MFC
souborové konverze na PC (Windows)

Obsah

1. Rychlý přehled

MFC d_0 Convertor

```
.d40 --> .d_0
.d80 --> .d_0
.d_0 --> .d80
.d80 --> .d40

.tap --> .000

.000 --> .tap
.d_0 --> .000
.000 --> .d_0
.bin --> .d_0
.tzx --> .tap
```

MFC snap Convertor

```
.z80 --> .mfc
.sna --> .mfc
.mfc --> .mfc
.__s --> .sna
```

MFC file cutter

```
*.* --> .bin
```

ZX Screens

```
.scr --> .bmp
.scr --> .gif
.scr --> .png
```

Vše je určeno pro PC s Windows, v linuxu jde spouštět pod Wine.
(psané v Borland C++ 4, zdrojové texty jsou přiloženy)

2. MFC d_0 Convertor

Program slouží pro práci s formáty:

.TAP .000 .D_0 .D40 .D80 .BIN .TZX

Levá strana v programu určuje zdrojové soubory (=co a **odkud** se převádí) a **pravá** cílový adresář (=**kam** se bude převádět). Označit lze samozřejmě i více souborů.

Nepůjde-li nějaký soubor převést, budete na to upozorněni chybovou hláškou (bude tam i důvod nepřevedení). Takovéto chybné soubory pak budou mít (ve výstupu) nulovou délku, nebo budou chybět úplně (záleží na druhu chyby). Po skončení konverze zůstávají problémové soubory označeny!

Zaškrtávací pole *High speed* nastavte (zaškrtněte), když budete převádět velké množství souborů (více než 30) najednou. Volba totiž způsobí, že se během konverze nebudou vypisovat v pravém okně výstupní soubory (musí-li se zobrazovat posuvník, tak to dlouho trvá... ...vyzkoušejte, uvidíte).

POZOR!

1. Soubory, které jsou zobrazeny, se automaticky "neREFRESHují". To znamená, že když něco v nějakém Commanderu odmažete, program si pořád bude myslet, že ty soubory tam jsou. K aktualizaci adresářů a souborů slouží tlačítko vpravo dole - *Refresh all file lists,* anebo klávesa F5.

2. Soubory v pravém okně mohou být bez varování přepsány (pokud má výstupní soubor stejné jméno jako ten, co už v pravém okně je, tak bude (s)prostě konvertorem přepsán).

3. Po dvojkliku na soubor v LEVÉM okně se vypíší informace o něm (funguje pouze pro .d_0).

Možnosti konverze:

(např.) **TZX --> TAP**

Tohle je naprosto neošetřená a šílená konverze napsaná pouze pro mé vlastní potřeby! Je třeba pracovat s DOSovským příkazovým řádkem (viz. volba ***DOS line...***).

exe file - DOSovský konvertor který se má volat (musí být uložen v *temp dir*!)

in file (mask) - vstupní soubor (resp. pouze jeho přípona)

out file - výstupní soubor (zase pouze přípona - musí být shodná s tím, co Vám vyhodí ten *exe file*)

temp dir - dočasné místo, kde bude konverze probíhat (ideální je RamDisk)

I/O synchro - synchronizační pauza (bude-li Vám program házet I/O vyjímky, musíte hodnotu o pár miliónů zvětšit)

Pozor na dávkovou konverzi (více souborů)! DOSovské programy se otevírají v příkazovém řádku a Windows s každým souborem otevře nové okno, které už nezavře (když tak byste si museli vytvořit .pif zástupce, který to okno po skončení sice zavře, jenže zase neuvidíte, jak ta konverze dopadla...). No a po pár desítkách takto otevřených okýnek Windows pěkně zatuhne :).

Rozhodně radím nepoužívat! Mám to tam jen proto, že příkazový řádek mi už leze krkem, a že *tzx2tap.exe* normálně bere jen DOSovsky pojmenované soubory (8+3); takhle to mám plně zautomatizované...

BIN --> D_0

Výsledná D_0 bude vždy typu "Bytes".

