine Marke in EQU-Anweisung
"8"	Hochkomma fehlt in Zeichenkette
"9"	Operandenfehler bei EX- oder LD-Befehl (Register falsch benutzt)
"A"	Falsche Flag-Bedingung (zulässig sind Z, NZ, C, NC, PE, PO, M, P)
"B"	Plus, Minus oder Komma im Operanden fehlt
"C"	Division durch Null
"D"	in den Zeilen, wo eine mehrfach definierte Marke definiert wurde
"E"	Indexregister falsch verwendet
"M"	Adresse unzulässig bei Option 'O' (überschreibt Quelltext oder System-Bereiche)

4.2.10. U880/Z80-Befehlsliste

Die in diesem Kapitel dargestellte Befehlsliste wurde aus [15] übernommen und für den CAOS-Assembler angepasst bzw. ergänzt.

Zeichenerläuterung:

r,r'	8-Bit-Register	B=000 C=001 D=010 E=011 H=100 L=101 A=111
x,x'	8-Bit-Register mit Indexregistern	B=000 C=001 D=010 E=011 HX=100 LX=101 A=111
y,y'		B=000 C=001 D=010 E=011 HY=100 LY=101 A=111
X Y	8-Bit-Hälfte der Indexregister IX, IY	HX=100 LX=101 HY=100 LY=101
dd, qq, pp, rr	16-Bit-Registerpaar	dd BC=00 DE=01 HL=10 SP=11 qq BC=00 DE=01 HL=10 AF=11 pp BC=00 DE=01 IX=10 SP=11 rr BC=00 DE=01 IY=10 SP=11
t		r, (HL), (IX+d), (IY+d), HX, LX, HY, LY
n	8-Bit-Binärzahl	
nn	16-Bit-Binärzahl	
d	8-Bit-Index	(-128 bis +127)
e	8-Bit-Distanz	(-126 bis +129)
b	Bitposition	D0=000 bis D7=111
cc	Sprungbedingung	NZ=000 (non zero) Z=001 (zero) NC=010 (non carry) C=011 (carry) PO=100 (odd) PE=101 (even) P=110 (plus) M=111 (minus)
p	Restartadresse	p=00H p=08H p=10H p=18H p=20H p=28H p=30H p=38H
Flagbitbeeinflussung		. unverändert X unbestimmt ? entsprechend Ergebnis gesetzt V Overflowfunktion des Paritätsflags P Parityfunktion des Paritätsflags I Inhalt des IFF2 bei NMI
T		Anzahl Taktzyklen

(HL) Inhalt des durch HL adressierten Speicherplatzes, statt (HL) kann auch die Kurzform M im Quelltext benutzt werden.

rot In roter Schrift zusätzliche „undokumentierte“ CPU-Befehle, die vom Assembler unterstützt werden

grün In grüner Schrift Code-Bestandteile, die für ähnliche Befehle durch andere Bitkombinationen zu ersetzen sind

Tabelle 61: U880/Z80-Befehlsliste

Befehl	T	Code	Erläuterung	C Z P S N H
8-Bit-Ladebefehle				
LD r,r'	4	01 r r'	r:=r'	. . . . .
LD r,(HL)	7	01 r 110	r:=(HL)	. . . . .
LD (HL),r	7	01 110 r	(HL):=r	. . . . .
LD r,n	7	00 r 110 – n –	r:=n	. . . . .
LD (HL),n	10	00 110 110 – n –	(HL):=n	. . . . .
LD x,x'	8	11 011 101 01 x x'	x:=x'	
LD y,y'	8	11 111 101 01 y y'	y:=y'	. . . . .
LD x,n	11	11 011 101 00 x 110 – n –	x:=n	. . . . .
LD y,n	11	11 111 101 00 y 110 – n –	y:=n	. . . . .
LD r,(IX+d)	19	11 011 101 01 r 110 – d –	r:=(IX+d)	. . . . .
LD r,(IY+d)	19	11 111 101 01 r 110 – d –	r:=(IY+d)	. . . . .
LD (IX+d),r	19	11 011 101 01 110 r – d –	(IX+d):=r	. . . . .
LD (IY+d),r	19	11 111 101 01 110 r – d –	(IY+d):=r	. . . . .
LD (IX+d),n	19	11 011 101 00 110 110 – d – – n –	(IX+d):=n	. . . . .
LD (IY+d),n	19	11 111 101 00 110 110 – d – – n –	(IX+d):=n	. . . . .

4.2. Assembler ASM

Befehl	T	Code	Erläuterung	C Z P S N H
LD A,(BC)	7	00 001 010	A:=(BC)	. . . . . .
LD A,(DE)	7	00 011 010	A:=(DE)	. . . . . .
LD A,(nn)	13	00 111 010 – n – – n –	A:=(nn)	. . . . . .
LD (BC),A	7	00 000 010	(BC):=A	. . . . . .
LD (DE),A	7	00 010 010	(DE):=A	. . . . . .
LD (nn),A	13	00 110 010 – n – – n –	(nn):=A	. . . . . .
LD A,I	9	11 101 101 01 010 111	A:=I	. ? I ? . .
LD A,R	9	11 101 101 01 011 111	A:=R	. ? I ? . .
LD I,A	9	11 101 101 01 000 111	I:=A	. . . . . .
LD R,A	9	11 101 101 01 001 111	R:=A	. . . . . .
16-Bit-Ladebefehle				C Z P S N H
LD dd,nn	10	00 dd0 001 – n – – n –	dd:=nn	. . . . . .
LD IX,nn	14	11 011 101 00 100 001 – n – – n –	IX:=nn	. . . . . .
LD IY,nn	14	11 111 101 00 100 001 – n – – n –	IY:=nn	. . . . . .

4.2. Assembler ASM

Befehl	T	Code	Erläuterung	C Z P S N H
LD HL,(nn)	16	00 101 010 – n – – n –	L:=(nn) H:=(nn+1)	. . . . . .
LD dd,(nn)	20	11 101 101 01 dd1 011 – n – – n –	ddL:=(nn) ddH:=(nn+1)	. . . . . .
LD IX,(nn)	20	11 011 101 00 101 010 – n – – n –	IXL:=(nn) IXH:=(nn+1)	. . . . . .
LD IY,(nn)	20	11 111 101 00 101 010 – n – – n –	IYL:=(nn) IYH:=(nn+1)	. . . . . .
LD (nn),HL	16	00 100 010 – n – – n –	(nn):=L (nn+1):=H	. . . . . .
LD (nn),dd	20	11 101 101 01 dd0 011 – n – – n –	(nn):=ddL (nn+1):=ddH	. . . . . .
LD (nn),IX	20	11 011 101 00 100 010 – n – – n –	(nn):=IXL (nn+1):=IXH	. . . . . .
LD (nn),IY	20	11 111 101 00 100 010 – n – – n –	(nn):=IYL (nn+1):=IYH	. . . . . .
Stack- und Stackpointerbefehle				C Z P S N H
LD SP,HL	6	11 111 001	SP:=HL	. . . . . .
LD SP,IX	10	11 011 101 11 111 001	SP:=IX	. . . . . .
LD SP,IY	10	11 111 101 11 111 001	SP:=IY	. . . . . .

Befehl	T	Code	Erläuterung	C Z P S N H
PUSH qq	11	11 qq0 101	(SP-2):=qqL (SP-1):=qqH	. . . . . .
PUSH IX	15	11 011 101 11 100 101	(SP-2):=IXL (SP-1):=IXH	. . . . . .
PUSH IY	15	11 111 101 11 100 101	(SP-2):=IYL (SP-1):=IYH	. . . . . .
POP qq	10	11 qq0 001	qqL:=(SP) qqH:=(SP+1)	. . . . . .
POP IX	14	11 011 101 11 100 001	IXL:=(SP) IXH:=(SP+1)	. . . . . .
POP IY	14	11 111 101 11 100 001	IYL:=(SP) IYH:=(SP+1)	. . . . . .
Registeraustauschbefehle				C Z P S N H
EX DE,HL	4	11 101 011	DE:==:HL	. . . . . .
EX AF,AF'	4	00 001 000	AF:==:AF'	. . . . . .
EXX	4	11 011 001	BC:==:BC' DE:==:DE' HL:==:HL'	. . . . . .
EX (SP),HL	19	11 100 011	L:==:(SP) H:==:(SP+1)	. . . . . .
EX (SP),IX	23	11 011 101 11 100 011	IXL:==:(SP) IXH:==:(SP+1)	. . . . . .
EX (SP),IY	23	11 111 101 11 100 011	IYL:==:(SP) IYH:==:(SP+1)	. . . . . .
Blocktransfer- und Blocksuchbefehle				C Z P S N H
LDI	16	11 101 101 10 100 000	(DE):=(HL) dann DE:=DE+1 HL:=HL+1 BC:=BC-1	. . ? . 0 0 BC-1=0 P=0 BC-1≠0 P=1
LDIR	21 (16)	11 101 101 10 110 000	(DE):=(HL) dann DE:=DE+1 HL:=HL+1 BC:=BC-1 bis BC=0	. . 0 . 0 0
LDD	16	11 101 101 10 101 000	(DE):=(HL) dann DE:=DE-1 HL:=HL-1 BC:=BC-1	. . ? . 0 0 BC-1=0 P=0 BC-1≠0 P=1
LDDR	21 (16)	11 101 101 10 111 000	(DE):=(HL) dann DE:=DE-1 HL:=HL-1 BC:=BC-1 bis BC=0	. . 0 . 0 0

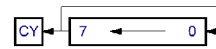
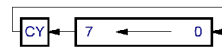
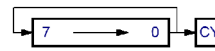
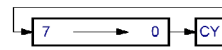
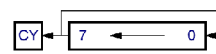
4.2. Assembler ASM

Befehl	T	Code	Erläuterung	C Z P S N H
CPI	16	11 101 101 10 100 001	A-(HL)=? dann HL:=HL+1 BC:=BC-1	. ? ? ? 1 ? A=(HL) Z=1 A≠(HL) Z=0 BC-1=0 P=0 BC-1≠0 P=1
CPIR	21 (16)	11 101 101 10 110 001	A-(HL)=? dann HL:=HL+1 BC:=BC-1 bis BC=0 oder A=(HL)	. ? ? ? 1 ? A=(HL) Z=1 A≠(HL) Z=0 BC-1=0 P=0 BC-1≠0 P=1
CPD	16	11 101 101 10 101 001	A-(HL)=? dann HL:=HL-1 BC:=BC-1	. ? ? ? 1 ? A=(HL) Z=1 A≠(HL) Z=0 BC-1=0 P=0 BC-1≠0 P=1
CPDR	21 (16)	11 101 101 10 111 001	A-(HL)=? dann HL:=HL-1 BC:=BC-1 bis BC=0 oder A=(HL)	. ? ? ? 1 ? A=(HL) Z=1 A≠(HL) Z=0 BC-1=0 P=0 BC-1≠0 P=1
8-Bit-Arithmetik- und Logikbefehle				C Z P S N H
ADD A,r	4	10 000 r	A:=A+r	? ? V ? 0 ?
ADD A,n	7	11 000 110 - n -	A:=A+n	? ? V ? 0 ?
ADD A,(HL)	7	10 000 110	A:=A+(HL)	? ? V ? 0 ?
ADD A,(IX+d)	19	11 011 101 10 000 110 - d -	A:=A+(IX+d)	? ? V ? 0 ?
ADD A,(IY+d)	19	11 111 101 10 000 110 - d -	A:=A+(IY+d)	? ? V ? 0 ?
ADD A,X	8	11 011 x 10 000 110	A:=A+X	? ? V ? 0 ?
ADD A,Y	8	11 111 y 10 000 110	A:=A+Y	? ? V ? 0 ?

4.2. Assembler ASM

Befehl	T	Code	Erläuterung	C Z P S N H
ADC A,s		001	A:=A+s+CY	? ? V ? 0 ?
SUB A,s		010	A:=A-s	? ? V ? 1 ?
SBC A,s		011	A:=A-s-CY	? ? V ? 1 ?
AND s		100	A:=A AND s	0 ? P ? 0 1
OR s		110	A:=A OR s	0 ? P ? 0 0
XOR s		101	A:=A XOR s	0 ? P ? 0 0
CP s		111	A-s=?	? ? V ? 1 ?
INC r	4	00 r 100	r:=r+1	. ? V ? 0 ?
INC (HL)	11	00 110 100	(HL):=(HL)+1	. ? V ? 0 ?
INC (IX+d)	23	11 011 101 00 110 100 - d -	(IX+d):=(IX+d)+1	. ? V ? 0 ?
INC (IY+d)	23	11 111 101 00 110 100 - d -	(IY+d):=(IY+d)+1	. ? V ? 0 ?
INC HX	8	11 011 101 00 100 100	IXH:=IXH+1	. ? V ? 0 ?
INC HY	8	11 111 101 00 100 100	IYH:=IYH+1	. ? V ? 0 ?
INC LX	8	11 011 101 00 101 100	IXL:=IXL+1	. ? V ? 0 ?
INC LY	8	11 111 101 00 101 100	IYL:=IYL+1	. ? V ? 0 ?
DEC t		101	t:=t-1	. ? V ? 1 ?
DAA	4	00 100 111	A: BCD-Dezimalkorrektur	? ? P ? . ?
CPL	4	00 101 111	A:=A negiert	. . . . 1 1
NEG	8	11 101 101 01 000 100	A:=-A	? ? V ? 1 ?
CCF	4	00 111 111	CY-Flag negiert	? . . . 0 X
SCF	4	00 110 111	CY-Flag setzen	1 . . . 0 X
16-Bit-Arithmetikbefehle				C Z P S N H
ADD HL,dd	11	00 dd1 001	HL:=HL+dd	? . . . 0 X
ADC HL,dd	15	11 101 101 01 dd1 010	HL:=HL+dd+CY	? ? V ? 0 X
SBC HL,dd	15	11 101 101 01 dd0 010	HL:=HL-dd-CY	? ? V ? 1 X
ADD IX,pp	15	11 011 101 00 pp1 001	IX:=IX+dd	? . . . 0 X
ADD IY,rr	15	11 111 101 00 rr1 001	IY:=IY+dd	? . . . 0 X

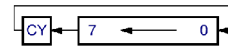
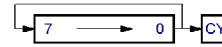
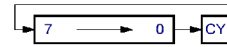
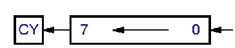
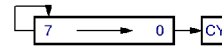
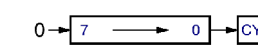
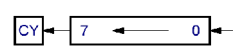
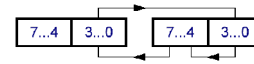
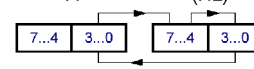
4.2. Assembler ASM

Befehl	T	Code	Erläuterung	C Z P S N H
INC dd	6	00 dd0 011	dd:=dd+1	. . . . . .
INC IX	10	11 011 101 00 100 011	IX:=IX+1	. . . . . .
INC IY	10	11 111 101 00 100 011	IY:=IY+1	. . . . . .
DEC dd	6	00 dd1 011	dd:=dd-1	. . . . . .
DEC IX	10	11 011 101 00 101 011	IX:=IX-1	. . . . . .
DEC IY	10	11 111 101 00 101 011	IY:=IY-1	. . . . . .
Rotations- und Verschiebebefehle				C Z P S N H
RLCA	4	00 000 111		? . . . 0 0
RLA	4	00 010 111		? . . . 0 0
RRCA	4	00 001 111		? . . . 0 0
RRA	4	00 011 111		? . . . 0 0
RLC r	8	11 001 011 00 000 r		? ? P ? 0 0
RLC (HL)	15	11 001 011 00 000 110		? ? P ? 0 0
RLC (IX+d)	23	11 011 101 11 001 011 - d - 00 000 110		? ? P ? 0 0
RLC (IY+d)	23	11 111 101 11 001 011 - d - 00 000 110		? ? P ? 0 0

RLC (IX+d),r	23	11 011 101
		11 001 011
		- d -
		00 000 r
			wie:
			LD r,(IX+d)
			RLC r
			LD (IX+d),r
				? ? P ? 0 0

RLC (IY+d),r	23	11 111 101
		11 001 011
		- d -
		00 000 r
			wie:
			LD r,(IY+d)
			RLC r
			LD (IY+d),r
				? ? P ? 0 0

4.2. Assembler ASM

Befehl	T	Code	Erläuterung	C Z P S N H
RL t		010		? ? P ? 0 0
RRC t		001		? ? P ? 0 0
RR t		011		? ? P ? 0 0
SLA t		100		? ? P ? 0 0
SRA t		101		? ? P ? 0 0
SRL t		111		? ? P ? 0 0
SLS t		110		? ? P ? 0 0
RLD	18	11 101 101 01 101 111		. ? P ? 0 0
RRD	18	11 101 101 01 100 111		. ? P ? 0 0
Bitbefehle				C Z P S N H
BIT b,r	8	11 001 011 01 b r	Z:=Bit(r) negiert	. ? X X 0 1
BIT b,(HL)	12	11 001 011 01 b 110	Z:=Bit(HL) negiert	. ? X X 0 1
BIT b,(IX+d)	20	11 011 101 11 001 011 - d - 01 b 110	Z:=Bit(IX+d) negiert	. ? X X 0 1
BIT b,(IY+d)	20	11 111 101 11 001 011 - d - 01 b 110	Z:=Bit(IY+d) negiert	. ? X X 0 1

4.2. Assembler ASM

Befehl	T	Code	Erläuterung	C Z P S N H
SET b,r	8	11 001 011 11 b r	Bit(r):=1	. . . . .
SET b,(HL)	15	11 001 011 11 b 110	Bit(HL):=1	. . . . .
SET b,(IX+d)	23	11 011 101 11 001 011 - d - 11 b 110	Bit(IX+d):=1	. . . . .
SET b,(IY+d)	23	11 111 101 11 001 011 - d - 11 b 110	Bit(IY+d):=1	. . . . .

SET b(IX+d),r	23	11 011 101
		11 001 011
		- d -
		11 b r
			wie:
			LD r,(IX+d)
			SET b,r
			LD (IX+d),r
				. . . . .

SET b(IY+d),r	23	11 111 101
		11 001 011
		- d -
		11 b r
			wie:
			LD r,(IY+d)
			SET b,r
			LD (IY+d),r
				. . . . .

RES b,r	8	11 001 011 10 b r	Bit(r):=0	. . . . .
RES b,(HL)	15	11 001 011 10 b 110	Bit(HL):=0	. . . . .
RES b,(IX+d)	23	11 011 101 11 001 011 - d - 10 b 110	Bit(IX+d):=0	. . . . .
RES b,(IY+d)	23	11 111 101 11 001 011 - d - 10 b 110	Bit(IY+d):=0	. . . . .

RES b(IX+d),r	23	11 011 101
		11 001 011
		- d -
		10 b r
			wie:
			LD r,(IX+d)
			RES b,r
			LD (IX+d),r
				. . . . .

RES b(IY+d),r	23	11 111 101
		11 001 011
		- d -
		10 b r
			wie:
			LD r,(IY+d)
			RES b,r
			LD (IY+d),r
				. . . . .

