Anmerkung: CP/M Programme laufen unter KOS mit Hilfe des Translators, eines Umsetztreibers für CP/M 2.2 Systemaufrufe

7. CP/M 2.2 Kompatibilität

Wer mit dem weitverbreiteten Betriebssystem CP/M des Software-Herstellers Digital Research vertraut ist, wird deutliche formale Gemeinsamkeiten mit Kontrons KOS erkennen. KOS ist jedoch benutzerfreundlicher und Hardware-unabhängiger konzipiert, und es bietet in sich den Zugang zum lokalen Netzwerk Kontron KOBUS. Darüber hinaus stehen flexiblere Möglichkeiten in der Datei- und Medienorganisation bei KOS zur Verfügung.

Formale Gemeinsamkeiten sind:

- '.COM' als Dateityp für Programme
- Startadresse 100 h für Programme voreingestellt
- KOS wie CP/M liegen im obersten Speicherbereich
- Medienanwahl, bei KOS durch Mediennummer 'mn:', bei CP/M durch A:, B:, ...,
  allerdings: KOS kann alle aktiven Medien nach einer Dateireferenz absuchen
- Dateizugriffe über Indexdatei
- interne Kommandos
- 8-stellige Dateinamen, 3-stelliger Dateityp
- '*' als Universalbezeichner in Dateiadressen
- Systemaufruf über Einsprungvektor möglich

Wegen dieser Gemeinsamkeiten ist der Ablauf von Programmen, die sich an die CP/M-Systemschnittstelle halten, zunächst einmal möglich.

Unterschiede bestehen in

- Medienkonzeption
- Working Directory
- Logische Ein-/Ausgabeorganisation

In diesen Punkten können CP/M-Programme die erweiterten Möglichkeiten von KOS nicht ausnutzen.

Gravierende Unterschiede bestehen jedoch in

- Systemaufrufformat
- Interruptbehandlung
- Zulässigkeit offener Dateien
- direkte Systemeinsprünge
- Zugriff auf Speicheradressen

Diese Unterschiede machen eine Prüfung der Ablauffähigkeit im einzelnen notwendig.

Die wesentliche Anpassung von CP/M-System-Aufrufen bietet der Umsetztreiber $CPM. Dieses Modul ist als Treiber organisiert und wird wie ein Ein-/Ausgabetreiber aktiviert. Er wirkt als ein dem Betriebssystem KOS vorgeschalteter Umsetzer, indem er die CP/M 2.2-Systemaufrufe in KOS-Systemaufrufe umwandelt (s. Utility/Umsetztreiber für CP/M-Aufrufe $CPM).

Parallel dazu arbeitet bei KOS5 der Bildschirmvortreiber XMON, der die X-Y-Koordinaten-orientierten Bildschirmausgaben empfängt und in die Hardware-spezifischen relativen Bildspeicheradressierungen umwandelt. XMON ist durch das KOS-Kommando 'RLOAD XMON' zu aktivieren für Programme, die Terminal-ähnliche Bildschirmausgabe voraussetzen, z.B. WordStar etc. In KOS6 ist die XMON-Funktion direkt integriert.

Bisher mit Erfolg erprobt als ablauffähig unter KOS mit CP/M-Treiber sind folgende CP/M-Programme:

MICROSOFT:	MBASIC	Redding:	LYNX
	BASCOM		
	FORTRAN80	SORCIM:	SUPERCALC2
	COBOL80		
	L80	Ashton-Tate:	dBASE II
	LD80		

MICRO PRO:	WORDSTAR	DIGITAL RESEARCH:	Pascal MT+
	WORDINDEX		
	DATASTAR	Micro Focus:	CIS-COBOL
	SUPERSORT		COBOL Level II

und weitere.

Bei welchen Produkten muß nun mit Anpassungsproblemen gerechnet werden?

Dies sind:

- Hardware-spezifische Programme
- Programme mit speziellen Terminalbedingungen
- Programme mit Systemeinsprüngen außerhalb der Standard CP/M-Systemaufrufe

Diese Anpassungen erfordern Aufwand, die Anpassung ist jedoch zumindest bei Vorliegen des Quellcodes immer lösbar.

