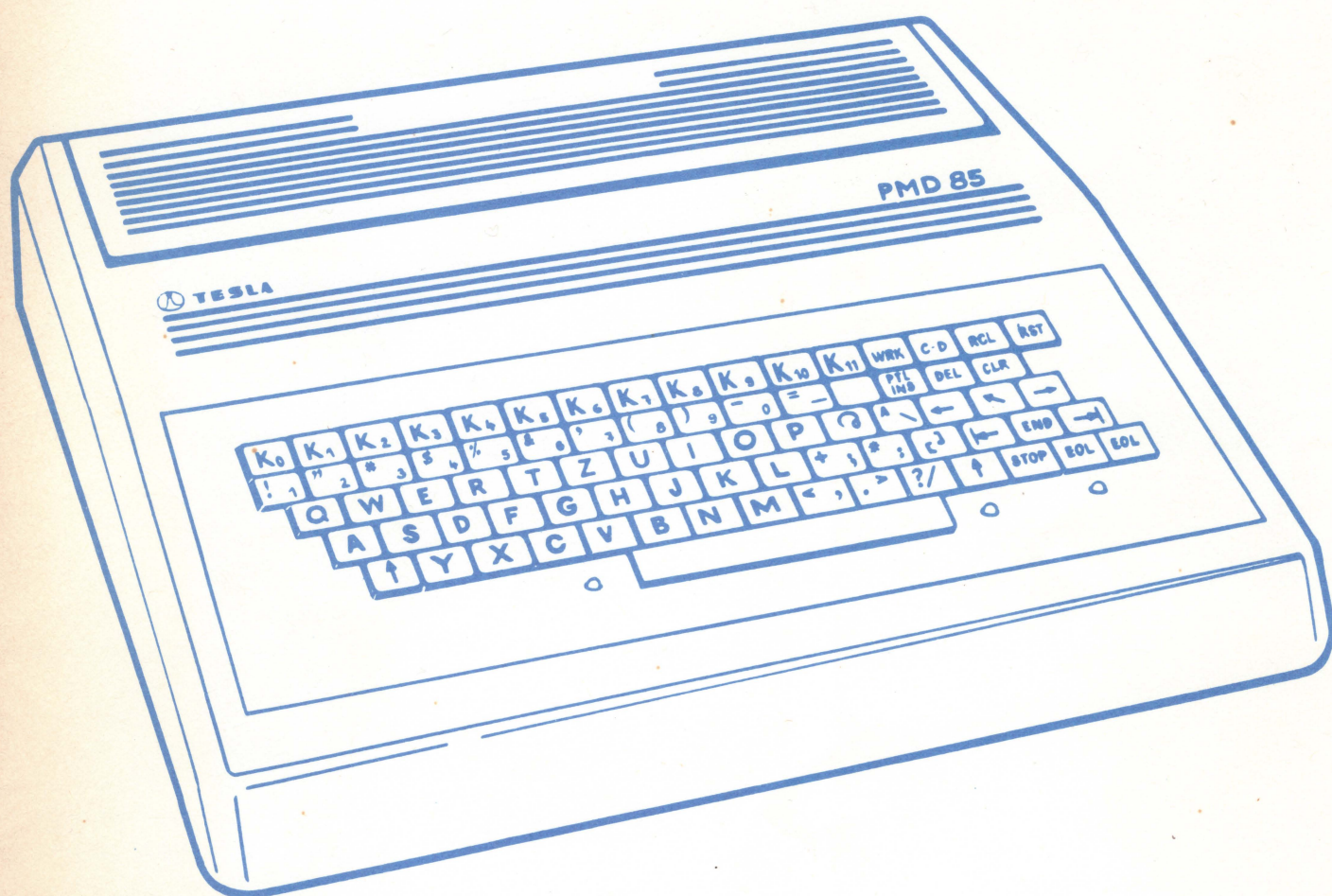


# OSOBNÝ MIKROPOČÍTAČ PMD 85



NÁVOD NA POUŽITIE A OBSLUHU  
MIKROPOČÍTAČA RADU PMD

**TESLA BRATISLAVA** 

## ÚVOD

PMD - 85 je osobný mikropočítač s grafickým spracovaním údajov. Je skonštruovaný na uľahčenie práce, vyplnenie voľných chvíľ a poslúži aj ako inteligentný spoločník.

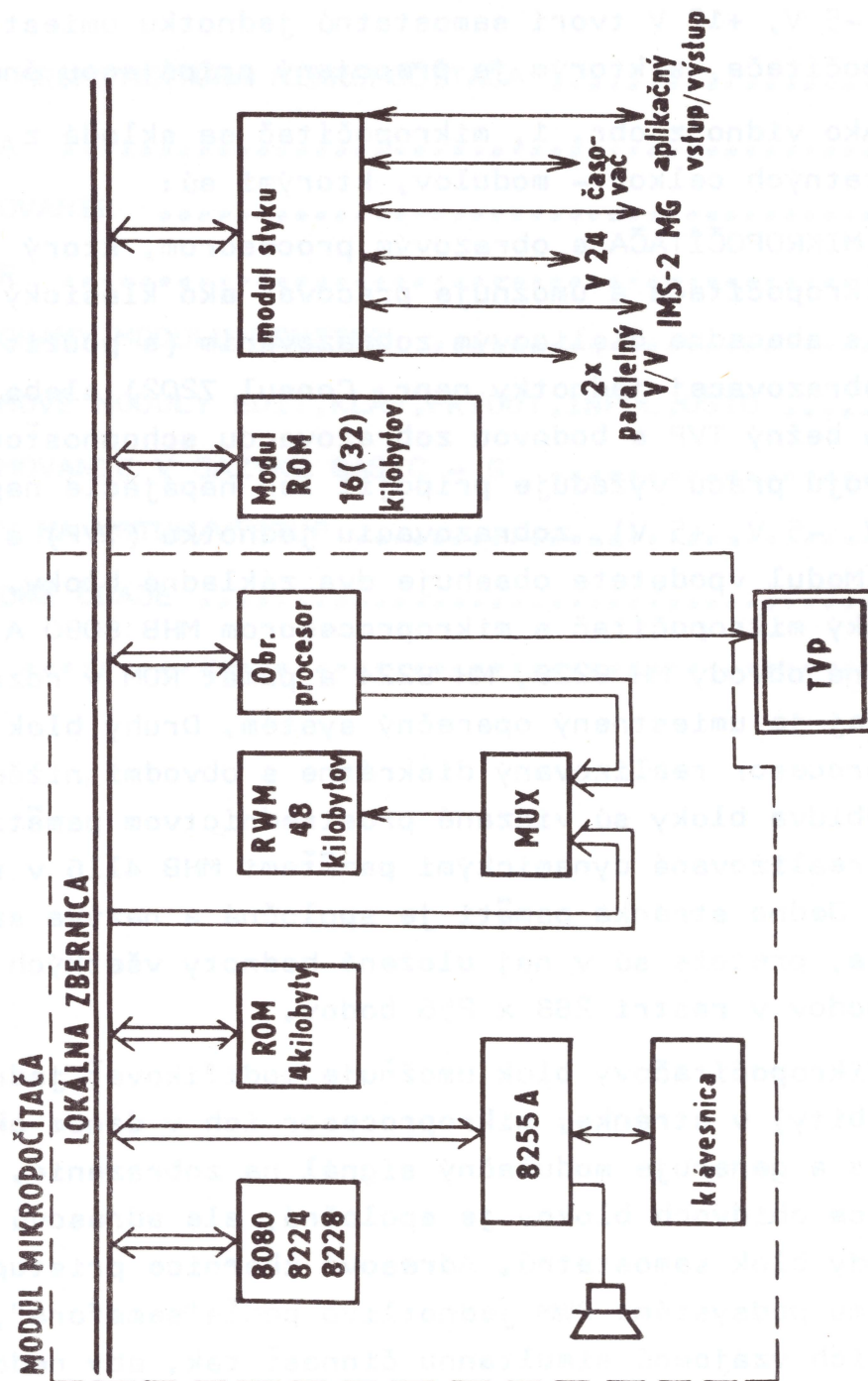
Komunikovanie s mikropočítačom je výlučne vizuálne. Pre dobré využitie vlastností a funkčných možností je dôležité oboznámiť sa so slovníkom, so znakmi a ovládacími prvkami, ktoré sa pri práci s mikropočítačom používajú.

Mikropočítač PMD - 85 sa so svojou architektúrou a možnosťami zaraďuje medzi osobné mikropočítače (Personal Computers). Pri jeho konštrukcii bola zvolená stratégia minimálneho obvodového riešenia orientovaného na stavebné prvky mikroprocesorovej rodiny 8080 s maximálnou podporou programovacích prostriedkov.

Daná architektúra umožňujúca zobrazovať (1 bit = 1 bod) obsah určenej pamäťovej stránky na zobrazovacej jednotke (ktorou môže byť monitor TV alebo bežný televízny prijímač) i napriek svojej jednoduchosti dovoľuje využiť mikropočítač v celom rade aplikácií.

Hlavnou prednosťou mikropočítača PMD - 85 je grafické spracovanie údajov pod vyšším programovacím jazykom BASIC GRAPHICS (ďalej len BASIC - G) a rozsiahle programové vybavenie, úlohou ktorého je naplniť pamäťovú obrazovú stránku bodovo orientovanú a organizovanú v rozsahu 288 x 256 bodov (šírka x výška).





Obr.1 Architektúra počítača PMD - 85

Napájací zdroj pre mikropočítač s napájacím napätím +5 V, -5 V, +12 V tvorí samostatnú jednotku umiestnenú mimo mikropočítača, s ktorým je prepojený pripájacou šnúrou.

Ako vidno z obr. 1, mikropočítač sa skladá z niekoľkých samostatných celkov - modulov, ktorými sú:

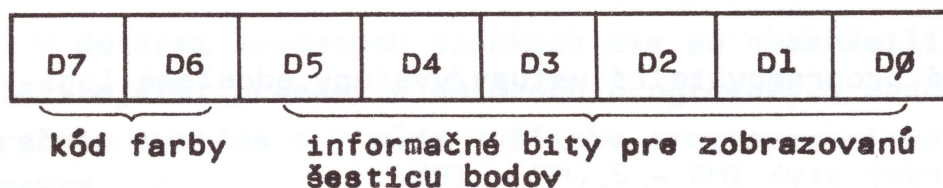
MODUL MIKROPOČÍTAČA s obrazovým procesorom, ktorý tvorí základ mikropočítača a umožňuje pracovať ako klasický mikropočítač s abecedne číslícovým zobrazovaním (s použitím externej zobrazovacej jednotky napr. Consul 7202) alebo s výstupom na bežný TVP s bodovou zobrazovacou schopnosťou. Modul pre svoju prácu vyžaduje pripojiť len napájacie napätie (+12 V, -5 V, +5 V), zobrazovaciu jednotku (TVP) a klávesnicu. Modul v podstate obsahuje dva základné bloky. Prvý je klasický mikropočítač s mikroprocesorom MHB 8080 A, jeho podporné obvody MH 8228, MH 8224 a pamäť ROM v rozsahu 4 kB, v ktorej je umiestnený operačný systém. Druhý blok je obrazový procesor realizovaný diskretné s obvody nižšej integrácie. Obidva bloky sú viazané prostredníctvom pamäti RWM, ktorá je realizovaná dynamickými pamäťami MHB 4116 v rozsahu 48 kB. Jedna stránka pamäti je spoločná a nazýva sa obrazová stránka, pretože sú v nej uložené hodnoty všetkých zobrazovacích bodov v rasti 288 x 256 bodov.

Mikropočítačový blok umožňuje modifikovať jednotlivé body (bity) v stránke; mikroprocesor ich v danom okamžiku prečíta a generuje modulačný signál na zobrazenie. Dátová zbernica obidvoch blokov je spoločná, ale adresovú zbernicu má každý blok samostatnú. Adresové zbernice pristupujú k pamäťovému podsystemu RWM jednotlivo podľa "semafora", ktorý riadi ich vzájomnú simultannu činnosť tak, aby nedošlo ku konfliktovým situáciám v lokálnej zbernici.

Prečítaný údaj z obrazovej stránky (videostránka) má formát podľa obr. 2.



Formát videoúdaja obr. 2.



Z uvedených osem bitov sa prvých šesť (D0 - D5) používa na zobrazenie šiestich bodov. Zvyšné dva (D6, D7) sú určené pre kód farby (v prípade FTV), prípadne na zmenu jasu o 50 % alebo na blikanie (pri ČBTv).

MODUL KLÁVESNICE predstavuje samostatnú časť, ktorá je pripojená cez 20-kolíkový FRB konektor na základný predchádzajúci modul. Klávesnica je riešená ako matica spínačov s organizáciou 15 stĺpcov x 5 riadkov. Klávesnica obsahuje 3 druhy klávesov, a to:

- štandardné klávesy ASCII,
- riadiace klávesy,
- programové klávesy.

Zobrazenie celkovej klávesnice s rozložením jednotlivých klávesov, ich význam a použitie uvedieme v ďalšej časti.

MODUL ROM patrí svojou štruktúrou medzi nevolateľné vonkajšie pamäte. Obsahuje 16 pozícií pre pamäte ROM typ MHB 8708 alebo s menšou úpravou pre pamäte MHB 2716.

Adresácia pamäťového bloku je realizovaná programovacími prostriedkami s podporou stykového obvodu MHB 8255 A. Vzhľadom na to, že modul ROM nepatrí do oblasti operačnej pamäti, používa sa ako „knížnica“ programových modulov. Pamäťový modul ROM sa aktivuje len počas prenosu údajov. V základnej zostave počítača PMD - 85 v module ROM obsahuje na desiatich pozíciách (9 kB) prekladač BASIC - G a zvyšných sedem pozícií je prístupných užívateľovi.

MODUL STYKOV umožňuje styk s okolím v nasledujúcich rozhraniach:

- dva paralelné programovateľné vstupy/výstupy oddelené logikou TTL;
- jeden štandardný styk IMS - 2 (HP - IB);
- jeden sériový styk V.24 oddelený optočlenmi;
- jeden styk pre komerčný kazetový magnetofón (MG) s prenosovou rýchlosťou 1200 Bd; odporúčaný typ kazetového magnetofónu: K - 10, MIRA;
- jeden časovač s podporou hodín reálneho času realizovaný obvodom 8253;
- jeden aplikačný (všeobecný) vstup/výstup lokálnej zbernice mikropočítača oddelený obvodom TTL umožňujúci pripojiť 8 ľubovoľných stykových obvodov.

Všetky rozhrania sú programovateľné a s podporou výkonných príkazov v prekladači BASIC ich možno modifikovať na žiadanú periférnu jednotku. Okrem toho styky pre MG a IMS - 2 obsahujú kompletne stykové procedúry volateľné priamo príkazmi BASIC - G.

Uvedené moduly počítača sú vzájomne prepojené pomocou konektorov FRB do jedného celku a umiestnené sú v skrinke z termoplastu, pričom modul ROM je riešený ako samostatná zasúvacia jednotka.

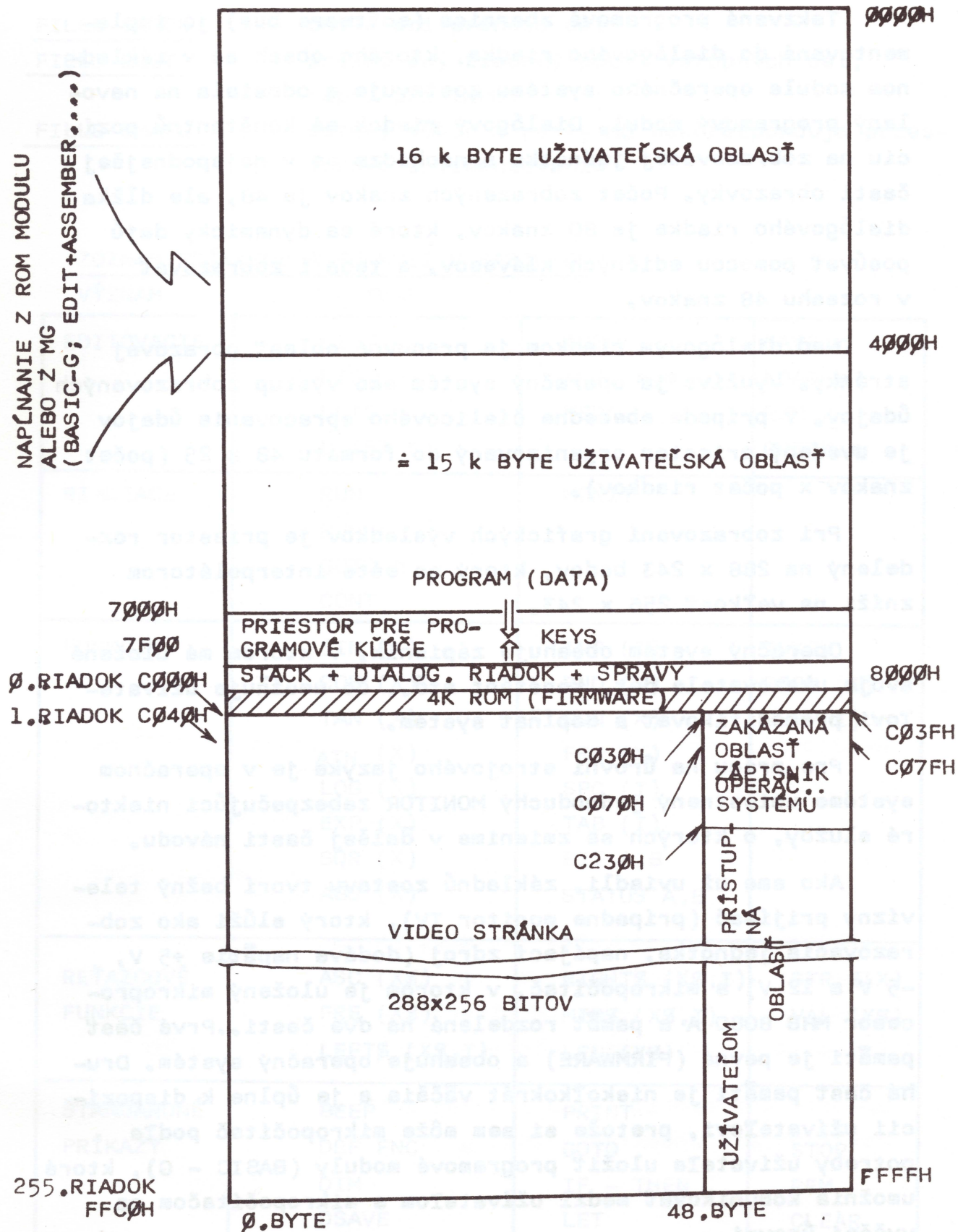
#### OPERAČNÝ SYSTÉM

Operačný systém v rozsahu 4 kilobyty je umiestnený v pamätiach ROM typu MHB 8608 a začína sa na adrese 8000 H. Užívateľovi je úplne k dispozícii pracovný priestor dvoch dolných stránok pamäti RWM od adresy 0000 H.

Operačný systém zabezpečuje organizáciu obrazovej stránky, ktorá sa začína od adresy C000 H, obsluhu klávesnice, styk pre MG, výpis znakov ASCII a organizáciu dialógového riadka. Rozdelenie pracovnej pamäti počítača PMD - 85 je na obr. č. 3.



NAPLNENIE Z ROM MODULU  
ALEBO Z MG  
(BASIC-G; EDIT+ASSEMBLER.....)



Obr.3 Rozloženie pamäte RWM

Takzvaná programová zbernica (software bus) je implementovaná do dialógového riadka, ktorého obsah sa v základnom module operačného systému zostavuje a odosiela na navolaný programový modul. Dialógový riadok má konštantnú pozíciu na zobrazovacej jednotke a nachádza sa v najspodnejšej časti obrazovky. Počet zobrazených znakov je 48, ale dĺžka dialógového riadka je 80 znakov, ktoré sa dynamicky dajú posúvať pomocou edičných klávesov, a teda i zobrazovať v rozsahu 48 znakov.

Nad dialógovým riadkom je pracovná oblasť obrazovej stránky. Využíva ju operačný systém ako výstup zobrazovaných údajov. V prípade abecedne číslícového spracovania údajov je uvedený priestor organizovaný do formátu 48 x 25 (počet znakov x počet riadkov).

Pri zobrazovaní grafických výsledkov je priestor rozdelený na 288 x 243 bodov, ktorý sa ešte interpolátorom zníži na veľkosť 256 x 243.

Operačný systém obsahuje zápisník, v ktorom má uložené svoje ukazovatele dát, konštanty atď., čo umožňuje užívateľovi premodifikovať a dopĺňať systém.

Pre prácu na úrovni strojového jazyka je v operačnom systéme umiestnený jednoduchý MONITOR zabezpečujúci niektoré služby, o ktorých sa zmienime v ďalšej časti návodu.

