Datei: ‘API-M-DE.DOK‘ (44 Seiten) Stand: 18. September, Jahr des CPC 2023

Datei zur Dokumentation der Programmierschnittstelle (API) des Future Operating System (FutureOS) im Erweiterungs-ROM der M4 Karte. Hier werden für Programmierer wichtige Funktionen beschrieben, die sich im Massenspeicher ROM des FutureOS befinden. Achtung: Das M4 XROM (ROM M) ist nur dann vorhanden, wenn es durch den FutureOS Installer in die M4 Karte installiert wurde! Die OS Funktionen werden in folgender Form beschrieben:

1. Kurzbeschreibung: in einem Satz wird kurz die Funktionsweise der OS Funktion beschrieben.

2. Label: Dieses Label wird für die OS Funktion im Source Code verwendet. In der mitgelieferten Label-Bibliothek (#EQU-API.DEU) mit den jeweils aktuellen ROM-Adressen findet ebenfalls dieses Label Verwendung. In eigenen Programmen sollte man alle OS Funktionen (und Systemvariablen) stets mit diesen Labels ansprechen. Dadurch wird der Quellcode einheitlich. Außerdem KÖNNTEN sich in zukünftigen OS Versionen die Startadressen der OS Funktionen ändern. Siehe Datei **'#EQU-API.DEU'**.

3. ROM-Nummer: Hier steht die logische Nummer des OS ROMs, in der die OS Funktion zu finden ist. Die in dieser Datei beschriebenen OS Funktionen kommen allerdings nur in ROM M vor.

4. Startadresse: Gibt die Einsprung-Adresse der OS Funktion an.

5. Einsprungbedingungen: Die Einsprungbedingungen der OS Funktion werden beschrieben und es wird sowohl auf Registerinhalte als auch auf RAM-Variablen eingegangen.

6. Aussprungbedingungen: Die Aussprungbedingungen der OS Funktion werden beschrieben und es wird sowohl auf Registerinhalte als auch auf RAM-Variablen eingegangen.

7. Manipuliert: Es werden alle manipulierten oder zerstörten Register und RAM-Variablen wiedergegeben. Manchmal werden auch manipulierte Peripheriebausteine angegeben.

8. Beschreibung: Es folgt eine vollständige Erklärung der Funktionen der beschriebenen OS Funktion.

9. Bitte Beachten: Es werden einige wichtige Details kurz angeschnitten. Dies ist besonders wichtig, denn da alle OS Funktion kompromißlos auf Höchstgeschwindigkeit getrimmt worden sind, kann es bei falscher Handhabung zu Problemen mit der Systemstabilität kommen.

Die im folgenden beschriebenen OS Funktionen dienen der Verwaltung der SD-Karte der M4 Erweiterung. Sie liegen alle im FutureOS ROM M.

Die jeweils aktuelle Version dieser Datei 'API-M-DE.DOK' ist auch im Internet erhältlich. Siehe FutureOS Homepage: http://www.FutureOS.de (unter Downloads)

M4: Aufruf einer OS Funktion des M4 XROMs via Funktionsnummer

Kurzbeschreibung: Mittels einer Funktionsnummer wird eine OS Funktion des XROMs M aufgerufen. Anschließend wird ROM A eingeblendet.

Label: API_M4

ROM-Nummer: M

Startadresse: &C0F7

Einsprungbedingungen: Register L enthält die Funktionsnummer

Aussprungbedingungen: Bei gesetztem Carry Flag wurde die Funktion aufgerufen. Alle anderen Parameter hängen von der aufgerufenen OS Funktion des M4 XROMs ab.

Manipuliert: F, H, ROM A ist eingeblendet, und das, was durch die aufgerufene OS Funktion verändert wurde.

Beschreibung: Dieser Einsprung dient dazu, von den anderen OS ROMs A-D aus, eine Funktion in ROM M aufrufen zu können. Sie kann aber auch vom Anwender sinnvoll verwendet werden.

Dabei wird die Funktionsnummer der aufzurufenden Funktion in ROM M mittels des Registers L übergeben. Zulässige Werte sind dabei ein Vielfaches von drei. Also z.B.: 0, 3, 6, 9, 12, 15, 18 etc.

Nach dem Aufruf einer Funktion in ROM M wird das Carry Flag gesetzt und anschließend das OS ROM A eingeblendet.

Bitte Beachten: Beim Rücksprung wurde FutureOS ROM A eingeblendet.

M4: Einlesen des Haupt-Inhaltsverzeichnis der M4-SD-Karte

Kurzbeschreibung: Das Haupt-Inhaltsverzeichnis der M4 SD-Karte wird eingelesen.

Label: GET_M4D

ROM-Nummer: M

Startadresse: &FC81

Einsprungbedingungen: Ein freies 16 KB Erweiterungs-RAM (E-RAM) muß vorhanden sein. Es wird für die Pufferung der M4 SD-Karten DIRs reserviert.

Die Variable FN_LR (=&BE11) entscheidet, ob in den ersten (=&00) bzw. in den zweiten (=&20) E-RAM Inhaltsverzeichnispuffer gelesen wird.

Aussprungbedingungen: Enthält die RAM Variable 'M4_ERAM' den Wert 0 so konnte kein Inhaltsverzeichnis eingelesen werden, denn es ist kein 16 KB E-RAM Puffer frei. Ansonsten wurde das Haupt-Inhaltsverzeichnis eingelesen.

Der (neu) reservierte E-RAM Block der M4 SD-Karte wurde eingeblendet.

Manipuliert: AF, BC, DE, HL, AF', BC', DE', HL', IX, REG16_2-4, L_RAM und der E-RAM Status. Der E-RAM Block der M4 SD-Karte wurde manipuliert.

Beschreibung: Diese OS Funktion erlaubt es einen der beiden Puffer für M4 SD-Karten Inhaltsverzeichnisse auf das Haupt-Inhaltsverzeichnis (auch Root genannt) zurückzusetzen. Dabei wird das Inhaltsverzeichnis sowohl gelesen als auch sortiert.

Welcher der beiden Puffer auf das Haupt-DIR zurückgesetzt wird entschiedet die RAM Variable FN_LR, die entweder den Wert &00 bzw. &20 enthalten muss.

Falls kein 16 KB E-RAM Puffer für die M4-DIRs reserviert worden ist, dann wird das automatisch getan.

Bitte Beachten: Ein 16 KB E-RAM zur Pufferung des M4 SD-DIRs muss vor Aufruf dieser OS Funktion zur Verfügung stehen.

M4: Einlesen eines Unter-Inhaltsverzeichnis der M4 SD-Karte

Kurzbeschreibung: Ein Unter-Inhaltsverzeichnis (Sub-DIR) der M4 SD-Karte wird eingelesen. Dabei muß zuvor bereits ein E-RAM für die M4 DIR-Puffer reserviert worden sein (d.h. es wurde zuvor schon ein DIR gelesen z.B. das Hauptverzeichnis mittels OS Funktion GET_M4D).

Label: GET_M4S

ROM-Nummer: M

Startadresse: &FC84

Einsprungbedingungen: Ein 16 KB Erweiterungs-RAM (E-RAM) wurde bereits für die Arbeit mit der M4 SD-Karte reserviert, z.B. mittels GET_M4D.
Die Variable FN_LR (=&BE11) entscheidet, ob in den ersten (=&00) bzw. in den zweiten (=&20) E-RAM Inhaltsverzeichnispuffer gelesen wird.
Ab Adresse &4100 (lesen in Puffer 1) bzw. &6100 (lesen in Puffer 2) des M4 E-RAMs befindet sich der komplette Pfad-Name des Sub-DIRs.

Aussprungbedingungen: Ein Unter-Inhaltsverzeichnis wurde ins M4 E-RAM eingelesen und sortiert.
Der E-RAM Block der M4 SD-Karte ist eingeblendet.

Manipuliert: AF, BC, DE, HL, AF', BC', DE', HL', IX, REG16_2-4, L_RAM und der E-RAM Status.
Der E-RAM Block der M4 SD-Karte wurde eingeblendet und manipuliert.

Beschreibung: Diese OS Funktion liest ein Unter-Inhaltsverzeichnis der M4 SD-Karte ein. Dabei wird das Inhaltsverzeichnis auch sortiert.
In welchen der beiden Puffer das Sub-DIR gelesen wird, entscheidet die RAM Variable FN_LR, die entweder den Wert &00 bzw. &20 enthalten muss.
Enthält FN_LR den Wert &00, so muss sich auch der Pfad-Name des Sub-DIRs ab Adresse &4100 im M4 E-RAM befinden. Wird jedoch in den zweiten Puffer gelesen, so enthält FN_LR den Wert &20 und der Pfad-Name befindet sich ab Adresse &6100 im M4 E-RAM.

Die RAM Auswahl des M4 E-RAM befindet sich in RAM Variable M4_ERAM. Dabei sind Werte von &C4, &C5 bis &FF möglich, ganz so wie es beim 16 KB RAM Banking des CPC üblich ist.

Es kann durch: LD B,&7F:LD A,(M4_ERAM):OUT (C),A eingeblendet werden.

Der Pfad-Name muss sich in diesem E-RAM befinden.

Beispiel eines Namens: '**/FutureOS**'

Bitte Beachten: Ein 16 KB E-RAM zur Pufferung des M4 SD-DIRs muss vor Aufruf dieser OS Funktion bereits reserviert worden sein. Dieses E-RAM muss in RAM Variable M4_ERAM vermerkt worden sein (&C4...&FF).

M4: Wechseln des M4 Inhaltsverzeichnisses

Kurzbeschreibung: Eines der beiden M4 (Unter-)Inhaltsverzeichnisse wird gewechselt.

Labels: M4_CDIR (Name im M4 E-RAM) bzw. M4_CD (Name ab HL)

ROM-Nummer: M

Startadressen: &FC87 (M4_CDIR) bzw. &FC8A (M4_CD)

Einsprungbedingungen: Das Haupt-Inhaltsverzeichnis (Root) der M4 SD-Karte muss bereits eingelesen worden sein.
Die RAM Variable FN_LR (=&BE11) definiert, ob das Inhaltsverzeichnis für Puffer 1 (=&00) bzw. Puffer 2 (=&20) geändert werden soll.

- Bei Aufruf mittels M4_CDIR befindet sich der volle Name das neuen M4 DIRs ab Adresse &4100 (Puffer 1) bzw. &6100 (Puffer 2).
- Bei Aufruf mittels M4_CD befindet sich der volle Name das neuen M4 DIRs ab Register HL im Speicher.

Aussprungbedingungen: A = &00 bei Erfolg.
A = &FF, wenn das M4 DIR nicht gefunden wurde.

Manipuliert: AF, BC, DE, HL und das aktuelle M4 DIR.
Der Speicher von &BE50 bis &BE61 wurde manipuliert.

Beschreibung: Diese beiden OS Funktionen des M4 ROMs können dazu genutzt werden, das Inhaltsverzeichnis der M4 SD-Karte zu wechseln.

Dabei entschiedet RAM Variable FN_LR darüber, ob das erste (&00) oder zweite (&20) gepufferte M4 Verzeichnis geändert und aktiviert werden soll. Der neue Name wird entweder im M4 DIR E-RAM übergeben (M4_CDIR) oder HL zeigt darauf (M4_CD). Der Name wird immer durch das Byte &00 abgeschlossen.

