Datei: API-D-DE.DOK (29 Seiten) Stand: 13. Februar, Jahr des CPC 2019

Datei zur Dokumentation aller Future Operating System Systemfunktionen der Version .8 des OS. Sämtliche für den Anwender oder Programmierer wichtige OS Funktionen in FutureOS ROM D (=Desktop) werden erklärt und lokalisiert.

Alle OS Funktionen werden in folgender Form beschrieben:

1. Kurzbeschreibung: Die OS Funktion wird in einem Satz kurz erklärt.

2. Label: Mit diesem Label wird die OS Funktion im Source Code bezeichnet. In der mitgelieferten Label-Bibliothek (#EQU-API.DEU) mit den jehweils aktuellen ROM Adressen findet ebenfalls dieses Label Verwendung. Aus Gründen der Kompatibilität sollte man in eigenen Programmen alle OS- / System-Funktionen (und System-Variablen) stets mit diesen Labels ansprechen.

3. ROM-Nummer: Hier ist die logische ROM - Nummer des FutureOS ROMs angegeben, in dem die entsprechende OS Funktion zu finden ist.
Manche OS Funktionen kommen, wegen ihrer Kürze, in mehreren ROMs vor, dann sind hier auch mehrere ROM Nummern angegeben.
Da die ROM Nummer logisch zu verstehen ist, sollte man sie nicht mit dem physikalischen ROM Select gleichsetzen. Wie man die physikalische ROM Nummer eines der FutureOS ROMs ermittelt wird im Handbuch erklärt.

4. Startadresse: Gibt die Einsprung-Adresse der OS Funktion an. Falls diese in mehreren ROMs vorkommt, wird hier auch eine entsprechende Anzahl von Adressen angegeben.

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 Pheripheriebausteine wiedergegeben.

8. Beschreibung: Es folgt eine vollständige Erklärung der Anwendung und Wirkungsweise der beschriebenen OS Funktion.

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

ACHTUNG: Einige OS Funktionen im FutureOS Desktop ROM D sind für den Programmierer nur indirekt nutzbar, da sie für die Benutzeroberfläche entwickelt worden sind.

Bei OS Funktionen ohne Rücksprung empfiehlt es sich z.B. die Variablen des OK-Icons zu nutzen um so in die Applikation zurück zu gelangen.

Die aktuelle Version dieser Datei finden Sie auch im Internet unter:

FutureOS Homepage: http://www.FutureOS.de

DARSTELLUNG EINES ICONS 6*3 MODE 2 ZEICHEN BEI 64*32 ZEICHEN

Kurzbeschreibung: Diese OS Funktion stellt ein 6 * 3 MODE 2 Zeichen großes Icon dar.

Label: ICON6ON

ROM-Nummer: D

Startadresse: &C017

Einsprungbedingungen:
DE = Ziel-Adresse im Bildschirmspeicher, normal &C000 ... &C7FA.
HL = Quell-Adresse des Icon Datenblocks.

Aussprungbedingungen: Icon wurde dargestellt.

Manipuliert: AF, BC, DE, HL und 144 Bytes Video-RAM.

Beschreibung: Diese OS Funktion dient dazu die quadratischen Icons des FutureOS auszugeben. So ein Icon ist 6 Mode 2 Zeichen breit und 3 Zeilen hoch. Das Icon hat also 48 * 24 Pixel. Außerdem muß der Bildschirm auf 64 Zeichen bei 32 Zeilen geschalten sein.

Beim Aufruf dieser Funktion enthält DE die Zieladresse innerhalb des Videorams, sie sollte normalerweise zwischen &C000 und &C7FA liegen.

HL enthält die Quell-Adresse der Icondaten. Dabei ist zu beachten, daß bei jedem Iocn die oberste und die unterste Zeile als ganzer Strich dargestellt wird. Dies spart pro Icon 12 Bytes Daten, außerdem haben die Icons alle einen Rahmen, es wird also auch etwas Rechenzeit gespaart.

Die Icon-Daten sind folgendermaßen aufgebaut: Zuerst kommen von links nach rechts die sechs Bytes der zweiten Zeile, dann die sechs Bytes der dritten, dann der vierten Zeile usw. bis zur 23. Zeile.

Bitte Beachten: Die Iconausgabe funktioniert nur im 64 * 32 Modus.
Die oberste und die unterste Zeile (1,24) eines Icons wird immer als Strich dargestellt.
Die Zeil-Adr. ist sehr frei wählbar.

DARSTELLUNG EINES ICONS 3*3 MODE 2 ZEICHEN BEI 64*32 ZEICHEN

Kurzbeschreibung: Darstellung: 3 * 3 Mode 2 Zeichen großes Icon.

Label: ICON3ON

ROM-Nummer: D

Startadresse: &C05B

Einsprungbedingungen:
DE = Ziel-Adresse im Bildschirmspeicher &C000 ... &C7FA.
HL = Quell-Adresse des Icon Datenblocks.

Aussprungbedingungen: Icon wurde dargestellt.

Manipuliert: AF, BC, DE, HL und 72 Bytes Video-RAM.

Beschreibung: Diese OS Funktion entspricht im weiteren genau der zuvor beschriebenen. Der Unterschied zu ICON6ON liegt darin, daß diese Funktion (ICON3ON) Icons von halber Breite Darstellt. Die Grafikdaten enthalten also lediglich je 3 Bytes pro Rasterzeile. Das FutureOS benutzt ICON3ON z.B. um die Ziffern der Uhrzeit und des Datums im Desktop darzustellen.

Bitte Beachten: Die Iconausgabe funktioniert nur im 64 * 32 Modus.
Die oberste und die unterste Zeile (1,24) eines Icons wird immer als Strich dargestellt.
Die Zeil-Adr. ist sehr frei wählbar.

UHRZEIT UND DATUM IM DESKTOP DARSTELLEN

Kurzbeschreibung: Diese OS Funktion stellt die aktuelle Uhrzeit und das Datum auf dem Bildschirm dar, oder nur einen Teil davon.

Label: DS_ZD (DS_YD, DS_XD, DS_WD, DS_VD, DS_UD)

ROM-Nummer: D

Startadressen:

&E35C (DS_ZD)	///	&E39E (DS_YD)	///	&E3E0 (DS_XD)
&E422 (DS_WD)	///	&E464 (DS_VD)	///	&E4A6 (DS_UD)

Einsprungbedingungen: Folgende RAM - Variablen müssen die korrekten Daten enthalten:

UHR_SEK = Sekunde . &00-&59
UHR_MIN = Minute ... &00-&59
UHR_STU = Stunde ... &00-&23
UHR_TAG = Tag ..... &00-&31
UHR_MON = Monat ... &01-&12
UHR_JAR = Jahr ..... &00-&99

Aussprungbedingungen: Uhrzeit, Datum in 24*24 Pixel Icons dargestellt.

