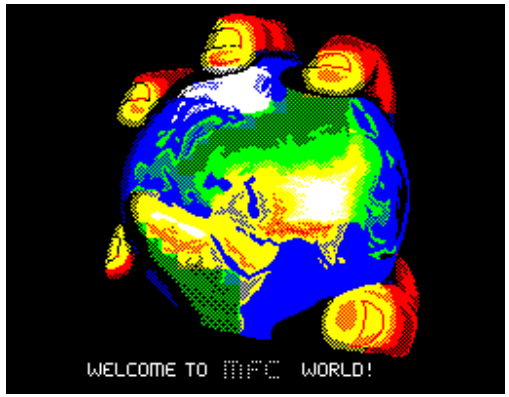
Příručka uživatele

MDOS File Commander
(MFC)

verze 3.0504



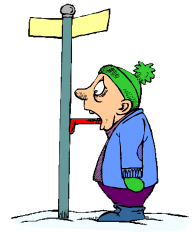
"Kostky jsou vrženy."

Rubik

*Určeno pro počítače **ZX Spectrum**, **Didaktik***
*s (nejen) diskovým systémem **D40/80***

Obsah

0. Autorská práva, poděkování



Oproti dřívějším verzím MFC ta třetí už tak "free" není. Ale nebojte, normálnímu uživateli nic nehrozí, jde mi hlavně o ty vyčůrance...

Licenční podmínky (MFC v3):

1. Program je šířen výhradně prostřednictvím autorových internetových stránek a to zdarma. Ostatní (libovolné) formy šíření jsou dovoleny pouze v případě, že ten, kdo program šíří, se tímto jednáním nesnaží něco "vytěžit". To něco znamená jakýkoliv prospěch, tedy nejen peníze.

2. Program může používat kdokoliv, kdykoliv, na jakýchkoliv platformách a to úplně zadarmo. Vyjímkou je nasazení programu do komerční sféry a v podstatě tak jeho podílení se na ziskovosti organizace. V tomto případě je třeba za používaní programu platit.

3. Kód programu si uživatel může libovolně upravit. Takováto upravenina (ani návod k ní) už však nesmí být šířena dále. Slušností je informovat původního autora programu a požádat ho, aby úpravu posoudil, příp. ji i zveřejnil nebo sám zahrnul do dalších verzí.

4. Pokud uživatel narazí na nějakou chybu v programu, měl by o této skutečnosti informovat autora. Chyba a veškeré její následky jsou však problémem uživatele, autor nezaručuje, že jeho program je bez chyb. Autor také v žádném případě nenese odpovědnost za škodu, kterou uživatel používáním programu způsobí jiným osobám.

5. Stejná pravidla platí i pro veškerý software přímo související s MFC (např. utility a šetřiče) pokud u něj nejsou stanoveny podmínky jiné.

6. Tyto licenční podmínky je možné porušit pouze s písemným souhlasem autora.

Musím také upozornit, že **úvodní obrázek v programu NENÍ mým výtvorem.** Našel jsem ho ve svém archívu a moc se mi líbil. Tak jsem si jej "vypůjčil" (a upravil). Autora bych moc rád

uvedl, ale bohužel jej neznám (vím jen, že ten obrázek pochází z nějakého setkání spectristů, kde soutěžil).

Nakonec bych rád poděkoval několika lidem, bez kterých by MDOS File Commander nebyl takový, jaký je.

V prvé řadě děkuji **Universumovi** za Prometheus 128; **Sweetovi** za ochotu spolupracovat, připomínky, kontrolu manuálu a hlavně sqělou formátovací rutinku pro MDOS 1.0; **Peterovi Turányimu** (alias Softhouse) za skvělý prográmek na výpis 24 bitového čísla; **Tritolovi** za šetřič, ukázku programu na spouštění PC snapů a podporu MFC snapu v DEMFIRu; **TAUovi** za povzbuzování a kontrolu manuálu; **Pokemu** za snahu u snap konvertorů pro PC (mrzí mě, že to dopadlo jak to dopadlo…); mému dlouholetému kamarádovi **Jindrovi Trojkovi** (alias Maniac Boy Computer) za pomoc při "user friendly" D_0 konvertoru a objasnění základů C++; **Velesoftovi** za vymyšlení Pentagon úpravy i na běžné 128/+2; no a hlavně **Busymu**, že mi ke konci práce na MFC sdělil, že už 8 let má doma zdroják EIMu…

MTs
mts.zxs@tiscali.cz

PS: A taky děkuji Billu Gatesovi, že se nikdy nepustil do vývoje softwaru pro ZX Spectrum.

1. Bláboly, které můžete přeskočit

Představuji Vám nový *File Commander* pro disketové mechaniky D40 a D80 (hlavně pro D80, neboť pouze s 3,5" disketou program pořádně využijete…).

Proč jsem vůbec začal dělat na tomto programu? Řeknu Vám, že Export-Import Manager (EIM) od MVD mi pil krev už ode dne prvního použití. Rozhodně tady ale EIM (ani jeho autora) nechci pomlouvat, nakonec mohli jsme být rádi, že je alespoň nějaký program než žádný. Časem jsem se ale naučil assembler a čím víc jsem toho uměl, tím víc věcí mi na EIMu vadilo (člověk si pořád říká, *sakra vždyť tohle přece není nic složitého, já bych to napsal daleko lépe a rychleji*…). Jak ale víme, tak programátor je velice líný člověk a je lepší kecat a pomlouvat, než sednout za klávesnici a "to lepší" udělat.

Mě se třeba vždycky líbil TOOLS od Proximy. Jediné mínus tohoto programu bylo (a stále je), že neumí práci EIMu (tj. pracovat s PC DD disketou). Když jsem zjistil, že Sweet Factory získal zdroják a předělal jej na 128K, neváhal jsem a okamžitě jsem mu navrhl, ať tam tu podporu PC disket dodělá (jó, "chytat hady cizíma rukama", na to jsem expert…). Sweet mě však bohužel nepotěšil. Řekl, že než se rýpat v TOOLSu, tak to raději napsat celé úplně od začátku. Myslím, že jsme tenkrát oba začali něco tvořit, ale zanedlouho jsme s tím přestali. Sweet mi poslal, co měl - kousek zdrojáku (myslím, že tam něco dělal i Shrek) a dokument o struktuře PC disket a disků. Do zdrojáku jsem ale nikdy pořádně nekoukal (pro mne je většinou rychlejší si program napsat sám, než hodiny zkoumat, co dělá ten cizí, abych si jej mohl předělat podle svého…).

Když už jsem tedy začínal úplně od nuly, rozhodl jsem se, že si kódování vychutnám a napíšu to, jak nejlíp dovedu. Řeknu Vám, stálo mě to šíleného času, nervů a potu, ale (jó, já vím, že vlastní chvála smrdí, ale...) mám z toho moc dobrý pocit a myslím, že MFC se opravdu povedl. V mých očích je prostě dokonalý :).

2. Přednosti programu

- funkčnost na obou oficiálních verzích MDOSu (1.0, 2.0) i na neoficiálních (3, 1.1, 2.1, divIDE)
- možnost práce i s mechanikami C, D
- D80 akcelerace (MFC narozdíl od MDOSu čte i z 80té stopy stejně rychle jako z nulté)
- podpora MS-DOS disket z PC (pouze DD 3,5")
- ScanDisk (podrobná kontrola a analýza každé diskety)
- ShakeDir ("setřesení" adresáře)
- naprosto unikátní formátovací a kopírovací systém
- možnost vlastních utilit
- šetrnost k disketě a mechanice (neplatí při 48K RAM)
- komfort, který není běžný ani na PC
- využití 128K/256K/512K/1024K RAM
- fungování i v emulátoru Real Spectrum na PC

V následujících kapitolách rozeberu vše podrobněji…

3. MFC a emulátory

Snažil jsem se program udělat plně funkční i na PC emulátorech. Bohužel žádné hnusné PC nikdy nedokáže emulovat ZX Spectrum (Didaktik) na 100%…

MFC jsem testoval **pouze** na emulátoru **Real Spectrum** (www.ramsoft.bbk.org/realspec.html), protože mimo jiné podporuje i systém MDOS (D40/80) a je asi i nejpoužívanější.

Real Spectrum (dále jen RS) ovšem není v emulaci MDOSu zcela bezchybný a emuluje opravdu jen tak, aby se neřeklo. Problémy mohou nastat při formátování diskety, protože RS jaksi zatuhne při špatném sektoru. Dále pracuje jen se standardními formáty (40x09, 80x09, 40x10, 80x10) a vždy počítá s oboustranným.

4. Práce s disketami

MFC pracuje se všemi typy disket (**5,25" a 3,5"**) a formáty (jednostranný, oboustranný) **MDOSu**. Pozor však na formáty s polovičním počtem stop - 40 u D80 a 20 u D40. MDOS díky emulaci 5,25" 16mi mechaniky na takto naformátované diskety použije dvojnásobný posun hlavy (viz. příloha). Výsledkem je, že disketa je prakticky nepoužitelná, protože je z ní dovoleno pouze číst. MFC tu emulaci (tj. ten dvojnásobný posun hlavy) zapne pouze v případě, že bude v mechanice, která má 80 stop vložena disketa naformátovaná na 40 stop. Jinak ne (narozdíl od Basicu či ostatních commanderů)!

MFC podporuje i **MS-DOS** (operační systém z PC). Ovšem **pouze 3,5"** diskety naformátované na DD (**720kB**). Podporu 5,25" disket jsem do programu úmyslně nedal (zbytečné prodloužení kódu), protože vidět dneska v PC pětačtvrtku je opravdový zázrak!

5. 128K/256K/512K/1024K

Program umí pracovat i stránkami klasické stodvacetosmičky nebo ruského Pentagonu (Pentagon má rozšířenou paměť na bitech 7, 6 a 5 portu 32765). Využívá je však **pouze** při kopírování souborů.

Paměť pro data:

- **48K** **10 240 bytes**
- **128K** **92 160 bytes**
- **256K** **223 232 bytes**
- **512K** **485 376 bytes**
- **1024K** **1 009 664 bytes**

Jak vidíte, tak **u 48K** kopírování rozhodně svižné vůbec nebude a doporučuji proto **používat opravdu jen v případě nouze.** Kopírování v MFC je děláno hlavně pro 128K a 2 mechaniky (nelépe obě 3,5"). Kopírování na 48K je hrozně pomalé, to raději zůstaňte u SingleCopy nebo TOOLSe.

Detekci paměti 48K vs. 128K MFC provádí automaticky. Vyšší paměť už ale netestuje! Chcete-li tedy používat 256K, 512K nebo 1024K, musíte si to nastavit sami (ručně) po každém spuštění MFC v setupu!

6. Spuštění MFC, Setup

Po nahrání MFC do počítače vidíte úvodní obrázek a program čeká na stisk libovolné klávesy. Pokud nyní **přidržíte SPACE** (mezerník), vyvolá se jednorázový setup. Jednorázový proto, protože později, během provozu MFC, už vyvolat nelze (není místo…). Můžete nastavit tyto parametry:

changes (klávesa **C**)

- ***MFC system*** (201) – změny (např. změna jména souboru, adresy, atributu…) nejsou ukládány ihned. Uživatel si je musí uložit sám klávesou X.
- ***save immediately*** (0) – změny jsou ukládány okamžitě.

Tato volba vznikla na žádost Sweeta. Zvyk (TOOLS) je totiž železná košile… Nicméně rozhodně doporučuji *save immediately* nezapínat a zvyknout si na klávesu X.

format (klávesa **F**)

- ***MFC system*** (0) – u MDOS 1 se použije speciální formátovací rutina, u MDOS 2 rutina z ROM.
- ***always DROM*** (201) – vždy se použije podprogram z ROM.

load-copy (klávesa **L**)

- ***skip when error*** (0) – pokud při LOADu u kopírování nebo exportu/importu dojde k chybě, sektor se přeskočí a pokračuje se dále.
- ***halt when error*** (201) – chyby jsou vždy hlášeny a operace je předčasně ukončena.

error (CRC) (klávesa **E**)

- ***fill "?"+128*** (0) – pokud při LOADu u kopírování nebo exportu/importu dojde k chybě *CRC error*, sektor se přeskočí a jeho místo v paměti zaplní kód "?"+128.
- ***ignore*** (201) – při CRC chybě se získaná data zachovají (jsou ale samozřejmě chybná).

Volba má pochopitelně smysl **pouze tehdy**, pokud je nastaveno *skip when error.*

hotkeys (klávesa **H**)

- ***original MFC*** (0) – horké klávesy tak, jak je popisuje tento manuál.
- ***Shrek's MBC*** (201) – některé horké klávesy jsou přizpůsobeny Shrekovému MBC:

SPACE	*označení souboru*
ENTER	*run souboru (SS+R odebráno)*
F	*quick formát (SS+F odebráno)*
R	*přejmenování diskety (V odebráno)*
5	*kopírování (C odebráno)*
8	*delete souboru (SS+D odebráno)*
9	*přejmenování souboru*
CS+3	*edit (CS+1 odebráno)*

memory (klávesa **M**)

- ***048K*** (0) – klasická čtyřicetosmička
- ***128K*** (1) – klasická stodvacetosmička
- ***256K*** (2) – Pentagon 256 (resp. bit 7 portu 32765)
- ***512K*** (3) – Pentagon 512 (resp. bity 7,6 portu 32765)
- ***1024*** (4) – Pentagon 1024 (resp. bity 7,6,5 portu 32765)

Uvědomte si, že MFC dovoluje nastavit větší paměť než třeba ve skutečnosti máte, následkem jsou pak samozřejmě chybně zkopírovaná data! Z tohoto důvodu nelze tuto volbu nastavit natvrdo přepoukováním MFC (resp. nastavit si to můžete, ale MFC stejně při startu provede detekci 48K vs. 128K a podle toho setup upraví...).

default (klávesy **CS+SS**) - nastaví všechny volby do doporučeného stavu.

exit (klávesa **ENTER**) – potvrzení voleb a spuštění MFC.

Číslo v závorce, které bylo uvedeno za každou volbou, je hodnota pro případné *POKE*.

Někteří z Vás jistě budou chtít, aby MFC okamžitě po svém spuštění obsahoval nějakou konkrétní (Vaši oblíbenou) utilitu, šetřič nebo PC boot sektor. Ne každému se bude líbit, co jsem (si) tam dal já. Totéž platí pro Setup.

Následující programek v Basicu by Vám to měl ulehčit:

```
10 CLEAR 27000

20 LOAD * "mfc???" CODE           ;otazníky značí pořadové číslo MFC
30 LOAD * "u@jmeno" CODE 55296    ;utilita (max 2816 bytes)
40 LOAD * "s@jmeno" CODE 58112    ;šetřič (max 512 bytes)
50 LOAD * "b@jmeno" CODE 58624    ;MS-DOS boot (přesně 512 bytes)

60 POKE 38429,201                 ;changes (0 nebo 201)
61 POKE 38430,0                   ;format (0 nebo 201)
62 POKE 38431,0                   ;load-copy (0 nebo 201)
63 POKE 38432,0                   ;crc (0 nebo 201)
64 POKE 38433,0                   ;hotkeys (0 nebo 201)
65 POKE 38434,0                   ;memory (bez účinku!)
66 POKE 38435,0                   ;zatím nepoužito
67 POKE 38436,0                   ;zatím nepoužito
68 POKE 38437,0                   ;utilita je přítomna v MFC
                                  ;viz. řádek 30 (0-ne, 1-ano)

99 SAVE * "mymfc" CODE 29800,29340

RUN
```

Vidíte, že vytvoření "vlastního" MFC je velice jednoduché a já věřím, že to zvládne každý z Vás levou zadní. A komu by se to nechtělo přepisovat, tak může použít program SET_MFC, se kterým je MFC dodáván…

7. Po spuštění MFC

Hned po spuštění se zobrazí tabulka (menu) se všemi možnými mechanikami a Vy musíte určit, se kterou chcete pracovat. Provedete to klávesou, která odpovídá jménu mechaniky (tj. klávesa A, B, C, D). Samozřejmě lze zvolit pouze mechaniku, která je fyzicky (příp. virtuálně :) připojena, což signalizuje bílá barva písma. Poté se MFC pokusí načíst BOOT, FAT a adresář (DIR) z vložené diskety. Pokud se nějaký sektor nepodaří načíst (kromě BOOTu – ten musí být vždy OK), uvidíte to vpravo dole na obrazovce (info řádek) – číslo nepřečteného sektoru bude blikat. **Zelená** barva znamená **čtení**, **červená zápis** a bílá vše OK. To, že se nějaký sektor patřící FAT nebo DIR nepřečte (CRC error, Sector not found atd. – ty chyby se ovšem nehlásí!!!), ještě **neznamená**, že data z celé diskety jsou ztracena! O tom rozhodne až ScanDisk!

Hned po přečtení BOOTu se vyhodnocuje o jakou disketu jde (MDOS, MS-DOS, či úplně něco jiného). Jak už jsem napsal, tak 5,25" PC disketa není v MFC podporována! Je-li vše v pořádku, nastaví se potřebné parametry v diskové SRAM (systémové proměnné) a pokračuje se v načítaní ostatních sektorů.