Beachte: Das Inhaltsverzeichnis wird zwar gewechselt, aber nicht eingelesen und sortiert. Dazu würde man OS Funktion GET_M4S nutzen.

Bitte Beachten: Es muß bereits ein E-RAM für die Pufferung der M4 DIRs reserviert worden sein. Dies geschieht z.B. durch einmaliges Einlesen des Haupt-DIRs der M4 SD-Karte.

Das Verzeichnis wird gewechselt, NICHT aber eingelesen!

M4: Zeige den Header der 1. markierten M4 Datei auf dem Bildschirm an

Kurzbeschreibung: Der Datei-Header der ersten markierten Datei der M4 SD-Karte wird auf dem Bildschirm angezeigt.

Label: DHED_M4

ROM-Nummer: M

Startadresse: &FC8D

Einsprungbedingungen: Eine Datei der M4 SD-Karte sollte markiert sein.

Aussprungbedingungen:
XL = &B3: Ein Header ist vorhanden und wird angezeigt. MODE 1 und das untere ROM sind nun aktiv.
XL = &8C: Es wurde kein Header im RAM ab &B400 gefunden. Es wurde eine Fehlermeldung ausgegeben und auf MODE 2 geschaltet, das untere ROM ist aktiv.

Manipuliert: AF, BC, DE, HL, AF', BC', DE', HL', IX, REG08_1-2, REG16_6 bis REG32_1, REG_PC, die RAM Bereiche von &B400 bis &B5FF, &B600 bis &B6FF und von &BE20 bis &BE36. Das untere ROM wurde aktiv geschaltet. Der Bildschirm-Modus ist 1 oder 2.

Beschreibung: Diese OS Funktion liest die ersten 512 Bytes der ersten markierten Datei der M4 SD-Karte ab Adresse &B400 ins RAM. Danach wird der Datei Header im MODE 1 angezeigt.

Beim Aufruf dieser OS Funktion sollte eine Datei der SD-Karte markiert sein, ansonsten kehrt DHED_M4 mit Fehlerkode XL = &8C zurück. In diesem Fall wurde auf MODE 2 geschalten und das untere ROM ist aktiv.

Kehrt DHED_M4 mit XL = &B3 zurück, so wird der Datei-Header auf dem Bildschirm angezeigt. Es ist auf MODE 1 geschalten und das untere ROM ist aktiv geschalten.

Bitte Beachten: Nach dem Aufruf diese OS Funktion ist das untere ROM eingeschalten. Der Aufruf muss also von einer Adresse oberhalb von &4000 erfolgen. Auch ist es nötig, den Bildschirm-Modus neu zu setzen.

M4: Lese den Header der ersten markierten Datei von M4 SD-Karte

Kurzbeschreibung: Der Datei-Header der ersten markierten Datei der M4 SD-Karte wird gelesen und ab &B400 im RAM abgelegt.

Label: LADAH_M4

ROM-Nummer: M

Startadresse: &FC90

Einsprungbedingungen: Ein M4 SD-Karten DIR muss eingelesen sein. Eine Datei sollte markiert sein.

Aussprungbedingungen: A = &FF: Der Datei-Header steht ab &B400 im RAM.
In diesem Fall enthalten die OS Variablen (REG08_1) und (REG_PC+1) die Nummer des Quell-Mediums: &0D für SD-Karten Verzeichnis 'N' oder &0E für das zweite SD-Karten Verzeichnis 'O'. Weiterhin enthält die OS Variable REG08_2 die E-RAM I/O Nummer des SD-Karten Verzeichnis E-RAM Puffers &C4...&FF. Und ab REG16_6 steht der Name der SD-Karten Datei im Format: '**`N--:FileNameExt`**' bzw. '**`O--:DateinmeExt`**'.

A = &00: Entweder es wurde keine markierte Datei auf der SD-Karte gefunden. Oder es gab einen Fehler beim Öffnen der Datei.

Manipuliert: AF, BC, DE, HL, AF', BC', DE', HL', IX, REG08_1-2, REG16_6 bis REG32_1, REG_PC, die RAM Bereiche von &B400 bis &B5FF, &B600 bis &B6FF und von &BE20 bis &BE36.

Beschreibung: Diese OS Funktion liest die ersten 512 Bytes der ersten markierten Datei der M4 SD-Karte ab der Adresse &B400 ins RAM. Enthält die Datei einen Header so befindet er sich als ab &B400 im RAM.

Beim Aufruf dieser OS Funktion sollte eine Datei der SD-Karte markiert sein, ansonsten kehrt LADAH_M4 mit Fehlerkode A = &00 zurück.

Kehrt LADAH_M4 mit A = &FF zurück, so wurde ein potenzieller Header erfolgreich nach &B400 gelesen. Ob es sich allerdings wirklich um einen Header handelt, muss mittels der Prüfsumme untersucht werden (siehe TST4HED in diesem ROM M oder TST_HED in ROM B).

Bitte Beachten: Einige ASCII-Dateien auf M4 SD-Karte können einen zusätzlichen Datei-Header enthalten, der mit der eigentlichen Datei nichts zu tun hat. Dies kann mittels OS Funktion T_HED_A (in ROM M) untersucht werden.

M4: Lese den Header einer Datei von M4 SD-Karte

Kurzbeschreibung: Der Datei-Header einer durch ihren Namen definierten Datei der M4 SD-Karte wird gelesen und ab &B400 im RAM abgelegt.

Label: M4_R_HED

ROM-Nummer: M

Startadresse: &FC93

Einsprungbedingungen: HL = Name der Datei
Ein M4 DIR muss eingelesen sein. Die Datei befindet sich im aktuellen Verzeichnis.

Aussprungbedingungen: A = &00: Der Datei-Header steht ab &B400 im RAM
A = &FF: Es gab einen Fehler beim Öffnen der Datei.

Manipuliert: AF, BC, DE, HL, AF', BC', DE', HL', der RAM Bereich von &B400 bis &B5FF und der RAM Bereich von &BE20 bis &BE36.

Beschreibung: Diese OS Funktion liest die ersten 512 Bytes einer Datei der M4 SD-Karte ins RAM ab der Adresse &B400. Enthält die Datei einen Header so befindet er sich ab da im RAM.

Zum Aufruf dieser OS Funktion muss Register HL auf den Namen der Datei zeigen. Des weiteren muß das Inhaltsverzeichnis, in dem sich die Datei befindet, aktiv sein (siehe z.B. M4_CDIR bzw. M4_CD).

Kehrt M4_R_HED mit A=&00 zurück, so wurde ein potenzieller Header gelesen. Ob es sich allerdings wirklich um einen Header handelt, muss noch mittels der Prüfsumme untersucht werden (siehe TST4HED in diesem ROM M oder TST_HED in ROM B).

Bitte Beachten: Einige ASCII-Dateien auf M4 SD-Karte können einen zusätzlichen Datei-Header enthalten, der mit der eigentlichen Datei nichts zu tun hat. Dies kann mittels OS Funktion T_HED_A (in ROM M) untersucht werden.

M4: Teste ob eine Datei einen Header enthält

Kurzbeschreibung: Es wird getestet, ob eine Datei einen Header enthält.

Label: TST4HED

ROM-Nummer: M

Startadresse: &FC96

Einsprungbedingungen: DE = Zeiger auf potenziellen Header = &XX00/80!

Aussprungbedingungen: Bei gesetztem Zero Flag wurde ein klassischer Datei-Header gefunden, so wie er von der CPC-Firmware und FutureOS genutzt wird. In diesem Fall enthält Register BC die Prüfsumme aus den Header-Daten selbst.
Bei gelöschtem Zero Flag konnte kein Header erkannt werden.

Manipuliert: AF, BC, DE, HL und XL

Beschreibung: Um zu testen, ob eine Datei einen Datei-Header enthält kann diese OS Funktion verwendet werden.

Vor dem Aufruf dieser OS Funktion TST4HED muß der potenzielle Header eingelesen werden. Die kann z.B. mittels OS Funktion M4_R_HED getan werden. Weiterhin ist Register DE mit dem Zeiger auf den Header zu laden (in diesem Beispiel &B400). Dabei muß sich der Header immer innerhalb einer 256 Bytes Seite (&XX00) befinden!

Nach der Rückkehr dieser OS Funktion gibt das Zero Flag Aufschluss über die Existenz eines Datei-Headers. Ist es gelöscht, so fehlt der Header.

Ist das Zero Flag nach Ende von TST4HED jedoch gesetzt, so wurde ein klassischer Datei-Header gefunden, so wie er von der CPC Firmware / Amsdos und FutureOS genutzt wird. In diesem Fall enthält Register BC die Prüfsumme aus dem Datei-Header.

Bitte Beachten: Achtung! Enthält Register BC nach Rücksprung den Wert &0000, so wurde ein Header zwar erkannt, ist aber nicht vorhanden!
Wenn eine Datei in ihren ersten 128 Bytes nur &00 Bytes enthält (dies kommt z.B. bei Bildern vor), dann wird fälschlicherweise ein Header erkannt. In diesem Fall ist jedoch BC = &0000!

M4: Teste ob eine ASCII-Datei einen M4 spezifischen Header enthält

Kurzbeschreibung: Es wird getestet, ob eine ASCII-Datei einen Header enthält, der von der M4 Karte zusätzlich generiert wurde.

Label: T_HED_A

ROM-Nummer: M

Startadresse: &FC99

Einsprungbedingungen: HL zeigt auf den Anfang des potenziellen Headers

Aussprungbedingungen: Bei gesetztem Zero Flag wurde ein zusätzlicher, von der M4 Karte generierter Header gefunden.
Bei gelöschtem Zero Flag wurde kein zusätzlicher M4 spezifischer Header gefunden. Diese Datei könnte jedoch einen echten Header besitzen, so wie er von der Firmware bzw. FutureOS her bekannt ist.

Manipuliert: AF und HL

Beschreibung: Manche ASCII-Dateien, die auf die SD-Karte der M4 Erweiterung geschrieben wurden, enthalten einen zusätzlichen &80 Bytes Header, der auf anderen Speicher-Medien nicht generiert wird. Um mit diesen ASCII-Dateien arbeiten zu können ist es wichtig zu wissen, ob vor eine Datei solch ein M4-spezifischer Header gesetzt wurde.
Dafür dient diese OS Funktion T_HED_A. Sie testet, ob sich am Dateianfang solch ein M4-Header befindet. Vor dem Aufruf ist es nötig den potenziellen Header mittels z.B. OS Funktion M4_R_HED einzulesen, um ihn untersuchen zu können.

Vor Aufruf von T_HED_A ist Register HL mit dem Zeiger auf den Header zu laden (in diesem Beispiel &B400).

Nach Rückkehr der Funktion gibt das Zero Flag Aufschluss über die Existenz eines M4-spezifischen Headers. Ist es gelöscht, so fehlt der zusätzliche Header. In diesem Fall kann die Datei jedoch einen normalen Header enthalten, dies ist gesondert zu testen z.B. mittels OS Funktion TST4HED in diesem ROM M oder mittels TST_HED auf ROM B.
Ist das Zero Flag nach Ende von T_HED_A jedoch gesetzt, so sollte man die ersten 128 Bytes der untersuchten ASCII-Datei ignorieren, da sie eben diesen zusätzlichen M4-spezifischen Header enthalten.

