MDOS 2.1

operační systém pro D40/80/Kompakt s řadičem
wd37c65c, GM82c765b, Intel 8272, a připojeným divIDE

Meyerův zákon:

"Je velmi jednoduché něco zkomplikovat, zato bývá značně komplikované něco zjednodušit."

Murphy

Obsah

1. Na začátek F.A.Q.

Co je to MDOS 2.1?

MDOS 2.1 je opravený a doplněný originální MDOS 2.0 od Didaktiku. Vytvořil jsem jej jako reakci na chyby, které jsem v původním MDOSu objevil za dlouhé roky používání a rovněž jako reakci na IDE řadič divIDE od Ziloga (Zilog, v tomto manuálu to bude osoba z ČR, autor divIDE, nikoliv ta nám všem známa firma, která vyrobila nejlepší procesor všech dob…).

Co lze získat přechodem na MDOS 2.1?

Především získáte stabilnější a lepší systém než byl MDOS 2.0, 2.1 můžete chápat jako takový update. Pokud nemáte divIDE, získáte jen na opravených chybách a vylepšeních (snap, NMI menu, LIST, CAT...). Pokud ovšem divIDE máte, pak se Vám otevře brána pracovat s jakýmkoliv IDE zařízením jako je harddisk, cd-rom, flash karty, zip mechaniky, atd. Podmínkou ovšem je LBA přístup, který neumí jen opravdu hodně staré IDE vykopávky. Práce bude naprosto stejná jako jste byli zvyklí u diskety. Výhody IDE oproti disketě jsou Vám doufám jasné – mnohonásobně rychlejší práce (čtení/zápis), delší životnost dat, větší jistota a stabilita.

Jsou třeba nějaké HW úpravy?

Ne. Naprosto nic. Stačí pouze výměna/přeprogramování původní ROM. I když, i toto vlastně lze považovat za jakýsi hardwarový zásah, protože asi tu původní ROM budete muset dostat ven, pak tam rovnou už jistě dáte patici a nakonec nasadíte novou EPROM.

MDOS 1.0 vs 2.0

Jak jistě víte, MDOS 2.0 vznikl v důsledku změny FDD řadiče. MDOS 2.1 je použitelný pouze na řadičích wd37c65c, GM82c765b, Intel 8272, na kterých běžel MDOS 2.0. Majitelé MDOSu 1.0 mají smůlu. Mohou však využít MDOS 3, který spáchal Sweet (http://ci5.speccy.cz).

2. Chyby MDOSu 2.0

Původně jsem zde chtěl detailně popsat všechny chyby, které jsem našel a jakým způsobem jsem je opravil. Ovšem, obyčejný uživatel by to asi stejně moc nepochopil a profíkovi jsem zpřístupnil zdrojový kód MDOSu 2.1, takže se touto kapitolou vůbec nebudu zatěžovat…

3. Vylepšení oproti MDOSu 2.0

SNAPSHOT

MDOS 2.1 rozeznává 2 druhy SNAPSHOTu – ***klasický*** bez uchování hodnoty z #3EF7, a ***nový*** kde hodnota z #3EF7 je uložena na #3FE7. MDOS 2.1 ukládá vždy v novém formátu. Samozřejmě že nahrát (spustit) lze snapy oba. Program si druh snapu detekuje sám a uživatel prakticky nikdy nic nepozná. A pokud budete nový snap nahrávat pod starým MDOSem (1.0, 2.0), tak to taky v pohodě půjde, resp. pouze #3EF7 zůstane beze změny, takže pokud ji snap potřebuje, pak máte smůlu.

Celé to funguje díky tomu, že na všech MDOSech je ukládán stav přerušení okamžitě po instrukci *LD A,I*. A jelikož víme, že ona instrukce způsobí vynulování flagu N(bit1) a H(bit4), lze starý snap poznat se 100% jistotou! Nový snap používá N(bit1) flag, je-li tam 1, při spuštění se obnoví i hodnota na adrese #3EF7.

Čistě uživatelsky to znamená, že půjde uložit a bez problému spustit více programů, především ty pro D40/80. Vyzkoušejte například CRACKSHOT2 od Proximy.