4.2. Assembler ASM

Befehl	T	Code	Erläuterung	C Z P S N H
Sprungbefehle				
JP nn	10	11 000 011 – n – – n –	PC:=nn	. . . . .
JP cc,nn	10	11 cc 010 – n – – n –	PC:=nn wenn Bedingung erfüllt, sonst nächster Befehl	. . . . .
JR e	12	00 011 000 – e-2 –	PC:=PC+e	. . . . .
JR C,e	12 (7)	00 111 000 – e-2 –	PC:=PC+e wenn CY=1 sonst nächster Befehl	. . . . .
JR NC,e	12 (7)	00 110 000 – e-2 –	PC:=PC+e wenn CY=0 sonst nächster Befehl	. . . . .
JR Z,e	12 (7)	00 101 000 – e-2 –	PC:=PC+e wenn Z=1 sonst nächster Befehl	. . . . .
JR NZ,e	12 (7)	00 100 000 – e-2 –	PC:=PC+e wenn Z=0 sonst nächster Befehl	. . . . .
DJNZ e	13 (8)	00 010 000 – e-2 –	B:=B-1, wenn B≠0 dann PC:=PC+e	. . . . .
JP (HL)	4	11 101 001	PC:=HL	. . . . .
JP (IX)	8	11 011 101 11 101 001	PC:=IX	. . . . .
JP (IY)	8	11 111 101 11 101 001	PC:=IY	. . . . .
CALL- und RETURN-Befehle				
CALL nn	17	11 001 101 – n – – n –	(SP-1):=PCH (SP-2):=PCL PC:=nn	. . . . .
CALL cc,nn	17 (10)	11 cc 100 – n – – n –	(SP-1):=PCH (SP-2):=PCL PC:=nn wenn Bedingung erfüllt	. . . . .
RET	10	11 001 001	PCL:=(SP) PCH:=(SP+1)	. . . . .
RET cc	11 (5)	11 cc 000	PCL:=(SP) wenn Bed. PCH:=(SP+1) erfüllt	. . . . .
RETI	14	11 101 101 01 001 101	RET von INT IFF1:=IFF2	. . . . .
RETN	14	11 101 101 01 000 101	RET von NMI IFF1:=IFF2	. . . . .
RST p	11	11 a 111	(SP-1):=PCH (SP-2):=PCL PCH:=0 PCL:=p	. . . . .

4.2. Assembler ASM

Befehl	T	Code	Erläuterung	C Z P S N H
Ein- und Ausgabebefehle, siehe auch [56]				
IN A,n	11	11 011 011 – n –	A:=(n) ADR:=A/n	. . . . .
IN r,(C)	12	11 101 101 01 r 000	r:=(C) ADR:=B/C	. ? P ? 0 ?
INI	16	11 101 010 10 100 010	1. (HL):=(C) ADR:=B/C 2. B:=B-1 HL:=HL+1 B-1=0 Z:=1 B-1≠0 Z:=0	. ? X X 1 X
INIR	21 (16)	11 101 101 10 110 010	1. (HL):=(C) ADR:=B/C 2. B:=B-1 HL:=HL+1 bis B=0	. 1 X X 1 X
IND	16	11 101 101 10 101 010	1. (HL):=(C) ADR:=B/C 2. B:=B-1 HL:=HL-1 B-1=0 Z:=1 B-1≠0 Z:=0	. ? X X 1 X
INDR	21 (16)	11 101 101 10 111 010	1. (HL):=(C) ADR:=B/C 2. B:=B-1 HL:=HL-1 bis B=0	. 1 X X 1 X
INF	12	11 101 101 01 110 000	F:=? ADR:=B/C (nur Flags setzen)	. ? P ? 0 ?
OUT (n),A	11	11 010 011 – n –	(n):=A ADR:=A/n	. . . . .
OUT (C),r	12	11 101 101 01 r 001	(C):=r ADR:=B/C	. . . . .
OUTI	16	11 101 101 10 100 011	1. B:=B-1 2. (C):=(HL) ADR:=B/C 3. HL:=HL+1 B-1=0 Z:=1 B-1≠0 Z:=0	. ? X X 1 X
OTIR	21 (16)	11 101 101 10 110 011	1. B:=B-1 2. (C):=(HL) ADR:=B/C 3. HL:=HL+1 bis B=0	. 1 X X 1 X
OUTD	16	11 101 101 10 101 011	1. B:=B-1 2. (C):=(HL) ADR:=B/C 3. HL:=HL-1 bis B=0 B-1=0 Z:=1 B-1≠0 Z:=0	. ? X X 1 X
OTIR	21 (16)	11 101 101 10 111 011	1. B:=B-1 2. (C):=(HL) ADR:=B/C 3. HL:=HL-1 bis B=0	. 1 X X 1 X
OTCL	12	11 101 101 01 110 001	(C):=0 ADR:=B/C (Ausgabe des Wertes 0)	. . . . .
CPU-Steuerbefehle				C Z P S N H
NOP	4	00 000 000	No Operationen	. . . . .
HALT	4	01 110 110	HALT	. . . . .
DI	4	11 110 011	Interrupt AUS	. . . . .
EI	4	11 111 011	Interrupt EIN	. . . . .

Befehl	T	Code	Erläuterung	C Z P S N H
IM 0	8	11 101 101 01 000 110	Interrupt Mode 0	. . . . . .
IM 1	8	11 101 101 01 010 110	Interrupt Mode 1	. . . . . .
IM 2	8	11 101 101 01 011 110	Interrupt Mode 2	. . . . . .

4.2.11. Arbeitszellen von Assembler, Editor und Reassembler

Die Lage der Arbeitszellen von ASM ab Version 2.0 weicht von den bekannten Adressen aus EDAS 1.4 bis EDAS 1.6 ab. Die seit CAOS 4.7 integrierte Software Assembler, Editor und Reassembler nutzt die gleichen Arbeitszellen. Deshalb ist ein Wechsel zwischen diesen Programmen problemlos möglich.

Beim Start des Assemblers lassen sich die zu benutzenden RAM-Bereiche einstellen. Der Assembler kann auch mit dem RAM-8 sowie einem RAM-Modul auf Adresse C000H arbeiten, wobei der IRM und der USER-ROM für den Zugriff automatisch ausgeblendet werden. Die beim Start vorgeschlagenen Werte entsprechen dem maximal nutzbaren Adressbereich. Im Adressbereich 0000H-00FFH des RAM0 und im IRM ab BFF0H liegen Arbeitszellen der drei Programme, hier eine Auswahl.

Tabelle 62: Speichernutzung Assembler/Editor/Reassembler

Name	Adr.	Länge	Bedeutung
NAME	0000H	11	Dateiname (8 Zeichen Name, 3 Zeichen Typ)
RANDL	0010H	1	Linke Randposition Editor
RANDR	0011H	1	Rechte Randposition Editor
MEML	0012H	1	Zeilennummer Editor
T1A	0018H	2	Beginn Teiltext 1 (Speicheranfang+1)
T2E	001AH	2	Ende Teiltext 2 (Speicherende-1)
T1E	001CH	2	Ende Teiltext 1
T2A	001EH	2	Beginn Teiltext 2
COL	0020H	1	Cursor-Spalte im Editor
LIN	0021H	1	Cursor-Zeile im Editor
STRING	0023H-003DH	27	Suchstring für FIND im Editor

4.2. Assembler ASM

Name	Adr.	Länge	Bedeutung
STAT	003EH	1	Merkzelle für verschiedene Einstellungen Bit 0 = 1 - Floppy vorhanden Bit 1 = 1 - DEP mit LW-Steuerung Bit 2 = 1 - Fehler bei Diskette oder MTAB-Überschreitung Bit 3 = 1 - Assembler-Option 'S' Bit 4 = 1 - Überschreiben (0=Einfügen) Bit 5 = 1 - Zeichensatz (0=IBM, 1=deutsch) Bit 6 = 1 - Zeichensatz (0=IBM, 1=CAOS) Bit 7 = 1 - END-Befehl erkannt (ASM)
OPT	003FH	1	Assembler-Optionen Bit 0 L - Listing ausgeben Bit 1 P - Umschaltung Drucker Bit 2 + - Markentabelle erweitern Bit 3 1 - nur PASS 1 Bit 4 3 - PASS 3 läuft (Save MC) Bit 5 B - Bildschirmformat (40 Spalten) Bit 6 O - MC in Speicher laden Bit 7 2 - nur PASS 2
OFFS	0040H	2	Assembler-Offset
RAMC	005AH-00A0H	69	Hilfsprogramme für Speicherzugriff
TEXT	A880H-A8FFH	128	Assemblierpuffer: erzeugte Listingzeile
NAME	B7F5H	11	Dateiname der Quelltext-Datei
IRM	BFCCH-BFFFH	52	Umschaltroutinen ASM ↔ EDIT sowie Menüworte QUIT/ASM im Editor

4.3. Disassembler

Ein Reassembler oder Disassembler ermöglicht das Rückübersetzen eines im Speicher des Computers stehenden Maschinenprogramms in die mnemonische Form, also wie ein Assemblerlisting darzustellen. Es sind zwei Varianten dieser Disassembler integriert: DISASS erzeugt einen Quelltext seitenweise am Bildschirm, durch den mit den Cursortasten geblättert werden kann. QMR ist ein Markenreassembler, der in 2 Durchläufen zunächst eine Markentabelle aufbaut und im 2. Durchlauf unter Verwendung dieser Markentabelle einen Quelltext erzeugt, der abgespeichert werden kann für eine Weiterbearbeitung.

4.3.1. CAOS-Kommando DISASS

Mit diesem CAOS-Menükommando wird der DISASS gestartet:

%DISASS Aadr [Zeilen [Ausadr [Prol]]]

Dabei bedeuten die Parameter:

Aadr	Anfangsadresse des zu disassemblierenden Programms
Zeilen	Anzahl der auf einer Bildschirmseite stehenden Zeilen
Ausadr	Ausgabeadresse am Bildschirm
Prol	Prologbytes, welche erkannt werden sollen

Wird DISASS ohne Argumente aufgerufen, dann erscheint eine Fehlermeldung mit Angabe der zu übergebenden Parameter.

Wird keine Zeilenanzahl eingegeben, dann benutzt DISASS das gesamte aktive Fenster. Nach dieser Anzahl von Zeilen wartet das Programm auf eine Tastatureingabe. Reaktionen der Tasten:

BRK: Rücksprung in das CAOS – Betriebssystem


CR: Ausgabe der nächsten nn Zeilen (Scrollen)


CUR: ein Befehl bzw. eine Zeile weiter


CUL: ein Befehl bzw. eine Zeile zurück


CUD: Bildschirm löschen und nn Zeilen weiter (blättern)


CUU: Bildschirm löschen und nn Zeilen zurück (blättern)

D: ein Byte als Datenbyte mit DB anzeigen

Durch die Angabe einer Ausgabeadresse hat man die Möglichkeit, ein Maschinenprogramm an eine andere Stelle in den Speicher zu laden und dann mit dem

4.3. Disassembler

Disassembler so anzeigen zu lassen, also ob sich das Programm auf der Adresse befinden würde, auf der es auch lauffähig ist. Das kann hilfreich sein für Programme, die auf der Adresse C000H laufen, weil sich dort auch der Disassembler selbst im Speicher befindet und kein anderes Programm auf diesem Adressbereich sehen kann.

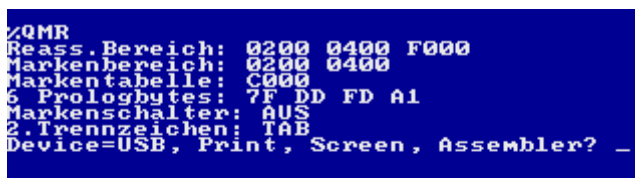
Mit dem Parameter Prol lassen sich bis zu zwei Prologbytes übergeben, die für die Erkennung von Menüworten genutzt werden. Sollen zwei verschiedene Prologbytes für die Erkennung definiert werden, so sind diese als ein Wert zu übergeben, beispielsweise als AA55, wenn nach Prologbyte AA und 55 zu suchen wäre. Wird der Parameter Prol nicht angegeben, dann werden die 6 bekanntesten Prologbytes 7FH, DDH, FDH, A1H, 3DH und 3EH benutzt. Wird Prol angegeben, dann werden die beiden letzten Prologbytes durch die Eingabewerte ersetzt.

4.3.2. CAOS-Kommando QMR

Mit dem Reassembler QMR wird aus im Speicher befindlichem Maschinencode ein Quelltext erzeugt, der auf verschiedene Arten ausgegeben werden kann. Danach kann man diesen Quelltext weiter bearbeiten.
Mit diesem CAOS-Menükommando wird der Reassembler QMR gestartet:

%QMR

Beim Aufruf des Kommandos QMR müssen keine Parameter angegeben werden. Die erforderlichen Parameter werden im Dialog abgefragt.



– Reassembler-Bereich ist der Adressbereich, in dem sich der zu übersetzende Maschinencode befindet. Eine optionale dritte Adresse (im Beispiel F000) kann angegeben werden, um dem Reassembler eine abweichende Originaladresse mitzuteilen.
– Markenbereich ist der Speicherbereich, in dem benutzte Sprungadressen als Marken erzeugt werden. Der als Reassembler-Bereich eingegebene Adressbereich wird dabei als Standardwert vorgeschlagen. Der Markenbereich sollte z. B. dann größer angegeben werden (gesamte Programmgröße), wenn nur ein Teilstück daraus reassembliert werden soll.

– Markentabelle ist die Adresse, unterhalb der sich die Markentabelle im Speicher rückwärts aufbaut. C000H wird hier als Standard vorgeschlagen und bedeutet am Ende des RAM8, also von BFFFH hin zu kleineren Adressen.
– Die 4 gängigsten Prologbytes werden für die Erkennung von Menüworten vorgeschlagen. Diese können geändert, und durch 2 weitere Prologbytes ergänzt werden.
– Der Markenschalter kann mit der Leertaste EIN bzw. AUS geschaltet werden. EIN bewirkt, dass Datenbereiche als Marken definiert werden, bei AUS werden direkte Adressen erzeugt.
– Das zweite Trennzeichen, also das Zeichen zwischen Operationscode und Operand kann wahlweise als Leerzeichen oder Tabulatorschritt erzeugt werden.
– Als letztes wird abgefragt, wo der Quelltext ausgegeben wird. Zur Auswahl stehen hier:

    D = das aktuelle Device
    P = Printer (Drucker)
    S = Screen (Bildschirm)
    A = Assembler

Wird Assembler gewählt, dann schreibt QMR den Quelltext direkt in den Speicher. Dazu wird noch die Anfangsadresse abgefragt, wo der Quelltext in den Speicher geschrieben wird (vergleiche Bild 37 Seite 377). Bei dieser Option ist besonders darauf zu achten, dass sich die einzelnen Adressbereiche (Maschinencode, Quelltext, Markentabelle) nicht überlagern. Kommt es zu keiner Überlagerung der Adressbereiche, ergibt sich eine sehr elegante Art, einen Quelltext aus einem Maschinencode zu erzeugen und sofort im Assembler zu bearbeiten. Im Zweifelsfall ist es jedoch sicherer, den Quelltext zunächst auf einen Datenträger abzuspeichern und dann in den Assembler bzw. Editor wieder einzulesen.

4.3.3. U880/Z80-Befehlsübersicht

Im Abschnitt des Assemblers ist bereits eine U880/Z80-Befehlsliste abgedruckt. Für eine Rückübersetzung, welche der Disassembler vornimmt, oder einfach nur zum Verständnis des Maschinencodes ist jedoch eine Befehlsliste hilfreich, die nach dem Befehlscode sortiert ist. Die folgenden Tabellen stellen genau diese Übersicht dar. In roter Schrift sind wieder die undokumentierten Z80-Befehlscodes dargestellt, welche sowohl vom U880 als auch vom Z80 ausgeführt werden und welche der Assembler ASM mit unterstützt.

4.3. Disassembler

Hex	Befehl	Hex	Befehl	Hex	Befehl	Hex	Befehl
00	NOP	40	LD B,B	80	ADD A,B	C0	RET NZ
01	BC,nn	41	LD B,C	81	ADD A,C	C1	POP BC
02	(BC),A	42	LD B,D	82	ADD A,D	C2	JP NZ,nn
03	INC BC	43	LD B,E	83	ADD A,E	C3	JP nn
04	INC B	44	LD B,H	84	ADD A,H	C4	CALL NZ,nn
05	DEC B	45	LD B,L	85	ADD A,L	C5	PUSH BC
06	LD B,n	46	LD B,(HL)	86	ADD A,(HL)	C6	ADD A,n
07	RLCA	47	LD B,A	87	ADD A,A	C7	RST 0H
08	EX AF,AF'	48	LD C,B	88	ADC A,B	C8	RET Z
09	ADD HL,BC	49	LD C,C	89	ADC A,C	C9	RET
0A	LD A,(BC)	4A	LD C,D	8A	ADC A,D	CA	JP Z,nn
0B	DEC BC	4B	LD C,E	8B	ADC A,E	CB	*siehe Tabelle "CB"*
0C	INC C	4C	LD C,H	8C	ADC A,H	CC	CALL Z,nn
0D	DEC C	4D	LD C,L	8D	ADC A,L	CD	CALL nn
0E	LD C,n	4E	LD C,(HL)	8E	ADC A,(HL)	CE	ADC A,n
0F	RRCA	4F	LD C,A	8F	ADC A,A	CF	RST 8H
10	DJNZ e	50	LD D,B	90	SUB A,B	D0	RET NC
11	LD DE,nn	51	LD D,C	91	SUB A,C	D1	POP DE
12	LD (DE),A	52	LD D,D	92	SUB A,D	D2	JP NC,nn
13	INC DE	53	LD D,E	93	SUB A,E	D3	OUT (n),A
14	INC D	54	LD D,H	94	SUB A,H	D4	CALL NC,nn
15	DEC D	55	LD D,L	95	SUB A,L	D5	PUSH DE
16	LD D,n	56	LD D,(HL)	96	SUB A,(HL)	D6	SUB A,n
17	RLA	57	LD D,A	97	SUB A,A	D7	RST 10H
18	JR e	58	LD E,B	98	SBC A,B	D8	RET C
19	ADD HL,DE	59	LD E,C	99	SBC A,C	D9	EXX
1A	LD A,(DE)	5A	LD E,D	9A	SBC A,D	DA	JP C,nn
1B	DEC DE	5B	LD E,E	9B	SBC A,E	DB	IN A,(n)
1C	INC E	5C	LD E,H	9C	SBC A,H	DC	CALL C,nn
1D	DEC E	5D	LD E,L	9D	SBC A,L	DD	*siehe IX-Befehle*
1E	LD E,n	5E	LD E,(HL)	9E	SBC A,(HL)	DE	SBC n
1F	RRA	5F	LD E,A	9F	SBC A,A	DF	RST 18H
20	JR NZ,e	60	LD H,B	A0	AND B	E0	RET PO
21	LD HL,nn	61	LD H,C	A1	AND C	E1	POP HL
22	LD (nn),HL	62	LD H,D	A2	AND D	E2	JP PO,nn
23	INC HL	63	LD H,E	A3	AND E	E3	EX (SP),HL
24	INC H	64	LD H,H	A4	AND H	E4	CALL PO,nn
25	DEC H	65	LD H,L	A5	AND L	E5	PUSH HL
26	LD H,n	66	LD H,(HL)	A6	AND (HL)	E6	AND n
27	DAA	67	LD H,A	A7	AND A	E7	RST 20H
28	JR Z,e	68	LD L,B	A8	XOR B	E8	RET PE
29	ADD HL,HL	69	LD L,C	A9	XOR C	E9	JP (HL)
2A	LD HL,(nn)	6A	LD L,D	AA	XOR D	EA	JP PE,nn
2B	DEC HL	6B	LD L,E	AB	XOR E	EB	EX DE,HL
2C	INC L	6C	LD L,H	AC	XOR H	EC	CALL PE,nn
2D	DEC L	6D	LD L,L	AD	XOR L	ED	*siehe Tabelle "ED"*
2E	LD L,n	6E	LD L,(HL)	AE	XOR (HL)	EE	XOR n
2F	CPL	6F	LD L,A	AF	XOR A	EF	RST 28H
30	JR NC,e	70	LD (HL),B	B0	OR B	F0	RET P
31	LD SP,nn	71	LD (HL),C	B1	OR C	F1	POP AF
32	LD (nn),A	72	LD (HL),D	B2	OR D	F2	JP P,nn
33	INC SP	73	LD (HL),E	B3	OR E	F3	DI
34	INC (HL)	74	LD (HL),H	B4	OR H	F4	CALL P,nn
35	DEC (HL)	75	LD (HL),L	B5	OR L	F5	PUSH AF
36	LD (HL),n	76	HALT	B6	OR (HL)	F6	OR n
37	SCF	77	LD (HL),A	B7	OR A	F7	RST 30H
38	JR C,e	78	LD A,B	B8	CP B	F8	RET M
39	ADD HL,SP	79	LD A,C	B9	CP C	F9	LD SP,HL
3A	LD A,(nn)	7A	LD A,D	BA	CP D	FA	JP M,nn
3B	DEC SP	7B	LD A,E	BB	CP E	FB	EI
3C	INC A	7C	LD A,H	BC	CP H	FC	CALL M,nn
3D	DEC A	7D	LD A,L	BD	CP L	FD	*siehe IY-Befehle*
3E	LD A,n	7E	LD A,(HL)	BE	CP (HL)	FE	CP n
3F	CCF	7F	LD A,A	BF	CP A	FF	RST 38H