Bitte Beachten: Eine Datei, die keinen zusätzlichen M4-spezifischen Header enthält, kann einen ganz normalen Header enthalten, so wie man es von der normalen Firmware des CPC oder von FutureOS her gewohnt ist enthalten.

M4: Laden einer durch ihren Namen definierten Datei

Kurzbeschreibung: Eine Datei, deren Name bekannt ist, wird von der M4 SD-Karte geladen. Dabei stehen zwei Verzeichnisse zur Verfügung (N/O).

Label: LADE_M4

ROM-Nummer: M

Startadresse: &FC0C

Einsprungbedingungen: Das Inhaltsverzeichnis der M4 SD-Karte, in dem sich die zu ladende Datei befindet, sollte vorhanden und im E-RAM gepuffert sein.

A' = Medium (13-14 = &0D-&0E) von dem die Datei geladen werden soll.
Ist das 8. Bit gesetzt (&8D / &8E), so wird der Header ignoriert

DE = Zeiger auf den M4 SD-Karten Dateinamen. Der Name kann bis zu 27 Zeichen enthalten.
Anschließend folgt der Punkt '.', dann drei Zeichen Erweiterung und Byte &00.
Oder der Dateiname ist Floppy kompatibel, er besteht dann aus User-Nummer (1 Byte), Name (8 Bytes) und der Erweiterung (3 Bytes) der Datei: 'UFileNameExt'.

Falls die zu ladende Datei keinen Header hat, oder der Header ignoriert werden soll, sind außerdem folgende Daten anzugeben:
REG08_4 = Ladeart (0,1,2,3) - siehe OS Funktion LADEN4 in ROM M
REG16_3 = Ladeadresse im RAM bzw. E-RAM, 16 Bit
AKT_RAM = physikalische E-RAM Nummer, &7FC4, &C5 .. &7FFF .. &78FF
RAMCHAR = Bildschirm MODE währed des Ladens

Aussprungbedingungen: Der Akku gibt Aufschluß über den Erfolg des Ladevorgangs:

- A = &00 ==> Das Inhaltsverzeichnis E-RAM von Medium N bzw. O ist nicht vorhanden, deshalb kann auch keine Datei geladen werden
- A = &01 ==> Das Medium, von dem die Datei geladen werden soll, ist nicht aktiv. Die Datei kann nicht geladen werden
- A = &02 ==> Die Datei existiert nicht im aktuellen Inhaltsverzeichnis des angegebenen Mediums. Oder es gab einen Fehler beim Laden
- A = &FF ==> Die Datei wurde ordnungsgemäß geladen

(REG_PC+1) = Nummer (13 oder 14) des Mediums (SD-Karte Verzeichnis N oder O) von dem die Datei geladen wurde
(FN_LR) = &00 / &20 je nachdem ob von Medium N bzw. O geladen wurde

Manipuliert: AF, BC, DE, HL, AF', BC', DE', HL', IX, IY und die RAM-Variablen REG08_0-3, 6 REG16_1, REG_PC, AKT_RAM und FN_LR. Außerdem die RAM Bereiche &B400-&B6FF, &BE20-&BEAF und &BC00-&BC7F, sowie der RAM-Status und der Bildschirm-Modus.

Beschreibung: Diese OS Funktion dient dazu, eine beliebige Datei zu laden. Die Datei wird durch ihr Quell-Medium und ihren Namen eindeutig identifiziert. Der Name kann entweder im regulären FutureOS Format vorliegen oder im M4 Format, bei Bedarf wird konvertiert.

Das Quell-Medium (&0D bzw. &0E für das aktuelle Verzeichnis in N bzw. O) wird in A' angegeben. Ist das 8. Bit gesetzt (A' = &8D / &8E), so wird ein eventueller Dateiheader ignoriert. Ansonsten entscheidet der Header wohin die Datei geladen wird. Ohne Datei-Header entscheiden folgende RAM-Variablen, an welche Adresse, in welchen RAM Block, und auf welche Art die Datei geladen wird.

REG08_4 = Ladeart: Die Ladeart &02 läd die Datei in den Hauptspeicher. Die Ladeart &03 läd die Datei ins E-RAM (siehe LADEN4)

REG16_3 = Ladeadresse im RAM bzw. E-RAM, 16 Bit

AKT_RAM = physikalische E-RAM Nummer: &7FC4 .. &7FFF .. &78FF (4 MB!)

Wurde die Datei korrekt geladen, so kehrt die OS Funktion mit &FF im Akku zurück, andernfalls enthält A den Fehlerkode.

REG_PC+1 enthält die Nummer des Mediums (&0D / &0E), von der die Datei geladen wurde. Dabei entsprechen &0D / &0E den Medien N / O.

Bitte Beachten: Diese Funktion wird von ROM C Funktion 'LADEN' genutzt

M4: Einlesen einer Datei von bis zu 64 KB

Kurzbeschreibung: Einlesen einer Datei von bis zu 64 KB Länge von der SD-Karte der M4 Erweiterung in die aktuelle RAM-Konfiguration.

Label: M4_R_64K

ROM-Nummer: M

Startadresse: &FC9C

Einsprungbedingungen: Es wird die aktuelle RAM Konfiguration benutzt
HL = Zeiger auf den kompletten Pfad+Dateinamen der zu lesenden Datei.
Der Pfad+Name wird vom Byte &00 abgeschlossen
DE = Zieladresse der zu lesenden Datei der M4 SD-Karte

Aussprungbedingungen:

- A = &00: Die Datei wurde erfolgreich gelesen. In diesem Fall befindet sich der Datei-Header ab &BC00 im RAM
- A = &FD: Fehler beim Öffnen der Datei (DE ist erhalten)
- A = &FE: Die Datei ist größer als 64 KB, also zu groß zum Laden
- A = &FF: Die Datei ist nicht vorhanden, oder wurde nicht gefunden

Manipuliert: AF, BC, DE, HL, AF', BC', DE', HL', der RAM Bereich von &BC00 bis &BC7F und der RAM Bereich von &BE20 bis &BE7F

Beschreibung: Diese OS Funktion dient zum Einlesen einer Datei von bis zu 64 KB von der M4 SD-Karte ins RAM. Dabei wird die aktuelle RAM-Konfiguration verwendet.

Dabei wird ein Zeiger auf den Pfad+Dateinamen im Register HL übergeben und die Zieladresse der Datei wird in Register DE übergeben.

Nach dem erfolgreichen Laden der Datei wird Register A auf den Wert &00 gesetzt. Enthält der Akku einen anderen Wert, so ist beim Laden ein Fehler aufgetreten.

Um eine Datei von mehr als 64 KB zu laden, kann man die OS Funktion 'M_R_X16' verwenden.

Bitte Beachten: nur Dateien bis zu maximal 64 KB verwendbar

M4: Einlesen einer Datei von bis zu 4 MB

Kurzbeschreibung: Einlesen einer Datei von bis zu 4 MB Länge von der SD-Karte der M4 Erweiterung in das Erweiterungs-RAM (E-RAM).

Label: M4_R_X16

ROM-Nummer: M

Startadresse: &FC9F

Einsprungbedingungen: AKT_RAM: Erster E-RAM Block in den gelesen wird
HL = Zeiger auf den kompletten Pfad+Dateinamen der zu lesenden Datei.
Der Pfad+Name wird vom Byte &00 abgeschlossen
DE = Zieladresse der zu lesenden Datei der M4 SD-Karte, kleiner &8000

Aussprungbedingungen:

- A = &00: Die Datei wurde erfolgreich gelesen
- A = &FD: Fehler beim Öffnen der Datei (DE ist erhalten)
- A = &FE: Die Datei ist größer als 4 MB, also zu groß zum Laden
- A = &FF: Die Datei ist nicht vorhanden, oder wurde nicht gefunden

Manipuliert: AF, BC, DE, HL, AF', BC', DE', HL', IX, die OS Variable AKT_RAM und der RAM Bereich von &BE20 bis &BEAF.

Beschreibung: Diese OS Funktion dient zum Einlesen einer Datei von bis zu 4 MB von der M4 SD-Karte ins Erweiterungs-RAM. Dabei wird die E-RAM Konfiguration aus OS Variable AKT_RAM verwendet. Sie gibt den ersten 16 KB E-RAM Block an, in den gelesen werden soll.

In Register HL wird ein Zeiger auf den Pfad+Dateinamen übergeben. Und die Zieladresse der Datei wird in Register DE übergeben. Die Ziel-Adresse muss kleiner &8000 sein. Üblicherweise befindet sie sich im Bereich von &4000 bis &7FFF.

Nach dem erfolgreichen Laden der Datei wird Register A auf den Wert &00 gesetzt. Enthält der Akku einen anderen Wert, so ist beim Laden ein Fehler aufgetreten.

Um Dateien größer 4 MB zu lesen ist die OS Funktion 'TEILA4' nützlich.

Bitte Beachten: nur Dateien bis zu maximal 4 MB verwendbar

M4: (Teilweise) Laden einer markierten Datei in Kurz-Zeit-Speicher

Kurzbeschreibung: Eine Datei wird, wenn möglich komplett, von der M4 SD-Karte in alle freien 16 KB Kurz-Zeit-Speicher E-RAMs (KZS) geladen.

Label: TEILA4 (erstes Laden) // TEILB4 (Rest(e) nachladen)

ROM-Nummer: M

Startadresse: &FCA2 (TEILA4) // &FCA5 (TEILB4)

Einsprungbedingungen:

TEILA4: System- und XRAM_??-Variablen enthalten korrekte Werte

TEILB4: Der Inhalt der RAM Variablen REG08_2, REG_IY, REG_SP, REG_PC+1 wurde seit dem Aufruf von TEILA4 nicht verändert!

Aussprungbedingungen: (gilt für TEILA4 und TEILB4)

- A = &00 ==> Die Datei wurde komplett in die KZS E-RAMs geladen
- A = &F0 ==> Die Datei wurde teilweise in die KZS E-RAMs geladen, alle KZS E-RAMs sind voll beladen. Der Rest der Datei kann mittels TEILB4 nachgeladen werden.
- A = &FC ==> Fehler beim 16 KB Block Laden von SD-Karte
- A = &FD ==> Fehler beim Öffnen der Datei von SD-Karte
- A = &FE ==> Kein (KZS) E-RAM frei, es kann also nichts geladen werden
- A = &FF ==> Keine Datei der M4 SD-Karte markiert

REG08_1 = REG_PC+1 = Nummer des Quell-Mediums = &0D / &0E für 'N'/'O'
REG08_2 = E-RAM-Block der die beiden SD-Karten Verzeichnisse puffert
REG_BC1 = / Bits 0-15 / 32 Bit Dateilänge der geladenen Datei
REG_DE1 = / Bits 16-31 /
REG_SP+1 = Datei-ID = Datei-Deskriptor der geöffneten Datei (SD-Karte)

Manipuliert: AF, BC, DE, HL, AF', BC', DE', HL', IX, IY, die OS RAM-Variablen REG08_0-5, REG16_1, REG16_6...REG32_1 (nur TEILA4), AKT_RAM, REG_IY, REG_SP, REG_PC, XRAM_C4..FF und der RAM-Status

Beschreibung: Diese OS Funktion ermöglicht es, eine beliebig lange, markierte Datei von der M4 SD-Karte in die Kurz-Zeit-Speicher (KZS) Blöcke (= 16 KB E-RAMs) zu laden. Reicht die Anzahl vorhandener KZS nicht aus, so wird die Datei nur soweit geladen, bis alle KZS Blöcke beladen sind. Der Dateirest kann anschließend nachgeladen werden, notfalls in mehreren Schritten. Es wird die erste markierte Datei geladen.