Manipuliert: AF, BC, DE, HL, IX und das Video-RAM

Beschreibung: Wenn das Desktop das aktuelle Datum und die Uhrzeit darstellen will, dann benutzt es diese OS Funktion.
Zeit und Datum (DS_ZD) werden an definierten Bildschirmadressen dargestellt. Dazu werden 24 * 24 Pixel große Icons verwendet.
Benutzt man ein anderes Label als DS_ZD, dann wird nur ein Teil der Information Dargestellt. Dies ist dann sinnvoll wenn sich nicht alle Datums- und Zeit-Werte geändert haben. Im einzelnen stellen die Labels folgende Daten dar:

DS_ZD stellt Sekunde, Minute, Stunde, Tag, Monat und Jahr dar.
DS_YD stellt Sekunde, Minute, Stunde, Tag und Monat dar.
DS_XD stellt Sekunde, Minute, Stunde und Tag dar.
DS_WD stellt Sekunde, Minute und Stunde dar.
DS_VD stellt Sekunde und Minute dar.
DS_UD stellt Sekunde dar.

Der Programmierer kann diese OS Funktion kaum sinnvoll einsetzen, da die Zieladressen der Icons/Ziffern festgelegt sind. Es sei denn daß das Desktop bei abgeschaltenen Interrupts teilaktiv bleiben soll.

Bitte Beachten: Die Zieladressen im Video-RAM sind vordefiniert.
ACHTUNG: Die Adressen diese OS Funktionen haben sich in der FutureOS Version 0.8 geändert!!!

KLICK - EINSPRUNG INS DESKTOP BEI INTAKTEM ICONSATZ

Kurzbeschreibung: Rücksprung ins Desktop, obere Bildhälfte intakt.

Label: KLICK

ROM-Nummer: D

Startadresse: &FE9A

Einsprungbedingungen: Bildschirm im 64 * 32 Zeichen Modus.
Icons in oberer Bildschirmhälfte unverändert.

Aussprungbedingungen: Es erfolgt KEIN Rücksprung

Manipuliert: - nicht relevant -

Beschreibung: Will ein Programm die Kontrolle an die Benutzeroberfläche zurückgeben, dann kann unter anderem KLICK als Einsprung gewählt werden.
Befindet sich der Bildschirm im 64 Zeichen auf 32 Zeilen Modus und sind alle Icons in der oberen Bildhälfte noch intakt, dann sollte man KLICK wählen, ansonsten lieber TUR_D oder TUR_E.

Folgenden Aktionen werden von KLICK ausgeführt:
- Laufwerksmotoren werden der Variable MO_ST angeglichen
- Systemvariablen für Pfeilverwaltung werden restauriert
- Standart 64K werden eingeschalten
- zuletzt wird die Kontrolle an das OS übergeben

Bitte Beachten: Es erfolgt kein Rücksprung, da es sich bei KLICK nicht um eine OS Funktion sondern um einen Einsprung in das OS handelt.

INITIALISIERUNG UND HAUPT-EINSPRUNG INS DESKTOP

Kurzbeschreibung: Haupteinsprünge in die Benutzeroberfläche.

Label: TUR_D (zus. LWs zurücksetzen) // TUR_E (normal)

ROM-Nummer: D

Startadresse: &FEA0 (TUR_D) // &FE9D (TUR_E)

Einsprungbedingungen: -

Aussprungbedingungen: KEIN Rücksprung

Manipuliert: s.u.

Beschreibung: Nach z.B. Beendigung eines Programms ist die Rückkehr ins Betriebssystem nötig. Wohin auch sonnst? ;-)
Diese beiden Einsprünge dienen dazu direkt in die Benutzeroberfläche zurückzukehren. Dabei erfüllen beide folgende Funktionen:
- Bildschirm auf Mode 2, 64 Zeichen auf 32 Zeilen.
- Bildschirm löschen.
- Laufwerks-, Grafik- und Zeit/Datums- Icons einblenden.
- Sprung nach KLICK (siehe dort für weitere Informationen).

Beim Einsprung in TUR_D werden zusätzlich alle eingelesenen Inhaltsverzeichnisse verworfen (RAM-Var. DIRIN auf &FF), außerdem werden die Laufwerksvariablen TURBO_A..M initialisiert (INI_MED s.o.).

Bitte Beachten: Es erfolgt KEIN Rücksprung. Bei Einsprung in TUR_D werden sämtliche eingelesene Inhaltsverzeichnisse verworfen.

(ENT-)MARKIEREN EINER DATEI FüR SPÄTERE BEARBEITUNG

Kurzbeschreibung: Die Datei, die unter der Pfeilspitze liegt, wird zur Bearbeitung markiert.

Label: TAG_DIR

ROM-Nummer: D

Startadresse: &FEB5

Einsprungbedingungen: Systemvariablen müssen korrekte Werte enthalten

Aussprungbedingungen: Es erfolgt KEIN Rücksprung, diese OS Funktion springt nach ihrer Abarbeitung nach KLICK (ins Desktop).

Manipuliert: Datei Tagging Byte der Datei unter dem Maus-Pfeil

Beschreibung: Hat man unter FutureOS das Inhaltsverzeichniss eines LWs eingelesen, so hat der Anwender erst die Dateien zu markieren, mit denen er später arbeiten will.
TAG_DIR übernimmt dies. Entweder wird eine unmarkierte Datei markiert, oder es wird eine bereits markierte Datei wieder entmarkiert.
Es wird diejenige Datei (ent-)markiert, die unter dem Mauspfeil liegt, und deren Inhaltsverzeichniss gerade angezeigt wird.

Bitte Beachten: Es erfolgt KEIN Rücksprung !!!
Diese OS Funktion dürfte für den Programmierer nur von geringem Interesse sein.

ANZEIGEN VON BIS ZU 64 DATEINAMEN EINES INHALTSVERZEICHISSES

Kurzbeschreibung: Anzeigen der, blättern in den eingelesenen Inhaltsverzeichnisseiten. Statuszeile wird angezeigt.

Label: SCRI zeigt erste Seite der aktuellen DIRs an. SC_AU nächste Seite aufwärts bzw. SC_AB nächste Seite abwärts in DIRs blättern.