Genauere Prüfung erfordern folgende Umstände:

- **Speicherbedarf:** Hier schneidet KOS6 besser ab als CP/M, KOS5 jedoch benötigt mehr Speicher als CP/M. Je nach Art des Programms wird die Anpassung möglich oder sehr schwierig sein, falls der Punkt 'Speicher' Probleme aufwirft.

- **Interrupt- und Stack-Probleme:** KOS ist ein Interrupt-orientiertes Betriebssystem. Jeder Interrupt belegt Stackraum. In Programmen, die mehrere kleine Stacks anlegen, treten sporadische Fehlfunktionen auf, wenn der Stack in Programm- oder Datenbereiche überläuft. Eine Programmanpassung ist möglich, wenn der Quelltext vorliegt, sie kann problematisch sein, wenn die Stackadressen dynamisch allokiert werden.
  Der Stackbereich wird von KOS dynamisch im Speicher verwaltet, die Größe beträgt 256 Bytes.

CP/M kompatible Programme, die den Stackbereich auf eine geringere Größe umdefinieren, können Fehlfunktionen des Gesamtsystems auslösen.

Beim Ablauf von CP/M kompatiblen Programmen gilt daher folgendes zu beachten:

- in welcher Weise wird der Stack verwaltet?
- auf welche Größe wird der Stack definiert?
- hat der Anwender Einfluß auf die Größe der Stacks?

Stellvertretend für eine Reihe von CP/M Software sei hier als Beispiel die Stackverwaltung von PASCAL MT+ Programmen dargelegt:

PASCAL MT+ Programme dimensionieren den Stackbereich auf 128 Byte (PASCAL MT+ Handbuch, Abschnitt 4.11).

Der Hersteller von PASCAL MT+, Digital Research, empfiehlt bei Interrupt betriebenen Systemen den Stack entsprechend zu erhöhen und stellt dafür eine Variable zur Verfügung.

Um Konflikte durch zu klein dimensionierten Stack zu vermeiden, ist in PASCAL MT+ Programmen der voreingestellte Wert von 128 Bytes auf 256 Bytes zu erhöhen.

Bei Kobusanlagen empfiehlt sich eine Stackgröße von 384 Bytes in PASCAL MT+ Programmen zu definieren.

- **Offene Dateien und DSB-(FCB-)-Zugriffe:** KOS verwaltet bis zu 16 offene Dateien. CP/M-Programme haben die Tendenz, beliebig viele Dateien offen zu halten, oder auf geschlossene Dateien wieder zuzugreifen. Der CP/M-Translator umgeht diese Problematik, indem er offene Dateien selbst verwaltet und automatisch schließt, so daß CP/M-Programme keine offenen Dateien zurücklassen können.

- Aufrufverhalten: Der CP/M-Translator schaltet sich beim ersten CP/M-Systemaufruf ein. Dieser darf jedoch kein Aufruf CREATE, OPEN oder DELETE sein. Falls Programme solchermaßen beginnen, stehen folgende Möglichkeiten offen: "Patchen" des Programmes, um einen Dummy System Call einzufügen, oder Vorschalten eines Dummy Programmes mit Chaining des eigentlichen Programmes.

- Bei Zugriffsfehlern auf Speichermedien erzeugt $CPM die Fehlermeldung "$CPM-KOS I/O-Error:nn", mit nn= 82...86, entsprechend den KOS-Fehlercodes. Die darauffolgende Benutzereingabe "ESC" bricht das laufende Programm ab, durch Eingabe von "RETURN" wird ins laufende Programm zur Fehlerbehandlung zurückverzweigt.

Insgesamt bedeutet die Anpassung von CP/M-Software an KOS dann keine oder wenig Mühe, wenn der Programmierer sich an die definierten CP/M-Systemaufrufe gehalten hat und es vermieden hat, CP/M-interne Abläufe und Speicherzellen zu verwenden. Dies jedoch sind Grundsätze jeder 'sauberen' Programmierung und sollte somit keine Einschränkung bedeuten.