Passt die Datei komplett in den Speicher, so kehrt die OS Funktion mit A=&00 zurück. Es muß also kein Dateirest nachgeladen werden. War die Datei zu groß, um sie komplett zu laden, so wurden alle KZS mit einem Teil der Datei beladen, und die OS Funktion kehrt mit A=&F0 zurück.
(Enthält der Akku einen anderen Wert, so ist ein Fehler aufgetreten).

Nach dem Laden eines Teils ist dafür Sorge zu tragen, dass die RAM-Variablen REG08_2, REG_IY, REG_PC+1 und REG_SP unverändert bleiben.
Nur dann kann der Dateirest mit TEILB4 nachgeladen werden. Die Aussprungbedingungen von TEILB4 entsprechen denen von TEILA4. Unter Umständen ist noch ein weiterer Dateirest zu laden, dies hängt von der Gesammtlänge der Datei ab.

Bitte Beachten: Um TEILB4 korrekt aufzurufen, müssen die RAM-Variablen REG08_2, REG_IY, REG_SP und REG_PC+1 nach dem Aufruf von TEILA4 unverändert bleiben!

M4: Datei entsprechend ihres Headers laden / Kein Header --> OS

Kurzbeschreibung: Eine Datei mit Datei-Header wird entsprechend der Daten ihres Datei-Headers geladen. Und zwar von der M4 SD-Karte.
Besitzt die zu ladende Datei keinen Header so wird ins OS gesprungen.

Label: X_LADEN_NO

ROM-Nummer: M

Startadresse: &FCA8

Einsprungbedingungen: Der E-RAM Puffer der M4 SD-Karte ist zwischen &4000 und &7FFF eingeblendet (siehe OS Variable M4_ERAM).
HL = Zeiger auf den Namen der Datei, die geladen werden soll
FN_LR = &00 (Laden vom SD-Karte 'N') oder &20 (Laden von SD-Karte 'O')
YH = &FF ---------> Kein Header --> Sprung nach KLICK in ROM D
YH = &00-&FE --> Kein Header --> Sprung in die M4 Shell in ROM M
RAMCHAR = Bildschirm MODE währed des Ladens

Aussprungbedingungen: Besitzt die zu ladende Datei wider Erwarten keinen Header, so springt diese OS Funktion entweder nach KLICK in ROM D (YH war &FF) oder in die Shell der M4 XROMs M (YH war kleiner &FF).
A = &00 --> Die Datei wurde entsprechend ihres Headers geladen
A > &00 --> Es ist ein Fehler beim Laden der Datei aufgetreten
Der in RAMCHAR angegebene Bildschirm-MODE wurde aktiviert

Manipuliert: AF, BC, DE, HL, AF', BC', DE', HL', IX, REG08_3, REG16_1, AKT_RAM, REG_PC, das Ziel RAM / E-RAM der geladenen Datei, das RAM von &B400-&B6FF, von &BC00-&BC7F und von &BE20 bis &BEAF. Außerdem der Bildschirm-Modus und die RAM / E-RAM Konfiguration

Beschreibung: Diese OS Funktion dient dazu eine Datei von der SD-Karte der M4 Erweiterung zu lesen. Dabei wird die Datei entsprechend ihrer Header-Daten geladen. Entweder in den Hauptspeicher oder in das E-RAM.
Sollte die Datei jedoch keinen Header besitzen, so wird ins OS zurück gesprungen, entweder via 'KLICK' (ROM D) oder in die M4 Shell. Dies hängt von YH ab.
Vor dem Aufruf dieser OS Funktion zeigt HL auf den Datei-Namen einer Datei der M4 SD-Karte. Das M4 E-RAM (siehe OS Variable M4_ERAM) muß bereits eingeblendet sein, denn darin befinden sich die Datei-Namen.
Die OS Variable FN_LR gibt an, ob das Erste bzw. Zweite gepufferte Verzeichnis der M4 SD-Karte verwendet werden soll. Also ob von Medium N bzw. O geladen werden soll. Entsprechend muss FN_LR den Wert &00 bzw. &20 enthalten.
Um zu definieren, welcher Bildschirm MODE während des Ladens aktiv sein, soll kann dieser MODE (0-3) in die unteren zwei Bits der OS Variable RAMCHAR geschrieben werden.
Nach dem Laden der Datei gibt der Akku Aufschluß über der Erfolg: Ist A = &00, so war das Laden erfolgreich. Ansonsten passierte ein Fehler.

Bitte Beachten: Diese OS Funktion springt zurück ins OS, wenn die zu ladende Datei keinen Header besitzt!

M4: Markierte Datei in Hauptspeicher oder E-RAM laden

Kurzbeschreibung: Die erste markierte Datei wird in den Hauptspeicher oder das Erweiterungs-RAM (E-RAM) geladen.

Label: LADEN4

ROM-Nummer: M

Startadresse: &FCAB

Einsprungbedingungen: A = &FF --> Datei-Markierungs-Byte unverändert
REG08_3 definiert die Ladeart der Datei:
= 0 --> Laden in den Hauptspeicher ab &0000
= 1 --> Laden in E-RAM ab Adresse &0000 und dem ersten E-RAM (&C4)
= 2 --> Laden in den Hauptspeicher ab Adresse in OS Variable (REG16_1)
= 3 --> Laden in E-RAM aus OS Variable (AKT_RAM) ab Adresse (REG16_1)
REG16_1 = Zieladresse - nur bei Ladeart 2 und 3!
AKT_RAM = Erster E-RAM Block, in den geladen wird - nur bei Ladeart 3!
RAMCHAR = Bildschirm MODE während des Ladens

Aussprungbedingungen: OS Variable (REG_PC+1) enthält das Quellmedium
A = &00 --> Die Datei wurde entsprechend ihres Headers geladen
A > &00 --> Es ist ein Fehler beim Laden der Datei aufgetreten
Der in RAMCHAR angegebene Bildschirm-MODE wurde aktiviert.

Manipuliert: AF, BC, DE, HL, AF', BC', DE', HL', IX, REG08_1-3, REG16_1-2, REG16_6-REG32_1, AKT_RAM, REG_PC, das Ziel RAM / E-RAM der geladenen Datei, das RAM von &B600-&B6FF, von &BC00-&BC7F und von &BE20 bis &BEAF. Außerdem der Bildschirm-Modus, die Datei-Tagging Bytes und die RAM / E-RAM Konfiguration.

Beschreibung: Diese OS Funktion dient dazu, die erste markierte Datei von der SD-Karte der M4 Erweiterung zu lesen. Dabei wird die Datei in den Hauptspeicher oder in das E-RAM geladen, dies hängt vom Inhalt der OS Variable REG08_3 ab. Enthält der Akku beim Aufruf den Wert &FF, so wird das Datei-Markierungs-Byte erhalten, andernfalls wird der Status der Datei von 'markiert' auf 'benutzt' gesetzt (Normalfall). Bei den Ladearten 2 und 3 sind zusätzliche Parameter anzugeben (siehe oben).

Um zu definieren, welcher Bildschirm MODE während des Ladens aktiv sein soll, kann dieser MODE (0-3) in die unteren zwei Bits der OS Variable RAMCHAR geschrieben werden.

Nach dem Laden der Datei gibt der Akku Aufschluß über der Erfolg: Ist A = &00, so war das Laden erfolgreich. Ansonsten passierte ein Fehler.

Bitte Beachten: Die Datei wird entsprechend der in REG08_3 definierten Lade-Art geladen, der Header hat keinen Einfluß.

M4: Sichern einer Datei auf M4 SD-Karte

Kurzbeschreibung: Ein definierter Speicherinhalt (Hauptspeicher oder E-RAM) wird als Datei auf die SD-Karte der M4 Erweiterung gesichert.

Label: SICHRE

ROM-Nummer: M

Startadresse: &FC12

Einsprungbedingungen: Der SAVE-Modus ist in REG08_3 eingetragen:
(REG08_3 = &32 (="1"+1) ==> Vordergrund Programm sichern) / **aktuell**
(REG08_3 = &33 (="2"+1) ==> Hintergrund Programm sichern) / **inaktiv!**
REG08_3 = &34 (="3"+1) ==> Aus Hauptspeicher sichern
REG08_3 = &35 (="4"+1) ==> Aus Erweiterungs-RAM sichern
REG16_6+1 = Ziel-Medium, auf das gesichert werden soll:
Dieses Medium wird durch ein ASCII-Zeichen symbolisiert, Entweder 'N/n' (=&4E/&6E) oder 'O/o' (=&4F/&6F) für eines der SD-Karten Verzeichnisse

Bei SAVE-Modus 3(&34) oder 4(&35) sind weitere Daten anzugeben:
REG16_8 = Beginn des Namens der zu sichernden Datei. Entweder sind das 12 Bytes: die User-Nummer (1 Byte), Dateiname (8 Bytes) und Extension (3 Bytes). Dabei wird die User# ignoriert!
Oder das erste Byte (User) ist &FF, dann enthält REG16_9 den Zeiger auf einen regulären M4 SD Dateinamen: Name (bis zu 27 Zeichen), '.'-Zeichen und die Extension (3 Zeichen).
REG_IX = Quell-Adresse, ab der gespeichert werden soll (16 Bit)
AKT_RAM = Quell-Block, ab dem gesichert werden soll: &7FC0/&7FC4-&78FF
REG_IY = Datei-Länge in KB (16 Bit). Dateilänge: maximal 4 MB

Aussprungbedingungen: Datei wurde auf die M4 SD-Karte gespeichert und im entsprechenden Verzeichnis eingetragen. Oder es erfolgt ein Sprung ins OS, falls ein Fehler aufgetreten ist.
In jedem Fall ist der Hauptspeicher eingeblendet (AKT_RAM = &7FC0)

Manipuliert: AF, BC, DE, HL, AF', BC', DE', HL', IX, IY und die RAM Variablen REG08_1, AKT_RAM, REG_R, FN_LR, der RAM-Bereich von &B600 bis &B70F. Außerdem wurde die E-RAM-Konfiguration und eines der SD-Karten Inhaltsverzeichnisse verändert.

Beschreibung: Diese OS Funktion erlaubt es einen Speicherbereich als Datei in eines der beiden gepufferten Verzeichnisse der M4 SD-Karte zu schreiben (und im Inhaltsverzeichnis zu vermerken).

Die Variable REG08_3 gibt an wie gesichert werden soll, es kann entweder ein bereits eingelesenes Programm gesichert werden, oder ein beliebiger RAM / E-RAM Bereich.

In jedem Fall ist in REG16_6+1 das Laufwerk 'N' oder 'O' anzugeben, und zwar als ASCII-Wert von &4E/&6E (N/n) oder &4F/&6F (O/o).

