GRAF computer

FLOMONCG

der Floppymonitor für den NDR-Z80 Computer

1.Ausgabe
Text: Rüdiger Nahm, Kaufbeuren
Gestaltung und Layout: Wolfgang Fink, Kempten GES
Druck und Bindung: Druckerei Rieder, Kempten
Bestellnr:11297

Erstellt mit "MicrosoftWord" und "PageMaker" auf einem mc-modular-AT/386

Inhaltsverzeichnis

1.0 Einführung

1.1. Zum NDR-Computer

Der NDR-Computer wird in der Fernsehserie "Computer Modular - Schritt für Schritt" aufgebaut, erklärt und in Betrieb genommen. Diese Serie wird vom Norddeutschen Rundfunk und vom Bayerischen Fernsehen ausgestrahlt. Es werden bald auch die Regionalsender anderer Bundesländer die Sendung in ihr Programm aufnehmen.

Zur Serie gibt es einige Begleitmaterialien, es ist daher nicht unbedingt notwendig, die Fernsehserie gesehen zu haben, um den NDR- Computer zu bauen und zu begreifen:

- **Bücher:** Rolf-Dieter Klein,
  "Rechner modular"
  Der NDR-Klein-Computer -
  selbstgebaut und programmiert
  ISBN 3-7723-8721-7, DM 68,-
  erschienen im Franzis-Verlag, München
  Auf diesem Buch baut die NDR-Serie auf

  Rolf-Dieter Klein,
  "Die Prozessoren 68000 und 68008"
  Rechnerarchitektur und Sprache im NDR-KLEIN-Computer
  ISBN 3-7723-7651-7, DM 78.-
  erschienen im Franzis-Verlag, München

- **Zeitschriften** "mc"und "ELO" des Franzis-Verlags

- **Zeitschrift** "LOOP" der Firma Graf
  Die Zeitschrift LOOP ist eine Kundenzeitschrift der Firma Graf und enthält Neuerungen, Änderungen, Tips und Tricks, Software usw. zum NDR-Computer

- **Videokassetten:**
  lizensierte Original Kassetten für den privaten Gebrauch.
  Auf diesen Kassetten sind die 26 Folgen der Fernsehserie enthalten.
  Systeme: VHS, Beta, Video 2000
  Preise: siehe gültige Preisliste

1.2 Vorwort

Digitale Schaltkreise in Computersystemen bestehen heutzutage aus einer hohen Integration von logischen Schaltelementen. Durch den Aufbau derartiger Schaltkreise ist der jetzige technische Standard und die damit verbundene, hohe Verarbeitungsgeschwindigkeit in der Elektronischen Datenverarbeitung zu erklären.

Diese Schaltkreise sind jedoch, wie sicher auch jedem nicht so versiertem Computeranwender bekannt, ohne die steuernde Software nur bis zu einem bestimmten Schwierigkeitsgrad eines Schaltungsproblems zu verwenden.

Die Softwaresteuerung der integrierten Schaltkreise beginnt bereits bei der untersten Stufe eines Mikrocomputers - ganz trivial gesagt bei ' Null'. Das bedeutet, der Prozessor des Computers beginnt bei der untersten Speicherstelle, die er besitzt: auf der Adresse '0'. An dieser Stelle muß sich in jedem Computersystem ein Programm befinden. Das an dieser Stelle stehende Programm ist im Fachjargon unter der Bezeichnung 'Monitorprogramm' oder auch 'ROM-BIOS' (ROM-BASIC Input/Output System) bekannt. Die Bezeichnung EPROM-BIOS erklärt auch schon, wo sich das Monitorprogramm im System befindet.
Der EPROM- Speicher ist ein programmierbarer 'nur' Lesespeicher (Read Only memory). Vereinfacht gesagt ist er vergleichbar mit einem 'IC'(Integrated Circuit), in das ein Programm mit Hilfe eines speziellen Epromprogrammiergeräts eingebrannt wurde. Unter 'Einbrennen' wird hierbei ein Vorgang verstanden, der es ermöglicht, bestimmte Werte in die Speicherstellen eines Silizium- chips zu übertragen, die auch nach Wegnahme der Spannungsversorgung erhalten bleiben.

Dieses Monitorprogramm ist also nach dem Einschalten des Computers die erste Anweisungsroutine, auf die der Prozessor trifft. Hier müssen die fundamentalsten Grunddeklarationen angeführt sein, die der Prozessor für seine Arbeit benötigt. Hierbei gehört die Speicherkonfiguration ebenso dazu, wie auch Ansteuerungsroutinen für die angeschlosssenen Massenspeicher, Ein/Ausgaberoutinen für die Tastaturschnittstellen, das Laden eines Betriebssystems oder aber auch die Bildschirmverwaltung.

Der NDR- Computer verfügt ebenfalls über ein solches Monitorprogramm. Das schon vielfach verbesserte FLOMON übernimmt hier diese Aufgabe.

1.2.1 Zum Handbuch

Das vorliegende Handbuch soll Ihnen als Vorlage und Begleiter bei der Arbeit mit dem Floppymonitor EFLOMONCG dienen. Es beinhaltet sämtliche Monitoreinsprünge und Funktionen, inklusive der Handhabung der Befehle und Erweiterungen.
Alle erklärten Befehle werden durch einfache Beispiele in der Programmiersprache BASIC aufgezeigt. Dadurch ist eine Integration in andere Hochsprachen problemlos gewährleistet.

Der erste Abschnitt erörtert die Erstinstallation des Floppymonitors, seine hardwaremäßigen Voraussetzungen und die Unterschiede gegenüber älteren Versionen.

Der zweite Abschnitt erklärt die grundlegenden Befehle des integrierten Debuggers.

Grafikbefehle, Mausansteuerung, Drucksequenzen und Floppybefehle werden in den darauffolgenden Kapiteln angesprochen.

Wichtig:

Die in diesem Handbuch wiedergegebenen Schaltungen und Verfahren werden ohne Rücksicht auf die Patentlage mitgeteilt. Sie sind ausschließlich für Amateur- und Lernzwecke bestimmt und dürfen nicht ohne Genehmigung des Lizensinhabers gewerblich genutzt werden. Bei gewerblicher Nutzung ist vorher die schriftliche Genehmigung des jeweiligen Lizensnehmers einzuholen.

Alle Schaltungen und technischen Angaben in diesem Handbuch wurden von dem Autor mit größter Sorgfalt erarbeitet bzw. zusammengestellt und unter Einschaltung von wirksamen Kontrollmaßnahmen reproduziert. Trotzdem sind Fehler nicht ganz auszuschließen. Der Lizensinhaber und der Autor sehen sich deshalb gezwungen, darauf hinzuweisen, daß sie weder Garantie noch die juristische Verantwortung oder irgendeine Haftung für Folgen, die auf fehlerhafte Angaben zurückzuführen sind, übernehmen.

2.0 Wozu dient der Monitor

Ein 'Monitor' ist das Minimalprogramm, mit dem ein Rechner ausgestattet werden muß, damit eine Programmierung von Hand oder von externen Speichermedien erfolgen kann. FLOMONCG beeinhaltet auch Unterprogramme, die die wichtigsten Ein/Ausgabearbeiten übernehmen (Zeichen einlesen, Diskettensektor lesen/schreiben ...). Neben diesen Unterprogrammen kann aber auch ein integrierter Tester (= Debugger) aufgerufen werden, mit dem im bescheidenen Maße Programme erstellt, verändert oder getestet werden können. Der wesentlichste Bestandteil ist aber die Terminalsoftware, die die Bildschirmverwaltung erledigt.

Das vorliegende Programm ist eine Weiterentwicklung der FLOMON- Versionen 1.0-4.2 und von RN-WINDOWS (Loop 15). Es beinhaltet also die erweiterten Fähigkeiten im Textmodus als auch die bekannten Grafikbefehle.

+ Geschwindigkeitssteigerung (200% beim SCROLL)
+ inverse Zeichen
+ Fensterverwaltung
+ Maus setzt Cursor (-> PULLUP-Menüs)
+ Hardcopy
+ Statuszeile am unteren Bildrand
+ EPSON-Modus
+ Uhrzeit ständig einblendbar

Als Terminalkarte kann wahlweise eine COL- oder eine GDP-Platine verwendet werden. Auch der Anschluß beider gleichzeitig ist möglich. (Fast) sämtliche Grafikbefehle der vorhergehenden FLOMON-Versionen sind auch bei der COL anwendbar, selbst die Wirkung von Registeränderungen wird simuliert. Allerdings wird die COL nur in der Auflösung 256*256 unterstützt; das bedeutet, daß nur 24+1 Zeilen mit je 42 Zeichen genutzt werden können. Manchen Programmen (Leider auch WS!) ist der Bildschirm dann aber zu klein. Da bei der COL der Z80 komplett für den Bildaufbau verantwortlich ist, muß natürlich mit einer Geschwindigkeitseinbuße gegenüber der GDP gerechnet werden.
Im Textmodus können bei der COL für Vordergrund, Hintergrund, Invers und Hintergrund Invers jeweils eigene Farben verwendet werden. Bei einer Hardcopy werden die Farben durch Schattierungen dargestellt.

Soll das System über ein separates Textterminal an der seriellen Schnittstelle bedient werden, wird dies am Fehlen der KEY- Baugruppe erkannt und die Ein/Ausgaben werden umgelenkt. GDP/COL bleiben für die Grafik über die Monitoreinsprünge verfügbar.

2.1 Wie setzt man den Monitor ein

Um korrekt arbeiten zu können, benötigt der Monitor außer 2*8k RAM auf der BANK/BOOT je mindestens eine der folgenden Komponenten (an Stelle der BANK/BOOT kann natürlich auch eine SBC3 o.ä. treten).

- (GDP64k oder GDP64HS oder COL256) und KEY oder
  SER und serielles Terminal.
- auf einer beliebigen Bank (z.B. ROA64) RAM im Bereich F000..FFFF.
- bei CP/M Betrieb mindestens 64Kbyte.

Jede weitere Bank, auf der sich noch erreichbare Programme befinden sollen (GRUNDPROGRAMM etc.), müssen ebenfalls in diesem Bereich mit RAM versehen sein.

Folgende Peripheriebaugruppen werden vom Monitor unterstützt oder benutzt:

- GDP64k
- GDPHS
- KEY
- COL256
- CLUT
- CENT
- SER
- HCOPY/MAUS
- FLO2
- SMART-WATCH

2.2 Technische Daten

Der Monitor ist in Z80-Assembler geschrieben und als Objektcode etwa 16kByte lang. Der Quelltext ist in verschiedene Funktionseinheiten (=Dateien) aufgeteilt und umfaßt insgesamt ca. 600 kByte (sechshundert!); es ergeben sich daraus ca. 1100 Seiten Assemblerlisting.

3.0 Inbetriebnahme

3.1. Umbau auf neue Version

Sollten Sie lediglich eine alte Version des FLOMON gegen FLOMONCG austauschen wollen, dann gehen Sie wie folgt vor:

Schalten Sie den Rechner aus und ziehen Sie die BANK/BOOT heraus.

Überprüfen Sie, ob die Brücken für die 8k-RAMS richtig gesetzt sind und setzen Sie in Fassung 2 und 3 ein 8k-RAM HM6264 o.ä. ein. FLOMONCG/0 kommt in Fassung 0 und FLOMONCG/1 in Fassung 1.
Nachdem Sie noch einmal die korrekte Ausrichtung der Speicher überprüft haben, setzen Sie die Platine wieder ein.

Spätestens einige Sekunden nach dem Einschalten der Versorgungsspannung erscheinen dann die Testmeldungen und schließlich das bekannte Menü. Sollte dies nicht der Fall sein, lesen Sie bitte das Kapitel über die automatische Platinenerkennung durch.

3.2. Erstinstallation

Bei der Erstinstallation des Monitors auf einem gerade fertiggestellten Rechner gehen Sie zunächst ähnlich wie im vorherigen Fall vor.
Vor dem ersten Einschalten muß dafür gesorgt werden, daß auf mindestens einer Bank im Bereich von F000..FFFFH RAM vorhanden ist. Dies kann z.B. mittels einer RAM64, RAM256 oder ROA64 geschehen (evtl. gemeinsam mit GRUNDPROGRAMM auf Bank E).

Des weiteren muß eine Terminalkarte / KEY oder die serielle Schnittstelle vorhanden sein.

Sollte sich der Monitor wider erwarten nicht melden, dann kontrollieren Sie noch einmal den gesamten Aufbau, insbesondere die bei den einzelnen Baugruppen eingestellten I/O-Adressen !

System-Konfigurationsvorschläge:

Eine der wohl wichtigsten Erweiterungen der FLOMONCG Version ist die beliebige Benutzung der Speicherbanken. FLOMONCG konfiguriert sich bei jedem Kaltstart selbst und erkennt die mit Speicher belegten Bänke. Dadurch ist es möglich, den Adreßraum variabel einzusetzen. Unter CP/M- Betrieb muß der Speicher nun nicht mehr auf Bank null, Adresse null liegen.

Programme wie etwa das Z80 Grundprogramm, der SPS- Interpreter, oder der Z80-Zeilenassembler, die auf Eprom vorliegen, können auf einer beliebigen freien Bank eingesetzt werden.Der Start derartiger Programme wird über den Monitor vorgenommen.

Ein Beispiel:

Das Z80 Grundprogramm sitzt auf BANK 0D, Adresse 0. Um das Programm aus dem Floppymonitor heraus zu starten, aktivieren Sie den Monitor und geben den Befehl
G D:0000 ein.
Der Bildschirm erlischt und das Grundprogramm wird gestartet.

Wichtig:
Benutzen Sie den ZEAT- Assembler, so ist es notwendig, den Speicherbereich zu konvertieren. Hierbei beachten Sie bitten den Hinweis unter 6.1.2.

3.2.1. GDP64k oder GDP64HS und KEY

Testen Sie die beiden Karten, soweit in der Platinenbeschreibung angegeben. Stecken Sie die Karten wieder auf den BUS und schließen Sie das Videokabel an. Nach dem Einschalten der Stromversorgung muß sich der Monitor auf dem Bildschirm melden.

3.2.2. COL256 und KEY

Testen Sie die beiden Karten, soweit in der Platinenbeschreibung angegeben. Stellen Sie die Adresse der COL auf 0C000H ein. Vergessen Sie nicht, die im Handbuch der COL angeführte Änderung auf der BANK/BOOT durchzuführen (nur bei Platine r1).
Stecken Sie die Karten wieder auf den BUS und schließen Sie das Videokabel an. Nach dem Einschalten der Stromversorgung muß sich der Monitor auf dem Bildschirm melden.

3.2.3. SER

Einstellungen am Terminal:

9600 baud, 8 bit, 2 stopbits, no parity

Die feste Baudrate wurde gewählt, damit auch ohne eine KEY-Baugruppe ein Terminalbetrieb möglich ist.

Testen Sie die SER-Platine und schließen Sie alle Kabel an. Nach dem Einschalten muß sich auch hier der Monitor auf dem Terminal melden.

3.3. Weitere Baugruppen

Sind die Tests bisher erfolgreich verlaufen, können weitere Baugruppen ins System eingefügt werden. Gehen Sie dabei nach den im entsprechenden Handbuch aufgeführten Anweisungen vor.

3.3.1. SMART-WATCH

Sollten Sie eine SMART-WATCH besitzen, dann setzen Sie sie auf der BANK/BOOT Karte in den Sockel 3.

