

**D A M + 2**

**Assembler a ladící program pro PMD-85-2**

---

**4004/482 ZO Svazarmu 1988**

## 1. Úvod

-----

DAM+2 je verze programu DAM určená pro počítač PMD 85-2. Proti předchozí verzi byly provedeny mnohé úpravy včetně změn v reprezentaci zdrojového textu, takže DAM+2 je s programem DAM-0000 na počítači PMD 85-1 z hlediska zdrojových textů neslučitelný. Naproti tomu byl vylepšen editor, zvětšen počet plněných funkcí a funkce z původního programu byly zdokonaleny.

DAM+2 umožňuje, stejně jako jeho předchůdce, tvorbu a ladění programů v assembleru a práci s pamětí počítače. Je rozdělen do tří částí.

Část první, ASSEMBLER, umožňuje tvorbu, záznam a překlad zdrojových textů v jazyce symbolických adres mikroprocesoru 8080.

Část MONITOR částečně nahrazuje služby operačního systému počítače. Umožňuje výpisy a opravy obsahu paměti, vyhledávání zadaných posloupností. Neumožňuje spolupráci s magnetofonem.

DEBUGGER, poslední část, pomáhá při ladění programů ve strojovém kódu. Umožňuje krokování a spouštění programů v mnoha režimech.

## 2. Nahrání programu a jeho spuštění

-----

DAM+2 lze nahrát do paměti počítače od libovolné adresy mezi 0000H až 5800H. Jedinou podmínkou je, že zvolená nahrávací adresa musí být adresou paměťové stránky, tedy musí končit dvěma nulami ( například adresy 0100H, 2500H, atd... ).

Nahrání provedete tak, že v operačním systému zadáte příkaz "MGLD 00 XX", kde "XX" je vámi zvolená stránka v paměti počítače, na kterou chcete DAM nahrát ( například pokud jej chcete nahrát od adresy 3700H, zadáte "MGLD 00 37" ). Pokud chcete DAM nahrát od adresy 0000H, stačí zadat pouze "MGLD 00".

Do paměti počítače se nahrává nejprve zavaděč, a to na adresy 7000H až 7300H. Tento zavaděč zjistí, od jaké adresy má vlastní program nahrát, provede jeho nahrání a nakonec program spustí, když předtím sám sebe smaže. Po nahrání je tedy vždy smazána paměť v místě zavaděče, zbytek paměti ale zůstává nedotčen.

Zavaděč, dříve než začne nahrávat program do paměti, kontroluje správnost zadání nahrávací adresy. Pokud bylo zadání u příkazu "MGLD" nesprávné, například zadaná adresa byla příliš vysoká ( vyšší než 58 ), nebo vůbec nešlo o adresu, vypíše hlášku "++ File error ++" a v nahrávání nepokračuje. Po stisku libovolné klávesy předá řízení zpět operačnímu systému. Ke stejné akci dojde, pokud se hlavní program nahraje s chybou.

Pokud se hlavní program nahraje správně, vypíše se po nahrání na obrazovku úvodní hlavička a po stisku libovolné klávesy se program spustí.

Znovuspuštění programu po vyskočení do operačního systému lze provést příkazem "JUMP XXXX", kde "XXXX" je adresa, od které je DAM v paměti nahrán. Nedoporučujeme tímto způsobem spouštět DAM, pokud se do paměti nahrál s chybou - mohli byste se dočkat nemilého překvapení v podobě "spadnutí" počítače.

Při spuštění programu zůstávají zachovány všechny systémové proměnné v tom stavu, v jakém byly při jeho opuštění, zůstávají tedy nastaveny hodnoty registrů v DEBUGGERu, zůstává připojen zdrojový text a zůstávají zapnuty či vypnuty výpisové funkce pro tiskárnu a obrazovku ( takže pokud jste prováděli těsně před opuštěním DAMu výpis na tiskárnu, může se program po novém spuštění při některém výpisu "kousnout", protože čeká na odpověď od tiskárny, kterou jste mezitím odpojili ).

Využíváte-li DAM+2 z ROM modulu, ohlásí se Vám ihned po zapnutí počítače zavaděč a požaduje zadání následujícího režimu práce. Stiskem 'B' zavedete BASIC, 'M' Vás přesune do monitoru PMD a 'D' volá DAM. Nejprve se vypíše požadavek "Enter DAM start address" a očekává zadání hexadecimálního dvojčíslí udávajícího horní polovinu zaváděcí adresy. Po stisku 'EOL' se během několika sekund objeví hlavička DAMu. Pokud by při přenosu dat došlo k chybě, zavaděč to ohlásí. Maximum zaváděcí adresy z modulu je 6200H.

### 3. Všeobecné zásady

-----

Pokud je v některé části DAMu vyžadován text jako data, píše se vždy mezi apostrofy, případný apostrof uvnitř textu musí být zdvojen ( týká se zadávání textů jako operandů instrukcí ve zdrojových textech a v datech v MONITORu ).

V DAMu se na mnoha místech setkáte s tím, že máte zadat nějakou adresu, číslo, hodnotu nebo text. Obvykle vás k tomu počítač vyzve nápisem v dialogovém řádku. Zadání je možno provést několika způsoby.

Předně pokud zadáte prázdnou řádku ( většinou to vyžaduje dvakrát stisknout 'EOL', protože po prvním stisknutí jen zmizí nápis, který Vás k zadání vyzývá, a teprve po druhém stisknutí se vložená řádka zpracuje ), způsobí to ve většině případů opuštění té funkce, která zadání vyžaduje. Například v MONITORu při výpisu paměti Vás počítač textem "Enter address" vybízí k zadání adresy k výpisu. Pokud nyní zadáte prázdnou řádku, opustíte funkci výpisu paměti a můžete vstoupit do další funkce. V některých případech zadání prázdné řádky nezpůsobí opuštění funkce - na tyto případy bude upozorněno při popisu dané funkce.

Pokud počítač vyžaduje zadání textu ( názvu souboru nebo zadání návěští ), stačí jednoduše tento text napsat a stisknout klávesu 'EOL'.

Zadávání adres a různých čísel lze provést ve třech soustavách - desítkové, šestnáctkové a dvojkové. Desítková čísla se zapisují způsobem zcela obvyklým - například "10", "1123", "458000" jsou desítková čísla. Čísla v šestnáctkové soustavě - hexadecimální - se zadávají obdobně, použitím šestnáctkových číslic, ale musí vždy začínat číslicí a končit znakem "H". Například "1234H", "0C1B0H" jsou hexadecimální čísla. Binární čísla se zadávají použitím číslic "0", "1" a končí písmenem "B". "10110100B", "1111011100110001B" jsou binární čísla.