Ab REG16_8 beginnt entweder der Name im klassischen Format: User(1) + Name(8) + Extension (3). Oder REG16_8 enthält das Byte &FF, in diesem Fall zeigt REG16_9 auf einen M4 SD-Karten Dateinamen im Format Name(x) + '.' Zeichen + Extension(3). Der Name sollte maximal 27 Zeichen enthalten. Die Extension darf maximal 3 Zeichen enthalten.

Bitte Beachten: Es kann nur gespeichert werden, wenn ein E-RAM für die M4 SD-Karte reserviert wurde und ein Ziel-Verzeichnis eingelesen ist.
Das Medium wird in REG16_6+1 als ASCII-Wert angegeben.
Tritt ein Fehler auf, so springt diese OS Funktion ins Desktop!

M4: Schreiben einer Datei von bis zu 64 KB

Kurzbeschreibung: Eine Datei von bis zu 64 KB Länge wird auf die SD-Karte der M4 Erweiterung geschrieben. Dabei wird die aktuelle RAM-Konfiguration verwendet.

Label: M4_W_64K

ROM-Nummer: M

Startadresse: &FCAE

Einsprungbedingungen: Es wird die aktuelle RAM-Konfiguration benutzt
HL = Zeiger auf den kompletten Pfad+Dateinamen der zu schreibenden Datei.
Der Pfad+Name wird vom Byte &00 abgeschlossen.
DE' = Länge der zu speichernden Datei in Bytes (max. 64 KB)
HL' = Quelladresse der zu speichernden Datei für die M4 SD-Karte

Aussprungbedingungen: A = &00: Die Datei wurde erfolgreich geschrieben
In diesem Fall ist das Zero Flag gesetzt und A' = Dateideskriptor
A = &FF: Beim Erstellen der Datei ist ein Fehler aufgetreten

Manipuliert: AF, BC, DE, HL, AF', BC', DE' und HL'

Beschreibung: Diese OS Funktion dient zum Sichern eines RAM Bereiches in eine Datei von bis zu 64 KB auf die M4 SD-Karte. Dabei wird die aktuelle RAM-Konfiguration verwendet.

Vor dem Aufruf dieser OS Funktion wird im Register HL ein Zeiger auf den Pfad+Dateinamen übergeben. DE' enthält die Dateilänge in Bytes und HL' die Start-Adresse, ab der geschrieben werden soll.

Nach dem erfolgreichen Schreiben der Datei enthält Register A den Wert &00. Enthält der Akku einen anderen Wert (A=&FF), so ist beim Schreiben ein Fehler aufgetreten.

Um eine Datei von mehr als 64 KB zu schreiben, kann man die OS Funktion 'M4_W_X16' verwenden.

Bitte Beachten: Nur Dateien bis zu maximal 64 KB können gesichert werden.

M4: Schreiben einer Datei von bis zu 4 MB

Kurzbeschreibung: Eine Datei von bis zu 4 MB Länge wird aus dem E-RAM auf die SD-Karte der M4 Erweiterung geschrieben.

Label: M4_W_X16

ROM-Nummer: M

Startadresse: &FCB1

Einsprungbedingungen: AKT_RAM: Erster E-RAM Block der geschrieben wird
HL = Zeiger auf den kompletten Pfad+Dateinamen der zu schreibenden Datei
Der Pfad+Name wird vom Byte &00 abgeschlossen
DE' = Länge der zu speichernden Datei in KB (max. 4096)
HL' = Quelladresse der zu speichernden Datei für die M4 SD-Karte

Aussprungbedingungen: Die Datei wurde erfolgreich geschrieben, wenn der Akku einen Wert kleiner &FF enthält, dieser Wert entspricht dann dem Dateideskriptor
A = &FF: Beim Erstellen der Datei ist ein Fehler aufgetreten

Manipuliert: AF, BC, DE, HL, AF', BC', DE', HL' und AKT_RAM

Beschreibung: Diese OS Funktion dient zum Sichern eines Erweiterungs-RAM Bereiches in eine Datei von bis zu 4 MB Länge auf die SD-Karte der M4 Erweiterung.

Vor dem Aufruf muß die E-RAM Konfiguration ab der geschrieben werden soll in die OS Variable AKT_RAM geschrieben werden. Sie gibt den ersten 16 KB E-RAM Block an aus dem gesichert werden soll. In Register HL wird ein Zeiger auf den Pfad+Dateinamen übergeben. DE' enthält die Dateilänge in KB (maximal 4096). Und HL' enthält die Start-Adresse, ab der geschrieben werden soll.

Nach dem erfolgreichen Schreiben der Datei enthält Register A den Wert des Dateideskriptors. Enthält der Akku jedoch den Wert &FF, so ist beim Schreiben ein Fehler aufgetreten.

Um eine Datei von mehr als 4 MB zu schreiben, kann man die OS Funktion 'TEISI4' verwenden.

Bitte Beachten: Es können nur Dateien bis zu maximal 4 MB gesichert werden.

M4: (Teilweise) Sichern einer Datei aus den Kurz-Zeit-Speicher E-RAMs

Kurzbeschreibung: Eine Datei wird (teilweise) aus dem KZS gesichert.

Label: TEISI4 (Beginn der Datei) und TEISK4 (Rest der Datei)

ROM-Nummer: M

Startadresse: &FCB4 (TEISI4), &FCB7 (TEISK4)

Einsprungbedingungen:
TEISI4:
A = &0D oder &0E, entsprechend ob ins Erste oder Zweite gepufferte Inhaltsverzeichnis der M4 SD-Karte gesichert werden soll, N oder O
REG16_8 = Zeiger auf den Dateinamen (OHNE Pfadangabe!) plus Byte &00
REG_BC1 = / Bits 0-15 / 32 Bit Länge der zu sichernden Datei
REG_DE1 = / Bits 16-31 / in Bytes angegeben (max. 4 GB)

TEISK4:
Die RAM Variablen REG_AF1, REG_BC1 und REG_DE1 müssen seit dem Aufruf von TEISI4 erhalten worden sein.

Aussprungbedingungen: Der Akku liefert Informationen über den Erfolg der Operation:

- A = &FF ==> Die Datei wurde komplett auf SD-Karte geschrieben
- A = &F0 ==> Die Datei wurde nur TEILWEISE geschrieben.
  Weitere Teile sind nun mittels TEISK4 zu speichern.
- A = &00 ==> Es ist kein E-RAM Puffer für die M4 SD-Karte vorhanden
- A = &01 ==> Es wurde kein Inhaltsverzeichnis von SD-Karte gelesen
- A = &02 ==> Die Datei hat eine Länge von 0 KB
- A = &05 ==> Es sind keine KZS im E-RAM frei, E-RAM anders genutzt
- A = &06 ==> Die Datei konnte nicht zum Schreiben angelegt werden

Manipuliert: AF, BC, DE, HL, AF', BC', DE', HL', IX, REG08_1, REG_AF1, REG_BC1, REG_DE1, FN_LR, die XRAM_?? Variablen, das RAM von &B600-&B6FF und von &BE20-&BE31. Außerdem wurden die RAM-Konfiguration und das Ziel-Inhaltsverzeichnis verändert.

Beschreibung: Die OS Funktion ist das Gegenstück zu TEILA(4). Während TEILA(4) eine Datei teilweise läd, dienen TEISI4 / TEISK4 dazu, die geladene Datei wieder abzuspeichern. Ganz oder in Stückchen, das hängt vom vorhandenen E-RAM ab. Freie Blöcke werden als KZS genutzt.
Anfangs wird TEISI4 aufgerufen. TEISI4 öffnet eine Datei zum Schreiben und speichert alle Kurz-Zeit-Speicher (KZS) auf SD-Karte. Der Erfolg der Aktion wird im Akku übergeben.
Kehrt TEISI4 mit dem Byte &FF im Akku zurück, dann wurde die Datei komplett gesichert. Enthält der Akku jedoch den Wert &F0, so wurden zwar alle KZS gespeichert, aber die Datei ist länger als das zur Verfügung stehende E-RAM. Die RAM-Variablen REG_AF1, REG_BC1 und REG_DE1 sind nun zu erhalten. Es muß also erst der Dateirest mit TEILB(4) nachgeladen werden, dann kann dieser Rest mit TEISK4 auf Disk geschrieben werden. Dieser Vorgang kann sich mehrmals wiederholen, wenn die Datei sehr lang oder das E-RAM gering ist.

Bitte Beachten: Nach dem Aufruf von TEISI4 müssen die RAM Variablen REG_AF1, REG_BC1 und REG_DE1 erhalten werden, da sie beim Sichern weiterer Teile einer sehr großen Datei von TEISK4 benötigt werden.

M4: Öffnen einer Datei zum Lesen und Ermitteln der Datei-Länge

Kurzbeschreibung: Eine Datei der SD-Karte der M4 Erweiterung wird zum Lesen geöffnet und ihre Datei-Länge wird ermittelt.

Label: M4_O_S

ROM-Nummer: M

Startadresse: &FCBA

Einsprungbedingungen: HL = Zeiger auf den kompletten Pfad+Dateinamen einer Datei auf der SD-Karte der M4 Erweiterung. Der Pfad+Name wird vom Byte &00 abgeschlossen

Aussprungbedingungen: A = &00: Die Datei wurde geöffnet und ihre Dateilänge wurde ermittelt. In diesem Fall enthält A' den Datei-Deskriptor und Register HL und DE enthalten die 32 Bit Datei-Länge.
A = &FD: Es ist ein Fehler aufgetreten -> HL und DE sind bedeutungslos

Manipuliert: AF, BC, DE, HL, AF', BC', DE', HL' und der RAM Bereich von &BE20 bis &BE4F.

Beschreibung: Die OS Funktion M4_O_S dient dazu, eine Datei von SD-Karte zum Lesen zu öffnen und ihre Datei-Länge zu ermitteln. Dazu wird in HL ein Zeiger auf den gesamten Pfad+Dateinamen übergeben. Dieser Pfad+Name endet mit dem Byte &00.

Nach dem erfolgreichen Aufruf von M4_O_S wird im Akku der Wert &00 übergeben. In A' befindet sich der Datei-Deskriptor, diese Datei-ID ist wichtig, um auf die Datei zuzugreifen bzw. sie zu schließen.
Weiterhin beinhalten die Register HL und DE die präzise Dateilänge in Bytes. Dabei enthält HL die Bits 31-16 und DE die Bits 15-0.
Sollte M4_O_S jedoch mit A = &FD zurückkehren, so ist ein Fehler aufgetreten.

Bitte Beachten: Eine geöffnete Datei sollte nach Verwendung wieder geschlossen werden. Dafür kann die OS Funktion M4_DSL genutzt werden.
Die geöffnete Datei kann nur gelesen werden, nicht beschrieben.

M4: Geöffnete Datei der M4 SD-Karte schließen

Kurzbeschreibung: Eine zuvor geöffnete Datei der SD-Karte der M4 Erweiterung wird wieder geschlossen.

Label: M4_DSL

ROM-Nummer: M

Startadresse: &FCBD

Einsprungbedingungen: A' = Datei-Deskriptor = Datei-ID

Aussprungbedingungen: BC = &FC00 // DE = &4304

Manipuliert: BC, DE und die Datei wurde geschlossen

Beschreibung: Geöffnete Dateien sollten nach ihrer Verwendung auch wieder geschlossen werden. Denn es ist nur möglich, eine bestimmte Anzahl von Dateien gleichzeitig offen zu halten.
Zum Schließen einer Datei wird die OS Funktion M4_DSL verwendet. Bei ihrem Aufruf muss Register A' den Datei-Deskriptor der zuvor geöffneten Datei enthalten.