ROM-Nummer: D

Startadresse: &FEAC (SCRI) // &FEB2 (SC_AU) // &FEAF (SC_AB)

Einsprungbedingungen: RAM Variablen korrekt:

DIRIN	=	&FF => kein DIR, Abbruch
TDRAM	=	RAM Block 16er DIR des aktuellen LWs.
TDHST	=	HighByte d. DIRs in diesem Block &40-&78
TDANZ	=	Anz.1K Seiten des akt.LWs max. 8 bei HD
TDAKT	=	akt.1K Seite des akt.LWs 0,1..3..7
TDLWK	=	aktuelles Laufwerk, dessen DIR angezeigt

Aussprungbedingungen: Eine 1 KB Seite, max. 64 DIR Einträge angezeigt

Manipuliert: Z80 Register, RAM-Banking, Bildschirm etc.

Beschreibung: Wer im Desktop in den Inhaltsverzeichnissen der verschiedenen LWs blättert benutzt diese OS Funktionen.
SCRI gibt die ersten 64 Dateinamen/Einträge auf dem Bildschirm aus.
Hat das DIR werniger als 64 Dateien, dann werden eben nur diese angezeigt.
Mit SC_AB kann man die nächste Seite anzeigen. Will man dagegen zurückblättern, dann benutzt man SC_AU, welches die vorige Seite anzeigt.
Angezeigt werden jehweils max. 64 Dateien, in der unteren Bildschirmhälfte, und zwar 16 Zeilen mit jeh 4 Dateinamen.
Dazu werden die sogenannten 16er DIRs verwendet, es handelt sich dabei um eine für die Ausgabe manipulierte Version des eigentlichen DIRs eine Laufwerks. Alle 16er DIRs sind RAM gepuffert.
Alle drei OS Funktionen zeigen zusätzlich noch eine Statuszeile an, diese enthält Informationen über Format, freien Platz in KB (Disk) und die verwendete Seite (Disk).

Bitte Beachten: Bitte prüfen Sie vor Aufruf einer dieser OS Funktionen nach, ob überhaupt ein DIR (Inhaltsverzeichnis) eingelesen ist. Die Variable DIRIN gibt darüber Aufschluß (siehe Datei #OS-VAR.DEU).
Die Ausgabeadresse der DIR-Seite ist festgelegt, und zwar von der 15 bis zur 31 Zeile. Unterhalb der DIR-Seite bleibt nur die Statuszeile frei.

DISKETTENLAUFWERK BEI FEHLFUNKTION DEAKTIVIEREN

Kurzbeschreibung: Diese OS Funktion vermerkt ein Laufwerk als nicht vorhanden, und deaktiviert es im Desktop.

Label: LADNT

ROM-Nummer: D

Startadresse: &FEA9

Einsprungbedingungen: A = zu deaktivierendes Laufwerk von 0..7 = A..H

Aussprungbedingungen: Es erfolgt kein Rücksprung, ab nach KLICK

Manipuliert: AF, BC, DE, HL, DIRIN=&FF und TURBO_?=&02

Beschreibung: Hat man bei einem Laufwerk einen Fehler festgestellt, dann empfiehlt es sich dies dem Anwender mitzuteilen, dazu kann diese OS Funktion verwendet werden.
Das zu deaktivierende LW wird in A übergeben, daraufhin wird sein LW tagging Byte TURBO_? mit &02 geladen, d.h. LW ist nicht vorhanden/inaktiv und nicht markiert.
Daraufhin wird das Inaktiv-Icon des entsprechenden LWs an korrekter Position auf dem Bildschirm ausgegeben. Außerdem wird die RAM Variable DIRIN mit &FF geladen, und damit sämtliche geladenen DIRs verworfen.
Diese OS Funktion kehrt nicht zurück, es erfolgt ein Sprung nach KLICK (also zum Desktop).
Ist ein Rücksprung durch den Anwender gewünscht, dann kann man zuvor das OK Icon vorbereiten.
Diese OS Funktion eignet sich vor allem für Programme, die aktiv mit dem Desktop zusammenarbeiten.

Bitte Beachten: KEIN Rücksprung !!! OS Funktion springt nach KLICK !!!
Es wird das "inaktiv" Icon des entsprechenden LWs auf dem Bildschrim dargestellt.

DIR-RAM'S IN XRAM_?? MARKIEREN

Kurzbeschreibung: Alle Exp. RAMs die DIRs enthalten, werden in den XRAM_?? Variablen auch als solche markiert.

Label: MADRA

ROM-Nummer: D

Startadresse: &E2FC

Einsprungbedingungen: TURBO_A..TURBO_M und XRAM_C4..XRAM_FF korrekt.

Aussprungbedingungen: DE = &0804
XRAM_C4..XRAM_FF u.U als DIR-Puffer markiert.

Manipuliert: AF, BC, DE, HL, BC', D', HL' und XRAM_C4..XRAM_FF

Beschreibung: Wenn man ein Inhaltsverzeichniss von Hand einließt und im Erweiterungs-RAM puffert, so sollte man dieses RAM auch als DIRectory-Puffer markieren. Dies geschieht in den XRAM Variablen, die Aufschluß über die Verwendung eines jeden Exp.-RAM-Blocks geben.

Bitte Beachten: Es werden die DIR-Exp.-RAMs aller markierten Laufwerke ermittelt, die Variablen TURBO_A..TURBO_M sollten also einen korrekten Inhalt haben.

ACHTUNG: Die Einsprungadresse ist seit FutureOS Version 0.8 geändert!

LAUFWERKS UND ANDERE ICONS EINBLENDEN

Kurzbeschreibung: Es werden die Laufwerks Icons und noch einige andere eingeblendet.

Label: AMON

ROM-Nummer: D

Startadresse: &E4E8

Einsprungbedingungen: 64 * 32 Bildschirmmodus aktiv, Mode 2 aktiv.
RAM-Variablen TURBO_A...TURBO_H korrekt.

Aussprungbedingungen: Icons wurden dargestellt.

Manipuliert: AF, BC, DE, HL und Video-RAM

Beschreibung: Diese OS Funktion dient dazu alle Laufwerks-Icons A...M und einige andere Icons an korrekter Position darzustellen. Dies ist u.U. wichtig, wenn man den Bildschirm nach Verwendung wieder restaurieren will.