7.1 ShakeDir

Potom se dostává ke slovu funkce ShakeDir. Jde o nové sestavení celého DIRu, kde ale už nebudou smazané soubory (s těmi se předem rozlučte). Já totiž přímo nesnáším ten hnusný (zlo)zvyk MDOSu, že nově zapisovaný soubor nemusí být vždycky na poslední pozici v DIR. Ty smazané soubory jsou stejně na ho*no, protože obnovení smazaného souboru je vždycky věc náhody. Nedá se totiž žádným algoritmem zjistit, ve kterých sektorech na disketě ten smazaný soubor ležel. Dá se to **jen odhadovat** (z čehož plyne, že ruční obnova je vždy spolehlivější).

7.2 ScanDisk

Následuje ScanDisk. Podle mne jde o vůbec nejlepší věc z celého MFC. Žádný jiný kopírovací program s takovou funkcí nikdy nepřišel (ani na PC). Na tomto podprogramu jsem pracoval opravdu velice dlouho a tvrdě a stojí na něm (resp. na jeho výsledcích) v podstatě celý MFC.

ScanDisk nedělá nic jiného než důkladnou prohlídku celé FAT a porovnání jejich údajů s DIR. Divili byste se kolik různých chyb může disketa obsahovat (viz. příloha). Pokud narazí na chybu, automaticky ji opraví. **Upozorňuji, že na disketu se opravy nezapisují** - vše se děje pouze v paměti, disketa je nedotčena!

Je důležité vědět, že **FAT má větší prioritu** než DIR. To znamená, že DIR (hlavně délka souboru) se bude opravovat podle FAT, nikdy ne naopak! Totiž software, který by s jistotou řekl, že ten a ten sektor souboru chybí (proto se ke stezce přidá), nebo že souboru nepatří (proto se ze stezky odstraní), napsat nejde. Vždy by to byly jen a jen odhady. Takto, s FAT jako prioritní, se nic věštit nemusí, prostě se vše podle ní zorganizuje.

U PC diskety se navíc porovnává shoda primární FAT se sekundární. Pokud nesouhlasí, musíte zvolit, která se bude používat.

Bylo-li něco na disketě opraveno, uslyšíte po ukončení ScanDisku "errorový beep" a dojde k automatické aktivaci "ochrany diskety proti jakémukoliv pokusu o zápis na ni" (viz. klávesa SS+X).

ScanDisk také při analýze diskety vytváří v paměti **mapu FAT** (zatím není uživateli přístupná). Každému sektoru je v paměti vyhrazen 1 byte. Jeho hodnota může být:

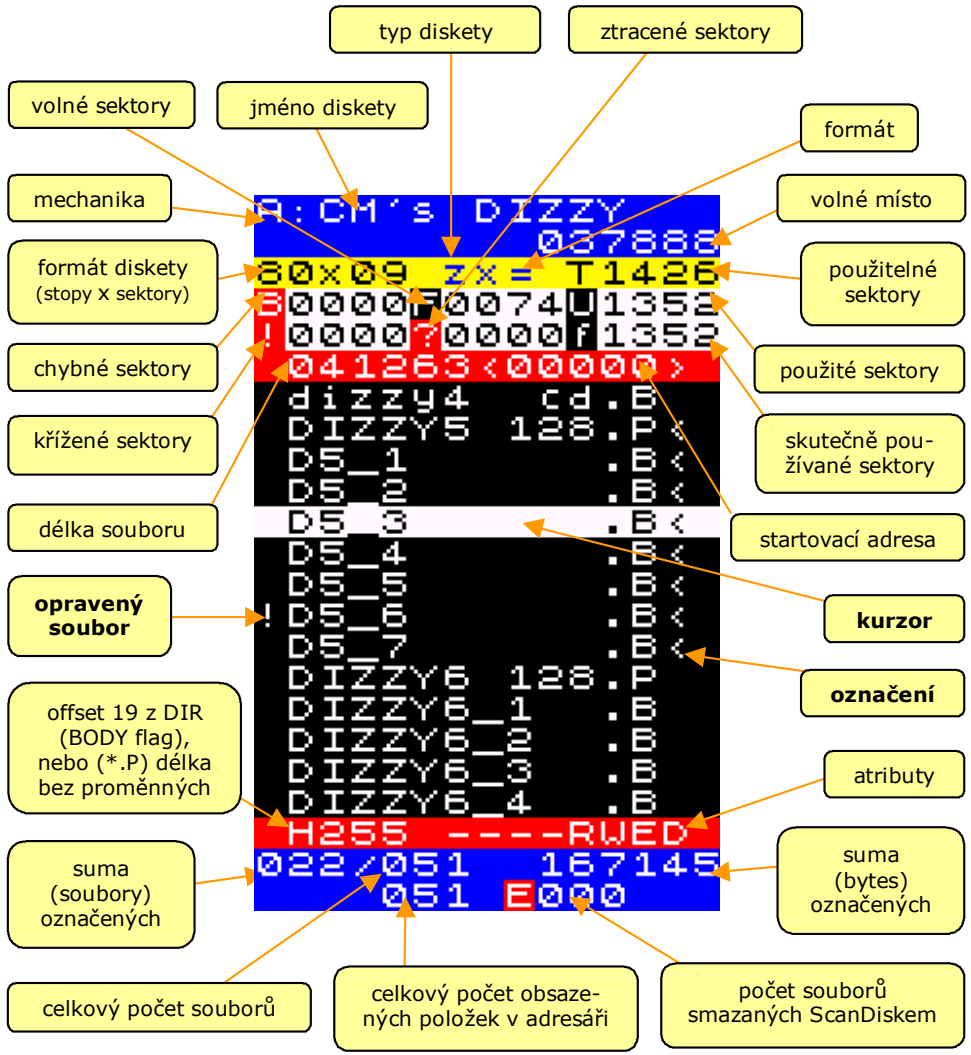
```
0 volný nebo rezervovaný sektor
B špatný sektor
f sektor používaný souborem (pouze 1x se na něj něco odkazovalo)
```

```
! křížený sektor (vícekrát se na něj odkazovalo)
? ztracený sektor (je použitý, ale nic se na něj neodkazovalo)
```

Ztracené a křížené sektory nemají na disketě co dělat. Pokud tam jsou, je velká pravděpodobnost, že některé soubory budou poškozeny. Ztracené sektory budou nejspíš nějakému souboru chybět *(nebo jde o virus, že Tritole!)*. U křížených je to ještě horší, protože stejný sektor používá více souborů. Ztracené sektory se většinou objeví kvůli opravě, kterou provede ScanDisk (např. když je nucen nějaký soubor zkrátit apod.). Ať už se tam objeví díky ScanDisku nebo ne, rozhodně byste s tím měli něco dělat (překopírovat co jde na jinou disketu a poškozenou zformátovat, nebo se pokusit poškozené soubory ručně opravit – dobře si rozmyslete, jestli to vůbec stojí za tu obrovskou námahu). Pozor ale, že některé firmy si pomocí ztraceného sektoru hlídaly, jestli se jedná o jejich originální disketu (např. Ultrasoft).

Možná víte, že na PC je program Scandisk součástí Windows (ten můj dělá téměř totéž). Dost jsem se nasmál, když jsem mu dal prohlédnout jednu plnou disketu – trvalo mu to věčnost (na AMD 550MHz). Jde o dokonalou ukázku rozdílu v programování na PC a na ZXS… Nenávidím lidi, co dělají program stylem "hlavně, že to funguje" - programátor musí myslet hlavou! Nenávidím Billa! Jeho krach a pád veškerých jeho "standardů" bude nejšťastnějším dnem v mém životě.

8. Hlavní okno (výpisy)

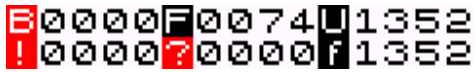


Doufám, že taky nepatříte k těm expertům, co jsou nadšeni z miniaturních fontů typu 3x6 a co uznávají pouze 2 barvy, a že se Vám okno MFC líbí. Já teda písmenka na obrazovce luštit nehodlám (na to fakt nemám nervy ani čas...) a všechny informace chci mít okamžitě na očích a to pěkně (barevně) rozlišené. Samozřejmě, snažil jsem se to nepřehnat (i když plno lidí je jiného názoru...).

Pouze několik věcí vysvětlím podrobněji, ostatní by Vám mělo být jasné už z obrázku na předcházející straně:

Tohle ukazuje formát diskety (80x09). Za ním může být:
pc= PC disk (= jako oboustranný formát)
zx= D40/D80 disk (= jako oboustranný formát)
zx- D40/D80 disk (- jako jednostranný formát)
zx# 5,25"; 16mi disketa v 8mi mechanice (viz. strany 71-76)

T???? značí celkový počet použitelných (pro soubory) sektorů/clusterů. Myšleno tak, že kdyby byla celá disketa prázdná, tak tolik by jich bylo volných pro data. **T** jako Total.



B???? **F**???? **U**???? špatné, volné, použité sektory/clustery
!???? **?**???? **f**???? křížené, ztracené, skutečně využité

B jako Bad, **F** Free, **U** Used, **f** files (jak vidíte, logika v tom je :-).

Pokud je vše na disketě v pořádku, tak by se **U** (used) mělo rovnat **f** (files). Kdyby se totiž nerovnalo, tak existují ztracené nebo křížené soubory (viz. kapitola o ScanDisku).

Někomu by ještě mohlo být nejasné v čem se vlastně liší **U** (used) a **f** (files). Tak tedy, Used se zjišťuje pouze z FATky

(obsazené položky ve FAT), kdežto files z DIR i FAT (sektory, které jsou soubory **skutečně využívány**).

BODY flag (tady H255) se zobrazuje u všech souborů, které mají jinou příponu než P (Basic program). U Basic programů se místo toho zobrazí údaj o délce programu bez proměnných. BODY flag má smysl pozorovat pouze u bezhlavičkových souborů, které byly převedeny z kazety (soubory BODY.H). Pozor! Soubor nemusí mít příponu H a přesto může být bezhlavičkový (to je tak, když si někdo soubor převede z kazety pomocí MASTERcopy, pak jej přejmenuje, změní příponu, a je to…). Pokud se nejedná o bezhlavičkový soubor, bude vypsáno H000 (takový má totiž vždycky na offsetu 19 v DIR nulu).

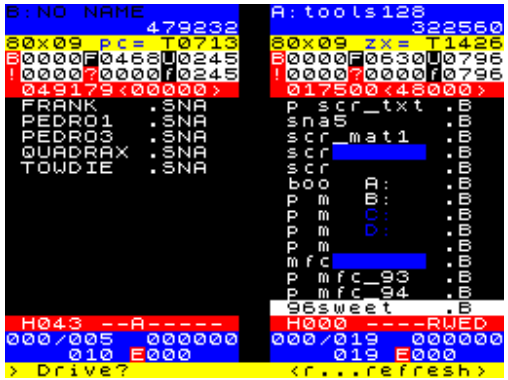
Pod BODY flagem je **počet souborů** (zde 022), **které jsou označeny**. Za lomítkem je **celkový počet souborů** na disketě (051). Nepleťte si to s údajem pod tím! Ten dole totiž určuje, kolik je **obsazeno položek v adresáři**!! Teď si asi říkáte, že jsem vůl, vždyť přece co obsazená položka to soubor na disketě… Jo, ale tohle platí u MDOSu a MS-DOSu, ale ten SHIT Windows díky podpoře dlouhých názvů (více než 8 znaků ve jméně souboru) dělá v adresáři pěkný BORDEL!

Poslední údaj (Erased000) doplňuje ScanDisk. Je to počet souborů, které musel smazat (viz. kapitola o ScanDisku). **V žádném případě nejde o počet smazaných souborů na disketě.** "Díry" (smazané soubory) se odstraňují už před aktivací ScanDisku – hláška *Shaking DIR…* !

9. Ovládací klávesy

N – nová disketa

Pokaždé, když v mechanice vyměníte disketu, musíte to MFC sdělit. K tomu slouží klávesa N. Po jejím stištění se zobrazí nabídka se všemi možnými mechanikami (tj. A, B, C, D) a je na Vás, jakou z nich vyberete. V nabídce se však "rozsvítí" pouze ty, které jsou připojené a připravené ke své činnosti (jen tyto MFC dovolí vybrat). Po výběru už následuje očmuchání diskety (viz. kapitola "Po spuštění MFC").



Jistě jste si všimli, že MFC podporuje i mechaniky C a D, které jsou normálně MDOSem zakázané. Ovšem máme taky "nenormální" :) MDOSy se spoustou vychytávek. Sweet Factory například dělá na MDOSu 3 (jádro MDOS 1.0), já zase na MDOSu 2.1 (jádro MDOS 2.0). Jde v nich o podporu IDE zařízení, resp. Zilogového divIDE. Přes NMI menu (tlačítko SNAP) je možné přiřazovat libovolné mechanice (A, B, C, D) virtuální disketu (část sektorů z HDD). Je to podobné jako v emulátoru RS na PC. Sqělé, ne? MFC nemůže jinak než to podporovat.

Kromě kláves A, B, C, D pro výběr mechaniky funguje také:

CS+1 nebo **SPACE** – krok zpět (jako ESC na PC)

Nové MDOSy dovolují odpojovat a připojovat mechaniky přímo za běhu programu, proto jsem přidal také funkci refresh a SNAP:

R (refresh) – znovuotestování stavu všech mechanik
M (menu) nebo **ENTER** – nasimulování stisku tlačítka SNAP

Když MFC detekuje upravený MDOS, který po stisku tlačítka SNAP zobrazuje NMI menu (testuje se adresa #01AC), provede se po stisku klávesy M nebo ENTER "softwarový" SNAP (tj. LD HL,#0067; PUSH HL; JP #0000). Smysl je v tom, že když chcete vyměnit virtuální disketu, stačí stisknout M a zobrazí se Vám NMI menu. Já jsem prostě zhýčkaný uživatel a nechce se mi někde vzadu na kabelu hledat tlačítko SNAP (nehledě na to, že po návratu s NMI menu se skočí na adresu, kterou chci já a tím pádem se automaticky provede i refresh :).

1, 2, SPACE – prohazování oken

Klávesy slouží pro přepínání se mezi okny.

1 – přepnutí se do levého okna
2 – přepnutí se do pravého okna
SPACE – přepnutí se do opačného okna

Pokud nějaké okno není viditelné, znamená to, že jste do něj ještě nenačetli žádnou disketu. V tomto případě by se při pokusu o přístup do něj provedl příkaz N – nová disketa.

Pohyb po souborech, označování

šipka nahoru – o jeden soubor nahoru - Up
šipka dolů – o jeden soubor dolů - Down
šipka vlevo – o celé okno (14 souborů) zpět - PgUP
šipka vpravo – o celé okno (14 souborů) dopředu - PgDown

Enter – označení/odznačení souboru

Pozn.: Pro pohyb po souborech a jejich spouštění funguje také **sinclair joystick** (klávesy 6, 7, 8, 9, 0).

K – označení všech souborů

Příkaz označí všechny soubory, které na disketě jsou (resp. které dovolil vypsat ScanDisk…).

J – odznačení všech souborů

Opak, dojde k odznačení všech souborů.

SS+K – označení souborů (maska)
SS+J – odznačení souborů (maska)

Pro označování/odznačování více souborů podle určitých kritérií slouží maska. **Maska může být tvořena max. 12ti znaky**. Hvězdičkovou a otazníkovou konvenci Vám doufám nemusím vysvětlovat (když tak viz příručka k D80). Na masce jsem si dal opravdu záležet…

Maska **musí obsahovat minimálně jeden znak**, tečku můžete použít pouze jednou nebo vůbec. Jinak Vám to program nesežere (varovný BEEP)!

MDOS má formát souboru 10+1 (10 znaků jméno, 1 znak přípona), jméno může obsahovat malá i velká písmena (např. *MYFILE.P* a *MyFile.P* jsou naprosto rozdílné soubory), přípona je vždy uložena jako velké písmeno.

MS-DOS má 8+3 (8 znaků jméno, 3 znaky přípona), jméno i přípona může obsahovat pouze velká písmena. (Teď by někdo mohl říct "a co jména ve Windows?", no tak aby bylo jasno, ten shit program mi je úplně fuk!).