D40/D80 --> D_0

Konvertor umí pracovat i s image soubory simulující disketu z disketové jednotky D40/D80 (používá např. emulátor Real Spectrum). Pro soubory .d40 a .d80 jsem nestanovil žádné extra omezení, důležité je, aby v BOOTu byla značka "SDOS" a délka image nebyla větší než 872960 bytes (odpovídá 1705 položkám ve FAT). Formát diskety může být libovolný (i jednostranný). Od verze 4.2 lze převádět i image s tzv. sweet (MDOS3) infosektorem - program s tím jedním sektorem navíc na začátku počítá, ale raději si kontroluje značku „SDOS" i v něm (není-li tam, tak ho odmítne s chybou).

D80 --> D40

Je dostupné od verze 4.6 a možné pouze tehdy, vyberete-li souborový filtr **.d80* a u textu *Output file* (vpravo) dáte d40 (místo defaultně nastaveného d_0).

Touto konverzí si můžete vytvořit image pro D40 (40x9) z image D80 (80x9). Může se to hodit, máte-li jen 5,25" mechaniku. Předpokladem úspěšné konverze je nemít na původním d80 image data v sektorech 720-1439. Pokud tam něco bude (tj. data na d80 image zabírají více než se dá nacpat na d40 image), program ohlásí chybu a konverzi odmítne udělat.

D_0 --> D80

Tato konverze je možná pouze tehdy, vyberete-li souborový filtr **.d_0* a u textu *Output file* (vpravo) dáte d80 (místo defaultně nastaveného 000).

Vytváří se pouze jeden výstupní d80 soubor (image). V něm budou všechny soubory, které jste označili, včetně pořadí.

Při konverzi se ověřuje XOR (CRC error souboru), pokud je ovšem přítomen (viz. definice d_0...). Nastane-li CRC error, budete na to upozorněni.

Defaultně se tvoří d80 image o formátu 80x09 oboustranný, tj. 1440 sektorů. Pokud je ale **v prvním označeném** d_0 souboru přítomen BOOT, image se vytvoří podle něj. BOOT však může být ještě dále upraven, protože konvertor dovolí pouze 80x09 oboustranně a 80x10 oboustranně.

Je-li detekována d_0 vytvořená z originální diskety firmy ULTRASOFT (archivovaný BOOT, kde jméno diskety je "ULTRASOFT" a formát 10 sektorů na stopu), konvertor automaticky vytvoří d80 image o formátu 80x10 oboustranný a přidá i ochranu, kterou Ultrasoft používal. To znamená, že hra bude v pohodě spustitelná (žádná hláška "tato kopie byla zhotovena nelegalně" :).

Na image lze dostat pouze 128 souborů (viz. definice MDOSU) a suma všech souborů nesmí překročit formát. Tyto chyby jsou když tak hlášeny... (...**pozor na** to, že při **"Disk full"** už image vetšinou bude obsahovat část konvertovaného souboru, který nebude ve FAT ukončen. Na ZXS jej pak MFC (ScanDisk) samozřejmě označí jako chybný!!).

TAP --> 000

Narozdíl od Busyho TAPTO0.EXE se tady výstupní .000 pojmenovávají jinak (lépe :-). Smysl je v tom, že i když máte v Commanderu na PC zapnuto třídění podle abecedy, soubory budete mít i přesto seřazeny pěkně za sebou tak, jak byly v TAPce.

Při konverzi se ověřuje XOR (CRC error souboru), **u hlavičky i u těla**. Nastane-li CRC error, budete na to upozorněni. Výstupní .000 bude samozřejmě uložen i s touto chybou (tzn. **nic se neopravuje!**). Pokud jste si 100% jisti, že soubor poškozen není a chcete jej opravit, musíte provést konverzi 000-->D_0 a hned zase zpět (D_0-->000).

000 --> TAP

Tato konverze je možná pouze tehdy, vyberete-li souborový filtr **.0??* a u textu *Output file* (vpravo) dáte TAP (místo defaultně nastaveného d_0).

Vytváří se pouze jeden výstupní TAP soubor. V něm budou všechny soubory, které jste označili, včetně pořadí.