Ako sme už uviedli, základnú zostavu tvorí bežný televízny prijímač (prípadne monitor TV), ktorý slúži ako zobrazovacia jednotka, napájací zdroj (dodáva napätie +5 V, -5 V a 12 V) a mikropočítač, v ktorom je uložený mikroprocesor MHB 8080 A a pamäť rozdelená na dve časti. Prvá časť pamäti je pevná (FIRMWARE) a obsahuje operačný systém. Druhá časť pamäti je niekoľkokrát väčšia a je úplne k dispozícii užívateľovi, pretože si sem môže mikropočítač podľa potreby užívateľa uložiť programové moduly (BASIC - G), ktoré umožnia komunikovať medzi užívateľom a mikropočítačom na vyššej úrovni.



V ďalších častiach návodu vás oboznámime s povelmi MONITORU, ktorý je súčasťou operačného systému mikropočítača, a s programovacím jazykom BASIC - G, pomocou ktorého budete s mikropočítačom najčastejšie konverzovať.

Aby sme Vám oboznamovanie s mikropočítačom uľahčili, postupne budeme uvádzať jednotlivé úkony, ktoré máte urobiť, a súčasne bude zobrazovaná i odpoveď mikropočítača.

Pri dodržiavaní rád a príkazov mikropočítača sa vyhnete nežiadúcim situáciám, ktoré môžu nastať pri strate programu.

Na začiatok dobrá rada: pri prípadnom neúspechu hľadajte najskôr chybu v zadaní úlohy (príkazu) pretože mikropočítač vykonáva iba to, čo sa mu dalo interpretovať.

## B. O B S L U H A

Po prvom oboznámení s mikropočítačom môžete pristúpiť k jeho pripojeniu na napájací zdroj a televízny prijímač. Ak máte externú pamäť mikropočítača, ktorá môže byť zastúpená odporúčaným kazetovým magnetofónom, a chcete ju použiť, pripojte magnetofón do príslušnej zásuvky na mikropočítač a prepínač voľby prepnete do polohy MG. Napájacie napätie, televízny prijímač a magnetofón na mikropočítač pripojte príslušnými pripájacími šnúrami. Pripojenie jednotlivých prístrojov na mikropočítač je znázornené na obr. 4.

Po zapnutí televízneho prijímača (alebo zobrazovacej jednotky) a zdroja napätia na znak, že je mikropočítač pripravený k činnosti, bude na spodnom riadku obrazovky zobrazená správa:

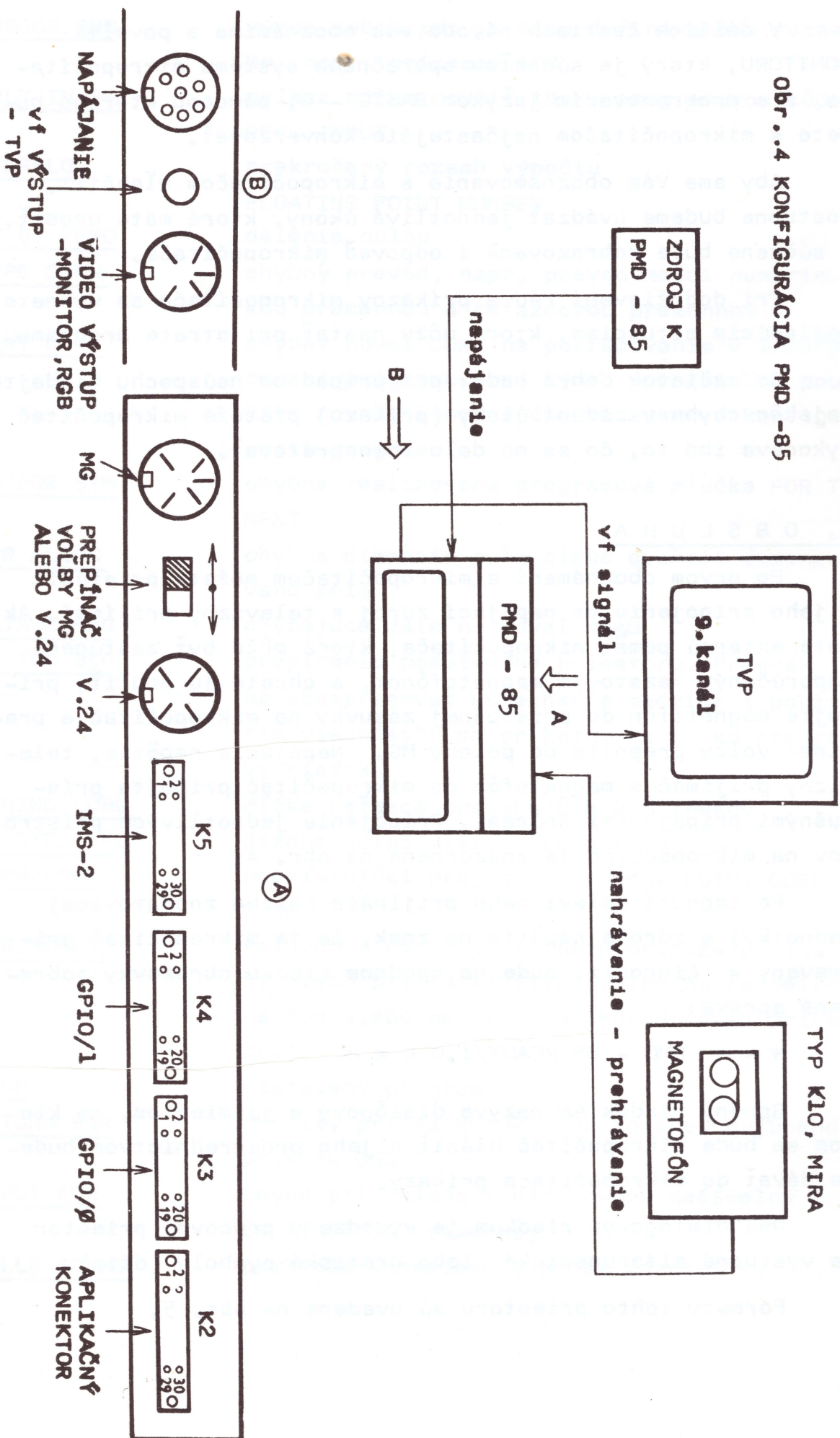
\* \* \* PMD - 85 READY/1,0 \* \* \*

Spodný riadok sa nazýva dialógový a je miestom, na ktorom sa bude mikropočítač hlásiť a jeho prostredníctvom budete dávať do mikropočítača príkazy.

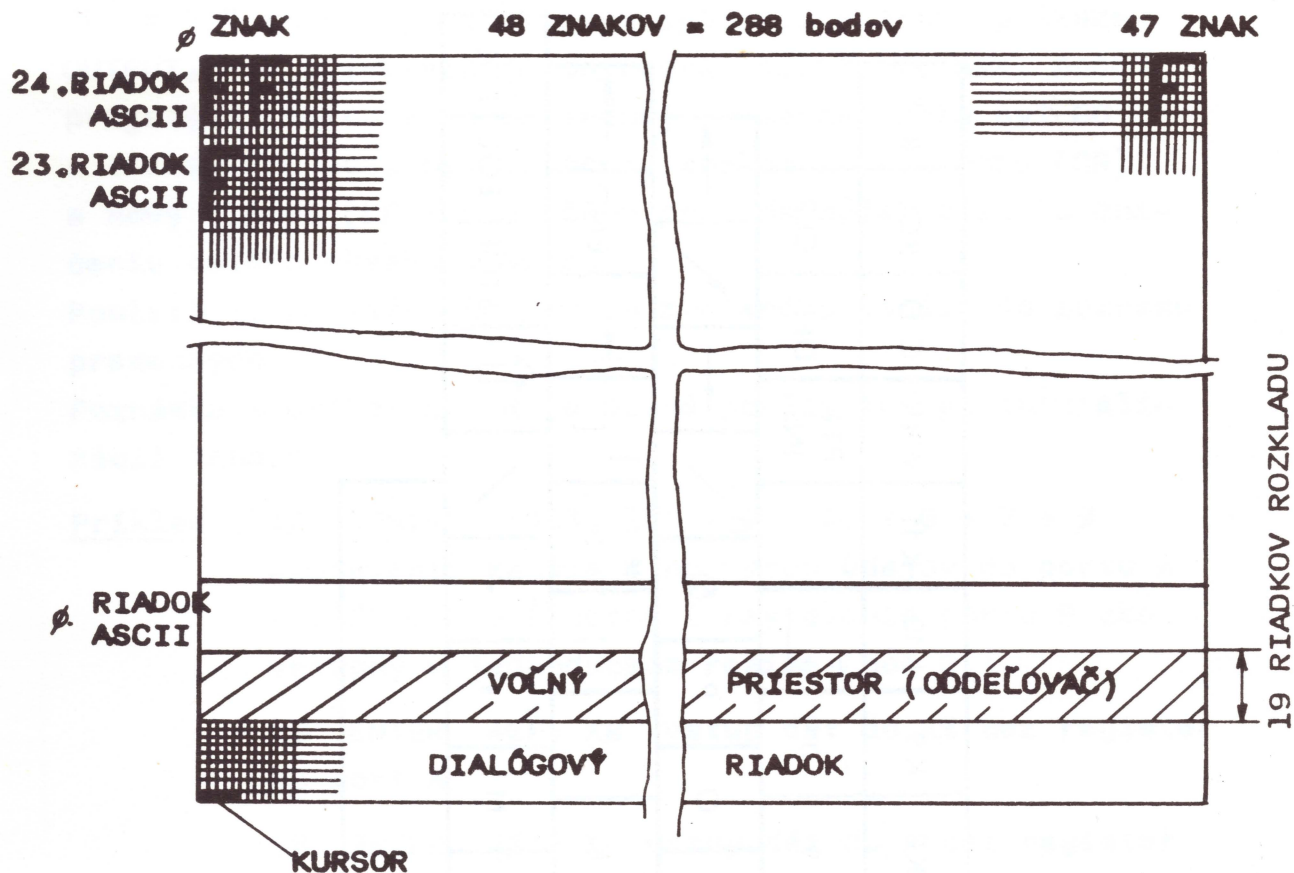
Nad dialógovým riadkom je vymedzený pracovný priestor na výstupné alfanumerické alebo grafické symboly (dáta).

Formáty tohto priestoru sú uvedené na obr. 5.

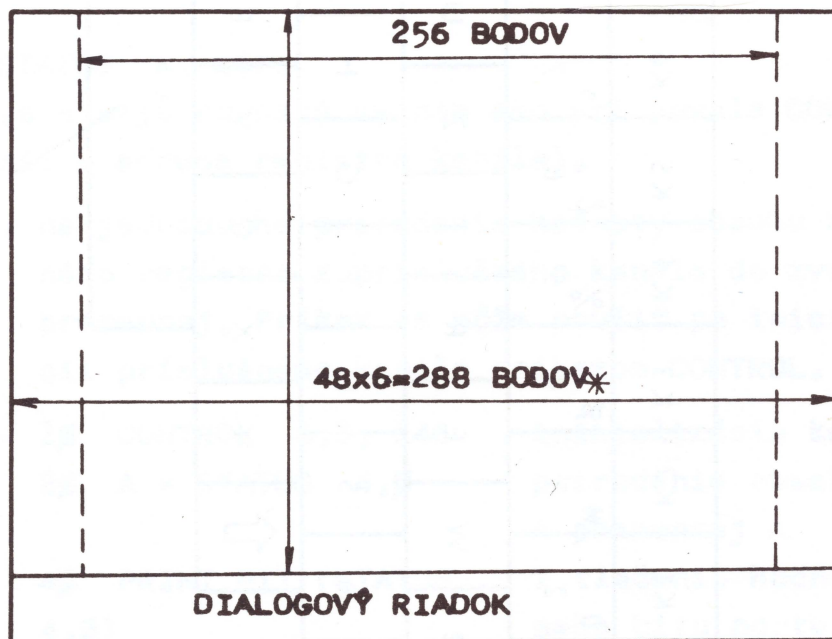
Obr..4 KONFIGURÁCIA PMD -85







a.) ALFANUMERICKÝ FORMÁT 48x25

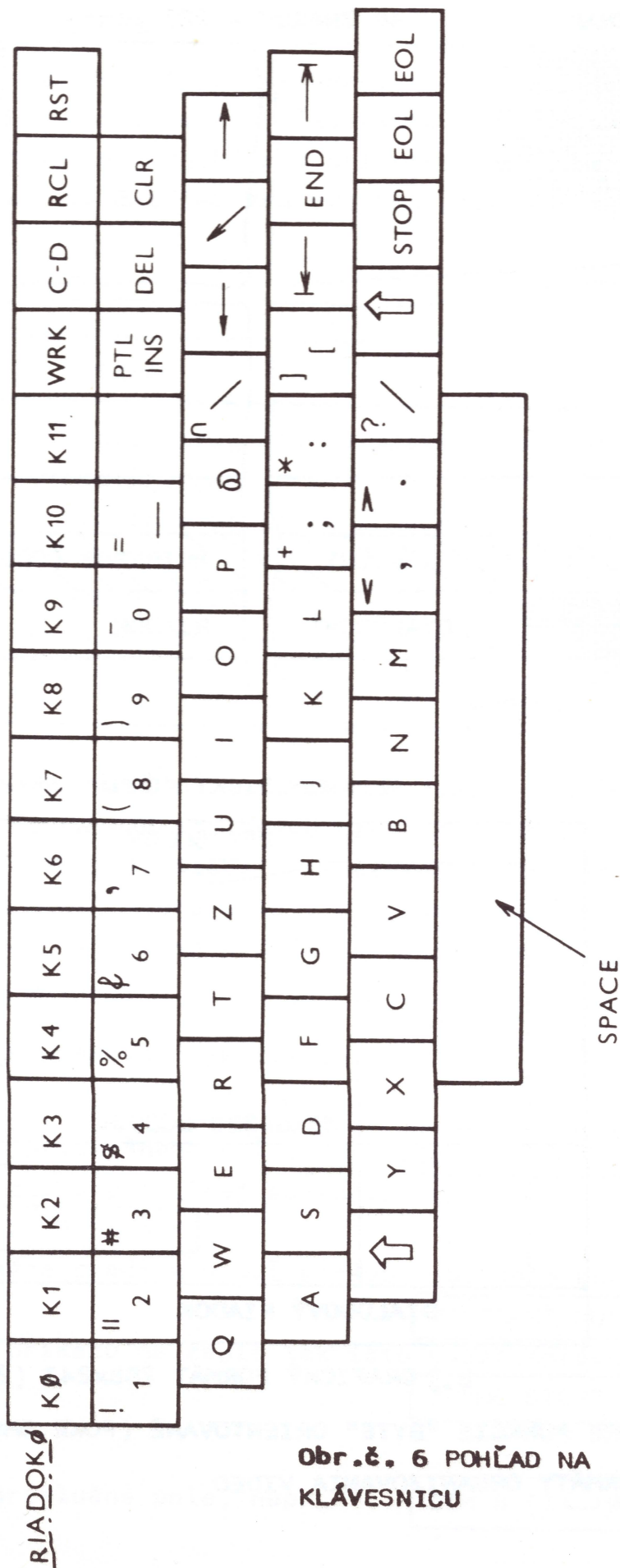


b.) GRAFICKÝ FORMÁT 288x243 (256x243)

\* PLATÍ PRE FUNKCIE "BYTE" ORIENTOvané (POKE, BMOVE, Bplot ....)

Obr.5 FORMÁTY ORGANIZOVANIA VIDEO

STĺPEČ ϕ




Obr.č. 6 POHĽAD NA  
KLÁVESNICU



Spôsob dávania príkazov do mikropočítača je uvedený v ďalších častiach.

#### VKLADANIE ZNAKOV

Každý príkaz je zostavený ako reťazec znakov ASCII. Sú to všetky znaky uvedené na klávesnici mikropočítača.

Klávesnica je zobrazená na obr. č.6. Niektoré klávesy majú priradené dva znaky. Horný znak je vyvolaný pri súčasnom stlačení klávesu  (SHIFT).



















Pokúste sa napísať pomocou uvedených klávesov svoje meno. Ak ste predtým nezatlačili žiaden kláves, mali by ste mať v dialógovom riadku text PMD-85 READY ... pred začiatkom vkladania prvého znaku Vášho mena. Ďalej mikropočítač zobrazí každý znak, ktorý ste zatlačili. Ak ste sa náhodou pomýlili a stlačili iný kláves, nič sa nestalo, lebo mikropočítač je pripravený aj na takéto situácie.


Na pravej strane klávesnice je skupina klávesov so zvláštnymi znakmi. To sú riadiace klávesy, pomocou ktorých možno opravovať text zobrazovaný v dialógovom riadku. Určite ste si všimli, že dialógový riadok má značku, tzv. kurzor, predstavujúcu čiarku pod úrovňou znaku. Táto značka je zobrazovaná preto, aby ste poznali miesto pre znak, ktorý práve chcete umiestniť. Kurzor i celý text možno posúvať vľavo, vpravo, na začiatok prvého znaku alebo na koniec textu, možno vymazať časť textu od kurzoru vpravo alebo naraz celý riadok. Okrem toho máte možnosť vsunúť znak alebo zrušiť znak v texte dialógového riadka. Uvedenú činnosť precvičte na tomto príklade.

#### Príklad č. 1 - Demonštrácia editovania obsahu dialógového riadka

H  
P

```
*** PMD 85 READY / 1.0 ***  
H_  
HP_
```

|   |   |                  |
|---|---|------------------|
|    |    | HP - _           |
|    |   | HP - 8           |
|    |   | HP - 85 _        |
|    |   | HP - 8 <u>5</u>  |
|    |   | <u>H</u> P - 85  |
|    |   | <u>P</u> - 85    |
|    |   | P <u>=</u> 85    |
|    |   | P _ - 85         |
|    |   | P _ -85          |
|   |   | PM _ - 85        |
|  |   | PMD <u>=</u> 85  |
|  |   | PMD - 8 <u>5</u> |
|  |  | PMD - 8 _        |
|  |   | MD - 8 _         |
|  |   | PMD - 8 _        |
|  |   | -                |

Ak ste ukončili editovanie, odporúčame opäť sa vrátiť k príkladu. Ešte raz si na ľubovoľnom texte precvičte používanie týchto klávesov. Toto zopakovanie sa Vám v budúcnosti vynahradí. Teraz Vás oboznámime s používaním programových klávesov. Sú určené na rýchle a pohodlné písanie textov (reťazec znakov), ktoré sa viackrát opakujú. Na klávesnici sú označené znakom K a číslicou (K0 + K11). Ak by Vám tento počet (12) nestačil, môžete použiť kláves , ktorý Vám umožní tento počet zdvojnásobiť na 24.



S týmito klávesmi súvisí kláves **WRK**, ktorý pri zatlačení dá povel mikropočítaču, že sa bude zapisovať celý text dialógového riadka. Zapísaný bude do pamäťového miesta, ktoré zodpovedá stlačenému programovému klávesu. Ak ste si to rozmysleli a nechcete modifikovať žiaden programový kláves a ak ste už stlačili kláves **WRK**, stačí stlačiť iný kláves, napr. **↵** a túto voľbu zrušíte. Opak zápisu na programový kláves je čítanie, teda výpis obsahu pamäťového miesta zodpovedajúceho príslušnému programovému klávesu do dialógového riadka. Tento úkon je jednoduchý, pretože stačí stlačiť žiadaný kláves a obsah (ak bol vložený) sa vypíše do dialógového riadka. Ak ste stlačili kláves, ktorý má nulový obsah, vypíšem Vám správu

**\*\* NO KEY \*\***

do dialógového riadka a stlačením ľubovoľného klávesu ju môžete zrušiť. Precvičte si to na nasledujúcom príklade:

#### Príklad č. 2 - Používanie programových klávesov

|       |     |    |   |   |   |       |                          |
|-------|-----|----|---|---|---|-------|--------------------------|
| CLR   | P   | M  | D | - | 8 | 5     | PMD-85                   |
| KØ    |     |    |   |   |   |       | ** NO KEY **             |
| SPACE |     |    |   |   |   |       | PMD - 85                 |
| WRK   | KØ  |    |   |   |   |       | ?                        |
| P     | O   | C  | I | T | A | C     | POCITAC                  |
|       |     |    |   |   |   |       | POCITAC                  |
| KØ    |     |    |   |   |   |       | POCITAC PMD - 85         |
| WRK   | KØ  |    |   |   |   |       | ?                        |
| O     | S   | O  | B | N | Y | SPACE | KØ OSOBNY POCITAC PMD-85 |
| CLR   | WRK | KØ |   |   |   |       | ?                        |
| KØ    |     |    |   |   |   |       | -                        |

Iste nie je potrebné vymenovať výhody, ktoré prináša používanie programových klávesov pri zostavovaní programu. Zistíte to predovšetkým pri úspore písania a hľadania znakov na klávesnici.

Na obsahy týchto programových klávesov je v mikropočítači rezervovaný dostatočný priestor v pamäti, ale v prípade iných, žiadanejších úloh sa tento priestor znižuje a môže nastať taká situácia, že si mikropočítač nebude môcť zapamätať (24x80) maximálne možné obsahy. Vtedy mikropočítač podá správu:

**\*\* MEMORY OVERFLOW \*\***

Keď dostanete túto správu, zvážte, ktorý programový kláves "zrušíte" prázdny textom (vymazaný dialógový riadok). Na záver Vám prezradíme (hoci by ste na to neskôr prišli), že pri stlačení systémového nulovania **RST** si mikropočítač vynuluje podstatnú časť pamäte.