Dále jsem na SNAPSHOTech upravil číslování. Ve starých MDOSech hrozilo, že snap se stejným číslem bude přepsán. Ve 2.0 to již opravili tak, že se čísla zvedala až do doby, dokud snap nebylo možné na disketu uložit aniž by se přepsal nějaký starý (99 snapů o velikosti 49.280 bytes není možné na disketu narvat, proto to celkem hezky fungovalo). Jelikož ale nyní v MDOSu 2.1, má FOTOSHOT (viz. dále) velikost 6912 bytes (těch už tedy na disketě může být víc jak 99), udělal jsem v číslování úpravu. Pokud se dosáhne čísla 99, pak místo přetečení zpět na "00" bude "xx" a takový snap/foto se bude normálně v případě potřeby přepisovat.

Ve starých MDOSech nastával také často problém se spadnutím snapu, pokud byl MDOSem chybně detekován mód přerušení. Nyní si může uživatel mód přerušení vynutit, a to tak že přidržíte *Caps shift* (vynutíte IM 1) nebo *Symbol shift* (vynutíte IM 2).

OUT 32765,16

Tento out se provede při každém resetu (i usr 0, non seek). Na 128K mašinách způsobí přestránkování do základní stránky 16.

BOOT

Boot sektor nyní obsahuje 3 náhodné bytes. Také je tam uložena informace "Formated with MDOS_2 (MTs edition)".

FYZLOG a LOGFYZ

Nahradil jsem je svými, bleskově rychlými. Viz. manuál k MFC.

Fonty

Nikdy jsem si nezvykl na ty hnusně tučné fonty na Didakticích. Při formátování se tedy budou zobrazovat klasické fonty ze ZXS 48.

LIST *, LIST !

Příkaz LIST * už nikdy nebude prosvěcovat mechaniky a vypisovat jména disket.

```
MDOS Release: 2.1 (27.11.06)
(c) 93-06 Didaktik & MTs

Drives Defined   : A, B, C, D
Drives Installed: A, B, C, D
Current Device   : C

Length of Program  : 0
Length of Variables: 0

Top of RAM : 65367
Free memory: 41513
```

Nově přidán příkaz LIST !, který vykoná IDE příkaz "Identify device". Na obrazovku se vypíší připojené master a slave zařízení.

```
IDENTIFY DEVICE (#EC)

master:

Transcend     128M

sn: SSSI128M04Z13A21419A
firmware: 1.1


slave:

WDC WD102AA-00BAA0

sn: WD-WMA3910604622?????
firmware: 10.09K11
```


CAT, CAT +

Příkaz CAT bude vždy vypisovat i skryté (hidden) soubory.

CAT + je trošku vylepšený. Rovněž vypisuje i skryté (hidden) soubory, dále je u každého souboru, jeho délka, startovní adresa a body flag.

```
Directory of kkk                fdd

P ide                 168 <00010> ---
B ide25              1555 <49152> ---
P run                  53 <00001> ---
B mfc135            29340 <29800> ---
B u@edit70           2816 <55296> ---
B u@flash8            167 <55296> ---
B u@xor6              200 <55296> ---
B xxrom_divi         4608 <00000> ---
P ide-                125 <00010> ---
B ide25-             1555 <49152> ---
P ii                  126 <00010> ---
P prom                 54 <09999> ---
B promdiskam        20124 <24000> ---
B p 98              17969 <24939> ---
B p xx               2629 <00000> ---
B p yy                 66 <00000> ---
B p vv                223 <00000> ---
S SNAPSHOT00        49280 <00000> ---
B +ide               6912 <16384> ---
```

NMI menu

Do NMI menu se lze dostat pomocí tlačítka SNAP. Signalizují ho jedoucí černé a modré pruhy (reg. I = 63, tj. snapnutý kód běží pravděpodobně v IM 1) nebo černé a červené pruhy (reg. I <> 63, pravděpodobně IM 2) v borderu. Upozorňuji, že tlačítko SNAP (u Didaktiku Kompakt CS+šipka vlevo+šipka vpravo) nefunguje, pokud je aktivní DROM (ROM D40/80, tedy MDOS).