Welche Vorgehensweise empfehlen wir Ihnen bei CP/M-Software-Problemen?

Es ist die gleiche, wie bei anderen Software-Problemen auch:

- Versuchen Sie, das Problem in eine einfach reproduzierbare Form zu bringen. Dies führt zu einer klaren Problemdefinition und möglicherweise bereits zur Lösung oder Umgehung des Problems.

- Teilen Sie uns - bitte schriftlich, ein Formularvorschlag dazu ist im Anhang abgedruckt - Ihre Fragestellung mit. Programmbeispiele auf Diskette ermöglichen uns im allgemeinen die schnellere Klärung Ihrer Frage.

Und als einfachste und doch wichtigste Maßnahmen:

- Verwenden Sie fehlerfreies, für Kontron PSI-Systeme zugelassenes Diskettenmaterial.

- Prüfen Sie die logische Konsistenz und physikalische Funktionalität Ihrer Medien (STATUS, FSCHECK, CHMED, FORMAT P V).

- Verwenden Sie freigegebene aktuelle Software-Stände.

- Beachten Sie die Grundregeln über Medieninitialisierung, Treiberorganisation etc, wie sie in diesem Handbuch beschrieben sind.

8. Gewährleistung bei Software

Betriebssysteme, System- und Anwendungsprogramme beinhalten komplexe Aufgabenstellungen. Fehler, Inkonsequenzen und unerwünschte Reaktionen bei Fehlbedienung sind weitgehend durch intensive Tests ausgeschlossen, die Garantie der Fehlerfreiheit kann jedoch kein Hersteller von Rechnern und Software übernehmen. Kontron ist bemüht, Ihnen nach dem Stand der Technik das bestmögliche Produkt zu liefern. Kontron übernimmt keine Gewähr für die Eignung von Software für eine bestimmte Anwendung.

Bei Beachtung der anerkannten Regeln der Kunst des Programmierens und der Rechnertechnologie werden Sie mit Kontron PSI-Rechnern leistungsfähige und benutzerfreundliche Hardware/Software Systeme für Ihre Anwendung aufbauen können.

Bei auftretenden Problemen sind wir bereit, an der Lösung zusammen mit Ihnen zu arbeiten, sei es durch Verbesserung der Dokumentation, durch Schulungskurse oder durch Test und Analyse von Problemstellungen. Weitergehende Haftungen und Ansprüche sind ausgeschlossen.

6. Umsetztreiber für CP/M-Aufrufe $CPM

$CPM ist notwendig für alle Programme, die Systemaufrufe über CP/M-'CALLS' durchführen, also z.B. FORTRAN, MBASIC, BASCOM, alle mit diesen Übersetzern erzeugten Programme, und für WORDSTAR, SuperCalc etc.
Bei fehlender Aktivierung wird das aufgerufene Programm abgebrochen.

Näheres zur Thematik "CP/M-Programme unter KOS" siehe Abschnitt Konzepte und Grundlagen, TB-A. Die Quelldatei CPM.SRC gibt weitere Detailinformationen. Sie ist Teil der UTILITY-Diskette.

CP/M-kompatible (CP/M-Versionen 1.4 und 2.2) Programme verwenden als Systemaufruf die Instruktion 'Call 5'. Über das Umsetzprogramm **CPM.OBJ** werden diese Funktionen in KOS-kompatible Aufrufe (RST8) transformiert. Dieses Programm ist als Treiber organisiert. Der CP/M-Translator $CPM Version 2.98 ist zum einen separat verfügbar (Aktivierung mit "IODC $CPM=ACTIVE") und zum anderen im Betriebssystemmodul KOSB.SYS (nur KOS6) eingebunden. Er benötigt als KOSB.SYS den geringeren Platz von 256 Bytes (100H Bytes) im Anwenderspeicher und braucht nicht mittels IODC-Kommando aktiviert werden.