Při konverzi se ověřuje XOR (CRC error souboru), **u hlavičky i u těla**. Nastane-li CRC error, budete na to upozorněni. Ve výstupní TAPce bude samozřejmě dotyčný soubor uložen i s touto chybou (tzn. **nic se neopravuje!**). Pokud jste si 100% jisti, že 000 soubor poškozen není a chcete jej opravit, musíte provést konverzi 000-->D_0 a hned zase zpět (D_0-->000).

d_0 (pouze MDOS_D4080) --> 000

Tuto konverzi budete potřebovat, pokud chcete dostat soubory d_0 např. do TAPky (000 je totiž takový meziformát, se kterým umí pracovat většina ostatních konvertorů). Pozor, že maximální délka dat u formátu .000 je 65535 (proto taky mohou být některé .d_0 odmítnuty). Uvědomte si, že formáty 000 a TAP jsou dělány pro kazetové soubory, kdežto d_0 pro diskové! d_0 uchovává daleko více informací o souboru než 000 či TAPka. Když tedy provedete d_0 --> 000, je zcela logické, že **některé informace se ztratí.**

Formát d_0 existuje buď s XOR ochranou nebo bez ní. XOR ochrana znamená, že data v souboru jsou vyxorována a výsledek je uložen (uloží ho tam aplikace, která soubor d_0 vytvořila - MDOS File Commander XOR nepodporuje!). Děla se to proto, aby bylo možné později zjistit, jestli náhodou není soubor poškozen. Pokud byste chtěli zjistit, je-li XOR ochrana v souboru přítomna nebo ne, stačí na něj 2x kliknout (funguje to pouze v LEVÉM okně) a zobrazí se okénko s informacemi.

Konvertor si **XOR** hodnotu hlídá a pokud v .d_0 je, tak ji prověří (provede nový xor a porovná). **Nebude-li sedět, dostanete chybové hlášení s možností nechat si (výstupní!) soubor opravit.** Opravte jej však pouze tehdy, jste-li si 100% jisti, že .D_0 soubor chybný není!

D_0 nepřevedete, pokud má menší velikost než 1024 bytes (to se pak zaručeně o soubor .d_0 nejedná, protože 1024 bytes je jen jeho hlavička...).

Důležitá pozn.: Program umí převádět d_0 i jako bezhlavičkový. Že jde o bezhlavičkový soubor se pozná z údajů z DIR, které má v sobě (offest 19 v DIR u MDOSu).

000 --> d_0 (pouze MDOS_D4080)

Že se jedná o formát 000 se bohužel nedá zjistit z přípony souboru. Lze ale využít toho, že 000 má vždy XOR ochranu. Takže, není-li přípona .bin, .d40, .d80, .d_0 nebo .TAP (to by se provedla jedna z předcházejících konverzí - viz. výše), a sedí-li xor, pak se jedná o 000.

XOR se kontroluje u hlavičky i u těla. **Narazí-li program na nesoulad, dostanete chybové hlášení s možností nechat si (výstupní!) soubor opravit.** Opravte jej však pouze tehdy, jste-li si 100% jisti, že .000 soubor chybný není!

Při testování programu se objevil problém u souborů, které měly stejné jméno (a lišily se pouze příponou - např. HEADLESS.000, HEADLESS.001, HEADLESS.002 atd.). Na výstupu pak totiž byl jen jeden soubor a to ten posledně převedený (protože nový se vytvořil vždy na úkor toho předchozího, tzn. přepsal jej). Problém jsem vyřešil tak, že na výstupu bude soubor pojmenován jako na vstupu (vč. přípony) a místo tečky se vloží znak ~. Např: soubor HEADLESS.005 se bude na výstupu jmenovat HAEDLESS~005.d_0.

Důležitá pozn.: Do d_0 se převádí i bezhlavičkové 000. "Bodyflag" se zapíše do DIR (offset 19).

.D_0 formát

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

32 bytes z DIR

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)

Formát D_0 jsem se snažil vymyslet co možná nejuniverzálněji. Podle mne je možné upravit (doplnit) 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 zase ty naše :).