Při zadání čísel je možno vždy použít pouze jednoho z těchto tří způsobů zadávání. Program při zpracování pozná, kterého z těchto tří způsobů jste použili. Pokud jste při zadání čísla provedli chybu, nemusí na to ale program vždy přijít. Pokud například zadáte "60A1", bude programem přečtená hodnota 60, protože zjistí, že dané číslo není hexadecimální ( nekončí "H" ) a při čtení skončí na prvním znaku, který není dekadickou číslicí. Pokud program přijde na chybu při zadávání, vypíše "++ Error in adres ++", čímž Vás požádá o nové zadání.

Některé funkce MONITORu vyžadují kromě adresy ještě zadání dat. Data se již zadávají pouze ve formě hexadecimálního dvojčíslí ( bez úvodních nul a "H" na konci ) nebo ve formě textů. Zadání adresy s daty může pak vypadat například takto: "601CH 00 20 'YOU'RE'0D0D".

Program si při běhu jistým způsobem organizuje paměť. Podrobnosti o této organizaci budou popsány ve zvláštní kapitole, důležité však je, že všechny části paměti, které jsou v daném okamžiku nutné ke správnému provozu programu, jsou jistým způsobem chráněny. Nadále budou označovány jako "chráněné oblasti".

## **4. Ovládání**

-----

Pro umožnění rychlého přecházení mezi jednotlivými částmi DAMu je ovládání provedeno způsobem "menu", kde si lze zvolit požadovanou funkci stiskem jediného tlačítka - toho tlačítka, které označuje první písmeno názvu zvolené funkce. Tato menu se vypisují do dialogové řádky.

Po spuštění programu se v dialogové řádce objeví hlavní menu - text "ASSEMBLER-MONITOR-DEBUGGER-QUIT". "QUIT" v tomto případě znamená opuštění programu a návrat do operačního systému počítače.

### **4.1. ASSEMBLER**

-----

Po spuštění provádí nejprve kontrolu zdrojového textu. Tato kontrola spočívá v tom, že zjistí, zda jeho systémové proměnné ukazují na platný zdrojový text. Pokud ano, umožní Vám další práci s tímto textem. Pokud ne, vytvoří na témže místě nový ( prázdný ) zdrojový text. Pak vypíše další menu: "EDIT-TRANSLATE-SAVE-CHECK-LOAD-MERGE-QUIT", přičemž "QUIT" znamená návrat do hlavního menu.

#### **4.1.1. EDIT**

-----

Tato část ASSEMBLERu "bdí" nad zdrojovým textem při opravách, umožňuje práci i nad více zdrojovými texty současně uloženými v paměti. Zpracování může být namířeno vždy jen na jeden zdrojový text, EDIT umožňuje právě přenášení zpracování z jednoho textu na druhý. Hlavní funkcí je však provádění změn ve zdrojovém textu pomocí celostránkového editoru.

##### **4.1.1.1. Vstup do editoru**

-----

Po spuštění EDITu se v dialogové řádce objeví text "Enter label". Pokud nyní zadáte číslo, chápe ho program jako adresu nového zdrojového textu. Pokud tato adresa ukazuje do chráněných oblastí, počítač pípne a vypíše znovu "Enter label". V opačném případě převezme program zdrojový text, který se na této adrese nachází, pokud se zde zpracovatelný text nenachází, založí na tomto místě nový zdrojový text. Pak znovu vypíše "Enter label".

Pokud nezadáte číslo, vstoupíte do celostránkového editoru tímto způsobem:

Pokud byla zadána prázdná řádka, objeví se Vám zdrojový text nastavený na začátku.

Pokud bylo zadáno návěští vyskytující se ve zdrojovém textu, vypíše se Vám zdrojový text počínaje řádkem označeným tímto návěští.

Pokud bylo zadáno návěští, které se v textu nevyskytuje, začnete s editací na samém konci zdrojového textu - objeví se prázdná obrazovka.

Kurzor pro editaci se vždy objeví v levém horním rohu obrazovky.

#### 4.1.1.2. Ovládání editoru

K ovládání editoru je využita celá klávesnice. Tlačítko stisknuté se shiftem ( nadále je budeme označovat `^'XXX'` ) má vždy jiný význam než totéž tlačítko stisknuté samostatně.

Pro pohyb kurzoru slouží tato tlačítka:

'<-' Pohybuje kurzorem v řádce doleva. Pokud se kurzor nalézá na prvním znaku na řádce, zůstane stát.

'->' Pohybuje kurzorem v řádce doprava. Pokud se kurzor nalézá na posledním znaku na řádce, zůstane stát.

'HOME' Nastaví kurzor na levý okraj řádky.

'END' Nastaví kurzor za poslední nemezerový znak na řádce, pokud je řádka prázdná, pak na její druhou pozici.

^'<-' Pohybuje kurzorem po řádkách nahoru. Pokud se kurzor nachází na první řádce na obrazovce, posune se text na obrazovce o patnáct řádek dolů ( odstraní se ) a kurzor se objeví ve spodní části obrazovky ( ne na nejspodnější řádce, ale na té, na kterou se měl posunout ). Pokud se kurzor nachází na první řádce zdrojového textu, vypíše se sice obrazovka znovu, ale kurzor zůstane na původním místě.

^'->' Pohybuje kurzorem po řádkách dolů. Pokud se kurzor nachází na poslední řádce obrazovky, posune se text na obrazovce o patnáct řádek nahoru a kurzor se objeví v horní řádce obrazovky. Pokud se kurzor nachází

na poslední řádce zdrojového textu, nestane se nic.

^'|<-' Posune kurzor o patnáct řádek nahoru ( pokud je ale kurzor méně než patnáct řádek od začátku zdrojového textu, pak jej posune jen na jeho první řádek ) a odstránkuje.

^'->|' Posune kurzor o patnáct řádek dolů ( ale opět maximálně na konec zdrojového textu ) a odstránkuje.

^'HOME' Nastaví kurzor na první řádku zdrojového textu.

^'END' Nastaví kurzor na poslední řádku zdrojového textu ( objeví se prázdná obrazovka s kurzorem v levém horním rohu ).

'EOL' Nastaví kurzor na první znak následující řádky. Pokud je zapnut vkládací režim, pak před následující řádku vloží prázdnou řádku a nastaví kurzor na její začátek.

Změny obsahu řádek umožňují tato tlačítka:

Všechna znaková tlačítka na klávesnici způsobí vypsání znaku na pozici kurzoru a posunutí kurzoru o znak doprava.

Klíčové klávesy 'K0' - 'K11', '^'K0' - '^'K11' lze používat stejně jako v operačním systému ( nadefinování obsahu pomocí klávesy 'WRK', samostatné stisknutí znamená výpis nadefinované hodnoty ).