Ak ste toho názoru, že vkladanie znakov, tvorenie textov a ich dodatočnú úpravu (editovanie) ste bez problémov zvládli, pristúpime k vysvetleniu zvyšných klávesov, ktoré majú systémový význam. Sú to klávesy:

**RST**   **RCL**   **C-D**   **EOL**   a   **STOP**

a majú tento význam:

kláves **RST** umožňuje tzv. studený štart systému. Vtedy nastane tzv. inicializácia programovateľných stykových obvodov a pamäťových buniek. Používajte ju rozumným spôsobom najmä v prípadoch, keď zostane mikropočítač na mŕtvom bode a treba ho znova oživiť. Aby nenastalo chybné stlačenie, je tento kláves závislý od stlačenia klávesu **↑** (SHIFT),

kláves **EOL** patrí medzi najdôležitejšie klávesy na klávesnici. Jeho stlačením dostáva mikropočítač impulz, že ste skončili vkladanie znakov do dialógového riadka, t.j. že súhlasíte s jeho obsahom. Bezprostredne po uvoľnení klávesu **EOL** si mikropočítač zapamätá jeho obsah na prípadné ďalšie použitie. Až potom ho posiela svojimi inštrukciami na dešifrovanie. Ďalšia činnosť závisí od konkrétneho obsahu dialógového riadka,

kláves **RCL** Vám umožňuje opäť privolať do dialógového riadka obsah posledného platného riadka, ktorý bol potvrdený klávesom **EOL**

kláves **C-D** - všetky správy, ktoré mikropočítač oznamuje,



budú zobrazené v dialógovom riadku. Po stlačení ľubovoľného klávesu sa táto správa vymaže. Ak by ste sa potrebovali pozrieť na posledné systémové hlásenie, stlačte kláves **C-D**, kláves **STOP** - už jeho názov hovorí, že ho treba stlačiť vtedy, ak chcete mikropočítač prerušiť v práci, ktorú vykonáva. Platí to však iba vo vymedzených prípadoch, o ktorých sa dozviete v ďalších kapitolách.

Ak ste pri štúdiu doposiaľ uvedených informácií pochopili význam jednotlivých klávesov a dokážete ich prakticky použiť, môžete pristúpiť k ďalšiemu oboznámeniu sa so zobrazovacími schopnosťami mikropočítača. V opačnom prípade sa vráťte späť a po opätovnom zopakovaní už uvedených informácií, pokračujte.

### C.. ZOBRAZOVANIE

Na zobrazovanie údajov, symbolov, znakov a pod. slúži televízna obrazovka. Na nej je vyhradená pracovná časť, do ktorej máte možnosť vstúpiť iba prostredníctvom výstupu Vášho programu. Nachádza sa nad dialógovým riadkom. Na obr. č. 5 sú znázornené dva formáty tejto pracovnej časti:

alfanumerický - umožňuje zobrazovať ľubovoľné znaky v rastru 6x8 bodov, ktoré sú organizované do 25 riadkov po 48 znakov. V svojom základnom vybavení obsahujú tvary tých alfanumerických znakov, ktoré sa nachádzajú na čiernych klávesoch klávesnice. Na jednom riadku môže mikropočítač zobraziť len 48 znakov, potom očakáva kód na návrat a posuv nového riadka. Ak tento kód (ØD) nedostane, bude sa zdržiavať na konci riadka. Písanie alfanumerického formátu sa vykonáva podobným spôsobom ako pri písacom stroji, teda smerom zdola nahor,

grafický - možno povedať, že aj predchádzajúca časť bola grafická, lebo každé zobrazenie alfanumerického znaku je vykonávané ako zobrazenie jednotlivých bodov v znaku. Rozdiel od tohto formátu spočíva najmä

v tom, že nie je organizovaný, ale predstavuje maticu bodov horizontálne 288 a vertikálne 243.

V základnom programovom vybavení má mikropočítač špeciálny program nazývaný interpolátor, ktorý pri zadaní súradníc začiatočného a koncového bodu umožňuje ich spojenie.

**Treba poznamenať, že video** pamäť mikropočítača je organizovaná v byte (t.j. 8 bitov). Nie všetky bity sú zobrazované; je zobrazovaných iba prvých 6 bitov, ostatné 2 slúžia ako farebné, prípadne špeciálne ich kódy. To pri čiernobielej verzii TVP umožňuje túto šesticu bodov alebo jej jednotlivé bity súčasne rozsvietiť plným jasom, polovičným jasom alebo kombináciou s blikaním. Pri alfanumerickom formáte nepostrehnete žiadne interferencie pri vzájomných šesticich bodov, ale pri grafickom (bodovom) zobrazovaní treba počítať s týmto javom. Nie je totiž možné, aby boli v danej šestici bodov zobrazené niektoré body plným a iné polovičným jasom. Na takýto spôsob pamäť mikropočítača nestačí. Toto na začiatok, v ktorom ste sa oboznámili so vstupnou a výstupnou stranou komunikácie, stačí. Pomocou týchto znalostí budete môcť pokračovať v sledovaní nasledujúceho výkladu, ktorý je zameraný na súbor (množinu) príkazov, pomocou ktorých môžete mikropočítač riadiť v hexadecimálnej forme - strojovým jazykom. Tento súbor príkazov sa nazýva MONITOR.

#### D. MONITOR

Toto programové vybavenie patrí medzi základné a umožňuje vykonávať úkony, nad pamäťovým priestorom mikropočítača. Pamäťový priestor je rozdelený tak, ako ukazuje obr. č. 3. Na obrázku vidno, že mikropočítač má veľkú operačnú pamäť, do ktorej môžete ukladať svoje programy. Špeciálna časť pamäťového priestoru je časť (12 k byte), v ktorej je uložený obrazec zobrazený na tienidle obrazovky TVP. Táto oblasť sa začína adresou C000 H. Ak je vám jasný pamäťový priestor, pristúpime k vysvetleniu príkazov monitoru:

|      |                                 |
|------|---------------------------------|
| SUB  | - modifikácia pamäťového miesta |
| MEM  | - výpis časti pamäti            |
| DUMP | - výpis stránky pamäti          |



|         |  |
|---------|--|
| JUMP    | - štart programu                             |
| MGSV    | - zápis na pásku kazetového magnetofónu (MG) |
| MGLD    | - čítanie z MG                               |
| MGEND   | - kontrola zápisu na MG                      |
| JOB     | - volanie programového modulu                |
| BASIC-G | - volanie programu BASIC                     |

Každý iný reťazec znakov je pre mikropočítač neznámy a na obrazovke vypíše správu

\*\* NO COMMAND \*\*

Okrem toho musíte vedieť, že pri práci v strojovom jazyku môžete vkladať do adresového a dátového poľa príkazov iba hexadecimálne znaky, a to: 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, F. V opačnom prípade je zobrazená správa

\*\* ERROR IN ADRESS \*\*

alebo \*\* ERROR IN DATA \*\*

Najprv sa oboznámite s príkazmi, ktoré umožňujú prehliadať a modifikovať pamäťové miesta..Sú to príkazy SUB, MEM a DUMP,

príkaz SUB - tvar príkazu: SUB    adresa dáta,

kde adresa znamená štvormiestne hexadecimálne číslo z priestoru pamäti RWM,

dáta sú dvojice hexadecimálneho čísla; ak zabudnete vložiť z tejto dvojice druhý znak, je zobrazená správa

\*\* ERROR IN DATA \*\*

alebo \*\* NO DATA \*\*,

ak nenapíšete vôbec žiadne znaky do daného dátového poľa.

Ak chcete mať pri písaní lepší prehľad medzi jednotlivými dátami, môžete vkladať ľubovoľný počet medzier, ktoré bude mikropočítač tolerovať. Ak by ste mali problémy s prepočítaním znakov ASCII, môžete ich vkladať rovnako ako dáta, ale na to musíte mikropočítač upozorniť pred prvým týmto znakom a za posledným znakom (ak ešte mienite vkladať hexadecimálne dáta). Urobíte to zvláštnym znakom - apostrofom (').

Tento príkaz treba používať rozumne, lebo môže spôsobiť mikropočítaču ťažkosti, ak "prepíšete" niektoré dáta, ktoré mikropočítač používa pri svojej práci. Radšej odporúčame na začiatku používať priestor od adresy 0000 - 7000 (HEX).

Príklad č. 3 - Demoštrácia príkazu SUB pri vkladaní do zvolenej pamäťovej lokácie.

|             |                     |
|-------------|---------------------|
| CLR         |                     |
| S U B SPACE | SUB _               |
| 0 0 0 0     | SUB 0000 _          |
| F F A B C D | SUB 0000 FFABCD _   |
| EOL         | SUB 0003 _          |
| 5 R         | SUB 0003 5R _       |
| EOL         | ** ERROR IN DATA ** |
| CLR RCL     | SUB 0003 5R _       |
| ← ← INS     | SUB 0003 _ 5R       |
| ↑ ,         | SUB 0003'5R         |
| EOL         | SUB 0005 _          |
| EOL         | ** NO DATA **       |

Ak ste sa zaoberali týmto príkladom, tak ste si určite všimli výhody klávesu **RCL**, ktorý vám umožnil vrátiť späť odoslaný obsah riadka, ktorý môžete ďalej použiť, prípadne opraviť.

#### príkaz MEM

Poslaním tohto príkazu je výpis dát do dialógového riadka od zvolenej pamäťovej lokácie.

Tvar príkazu: **MEM** adresa, kde adresa znamená adresované miesto, od ktorého mikropočítač vypíše 16 za sebou idúcich dát z tejto pamäťovej lokácie. Na lepšie čítanie a na ďalšie použitie s príkazom SUB je robený výpis vo formáte.

Prvý dátový znak sa vypisuje na 8 pozícii, druhý znak na 9 pozícii, potom je jeden znak medzera, atď.



Priestor na začiatku riadka má práve toľko voľných pozícií, koľko potrebuje príkaz SUB s udaním adresy.

V ďalšom príklade si vysvetlíme funkciu používania príkazu MEM a zopakujeme si niektoré predchádzajúce známe úkony.

Príklad č. 4 - Demonštrácia príkazu MEM

|                         |                       |
|-------------------------|-----------------------|
| CLR                     |                       |
| S U B SPACE Ø Ø Ø Ø     | SUB ØØØØ _            |
| WRK KØ                  | ?                     |
| KØ Ø 1 Ø 2 Ø 3 Ø 4      | SUB ØØØØØ1Ø2Ø3Ø4 _    |
| EOL                     | SUB ØØØ4 _            |
| CLR                     | _                     |
| M E M SPACE Ø Ø Ø Ø     | MEM ØØØØ _            |
| EOL                     | _ Ø1 Ø2 Ø3 Ø4         |
| RCL                     | MEM ØØØØØ1 Ø2 Ø3 Ø4   |
| ↖                       | MEM ØØØØØ1 Ø2 Ø3 Ø4   |
| KØ                      | SUB ØØØØØ1 Ø2 Ø3 Ø4   |
| INS INS Ø Ø             | SUB ØØØØØØØ1 Ø2 Ø3 Ø4 |
| EOL                     | SUB ØØ11 _            |
| CLR M E M SPACE Ø Ø Ø Ø | MEM ØØØØ _            |
| EOL                     | _ ØØ Ø1 Ø2 Ø3 Ø4      |

Ak ste pozorne pracovali podľa príkladu, mohli ste si všimnúť, že pomocou príkazov MEM a SUB môžete uskutočňovať vkladanie dát. Musíte si však uvedomiť, že príkaz MEM zobrazil iba 16 údajov a vloženie ďalších dát by sa nasledujúce dáta mohli zničiť.

Poznáte už dva príkazy Monitoru. Tieto príkazy stačia na modifikáciu pamäti, ale mikropočítač poskytuje ešte jeden komfortný výpis obsahu pamäťovej lokácie. Tento výpis vykonáva do svojej pracovnej časti v alfanumerickom formáte. Príkaz má názov DUMP.

## príkaz DUMP

Tvar príkazu `DUMP [ ] adresa`, kde adresa znamená štvormiestne hexadecimálne číslo určujúce prvú začiatočnú adresu pamäťovej lokácie, ktorá sa bude vypisovať. Výpis začína v riadku tesne nad dialógovým riadkom v nasledujúcom formáte:

|        |          |            |
|--------|----------|------------|
| ADRESA | 8 x DATA | ASCII TVAR |
|--------|----------|------------|

Zvláštnosťou tohto formátu je to, že pri zobrazovaní dát sa v prvej časti formátu zobrazujú ich alfanumerické tvary (ASCII). Ak dáta nie sú z tejto množiny, je zobrazená bodka alebo nedefinovateľný tvar znaku v rastru 6 x 7 bodov.

Po zobrazení prvého riadka nastáva tzv. rolovanie obsahu. Vtedy sa presunú riadky smerom nahor. Táto operácia zabera najviac času, pretože mikropočítač musí posunúť asi 80 000 bodov v pracovnej časti a vymazať prvý riadok, aby ste mohli mať zobrazený ďalší riadok výpisu. Toto mikropočítač urobí až 25x a zostane v klude. Zatiaľ sa môžete rozhodnúť, či požadujete ďalšie výpisy. Potom nasleduje z mikropočítača dopyt správou

`CONTINUE?`

Ak požadujete ďalší riadok, stačí stlačiť kláves EOL a cyklus bude opakovaný dovtedy, kým nestlačíte iný kláves, odlišný od EOL.

Ako príklad na tento príkaz si urobíte výpis prvej stránky operačného systému mikropočítača od adresy 8000 H. Preto vložte do dialógového riadka príkaz

`D U M P SPACE 8 0 0 0`

a stlačte kláves EOL. Bezprostredne potom začne opísaným spôsobom výpis dát. Prerušenie mikropočítača v tejto činnosti nie je možné, iba "násilným spôsobom" - súčasným stlačením klávesov `↑ + RST`.

Pomocou nasledujúceho príkazu Monitoru uskutočnite štart programu zostaveného už známymi príkazmi. Ak ho chcete používať, je potrebné, aby ste dôkladne poznali inštrukcie mikropočítača v strojovom kóde. Môžete sa ich, samozrejme, naučiť aj s mikropočítačom, ale v tom prípade nie je zaručený úspech.



Mikropočítač bude totiž vykonávať všetky za sebou idúce inštrukcie (kódy), ktoré ste si zaradili do programu.

### - príkaz JUMP

Tvar príkazu **JUMP** adresa, kde adresa predstavuje štvormiestne hexadecimálne číslo udávajúce štartovaciu adresu.

Ešte chceme upozorniť, že ide o dôležitý príkaz, ktorý zbavuje mikropočítač zodpovednosti nad riadením systému. Riadenie preberá váš užívateľský program. Mikropočítač ešte stihne podať správu

\* \* EXECUTIVE \* \*

Príklad č. 5 - Demonštrácia príkazu JUMP na zostavenom programe na výpis znaku "A" do pracovnej časti.

Program:

|           |        |   |
|-----------|--------|---|
| MVI A, 41 | 3E41   | ; uloženie znaku "A" do akumulátora       |
| STA C034  | 3234C0 | ; zápis akumulátora do preberajúcej bunky |
| CALL 8500 | CD0085 | ; výkonný podprogram na zápis znaku       |
| HLT       | 76     | ; zastavenie programu                     |

Uloženie programu a jeho odštartovanie od adresy 0000H je v nasledujúcich krokoch:

|     |     |     |   |       |   |     |   |   |                   |
|-----|-----|-----|---|-------|---|-----|---|---|-------------------|
| CLR | S   | U   | B | SPACE | 0 | 0   | 0 | 0 | SUB 0000          |
| 3   | E   | 4   | 1 | EOL   |   |     |   |   | SUB 0002          |
| 3   | 2   | 3   | 4 | C     | 0 | EOL |   |   | SUB 0005          |
| C   | D   | 0   | 0 | 8     | 5 | EOL |   |   | SUB 0008          |
| 7   | 6   | EOL |   |       |   |     |   |   | SUB 0009          |
| CLR | M   | E   | M | SPACE | 0 | 0   | 0 | 0 | 3E 41 32 34 C0    |
| EOL | CLR |     |   |       |   |     |   |   |                   |
| J   | U   | M   | P | SPACE | 0 | 0   | 0 | 0 | JUMP 0000         |
| EOL |     |     |   |       |   |     |   |   | * * EXECUTIVE * * |

Súčasne s touto správou sa na začiatok prvého riadka v pracovnej časti zobrazí znak "A". Pretože tento program má poslednú inštrukciu HLT, môžete mikropočítač zaktivizovať iba prostredníctvom systémového nulovania RESET.



Na zopakovanie uvedieme, že sme sa naučili tieto príkazy: SUB, MEM a DUMP. Umožňujú prácu nad pamäťovým obsahom. Ďalší príkaz, ktorý už ovládáte, JUMP, je štartovacím príkazom vášho programu. Teraz sa budeme zaoberať ďalšími príkazmi, ktoré sú dopĺňajúce a majú špeciálny význam. Sú to príkazy na ukladanie a čítanie dát z vonkajšej pamäti, ktorá môže mať formu bežného kazetového magnetofónu alebo tzv. modulu ROM. Najprv si vysvetlíme, ako treba pracovať s vonkajšou pamäťou - s kazetovým magnetofónom (MG).

#### D. 1 Práca s kazetovým magnetofónom

Kazetový magnetofón predstavuje vonkajšiu pamäť, ktorá umožňuje uchovávať súbory dát, programov. Aby mikropočítač mohol tieto súbory vyhľadať, označuje si ich hlavičkou, ktorú musíte vyplniť. Hlavička je súčasťou daného príkazu. Pri tejto práci dávajte pozor, aby nenastalo prepísanie dát, lebo ovládanie pásky nie je pod "velením" mikropočítača. Zápis a čítanie dát sa koná rýchlosťou 1 200 Bd, čo znamená, že za 1 sek. je prenesených 1 200 bitov.

Ak ste už nastavili pásku a magnetofón na nahrávanie, prekontrolujte prepnutie prepínača voľby V24 - MG do polohy MG a vysvetlíme si, ako budete nahrávať dáta (binárne) na kazetu. Bude to ďalší príkaz.

#### príkaz MGSV

tvár príkazu: MGSV\_ file; od - do; komentár , kde file znamená dvojmiestne decimálne číslo v rozsahu 00 až 63 a znamená číselné označenie daného súboru alebo poradové číslo súboru na páske.

od - štvormiestne hexadecimálne číslo udávajúce adresu prvého pamäťového miesta, odkiaľ majú byť zaznamenané dáta,

do - štvormiestne hexadecimálne číslo udávajúce adresu posledného pamäťového miesta z danej lokácie. Musí platiť, že "od" < "do".

komentár - reťazec ôsmich znakov, ktoré slúžia ako poznámka. Komentár nemusíte uvádzať, ale je lepšie, keď máte pri neskoršom prehľadávaní súboru poznámku, ktorá slovne vystihuje daný súbor.



Použitie príkazu si vysvetlíme na ďalšom príklade.

Príklad č. 6 - Demonštrácia príkazu MGSV

|                     |                              |
|---------------------|------------------------------|
| CLR                 |                              |
| M G S V SPACE Ø Ø ; | MGSV Ø Ø ; _                 |
| Ø Ø Ø Ø - Ø 1 Ø Ø ; | MGSV O O ; ØØØØ-Ø1ØØ; _      |
| M E N O             | MGSV Ø Ø ; ØØØØ-Ø1ØØ; MENO _ |

Teraz uvoľníte tlačidlo STOP na kazetovom magnetofóne a pokračujte

|     |   |
|-----|---|
| EOL | ? |
|-----|---|

Po nahratí súboru mikropočítač vypíše správu a okamžite môžete zastaviť nahrávanie magnetofónu.

|         |          |
|---------|----------|
| Správa: | MG STOP! |
|---------|----------|

Takto mikropočítač oznamuje, že sa súbor nahral, ale odporúčame zistiť, či sa nahral bezchybne. Na to slúži nasledujúci príkaz:

príkaz MGEND

Tvar príkazu MGEND \_file, kde file predstavuje dvojmiestne hexadecimálne číslo udávajúce žiadaný súbor.

Príkaz MGEND má dve funkcie:

- zisťuje koniec žiadaného súboru, čo je potrebné na ďalšie pridávanie súborov,
- zisťuje správnosť záznamu, keďže sa na kazetovej páske môže vyskytnúť nečistota alebo chyba.

Toto mikropočítač zisťuje tak, že pri nahrávaní počíta všetky idúce dáta; ich osembitový súčet zapíše ako posledný údaj tak, že keď bude znova robiť súčet týchto dát z pásky musia sa tieto súčty rovnať. V opačnom prípade je podaná správa

" FILE ERROR"

Teraz sa pokúsime skontrolovať predošlý záznam.