.TAP file format

This chapter is part of documentation to Sinclair ZX Spectrum 48/128/ SamRam/Interface1/RS232/AY-3-8912 Emulator 'Z80' v2.01 - 3/5/93 - by G.A. Lunter:

The .TAP files contain blocks of tape-saved data. All blocks start with two bytes specifying how many bytes will follow (not counting the two length bytes). Then raw tape data follows, including the flag and checksum bytes. The checksum is the bitwise XOR of all bytes including the flag byte. For example, when you execute the line SAVE "ROM" CODE 0,2 this will result:

```
      |------ Spectrum-generated data -------|       |---------|
13 00 00 03 52 4f 4d 7x20 02 00 00 00 00 80 f1 04 00 ff f3 af a3

^^^^^       first block is 19 bytes (17 bytes+flag+checksum)
      ^^    flag byte (A reg, 00 for headers, ff for datablocks)
         ^^ first byte of header, indicating a code block

filename    ^^^^^^^^^^^^^
header info               ^^^^^^^^^^^^^^^^^^
checksum of header                          ^^
length of second block                         ^^^^^
flag byte                                             ^^
first two bytes of rom                                  ^^^^^
checksum (checkbittoggle would be better)                     ^^
```

Note that it is possible to join .TAP files by simply stringing them together, for example COPY /B FILE1.TAP + FILE2.TAP ALL.TAP

For completeness, I'll include the structure of a tape header. A header always consists of 17 bytes:

Byte	Length	Description
0	1	Type (0,1,2 or 3)
1	10	Filename (padded with blanks)
11	2	Length of data block
13	2	Parameter 1
15	2	Parameter 2

The type is 0,1,2 or 3 for a Program, Number array, Character array or Code file. A screen$ file is regarded as a Code file with start address 16384 and length 6912 decimal. If the file is a Program file, parameter 1 holds the autostart line number (or a number >=32768 if no LINE parameter was given) and parameter 2 holds the start of the variable area relative to the start of the program. If it's a Code file, parameter 1 holds the start of the code block when saved, and parameter 2 holds 32768. For data files finally, the byte at position 14 decimal holds the variable name.

.000 file format

This format is like TAP, but main difference is each block (with or without 17-byte header) is stored in his own file. First byte of file is #00 for block with header and #FF for block without header. Extensions of file with this format may be 000,001,002...999.

Blocks without header:
For example if you run this routine:

```
   ... ld ix,0  ;  ld de,2  ;  ld a,#FF  ;  call #04C2 ...
```

then you will make this file (length 5 bytes):

```
                                      |---------| Spectrum generated
                                   FF FF F3 AF A3               data
Mark byte (file is without header)  ^^          ^^ parity
                      flagbyte of body ^^
                    first two bytes of rom ^^^^^
```

Blocks with header:
For example the same example as in TAP-section: SAVE "ROM" CODE 0,2

```
             |------ Spectrum-generated data -------| |---------|
          00 00 03 52 4f 4d 7x20 02 00 00 00 00 80 f1 ff f3 af a3
Mark byte ^^                                                  ^^ parity
             ^^ <- flagbyte of header (always 0)      ^^ flagbajt of body
17-byte header  ^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^
(the same as in TAP)              parity of header ^^    ^^^^^ first two
                                                              bytes of rom
```

The parity (or checksum, checkbittoggle) is the bitwise XOR of all bytes including the flag byte - the same as in TAP files.

3. MFC snap Convertor

Program slouží pro práci s formáty:

.MFC .SNA .Z80 .__S

Levá strana v programu určuje zdrojové soubory (=co a **odkud** se převádí) a **pravá** cílový adresář (=**kam** se bude převádět). Označit lze samozřejmě i více souborů.

Nepůjde-li nějaký soubor převést, budete na to upozorněni chybovou hláškou (bude tam i důvod nepřevedení). Takovéto chybné soubory pak budou mít (ve výstupu) nulovou délku, nebo budou chybět úplně (záleží na druhu chyby). Po skončení konverze zůstávají problémové soubory označeny!