3.3.2. GDP64k

Beim Betrieb einer GDP64k kann es sein, daß der Bildschirm bei jedem Scrollen kurz flackert. Dieses Flackern läßt sich durch eine minimale Hardwareänderung beheben. Die Zuführung der /WE Leitung zu den RAMS 4164 muß unterbrochen und stattdessen auf J15, Pin 1 gelegt werden. J15, Pin 3 liefert das neue /WE-Signal für die RAM's; Pin 2 und 9 von J15 werden verbunden. Dadurch findet grundsätzlich ein READ-MODIFY-WRITE Zyklus statt; es ist somit möglich, eine Seite zu löschen, während sie noch angezeigt wird. Beim nächsten SYNC kann dann auf eine bereits fertig aufgebaute neue Seite umgeschaltet werden.

Die Bezeichnung der IC's und Pins bezieht sich auf GES-Platinen. Besitzer von anderen Platinen mögen sich die Veränderung anhand des Buches über den NDR-Computer oder ähnlicher Unterlagen auf ihre Verhältnisse übertragen.

3.4. Automatische Platinenerkennung

FLOMONCG prüft bei jedem RESET eine Reihe von Baugruppen auf ihre Anwesenheit. Daraus ergeben sich einige Konventionen, die beachtet werden müssen. Im folgenden wird nur auf Besonderheiten hingewiesen.

3.4.1. Speicher

Damit eine Speicherkarte von FLOMONCG aus erreicht werden kann, muß dieselbe im Bereich von F000..FFFF mit RAM bestückt sein.
Nach POWER-ON-RESET erfolgt ein kompletter Speichertest auf allen erkannten Karten. Nach dem Speichertest enthalten alle Zellen den Wert 0E5h; dieser Wert findet sich auf einer frisch formatierten Diskette. Somit ist jede RAMDISK nach einem Systemstart leer. Die angegebene Fehlerzahl stellt nur die Anzahl der nicht beschreibbaren Zellen dar, beinhaltet also auch bestückte EPROM- Bereiche!
Bei diesem Test wird der Speicher nicht verändert, der Inhalt einer batteriegepufferten RAM-Floppy bleibt erhalten. Das heißt aber auch, daß eine normale RAM-Floppy irgend welche Zufallswerte enthält. Auf ein Löschkammando innerhalb des Monitors wurde bewußt verzichtet, da Größe und Lage der RAM-Floppy unbekannt sind. So bleibt nur die übliche Sequenz in CP/M:

```
'STAT E:*.* $R/W '
'ERA E:*.*'
```

Nach jedem RESET (Tastendruck) erfolgt nur ein Kurztest, um die RAM-Bestückung zu ermitteln und die Belegungstabelle aufzubauen. Die RAM-Floppy wird dabei NICHT zerstört!

3.4.2. KEY

Eine Erkennung der KEY-Platine wurde eingeführt, um auch einen Betrieb mit Terminal über die serielle Schnittstelle zu ermöglichen. Die Platine gilt als vorhanden, wenn am Port KEYSTS ein von 0FFH verschiedener Wert anliegt. Das wird erreicht, indem die DIP-Schalter auf der KEY nicht alle auf OFF liegen.

3.4.3. CENT

Um ein Aufhängen bei nicht vorhandenem Drucker zu vermeiden, wird die Anwesenheit der CENT-Platine (oder entspechender IOE) überprüft. Leider kann aber zwischen nicht vorhandenem und nicht betriebsbereitem Drucker keine Unterscheidung gemacht werden. D.h. im Klartext:

Zum Zeitpunkt der Ausgabe der ersten Zeichens muß der Drucker empfangsbereit sein. Ist er das nicht, dann werden ALLE folgenden Ausgaben auf den Drucker ignoriert.

3.4.4. SMART-WATCH

Wird die SMART-WATCH auf der BANKBOOT- Baugruppe erkannt, wird die Zeit permanent im linken unteren Teil des Bildschirmes eingeblendet. Bei Bedarf kann sie über den Terminaltester deaktiviert werden.

4. Kommandobeschreibung

Sollte im Verlauf des Betriebs einer GDP64k oder GDPHS irgendwann der Bildschirm plötzlich dunkel werden, hat es weder einen Systemabsturz gegeben, noch muß eine Platine defekt sein. Zur Bildschirmschonung wird nach einigen Minuten ohne Zeichenausgabe das Videosignal abgeschaltet. Beim nächsten eintreffenden Zeichen ist die Anzeige wieder unverändert da. Falls das gerade laufende Programm keine Reaktion auf eingegebene Zeichen zeigt, bleibt der Bildschirm dunkel. In diesem Fall tippen Sie am einfachsten ein '^Ö' und rufen den Monitor auf.

4.1. Monitor/Tester

Durch Eingabe der Ziffer '2' kommen Sie vom Menü in den Monitor/Tester. Sie erkennen dies dadurch, daß der Cursor in einem Kasten in der oberen Bildschirmhälfte blinkt und darunter invers der Text 'Monitor' erscheint. Die Kommandos, die Sie jetzt geben können, sind ähnlich denen des Monitors im MC-CP/M- Computer. Sämtliche Eingaben sind mit <RETURN> abzuschließen. Die Ausgabe kann durch Betätigen der Taste '^Ö' jederzeit abgebrochen werden.
Nach dem Starten von CP/M oder ZEAT kann der Monitor jederzeit wieder durch Drücken der Taste ^Ö (läßt sich ändern) aktiviert werden. Auch der Zugriff auf das Notizbuch läuft über einen Aufruf des Monitors.
ACHTUNG: Das Zeichen '^' steht für die 'CONTROLL- Taste'.
Bei der Syntaxbeschreibung sind optionale Teile durch () angegeben, ()* bedeutet beliebige Wiederholung des Klammerinhalts.

4.1.1. '?' : Hilfe oder Berechnung ausführen

Syntax : '?'
'?' Ausdruck (',' Ausdruck)*

Geben Sie ein simples '?' ein. Sie erhalten daraufhin eine Liste mit allen Befehlen und deren Syntax. Kümmern Sie sich nicht darum, wenn ein Teil nach oben verschwindet, den Sie nicht schnell genug lesen konnten. Sie werden später noch lernen, wie Sie den Ausgabebereich beliebig vergrößern können.

Mit dem Befehl '?' können Sie aber auch beliebige Rechnungen ausführen. Als Argument kann eine beliebige Liste von Ausdrücken, getrennt durch ' ' oder ',' angegeben werden.

Beispiel: '? 1'
Ausgabe: '01 #00001 §00000001 "."

^ ASCII-Z. = nicht druckbar
^ Binärdarstellung
^ Dezimaldarstellung
^ Hexdarstellung

ACHTUNG: Bei manchen Tastaturen gilt -> Paragraphzeichen = Klammeraffenzeichen.

Das Ergebnis wird in 4 verschiedenen Darstellungen gedruckt, so daß Sie auf diese Weise zwischen verschiedenen Zahlensystemen umrechnen können; auch das Nachschauen in einer ASCII-Tabelle kann so entfallen.
Als Basis ist standardmäßig 16 eingestellt, jedoch können Werte in einer anderen Darstellung durch ein vorangestelltes Sonderzeichen gekennzeichnet werden.

Beispiel:
```
'? 40'
'? #64'
'? §01000000'
'? "§'
```

Ausgabe jeweils:
```
'40 #00064§ 01000000 "§'
```

Mit '$' kann auf Prozessorregister Bezug genommen werden, der Stand der Register ist der zum Zeitpunkt des Monitoraufrufs (durch Tastendruck, Haltepunkt oder expliziten CALL). Der Inhalt des Stackpointers wird z.B. durch

```
'? $s'
```

angezeigt.

Die gültigen Registerbezeichnungen sind bei dem Befehl 'X' näher beschrieben.

Anstatt einer Konstanten können Sie sich aber auch den Wert eines beliebigen Ausdrucks ausgeben lassen.

Beispiel: '? 1 + 2 * (3 + 4)'

Die Arithmetik wird immer mit 16 bit ausgeführt, ohne Behandlung von Überläufen.
Als Operatoren stehen zur Verfügung (mit abnehmendem Vorrang; x und y bezeichnen Ausdrücke):

Einstellige (unäre) Operatoren:

x ^
liefert den Inhalt der Speicherzellen (x,x+1) als 16 bit- Wert.

x ?
liefert den Inhalt von Port x.

x !
liefert die bitweise Negation von x.

Zweistellige (binäre) Operatoren:

x % y
berechnet x modulo y.

x * y
berechnet das Produkt von x und y.

x / y
berechnet den Quotienten; Priorität wie '*'.

x + y
berechnet die Summe von x und y.

x - y
berechnet die Differenz von x und y; Priorität wie '+'.

x < y
berechnet x * 2 hoch y = SHIFT LEFT.

x > y
berechnet x / 2 hoch y = SHIFT RIGHT; Priorität wie '<'.

x \ y
berechnet das bitweise XOR von x und y.

x & y.
berechnet das bitweise AND von x und y

x ö y.

berechnet das bitweise OR von x und y
Bankoperator: x : y

Der letzte Operator wurde eingeführt, um quer über Bänke arbeiten zu können. Soll etwa der Inhalt der Speicherzellen 2000h, 2001h auf Bank 07 abgefragt werden, dann geht das z.B. mit

'? (7:2000)^'

Die Klammern sind hier notwendig, weil ':' schwächer bindet als der unäre Operator '^'. Ist beim Differenzieren noch keine Bank ausdrücklich angegeben, wird die momentan aktive Bank verwendet.

Weitere Beispiele: '? $s^'
oberster Wert auf dem Stack

'? ($s + 2) & #255
unteres Byte des zweitobersten Wertes auf dem Stack

...

4.1.2. 'B' : aktive Bank wechseln

Syntax: 'B' (n)

Ohne Parameter werden die momentan aktive Bank und alle vom System erkannten Bänke ausgegeben.

Beispiel: 'B'
Ausgabe: '00/03 02 01 00'

Mit Parameter n wird die Wertvorgabe für alle Befehle, die eine Bankangabe verlangen, auf n gesetzt. Wird der Monitor verlassen, dann bleibt Bank n aktiv. Dies sollte UNBEDINGT beachtet werden, da sich sonst leicht ein Systemabsturz provozieren läßt. Vor dem Starten von CP/M muß unbedingt die gleiche Bank wie beim Eintritt in den Monitor angewählt sein.

Beispiel: 'B 0e'
Ausgabe: -

4.1.3. 'C' : Speicherbereiche vergleichen

Syntax: 'C' v b m

Der Speicherbereich von v bis b einschließlich wird mit dem Speicherbereich ab m verglichen, Unterschiede werden ausgegeben.

Beispiel: 'C 1000 1fff 3:4000'
Ausgabe: '00:1010 11 <-> 03:4010 22'
...

4.1.4. 'D' : Speicherbereich anzeigen

Syntax: 'D' (v (b))

Ohne Parameter werden 128 Bytes ab der letzten Adresse eines 'D'- Kommandos in HEX und ASCII-angezeigt.Ist v gegeben, beginnt die Ausgabe des Speicherbereichs bei v; bei Angabe von b wird der Inhalt bis einschließlich b angezeigt.

Beispiel: 'D 100 101'
Ausgabe: '00:0100 40 41 § A'

^ ASCII
^ HEX
^ Adresse
^ Bank

Beispiel: 'D'
Ausgabe: '00:102 00 02 04 08 40 40 ....§§'
...

4.1.5. 'F' : Speicher füllen

Syntax: ' F ' v b m

Der Speicherbereich von v bis b einschließlich wird mit dem Wert m gefüllt.VORSICHT!
Beispiel: 'F 1000 1FFF "A'

Ausgabe: -

4.1.6. 'G' : Programme starten

Syntax: 'G'
'G' s
'G' s b0 (',' b)7

Mit 'G' ohne Parameter wird ein Programm an der Stelle fortgesetzt, an der es unterbrochen wurde oder an der der Monitor aufgerufen wurde. Die aktive Bank muß zuvor wieder korrekt eingestellt worden sein.
Der Parameter s gibt eine Adresse an, an der die Ausführung begonnen oder weitergeführt werden soll. Als Haltepunkte können maximal 8 Adressen b0..b7 angegeben werden, bei deren Erreichen das laufende Programm unterbrochen und der Monitor wieder aufgerufen wird. FLOMONCG verwendet hierfür den Befehl RST30h, in der Hoffnung, daß der zugehörige Vektor nicht von anderen Programmen auch benutzt wird (wie z.B. RST38h).
Erreicht das Programm keinen der Haltepunkte oder ist keiner spezifiziert, dann kann der Monitor nur dann aktiviert werden, wenn das laufende Programm den Tastaturstatus abfragt oder Zeichen einliest (und natürlich durch RESET).

Beispiel: 'G'
'G 1000'
'G 100 110 130 1:200 3:$s^'
Ausgabe: -

Bevor die Kontrolle an das unterbrochene Programm zurückgegeben wird, verschwindet das Fenster, in dem die Monitorausgaben erfolgen. Bei einer Unterbrechung erscheint es wieder. Probleme kann es geben, wenn gerade der Grafikmodus aktiviert ist oder eine unvollständige Steuersequenz ans Terminal geschickt wurde. Normalerweise wird aber durch eine Unterbrechung am Bildschirminhalt nichts verändert.

4.1.7. 'I' : Port abfragen

Syntax: 'I' p

Der Inhalt des Ports p wird ermittelt und ausgegeben. Der Befehl kann auch über '?' simuliert werden.

Beispiel: 'I 60'
Ausgabe: 'FF #00255§11111111 "."

4.1.8. 'M' : Speicher verschieben

Syntax: 'M' v b n

Der Speicherbereich von v bis b einschließlich wird nach n verschoben.

Beispiel: 'M e:2000 3fff 0:2000'
Ausgabe: -

4.1.9. 'N' : Notizbuch

Syntax: 'NE'

Nach dem Befehl 'NE' erscheint ein neuer Bildschirm; am linken unteren Rand erscheint 'Notizbuch'. Jetzt kann ein Text eingegeben und editiert werden, auf den später jederzeit von (fast) jedem Programm aus zugegriffen werden kann. Innerhalb des Textes stehen alle Cursor- Lösch- und Einfügebefehle zur Verfügung, die in der Terminal-Anleitung näher beschrieben sind. Für's erste reichen '^S', '^E', '^D', '^X', die schon von anderen Editoren bekannt sein dürften. Mit <ESC> '$' kommt man wieder auf die normale Kommandoebene.

ACHTUNG: Das Zeichen '^' steht für die 'CONTROLL- Taste'.

```
Das ist ein Notizbuch



          Hier stehen eigene Notizen
```

Notizbuch

Bild 4-1: Notizbuch

Syntax: 'NW'

Mit diesem Befehl wird der Inhalt des obersten Fensters (nach dem Monitor) in das Notizbuch kopiert. Zu diesem Zweck verschwindet kurzzeitig das Monitorfenster. Überzeugen Sie sich von der korrekten Ausführung des Befehls durch ein 'NE'. Jetzt müßte der Bildschirm genauso erscheinen wie ohne Monitorfenster. Nur links unten kündet ein Vermerk 'Notizbuch' von der Transaktion.

Syntax: 'NR'

Das Gegenstück zu 'NW' ist der Befehl 'NR'. Hierbei wird der Inhalt des Notizbuchs an das unterbrochene Programm übergeben. Die auszulesenden Zeichen werden in die Konsoleingabe eingeschleust, bis das Fenster komplett übermittelt ist.
Somit kann die Ausgabe eines beliebigen Programms als Eingabe für ein anderes Programm verwendet werden.

4.1.10. 'O' : Port schreiben

Syntax: 'O' p n

An den Port mit der Adresse p wird der Wert n ausgegeben.