Abgesehen von den LW-Icons werden noch folgende Icons eingeblendet: OK, REN, DRUCKEN, RETAG, UNTAG, WECKER, END, GFX2TXT und INFO.

Bitte Beachten: Der Bildschirm sollte auf MODE 2, mit 64 Zeichen auf 32 Zeilen gesetzt sein.

ACHTUNG: Die Einsprungadresse ist seit FutureOS Version 0.8 geändert!

RAM-VARIABLEN DER LAUFWERKE DURCH KONFIG BYTES INITIALISIEREN

Kurzbeschreibung: TURBO_A..M werden den RAM-Bytes der Koniguration entsprechend gesetzt.

Label: INI_MED

ROM-Nummer: D

Startadresse: &E6AD

Einsprungbedingungen: RAM-Konfig Bytes korrekt.

Aussprungbedingungen: TURBO_A..M initialisiert (&00 oder &02).

Manipuliert: AF, B, DE, HL und TURBO_A..M

Beschreibung: Diese OS Funktion dient dazu die Variablen TURBO_A..M zu initialisieren. Hat man z.B. die Konfig-Bytes im RAM (!) geändert, und will die neue Konfiguration auch bei den Speicher-Medien anwenden, so springt man diese OS Funktion an.

In TURBO_A..M ist vorallem Bit 1 relevant, es gibt an, ob ein Laufwerk vorhanden ist oder nicht. Alle anderen Bits werden in jedem Fall auf Null gesetzt. Jedes Laufwerk ist also unmarkiert und Seite 0 ist aktiv.

TURBO_A..M enthalten nun entweder das Byte &00 (LW vorhanden) oder &02 (LW ist NICHT vorhanden).

Bitte Beachten: Es werden die Konfig-Bytes des RAMs gelesen, nicht die im ROM. Hat man die RAM Konfig der LWs geändert, dann wird diese nun aktiv.

ACHTUNG: Die Einsprungadresse ist seit FutureOS Version 0.8 geändert!

EINBLENDEN DER TEXT ICONS

Kurzbeschreibung: Alle Text-Icons werden eingeblendet.

Label: TXT_ICO - DIESE FUNKTION IST AB FutureOS SYSTEM .8 HISTORISCH!

ROM-Nummer: D

Startadresse: &FEA6

Einsprungbedingungen: 64 * 32 Bildschirmmodus, Mode 2 aktiv.

Aussprungbedingungen: KEIN Rücksprung, OS Funktion springt nach KLICK

Manipuliert: AF, BC, DE, HL und Video-RAM.

Beschreibung: Unter FutureOS kann der Anwender zwischen Text und Grafik Icons wählen.

Diese OS Funktion stellt alle Text-Icons dar, das sind im einzelnen folgende: DIR, TYPE, LOAD, SAVE, ERA, COPY, MONITOR und RUN.

Bitte Beachten: Man sollte REG08_7 auf &01 setzen, um dem OS zu sagen, dass die Text-Icons aktiv sind.

DIESE OS FUNKTION IST AB FutureOS SYSTEM .8 HISTORISCH!

EINBLENDEN DER GRAFIK ICONS

Kurzbeschreibung: Alle Grafik-Icons werden eingeblendet.

Label: GRA_ICO - DIESE FUNKTION IST AB FutureOS SYSTEM .8 HISTORISCH!

ROM-Nummer: D

Startadresse: &FEA3

Einsprungbedingungen: 64 * 32 Bildschirmmodus, Mode 2 aktiv.

Aussprungbedingungen: Kein Rücksprung, OS Funktion springt nach KLICK

Manipuliert: AF, BC, DE, HL, und REG08_7 wird mit &00 geladen.

Beschreibung: Unter FutureOS kann der Anwender zwischen Text und Grafik Icons wählen.

Diese OS Funktion stellt alle Grafik-Icons dar, das sind im einzelnen folgende: DIR, TYPE, LOAD, SAVE, ERA, COPY, MONITOR und RUN.

Bitte Beachten: REG08_7 wird auf &00 gesetzt, um dem OS zu sagen, dass die Grafik-Icons aktiv sind.

DIESE OS FUNKTION IST AB FutureOS SYSTEM .8 HISTORISCH!

DOBBERTIN-ZEIT IN EINZELNE BYTES EXPANDIEREN

Kurzbeschreibung: Drei Dobbertin-kompatible Uhrzeit Bytes werden zu sechs Bytes expandiert. Jedes dieser sechs Bytes enthält ein Nibble.

Label: Z_D2Z

ROM-Nummer: D

Startadresse: &FE88

Einsprungbedingungen: Das Highbyte aller drei Quellbytes muß identisch sein, das Highbyte aller sechs Zielbytes ebenfalls.
DE = Zeiger auf erstes von sechs Zielbytes: H,H,M,M,S,S.
HL = Zeiger auf letztes v. drei Quellbytes: Sek,Min,Stu. z.B.: UHR_STU

Aussprungbedingungen: Ab DE stehen sechs Zeit-Bytes im RAM.

Manipuliert: AF, BC, E, L und die sechs Zielbytes.

Beschreibung: Die Echtzeituhr von Dobbertin liefert die Zeitdaten in einem speziellem Format. Sekunden, Minuten und Stunden sind je in einem Byte zusammengefasst. Dabei wird das BCD Format verwendet. Enthalten die Zeit-Bytes die Werte &56 (Sekunde), &37 (Minute) und &18 (Stunde), dann ist es eben 18:37:56 Uhr.
Das Dobbertin Format ist zwar platzsparend, will man aber die Uhrzeit editieren, hätte man gerne jedes Nibble für sich.
Diese OS Funktion wandelt drei Dobbertin-kompatible Bytes (Sekunde -> Minute -> Stunde, wobei HL auf die Stunde zeigt) in sechs einzelne Nibbels um (Stunde high, Stunde low, Minute high, Minute low, Sekunde high, Sekunde low). Diese sechs Nibbles werden ab DE abgelegt, und zwar in der Reihenfolge, in der sie auch gelesen werden. -->
Wenn wir bei obigem Beispiel bleiben: &01,&08,&03,&07,&05,&06 ab DE.
Achtung: Der Zeiger auf die drei Quellbytes, also HL, zeigt auf das LETZTE der drei Bytes. HL zeigt also auf die Stunde.
Das von dieser OS Funktion erzeugte Zeit Format ist zur OS Funktion ZEEZ kompatibel, ZEEZ erlaubt es die Zeit-Daten video-orientiert zu editieren. Siehe dort ...