Zaškrtávací pole *High speed* nastavte (zaškrtněte), když budete převádět velké množství souborů (více než 30) najednou. Volba totiž způsobí, že se během konverze nebudou vypisovat v pravém okně výstupní soubory (musí-li se zobrazovat posuvník, tak to dlouho trvá... ...vyzkoušejte, uvidíte).

POZOR!

1. Soubory, které jsou zobrazeny, se automaticky "neREFRESHují". Tzn, že když něco v nějakém Commanderu odmažete, program si pořád bude myslet, že ty soubory tam jsou. K aktualizaci adresářů a souborů slouží tlačítko vpravo dole - *Refresh all file lists,* anebo klávesa F5.

2. Soubory v pravém okně mohou být bez varování přepsány (pokud má výstupní soubor stejné jméno jako ten, co už v pravém okně je, tak bude (s)prostě konvertorem přepsán).

3. Zjistil jsem, že mnoho snapů je ve verzi 128K, ale přitom jde o normální 48K hry a 128K vůbec nepotřebují. Proto máte možnost si výsledný .mfc snap ještě analyzovat a když tak si z 128K verze udělat 48K.

4. Ne každý snap, který je spustitelný na PC v emulátoru, půjde rozběhnout na pravém ZXS. Důvodů je více a vysvětlovat je zde nemá smysl. Takže s tím počítejte.

5. Kdo narazí na nějaký snap, který mu v pohodě poběží v emulátoru, ale tento konvertor jej z nějakého důvodu nebude chtít na .MFC převést, ať se mi určitě ozve! mts.zxs@tiscali.cz

6. Po dvojkliku na soubor v LEVÉM okně se vypíší jeho informace (funguje pouze pro .MFC a .Z80).

Možnosti konverze:

Z80 --> MFC

Formát .z80 existuje v mnoha verzích a každá verze ještě v mnoha variantách (věřte, že je v tom neuvěřitelný bordel). Tento konvertor jsem kvůli .z80kám hodně často aktualizoval... Současná verze by si už měla umět poradit s DRTIVOU VĚTŠINOU těchto snapů.

Přidal jsem i podporu SAMa (snap pro SAMa se "okleští" na 48K verzi spustitelnou na ZXS).

SNA --> MFC

Lze převádět 48K i 128K verze těchto snapů.

Pozn.: 48K verzi SNA umí MDOS File Commander spouštět i přímo (bez nutnosti konverze na .MFC).

__S --> SNA

__S je formát SNAPu z D40/80. Bohužel jde o jeden z nejhorších typů snapů, jaký jsem kdy viděl. Neuchovává některé důležité informace (např. mód přerušení, registr R, Program Counter). Proto (jak už se Vám určitě stalo) se některé hry/programy po snapnutí zhroutí či

zatuhnou. Ovšem nevěšte hlavu, protože i takový špatný snap lze převodem do .SNA někdy zfunkčnit...

Při konverzi máte možnost nastavit si u každého převáděného souboru mód přerušení (0,1,2). Tohle totiž MDOS do snapu neukládá a pak (při spuštění) se to pouze odhaduje z registru I (no a když se netrefí, tak hra spadne...). Program Vám nabídne výchozí hodnotu módu přerušení (té odpovídá stav registru I). Vy ji buď potvrďte nebo změňte (když hra padá, je nutné místo výchozího IM 2 dát IM 1).

Pozn.: Pokud máte zaškrtnuto *high speed*, nebude se program na mód přerušení ptát a nastaví ten výchozí (tj. takový, co by nastavil i MDOS).

MFC --> MFC

Nebojte, nezbláznil jsem se... Tady nejde o konverzi, ale spíše o analýzu a znovusestavení .MFC souboru. Testuje se, zda .MFC není náhodou poškozen (XOR test) a zda by u 128K verze snapu nestačilo 48K. Také se aktualizuje spouštěcí rutina (do hlavičky se nahraje aktuální "mfcrun.__b"), což je životně důležité v případě, že má snap spustit MDOS File Commander a né nějaký PC emulátor či jiný ZXS operační systém.