Po spuštění NMI menu se vždy umlčí AY a vypnou disketové mechaniky.

Funkční klávesy:

Q	quit (návrat do přerušeného programu)
SS+Q	reset AY a celého počítače
SS+CS	reset AY a celého počítače (non seek)

B **pokus** o výskok do Basicu
S save SNAPSHOTnn (49280 bytes) a quit
F save FOTOSHOTnn (6912 bytes screen) a quit
G přenos VRAM2 do VRAM1, save FOTOSHOTnn a quit
V přepnutí do VRAM1, tj. OUT 32765,16
M multifunkční *command line* (MOVE, POKE, OUT)
D spuštění DevastAce (musí být v ROM DivIDE !)

M – multifunkční *command line*

Vyvolání příkazového řádku může i chvilku trvat (max. však 2 sekundy). Je to z toho důvodu, že program se snaží najít někde v paměti blok stejných bytes, aby tam mohl dočasně přesunout poslední dva řádky z obrazovky. A aby to trvalo max. ty dvě sekundy, hledá se pouze blok s bajty 0 nebo 229. Pokud se nepodaří "volné" místo najít, při opuštění příkazového řádku budou ty poslední dva řádky obrazovky bohužel zničeny (neobnoví se do původního stavu).

Do příkazovém řádku je možno zadat pouze tato písmena:
0, 1, 2, 3, 4, 5, 6, 7, 8, 9, a, b, c, d, z (:), n (,), m, s, f, w
ENTER slouží pro odeslání příkazu, CS+1 pro návrat do NMI menu.

Příkazový řádek slouží pro zadání příkazu MOVE, POKE nebo OUT.

MOVE

Max. 12 znaků, viz. syntaxe příkazu níže. Klíčové slovo *MOVE* se nezadává. To, že jde o MOVE se pozná podle prvních dvou zadaných znaků – musí jít o určení *drive* (a,b,c,d) a dvojtečku. Pokud po odeslání příkazu border zčervená, zadali jste nějakou blbost nebo došlo k chybě.

POKE

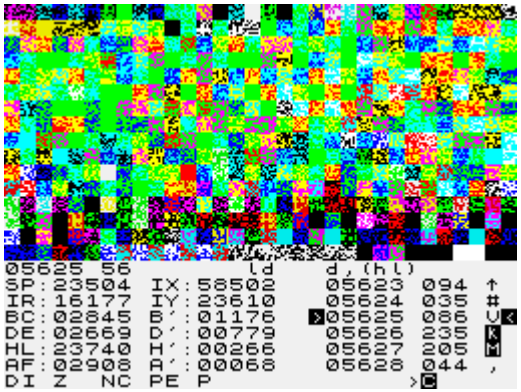
Přesně 9 znaků, maska: ***nnnnn , nnn***
(Příklad: 22528,056 - provede POKE 22528,56)
Úspěšné provedení tohoto příkazu signalizuje zelený border.

Pozn.: POKE 15968,128 aktivuje *SRAM debug mode* - po stisku F v NMI menu pak dojde k přenesení diskové SRAM (2048 bytes) na adresu 16384.

OUT

Přesně 9 znaků, maska: ***nnnnn : nnn***
(Příklad: 00191:144 - provede OUT 191,144)
Úspěšné provedení tohoto příkazu signalizuje zelený border.

D – DevastAce



Po stisku klávesy D dojde ke spuštění obrazovkové DevastAce. Devastace zabírá 4 kB a nebylo možné ji narvat do samotného MODSu 2.1. Proto musí být nahrána v ROM divIDE (*u EEPROM může pro flashnutí využít utilitu pro MFC s názvem u@flash)*. Touto volbou v NMI jsem v podstatě nahradil program CRACKSHOT2 od Proximy. DevastAce se totiž spustí už s přednastavenými registry (hodnoty jsou stejné jako v okamžiku stisku tlačítka SNAP) a první vypsanou adresou (je to ta, na kterou by se skočilo při rozbíhání snapu). Pozor však na případ, kdy snapnutý program má zásobník někde mezi 0 a 23295 ! To se DevastAce nastaví na adresu 0.