Bitte Beachten: Es erfolgt keine Rückmeldung über den Erfolg.

M4: Lesen von 16 KB einer geöffneten Datei nach Adresse &4000

Kurzbeschreibung: Ein Block von 16 KB wird nach &4000 gelesen.

Label: M4_R16

ROM-Nummer: M

Startadresse: &FCC0

Einsprungbedingungen: Eine Datei muss bereits zum Lesen geöffnet worden sein. Zum Lesen wird die aktuelle E-RAM-Konfiguration genutzt.
A' = Datei-Deskriptor der geöffneten Datei der M4 SD-Karte

Aussprungbedingungen: A = &00: Daten wurden erfolgreich gelesen
A > &00: Es ist ein Fehler aufgetreten!

Manipuliert: AF, BC, DE, HL, BC', DE', HL', der RAM Bereich von &4000 bis &7FFF und der RAM Bereich von &BE50 bis &BE7F

Beschreibung: Diese OS Funktion dient dazu 16 KB aus einer Datei zu lesen und sie ab &4000 ins RAM zu schreiben. Die Datei der M4 SD-Karte muss dazu bereits zum Lesen geöffnet worden sein. Ihr Datei-Deskriptor wird im Register A' an M4_R16 übergeben. Die gelesenen 16 KB werden in den Block von &4000 bis &7FFF der gerade aktuellen E-RAM-Konfiguration geschrieben. Dadurch ist es möglich 16 KB Daten-Blöcke direkt in 16 KB E-RAM Blöcke zu schreiben.
Nach dem Rücksprung enthält der Akku den Wert &00, falls das Lesen erfolgreich war, andernfalls hat der Akku einen Wert größer &00.

Bitte Beachten: Vor dem Aufruf dieser OS Funktion muss eine M4 SD-Karten-Datei zum Lesen geöffnet worden sein! Dies kann z.B. mittels der OS Funktion 'M4_O_S' getan werden.

M4: Schreibe 16 KB aus dem RAM in eine zum Schreiben geöffnete Datei

Kurzbeschreibung: Schreibe 16 KB Daten in eine geöffnete Datei

Label: M4_W16

ROM-Nummer: M

Startadresse: &FCC3

Einsprungbedingungen: Eine Datei muss bereits zum Schreiben geöffnet worden sein. Zum Schreiben der Daten in die Datei wird die aktuelle E-RAM-Konfiguration genutzt.
A' = Datei-Deskriptor der geöffneten Datei der M4 SD-Karte

Aussprungbedingungen: 16 KB wurden aus der aktuellen RAM Konfiguration in eine geöffnete Datei geschrieben. Der 16 KB Block wurde von &4000 bis &7FFF geschrieben.

Manipuliert: AF, BC, DE, HL und die zu beschreibende Datei wurde um 16 KB länger.

Beschreibung: Diese OS Funktion dient dazu 16 KB Daten aus dem RAM von &4000 bis &7FFF in eine Datei zu schreiben. Die Datei der M4 SD-Karte muss dazu bereits zum Schreiben geöffnet worden sein.
Beim Aufruf von M4_W16 wird der Datei-Deskriptor der zum Schreiben geöffneten Datei in Register A' übergeben. Die zu schreibenden 16 KB werden aus dem RAM Block von &4000 bis &7FFF der gerade aktuellen E-RAM-Konfiguration geholt. Dadurch ist es möglich komplette 16 KB E-RAM Blöcke am Stück in eine geöffnete Datei zu schreiben.

Bitte Beachten: Vor dem Aufruf dieser OS Funktion muss eine M4 SD-Karten-Datei zum Schreiben geöffnet worden sein!

M4: Datei von der M4 SD-Karte löschen

Kurzbeschreibung: Eine Datei wird von der M4 SD-Karte gelöscht.

Label: M4_ERA

ROM-Nummer: M

Startadresse: &FCC6

Einsprungbedingungen: HL = Zeiger auf Pfad+Dateiname der zu löschenden Datei. Der Pfad+Dateiname endet mit dem Byte &00.

Aussprungbedingungen: A = &00: Löschen der Datei war erfolgreich
A = &FF: Ein Fehler ist aufgetreten

Manipuliert: AF, BC, DE, HL und der RAM Bereich von &BE50 bis &BE61

Beschreibung: Diese OS Funktion dient zum Löschen von Dateien oder (leeren) Verzeichnissen. Dazu wird in HL ein Zeiger auf den gesamten Pfad+Namen der Datei / des Verzeichnisses übergeben.
Kehrt M4_ERA mit A = &00 zurück, dann war das Löschen erfolgreich. Bei Auftreten eines Fehlers steht im Akku der Wert &FF.

Achtung: Soll ein Verzeichnis gelöscht werden, so sind zuvor alle Dateien und Unterverzeichnisse zu löschen. Um ein Verzeichnis löschen zu können, muß es leer sein.

Bitte Beachten: Beim Löschen von Verzeichnissen müssen diese leer sein

M4: Erstellen eines Inhaltsverzeichnisses auf SD-Karte

Kurzbeschreibung: Erstelle ein (Unter-)Inhaltsverzeichnis auf der SD-Karte der M4 Erweiterung.

Label: M4_MKD

ROM-Nummer: M

Startadresse: &FCC9

Einsprungbedingungen: HL = Zeiger auf den Namen des zu erzeugenden neuen Inhaltverzeichnisses. Der Name endet mit dem Byte &00.

Aussprungbedingungen: A = &00: Das neue Verzeichnis wurde erzeugt
A = &FF: Ein Fehler ist aufgetreten, z.B. Verzeichnis existiert schon

Manipuliert: AF, BC, DE, HL und der RAM Bereich von &BE50 bis &BE61

Beschreibung: Will man ein neues (Unter-)Inhaltsverzeichnis auf der SD-Karte der M4 Erweiterung erzeugen, so nutzt man diese OS Funktion M4_MKD.
Dabei wird das neue Verzeichnis im aktuell gültigen Verzeichnis erzeugt. Das aktuelle Verzeichnis läßt sich mit den OS Funktionen M4_CDIR oder M4_CD wechseln (siehe oben).

Vor der Aufruf dieser Funktion muß im Register HL ein Zeiger auf den Namen des neuen Verzeichnisses übergeben werden. Der Name muß mit Byte &00 enden. Nach dem Aufruf von M4_MKD enthält der Akku entweder den Wert &00, dann wurde das neue DIR erfolgreich erzeugt. Oder der Akku enthält den Wert &FF, dann ist ein Fehler aufgetreten. So ein Fehler kann z.B. ein unbrauchbarer Name sein, oder das neu anzulegende Verzeichnis existiert bereits.

Bitte Beachten: Wird versucht eine bereits existierendes Verzeichnis nochmals anzulegen, so bricht diese OS Funktion mit dem Fehlerkode &FF im Akku ab.

M4: Umbenennen einer Datei auf der M4 SD-Karte

Kurzbeschreibung: Eine Datei der M4 SD-Karte wird umbenannt.

Label: M4_REN

ROM-Nummer: M

Startadresse: &FCCC

Einsprungbedingungen:
HL = Zeiger auf den alten Pfad+Namen der Datei / Beide Namen enden
DE = Zeiger auf den neuen Pfad+Namen der Datei / mit dem Byte &00!

Aussprungbedingungen: A = &00: Die Datei wurde erfolgreich umbenannt
A = &04: Die Datei existiert nicht (alter Dateiname oder Pfad falsch)
A = &08: Eine Datei mit dem neuen Namen existiert bereits
A = &FF: Es ist ein anderer Fehler aufgetreten

Manipuliert: AF, BC, DE, HL und der RAM Bereich von &BE50 bis &BE61

Beschreibung: Die OS Funktion M4_REN erlaubt es Dateien umzubenennen. Dazu müssen in HL und DE Zeiger auf den alten und neuen Pfad+Namen der Datei übergeben werden.

Kehrt die Funktion mit &00 im Akku zurück, so war das Umbenennen erfolgreich. Andernfalls gibt der Inhalt von A über den Fehler Auskunft.

Bitte Beachten: Beim Auftreten von Fehlern bitte die beiden Pfad+Namen überprüfen.

M4: Ermitteln der Datei-Länge einer Datei auf SD-Karte

Kurzbeschreibung: Die Größe einer Datei der SD-Karte wird ermittelt.

Label: M4_G_SIZE

ROM-Nummer: M

Startadresse: &FCCF

Einsprungbedingungen: HL = Zeiger auf den kompletten Pfad+Dateinamen einer Datei auf der SD-Karte der M4 Erweiterung. Der Pfad+Name wird vom Byte &00 abgeschlossen.

Aussprungbedingungen: A = &00: Die Dateilänge (Bytes) wurde ermittelt.
Sie wird in den Registern HL und DE übergeben.
A = &FD: Es ist ein Fehler aufgetreten -> HL und DE sind bedeutungslos

Manipuliert: AF, BC, DE, HL, AF', BC', DE', HL' und der RAM Bereich von &BE20 bis &BE4F

Beschreibung: Die OS Funktion M4_G_SIZE dient dazu, die Länge einer Datei von SD-Karte im Bytes zu ermitteln.
Dazu wird in HL ein Zeiger auf den gesamten Pfad+Dateinamen übergeben. Dieser Pfad+Name endet mit dem Byte &00.
Nach dem erfolgreichen Aufruf von M4_G_SIZE wird im Akku der Wert &00 zurückgegeben. Und die Register HL und DE enthalten die präzise Dateilänge in Bytes. Dabei enthält HL die Bits 31-16 und DE die Bits 15-0.
Sollte M4_G_SIZE jedoch mit A = &FD zurückkehren, so ist ein Fehler aufgetreten. In diesem Fall sind HL und DE bedeutungslos.

Bitte Beachten: Die überprüfte Datei wurde wieder geschlossen.

M4: Generiere den Pfad+Namen aus aktuellem Pfad und Dateinamen

Kurzbeschreibung: Aus dem aktuellen Pfad der M4 SD-Karte und einem Dateinamen wird ein String generiert, der Pfad und Name + 0 enthält.

Label: ER_PN oder EP_B6 (Pfad+Name beginnt ab &B600)

ROM-Nummer: M

Startadresse: &FCD2 (ER_PN) oder &FCD5 (EP_B6)

Einsprungbedingungen: Ein Verzeichnis der M4 SD-Karte muss eingelesen worden sein. Und das Puffer-E-RAM der SD-Karten Verzeichnisse muss eingeblendet sein.
DE = Zieladresse, ab dem Pfad + Name + Byte &00 generiert werden sollen. Beim Aufruf von EP_B6 muss DE nicht geladen werden, denn DE wird automatisch auf &B600 gesetzt.
HL = Zeiger auf einen Dateinamen in einem der beiden gepufferten SD-Karten Verzeichnisse im SD DIR Puffer E-RAM.
Die RAM Variable FN_LR enthält entweder den Wert &00 oder &20. Dadurch wird bestimmt, ob der Pfad vom gepufferten SD-Karten Verzeichnis des SD-Karten-Mediums 'N' oder 'O' benutzt werden soll.