Beispiel: 'O 70 0c'
Ausgabe: Bildschirm flackert

4.1.11. 'Q' : Monitor verlassen

Syntax: 'Q'

Der Monitor wird verlassen und das aufrufende oder unterbrochene Programm wird weiter ausgeführt. Entspricht einem 'G'-Kommando ohne Startadresse und Haltepunkte.

4.1.12. 'S' : Speicherzellen ändern

Syntax: 'S' n

Beginnend bei Speicherzelle n können die Inhalte der folgenden Speicherzellen modifiziert werden. Bei jeder Zelle werden zuerst Adresse und aktueller Inhalt ausgegeben. Als Eingabe wird der neue Wert erwartet. Eine leere Eingabe läßt die Speicherzelle

unverändert. Sollen keine weiteren Zellen mehr geändert werden, braucht nur ein illegaler Wert als Eingabe erfolgen, z.B. '.'. Der Monitor gibt dann ein '?' als Hinweis für einen Fehler aus, der aber nicht weiter zu stören braucht.
Der Wert kann ein beliebiger Ausdruck sein, von dem aber nur die unteren 8 Bit zählen. Zusätzlich kann aber auch ein String, gekennzeichnet durch ein führendes " ' ", eingegeben werden. In diesem Fall werden die einzelnen Zeichen der Reihenfolge nach im Speicher abgelegt.

Beispiel: 'S 100'
Ausgabe: '00:0100 00 #00000§ 00000000 ".->'
Eingabe: ' '
Ausgabe: '00:0101 00 #00000§ 00000000 ".->'
Eingabe: '12'
Ausgabe: '00:0102 00 #00000§ 00000000 ".->'
Eingabe: " 'ABCDE"
Ausgabe: '00:0107 00 #00000§ 00000000 ".->'
Eingabe: '.'
Ausgabe: '?'

4.1.13. 'X' : Register anzeigen und ändern

Syntax: 'X'

Ohne Parameter werden alle Prozessorregister des unterbrochenen Programms angezeigt.

Beispiel: 'X'
Ausgabe:

```
"f=    a=  b=    d=   h=   x    y"
"-Z---- 80  001a  00df 00b8 3544 f38c"
"F=    A'  B'    D'   H'   s    p"
"-Z---- 32  0000  44ef f38c f78c f37f"
```

Hierbei ist folgende Zuordnung zwischen Register und Buchstaben getroffen:

Register	*Buchstabe*	*Register*	*Buchstabe*
A	a	A'	A
F	f	F'	F
BC	b	BC'	B
DE	d	DE'	D
HL	h	HL'	H
IX	x	SP	s
IY	y	PC	p

An Flagbezeichnungen gibt es : M, Z, H,V, N, C.
Ein Flag ist gesetzt, wenn der zugehörige Buchstabe erscheint, es ist nicht gesetzt, wenn ein '-' an der entsprechenden Position steht.

Syntax: 'X' r v

Setzt das Register r auf den Wert v. Als Registerbezeichner sind die gerade erläuterten Buchstaben mit Ausnahme von 'f' und 'F' zugelassen.

Beispiel: 'X h 3*4'
Wirkung: HL:= 12D

Beispiel: 'X s $s+2'
Wirkung: SP:= SP+2

Syntax: 'Xf' fl v
'XF' fl v

Das Flag fl im Register F(''f'') oder F'(''F'') wird auf den Wert v gesetzt. v kann '0' oder '1'sein, ein Rechenausdruck ist hier nicht zugelassen.

Beispiel: 'XfZ0'
Wirkung: setzt das Z-Flag im Register F zurück

Beispiel: 'XFC1'
Wirkung: setzt das C-Flag im Register F'

4.1.14. 'Y' : Liste suchen

Syntax: 'Y' v b l
'Y' v b " " "s
'Y' v b l " " "s

Sucht im Bereich von v bis b nach allen Vorkommen der Liste l. l ist entweder eine Liste von Ausdrücken, getrennt durch ',' oder ' ', oder ein String s, gekennzeichnet durch ein führendes " " ". Auch besteht die Möglichkeit, eine Liste und einen String anzugeben.

Beispiel: "Y 100 200 40 41 42 43 44"
"Y 100 200 '§ABCD"
"Y 100 200 40 "§+1 'CD"
Wirkung: Alle 3 Kommandos suchen den String "ABCD" im Bereich von 100h bis 200h

4.2. Monitoraufruf von CP/M aus

Innerhalb von Anwendungsprogrammen, die unter CP/M oder ZEAT laufen (und möglichst nicht den Grafikmodus benutzen), kann jederzeit der Monitormodus wieder aktiviert werden (sofern irgendwann CSTS oder CI gerufen werden). Es kann dann der Speicher inspiziert oder verändert werden; auch die Benutzung des Notizbuchs ist so möglich. Auch eine Umwandlung DEZ-HEX-ASCII-BIN kann so leicht erfolgen.

Beispiel: Das Inhaltsverzeichnis einer Diskette soll in eine Wordstardatei übernommen werden.

Voraussetzung: Sie sind auf CP/M- Kommandoebene.

Eingabe: DIR<RET>
Ausgabe: <Inhaltsverzeichnis>
Eingabe: ^Ö
Ausgabe: <Monitorfenster erscheint>
Eingabe: NW<RET>
Ausgabe: <Fenster verschwindet kurz>
Eingabe: Q<RET>
Ausgabe: <Monitorfenster verschwindet>
Eingabe: WS TEST
Ausgabe: <Wordstar-Maske>
Eingabe: ^Ö

Ausgabe: <Monitorfenster erscheint>
Eingabe: NR<RET>
Ausgabe: <nach einiger Zeit ist der Bildschirminhalt übernommen>

Bevor Sie das Notizbuch lesen lassen, können Sie es natürlich auch mit 'NE' verändern.

4.3. Textmodus

Vom Menü aus kommen Sie mit '^C' in den Terminaltest, mit einem weiteren ^C kehren Sie in das Menü zurück.Wenn der Terminaltest aktiv ist, werden alle Zeichen von der Tastatur direkt an den Bildschirm geschickt. Sie können so erste Erfahrungen mit der Ansteuerung des Bildschirms sammeln. Die Beispiele sind in BASIC angegeben; eine Übertragung in andere Sprachen fällt Ihnen sicherlich nicht schwer.

ACHTUNG:Einige Basic Dialekte erkennen statt dem einfachen Anführungszeichen nur ein doppeltes, z.b HEBAS.

4.3.1. Cursor nach rechts

04h : ^D

Der Cursor wird um eine Position nach rechts verschoben, am Zeilenende springt er in die nächste Zeile; am Seitenende wird der Bildschirm gescrollt.

Beispiel: PRINT CHR$(4);

4.3.2. Cursor nach oben

05h : ^E

Der Cursor wird um eine Zeile nach oben verschoben. Ist er schon am oberen Rand, erfolgt keine Veränderung.

Beispiel: PRINT CHR$(5);

4.3.3. BELL

07h : ^G

Keine Wirkung.

Beispiel: PRINT CHR$(7);

4.3.4. BACKSPACE; Cursor nach links

08h : ^H

Der Cursor wird um ein Zeichen nach links verschoben, am Zeilenende springt er an den Anfang der vorherigen Zeile. In der obersten Zeile erscheint er wieder am Zeilenende.

Beispiel: PRINT CHR$(8);

4.3.5. TAB; Cursor nach rechts

09h : ^I

Der Cursor wird um eine Position nach rechts verschoben, am Zeilenende springt er in die nächste Zeile; am Seitenende wird der Bildschirm gescrollt.

Beispiel: PRINT CHR$(9);

4.3.6. LINEFEED

0ah : ^J

Der Cursor wird um eine Zeile nach unten bewegt, in der untersten Zeile wird der Bildschirm gescrollt.

Beispiel: PRINT CHR$(10);

4.3.7. Cursor nach oben

0bh : ^K

Der Cursor wird um eine Zeile nach oben verschoben. Ist er schon am oberen Rand, keine Veränderung.

Beispiel: PRINT CHR$(11);

4.3.8. Cursor nach rechts

0ch : ^L

Der Cursor wird um eine Position nach rechts verschoben, am Zeilenende springt er in die nächste Zeile; am Seitenende wird der Bildschirm gescrollt.

Beispiel: PRINT CHR$(12);

4.3.9. CARRIAGE RETURN

0dh : ^M

Der Cursor geht zum Zeilenanfang.

Beispiel: PRINT CHR$(13);

4.3.10. Hardcopy

10h : ^P

Der Bildschirm wird als Hardcopy auf dem Drucker ausgegeben, falls eine Möglichkeit dazu besteht (COL / GDPHS oder HCOPY).

Beispiel: PRINT CHR$(16);

4.3.11. Cursor nach links

13h : ^S

Der Cursor wird um ein Zeichen nach links verschoben, am Zeilenende springt er an den Anfang der vorherigen Zeile. In der obersten Zeile erscheint er wieder am Zeilenende.

Beispiel: PRINT CHR$(19);

4.3.12. Cursor nach unten

16h : ^V

Der Cursor wird um eine Zeile nach unten bewegt. In der untersten Zeile keine Veränderung.

Beispiel: PRINT CHR$(22);

4.3.13. Cursor nach unten

18h : ^X

Der Cursor wird um eine Zeile nach unten bewegt.In der untersten Zeile keine Veränderung.

Beispiel: PRINT CHR$(24);

4.3.14. Bildschirm löschen

1ah : ^Z

Der Bildschirm wird gelöscht,die Statuszeile bleibt erhalten.
Der Cursor steht links oben.

Beispiel: PRINT CHR$(26);

4.3.15. Cursor home

1eh : ^^

Der Cursor geht in die linke obere Ecke.

Beispiel: PRINT CHR$(30);

Escape-Sequenzen im Textmodus

4.3.16. Doppelescape

1bh 1bh 47h : <ESC> <ESC> G

Aufruf des -> Grafikmodus

Beispiel: PRINT CHR$(27)+CHR$(27)+'G';

4.3.17. Datum sichtbar

1bh 20h 30h : <ESC> ' ' '0'

Falls eine SMART-WATCH eingesetzt ist, können Uhrzeit und Datum abgfragt oder in den Bildschirm eingeblendet werden.
Mit diesem Komando wird das Datum eingeblendet, falls es noch nicht sichtbar ist.

Beispiel: PRINT CHR$(27)+' 0`;

4.3.18. Uhrzeit setzen

1bh 20h 31h h h m m s s : <ESC> ' ' '1' h h m m s s

Die Uhrzeit wird gesetzt.Stunden, Minuten und Sekunden als ASCII-Ziffern '0'..'9'.

Beispiel: PRINT CHR$(27)+' 1`+'123027`;

4.3.19. Uhrzeit abfragen

1bh 20h 32h : <ESC> ' ' '2'

Die Uhrzeit wird abgefragt.Format der Ausgabe wie beim Setzen.

Beispiel: PRINT CHR$(27)+' 2';

4.3.20. Uhrzeit unsichtbar

1bh 20h 33h : <ESC> ' ' '3'

Die Uhrzeit wird ausgeblendet, falls sie sichtbar ist.

Beispiel: PRINT CHR$(27)+' 3';

4.3.21. Uhrzeit sichtbar

1bh 20h 34h : <ESC> ' ' '4'

Die Uhrzeit wird eingeblendet, falls sie noch nicht sichtbar ist.

Beispiel: PRINT CHR$(27)+' 4';

4.3.22. Alarmzeit setzen

1bh 20h 35h h h m m s s : <ESC> ' ' '5' h h m m s s

Die Alarmzeit wird gesetzt.
Stunden, Minuten und Sekunden als ASCII-Ziffern '0'..'9'. Achtung: Die Alarmzeit geht beim Ausschalten des Rechners verloren !
Bei Erreichen der Alarmzeit blinkt die Uhrzeit.

Beispiel: PRINT CHR$(27)+' 5'+'201500';

4.3.23. Alarm aus

1bh 20h 36h : <ESC> ' ' '6'

Ein bereits gesetzter oder schon ausgelöster Alarm wird gelöscht.

Beispiel: PRINT CHR$(27)+' 6';

4.3.24. Datum setzen

1bh 20h 37h y y m m d d : <ESC> ' ' '7' y y m m d d

Das Datum wird gesetzt. Jahr, Monat und Tag als ASCII- Ziffern '0'..'9'.

Beispiel: PRINT CHR$(27)+' 7'+'880601';

4.3.25. Datum abfragen

1bh 20h 38h : <ESC> ' ' '8'

Das Datum wird abgefragt. Format der Ausgabe wie beim Setzen.

Beispiel: PRINT CHR$(27)+' 8';

4.3.26. Datum unsichtbar

1bh 20h 39h : <ESC> ' ' '9'

Das Datum wird ausgeblendet, falls es sichtbar ist.

Beispiel: PRINT CHR$(27)+' 9';

4.3.27. Invers AUS

1bh 28h : <ESC> '('

Alle folgenden Zeichen werden normal dargestellt.

Beispiel: PRINT CHR$(27)+'(';

4.3.28. Invers EIN

1bh 29h : <ESC> ')'

Alle folgenden Zeichen werden invers dargestellt.

Beispiel: PRINT CHR$(27)+')';

4.3.29. Bildschirm löschen

1bh 2ah : <ESC> '*'

Der Bildschirm wird gelöscht, die Statuszeile bleibt erhalten.Der Cursor steht links oben.

Beispiel: PRINT CHR$(27)+'*';

4.3.30. Bildschirm löschen

1bh 2bh : <ESC> '+'

Der Bildschirm wird gelöscht, die Statuszeile bleibt erhalten.Der Cursor steht links oben.

Beispiel: PRINT CHR$(27)+'+';

4.3.31. Bildschirm löschen

1bh 2ch : <ESC> ','

Der Bildschirm wird gelöscht, die Statuszeile bleibt erhalten.
Der Cursor steht links oben.

Beispiel: PRINT CHR$(27)+',';

4.3.32. Cursorattribut setzen

1bh 2eh n : <ESC> '.' n

Als Attribute n sind zugelassen:
'0' = kein Cursor
'1' = Block, blinkend
'2' = Block
'3' = Unterstreichung, blinkend
'4' = Unterstreichung

Beispiel: PRINT CHR$(27)+'.1`;

4.3.33. Fadenkreuz-Symbol umdefinieren

1bh 30h <STRING> 0dh : <ESC> '0' <STRING> <CR>

Das Muster für das Fadenkreuz wird umdefiniert. Ist <STRING> leer, dann wird das Standardsymbol (Pfeil) gewählt.
Falls eine GDP64k mit HCOPY angeschlossen ist, wird das Hardware-Fadenkreuz verwendet.
Die Bedeutung der Zeichen in STRING wird in der Erläuterung zum Grafik-Modus beschrieben.

Beispiel: PRINT CHR$(27)+'0`+'1XZÖ^0I1HJLN'

4.3.34. Zeile senden

1bh 34h : <ESC> '4'

Die Zeile, in der der Cursor steht, wird an den Rechner geschickt, abgeschlossen mit CR.

Beispiel: PRINT CHR$(27)+'4';: LINE INPUT L$

4.3.35. Zeile senden

1bh 36h : <ESC> '6'

Die Zeile, in der der Cursor steht,wird an den Rechner geschickt, abgeschlossen mit CR.

Beispiel: PRINT CHR$(27)+'6';: LINE INPUT L$

4.3.36. GDP-Platine anwählen

1bh 38h : <ESC> '8'

Falls eine GDP-Platine angeschlossen ist, wird diese initialisiert und ab jetzt zur Textdarstellung verwendet. Bildschirmformat 84*(24 +1). Bei der Umschaltung wird das Notizbuch gelöscht.