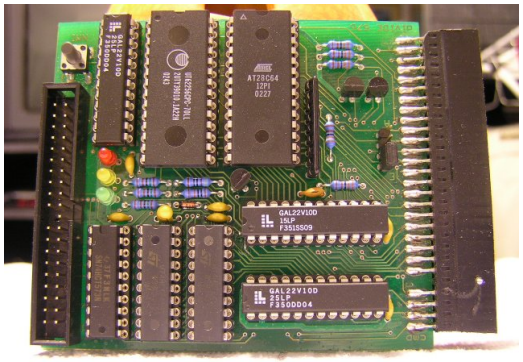
Při přetahování devastace z ROM divIDE se testuje, zda není poškozena (na adrese 8191 mám uložen XOR). Pokud není v pořádku provede se non seek reset.

Podpora DivIDE

Takže se dostávám k tomu nejzajímavějšímu. divIDE způsobilo takovou malou revoluci a neobešlo se bez křiku a hádek.

Asi prvními kdo se začali harddiskem zabývat a dotáhli to do nějakého zdárného konce byli PVL a TRITOL. Mrkněte třeba do nějakého staršího ZX-Magazínu. PVL se také zabýval vylepšením MDOSu. S Tritolem totiž měli úpravu, kterou si rozšířili SRAM v D80 a umístili si tam vlastní verzi MDOSu, kterou provozovali. Vyvíjeli také operační systém MATRIX, který měl umožnit práci s D40/80, harddiskem, cd-rom atd. Ovšem ten systém nikdy nedodělali.

Pak se objevil Zilog a navrhl zcela nový řadič IDE pro ZX Spectrum. A tak přišlo divIDE. Nakonec byl Zilog dokopán i k tomu, aby bylo jeho zařízení kompatibilní s tím, co tenkrát splácali TRITOL a PVL.



Koukněme na divIDE trošku podrobněji. Od začátku bylo koncipováno jako samostatné zařízení, které nahradí všechny dosud používané zařízení jako kazeťák, D40/80, MB-02, Betadisk atd. divIDE obsahuje 8 kB paměti ROM (flash ROM) a 32 kB paměti RAM. RAM je teoreticky rozšířitelná na 512 kB, ale Zilog o rozšíření nechce ani slyšet. Jakmile se divIDE připojí k počítači, automaticky mapuje určité body (adresy) a když na ně nějaký program doskáče, přestránkuje do své vlastní ROM (mj. přesně takhle funguje i D40/80).

Mapované vstupní body (přistránkuje se ROM divIDE):

adresa (dec)	adresa (hex)	1 bytes zpoždění
0	#0000	ano
8	#0008	ano
56	#0038	ano
102	#0066	ano
1222	#04c6	ano
1378	#0562	ano
15616-15871	#3d00 - #3dff	ne

1 bytes zpoždění "ano" znamená, že první byte se na dané adrese provede ještě v klasické ROM a až ten druhý se pak provede v ROM divIDE.

1 bytes zpoždění "ne" znamená, že se ihned přistránkuje ROM divIDE a začne se provádět kód, který tam leží.

Mapované výstupní body:

adresa (dec)	adresa (hex)	1 bytes zpoždění
8184	#1ff8	ano
8185	#1ff9	ano
8186	#1ffa	ano
8187	#1ffb	ano
8188	#1ffc	ano
8189	#1ffd	ano
8190	#1ffe	ano
8191	#1fff	ano

1 bytes zpoždění "ano" znamená, že první byte se na dané adrese provede ještě v ROM divIDE a až ten druhý se pak provede v klasické ROM.

ROM divIDE leží v oblasti 0 - 8191
RAM divIDE leží v oblasti 8192 - 16383

Stránkuje se na portu 227 (#e3).

Takže jsme si tedy divIDE hezky popsali a už Vám musí být jasné, že takhle to s naší D40/80/Kompaktem nikdy fungovat nebude, protože disketovka mapuje adresy také. Takže by se obě zařízení akorát přetlačovaly a nefungovalo by ani jedno.

Možnosti tu ale jsou, a to hned dvě.