Grundtabelle ohne Vorbyte

4.3. Disassembler

Hex	Befehl	Hex	Befehl	Hex	Befehl	Hex	Befehl
00	RLC B	40	BIT 0,B	80	RES 0,B	C0	SET 0,B
01	RLC C	41	BIT 0,C	81	RES 0,C	C1	SET 0,C
02	RLC D	42	BIT 0,D	82	RES 0,D	C2	SET 0,D
03	RLC E	43	BIT 0,E	83	RES 0,E	C3	SET 0,E
04	RLC H	44	BIT 0,H	84	RES 0,H	C4	SET 0,H
05	RLC L	45	BIT 0,L	85	RES 0,L	C5	SET 0,L
06	RLC (HL)	46	BIT 0,(HL)	86	RES 0,(HL)	C6	SET 0,(HL)
07	RLC A	47	BIT 0,A	87	RES 0,A	C7	SET 0,A
08	RRC B	48	BIT 1,B	88	RES 1,B	C8	SET 1,B
09	RRC C	49	BIT 1,C	89	RES 1,C	C9	SET 1,C
0A	RRC D	4A	BIT 1,D	8A	RES 1,D	CA	SET 1,D
0B	RRC E	4B	BIT 1,E	8B	RES 1,E	CB	SET 1,E
0C	RRC H	4C	BIT 1,H	8C	RES 1,H	CC	SET 1,H
0D	RRC L	4D	BIT 1,L	8D	RES 1,L	CD	SET 1,L
0E	RRC (HL)	4E	BIT 1,(HL)	8E	RES 1,(HL)	CE	SET 1,(HL)
0F	RRC A	4F	BIT 1,A	8F	RES 1,A	CF	SET 1,A
10	RL B	50	BIT 2,B	90	RES 2,B	D0	SET 2,B
11	RL C	51	BIT 2,C	91	RES 2,C	D1	SET 2,C
12	RL D	52	BIT 2,D	92	RES 2,D	D2	SET 2,D
13	RL E	53	BIT 2,E	93	RES 2,E	D3	SET 2,E
14	RL H	54	BIT 2,H	94	RES 2,H	D4	SET 2,H
15	RL L	55	BIT 2,L	95	RES 2,L	D5	SET 2,L
16	RL (HL)	56	BIT 2,(HL)	96	RES 2,(HL)	D6	SET 2,(HL)
17	RL A	57	BIT 2,A	97	RES 2,A	D7	SET 2,A
18	RR B	58	BIT 3,B	98	RES 3,B	D8	SET 3,B
19	RR C	59	BIT 3,C	99	RES 3,C	D9	SET 3,C
1A	RR D	5A	BIT 3,D	9A	RES 3,D	DA	SET 3,D
1B	RR E	5B	BIT 3,E	9B	RES 3,E	DB	SET 3,E
1C	RR H	5C	BIT 3,H	9C	RES 3,H	DC	SET 3,H
1D	RR L	5D	BIT 3,L	9D	RES 3,L	DD	SET 3,L
1E	RR (HL)	5E	BIT 3,(HL)	9E	RES 3,(HL)	DE	SET 3,(HL)
1F	RR A	5F	BIT 3,A	9F	RES 3,A	DF	SET 3,A
20	SLA B	60	BIT 4,B	A0	RES 4,B	E0	SET 4,B
21	SLA C	61	BIT 4,C	A1	RES 4,C	E1	SET 4,C
22	SLA D	62	BIT 4,D	A2	RES 4,D	E2	SET 4,D
23	SLA E	63	BIT 4,E	A3	RES 4,E	E3	SET 4,E
24	SLA H	64	BIT 4,H	A4	RES 4,H	E4	SET 4,H
25	SLA L	65	BIT 4,L	A5	RES 4,L	E5	SET 4,L
26	SLA (HL)	66	BIT 4,(HL)	A6	RES 4,(HL)	E6	SET 4,(HL)
27	SLA A	67	BIT 4,A	A7	RES 4,A	E7	SET 4,A
28	SRA B	68	BIT 5,B	A8	RES 5,B	E8	SET 5,B
29	SRA C	69	BIT 5,C	A9	RES 5,C	E9	SET 5,C
2A	SRA D	6A	BIT 5,D	AA	RES 5,D	EA	SET 5,D
2B	SRA E	6B	BIT 5,E	AB	RES 5,E	EB	SET 5,E
2C	SRA H	6C	BIT 5,H	AC	RES 5,H	EC	SET 5,H
2D	SRA L	6D	BIT 5,L	AD	RES 5,L	ED	SET 5,L
2E	SRA (HL)	6E	BIT 5,(HL)	AE	RES 5,(HL)	EE	SET 5,(HL)
2F	SRA A	6F	BIT 5,A	AF	RES 5,A	EF	SET 5,A
30	SLS B	70	BIT 6,B	B0	RES 6,B	F0	SET 6,B
31	SLS C	71	BIT 6,C	B1	RES 6,C	F1	SET 6,C
32	SLS D	72	BIT 6,D	B2	RES 6,D	F2	SET 6,D
33	SLS E	73	BIT 6,E	B3	RES 6,E	F3	SET 6,E
34	SLS H	74	BIT 6,H	B4	RES 6,H	F4	SET 6,H
35	SLS L	75	BIT 6,L	B5	RES 6,L	F5	SET 6,L
36	SLS (HL)	76	BIT 6,(HL)	B6	RES 6,(HL)	F6	SET 6,(HL)
37	SLS A	77	BIT 6,A	B7	RES 6,A	F7	SET 6,A
38	SRL B	78	BIT 7,B	B8	RES 7,B	F8	SET 7,B
39	SRL C	79	BIT 7,C	B9	RES 7,C	F9	SET 7,C
3A	SRL D	7A	BIT 7,D	BA	RES 7,D	FA	SET 7,D
3B	SRL E	7B	BIT 7,E	BB	RES 7,E	FB	SET 7,E
3C	SRL H	7C	BIT 7,H	BC	RES 7,H	FC	SET 7,H
3D	SRL L	7D	BIT 7,L	BD	RES 7,L	FD	SET 7,L
3E	SRL (HL)	7E	BIT 7,(HL)	BE	RES 7,(HL)	FE	SET 7,(HL)
3F	SRL A	7F	BIT 7,A	BF	RES 7,A	FF	SET 7,A

Vorbyte CB

4.3. Disassembler

Hex	Befehl	Hex	Befehl	Hex	Befehl	Hex	Befehl
00	-	40	IN B,(C)	80	-	C0	-
01	-	41	OUT (C),B	81	-	C1	-
02	-	42	SBC HL,BC	82	-	C2	-
03	-	43	LD (nn),BC	83	-	C3	-
04	-	44	NEG	84	-	C4	-
05	-	45	RETN	85	-	C5	-
06	-	46	IM 0	86	-	C6	-
07	-	47	LD I,A	87	-	C7	-
08	-	48	IN C,(C)	88	-	C8	-
09	-	49	OUT (C),C	89	-	C9	-
0A	-	4A	ADC HL,BC	8A	-	CA	-
0B	-	4B	LD BC,(nn)	8B	-	CB	-
0C	-	4C	NEG	8C	-	CC	-
0D	-	4D	RETI	8D	-	CD	-
0E	-	4E	-	8E	-	CE	-
0F	-	4F	LD R,A	8F	-	CF	-
10	-	50	IN D,(C)	90	-	D0	-
11	-	51	OUT (C),D	91	-	D1	-
12	-	52	SBC HL,DE	92	-	D2	-
13	-	53	LD (nn),DE	93	-	D3	-
14	-	54	NEG	94	-	D4	-
15	-	55	RETN	95	-	D5	-
16	-	56	IM 1	96	-	D6	-
17	-	57	LD A,I	97	-	D7	-
18	-	58	IN E,(C)	98	-	D8	-
19	-	59	OUT (C),E	99	-	D9	-
1A	-	5A	ADC HL,DE	9A	-	DA	-
1B	-	5B	LD DE,(nn)	9B	-	DB	-
1C	-	5C	NEG	9C	-	DC	-
1D	-	5D	RETN	9D	-	DD	-
1E	-	5E	IM 2	9E	-	DE	-
1F	-	5F	LD A,R	9F	-	DF	-
20	-	60	IN H,(C)	A0	LDI	E0	-
21	-	61	OUT (C),H	A1	CPI	E1	-
22	-	62	SBC HL,HL	A2	INI	E2	-
23	-	63	LD (nn),HL	A3	OUTI	E3	-
24	-	64	NEG	A4	-	E4	-
25	-	65	RETN	A5	-	E5	-
26	-	66	-	A6	-	E6	-
27	-	67	RRD	A7	-	E7	-
28	-	68	IN L,(C)	A8	LDD	E8	-
29	-	69	OUT (C),L	A9	CPD	E9	-
2A	-	6A	ADC HL,HL	AA	IND	EA	-
2B	-	6B	LD HL,(nn)	AB	OUTD	EB	-
2C	-	6C	NEG	AC	-	EC	-
2D	-	6D	RETN	AD	-	ED	-
2E	-	6E	-	AE	-	AE	-
2F	-	6F	RLD	AF	-	AF	-
30	-	70	INF	B0	LDIR	F0	-
31	-	71	OTCL	B1	CPIR	F1	-
32	-	72	SBC HL,SP	B2	INIR	F2	-
33	-	73	LD (nn),SP	B3	OTIR	F3	-
34	-	74	NEG	B4	-	F4	-
35	-	75	RETN	B5	-	F5	-
36	-	76	-	B6	-	F6	-
37	-	77	-	B7	-	F7	-
38	-	78	IN A,(C)	B8	LDDR	F8	-
39	-	79	OUT (C),A	B9	CPDR	F9	-
3A	-	7A	ADC HL,SP	BA	INDR	FA	-
3B	-	7B	LD SP,(nn)	BB	OTDR	FB	-
3C	-	7C	NEG	BC	-	FC	-
3D	-	7D	RETN	BD	-	FD	-
3E	-	7E	-	BE	-	FE	-
3F	-	7F	-	BF	-	FF	-

Vorbyte ED

4.3. Disassembler

Hex	Befehl	Hex	Befehl	Hex	Befehl	Hex	Befehl
00	-	40	-	80	-	C0	-
01	-	41	-	81	-	C1	-
02	-	42	-	82	-	C2	-
03	-	43	-	83	-	C3	-
04	-	44	LD B,Hi	84	ADD A,Hi	C4	-
05	-	45	LD B,Li	85	ADD A,Li	C5	-
06	-	46	LD B,(ii+d)	86	ADD A,(ii+d)	C6	-
07	-	47	-	87	-	C7	-
08	-	48	-	88	-	C8	-
09	ADD ii,BC	49	-	89	-	C9	-
0A	-	4A	-	8A	-	CA	-
0B	-	4B	-	8B	-	CB	*siehe Tabelle ii+CB*
0C	-	4C	LD C,Hi	8C	ADC A,Hi	CC	-
0D	-	4D	LD C,Li	8D	ADC A,Li	CD	-
0E	-	4E	LD C,(ii+d)	8E	ADC A,(ii+d)	CE	-
0F	-	4F	-	8F	-	CF	-
10	-	50	-	90	-	D0	-
11	-	51	-	91	-	D1	-
12	-	52	-	92	-	D2	-
13	-	53	-	93	-	D3	-
14	-	54	LD D,Hi	94	SUB A,Hi	D4	-
15	-	55	LD D,Li	95	SUB A,Li	D5	-
16	-	56	LD D,(ii+d)	96	SUB A,(ii+d)	D6	-
17	-	57	-	97	-	D7	-
18	-	58	-	98	-	D8	-
19	ADD ii,DE	59	-	99	-	D9	-
1A	-	5A	-	9A	-	DA	-
1B	-	5B	-	9B	-	DB	-
1C	-	5C	LD E,Hi	9C	SBC A,Hi	DC	-
1D	-	5D	LD E,Li	9D	SBC A,Li	DD	-
1E	-	5E	LD E,(ii+d)	9E	SBC A,(ii+d)	DE	-
1F	-	5F	-	9F	-	DF	-
20	-	60	LD Hi,B	A0	-	E0	-
21	LD ii,nn	61	LD Hi,C	A1	-	E1	POP ii
22	LD (nn),ii	62	LD Hi,D	A2	-	E2	-
23	INC ii	63	LD Hi,E	A3	-	E3	EX (SP),ii
24	INC Hi	64	LD Hi,Hi	A4	AND Hi	E4	-
25	DEC Hi	65	LD Hi,Li	A5	AND Li	E5	PUSH ii
26	LD Hi,n	66	LD H,(ii+d)	A6	AND (ii+d)	E6	-
27	-	67	LD Hi,A	A7	-	E7	-
28	-	68	LD Li,B	A8	-	E8	-
29	ADD ii,ii	69	LD Li,C	A9	-	E9	JP (ii)
2A	LD ii,(nn)	6A	LD Li,D	AA	-	EA	-
2B	DEC ii	6B	LD Li,E	AB	-	EB	-
2C	INC Li	6C	LD Li,Hi	AC	XOR Hi	EC	-
2D	DEC Li	6D	LD Li,Li	AD	XOR Li	ED	-
2E	LD Li,n	6E	LD L,(ii+d)	AE	XOR (ii+d)	AE	-
2F	-	6F	LD Li,A	AF	-	AF	-
30	-	70	LD (ii+d),B	B0	-	F0	-
31	-	71	LD (ii+d),C	B1	-	F1	-
32	-	72	LD (ii+d),D	B2	-	F2	-
33	-	73	LD (ii+d),E	B3	-	F3	-
34	INC (ii+d)	74	LD (ii+d),H	B4	OR Hi	F4	-
35	DEC (ii+d)	75	LD (ii+d),L	B5	OR Li	F5	-
36	LD (ii+d),n	76	-	B6	OR (ii+d)	F6	-
37	-	77	LD (ii+d),A	B7	-	F7	-
38	-	78	-	B8	-	F8	-
39	ADD ii,SP	79	-	B9	-	F9	LD SP,ii
3A	-	7A	-	BA	-	FA	-
3B	-	7B	-	BB	-	FB	-
3C	-	7C	LD A,Hi	BC	CP Hi	FC	-
3D	-	7D	LD A,Li	BD	CP Li	FD	-
3E	-	7E	LD A,(ii+d)	BE	CP (ii+d)	FE	-
3F	-	7F	-	BF	-	FF	-

Vorbyte DD (ii=IX, i=X) oder Vorbyte FD (ii=IY, i=Y)