Vy můžete vždy psát jména a přípony jakými písmeny chcete. Program si totiž hlídá s jakou disketou se právě pracuje a znaky si na velké případně opraví sám…

Vámi zadaná maska se vždy převádí do otazníkové formy, podle které se pak vybírají soubory. Ukážu nejlépe na příkladech:

zadaná maska	***MDOS***	***MS-DOS***
`*.*`	`??????????.?`	`????????.???`
`.` *(pouze tečka)*	`??????????.?`	`????????.???`
`b*.*`	`b?????????.?`	`B???????.???`
`b`	`b?????????.?`	`B???????.???`
`b.`	`b?????????.?`	`B???????.???`
`*.p`	`??????????.P`	`????????.P??`
`.p`	`??????????.P`	`????????.P??`
`.tXT`	`??????????.T`	`????????.TXT`
`12345678901.`	`1234567890.?`	`12345678.???`
`MTs.pS`	`MTs???????.P`	`MTS?????.PS?`
`*mts.*p`	`??????????.?`	`????????.???`
`?mts.?s`	`?mts??????.?`	`?MTS????.?S?`

Z příkladů vidíte, že maska se vyhodnocuje zleva doprava, a že na hvězdičku se můžete klidně vykašlat (stejný výsledek dostanete i bez ní). Udělal jsem to kvůli příjemnější práci. Sporné to je však třeba u MTs.Ps, kde by se správně měl označit pouze jeden soubor s názvem MTs.P (MDOS), příp. MTS.PS (MS-DOS), ale pochybuji, že by to někdo dělal přes masku (jeden soubor je rychlejší označit ručně). Vycházím z toho, že pokud se použije maska, tak se nehledá soubor jeden ale více!

SS+8 – suma označ. v clusterech

Spočítá, kolik by označené soubory zabraly clusterů (tj. položek ve FAT) na PC disketě. Pozor, že nulový soubor u MS-DOSu neobsazuje žádnou položku, proto také nebude započítán.

SS+9 – suma označ. v sektorech

Spočítá, kolik by označené soubory zabraly sektorů (tj. položek ve FAT) na D40/80 disketě. Nulový soubor obsazuje (podle MDOSu) jednu položku, proto bude také započítán.

P – "potáhnutí" souboru dolů
L - "potáhnutí" souboru nahoru

MFC zobrazuje soubory v okně za sebou tak, jak by je vypisoval příkaz CAT. Někdy se může stát, že potřebujete soubory různě popřeházet (pro lepší orientaci při CAT). K tomu slouží právě klávesy P a L, které soubor, na němž stojí kurzor posunou dolů či nahoru.

Pozn.: "Potažením" se samozřejmě mění i DIR.

SS+D - výmaz souborů

Vymazaní souborů musíte vždycky ještě potvrdit. Mazat se budou všechny označené soubory na disketě. Pokud není označený žádný, smaže se ten, na němž stojí kurzor.

Při mazání se nebere ohled na atributy souboru!!!

V – změna jména diskety

Změnit jméno diskety je možné jen u zx diskety. U pc se klávesa ignoruje.

R – změna jména souboru

Na jména (i přípony) u pc diskety je třeba dávat obzvláště **velký pozor**. Pokud byste použili nějaký MS-DOSem zakázaný znak, dočkali byste se na PC pouze hlášky *Soubor nelze přečíst!* MFC proto nepovolené znaky hlídá a nahrazuje je podtržítkem. Jde o tyto: ***mezera " / \ | [ ] : + = . ; , * ? < >***

Dále MS-DOS používá pouze velká písmena, proto MFC automaticky provede i transformaci malých písmen na velká.

S - změna přípony souboru

Přípona bude vždy (jak u MS-DOSu tak MDOSu) převedena na velké písmena. Jinak zde platí totéž, co pro jména souborů.

B – změna startovní adresy

Tady není co dodat. Snad jen, že odeslání prázdného pole se bere jako byste zadali 0 (nulu).

A – změna atributů souboru

Nastavení/zrušení nějakého atributu se provádí klávesou, která danému atributu odpovídá. Můžete ale také použít **SS+Q** pro vymazaní všech nastavených atributů a **SPACE** pro vymazaní všech nastavených a nastavení pouze standardních (pc - A, zx - RWED).

MDOS

	celý název	význam (je-li nastaven)
H	hidden	skrytý soubor (CAT nevypisuje)
S	system	systémový (užito pro "run" od Proximy)
P	protected	chráněný **(používá MFC)**
A	archive	archivační (užito pro "run" od Proximy)
R	readable	soubor je možné číst
W	writeable	soubor je možné přepsat (není "read only")
E	executable	spustitelný soubor
D	deleteable	soubor je dovoleno smazat

Atribut **P** nastaví MFC každému souboru, který zkopírujete z pc diskety.

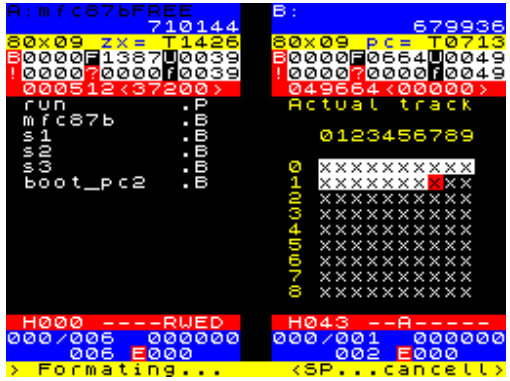
MS-DOS

	celý název	význam (je-li nastaven)
A	archive	archivační
S	systém	systémový soubor
H	hidden	skrytý soubor
R	read only	soubor pouze pro čtení

CS+F – formátování

Formátování je další silnou stránkou MFC. V diskové ROM je totiž chyb až moc, takže jsem celé formátování přepracoval (a to bych nebyl já, kdybych i něco nevylepšil ;).

Možná víte, že formátování se na MDOSu 1.0 a 2.0 výrazně liší (a to nejen vzhledově ale i programově). MFC je však natolik inteligentní, že si verzi Vašeho MDOSu automaticky otestuje a použije správné formátovací podprogramy. Vy ale prakticky nic nepoznáte – na 1.0 i 2.0 se formátuje (vzhledově!!) úplně stejně.



Jelikož je tu teď MFC, nikomu už nedoporučuji používat pro formátování Basic. Těch chyb je tam fakt dost a jsou pěkně hnusné. Mimochodem, MDOS 1.0 má chybu i v číslovaní stran diskety a index pulzu (u 2.0 neplatí – tam tyto chyby nejsou, protože formátování je díky novému řadiči celé předěláno), takže kdybyste takto naformátovanou disketu chtěli použít na PC (v MS-DOSu nebo třeba emulátoru Real Spectrum), tak neuspějete a budou se Vám hlásit různé errory. MFC má však pro MDOS 1.0 připravenou speciální formátovací rutinu, se kterou bude každá disketa

naformátována podle platných standardů a bude tedy plně čitelná i na PC (Sweet Factory opět zabodoval…).

Před formátováním musíte ještě zadat počet stop (tracks) a sektorů (sectors). **Vřele Vám doporučuji používat standardní formáty (tj. 80x09 u D80 a 40x09 u D40). Méně klidně taky**, ale nikdy ne více! A dávejte pozor na formáty s polovičním počtem stop než zvládne mechanika (viz. příloha).

Formátování (i verify) můžete kdykoli přerušit stiskem SPACE. Status formátování (na které stopě právě jste) můžete sledovat na obrazovce. Verifikace se nevypisuje (z důvodů rychlosti programu a úspory kódu). Pouze kdyby se narazilo na špatný sektor, BORDER zčervená.

Po skončení formátování se program zeptá (ale pouze pokud jste formátovali v D80 80x09), jestli má vytvořit PC (MS-DOS) nebo ZX (MDOS) disketu.

Pozn.: Trošku jsem vylepšil i BOOT diskety, kterou naformátuje MFC (viz. příloha). Mimo jiné jsem snížil pravděpodobnost výskytu 2 stejných BOOTů na 2 různých disketách (1:16777216 je si myslím daleko lepší než 1:65536).

Pozn.2: Jednostranný formát MFC nedovolí (disketovka má 2 hlavy, takže je taky koukejte využívat!).

Pozn.3: Automatická detekce verze MDOSu (kvůli zvolení patřičných formátovacích podprogramů) není zrovna nejlepší řešení a u nových MDOSů postavených na jádru 1.0 (např. ten Sweetův pro spolupráci s HDD) způsobuje spíše nepříjemnosti. Pokud totiž MFC vyhodnotí, že máte MDOS verze 1 (hodnota 129 na adrese 9105), nepoužije pro naformátování diskety standardní formátovací bod (adresa 9176), ale svou vlastní rutinu (důvody jsou popsány výše), která je ovšem dost "low level" a šahá přímo na řadič. No a to je průser… V setupu tedy máte možnost si tohle nastavit.

SS+F – rychlé formátování

Zde nejde o formátování v pravém slova smyslu. Rychlé formátování spočívá v tom, že se vytvoří pouze nový BOOT, FAT a DIR a vše se to uloží na disketu.

Jméno diskety bude v novém BOOTu stejné jako v tom starém (náhodné bytes však vygenerovány nově jsou).

Může ale nastat jedna vyjímka. Když budete chtít pomocí SS+F naformátovat disketu s vadnými sektory (MFC si to zjišťuje z FAT), bude se provádět i VERIFY (špatné sektory se totiž mohly rozšířit…).

Pozn.: Jednostranný formát je při rychlém formátování akceptován a dovolen.

X – zápis BOOT, FAT, DIR

Základní filozofie MFC je šetrnost k mechanikám. Proto se s disketou "točí" co nejméně. Když nějaký soubor, přejmenujete, změníte příponu, startovní adresu, "posunete" atd., **na disketu se změny hned nezapíší!** Možná, že to mnohým z Vás bude působit problémy ale radím Vám, zvykněte si (i když, setup řeší vše...).

Člověk je ale přece jen tvor zapomětlivý, takže MFC hlídá před stiskem kritických kláves, jestli je třeba BOOT, FAT a DIR na disketu zapsat. Kritické klávesy jsou takové, jejichž volba zničí (vymaže) obsah BOOT, FAT a DIR v paměti (např. CS+Q, SS+R, N, apod.). **Pozor!** U formátování se to nehlídá Můžete klidně na disketě změnit co chcete, a jestli pak dáte SS+F nebo CS+F, disketa se v klidu naformátuje bez jakéhokoliv upozornění (je totiž zbytečné zapisovat na ni těch 14ti sektorů, když ji stejně v zápětí zformátujete). Totéž platí pro kopírování (není třeba sektory zapisovat, neboť u cílové diskety se stejně po skončení kopírování zapíší).

SS+X – deaktivace ochrany

Pokud ScanDisk opraví nějaké soubory (tj. provede zápis do adresáře nebo FATky), automaticky se aktivuje ochrana, která znemožní na takovou disketu cokoliv uložit a z diskety tak bude možné pouze číst! SS+X tuto ochranu deaktivuje. Dobře si však rozmyslete, co děláte než SS+X použijete! Měli byste 100% vědět, co ScanDisk opravil a plně s těmito změnami souhlasit.

SS+U – nahrání utility/šetřiče/bootu

Utilitám a šetřičům je věnována samostatná kapitola, takže teď se moc rozepisovat nebudu. **Utilitu nebo šetřič obrazovky lze nahrát pouze ze ZX** (tj. MDOS) **diskety** a to ještě za předpokladu, že soubor je správně pojmenován (začíná znaky ***u@*** nebo ***s@***). Jinak se nestane vůbec nic.

Šetřič se spouští okamžitě po nahraní (člověk je zvědavý hned a nechce se mu čekat než naskočí sám :), utilitu si spouští uživatel sám (viz. následující volba).

Přidal jsem i **možnost nahrání Vašeho vlastního** PC **BOOT sektoru** (MFC jej použije při vytváření PC diskety). Jeho jméno na disketě musí začínat znaky ***b@*** a lze jej opět nahrát pouze ze zx diskety.

U, 3 – spuštění utility

Po stlačení U (jako Utility) nebo 3 (jako F3 na PC) se **MFC přepne do "Utility módu" a veškeré řízení předá utilitě** (tj. uživatelskému programu), která je v paměti.

Utilitu se Vám nepodaří spustit, pokud není nahrána v paměti (uvědomte si, že při kopírování souborů dochází k jejímu smazání!!) nebo nemáte vloženu správnou disketu (ověřuje se boot).

SS+R – spuštění souboru

U zx (**MDOS**) diskety jdou spouštět pouze soubory typu **Program** (*.P), **Bytes** (*.B) a **Snapshot** (*.S). Základní algoritmus spouštění jsem převzal z TOOLSe, neboť je opravdu geniální. Jde v podstatě o provedení příkazu NEW a "naťukání" potřebného LOAD příkazu do editačního řádku.

Pokud jde o Bytes, musíte zadat 3 adresy:

CLEAR - pro nastavení RAMTOPu, akceptuje se 0-65535. Jestli odešlete prázdný řádek nebo 0 (nulu), tak se hodnota upraví na 65367 (jako po RESETu). Pozor! Neprovádí se kontrola hodnoty (můžete klidně zadat i CLEAR 10, pak ale čekejte chybové hlášení BASICu…).

CODE - na jakou adresu se má blok nahrát, akceptuje se 0-65535 (prázdný řádek znamená 0).

RANDOMIZE USR - odkud se má blok spustit, akceptuje se 0-65535. Prázdný řádek nebo 0 znamená, že se program spouštět nebude (resp. provede se RANDOMIZE USR #1700, kde je instrukce RET).

Přímo z PC (**MS-DOS**) diskety lze spouštět snapy s příponou SNA (pouze 48K) a MFC.

.SNA je SNAP, který používají emulátory na PC. MFC umí spouštět jen 48K verzi takového snapu. Formát **.MFC** (popis viz. příloha) jsem zavedl já. Jde také o SNAP, ale na rozdíl od všech ostatních "hnusů" má jednoduchou a jasnou strukturu.

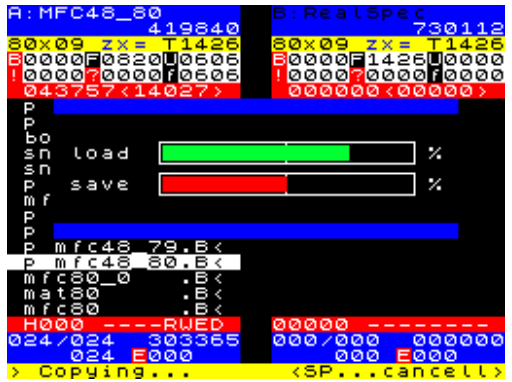
Když budete spouštět 48K snap na počítači se 128K (a více), máte možnost ještě zakázat stránkování, tj. zadat příkaz OUT 32765,48). 128K snap na "obyčejné" čtyřicetiosmičce ze zcela logických důvodů nespustíte nikdy :D.

Nic jiného MFC spouštět neumí. Samozřejmě vím, že hodně lidí (i já) potřebuje spouštět taky .Z80 a 128K verzi .SNA. Proto jsem na PC v C++ vytvořil ***MFC snap Convertor***.

C – kopírování (1:1)

Tak konečně se dostávám k tomu nejzajímavějšímu a nejlepšímu z celého MFC. Jde o zbrusu nový a zcela jedinečný kopírovací systém, který jsem vypiloval do nejmenších detailů.

Kopírovat je možno libovolně: PC->PC, ZX->ZX, ZX->PC, PC->ZX.



Při kopírovaní se poslední sektor každého souboru **"očišťuje"**. Vysvětlím detailněji. Jak víte, tak z diskety se načítá po sektorech (=512 bytes), takže má-li soubor např. 6912 bytes, tak zabere 13,5 sektorů. Načíst a uložit se však musí 14 s tím, že v tom 14ém je i několik bytes navíc, co už souboru nepatří (v našem příkladě 256). Tyto bytes navíc MFC "očišťuje" = nuluje. To proto, aby se pak při editaci diskety v sektorech dobře orientovalo.

Pokud máte v Setupu správně nastaveny patřičné volby, MFC kopíruje i soubory, které by normálně nešly - dokáže ignorovat chyby *Sector not found* nebo *CRC error*. U těchto chyb jsem zjistil opravdu zajímavé věci:

***Sector not found** – tato chyba nastává i v případě, kdy sektor existuje a je v naprostém pořádku (jak jsem ale zjistil později, tohle platí pouze pro MDOS 2.0). Pokud disketu zaplníte až do úplného konce, tak při formátu 80x9 chyba zaručeně nastane u čtení posledních (logických) sektorů diskety. Běžnými kopírovacími programy nebo příkazem SAVE je tedy nemožné na MDOSu 2.0 disketu 100% zaplnit (0 bytes volných). MFC to však dokáže a se Sector not found problémy nemá. Při výskytu chyby stačí jen poslat hlavu "domů" a operaci zopakovat. Když sektor skutečně existuje, tato finta zajistí jeho nalezení a načtení.*

CRC error - *při ukládaní dat se chyba vůbec neprojeví! Máte-li poškozenou disketu a budete na ni ukládat data, neohlásí se jediná chyba. CRC error se projeví až při čtení těch dat. Pěkně hnusné co? Slabou útěchou je fakt, že CRC error, i když nastane, tak v paměti něco načteno bude (chybně ale bude). Proto je možné z těch dat ještě něco smysluplného dostat. MFC má na tohle speciální volbu v setupu, příp. utilitu u@edit. Jen tak pro úplnost, u CRC error je zbytečné hlavu domů posílat a zkoušet to znovu. Je lepší použít registr E před vstupem do DREAD/DWRITE. Ten udává počet pokusů než bude CRC chyba ohlášena (MFC tam má 3, tj. 3 pokusy).*

Když při LOADu (**načítaní**) dojde k chybě *Sector not found,* neúspěšný sektor (ne soubor!) se přeskočí, v paměti jeho místo vyplní invertovaný otazník (tj. "?"+128), BORDER zčervená a pokračuje se dále. Pro *CRC error* platí totéž, ale místo invertovaných otazníků je možno ponechat i to, co se skutečně z diskety načetlo (viz. Setup). Na cílové disketě budou potom všechny neúspěšně zkopírované soubory označeny ScanDiskem - vykřičníkem (neboť v adresáři je uložena nulová délka, ale ve FAT je uložená celá cesta). Věřte, že smysl to má, protože někdy je třeba z poškozené diskety dostat byť jen útržky souboru. **POZOR** ale, že nesmíte takto upravený (ScanDiskem) adresář uložit. Přepsali byste totiž tu nulovou délku za správnou a vykřičníky by zmizely (i když, chybné soubory poznáte i tak - nebudou mít nastaven žádný atribut).