Za prvé, zakázat to mapování adres na D40/80/Kompaktu a využít na 100% pouze divIDE, tj. nacpat do jeho ROM a RAM systém a pracovat. Touto cestou jdou všichni, které znám - Baze (fatware), Tritol (demfir), Sweet (MDOS3). Disketové mechaniky a diskety již dávno zahodili a provozují pouze IDE.

Za druhé, zakázat to mapování adres na divIDE a nechat D40/80/Kompakt dělat svoji práci. divIDE tak bude zcela mrtvé, ale jeho porty fungovat budou. Hardwarové zásahy jsou zcela minimální (na divIDE se vytáhne pouze jeden switch a na D40/80/kompaktu se přeprogramuje EPROM). Touto cestou jdu já a můj MDOS 2.1. Abych byl upřímný, já mám Kompakt, takže když ze sběrnice vytáhnu divide, chci mít funkční disketovku jako dříve; když na sběrnici divIDE nasadím, chci zase mít okamžitě k dispozici disketovku i harddisk. Nikde nic zapojovat/pájet/přeškrabávat nehodlám! To jest "*vše, na co jsem zvyklý, mi zůstane, ale mám možnost pracovat kromě disket v D40/80 i s "disketami" na harddisku*."

Ještě k těm switchům (jumperům) co má divIDE. Jumper **A** musí být nasazen jen v případě ZXS +2A nebo +3. Jumperem **E** se zakazuje zápis do ROM a povoluje mapování paměti. Pro MDOS 2.1 musí být jumper E vytažen, protože mapování paměti nechceme (povolen zápis do ROM divIDE sice taky nechceme, ale toto bohužel neovlivníme, takže se s tím nezbývá než smířit, nebude nám to nijak vadit)!

MDOS 2.1 umí pracovat s IDE médii, které splňují tyto 3 základní předpoklady:

- **LBA přístup**
- **kapacita max. 128 GB**
  (resp. dokáže se obsloužit sektor s číslem 0 až 2^{28}-1)
- **512 bytes na sektor**

Dále je třeba souhlasit s už s definovaným (Sweetem) formátem virtuální diskety:

- **1693 sektorů na 1 disketu** (infosektor + formát 94x09)

Disketa může mít libovolně menší formát (klidně 30x07 jednostranně), ale na IDE médiu bude mít stejně rezervováno těch 1693 sektorů. **Basic formátuje standardně na 80x09 oboustranně,** proto pokud chcete disketu s maximální kapacitou, musíte použít něco jiného (MFC, TOOLS).

Pozn.: Infosektor MDOS 2.1 k ničemu nevyužívá, ale z důvodu kompatibility s MDOSem 3 jej vytváří.

Po vzniku divIDE a při jeho testování se zjistilo, že jsou problémy se zákmity.

Stručně řečeno, že komunikace s IDE není bezchybná a objevují se chyby při čtení nebo zápisu. Toto je samozřejmě katastrofa. Zilog to řešil více verzemi divIDE a také obsahem GALu. Ukázalo se také, že problémy se zákmity mají většinou pouze Flash karty, a že harddisky v drtivé většině fungují zcela perfektně. Pokud uživatel narazí na problém se zákmity, pak se doporučuje vyrobit si buď speciální kábel, anebo provozovat

problematické zařízení jako master a na slave dát plnohodnotný harddisk, který tam bude jen jako zátěž. Pro detekci zákmitů doporučuji svůj prográmek "ide25", který mám na webu.

Protože já jsem celkem paranoidní a potřebuji 100% jistotu, že se mi data zapíší/přečtou správně, musel jsem do MDOSu zabudovat detekci zákmitů. Funguje to tak, že pokud nastane zákmit, pak se operace (read/write/identify) provádí znova a to tak dlouho, dokud se neprovede bezchybně. Abych svou paranoiu ještě umocnil, tak jsem po operaci "zápis na disk" přidal ještě sekci na verifikaci. To znamená, že každý zapsaný sektor se ihned zase přečte zpět a bajt po bajtu porovná pamětí, která se měla uložit. Tímto zápis trvá jednou tak déle než čtení, ale radši tak, než riskovat…

Rychlost čtení je +- 16 kB/s (+- 33 sektorů za sekundu)
Rychlost zápisu je +- 8 kB/s (+- 16 sektorů za sekundu)

Pozn: Oblast 15872-16383 se při zápisu neverifikuje, protože těchto 512 bytes se mění (jde o diskovou SRAM, ve které jsou proměnné).