4.3. Disassembler

Hex	Befehl	Hex	Befehl	Hex	Befehl	Hex	Befehl
00	RLC (ii+d),B	40	-	80	RES 0,(ii+d),B	C0	SET 0,(ii+d),B
01	RLC (ii+d),C	41	-	81	RES 0,(ii+d),C	C1	SET 0,(ii+d),C
02	RLC (ii+d),D	42	-	82	RES 0,(ii+d),D	C2	SET 0,(ii+d),D
03	RLC (ii+d),E	43	-	83	RES 0,(ii+d),E	C3	SET 0,(ii+d),E
04	RLC (ii+d),H	44	-	84	RES 0,(ii+d),H	C4	SET 0,(ii+d),H
05	RLC (ii+d),L	45	-	85	RES 0,(ii+d),L	C5	SET 0,(ii+d),L
06	RLC (ii+d)	46	BIT 0,(ii+d)	86	RES 0,(ii+d)	C6	SET 0,(ii+d)
07	RLC (ii+d),A	47	-	87	RES 0,(ii+d),A	C7	SET 0,(ii+d),A
08	RRC (ii+d),B	48	-	88	RES 1,(ii+d),B	C8	SET 1,(ii+d),B
09	RRC (ii+d),C	49	-	89	RES 1,(ii+d),C	C9	SET 1,(ii+d),C
0A	RRC (ii+d),D	4A	-	8A	RES 1,(ii+d),D	CA	SET 1,(ii+d),D
0B	RRC (ii+d),E	4B	-	8B	RES 1,(ii+d),E	CB	SET 1,(ii+d),E
0C	RRC (ii+d),H	4C	-	8C	RES 1,(ii+d),H	CC	SET 1,(ii+d),H
0D	RRC (ii+d),L	4D	-	8D	RES 1,(ii+d),L	CD	SET 1,(ii+d),L
0E	RRC (ii+d)	4E	BIT 1,(ii+d)	8E	RES 1,(ii+d)	CE	SET 1,(ii+d)
0F	RRC (ii+d),A	4F	-	8F	RES 1,(ii+d),A	CF	SET 1,(ii+d),A
10	RL (ii+d),B	50	-	90	RES 2,(ii+d),B	D0	SET 2,(ii+d),B
11	RL (ii+d),C	51	-	91	RES 2,(ii+d),C	D1	SET 2,(ii+d),C
12	RL (ii+d),D	52	-	92	RES 2,(ii+d),D	D2	SET 2,(ii+d),D
13	RL (ii+d),E	53	-	93	RES 2,(ii+d),E	D3	SET 2,(ii+d),E
14	RL (ii+d),H	54	-	94	RES 2,(ii+d),H	D4	SET 2,(ii+d),H
15	RL (ii+d),L	55	-	95	RES 2,(ii+d),L	D5	SET 2,(ii+d),L
16	RL (ii+d)	56	BIT 2,(ii+d)	96	RES 2,(ii+d)	D6	SET 2,(ii+d)
17	RL (ii+d),A	57	-	97	RES 2,(ii+d),A	D7	SET 2,(ii+d),A
18	RR (ii+d),B	58	-	98	RES 3,(ii+d),B	D8	SET 3,(ii+d),B
19	RR (ii+d),C	59	-	99	RES 3,(ii+d),C	D9	SET 3,(ii+d),C
1A	RR (ii+d),D	5A	-	9A	RES 3,(ii+d),D	DA	SET 3,(ii+d),D
1B	RR (ii+d),E	5B	-	9B	RES 3,(ii+d),E	DB	SET 3,(ii+d),E
1C	RR (ii+d),H	5C	-	9C	RES 3,(ii+d),H	DC	SET 3,(ii+d),H
1D	RR (ii+d),L	5D	-	9D	RES 3,(ii+d),L	DD	SET 3,(ii+d),L
1E	RR (ii+d)	5E	BIT 3,(ii+d)	9E	RES 3,(ii+d)	DE	SET 3,(ii+d)
1F	RR (ii+d),A	5F	-	9F	RES 3,(ii+d),A	DF	SET 3,(ii+d),A
20	SLA (ii+d),B	60	-	A0	RES 4,(ii+d),B	E0	SET 4,(ii+d),B
21	SLA (ii+d),C	61	-	A1	RES 4,(ii+d),C	E1	SET 4,(ii+d),C
22	SLA (ii+d),D	62	-	A2	RES 4,(ii+d),D	E2	SET 4,(ii+d),D
23	SLA (ii+d),E	63	-	A3	RES 4,(ii+d),E	E3	SET 4,(ii+d),E
24	SLA (ii+d),H	64	-	A4	RES 4,(ii+d),H	E4	SET 4,(ii+d),H
25	SLA (ii+d),L	65	-	A5	RES 4,(ii+d),L	E5	SET 4,(ii+d),L
26	SLA (ii+d)	66	BIT 4,(ii+d)	A6	RES 4,(ii+d)	E6	SET 4,(ii+d)
27	SLA (ii+d),A	67	-	A7	RES 4,(ii+d),A	E7	SET 4,(ii+d),A
28	SRA (ii+d),B	68	-	A8	RES 5,(ii+d),B	E8	SET 5,(ii+d),B
29	SRA (ii+d),C	69	-	A9	RES 5,(ii+d),C	E9	SET 5,(ii+d),C
2A	SRA (ii+d),D	6A	-	AA	RES 5,(ii+d),D	EA	SET 5,(ii+d),D
2B	SRA (ii+d),E	6B	-	AB	RES 5,(ii+d),E	EB	SET 5,(ii+d),E
2C	SRA (ii+d),H	6C	-	AC	RES 5,(ii+d),H	EC	SET 5,(ii+d),H
2D	SRA (ii+d),L	6D	-	AD	RES 5,(ii+d),L	ED	SET 5,(ii+d),L
2E	SRA (ii+d)	6E	BIT 5,(ii+d)	AE	RES 5,(ii+d)	EE	SET 5,(ii+d)
2F	SRA (ii+d),A	6F	-	AF	RES 5,(ii+d),A	EF	SET 5,(ii+d),A
30	SLS (ii+d),B	70	-	B0	RES 6,(ii+d),B	F0	SET 6,(ii+d),B
31	SLS (ii+d),C	71	-	B1	RES 6,(ii+d),C	F1	SET 6,(ii+d),C
32	SLS (ii+d),D	72	-	B2	RES 6,(ii+d),D	F2	SET 6,(ii+d),D
33	SLS (ii+d),E	73	-	B3	RES 6,(ii+d),E	F3	SET 6,(ii+d),E
34	SLS (ii+d),H	74	-	B4	RES 6,(ii+d),H	F4	SET 6,(ii+d),H
35	SLS (ii+d),L	75	-	B5	RES 6,(ii+d),L	F5	SET 6,(ii+d),L
36	SLS (ii+d)	76	BIT 4,(ii+d)	B6	RES 6,(ii+d)	F6	SET 6,(ii+d)
37	SLS (ii+d),A	77	-	B7	RES 6,(ii+d),A	F7	SET 6,(ii+d),A
38	SRL (ii+d),B	78	-	B8	RES 7,(ii+d),B	F8	SET 7,(ii+d),B
39	SRL (ii+d),C	79	-	B9	RES 7,(ii+d),C	F9	SET 7,(ii+d),C
3A	SRL (ii+d),D	7A	-	BA	RES 7,(ii+d),D	FA	SET 7,(ii+d),D
3B	SRL (ii+d),E	7B	-	BB	RES 7,(ii+d),E	FB	SET 7,(ii+d),E
3C	SRL (ii+d),H	7C	-	BC	RES 7,(ii+d),H	FC	SET 7,(ii+d),H
3D	SRL (ii+d),L	7D	-	BD	RES 7,(ii+d),L	FD	SET 7,(ii+d),L
3E	SRL (ii+d)	7E	BIT 7,(ii+d)	BE	RES 7,(ii+d)	FE	SET 7,(ii+d)
3F	SRL (ii+d),A	7F	-	BF	RES 7,(ii+d),A	FF	SET 7,(ii+d),A

Vorbytes DD CB (ii=IX) oder Vorbytes FD CB (ii=IY)

4.4. Debugger TEMO

Haupteinsatzgebiet des Programmpaketes „Debugger“ bzw. "Testmonitor" ist die Unterstützung beim Test von Maschinenprogrammen. Es dient zur Fehlersuche bzw. Programmanalyse mittels Einzelbefehlsabarbeitung und Unterbrechungspunktsteuerung mit nachfolgender Anzeige der Registerinhalte. Für die zu testenden Anwenderprogramme werden eigene USER-Register definiert, welche nach der Anwenderprogrammabarbeitung im RAM gerettet werden und zur Abarbeitung in den CPU-Registersatz geladen werden. Die Einzelschrittabarbeitung erfolgt interruptgesteuert.

Der Kanal 0 des CTC-Bausteins im Grundgerät wird so programmiert, dass nach dem Sprung in das Anwenderprogramm auf dem ersten Befehl ein Interrupt ausgelöst wird. Anschließend werden die Register gerettet und angezeigt.

Die Unterbrechungspunktsteuerung ist auf zwei Arten möglich:

1. Abarbeitung im Echtzeitbetrieb; die Unterbrechung wird durch Eintragen eines RST 38 - Befehls auf den Unterbrechungspunkt realisiert,
2. die Unterbrechung wird über Adressvergleich nach Abarbeitung jedes Befehls realisiert (interruptgesteuert wie Einzelschrittabarbeitung)

Weiterhin beinhaltet der Testmonitor Kommandos zur Verschiebung von Speicherbereichen, Kommandos zur Einzelbyte-Ein- und -Ausgabe über die anzugebenden Ein-/Ausgabekanaladressen und weitere allgemein nutzbare Kommandos.

Hinweis: Mit dem Debugger können keine Programme oder Speicherinhalte im Adressbereich von C000H bis DFFFH untersucht werden, da in diesem Bereich der Programmcode des Debuggers selbst liegt.

4.4.1. Starten des Debuggers

Der Debugger wird aus dem CAOS-Menü über diese Kommandos gestartet:

%TEMO	Kaltstart Debugger/Testmonitor
%RETEMO	Warmstart Debugger/Testmonitor

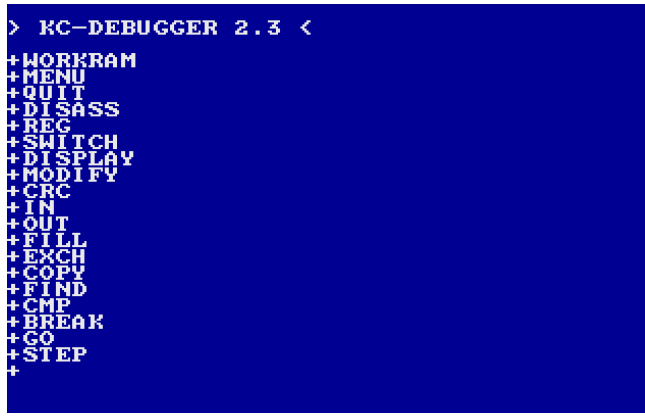
Ab Version 2.3 des Debuggers gibt es noch eine dritte Möglichkeit zum Aufruf einzelner Debugger-Kommandos direkt aus dem CAOS-Menü heraus, ohne vorher in das Debugger-Menü wechseln zu müssen. Dies gilt für die Kommandos CRC, IN, OUT, FILL, EXCH, COPY, FIND und CMP.

%TEMO <cmd> Aufruf eines einzelnen Debugger-Kommandos

4.4. Debugger TEMO

Beim Kalt- oder Warmstart wechselt der Debugger zu Bild 1, sodass ein Anwenderprogramm das Bild 0 benutzen kann und nicht von den Debugger-Anzeigen zerstört wird.

Nach dem Start erscheint das Debugger-Menü:



Durch den Sprung in das Debugger-Menü über Kommando TEMO wird der Unterbrechungspunkt- und Schrittbetrieb initialisiert. Dazu werden
- die USER-Register AF, BC, DE, HL und PC auf Null gesetzt
- der Anwender-Stack unterhalb des Systemstacks auf 160H gelegt
- der Interruptvektor des CTC0 für den Einzelschrittbetrieb vorbereitet

4.4.2. Das Debugger-Menü

Debugger-Kommando WORKRAM

+WORKRAM [aa]

verlagert den IX-Bereich, die Interrupttabellen und den Stack in einen anderen RAM-Bereich. Der Parameter aa ist dabei der höherwertige Teil der Zieladresse. Ohne Parameter aa wird der Standardwert 01H eingestellt.

Beispiel: WORKRAM 3F – Verlegen von IX-, Interrupttabelle und Stack an das Ende des RAM0, auf Adresse 3F00H

Bei Kommandoausführung wird der Bildschirm gelöscht, das TEMO-Menü erscheint und der USER-Stack wird neu initialisiert (im Beispiel auf Adresse 3F60H).

4.4. Debugger TEMO

Debugger-Kommando MENU

+MENU [n] schreibt das Testmonitor-Menü neu aus. Wie beim gleichlautenden CAOS-Menüwort (siehe Seite 45) kann auch hier durch einen Parameter die Anzeige versteckter Menüworte und die Auflistung der Startadressen veranlasst werden.

Debugger-Kommando QUIT

+QUIT verlässt den Debugger und kehrt zum CAOS-Menü mit Wechsel zu Bild 0 zurück. Solange die Arbeitszellen vom Debugger nicht verändert werden, kann mit dem Kommando %RETEMO der Debugger an der Stelle fortgesetzt werden, wo er mit QUIT verlassen wurde.

Debugger-Kommando DISASS

+DISASS ruft den Disassembler auf, Funktion und Parameter siehe Seite 412.

Debugger-Kommando REG

+REG [name wert]

Dieses Kommando zeigt die USER-Register an und gestattet auch das gezielte Verändern einzelner Registerinhalte. Für die Anzeige aller Register wird REG ohne Parameter aufgerufen. Die Anzeige erscheint in der Form:

```
A  -FLAGS-- B C  D E  H L  M  IX   IY
00          0000 0000 0000 00 01F0 0000
A' -FLAGS'- B'C' D'E' H'L' M' SP   (SP)
00          0000 0000 0000 00 0160 0000
```

Zum Ändern eines Registers ist hinter das Menüwort REG der Name des Registers und der Wert anzugeben. Zulässig als Registernamen sind:

- Die Einzelregister: F A C B E D L H F' A' C' B' E' D' L' H'
- Die Doppelregister: AF BC DE HL IY SP PC AF' BC' DE' HL'
- Die Flags: S Z HY P N CY S' Z' HY' P' N' CY'

Debugger-Kommandos SWITCH, DISPLAY und MODIFY

Funktion und Parameter entsprechen den gleichlautenden CAOS-Menüworten.

+SWITCH siehe Seite 71
+DISPLAY siehe Seite 77
+MODIFY siehe Seite 78

Debugger-Kommando CRC

+CRC aadr eadr+1
%TEMO CRC aadr eadr+1

Berechnung einer 16Bit-Püfsumme und CRC16 über einen Speicherbereich, es ist die Anfangsadresse und Endadresse+1 anzugeben.

Beispiel:

+CRC E000 0000
CRC16=0540 SUM16=9C54

Debugger-Kommando IN

+IN aaaa
%TEMO IN aaaa

Anzeige eines Datenbytes vom Eingabekanal mit der Portadresse aaaa. Die Portadresse kann sowohl als 8-Bit als auch als 16 Bit-Wert angegeben werden, je nachdem was die entsprechende Hardware benötigt.

Beispiel: IN 880 - Lesen des Strukturbytes des Moduls im rechten Steckplatz

Debugger-Kommando OUT

+OUT aaaa nn
%TEMO OUT aaaa nn

Ausgabe des Datenbytes nn zum Ausgabekanal mit der Portadresse aaaa. Die Portadresse kann sowohl als 8-Bit als auch als 16 Bit-Wert angegeben werden, je nachdem was die entsprechende Hardware benötigt.

Beispiel: OUT C80 0 - Abschalten des Moduls im linken Steckplatz

Debugger-Kommando FILL

+FILL aadr eadr+1 [Byte [Byte …]]
%TEMO FILL aadr eadr+1 [Byte [Byte …]]

Beschreiben (Füllen) eines Speicherbereiches mit einem Datenbyte bzw. einer Bytefolge. Wird kein Byte angegeben, dann wird der Speicherbereich mit 0-Bytes gefüllt, wird ein Byte angegeben, dann wird der gesamte Speicherbereich mit diesem Wert gefüllt. Es kann eine Bytefolge von bis zu 8 Bytes angegeben werden, mit denen der Speicher dann beschrieben wird.

Beispiele:

+FILL 4000 8000	Beschreiben des gesamten RAM4 mit 00H
+FILL 200 500 FF	Beschreiben des Bereiches von 200H bis 4FFH mit FFH
+FILL 200 300 AA 55	Beschreiben des Bereiches mit dem Bytemuster AA, 55, AA, …

Debugger-Kommando EXCH

+EXCH aadr eadr+1 adr2
%TEMO EXCH aadr eadr+1 adr2

Austausch des Inhaltes von zwei Speicherbereichen. Anzugeben ist dabei die Anfangs- und Endadresse eines Bereiches und die Anfangsadresse des zweiten Speicherbereiches.

Debugger-Kommando COPY

+COPY aadr eadr+1 adr2
%TEMO COPY aadr eadr+1 adr2

Kopieren des Inhaltes eines Speicherbereiches auf einen zweiten Speicherbereich. Anzugeben ist dabei die Anfangs- und Endadresse des Quellbereiches und die Zieladresse. Die Bereiche dürfen sich dabei auch überlappen.

Debugger-Kommando FIND

+FIND adr byte [Byte [Byte …]]
%TEMO FIND adr byte [Byte [Byte …]]

Sucht im Speicher ab der angegebenen Adresse nach einem bestimmten Byte bzw. einer Bytefolge. Durchsucht wird immer der gesamte 64K-Adressraum. Bei einer gefundenen Adresse wird diese angezeigt und auf eine Tastatureingabe gewartet. Die BRK-Taste bricht die Funktion ab, alle anderen Tasten setzen die Suche fort. Erscheint der Prompt ohne Anzeige einer Adresse, dann wurde die Bytefolge nicht gefunden.

Debugger-Kommando CMP

+CMP adr1 adr2
%TEMO CMP adr1 adr2

CMP vergleicht den Inhalt von zwei Speicherbereichen Byte für Byte. Bei einem gefundenen Unterschied werden Adressen und Inhalte angezeigt und eine Eingabe abgewartet. BRK bricht den Vergleich ab, jede andere Taste setzt den Vergleich fort.

Debugger-Kommando BREAK

+BREAK [adresse]

Setzen oder Anzeigen eines Unterbrechungspunktes. Ohne Parameter wird der aktuelle Unterbrechungspunkt angezeigt. Wird eine Adresse angegeben, dann wird der Unterbrechungspunkt auf diese Adresse gesetzt.

Debugger-Kommando GO

+GO [nnnn]

Sprung in ein Anwenderprogramm mit Initialisierung des Unterbrechungspunktes. Wird keine Adresse nnnn angegeben, dann erfolgt die Abarbeitung ab der gespeicherten PC-Adresse.

Liegt der Unterbrechungspunkt im RAM, dann wird für die Unterbrechung im Programm ein Befehl RST 38H eingetragen, der die Unterbrechung bewirkt. Die Programmabarbeitung erfolgt dann im Echtzeitbetrieb bis zum Unterbrechungspunkt.

Liegt der Unterbrechungspunkt im ROM oder in einem schreibgeschützten RAM, dann wird der CTC-Kanal 0 so initialisiert, dass er nach jedem Befehl den Befehlszählerstand kontrolliert bis zum Erreichen des Unterbrechungspunktes. Bei Erreichen des Unterbrechungspunktes wird in den Schrittbetrieb übergegangen - siehe auch Menüwort STEP.

Debugger-Kommando STEP

+STEP [nnnn]

Abarbeiten eines Programms im Schrittbetrieb. Ohne Angabe einer Adresse wird das Programm an der gespeicherten PC-Adresse weiter abgearbeitet. Das Kommando STEP springt beim Aufruf sofort in den Schrittbetrieb, ohne einen Befehl auszuführen. Die Programmabarbeitung kann über den Schrittbetrieb gesteuert werden.

Debugger-Kommando STACK

Dieses normalerweise nicht sichtbare Menüwort zeigt den Stackpointer (Nicht USER-Stack!) und den ersten Wert auf dem Stack an. Dies kann zur Fehlersuche hilfreich sein.

Beispiel:
+STACK
01C0 F296

Das Stackpointer zeigt auf 01C0H und dort ist der Wert F296H abgelegt. Das entspricht hier im Beispiel der Adresse der CAOS-Menükommando-Routine.

Debugger-Kommando „?“

Aufruf des Taschenrechners CALC, Funktion und Parameter siehe Seite 85.
Dieses Menüwort ist standardmäßig nicht im Menü sichtbar.

Debugger-Kommando „Punkt“

Der Punkt dient als Kommando zum Manipulieren des USER-Stacks. Dazu wird die MODIFY-Routine mit der Stack-Adresse und 2 Byte Breite aufgerufen.
Dieses Menüwort ist standardmäßig nicht im Menü sichtbar.

4.4.3. Schrittbetrieb

Der Schrittbetrieb wird mit dem Kommando STEP oder bei Erreichen des Unterbrechungspunktes nach dem Kommando GO erreicht. Im Schrittbetrieb wird am Bildschirm ab der ersten Zeile zunächst der Inhalt aller USER-Register angezeigt. Darunter drei reassemblierte Befehlszeilen:

- der zuletzt abgearbeitete Befehlscode
- der aktuelle Befehlscode (in roter Schriftfarbe)
- der Befehlscode, der dem aktuellen Befehl im Speicher folgt.