Když dojde k chybě při SAVE (**ukládání**), chyba je ohlášena ihned a celé kopírování je předčasně ukončeno.

Nezapomeňte ale, že vše záleží na Vašem nastavení v Setupu! I při LOADu mohou být chyby okamžitě hlášeny a kopírovaní předčasně ukončeno...

Možnost R=Retry je umožněna jen u *Drive is not ready* (jinde ani nemá smysl). Po této chybě se po stisku P nebo R bude ověřovat i správnost vložené diskety.

Při kopírování na 2 mechanikách se ověřuje, **zda je vložena správná disketa POUZE na začátku** (před prvním čtením

a před prvním zápisem). **Při kopírování na jedné** to neplatí (tam **se ověřuje pokaždé**).

Kontrola správnosti vložených disket se ověřuje pomocí BOOTu. **POZOR!** U ZX diskety se kontroluje od offsetu 202 po 511 (od náhodných bajtů dále). Máte-li tedy 2 diskety se stejným náhodným číslem, pak bude tato kontrola s velkou pravděpodobností neúčinná! Celých 512 bytes se neporovnává, protože jsem chtěl, aby bylo možné si změnit jméno diskety a pak na ni kopírovat aniž by před tím musel být uložen BOOT. U PC disket se porovnává celých 512 bytes.

Pozn.: Kopírují se pouze označené soubory (když není označen žádný, tak se vezme ten, na němž stojí kurzor).

Pozn.2: Kopírování je možné mezerníkem (SPACE) předčasně ukončit.

Pozn.3: Na ukazateli procent můžete sledovat, kolik už je načteno/uloženo a kolik ještě zbývá. A narozdíl od ukazatele ve Windows, ten můj si z uživatele srandu nedělá!

SS+C – kopírování (Export/Import)

Jedním z důvodů, proč jsem vůbec začal na MFC dělat, byl fakt, že si chci soubory z D80 ukládat na PC (životnost disket totiž není (bohužel) věčná). Na PC je HDD, CD, internet atd., což zaručuje celkem neomezenou životnost souborů. Potřeboval jsem tedy vymyslet nějaký archivační formát, do kterého bych si soubory převedl. Důležitou podmínkou ale bylo, že až budu soubor natahovat zpět na MDOS disketu, musím být schopen obnovit všechny důležité informace o souboru (atributy, příponu atd.) i o disketě, na které ležel.

EIM používal formát .000, který je ale vhodný pouze pro soubory z kazeťáku. Nedokáže totiž uchovat všechny informace z DIR a se souborem delším než 65535 jsou problémy. Jo a kdyby teď někdo

namítal, že je tu ještě emulátor Real Spectrum a soubor .d80, tak věřte, že tohle má určité mouchy (d80 je velký soubor, ze kterého si jen tak jednoduše jednotlivé soubory nevytáhnete, a další problém je v tom, že RS neumí pracovat s nestandardním formátem diskety).

Můj nový formát má **příponu .D_0** (popis viz. příloha) a má vždy o 1024 bytes (tj. o 2 sektory) větší délku než archivovaný soubor. Vím, že mnohým se to nebude vůbec líbit, ale věřte, že žádný argument ostatních mě nezlomil a výhody tu jsou (velice jednoduchá práce s takovým souborem jak na PC tak na ZXS). Nebudu Vám zde popisovat, proč jsem to udělal tak a ne jinak, berte to prostě jako fakt. Buď se s tím smíříte nebo D_0 nebudete používat. Vaše věc.

Převod z/do .D_0 funguje jen tehdy, když je v jednom okně MS-DOS disketa a ve druhém MDOS disketa. Jinak ne! Jestli se jedná o Import či Export souborů, si MFC rozpozná sám (záleží, na kterém okně stisknete SS+C).

U exportu/importu platí naprosto stejné zásady jako u kopírování 1:1, tj. "očišťování" souborů, ověřování správnosti disket, přeskakování chybných sektorů. Při importu (.D_0 -> zx disk) si ale dejte pozor na to, aby soubory, které chcete přenášet, byly skutečně ve formátu .D_0 a určeny pro MDOS (jinak chyba a násilný konec…).

XOR dat MFC u jednotlivých soborů neprovádí (kvůli zachování co nejvyšší rychlosti kopírování).

Pozn.: Na PC jsem v C++ vytvořil "user friendly" konvertor, který Vám pomůže při převodech (konverzích) .000 .D_0 a .TAP. Jmenuje se ***MFC d_0 Convertor***.

SS+Z – XOR souboru

Volba slouží na XORování souboru, na kterém stojí kurzor. Kdybyste přemýšleli k čemu je to vlastně dobré, tak věřte, že si tím můžete porovnávat soubory (hodí se, když chcete mít jistotu, že právě zkopírovaný soubor je bez CRC chyb a identický s originálem…).

SS+I – info o systému

Zde můžete zjistit verzi Vašeho MDOSu, připojené mechaniky a typ počítače (48/128).

U MDOSu 1.0 se také zjišťuje, zda je opravena chyba při kopírování příkazem MOVE (ono slavné LD (49152),BC na adrese #1AAB). Pokud to tam není (tzn. chyba je opravena), zobrazí se za verzí MDOSu znak +.

SS+Q – Quit

Ukončení MFC a **reset** počítače. Dříve byla tato volba pod klávesou CS+Q, ale lepší (resp. bezpečnější) je to pod SS+Q *(kolikrát se mi stalo, že při CS+Q jsem si MFC ukončil nechtěně…).* Odezva na reset (tj. čas od stisknutí SS+Q do "zčervenání" obrazovky) může trvat až několik sekund. Je to dáno tím, že MFC si po sobě v paměti čistí veškerý bordel (paměť, která byla nastavena v setupu).

10. Utility

Už na začátku psaní MFC mi bylo jasné, že paměť na ZXS není nafukovací, a že dříve nebo později zjistím, že víc funkcí už do programu prostě nedostanu. S tím jsem se ovšem nehodlal smířit… …**a proto, jsem MFC napsal tak, že on sám je v podstatě jen jakési vychytané jádro, které umí pouze základní operace s disketou a na tomto jádru může kdokoliv cokoliv** (co MFC neumí) **postavit, aniž by bylo nutné MFC nějak změnit** (vydat novou verzi)! Ptáte se jak? No přece pomocí utilit!

Co to vlastně utilita je? Utilita je kus programu (kódu), který se přihraje do hotového programu (aplikace) a rozšíří tak jeho funkce. S utilitami se můžete setkat např. v Desktopu *(tam jsem Vás o jejich síle doufám dostatečně přesvědčil díky utilitě+ DTPmenu)*. V MFC se jimi bude moci realizovat hlavně nadstandardní práce se soubory (konverze, edit atd.) a to aniž by v jejich kódu musela být jediná I/O rutina. Dobré ne? Udělal jsem to podle vzoru Pascalu, C++ aj., kde se k souborům přistupuje přes OPEN FILE, READ BYTE, WRITE BYTE, CLOSE FILE. Výhoda je jasná, programátor se vůbec nestará o spodní I/O rutiny, ale pracuje jen s obsahem (byty) souboru, který mu je postupně naservírován doslova na stříbrném podnose (MFC vystupuje jako číšník a utilita jako host, který je číšníkem obskakován :). Funguje samozřejmě i zpětná vazba, tzn., že utilita předá data MFC a ten se postará o jejich zápis na disketu do nového souboru.

S utilitami je však spojeno jedno veliké nebezpečí (zvláště v MFC, kde se pracuje se soubory…). Nějaký zmetek programátor může napsat jako utilitu (nebo i šetřič) virus. Chudák uživatel se pak může dočkat čehokoliv - od neškodných blbinek až po nečekaně a bleskově zformátovanou disketu. Abych tomu zabránil, rozhodl jsem se, že **na svém webu budu mít zveřejněny všechny utility a šetřiče, které jsou bezpečné**, tzn. mám k nim zdrojový text a otestoval jsem je. Doporučuji tam tedy v případě pochybností nakouknout…

10.1 Omezení utilit

- ***jméno utility na disketě***

MFC utilitu rozeznává podle jména, pod kterým je uložena na disketě (délku si netestuje). Jméno musí vyhovovat masce **u@*.*** (logika v tom je – **u** jako utility a **@** je převzat z Desktopu).

- ***možnost práce pouze na jedné disketě***

Bohužel. Nejde číst soubory např. z mechaniky A a zapisovat je na B. Nelze ani použít 2 různé diskety na stejné mechanice (tj., že z jedné diskety by se četlo a na druhou se zapisovalo). Je možná pouze jedna (konkrétní) disketa a mechanika.

- ***zápis dat (U_WBYTE) možný pouze na zx disketě***

Podpora zápisu i na pc disketu by přinesla mnoho komplikací… Čtení (U_RBYTE) však s pc disketou funguje!

- ***omezená délka utility a paměť***

MFC má na utilitu vyhrazeno místo od **adresy 55296 do 58111**. Její **spouštěcí adresa je 55296**. Dočasně (např. pro různé buffery) je však možné použít ještě obrazovku, dále 256 bytes od adresy 58112 (tj. 58112-58367) a sekundární stránky 128K (17,19,20,22,23).

Délka utility by vždy měla být **2816 bytes** (není podmínkou, může být i kratší *(delší samozřejmě ne :)*, ale vřele Vám doporučuji tu konstantní délku dodržovat).

- ***možnost nechtěného výmazu utility z paměti***

Pokud je utilita nahrána v paměti (SS+U), ještě to neznamená, že tam zůstane natrvalo. MFC ji může za určitých okolností (kopírování souborů) přepsat a je třeba ji pak nahrát znovu. Připouštím, že tohle není zrovna příjemné, ale kopírovaní jsem dal vyšší prioritu než utilitám, protože při kopírovaní je těch 2816 bytes cítit…

- ***vše běží pod DROM (ROM D40/80) a IM 1***

10.2 Programování vlastních utilit

38343,ERROR elegantní "spadnutí" celého MFC

IN: ROM D40/80, normální ROM
(je možné volat kdykoliv a odkudkoliv)

OUT: vypnutí mechanik
"odstřelení" celého MFC a návrat do BASICu

Semtam se Vám může tento podprográmek hodit při vývoji a testování Vaší utility…

38346,U_RET návrat z utility a předání řízení MFC

IN: ROM D40/80
B – druh návratu
C – má se obnovit i obrazovka? (možné pouze s B=0)
HL – adresa s textem, který se má vypsat (0=žádný text)

OUT: vypnutí mechaniky, vypsání textu a návrat podle reg. BC

Akceptované hodnoty v registru B:

B=0 návrat do MFC *(požijte pouze tehdy, pokud jste na disketě vůbec nic nezměnili)*.

B=1 nastavení FLAG5 a návrat do MFC *(znamená, že v DIR byly provedeny změny, a aby MFC později příp. upozornil na jejich neuložení).*

B=2 uložení BOOT, FAT, DIR a návrat do MFC *(musíte použít vždy, když na disketě vytvoříte nový soubor).*

B=3…254 zatím nevyužito

B=255 nastavení FLAG0 a návrat do MFC *(znamená, že došlo k chybě; nebude se nic ukládat a v okně MFC se nastaví NOT READY).*

Akceptované hodnoty v registru C:

C=0 obrazovka se nebude obnovovat

C=1…255 refresh obrazovky (znovu se vykreslí obě okna)

Pozn.: Pokud B<>0, tak se refresh provede VŽDY (i když C=0)!

38349,U_BREAD načte sektor z disku (512 bytes na adr 14336)

IN: ROM D40/80
HL - logický sektor

OUT: ROM D40/80 (i po chybě!)
HL – adresa načtení zvětšená o 512 (14336+512)
BC – fyzická stopa a sektor (B-číslo stopy, C-číslo sektoru na ní)
z – sektor nahrán OK, motory mechaniky běží
nz – chyba při čtení sektoru (HL nyní neplatí!)
po chybě je automaticky vypsána chybová hláška
a zastaveny motory mechaniky

U_BREAD a U_RBYTE lze používat současně (každý má jinou oblast pro načtená data).

38420,U_BREAD0 načte sektor z disku

IN: ROM D40/80
BC – fyzická stopa a sektor (B stopa, C sektor)
HL – adresa kam se bude nahrávat

OUT: ROM D40/80 (i po chybě!)
HL – adresa načtení zvětšená o 512
BC – fyzická stopa a sektor (stejné jako na vstupu)
z – sektor nahrán OK, motory mechaniky běží
nz – chyba při čtení sektoru (HL nyní neplatí!)
po chybě je automaticky vypsána chybová hláška
a zastaveny motory mechaniky

38352,U_BWRITE uloží sektor (512 bytes od adr 14336) na disk

IN: ROM D40/80
HL - logický sektor

OUT: ROM D40/80 (i po chybě!)
HL – adresa zvětšená o 512 (14336+512)
BC – fyzická stopa a sektor (B-číslo stopy, C-číslo sektoru na ní)
z – sektor zapsán OK, motory mechaniky běží
nz – chyba při zápisu sektoru (HL nyní neplatí!)
po chybě je automaticky vypsána chybová hláška
a zastaveny motory mechaniky

U_BWRITE **nesmíte** použít, pokud právě aktivně pracujete přes U_WBYTE (oba mají stejnou oblast pro data).

38423,U_BWRIT0 uloží sektor na disk

IN: ROM D40/80
BC – fyzická stopa a sektor (B stopa, C sektor)
HL – adresa odkud se bude ukládat

OUT: ROM D40/80 (i po chybě!)
HL – adresa zvětšená o 512 (tj. HL=HL+512)
BC – fyzická stopa a sektor (stejné jako na vstupu)
z – sektor zapsán OK, motory mechaniky běží
nz – chyba při zápisu sektoru (HL nyní neplatí!)
po chybě je automaticky vypsána chybová hláška
a zastaveny motory mechaniky

38355,U_ROPN otevření souboru ke čtení

IN: ROM D40/80

OUT: ROM D40/80
carry=1 – už není co otevřít
carry=0
IX – ukazatel do DIR (na hlavičku)
DHL – délka souboru
EBC – přípona souboru pc disk
E – přípona souboru zx disk
B – body flag (IX+19) zx disk
C – atribut (IX+20) zx disk
z – zx disk
nz – pc disk

U_ROPN vrací jeden označený soubor za druhým (co zavolání to vyzvednutí jednoho označeného souboru a příprava na jeho čtení).

38396,U_RHOME přetočení souboru (který je čten) na začátek

Podprogram provede znovuotevření právě otevřeného souboru, který je čten. Vrací úplně stejné hodnoty jako předchozí U_ROPN.

38358,U_RBYTE načtení jednoho byte z otevřeného souboru

IN: ROM D40/80

OUT: ROM D40/80 (i po chybě!)
carry=1 a A=0 – konec souboru (vše už bylo načteno a OK)
carry=1 a A=1 – došlo k chybě při čtení z disku
chybová hláška je automaticky vypsána
carry=0
A – načtený byte
HL, DE, BC, IX jsou stejné jako na vstupu (IN)

Čtení souboru je bufferováno. To znamená, že MFC se snaží načíst ze souboru co nejvíce bytes do paměti, mechaniku vypne a registr A plní už jen z paměti. Pokud dojde na konec bufferu, znovu zapne mechaniku a zase načte co může. Takto se pokračuje, dokud nenastane konec souboru. Velikost bufferu je v této verzi MFC 7168 bytes.

Vypínání mechaniky může někdy vadit *(dobré to totiž je pouze pro edit souboru, kdy se čeká na reakci uživatele a je hloupost, aby mechanika svítila…)*, např. když se od uživatele nic nechce a s daty pracuje pouze počítač (třeba XORování souboru). Vypínání lze tedy zakázat (viz. adresa 38394).

Abyste mohli ze souboru číst, musí být otevřen (U_ROPN)!

xxxxx,U_RCLOSE uzavření souboru otevřeného pro čtení

Tento podprogram neexistuje. Uzavírat soubor, který byl otevřen pro čtení (U_ROPN) není třeba. Pokud chcete přestat s aktuálním souborem pracovat a otevřít si další označený soubor v pořadí, stačí zavolat U_ROPN.