Příkaz MOVE

MOVE "*drive*:" *drive* může být a,b,c,d

Tohle je klasika. Slouží k výběru mechaniky, se kterou se následně budou provádět veškeré diskové operace.

MOVE "*drive*:*dN*,*nnnnn*[,*w*]"

Tak na tomto příkazu stojí celá podpora virtuálních disket (a tedy i DivIDE). Slouží totiž k "napíchnutí" konkrétní virtuální diskety na nějakou *drive*.

parametr ***d***

Určuje zařízení a je povinný

f	fdd (drive jako po resetu)
s	slave
m	master

parametr ***N***

Určuje StartSektor, tj. sektor, ve kterém leží infosektor PRVNÍ virtuální diskety. Je povinný.

0	uživatelem definovaný (je v SRAM),
1 až 4	číslo primární oddílu na HDD (start se tedy zjistí z MBR),
d	start na sektoru č. 2 (default)

Pozn.: Oddíl nemusí mít Sweetovu signaturu! Nepodaří-li se boot načíst, tak border zčervená a nastaví se default hodnota. Každá *drive* může mít klidně "napíchnutou" disketu z úplně jiného oddílu (N) či disku (d).

Pozn 2: pokud za N dáte "0", bude to stejné jako byste zadali "d", protože do SRAM se při resetu dává default hodnota, tj. sektor č. 2. Tu nulu byste ale měli používat opravdu jen v případech, kdy chcete využívat svůj vlastní start sektor.

parametr ***nnnnn***

Určuje číslo virtuální diskety. Těch může být 65536 (0 až 65535). Číslo nemusíte psát vždy jako 5ti ciferné (místo 00011 můžete napsat jen 11).
00000

parametr ***W***

Není povinný, určuje write protect (ochranu proti zápisu).

,w write protect ON (disketa je chráněna proti zápisu)
cokoliv jiného write protect OFF (default)

Pozn.: Sweet (MDOS 3) má tuto informaci v infosektoru. MDOS 2.1 infosektor ignoruje!

Příklady:

MOVE "b:md,99"	***B, master, start 2, disketa 99***
MOVE "b:fdd"	***B, nastavit zpět FDD jako byla po resetu***
MOVE "b:f"	***B, nastavit zpět FDD jako byla po resetu***
MOVE "c:s4,50,w"	***C, slave, oddíl 4, disketa 50, zákaz zápisu***

Nové vstupní body

MDOS 2.1 dovoluje přímý přístup na IDE zařízení. Slouží k tomu nový vstupní bod 14333, #37FD. Délka sektoru je stejná jako u diskety, tj. 512 bytes.

14333, #37FD,REWR_LBA čtení/zápis sektoru

IN:

reg BCDE	LBA sektor (BCDE je fiktivní 32bitový registr)
reg HL	adresa kam načíst, příp. odkud zapsat
reg A	operace (0=čtení, 1=zápis, 255=identify)
reg IX	klíčová je pouze hodnota na (IX+11), stačí tyto bity:

bit 4	0 – master 1 – slave
bit 5	0 – write protect OFF 1 – write protect ON
bit 6	0 – LBA přístup OFF 1 – LBA přístup ON
bit 7	0 – vždy musí být nula !!!

bit 7 na (IX+11) nemusí být nula, ovšem v tom případě musí IX+11 ukazovat do RAM (já už si tam ten bit 7 vynuluju sám). Jinak (bit 7 je nula) můžete klidně použít na IX+11 i oblast z ROM.

OUT:

reg C	kód chyby

128	not ready
64	write protected
16	sector not found
8	CRC error
0	žádná chyba (načteno/zapsáno OK)

Pozor, že MDOS 2.1 neumí obsluhovat disky, které neumí LBA (ačkoliv to nastavení bitu 6 umožňuje) !!