Príklad č. 7 - Demonštrácia príkazu MGEND

Najprv si musíte pretočiť pásku na začiatok hľadaného (zaznamenávaného) súboru, ináč by ste ho nikdy nevyhľadali.

|     |   |   |   |   |   |       |   |   |           |
|-----|---|---|---|---|---|-------|---|---|-----------|
| CLR | M | G | E | N | D | SPACE | Ø | Ø | MGEND Ø Ø |
| EOL |   |   |   |   |   |       |   |   | ?         |

Spustíte magnetofón na prehratie a čakajte správu v dialógovom riadku, Ak mikropočítač našiel daný súbor, vypíše jeho hlavičku:

ØØ/? MENO

a v hlavičke zároveň doplní znak ?, lebo daný súbor bol uskutočnený z operačného systému (z Monitoru).

Ak bol záznam skontrolovaný a je správny, objaví sa prázdny riadok:

?

a môžete zastaviť magnetofón.

V opačnom prípade mikropočítač podá správu:

FILE ERROR

To bude impulz, aby ste znova opakovali nahrávanie daného súboru, samozrejme, i jeho kontrolu. Po úspešnej kontrole odporúčame viesť denník (záznamník) o danej kazete, aby nedošlo k deformácii jej obsahu.

Posledným príkazom na prácu s magnetofónom je čítanie zaznamenaného súboru, Umožňuje to príkaz MGLD.

-príkaz MGLD

Tvar príkazu: MGLD file

kde file je dvojmiestne decimálne číslo udávajúce číslo súboru, ktorý sa má prečítať a uložiť do operačnej pamäti.. Miesto, na ktoré sa bude ukladať, bolo zadané už pri jeho



zápise - je zadané v hlavičke súboru. Ak budete netrpezliví pri hľadaní daného súboru, môžete prácu mikropočítača s magnetofónom prerušiť stlačením klávesu **STOP**. Vtedy mikropočítač podá správu:

"FILE ERROR"

Pri vyhľadávaní daného súboru mikropočítač vypisuje každú hlavičku, cez ktorú prechádza. Ak nejde o daný súbor, vydá ešte akustický signál - krátky tón. Ak nájde daný súbor, nestihne už vydať akustický signál, lebo okamžite začína presúvať dáta z pásky do operačnej pamäti, pri súčasnej kontrole spomínaného súčtu. Ak ste ukladali označovanie súboru postupne a viete, že by ste na daný súbor čakali dlhšie, môžete pretočiť pásku a MG znova **spustiť na prehrávanie**. To znamená, že hľadanie daného súboru mikropočítačom, môžete manipulovať s magnetofónom podľa vlastného uváženia.

Príklad č. 8 - Demonštrácia príkazu MGLD:

**CLR** **M** **G** **L** **D** **SPACE** **Ø** **Ø** **EOL** **?**

Spustíte magnetofón na prehrávanie od začiatku pásky. Pri prechode hlavičky súboru vypisuje mikropočítač jej obsah

**ØØ/? MENO**

Po uložení posledného znaku a po kontrole súčtu sa dialógový riadok vymaže

**?**

To boli príkazy na prácu s magnetofónom. Viackrát si precvičte tieto príkazy, aby ste si ich osvojili a mohli bez problémov operatívne používať.

Teraz sa oboznámite s poslednými dvoma príkazmi Monitoru. Sú to špecifické príkazy a umožňujú kvalitatívne zvýšiť programové vybavenie mikropočítača.

Podmienkou ich využívania je prítomnosť modulu ROM s daným programovým vybavením. Tieto príkazy presúvajú do operačnej pamäti mikropočítača obsah modulu ROM a súčasne ho šartujú. Od tohto momentu pracuje mikropočítač s "vyššou inteligenciou".

Príkaz vyvolávajúci takýto programový modul je príkaz Basic-G.

#### príkaz Basic-G

Tvar príkazu BASICG

Po vložení tohto príkazu do dialógového riadka

BASIC GEOL

opúšťa mikropočítač operačný systém a prihlasuje sa novým hlásením a znakom ">". Ide o základné vyššie programové vybavenie, a to o BASIC s grafickými príkazmi. Popis tohto jazyka je podrobne uvedený v príslušnej literatúre.

Všeobecný príkaz na prenos programových modulov do operačnej pamäti je

#### -príkaz JOB

Tvar príkazu: JOB └ kód

kde kód je dvanásťmiestne hexadecimálne číslo, ktoré má priradený daný zvolený programový modul z modulu ROM.

Je to vlastne číslo zostavené z týchto parametrov:

xxxx      adresa v module ROM (odkiaľ)  
xxxx      dĺžka bloku (koľko)  
xxxx      adresa RWM (kam)

Na privolanie programového modulu BASIC - G by sme použili tento príkaz:

JOB SPACE    25      EOL

kde: 0000 je adresa prvej pozície pamäti ROM,

2500 je dĺžka bloku v rozsahu 9 k byte,

0000 je adresa pamäti RWM, kde sa začne ukladať programový modul.

#### E. PODPROGRAMY MODULU MONITOR

Okrem už uvedených štandardných povelov (v predchádzajúcich článkoch) mikropočítač môže užívateľovi poskytnúť



celý rad podprogramov z programového modulu MONITOR. Tieto podprogramy môže užívateľ s výhodou využiť pri zostavovaní vlastných programov. Predpokladom však je znalosť systému 8080 a jeho programovanie.

HEX - podprogram na prevod čísla ASCII do HEX. V prípade chyby, ak nejde o znak z množiny hexa znakov (0 F), sa nastavuje CARRY bit.

Vstup: akumulátor

Výstup: akumulátor a príznakový bit CARRY

Volanie: 80E0H

Mení obsah registrov: PSW

PAIR IN - prevod 2 x ASCII do HEXA,

Vstup: ukazovateľ HL

Výstup: akumulátor a príznakový bit CARRY

Volanie: 80F7H

Mení obsah registrov: PSW, HL, B

ADRIN - prevod 4x ASCII do 2 x HEXA

Vstup: ukazovateľ HL

Výstup: registrovaný pár DE, príznakový bit CARRY

Volanie: 8109H

Mení obsah registrov: všetkých

PREVOD 1 - prevod HEXA na dva ASCII pri ich súčasnom zobrazovaní do pracovnej časti obrazovky. Využíva podprogram PRTOUT.

Vstup: znak HEXA v akumulátore

Výstup: podprogram PRTOUT (8500)

Volanie: 8125 H

Mení obsah registrov: B, PSW

PREVOD 2 - prevod HEXA na dva ASCII a ich uloženie do pamäti na miesto určené ukazovateľom v HL

Vstup: znak HEXA v akumulátore, ukazovateľ HL

Výstup: pamäťové miesto určené v HL

Volanie: 813BH

Mení obsah registrov: B, PSW

BIN/BCD - prevod binárneho čísla (HEXA) na BCD v rozsahu (0-63)

Vstup: akumulátor

Výstup: akumulátor

Volanie: 8E73H

Mení obsah registrov: HL, B, PSW

TRANSFER - prenáša obsah z modulu ROM do operačnej pamäti.

Argumenty sa ukladajú za voláciu inštrukciu v poradí: počiatková adresa ROM, dĺžka bloku, ukladacia adresa RAM.

Volanie: 8C00H

Mení obsah registrov: všetkých

Príklad: CALL TRANSFER

C D 0 0 8 C

DW ADR 1

0 0 0 0

DW BATCH

0 0 2 5

DW ADR 2

0 0 0 0

STOP - podprogram zabezpečuje stlačenie klávesu STOP. V prípade, že sa tak stalo, akumulátor nadobudne hodnotu 03.

Vstup: kláves STOP

Výstup: PSW

Volanie: 8C74H

Mení obsah registrov: PSW

BREAD - podprogram slúžiaci ako podpora pre grafický BASIC a umožňujúci načítať hodnotu byte z pracovnej časti obrazovky, z miesta, na ktoré ukazuje kurzor.

Vstup: ukazovateľ miesta (kurzor) C17AH

Výstup: akumulátor

Volanie: 8473H

RPOINT - podprogram slúžiaci ako podpora k interpolátoru (BASICu) a umožňujúci zo zadaných súradníc bodu zistiť jeho stav (0 alebo 1). Využíva podprogram POINT a s ním súvisiace jeho vstupné argumenty.



Vstup: argumenty podprogramu POINT

Výstupy: akumulátor nadobudne hodnoty **00 alebo 01**

Volanie: 85E6H

Mení obsah registrov: všetkých

MONIT - podprogram zabezpečujúci vstup do monitoru, nastavuje prompt znak "?" a vypisuje text "OS READY" do dialógového riadka.

Volanie: 8C40H

#### F. PROGRAMOVÉ MODULY EDIT, KLÁV, PRTOUT, INPOL, OSIO

V predchádzajúcej časti textu sme vás oboznámili s programovým modulom MONITOR. V ďalšej časti si rozoberieme ďalšie programové moduly operačnej pamäti mikropočítača.

#### EDIT

Tento programový modul zabezpečuje interaktívnu činnosť s obsahom dialógového riadka. Pri prijatí vstupného argumentu mikropočítač rozhodne, či ide o alfanumerický alebo riadiaci znak a v prípade alfanumerického znaku vykoná okamžité rozkreslenie nad pozíciou kurzoru. Ostatné prijaté znaky roztriedi a vykoná príslušný podprogram. Ide o tieto riadiace znaky:

|       |       |   |
|-------|-------|---|
| →     | ..... | kurzor vpravo   |
| ←     | ..... | kurzor vľavo  |
| ←     | ..... | riadok vľavo  |
| →     | ..... | riadok vpravo   |
| CLR   | ..... | riadok vymaž  |
| ↑ CLR | ..... | od kurzoru do konca riadka vymaž                              |
| ↑ PTL | ..... | zobraz do pracovnej oblasti celý riadok                       |
| DEL   | ..... | vymaž znak v mieste kurzoru                                   |
| INS   | ..... | ulož medzeru v mieste kurzoru                                 |
| ↖     | ..... | kurzor na začiatok riadka                                     |
| END   | ..... | kurzor na posledný znak v riadku                              |
| WRK   | ..... | bude sa zapisovať do programových klávesov (tzv. soft - keys) |

RCL ..... privolanie odoslaného riadka  
 EOL ..... .. ukončenie práce nad obsahom dialógového  
 riadka  
 C-D .....privolanie návestia od OS

Volanie: adr, 8800H

Vstupný argument: znak na adresu C134H

Okrem uvedených edičných **podprogramov má mikropočítač** ďalšie podprogramy, ktoré môžu byť k dispozícii užívateľovi.

Sú to:

BELL - podprogram zabezpečujúci akustický výstup.

Volanie: adr, 88A3H

Mení obsah registrov: PSW, DE, HL

V prípade vášho vlastného komponovania akustického výstupu je možné nastaviť ukazovateľ tabuľky do registra HL a volanie uskutočniť s adresou 88A6H.

Zloženie tabuľky:

|     |      |  |     |      |       |
|-----|------|--|-----|------|-------|
| TÓN | WAIT |  | TÓN | WAIT | MARKA |
|-----|------|--|-----|------|-------|

pričom: TÓN - môže nadobudnúť hodnoty 00, 01, 02, 03,

WAIT - hodnota 00 až FF,

MARKA - hodnota FF ukončuje činnosť a znamená výstup z podprogramu BELL.

Príklad: LXI H, TAB

6 0 0 0

21 06 60

JMP BELL + 1

63 A6 88

TAB: DB TON 1

01

DB WAIT 1

50

DB TON 2

02

DB WAIT 2

80

DB MARKA

FF

Príklad znázorňuje vytvorenie podprogramu s volacou adresou 6 0 0 0, s akustickým, výstupom daným hodnotami uloženými na TAB (adresa 6 0 0 6).



ZOBR BUF - podprogram vykoná zobrazenie reťazca uloženého v pamäti, ktorého začiatok je daný ukazovateľom v registroch HL, do dialógového riadka.

Volanie: adr. 8858 H

Mení obsah registrov: všetkých

Vstupný argument: adresa začiatku reťazca do HL.

PRTTEXT - vypísanie reťazca ASCII do dialógového riadka s možnosťou akustického návestia.

Volanie: adr. 8AB9H + BELL

adr. 8A8CH bez BELL

Mení obsah registrov: všetkých

Vstupný argument: ukazovateľ začiatku reťazca uložený v zápisníku na adresu

C074H a C075H.

Poznámka: reťazec ASCII musí byť ukončený znakom 0DH!

#### KLAV

Zabezpečuje priradenie kódu ASCII znakov a riadiacich znakov z klávesnice, ktoré sú umiestnené v tabuľke od adresy 8400 H tak, že každá päťnástina obsahuje príslušnú značku určujúcu kód riadka.

Ak mikropočítač nájde príslušný riadok (značku), potom pripočíta pozíciu klávesu v riadku ( $\overline{0}14$ ), a tým určí príslušný bod znaku. Ten je možné prevziať priamo z akumulátora alebo zo zápisníka adresy C134H. Adresový vektor programovacieho modulu je 84A1 a uchováva všetky registre okrem PSW.

#### PRTOUT

Pomocou tohto programového modulu mikropočítač organizuje rozkreslenie znaku ASCII do videostránky pamäti RWM na miesto, ktoré mu udáva argument kurzoru.

Vstupné argumenty: akumulátor - znak výpisu

|                                |                           |
|--------------------------------|---------------------------|
| údaje do zápisníka - C 0 3 E H | } pozícia výpisu (kurzor) |
| C 0 3 F H                      |                           |
| - C 0 3 A H                    | kód farby                 |
| - C 0 3 C H                    | } tabuľka znakov          |
| C 0 3 D H                      |                           |

Uvedené údaje má mikropočítač inicializované takto:

C 0 3 E 0 0                      C 0 3 A - 0 0 (80, C0, 40)

C 0 3 F FB

C 0 3 C 00

C 0 3 D 85

Volanie: adr. 8500 H

Mení obsah registrov: PSW

Tento programový modul obsahuje výkonný podprogram, ktorým mikropočítač vyhľadá z tabuľky znakov príslušný kód a rozkreslí ho do nastavenej časti pamäti. Je volený cez adresu 8584 H a používa uvedené argumenty.

Na obr. 7 je uvedená tabuľka znakov a z nej je zrejmé, ako môže mikropočítač modifikovať ľubovoľné typy, prípadne symboly v rastri 6 x 8 bodov. Treba zdôrazniť, že hodnotu ukazovateľa tabuľky nastavuje mikropočítač pri rozkreslení znakov v rastri 5 x 7 zmenšenú o FFH a pri plnom vykreslení v rastri 6 x 8 bodov zmenšenú o FEH. Je to z dôvodu rýchlejšieho prenosu pri alfanumerickom výstupe.

Vrátime sa ešte k exekutive PRTOUT. Musíme však dodať že ako vstupný znak okrem štandardných alfanumerických znakov ASCII môže mikropočítač prijať i tieto znaky:

LF (0AH) - prázdny znak

CR (0DH) - návrat na nový riadok + posun

FS (1CH) - nulovanie obrazovky

#### INPOL

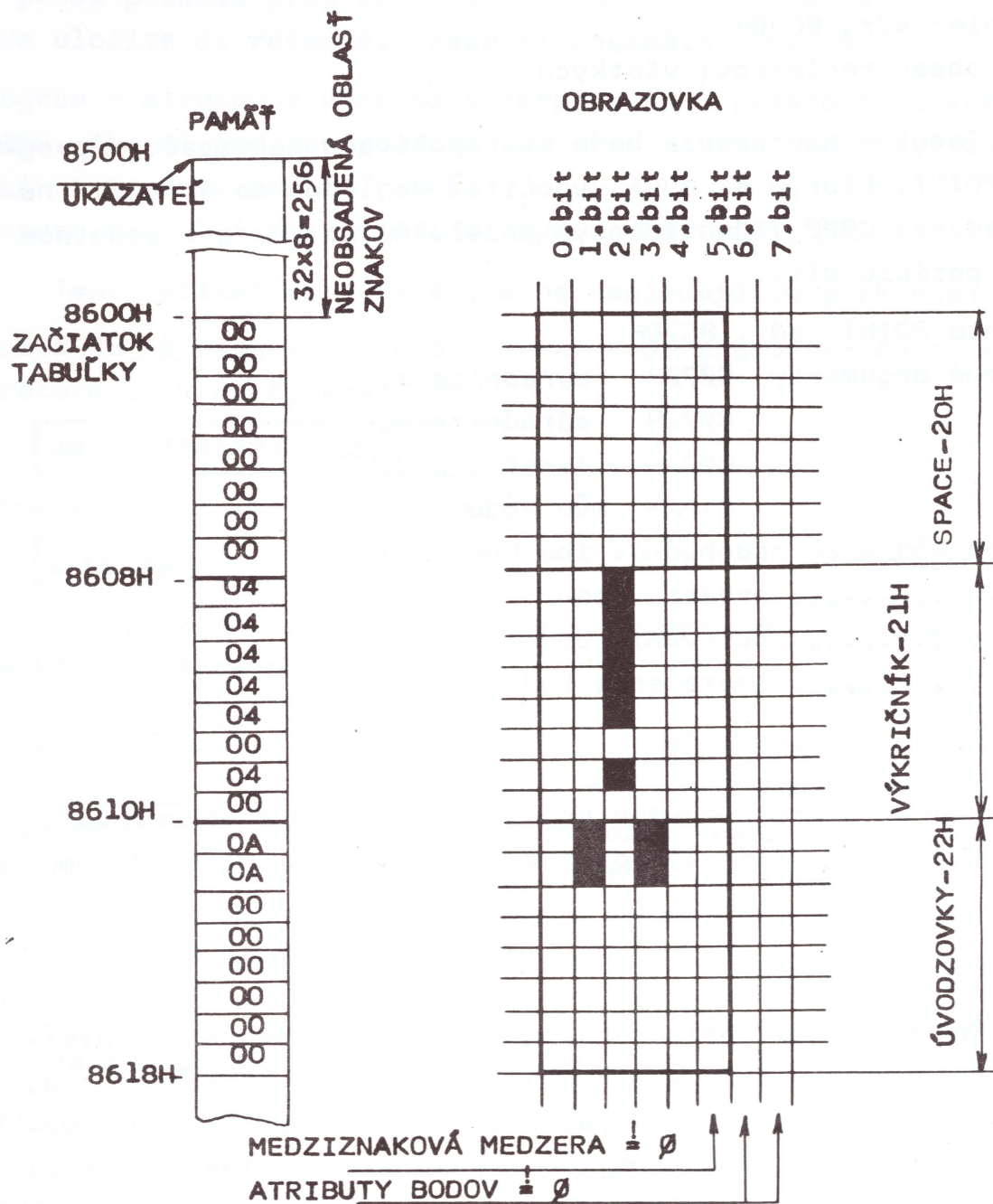
Týmto programovým modulom zabezpečuje mikropočítač vytvorenie lineárneho spojenia (interpolácia) medzi dvoma ľubovoľne zadanými bodmi v pracovnej časti obrazovky. Z dôvodu väčšej rýchlosti si mikropočítač zníži šírku pracovnej časti na 256 bodov, čo predstavuje jeden byte a samozrejme všetky aritmetické operácie s osembitovými číslami.

Interpolátor vyžaduje tieto vstupné argumenty, ktoré má mikropočítač uložené v zápisníku OS:

C170H - hodnota x posledného bodu

C172H - hodnota y posledného bodu





Obr.7 VZŤAH MEDZI ULOŽENÝMI DÁTAMI A ZOBRAZOVANÝMI BODMI

C173H - hodnota x žiadaného bodu

C174H - hodnota y žiadaného bodu

Volanie: adr. 8CD0H

Mení obsah registrov: všetkých

Samotné zobrazenie bodu mikropočítača zabezpečuje program POINT, ktorým si musí vypočítať vzhľadom na strojovú nulu (adresa C000H videostránky) príslušnú adresu a v samotnom byte pozíciu bitu.

Volanie POINT: adr. 8C7DH

Vstupné argumenty: C070H súradnica x,  
C072H súradnica y,  
C03AH farebný kód,  
C1FAH mód kódu,

pričom mód môže nadobudnúť iba hodnoty:

A8H .....negácia bodu

AFH .....nulovanie bodu

B0H .....nastavenie bodu

Mení obsah registra: PSW

Návrat z programového modulu INPOL vykoná mikropočítač pri stotožnení údajov súradníc posledného bodu so žiadaným bodom.

## OSIO

Pre vzájomnú spoluprácu operačného systému s užívateľským programom vyžaduje mikropočítač správnu korešpondenciu. Volanie podprogramu ENTER (adr. 8BEEH) umožní odloženie vrcholu zásobníka, návratovej adresy a nastaví vlastný vrchol zásobníka na hodnotu 7FFFH.

Výstup z podprogramu a pokračovanie v interpretovaní užívateľského programu mikropočítač vykoná pri stlačení klávesu EOL. Na tomto základe mikropočítač presunie obsah dialógového riadka do lokácie zadanej ukazovateľom C078 H a daný reťazec ukončí znakom CR (0DH). Potom uloží celý riadok do oblasti vyhradenej programovým kľúčom. Späť si ho môžete privolať klávesom RCL. Mikropočítač potom nastaví vrchol zásobníka, ktorý bol pred vstupom do programu, a pokračuje v in-



terpretovaní prerušeného programu.