Beispiel: PRINT CHR$(27)+'8';

4.3.37. COL-Platine anwählen

1bh 39h : <ESC> '9'

Falls eine COL-Platine angeschlossen ist, wird diese initialisiert und ab jetzt als Textdarstellung verwendet. Bildschirmformat 42*(24 +1). Bei der Umschaltung wird das Notizbuch gelöscht.

Beispiel: PRINT CHR$(27)+'9';

4.3.38. Bildschirm löschen

1bh 3ah : <ESC> ':'

Der Bildschirm wird gelöscht, die Statuszeile bleibt erhalten. Der Cursor steht links oben.

Beispiel: PRINT CHR$(27)+':';

4.3.39. Bildschirm löschen

1bh 3bh : <ESC> ';'

Der Bildschirm wird gelöscht, die Statuszeile bleibt erhalten. Der Cursor steht links oben.

Beispiel: PRINT CHR$(27)+';';

4.3.40. Tastenklick aus

1bh 3ch : <ESC> '<'

Noch keine Funktion.

Beispiel: PRINT CHR$(27)+'<';

4.3.41. Cursor setzen

1bh 3dh y x : <ESC> '=' <Zeile + 20h> <Spalte + 20h>

Der Cursor wird an die spezifizierte Position gesetzt. Illegale Koordinaten werden ignoriert.

Beispiel: PRINT CHR$(27)+'='+chr$(32+10)+chr$(32+50);

4.3.42. Tastenklick ein

1bh 3eh : <ESC> '>'

Noch keine Funktion.

Beispiel: PRINT CHR$(27)+'>';

4.3.43. Cursor abfragen

1bh 3fh : <ESC> '?'

Antwort : <Zeile + 20h> <Spalte + 20h> <CR>
Beispiel: PRINT CHR$(27)+'?';:Y=ASC(INPUT$(1))-32:
X=ASC(INPUT$(1))-32

4.3.44. Lokal-Modus ein/aus

1bh 44h x : <ESC> 'D' x

Bei x='L' wird der Lokalmodus eingeschaltet, ab jetzt werden von der Tastatur aus keine Zeichen mehr an den Rechner geschickt, sondern direkt ans Terminal (ähnlich Terminaltest). Ein beliebiges anderes x schaltet den Lokalmodus aus.

Beispiel: PRINT CHR$(27)+'DL';

4.3.45. Zeile einfügen

1bh 45h : <ESC> 'E'

Vor der Zeile, in der der Cursor steht, wird eine Leerzeile eingefügt. Die letzte Zeile im Bildschirm wird dafür herausgeschoben.

Beispiel: PRINT CHR$(27)+'E';

4.3.46. Hintergrundfarbe normal setzen.

1bh 4ch x : <ESC> 'L' x

Bei der COL256 besteht die Möglichkeit, für die Textdarstellung 4 Farben zu benutzen: Je eine Farbe für den Hintergrund von inversen und von normalen Zeichen sowie für die Darstellung von normalen und inversen Zeichen.
Mit <ESC> 'L' wird der Hintergrund für normale Zeichen bestimmt. Standardwert x=00h

Beispiel: PRINT CHR$(27)+'L'+chr$(240);: REM ROT

4.3.47. Hintergrundfarbe invers setzen

1bh 4dh x : <ESC> 'M' x

Die Farbe für den Hintergrund bei inversen Zeichen wird gesetzt.
Standardwert x=0cfh

Beispiel: PRINT CHR$(27)+'L'+chr$(204);: REM GRUEN

4.3.48. Zeichen einfügen

1bh 51h : <ESC> 'Q'

Vor dem Zeichen, auf dem der Cursor steht, wird ein Leerzeichen eingefügt.Das letzte Zeichen in der Zeile wird dafür herausgeschoben.

Beispiel: PRINT CHR$(27)+'Q';

4.3.49. Zeile löschen

1bh 52h : <ESC> 'R'

Die Zeile, in der der Cursor steht, wird gelöscht. Alle weiter unten stehenden Zeilen werden nachgeschoben.

Beispiel: PRINT CHR$(27)+'R';

4.3.50. Zeile bis Ende löschen

1bh 54h : <ESC> 'T'

Ab Cursorposition bis Zeilenende werden alle Zeichen gelöscht.

Beispiel: PRINT CHR$(27)+'T';

4.3.51. Monitormodus ein

1bh 55h : <ESC> 'U'

Ab jetzt werden Steuerzeichen nicht mehr ausgeführt, (Ausnahme ENDE Monitormodus) sondern als '<nn>' ausgegeben, wobei nn der ASCII-Code des Steuerzeichens in HEX ist. Dieser Befehl wird verwendet, wenn die Ausgabe eines Programms überprüft werden soll, das selbst den Bildschirm stark verändert oder im Grafikmodus arbeitet.

Beispiel: PRINT CHR$(27)+'U';

4.3.52. Zeichen löschen

1bh 57h : <ESC> 'W'

Das Zeichen unter dem Cursor wird gelöscht. Die restlichen Zeichen der Zeile werden nachgeschoben.

Beispiel: PRINT CHR$(27)+'W';

4.3.53. Monitormodus aus

1bh 58h : <ESC> 'X'

Ab jetzt werden Steuerzeichen wieder ausgeführt.

Beispiel: PRINT CHR$(27)+'X';

4.3.54. Bildschirm bis Ende löschen

1bh 59h : <ESC> 'Y'

Der Bildschirm wird ab der Cursorposition bis zum Ende gelöscht.

Beispiel: PRINT CHR$(27)+'Y';

4.3.55. Farbe normal setzen

1bh 5bh n : <ESC> 'Ä' n

n bestimmt die Farbe, in der normale Zeichen ausgegeben werden.

Beispiel: PRINT CHR$(27)+'Ä'+chr$(195);: REM BLAU

4.3.56. Farbe invers setzen

1bh 5dh n : <ESC> 'Ü' n

n bestimmt die Farbe, in der inverse Zeichen ausgegeben werden.

Beispiel: PRINT CHR$(27)+'Ü'+chr$(255);: REM WEISS

4.3.57. Unterbrechungstaste setzen

1bh 5eh c : <ESC> '^' c

c bestimmt die Taste, bei deren Betätigung der Monitor aufgerufen werden soll; Vorgabe ist ^Ö.

Beispiel: PRINT CHR$(27)+'^'+'^';

4.3.58. Hardcopytaste setzen

1bh 5fh c : <ESC> '_' c

c bestimmt die Taste, bei deren Betätigung eine Hardcopy ausgelöst wird; Vorgabe ist ^§.

Beispiel: PRINT CHR$(27)+'_'+'§';

4.3.59. Hintergrund hell

1bh 62h : <ESC> 'b'

Ab jetzt erfolgt die Darstellung mit dunkler Schrift auf hellem Hintergrund. (nur GDP)

Beispiel: PRINT CHR$(27)+'b';

4.3.60. Hintergrund dunkel

1bh 64h : <ESC> 'd'

Ab jetzt erfolgt die Darstellung mit heller Schrift auf dunklem Hintergrund. (nur GDP)

Beispiel: PRINT CHR$(27)+'d';

4.3.61. Druckermodus ein

1bh 65h : <ESC> 'e'

Ab jetzt arbeitet das Terminal im -> Druckermodus

Beispiel: PRINT CHR$(27)+'e';

4.3.62. Statuszeile setzen

1bh 66h <TEXT> 0dh : <ESC> 'f' <TEXT> <CR>

TEXT wird in die Statuszeile übernommen. Zeichen < 20h (ASCII) werden ignoriert.

Beispiel: PRINT CHR$(27)+'f'+'Statuszeile'+chr$(13);

4.3.63. Invers EIN

1bh 6ah : <ESC> 'j'

Alle folgenden Zeichen werden invers dargestellt.

Beispiel: PRINT CHR$(27)+'j';

4.3.64. Invers AUS

1bh 6bh : <ESC> 'k'

Alle folgenden Zeichen werden normal dargestellt.

Beispiel: PRINT CHR$(27)+'k';

4.3.65. Unterstreichen EIN

1bh 6ch : <ESC> 'l'

Alle folgenden Zeichen werden unterstrichen dargestellt.
Kann nicht zusammen mit inverser Schrift verwendet werden!

Beispiel: PRINT CHR$(27)+'l';

4.3.66. Unterstreichen AUS

1bh 6dh : <ESC> 'm'

Alle folgenden Zeichen werden normal dargestellt.

Beispiel: PRINT CHR$(27)+'m';

4.3.67. Maus ein

1bh 6eh : <ESC> 'n'

Ab jetzt kann die Maus (wenn vorhanden) Aktionen auslösen.

Beispiel: PRINT CHR$(27)+'n';

4.3.68. Maus aus

1bh 6fh : <ESC> 'o'

Ab jetzt kann die Maus keine Aktionen mehr auslösen.
Beispiel: PRINT CHR$(27)+'o';

4.3.69. Zeile bis Ende löschen

1bh 74h : <ESC> 't'

Ab Cursorposition bis Zeilenende werden alle Zeichen gelöscht.

Beispiel: PRINT CHR$(27)+'t';

4.3.70. Monitormodus aus

1bh 58h : <ESC> 'u'

Ab jetzt werden Steuerzeichen wieder ausgeführt.

Beispiel: PRINT CHR$(27)+'u';

4.3.71. Bildschirm bis Ende löschen

1bh 79h : <ESC> 'y'

Der Bildschirm wird ab Cursorposition bis zum Ende gelöscht.

Beispiel: PRINT CHR$(27)+'y';

4.3.72. Zeichensatz wählen

1bh 7ah n : <ESC> 'z' n

n= '0' : US Zeichensatz
n= '1' : Dt Zeichensatz

Beispiel: PRINT CHR$(27)+'z1';: REM dt. Zeichensatz

4.4. Steuerzeichen und Escape-Sequenzen im Druckermodus

Vom Textmodus gelangt man mit <ESC> 'e' in den Druckermodus. Das Terminal verhält sich jetzt ungefähr wie ein EPSON-kompatibler Drucker. Dieser Modus wurde eingeführt, um aus Einzelpunkten bestehende Bilder auch auf dem Terminal sichtbar machen zu können.

Normale ASCII-Zeichen erscheinen auf dem Bildschirm, ohne daß ein Cursor sichtbar ist. Ist eine Seite vollgeschrieben, dann wird der Bildschirm nicht nach oben geschoben, sondern gelöscht. Zuvor wird noch eine Bestätigung vom Benutzer verlangt. Wird als Bestätigung ein '^P' eingegeben, dann wird das Bild, falls möglich, auf dem Drucker als Hardcopy ausgegeben.

4.4.1. BELL

07h : <BEL>

Noch keine Funktion

Beispiel: PRINT CHR$(7);

4.4.2. BACKSPACE

08h : <BS>

Beispiel: PRINT CHR$(8);

4.4.3. LINEFEED

0ah : <LF>

Beispiel: PRINT CHR$(10);

4.4.4. FORMFEED

0ch : <FF>

Eine neue Bildschirmseite wird angefangen

Beispiel: PRINT CHR$(12);

4.4.5. CARRIAGE RETURN

0dh : <CR>

Beispiel: PRINT CHR$(13);

4.4.6. Zeilenabstand Standard

1bh 30h : <ESC> '0'

Beispiel: PRINT CHR$(27)+'0';

4.4.7. Zeilenabstand Standard

1bh 32h : <ESC> '2'

Beispiel: PRINT CHR$(27)+'2';

4.4.8. Druckermodus aus

1bh 40h : <ESC> '§'

Beispiel: PRINT CHR$(27)+'§';: REM Rückkehr in Textmodus

4.4.9. Zeilenabstand setzen

1bh 41h x : <ESC> 'A' x

Der neue Zeilenabstand wird auf x/72 " gesetzt

Beispiel: PRINT CHR$(27)+'A'+chr$(8);: REM für Grafiken

4.4.10. Graphics low-resolution

1bh 4bh n1 n2 <BIT-IMAGE> : <ESC> 'K' n1 n2 <BIT-IMAGE>

Die nächsten (n2 * 256 + n1) Bytes werden als Bitmuster interpretiert und ausgegeben.

Beispiel:
```
PRINT CHR$(27)+'K'+CHR$(0)+CHR$(1);
FOR T=0 TO 255: PRINT CHR$(T);: NEXT T
```

4.4.11. Graphics high-resolution

1bh 4ch n1 n2 <BIT-IMAGE> : <ESC> 'L' n1 n2 <BIT-IMAGE>

Die nächsten (n2 * 256 + n1) Bytes werden als Bitmuster interpretiert und ausgegeben.

Beispiel:
```
PRINT CHR$(27)+'L'+CHR$(0)+CHR$(1);
FOR T=0 TO 255: PRINT CHR$(T);: NEXT T
```

4.5. Fensterkommandos

Ein Fenster oder Window ist ein Bereich innerhalb der Bildschirms, der logisch vom Rest abgetrennt ist. Die unter TEXTMODUS aufgeführten Kommandos beeinflussen (mit einigen Ausnahmen) ausschließlich das aktive Fenster. Der Befehl zum Löschen des Bildschirms wirkt sich also auch nur innerhalb des aktiven Fensters aus, der Rest bleibt unberührt. So ist es z.B. möglich, daß ein laufendes Programm jederzeit unterbrochen werden kann und der Monitor aufgerufen wird, ohne daß der Bildschirm verändert wird. Der Monitor benutzt sein eigenes Fenster.

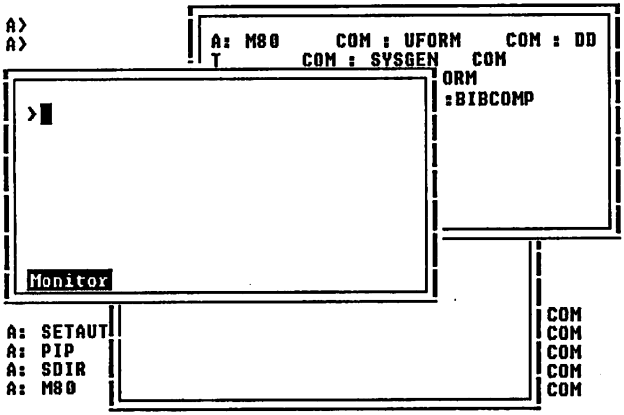


Bild 4-2: Fenstertechnik

Ein Programm braucht sich nicht unbedingt darum zu kümmern, in welches Fenster seine Ausgabe gelangt. Sind allerdings an die Abmessungen bestimmte Anforderungen gestellt (Editor, WS, TURBO, ...), dann kann es natürlich Probleme geben.
Fenster können innerhalb gewisser Grenzen beliebig vergrößert, verkleinert und verschoben werden; Fenster dürfen sich gegenseitig überlappen. Jedes Fenster kann über einen eindeutigen Bezeichner erreicht werden. Die Anzahl der maximal möglichen Fenster ist einerseits durch den Speicherplatz beschränkt, andererseits durch die Eindeutigkeit eines 7 bit-Bezeichners. D.h. maximal können 127 Fenster eröffnet werden, wenn nicht vorher der Speicher überläuft.

Da nicht alle folgenden Kommandos immer ausführbar sind, gibt es eine Möglichkeit, die korrekte Ausführung abzufragen. Auf alle mit '?' gekennzeichneten Kommando- Fragen kann eine Anwort in Form eines ASCII-Zeichens erhalten werden.

'1' : Operation erfolgreich
'0' : Operation nicht erfolgreich

Diese Antwort kann dann z.B. über a$=INPUT$(1) ins Programm übernommen werden.

4.5.1. Fenster- ID's holen

1bh 24h 21h : <ESC> '$' '!'

Die ID's aller vorhandenen Fenster werden an den Rechner gegeben, gefolgt von CR.