'DEL' Smaže znak, pod kterým stojí kurzor, a zbytek řádky posune o znak doleva.

'INS' V místě kurzoru vloží mezeru, zbytek řádky se posune o znak doprava.

'CLR' Smaže řádku a nastaví kurzor na její začátek.

^'CLR' Smaže zbytek řádky od kurzoru doprava.

'RCL' Obnovuje původní obsah řádky ( původní znamená ten, který byl v řádce zapsán těsně před tím, než se na ní ocitl kurzor, nebo těsně před tím, než byl definován začátek či konec bloku ).

'WRK' Slouží k nadefinování obsahu klíčových kláves.



^'WRK' Posunuje znak, pod kterým se nachází kurzor, o jednu pozici dopředu v ASCII-tabulce.

K práci s textem dále slouží tato tlačítka:

'|<-' Definuje začátek bloku. Řádka, na které stojí kurzor, se stává první řádkou bloku.

'->|' Definuje konec bloku. Řádka, na které stojí kurzor, se stává poslední řádkou bloku.

Pokud je v editoru nadefinován blok, rozsvítí se červená LED-dioda na klávesnici. Pokud tato dioda svítí, zůstává blok nadefinován. Blok obsahuje vždy pouze celé řádky.

^'DEL' Pokud je nadefinován blok, smaže jej a přesune kurzor na první řádku za smazaným blokem. Pokud není nadefinován blok, smaže řádku, na které stojí kurzor.

'C-D' Kopíruje nadefinovaný blok před řádku, na které stojí kurzor. Pak přesune kurzor na začátek této kopie. Tato funkce je u delších zdrojových textů značně časově náročná.

^'RCL' Vypisuje nadefinovaný blok na tiskárnu. Výpis lze mezi jednotlivými řádky zastavit stiskem klávesy 'STOP'.

^'INS' Zapíná a vypíná vkládací režim. Při zapnutí se před řádku, na které stojí kurzor, vloží jedna prázdná řádka a kurzor se přesune na tuto prázdnou řádku. Pak se po každém stisku 'EOL' vytvoří za zadanou řádkou jedna prázdná řádka a na ni se přesune kurzor.

^'C-D' Slouží k opuštění editoru, k přemístění kurzoru podle zadaného návěští a ke změnám zdrojového textu. Po stisknutí této klávesy se smaže obrazovka a kurzor se objeví v jejím levém dolním rohu. Zadáním prázdné řádky opustíte editor a vrátíte se do menu ASSEMBLERu, jinak lze postupovat stejně jako při vstupu do editoru (zadat návěští nebo adresu zdrojového textu).

Poznámky k editaci:

Řádky zdrojového textu se do textu zařazují až v okamžiku, kdy kurzor opouští řádku, když je na řádce definován blok,

nebo když je prováděna bloková operace ( k zařazení nedochází při stisku kláves sloužících k změnám obsahu řádky a při stisku kláves pro pohyb kurzoru, které neznamenají opuštění řádku ). Pokud řádku v takovém případě nelze zařadit, protože neodpovídá její syntaxe, pak se nepokračuje v plnění funkce klávesy, počítač pípne a nastaví kurzor na místo, kde byla nalezena chyba.

Pokud při zařazení řádky do zdrojového textu dojde ke změně jeho délky, zruší se definice bloku ( i pokud je nadefinován jen jeho začátek či konec ). To může způsobit například "záhadnou" ztrátu funkce klávesy '^DEL', protože pokud máte definován blok, můžete udělat změny v řádce, aniž by se blok zrušil, protože řádka dosud nebyla zařazena do textu. Po stisku '^DEL' se zjistí, že je nadefinován blok, takže se provede zařazení řádky do textu ( kurzor při této operaci opouští řádku ), po zařazení ale blok definován není ( změnila se délka zdrojového textu ), takže se smazání bloku neprovede ( neprovede se vlastně vůbec nic ).

Může se stát, že zdrojový text je příliš dlouhý a přetekl by do chráněných oblastí. Tomu se editor brání tímto způsobem: Pokud chcete kopírovat příliš dlouhý blok, ( klávesou 'C-D' ), počítač pípne a neprovede se nic ( blok zůstane nadefinován ). Pokud by Vámi opravený řádek prodloužil text do chráněných oblastí, počítač pípne a v řádku se objeví jeho původní obsah ( jako po stisku klávesy 'RCL' ). Pokud chcete vkládat řádku ( tato operace prodlužuje text o jeden byte ) a text by se nevešel, řádka se jednoduše nevloží. Ve všech těchto případech se také samozřejmě nepokračuje ve vykonávání funkce stisknuté klávesy.

Pokud stisknete klávesy určené pro práci s blokem, aniž by byl nadefinován blok, neprovede se nic.

#### **4.1.1.3. Syntaktická pravidla**

-----

V této kapitole se budeme věnovat popisu syntaxe zdrojového textu, která souvisí s editací i překladem.

DAM provádí syntaktickou kontrolu ve třech fázích.

První fáze probíhá v okamžiku, kdy je k programu připojován zdrojový text z paměti. Tato fáze probíhá bez vnějších efektů a jejím důsledkem v případě chyby je, že se text "smaže" ( v této fázi se kontroluje, zda je editor schopen tento text zpracovávat ).

Druhá fáze probíhá během práce s editorem. Kontrola se provádí v okamžiku, kdy je řádka zařazována do zdrojového textu. V případě chyby je na to uživatel vždy upozorněn, takže se nemůže stát, že by se obsah řádky v textu neshodoval s obsahem řádky na obrazovce (pokud by k tomu mělo dojít, editor nepovolí opustit řádku kurzorem, nelze ani opustit editor).

Třetí fáze probíhá v době překladu. Spočívá v kontrole zápisu výrazu. V případě chyby se vypisuje chybové hlášení a provádí se návrat do editoru s vyznačením řádky, ve které k chybě došlo.

Řádka zdrojového textu může být pouze v jednom z těchto dvou tvarů:

#### 1. komentářová řádka

" ; komentář....."

Středník musí být prvním znakem na řádce, obsah zbytku řádky může být libovolný.

#### 2. příkazová řádka

"<návěští> <operace> <operandy>"

<návěští> se na řádce nemusí vyskytovat, pokud ale je uvedeno, musí začínat na prvním znaku řádky. Začínat musí velkým písmenem nebo znakem '@' ( "zavináč" ), ve zbytku se mohou vyskytovat velká písmena, číslice a znaky '!', '"', '#', '\$', '%', '&', '\_', '@'. Maximální povolená délka návěští je osm znaků.

