MZ-80B - intern Nr. 5

Die hier beschriebene Verfahrensweise beschleunigt das Lesen und Speichern von größeren Datenmengen von/auf Kassette rund um den Faktor vier. Dies wurde durch zwei Änderungen ereicht: Zum einen wird auf die zweimalige Aufzeichnung der Daten verzichtet. Dies ist zwar eine Sicherheitsmaßnahme im Monitor, die die Wahrscheinlichkeit von Datenverlusten mindern soll, die Erfahrung hat jedoch gezeigt, das Aufzeichnungsfehler extrem unwahrscheinlich sind. Zum zweiten wird der interne 256-Byte-Buffer, der für das andauernde Ein- und Ausgeschalte verantwortlich ist, deaktiviert und stattdessen ein anderer Speicherbereich eingerichtet, der wesentlich größer ist. Der Vorteil dieser Verfahrensweise ist, daß sämtliche Kassettenbefehle bestehen bleiben, was bei anderen Verfahrensweisen, bei dem Teile des Speicherbereichs für Variablen direkt als Maschinenprogramm aufgezeichnet werden, nicht der Fall ist.

Die erste angesprochene Änderung läßt sich mit POKE $02DB,$01 und POKE $030C,$01 durchführen. Da dies Änderungen im Monitor sind, gilt dies für alle Interpreter, die den Monitor SB-1510 verwenden. Die Änderungen werden mit POKE $02DB,$02 und POKE $030C,$02 wieder rückgängig gemacht. übrigens werden Aufzeichnungen ohne Wiederholung auch ohne Änderungen im Monitor gelesen, solange kein Aufzeichnungsfehler vorliegt. Dann nämlich versucht der Monitor, die Wiederholung zu lesen, und die gibt es ja nun nicht mehr.

Nun zur zweiten Änderung. Der interne Buffer ist durch drei Größen gekennzeichnet: Länge ($0100 = 256), Anfangsadresse ($4D93 bei BASIC SB-5510 und $63D3 bei BASIC SB-6510) und Endadresse ($4E93 bzw. $64D3). Wenn man nun einen größeren Buffer vorsehen will, müssen diese Werte geändert werden. Dabei bietet sich ein Speicherbereich am oberen Speicherende an, der mit LIMIT vom Interpreter abgetrennt wird. Die Länge dieses Bereichs hängt von der anfallenden Datenmenge ab. Umso größer er ist, um so seltener braucht er auf Band geschrieben werden (Recorder ein, Schreiben, Recorder aus), da er logischerweise nicht so schnell voll wird. Andererseits wird dadurch der für Programme und Variablen verfügbare Platz kleiner. überlegen Sie sich also diese drei Größen und setzen Sie sie in die POKE-Befehle von Bild 1 ein. Denken Sie dabei daran, daß zuerst das niederwertige Byte und dann das höherwertige Byte der Adresse bzw. Länge angegeben werden muß.

In Bild 1 wurden als Beispiel verwendet: Anfang = $F000, Ende = $FE00, Länge = $0F00. Der Buffer wurde mit LIMIT $EFFF vom BASIC-Speicher abgetrennt. Anschließend wurden gut 10000 Bytes Daten (200 Strings mit 50 Zeichen) aufgezeichnet, einmal ohne und einmal mit der hier beschriebenen Verfahrensweise. Im ersten Fall dauerte die Aufzeichnung 5 M. 37 S., es wurden 108 Einheiten Band verbraucht, der Recorder wurde 41 mal ein- und ausgeschaltet. Im zweiten Fall dauerte die Aufzeichnung 1 M. 30 S., es wurden 34 Einheiten Band verbraucht, der Recorder wurde 3 mal ein- und ausgeschaltet. Das ist doch wohl eine sehr deutliche Verbesserung.

Zwei Dinge sollten bei der Verwendung der Kassettenbefehle immer beachtet werden: Strings werden sequentiell, daß heißt einer nach dem anderen aufgezeichnet. Zur Trennung dient CR (=$0D=13). Die Länge der Strings wird im Gegensatz zur Variablenverwaltung im Speicher nicht aufgezeichnet. Daraus folgt, daß Strings, die mehrere CR enthalten, nur bis zum ersten CR aufgezeichnet werden, wobei dieses CR als Endzeichen gilt und bei der Abfrage der Variablen auch nicht erscheint. Beispiel: Aus "A"+CHR$(13)+"B"+CHR$(13)+"C" wird beim Aufzeichnen "A"+CHR$(13), wobei CHR$(13) nicht als Variableninhalt, sondern Endzeichen aufzufassen ist. Nach dem Wiedereinlesen in eine Variable ist also nur ein "A" in dieser. Zweitens gilt das Pi-Zeichen (=$FF=255)

als erstes Zeichen eines aufgezeichneten Strings als Kennung für das Dateiende. Es muß also unbedingt darauf verzichtet werden, Strings, die mit einem Pi-Zeichen beginnen, aufzuzeichnen. Hier bietet sich als Lösung an, grundsätzlich irgendein anderes Zeichen vor den eigentlichen String zu bringen.