Unter den drei Befehlscodezeilen erfolgt schließlich eine Tastaturabfrage mit mehreren Möglichkeiten zur Fortsetzung.

Nach der Anzeige erscheint der Cursor und es kann mit folgenden Tasten fortgesetzt werden:


CR: einen Befehl im Schrittbetrieb abarbeiten


LF: Go und Unterbrechungspunkt auf nächsten Befehl setzen zum kompletten Abarbeiten von Programmschleifen und Unterprogrammen, einschließlich CAOS-Systemrufen

BRK: Rücksprung zum TEMO-Menü

G: Go - Programm weiter abarbeiten

U: Anzeige USER-Bildschirm bis zum nächsten Tastendruck

I: nächsten Befehl ignorieren

S: Schrittbetrieb mit laufender Befehls- und Registeranzeige

Die zeitliche Dehnung bei Unterbrechungspunkten im ROM ist etwa 120fach. Es kann jederzeit mit der 'BRK'-Taste unterbrochen und in die Schrittbetriebsanzeige gesprungen werden. (Die zeitliche Dehnung macht sich extrem beim Bildschirmlöschen und -rollen bemerkbar!)

Einschränkung: Der Schrittbetrieb und die Unterbrechungspunkte arbeiten nicht korrekt, wenn Speicherschaltungen im Adressbereich C000H-DFFFH ausgeführt werden, wo auch der Programmcode vom Debugger selbst liegt.

4.5. Editor EDIT

Seit CAOS 4.7 steht im USER-ROM erstmals ein universeller Texteditor zur Verfügung (vorausgesetzt es wird nicht die alternative USER-ROM-Variante mit FORTH benutzt). Mit CAOS 4.8 ist der Editor fester Bestandteil des USER-ROM. EDIT ist eine Weiterentwicklung auf Basis des EDAS-Editors, weist aber viele Eigenschaften von WordPro auf. Die Hauptmerkmale sind:

- 80-Zeichen breite Textzeilen
- 3 verschiedene Zeichensätze mit Grafikzeichen
- großer Textspeicher bis 55,5 KByte
- Import von Texten anderer Textprogramme
- Texte mit Zeilenendezeichen CR, LF oder beides werden gelesen
- schnelle Scrolling-Routinen für effektive Zeilenwechsel

Der Editor wird vom CAOS-Menü mit diesen Kommandos gestartet:

%EDIT	Kaltstart Editor mit Initialisierung der Arbeitszellen
%REEDIT	Warmstart Editor ohne den Text zu löschen

Der Editor kann außer mit dem RAM8 auch mit einem RAM auf der Adresse C000H arbeiten. Deshalb sucht der Editor beim Start nach einem 64K- oder 16K-RAM-Modul und schaltet das erste gefundene Modul auf die Adresse C000H ein. Für den Zugriff auf die vom IRM bzw. ROM verdeckten RAM-Bereiche werden kurze Hilfsprogramme in den RAM0 kopiert, die Arbeitszellen sind eine Teilmenge des Assemblers, siehe Kapitel 4.2.11 Seite 410.

ACHTUNG! Soll neben einem 16K- oder 64K-RAM-Modul auch noch ein USB-Modul M052 im KC-System betrieben werden, so ist das M052 auf einem niedrigeren Steckplatz als das RAM-Modul anzuordnen, da sonst der Zugriff auf den Speicher des M052 durch das RAM-Modul verdeckt wird.

Vom Assembler wird der Editor mit diesem Kommando gestartet:

>EDIT	Wechsel vom Assembler zum Editor

Bei Aufruf des Editors aus dem Assembler heraus, werden die Arbeitszellen des Assemblers weiter verwendet. Damit gelten die beim Start des Assemblers vereinbarten Adressbereiche und eine im Speicher befindliche Markentabelle bleibt erhalten. Die Rückkehr zum Assembler erfolgt mit dem Kommando

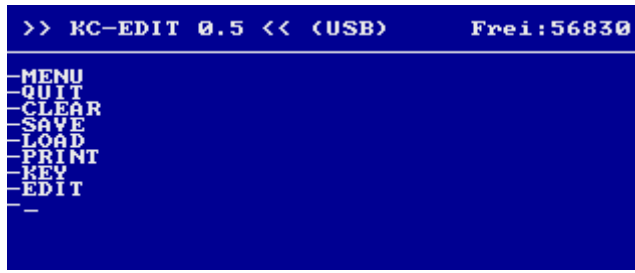
-QUIT	Beenden des Editors

4.5. Editor EDIT

4.5.1. Das Editor-Menü

Beim Start des Editors bzw. bei Rückkehr aus der Textbearbeitung erscheint ein kurzes Menü. Das komplette, lange Menü mit weiteren Menüpunkten kann durch das Menüwort **-MENU** angezeigt werden.

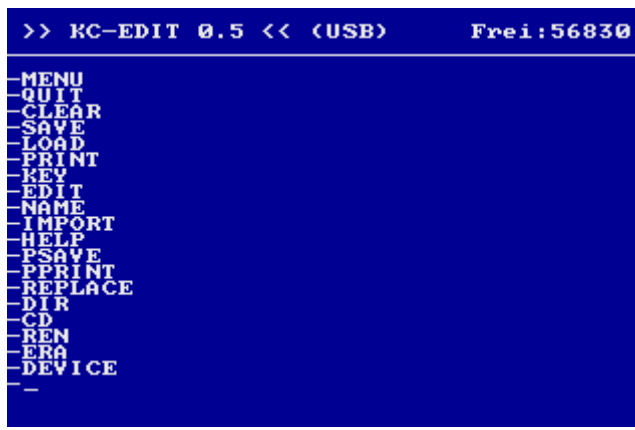
In der Titelzeile wird in der Mitte das aktuell eingestellte DEVICE und bei Diskette zusätzlich das eingestellte Laufwerk mit USER-Bereich angezeigt. Rechts wird außerdem noch die Anzahl freier Zeichen dezimal angezeigt.



Im Einzelnen bedeuten die Menüworte:

EDIT-Kommando MENU

Ausschreiben des vollständigen Menüs.
Befindet sich eine geladene oder bereits abgespeicherte Datei im Speicher, dann zeigt die Titelzeile an Stelle der EDIT-Version den Dateinamen an.



EDIT-Kommando QUIT

Verlassen des Assemblers zu CAOS bzw. ASM, je nachdem woraus der Editor gestartet wurde.

EDIT-Kommando CLEAR

Löschen des Textspeichers und des aktuellen Dateinamens. Eine Sicherheitsabfrage soll ein versehentliches Löschen des Textes verhindern.

EDIT-Kommando SAVE

Abspeichern des im Speicher befindlichen Textes auf das aktuell eingestellte Speichergerät. Der aktuelle Dateiname kann vor dem Abspeichern noch einmal geändert werden. Wurde noch kein Dateiname vergeben, dann wird bei der Eingabe der Dateityp TXT vorgeschlagen. Abgespeichert wird im Textformat mit CR+LF als Zeilenwechsel und mit 03H als Ende-Kennung. Der Rest des letzten Datenblockes bis zu vollen 128 Bytes wird mit 1AH (CP/M-Ende-Zeichen) aufgefüllt. Ein CAOS-Vorblock mit dem Dateinamen wird nur bei Kassette vorangestellt. Eine gespeicherte Textdatei hat systembedingt eine Mindestgröße von 256 Byte.

! Beim Weiterbearbeiten einer mit EDIT gespeicherten Textdateien mit anderen Programmen ist auf das Textende-Zeichen 03H zu achten und alle danach folgenden Steuerzeichen zu ignorieren.

EDIT-Kommando LOAD

Einlesen eines Textes vom aktuell eingestellten Speichergerät.
Befindet sich bereits Text im Speicher, dann wird der neu geladene Text stets vor die erste Zeile der zuletzt bearbeiteten Textseite, also nach Teiltext 1 eingefügt. Damit kann eine „merge“-Funktion realisiert werden. Für einen neuen Text ist der Speicher zunächst mit CLEAR zu löschen!
Befindet sich bereits ein Dateiname im Speicher, dann wird dieser Name nicht mit dem Namen der geladenen Datei überschrieben.

Ist als DEVICE Kassette eingestellt, dann wird die nächste Datei vom Recorder eingelesen und dabei der Dateiname aus dem Vorblock angezeigt und abgespeichert. Tritt während des Ladevorgangs ein Lesefehler auf ("*" erscheint), so kann nach Rückspulen der Kassette erneut versucht werden, den fehlerhaften Block zu lesen. Gelingt das trotz mehrmaligen Probierens nicht, so ist der Ladevorgang mit der BREAK-Taste abzubrechen. Bei Abbruch ist der bis dahin eingelesene Text verfügbar. Auf die BREAK-Taste reagiert der Rechner nur, wenn ein Signal vom Kassettenrecorder anliegt.

4.5. Editor EDIT

EDIT-Kommando PRINT

PRINT [n] ohne Parameter: Ausdruck des Quelltextes im Endlosdruck
n = 1 bis 255: Ausgabe auf Drucker mit Blattwechsel nach n Zeilen.
(n ist dezimal anzugeben)

EDIT-Kommando KEY

KEY [n] ohne Parameter: KEYLIST (Aufruf Menükommandoroutine)
n = 0 : Grundbelegung des F-Tastenspeichers herstellen
1 <= n <= F : wie %KEY (Aufruf Menükommandoroutine)
n = D5 : Grundbelegung für D005-Tastatur herstellen
Beim Kaltstart des Editors wird die Grundbelegung der KC-Tastatur initialisiert, dabei ist F1=ESC, F2=TAB und F3=weitersuchen für Find-Funktion.

EDIT-Kommando EDIT

Wechsel zum Editiermodus, an der Stelle, die zuletzt bearbeitet wurde. Siehe nächstes Kapitel auf Seite 434.

Im vollständigen, langen Menü gibt es weitere Kommandos:

EDIT-Kommando NAME

Befindet sich noch kein Text im Speicher, bzw. hat man einen neuen Text eingegeben, dann ist der Dateiname zunächst leer. Mit diesem Kommando kann der aktuelle Dateiname eingegeben oder geändert werden. Ist noch kein Dateiname vorhanden, dann wird der Dateityp TXT vorgeschlagen. Dieser Dateiname wird bei einem danach folgenden SAVE vorgegeben, kann aber auch dort noch einmal geändert werden.

Nachladbares EDIT-Kommando IMPORT

Um Dateien verschiedener Textsysteme weiter bearbeiten zu können, existiert ein nachladbarer Konverter in Form einer Datei IMPORT.KCC. Beim ersten Aufruf des Kommandos IMPORT wird diese Datei in den IRM ab BA00H geladen und ausgeführt. Ist die IMPORT-Funktion bereits im Speicher, entfällt das Nachladen. Die Datei IMPORT.KCC muss also immer auf dem gleichen Speichergerät/ Verzeichnis vorhanden sein, von dem Dateien importiert werden sollen. IMPORT funktioniert im Prinzip genauso wie LOAD, nur dass zusätzlich ein Dateikonverter zwischengeschaltet ist. Folgende Konverter sind verfügbar:

Tabelle 63: Import-Konverter für Fremdformate des Editors

Nr.	Textprogramm	Beschreibung
1	WordPro"86 und WordPro9	Dateien liegen im KCC-Format vor, besitzen einen CAOS-Vorblock mit Anfangs- und Endadresse. Es handelt sich um reine Speicherabzüge ohne Zeilenschaltung CR/LF. Eine Komprimierung mit Kennbyte FFH, siehe [56] oder EAH bei WordPro9 wird automatisch dekomprimiert. Der Wordpro9-Zeichensatz und die wesentlichen Druckercodes werden konvertiert. Leerzeichen am Zeilenende werden entfernt, Leerzeichen im Text werden soweit möglich in Tabulator-Schritte konvertiert.
2	TypeStar	Dateien liegen im KCC-Format vor, also mit Vorblock und darin enthaltenen Adressen. Es handelt sich um reine Speicherabzüge. Eine Komprimierung mit Kennbytes C0-FFH wird automatisch dekomprimiert.
3	TEXOR	Dateien liegen im KCC-Format vor, also mit Vorblock und darin enthaltenen Adressen. Es handelt sich um reine Speicherabzüge. Umlaute, das Zeilenende-Zeichen 00H und das Textende-Zeichen FFH werden konvertiert.
4	WordPro‘89 und WordPro 5	Dateien liegen im Textformat vor, Druckersteuerzeichen, Umlaute, Sonder- und Grafikzeichen werden konvertiert.
5	TPKC bzw. WordStar	Dateien von TPKC oder WordStar haben einen ASCII-Zeichensatz mit 7 Bit. Das Bit 8 wird zur Markierung des Wortendes benutzt. Um solche Dateien weiterverarbeiten zu können, wird das Bit 7 beim Einlesen zurückgesetzt. Einige Steuerzeichen und Druckercodes werden konvertiert.

Nachladbares EDIT-Kommando HELP

Wie bei IMPORT handelt es sich hier ebenfalls um eine nachladbare Funktion. Beim ersten Aufruf des Kommandos HELP wird eine Datei HILFE.KCC in den IRM ab BA00H geladen und ausgeführt. Im Speicher kann sich also nur eine der beiden nachladbaren Programmteile befinden. Die HILFE-Funktion wird auf dem Bild 1 dargestellt und anschließend wieder zu Bild 0 gewechselt. Später kann im EDIT-Modus mit ESC-H kurz auf Bild 1 umgeschaltet werden, um diese Informationen zu sehen.

Tipp: Soll sowohl mit der HILFE-Anzeige als auch mit der IMPORT-Funktion gearbeitet werden, dann zunächst die HILFE.KCC aktivieren und danach die IMPORT-Funktion laden. Die HILFE-Anzeige bleibt dabei im Bild 1 des Computers erhalten.

4.5. Editor EDIT

EDIT-Kommando PSAVE

PSAVE dient zum Abspeichern eines Textblockes, der durch die Blockmarken ◄ und ► eingegrenzt wurde. Sind keine Blockmarken gesetzt oder nicht in der richtigen Reihenfolge, dann erscheint eine Fehlermeldung.

EDIT-Kommando PPRINT

-PPRINT [n]

Abspeichern eines Textblockes, der durch die Blockmarken ◄ und ► eingegrenzt wurde. Sind keine Blockmarken gesetzt oder nicht in der richtigen Reihenfolge, dann erscheint eine Fehlermeldung. Der optionale Parameter n hat die gleiche Bedeutung wie beim Kommando PRINT.

EDIT-Kommando REPLACE

-REPL [s]

Ersetzen von Zeichenketten ohne Schutzabfrage. Abfrage der zu suchenden (ersetzenden) Zeichenkette und der Ersatzkette. Ausgabe der Anzahl der Ersetzungen (dezimal). Besonders nützlich bei massiven Markenumbenennungen, z. B. nach Durchlauf von Maschinencode durch den Markenreassembler QMR.

– ohne Schalter s von vorn
– mit Schalter s ab aktuellem Seitenanfang.

4.5. Editor EDIT

EDIT-Kommandos DIR, CD, REN, ERA und DEVICE

Diese 5 Kommandos haben die gleiche Syntax und Wirkung wie die gleichnamigen Kommandos im CAOS-Menü, siehe Seite 51 bis 60.

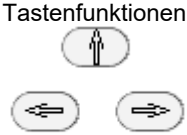
4.5.2. Der Editiermodus

Mit dem Kommando EDIT gelangt man in den Editiermodus. Hiermit wird der eigentliche Text bearbeitet. Die Zeichendarstellung wechselt jetzt in den 80-Zeichen-Modus. Am oberen Bildschirmrand befindet sich eine ständig sichtbare Statuszeile.

In der Statuszeile werden folgende Informationen angezeigt:
- die aktuelle Cursorposition (Zeile, Spalte)

- CAPS-Status ↑ für Groß- bzw. ↓ für Kleinbuchstaben ohne Shift-Taste
- Tastaturmodus: Einfügen/Überschreiben
- Tastenklick ein/aus (Glöckchen-Symbol bei aktiviertem Tastenklick)
- Zeichensatz (ASCII, deutsch, CAOS)
- HEX-Code des Zeichens auf der aktuellen Cursorposition
- Dateiname der gerade zu bearbeitenden Datei
- Lineal mit linker und rechter Randmarke sowie Tabulatoren

Die Bedienung des Editiermodus sollte möglich intuitiv sein. Es sind folgende Tastenfunktionen verfügbar:



Mit den Cursortasten bewegt man sich durch den Text, auch über die Bildschirmseiten hinaus. Bei Bedarf wird am Anfang und Ende des Bildschirms er Text gescrollt.

Shift+CUR und Shift+CUL springt wortweise weiter.

Shift+CUU und Shift+CUD blättert jeweils eine Seite.

BRK beendet den Editor und geht zum Menü zurück.

STOP schaltet zwischen den 3 Zeichensätzen um

Shift+STOP bzw. ESC und dient der Eingabe von Druckersteuerzeichen und dem Aufruf von Sonderfunktionen

INS schaltet zwischen Einfügen und Überschreiben um

Shift+INS schaltet den Tastenklick ein und aus

DEL löscht das Zeichen unter der Cursorposition, der Folgetext bis zum nächsten Tabulator rückt nach links nach

Shift+DEL löscht die gesamte Zeile, der Text unterhalb wird um eine Zeile nach oben geholt

CLR löscht das Zeichen links vom Cursor, der Folgetext bis zum nächsten Tabulator rückt nach links nach

Shift-CLR ruft die Hardcopy-Funktion auf, z. B. CALC

HOME setzt den Cursor in die linke/obere Bildschirmecke

Shift+HOME setzt den Cursor nach rechts unten

F1 dient bei der KC-Tastatur als ESC-Taste und hat hier die gleiche Funktion wie Shift-STOP.

F2 (F5) ist als Tabulator zu verwenden, im Einfügemodus wird der Text ab der Cursorposition mitgenommen

F3 (F6) dient zum Weitersuchen, falls mit FIND in den Editor gegangen wurde

ENTER im Einfügemodus zieht alles ab der aktuellen Cursorposition auf eine neue Zeile, so kann auch eine Leerzeile eingefügt werden, wenn man hinter das letzte Wort in der Zeile geht. Befindet sich der Cursor links von der linken Randmarke, dann steht der Cursor in der neuen Zeile auf der ersten Spalte, ansonsten auf der linken Randmarke.

Bei allen Funktionen, die die Zeile wechseln (außer ENTER), bleibt die Spaltenposition erhalten, wenn in dieser Spalte Zeichen stehen, ansonsten wird hinter das letzte Textzeichen zurückgegangen.

4.5.3. ESC-Funktionen im Editor

Mit der Kombination Shift-STOP, der Taste F1 auf der KC-Tastatur oder der ESC-Taste auf der D005-Tastatur und einer nachfolgenden Ziffern- oder Buchstabentaste werden Steuerfunktionen aufgerufen oder Drucksteuerzeichen eingegeben. Die folgende Tabelle zeigt die gesamte Liste der im Editiermodus verfügbaren ESC-Funktionen.