<operace> je jedna z instrukcí 8080 nebo jedna z pseudoinstrukcí, jejichž výčet bude uveden níže. V řádce musí být uvedena vždy a musí být oddělena od návěští či od začátku řádky alespoň jednou mezerou.

<operandy> musí odpovídat použité operaci. Nemusí již být od operace odděleny mezerou.

V příkazové řádce není povoleno psát komentáře.

U některých instrukcí a pseudoinstrukcí je jako jeden z operandů aritmetický výraz. Hodnota výrazu se počítá v šestnáctibitové celočíselné aritmetice bez znaménka, jeho zápis se řídí těmito pravidly:

Jako operandy výrazu je možno používat návěští, čísla, texty a výrazy v závorkách.

Z textů uvedených ve výrazu se uchovávají maximálně poslední dva znaky ( v případě jednoznakového textu jde o ASCII-hodnotu znaku, u více znaků se znaky posunují vždy o osm bitů ).

Závorky jsou povoleny do pěti úrovní.

Za každým operandem lze uvést libovolné množství unárních operátorů, které bezprostředně upravují hodnotu tohoto operandu.

Unární operátory jsou:

' ' ( '^@' ) znamená negaci ( pokud je hodnota operandu různá od nuly je výsledkem nula, pokud je operand nulový, je výsledkem jednička ).

'\ ' znamená prohození horního a dolního byte operandu ( vhodné například pokud je třeba dostat do jednoduchého registru horní část nějaké adresy ).

S operandy je možno provádět tyto operace:

Sčítání - znaménkem '+'

Odčítání - znaménkem '-'

Násobení - znaménkem '\*'

Na začátku výrazu lze psát unární minus. Vyčíslení pak probíhá tak, jako by před tímto znaménkem byla ještě nula.

Výpočet výrazu s těmito operátory probíhá běžně známým způsobem zleva doprava ( násobení má vyšší prioritu ).

Výraz se může skládat ze dvou výrazů oddělených znaménkem '=' nebo '<'. Při vyčíslování se výrazu přiřadí hodnota pravdivosti tohoto predikátu ( tj. hodnotou výrazu "12=12" je 1, hodnotou výrazu "64<20" je 0 ). Výrazy typu A=B=C nejsou povoleny. Tento způsob lze použít například při podmíněném překladu ( je možno provádět i další logické operace, např. logický součin operací '\*', součet '+', dosažení logické hodnoty pomocí dvojité negace ).

#### 4.1.1.4. Pseudoinstrukce

Úmluva: <výraz> znamená libovolný aritmetický výraz.

<výrazx> znamená výraz, ve kterém se všechna uvedená návěští vyskytují na předchozích řádkách ve zdrojovém textu ( vyčíslení tohoto výrazu se provádí již při prvním průchodu překladače a je nutné aby všechna uvedená návěští byla definována ).

Překladač zná všechny obvyklé pseudoinstrukce, k dosažení speciálních funkcí byly zavedeny i některé nové. U žádných z pseudoinstrukcí není uvedení <návěští> povinné.

"ORG"

Syntaxe "<návěští> ORG <výrazx>"

Pokud je návěští uvedeno, nabývá hodnoty minulého čítače instrukcí ( například "A ORG A-1" způsobí při překladu návrat o jeden byte zpět ).

Pseudoinstrukce způsobí přenos překladu na adresu určenou výrazem. Pokud není na začátku zdrojového textu tato pseudoinstrukce uvedena, provádí se překlad od adresy 6000H.

"EQU"

Syntaxe "<návěští> EQU <výraz>"

Pokud návěští není uvedeno, neprovede se nic ( není ani vyčíslována hodnota výrazu ). Návěští se přiřadí hodnota výrazu.

"DS"

Syntaxe "<návěští> DS <výraz>"

Vyhradí v paměti počet byte odpovídající hodnotě výrazu, aniž by do paměti zasahoval ( lze tedy vyhrazovat i v chráněných oblastech ). Při překladu na magnetofon generuje odpovídající počet nul.

"DW"

Syntaxe "<návěští> DW <výraz>, ... ,<výraz>"

Je generována posloupnost dvoubyte v intel-notaci ( nižší byte je první ) s hodnotami výrazů.

"DB"

Syntaxe "<návěští> DB <výraz>, ... ,<výraz>"

Je generována posloupnost byte s hodnotami výrazů ( ukládá se nižší byte výsledku ). Neumožňuje zadání znakových řetězců, k této funkci je nutno použít pseudoinstrukci "DM" nebo "DT".

"DM"

Syntaxe "<návěští> DM <posloupnost byte>"

Umožňuje přímé zadání textových řetězců a hexadecimálních kódů jako u příkazu "SUB" v operačním systému.

"DT"

Syntaxe "<návěští> DT <text>, ... ,<text>"

Umožňuje zadání textových řetězců, jejichž poslední znak má nastavený nejvyšší bit. Texty musí být zapsány standardním způsobem, tedy mezi apostrofy a musí obsahovat alespoň jeden znak.

"FILE"

Tato pseudoinstrukce se používá k označení začátku bloku při překladu na magnetofon. Pokud se neprovádí překlad na magnetofon, je ignorována. Slouží k zadání způsobu ukládání bloku co se týče jeho začátku:

Syntaxe "<návěští> FILE <číslo>,<název>"

Požádá o puštění magnetofonu, pak generuje hlavičku bloku, která obsahuje uvedené číslo a název ( místo čísla lze napsat aritmetický výraz, ten ale nesmí začínat písmeny 'N' ani 'H' ). Počáteční adresa generovaného bloku je dána čítačem instrukcí v okamžiku zpracování pseudoinstrukce, tedy je nutné nejprve uvádět "ORG", pak teprve definovat blok pomocí "FILE".

Syntaxe "<návěští> FILE H"

Požádá o puštění mgf., ale negeneruje hlavičku bloku.

Syntaxe "<návěští> FILE N"

Generuje bezhlavičkový blok, aniž požádá o puštění magnetofonu. Lze použít například pokud zavaděč nevyžaduje mezeru mezi svým koncem a začátkem zaváděného bloku.

"EOF"

Slouží k označení konce bloku při překladu na magnetofon. Pokud se neprovádí překlad na magnetofon, je i tato pseudoinstrukce ignorována.

Syntaxe "<návěští> EOF"

Ukládá blok včetně kontrolního součtu na konci ( kontrolní součet je počítán způsobem běžným pro PMD ).

Syntaxe "<návěští> EOF -"

Ukládá blok bez kontrolního součtu.