Zhrnutie:

V doteraz uvedenom texte sme prebrali prvú časť informácií o mikropočítači PMD - 85. Druhá časť bude zameraná na úplné inú problematiku, a to na komunikovanie mikropočítača s užívateľom na vyššej úrovni. Aby ste mohli celkovo zhodnotiť, ako ste pochopili dosiaľ uvedené informácie o funkcii monitора, urobte nasledujúcu úlohu.

Úloha č. 1

- a/ do videostránky vložte údaje svojho mena,
- b/ skontrolujte tento obsah príkazom MEM,
- c/ skontrolujte tento obsah príkazom DUMP a porovnajte ho s riešením podľa bodu b/,
- d/ hľadajte odlišnosti a zvažte ich dôvod,
- e/ zostavte a vykonajte program na naplnenie videostránky jednotkami.

#### G. PROGRAMOVANIE V JAZYKU BASIC - G

V tejto časti textu vás oboznámime s programovacím jazykom BASIC - G, ktorý je špeciálne orientovaný na grafiku. Zároveň upozorňujeme, že nie je v našich silách popísať všetky možné kombinácie príkazov. Od metodiky programovania a šikovnosti programátora závisí, ako bude komponovať príkazy v programe.

Pretože v tomto programovacom jazyku sa často píše (dlhé texty), je vynechané pri niektorých príkladoch vkladanie dlhých textov s tým, že bude zobrazený iba text v dialógovom riadku.

Teraz vyvolajte vyšší programovací jazyk mikropočítača. Urobíte to príkazom

|   |   |   |   |   |       |   |     |
|---|---|---|---|---|-------|---|-----|
| B | A | S | I | C | SPACE | G | EOL |
|---|---|---|---|---|-------|---|-----|

Bezprostredne potom mikropočítač podá správu:

BASIC-G/v1.0

a v dialógovom riadku sa objaví " OK ". Od tohto momentu môžete s mikropočítačom komunikovať na vyššej úrovni.

To, že sa nachádza v tomto programovom module, bude vidieť aj zo znaku ">", ktorý sa bude zobrazovať vľavo v dialógovom riadku.

Programovací jazyk BASIC - G, ktorý pracuje interpretačným spôsobom, možno rozdeliť na dve časti. Sú to tieto časti:

- štandardná časť (numerická)
- špeciálna časť (grafická).

Každá časť programovacieho jazyka BASIC - G sa dá interpretovať v tzv. priamom režime a v programovom móde. Najprv sa budeme venovať jednoduchšiemu, preto začneme programovaním v priamom režime pre štandardné (numerické) príkazy.

#### G. 1. Priamy režim

charakterizuje to, že po vložení povelu BASIC do dialógového riadka je tento povel bezprostredne mikropočítačom vykonaný. Mikropočítač v tomto režime pracuje tak, ako kalkulačka. Aby mohol mikropočítač vypísať výslednú hodnotu alebo hodnotu niektorej premennej, prípadne reťazca znakov, musíte vložiť povel:

alebo

Rozdiel medzi týmito povelmi je len v mieste výpisu. V prípade PRINT bude mikropočítač výstup zobrazovať do pracovnej časti obrazovky, v druhom prípade do dialógového riadka, ale jeho obsah treba potvrdiť stlačením ľubovoľného klávesu.

Pre prácu BASIC má mikropočítač vyhradené niektoré klávesy na špeciálne funkcie.

Sú to tieto klávesy:

označenie reťazcovej premennej

označenie na začiatok a na koniec reťazca znakov ASCII

desatinná bodka

čiarka ako oddeľovač

bodkočiarka na formát výpisu

skrátенý povel na výpis hodnoty do dialógového riadka



|           |                                   |
|-----------|-----------------------------------|
| ( )       | zátvorky pre aritmetické výrazy   |
| > = <     | relačné operátory                 |
| + - * / ^ | aritmetické operátory             |
| :         | dvojbodka pre viacnásobné príkazy |
| SPACE     | medzera                           |
| STOP      | zastavenie programu               |

Pred uvedením prvého príkladu sa musíte oboznámiť s aritmetickým aparátom mikropočítača. Popíšeme ho v nasledujúcich častiach.

### G. 1.1 Číselný formát

Všetky čísla, ktorých absolútna hodnota je v rozsahu 0,01 až 999999 sú zobrazované v pevnej desatinnej čiarke bez exponentu. Všetky ostatné okrem 0 majú formát

SM,MMMMMESXX,

|       |   |
|-------|---|
| kde S | predstavuje znamienko + alebo -, pričom sa + nezobrazuje, ale počíta sa s jeho miestom, |
| M     | mantisa čísla,  |
| X     | exponent čísla,   |
| E     | identifikátor exponentu.  |

Matematický aparát pracuje v číselnom rozsahu:

$1 \times 10^{-38}$  až  $1 \times 10^{+38}$

### G. 1.2 Aritmetické premenné

Aritmetické premenné môžete označiť až dvoma znakmi, pričom prvý znak musí byť písmeno a druhý písmeno alebo číslica.

Napríklad: X, A, A1, ER atď.

Aritmetické premenné slúžia na uchovávanie hodnoty aritmetického, logického alebo relačného výrazu.

### G. 1.3 Operátory aritmetického výrazu

| SYMBOL      | VÝRAZ        | VÝZNAM                     |
|-------------|--------------|----------------------------|
| +           | $A + B$      | sčítanie                   |
| -           | $A - B$      | odčítanie                  |
| $\wedge$    | $A \wedge B$ | umocnenie čísla A číslom B |
| *           | $A * B$      | násobenie                  |
| /           | $A / B$      | delenie                    |
| znamienko + | $+ A$        | kladné číslo A             |
| znamienko - | $- A$        | záporné číslo A            |

Jednotlivé operátory môžete v aritmetickom výraze použiť viackrát, prípadne ich môžete kombinovať,

napríklad:  $20 - ((A + B) + C) \wedge D$   
 $AA + VG + (-X/32)$  a pod.

### G. 1.4 Operátory v logických výrazoch

| SYMBOL | VÝRAZ              | VÝZNAM                |
|--------|--------------------|-----------------------|
| OR     | $A \text{ OR } B$  | logický súčet         |
| AND    | $A \text{ AND } B$ | logický súčin         |
| NOT    | $\text{NOT } A$    | dvojkový komplement A |

Logické operátory používame vtedy, keď chceme pri rozhodovaní podmieňovať výsledok operácie splnením daných vstupných podmienok.

### G. 1.5 Operátory v relačných výrazoch

| SYMBOL | VÝRAZ      | VÝZNAM                        |
|--------|------------|-------------------------------|
| =      | $A = B$    | A je rovné B                  |
| <      | $A < B$    | A je menšie ako B             |
| >      | $A > B$    | A je väčšie ako B             |
| <=     | $A \leq B$ | A je rovné alebo menšie ako B |
| >=     | $A \geq B$ | A je rovné alebo väčšie ako B |
| <>     | $A \neq B$ | A sa nerovná B                |

Pri logických a relačných výrazoch si treba uvedomiť, že výsledná hodnota môže nadobúdať iba dve hodnoty, a to: pre pravdivý výraz hodnotu "1", pre nepravdivý výraz hodnotu "0".



Musíme poznamenať, že operátor "NOT" je výnimkou, pretože robí dvojkový komplement čísla. Mikropočítač má v matematickom aparáte širokú škálu definovaných matematických funkcií, ktoré vám určite uľahčia prácu. Teraz si ich vysvetlíme.

#### G, 1.6 Matematické funkcie

|         |   |
|---------|---|
| SIN (X) | sínus čísla X zadaného v radiánoch                          |
| COS (X) | kosínus čísla X zadaného v radiánoch                        |
| TAN (X) | tangens čísla X zadaného v radiánoch                        |
| ATN (X) | arctangens čísla X, pričom $-\pi/2 < \text{ATN}(X) < \pi/2$ |
| LOG (X) | prirodzený logaritmus čísla väčšieho ako nula               |
| EXP (X) | X-tá mocnina základov prirodzených logaritmov e             |
| SQR (X) | druhá odmocnina čísla X väčšieho ako nula                   |
| ABS (X) | absolútna hodnota čísla X                                   |
| INT (X) | celočíselná hodnota čísla X                                 |
| RND (X) | generovanie náhodného čísla z oblasti $0 \div 1$            |
| SGN (X) | algebraická hodnota čísla X                                 |

Je samozrejmé, že namiesto čísla X môžete uvádzať aj aritmetické výrazy. Kým sa začneme zaoberať príkladmi výpočtov kalkulačkového typu, vysvetlíme si ešte, ako budú mikropočítačom spracúvané zadané aritmetické výrazy.

Matematická hierarchia je táto:  
exponenciálne výrazy  
znamienka mínus alebo plus  
násobenie alebo delenie  
sčítanie alebo odčítanie  
relačné operátory  
logické operátory NOT, AND a nakoniec OR.

Ak majú dva operátory rovnakú úroveň spracovania, postupuje počítač smerom zľava doprava.

Napríklad výraz

$$20 - A + B \times C^{\wedge}D$$

bude mikropočítačom spracúvaný týmto spôsobom:

1. vypočíta  $20 - A$ ,
2. vypočíta  $C^{\wedge}D$ ,

3. potom B vynásobí hodnotou získanou v operačnom kroku 2,
4. hodnotu operačného kroku 1 pripočíta k hodnote získanej v operačnom kroku 3.

Ak ste si neistý v tom, ako bude mikropočítačom spracúvaný výraz, odporúčame používať zátvorky.

Teraz pristúpime k dialógu. Ak vám nie je ešte všetko jasné, bude lepšie, ak po krátkej prestávke začnete znova od kapitoly G. a zároveň si budete robiť krátke poznámky.

#### G.1.7 Kalkulátor

Kalkulačkový spôsob práce pre mikropočítač spočíva v zadaní výpočtu výrazu, ktorý môže byť aritmetický, logický alebo relačný. Ak správne vložíte výraz na výpočet, objavíte výsledok mikropočítača podľa želania alebo v pracovnej časti obrazovky, alebo v dialógovom riadku. Keďže sa však mikropočítač nemôže spoliehať iba na ľudskú dôslednosť, musí užívateľa kontrolovať popri svojej práci so zadaním. Vzniknuté chyby môžu byť syntaktického alebo sémantického charakteru. V každom prípade však musí mikropočítač na ne reagovať tým, že vydá príslušnú správu do dialógového riadka. Zoznam všetkých správ, ktoré môže odoslať, je zahrnutý v zvláštnej časti na konci výkladu. Je to preto, aby ste ich mali vedľa seba a tým zabránili možnému omylu.

V závere tejto časti predkladáme skôr uvádzané príklady na precvičenie.

#### Príklad č. 9

Vypíšte názov PMD-85

CLR P R I N T " P M D - 8 5 "

stlačením klávesu EOL sa zobrazí v pracovnej časti

PMD-85

pracovná časť obrazovky

OK

dialógový riadok



### Príklad č. 10

Zobraz číslo 2,000,000 v dialógovom riadku

[?] [2] [0] [0] [0] [0] [0] [0] [EOL]

[2] E + 06

[SPACE]

[OK]

Ak žiadate priradiť premenné určitou hodnotou, môžete to urobiť takto:

[A] [=] [2] [E] [1] [0] [EOL]

[OK]

Teraz sa môžete presvedčiť, či si mikropočítač túto hodnotu zapamätal. Môžete to urobiť tak, ako sme už skôr uviedli, príkazmi PRINT alebo ?

[?] [A] [EOL]

[2E + 10]

Iste vám prišli na um rozličné možnosti využitia tohto priameho režimu práce. Ak napríklad chcete poznať číselnú hodnotu aritmetického výrazu, ktorý často používate pri svojej práci, môžete si tento výraz definovať premennými. Potom stačí, keď budete vkladať hodnoty týchto premenných. V niekoľkých ďalších príkladoch bude vykonaná práca v priamom (kalkulačnom) režime. Pozorne sledujte jednotlivé úkony, lebo ich mikropočítač nebude komentovať. Výsledky bude vypisovať do pracovnej časti, teda s použitím príkazu PRINT.

### Príklad č. 11

Výpočet SIN (10) (v rad)

[P] [R] [I] [N] [T] [S] [I] [N] [(] [1] [0] [)] [EOL]

[ - . 545358

[OK]

### Príklad č. 12

Výpočet priemeru dvoch čísel. Vzorec:  $\frac{A + B}{2}$

[(] [A] [+] [B] [)] [/] [2] [WRK] [K0]

[A] [=] [1] [0] [EOL]

[B] [=] [S] [I] [N] [(] [1] [0] [)] [EOL]

P R I N T K Ø EOL

4. 72732

OK

Ak máte záujem o výpis viacerých hodnôt pod príkazom PRINT, môžete ich oddeliť čiarkou alebo dvojbodkou.

Vtedy napr.

P R I N T K Ø , A ; B EOL

sa zobrazí nasledujúcim spôsobom:

4. 72732

voľných 6 pozícií

10 -.545358

voľná jedna pozícia

OK

Príklad č. 13

Výpočet vzťahu

$$\frac{\sqrt{B^2 - 4AC}}{2}$$

( S Q R ( B ^ 2 - 4 \* A \* C ) ) / 2 WRK K1

Vloženie premenných:

B = 9 EOL

A = 2 : C = 1 EOL

Výpočet vzťahu:

P R I N T K1 EOL

4.272

OK

Ak použijete ďalší "key", na ktorý uložíte aj príslušný vzťah, môžete mať výsledok v žiadanej forme, na akú ste zvyčajne zvyknutí.

P R I N T " K1 = " ; K1 EOL

Zobrazenie:

SQR ( B ^ 2 - 4 \* A \* C )

OK



#### Príklad č. 14

Demonštrácia sémantickej chyby pri delení

P R I N T 1 Ø / Ø EOL      \* \* \* DV BY ZERO \* \* \*

Zámerne sme uviedli tento príklad, v ktorom mikropočítač demonštruje vzniknutú chybu a spôsob oznámenia. Ide konkrétne o delenie nulou. Túto správu mikropočítač podáva do dialógového riadka aj s krátkym akustickým tónom.

Ak vás tieto príklady presvedčili, že už ovládáte postup, pokúste sa samostatne zadávať rozličné výpočty a realizovať ich, aby ste získali potrebnú rutinu a odhalili matematické možnosti mikropočítača.

Teraz vysvetlíme, ako môžete programovať celé výpočty. Pôjde o druhý programový režim mikropočítača.

#### G. 2 Programový režim

Ľahko zistíte, že priamym režimom sa ťažko realizujú zložitejšie, viacnásobné príkazy. Je to preto, lebo musíte vykonať niekoľko príkazov za sebou, aby ste mohli použiť žiadaný vzťah. Máte možnosť zaradiť jednotlivé myšlienky za sebou. Takému usporiadaniu hovoríme program. Na program má mikropočítač vyhradené pamäťové miesto. Ak chcete vedieť, akú má práve teraz voľnú kapacitu, odošlite nasledujúci príkaz:

? F R E ( A ) EOL

V dialógovom riadku bude uvedená jej veľkosť v byte.

Aby bola možnosť evidovať jednotlivé príkazy, budeme definovať jednotlivé pojmy.

Najmenšou programovou jednotkou je jeden programový riadok. Na jeho evidenciu v programe je potrebné, aby ste mu priradili číslo. To znamená, ak napíšete do dialógového riadka obsah, ktorý sa začína číslom, mikropočítač ho zaeviduje do svojej pamäti ako programový riadok a na vytvorenie programu, jeho editovanie, prípadne spustenie má rezervované tieto príkazy:





Všimnite si programové riadky 30 a 40. V riadku 30 je uvedený príkaz DISP, ktorý má podobnú funkciu ako príkaz PRINT, ale ho mikropočítač vypisuje do dialógového riadka. Jeho úlohou je v našom prípade upozorniť na to, že máte vložiť vstupnú hodnotu, lebo príkaz INPUT nemôže prijať návestie. Teraz sa pokúste prečítať program, ktorý ste napísali, príkazom LIST.

CLR L I S T EOL

V pracovnej časti sa zobrazí takto:

```
10      PRINT "PLOCHA"
20      PI = 3.14159
30      DISP "POLOMER="
40      INPUT R
50      F = PI * R * R
60      PRINT R, F
70      END
```

Teraz si precvičte editovanie programového riadka číslo 10. Urobíte to príkazom LLIST

L L I S T EOL      10 PRINT "PLOCHA"  
END SPACE SPACE K R U H U " EOL      OK

Spätná kontrola:

L L I S T EOL      10 PRINT "PLOCHA KRUHU"

Ak je Vám jasný spôsob vkladania a editovania programových riadkov, vyskúšajte si ešte príkaz na spustenie programu.

R U N EOL      OK

Bezprostredne po tomto príkaze začne mikropočítač interpretovať program. Najprv vypisuje text

P L O C H A

a v dialógovom riadku sa objaví žiadosť o zadanie vstupného argumentu

POLOMER =

Teraz bude mikropočítač čakať, kým neuložíte prvý znak. Uložte číslicu 1 a riadok ukončíte klávesom EOL. Ak ste vložili iný znak ako číslo, bude mikropočítač stále čakať a vracať sa,

kým mu nevložíte číselný znak. Keď chcete opustiť tento stav, stačí súčasne stlačiť klávesy **STOP** a **EOL**. Ak ste vložili hodnotu 1, umožnili ste mu pokračovať v interpretovaní programu v riadku 50, kde má urobiť výpočet premennej F. V riadku 60 žiadate vypísať tento výsledok do pracovnej časti obrazovky i s vloženým polomerom. Z mikropočítača dostávate výstup, ktorý má takýto tvar:

|    |          |
|----|----------|
| 1  | 3. 14159 |
| OK |          |

Odštartovať program môžete aj od čísla riadka, ktorý ste zadali. Vtedy príkaz RUN dostane argument s celočíselnou hodnotou udávajúcou číslo existujúceho riadka. Ak chcete program testovať v určitých vami zadanych bodoch, môžete použiť príkazový riadok s obsahom:

|   |   |   |   |
|---|---|---|---|
| S | T | O | P |
|---|---|---|---|

Vtedy vám mikropočítač vypíše do dialógového riadka, že sa zastavil na danom čísle riadka a očakáva príkazy. V tomto bode môžete používať spôsob priameho režimu, ktorý sme si už vysvetlili, môžete sa spytovať na obsahy premenných, prípadne ich môžete meniť. Ak chcete pokračovať v prerušenom programe, máte zakázané meniť rozsah programu; teda nemôžete upravovať program.

Pokračovanie programu vykonáte príkazom:

|   |   |   |   |     |
|---|---|---|---|-----|
| C | O | N | T | EOL |
|---|---|---|---|-----|

alebo môžete priamo napísať skok na žiadaný riadok. Napr:

|   |   |   |   |   |   |     |
|---|---|---|---|---|---|-----|
| G | O | T | O | 5 | 0 | EOL |
|---|---|---|---|---|---|-----|

Činnosť interpretovania programových riadkov mikropočítača môžete zastaviť aj druhým, veľmi častým spôsobom: stlačením klávesu **STOP**. Toto prerušenie nastane len pri prechode na nový riadok, takže sa môže stať, že musíte počkať, ak mikropočítač dlhšie pracuje v danom riadku.

Tieto nové príkazy vykonajte na príklade č. 14 viackrát, aby ste si ich dôkladne osvojili.