4.5. Editor EDIT

Tabelle 64: ESC-Steuerfunktionen im Editor

Taste	Funktion
ESC 0-9	Eingabe Druckersteuerzeichen, siehe Kapitel 4.5.4
ESC-A	ANFANG: Eingabe Blockanfangsmarke ◄ (Code 1CH)
ESC-B	BOTTOM: Gehe zum Textende
ESC-C	COPY: Block kopieren (kopiert einen durch die Blockmarken ◄ und ► markierten Textblock an die aktuelle Cursorposition)
ESC-D	DOWN: Eingabe Pfeil nach unten 1FH (derzeit ohne Funktion)
ESC-E	ENDE: Eingabe Blockendemarke ► (Code 1EH)
ESC-F	FIND: Suchen einer Zeichenkette, Weitersuchen mit F3 (F6 bei D005-Tastatur)
ESC-G	GRAFIK: Eingabe eines Grafikzeichens, welches mit den Cursortasten angewählt und mit Enter bestätigt wird. BRK bricht die Eingabe ab.
ESC-H	HELP: Umschaltung zu Bild 1 und Anzeige einer vorher dort geladenen Hilfe
ESC-J	Eject: Eingabe eines Seitenwechsel-Zeichens 0CH
ESC-K	KILL: Block löschen (löscht einen durch die Blockmarken ◄ und ► markierten Textblock)
ESC-L	Linke Randmarke auf TAB-Stop vor aktueller Cursorposition setzen
ESC-N	NoCR/LF: Zeile binden 0EH
ESC-R	Rechte Randmarke auf TAB-Stop vor aktueller Cursorposition setzen
ESC-T	TOP: Gehe zum Textanfang
ESC-U	UP: Eingabe Pfeil nach oben 1DH (derzeit ohne Funktion)
ESC-V	Block verschieben (verschiebt einen durch die Blockmarken ◄ und ► markierten Textblock zur aktuellen Cursorposition)
ESC-X	Eingabe Steuerzeichen für nachfolgende Hex-Sequenz 0FH (nachfolgende HEX-Codes werden direkt an Drucker gesendet)
ESC-Z	ZEILE: Gehe zur eingegebenen Zeilennummer

4.5.4. Druckersteuerzeichen

Mit der Kombination ESC + Zifferntaste werden Drucksteuerzeichen erzeugt, welche als kleine Zahlen im Text erscheinen und beim Druck in die entsprechenden Druckerbefehle umgewandelt werden. Für jede Funktion existiert nur ein Steuerzeichen. Beim ersten Auftauchen im Text wird die Funktion aktiviert und danach mit demselben Zeichen wieder abgeschaltet. Dies ist besonders zu beachten, wenn nur Teile von Texten gedruckt werden sollen! Die Druckersteuerzeichen sind mit den Codes belegt, die von WordPro 6 bekannt sind.

Tabelle 65: Druckercodes im Editor

Nr.	Funktion	ein	aus
0	unidirektional	ESC U 01	ESC U 00
1	Exponent (superscript)	ESC S 00	ESC T
2	Schmalschrift (condensed)	SI	DC2
3	Unterstreichen (underline)	ESC - 1	ESC - 0
4	Elite (Pica)	ESC M	ESC P
5	Breitschrift (enlarged)	ESC W 01	ESC W 00
6	Doppeldruck (double strike)	ESC G	ESC H
7	Index (subscript)	ESC S 01	ESC T
8	Fettschrift (emphasized)	ESC E	ESC F
9	Schönschrift (NLQ)	ESC x 01	ESC x 00

5. ANHANG

5.1. Technische Parameter

Tabelle 66: Technische Daten

Bezeichnung:	Kleincomputer KC 85/5
Hersteller:	VEB Mikroelektronik „Wilhelm Pieck“ Mühlhausen im Kombinat Mikroelektronik
Bauform:	Grundgerät mit abgesetzter Tastatur
Abmessungen:	Grundgerät 385 * 270 * 77 (in mm) Tastatur 296 * 152 * 18/19 (in mm)
Masse:	ca. 4800 g (Grundgerät + Tastatur)
Schutzgrad:	IP 20 nach TGL 15165
Betriebsspannung:	230 V (Sicherungen: 0,315 A träge ; 2,5 A flink)
Leistungsaufnahme:	ca. 25 VA, bei Nennbetriebsspannung ohne Module ca. 35 VA, bei Nennbetriebsspannung mit Modulen
Prozessortyp:	U 880 D (Z80)
Taktfrequenz:	1,77 MHz *
Schreib-Lesespeicher: für Anwender nutzbar:	320 KByte dRAM ca. 256 KByte
Festwertspeicher:	48 KByte ROM
Bildaufbau:	vollgrafisch, 320 x 256 Bildpunkte
freiprogrammierbare Bildpunktzahl:	81.920
Vordergrundfarben:	16
Hintergrundfarben:	8
Anzeigeeinheit:	handelsübliches Farb-Fernsehgerät
Anschlussmöglichkeiten an TV:	Antenneneingang, FBAS-Anschluss, RGB-Eingang (SCART)
verwendete Farbfernsehform:	PAL-COLOR
Tonerzeugung:	2 Tongeneratoren
Tonhöhenumfang:	2 * 7 Oktaven

TECHNISCHE PARAMETER

Tonwiedergabe:	– über Fernsehgerät (Mono) FBAS-RGB-Eingang, Lautstärke in 16 Stufen regelbar – über Stereoanlage bei konstantem Pegel – über eingebauten Piezosummer
Externer Programm- und Datenspeicher	– handelsüblicher Magnetbandkassetten-recorder oder Spulentonbandgerät – mit Erweiterung D004: 5,25“-Diskette – mit Erweiterung D008: zusätzlich 3,5“-Diskette und IDE-Festplatte
Motorschaltung:	vorhanden (TTL-Pegel)
Erweiterungsmöglichkeiten:	2 Modulsteckplätze im Grundgerät, Anschluss für Erweiterungsaufsatz
Besonderheiten:	– interne Speicher über Programme abschaltbar – mehrere Module vom gleichen Typ quasi gleichzeitig nutzbar – Zeichenbilder und Tastencodes frei wählbar – abgesetzte Schreibmaschinentastatur ergonomisch gestaltet
Anzahl der Tasten:	64
frei programmierbare Tasten:	6 Funktionstasten, doppelt belegbar
Programmiersprachen:	BASIC, FORTH, Assembler (U880) …

** Der Systemtakt im KC 85/5 ist theoretisch 1/8tel von VCOT (14,187580 MHz) also 1,7734475 MHz. Der Horizontalzähler wird jedoch regelmäßig durch das Signal HZR zurückgesetzt, läuft also nicht linear durch. HZR wird aktiv, wenn die Signale m1 + m3 + h3 + h4 + h5 auf High stehen, der Zähler zählt damit von 0 bis 906. Wenn HZR einsetzt, sind gerade m0=Low, m2=Low und m1=High geworden, da ist vom Zählerstand her mindestens 1/4tel von m2 erreicht. Bevor HZR tatsächlich wirksam wird, gehen aber auch noch 2 Gatterlaufzeiten in der Auskodierung verloren (> 20 ns). Nach dem Rücksetzen wartet der Zähler auf die nächste H-Flanke von VCOT, um neu bei 0 zu starten.*

Damit braucht der Systemtakt alle 907 Takte (VCOT) ein 3/8tel länger.

907 / 8 = 113,375 (m2 Takte bis HZR)
113,375 - 0,375 = 113 (m2 Takte effektiv)
113 x 1,7734475MHz / 113,375 ≈ 1,7675816 MHz

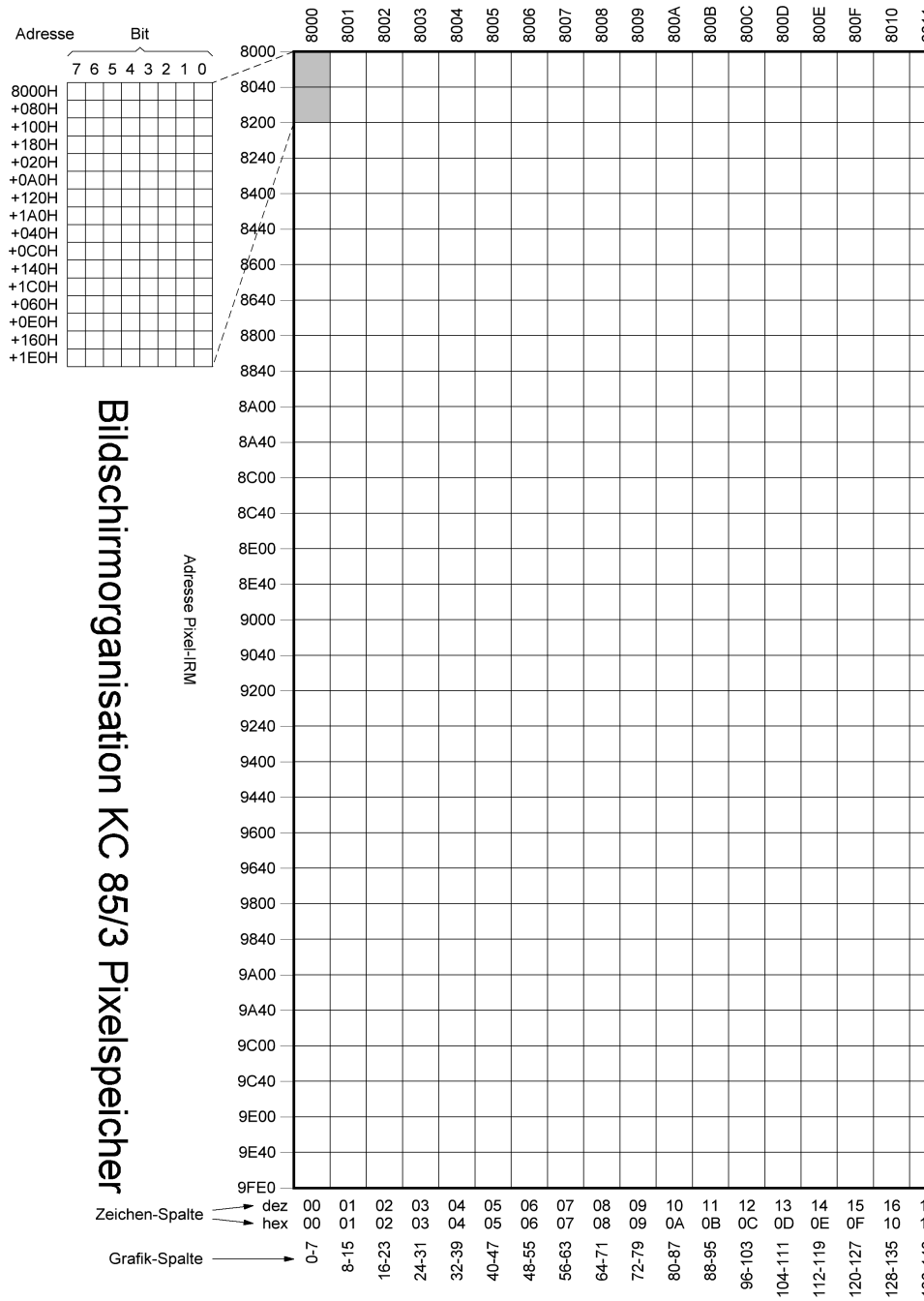


Bild 41: Bildschirmorganisation KC 85/3 Pixel-IRM

8012	8013	8014	8015	8016	8017	8018	8019	801A	801B	801C	801D	801E	801F	A000	A001	A002	A003	A004	A005	A006	A007	Grafik-Zeile	
														A000								00 00H	248-255
														A040								01 01H	240-247
														A008								02 02H	232-239
														A048								03 03H	224-231
														A010								04 04H	216-223
														A050								05 05H	208-215
														A018								06 06H	200-207
														A058							A1FF	07 07H	192-199
														A200								08 08H	184-191
																						09 09H	176-183
																						10 0AH	168-175
																						11 0BH	160-167
																						12 0CH	152-159
																						13 0DH	144-151
																						14 0EH	136-143
																					A3FF	15 0FH	128-135
														A400								16 10H	120-127
																						17 11H	112-119
																						18 12H	104-111
																						19 13H	96-103
																						20 14H	88-95
																						21 15H	80-87
																						22 16H	72-79
																					A5FF	23 17H	64-71
														A600								24 18H	56-63
																						25 19H	48-55
																						26 1AH	40-47
																						27 1BH	32-39
																						28 1CH	24-31
																						29 1DH	16-23
																						30 1EH	8-15
													9FFF								A7FF	31 1FH	0-7
18 12	19 13	20 14	21 15	22 16	23 17	24 18	25 19	26 1A	27 1B	28 1C	29 1D	30 1E	31 1F	32 20	33 21	34 22	35 23	36 24	37 25	38 26	39 27		
144-151	152-159	160-167	168-175	176-183	184-191	192-199	200-207	208-215	216-223	224-231	232-239	240-247	248-255	256-263	264-271	272-279	280-287	288-295	296-303	304-311	312-319	Zeichen-Zeile	

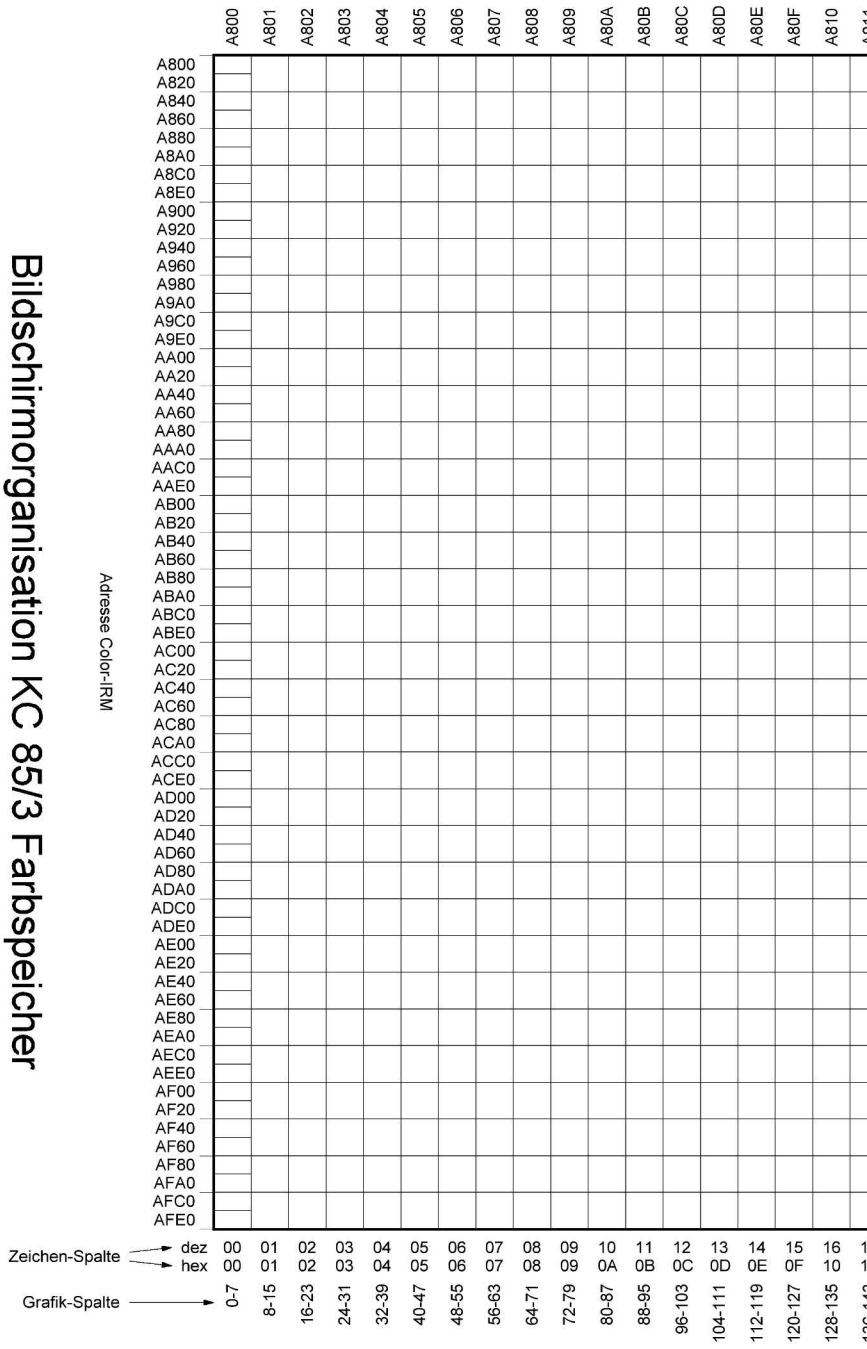


Bild 42: Bildschirmorganisation KC 85/3 Farb-IRM

A812	A813	A814	A815	A816	A817	A818	A819	A81A	A81B	A81C	A81D	A81E	A81F	B000	B001	B002	B003	B004	B005	B006	B007	Grafik-Zeile	
														B000 B020								00 00H	248-255
														B040 B060								01 01H	240-247
														B008 B028								02 02H	232-239
														B048 B068								03 03H	224-231
														B010 B030								04 04H	216-223
														B050 B070								05 05H	208-215
														B018 B038								06 06H	200-207
														B058 B078							B05F B07F	07 07H	192-199
														B080 B0A0								08 08H	184-191
														B0C0 B0E0								09 09H	176-183
														B088 B0A8								10 0AH	168-175
														B0C8 B0E8								11 0BH	160-167
														B090 B0B0								12 0CH	152-159
														B0D0 B0F0								13 0DH	144-151
														B098 B0B8								14 0EH	136-143
														B0D8 B0F8							B0DF B0FF	15 0FH	128-135
														B100 B120								16 10H	120-127
														B140 B160								17 11H	112-119
														B108 B128								18 12H	104-111
														B148 B168								19 13H	96-103
														B110 B130								20 14H	88-95
														B150 B170								21 15H	80-87
														B118 B138								22 16H	72-79
														B158 B178							B15F B17F	23 17H	64-71
														B180 B1A0								24 18H	56-63
														B1C0 B1E0								25 19H	48-55
														B188 B1A8								26 1AH	40-47
														B1C8 B1E8								27 1BH	32-39
														B190 B1B0								28 1CH	24-31
														B1D0 B1F0								29 1DH	16-23
														B198 B1B8								30 1EH	8-15
													AFDF AFFF	B1D8 B1F8							B1DF B1FF	31 1FH	0-7
18 12	19 13	20 14	21 15	22 16	23 17	24 18	25 19	26 1A	27 1B	28 1C	29 1D	30 1E	31 1F	32 20	33 21	34 22	35 23	36 24	37 25	38 26	39 27	↑	
144-151	152-159	160-167	168-175	176-183	184-191	192-199	200-207	208-215	216-223	224-231	232-239	240-247	248-255	256-263	264-271	272-279	280-287	288-295	296-303	304-311	312-319	Zeichen-Zeile	