Beispiel: PRINT CHR$(27)+'$!';: LINE INPUT A

4.5.2. Parameter holen

1bh 24h 3fh : <ESC> '$' '?'
Die Parameter des aktiven Fensters werden an den Rechner gegeben.
<ID> <Höhe+20h> <Breite+20h> <Y-Offset+20h> <X-Offset + 20h>

Beispiel: PRINT CHR$(27)+'$?';: LINE INPUT A$

4.5.3. Fenster normieren

1bh 24h 40h : <ESC> '$' '§'

Es werden alle Fenster geschlossen. Dann wird Fenster 0 mit der Standardgröße wieder geöffnet.

Beispiel: PRINT CHR$(27)+'$';

4.5.4. Antwort ein

1bh 24h 41h : <ESC> '$' 'A'

Ab jetzt erfolgt auf die mit (?) gekennzeichneten Fenster-Kommandos eine Antwort mit '0' oder '1'. Falls die Operation ok war, '1'.

Beispiel: PRINT CHR$(27)+'$A';

4.5.5. Antwort aus

1bh 24h 42h : <ESC> '$' 'B'

Ab jetzt erfolgt auf die mit (?) gekennzeichneten Fenster- Kommandos keine Antwort.

Beispiel: PRINT CHR$(27)+'$B';

4.5.6. Fenster schließen

1bh 24h 43h : <ESC> '$' 'C'

Das aktive Fenster wird gelöscht. War es das einzige Fenster, dann wird die Initialisierung durchgeführt. Neues aktives Fenster wird das am weitesten vorne liegende.

Beispiel: PRINT CHR$(27)+'$C';

Bild 4-3: Fenster vor dem Schließen

```
FLOMONH V1.0 (C)1987/88 R. Nahm

Test...
Speicherbank 03:0000 Fehler, RAM: 0000..
Speicherbank 02:0000 Fehler, RAM: 0000..
Speicherbank 01:0000 Fehler, RAM: 0000..
Speicherban ┌──────────────────────────────┐
KEY aktiv   │ Dieses Fenster wird jetzt    │
COL aktiv   │ geschlos-                    │
GDP64k akti │ sen                          │
GDPHS nicht │                              │
MAUS aktiv  │                              │
HCOPY aktiv │                              │
CENT aktiv  │                              │
SER nicht a │                              │
Fenster: 60 │                              │
            │                              │
1 =Floppy   │                              │
2 =Monitor  └──────────────────────────────┘
↑C=Terminaltest

FLOMON-Menu
```

Bild 4-4: Fenster nach dem Schließen

```
FLOMONCG V1.0 (C)1987/88 R. Nahm
Test...
Speicherbank 0E:0000 Fehler, RAM: 0000..
Speicherbank 03:0000 Fehler, RAM: 0000..
Speicherbank 02:0000 Fehler, RAM: 0000..
Speicherbank 01:0000 Fehler, RAM: 0000..
Speicherbank 00:0000 Fehler, RAM: 0000..
KEY aktiv
COL aktiv
GDP64k aktiv
GDPHS nicht aktiv
MAUS aktiv
HCOPY aktiv
CENT aktiv
SER nicht aktiv
Fenster: 6086..7FFF

1 =Floppy Boot
2 =Monitor
↑C=Terminaltest
▌

FLOMON-Menu
```

4.5.7. Fenster verschieben (?)

1bh 24h 44h : <ESC> '$' 'D'

Das aktive Fenster wird um eine Zeile nach unten verschoben, sofern es nicht schon am unteren Rand ist.

Beispiel: PRINT CHR$(27)+'$D';

4.5.8. Fenster sichtbar

1bh 24h 45h : <ESC> '$' 'E'

Das aktive Fenster wird ab jetzt sichtbar.

Beispiel: PRINT CHR$(27)+'$E';

4.5.9. Fenster unsichtbar

1bh 24h 46h : <ESC> '$' 'F'

Das aktive Fenster wird ab jetzt unsichtbar.

Beispiel: PRINT CHR$(27)+'$F';

4.5.10. Fenster in den Hintergrund

1bh 24h 47h : <ESC> '$' 'G'

Das aktive Fenster wird in den Hintergrund geschoben. Neues aktives Fenster wird das dann am weitesten vorne liegende Fenster.

Beispiel: PRINT CHR$(27)+'$G';

4.5.11. Position neu setzen (?)

1bh 24h 49h y x : <ESC> '$' 'I' <Y-Offset + 20h> <X-Offset + 20h>

Der Offset des aktiven Fensters wird neu gesetzt. Bei illegalen Parametern keine Veränderung.

Beispiel: PRINT CHR$(27)+'$I'+CHR$(32+2)+CHR$(32+10);

4.5.12. Fenster fixieren

1bh 24h 4ah : <ESC> '$' 'J'

Ab jetzt kann durch die Maus keine Verschiebung, kein Öffnen ..., erfolgen. Das aktuelle Fenster bleibt unverändert.

Beispiel: PRINT CHR$(27)+'$J';

4.5.13. Fenster lösen

1bh 24h 4bh : <ESC> '$' 'K'

Ab jetzt kann die Maus wieder das aktuelle Fenster verschieben.

Beispiel: PRINT CHR$(27)+'$K';

4.5.14. Fenster verschieben (?)

1bh 24h 4ch : <ESC> '$' 'L'

Das aktive Fenster wird um ein Zeichen nach links verschoben, sofern es nicht schon am linken Rand ist.

Beispiel: PRINT CHR$(27)+'$L';

4.5.15. Menüfenster setzen (?)

1bh 24h 4dh id <n+ 20h> : <ESC> '$' 'M' id <n+20h>

Das aktuelle Fenster wird zum Menüfenster Nummer n für Fenster ID. (-> Mausbeschreibung) Fehler, falls Fenster id noch nicht existiert.

Beispiel: PRINT CHR$(27)+'$M'+'1'+chr$(32+0);

4.5.16. Fenster eröffnen (?)

1bh 24h 4fh id <y +20h> <x+20h> <dy+20h> <dx+20h> : <ESC> '$' 'O' id y x dy dx

Fenster id wird unsichtbar eröffnet; y= Anzahl der Zeilen; x= Anzahl der Spalten; dy= Offset in Zeilen; dx= Offset in Spalten. Das Fenster ist leer, der Cursor links oben. Fehler, falls illegale Parameter oder nicht genügend Speicherplatz.

Beispiel:
```
PRINT CHR$(27)+'$O1';
PRINT chr$(32+24)+chr$(32+80)+chr$(32+0)+chr$(32+0);
REM Fenster mit Abmessungen wie bisheriger Bildschirm
```

```
FLOMONCG V1.0 (C)1987/88 R. Nahm

Test...
Speicherbank 03: 0000 Fehler, RAM: 0000..
Speicherbank 02: 0000 Fehler, RAM: 0000..
Speicherbank 01: 0000 Fehler, RAM: 0000..
Speicherbank 00: 0000 Fehler, RAM: 0000..
KEY aktiv
COL aktiv
GDP64k aktiv
GDPHS nicht aktiv
MAUS aktiv
HCOPY aktiv
CENT aktiv
SER nicht aktiv
Fenster: 6086..7FFF

1 =Floppy Boot
2 =Monitor
↑C=Terminaltest
▌

FLOMON-Menu
```

Bild 4-5: Fenster vor dem Öffnen

```
FLOMONCG V1.0 (C)1987/88 R. Nahm

Test...
Speicherbank 03:0000 Fehler, RAM: 0000..
Speicherbank 02:0000 Fehler, RAM: 0000..
Speicherbank 01:0000 Fehler, RAM: 0000..
Speicherbank 00:0000 Fehler, RAM: 0000
KEY aktiv
COL aktiv
GDP64k akti
GDPHS nicht
MAUS aktiv
HCOPY aktiv
CENT aktiv
SER nicht a
Fenster: 6(

1 =Floppy E
2 =Monitor
↑C=Terminaltest
```

FLOMON-Menu

Bild 4-6: Fenster nach dem Öffnen

4.5.17. Bedingung setzen

1bh 24h 50h <y+20h> <x+20h> <STRING> 0dh :

<ESC> '$' 'P' <y+20h> <x+20h> <STRING> <CR>

String muß an Position y, x stehen, damit das aktive Fenster als Menüfenster mit der Maus eröffnet werden kann. (siehe Mausbeschreibung)

Beispiel: PRINT CHR$(27)+'$P'+CHR$(32+0)+CHR$(32+10);
PRINT ':'+chr$(13);:REM Spezifisch für WORDSTAR

4.5.18. Fenster verschieben (?)

1bh 24h 52h : <ESC> '$' 'R'

Das aktive Fenster wird um ein Zeichen nach rechts verschoben, sofern es nicht schon am rechten Rand ist.

Beispiel: PRINT CHR$(27)+'$R';

4.5.19. Fenster anwählen (?)

1bh 24h 53h id : <ESC> '$' 'S' id

Fenster n wird in den Vordergrund gebracht.
Alle folgenden Ausgaben erfolgen in diesem Fenster;

Beispiel: PRINT CHR$(27)+'$S0';

4.5.20. Fenster verschieben (?)

1bh 24h 55h : <ESC> '$' 'U'

Das aktive Fenster wird um eine Zeile nach oben verschoben, sofern es nicht schon am oberen Rand ist.

Beispiel: PRINT CHR$(27)+'$U';

4.5.21. Notizbuch anwählen

1bh 24h 56h : <ESC> '$' 'V'

Das Notizbuch wird sichtbar. Ab jetzt dürfen KEINE weiteren Fensterkommandos folgen, solange bis es wieder unsichtbar ist !!

Beispiel: PRINT CHR$(27)+'$V';

4.5.22. Notizbuch unsichtbar

1bh 24h 57h : <ESC> '$' 'W'

Das Notizbuch wird unsichtbar. Jetzt dürfen wieder Fensterkommandos folgen.

Beispiel: PRINT CHR$(27)+'$X';

4.5.23. Notizbuch schreiben

1bh 24h 58h : <ESC> '$' 'X'

Der Inhalt des aktiven Fensters wird in das Notizbuch kopiert.

Beispiel: PRINT CHR$(27)+'$X';

4.5.24. Notizbuch lesen

1bh 24h 59h : <ESC> '$' 'Y'

Der Inhalt des Notizbuchs wird in die Konsoleingabe eingeschleust.

Beispiel: PRINT CHR$(27)+'$Y';

4.5.25. Fenster vergrößern (?)

1bh 24h 64h : <ESC> '$' 'd'

Das aktive Fenster wird um eine Zeile nach unten vergrößert. Fehler, falls am unteren Bildschirmrand oder nicht genügend Speicher.

Beispiel: PRINT CHR$(27)+'$d';

4.5.26. Fenstergröße setzen (?)

1bh 24h 69h <y+20h> <x+20h> : <ESC> '$' 'i' <y+20h> <x+20h>

Die Größe des aktiven Fensters wird neu gesetzt. Fehler, falls illegale Parameter oder nicht genügend Speicher.

Beispiel: PRINT CHR$(27)+'$i'+CHR$(32+24)+CHR$(32+80);

4.5.27. Fenster verkleinern (?)

1bh 24h 6ch : <ESC> '$' 'l'

Das aktive Fenster wird um ein Zeichen auf der rechten Seite verkleinert. Fehler, falls Fenster zu klein oder nicht genügend Speicher.

Beispiel: PRINT CHR$(27)+'$l';

4.5.28. Fenster vergrößern (?)

1bh 24h 72h : <ESC> '$' 'r'

Das aktive Fenster wird um ein Zeichen nach rechts vergrößert. Fehler, falls am rechten Bildschirmrand oder nicht genügend Speicher.

Beispiel: PRINT CHR$(27)+'$r';

4.5.29. Fenster verkleinern (?)

1bh 24h 75h : <ESC> '$' 'u'

Das aktive Fenster wird um eine Zeile auf der Unterseite verkleinert. Fehler, falls Fenster zu klein oder nicht genügend Speicher.

Beispiel: PRINT CHR$(27)+'$u';

4.6. Grafik-Modus

Durch Eingabe von <ESC> <ESC> 'G' gelangt man vom Textmodus in den Grafik-Modus. Der Cursor verschwindet, der Bildschirm bleibt aber vorläufig noch unverändert. Wird der Grafikmodus im Terminaltest aufgerufen, dann werden alle eingegeben Zeichen zur Kontrolle in der obersten Zeile ausgegeben. Dies ist ein wesentlicher Vorteil gegenüber früheren FLOMON-Versionen, bei denen grundsätzlich blind getippt werden mußte.

Im Grafikmodus beträgt die Auflösung bei einer GDP-Platine 512*256, bei einer COL256- 256*256 Punkte. Der Koordinatenursprung ist links unten. Bei einer GDP stehen 4 Bildschirmseiten zur Verfügung, die alternativ angezeigt werden können; bei einer COL ist normalerweise nur eine Seite verfügbar.

Zahlenwerte können Dezimal oder Hexadezimal eingegeben werden. Auch negative Werte sind erlaubt.

Beispiel: 100, $FF, -50, -$0F

Soll ein Wert an einer Stelle, wo eine Absolutangabe stehen müßte, als relativ zur aktuellen Koordinate behandelt werden, wird ein '%' vorangestellt.

Beispiel: 'D%30 %50;'
Ausgabe: Linie von (x0,y0) nach (x0+30,y0+50)

Aus Platzgründen wurde die Syntaxprüfung vereinfacht, d.h. bei unsinnigen Angaben oder fehlenden Parametern wird das Kommando nicht unbedingt ignoriert, sondern mit willkürlichen Werten ausgeführt.
An Stellen, wo in der Kommandobeschreibung ein ';' erscheint, kann genauso gut auch <CR> verwendet werden, aber nicht umgekehrt!

4.6.1. Hardcopy

10h : ^P

Es wird eine Hardcopy auf den Drucker ausgegeben.

Beispiel: PRINT CHR$(16);

4.6.2. Linie zeichnen, Binärkoordinaten

64h xh xl yh yl : 'd' xh xl yh yl

Von der aktuellen Koordinate aus wird eine Linie zum Punkt (xh*256 + xl, yh*256 + yl) gezeichnet.

Beispiel:
```
PRINT 'd';CHR$(dx/256);CHR$(dx - dx/256);
PRINT CHR$(dy/256);CHR$(dy-dy/256);
```

4.6.3. Dreieck füllen

6ch x0 y0 x1 y1 x2 y2 3bh : 'l' x0 y0 x1 y1 x2 y2 ';'

Das durch die drei Punkte (x0,y0), (x1,y1) und (x2,y2) bestimmte Dreieck wird ausgefüllt.

Beispiel:
```
PRINT 'l0 0 100 10 10 100;';
```

```
l10 10 400 50 50 200;
```

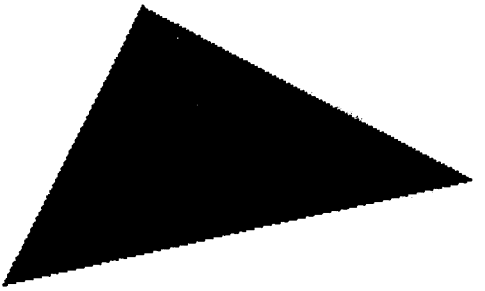


Bild 4-7: Dreieck füllen

4.6.4. Position setzen, Binärkoordinaten

6dh xh xl yh yl : 'm' xh xl yh yl

Die aktuelle Position wird auf (xh*256 + xl, yh*256 + yl) gesetzt.