Na úplnosť príkazov týkajúcich sa zostavovania programu ešte treba uviesť príkazy na prácu s magnetofónom. Práca s magnetofónom je založená na tom istom princípe, ktorý sme si vysvetlili v kapitole D.1., len treba poznať iné názvy príkazov. Tu sú uvedené:

| PRÍKAZ                 | VÝZNAM  |
|------------------------|---|
| SAVE n<br>("komentár") | Uloženie programu začínajúceho sa prvým riadkom na pásku magnetofónu. Záznam bude mať poradové číslo "n". Komentár nie je povinný, ale ak ho napíšete, musí sa začínať úvodzovkami.         |
| LOAD n                 | Vyhľadanie a uloženie záznamu s poradovým číslom "n" do programovej pamäti BASICu.  |
| CHECK n                | Kontrola záznamu z pásky magnetofónu s poradovým číslom "n". Tento príkaz je vhodný aj na vyhľadanie konca záznamu pre možnosť ďalšieho zápisu, pretože sa nemení obsah programovej pamäti. |

Doteraz by ste podľa prebratého výkladu mali ovládať príkazy na prácu v priamom i programovom režime. Mali by ste hravo zvládnuť napísanie programu, jeho editovanie, prípadne uloženie na kazetovú pásku. Viete už používať príkaz na odštartovanie a ladenie programu. Ešte si musíte osvojiť príkazy používané v samotnom programe. Tieto príkazy si iba zhrnieme do tabuľky. V tabuľke uvádzame iba štandardné príkazy, teda bez príkazov pre grafiku. S nimi sa oboznámime v samostatnej časti.

| PRÍKAZ             | VÝZNAM  |
|--------------------|---|
| BEEP               | Akustické návestie  |
| DATA               | Označenie dát alebo reťazcových znakov na ich čítanie príkazom READ |
| DEF FNC            | Definovanie funkcií užívateľom                                      |
| DIM                | Deklarácia reťazcových a dátových polí                              |
| FOR-TO-NEXT (STEP) | Tvorenie programových slučiek                                       |
| DISP               | Zobrazenie výrazu do dialógového riadka                             |

|              |   |
|--------------|---|
| GCLEAR       | Vymazanie obrazovej časti                                       |
| GOSUB-RETURN | Príkazy na volanie a skončenie programového podprogramu         |
| GOTO         | Prechod na vykonávanie určeného programového riadka             |
| IF-THEN      | Príkaz na podmienené vykonávanie príkazu THEN                   |
| INPUT        | Zadanie vstupných údajov z klávesnice                           |
| MONIT        | Návrat do operačného systému                                    |
| LET          | Priradovací príkaz  |
| PAUSE        | Časové zdržanie vo vykonávacom programe                         |
| PRINT        | Zobrazovanie výrazov do pracovnej časti obrazovky               |
| POKE         | Zápis do pamäti   |
| READ, DATA   | Čítanie údajov do premenných z programu                         |
| RESTORE      | Nastavenie ukazovateľa údajového bloku na čítanie príkazov READ |
| STOP         | Zastavenie interpretácie programu                               |
| REM          | Vloženie komentárov do programu                                 |
| ON           | Príkaz programovému prepínaču volania podprogramov alebo skokov |
| OUT          | Zápis na port systému 8080                                      |

Doposiaľ vám nebolo vysvetlené, aké má mikropočítač možnosti na uskladnenie premenných, prípadne reťazcov. Všeobecne sa tieto skupiny dát nazývajú "pole". S jednotlivými prvkami poľa sa dá pracovať ako s premennými. Všeobecné pole sa označuje dvoma znakmi. Prvý znak je písmeno, druhý znak môže byť písmeno alebo číslo.

Napríklad:

- A1 (15)            znamená 15-ty prvok z jednorozmerného číselného poľa A1,
- AA (10, 10)        znamená prvok z dvojrozmernej matice číselného poľa AA; pozícia prvku je v súradniciach 10, 10.
- AB (10, 10, 10)    znamená prvok z trojrozmerného číselného poľa AB; pozícia prvku je v súradniciach 10, 10, 10.



Zapamätajte si, že pre väčší počet prvkov ako 10 je potrebné, aby ste deklarovali dané pole (aby ste určili dimenziu),

Ak chcete vytvoriť pole, ktoré bude obsahovať reťazce, prijíma značka poľa ešte znak \$.

Reťazce sú znaky ASCII, ktoré sú vpísané do dvoch úvodzoviek. Ich dĺžka môže byť maximálne 255 znakov. Reťazcové premenné môžu byť označované jednoznakovo (napr. A\$) alebo dvojznakovo (Al\$.)

POZOR! Nezabudnite rozlišovať druh premenných:

A, A 1 a A\$

Reťazcové premenné môžu byť deklarované aj v poliach, pričom max. dimenzia je 3.

Pri práci s reťazcami môžete používať tieto funkcie:

| FUNKCIA           | VÝZNAM   |
|-------------------|--|
| ASC (X\$)         | Dekadická hodnota prvého znaku reťazca X\$               |
| CHR\$ (I)         | Prevod dekadickej hodnoty I (0 - 255) na hodnotu ASCII   |
| FRE (X\$)         | Zistenie voľného miesta na uloženie obsahu reťazcov      |
| LEFT\$ (X\$, I)   | Načítanie zľava, I počet znakov z X\$                    |
| RIGHT\$ (X\$, I)  | Načítanie sprava, I počet znakov z X\$                   |
| MID\$ (X\$, I, K) | Načítanie časti reťazca od znaku s pozíciou I a dĺžkou K |
| LEN (X\$)         | Určenie dĺžky reťazca X\$                                |
| STR\$ (X)         | Premena čísla X na reťazec                               |
| VAL (X\$)         | Premena reťazca na číslo                                 |

Pre úplnosť uvádzame aj ostatné zvyšné funkcie, ktoré majú špeciálny význam. Sú to tieto funkcie:

| FUNKCIA | VÝZNAM   |
|---------|--|
| FRE (X) | Voľná pamäťová oblasť na uloženie programu alebo dát |
| SPC (I) | Vkladanie I medzier na výstupné zariadenie           |
| TAB (I) | Tabulátor  |

|             |   |
|-------------|---|
| BIT A, B    | Určenie hodnoty bitu pri A na váhe B  |
| STATUS A, B | Načítanie hodnoty z I/O kanála A a registra B   |
| INKEY       | Načítanie hodnoty KEY klávesu (K0-K10)  |
| PEEK (I)    | Načítanie hodnoty byte z pamäťového miesta I  |
| USR (I)     | Volanie užívateľského podprogramu na adrese I; odovzdávajúci argument sa nachádza v akumulátore |
| INP (I)     | Načítanie hodnoty z portu I   |

Ak vám nie sú jasné niektoré príkazy alebo funkcie, s ktorými ste sa doposiaľ oboznámili, máte možnosť precvičiť si ich na nasledujúcich príkladoch. Jednotlivé príklady sú spracované ako výpisy programov, ktoré musíte vložiť prostredníctvom klávesu do pamäti. Ladenie a štartovanie programov vykonajte podľa predchádzajúcich pokynov.

#### Príklad č. 15

```
10 REM KVADRATICKA ROVNICA  $AX^2 + BX + C = 0$ 
20 DISP "ZADAJ KONŠTANTY A, B, C"
30 INPUT A, B, C
40 D =  $B^2 - 4 * A * C$ 
50 IF D < 0 GOTO 100
60 X1 =  $(-B + \text{SQR}(D)) / 2 / A$ 
70 X2 =  $(-B - \text{SQR}(D)) / 2 / A$ 
80 PRINT "REALNE KORENE : X1 = "; X1;"X2 = "; X2
90 END
100 R =  $B / 2 / A$  : I =  $\text{SQR}(-D) / 2 / A$ 
110 PRINT "IMAG. KOR. X1 = "; R; " + I"; I;"X2 = "; R;
    " - I " ; I
```

Pri vykonávaní príkazu na riadku 30 môžete urobiť chybu, na miesto číselnej hodnoty vložíte nečíselný znak. Hodnoty môžete vložiť naraz s oddelením čiarkou alebo postupne po jednej.

Úloha č. 2 Vložte programový riadok na zopakovanie výpočtu



Príklad č. 16

Komentár

```
10 REM TABULKA FUNKCIA
20 T = 8 : DEF FNC A (X) = EXP (X/T)      Definovanie funkcie
30 FOR X = 0 TO 100 STEP 2
40 PRINT "X = "; X; "A(X)="; FNC A(X)
50 NEXT X
```

Poznámka: Všimnite si možnosti písania viacerých príkazov na jeden programový riadok. Oddeľuje ich dvojbodka.

Úloha č. 3 Zovšeobecniť úlohu pre definičný obor zadany z klávesnice.

Príklad č. 17

```
10 REM SPORTKA
20 FOR I = 1 TO 6
30 PRINT INT (48 * RND (1) + 1);
40 NEXT I
```

Úloha č. 4 Ochráňte program tak, aby nedošlo ku generovaniu rovnakých čísel.

Príklad č. 18

Komentár

|                                    |                    |
|------------------------------------|--------------------|
| 10 REM PREVOD KCAL - JOUL          |                    |
| 20 DISP "ZADAJ HODNOTU KCAL"       | hlásenie           |
| 30 INPUT K                         | vstup z klávesnice |
| 40 J = K * 4210                    | výpočet            |
| 50 PRINT K; "KCAL JE"; J; "JOULOV" | hlásenie           |

Úloha č. 5 Zmeňte program na prevod JOUL na KCAL

Príklad č. 19

Komentár

|   |                        |
|---|------------------------|
| 10 REM : SUCET 10 CISEL ZADANYCH AKO DATA |                        |
| 20 FOR I = 1 TO 10                        | začiatok cyklu         |
| 30 READ N                                 | do N sa ukladajú údaje |
| 40 S = S + N                              | súčet dát              |
| 50 NEXT I                                 | koniec cyklu           |
| 60 PRINT "SUCET JE"; S                    | hlásenie               |
| 70 DATA 5, 4, 2, 3, 6, 7, 8, 4, 1, 3      | dáta                   |

- Úloha č. 6 a) Doplňte jeden programový riadok tak, aby sa na obrazovke objavili aj zadané dáta.  
b) Zmeňte program pre súčet sčítancov (10) zadaný z klávesnice.

Príklad č. 20

Komentár

```
10 REM DEFINOVANIE a TLAČENIE
   MATICE A (I,J) = I * J
20 DISP "ZADAJ RAD MATICE"
30 INPUT N
40 DIM A (N,N)                deklarácia dvojrozmerného
                                poľa
50 FOR I = 1 TO N
60 FOR J = 1 TO N
70 A (I,J) = I * J            definovanie prvku A(I,J)
80 PRINT A (I,J)
90 NEXT J
100 PRINT
110 NEXT I
```

Všimnite si správnosť tvorenia programových slučiek I, J.

- Úloha č. 7 Napíšte program na definovanie matíc (A), (B) a výpočet súčtovej matice (C) = (A) + (B)

Príklad č. 21

```
10 REM VYPOCET DVOCH PARALELNE ZAPOJENYCH ODPOROV
20 DISP "ZADAJ HODNOTY R1, R2, " " " :INPUT R1,R2
30 R = 1/R1 + 1/R2 : R = 1/R
40 PRINT R1;"PARALELNE S" ; R2; "JE" ; R
50 BEEP
60 DISP "CHCES NOVE HODNOTY? [ANO/NIE]"
70 INPUT A$
80 IF A$ = "ANO" THEN 10
90 END
```

Všimnite si riadok 20, v ktorom je príkaz DISP zakončený prázdny reťazcom, lebo nasledujúci DISP v riadku 60 je dlhší a spôsobil by zvyšok v zobrazení, Môžete sa o tom presvedčiť,



Úloha č. 8 Doplňte program o možnosť voľby z klávesnice na výpočet sériovej alebo paralelnej kombinácie odporov.

Príklad č. 22

```
10 REM TABULKA
20 PRINT "CISLO"; TAB (5); "NAZOV"; TAB (25); "CENA"; TAB (33);
   "POCET"
30 DISP "ZADAJ POCET POLOZIEK:", " "
40 INPUT N
50 FOR I=1 TO N
60 DISP "NAZOV, CENA, POCET", " "
70 INPUT A$, C, P
80 PRINT I; TAB (5); A$; TAB (25); C; TAB (33); P
90 NEXT I
```

Úloha č. 9 Rozšírte program o možnosť vyhľadávania položky, ktorej názov je zadany cez klávesnicu.

Príklad č. 23 Demonštrácia podmienovacích príkazov.

A sa mení od 0 do 100 opakovane, B sa náhodne generuje od 0 do 1. Zistite prípad, keď A = 22 alebo 33 alebo 44 a B je zároveň v rozsahu

$0.3 < B < 0.5$ .

10 B = RND (1)

20 IF (A = 22 OR A = 33 OR A = 44) THEN GOSUB 100

30 A = A + 1

40 IF A = 100 THEN LET A = 0

50 GOTO 10

60 END

100 IF (B > .3 AND B < .5) THEN PRINT A,B: BEEP

110 RETURN

Úloha č. 10 Zjednodušte (skompresujte) daný program použitím viacnásobných príkazov na riadku.

Príklad č. 24 Demonštrácia príkazov ON - INKEY - GOSUB

10 PAUSE 10

20 ON INKEY GOSUB 100, 200

30 GOSUB 300

```
40 GOTO 10
100 A = 1 : B = 0
110 PRINT "A = "; A, "B = "; B
120 RETURN
200 A = 0 : B = 1
210 PRINT "A = "; A, "B = "; B
220 RETURN
300 C = C + A : D = D + B
310 ? "C="; C; "D="; D; " "
320 RETURN
```

V tomto príklade je použitý v riadku 310 skrátený príkaz "?", ktorý umožňuje zastavenie programu do stlačenia ľubovoľného klávesu.

Úloha č. 11 Doplňte program o podmienený skok, ktorý skončí program, ak je C alebo D 1000.

Príklad č. 25 Demonštrácia príkazov s reťazcovými premennými

```
10 A$ = "PMD-85" : B$ = " "
20 FOR I = 1 TO LEN (A$)
30 B$ = LEFT$ (A$, I )
40 PRINT B$
50 NEXT I
```

Po vykonaní programu sa zobrazí na obrazovke :

|          |
|----------|
| P        |
| PM       |
| PMD      |
| PMD -    |
| PMD - 8  |
| PMD - 85 |

|    |
|----|
| OK |
|----|

Úloha č. 12 Zmeňte program tak, aby sa na obrazovke postupne objavovalo :

5

58



58 -  
58 - D  
58 - DM  
58 - DMP

Príklad č. 25/•

```
10 REM VETA
20 A$ = "MA": B$ = "TA"
30 A$ = A$ + A$: B$ = B$ + B$: D$ = RIGHT$(A$,1)
40 PRINT A$, D$, B$
```

Úloha č. 13 Vytvorte z reťazcov A\$, B\$ ďalší reťazec, v ktorom bude slovo "MAT".

A na záver ešte jeden príklad, ktorý je trochu dlhší.

Príklad č. 26

```
10 REM PREVOD : FORINT - KORUNA, RUBEL - KORUNA
20 F$ = "FT" : R$ = "RB"
30 PRINT "ZA SUMOU UVEDTE SKRATKU FT (FORINT) ALEBO RB (RUBEL)"
40 DISP "ZADAJ SUMU A SKRATKU MENY"
50 INPUT A$
60 M$ = RIGHT$(A$, 2) : P = LEN(A$) - 2 : B$ = LEFT$(A$,P)
70 S = VAL(B$)
80 IF M$ = F$ THEN LET S = S * .56
90 IF M$ = R$ THEN LET S = S * 10
100 IF S = 0 THEN PRINT "ZADAJ ZNOVA" : GOTO 40
110 PRINT A$ ; "JE"; S ; "KCS"
```

Úloha č. 14 Rozšírte program o ďalšie meny.

S oboznámením doteraz uvedených informácií sme skončili jednu z väčších častí práce s interpretérom BASIC-G. Pracovali ste výlučne iba s alfanumerickými znakmi. Aj v tomto režime by ste mohli graficky (semigraficky) spracúvať výpočty prostredníctvom príkazu PRINT a zmenou výstupnej tabuľky na výpis znaku ASCII. Táto práca je však nepohodlná a nevyhovujúca.. Pohodlnejšie a účelnejšie využitie nájdete v nasledujúcej časti popisu, v ktorej bude vysvetlená práca s grafickými povelmi. Interpretovanie týchto príkazov je pre mikropočítač

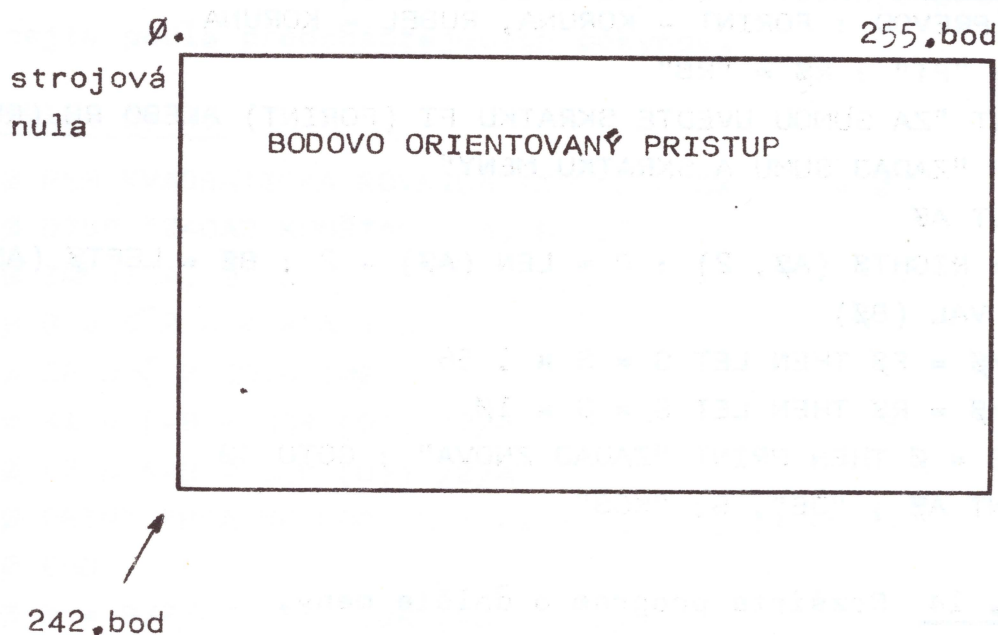
najlepšie, lebo pracuje naplno, s veľkým efektom a hlavne pre vašu spokojnosť.

### G. 3.. Príkazy pre grafiku

Všetky príkazy pre grafiku môžete používať v priamom i v nepriamom režime. Nie je možné umiestňovať ich viac na jeden programový riadok. Okrem toho je zakázané použiť premenné  $X0$ ,  $X1$ ,  $X2$ ,  $Y0$ ,  $Y1$ ,  $Y2$ ,  $Y3$ , preto, lebo sú potrebné pre výpočet transformácie bodov v príkaze SCALE.

Zobrazovať motívy a funkcie je možné prakticky v dvoch formách: jednotlivo bod po bode (bit po bite) alebo slovne naraz šesticou bodov (byte po byte).

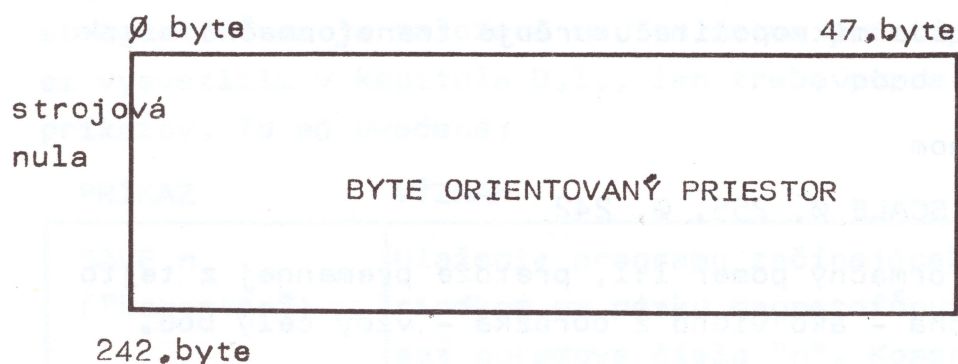
Pri prvom spôsobe si treba zapamätať rozloženie pracovnej časti tak, ako ukazuje nasledujúci obrázok:



Nemusíte mať obavu, že úsečky, spojenia atď. musíte vytvárať vo vlastných programoch. Mikropočítač má veľmi rýchly lineárny interpolátor, ktorý dokáže spojiť ľubovoľné dva body v danej oblasti. Dokonca nemusíte pracovať na úrovni mikropočítača, pretože si môžete dovoliť určiť aj svoju mierku a pracovať implicitne.



Pri druhom spôsobe je rozdelenie obrazovky takéto:



V tomto prípade sa naraz ukladajú celé slová (byte) do pracovného priestoru. Sú na to špeciálne príkazy, ktoré sú odlišné od bodovo (bitovo) orientovaných príkazov.

Každé zobrazenie v základnom stave sa koná ako komplement súčasného stavu na obrazovke. Ak chcete zmeniť spôsob zobrazovania (nastavovania, nulovania), môžete mikropočítač premodifikovať tak, ako je uvedené v časti Modifikácia systému.

Teraz sa zameriame na príkazy pre prácu v bodovo orientovanom priestore. Najprv však zrušíme všetky informácie na obrazovke odoslaním príkazu

G C L E A R EOL

V nasledujúcej tabuľke sú zhrnuté všetky príkazy určené na túto činnosť:

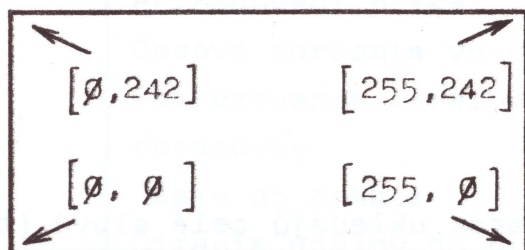
| PRÍKAZ | VÝZNAM                          |
|--------|---------------------------------|
| SCALE  | Nastavenie mierky               |
| MOVE   | Presuň pero do bodu X,Y         |
| PLOT   | Vykresli spojnicu do bodu X, Y  |
| FILL   | Vyplň danú oblasť bodov         |
| LABEL  | Zobrazenie výrazu               |
| AXES   | Zobrazenie osi priesečníka X, Y |

Najdôležitejším začínajúcim príkazom pri grafickom zobrazovaní je SCALE. Tento príkaz slúži na nastavenie mierky v smere x, y, teda mikropočítaču určuje transformačnú mierku na priraďovanie bodov.