38361,U_WOPN otevření souboru pro zápis

IN: ROM D40/80

OUT: ROM D40/80
carry=1 – nelze otevřít (pc disketa, aktivní DIR ochrana,
není místo v DIR nebo ve FAT)
žádné hlášení není vypsáno

carry=0 – vytvoří novou položku v DIR
IX – ukazatel do DIR (na hlavičku) vytvořeného souboru

Pokud soubor nelze otevřít, do DIR ani FAT není zapsáno vůbec nic.

38364,U_WBYTE uložení jednoho byte do otevřeného souboru

IN: ROM D40/80
A – byte pro uložení

OUT: ROM D40/80 (i po chybě!)
carry=1 – došlo k chybě při pokusu o zápis na disk
chybová hláška je automaticky vypsána (kromě chyby, že nebyl nalezen volný sektor ve FAT!)

carry=0 – úspěšně zapsáno
HL, DE, BC, IX jsou stejné jako na vstupu (IN)

Zápis je také bufferován. Velikost bufferu je 512 bytes (od adresy 14336). Při použití U_WBYTE dojde k zakázání vypínání mechaniky (pořád bude svítit – i při U_RBYTE), protože by to zpomalovalo vlastní zápis na disketu.

S každým úspěšně zapsaným sektorem je v paměti aktualizována FATka (zapsána koncová značka) a DIR (délka souboru). Je to dost výhodné, protože kdyby příště nastala chyba, budete mít uloženu alespoň část souboru a to zcela korektně.

Nelze používat U_WBYTE a U_BWRITE současně! Oba totiž používají stejnou oblast pro data (adresu 14336).

38367,U_WCLOSE uzavření souboru otevřeného pro zápis

IN: ROM D40/80

OUT: ROM D40/80 (i po chybě!)
carry=1 – došlo k chybě při pokusu o zápis na disk
chybová hláška je automaticky vypsána (kromě chyby, že nebyl nalezen volný sektor ve FAT!)

carry=0 – úspěšně zapsáno a uzavřeno

U_WCLOSE nesmíte zavolat, jestliže došlo při U_WBYTE k chybě!

38370,U_SHADE přestránkování do ROM D40/80

IN: normální ROM (je možné ale i ROM D40/80)

OUT: ROM D40/80 + povolené přerušení (EI)
AF beze změny

38394,defw U_R9526+1 adresa pro vypínání mechaniky

Pokud budete chtít zakázat vypínání mechaniky při U_RBYTE, musíte provést (nejlépe hned při startu Vaší utility) toto:

```
LD    HL,(38394)
LD    (HL),1                ;to je jednička, ne malé L!
```

Normálně má MFC vypínání mechaniky povoleno (povoluje se při každém spuštění utility).

38373,U_INKEY čekání na stisk klávesy

IN: ROM D40/80

OUT: A – kód stisknuté klávesy
ROM D40/80

38376,U_PMESAG PRINT (tisk hlášení)

IN: ROM D40/80
HL – ukazatel na text (text je ukončen invertovaným znakem)

OUT: vytištění hlášky do editačního řádku
ROM D40/80

38379,U_CHARS úprava jména souboru

IN: HL – ukazatel na text (10 znaků – jméno souboru)
DE – kam se má upravený text uložit

OUT: upraví text do tisknutelné podoby
(0 = mezera, 1-31 AND 128-255 = otazník)
HL=HL+10; DE=DE+10; B=0
alternativní (EXX) BC, DE, HL registry jsou beze změny

38382,U_DEC3H0 NUM (0-255) to ASCII

IN: A – číslo (0-255)
DE – kam se má ASCII ukládat

OUT: převede 8bit číslo do ASCII podoby
DE=DE+3
alternativní (EXX) BC, DE, HL registry jsou beze změny
alternativní (EX AF,AF') AF registr je beze změny

38385,U_DEC5 NUM (0-65535) to ASCII

IN: HL – číslo (0-65535)
DE – kam se má ASCII ukládat

OUT: převede 16bit číslo do ASCII podoby
DE=DE+5
alternativní (EXX) BC, DE, HL registry jsou beze změny
alternativní (EX AF,AF') AF registr je beze změny

38388,U_DOWNHL DOWNHL

IN: HL – adresa mikrořádku na obrazovce

OUT: HL – adresa následujícího mikrořádku
BC, DE jsou beze změny
alternativní (EXX) BC, DE, HL registry jsou beze změny
alternativní (EX AF,AF') AF registr je beze změny

38411,U_LDIRIQ LDIR s "inteligencí"

IN: HL – odkud
DE – kam
BC – kolik

OUT: provede výměnu (HL)<-->(DE)
BC=0
alternativní (EXX) BC, DE, HL registry jsou beze změny
alternativní (EX AF,AF') AF registr je beze změny

38402,U_INPDTXT příprava pro INPUT

IN:

OUT: vymaže paměťovou oblast pro INPUT (30 bytes)
DE – adresa počátku této oblasti
BC = 0; A=0
alternativní (EXX) BC, DE, HL registry jsou beze změny
alternativní (EX AF,AF') AF registr je beze změny

Pokud budete chtít, aby Vám INPUT už přednastavil nějaký text, tak jej musíte na adresu, kterou udá DE, LDIRnout. Maximálně však 30 bytes a v textu **nesmí** být ASCII kód menší než 3.

38391,U_INPUT INPUT

IN: **normální** ROM
B – počet znaků, které může uživatel zadat (**max. 30**)
C – kód pro možnost ALTxxx
(MFC používá 14 = CS+SS; 0 = ALTxxx zakázáno)
HL – rozsah tolerovaných kláves <H, L)
(H=32, L=128 čísla+text; H=48, L=58 pouze čísla)
DE – kam se bude vypisovat - pozice pro AT
(AT x,y; x=D, y=E)

OUT: normální ROM
z = stisk CS+1 (tzn. stornování INPUTu uživatelem)
nz = stisk ENTERu
když při vstupu B=3 a H=48, tak HL – 0…999
když při vstupu B=5 a H=48, tak DHL – 0…99999
jinak HL – ukazatel na zadaný text
B – počet znaků, které uživatel zadal (může být i 0)

38417,U_INPUT0 "edit" INPUT

IN: **normální** ROM
BC, HL – jako U_INPUT
IX – ukazatel na text, který se má vypsat do editačního řádku

OUT: jako U_INPUT, (+provede se i U_PANEL1)

Registr DE na vstupu se nezadává. Program si hodnotu vypočte sám (bude vždy **za** textem, který byl vypsán do editačního řádku).

38405,U_BEEP pípnutí

IN: **normální** ROM

OUT: normální ROM
pípnutí, které používá MFC

38408,U_BEEPER errorové pípnutí

IN: **normální** ROM

OUT: normální ROM
pípnutí, které používá MFC při chybě
AF, HL, DE, BC, IX jsou beze změny

38414,U_PANEL1 edit řádek s MFC info textem

IN: **normální** ROM

OUT: normální ROM

BC, DE, HL, AF registry jsou beze změny

38399,U_REFAT čtení položky z FAT

IN: HL – logický sektor (zx disk) nebo cluster (pc disk)
(o jaký disk se jedná si program zjistí sám)

OUT: DE – obsah položky
HL – adresa položky
BC beze změny
alternativní (EXX) BC, DE, HL registry jsou beze změny

38426,U_DIVIDE přepočet log. sektoru na fyz.

IN: ROM D40/80
HL – logický **sektor**

OUT: ROM D40/80
IX – adresa proměnných aktuální mechaniky
A – aktuální mechanika (0…3 = A…D)
BC – B fyzická stopa, C sektor
E – počet sektorů na stopu (IX+3)
D=0
alternativní (EXX) BC, DE, HL registry jsou beze změny

10.3 Několik rad programátorům

1. Utilita se spouští s přistránkovánou **ROM D40/80**! A pozor, že né každý podprogram lze pod ní volat.

2. **Pokud není** u podprogramů **uvedeno**, že hodnota registru je na výstupu stejná jako na vstupu, **tak** předpokládejte, že hodnoty všech (i EXX) registrů neuvedených ve výstupu **jsou** po provedení podprogramu **změněny.**

3. Registr IY a mód přerušení (IM 1) byste neměli měnit.

4. MFC dokáže v utility módu pracovat **pouze se dvěmi** zcela nezávislými soubory (jeden jako U_RBYTE a druhý jako U_WBYTE). Více než 2 soubory najednou neotevřete.

5. **U_RET, U_INPDTXT, U_INPUT, U_INPUT0** využívají stránku 16, proto, žádný z těchto podprogramů nesmíte volat, pokud ji nemáte přistránkovanou (jste v nějaké sekundární stránce 128ičky).

6. Budete-li dělat utilitu pro práci s DIR (např. hromadná změna adresy, atributu, přípony atd.) pomocí U_ROPN, nevolejte po jejím skončení U_RET s B=2 ale s B=1. Uživatel by si měl ty změny potvrdit (uložit) sám!.

7. Správnost vložené diskety (jestli ji uživatel nevyměnil) je ověřována pouze před vstupem do utlity módu. Tohle však nestačí a ducha(ne)přítomný uživatel může disketu vyměnit později. Při LOADu se nic moc nestane (data přežijí), ale při SAVE ztratí vše (MFC si myslí, že tam je pořád ta původní disketa…). Proto ve Vašem manuálu k utilitě uživatele upozorněte, že **utilita může pracovat pouze s jednou jedinou konkrétní disketou a je zakázáno ji v utility módu vyměnit za jinou.**

8. Vaši utilitu jsem ochoten umístit na své webové stránky (příp. i zkontrolovat či Vám pomoci s jejím naprogramováním).

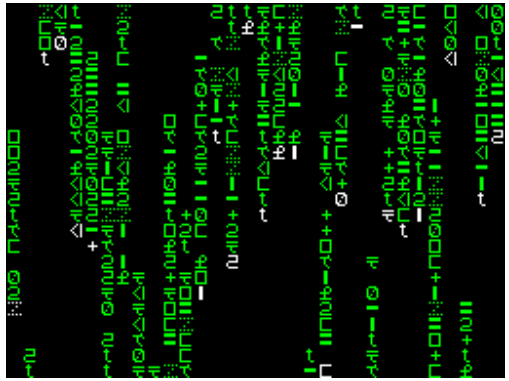
11. Spořiče obrazovky

Screensaver má v MFC **délku max 512 bytes**. **Spouští se od adresy 58112**. Jako buffer může po svém spuštění využít i paměť za ním (58624-65535 = 6912 bytes).

Jméno šetřiče musí na disketě vyhovovat masce **s@*.*** (**s** jako screensaver a **@** jako že jde o utilitu).

Šetřič není umístěn v klasické 48K paměti, ale je v SRAM D40/80 (to aby zbytečně nezabíral místo při kopírování). Pokud má dojít k jeho aktivaci, MFC ho přenese na adresu 58112 a pod klasickou ROM (tedy ne pod DROM jako u utility!) spustí CALLem 58112. Pro návrat z šetřiče pak stačí pouze instrukce RET.

Spořič může být klidně napsán stylem "na jedno použití". Díky jeho uzamčení v SRAM, je počet použití naprosto neomezený Uzamčení znamená, že po skončení šetřiče se jeho 512 bytes **nepřenáší** zpět do SRAM! Nevýhodou je, že šetřič nemůže začít příště tam, kde minule skončil.



Pozor, že u spořičů hrozí stejné **nebezpečí** jako u utilit, takže si jejich bezpečnost raději ověřte na mých www…

12. Známé chyby

MFC obsahuje jednu chybku, na kterou jsem přišel, až už byl program téměř celý hotov. Podle mne je celkem zbytečné s tím teď něco dělat, protože jde spíše o raritu.

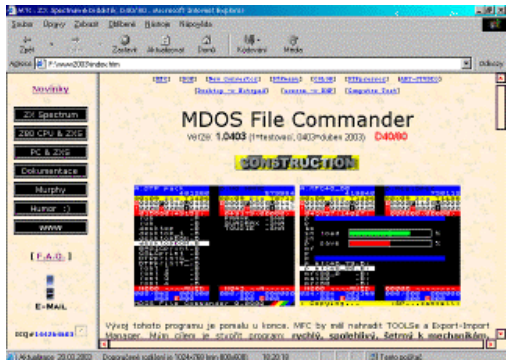
O co že jde? **Problém** je **s nulovým souborem** na **MS-DOS** disketě. V dokumentech, které jsem studoval na samém počátku, bylo napsáno, že nulový soubor je zakončen již v adresáři a to tak, že jako odkaz do FAT se dá #FFF. Tohle jsem si samozřejmě tenkrát raději ověřil, bohužel ale pouze tak, že jsem vzal disketu a ručně (v disk editoru) tam takový nulový soubor vytvořil. Vše bylo OK. Když ale nulový soubor vytvoří MS-DOS od Win98 (nemůžu hovořit za všechny verze, protože na jiných jsem to nezkoušel), hodí jako odkaz do FAT nulu, což je naprostý nesmysl a pro můj ScanDisk je to nepřípustné. Nula by tam měla být pouze tehdy, když jde o přebytečnou položku v adresáři (podpora dlouhých jmen souborů ve Windows).

Dále jsem zjistil, že **čtení originálních disket od Proximy** mi **trvá** strašně **dlouho**. Bohužel nevím proč. Z počátku jsem měl podezření na ujetí sektoru (viz. kapitola "kritické" podprogramy), ale tohle jsem musel vyloučit, protože když jsem zkoušel kopírovat z té kopie co jsem vytvořil, tak už vše bylo OK. Chyba tedy nemůže být na mé straně (v "pomalosti" MFC), ale na straně mechaniky (zřejmě má problémy se čtením těch disket).

Musím Vás také varovat před **použitím tlačítka SNAP**. Pokud MFC snapnete, tak přepíšete SRAM v D40/80, ve které je uložen šetřič a boot sektor PC diskety. Následky si domyslete sami :).

13. www

MDOS File Commander a veškerý "support" (šetřiče, utility, konvertory, dokumentace…) k němu je šířen především díky internetu a mých www stránek. V současné době je jejich adresa http://mts.web.wo.cz nebo http://home.tiscali.cz/mts.zxs (můj email mts.zxs@tiscali.cz).



Samozřejmě to, co platí teď, nemusí platit za několik měsíců. Stát se může lecos a stránky mohou být ze dne na den (s)prostě fuč. Pokud se tohle stane, nepropadejte hned panice *(ať Vás ani nenapadne myslet si, že jsem se ZXS skončil – to se NIKDY nestane!).* Zkuste použít webovské vyhledávače (seznam, google atd.) a magickou formuli "*MTs ZX Spectrum*" nebo "*MDOS File Commander*".

"Černé díry jsou tam, kde Bůh dělil nulou."

anonym z internetu

Příloha č.1 – seznam hlášení MFC

Zpráva (hláška)	Význam
Any file is not .D_0!	Pokoušíte se importovat soubor, který není ve formátu .D_0
Bad sector 0-13 (SHIT disk!)	Verifikace po formátování odhalila, že je špatný nějaký ze sektorů 0 až 13 (BOOT, FAT, DIR). Disketa je tedy úplně na nic. Vyhoďte ji!
Cancelled by user.	Operace byla stornována (přerušena) uživatelem.
Copying... <SP...cancel>	Probíhá kopírování (dá se přerušit stiskem SPACE).
CRC error!	Chyba CRC. Tuto chybu si testuje sama disketová mechanika. Data byla načtena úspěšně, ale nesouhlasí CRC součty. Data jsou tedy zřejmě poškozena.
Create MS-DOS disk? <a,y/n>	Pokud jste formátovali na 80x09, máte možnost vytvořit si po skončení formátování a verifikaci PC DD MS-DOS disketu (a, y = vytvořit MS-DOS disk, cokoliv jiného = MDOS disk).
Data changed in 1,2 <p...OK>	Upozornění, že byly (Vámi) změněny nějaké údaje na disketě a že byste si změny měli uložit. Hláška se zobrazí, když hodláte provést operaci, která zničí BOOT, FAT a DIR, který je v paměti. (p, r = ignorování hlášení a pokračování ve Vámi požadované operaci, cokoliv jiného = zrušení Vámi požadované operace a možnost si změny uložit).
Delete file(s)? Sure?	Žádost o potvrzení výmazu označených souborů. Pokud není žádný označený, vymaže se ten, na nemž stojí kurzor.
Device unavailable!	Mechanika není fyzicky přítomná.
DIR protection disabled.	Potvrzení, že jste deaktivovali ochranu diskety proti zápisu.
DIR protection is active!	Informace, že je aktivní ochrana diskety proti zápisu, kterou zapnul ScanDisk. Na disketu není možné cokoliv uložit, dokud ochranu sami nedeaktivujete.
Directory full!	Překročen maximální počet souborů na disketě (MDOS 128, MS-DOS 112).
Disable DIR protection? Sure?	Dotaz, zda se má opravdu deaktivovat ochrana diskety proti zápisu, kterou zapnul ScanDisk (a, y = souhlas, cokoliv jiného = nesouhlas).