Aussprungbedingungen: Pfad + Dateiname + Byte &00 stehen ab der angegebenen Zieladresse im RAM. Beim Aufruf von EP_B6 ist das &B600.
DE = Zeiger auf das Byte nach Pfad+Name+&00. A = &00.

Manipuliert: AF, BC, DE, HL und der RAM Bereich des Pfad + Namen + &00

Beschreibung: Bei der Verwendung der M4 SD-Karte können ein oder zwei Verzeichnisse gepuffert werden. Dabei gibt die Variable FN_LR an, ob das erste SD-Karten Laufwerk 'N' bzw. das zweite 'O' benutzt werden soll. Dazu enthält FN_LR (=&BE11) entweder &00 bzw. &20.

Die OS Funktion ER_PN benutzt nun das selektierte SD-Karten Verzeichnis, um aus dem selektierten Pfad dieses Verzeichnisses und einem Dateinamen (HL) den Pfad + Namen + Byte &00 zusammen zustellen. Dabei gibt DE die Zieladresse an. Bei Verwendung von EP_B6 muss DE nicht geladen werden, denn es wird &B600 als Zieladresse angenommen.

Nach Aufruf einer der beiden Funktionen zeigt DE auf das Byte nach dem neue generieren 'Pfad + Dateinamen + Byte &00'. Das Byte &00 dient dabei als Endmarkierung.

Bitte Beachten: Wird ER_PN oder EP_B6 mit HL = &FFFF aufgerufen, so wird lediglich der selektierte Pfad des selektierten SD-Karten-Mediums kopiert.

Finde die erste markierte Datei aller Medien (A bis O)

Kurzbeschreibung: Alle Medien von A bis O (Laufwerke, HD20 Festplatte und beide gepufferten M4 SD-Karten-Verzeichnisse) werden nach der ersten markierten Datei durchsucht. Diese OS Funktion entspricht FMD32 in ROM C.

Label: FMD_AO

ROM-Nummer: M

Startadresse: &FCD8

Einsprungbedingungen: Siehe FMD32 in ROM C

Aussprungbedingungen: Siehe FMD32 in ROM C

Manipuliert: Siehe FMD32 in ROM C

Beschreibung: Siehe FMD32 in ROM C
Die OS Funktion 'FMD_AO' des ROM M ruft die OS Funktion 'FMD32' in ROM C auf. Bitte dort nachlesen.

Bitte Beachten: Siehe FMD32 in ROM C

M4: Finde erste markierte Datei der SD-Karte

Kurzbeschreibung: Alle Datei-Tagging-Bytes der Medien N und O (der M4 SD-Karte) werden nach einer markierten Datei durchsucht.

Label: FMD_NO

ROM-Nummer: M

Startadresse: &FC03

Einsprungbedingungen: Die Datei-Tagging-Bytes der Medien 'N' und 'O' müssen korrekt sein.
Sie befinden sich im E-RAM der M4 Erweiterung.
A < &FF ==> normale Funktion
A = &FF ==> Dateistatus wird NICHT verändert!

Aussprungbedingungen:
HL = &0000 und Zero-Flag = 1 ==> Keine Datei auf M4 SD-Karte markiert!
HL > &0000 und Zero-Flag = 0 ==> HL zeigt auf die 'Adresse' des Namens der ersten markierten Datei im DIRectory E-RAM der M4 SD-Karte.
REG08_1 = A = Medium auf dem sich die Datei befindet, das ist entweder &0D für Medium 'N' oder &0E für Medium 'O'
REG08_2 = E-RAM Nummer des M4 SD-Karten DIRectories &C4-&FF. Dieses E-RAM ist bereits eingeblendet.
REG16_6 bis REG32_1 = 16er Dateiname im Format 'N--:FileNameExt' bzw. 'O--:FileNameExt'. Der Punkt wurde entfernt!
FN_LR wurde auf &00 (für Medium N) oder auf &20 (für Medium O) gesetzt, je nachdem auf welchem Medium die erste markierte Datei gefunden wurde.

Wenn A bei Aufruf von FMD_NO ungleich &FF war:
Der Datei-Status wird von Tagged (markiert) auf Retagged (bearbeitet) gesetzt, eine solche Datei wird durchgestrichen dargestellt.

Manipuliert: Alle Register außer IY. E-RAM Status, Datei-Tagging-Bytes, REG08_1..2, REG16_6..REG32_1

Beschreibung: Dieses Unterprogramm dient dazu, in den Medien 'N' und 'O' der M4 SD-Karte nach einer markierten Datei zu suchen. Beim Einsprung müssen die Systemvariablen in Ordnung sein.

Diese OS Funktion übergibt in HL die Adresse des Datei-Namens der ersten markierten Datei. Dieser Dateiname wird zusätzlich ins 16er Format umgewandelt und ab REG16_6 im RAM abgelegt, um zu 'FMD32' kompatibel zu bleiben.
Hat das Register HL den Wert &0000 und ist das Zero-Flag gesetzt, dann wurde keine markierte Datei gefunden.
Bei HL > &0000 wird im Akku und in REG08_1 das Medium angegeben, auf dem sich die erste markierte Datei befindet. Entweder &0D oder &0E für Medium 'N' oder Medium 'O'. Beide sind gepufferte Inhaltsverzeichnisse der M4 SD-Karte.

Außerdem wird in REG08_2 die physikalische E-RAM-Nummer von &C4 - &FF übergeben, in dem die beiden Inhaltsverzeichnisse der M4 SD-Karte gepuffert werden. Dieses E-RAM ist bereits eingeblendet.

Bitte Beachten: Die Systemvariablen müssen unbedingt korrekt sein, da sonst korrupte Daten übergeben werden könnten.

Achtung: Hat Register A vor Aufruf der OS Funktion den Wert &FF, so wird der Status der gefundenen Datei NICHT verändert. Bei nochmaligem Aufruf von FMD32 würde man dieselbe Datei nochmals finden!

Anstatt den Akku mit &FF zu laden, kann man diese OS Funktion auch über **EMD_NO** aufrufen, an Adresse: **&FC00**.

M4: Finde erstes markiertes Datei-Markierugs-Byte von 'N' oder 'O'

Kurzbeschreibung: Alle Datei-Tagging-Bytes der Medien N oder O (der M4 SD-Karte) werden nach der Markierung 'Markiert' durchsucht. Die Nummer der ersten markierten Datei wird zurückgegeben.

Label: FTBNO

ROM-Nummer: M

Startadresse: &FCDB

Einsprungbedingungen: Die Datei-Tagging-Bytes der Medien 'N' und 'O' müssen korrekt sein. Sie befinden sich im E-RAM der M4 Erweiterung. Dieses E-RAM muß eingeblendet sein.
HL = Zeiger auf den Start der Datei-Markierungs-Bytes
HL = &4400 für Medium 'N' / N und O sind die beiden gepufferten M4
HL = &6400 für Medium 'O' / SD-Karten Verzeichnisse

Aussprungbedingungen:
HL = &0000 ==> Keine Datei auf M4 SD-Karte markiert!
HL > &0000 ==> HL enthält die Nummer der markierten Datei &0001-&03FF und DE zeigt auf das aktuelle markierte Tag-Byte --> DE = &4400-&47FF (Medium N) bzw. = &6400-&67FF (Medium O)

Manipuliert: AF, C, DE und HL

Beschreibung: Diese OS Funktion sucht nach dem ersten markierten Datei-Markierung-Byte / File-Tagging-Byte entweder des Mediums 'N' oder von Medium 'O'. Beide Medien sind gepufferte Inhaltsverzeichnisse der M4 SD-Karte.
Beim Aufruf der Funktion muss HL entweder mit &4400 bzw. für &6400 geladen sein. Diese Adresse entspricht dem Start der Markierungs-Bytes des Mediums N bzw. O.
Nach dem Rücksprung gibt HL Aufschluss über den Erfolg der Aktion. Ist HL mit &0000 geladen, so wurde keine markierte Datei gefunden. Enthält HL jedoch einen Wert zwischen &0001 und &03FF, so entspricht dieser Wert der Nummer der markierten Datei des durchsuchten Mediums.
Weiterhin enthält das Register DE in diesem Fall einen Zeiger auf das aktuelle (markierte) Datei-Markierungs-Bytes.

Um die Datei-Nummer der ersten gefundenen markierten Datei in einen Zeiger auf ihren Namen umzurechnen, kann die OS Funktion FNOFN benutzt werden.

Bitte Beachten: Vor dem Aufruf dieser Funktion muß das Puffer-E-RAM der M4 Erweiterung eingeblendet werden. Beispiel:

```
LD B,&7F:LD A,(M4_ERAM):OUT (C),A ;M4 E-RAM einblenden
```

M4: Berechne Zeiger auf Dateinamen aus Markierungs-Byte-Nummer

Kurzbeschreibung: Die Nummer eines Datei-Markierungs-Bytes wird in die Adresse des Namens dieser Datei der M4 SD-Karte umgerechnet.

Label: FNOFN

ROM-Nummer: M

Startadresse: &FCDE

Einsprungbedingungen: Das E-RAM der M4 SD-Karte muss eingeblendet sein
DE = Nummer der (markierten) Datei &0001-&0400. Achtung: 0 nicht verwenden!
HL = Zeiger auf den 1. Namen des Mediums 'N' bzw. 'O' der M4 SD-Karte
HL = &4800 --> Start 1. Dateiname des Mediums 'N'
HL = &6800 --> Start 1. Dateiname des Mediums 'O'

Aussprungbedingungen: HL = Zeiger auf den gesuchten Namen

Manipuliert: AF, DE und HL

Beschreibung: Diese OS Funktion dient dazu, die Nummer einer Datei in einen Zeiger auf ihren Namen umzurechnen. Dabei handelt es sich um eine Datei auf der SD-Karte der M4 Erweiterung.

Mit der zuvor beschriebenen OS Funktion 'FTBNO' kann die Nummer der ersten markierten Datei berechnet werden. Und mit der OS Funktion 'FNOFN' kann aus der Nummer dieser Datei deren Adresse im Puffer-E-RAM berechnet werden.

Beim Aufruf von 'FNOFN' muß das Puffer-E-RAM der Inhaltsverzeichnisse der SD-Karte der M4 Erweiterung bereits eingeblendet sein. Weiterhin enthält Register HL einen Zeiger auf das erste Zeichen des ersten Namens des genutzten Mediums. Für Medium 'N' ist HL = &4800 zu setzen und für Medium 'O' ist HL = &6800 zu setzen. Die Nummer der gesuchten Datei wird in Register DE übergeben.

Nach dem Rücksprung dieser OS Funktion zeigt HL auf das erste Zeichen des gesuchten Dateinamens im Puffer E-RAM der M4 Erweiterung.

Bitte Beachten: Vor dem Aufruf dieser Funktion muß das Puffer-E-RAM der M4 Erweiterung eingeblendet werden, wenn es das noch nicht ist.

Beispiel:

```
LD B,&7F:LD A,(M4_ERAM):OUT (C),A ;M4 E-RAM einblenden
```

Achtung: DE darf beim Aufruf der Funktion nicht den Wert &0000 enthalten! Denn 0 würde als &FFFF verwendet werden!