Bacha, že když bude zdrojový i cílový adresář stejný, soubory se Vám budou přepisovat!!

.MFC formát

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)

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
```

Tento kód (mfcrun.__b) je napsán univerzálně pro jakýkoliv „systém", který snap bude rozbíhat (4 bajtový prostor pro instrukci, která má za úkol přestránkovat do normální ZXS ROM, by měl bohatě dostačovat). Kód by měl být přítomen v každém .MFC souboru. Pokud tam nebude (offset 34 = nula), MDOS File Commander jej nespustí, ale elegantně „vytuhne" s jedoucím borderem jako signálem, že je něco špatně.

3. MFC file cutter

Program slouží pro "vystřihnutí" dat z jednotlivých souborů.

Sem tam je totiž potřeba dostat z .MFC úvodní obrázek či data z konkrétní stránky, nebo z .d_0 dostat jen data bez 1024 bytes dlouhé hlavičky atd. atd. Možností použití je spousta. Od verze 2.9 umí program také 2 soubory binárně porovnat :)

Levá strana v programu určuje zdrojové soubory (=co a **odkud** se převádí) a **pravá** cílový adresář (=**kam** se bude převádět). Označit lze samozřejmě i více souborů.

Zaškrtávací pole *High speed* nastavte (zaškrtněte), když budete převádět velké množství souborů (více než 30) najednou. Volba totiž způsobí, že se během konverze nebudou vypisovat v pravém okně výstupní soubory (musí-li se zobrazovat posuvník, tak to dlouho trvá... ...vyzkoušejte, uvidíte).

POZOR!

1. Soubory, které jsou zobrazeny, se automaticky "neREFRESHují". Tzn, že když něco v nějakém Commanderu odmažete, program si pořád bude myslet, že ty soubory tam jsou. K aktualizaci adresářů a souborů slouží tlačítko vpravo dole - *Refresh all file lists,* anebo klávesa F5.

2. Soubory v pravém okně mohou být bez varování přepsány (pokud má výstupní soubor stejné jméno jako ten, co už v pravém okně je, tak bude (s)prostě konvertorem přepsán).

3. Tlačítko **fc - *files compare*** slouží pro binární porovnání označeného souboru v levém okně s označeným souborem v pravém okně.

Možnosti konverze:

. --> BIN

Ty dvě žluté pole vlevo slouží pro zadání číselných hodnot. V prvním (**levém**) jde o počet počátečních **bytes, které budou ignorovány** (0=nic se neignoruje) a v druhém (**pravém**) o **délku výstupního souboru** (0=dokud není konec vstupního souboru). Kdybyste zadali do obou nulu, výstupní soubor bude naprosto stejný jako vstupní (vůbec nic byste nevystřihli).

Hodnotami 512 (vlevo) a 6912 (vpravo) u souboru .mfc dosáhnete .bin souboru o délce 6912 bytes (tj. úvodní obrázek z jednotlivých snapů). 512 znamená že prvních 512 bytes se u vstupního souboru přeskočí (u .mfc je tam hlavička).

Kdybyste např. chtěli dostat holá data z .d_0 souboru, musíte zadat 1024 (hlavička .d_0) a 0 (jako "až do konce vstupního souboru").

Tím třetím žlutým editačním polem vlevo máte možnost změnit si příponu výstupního souboru (default je *bin*).

Žlutá pole vpravo Vám ukazují XOR hodnotu výstupního souboru. První číslo je v dekadickém tvaru, druhé (symbolizuje #) v hexadecimálním.

. --> DEF

Od verze 4.3 (květen 2020) umí program také vytvořit tzv. GENS soubor (jako import do Promethea). GENS soubor bude tvořen pouze instrukcemi „defb" a jednotlivými bytes ze vstupního souboru. Hodnoty mohou být buďto v dekadickém nebo hexa tvaru. Konverzi vyvoláte zadáním výstupní přípony souboru jako ***defb*** anebo ***defb#*** (vše malými písmeny!)