4. Systémové proměnné

Klasika na 15872, #3E00

offset	bit	význam	
0	bit 0	0 - mechanika nepřipojena 1 - připojena	
	bit 1,2,3,4	(nepoužito)	
	bit 5	0 - strana 1 = 1 (správně naformátovaná disketa) 1 - strana 1 = 2 (špatně naformátovaná disketa)	
	bit 6	0 - vše OK (SEEK není nutný) 1 – došlo k chybě při práci s disketou (nutný SEEK)	
	bit 7	0 – mechanika zastavena přes 9526 1 – mechanika se točí (LED svítí)	
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)	
8,9,10		LBA 0-23 (start sektor vybrané virtuální diskety)	DIVIDE
11	bit 0-3	LBA 24-27 (start sektor vybrané virtuální diskety)	
	bit 4	0 – master 1 – slave	
	bit 5	0 – write protect OFF 1 – write protect ON	
	bit 6	0 – LBA přístup OFF 1 – LBA přístup ON	
	bit 7	0 – jde o FDD 1 – jde o IDE	

Offset 8-11 se po příkazu MOVE "*drive*:*dN*,*nnnnn*[,*w*]" XORuje a výsledek se ukládá do kopie (viz dále) na offset 11. Při každé operaci (R/W) se pak XORuje znova a není-li výsledek stejný, vyhodí se *Internal error*. Toto je taková bezpečnostní pojistka.

Vše co je červeně je nové, příp. staré ale pojato trochu jinak.

Kopie na 16128 (15872+256)

offset	bit	význam	
0-7		jako po resetu (používá se pak po *MOVE "drive:fdd"*)	DivIDE
8,9		číslo vybrané virtuální diskety (0-65535)	
10		volno (nepoužito)	
11		výsledek XORu	

Další pomocné proměnné

15968,#3E60,DEBUG 1

Veškeré ladící tisky, které tato proměnná způsobovala u MDOSu 1.0, byly u 2.0 zrušeny. Použil jsem ji tedy použil pro své vlastní účely. POKE 15968,128 (resp. bit7 nastavený na 1) aktivuje SRAM debug mode, který pak po stisku S nebo F v NMI přenese

na adresu 16384 diskovou SRAM a uloží to (snap, nebo foto, záleží co jste stiskli). Po přenosu SRAM se bit7 zase vynuluje.

15976,#3E68,OPER 1
Při každém volání DREAD, DWRITE, DFORMA požadující IDE zařízení se zde ukládá číslo operace (#20 pro read, #30 pro write).

15977,#3E69,ADR 2
Při každém volání DREAD, DWRITE, DFORMA požadující IDE zařízení se zde ukládá adresa kam/odkud se bude číst/zapisovat.

15997,#3E7D,NMIEXIT 1
Používá pouze NMI menu.

16106,#3EEA,FORMAT 1
MDOS 2.1 už při prvním zavolání DFORMA zformátuje **celou** virtuální disketu. Nenulová (1-255) hodnota při volání DFORMA značí "formátuj teď celou disketu a do proměnné nastav nulovou (0) hodnotu". Zpět na nenulovou hodnotu se nastaví při každém skoku na 9526 (vypnutí motoru). Formátování znamená **vyplnění všech 1693 sektorů hodnotou 229**.

16124,#3EFC,SEC1USER 4
Při resetu se zde uloží číslo 2 (ve 32bitové podobě), které značí LBA sektor. Tato proměnná má velký význam pokud parametr N u příkazu MOVE bude "0" (ascii nula). Lze si tak ručně nastavit začátek oddílu na HDD (CF).

16128,#3F00,MECHALL 48
Parametry mechaniky a diskety (jejich kopie).

16176,#3F30,SECMAX 2
Zde je uložen počet sektorů každé virtuální diskety. Je to 1693 (viz. Swetova definice).

16178,#3F32,LOGSEC 2
Při každém volání DREAD, DWRITE, DFORMA požadující IDE zařízení se zde ukládá číslo logického sektoru. Může zde být

pouze číslo 0 až 1692, přičemž 0 neznačí BOOT sektor ale Infosektor (BOOT je až na 1), pozor na to!