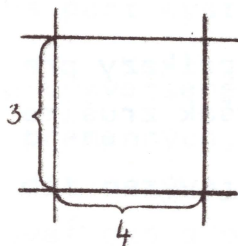
Pri zvolenom

SCALE 0, 255, 0, 242

vychádza transformačný pomer 1:1, pretože premennej z tejto oblasti prislúcha - ako vidno z obrázka - vždy celý bod.



Vzhľadom na televízny rozklad obrazu to nie je štvorec, ale pomer strán je  $x = 4/3 y$ .



Pri kreslení asymetrických obrazcov musíte mať na zreteľli zvolenie správnej mierky (SCALE). Tak si, napríklad, môžete zvoliť mierku:

SCALE -1, 1, -1, 1

Vtedy bude mať transformácia taký pomer, aby hodnota -1 zodpovedal nultý bod a hodnote 1 maximálne bod 255 v smere x a 242 v smere y. Počet bodov pripadajúcich k premennej je celočíselná hodnota.

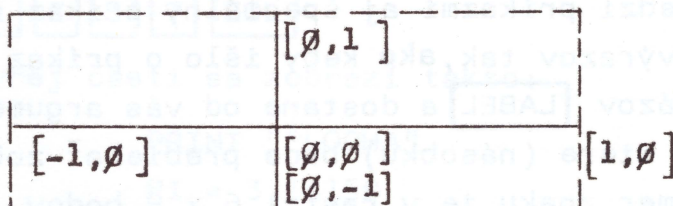
Na vykreslenie osí v danom SCALE sa používa príkaz **AXES**, ktorý dostane súradnice daného priesečníka osí. Ak chcete,



napríklad, vykresliť osi pre predchádzajúci SCALE v strede, príkazový riadok bude vyzeráť takto:

**AXES  $\emptyset$ ,  $\emptyset$**

Po vykonaní tohto príkazu sa zobrazia vyžadované osi v danej mierke:



Aby ste mohli premiestňovať miesto začiatku kreslenia v danom priestore SCALE, bol vytvorený príkaz **MOVE**, ktorý dostane súradnice x, y daného bodu. Teraz budeme pokračovať v našom príklade. Žiadajte, aby sa začalo kresliť najprv so súradnicami  $[0,0]$ .

Preto bude mať príkaz tvar:

**MOVE  $\emptyset$ ,  $\emptyset$**

Po vykonaní príkazu si mikropočítač zapamätá túto súradnicu ako začiatočnú. Výkonným príkazom na kreslenie spojnic s podporou lineárneho interpolátora alebo iba na aktivovanie daného bodu je príkaz **PLOT**. Tento príkaz má povinnú časť argumentov, ktoré majú udať dané miesto (súradnice x, y), a nepovinnú časť, ktorá udáva, či ide o spojnicu tohto bodu s predchádzajúcim bodom, ktorý je teraz začiatočným bodom, alebo len aktivovanie bodu danými súradnicami.

Príklad: V nadväznosti na predchádzajúce príkazy treba spojiť začiatok osí s bodom, ktorý má súradnice  $[1, 1]$ . Tvar -

**PLOT 1,1** vykoná požadované zobrazenie.

Ak máte záujem iba o zobrazenie bodu, dostane príkaz aj časť určená na zdvihnutie kresliaceho pera,

**PLOT 1, 1, 1**

Táto časť má byť iná ako nula, lebo v opačnom prípade (ak sa rovná  $\emptyset$ ) má prázdny účinok.

Keď sa zamyslíme nad týmto príkazom zistíte, že určitým spôsobom nahradí predchádzajúci príkaz MOVE premiestňovanie kresliaceho pera a v spojení s príkazom MOVE nahradza príkaz AXES na kreslenie súradnicových osí.

S týmito príkazmi by sme mohli vystačiť pri grafickom znázorňovaní funkcií, motívov a podobne. Písanie výrazov v pracovnej bodovo orientovanej oblasti je obtiažne, preto má mikropočítač medzi príkazmi aj špeciálny príkaz na vykresľovanie znakov, výrazov tak, ako keby išlo o príkaz **PRINT**. Tento príkaz má názov **LABEL** a dostane od vás argumenty, ktoré určujú, v akej miere (násobku) bude prebiehať zobrazovanie. Základný rozmer znaku je v rastrí 6 x 8 bodov, z toho znak zaberá 5 x 7 bodov. Podľa želania môžete tento raster zväčšiť v obidvoch smeroch. Vysvetlím to na nasledujúcom príklade:

Príklad: Žiadúce je v danom **SCALE** a v danom **MOVE** zobraziť text "PMD-85" s dvojnásobným rozmerom (základný rozmer je rozmer znázornený v dialógovom riadku), teda v rastrí 12 x 16 bodov, z toho samotné znaky budú zaberáť priestor 10 x 14 bodov.

Potom musíme napísať príkaz takto:

```
LABEL 2, 2; "PMD-85"
```

Je len samozrejmé, že v prípade väčšej mierky násobku rozmeru sa zobrazuje iba tá časť, ktorá je viditeľná v oblasti SCALE. Po vykonaní tohto príkazu zostáva kresliace pero za posledným vykresleným znakom.

Ak použijete príkaz LABEL v programovom režime, po prvom definovanom násobku rozmeru môžete uvádzať skrátený zápis vo forme "x" namiesto konkrétnych opakujúcich sa výrazov mierky. Napríklad:

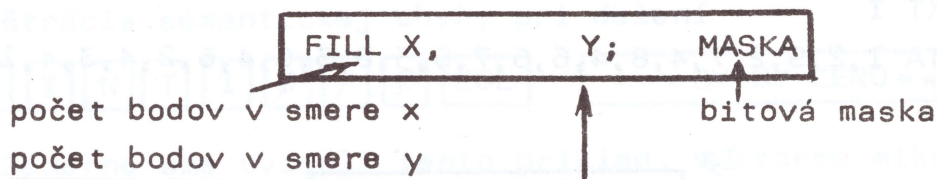
```
15 LABEL 1,5; SIN (A)
```

```
20 LABEL x; COS (A)
```

Dostali sme sa k poslednému príkazu, a to k príkazu **FILL**. Tento príkaz je veľmi užitočný vzhľadom na to, že môže priamo od ľubovoľného bodu (MOVE) v danom SCALE vykresliť určitý počet bodov v smere x, y v zvolenej bitovej maske.



### Formát príkazu



Na porozumenie výrazu maska uvedieme niekoľko príkladov, z ktorých bude zrejmy jej význam.

FILL 1, 1; 5

bitová maska je 5, t.j..1 0 1 0 0 0 0 0  
a vykonaním príkazu sa jednotlivé bity zľava vykreslia v maske s veľkosťou 1 x 1 (na obrazovke sa objaví maska vo forme bodka, medzera, bodka)

FILL 1, 242; 1

vykreslenie zvislej čiary s hrúbkou jedného bodu

FILL 2, 242; 2 1

vykreslenie troch zvislých čiar s hrúbkou dva body

FILL 255, 1; 1

vykreslenie vodorovnej čiary

FILL 255, 242; 1

ak sa začína v ľavom dolnom mieste, tak príkaz vykreslí celý pracovný priestor

#### Poznámka:

Vykresľovanie bodov sa uskutočňuje zdola nahor, preto po splnení príkazu zostáva pracovný bod v tomto poslednom bode. Ak máte nejaké nejasnosti, ozrejmíte si ich na ďalších príkladoch. Tieto príklady vám v spolupráci so štandardnými príkazmi dajú obraz o používaní grafických povelov. Každý príklad vložte do pamäti a odštartujte ho.

#### Príklad č. 27 Demonštrácia príkazu PLOT

10 REM \*\* STROMCEK \*\*

20 GCLEAR

25 SCALE 0, 10, 0, 10

30 READ X, Y

40 MOVE X, Y

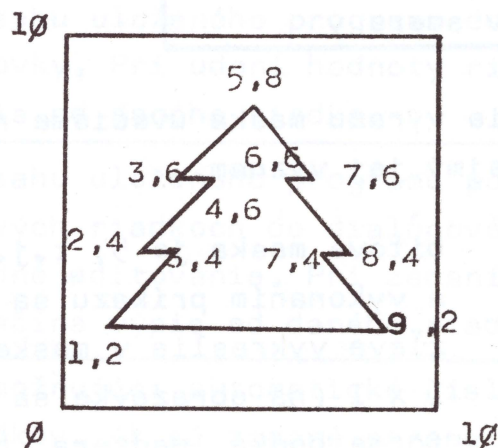
50 FOR I = 1 TO 11

60 READ X, Y

70 PLOT X, Y

80 NEXT I

90 DATA 1,2,9,2,7,4,8,4,6,6,7,6,5,8,3,6,4,6,2,4,3,4,1,2



Úloha č. 15 Rozšířte program tak,

a) aby mal strom kmeň,

b) na celú obrazovku sa vykreslí 100 znakov  
"X" s náhodne generovanou polohou

Príklad č. 28

10 REM X HISTOGRAM percentuálnych hodnôt X

20 GCLEAR

30 SCALE 0, 10, 0, 250

40 FOR I = 0 TO 9

50 READ A

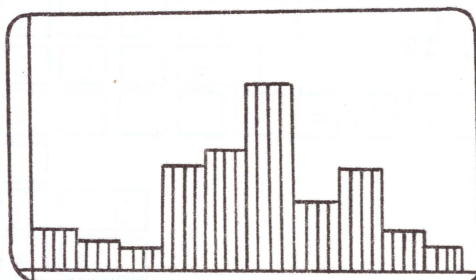
60 MOVE I, 0

70 FILL 15, 2 X A; 1

80 NEXT I

90 AXES 0, 0

100 DATA 23, 13, 10, 58, 64, 80, 25, 60, 22, 1





Úloha č. 16 Upravte program tak, aby sa nad každým stĺpcom objavilo jeho poradové číslo.

Príklad č. 29

1Ø REM NAPIS

2Ø GCLEAR

3Ø X = 1984

4Ø SCALE Ø, 1Ø, -1, 1

5Ø MOVE 1, Ø

6Ø LABEL 3, 2; "ROK"; X

ROK 1984

Poznámka: Všimnite si väčší rozdiel vzdialenosti medzi znakom "K" a číslicou "1". Je to spôsobené tým, že pri kladnom čísle sa nezobrazuje znamienko plus a jeho miesto sa ponecháva. Ak chcete túto záležitosť odstrániť, máte jedinou možnosť: vypísať prevod čísla do reťazca (bez znamienka).

Úloha č. 17

a) urobte úpravu podľa poznámky

b) zmeňte veľkosť nápisu

c) zmeňte polohu nápisu

Príklad č. 30 Demonštrácia príkazu AXES

1Ø REM \* MREZA \*

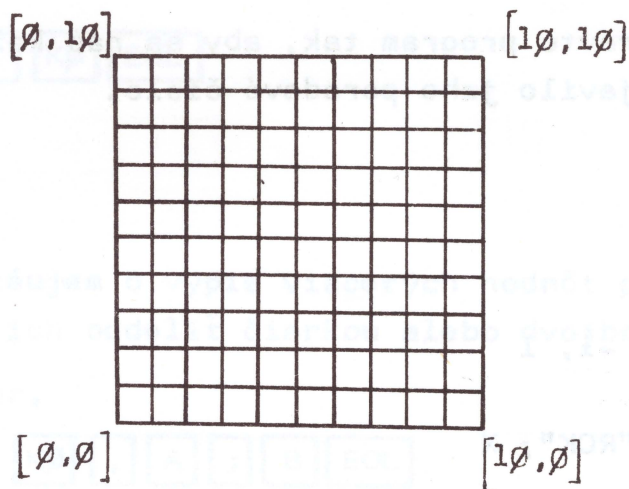
2Ø GCLEAR

3Ø SCALE Ø, 1Ø, Ø, 1Ø

4Ø FOR I = Ø TO 1Ø

5Ø AXES I, I

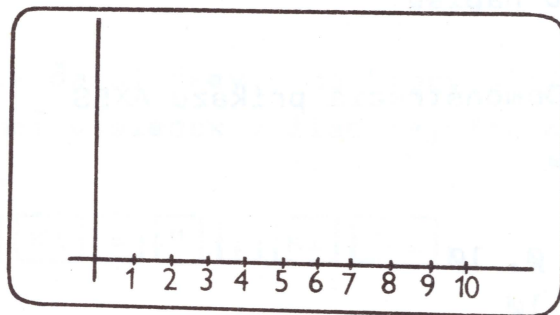
6Ø NEXT I



Úloha č. 18 Zvoľte SCALE tak, aby ste mali rozmerovo rovnaké štvorce.

Príklad č. 31 Demonštrácia príkazu FILL

```
10 REM POPIS OSI X
20 GCLEAR
30 SCALE -1, 11, -1, 11
40 AXES 0, 0
50 FOR I = 1 TO 10
60 MOVE I, 0
70 FILL 1, 4; 1
80 MOVE I-.4, -1
90 LABEL 1, 1; I
100 NEXT I
```



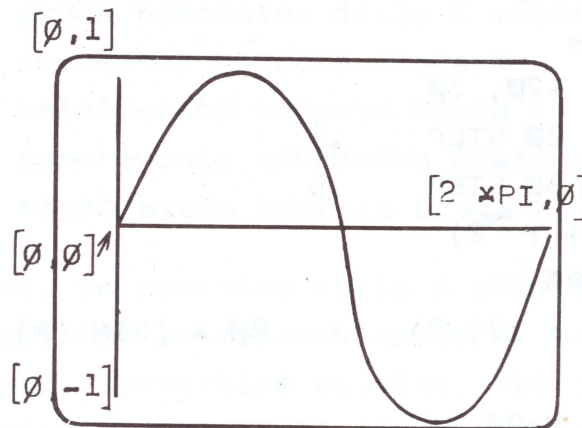


Úloha č. 19 Popíšte aj os Y

Doplňte program o vykreslenie zvolenej funkcie

Príklad č. 32 Demonštrácia príkazu PLOT vo funkcii sin (x)

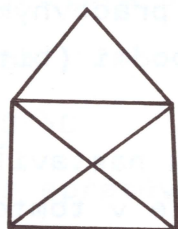
```
10 REM * SIN (X) *  
20 GCLEAR  
30 P I = 22/7  
40 SCALE 0, 2 * P I, -1,1  
50 AXES 0,0  
60 MOVE 0, 0  
70 FOR X=0 TO 2 * P I + .3 STEP P I/20  
80 PLOT X, SIN (X)  
90 NEXT X  
100 END
```



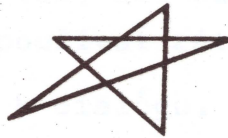
Úloha č. 20 Vyšrafujte funkciu sin X zvislými čiarami.

Ďalšie úlohy budú na zostavenie programov.

Úloha č. 21 Zostavte program na znázornenie elipsy, kružnice a nasledujúceho obrazca,



Úloha č. 22 Zostavte program na náhodné generovanie miesta na vykreslenie rovnakých desiatich hviezdíčiek, ktoré sú na obrázku:



Na záver tejto časti vám predkladáme nasledujúci program. Vložte ho do pamäti a odštartujte ho.

Príklad č. 33

```
5 REM * BUBLINA *
10 SCALE -20, 30, -20, 30
20 FOR Y = -20 TO 20 STEP .5
30 FOR X = -40 TO 30 STEP .5
40 R = SQR (X^2 + Y^2)
50 IF R = 0 THEN 80
60 PLOT X+1 * ((Y + 17)/2), Y + 20 * (SIN (R)/R)
70 GOTO 90
80 PLOT (Y+17)/2, Y+20
90 NEXT X
100 MOVE -20, -20
110 NEXT Y
120 END
```

Doteraz sme sa zaoberali prácou nad bodovo prístupným pracovným priestorom: Pri niektorých aplikáciach je však vykonávanie týchto príkazov pomalé - vyžadujú si kreslenie viacerých bodov naraz. Takéto možnosti poskytujú nasledujúce dva príkazy, ktoré pracujú nad byte orientovaným pracovným priestorom. Umožňujú naraz manipulovať so šiestimi bodmi (bitmi) vrátane ich farebného kódu.

Prvý príkaz **BMOVE** má za úlohu nastaviť ukazovadlo miesta kreslenia - podobne ako MOVE - , ale v tomto prípade nie je možné meniť mierku pracovnej časti (SCALE), pretože je pevne definovaná, a to v smere osi X v rozsahu 0 až 47 a v smere



osi Y, 0 až 242,

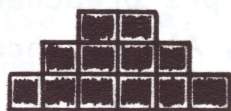
Pozícia 0, 0 je pevná a nachádza sa v ľavom hornom rohu pracovnej časti. Ako sme už skôr uviedli v smere osi X je táto oblasť o 32 bodov väčšia, má  $48 \times 6 = 288$  bodov,

Príklad: `BMOVE 10, 10`

Nastavenie miesta vykresľovania v smere osi X desiaty byte a v smere osi Y 10 riadkov,

Výkonným povelom na kreslenie znaku je príkaz `BPLOT`, ktorý musí od vás dostať názov reťazca a argument určujúci, ako sa má tento reťazec rozkladať v smere osi X. Postupnosť kreslenia je v smere zhora nadol. Tento príkaz dáva možnosť vykresľovania rozličných motívov a znakov, ktoré nie sú štandardné. Postup vytvorenia tohto reťazca údajov vám vysvetlíme na jednoduchom príklade,

Potrebuje vykresliť trojuholník so základňou 6 bodov a s výškou 3 body - pozri obrázok:



Jednotlivé riadky treba definovať, preto nakreslíme do rastra  $6 \times 3$

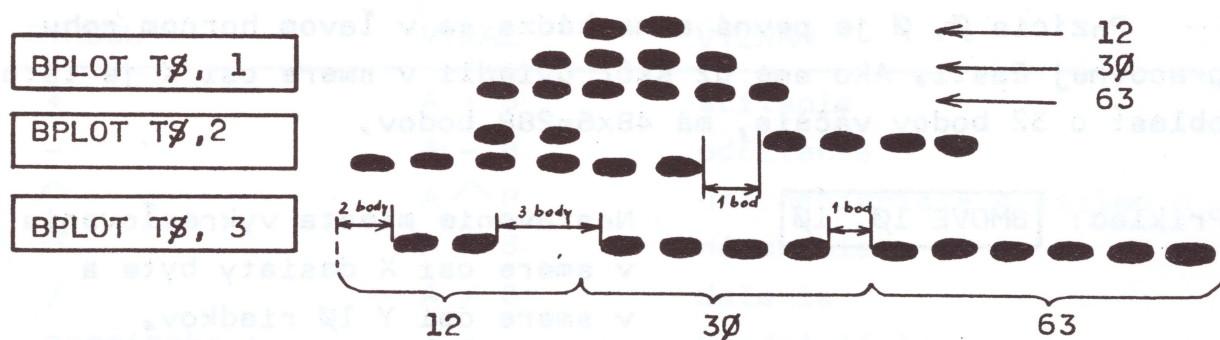
| raster | binárne | decimálne |
|--------|---------|-----------|
|        |         | 12        |
|        |         | 30        |
|        |         | 63        |

najnižšia váha      farba

a jemu zodpovedajúce stavy v binárnej a decimálnej forme. Takto sme získali údaje pre reťazec, a preto už stačí ich vložiť,

`T$ = CHR$(12) + CHR$(30) + CHR$(63)`

Rozkreslenie definovaného reťazca `T$` umožní príkaz `BPLOT` v týchto možných variantoch:



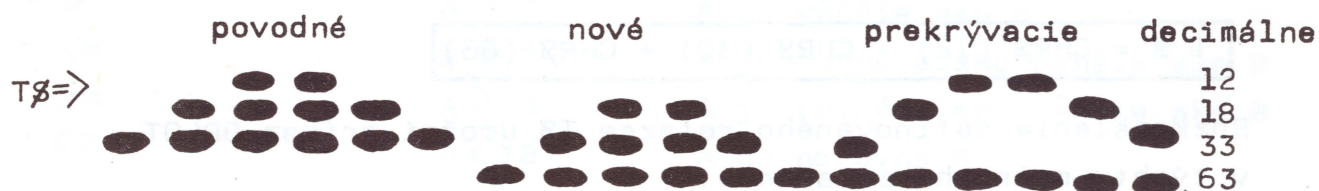
Ďalšie zvyšovanie argumentu nemá zmysel; rozkresľovanie sa skončí už pri treťom znaku.

Ak by ste požadovali vykreslenie motívu s väčším rastrom ako 6 bodov, použijete rovnaký postup. Argument sa zväčšuje len o toľko, koľko šestic bodov treba zoradiť vedľa seba.

Na niekoľkých príkladoch si precvičte tento príkaz, aby ste si ho osvojili. Vytvorte si takto napríklad motív závodného autíčka.

Zmazať motív môžete tak ako pri predchádzajúcich príkazoch, teda opätovným vykreslením. Ale ak chcete, aby sa motív pohyboval, nepoužite tento spôsob: postrehli by ste (blikanie) toto zmazávanie a znovuvykreslenie.