M4: Konvertiere SD-Karten Dateiname in OS Format

Kurzbeschreibung: Ein Dateiname der M4 SD-Karte wird in das FutureOS Format für Dateinamen konvertiert.

Label: FNOCN

ROM-Nummer: M

Startadresse: &FCE1

Einsprungbedingungen: A = 'N' oder A = 'O' für Medium N oder O
HL = Zeiger auf den M4 SD-Karten Dateinamen
Das Puffer-E-RAM für die M4 SD-Karte ist vorhanden und eingeblendet.

Aussprungbedingungen: Von REG16_6 bis REG32_1 steht der neu erzeugte Dateiname im üblichen FutureOS Format:
Für Medium N: '**`N:--FileNameExt`**'
Für Medium O: '**`O:--FileNameExt`**'
Hierbei ist keine User-Nummer vorhanden, es ist auch kein Punkt vor der Extension vorhanden. Das Format entspricht dem üblichen Stil, wie Dateinamen unter FutureOS angezeigt werden.

Manipuliert: AF, BC, DE, HL und REG16_6 bis REG32_1

Beschreibung: Die SD-Karte der M4 Erweiterung unterscheidet sich in zwei Punkten wesentlich von den CPC / FutureOS üblichen Dateinamen:
- Es gibt keine User-Nummern
- Namen können vor dem Punkt mehr als acht Zeichen enthalten

Diese OS Funktion 'FNOCN' wandelt einen Dateinamen der M4 SD-Karte in das übliche Format '**`MUU:FileNameExt`**' um:

M	=	Medium (hier N oder O)
UU	=	User Nummer (hier '--')
FileName	=	Acht Zeichen des Dateinamens
Ext	=	Extension

Der bei SD-Karten-Dateinamen vorhandene Punkt entfällt hierbei.
Es werden maximal acht Zeichen des Dateinamens benutzt, überzählige Zeichen werden nicht kopiert. Ist der Name kürzer, so wird mit Leerzeichen aufgefüllt.

Diese OS Funktion wird von der OS Funktion 'FMD_NO' benutzt.

Bitte Beachten: Ist der Dateinamen der SD-Karte zu lang, so werden nur die ersten acht Buchstaben vor dem Punkt kopiert!

M4: Entferne alle Leerzeichen aus einem Dateinamen / String

Kurzbeschreibung: Aus einem Dateinamen der M4 SD-Karte (oder einem String) werden alle Leerzeichen entfernt.

Label: LZEN

ROM-Nummer: M

Startadresse: &FCE4

Einsprungbedingungen: HL = Zeiger auf den Anfang des Dateinamens

Aussprungbedingungen: Alle Leerzeichen wurden entfernt

Manipuliert: AF, BC, DE, HL und der Dateiname / String

Beschreibung: Diese OS Funktion entfernt alle Leerzeichen (&20) aus einem Dateinamen der M4 SD-Karte bzw. aus irgendeinem String. Dabei muß der Name / String mit dem Byte &00 enden.
Bei der Verwendung der SD-Karte der M4 Erweiterung dürfen Dateinamen keine Leerzeichen (= Zeichen &20 = 32) enthalten. Diese OS Funktion eliminiert die Leerzeichen und verkürzt dadurch die Länge. Das Ende wird weiterhin durch das Byte &00 markiert. Die Anfangsadresse des Dateinamens bzw. Strings wird nicht verändert.

Bitte Beachten: Das Ende des Dateinamens bzw. Strings wird durch ein &00-Byte angezeigt. Nach dem ersten Byte &00 endet die Bearbeitung.

M4: Selektiere folgenden E-RAM Block, setze DE = &4000

Kurzbeschreibung: Innerhalb der maximal möglichen 4 MB Erweiterungs-RAM wird der nachfolgende Block selektiert und eingeblendet. Und das Register DE wird auf &4000 gesetzt.

Label: EX_ERAM

ROM-Nummer: M

Startadresse: &FCE7

Einsprungbedingungen: OS Variable AKT_RAM enthält aktuelles E-RAM

Aussprungbedingungen: DE = &4000
AKT_RAM enthält den nachfolgenden, eingeblendeten 16 KB E-RAM Block

Manipuliert: F, DE und AKT_RAM

Beschreibung: An den CPC lassen sich maximal 4 MB Erweiterungs-RAM (E-RAM) anschließen. Unter FutureOS wird dieses E-RAM in Blöcken von 16 KB angesprochen, diese werden zwischen &4000 und &7FFF eingeblendet.

Die E-RAM Auswahl des aktuellen E-RAM Blocks solle in der OS Variable AKT_RAM (&7FC4-&7FFF, &7EC4-&7EFF, ... &78C4-&78FF) gespeichert sein.

Die OS Funktion EX_ERAM liest den aktuellen Wert aus AKT_RAM und schaltet auf den nächsten 16 KB Block der Erweiterungs-RAMs. Dabei wird sowohl das nachfolgende E-RAM eingeblendet, als auch AKT_RAM auf den neuen Stand gebracht. Weiterhin wird das Register DE auf &4000 gesetzt, das ist der Anfang des neuen E-RAM Blocks.

Bitte Beachten: Es wird nicht geprüft, ob der neu selektierte E-RAM Block physikalisch vorhanden ist.

M4: Selektiere folgenden E-RAM Block, setze HL = &4000 / BC = &FE00

Kurzbeschreibung: Innerhalb der maximal möglichen 4 MB Erweiterungs-RAM wird der nachfolgende Block selektiert und eingeblendet. Und das Register HL wird auf &4000 gesetzt. Und BC = &FE00, der M4 Datenport.

Label: NX_ERAM

ROM-Nummer: M

Startadresse: &FCEA

Einsprungbedingungen: OS Variable AKT_RAM enthält aktuelles E-RAM

Aussprungbedingungen: HL = &4000 // BC = &FE00
AKT_RAM enthält den nachfolgenden, eingeblendeten 16 KB E-RAM Block

Manipuliert: F, BC, HL und AKT_RAM

Beschreibung: An den CPC lassen sich maximal 4 MB Erweiterungs-RAM (E-RAM) anschließen. Unter FutureOS wird dieses E-RAM in Blöcken von 16 KB angesprochen, diese werden zwischen &4000 und &7FFF eingeblendet.

Die E-RAM Auswahl des aktuellen E-RAM Blocks solle in der OS Variable AKT_RAM (&7FC4-&7FFF, &7EC4-&7EFF, ... &78C4-&78FF) gespeichert sein.

Die OS Funktion NX_ERAM liest den aktuellen Wert aus AKT_RAM und schaltet auf den nächsten 16 KB Block der Erweiterungs-RAMs. Dabei wird sowohl das nachfolgende E-RAM eingeblendet, als auch AKT_RAM auf den neuen Stand gebracht. Weiterhin wird das Register HL auf &4000 gesetzt, das ist der Anfang des neuen E-RAM Blocks. Und Register BC wird auf &FE00 gesetzt, das ist der Datenport der M4 Erweiterung.

Bitte Beachten: Es wird nicht geprüft, ob der neu selektierte E-RAM Block physikalisch vorhanden ist.

M4: Bestimme die Anzahl der Einträge in einem M4 Inhaltsverzeichnis

Kurzbeschreibung: Die Anzahl der Unterinhaltsverzeichnisse und Dateien im aktuellen M4 Verzeichnis wird ermittelt.

Label: ZDN

ROM-Nummer: M

Startadresse: &FCED

Einsprungbedingungen: Das Puffer-E-RAM für die M4 SD-Karte existiert und ist eingeblendet. Ein Verzeichnis wurde gelesen.
HL = Zeiger auf den 1. Names des zu untersuchenden Verzeichnisses. Üblicherweise &4800 (Medium 'N') oder &6800 (Medium 'O').

Aussprungbedingungen:
DE = Anzahl der Einträge (SubDIRs + Dateinamen) des untersuchten M4 Verzeichnisses.

Manipuliert: AF, BC, DE und HL

Beschreibung: Diese Funktion ermittelt die Anzahl aller Einträge des aktuellen Inhaltsverzeichnisses von M4 Medium 'N' oder 'O'. Dabei werden sowohl Unter-Inhaltsverzeichnisse als auch Dateien gezählt.

Beim Aufruf der Funktion mu: das M4 E-RAM (= Puffer der Verzeichnisse) eingeblendet worden sein und Register HL muss auf den Start des zu untersuchenden Verzeichnisses zeigen. Normalerweise mu: also HL mit &4800 für Medium 'N' oder mit &6800 für Medium 'O' geladen werden.

Nach den Rücksprung der Funktion enthält das Register DE die Anzahl der gefundenen Einträge (Verzeichnisse + Dateinamen).

Bitte Beachten: Das M4 DIR-Puffer E-RAM muss eingeblendet sein.
Es werden sowohl Datei-Namen als auch Unter-Inhaltsverzeichnis-Namen gezählt.

M4: Konvertiere einen M4 Dateinamen ins 9P3 Format

Kurzbeschreibung: Ein beliebiger Dateiname eines M4 Verzeichnisses wir ins 9P3 Format konvertiert, also 9 Zeichen Name + Punkt + 3 Zeichen Erweiterung.

Label: KM4OS

ROM-Nummer: M

Startadresse: &FCF0

Einsprungbedingungen: Sollte sich der Quellname im E-RAM befinden, so muß dieses eingeblendet sein.
HL = Quellname = Zeiger auf das 1. Zeichen des M4 SD-Karten Dateinames
DE = Zielname = Zeiger auf 13 Bytes freien Speicher (innerhalb &nn??)

Aussprungbedingungen:
HL = Zeiger auf das 1. Zeichen des nachfolgenden SD-Karten Dateinamens

Manipuliert: AF, BC, DE und HL

Beschreibung: Um die Dateinamen von der SD-Karte der M4 Erweiterung im FutureOS gordnet darstellen zu können, müssen sie in ein definieres Format gebracht werden. Diese Funktion übernimmt die Funktion 'KM4OS'.

Dabei wird der Name eines M4 SD-Karten-Eintrags in das 9P3 Format umgewandelt. Ein Eintrag der SD-Karte kann bis zu einigen Dutzend Zeichen lang sein. Das 9P3 Format besteht aus: 9 Zeichen Name, dann folgt der Punkt, und schlie:lich 3 Zeichen Dateierweiterung. Ein Beispiel ist: 'DateiName.Ext'.

Allerdings werden die Namen von Unter-Inhaltsverzeichnissen nicht ins 9P3 Format ungewandelt, es werden lediglich die ersten 13 Zeichen kopiert.

Beim Aufruf von 'KM4OS' muss HL einen Zeiger auf den umzuwandelnden Eintrag (im M4 SD-Karten Format) enthalten. Und DE zeigt auf einen Speicherbereich von 13 Bytes, in die der Eintrag im 9P3 Format geschrieben wird. Dabei müssen diese 13 Bytes innerhalb einer Code-Seite liegen (&nn??).

Nach dem Rücksprung der Funktion zeigt HL auf den Beginn des nächsten Namens. Damit ist der nachfolgende Name im Quell-Inhaltsverzeichnis gemeint.

Bitte Beachten: Unterinhaltsverzeichnisse werden nicht ins 9P3 Format überführt, sondern es werden nur die ersten 13 Zeichen kopiert.