Noch ein Trick am Rande: Mit USR($04B1) bzw. USR($04B7) wird eine Kassette bis zur nächsten Lücke vorgespult. Der Unterschied zwischen beiden USR-Aufrufen besteht darin, daß der erste erst die Kassette stoppt, bevor der Suchlauf eingeschaltet wird, während der zweite gleich den Suchlauf aktiviert. Wenn diese Lücke erreicht ist, gibt das Maschinenunterprogramm an das Aufrufprogramm zurück, ohne jetzt jedoch den Recorder zu stoppen. Dies muß gleich anschließend mit USR($04CE) gemacht werden. Der Vorteil des Nicht-Stoppens vor Einschalten des Suchlaufs liegt darin, daß man durch wiederholtes Aufrufen von $04B7 eine bestimmte Anzahl von Lücken überspringen kann, ohne den Recorder jedesmal neu starten zu müssen, was Zeit kostet. Zum Beispiel bei der dritten Lücke stoppen: USR($04B1): USR($04B7): USR($04B7): USR($04CE). Der erste USR-Aufruf sollte aber immer USR($04B1) sein, damit eine eventuell laufende Kassettenfunktion erst einmal gestoppt wird, bevor der Suchlauf aktiviert wird. Es kann aber passieren, daß der Suchlauf durch überlange Lücken oder Reste vorheriger Aufzeichnungen mehr Lücken erkennt, als er eigentlich sollte. Dem kann eventuell durch (sehr) kurze Pausen zwischen den USR-Aufrufen abgeholfen werden.

Bild 1: POKE-Befehle für anderen Buffer , vorher unbedingt: LIMIT $EFFF
Beispiel: Anfang $F000 / Ende $FE00 / Länge $0F00

BASIC SB-5510	BASIC SB-6510	
POKE $3E38,$00	POKE $505A,$00	Bufferanfang niederw. Byte
POKE $3E39,$F0	POKE $505B,$F0	Bufferanfang höherw. Byte
POKE $3E7F,$00	POKE $50A1,$00	Bufferanfang niederw. Byte
POKE $3E80,$F0	POKE $50A2,$F0	Bufferanfang höherw. Byte
POKE $3ED3,$00	POKE $50EB,$00	Bufferanfang niederw. Byte
POKE $3ED4,$F0	POKE $50EC,$F0	Bufferanfang höherw. Byte
POKE $3F40,$00	POKE $514F,$00	Bufferanfang niederw. Byte
POKE $3F41,$F0	POKE $5150,$F0	Bufferanfang höherw. Byte
POKE $3F91,$00	POKE $5195,$00	Bufferanfang niederw. Byte
POKE $3F92,$F0	POKE $5196,$F0	Bufferanfang höherw. Byte
POKE $3E25,$00	POKE $5047,$00	Bufferende niederw. Byte
POKE $3E26,$FE	POKE $5048,$FE	Bufferende höherw. Byte
POKE $3E5D,$00	POKE $507F,$00	Bufferende niederw. Byte
POKE $3E5E,$FE	POKE $5080,$FE	Bufferende höherw. Byte
POKE $3F4C,$00	POKE $515B,$00	Bufferende niederw. Byte
POKE $3F4D,$FE	POKE $515C,$FE	Bufferende höherw. Byte
POKE $3F6E,$00	POKE $5173,$00	Bufferende niederw. Byte
POKE $3F6F,$FE	POKE $5174,$FE	Bufferende höherw. Byte
POKE $3ECD,$00	POKE $50E5,$00	Bufferlänge niederw. Byte
POKE $3ECE,$0F	POKE $50E6,$0F	Bufferlänge höherw. Byte
POKE $3F46,$00	POKE $5155,$00	Bufferlänge niederw. Byte
POKE $3F47,$0F	POKE $5156,$0F	Bufferlänge höherw. Byte

NUR DISC-BASIC!

NICHT Double Precision Basic!

* * *

POKE $ 02DB,$01 [$02] = nur einmaliges Aufzeichnen von Daten
POKE $ 030C,$01 [$02] = ~~nur~~ einmaliges Lesen von Daten

$029C } = write
$02CD } = read
$031A } für Monitor
= Kopf 1511

TEST: a) USR ($04B1) Stoppt Cassette und schaltet Suchlauf ein (Nr 1)
b) USR ($04B7) aktiviert Suchlauf bis Lücke gefunden. (mehrmaliges Benutzen um Lücken überspringen)
c) USR ($04CE) Stoppt Cassettenlauf

REW: a) mit Poke $1150,0 geht's manchmal!