"ASM"

Syntaxe "<návěští> ASM <výraz>"

Pokud je hodnota výrazu nenulová, pokračuje dále v překladu normálním způsobem. Pokud je hodnota výrazu nulová, ignoruje následující řádky až do další pseudoinstrukce "ASM" nebo do konce zdrojového textu.

Zařazení návěští do tabulky se řídí tím, jestli byl překlad povolen na minulé řádce.

Například u textu	ASM 1
NAV0	ASM 0
NAV1	ASM 1

se do tabulky zařadí návěští NAV0, návěští NAV1 bude ignorováno.

#### **4.1.2 TRANSLATE**

-----

Tato část ASSEMBLERu provádí překlad zdrojového textu. Vyžádá si zadání způsobu překladu a případně další údaje. Pokud při překladu nalezne chybu, oznámí ji a pak přejde do editoru tak, že se kurzor nachází na řádce, na které došlo k chybě. Pokud překlad proběhne dobře, vrátí se do menu ASSEMBLERu nebo přímo spustí přeložený program.

##### **4.1.2.1 Spuštění překladu**

-----

Po vstupu do této části se v dialogové řádce objeví text "Enter options" a program očekává zadání čísla kombinace podmínek překladu. Toto číslo se sestavuje sečtením čísel jednotlivých podmínek ( 1,2,4,8 ).

Po zadání čísla 0 se provede normální překlad do paměti. Během prvního průchodu překladu se na obrazovku vypisují návěští a jejich hodnoty v tom pořadí, v jakém se nacházejí ve zdrojovém textu. Pseudoinstrukce týkající se překladu na magnetofon jsou ignorovány.

Podmínka s číslem 1 Vám umožní určit si místo v paměti, na kterém se během překladu bude ukládat tabulka návěští ( toto zadání si program vyžádá automaticky, pokud tabulku nelze umístit na implicitní adresu - těsně za zdrojový text ). Vypíše se "Enter table address" a program počká na zadání adresy. Pokud vložená adresa ukazuje do chráněných oblastí, program požádá o nové zadání.

Podmínka s číslem 2 způsobuje překlad do jiných částí paměti, než je určeno pseudoinstrukcemi "ORG". Vypíše se "Enter start address" a program počká na zadání počáteční adresy pro ukládání dat. Zadanou adresu program nekontroluje, ale generování kódu do chráněných oblastí není umožněno jinými kontrolami. Pak se vypíše "Enter run label" a program čeká na

zadání návěští, od kterého se pak má přeložený program po přenesení na správnou adresu spustit. Lze také zadat pseudonávěští "DAM" či "OS", která způsobí skok do DAMu či operačního systému, pokud nechcete spustit přeložený program. Pokud se ve zdrojovém textu návěští "DAM" či "OS" vyskytují, mají při zadání přednost pseudonávěští. Pokud zadáte návěští, které se ve zdrojovém textu nevyskytuje nebo pokud zadáte prázdnou řádku, provede se pouze překlad a program zůstává na tom místě, kam byl přeložen. Během překladu se pak u každé pseudoinstrukce "ORG" vypisuje hodnota, kam je následující blok adresován a vedle adresy, na které je skutečně uložen ( tento výpis se neprovádí, pokud je tabulka návěští nebo generovaný kód v obrazovce ). Po skončení překladu, pokud bylo správně zadáno startovní návěští, se celý blok vygenerovaného kódu jako jeden kus přenesení na adresu určenou prvním generovaným kódem. Přenos se provádí směrem od nižších adres k vyšším. Po skončení přenosu se spustí program podle zadaného návěští. Hodnota ukazatele zásobníku po skončení přenosu není definována ( zůstává na tom místě, kde ho měl DAM ).

Podmínka s číslem 4 ruší výpis hodnot návěští během prvního průchodu překladu ( tento výpis je zrušen automaticky, pokud máte tabulku návěští v obrazovce ).

Podmínka s číslem 8 zapíná překlad na magnetofon. Pokud zapnete tuto podmínku společně s podmínkou 2, je podmínka 2 ignorována ( nemá význam ). Během prvního průchodu překladu se založí všechny bloky definované pseudoinstrukcemi "FILE" a "EOF" ( může jich být maximálně 16 ), během druhého průchodu se obsah těchto bloků místo ukládání do paměti nahraje na magnetofon. Všechny instrukce mimo specifikované bloky se překládají, ale nikde se neukládají.

Překlad lze kdykoli zastavit stiskem klávesy "STOP". Pak se vypíše "Press 'EOL' to continue, 'Q' to quit" do dialogové řádky. Po stisku 'EOL' se pokračuje v překladu, po stisku 'Q' je překlad zrušen a provede se návrat do menu ASSEMBLERu.

#### **4.1.2.2. Chyby při překladu**

-----

Během překladu se stále kontroluje, zda nejsou přepisovány chráněné úseky paměti a zda lze zdrojový text správně přeložit. Pokud dojde k chybě, počítač pípne a v dialogové řádce se objeví chybové hlášení. Po stisku libovolného tlačítka pak program



přejde do editoru a kurzor se objeví na řádce, na níž k chybě došlo. Nastat mohou tyto chyby:

"++ Memory overflow ++"

Text obsahuje více návěští než se jich vejde do prostoru pro tabulku. V tomto případě je vhodné překládat nadále s tabulkou návěští na jiném místě v paměti, nejlépe v obrazovce.

"++ Bad org ++"

Překládané instrukce nebo data jsou směrována do chráněných oblastí. Změňte pseudoinstrukce ORG nebo překládejte s podmínkou 2 do dostatečně velkého prostoru ( například do obrazovky ).

"++ Dual label ++"

Návěští, kterým je řádek označen, se v textu vyskytuje dvakrát nebo vícekrát.

"++ Missing label ++"

Některé z návěští ve výrazu se v textu nevyskytuje, popřípadě se nevyskytuje před tímto řádkem ( u některých pseudoinstrukcí ).

"++ Syntax error ++"

Chybně syntakticky zapsaný výraz nebo chybně syntakticky zapsané operandy pseudoinstrukce.

"++ Error in data ++"

Chybně zapsaná data u pseudoinstrukce "DM" nebo "DT".

"++ File error ++"

U "EOF" - nebyl deklarován soubor pomocí "FILE".  
U "FILE" - minulý soubor nebyl uzavřen pomocí "EOF".  
- bylo založeno více než šestnáct souborů.  
Konec textu - poslední soubor nebyl uzavřen pomocí "EOF".

"++ No file ++"

Soubor mezi "FILE" a "EOF" je prázdný, a proto jej nelze nahrát.