Bitte Beachten: Achtung, die Highbytes aller Quellbytes bzw. aller Zielbytes müssen identisch sein.
Es darf die Code-Page nicht verlassen werden. >=> DE < &XXFA /// HL > &XX02.
Die Zielbytes (ab DE) werden zwar ganz normal speicheraufwärts geschrieben, jedoch werden die Quellbytes (unter HL) speicherabwärts gelesen!!!

EINZELNE BYTES IN DOBBERTIN-ZEIT KOMPRIMIEREN

Kurzbeschreibung: Die von Z_D2Z(s.o.) expandierten Bytes werden wieder komprimiert, und so zur Dobbertin-Echtzeituhr kompatibel gemacht.

Label: Z_Z2D

ROM-Nummer: D

Startadresse: &FE8B

Einsprungbedingungen: Das Highbyte aller sechs Quellbytes muß gleich sein, das Highbyte aller drei Zielbytes ebenfalls.
DE = Zeiger auf erstes v. sechs Quellbytes: H,H,M,M,S,S.
HL = Zeiger auf letztes von drei Zielbytes: Sek,Min,Stu. z.B.: UHR_STU

Aussprungbedingungen: Unter HL stehen drei Zeit-Bytes im RAM.

Manipuliert: AF, BC, E, L und die drei Zielbytes.

Beschreibung: Diese OS Funktion ist das Gegenstück zu Z_D2Z (siehe oben).
Hat man zuvor die Dobbertin-Zeit in ZEEZ kompatibles Format expandiert, so dient diese OS Funktion dazu, diese sechs Nibbles wieder zu drei Bytes zusammenzufassen. Die sechs Quell-Nibbles werden ab DE gelesen, und die drei Ziel-Bytes werden unter HL geschrieben. Auch hierbei zeigt HL wieder auf das letzte der drei Ziel-Bytes. Im Speicher sind drei Bytes für Sekunde, Minute und Stunde vorgesehen, dabei zeigt HL auf das Byte für die Stunde.

Bitte Beachten: Achtung, die Highbytes aller Quellbytes bzw. aller Zielbytes müssen identisch sein. Es darf die Code-Page nicht verlassen werden. >=> DE < &XXFA /// HL > &XX02.

DOBBERTIN-DATUM IN EINZELNE BYTES EXPANDIEREN

Kurzbeschreibung: Drei Dobbertin-kompatible Datum Bytes werden zu sechs Bytes expandiert. Jedes dieser sechs Bytes enthält ein Nibble.

Label: Z_D2J

ROM-Nummer: D

Startadresse: &FE8E

Einsprungbedingungen: Das Highbyte aller drei Quellbytes muß identisch sein, das Highbyte aller sechs Zielbytes ebenfalls.
DE = Zeiger auf sechs Zielbytes: T,T,M,M,J,J.
HL = Zeiger auf drei Quellbytes: T, M, J. // z.B. UHR_TAG

Aussprungbedingungen: Ab DE wurden sechs Datums-Bytes geschrieben.

Manipuliert: AF, BC, E, L und die sechs Zielbytes.

Beschreibung: Die Echtzeituhr von Dobbertin liefert auch die Datums-Daten in einem speziellem Format. Tag, Monat und Jahr sind je in einem Byte zusammengefasst. Dabei wird das BCD Format verwendet.
Enthalten die Datums-Bytes die Werte &21 (Tag), &04 (Monat) und &69 (Jahr), dann bedeutet das den 21. April '69.
Will man das Datum editieren, dann hätte man gerne jedes Nibble für sich.
Diese OS Funktion wandelt drei Dobbertin-kompatible Bytes ab HL (Tag, Monat, Jahr) in sechs einzelne Nibbels um (Tag high, Tag low, Monat high, Monat low, Jahr high, Jahr low).
Diese sechs Nibbles werden ab DE abgelegt, und zwar in der Reihenfolge, in der sie auch gelesen werden.
Wenn wir bei obigem Beispiel bleiben: &02,&01,&00,&04,&06,&09 ab DE.
Das von dieser OS Funktion erzeugte Datums Format ist zur OS Funktion ZEED kompatibel, ZEED erlaubt es die Datums-Daten videoorientiert zu editieren. Siehe dort ...

Bitte Beachten: Achtung, die Highbytes aller Quellbytes bzw. aller Zielbytes müssen identisch sein. Es darf die Code-Page nicht verlassen werden. >=> DE < &XXFA /// HL < &XXFD.

EINZELNE BYTES IN DOBBERTIN-DATUM KOMPRIMIEREN

Kurzbeschreibung: Die von Z_D2J(s.o.) expandierten Bytes werden wieder komprimiert, und so zur Dobbertin-Echtzeituhr kompatibel gemacht.

Label: Z_J2D

ROM-Nummer: D

Startadresse: &FE91

Einsprungbedingungen: Das Highbyte aller sechs Quellbytes muß gleich sein, das Highbyte aller drei Zielbytes ebenfalls.
DE = Zeiger auf sechs Quellbytes: T,T,M,M,J,J.
HL = Zeiger auf drei Zielbytes: T, M, J. // z.B. UHR_TAG

Aussprungbedingungen: Ab HL wurden drei Datums-Bytes geschrieben.

Manipuliert: AF, BC, E, L und die drei Zielbytes.

Beschreibung: Diese OS Funktion ist das Gegenstück zu Z_D2J (siehe oben).
Hat man zuvor das Dobbertin-Datum in ZEED kompatibles Format expandiert, so dient diese OS Funktion dazu, diese sechs Nibbles wieder zu drei Bytes zusammenzufassen. Die sechs Quell-Nibbles werden ab DE gelesen, und die drei Ziel-Bytes werden ab HL geschrieben.

Bitte Beachten: Achtung, die Highbytes aller Quellbytes bzw. aller Zielbytes müssen identisch sein. Es darf die Code-Page nicht verlassen werden. >=> DE < &XXFA /// HL < &XXFD.

UHR-ZEIT ODER DATUM VIDEO-ORIENTIERT EDITIEREN

Kurzbeschreibung: Diese beiden OS Funktionen erlauben es das Datum (ZEED) bzw. die Uhr-Zeit (ZEEZ) video-orientiert zu editieren.

Label: ZEED (Datum) bzw. ZEEZ (Uhr-Zeit).