Beispiel:
```
PRINT 'm';CHR$(x/256);CHR$(x - x/256);
PRINT CHR$(y/256);CHR$(y-y/256);
```

4.6.5. RMW-Modus setzen

72h m : 'r' m

Der Modus beim Zeichnen wird auf m gesetzt. Dieses Kommando hat keine Auswirkung bei einer GDP64k. RMW bedeutet, daß beim Setzen eines Punktes der alte Helligkeits- oder Farbwert mit dem neuen über XOR verknüpft wird. Wird der Punkt ein zweites Mal gesetzt, ist der Originalzustand wiederhergestellt.

m= 00h: kein RMW
m= 01h: RMW

Beispiel:
```
PRINT 'r'+CHR$(1);
```

4.6.6. Grafikmodus verlassen

41h : 'A'

Der Grafikmodus wird verlassen; der Bildschirm wird wieder auf den Zustand vor dem Aufruf gebracht, alle gezeichneten Linien verschwinden.

Beispiel:
```
PRINT 'A';
```

4.6.7. Text ausgeben

42h text 0dh : 'B' text <CR>

An der aktuellen Position wird text ausgegeben. Zeichen <20h werden ignoriert.

Beispiel: PRINT 'Dies ist ein Text'+CHR$(13)

Bild 4-8: Text ausgeben

4.6.8. Seite löschen

43h : 'C'

Die aktuelle Schreibseite wird gelöscht.

Beispiel: PRINT 'C';

4.6.9. Linie zeichnen

44h x y 3bh : 'D' x y ';'

Es wird eine Linie von der aktuellen Position nach (x,y) gezeichnet.

Beispiel: PRINT 'D100 200;';

4.6.10. Fadenkreuz setzen

46h x y s 3bh : 'F' x y s ';'

Es wird ein Fadenkreuz an Position (x, y) auf Seite s gesetzt. Ein bereits vorher gesetztes Fadenkreuz wird entfernt.
Aktuelle Position und Seite werden nicht verändert.

Beispiel: PRINT 'F100 200 0;';

4.6.11. GDP-Register setzen

47h r i 3bh : 'G' r i ';'

Bei einer GDPxx-Platine werden die Register des Prozessors EF9366 gesetzt. Welche Wirkung damit ausgelöst wird, kann aus dem zugehörigen Datenblatt entnommen werden. Im Falle einer COL256 hat der Befehl aber die gleiche Wirkung, da der Prozessor simuliert werden kann.

Beispiel: PRINT 'G3 $22;';
REM Schrift doppelt breit und hoch

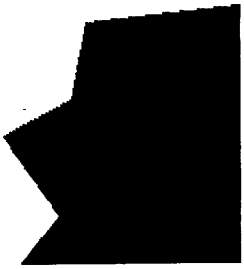
4.6.12. Füllmodus X-Achse

48h 3bh : 'H;'
48h y 3bh : 'H' y ';'

Ab jetzt wird bei den Kommandos 'D' und 'J' jeder gesetzte Punkt (x0 ,y0) mit einem Punkt (x0,y) verbunden. Ohne Angabe von y wird der Füllmodus abgeschaltet.

Beispiel: PRINT 'H 0;'
REM alle Punkte mit dem unteren Rand verbunden

```
I200;M10 10;D40 40;D20 100;D70 120;D80 190;D190 200;
```



4.6.13. Füllmodus Y-Achse

49h 3bh : 'H;'
49h x 3bh : 'I' x ';'

Ab jetzt wird bei den Kommandos 'D' und 'J' jeder gesetzte Punkt (x0, y0) mit einem Punkt (x,y0) verbunden.
Ohne Angabe von x wird der Füllmodus abgeschaltet.

Beispiel: PRINT 'I 0;'
REM alle Punkte mit dem linken Rand verbunden

```
H20;M100 100;D200 150;D230 100;D250 120;D300 40;
```

Bild 4-10: Füllmodus y-Achse

4.6.14. Linie zeichnen, Relativkoordinaten

4ah dx dy 3bh : 'J' dx dy ';'

Es wird eine Linie von der aktuellen Position (x0,y0) nach (x0+dx, y0+dy) gezeichnet.

Beispiel: PRINT 'J10 10;';

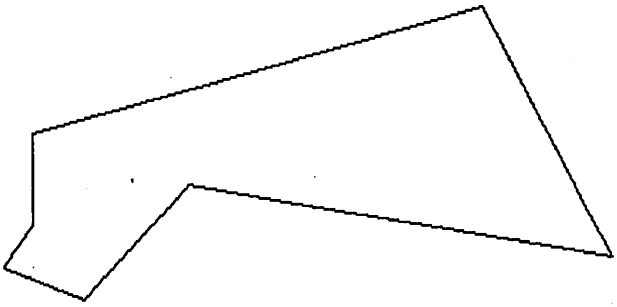
4.6.15. Polygon zeichnen

4ch x0 y0 .. xn yn 3bh : 'L' x0 y0 .. xn yn ';'

Die Punkte (x0,y0) bis (xn,yn) werden der Reihe nach verbunden.
Von (xn,yn) wird wieder eine Linie nach (x0 y0) gezeichnet.
Somit entsteht ein Polygon oder (n+1)-Eck.

Beispiel: PRINT 'L0 0 100 10 20 30 100 0;';

```
L30 30 50 50 50 100 400 200 500 10 200 80 100 20;
```



4.6.16. Position setzen

4dh x y 3bh : 'M' x y ';'

Die aktuelle Position wird auf (x,y) gesetzt.

Beispiel: PRINT 'M0 0;';

4.6.17. Ellipsensegment zeichnen

4fh rx ry phi0 phi1 3bh : 'O' rx ry phi0 phi1 ';'
4fh rx ry phi0 phi1 31h 3bh : 'O' rx ry phi0 phi1 '1' ';'

Es wird an der aktuellen Position ein Ellipsensegment gezeichnet. Länge der Halbachse in x-Richtung=rx; Länge der Halbachse in y-Richtung=ry; Startwinkel phi0, Endwinkel phi1. Ist als Option '1' angegeben, dann wird der vom Radius überstrichene Bereich aufgefüllt (Torte).

Beispiel: PRINT 'O50 20 0 360;';

```
M200 100;0100 50 0 270 1;
```

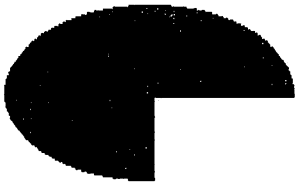


Bild 4-12: Ellipsensegment

4.6.18. Seite setzen, asynchron

50h s 3bh : 'P' s ';'

Schreib- und Leseseite werden sofort gesetzt. s errechnet sich zu: Leseseite+ 4*Schreibseite.

Beispiel: PRINT 'P13;';
REM Schreibseite 3, Leseseite 1

4.6.19. Stack löschen

51h : 'Q'

Der Koordinatenstack wird gelöscht.

Beispiel: PRINT 'Q';

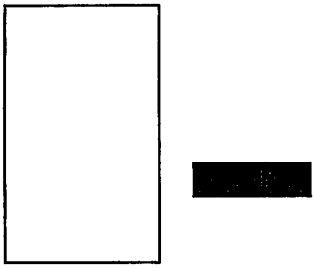
4.6.20. Rechteck zeichnen

52h dx dy 3bh : 'R' dx dy ';'
52h dx dy 31h 3bh : 'R' dx dy '1' ';'

Es wird ein Rechteck mit der aktuellen Position in die linke untere Ecke, Breite dx und Höhe dy, gezeichnet. Mit der Option '1' wird der Bereich innerhalb des Rechtecks ausgefüllt.

Beispiel: PRINT 'R100 50 1;';

```
M10 10;R100 200;M120 50;R80 20 1;
```



4.6.21. Seite setzen, synchron

51h s 3bh : 'S' s ';'

Schreib- und Leseseite werden außerhalb des Anzeigezyklus (und somit flackerfrei) gesetzt. s errechnet sich zu: Leseseite+ 4*Schreibseite.

Beispiel: PRINT 'S10;';
REM Schreibseite 2, Leseseite 2

4.6.22. Push Position

52h : 'T'

Die aktuelle Position wird auf den Koordinatenstack gebracht, falls noch Platz ist. Der Koordinatenstack liegt zwischen dem Bereich für die Textfenster und 8000h; d.h. je mehr Fenster offen sind, desto weniger Platz.

Beispiel: PRINT 'T';

4.6.23. Pop Position

53h : 'U'

Die aktuelle Position wird auf den obersten Wert gesetzt, der sich noch auf dem Koordinatenstack befindet. Bei leerem Stack keine Veränderung. Das oberste Element wird entfernt. +

Beispiel: PRINT 'U';

4.6.24. Kommando an GDP

54h STRING 00h : 'V' STRING <NUL>

Die einzelnen Bytes in String werden als Kommando unmittelbar an den GDP geschickt. Im Falle der COL256 wird die Wirkung des Kommandos durch die Software simuliert.

Beispiel: PRINT 'V'+CHR$(4)+CHR$(0);
REM Schirm löschen

4.6.25. Seitenwechsel

58h n 3bh : 'X' n ';'
58h 3bh : 'X' ';'

Die Seiten 0..3 erscheinen der Reihe nach jeweils für n*20ms als Leseseite; nach Seite 3 kommt wieder Seite 0. Ist n=0 oder fehlt n, dann wird der Seitenwechsel beendet. Die eingestellte Leseseite ist dann zufällig. Die gewünschte Wechselrate kann nur eingehalten werden, wenn der Z80 mindestens alle 20ms dieentsprechende Schleife durchläuft. Fragt ein Anwenderprogramm zu selten oder gar nicht CSTS ab, dann wird der Seitenwechsel im Zeitverhalten dadurch gestört.

Beispiel: PRINT 'X1;';

4.6.26. Seitenwechsel

59h n 3bh : 'Y' n ';'
59h 3bh : 'Y' ';'

Die Seiten 0/1 oder 2/3 (je nach der zuvor eingestellten Leseseite) erscheinen abwechselnd für jeweils n*20ms als Leseseite.
Ist n=0 oder fehlt n, dann wird der Seitenwechsel beendet. Die eingestellte Leseseite ist dann zufällig. Die gewünschte Wechselrate kann nur eingehalten werden, wenn der Z80 mindestens alle 20ms die entspechende Schleife durchläuft. Fragt ein Anwenderprogramm zu selten oder gar nicht CSTS ab, dann wird der Seiten wechsel im Zeitverhalten dadurch gestört.

Beispiel: PRINT.'Y1;';

4.6.27. Fadenkreuz-Symbol umdefinieren

57h 41h STRING 0dh : 'WA' STRING <CR>

Das Symbol für das Fadenkreuz wird umdefiniert. Die einzelnen Zeichen in STRING definieren durch Angabe von Richtung und Länge eine Figur.
Im einzelnen haben die Zeichen folgende Bedeutung:

'0' = Stift hoch, '1'=Stift runter

```
   Ä     Z     Y        Länge 3
    S    R    Q         Länge 2
      K  J  I           Länge 1
        DB A            Länge 0
Ö  T L D  § H P X
        EF G            Länge 0
      M  N  O           Länge 1
     U   V   W          Länge 2
   Ü         ^_         Länge 3
```

Das Diagramm veranschaulicht die Zuordnung von Zeichen und Vektor. Der zu einem Vektor gehörende Buchstabe läßt sich aber auch errechnen:

B:= Richtung + (8*Länge) + 40h

wobei Länge=0..3 und Richtung=0..7 (*45 Grad). Ein Vektor der Länge 2 nach links oben hätte demnach den Code

3+ (8*2) + 40h = 53h = 'S'

Falls STRING leer ist, wird das Standardsymbol gewählt.
Nachdem Befehl 'WA'... ist das Fadenkreuz unsichtbar.

Beispiel: PRINT 'WA' + '1^0X1ÄX0^1_' + CHR$(13);
REM Standardmäßig definierter Pfeil

4.6.28. Fadenkreuz-Symbol zeichnen

57h 42h : 'WB'

Das Fadenkreuzsymbol wird an der aktuellen Position auf der aktuellen Schreibseite gezeichnet. Keine Auswirkung auf ein evtl. definiertes Fadenkreuz.

Beispiel: PRINT 'WB';

4.6.29. Fadenkreuz-Symbol drehen

57h 43h f d 3bh : 'WC' f d ';'
57h 43h 3bh : 'WC;'

Das Fadenkreuzsymbol wird um den Winkel d * 45 Grad nach links gedreht und auf den Faktor f vergrößert.Fehlen die Angaben f und d, wird der Ausgangszustand f=1, d=0 hergestellt.
Nach diesem Befehl ist das Fadenkreuz unsichtbar.

Beispiel: PRINT 'WC 2 1';
REM doppelt groß und um 45 Grad nach links

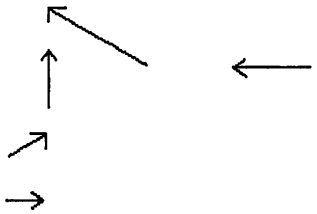


Bild 4-14: Fadenkreuz

4.7. Monitorschnittstelle

Genau wie alle anderen vorhergehenden Versionen bietet FLOMONCG dem Benutzer eine Schnittstelle, über die er Funktionen des Monitors aufrufen kann. Der Umfang dieser Funktionen hat sich weiter vergrößert, etwa um Sprünge zwischen den Bänken und um eine globale Speicherverwaltung. Im Gegensatz zu früher, können die Monitorfunktionen von einer BELIEBIGEN Bank aus aufgerufen werden; die notwendige Sprungleiste ist auf allen Bänken vorhanden, die im Bereich von f000..ffff RAM haben.

Die Register A und F werden grundsätzlich zerstört bzw. als Ausgaberegister verwendet. Alle anderen nicht angegebenen Register bleiben unverändert.

4.7.1. Kaltstart

Einsprungadresse: 0F000h
Eingabeparameter: -
Ausgabeparameter: -

Es wird die gleiche Befehlssequenz wie nach einem RESET ausgeführt, d.h. Überprüfung der Systemkomponenten, Initialisierung aller benutzen Baugruppen etc. Es erfolgt keine Rückkehr zum aufrufenden Programm.

4.7.2. Zeichen von Tastatur

Einsprungadresse: 0F003h
Eingabeparameter: -
Ausgabeparameter: A= eingelesenes Zeichen

Normalerweise wird hier ein Zeichen aus dem Tastaturpuffer gelesen.In Sonderfällen (->Maus setzt Cursor, ->Notizbuch lesen) ist die Quelle der Zeichen auch eine andere.

4.7.3. Zeichen von serieller Schnittstelle

Einsprungadresse: 0F006h
Eingabeparameter: -
Ausgabeparameter: A= eingelesenes Zeichen

Es wird ein Zeichen von der seriellen Schnittstelle eingelesen. Ist die SER-Platine nicht angeschlossen, dann ist das Ergebnis A=1ah=^Z; ist dagegen die Platine vorhanden und kein Peripheriegerät angeschlossen, kommt der Rechner in eine Endlosschleife.

4.7.4. Zeichen an Bildschirm

Einsprungadresse: 0F009h
Eingabeparameter: C= auszugebendes Zeichen
Ausgabeparameter: -

Das Zeichen wird an die Terminal-Software übergeben.

4.7.5. Zeichen an serielle Schnittstelle

Einsprungadresse: 0F00Ch
Eingabeparameter: C= auszugebendes Zeichen
Ausgabeparameter: -

Das Zeichen wird über die serielle Schnittstelle ausgegeben; es ist kein XON/XOFF oder ETX/ACK Protokoll implementiert. Ist die SER-Platine nicht angeschlossen, hat der Aufruf keine Wirkung; ist jedoch kein Peripheriegerät angeschlossen, kommt der Rechner in eine Endlosschleife.