#### 4.1.3. SAVE

-----

Pomocí této funkce lze uložit vytvořený zdrojový text na magnetofonovou pásku. Po spuštění vypíše do dialogové řádky výzvu "Enter name" a očekává zadání jména, pod jakým má být zdrojový text na pásce uložen. Po zadání jména vypíše "Enter segment number" a očekává zadání čísla, pod kterým má být text uložen. Pak vypíše "Start tape, then press any key" a po stisku libovolného tlačítka vypíše do dialogové řádky obsah hlavičky ( číslo, typ a název nahrávaného bloku stejným způsobem, jakým se v operačním systému vypisuje obsah hlavičky u příkazu "MGLD" ) a nahraje text na magnetofon ( po stisku tlačítka nenechává mezeru jako operační systém u příkazu "MGSV", tuto mezeru je nutno vyrobiť "ručně" ). Po skončení nahrávání pípne a vypíše "++ Mg stop! ++" a po stisku libovolného tlačítka se vrátí do menu ASSEMBLERu.

#### 4.1.4. CHECK

-----

Slouží ke kontrole záznamů na magnetofonové pásce ( nemusí jít jen o zdrojové texty - funguje stejně jako "MGEND" v operačním systému ). Po spuštění vypíše do dialogové řádky "Enter segment number" a očekává zadání čísla bloku, který má kontrolovat. Pak očekává hlavičku souboru na magnetofonu. Když ji nahraje, vypíše její obsah do dialogové řádky. Pokud se její číslo shoduje se zadaným, zkontroluje kontrolní součet a pokud ten je v pořádku, pípne a vrátí se do menu ASSEMBLERu. Pokud je kontrolní součet kontrolovaného záznamu chybný, pípne, vypíše "++ File error ++" a po stisku libovolného tlačítka se vrátí do menu ASSEMBLERu. Když se číslo v přečtené hlavičce neshoduje se zadaným, pípne a očekává další hlavičku ( obsah záznamu nekontroluje ). Funkci lze kdykoli přerušit stiskem klávesy 'STOP'.

#### 4.1.5. LOAD

-----

Slouží k nahrání zdrojového textu z magnetofonu místo stávajícího zdrojového textu v paměti ( tedy vždy místo toho textu, který je zrovna zpracováván - lze použít, pokud je třeba dostat do paměti více než jeden zdrojový text ). Obsluhuje se stejně jako "CHECK", pouze po nahrání hlavičky navíc kontroluje, zda se nahrávaný text na dané místo do paměti vejde.

V případě, že by se text nevešel, okamžitě po nahrání hlavičky vypíše "++ Memory overflow ++" a po stisku libovolné klávesy se vrátí do menu ASSEMBLERu. V takovém případě zůstane v paměti původní zdrojový text. Pokud se záznam nahraje s chybou, vypíše se "++ File error ++" a text se z paměti vymaže ( nahradí se prázdným textem ). Funkci lze kdykoli přerušit stiskem klávesy 'STOP'. Pokud nahrávání přerušíte dříve, než se nahraje požadovaná hlavička, zůstane v paměti původní zdrojový text. Pokud nahrávání přerušíte během nahrávání záznamu, text se z paměti vymaže.

#### **4.1.6. MERGE**

-----

Tato funkce je shodná s funkcí "LOAD", pouze zdrojový text z magnetofonu nahrává těsně za poslední řádek stávajícího zdrojového textu ( připojí nový zdrojový text na konec starého ). Pokud dojde k chybě při nahrávání ( chyba v kontrolním součtu nebo stisk klávesy 'STOP' ), smaže pouze nahranou část, původní zdrojový text v paměti zůstane.

#### **4.2. MONITOR**

-----

Vypisuje menu "EDIT-FIND-LIST-DISASSEMBLER-QUIT". "QUIT" v tomto případě znamená návrat do hlavního menu.

##### **4.2.1. EDIT**

-----

Slouží k opravám a prohlížení obsahu paměti ( kombinace příkazů "MEM" a "SUB" v operačním systému ). Po spuštění vypíše "Enter address & data" a očekává zadání. Pokud zadáte pouze adresu, zobrazí do dialogového řádku opět tuto adresu a za ní dvanáct byte obsahu paměti od této adresy v podobě hexadecimálních dvojčíslí. Zadáte-li adresu i data, zapíše zadaná data do paměti a zobrazí následující adresu ( tentokrát bez obsahu paměti ). Data je možno zadávat v podobě hexadecimálních dvojčíslí ( mezery mezi těmito dvojčíslími nejsou povinné ) nebo textů ( uzavřených mezi apostrofy, apostrof v textu je nutno psát dvakrát za sebou ). Pokud v zadání dat uděláte chybu, počítač pípne a vypíše "++ Error in data ++". V tomto případě se data do paměti nezapiší. Zadání lze opravit např. po stisku klávesy 'RCL'.

#### **4.2.2. FIND**

-----

Slouží k vyhledávání zadané posloupnosti dat v paměti. Po spuštění vypíše "Enter address & data" a očekává zadání ve stejné podobě jako "EDIT". Pak vyhledává vložená data v paměti počínaje zadanou adresou. Pokud zadáte adresu, ale nezadáte data, vyžádá si zadání znova. Pak se do dialogové řádky vypíše text "Press 'STOP'" a program začne vyhledávání. Pokud při vyhledávání zadanou posloupnost nalezne, vypíše adresu prvního byte této posloupnosti na obrazovku a pokračuje ve vyhledávání. Vyhledávání pokračuje v paměti stále dokola, je nutno ho zastavit stiskem klávesy 'STOP'. Pak požádá o nové zadání adresy a dat.

#### **4.2.3. LIST**

-----

Slouží k výpisu obsahu paměti obdobně jako příkaz "DUMP" v operačním systému. Vypíše do dialogové řádky text "Enter address" a očekává zadání počáteční adresy pro výpis. Potom vypisuje obsah paměti tak, že na řádce je vždy vlevo hexadecimální adresa, následuje osm byte nejprve v podobě hexadecimálních dvojčíslí a pak ve znakové formě ( znaky jsou z obou stran ohraničeny vykřičníky, netisknutelné znaky jsou nahrazeny tečkami ). Výpis lze kdykoli zastavit stiskem klávesy "STOP". Po zastavení se v dialogové řádce objeví následující adresa, takže stiskem klávesy 'EOL' lze kdykoli ve výpisu pokračovat, popřípadě zadat novou adresu.

#### **4.2.4. DISASSEMBLER**

-----

Slouží k výpisu obsahu paměti v podobě zpětného překladu do strojových instrukcí mikroprocesoru 8080. Ovládá se zcela stejně jako funkce "LIST". Výpis vypadá tak, že na řádce je vždy adresa příslušné instrukce, za ní hexadecimální výpis této instrukce v její délce a výpis mnemoniky ( například "4035 C3 22 31 JMP 3122" ). Kódy, které nejsou instrukcemi, se vypisují shodně s definicí assembleru pomocí pseudoinstrukce "DM".