ROM-Nummer: D

Startadresse: &FE97 (ZEED) /// &FE94 (ZEEZ)

Einsprungbedingungen: Die Variable C_POS markiert die linke obere Ecke, des Editierfeldes. Das Editierfeld ist 22 Zeichen breit und drei Zeilen hoch. Es muß im Mode 2 (64*32) aktiv sein.
HL = Zeiger auf sechs Bytes Datum bzw. Zeit, wobei HL < &XXFA !!!

Aussprungbedingungen: Das Z-Flag informiert über den Ausgang des Editiervorgangs:
Zero-Flag geleert => alles o.k. es wurde mit Copy beendet.
Zero-Flag gesetzt => Das Editieren wurde mit ESC abgebrochen!

Manipuliert: AF, BC, DE, HL, IX, C_POS, REG08_0,1, REG16_0,1,2 und die sechs Bytes ab HL.

Beschreibung: Mit Hilfe dieser beiden OS Funktion läßt sich das Datum bzw. die Uhrzeit video-orientiert editieren. Will man das Datum verändern, dann benutzt man ZEED, um die Zeit auf den neusten Stand zu bringen benutzt man ZEEZ. Beide OS Funktionen sind sich sehr ähnlich.
Vor dem Einsprung ist auf Bildschirmmodus 2 und 64 Zeichen mal 32 Zeilen zu schalten. Die Variable C_POS ist mit der linken oberen Ecke des Editierfensters zu laden. Dieses Editierfenster ist 22 Zeichen breit und 3 Zeilen hoch, denn es werden die großen Zahlen-Sätze verwendet. Außerdem muß HL auf sechs Bytes zeigen, die entweder das Datum oder die Zeit beinhalten.

```
Format des Datums: &02,&01,&00,&04,&06,&09 => 21. 4.69
Format der Zeit..: &01,&08,&03,&07,&05,&06 => 18:37:56
```

Um diese sechs Bytes aus den Daten der Dobbertin-Uhr zu generieren kann man die OS Funktion Z_D2J (Datum) bzw. Z_D2Z (Zeit) benutzen.
Der Editiervorgang wird mit den Cursortasten durchgeführt. Copy beendet korrekt, wogegen die ESC Taste abbricht.
Das Zero Flag gibt Aufschluß ob mit Copy beendet wurde (Z=0) oder ob mit ESC abgebrochen wurde (Z=1).
In jedem Fall ist es aber mögich dass die Bytes ab HL manipuliert wurden, u. U. kann es sinnvoll sein sie an anderer Stelle zu sichern.

Bitte Beachten: Vor Aufruf der OS Funktion muß Mode 2 aktiv sein. Der Bildschirm muß auf 64 Zeichen mal 32 Zeilen geschaltet werden.
Ist das Z-Flag beim Rücksprung gesetzt, dann hat der Anwender mit ESC abgebrochen.
Beachte: HL muß kleiner als &XXFA sein !!! Die sechs Edit-Bytes dürfen keinen Page-übertritt haben. HL & HL + 6 müssen selbes Highbyte haben.

RE-MARKIEREN ALLER ALT-MARKIERTEN DATEIEN

Kurzbeschreibung: Alle bearbeiteten Dateien, also alle alt-markierten Dateien eines Laufwerks werden neu markiert.

Label: RTLW

ROM-Nummer: D

Startadresse: &FE82

Einsprungbedingungen: HL = TURBO_A..M, Systemvariablen korrekt!

Aussprungbedingungen: Alle bearbeiteten Dateien sind neu markiert.

Manipuliert: AF, BC', D', HL' und u.U. einige TMS_? des LWs.

Beschreibung: Hat man eine Reihe von markierten Dateien bearbeitet, dann hat sich ihr Status von "markiert" auf "alt-markiert" geändert.
Alle markierten Dateien werden im Desktop unterstrichen dargestellt, alle alt-markierten Dateien werden durchgestrichen dargestellt.
Will man alle bereits bearbeiteten Dateien nochmals bearbeiten, dann sollte man sie zuerst neu markieren. Dies erledigt diese OS Funktion.
Alle alt-markierten Dateien eines Laufwerks / einer Partition A...M werden neu markiert.
Andere, bereits markierte, Dateien werden nicht beeinflusst, bleiben also ebenfalls markiert.
Beim Einsprung ist das Register HL mit dem Zeiger auf eine der RAM-Variablen TURBO_A bis TURBO_M zu laden. Dadurch wird das zu bearbeitende Laufwerk selektiert. Beispiel einer Anwendung:

Zuerst wird durch das Register A das Laufwerk gewählt.

```
LD   A,LW  ;Laufwerk A..M, symbolisiert durch 0..12.
RLCA
RLCA
RLCA            ;LW mit 8 multiplizieren
LD   HL,TURBO_A ;Zeiger auf TURBO_A (erstes Laufwerk)
ADD  A,L
LD   L,A   ;TURBO_A + (LW * 8)

CALL RTLW  ;OS Funktion aufrufen, alte Dateien neu markieren
```

Vor Aufruf der OS Funktion kann man das LW-Tag-Byte direkt aus TURBO_? laden: " LD A,(HL) " und testen, ob das LW überhaupt aktiv ist.

Bitte Beachten: Bereits markierte Dateien bleiben weiterhin markiert.
Die Laufwerkswahl funktioniert etwas umständlich über HL.

ALLE DATEIEN EINES LAUFWERKS ENTMARKIEREN

Kurzbeschreibung: Alle Dateien eines Laufwerks werden entmarkiert.

Label: UTLW

ROM-Nummer: D

Startadresse: &FE85

Einsprungbedingungen: HL = TURBO_A..M, Systemvariablen korrekt!

Aussprungbedingungen: Alle bearbeiteten Dateien sind entmarkiert.

Manipuliert: AF, BC', D', HL' und u.U. einige TMS_? des LWs.

Beschreibung: Alle markierten Dateien eines Laufwerks / einer Partition A..M werden entmarkiert.
Alle markierten Dateien werden im Desktop unterstrichen dargestellt, alle nicht-markierten Dateien werden normal dargestellt.
Beim Einsprung ist das Register HL mit dem Zeiger auf eine der RAM-Variablen TURBO_A bis TURBO_M zu laden. Dadurch wird das zu bearbeitende Laufwerk selektiert. Beispiel einer Anwendung:

Zuerst wird durch das Register A das Laufwerk gewählt.

```
LD   A,LW   ;Laufwerk A..M, symbolisiert durch 0..12.
RLCA
RLCA
RLCA            ;LW mit 8 multiplizieren
LD   HL,TURBO_A ;Zeiger auf TURBO_A (erstes Laufwerk)
ADD  A,L
LD   L,A    ;TURBO_A + ( LW * 8 )

CALL UTLW   ;OS Funktion aufrufen, alle Dateien ent-markieren
```

Vor Aufruf der OS Funktion kann man das LW-Tag-Byte direkt aus TURBO_? (? = A-M) laden: " LD A,(HL) " und testen, ob das LW überhaupt aktiv ist.