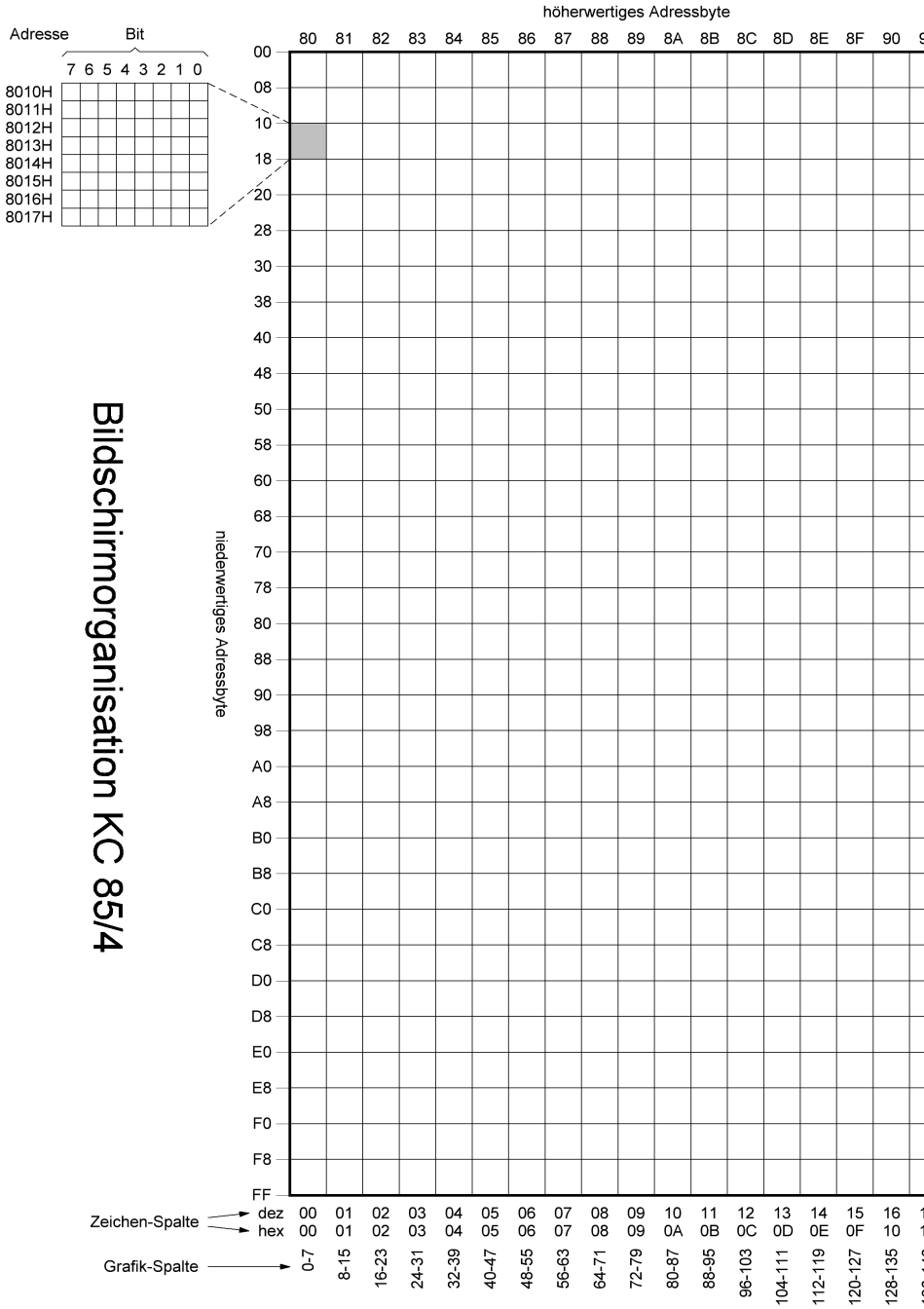
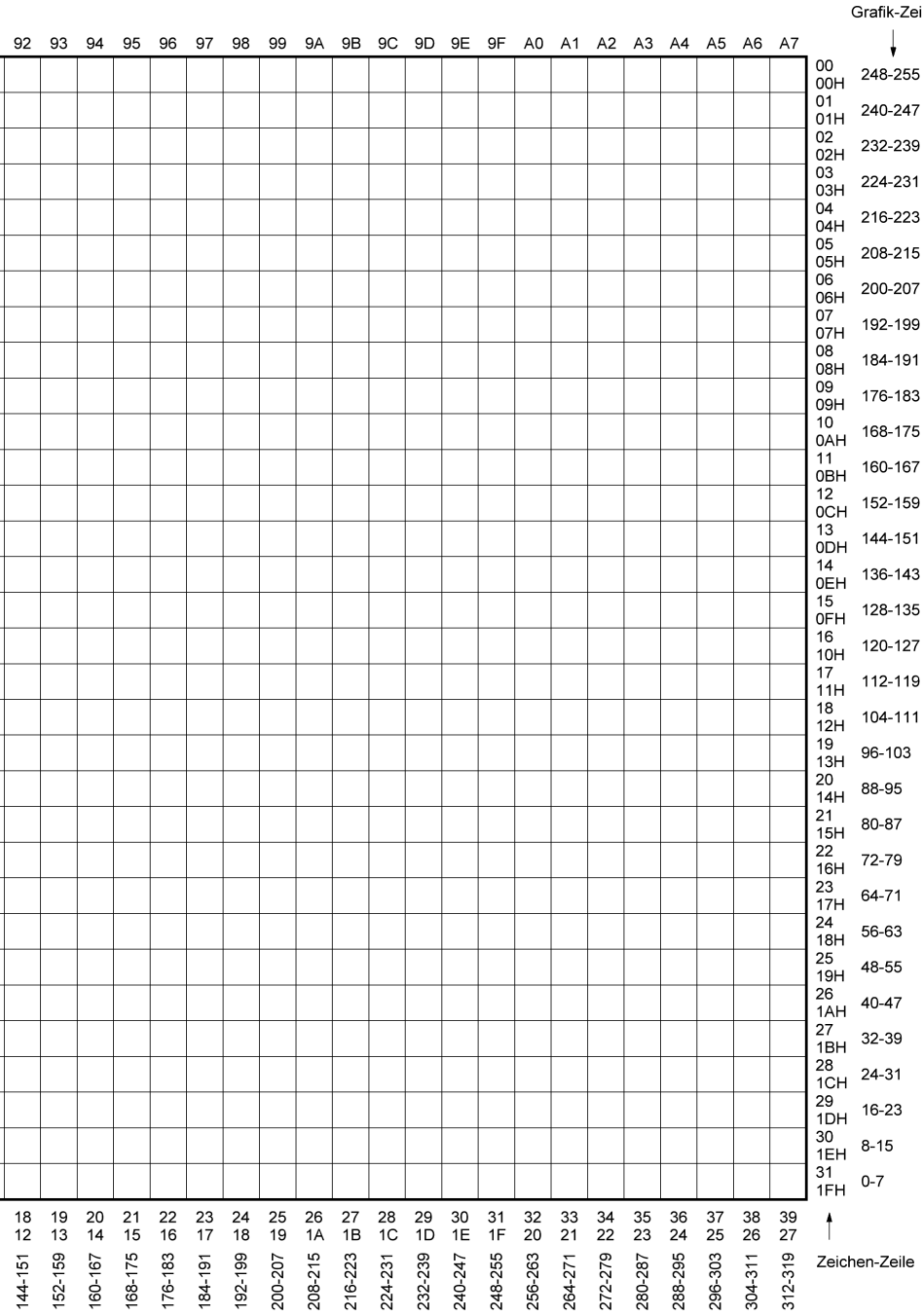


Bild 43: Bildschirmorganisation KC 85/4 Pixel- und Farb-IRM


Grafik-Zeile
92 93 94 95 96 97 98 99 9A 9B 9C 9D 9E 9F A0 A1 A2 A3 A4 A5 A6 A7
00 00H 248-255
01 01H 240-247
02 02H 232-239
03 03H 224-231
04 04H 216-223
05 05H 208-215
06 06H 200-207
07 07H 192-199
08 08H 184-191
09 09H 176-183
10 0AH 168-175
11 0BH 160-167
12 0CH 152-159
13 0DH 144-151
14 0EH 136-143
15 0FH 128-135
16 10H 120-127
17 11H 112-119
18 12H 104-111
19 13H 96-103
20 14H 88-95
21 15H 80-87
22 16H 72-79
23 17H 64-71
24 18H 56-63
25 19H 48-55
26 1AH 40-47
27 1BH 32-39
28 1CH 24-31
29 1DH 16-23
30 1EH 8-15
31 1FH 0-7
18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39
12 13 14 15 16 17 18 19 1A 1B 1C 1D 1E 1F 20 21 22 23 24 25 26 27
144-151 152-159 160-167 168-175 176-183 184-191 192-199 200-207 208-215 216-223 224-231 232-239 240-247 248-255 256-263 264-271 272-279 280-287 288-295 296-303 304-311 312-319
Zeichen-Zeile

5.3. Internetadressen für den KC 85

Die folgende Auswahl an Internetadressen, die sich mit dem KC 85 befassen, ist Stand März 2023 und erhebt keinen Anspruch auf Vollständigkeit.
Weiterführende Links sind auf jeder der angegebenen Websites zu finden.

Offizielle Website des KC-Clubs:	www.kcclub.de
CAOS-Quelltexte:	https://github.com/maleuma/CAOS
Alternative Adressen:	www.kc-club.de www.kc-club.net
KC 85-Labor:	http://kc85.info
KC 85/4 Aufbau und Bedienung: Der KC 85	http://www.mpm-kc85.de www.kc85.net
Robotrontechnik mit Forum:	http://www.robotrontechnik.de
CAOS 3.4 für KC 85/3:	https://www-user.tu-chemnitz.de/~heha/basteln/8bit/KC85/KC85-3 mit CAOS 3.4

Emulatoren für den KC 85:

KCemu (Torsten Paul):	http://kcemu.sourceforge.net http://sourceforge.net/projects/kcemu
KC85EMU (Frank Ludwig):	http://www.kc85emu.de
JKCEMU (Jens Müller):	http://www.jens-mueller.org/jkcemu
KCEMU (Henrik Haftmann):	http://www-user.tu-chemnitz.de/~heha/hs_freeware/kcemu
KC-Emulator für DOS: (Arne Fitzenreiter)	http://www.fitzenreiter.de
Virtueller KC im Browser (A. Weissflog)	https://floooh.github.io/virtualkc/index.html
KC-Emulator im Browser (Alexander Lang)	https://www.lanale.de/kc85_emu/KC85_Emu.html

5.4. CAOS Versionsgeschichte

Neben den CAOS-Betriebssystemen gab es auch Versionen mit anderen Namen. In der nachfolgenden Übersicht soll ohne Anspruch auf Vollständigkeit ein grober Überblick gegeben werden:

Version	Datum	Für Hardware KC 85/2 und KC 85/3
HC900 CAOS	ca. 1984	Betriebssystem des HC900 (erste Modelle des KC 85/2) funktionell fast identisch zu CAOS 2.2
CAOS 2.1		Entspricht HC900 CAOS – nur umbenannt?
CAOS 2.2	**ca. 1984**	**Standard-Betriebssystem des KC 85/2**
HC901 CAOS	ca. 1985	Betriebssystemerweiterung für den HC900 im Modul M006, funktionell fast identisch zu CAOS 3.1 Unterschiede: - SP = 01B4H statt 01D4H - UP 43H bis 45H noch nicht vorhanden - Prologbyte fest 7FH, noch nicht per (IX+9) änderbar
CAOS 2.3	1987	Alternatives Betriebssystem für den KC 85/2 mit schnelleren Bildschirmroutinen
CAOS 2.4	1988	Alternatives Betriebssystem für den KC 85/2 mit schnelleren Bildschirmroutinen und EDAS statt BASIC
CAOS 2.5	1989	Alternatives Betriebssystem für den KC 85/2 mit schnelleren Bildschirmroutinen und EDAS 2.0
CAOS 3.1	**ca. 1986**	**Standard-Betriebssystem des KC 85/3 mit integriertem BASIC-Interpreter im ROM**
CAOS 3.3	1987	Weiterentwickeltes CAOS von Frank Klemm. - Tastaturpuffer 0200h-02FFh - Centronics-Druckertreiber für PIO auf Portadresse C0h - V.24-Druckertreiber für M003/M053 - CAOS Unterprogramme PUSE und PUDE modifiziert
OS pi/88	1988	Alternatives Betriebssystem für den KC 85/3
OS pi/90	1990	Alternatives Betriebssystem für den KC 85/3
CAOS 3.4	1991	Alternatives KC 85/3-Betriebssystem mit KC 85/4-Optik und wesentlich schnelleren Bildschirmroutinen

VERSIONSHISTORIE

Version	Datum	Für Hardware KC 85/4 und KC 85/5
CAOS 4.1		Erstes bekanntes Betriebssystem für den KC 85/4 mit einem Fehler bei der Fensterinitialisierung
CAOS 4.2	**1987**	**Standard-Betriebssystem des KC 85/4**
CAOS 5.0	1990	Weiterentwicklung von Frank Pischel - Autostart D004 (3 Versuche) - DISPLAY heißt hier DUMP - MODIFY entfällt als Menüwort - MODUL entfällt als Menüwort - SYSTEM (und ESC-6) zeigt Module mit an - KEYLIST entfällt als Menüwort - KEY ohne Argumente ersetzt KEYLIST - Centronicstreiber für M021 - Joysticktreiber für M021
CAOS 6.0	1992	Hauptsächlich Ideen zur Weiterentwicklung im Quelltext eingetragen, kaum Änderungen gegenüber CAOS 5.0
		Ab hier spricht man vom KC 85/5
CAOS 4.3	1995	Erste Version für 256K RAM und 32K USER-ROM Centronics-Druckertreiber für M001 und M021 Neue Menüworte: view, PRINT, FLOAD, FSAVE, DIR, REN, ERA, SETRO, SETWR, DRIVE, TYPE, DUMP, INIT, go, CEN - SERVICE-kompatibler DISK-Verteiler auf F021H
CAOS 4.4	2003	Fehlerkorrekturen zu CAOS 4.3
CAOS 4.5	2010	Weiterentwicklung Daniel Elstner, Mario Leubner Joysticktreiber, LSTDEV statt V24 + CEN
CAOS 4.6	2016	DEVICE-Umschaltung TAPE und DISK, BASIC mit TAPE nicht nutzbar (DEVICE-Umschaltung zu langsam) - Taschenrechner CALC und Joystick-Editor JEDIT - DEVTAB und SUTAB standardmäßig im IRM - DISK-Verteiler auf F021H deaktiviert
CAOS 4.7	2018	DEVICE-Umschaltung für USB-Modul hinzugefügt, Byte-Routinen für BASIC neu programmiert, - neuer Programmverteiler PV7 auf F021H - 80-Zeichen-Editor im USER-ROM

VERSIONSHISTORIE

Version	Datum	Für Hardware KC 85/4 und KC 85/5
CAOS 4.8	2021	Fehlerkorrekturen zu CAOS 4.7: - Fehlerauswertung bei UP 36H/37H/38H (SAVE/ MBIN/MBOUT) mit CY-Flag - CSAVE/CLOAD bei TAPE korrigiert - Korrektur bei Menüwort V24DUP ohne Argumente - Fehlerbeseitigung bei %INIT (USB) - weniger DI/EI bei Zeichenausgabe und Grafikroutinen - TCIF überwacht jetzt auch Fenstergröße 0x0 - LOAD und MBIN und realisieren jetzt alle Bildschirmanzeigen über OCHR statt CRT - Dateinamensanzeige bei DIR jetzt über ZKOUT - HardCopy-Adresse bei Initialisierung V.24-Modul korrigiert - nach TAPELIST wird der CAOS-ROM-C wieder abgeschaltet Neuerungen: - USER-ROM nur noch mit Editor - Parameter A=FFH bei UP 31H (SIXD) neu definiert - Autostart INITIAL.UUU von USB - Erweiterung HELP-Kommando, ESC-H - ESC-E (Soft-Reset) und ESC-F (Soft-Power-On) - alternativer ROM mit dünnem Zeichensatz - Power on: erst Module abschalten, dann Speicher löschen - „2-Monitor-Modus“ mit ESC-G zur Umschaltung - Neuverteilung der IRM-Arbeitszellen ab AA00-AAFFh - verbessertes RANDOMIZE im BASIC-Interpreter - verbesserte Tonausgabe bei gleichen Parametern für linken und rechten Kanal. - neues Kommando FIND im Debugger - CALC berechnet nun auch 24Bit-Hexzahlen - TAPELIST erkennt nun auch PASCAL-Quelltexte - neue CAOS-UP HLDEZ und RDEZ - dezimale Argumente bei WINDOW, COLOR und dem User-Bereich im Kommando CD Treiber-Version 3.0: - Abfrage der Treiber-Version implementiert - Treiber-spezifische USB-Funktionen - Fehlercodes in DE-Register bei DEVICE-Funktionen ab Treiber-Version 3.0 (USB) - optionale Pfadangabe bei allen USB-Dateioperationen

5.5. Bekannte Probleme ab CAOS 4.7

Mit CAOS 4.7 hat sich einiges an den genutzten Speicherbereichen verändert und auch die Parameter der Systemunterprogramme sind zum Teil erweitert worden, z. B. für die DEVICE-Funktionen. Deshalb kann nicht ausgeschlossen werden, dass einige ältere Programme nicht mehr oder nur noch eingeschränkt lauffähig sind. Generell gilt:

A) Programme, welche den IRM im Adressbereich A900-AAFFh nutzen, funktionieren nicht mehr, da hier die DEVICE-Tabelle und die SUTAB von CAOS liegt. (Programme vom KC 85/3 schreiben vielmals direkt in den COLOR-RAM, was bis CAOS 4.5 zumindest keinen Programmabsturz verursacht hatte)
B) Programme, welche direkt auf die D004-Schnittstelle zugreifen, können nicht auf einen anderen Treiber umgestellt werden (z. B. UNIPIC)
C) Alte CAOS-Programme, welche die TAPE-Routinen nutzen, können theoretisch auf die neue DEVICE-Schnittstelle zugreifen. Nur fehlt beim LOAD die Angabe des Dateinamens, da dies beim Kassettenbetrieb nicht erforderlich war. Neue oder umgeschriebene Software muss bei LOAD oder ISRI den Dateinamen im Register HL übergeben.
D) BASIC-Programme, welche CALL-Aufrufe von SERVICE.KCC oder CAOS 4.5 nutzen, müssen umgestellt werden.
   - CALL*D8 → BLOAD“NAME“
   - CALL*DE → FILES
   - CALL*F0 → CHDIR
   - alle anderen CALL‘s müssen ersatzlos entfernt werden.
E) Für eine Nutzung der DEVICE-Schnittstelle ist es zwingend notwendig, jede Datei zu öffnen (ISRI/ISRO) und auch wieder zu schließen (CSRI/CSRO), sonst treten bei USB Fehler der Art „File Open“ auf.
F) Programme, welche den Inhalt von Adresse 000B verändern oder den Adressbereich 00AEH bis 00E0H überschreiben, können nicht mit der USB-Tastatur genutzt werden.
G) Bei Programmen, welche die beiden Bilder nutzen, können ab CAOS 4.8 Darstellungsprobleme auftreten. Abhilfe: Vor Programmstart ESC-G drücken.
H) Programme, welche nicht standardisierte ROM-Adressen aus einer speziellen CAOS-Version direkt benutzen, laufen nur unter dieser Version.
I) KCB-Programme mit Selbststartroutinen, welche den Modulsteuerbytespeicher des USER-ROM nicht aktualisieren. Siehe Kapitel 4.1.25 Seite 362.