4.7.6. Zeichen an Drucker

Einsprungadresse: 0F00Fh
Eingabeparameter: C= auszugebendes Zeichen
Ausgabeparameter: -

Das Zeichen wird über die Centronics-Schnittstelle an einen Drucker ausgegeben. Ist die CENT- Platine (o.ä.) nicht vorhanden oder war der Drucker zum Zeitpunkt des letzten RESET nicht bereit, dann hat der Aufruf keine Wirkung.

4.7.7. Eingabestatus

Einsprungadresse: 0F012h
Eingabeparameter: -
Ausgabeparameter: A= 0ffh, falls Zeichen da
A= 0 sonst
F= NZ, falls Zeichen da
F= Z sonst

Es wird abgefragt, ob ein Zeichen im Eingabepuffer vorhanden ist.

4.7.8. IOBYTE holen

Einsprungadresse: 0F015h
Eingabeparameter: -
Ausgabeparameter: A= 0

Funktion ist nicht implementiert.

4.7.9. IOBYTE setzen

Einsprungadresse: 0F018h
Eingabeparameter: -
Ausgabeparameter: -

Funktion ist nicht implementiert.

4.7.10. Speicherobergrenze ermitteln

Einsprungadresse: 0F01Bh
Eingabeparameter: -
Ausgabeparameter: A= MSB
B= LSB

Die Funktion ermittelt die höchste Speicheradresse, die ein Benutzerprogramm noch belegen darf (=0EFFFh).

4.7.11. Monitor aufrufen

Einsprungadresse: 0F01Eh
Eingabeparameter: -
Ausgabeparameter: -

Der Kommandointerpreter des Monitors wird aufgerufen.
Alle Register bleiben erhalten, wenn sie nicht explizit vom Benutzer geändert werden. Kommandos siehe ->Monitor.

4.7.12. Floppy-Bedienung

Einsprungadresse: 0F021h

Eingabeparameter: B=0
D: Bit 0..1 Steprate
Bit 7 = 0, dann kein Seitenvergleich
Bit 7 = 1, dann Seitenvergleich

Ausgabeparameter: -

Eingabeparameter: B=1 Sektor lesen
2 Sektor schreiben

C: Bit 3..0 = 1 für LW 1
= 2 für LW 2
= 4 für LW 3
= 8 für LW 4
Bit 4 = 0 für DD
= 1 für SD

Bit 5 = 0 für 8"
= 1 für 5 1/4"

Bit 6 = 0 für Motor ein
= 1 für Motor aus

Bit 7 = Seite

D = Spurnummer
E = Sektornummer
HL = Quell- oder Zieladresse

Ausgabeparameter: A= 0FFh, falls Fehler
= 0 sonst
F= NZ,C, falls Fehler
= Z,NC sonst

Mit diesem Einsprung kann auf die Diskettenlaufwerke zugegriffen werden.

4.7.13. Floppy-Bedienung, 8"

Einsprungadresse: 0F024h

Eingabeparameter: B=0
D: Bit 0..1 Steprate
Bit 7 = 0, dann kein Seitenvergleich
Bit 7 = 1, dann Seitenvergleich

Ausgabeparameter: -

Eingabeparameter: B=1 Sektor lesen
2 Sektor schreiben

C: Bit 3..0 = 0 für LW 1, Seite 0
= 1 für LW 2, Seite 0
= 2 für LW 1, Seite 1
= 3 für LW 2, Seite 1

D= Spurnummer
E= Sektornummer
HL= Quell- oder Zieladresse

Ausgabeparameter: A= 0FFh, falls Fehler
= 0 sonst
F = NZ,C, falls Fehler
= Z,NC sonst

Mit diesem Einsprung kann auf 8" Diskettenlaufwerke zugegriffen werden. Er ist wegen der Kompatibilität zu früheren Versionen vorhanden.

4.7.14. Floppy-Bedienung, 5 1/4"

Einsprungadresse: 0F027h

Eingabeparameter: B=0
D: Bit 0..1 Steprate
Bit 7 = 0, dann kein Seitenvergleich
Bit 7 = 1, dann Seitenvergleich

Ausgabeparameter: -

Eingabeparameter: B=1 Sektor lesen
2 Sektor schreiben

C: Bit 0 = Seite
Bit 2..1= 0 für LW 1
= 1 für LW 2
= 2 für LW 3
= 3 für LW 4
Bit 6 = 0 für SD
= 1 für DD

D = Spurnummer
E = Sektornummer
HL= Quell- oder Zieladresse

Ausgabeparameter: A= 0FFh, falls Fehler
= 0 sonst

F= NZ,C, falls Fehler
= Z,NC sonst

Mit diesem Einsprung kann auf 5 1/4" Diskettenlaufwerke zugegriffen werden. Er ist wegen der Kompatibilität zu früheren Versionen vorhanden.

4.7.15. Harddisk-Bedienung

Einsprungadresse: 0F02Ah
Eingabeparameter: -
Ausgabeparameter: -

Mit diesem Einsprung soll die Harddisk bedient werden. Zum Zeitpunkt der Erstellung dieser Version war gerade die Umstellung auf die preisgünstigen IBM-kompatiblen Laufwerke im Gespräch.
Deshalb ist hier noch keine Prozedur implementiert.

4.7.16. Sprungtabelle

Adresse: 0F02Dh

An dieser Adresse stand in früheren Versionen ein Zeiger auf eine Sprungtabelle.

4.7.17. Freier Speicher

Adresse: 0F02Fh

Der Inhalt von (0F02Fh,0F030h) gibt an, ab welcher Speicherzelle (zwischen 0F000h und 0FFFFh) das RAM nach dem Booten von CP/M zur Verfügung steht.

4.7.18. Freier Speicher

Adresse: 0F031h

Der Inhalt von (0F031h,0F032h) gibt an, ab welcher Speicherzelle (zwischen 0F000h und 0FFFFh) das RAM für den Benutzer zur Verfügung steht. Es gibt die Möglichkeit, in diesen freien Bereich kleine Unterprogramme einzulagern.
Tut dies der Benutzer, dann sollte der Inhalt dieser Adresse unbedingt um die Länge des eingeschobenen Programms heraufgesetzt werden. Bisher ist dies aber leider meistens aus Bequemlichkeit unterlassen worden; mit der Folge, daß es nicht immer möglich ist, 2 dieser Erweiterungen gleichzeitig zu laden !!!

4.7.19. Fadenkreuz setzen

Einsprungadresse: 0F03Ah
Eingabeparameter: HL= x-Koordinate
DE= y-Koordinate
Ausgabeparameter: -

An die angegebene Position wird ein Fadenkreuz gesetzt. Ein zuvor gesetztes wird gelöscht.Diese Funktion sollte nur im Grafikmodus verwendet werden.

4.7.20. Maus abfragen

Einsprungadresse: 0F03Dh
Eingabeparameter: -
Ausgabeparameter: C: Bit 7..6 Maustasten (0= gedrückt)
D= Anzahl Schritte nach oben
E= Anzahl Schritte nach unten
H= Anzahl Schritte nach rechts
L= Anzahl Schritte nach links
F= NZ, falls die Maus bewegt wurde
= Z sonst

Mit dieser Funktion kann eine Mausbewegung abgefragt werden.
Nur im Grafikmodus verwenden !

4.7.21. Seiten löschen

Einsprungadresse: 0F040h
Eingabeparameter: -
Ausgabeparameter: -

Es werden alle Bildschirmseiten gelöscht.
Nur im Grafikmodus verwenden !

4.7.22. Seite löschen

Einsprungadresse: 0F043h
Eingabeparameter: -
Ausgabeparameter: -

Es wird die aktuelle Schreibseite gelöscht.
Nur im Grafikmodus verwenden !

4.7.23. Position setzen

Einsprungadresse: 0F046h
Eingabeparameter: HL= x
DE= y
Ausgabeparameter: -

Die aktuelle Position wird auf (x,y) gezsetzt.
Nur im Grafikmodus verwenden.

4.7.24. Linie zeichnen

Einsprungadresse: 0F049h
Eingabeparameter: HL= x
DE= y
Ausgabeparameter: -

Es wird eine Linie von der aktuellen Position nach (x,y) gezeichnet.
Nur im Grafikmodus verwenden.

4.7.25. Schreibseite setzen

Einsprungadresse: 0F04Ch
Eingabeparameter: C= Schreibseite
Ausgabeparameter: -

Es wird eine neue Schreibseite gesetzt.
Nur im Grafikmodus verwenden !

4.7.26. Leseseite setzen

Einsprungadresse: 0F04Fh
Eingabeparameter: C= Leseseite
Ausgabeparameter: -

Es wird eine neue Leseseite gesetzt.
Nur im Grafikmodus verwenden !

4.7.27. RMW-Modus setzen

Einsprungadresse: 0F052h
Eingabeparameter: C= 0 für normales Zeichnen
= 1 für RMW-Modus
Ausgabeparameter: -

Falls der RMW-Modus aktiv ist, findet beim Zeichnen kein Überschreiben des alten Inhalts, sondern eine XOR-Verknüpfung statt (wenn die Hardware dafür ausgelegt ist).
Nur im Grafikmodus verwenden !

4.7.28. Warten auf GDP

Einsprungadresse: 0F055h
Eingabeparameter: -
Ausgabeparameter: -

Es wird gewartet, bis der GDP wieder für eine neues Kommando bereit ist.
Nur im Grafikmodus verwenden !

4.7.29. Kommando an GDP

Einsprungadresse: 0F058h
Eingabeparameter: C= Kommando
Ausgabeparameter: -

Es wird ein Kommando an den GDP übergeben; falls die COL256 aktiv ist, wird die Wirkung durch Software simuliert.

Nur im Grafikmodus verwenden !

4.7.30. Speicherbereich verschieben

Einsprungadresse: 0F05Bh
Eingabeparameter: C = Quellbank
B = Zielbank
HL = Quelladresse
DE = Zieladresse

Ausgabeparameter: F = C, falls eine Bank nicht da
= NC sonst

Ein 128-Byte langer Speicherbereich wird verschoben. Quell- und Zielbank müssen im Bereich von 0F000h bis 0FFFFh mit RAM bestückt sein.

4.7.31. Aktive Bank holen

Einsprungadresse: 0F05EH
Eingabeparameter: -
Ausgabeparameter: A= aktive Bank

Es wird die Bank ermittelt, auf der sich das Benutzerprogramm gerade befindet.

4.7.32. Sprung auf andere Bank

Einsprungadresse: 0F061h
Eingabeparameter: C' = Bank
HL' = Sprungadresse
Ausgabeparameter: -
Zerstörte Register: B', IY

Mit diesem Monitoraufruf kann ein Unterprogramm auf einer anderen Bank aufgerufen werden.Durch das aufgerufene Unterprogramm dürfen sich SP, C',und HL' nicht ändern. Dann kann die Rückkehr wieder durch ein einfaches JP0F061h erfolgen.

Achtung: Der Stackpointer muß vor dem ersten PUSH auf einen lokalen Stack umgesetzt werden !!!

Beispiel: auf Bank 0 steht

```
...
LD  C,11        ; beliebiger Parameter
EXX
LD  C,1         ; Bank 1 rufen
LD  HL,PROG     ; Adresse der Prozedur
EXX
CALL 0F061h     ; Prozedur aufrufen
...
```

auf Bank 1 steht

```
PROG:

LD  (HSP),SP            ; Stackpointer aufheben
LD  SP,LOCAL_STACK      ; auf lokalen Stack schalten
PUSH AF                 ; jetzt erlaubt
LD  A,C                 ; (fast) beliebiger CODE
POP  AF
LD  SP,(HSP)            ; restaurieren
CALL 0F061h             ; = RETURN nach Bank 0
```

4.7.33. Speicher belegen

Einsprungadresse: 0F064h
Eingabeparameter: A = Bank
H = MSB Adresse (1k-Grenze !)
L = Länge in kByte
Ausgabeparameter: F = NZ, falls Bank nicht da oder der angeforderte Speicher schon belegt ist
= Z sonst

Die Speicherverwaltung wurde eingeführt, um eine einheitliche Handhabung des gesamten RAMS zu gewährleisten. Zusammen mit dem Sprung auf eine andere Bank eröffnet sich so die Möglichkeit, dynamisch Hilfsprogramme in den Speicher nachzuladen, ohne an irgendwelche vorher festgelegten Adressen gebunden zu sein.Allerdings müssen die Hilfsprogramme dann relokativ geschrieben sein oder zur Laufzeit gelinkt werden. Eine andere Anwendung ist die dynamische Bereitstellung von Speicherplatz für Puffer etc.
Auf diese Weise ließe sich vom BIOS feststellen, wieviel Platz für eine RAM-Floppy zur Verfügung steht; es müßte dann nicht beim Hinzufügen einer neuen Spei-

cherbaugruppe das BIOS neu assembliert werden.

4.7.34. Speicher freigeben

Einsprungadresse:	0F067h
Eingabeparameter:	A= Bank
	H= MSB Adresse (1k-Grenze !)
	L= Länge in kByte
Ausgabeparameter:	-

Zuvor angeforderter Speicher wird wieder freigegeben. Es muß sichergestellt sein, daß nur wirklich zuvor belegter Speicher freigegeben wird.

4.7.35. Einsprung umsetzen

Einsprungadresse:	0F06Ah
Eingabeparameter:	HL= Adresse des Programmstücks
	BC= Länge des Programmstücks
	DE = Adresse des Sprunbefehls
	HL´= Adresse der Korrekturtabelle
	BC´= Länge der Korrekturtabelle
	DE´= Basisadresse des Programmstücks
Ausgabeparameter:	F=C, falls kein Platz mehr = NC
Ausgabeparameter:	F =NZ, falls Liste schon voll
	=Z sonst

Mit diesem Aufruf kann eine Umleitung in einen beliebigen Einsprung eingebaut werden; d.h. es kann so z.B. eine kleine Prozedur in die Konsolausgabe eingehängt werden, die alleKleinbuchstaben in Großbuchstaben umwandelt. Die Anwendung dieses Aufrufs wird am Besten durch das folgende Beispiel deutlich:

```
.z80

    INCLUDE RSX.MAC

BEGIN   0f009h          ; Adresse des zu ersetzenden Einsprungs
REL_BEGIN
    ld  a,c
```

```
      cp    'a'
REL <jp     c,next>
      cp    'z'+1
REL <jp     nc,next>
      sub   32
      ld    c,a
REL <jp     next>

REL_END
end   start
```

Ende Beispiel

Im Beispiel wurden einige Makros verwendet, um häufig wiederkehrende Sequenzen abzukürzen. Die Makrodefinition steht weiter unten. Aus dem kurzen Programmstück wird nach deren Aufruf alle Kleinbuchstaben in Großbuchstaben umgewandelt werden. Die benutzten Makros befinden sich auch auf der Quellendiskette zu FLOMONCG.

```
BEGIN     macro     jump
set_jp          equ  0f06ah       ; Umleitung einsetzen
reset_jp  equ  0f06dh             ; Umleitung entfernen
bdos      equ  00005h

start:
      ld    hl,relbeg
      ld    bc,relend-relbeg
      ld    de,jump
      exx
      ld    hl,reltbl
      ld    bc,reltbl_len
      ld    de,relbeg
      exx
      call  set_jp
      ret   nc
      ld    c,9
      ld    de,msg
      jp    bdos
```

```
msg: db   'RSX: nicht mehr genuegend Platz',10,13,'$'
     endm


REL_BEGIN   macro
dseg
reltbl:                  ; Anfang der Tabelle mit Korrekturadressen
cseg
relbeg:
next: jp    0000         ; dummy-Eintrag
     dw  0               ; intern verwendet
     endm


REL_END macro    lbl
dseg
reltbl_len  equ  ($-reltbl)/2
cseg
relend:
     endm


REL macro    m1
local x
m1
x    equ  $-2
dseg
dw   x
cseg
     endm
```

Wenn sich ein Programmierer an das vorstehende Schema des Beispiels hält,dann ist gewährleistet, daß die Bankunabhängigkeit von FLOMONCG erhalten bleibt. Es ist dann auch möglich, mehrere Prozeduren einzuhängen, ohne daß diese etwas voneinander wissen. Insbesondere ergeben sich keine Probleme mehr mit Überschneidungen im Adressbereich. Bisher konnten dort Schwierigkeiten auftreten, falls etwa eine Hardcopyroutine und eine Routine für die serielle Schnittstelle und vielleicht noch ein paar Prozeduren hinzugefügt werden sollten.