16180,#3F34,SPCKO 76
Zásobník se před načítáním SNAPSHOTu nastavuje u každého MDOSu na 16256. 76 bytes by mělo stačit (počítal jsem to pro disketu i HDD).

16256,#3F80,BUFF10 10
16266,#3F8A,BUFF32 32
42 bytes jsem potřeboval na různé buffery, resp. na dočasně přidělenou paměť.

16298,#3FAA,NMISP 60
Zásobník pro NMI menu.

16358,#3FE6,SYSFLAG2 2
Na #3FE7 je hodnota z #3EF7 (SYSFLAG), kterou dává MDOS 2.1 do SNAPSHOTu navíc. Snap je tak zase o něco dokonalejší.

16360, #3EF8,SNAPREG 22
Registry SNAPu.

16382,#3FFE,SAVESP 2
Uložení registru SP (zásobníku) při SNAPu

Pokud si při MOVE vynutíte oddíly, tak se musí načítat MBR (to aby se zjistilo, kde daný oddíl začíná…). Na buffer se v tomto případě použije oblast o délce **58** bytes od adresy **16178** (**#3F32**).

Příkaz "LIST !" používá pro uložení hodnot prostor od 15360 (#3C00). Volám standardně program na adrese 14333 a #3C00 slouží pro registr IX. Dále používá 512 bytes od 14336 (#3800), kde se uloží údaje načtené z IDE zařízení.

Kvůli GAL zákmitům požívám také 16084-16097 (#3ED4-#3EE1). Původně to MDOS používá pouze pro výpis 24 bitových čísel. Moje použití by funkčnost systému nemělo nijak ovlivnit.

5. Pár (kritických) slov k divIDE

V současné době nese poslední verze divIDE označení 57c. Od předcházejících verzí (57b, 57), kterou má asi většina se moc neliší. Zilog nepřidává nové funkce, spíše verzemi zlepšuje stabilitu hardwaru, příp. optimalizuje.

Teď uvedu několik věcí, které mi na divIDE vadí. Berte to prosím Vás jako můj subjektivní názor, který nikomu nevnucuji, pouze jej ventiluji ven :).

1. Zařízení je osazeno pouze 32K RAM. Zilog tvrdí, že to každému musí stačit, že více není potřeba. S tím naprosto nesouhlasím. divIDE bylo již od začátku (papírově) prezentováno tak, že dovoluje až 512K. Bohužel o více jak 32K si podle současné HW koncepce můžeme nechat jen zdát.

2. Chybí resetovací IDE registr. Softwarově tak není možné IDE zařízení zresetovat.

3. Chybí obvod 8255. Nějak jsem si už za ty roky na přítomnost paralelního interface zvyknul a považuji jej za šikovnou věc, která dovolí připojit tiskárnu, spojit dva počítače… No prostě i blbá D40/80 ho měla.

4. Výroba a servis. Zpočátku bylo divIDE dodáváno pouze jako stavebnice a každý nechť si ho postaví a oživí sám. Naštěstí v současné době se výroby ujal Noby.

5. Vzhled. Někomu se možná líbí kus zeleného integráče osazeného součástkami, který čouhá z kompu. Já bych ale raději preferoval klasickou černou krabičku, ze které by čouhaly akorát tři ledky, jedno NMI tlačítko a jeden IDE konektor.

6. Nechuť autora (Ziloga) divIDE vylepšovat. Zilog nechce narušit svůj minimalistický koncept, vždy si jej obhájí a vylepšovací návrh zavrhne.

6. Užijte si to

Co dodat? Snad jen, že doufám, že MDOS 2.1 využije i někdo jiný než já sám. Také podotýkám, že zdrojový text jsem uvolnil a budu rád za vaše názory, nápady a vůbec jakoukoliv odezvu.

Nakonec díky. Díky patří **Nobymu** (postavil mi divIDE, vyměnil GAL, opravil Ivanovi Kompakt), **Ivanovi** (půjčil mi svůj Kompakt na testování), **Sweetovi** (spolupráce).

Škoda jen, že jsem MDOS nedodělal už v roce 2003, tři roky (do teď) to leželo jen tak rozpracované, což nemívám ve zvyku…

mts.zxs@tiscali.cz

ICQ: 144264603