### 4.3. DEBUGGER

-----

Po spuštění této části programu se smaže obrazovka a vypíše se Vám "čelní panel". Pak se v dialogové řádce vypíše text "Enter command" a program čeká na Vaše příkazy.

#### 4.3.1. Čelní panel

-----

V horní části obrazovky se vypisuje "okénko" do paměti, celkem 64 byte vypsaných stejným způsobem, jakým vypisuje "LIST" v MONITORu. Výpis okénka může být také vypnut, pak zůstává horní část obrazovky prázdná.

Pod okénkem jsou vypsané obsahy jednotlivých dvojregistrů, vždy vlevo je název dvojregistru, za ním znak '=' a pak hexadecimální výpis obsahu dvojregistru. Následuje tentýž obsah vypsaný ve dvojkové soustavě a pak dvojice znakových ekvivalentů obsahu. Pak následuje šipka "->" a hexadecimální a znakový výpis toho byte paměti, na který dvojregistr ukazuje.

Pod obsahy registrů je zobrazeno osm nejbližších položek na zásobníku ( ve tvaru "SP -> XXXX ..." ), přičemž položka nejvíce vlevo se právě nachází na vrcholu zásobníku.

Pod výpisem zásobníku se vypíše disassembler instrukce, na kterou právě ukazuje registr PC ( tato instrukce právě má být simulována ).

Pohled na čelní panel ukazuje obrázek v příloze.

#### 4.3.2. Ovládání

-----

Příkaz DEBUGGERu se zadává vždy stiskem jednoho tlačítka, když je v dialogové řádce vypsán text "Enter command".

##### 4.3.2.1. Zadávání obsahu registrů

-----

Obsahy registrů lze změnit stiskem počátečního písmene daného dvojregistru ( 'A' pro AF, 'P' pro PC atd.. ). Pak se v dialogové řádce objeví text "Enter XX", kde "XX" je název daného dvojregistru, a program čeká na zadání nového obsahu. Stejným způsobem lze měnit i polohu "okénka" v paměti - klávesou 'W'. Pokud zadáte prázdný řádek, obsah registru se nezmění.

#### 4.3.2.2. Zapínání a vypínání výpisu okénka

Výpis okénka lze zapnout či vypnout stiskem klávesy 'V'. Když je výpis okénka zapnut, je práce s DEBUGGERem pomalejší, protože objem vypisované informace při výpisu čelního panelu je přibližně dvakrát větší.

#### 4.3.2.3. Krokování jednotlivých instrukcí

Po stisku klávesy 'EOL' se provede simulace jedné instrukce, a to té, na kterou před stiskem klávesy ukazoval registr PC a která byla vypsána ve spodní části obrazovky. Pak se registr PC automaticky posune na další instrukci.

Stiskem klávesy 'C' lze dosáhnout téhož efektu jako stiskem klávesy 'EOL', pouze s tím rozdílem, že při simulaci instrukcí volání podprogramu ( "CALL", "RST" ) se tento podprogram provede v reálném čase.

Pokud má být krokován kód, který není instrukcí, simulace se neprovede, počítač pípne a v dialogové řádce se vypíše text "++ Bad code ++".

#### 4.3.2.4. Krokování delších programů

Chcete-li krokovat více instrukcí ( například se chcete přenést na konec cyklu nebo nechat program krokovat do určité adresy ) je možné postupovat podle této tabulky:

simulace	rychlá	pomalá
do zadané adresy	'Y'	'R'
do následující instrukce	'X'	'O'
do návratové adresy	'Z'	'F'

Po stisku kláves pro simulaci do zadané adresy se vypíše do dialogové řádky "Enter break" a program čeká na zadání adresy, na které má simulace skončit. Po zadání adresy rozběhne simulaci. Ostatní druhy simulace se rozbíhají okamžitě.

Simulaci do následující instrukce lze výhodně použít při průchodu cyklem. Jako adresu skončení simulace chápeme adresu instrukce následující za tou, na kterou ukazuje PC.

Simulace do návratové adresy chápe jako koncovou adresu položku na vrcholu zásobníku v okamžiku začátku simulace.

Pomalá simulace znamená, že po simulaci každé instrukce se vypisuje čelní panel.

Rychlá simulace znamená, že čelní panel se během simulace nevypisuje, při započetí simulace se smaže obrazovka ( toto mazání lze vypnout - viz systémový popis programu ).

Během simulace je v dialogové řádce vypsán text "Press 'STOP'". Simulaci lze kdykoli přerušit stiskem klávesy 'STOP' ( pokud není zrovna prováděn podprogram v reálném čase ).

Stisknete-li některou z uvedených kláves bez shiftu, probíhá simulace stejně jako cyklické mačkání klávesy 'EOL'. Při stisku uvedených kláves se shiftem se provádí volané podprogramy v reálném čase.

#### **4.3.2.5. Volání podprogramů, spouštění programů**

-----

Po stisku klávesy 'U' se provede volání podprogramu, jehož adresa je v PC, s nastavenými obsahy registrů. Návrat z tohoto podprogramu zpět do DEBUGGERu se provádí instrukcí "RET".

Po stisku klávesy 'G' se provede skok do programu, jehož adresa je v PC, s nastavenými obsahy registrů. Návrat je nutno provést jiným způsobem, protože v tomto případě se návratová adresa na zásobník neukládá.

#### **4.3.2.6. Opuštění DEBUGGERu**

-----

DEBUGGER lze opustit stiskem klávesy 'Q'. Po stisku této klávesy se smaže obrazovka a program se vrátí do hlavního menu.

#### 4.4. Tisková funkce

-----

V této kapitole bude popsáno, jakým způsobem lze nasměrovat výpis na tiskárnu ( předpokládá se, že tuto tiskárnu máte již připojenu - připojení tiskárny bude věnována pozornost ve zvláštní kapitole ).

Pokud potřebujete nějakým způsobem ovládat výpisové funkce, musíte se nejprve dostat do situace, kdy je v dialogové řádce vypsán text DAMu a program čeká na stisk klávesy ( například v kterémkoli menu nebo před zadáním adres, ... ). Pokud v této situaci stisknete klávesu 'C-D', rozsvítí se červená LED-diody na klávesnici a program čeká na Vaše zadání tiskových funkcí.

Tiskové funkce se přepínají klávesami 'K0', 'K1', 'K2', stisk jiných kláves způsobí stejnou akci, jako byste předtím klávesu 'C-D' nestiskli.

Pokud klíčové klávesy stisknete bez shiftu, znamená to zapnutí dané funkce, stisk se shiftem danou funkci odpojuje.