In der folgenden Übersicht sind einige erkannte Probleme aufgeführt – ohne Anspruch auf Vollständigkeit:

VERSIONSHISTORIE

Programm	Beschreibung der Unverträglichkeit	
BAC854.SSS BAC854-C.SSS	Die Bascoder erstellen eine Kopie der Tastaturtabelle KTAB im Adressbereich ab 0048H und überschreiben die Interruptroutine der USB-Tastatur	F)
BigTurn	Benutzt Speicherbereich ab A900H und überschreibt dadurch die SUTAB	A)
BoulderDash	Lädt Programmteile nach	B)
BREAKOUT	Benutzt 000B-000F als Arbeitszellen, inkompatibel mit USB-Tastatur	F)
CAVE	Benutzt 000B-000F als Arbeitszellen, inkompatibel mit USB-Tastatur	F)
CLUB-X.SSS	Enthält MC zum direkten Löschen des Farb-IRM vom KC 85/3.	A)
DANGER.KCC	Lädt Fenster-Definitionen mit der WEND-Adresse von CAOS 4.2 → Patch erstellt, um WEND-Adresse zu ermitteln.	H)
FORTH (M026)	wurde noch nicht für die DEVICE-Schnittstelle angepasst, FORTHEX ist nur für D004 verfügbar.	B)
JUNGLE	JUNGLE 2.0 enthält einen Diskettentreiber für D004/DISK → JUNGLE 2.1 ist für den DEVICE-Treiber angepasst worden.	B)
OTHELLO	3D-OTHELLO ist ein KC 85/3-Programm mit IRM-Effekten, welche durch einen direkten IRM-Zugriff realisiert werden.	A)
PASCAL	KC-Pascal 2.1 macht kein ISRI beim Einlesen. Wenn (IX+2) von 0FFh auf 0 wechselt, dann wird Datei geöffnet und Name steht in 005CH… siehe PASEX.ASM	E)
SCHOCKY.KCC	Lädt Programmteile von Diskette nach, Diskettenroutinen sind fest einprogrammiert	B)
SERVICE.KCC	Das Nachladen des D004-Dienstprogramms SERVICE.KCC überschreibt den USB-Tastaturinterrupt, falls ein M052 mit Software 2.6 oder höher für VNC2 eingesetzt wird. → SERVICE.KCC ohne Autostart laden, dann wird der BASIC-Verteiler nicht nach 00D8H-00F8H kopiert!	F)
SNAKE.KCC	Direkter Zugriff auf den IRM (KC 85/3) → Programm umgeschrieben als SNAKE4.KCC	A)
SPSV4	SPSV4.OVR wird nicht nachgeladen. Beim manuellen Laden startet Programm nicht: Es ist ein spezieller D004-Umschalter enthalten, Menüworte TAPE/DISK	B)
TEXOR (M012)	Kommt nur teilweise mit der DEVICE-Schnittstelle zurecht. Abspeichern funktioniert, Laden funktioniert nicht, da kein Dateiname abgefragt wird. TEXOREX ist nur für D004 verfügbar.	C)
UNIPIC	Lädt alle Overlays und Dateien mit einem direkten D004-DISK-Loader, kennt also keine DEVICES und kann deshalb nicht mit USB arbeiten	B)
VOLLGAS	Lädt Programmteile nach, Laderoutine nutzt die FLOAD-Funktion der SERVICE-Schnittstelle von CAOS 4.5 auf Adresse F021H	B)

Programm	Beschreibung der Unverträglichkeit	
WordPro5, WordPro6	Die Zeichenbildtabelle liegt im IRM A820H…ABFFH und überschreibt die SUTAB von CAOS 4.7 → Interner Editor von CAOS 4.7 kann WordPro-Dateien bearbeiten oder WordPro 7 benutzen.	A)

5.6. Literatur

[1] Beschreibung zu M003 V.24-Modul *[34]
VEB Mikroelektronik „Wilhelm Pieck“ Mühlhausen

[2] Beschreibung zur Programmkassette C0171/1 V.24-Software *[34]
VEB Mikroelektronik „Wilhelm Pieck“ Mühlhausen

[3] Bedienungshandbuch Punkt Matrixdrucker *[34]
Seiko EPSON. Deutschland GmbH, Düsseldorf 1986

[4] Manual K 6311, K 6312 Hard-Copy-Drucker *[34]
VEB Robotron Büromaschinenwerk Sömmerda

[5] Manual K 6313, K 6314 Hard-Copy-Drucker *[34]
VEB Robotron Büromaschinenwerk Sömmerda

[6] Manual K 6327, K 6328 Hard-Copy-Drucker *[34]
VEB Robotron Büromaschinenwerk Sömmerda

[7] Manual K 6303, K 6304 Thermodrucker *[34]
VEB Robotron Büromaschinenwerk Sömmerda

[8] Manual S 3004 Schreibmaschine *[34]
VEB Robotron-Optima Büromaschinenwerk Erfurt

[9] Manual S 6005 Schreibmaschine „Erika“ *[34]
VEB Robotron-Optima Büromaschinenwerk Erfurt

[10] Manual S 6009 Schreibmaschine *[34]
VEB Robotron Büromaschinenwerk Karl-Marx-Stadt

[11] Manual S 6010 Schreibmaschine *[34]
VEB Robotron Büromaschinenwerk Karl-Marx-Stadt

[12] Manual S 6120 Schreibmaschine *[34]
VEB Robotron-Optima Büromaschinenwerk Erfurt

[13] Manual S 6130 Schreibmaschine *[34]
VEB Robotron-Optima Büromaschinenwerk Erfurt

[14] Gesetz über das Post- und Fernmeldewesen vom 29.11.1985,
Gesetzblatt Teil I, Nr. 31, Paragrafen 12 und 21

[15] Claßen, L.; Oefler, U.: Wissensspeicher Mikrorechnerprogrammierung
VEB Verlag Technik, Berlin 1986

[16] Barth, P.; Bohnsack, S.: Mikrorechentechnik Programmierung, Grundwissen für Lehrer
Volk und Wissen Volkseigener Verlag, Berlin 1987

LITERATURVERZEICHNIS

[17] Barthold, H.; Bäurich, H.: Mikroprozessoren-Mikroelektronische Schaltkreise und ihre Anwendung (Teile 1 und 2) Reihe electronica, Band 222/223 Band, 224/225.
VEB Militärverlag der Deutschen Demokratischen Republik, Berlin 1985

[18] Bückner, U.: Kleincomputer leichtverständlich
VEB Fachbuchverlag, Leipzig 1986

[19] Gutzer, H.: Spiel + Spaß mit dem Computer
Urania-Verlag. - Leipzig; Jena; Berlin 1987

[20] Heblik, P.: Wissensspeicher BASIC
Volk und Wissen Volkseigener Verlag, Berlin 1986

[21] Kieser, H.; Meder, M.: Mikroprozessortechnik
VEB Verlag Technik, Berlin 1985

[22] Kreul, H.; Leupold, D.; Horn, T.: Kleinstrechner-TIPS
(Broschürenreihe für Kleinstrechentechnik)
VEB Fachbuchverlag, Leipzig 1986

[23] Müller, S.: Programmieren mit BASIC
REIHE AUTOMATISIERUNGSTECHNIK, Band 216
VEB Verlag Technik, Berlin 1985

[24] Schlenzig, St.; Schlenzig, K.: Tips und Tricks für kleine Computer.
VEB Militärverlag der Deutschen Demokratischen Republik, Berlin 1988

[25] Scholz, K. P.: 1000 Begriffe für den Praktiker
VEB Verlag Technik, Berlin 1988

[26] Völz, H.: Elektronik Grundlagen, Prinzipien, Zusammenhänge
Akademie-Verlag, Berlin 1986

[27] Werner, D.: BASIC für Mikrorechner
VEB Verlag Technik, Berlin 1986

[28] Mikroprozessortechnik. Zeitschrift für Mikroelektronik, Computertechnik, Informatik.
VEB Verlag Technik Berlin (erscheint seit 1987)

[29] Funkamateur. Zeitschrift der Gesellschaft für Sport und Technik
(Artikel für Computertechnik ca. ab Jahrgang 34 (1985))
Militärverlag der Deutschen Demokratischen Republik

[30] rundfunk fernsehen elektronik. Fachzeitschrift
(Artikel zur Computertechnik ca. ab Jahrgang 34 (1985))
Verlag Technik, Berlin

[31] Jugend und Technik, Populärwissenschaftlich-technisches Jugendmagazin. (Artikel zur Computertechnik ab Jahrgang 32 (1984)).
Verlag Junge Welt, Berlin

[32] Domschke, W.: Kleincomputer KC 85/3-Hardwarekonzept
Mikroprozessortechnik, Berlin 1 (1987) 2, S. 56–59

[33] Domschke, W.: Das Softwarekonzept KC 85/3
Mikroprozessortechnik, Berlin 1 (1987) 3, S. 89–91

[34] Domschke, W.; Katzmann, K.: Der Modul M026 FORTH für Kleincomputer KC 85/2 und KC 85/3
Mikroprozessortechnik, Berlin 1 (1987) 8, S.244–246

[35] Kirves, K.-D.: V.24-Modul M003
Mikroprozessortechnik, Berlin 1 (1987) 4, S. 124–125

[36] Kirves, K.-D.; Schenk, B.; Schiwon, K.: Modul M011, 64 KByte-RAM
Mikroprozessortechnik, Berlin 1 (1987) 5, S. 147–148

[37] Kirves, K.-D.; Schenk, B.; Schiwon, K.: Digital-Ein-Ausgabemodul für KC 85/2 und KC 85/3
Mikroprozessortechnik, Berlin 1 (1987) 10, S. 308–310

[38] Kirves, K.-D.: Modul M027 Development-Assemblerprogrammierung für KC 85/3
Mikroprozessortechnik, Berlin 1 (1987) 8, S. 247–249

[39] Poppe, D.: Bustreiberaufsatz D002
Mikroprozessortechnik, Berlin 2 (1988) 5, S. 149–151

[40] Sieder, R.; Kraft, D.; Schenk, B.: Analogeingabemodul M010 ADU1 für KC 85/2 und KC 85/3
radio fernsehen elektronik Berlin 36 (1987) 4, S. 253–254

[41] Völz, H.: Textverarbeitung auf Kleincomputern
Mikroprozessortechnik, Berlin 1 (1987) 4, S. 118–120

[42] Völz, H.: BASIC 1 x 1 des Programmierens
Aufzeichnung der gleichnamigen Rundfunksendereihe von Radio DDR II aus dem Jahre 1987 auf 6 Hörspielkassetten.
Koproduktion Radio DDR II, VEB Deutsche Schallplatten, Berlin DDR

[43] Sonderheft der Zeitschrift URANIA
URANIA-Verlag Leipzig-Jena-Berlin 1987

[44] BASIC 1 x 1 des Programmierens
Sendereihe des Rundfunks der DDR. Ausstrahlungsbeginn: 1987

[45] Computerstunde
Sendereihe des Fernsehens der DDR. Ausstrahlungsbeginn: 1987

[46] BASIC für Fortgeschrittene
Sendereihe des Rundfunks der DDR. Ausstrahlungsbeginn: 1988

[47] DT 64-Computerclub
Sendereihe von Jugendradio DT 64. Ausstrahlungsbeginn: 1987

[48] KC 85/4 Systemhandbuch *[34]
VEB Mikroelektronik „Wilhelm Pieck“ Mühlhausen, Juli 1988

[49] KC 85/4 BASIC-Handbuch und BASIC-Übersichten *[34]
VEB Mikroelektronik „Wilhelm Pieck“ Mühlhausen

[50] D004 Manual *[34]
VEB Mikroelektronik „Wilhelm Pieck“ Mühlhausen, Dezember 1988

[51] D004 Handbuch für den Bediener *[34]
VEB Mikroelektronik „Wilhelm Pieck“ Mühlhausen, Oktober 1988

[52] D004 Handbuch für den Programmierer *[34]
VEB Mikroelektronik „Wilhelm Pieck“ Mühlhausen, Dezember 1988

[53] Beschreibung zum Modul M027 DEVELOPMENT *[34]
VEB Mikroelektronik „Wilhelm Pieck“ Mühlhausen, 1989

[54] W. Thiel: "Befehlsstruktur des U880"
rfe 7/1984 S. 419–421

[55] KC 85/3-Assemblertip: "Markenanzeige"
rfe 2/1988 S. 31

[56] INIR und OTIR sind unsymmetrisch
rfe 5/1987 S. 298

[57] Kürzere Ladezeiten für WordPro
Funkamateur 10/1989, S. 481

[58] Völz, H.: BASIC Effektiv programmieren auch mit Kleinstrechnern
Verlag Die Wirtschaft Berlin, Januar 1989

[59] Dr.-Ing. Werner Domschke, Kleinplotter XY 4131
radio fernsehen elektronik 4/1989, S. 217-218

[60] Beschreibung zur Ansteuersoftware für den Kleinplotter XY 4131 vom KC 85 des MPM in BASIC
VEB Mikroelektronik „Wilhelm Pieck“ Mühlhausen, 28.03.1989

[61] Claßen: Programmierung des Mikroprozessorsystems U880 – K1520
VEB Verlag Technik Berlin, 1981

[62] Roth, M.: Mikroprozessoren
Wissenschaftliche Zeitschrift KdT, Hochschulsektion TH Ilmenau 1979

[63] Beschreibung zu M001 DIGITAL IN/OUT *[34]
VEB Mikroelektronik „Wilhelm Pieck“ Mühlhausen

[64] Beschreibung zu M005 USER *[34]
VEB Mikroelektronik „Wilhelm Pieck“ Mühlhausen

[65] Beschreibung M007 ADAPTER *[34]
VEB Mikroelektronik „Wilhelm Pieck“ Mühlhausen

[66] Beschreibung zu M008 JOY-MODUL *[34]
VEB Mikroelektronik „Wilhelm Pieck“ Mühlhausen

[67] Beschreibung zu M052 USB
KC-Club, Mario Leubner, März 2018

[68] Beschreibung zu Ersatzleiterplatte für die 3 EPROMS mit einem Flash
Andreas Schlechte, 05.09.2021

[69] Handbuch zu Plotter BASIC 2.0 für XY 4131
KC-Club, Mario Leubner, Dezember 2022

Diese Auswahl der Literaturstellen erhebt keinen Anspruch auf Vollständigkeit.

*34 Diese Literatur wurde beim Kauf als gerätebezogene Dokumentation mitgeliefert

5.7. Abkürzungen

ADR	ADResse
ADU	Analog-Digital-Umsetzer
AFC	Automatic Frequency Control (Automatische Frequenzkontrolle)
ASCII	American Standard Code for Information Interchange (international standardisierter Code zur digitalen Verschlüsselung von Texten)
AV	Audio-Video
B	Binär (am Zahlenende zur Kennzeichnung)
Baud	Maßeinheit für die Übertragungsgeschwindigkeit von Daten (Symbolrate), hier gleichbedeutend mit Bit/s (1 Symbol entspricht 1 Bit)
BC	BüroComputer
BRK	BReaK (Abbruch)
CAOS	Cassette Aided Operating System (die Kassettenarbeit unterstützendes Betriebssystem)
CCR	Cursor Carriage Return (Cursor an den Anfang der Zeile)
CEL	Cursor to End of Line (Cursor an das Ende der Zeile)
CLL	CLear a Line (Löschen einer Zeile)
CLR	CLeaR (Löschen eines Zeichens)
CLS	CLear Screen (Löschen des aktuellen Fensters)
CPU	Central Processor Unit (zentrale Verarbeitungseinheit)
CR	Carriage Return (ENTER)
CRT	Cathode-Ray-Tupe (Katodenstrahlröhre = Bildschirm)
CTC	Counter Timer Circuit (Zähler-Zeitgeber-Baustein)
CUD	CUrsor Down (Cursor nach unten)
CUL	CUrsor Left (Cursor nach links)
CUR	CUrsor Right (Cursor nach rechts)
CUU	CUrsor Up (Cursor nach oben)
DAU	Digital-Analog-Umsetzer
DEL	DELete (Löschen)
E/A	Ein-/Ausgabe
EAS	Ein- und Ausgabesteuerung
ESC	ESCape (Umschaltcode)
FBAS	Farb-Bild-Austast-Synchronsignal
GByte	Gigabyte (1 GByte = 2^{30} Byte = 1.073.741.824 Byte)
H	Hexadezimal (am Zahlenende zur Kennzeichnung)
HCOPY	HardCOPY (Aufruf eines Sonderprogramms)
HF	HochFrequenz
HRG	High-Resolution-Grafik = Hochauflösender Grafikmodus

ABKÜRZUNGSVERZEICHNIS

IBM	IBM (International Business Machines Corporation) US-amerikanisches IT-Unternehmen und eines der weltweit führenden Unternehmen für Hardware und Software.
INS	INSert (Einfügen)
ISR	Interrupt-Service-Routine
I/O	Input/Output (Eingabe/Ausgabe)
IRM	Image Repetition Memory (Bildwiederholspeicher)
KBD	KeyBoarD (Tastatur)
KByte	Kilobyte (1 KByte = 2^{10} Byte = 1024 Byte)
KC	KleinComputer
LED	Light Emitting Diode (Licht emittierende Diode, Leuchtdiode)
LS	Lesen und Schreiben
MByte	Megabyte (1 MByte = 2^{20} Byte = 1.048.576 Byte)
MC	MaschinenCode
NF	NiederFrequenz
NL	Nur Lesen
PC	PersonalComputer
PIO	Parallel Input-Output (parallele Ein- und Ausgabe)
PV	ProgrammVerteiler
RAM	Random Access Memory (Schreib-Lese-Speicher)
RGB	Rot-Grün-Blau-Anschluß
ROM	Read Only Memory (Nur-Lese-Speicher)
SIO	Serial Input-Output (serielle Ein- und Ausgabe)
SPC	SPaCe (Leerzeichen)
TTL	Transistor-Transistor-Logic (Standard, der festlegt, welche elektrischen Kenngrößen zu den logischen Werten 0 und 1 gehören.)
TV	TeleVision
UHF	Ultra Hight Frequency (Fernsehkanäle von 20 bis 60)
UP	UnterProgramm
USB	Universal Serial Bus, ist ein serielles Bussystem zur Verbindung eines Computers mit externen Geräten
VHF	Very Hight Frequency (Fernsehkanäle von 3 bis 12)
VIF	VideoInterFace (Bildschirm-Controller)
ZRE	Zentrale RechenEinheit (Synonym: CPU)

5.8. Stichwortverzeichnis

Es bedeuten: f und folgende Seite
ff und mehrere folgende Seiten
fett Haupteintrag

Stichwortverzeichnis

5.9. Abbildungsverzeichnis

5.10. Tabellenverzeichnis

Notizen

mikroelektronik

RFT

veb mikroelektronik
wilhelm pieck
mühlhausen

KC-CLUB