Die Aktivierung erfolgt automatisch beim Laden von KOS V6.06, vorausgesetzt, daß das KOSB.SYS-Flag in KOS0.SYS gesetzt ist. Das Setzen und Rücksetzen des KOSB.SYS-Flags erfolgt durch die KOS Kommandofolge (siehe Techn.Beschreibung):

```
KOSGEN KOS0<---
```

Vor der ersten Verwendung des CP/M-Translators muß dieser eröffnet werden mit dem Systemkommando "öffne Treiber":

```
O $CPM<---
```

Die Frage nach "debug mode" bei eventuellem mehrfachen Eröffnen wird mit N(ein) beantwortet. Ist der CP/M Translator nicht geöffnet, erscheint die Systemmeldung "$CPM aktiv?". Die verwendete Translator-Version kann festgestellt werden durch Neuinitialisierung von $CPM mit dem Systemkommando:

```
N $CPM<---
```

Wir empfehlen in Zukunft nur noch mit CP/M-Translatermodul Version V2.98 (oder später) zu arbeiten.
Ab der Version 2.93 sorgt der Translator selbst dafür, daß Anwenderdateien geschlossen werden.

Bei früheren $CPM-Versionen wurde eine Utility CLOSEX angeboten, mit deren Hilfe Dateien von Anwenderprogrammen aus selektiv geschlossen werden konnten. Dies ist ab $CPM 2.93 nicht mehr erforderlich.

CLOSEX-Aufrufe müssen beim Arbeiten ab CP/M-Translator Version 2.93 aus den Anwenderprogrammen entfernt werden, da sie ansonsten zu einem undefinierten Systemverhalten führen können.

Einige CPM-Programme (z.B.: COBOL und MTPLUS) reservieren nur einen relativ kleinen Stackbereich. Dies kann eventuell zu Konflikten mit dem Multi-Tasking von KOS führen. Näheres zu diesem Thema entnehmen Sie bitte dem Abschnitt 7 CP/M 2.2 Kompatibilität in der KOS Technischen Beschreibung TB-A.

Programmbeschreibung: $CPM, Version 2.98

$CPM setzt CP/m 2.2 kompatible Systemaufrufe (Call 0005) in KOS kompatible Systemaufrufe (RST 8) um. $CPM belegt 2 kByte (16 Segmente) des Anwenderspeichers. Folgende CP/M-Systemaufrufe sind implementiert:

A: Simple Device Operations

00:	System Reset	-
01:	Console Input	(KOS Kanal I-1/0-1)
02:	Console Output	(KOS Kanal 0-1)
03:	Reader Input	(KOS Kanal I-2)
04:	Punch Output	(KOS Kanal 0-4)
05:	List Output	(KOS Kanal 0-2)
06:	Direct Console I/O	(KOS Kanal I-1/0-1)
07:	Get I/O Byte	
08:	Set I/O Byte	-
09:	Print String	(KOS Kanal 0-1)
10:	Read Console Buffer	(KOS Kanal I-1/0-1)
11:	Get Console Status	(KOS Kanal I-1)
12:	Return Version Number	

B: FDOS Operations		**KOS-Funktion**
13:	Reset Disk System	-
14:	Select Disk	61H
15:	Open File	42H
16:	Close file	63H/6DH
17:	Search for First	64H
18:	Search for Next	65H
19:	Delete File	66H
20:	Read Sequential	77H
21:	Write Sequential	68H/78H
22:	Make File	49H
23:	Rename File	4AH
24:	Return Log-in Vector	-
25:	Return Current Disk	-
26:	Set DMA Address	-
27:	Get ADDR (Alloc)	-
28:	Write Protect Disk (*)	-
29:	Get Read Only Vector	-
30:	Set File Attributes (*)	-
31:	Get ADDR (Disk Parms)	-
32:	Set/Get User Code (*)	-
33:	Read Random	77H
34:	Write Random	78H
35:	Compute File Size	79H
36:	Set Random Record	-
37:	Reset Drive (*)	-
40:	Write Random with Zero File	78H

(*) keine Wirkung unter KOS