Disk exchanged!	Disketa byla vyměněna. Kontrola BOOTu nesouhlasí. Musíte vložit správnou disketu.
Disk full!	Disketa je plná (data se na ni nevejdou).
Disk is not supported!	Disk není podporován. Stane se Vám to například při DD 5,25" MS-DOS…
Disk is write protected!	Disketa je chráněna proti zápisu.
Drive is not ready!	Mechanika není připravena (nevložili jste disketu apod.)
FAT1<>FAT2 <2...use FAT2>	Primární FAT se nerovná sekundární. Musíte zvolit, kterou má MFC použít (2 = požije sekundární, cokoliv jiného = primární). Doporučuji Vám použít tu FAT, ze které dostane nejvíc údajů ScanDisk…
Format disk? Are you sure?	Žádost o potvrzení naformátování diskety (a, y = souhlas, cokoliv jiného = nesouhlas). Poté musíte ještě zadat počet stop (tracks) a sektorů (sectors).
Format=lost all data! <p...OK>	Varování, že formátování způsobí zničení všech dat na disketě (p, r = souhlas, cokoliv jiného = nesouhlas). Formátování je zvláště nebezpečná volba, takže MFC se raději ptá 2x a pokaždé je třeba potvrdit to jinou klávesou!
Formatting... <SP...cancel>	Probíhá formátování (dá se přerušit stiskem SPACE)
Insert source disk.	Zobrazuje se při kopírování pouze s jednou mechanikou. Říká, že máte vložit **zdrojový** (=ten, **ZE** kterého kopírujete) disk a stisknout libovolnou klávesu.
Insert target disk.	Zobrazuje se při kopírování pouze s jednou mechanikou. Říká, že máte vložit **cílový** (=ten, **NA** který kopírujete) disk a stisknout libovolnou klávesu.
Internal error!	Neidentifikovatelná chyba při práci s disketou/mechanikou.
OUT "32765,48"? <a,y/n>	Před spuštěním 48K PC snapu (.SNA, .MFC) na 128K počítači máte možnost provést příkaz OUT 32765,48 (a, y = provedení příkazu, cokoliv jiného = žádný OUT)
Preparing for LOAD...	Příprava na nahrávání, zobrazuje se po SS+R
R=Retry	Dotaz, zda se má operace provést znovu (p, r = ano, cokoliv jiného = ne).
Reading sector 0 (boot)...	Čtení pouze BOOT sektoru (kvůli ověření správnosti vložené diskety)

`Reading...  0123456`	Zobrazuje se při rychlém formátování (je třeba zjistit typ diskety a přítomnost vadných sektorů)
`Reading...  01234567890123`	Načítaní BOOT, FAT, DIR. Čísla znamenají jednotlivé sektory (zelená = probíhá čtení, bílá = načteno OK, blikání = chyba při čtení)
`ScanDisk...`	Probíhá ScanDisk.
`Sector not found!`	Sektor nenalezen.
`Shaking DIR...`	Provádí se znovusestavení adresáře.
`SP empty :-)  Contact author!`	"Spadnutí" celého MFC. Zapamatujte si prosím, jak se to stalo a kontaktujte mne.
`Utility mode`	MFC předal řízení utilitě
`Verifying...  <SP...cancel>`	Probíhá verifikace (dá se přerušit stiskem SPACE)
`Where is target disk?`	Chcete kopírovat, ale není kam (target disk neexistuje nebo je Not ready).
`Write BOOT,FAT,DIR?  Sure?`	Žádost o potvrzení zápisu (a, y = souhlas, cokoliv jiného = nesouhlas). Aby se Vámi provedené změny (přejmenování, výmaz…) na disketě projevily, je nutno BOOT, FAT a DIR zapsat!
`Writing...  01234567890123`	Zápis BOOT, FAT, DIR. Čísla znamenají jednotlivé sektory (červená = probíhá zápis, bílá = uloženo OK)

Příloha č.2 - podprogramy

Podprogram INPUT

Tento podprogram je volán, pokud je třeba, aby uživatel zadal nějaká data (při přejmenování souboru, změně přípony, startovní adresy atd.). Oproti TOOLSu, EIMu aj. je značně vylepšen.

šipka nahoru – HOME (kurzor před první písmeno)
šipka dolů – END (kurzor za poslední písmeno)
šipka vlevo – LEFT (kurzor o jedno doleva)
šipka vpravo – RIGHT (kurzor o jedno doprava)

CS+0 – DELETE zprava (vymaže znak před kurzorem)
CS+9 – DELETE zleva (vymaže znak za kurzorem)
SS+Q – DELETE všech znaků

CS+2 – velké/malé znaky

ENTER – ukončení zadávání znaků (OK)
CS+1 – ukončení zadávání znaků (storno)

CS+SS – nyní musíte zadat **3** čísla (ASCII kód znaku) nebo
znovu CS+SS pro exit
akceptuje se pouze 032-255 (obdoba ALT *číslo* na PC)

Při vkládání dat se na obrazovku **vypisují pouze tisknutelné znaky** (ASCII 32-127), všechny ostatní se nahradí červeným otazníkem. Ale nebojte, **v paměti je přesně to, co jste pomocí CS+SS zadali**!

MFC si podprogram INPUT může volat pokaždé s jinými parametry. Výsledkem je, že lze zadat pouze čísla (např. při změně startovní adresy programu), nebo je zakázáno CS+SS apod.

Podprogram ScanDisk (možné chyby)

Možné chyby u D80 diskety (MDOS):

Nejdříve však jak by to mělo vypadat:

DIR
odkaz do FAT 15
délka souboru 3077

FAT
položka č. 15 16 17 18 30 31 50
obsah pol. 16 17 18 30 31 50 3589

Z FAT se dá zjistit, že soubor zabírá sektory 15, 16, 17, 18, 30, 31, 50 a jeho délka by měla být 6(sektorů)*512(bytes na sektor)+3589-3584=3077. Vidíte, že vše je tedy OK, protože soubor je ukončen (ve FAT je obsah poslední položky 3584+xx) a délka ve FAT sedí s délkou v DIR.

Co kdyby to ale vypadalo trošku jinak? Třeba takto:

FAT
položka č. 15 16 17 18 30 31 50
obsah pol. 50 17 18 3589 31 16 30

Soubor by se načítal: 15, 50, 30, 31, 16, 17, 18. Zase je vše OK. MDOS soubor v pohodě přečte, ale nikdy by ho takhle neuložil (už teď Vám mohu říci, že MS-DOS ano)! MDOS bude vždy ukládat od nejmenšího sektoru k nejvyššímu! Proto tohle bude považováno za chybu, je nepřípustné odkázat se na nižší sektor z vyššího!

DIR
odkaz do FAT: 0-13 nebo mimo rozsah formátu diskety

oprava: soubor je smazán

DIR
odkaz do FAT: je v povoleném rozsahu (např. 14)

FAT
položka č. 14
obsah pol. 3583 (vadný)

oprava: soubor je smazán

DIR
odkaz do FAT: je v povoleném rozsahu (např. 14)
délka souboru: >0

FAT
položka č. 14
obsah pol. 3072 (nulová délka)

oprava: opraven DIR (zapsána nulová délka)

DIR
odkaz do FAT: je v povoleném rozsahu (např. 14)

FAT
položka č. 14 15
obsah pol. 15 3583(vadný)

oprava: ve FAT zakončen (pol. 14 bude obs. 3584) a do DIR zapsána nová délka (tady 512)

DIR
odkaz do FAT: je v povoleném rozsahu (např. 14)

FAT
položka č. 14 15
obsah pol. 15 3072

oprava: ve FAT zakončen (pol. 15 bude obs. 3584) a do DIR zapsána nová délka (tady 1024)

DIR
odkaz do FAT: je v povoleném rozsahu (např. 14)

FAT
položka č. 14 15
obsah pol. 15 0-13 nebo mimo rozsah formátu (není 3583 nebo 3072 a není ani ukončen(3584+xx))

oprava: ve FAT zakončen (pol. 15 bude obs. 3584) a do DIR zapsána nová délka (tady 1024)

DIR
odkaz do FAT: je v povoleném rozsahu (např. 20)

FAT
položka č. 20
obsah pol. <=20

oprava: ve FAT zakončen (pol. 20 bude obs. 3584) a do DIR zapsána nová délka (tady 512)

---

stezka ve FAT je OK
délka souboru z DIR <> délce souboru ve FAT

oprava: délka v DIR opravena podle FAT

Možné chyby u DD PC diskety (MS-DOS):

DIR
odkaz do FAT: #FFF
délka souboru: >0

oprava: v DIR bude nulová délka

DIR
odkaz do FAT: 1 nebo mimo rozsah formátu diskety (>714) (není ani #FFF)

oprava: odkaz do FAT v DIR = #FFF a nulová délka

DIR
odkaz do FAT: je v povoleném rozsahu (např. 14)

FAT
položka č. 14
obsah pol. 4087(vadný)

oprava: odkaz do FAT v DIR = #FFF a nulová délka

DIR
odkaz do FAT: je v povoleném rozsahu (např. 14)

FAT
položka č. 14 15
obsah pol. 15 4087(vadný)

oprava: ve FAT zakončen (pol. 14 bude obs. #FFF) a do DIR zapsána nová délka (tady 1024)

DIR
odkaz do FAT: je v povoleném rozsahu (např. 14)

FAT
položka č. 14 15
obsah pol. 15 0-1 nebo mimo rozsah formátu (není 4087 a není ani ukončen(#FFF))

oprava: ve FAT zakončen (pol. 15 bude obs. #FFF) a do DIR zapsána nová délka (tady 2048)

---

nekonečná stezka ve FAT, soubor právě dosáhl 714ti clusterů

oprava: odkaz do FAT v DIR = #FFF a nulová délka

---

stezka ve FAT je OK
délka (přepočítána na počet clusterů) souboru z DIR <> počtu clusterů ve FAT

oprava: délka v DIR opravena podle FAT

"Kritické" podprogramy

DIVIDE (LOGFYZ) – přepočet logického sektoru na fyzický
FYZLOG – přepočet fyzické stopy a sektoru na logický sektor

Jak jistě víte, tak ve FAT jsou uloženy **logické** čísla sektorů, ve kterých daný soubor leží. Např. u formátu 80x09 jsou možné pouze sektory 0 až 1439. Řadič disketovky chce ale **fyzický** sektor (tj. která stopa a který sektor na ní).

To, jak rychle je možné načíst všechny sektory z každé stopy, udává **Interleave** (tzn. na kolik "otáček" bude celá stopa přečtena a podle toho jsou také sektory ve stopě očíslovány).

0	1	2	3	4	5	6	7	8

Interleave 1:1 (jedna stopa na jednu otáčku)
(1 otáčka, načte se vše, tj. 0,1,2,3,4,5,6,7,8)

0	5	1	6	2	7	3	8	4

Interleave 1:2 (jedna stopa na dvě otáčky)
(1. otáčka, načte se 0,1,2,3,4; 2. otáčka 5,6,7,8)

MDOS používá INTERLEAVE 1:1 (tzn. načítá celou stopu na jedinou otáčku). Vtip je ale v tom, že aby se všechny sektory na dané stopě skutečně stihly na tu jedinou otáčku přečíst, musí být ďábelsky rychlé 2 podprogramy - výběr položky z FAT a přepočet logického sektoru na fyzický. Vysvětlení: Když načtete jeden sektor (pojmenujme ho např. sektor A), tak se hlava nachází nad začátkem dalšího sektoru (sektor B). Jenže vy ještě nevíte, který budete načítat teď a musíte to nejprve zjistit (výběr položky (sektoru) z FAT a jeho přepočet na fyzický). Na 95% však vždy zjistíte, že načítat se má sektor B. Jenže než tohle zjistíte, hlava už nemusí být na začátku sektoru B, ale někde uprostřed B nebo dokonce úplně jinde (na C, D…). A vy teď musíte čekat celou jednu otáčku než sektor B budete moct načíst. Chápete? KVAKSOFT v komentovaném výpisu MDOSu psal, že autoři na to mysleli a napsali ty 2 kritické podprogramy tak, aby se vše stihlo.

A můj názor? Jo u D40 možná jo, ale u D80 **zaručeně ne**! Oni ty podprogramy totiž napsali tak, že čím větší logický sektor (a taky čím menší počet sektorů na stopu), tím delší doba trvání. U D40 je vše celkem v pohodě (při 9 sektorech na stopu), ale na D80 je to u vzdálenějších stop katastrofa! Já sám se nepovažuju za kdovíjak skvělého programátora, ale ten jejich LOGFYZ (#1DF9) a FYZLOG (#1DE9) je hnus! Oba nechutně zacyklené a přitom úplně zbytečně.

Takže jsem sednul a napsal oba úplně znova. Díky nim už sektor hlavě **nikdy** neujede a vše se stihne! Taky silně uvažuju, že si je implantuju i přímo do MDOSu…

Pro pobavení malá ukázka:
(jde o čisté časy, tj. bez "zbytečných" odskoků - call IXAB)

podprogram LOGFYZ (DIVIDE)			
sektor	sektory na stopu	čas (T-cykly)	
		MDOS	můj
0	9	91	**976**
359	9	1222	**936**
719	9	2382	**926**
1079	9	3542	**916**
1439	9	4702	**916**
0	6	91	**976**
359	6	1802	**926**
719	6	3542	**916**
1079	6	5282	**926**
1439	6	7022	**906**

podprogram FYZLOG			
číslo stopy	sektory na stopu	čas (T-cykly)	
		MDOS	můj
0	9	89	**84**
39	9	1025	**309**
79	9	1985	**309**
159	9	3905	**309**
0	6	89	**84**
39	6	1025	**237**
79	6	1985	**237**
159	6	3905	**237**

MFC bohužel nedokáže ujetí sektoru zabránit vždy. Může za to "čištění" posledního sektoru u každého souboru (čím míň bytes souboru skutečně patří, tím pomalejší LDDR…).

Od MFC verze 3 je LOGFYZ ještě kratší a rychlejší. :-)

DIVIDE (LOGFYZ)

```
;OUT: IX=adresa parametrů disku, A=číslo aktuálního drivu (0=A, 1=B)

IXAB     ld   a,(15979)
         or   a
         ld   ix,15872
         jr   z,IXAB2
         ld   ix,15872+12
IXAB2    ld   (#3EE7),ix            ;to je zde kvůli chybě v MDOSu...
         ret

;IN: HL=číslo logického sektoru
;OUT:B=číslo stopy, C=číslo sektoru na stopě

DIVIDE   call IXAB                  ;MFC verze 1 a 2
         ex   af,af'
         ld   a,h
         ld   c,l
         ld   hl,0
         ld   e,(ix+3)              ;počet sektorů na stopu
         ld   d,h
         slia c                     ;5x slia a rla
         rla                        ;rozepsáno kvůli rychlosti
         slia c                     ;(nemusí být, pak ale
         rla                        ;je třeba níže dát ld b,16 místo ld b,16-5)
         slia c
         rla
         slia c
         rla
         slia c
         rla
         ld   b,16-5                ;viz. ASSEMBLER a ZXS 1. díl
DIVIDE2  slia c                     ;dělíme číslo sektoru
         rla                        ;počtem sektorů na stopu
         adc  hl,hl
         sbc  hl,de
         jr   nc,DIVIDE3
         add  hl,de
         dec  c
DIVIDE3  djnz DIVIDE2
         ld   b,c                   ;v C je výsledek dělení (fyz. stopa)
         ld   c,l                   ;v L je zbytek po dělení (fyz. sector)
         ex   af,af'                ;aktuální drive
         ret

;IN: HL=číslo logického sektoru
;OUT:B=číslo stopy, C=číslo sektoru na stopě

DIVIDE   call IXAB                  ;MFC verze 3
         ex   af,af'
         xor  a
         ld   c,(ix+3)
         ld   b,16
DIV1     add  hl,hl
         rla
         cp   c
         jr   c,DIV2
         sub  c
```

```
        inc  l
DIV2    djnz DIV1
        ld   b,l
        ld   c,a
        ex   af,af'
        ret


;tentýž podprogram z MDOSu

DIVIDE  call IXAB
        ld   e,(ix+3)
        ld   d,0
        ld   b,255
        or   a
DV0     inc  b
        sbc  hl,de
        jp   nc,DV0             ;tohle je kámen úrazu (od logického
        add  hl,de              ;se odečítá počet sektorů na stopu
        ld   c,l                ;dokud nedojde k přetečení)
        ret
```

FYZLOG

```
;IN: B=stopa, C=sektor
;OUT: HL=logický sektor

FYZLOG  push de
        ld   e,b
        ld   hl,0
        ld   b,h
        ld   d,h
        inc  e
        dec  e
        jr   z,FYZL2
        ld   b,(ix+3)           ;cyklí se podle počtu sektorů na stopu
FYZL1   add  hl,de              ;(těch je od 6 do 10 - takže rychlovka)
        djnz FYZL1
FYZL2   add  hl,bc
        pop  de
        ret


;tentýž podprogram z MDOSu

FYZLOG  push de
        ld   e,(ix+3)
        ld   d,0
        ld   h,d
        ld   l,c
        inc  b
        jr   CALC1

CALC    add  hl,de
CALC1   djnz CALC               ;cyklí se podle čísla stopy - GOOD SHIT!
        pop  de
        ret
```


Příloha č.3 – popis souborů

soubor .D_0

Offsest	Délka	Obsah	Význam
0	1	0	verze souboru .D_0
1-3	3	"D_0"	přípona (kvůli identifikaci souboru)
4-13	10	"MDOS_D4080"	info o diskovém systému
14-45	32	údaje z DIR	32 bytes o souboru přenesených z DIR
46	1	0 nebo 1	BOOT archivován? (0=NE, 1=ANO)
47	1	0 nebo 1	XOR souboru? (0=NE, 1=ANO)
48-510	463	samé 0	volno
511	1	výsledek XORu	pokud offset 47=1, potom jde o XOR souboru (výsledek XORu od offsetu 1024 do konce)
512-1023	512	boot	pokud offset 46=1, potom je zde uložen boot diskety, ze které byl soubor archivován
1024-??	??	soubor	data archivovaného souboru

Pozn.: MFC XORování souboru z časových důvodů neprovádí.