5.0 Maus

Bisher mußten alle Kommandos von Hand über die Tastatur oder von einem Programm aus gegeben werden. Sollten Sie allerdings glücklicherweise Besitzer einer Maus und der dazugehörigen Platine sein, dann können Sie die Fähigkeiten von FLOMONCG erst richtig ausschöpfen. Im folgenden werden die Aktionen beschrieben, die Sie mit Hilfe der Maus auslösen können. Da bei manchen Platinen die Zuordnung der Maustasten links und rechts vertauscht ist, müssen Sie evtl. umdenken oder die Leitungen kreuzen.

Drücken Sie die RESET-Taste oder schalten Sie den Rechner aus. Im Menü wählen Sie Punkt 2= Monitor. Sie sehen jetzt in der oberen Hälfte des Bildschirms einen Kasten, das Monitorfenster. Es ist durch einen unterbrochenen und einen durchgezogenen Rahmen hervorgehoben. Jedes Fenster erhält einen solchen unterbrochenen Rahmen, falls es die Abmessungen zulassen. Der durchgezogene Rahmen kennzeichnet das aktive Fenster, in das alle weiteren Ausgaben erfolgen.

Wenn Sie jetzt die Maus bewegen, werden Sie feststellen, daß ein Fadenkreuz oder ein Pfeil der Bewegung über den Bildschirm folgt.
Drücken Sie bitte erst dann irgendwelche Tasten, wenn Sie dazu aufgefordert werden ! Lassen Sie die Maus einige Sekunden in Ruhe, dann verschwindet das Fadenkreuz wieder.

Die Wirkung der Maustasten hängt nun wesentlich davon ab, ob sich das Fadenkreuz innerhalb oder außerhalb des aktiven Fensters befindet. Sollten Sie in der weiteren Beschreibung irgendwo nicht die gewünschte Reaktion sehen, so versuchen Sie zuerst festzustellen, ob Sie links/rechts bei den Tasten richtig zugeordnet haben. Im Zweifelsfall drücken Sie eben noch einmal RESET.

Nicht alle Funktionen sind jederzeit von jedem Programm aus verfügbar. Aus Sicherheitsgründen kann das Öffnen, Schließen und Verschieben von Fenstern gesperrt worden sein.

5.1. Maus außerhalb des aktiven Fensters

5.1.1. Linke Taste

Funktion: Zyklisches Aktivieren der Fenster

Fahren Sie mit der Maus an den oberen Bildschirmrand; hier ist sie mit Sicherheit außerhalb eines jeden Fensters.

Wenn Sie jetzt einmal die linke Taste drücken und gleich wieder loslassen, dann verschwindet das kleine Fenster, und der gesamte Bildschirm wird umrahmt. Ein zweiter Tastendruck stellt wieder den Ausgangszustand her.

Auf diese Art und Weise wird jeweils das nächste Fenster zum aktiven Fenster. Momentan sind aber nur 2 vorhanden.

5.1.2. Rechte Taste

Funktion: Fenster verschieben

Machen Sie das kleine Fenster sichtbar und fahren Sie mit der Maus an den oberen Bildschirmrand. Halten Sie jetzt die rechte Taste gedrückt. Das Fadenkreuz springt an den linken oberen Rand des aktiven Fensters, ein Zeichen dafür, daß Sie es jetzt verschieben können. Bewegen Sie jetzt die Maus ein Stück nach unten. Nach dem Loslassen müßte sich das Fenster nach unten verschoben haben. Ist dies nicht der Fall, dann haben Sie das Fenster zu weit bewegen wollen und die gesamte Aktion wird ignoriert.

5.1.3. Linke Taste + Rechte Taste

Funktion: Fenster öffnen

Fahren Sie die Maus wieder zum oberen Bildschirmrand. Drücken Sie die linke Taste und halten Sie sie unten. Jetzt betätigen Sie zusätzlich auch noch die rechte Taste und lassen beide los.
Der Cursor verschwindet, aber das Fadenkreuz bleibt und folgt der Maus.

Bewegen Sie das Fadenkreuz etwa in die Mitte des Bildschirms. Drücken Sie die linke Taste und halten Sie sie unten, während Sie jetzt mit der Maus ein Stück nach rechts unten fahren; aber bitte nicht zu weit an den rechten oder unteren Rand. An der Stelle, an der Sie die Taste betätigt haben, bleibt ein Kreuz stehen. Das Rechteck, das von diesem Kreuz

und dem Fadenkreuz begrenzt wird, gibt die Größe des Fensters an. Lassen Sie jetzt die linke Taste los; es müßte nun ein neues Fenster erscheinen, in dem der Cursor wieder blinkt.

5.1.4. Linke Taste

Funktion: Fenster gezielt aktivieren

Bewegen Sie das Fadenkreuz in das jetzt nicht mehr aktive Fenster mit der Statuszeile 'Monitor'. Wenn Sie jetzt kurz die linke Taste betätigen, sehen Sie, daß dieses Fenster aktiviert wird. Fahren Sie zurück in das andere und wiederholen Sie den Vorgang entsprechend.

5.2. Maus innerhalb des aktiven Fensters

Begeben Sie sich bitte vom Menü aus mit ^C in den Terminaltest.

5.2.1. Linke Taste

Funktion: Cursor positionieren

Bewegen Sie das Fadenkreuz an eine beliebige Position innerhalb des aktiven Fensters (mit Ausnahme der Statuszeile!). Drücken Sie jetzt kurz die linke Taste. Nach einigen Augenblicken müßte sich der Cursor unter dem Fadenkreuz befinden. Diese Positionierung funktioniert

1. im Testmodus
2. beim Editieren des (->) Notizbuches.
3. bei allen Editoren, bei denen der Cursor mit den Tasten ^S,^E,^D,^X bewegt werden kann (WORDSTAR, TURBO, ...).
   Bemerkung: Wordstar z.B. akzeptiert nur dann, wenn sich darunter noch Text befindet.

5.2.2. Rechte Taste

Funktion: Zeile abschicken

Fahren Sie mit der Maus zum Anfangsbuchstaben eines Wortes (im aktiven Fenster), z.B. 'Terminaltest'. Wenn Sie jetzt kurz die rechte Taste drücken, dann sehen Sie, daß dieses Wort und der gesamte Text rechts davon an der Cursorposition erscheint. Mit dieser Funktion können Sie z.B. Teile einer Eingabezeile wiederholen.

5.2.3. Linke Taste + Rechte Taste

Funktion: Fenster schließen

Bewegen Sie die Maus mitten in das aktive Fenster (falls gerade keines sichtbar ist, eröffnen Sie sich eben eines). Drücken Sie die linke Taste, halten Sie sie gedrückt und betätigen Sie die rechte Taste zusätzlich. Wenn Sie jetzt beide Tasten loslassen, dann verschwindet das aktive Fenster, es wird geschlossen bzw. gelöscht. Wiederholen Sie den Vorgang, bis der Bildschirm komplett gelöscht ist. Nachdem das letzte Fenster geschlossen ist, wird automatisch das Standardfenster wieder eröffnet.

5.2.4. Rechte Taste + Linke Taste

Funktion: Hardcopy

Falls ein Drucker angeschlossen ist, können Sie eine Hardcopy des Bildschirms erzeugen. Fahren Sie dazu mit der Maus mitten in das aktive Fenster. Drücken Sie die rechte Taste, halten Sie sie gedrückt und betätigen Sie die linke zusätzlich. Wenn Sie eine HCOPY-Platine haben, sehen Sie, wie der Bildschirm abgetastet wird. Ansonsten können Sie den Erfolg der Aktion auf dem Drucker betrachten.

5.3. Maus auf der Statuszeile des aktiven Fensters

5.3.1. Rechte Taste

Funktion: Fenstergröße verändern

Bewegen Sie die Maus auf die Statuszeile (unterste Zeile) des aktiven Fensters. Drücken Sie die rechte Taste und halten Sie sie gedrückt. Der Cursor hört auf zu blinken und das Fadenkreuz springt an den rechten unteren Rand. Sie können jetzt mit der Maus die rechte untere Ecke des Fensters an eine beliebige Position setzen. Beim Loslassen der Taste wird das Fenster dann mit der neuen Größe gezeichnet.

5.3.2. Linke Taste

Funktion: Menü wählen

Diese Funktion können Sie nur dann sinvoll einsetzen, wenn Sie einen Massenspeicher zur Verfügung haben. Für CP/M sind ein Menügenerator und mehrere Beispiele verfügbar. Sollten Sie RNWINDOWS besitzen, dann können Sie auch darauf zurückgreifen.

Ansonsten brauchen Sie dazu die DEMO/Utility-Diskette für FLOMONCG, auf der die Benutzung auch näher beschrieben ist.

Wird die linke Taste auf der Statuszeile betätigt, dann wird die Anzahl der Leerzeichen zwischen dem linken Rand und der Position des Fadenkreuzes ermittelt. Diese Anzahl gibt die Nummer des zu öffnenden Fensters an (siehe ->Fensterkommandos, Menüfenster). Falls ein solches existiert, wird es sichtbar. Mit gedrückter Taste wird das Fadenkreuz auf eine Zeile des Menüfensters positioniert. Beim Loslassen der Taste wird die entsprechende Zeile an den Rechner geschickt, so als ob der Text von der Tastatur eingegeben worden wäre.
Um die Funktionen dokumentieren und um Steuerzeichen schicken zu können, wurde folgende Konvention vereinbart:

Alle Zeichen links des ersten Doppelpunkts in der Zeile sind Kommentar. Alle Zeichen rechts davon werden an den Rechner geschickt, wobei '^' und ein Zeichen das entsprechende Steuerzeichen ergeben.
Eine Zeile des Beispielmenüs für WORDSTAR könnte lauten

'Cursor nach oben:^E'

6.0 Diverses

6.1. Aufruf anderer Programme

Obwohl beim Entwurf des neuen Monitors auf größtmögliche Kompatibilität zu vorhergehenden Versionen geachtet wurde, haben sich in der Bedienung einige Änderungen ergeben.

6.1.1. GO E0000

Dieser Punkt ist aus dem Menü herausgefallen. Die gleiche Wirkung wird erzielt, wenn zuerst der Monitor aufgerufen wird und dann das Kommando 'G E:0' gegeben wird. Hier liegt der Vorteil darin, daß die EPROMS nicht unbedingt auf Bank 0E stecken müssen, sondern beliebig eingesetzt werden können. Allerdings ist jetzt dafür unabdingbare Voraussetzung, daß die entsprechende Bank im Bereich von F000..FFFF mit RAM bestückt ist.

6.1.2. GO BANK 2000, ZEAT

Auch dieser Punkt erscheint nicht mehr im Menü. Dies hat seinen Grund darin, daß die BANK/BOOT Karte allein für FLOMONCG reserviert bleibt. Es können dort keine anderen Programme gleichzeitig untergebracht werden. Insbesondere fällt ZEAT unter diese Einschränkung.
Soll ZEAT weiterhin betrieben werden, dann können die beiden Eproms irgendwo auf einer beliebigen Bank untergebracht werden, z.B. auf Bank 0F, Adresse 4000..7FFF. Um ZEAT dann erfolgreich aufrufen zu können, gehen Sie folgendermaßen vor:

1. Gehen Sie in den Monitor
2. Geben Sie das Kommando 'M F:4000 7FFF 0:2000'; damit wird das Programm auf Bank 0, Adresse 2000h verschoben.
3. Starten Sie ZEAT mit 'G 0:2000'

Der Bildschirm ist zunächst in Unordnung, weil ZEAT seinen Begrüßungstext als Grafik ausgibt, die sich nicht mit dem neuen Textmodus verträgt. Allerdings braucht Sie dieser Effekt nicht weiter zu stören.
Wesentlichster Nachteil ist momentan, daß ZEAT von einem 80 Zeichen breiten Bildschirm ausgeht und somit ein Betrieb auf der COL256 nicht vernünftig möglich ist.

6.2. Unstimmigkeiten

Unter besonderen Umständen können bei Programmen seltsame Effekte auftreten. Diese werden im folgenden beschrieben.

6.2.1. Text und Grafik

Es ist nicht mehr möglich, Text und Grafik beliebig zu mischen. Wird der Grafik-Modus aufgerufen, dann bleibt das Textbild noch erhalten, allerdings nur auf EINER Seite und nicht auf mehreren. Dies war allerdings bereits schon ab Version 4.2 der Fall.

Bei der Rückkehr von der Grafik in den Textmodus wird der Textbildschirm komplett neu aufgebaut, eine jegliche Grafik geht verloren. Somit ist es nicht möglich, z.B. einen Begrüßungstext in großer Schrift auf dem Bildschirm stehen zu lassen (ZEAT).

Die dadurch hervorgerufene Einschränkung sollte sich aber dennoch verkraften lassen. Falls es gar nicht mehr anders geht, besteht die Möglichkeit, FLOMON 3.2 von der Diskette nachzuladen. Auf der Quelltext-/Demodiskette befindet sich deshalb eine Datei FLOMON32.COM, die sich selbst auf die Bank/Boot lädt und die Einsprünge umsetzt. Jetzt müßten alle Programme laufen, die keine Probleme mit FLOMON 3.2 hatten, so auch LOG16 und CAD. Erst beim nächsten RESET wird wieder FLOMONCG aktiv.

Achtung: Nach dem Laden von FLOMON32 besteht keine Möglichkeit mehr, auf die erweiterten Fähigkeiten von FLOMONCG zuzugreifen !

6.2.2. Steuerzeichen und Bildschirmaufbau

Aus Optimierungsgründen wird nicht nach jedem ausgegebenen Steuerzeichen (Zeichen/Zeile löschen/einfügen) der Bildschirm sofort aufgefrischt. Erst nach einigen Aufrufen von CSTS oder CI wird der Bildschirm neu aufgebaut. Gibt ein Programm aber immer nur Zeichen aus, ohne entsprechend oft CI aufzurufen, dann kann es sein, daß die Ausgabe unsichtbar bleibt. Beispiele hierfür sind der Formatierer UFORM und das CP/M-Spiel LADDER. Sollte dieser Effekt zu störend wirken, wird bei neueren FLOMONCG-Versionen darauf Rücksicht genommen. Allerdings kann wie im vorigen Fall 'Text und Grafik' auch hier im Bedarfsfall FLOMON32 nachgeladen werden.

6.3. Verbesserungsmöglichkeiten

6.4. Ausblick

Korrekturen für dieses Handbuch werden in der Zeitschrift LOOP bekanntgegeben. Man sollte dann die fehlerhaften Stellen von Hand korrigieren.

6.5. Kritik

Bitte senden Sie uns die ausgefüllte Kritikkarte, die dem Handbuch beiliegt, zurück. Sie helfen uns, unsere Produkte und unseren Service noch besser zu gestalten.
Für Fehlermeldungen und Verbesserungen, die dieses Handbuch betreffen, sind wir immer dankbar!