'K0' zapíná a vypíná výpis na obrazovku. Po nahrání je tento výpis zapnut.

'K1' zapíná a vypíná výstup výpisu na tiskárnu. Po nahrání je tento výpis vypnut.

'K2' zapíná a vypíná stránkování při výpisu na tiskárnu. Po zapnutí se vždy odstránkuje, dříve než je vypsána první řádka výpisu. Po nahrání je stránkování vypnuto.

Těmito zadáními se řídí všechny výpisy MONITORu a výpisy ASSEMBLERu při překladu. Navíc není vypnut výpis na tiskárnu v případě, že nelze provádět výpis na obrazovku ( je v ní tabulka návěští či generovaný kód ), takže lze provádět výpis návěští na tiskárnu, i když je tabulka návěští v obrazovce. Zadání stránkování respektuje i výpis bloku na tiskárnu, který lze provést v editoru.

Stránkování se provádí tak, že po každých 60 řádkách tisku se 11 krát naprázdno odřádkuje.

Standardně je DAM+2 verze 1989 vybaven tiskovou rutinou spolupracující s tiskárnou CONSUL 2111 ( resp. tiskárnou s rozhraním IRPR ). Zapojení kabelu k tiskárně je shodné jako při tisku z BASICu. Popis je uveden např. ve čtvrtém dílu manuálu PMD ( OUTPUT/ENTER manuál ).



## 5. Systémový popis

Tato kapitola je věnována různým "kutilům", kteří se nespokojí s DAMem takovým, jaký ho získali, a chtějí na něm provádět některé úpravy, například připojení tiskárny, změna tvaru výpisu či jiná vylepšení.

### 5.1. Organizace paměti

Popis organizace bude proveden pro DAM nahraný od obecné adresy "XX00H".

počáteční adresa	obsah	poznámka
0000H	volná paměť	volné
XX00H	DAM	chráněno
XX00H+1800H	zásobník a pracovní oblast DAMu	chráněno
XX00H+1900H	volná paměť	volné
7B00H	oblast klíčů, ROM, systémové proměnné	chráněno
C200H	volná videoram	volné, nelze ukládat zdrojový text

Do chráněných oblastí dále patří zdrojový text a během překladu tabulka návěští.

Implicitní adresa začátku zdrojového textu po nahrání DAMu je těsně za DAMem, tedy na adrese XX00H + 1900H.

### 5.2. Povolené úpravy

Úpravy, které lze na DAMu provádět, úzce souvisejí s jeho začátkem. Začátek zdrojového textu vypadá takto:

```

DAMZAC    JMP    DAMGO
PRINT     JMP    PRINC
PAGE      JMP    PAGEC
TINI      JMP    PRINI
;
FSPC      DB     8
PAGR      DB     60
PAGP      DB     11
;
CLSBYTE   DB     0
;
OCHTAB    DW     DAMZAC,DAMZAC
          DW     7B00H,0C1FFH
          DW     DAMZAC,DAMEND
          ...

```

Vektor označený "DAMZAC" je spouštěcí vektor DAMu. Jeho směr není vhodné měnit.

Vektor označený "PRINT" je odskok na rutinu, která provádí výtisk jednoho byte na tiskárnu. Dostává tento byte v registru A, obsahy registrů není třeba zachovávat. Na konci řádky dostává pouze kód 0DH.

Vektor označený "PAGE" je odskok na rutinu, která obsluhuje odstránkování. Nedostává žádné parametry, obsahy registrů není nutno zachovávat.

Vektor označený "TINI" je odskok na rutinu pro inicializaci tiskárny. Tato rutina je volána vždy jednou těsně po spuštění DAMu. Nedostává žádné parametry, obsahy registrů není nutno zachovávat.

Byte označený "FSPC" udává, kolik mezer se vypisuje na řádku před vlastním tiskem řádky ( při tisku řádky se nejprve vypíše tento počet mezer, pak se vypíše obsah řádky a nakonec se odřádkuje znakem 0DH ).

Byte označený "PAGR" udává, po kolika řádcích textu se volá rutina pro odstránkování.

Byte označený "PAGP" udává standardně implementované stránkovací rutinu, kolikrát při stránkování odřádkovat.

"CLSBYTE" udává, zda se má v DEBUGGERu při rychlé simulaci mazat obrazovka. Pokud je jeho hodnota nenulová, mazání se potlačí.

"OCHTAB" je tabulka chráněných oblastí. Obsahuje celkem pět položek, pro naše účely jsou ale potřebné jen uvedené tři. Zásahy do této tabulky je nutno provádět uváženě, protože mohou mít destruktivní následky co se týče existence zdrojového textu i vlastního programu. Každá chráněná oblast je popsána dvěma ukazateli do paměti, první ukazuje na první byte této oblasti a druhý na poslední byte této oblasti. Druhý ukazatel musí být vždy větší nebo roven prvnímu, nelze tedy chránit oblast přes 0000H.

První chráněná oblast je uživatelsky definovatelná. Po nahrání je namířena dovnitř DAMu a neprojevuje se. Její hodnoty lze libovolně nastavit a tak získat novou chráněnou oblast, do které lze uložit vlastní podprogramy či programy.

Druhá oblast je oblast klíčů, ROM a systémových proměnných. Pokud vlastníte PMD s rozšířenou pamětí, lze tuto oblast s použitím uživatelsky definovatelné oblasti rozdělit na dvě a tak získat mezi ROM a videoram nové volné místo, kde lze ukládat programy či tabulku návěští. V této oblasti však nelze uchovávat zdrojové texty. Nedoporučuje se nechávat oblast ROM či systémových proměnných nechráněnou !

Třetí oblast je oblast DAMu. Lze ji "natáhnout" nahoru či dolů a uložit si programy do těsné blízkosti DAMu. Takto je vhodné např. ukládat tiskové rutiny. Pozor však na to, že těsně za DAMem je jeho pracovní oblast ( 256 byte ), která musí zůstat netknuta ( tato oblast ještě spadá do chráněné oblasti definované třetím údajem v tabulce ) !

Při změnách obsahu tabulky je třeba mít na paměti, že musí zůstat chráněny všechny oblasti, které by mohly poškodit zdrojový text ( i tím, že do nich zdrojový text prostě "přeteče" ), tedy oblast klíčů, ROM, systémové proměnné operačního systému, pracovní oblast DAMu i vlastní DAM.

Pokud Váš DAM nebude fungovat, jak zde bylo popsáno, obraťte se laskavě přímo na nás.

4004/482 ZO Svazarmu  
Klub digitální a měřicí techniky  
Lopatecká 615/5  
147 00 PRAHA 4