Formát D_0 jsem se snažil vymyslet co možná nejuniverzálněji. **Podle mne je možné upravit jej pro archivaci souborů z jakéhokoliv existujícího file systému (FS).**

Kdyby náhodou někdo z Vás chtěl tento formát aplikovat i na svůj FS, který používá u ZXS (z Betadisku, MB02 atd.), tak se prosím snažte využít toho, co už je definováno (v současné době pouze "MDOS_D4080"). Pokud Váš FS má DIR podobný s MDOSem (tj. uchovává stejné info - třebas i jinak uspořádané, to nevadí), tak nemusíte D_0 vůbec modifikovat! A bude to mít i jeden příjemný vedlejší efekt – my (D80káři) v pohodě přečteme Vaše soubory a Vy zas ty naše :).

soubor .MFC

Offsest	Délka	Obsah	Význam
0	1	0	verze souboru .MFC
1-3	3	"MFC"	přípona (kvůli identifikaci souboru)
4	1		verze snapu 0=48K, 1=128K
5	1	0-7	BORDER color
6,7	2		SP
8,9	2		PC (když 0 tak se spouští pouze přes SP)
10	1		IM (0,1,2)
11	1		reg I
12,13	2		HL`
14,15	2		DE`
16,17	2		BC`
18,19	2		AF`
20,21	2		HL
22,23	2		DE
24,25	2		BC
26,27	2		IY
28,29	2		IX
30	1		bit 2 = IFF (0 - DI, 1 - EI)
31	1		reg R
32,33	2		AF
34	1		spouštěcí rutina od offsetu 100? (0=NE, 1=ANO)
35	1		OUT 32765 (stránka pro start)
36	1		OUT 65533 (last out)
37-52	16		hodnoty z AY registru
53	1		je **nutná** ROM z offsetu 54 (0=NE, 1=ANO)
54	1		XOR ROMKy (0-16383)
55-99	45		nevyužito
100-355	256		spouštěcí rutina (na offsetu 355 je XOR rutiny)
356-510	155		nevyužito
511	1		XOR dat (512 až 49663 nebo 512 až 131583)
512-49663	49152		48K data (16384-65535)
512-131583	131072		128K (16384-65535 + stránky 17,19,20,22,23)

Pozn.: MFC XORování dat z časových důvodů neprovádí.

Spouštěcí rutina

```
IN: IX odkaz na offset 0
BC,DE 4 bytes pro instrukci na spuštění (resp. přestránkování)
 (u D80 JP #1700 a NOP: B=0, C=195 (JP), D=0, E=23)

offset 100
RUN_MFC  jr   RUN_2
         defm "(c)MTs 2.0"          ;10 bytes pro identifikaci (verze atd.)

RUN_2    di
         ld   sp,ix
         pop  hl
         pop  hl

         push ix
         pop  iy
         ld   h,d
         ld   l,e
         ld   de,100+1
         add  iy,de

         ld   (iy+125-1),c          ;hodnoty pro spuštění (přestránkování)
         ld   (iy+126-1),b
         ld   (iy+127-1),l
         ld   (iy+128-1),h

         pop  hl
         ld   a,h
         out  (254),a
         ld   a,l
         or   a
         jr   z,RUN_48

         ld   a,(ix+35)
         and  %00001111
         add  a,16                  ;musí fungovat D40/80 takže vždy ROM48!
         ld   bc,32765
         out  (c),a

RUN_48   pop  hl
         pop  bc
         ld   a,b
         or   c                     ;Program Counter = 0?
         jr   z,RUN_NOPC

         dec  hl
         ld   (hl),b                ;uložení reg. PC na zásobník hry
         dec  hl
         ld   (hl),c

RUN_NOPC ld   (iy+122-1),l          ;v HL je zásobník hry (reg. SP)
         ld   (iy+123-1),h
         ld   a,(ix+30)
         bit  2,a                   ;stav přerušení (EI nebo DI?)
         jr   z,RUN_DI
         ld   (iy+124-1),251        ;kód instrukce EI
```

```
RUN_DI    pop  bc
          ld   a,b
          ld   i,a
          im   0
          inc  c
          dec  c
          jr   z,RUN_IMOK
          im   1
          dec  c
          jr   z,RUN_IMOK
          im   2
RUN_IMOK  pop  hl
          pop  de
          pop  bc
          pop  af
          exx
          ex   af,af'
          pop  hl
          pop  de
          pop  bc
          pop  iy
          pop  ix
          pop  af
          ld   r,a
          pop  af
          ld   sp,00000             ;zásobník
          nop                       ;tady bude případně EI
          nop                       ;tady se uloží CBLH (JP #1700, NOP)
          nop
          nop
          nop

          defs 126                  ;volno (NOP)

XOR       defb 32                   ;výsledek XORu celé rutiny
```


Příloha č.4 – popis OS

MDOS (D40/80)

Úvod

Bytes na sektor	512
BOOT, FAT, DIR	prvních 14 sektorů (7168 bytes)
sektorů na cluster	1 (=512 bytes)
max položek ve FAT	1705
první volná položka	14
poslední volná položka	1704
FAT	12ti bitová (FAT12)
max počet souborů	128 (32 bytes na soubor v DIR)

BOOT

(0.sektor, 512 bytes, 0-511)

Standardní BOOT

Offset	Délka	Význam
0-127	128	volno
128-176	48	info o všech mechanikách připojených při formátování diskety
176-187	12	info o disketě a mechanice
188-191	4	volno
192-201	10	jméno diskety
202-203	2	náhodný dvojbajt
204-207	4	zapsáno "SDOS"
208-511	304	volno

BOOT vytvořený MFC

Offset	Délka	Význam
0-39	40	zapsáno "Formated with MDOS File Commander (MFC)."
40-127	88	volno
128-176	48	info o všech mechanikách připojených při formátování diskety
176-187	12	info o disketě a mechanice
188-191	4	volno
192-201	10	jméno diskety
202-203	2	náhodný dvojbajt
204-207	4	zapsáno "SDOS"
208	1	další (třetí) náhodný bajt
209-511	303	volno

Na adrese 128-176 jsou zkopírovány systémové proměnné z diskové ROM (od 15872). Jde o 48 bytes, kde každých 12 bytes je vyhrazeno pro 1 mechaniku (A,B,C,D – řadič umožňuje pracovat až se 4mi mechanikami, bohužel MDOS pouze se 2mi).

Na 176-187 jsou ty samé sys. proměnné, ale pouze těch 12 bytes, které se týkaly mechaniky a diskety při formátování.

Raději příklad. Když zformátujete disketu na mechanice B, tak se do bootu na offset 128-176 přenese 48 bytes z adresy 15872 (sys. proměnné o mechanice A, B, C, D) a na offset 176-187 se přenesou ty samé sys. proměnné, ale už ne pro všechny 4 mechaniky, ale pouze pro tu aktuální (v našem příkladě B).

Pokud se nad tím zamyslíte, zjistíte, že offset 128-176 je v bootu úplně na nic. Smysl by měl pouze tehdy, když by byl zničený offset 176-187.

Těch 12 bytes systémových proměnných pro každou mechaniku je životně důležitých, takže je proberu podrobněji:

offset	bit	význam	
0	bit 0	0 - mechanika nepřipojena 1 – připojena	
	bit 1-6	???	
	bit 7	0 – buď mechanika zastavena, nebo došlo k chybě 1 – mechanika se točí a je připravena k nějaké operaci	
1	bit 0	???	DISKETA
	bit 1	???	
	bit 2	0 – mechanika A 1 – mechanika B	
	bit 3	0 – jde o D80 1 – jde o D40	
	bit 4	0 – jednostranný formát 1 – oboustranný	
	bit 5	0 – normální režim 1 – disketa má poloviční počet stop než mechanika	
	bit 6,7	rychlost krokování mechaniky	
2		počet stop diskety	
3		sektorů na stopu	
4		číslo stopy kam byla naposledy nastavena hlava	
5	bit 0	???	MECHANIKA
	bit 1	???	
	bit 2	0 – mechanika A 1 – mechanika B	
	bit 3	0 – jde o D80 1 – jde o D40	
	bit 4	0 – jednostranná mechanika (pouze 1 hlava) 1 – oboustranná mechanika	
	bit 5	0 – normální režim (3,5"=80stop, 5,25"=40stop) 1 – 80ti stopá 5,25" mechanika ale chceme D40	
	bit 6,7	rychlost krokování mechaniky	
6		počet stop mechaniky (40 nebo 80)	
7		počet sektorů na stopu (9)	
8-11		MDOS 1.0 nevyužito, MDOS 2.0 to ale používá!!!	

MDOS pracuje následovně. V diskové ROM jsou natvrdo uloženy proměnné mechanik A a B (24 bytes). Tyto se při resetu přesunou na adresu 15872. Důležité je, že přesunem se vyplní pouze parametry mechaniky (vše ostatní je nulové). Parametry diskety se vyplní **až při práci s disketou** (no, tohle není úplně pravda, v MDOSu je chyba a počet stop mechaniky se při resetu nastaví i na offset 3 – ale jak říkám tohle je "bug" a naštěstí naprosto neškodný). Získají se tak, že se nejprve zkopírují z parametrů mechanik a tato kopie se pak modifikuje podle BOOTu (využití offsetu 177,178,179).

Pozor! Těch 24 bytes (parametry mechanik v ROM) je nastaveno tak, že připojené mechaniky budou oboustranné (2 hlavy) a taky normální režim = offset 5, bit 4 a 5 nastaveno na nulu. Tohle je OK, ale MDOS má špatně test ohledně bitu 5 a pokud je potom vložená disketa naformátována na poloviční počet stop (20 stop u D40 nebo 40 stop u D80), bit 5 na offsetu 1 se nastaví na 1 a vše je v hajzlu… Ten bit by se měl nastavit **pouze** tehdy, když máte připojenou **5,25" mechaniku, která má 80 stop** (1,2MB) a ta má pracovat jako D40. Rozdíl mezi 40stopou a 80stopou 5,25" mechanikou je v jiné technologii šířky stopy (80stopá musí při emulaci 40stopové provést dvojnásobný posun hlavy). U 3,5" nic takového není!

MFC verze 1 a 2 bit 5 nikdy na 1 nenastavovaly (resp. nastavily ho jedině tehdy, pokud už tak byl v ROM, což ale u oficiálních verzí MDOSu nehrozilo…). **MFC verze 3 emulaci povoluje! Ale pouze u diskety, která je naformátována na 40 stop a je v mechanice, která má 80 stop**.

Při zkoumání MDOSu jsem dále zjistil, že bit 3 na offsetu 1 a 5 je úplně zbytečný (systém jej nijak netestuje ani nevyužívá).

Tam, kde jsou v tabulce otazníky, mi není přesně jasné k čemu tyto informace systém používá (jestli je vůbec používá…).

Přiznám se, že jsem kdysi zkoušel připojit 80ti stopou 5,25" jako B. Bohužel už si nepamatuju, jestli četla diskety z D40 (myslím ale, že ano). Z výpisu MDOSu jsem později zjistil, že když je onen kritický bit 5 nastaven na 1, tak se při pokusu na disketu něco zapsat nebo ji naformátovat, objeví hláška *Internal error*. Rovněž nastane problém se čtením D40 diskety v 80stopé 5,25" mechanice, pokud nebude standardně naformátovaná (jinak než na 40 stop). Bit 5 se totiž nenastaví. Ono je to celé s tím bitem 5 na prd, neboť když se má nastavit tak se nenastaví a když nemá tak se zas nastaví… Tu emulaci D40 na 80ti stopé mechanice měl autor úplně vynechat (způsobil více škody než užitku) a nebo ji měl domyslet (napsat spolehlivější testování 80ti stopé pětačtvrtky).

Problematika 80ti stopé 5,25" mechaniky sahá hluboko do minulosti… 360kB (40 stop 5,25") mechanika se od 1,2MB (80 stop 5.25") výrazně liší v hardwarové technologii. Hlava u 360kB zapisuje 16 mi (milipalců) širokou řádku. Ovšem hlava na 1.22MB pouze 8 mi !! Takže když něco zapíšete technologií 8 mi a pak tu disketu chcete přečíst zase 16 mi, tak se smysluplných dat s velkou pravděpodobností nedočkáte. Takže aby bylo jasno - všechno z 40stopé mechaniky byste měli přečíst na 80stopé, ale to, co vytvoří (formát/zápis) 80stopá, už na 40stopé bezchybně přečíst nelze (resp. je to věc náhody). Je proto celkem logické, že MDOS dovolí u diskety s polovičním počtem stop pouze číst a při pokusu o formát nebo zápis vypíše Internal error.

*Časem ovšem někoho (Billa ale určitě ne!!) napadlo, že "když to nejde přes hardware, proč na to nejít přes software? Prostě budem na 80stopé jednotce zapisovat po **dvojici** "úzkých" stop a ta 40stopá to pak přece musí sežrat!". A taky že jo, nápad to byl dobrý a fungoval.*

Mechanika	Počet stop na palec (tracks per inch – TPI)	Šířka stopy (tisícina palce – mi)	Doporučená citlivost hlavy tj. 80% šířky stopy	stopy x sektory na stopu
5,25" 360 kB	48 TPI	20,8 mi	16 mi (± 8mi od středu stopy)	40 x 09
5,25" 1,2 MB	96 TPI	10,4 mi	8 mi (± 4mi od středu stopy)	80 x 15
3,5" 720 kB	135 TPI	?? mi	??	80 x 09
3,5" 1,44 MB	135 TPI	?? mi	??	80 x 18

Z tabulky je vše jasně vidět. U 3.5" neznám přesná čísla (v knize, ze které jsem čerpal je zřejmě chyba, protože mi ty výsledky pořád nevychází…), ale šířku stopy i citlivost má DD i HD 3.5" stoprocentně stejnou a daný HW problém se u nich nevyskytuje!! U 3.5" se kapacita zvedla tak, že se zdokonalilo médium (disketa) a trošku "vylepšila" mechanika. Díky HD (High Density – vysoká hustota) disketě bylo možné zdvojnásobit počet sektorů na stopu. 3,5" HD mechanika umí vše přesně tak, jako její DD předchůdce (naopak to samozřejmě neplatí). Proto je klidně možné vyměnit originální 3,5" DD mechaniku z D80(B) a nahradit ji HDčkovou.

Když už jsem se tak rozepsal, tak napíšu i pár vět o médiu (disketě). Disketa je tvořena stopami a sektory. Stopa je v podstatě kružnice a její šířka závisí na hw technologii (viz. ta tabulka výše). Mechanika by měla mít hlavu nastavenou tak, aby byla vždy uprostřed každé stopy. Kdyby mechanika byla schopna data zapsat/přečíst přesně na/ze střed(u) každé stopy, šlo by o naprosto dokonalou mechaniku (nutno podotknout, že takovou asi ještě nikdo nikdy neviděl…). Proto jsou dovoleny odchylky – říká se tomu citlivost hlavy (viz. zase ta tabulka). Je doporučeno, že odchylka by měla být v rozmezí 80% celkové šířky stopy. A toho by už měla být každá mechanika schopná splnit (jestli ne tak do koše s ní!). Ono by se mohlo zdát že čím více milipalců citlivost hlavy bude, tím líp (nejlépe 100%, tj. celou šířku stopy), ale tohle je blbost a hrozí CRC a jiné errory! Proč? Protože při 100% by se mohly při čtení přimíchat do čtených dat i informace (data) z vedlejší stopy. 80% je ale minimum, bude-li mechanika umět víc, tím lépe.