Bitte Beachten: Die Laufwerkswahl erfolgt etwas umständlich über HL.

AUF HEGOTRON GRAFPAD 2 TESTEN

Kurzbeschreibung: Testen, ob ein Hegotron Grafpad 2 angeschlossen ist.

Label: T_GP2

ROM-Nummer: D

Startadresse: &E714

Einsprungbedingungen: -

Aussprungbedingungen: Der Akku gibt Aufschluss darüber, ob das Hegetron Grafpad 2 am CPC angeschlossen ist.
A = &00 und Zero-Flag gesetzt --> es ist kein GP2 angeschlossen
A = &01-05 und Zero-Flag geleert --> das GP2 ist angeschlossen

Manipuliert: AF und BC.

Beschreibung: Diese OS Funktion überprüft, ob am CPC das Grafpad 2 der Firma Hegotron angeschlossen und betriebsbereit ist. Diese OS Funktion wird ohne Parameter aufgerufen, nach ihrer Rückkehr geben Akku und das Zero Flag Aufschluss über das eventuelle Vorhandensein des Grafpad 2.
Kehrt die OS Funktion mit dem Wert &00 im Akku und gesetztem Zero Flag zurück, so ist kein Grafpad 2 angeschlossen. Enthält der Akku einen anderen Wert als Null und ist das Zero Flag gelöscht, so ist das Grafpad 2 angeschlossen.

Beispiel:

```
 CALL T_GP2    ;Auf Grafpad 2 testen.
 JR   Z,NO_GP2 ;Es ist KEIN GP 2 vorhanden!

GP2 ...        ;GP2 ist vorhanden!

...
...
```

Bitte Beachten: -

ACHTUNG: Die Einsprungadresse ist seit FutureOS Version 0.8 geändert!

ABFRAGE DES HEGOTRON GRAFPAD 2

Kurzbeschreibung: Abfrage der X- und Y-Koordinaten, sowie der beiden Knöpfe E und S des Hegotron Grafpad 2.

Label: G_GP2

ROM-Nummer: D

Startadresse: &E725

Einsprungbedingungen: -

Aussprungbedingungen: Die Register DE und HL geben über X- und Y-Koordinaten und die Knöpfe E und S Aufschluss.
DE = Y-Koordinate, theoretisch: &0000-&04FF, praktisch: &001B-&0484
HL = X-Koordinate, theoretisch: &0000-&05FF, praktisch: &0020-&05DF
In Register H ist ausserdem der Status der beiden Knöpfe E (Bit 7, MSB von H) und S (Bit 6 von H) enthalten. Dabei symbolisiert ein geleertes Bit eine gedrückte Taste. Ist das entsprechende Bit gesetzt, so wird die zugehörige Taste nicht gedrückt.

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

Beschreibung: Ist am CPC ein Hegotron Grafpad 2 angeschlossen (bitte zuvor testen!), dann lassen sich mit G_GP2 die aktuellen Koordinaten des Eingabestiftes und der Knöpfe abfragen. Nach dem Aufruf der OS Funktion steht im Register DE die Y-Koordinate des Stiftes, dabei sind theoretisch Werte zwischen &0000 und &04FF möglich, praktisch deckt das Grafiktablett etwa den Bereich von &001B bis &0484 ab (das kann allerdings von Gerät zu Gerät leicht variieren). Kleine Y-Werte sind unten, nahe beim Anwender. Große Y-Werte sind oben, vom Anwender weg.

Im Register HL wird die X Koordinate (Bits 0-10) und der Status der beiden Knöpfe übergeben. Die X-Koordinate kann dabei theoretisch zwischen &0000 und &05FF liegen, praktisch wird der Bereich von etwa &0020 bis &05DF abgedeckt. Kleine X-Werte befinden sich links, große X-Werte befindne sich auf der rechten Seite des Grafiktabletts.

In den obersten beiden Bits (7, 6) von Register H sind die Knöpfe E (Bit 7) und S (Bit 6) verschlüsselt:

- Ist Bit 7 gelöscht, so wird Taste E (Entry, Exit) gerade gedrückt.
  Ist Bit 7 dagegen auf 1 gesetzt, so wird E gerade nicht gedrückt.

- Ist Bit 6 auf 0 geleert, so wird Taste S (Select) gerade gedrückt.
  Ist Bit 6 dagegen auf 1 gesetzt, so wird S gerade nicht gedrückt.

Beispiel:

```
 CALL G_GP2 ;Grafpad 2 Koordinaten und Knöpfe abfragen.

 LD   (GP2_X_Koordinate),DE ;X-Koordinate ins RAM
 LD   A,H
 AND  A,&C0                 ;Bits 7 und 6 (Tasten E und S) isolieren,
 LD   (KEYS_E_AND_S),A      ;und Tasten-Status (??00 0000) im RAM sichern
 RES  7,H
 RES  6,H
 LD   (GP2_Y_Koordinate),HL ;Y-Koordinate ins RAM
...
```

Bitte Beachten: Diese OS Funktion darf NUR dann verwendet werden, wenn zuvor die Bereitschaft des Grafpad 2 am CPC geprüft wurde (siehe OS Funktion T_GP2).

ACHTUNG: Die Einsprungadresse ist seit FutureOS Version 0.8 geändert!

ABFRAGE DER KNöPFE E UND S DES HEGOTRON GRAFPAD 2

Kurzbeschreibung: Die beiden Knöpfe E und S des Hegotron Grafpad 2 werden abgefragt.

Label: K_GP2

ROM-Nummer: D

Startadresse: &E750

Einsprungbedingungen: -

Aussprungbedingungen: Der Akku gibt Aufschluss darüber, ob einer der beiden Knöpfe E (Entry, Exit) und/oder S (Select) gedrückt wird.
Register A Bit 7 entspricht Knopf E.
Register A Bit 6 entspricht Knopf S.
Dabei symbolisiert ein gelöschtes Bit eine gedrückte Taste, ein gesetztes Bit zeigt an, dass die Taste nicht gedrückt wird.

Manipuliert: AF und BC.