4. ZX Screens 2.03

(c) únor 2002 - červenec 2004 Pavel Plíva, Pavero software

Program slouží k prohlížení titulních obrazovek her pro ZX-Spectrum, jedná se o soubory s příponou SCR. Dále aplikace dokáže načíst obrázek ve formátech BMP, GIF a PNG.

Forma licence: Freeware

Platforma: Windows 95/98/ME/2000/XP

Kontakt: pavero@seznam.cz
www.zx-spectrum.cz

Holandský překlad: Jeroen Adolfse

Načtené obrázky lze libovolně zvětšovat, stačí pouze měnit velikost okna aplikace. Poslední verze přináší zrychlený algoritmus načítání obrázků, kopírování obrázku do schránky, jejich ukládání ve formátech SCR, BMP, GIF a PNG, postupné (automatické) prohlížení všech obrázků nacházejících se v aktuálním adresáři a hromadný převod obrázků z jednoho formátu do druhého (SCR2GIF, SCR2PNG, SCR2BMP, GIF2SCR, GIF2PNG, GIF2BMP, PNG2SCR, PNG2GIF, PNG2BMP, BMP2SCR, BMP2GIF a BMP2PNG).

Menu Soubor

Otevřít (F2) - otevření obrázku ve formátu SCR, GIF, PNG nebo BMP.

Uložit jako (F3) - uložení obrázku ve formátu SCR, GIF, PNG nebo BMP.

Hromadný převod - pomocí této položky můžete mezi sebou převádět jednotlivé formáty obrázků. Stačí vybrat soubory, které chcete převést, určit kam se mají uložit a vybrat zdrojový a cílový formát.

Předchozí (F6 nebo '-') - zobrazení předchozího obrázku v aktuálním adresáři.

Další (F7 nebo '+') - zobrazení následujícího obrázku v aktuálním adresáři.

Automatické prohlížení (F8) - obrázky v aktuálním adresáři se budou zobrazovat automaticky.

Časový interval - čas v sekundách, kdy se zobrazí další obrázek, pokud je zaškrtnuto automatické prohlížení.

Vymazat (F5) - vymaže právě načtený obrázek.

Kopírovat do schránky (Ctrl + C) - zkopírovaní obrázku do schránky Windows

Asociace SCR obrázků - pokud je zaškrtnuta tato volba a vy kliknete v Průzkumníkovi pravým tlačítkem myši na libovolný SCR obrázek, objeví se plovoucí nabídka, ze které můžete vybrat položku *Open by ZX Screens*. Nejedná se tedy o přímou asociaci (dvojklik levým tlačítkem na daný soubor), protože příponu SCR mají ve Windows registrovánu šetřiče obrazovky.

Jazyk - aktuální jazyk aplikace.

Konec (Alt + F4) - ukončí aplikaci.

Menu Vzhled

Průhledně - průhledné zobrazení obrázku.

Černobíle - u souborů SCR se zobrazí pouze černobílá složka.

Pouze 8 barev - tmavší odstíny se převedou na světlejší, soubory SCR tak budou mít pouze 8 barev.

Invertovat barvy - prohození barev.

Paleta barev - možnost definování vlastních barev.

Rozměry - přednastavené rozměry okna.

Menu Nápověda

Soubor s nápovědou (F1) - zobrazí se tento dokument.

O aplikaci (Alt + F1) - zobrazí se stručné informace o aplikaci.

Jak přidat nový jazyk?

Všechny texty, které aplikace obsahuje, najdete v souboru *Language.ini*. Pokud chcete přidat nový jazyk, zkopírujte sekci *English* na konec tohoto souboru a změňte její jméno (např. na *Deutch*). Potom přeložte pravou část jednotlivých řádků (tu, která se nachází za rovnítkem). Nakonec ještě přidejte jméno nového jazyka do sekce *Languages* a v sekci *Language* přiřaďte proměnné *Current* číslo jazyka. Výsledek by měl být následující:

```
[Language]
Current=3
[Languages]
1=English
2=Česky
3=Deutch
....
[Deutch]
Proměnná1=Překládaný text
....
ProměnnáX=Překládaný text
```

mts.zxs@tiscali.cz