Při formátování diskety se do jednotlivých stop (kružnic) vytvářejí sektory. V tabulce jsou uvedeny doporučené (=zaručena bezchybnost) formáty na jednotlivých mechanikách. Méně (sektorů či stop) můžete použít úplně bez obav. Více ovšem ne (při větším počtu stop může v nejhorším případě dojít až k utržení celé hlavy) a když už se formát podaří, tak na jiných mechanikách můžou nastat problémy (to, že Vaše 3,5" umí 83 stop ještě neznamená, že to umí každá…).

Pro zajímavost Vám ještě prozradím, jak zjistíte, kolik palců na disketě Váš formát skutečně zabírá (tj. jak velkou plochu na disketě data budou potřebovat):

šířka dat na disketě ["] = šířka stopy [mi]*počet stop/1000

Kdybyste to chtěli na cm (jako že asi jo :-), tak výsledek musíte vynásobit číslem 2,54 (tolik je 1 palec centimetrů). Pokud šířku dat

v palcích vynásobíte údajem TPI z tabulky, mělo by (musí) Vám vyjít počet stop, které jste dosazovali do vzorce.

*Příklad pro 5,25" 360kB: 20,8*40/1000**=0,832"** * 2,54**=2,1cm***
*0,832*48TPI=40stop*

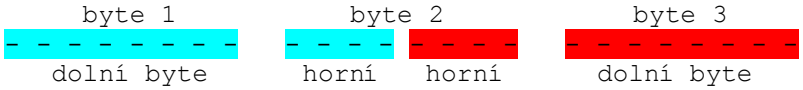
Disketa je samozřejmě vždycky o dost větší. To také vysvětluje, že třeba vpich špendlíkem do určitých míst, data na disketě nijak neovlivní (nevyžitá oblast je několik mm na začátku (okraji) i konci (od plíšku) diskety). Na okraji (začátku diskety) ale raději neexperimentujte, protože hned první sektor je boot…

Jo málem bych zapomněl ještě na HD a DD diskety. Rozdíl je v jejich povrchu. Povrch u HD je méně citlivý než u DD. Jinými slovy pro HD je třeba větší magnetický výkon. Udělat z DD diskety HDčkovou a používat ji se vysoce nedoporučuje (data se tam začnou různě přemisťovat – DD povrch je citlivější a HD mechanika ho zmagnetizuje až moc). Chvíli (pár hodin/dnů) to ale funguje a zapsaná data jsou i bezchybně přečtena. Co se stane, když uděláte z HD Ddčkovou, Vám nepovím (přesně nevím). Teoreticky by magnetický výkon DD mechaniky neměl na HD povrch stačit, ale většinou to funguje (sám mohu potvrdit, neboť DD diskety už dneska člověk u nás nesežene (v zahraničí v pohodě ano) a i já je v Kompaktu požívám – zálohu mám ale raději i na PC díky MFC (soubory D_0), člověk nikdy neví, zvláště u hw a sw je všecko možné… :-).

FAT

(1. - 5. sektor, 2560 bytes, 512-3071)

maximálně 1705 položek (0.-1704.)



DEC	HEX	Význam
0	#000	volný sektor
3072	#C00	nulový soubor (je zapsáno ve FAT ne v adresáři!)
3583	#DFF	vadný sektor
3549	#DDD	nedostupný (boot, fat, dir, a to co přesahuje formát.)
3584-4095	#E00-#FFF	koncová značka souboru (DEC - 3584 = délka, 0=512)

FATka je v každém svém sektoru zalomena!!! To znamená, co sektor s částí FAT, to na jeho konci kód 13 (jde o naprosto zbytečný "půlbajt" navíc). Bacha na to!

Zjistil jsem také to, že MDOS se vždy při ukládání souboru snaží zajistit, aby byl celý soubor uložen pěkně "za sebou" (žádné přeskakování sektorů). MFC ale ukládá vždy do nejbližšího volného sektoru!

DIRectory

(6. – 13. sektor, 4096 bytes, 3072-7167)

Offset	Délka	Význam
0	1	přípona souboru (P, B, S, Q, N, C) nebo hodnota 229 (=vymazaný soubor, nebo volné místo)
1-10	10	jméno souboru
11-12	2	délka souboru
13-14	2	počáteční (startovací) adresa nebo řádek
15-16	2	délka programu (*.P) bez proměnných, jinak nemá význam (32768)
17-18	2	číslo prvního sektoru souboru
19	1	nula, u bezhlavičkového souboru BODY flag
20	1	atributy souboru, každý bit (76543210) odpovídá jednomu příznaku (HSPARWED)
21	1	pro uložení délky souborů delších než 65535 (*.Q)
22-31	10	nevyužito (zaplněno hodnotou 229)

Po zformátování je celý adresář zaplněn hodnotou 229. Ukládáním (SAVE*) se postupně zaplňuje. **Při vymazání souboru (ERASE) se odstraní pouze přípona souboru a místo ní je uloženo 229.**

MS-DOS (FAT12, DD 720kB)

Úvod

Bytes na sektor	512
BOOT, FAT, DIR	prvních 14 sektorů (7168 bytes)
sektorů na cluster	2 (=1024 bytes)
max položek ve FAT	1024
první volná položka	2 (=sektor 14,15)
poslední volná položka	714 (=sektor 1438,1439)
FAT	12ti bitová (FAT12)
max počet souborů	112 (32 bytes na soubor v DIR)

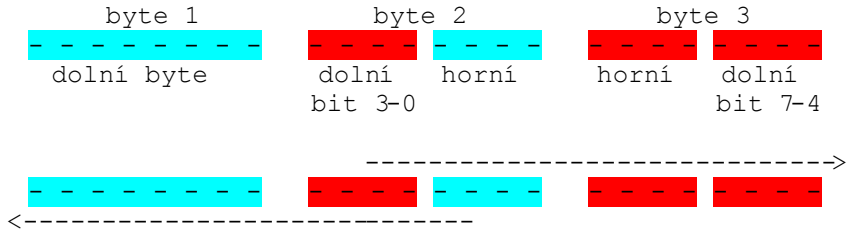
BOOT
(0.sektor, 512 bytes, 0-511)

Offset	Délka	Hodnota	Význam
11-12	1	512	bytes na sektor
13	1	2	počet sektorů na cluster
16	1	2	počet FAT
19-20	1	1440	celkový počet sektorů
22-23	2	3	počet sektorů pro jednu FAT
24-25	2	9	sektory (tracks)
26-27	2	2	počet hlav
43-53	11		jméno diskety
54-58	304	5	zapsáno "FAT12"

Pozn.: Tabulka rozhodně není kompletním popisem BOOTu. Jde pouze o výcuc!

FAT hlavní & záložní
(1. - 3. sektor, 1536 bytes, 512-2047)
(4. - 6. sektor, 1536 bytes, 2048-3583)

maximálně 1024 položek



DEC	HEX	Význam
0	#000	volný cluster
4087	#FF7	vadný cluster
4080-4086	#FF0-#FF6	nedostupný/rezervovaný
4088-4095	#FF8-#FFF	koncová značka souboru

Přepočet clusteru na sektor: sektor=cluster*2+10

Nulový soubor by měl být ukončen již v adresáři (#FFF). Bohužel na mnou testovaném MS-DOSu od W98 se u nulového souboru zapsala jako odkaz do FAT nula. Což je ale pěkná pí****na.

Záložní FAT by měla být vždy kopie hlavní. Zjistil jsem ale, že Win98 klidně pracuje i s disketou kde záložní FAT chybí (ačkoli by tam být měla) a neohlásí jedinou chybu (pokud je ovšem ta hlavní naprosto v pořádku). Dobré co?

Po zformátování jsou v celé FAT samé nuly (pokud ovšem na disketě není nějaký vadný sektor…). To je dost nevýhodné, podle mě by bylo lepší, kdyby položky ve FAT, které nejdou použít (tj. ty co přesahují formát) obsahovaly nějakou nenulovou hodnotu.

Narozdíl od MDOSu zde "zalomení" FATky (naštěstí) není.

DIRectory
(7. – 13. sektor, 3584-7167)

Offset	Délka	Význam
0-7	8	jméno souboru
8-10	3	přípona souboru
11	1	atributy souboru, každý bit (76543210) odpovídá jednomu příznaku (RHS--A--), pokud bit4=1 jde o adresář
12-21	10	nepoužito
22-25	4	datum
26-27	2	číslo prvního sektoru souboru
28-31	4	velikost souboru

datum

bit	Význam
0-4	sekundy/2 (0-29)
5-10	minuty (0-59)
11-15	hodiny (0-23)
16-20	den (1-31)
21-24	měsíc (1-12)
25-31	rok (skutečný rok = rok +1980)

Windows dělá v adresáři pěkný bordel *(jak já ty okna nenávidím… …zaberou plno místa a stejně jsou k ho…)*. Je to díky podpoře dlouhých názvů, a tak 1 soubor klidně obsadí více položek (po 32 bytech) v adresáři. Ale takováto přebytečná položka se dá rozpoznat (odkaz do FAT je u této položky nula, což je nesmysl).

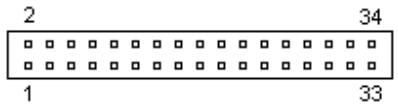
Po naformátování jsou všude samé **nuly**. Když nějaký soubor vymažete odstraní se pouze první písmeno souboru (místo něj je vloženo **229**). Pokud tedy nenaleznete v adresáři nějakou položku začínající nulou, ještě to neznamená, že disketa (adresář) je už plná, ještě musíte vyzkoušet kód 229.

Příloha č.5 - jak si vyrobit D80B

Jak víte, tak MFC nepodporuje diskety 5,25" z PC (MS-DOS). A jelikož mi Sweet Factory řekl, že zná minimálně jednoho člověka, který tímto (prehistorickým) způsobem ještě data přenáší, tak přicházím s perfektním řešením. **Pořiďte si D80B! :)**

Vyrobit si D80B lze celkem snadno. Stačí k tomu zdroj, 3,5" mechanika a propojovací kabel. Schéma kabelu je uvedeno níže. Zdrojem se tu zabývat nebudu (ten si obstarejte jak chcete – vlastní výroba, PC zdroj, vytáhnutí drátů z počítače, upgrade D40…), pouze podotknu, že na 3,5" stačí pouze 5V (12V není vůbec třeba). Mechaniku můžete použít jakoukoliv běžně dostupnou v počítačovém obchodě. Dneska už se stejně dostanou jen AT-čkové HD mechaniky s kapacitou 1,44MB. DD by byla sice vhodnější (řadič v D40(80)A totiž pracuje vždy v režimu DD), ale ty už se nevyrábí. HD mechanika je naštěstí pro nás v pohodě použitelná, protože je plně kompatibilní s DD (pozor u 5,25" tohle neplatí, tam je mezi DD a HD velký rozdíl...). Z uvedeného samozřejmě automaticky vyplývá, že pokud Vám odejde originální mechanika v D80A(B) či Kompaktu, můžete ji klidně vyměnit za HD…

Kdybyste chtěli náhodou někdy vyrábět D40B (příp. měnit mechaniku v D40A(B)), tak věřte, že s tím jsou jen problémy. Tady už DD mechaniku potřebujete (a sehnat ji dneska je hotové mistrovské umění) a stejně i tak Vám to ještě nemusí chodit. Schéma kabelu by bylo stejné jako předtím (mám vyzkoušeno - DD mechanika z frcu za 50Kč a univerzální (tj. 3,5" i 5,25" koncovky) kabel pro připojení FDD do PC). Jo a u 5,25" už těch 12V potřebujete…



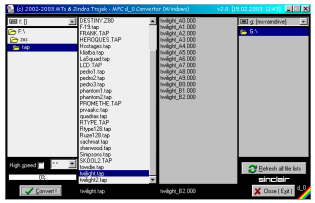
	EXTENDED			**FDD 3,5" HD**
	strana součástek			
2	/HEAD LOAD	o-------o	4	???
4	INDEX	o-------o	8	/INDEX
6	/DS1	o-------o	10	/DS0
7	/MOTOR 1	o-------o	16	/MOTOR ON
9	DIRECTION SELECT	o-------o	18	/DIRECTION SELECT
10	/STEP	o-------o	20	/STEP
11	/WRITE DATA	o-------o	22	/WRITE DATA
12	/WRITE GATE	o-------o	24	/WRITE GATE
13	/TRACK 00	o-------o	26	/TRACK 00
14	/WRITE PROTECT	o-------o	28	/WRITE PROTECT
15	READ DATA	o-------o	30	/READ DATA
16	SIDE SELECT	o-------o	32	/SIDE SELECT
	strana spojů			
1…15+16	GND + READY	o-------o	19,21,23,25	GND

Důležitá pozn.: Zapojení počítá s tím, že **obě** mechaniky jsou nastaveny jako A. Musíte tedy zajistit, aby tomu tak skutečně bylo (většinou stačí nastavit DS0 – u starých mechanik jumper/switch, u nových "kapka" cínu). U dnešních mechanik je vždy už z výroby nastaveno DS1, takže s tou malou ruční úpravou počítejte…

Příloha č.6 – podpora MFC na PC

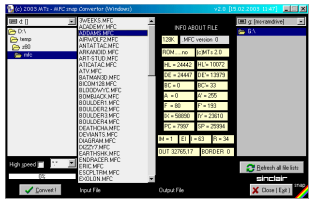
Na PC jsem v C++ vytvořil programy pro práci s .D_0 a .MFC soubory. Fungují na OS Windows 95 a vyšších (na Linuxu jdou ovšem díky projektu Wine spustit také). Jsou zadarmo a plně k dispozici na mých www stránkách.

MFC d_0 Convertor



D40 → D_0
D80 → D_0
TAP → 000
000 → TAP
D_0 → 000
000 → D_0

MFC snap Convertor



SNA → MFC
Z80 → MFC
__S → SNA
MFC → MFC

__S je snap disketové jednotky D40/80
U MFC → MFC jde o aktualizaci (spouštěcí rutina) a otestování .MFC souboru.

Příloha č.7 – seznam hotkeys

Okno	
levé	1
pravé	2
přepnutí	SPACE
Disketa	
nová	N
přejmenování	V
quick format	SS+F
full format	CS+F
zrušení DIR ochrany	SS+X
uložení DIR	X
Soubor	
označení	ENTER
přejmenování	R
změna atributů	A
změna přípony	S
změna start adresy	B
posun v DIR ↑	P
posun v DIR ↓	L
xor	SS+Z
run (spuštění)	SS+R, 0
Blok	
označení (maska)	SS+K
označení všech	K
odznačení (maska)	SS+J
odznačení všech	J
suma (clustery)	SS+8
suma (sektory)	SS+9
vymazání	SS+D
kopírování (1:1)	C
kopírování (EX/IM)	SS+C
Ostatní	
load @	SS+U
spuštění utility	U,3
systémové info	SS+I
quit (reset)	SS+Q
Input	
LEFT	←
RIGHT	→
HOME	↑
END	↓
DELETE zprava	CS+0
DELETE zleva	CS+9
DELETE všech znaků	SS+Q
velké/malé znaky	CS+2
OK	ENTER
storno	CS+1
ALT xxx	CS+SS

original MFC

Okno	
levé	1
pravé	2
přepnutí	CS+SS
Disketa	
nová	N
přejmenování	R
quick format	F
full format	CS+F
zrušení DIR ochrany	SS+X
uložení DIR	X
Soubor	
označení	SPACE
přejmenování	9
změna atributů	A
změna přípony	S
změna start adresy	B
posun v DIR ↑	P
posun v DIR ↓	L
xor	SS+Z
run (spuštění)	ENTER
Blok	
označení (maska)	SS+K
označení všech	K
odznačení (maska)	SS+J
odznačení všech	J
suma (clustery)	SS+8
suma (sektory)	SS+9
vymazání	8
kopírování (1:1)	5
kopírování (EX/IM)	SS+C
Ostatní	
load @	SS+U
spuštění utility	U,3
systémové info	SS+I
quit (reset)	SS+Q
Input	
LEFT	←
RIGHT	→
HOME	↑
END	↓
DELETE zprava	CS+0
DELETE zleva	CS+9
DELETE všech znaků	SS+Q
velké/malé znaky	CS+2
OK	ENTER
storno	CS+3
ALT xxx	CS+SS

Shrek's MBC

Murphologická filozofie:

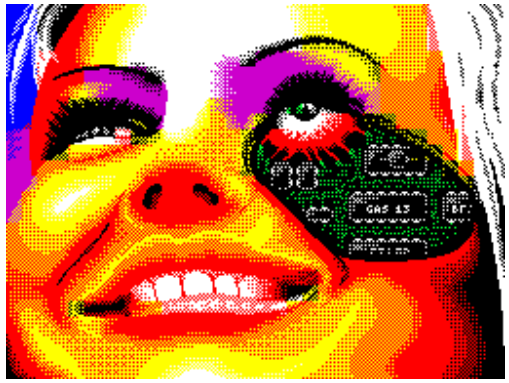
"Usmívejte se... zítra bude hůř."

Murphy

mts.zxs@tiscali.cz

ICQ: 144264603

(připomínky jsou vítány)



To je vše, přátelé.