Beschreibung: Ist am CPC ein Hegotron Grafpad 2 angeschlossen (bitte zuvor testen!), dann läßt sich mit K_GP2 der aktuelle Status der Knöpfe abfragen. In den obersten beiden Bits (7, 6) von Register A sind die Knöpfe E (Bit 7) und S (Bit 6) verschlüsselt:

- Ist Bit 7 gelöscht, so wird Taste E (Entry, Exit) gerade gedrückt.
  Ist Bit 7 dagegen auf 1 gesetzt, so wird E gerade nicht gedrückt.

- Ist Bit 6 auf 0 geleert, so wird Taste S (Select) gerade gedrückt.
  Ist Bit 6 dagegen auf 1 gesetzt, so wird S gerade nicht gedrückt.

Beispiel:

```
CALL K_GP2 ;Grafpad 2 Koordinaten und Knöpfe abfragen.

AND A,&C0 ;Bits 7 und 6 isolieren (??00 0000)
LD (KEYS_E_AND_S),A) ;und Status der Tasten im RAM sichern

...
```

Bitte Beachten: Diese OS Funktion darf NUR dann verwendet werden, wenn zuvor die Bereitschaft des Grafpad 2 am CPC geprüft wurde (siehe OS Funktion T_GP2).

ACHTUNG: Die Einsprungadresse ist seit FutureOS Version 0.8 geändert!

EINLESEN DER INHALTSVERZEICHNISSE ALLER MARKIERTEN LAUFWERKE

Kurzbeschreibung: Die Inhaltsverzeichnisse aller markierten Laufwerken und Festplatten-Partitionen werden eingelesen und im E-RAM gepuffert.

Label: GET_DIR

ROM-Nummer: D

Startadresse: &FE7C

Einsprungbedingungen: Laufwerke und Festplatten-Partitionen deren Inhaltsverzeichnis eingelesen werden soll müssen in den System-Variablen TUBBO_A bis TURBO_M markiert sein (siehe Datei #OS-VAR.DEU).
Es muss genügend freies Erweiterungs-RAM (E-RAM) vorhanden sein, um die Inhaltsverzeichnisse puffern zu können.

Aussprungbedingungen: Die Inhaltsverzeichnisse wurden eingelesen.
Der Akku gibt Aufschluss über den Erfolge der OS Funktion:
A = &00 => Alle Inhaltsverzeichnisse wurden erfolgreich eingelesen.
A = &01 => Es war kein Laufwerk markiert, es wurde nichts gelesen.
A = &02 => Es war nicht genügend E-RAM frei. Abbruch. Nichts gelesen!

Manipuliert: AF, BC, DE, HL, AF', BC', DE', HL', IX, TURBO_A..M, FDC, Laufwerks-Motoren und einige E-RAMs.

Beschreibung: Diese High-Level Funktion des FutureOS erlaubt es die Inhaltsverzeichnisse von Floppy Disk Laufwerken und Partitionen der HD20 Festplatte (Dobbertin) einzulesen und im E-RAM zu puffern.

Die eingelesenen Inhaltsverzeichnisse werden sortiert. Anschließend wird eine Kopie davon (das 16er DIR) erzeugt, welches nur die Datei-Namen enthält und in dieser Form direkt am Bildschirm angezeigt werden kann (siehe OS Funktionen SCRI, SC_AU, SC_AB im ROM D).

Um Laufwerke und/oder Partitionen zum lesen des Inhaltsverzeichnisses auszuwählen muss Bit 0 der entsprechen Variablen (TURBO_A, B, C,... M) gesetzt werden. Bei allen andernen muss das Bit 0 gelöscht werden, um sie nicht auch einzulesen.

Nach dem Aufruf der OS Funktion GET_DIR gibt der Akku über den Erfolg der Funktion Aufschluss. Wenn der Akku auf &00 gesetzt wurde, dann ist alles in Ordnung. Sollte der Akku allerdings &01 enthalten, dann war beim Aufruf der OS Funktion gar kein Speicher-Medium markiert, es konnte also auch nichts gelesen werden. Enthält der Akku den Wert &02, dann war nicht genügend E-RAM vorhanden, auch in diesem Fall wurde nichts gelesen.

Für den Speicher-Verbrauch kann folgende Regen zur Anwendung konnen:
Data, System Format benötigen maximal: 3 KB
Vortex Format benötigt maximal.....: 6 KB
HD20 Partition benötige maximal.....: 24 KB
RAM-Diskette (M) benötigt maximal.....: 2 KB
Zusätzlich werden 16 KB als Puffer für das untere RAM benötigt.

Beispiel: Laufwerk A (Data) und B (Vortex) benötigen 25 KB E-RAM.

Bitte Beachten: Diese High-Level OS Funktion verändert alle Register ausser IY und I. Bei zu wenig E-RAM kommt es zum Abbruch.
Laufwerke bzw. Partitionen die nicht eingelesen werden sollen müssen in ihren TURBO_? (? = A..M) Variablen deaktiviert werden.

In älteren FutureOS Versionen ist diese OS Funktion nicht vorhanden!

KONVERTIERE ZWEI ASCII-ZEICHEN IN EINEN 8 BIT WERT

Kurzbeschreibung: Zwei ASCII-Zeichen werden in einen 8 Bit Wert konvertiert.

Label: CC2N

ROM-Nummer: D

Startadresse: &FE7F

Einsprungbedingungen: HL enthält zwei ASCII-Zeichen ("0"-"9", "A"-"F")
H enthält das höherwertige und L das niederwertige Zeichen.

Aussprungbedingungen: A = 8 Bit Wert &00-&FF

Manipuliert: AF und B

Beschreibung: Diese OS Funktion dient dazu zwei ASCII-Zeichen (0-9, A-F), die im Register HL enthalten sind in einen 8 Bit Wert zu konvertieren.
Bei Aufruf der OS Funktion enthält H das höherwertige Zeichen und L enthält das niederwertig Zeichen. Nach dem Rücksprung der Funktion enthält der Akku einen 8 Bit Wert, der sich folgendermaßen errechnet:

Beispiel:

Das Register HL wird mit &3742 geladen. Wobei &42 = ASC("B") ist.

```
LD   HL,&3742
CALL CC2N
```

Der Akku enthält nun &7B.

Diese OS Funktion eignet sich lediglch dazu Zeichen von "0" bis "9" und "A" bis "F" in einen 8 Bit Wert zu konvertieren.

Bitte Beachten: Die zu konvertierenden Zeichen müssen zwischen "0" und "9" (&30-&39) bzw. "A" und "F" (&41-&46) sein.