V tomto prípade odporúčame použiť filozofiu vzájomného prekrývania, teda vytvorenie nového reťazca, ktorý bude väčší o toľko, o koľko chcete posunúť v danom smere; časť, ktorá sa má vymazať, sa premaže novými údajmi, pretože vykresľovanie bodu je dané funkciou EXOR, teda negáciou pôvodného bodu. Vysvetlíme to na nasledujúcom príklade. Váš trojuholník chcete posunúť o jeden riadok smerom dolu, Vytvoríte si nový reťazec, ktorý bude mať o jeden údaj viac.





Nové (prekrývacie) údaje vložte do ďalšieho reťazca

T1\$ = CHR\$(12) + CHR\$(18) + CHR\$(33) + CHR\$(63)

Potom vykonajte túto sekvenciu:

```
BMOVE 20, 20
BPL0T T$, 1
BPL0T T1$, 1
..BMOVE 20, 21
. BPL0T T1$, 1
.
.
.
atď.
```

Pokúste sa samostatne urobiť posun do smeru X a zostavte program plynulého behu motívu. Ak by ste to hravo zvládli, dávame vám na záver tejto časti dva príklady na preštudovanie,

Vložte ich do pamäti a odštartujte !

#### Príklad č. 34

```
10 REM * STRINGY *
15 GCLEAR
20 A$ = " " : M$ = " "
30 FOR I = 1 TO 7
40 READ A
50 A$ = A$ + CHR$(A)
60 NEXT I
70 FOR I = 1 TO 7 : READ A : M$ = M$ + CHR$(A) : NEXT I
80 DATA 0, 14, 17, 16, 30, 17, 46, 0, 42, 62, 43, 42, 42, 42
90 D$ = M$ + A$ + M$ + A$
100 BMOVE 20, 150
110 BPL0T D$, 1
```

Zobrazenie vám neprezradíme, ale vznikne nápis malými písmenami. Zistite si ho spustením programu!

Úloha č. 23 Vytvorte iné nápisy!

Príklad č. 35

```
10 REM * tranzistor *
20 GCLEAR
30 A$ = ""
40 FOR I = 1 TO 54
50 READ A
60 A$ = A$ + CHR$(A)
70 NEXT I
80 DATA 0, 63, 4, 48, 0, 3, 8, 56, 5, 4, 32, 9, 2, 16, 17, 2, 8
90 DATA 17, 49, 4, 32, 49, 2, 32, 63, 1, 32, 49, 1, 32
100 DATA 49, 2, 32, 50, 4, 16, 2, 8, 16, 4, 16, 8, 8, 32, 4
110 DATA 48, 0, 3, 0, 63, 4, 0, 0, 0
120 DISP "SURADNICE X, Y"
130 INPUT X, Y
140 BMOVE X, Y
150 BPLOT A$, 3
160 GOTO 120
```

Úloha č. 24

- a) vytvorte ďalšie schematické značky v ďalších reťazcoch,
- b) doplňte program o možnosť kreslenia pomocou klúčov K0-K10.

Osvojením doteraz uvedených informácií a úkonov ste sa dostali na koniec popisu grafických príkazov BASIC-G. Ak ste postupovali podľa uvedených pokynov a vyriešili zadané úlohy, sme si istí, že pochopíte nasledujúcu časť. Táto časť sa týka príkazov určených na komunikáciu so strojovým jazykom 8080. Na túto činnosť sú v mikropočítači vymedzené nasledujúce príkazy:

| PRÍKAZ | VÝZNAM  |
|--------|---|
| CODE   | Vykonanie programu v strojovom kóde zadanom v reťazci                     |
| ROM    | Presun programov, dát do pevne zvolenej časti pamäti a jeho odštartovanie |
| URS    | Volanie podprogramu v strojovom kóde                                      |



Príkaz **CODE** umožňuje napísať program v strojovom kóde priamo počas písania programu v jazyku BASIC. Program v strojovom kóde uložíte do reťazca. Treba si zapamätať dve veci:

program v strojovom kóde sa interpretuje v pamäťovej lokácii 7F00H. Ak váš program používa vnútorné slučky, musí byť zadaná správna stránka pamäti (skoku), program musí byť ukončený inštrukciou RET (C9).

Tento príkaz si vyskúšajte na nasledujúcom príklade:

Vložte do A\$ program na výpis znaku A do pracovnej časti pod spracovaním v strojovom kóde.

```
A$ = "3E413234C1CD0085C9"
```

Potom stačí už len vydať povel

```
CODE A$
```

a vypíše sa žiadaný znak. Ak chcete podprogram zopakovať viackrát, môžete použiť zápis:

```
CODE A$, A$, A$
```

Príkaz **CODE** sa s obľubou používa pri premodifikovaní určitých dát, sekvencií v samotnej interpretácii. Treba len správne zvážiť, čo sa dá modifikovať, aby nenastalo zrušenie systému. Možno ho takisto použiť pri doplnení - rozšírení príkazov, ale na to musíte mať detailnejšie znalosti o vnútornej stavbe mikropočítača.

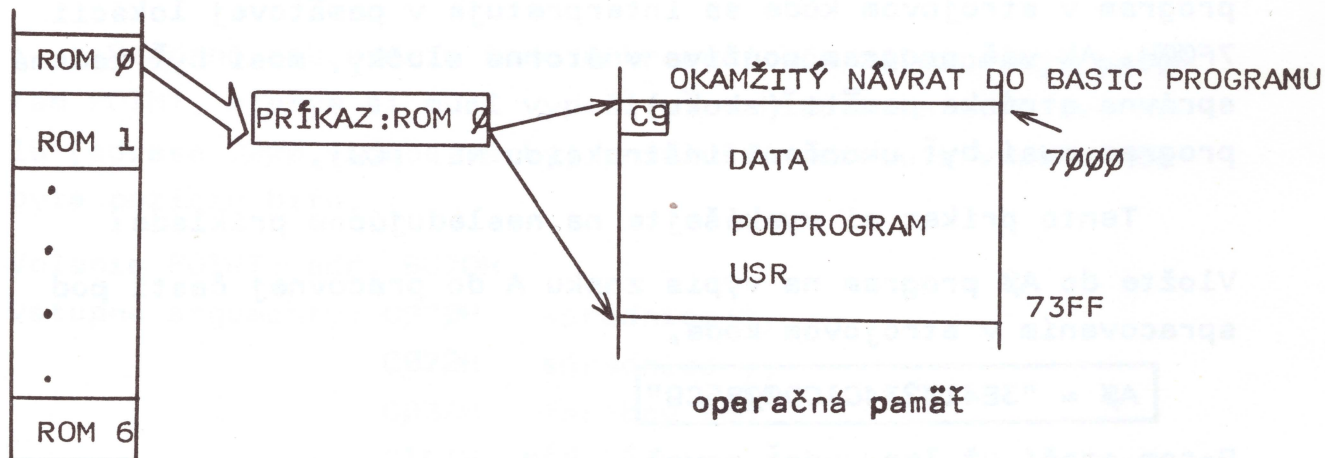
Ďalším príkazom je príkaz **ROM**.

Tento príkaz bol vytvorený pre tých užívateľov, ktorí majú modul ROM, pretože umožňuje transportovať dáta zo siedmich možných užívateľských pozícií. Tieto pozície sú očíslované od 0 do 6. Každá pozícia má rozsah 1 k byte, premiestni sa na stránku 7000H a odštartuje sa od tejto adresy. Obsahom pamäti môže byť aj samotný program, prípadne data, ktoré program využíva, alebo nové exekutívy, ktoré sa včlenia do interpretéra, premodifikujú ho a podobne. Užívateľ v nich môže mať uložené štandardné programy, takže magnetofón môže používať len ako pamäť dát. Možností kombinácií je veľmi veľa, pretože interpretér je uložený v pamäti RWM.

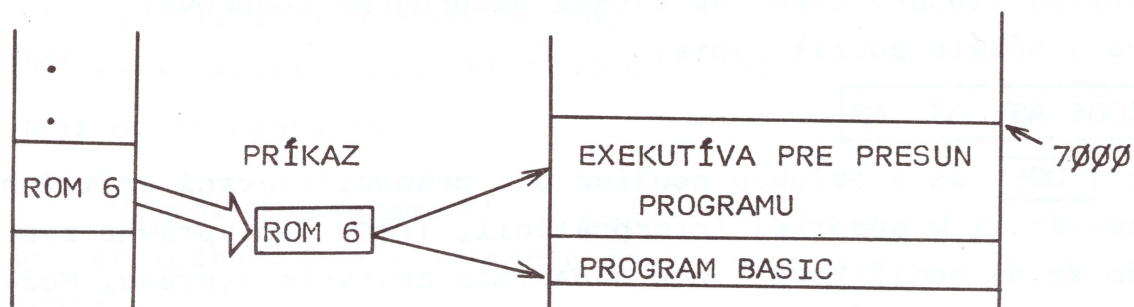
Príklad na použitie príkazu:

```
ROM 0
```

Východ, teda návrat nazad do vlastného programu BASIC sa uskutočňuje pomocou inštrukcie RET (C9). Na nasledujúcich obrázkoch sú uvedené niektoré aplikácie tohto obrázka:



Užívateľské pozície na module ROM:



**Poznámka 1:** Po presune obsahu užívateľskej ROM 6 sa odštartuje exekutíva, ktorej úlohou je naplniť programovú pamäť užívateľským programom. Maximálna dĺžka tohto programu je 6,5 kB,

Tvar exekutívy:

```
PUSH H          7000 E5      ; Programový ukazovateľ
CALL TRANSF     CD 00 8C    ; Presun dát
DW1             00 00      ; od
DW2             00 00      ; koľko
DW3             00 24      ; kam
JMP RESTART     C3 04 1E    ; návrat + ochrana
```

Podprogram TRANSF zabezpečuje transport dát z užívateľského modulu ROM; jeho argumentmi sú DW1 adresa prvej pozície ROM, odkiaľ sa bude uskutočňovať transport,

DW2 - dĺžka transformovaných dát zvýšená o 256, teda pre 1 kB



je to hodnota 05 00 (HEX).

DW3 - miesto - adresa, kam sa majú ukladať prenášané dáta.

Adresa 2400 (H) je začiatkové miesto pre programovú pamäť.

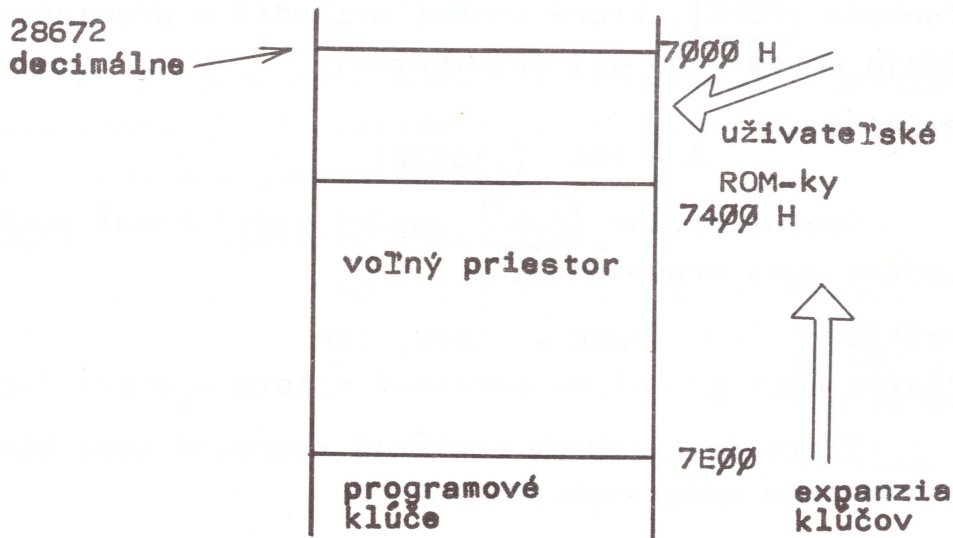
Podprogram RESTART zabezpečí ochranu uloženého programu vzhľadom na existujúce nastavené ukazovatele v zápisníku BASIC.

Poznámka 2 : Zapamätajte si, že vždy pri vyvolaní príkazu ROM sa tento obsah ukladá na konštantné miesto - adresa 7000 (H). Ak je vyvolaných viacero príkazov ROM za sebou, vždy sa v tejto oblasti nachádza obsah poslednej ROM-ky.

Poznámka 3 : Pretože interpretér BASIC pracuje s medzikódom a jeho programové riadky explicitne obsahujú práve platné pozície v programe, treba presne dodržať obsah programovej pamäti, ktorá sa začína na pevnej hodnote - na adrese 2400 (H) a končí sa na adrese udanej v ukazovateli 5E7A (H).

Na záver tohto príkazu vám odporúčame zo začiatku využívať tento príkaz iba na dáta prípadne na programy USR (strojový kód) alebo používať štandardne vypracované príkazy ROM. Prvá vyvinutá pamäť ROM je užívateľský program na programovanie pamäti 8708/8608, ku ktorému bude dodávaný aj programátor.

Posledným príkazom na prácu v strojovom kóde je príkaz - funkcia **USR**. Umožňuje vykonať program v strojovom kóde, ktorý sa začína na zadnej adrese a priradiť obsah akumulátora premennej v programe BASIC. Na písanie vlastných strojových (binárnych) programov je určená voľná pamäťová oblasť, ktorá je znázornená na nasledujúcom obrázku :



Užívateľský podprogram sa ukončuje inštrukciou RET (C9).

Tvar príkazu:

A = URS (28672)

V operačnom systéme sa nachádzajú podprogramy, ktoré môžete využiť pri práci v BASIC-u.

Najbežnejšie podprogramy, ktoré môžete využiť, sú tieto:

| Názov  | Adresa volania | Význam  |
|--------|----------------|---|
| RPOINT | - 31258        | Čítanie hodnoty bodu ukazovaného podľa MOVE. Priradená premenná môže mať hodnotu 0 alebo 1 podľa toho, či je zobrazený daný bod.  |
| BREAD  | - 31501        | Čítanie hodnoty bodu ukazovaného podľa príkazu BMOVE. Priradená premenná nadobudla hodnoty 0 - 255 včítane farebného kódu. Túto funkciu možno využiť pri tzv. "hard copy" pracovného priestoru. |
| KEYBD  | - 31583        | Čítanie hodnoty stlačeného klávesu. Priradená premenná nadobudne hodnotu práve stlačeného klávesu.  |

Vzhľadom na ďalší príklad si zopakujeme ešte dva príkazy, s ktorými môžete pracovať so systémom. Je to príkaz - funkcia **PEEK**, ktorá umožní priradiť k premennej hodnotu údaj adresovanú jej argumentom.

Príklad:

A = PEEK (-16322)

Opačný príkaz **POKE** umožní modifikovať pamäťové miesto zadané jeho argumentom.

Príklad:

POKE - 15878, 168

Zapíše údaj AB (H) na adresové miesto - 15878 decimálne (ClFA H).

Tieto dva príkazy umožňujú prenášať celé bloky dát medzi systémom a programom BASIC.



V ďalšom príklade je znázornené, ako možno spojiť tieto dva príkazy,

Príklad č. 36

```
1Ø REM * VYPIS ZNAKU *  
2Ø A = 28672  
3Ø FOR I = Ø TO 7  
4Ø READ D  
5Ø POKE A + I, D  
6Ø NEXT I  
7Ø DATA 62, 65, 50, 52, 193, 195, Ø, 133  
8Ø U =USR (A)  
9Ø END
```

Komentár k jednotlivým riadkom:

2Ø - nastavenie pamäťovej lokácie pre binárny program  
3Ø, 4Ø, 5Ø - presun binárneho programu zapísaného ako DATA  
8Ø - vykonanie binárneho programu

Úloha č. 25

Napište program posunu binárneho programu na inú pamäťovú lokáciu.

Zjednodušte daný program podľa doteraz známych príkazov a porovnajte ich.

To boli príkazy, funkcie na prácu s komunikáciou so základným systémom. Táto činnosť si vyžaduje dôkladné poznanie strojového kódu mikropočítača. V opačnom prípade by ste boli iba pasívnymi užívateľmi týchto vysoko efektívnych príkazov. V ďalšej kapitole sa budeme zaoberať tým, ako možno robiť do mikropočítača určité zásahy, modifikácie. Prísne dodržiavajte pokyny mikropočítača. Môžete ho totiž úplne zničiť. Vtedy vám neostáva nič iné, iba stlačiť tlačidlo RST, čím sa dostanete na úplný začiatok.

G..4 MODIFIKÁCIA SYSTÉMU

Keďže si všetky dôležité adresové miesta mikropočítač ukladá do svojho zápisníka, ktorý sa nachádza v operačnej pamäti RWM, môžete ich modifikovať.

Na to mikropočítač sprístupní nasledujúce ukazovatele:

| Ukazovateľ<br>[DEC] | Hodnota<br>[DEC] | Význam   |
|---------------------|------------------|--|
| - 15878             | 168              | modifikácia bodu na jeho komple-<br>ment (EXOR)                                    |
|                     | 176              | modifikácia bodu na jeho nastave-<br>nie (SET)                                     |
|                     | 175              | modifikácia bodu na jeho nulovanie<br>(RESET)                                      |
| - 16326             | Ø                | modifikácia farby (módu)- 100 % jas  |
|                     | 64               | - 50 % jas   |
|                     | 128              | -100 % jas + blikanie  |
|                     | 192              | - 50 % jas + blikanie  |
| - 16324             | NIŽŠÍ            | } hodnoty adresy ukazovateľa tabuľky<br>na výpis znakov ASCII                      |
| - 16323             | VYŠŠÍ            |  |
| - 16322             | NIŽŠÍ            | } hodnoty adresy ukazovateľa výpisu<br>znaku na miesto výpisu v pracovnej<br>časti |
| - 16321             | VYŠŠÍ            |  |

Tieto údaje v zápisníku sa modifikujú príkazom **POKE**. Na ovládanie akustického tónu a diód LED možno použiť príkaz **OUT** s nasledujúcou modifikáciou.

| Adresa<br>PORT<br>[DEC] | Hodnota<br>[DEC] | Význam                |
|-------------------------|------------------|-----------------------|
| 134                     | Ø                | akustický tón vypnutý |
|                         | 1                | akustický tón 1       |
|                         | 2                | akustický tón 2       |
|                         | 3                | akustický tón 3       |
| 134                     | Ø                | diódy LED vypnuté     |
|                         | 4                | dióda LED zapnutá     |
|                         | 8                | dióda LED zapnutá     |

Zvláštnosťou LED 1 je to, že je galvanicky spojená s akustickým meničom. Menič možno programovo ovládať prostredníctvom tejto linky, a tým vytvárať svoje užívateľské tóny.



Nasledujúci príklad je demonštráciou zahniezdzenia binárneho programu do programu BASIC. Tento program umožňuje vydávať tónovú stupnicu použitím jestvujúcej klávesnice.

Príklad č. 37

































```

10 DIMT (255)
20 FOR I = 1 TO 30
30 READ K
40 READ T
50 T (K) = T: NEXT I
60 DATA 81, 255, 87, 228, 69, 203, 82, 192, 84, 170, 90, 151
65 DATA 85, 134, 73, 127, 79, 112, 80, 100, 64, 94, 92, 83, 131
70 DATA 74, 132, 65, 133, 61, 49, 255, 50, 242, 51, 215, 52, 203
75 DATA 53, 181, 54, 160, 55, 142, 56, 134, 57, 119, 48, 106, 95
80 DATA 48, 106, 95, 100, 255, 88, 128, 78, 119, 69, 141, 61
100 A$ = "0E002A007055DBF6EE04D3F615C20C7F798D4FD2057F25C2057FC9"
110 A1 = 28672 : POKE A1 + 1, 80
120 K = USR (-31583)
130 T = T (K)
135 IF T = 0 THEN 110
140 POKE A1, T : CODE A$
150 GOTO 120

```

Na záver tejto kapitoly si prevedieme, ako možno premodifikovať systém na výpis iných znakov, ako sú znaky ASCII.

Na program je táto požiadavka: zobrazíť v bodovo orientovanom priestore znak "autíčko". Jeho kreslenie má byť v režime nastavovania bodov (nie komplement). Ďalej žiadame väčší rozmer autíčka, ako dovoľuje základný raster bodov na znaky.

| 1   | 2   | 4   | 8   | 16  | 32  | kód HEX |
|---|---|---|---|---|---|---------|
|  |   |  |  |   |  | 2D      |
|  |  |  |  |  |  | 3F      |
|  |   |  |  |   |  | 2D      |
|   |   |  |  |   |   | 0C      |
|   |   |  |  |   |   | 0C      |
|  |   |  |  |   |  | 2D      |
|  |  |  |  |  |  | 3F      |
|  |   |  |  |   |  | 2D      |

Kód motívu "autíčko" priradíte znaku ASCII, a to SPACE (medzera), ktorý sa nachádza v tabuľke na pozícii 32 (DEC).

Ukazovateľ tabuľky ASCII sa nachádza na adrese - 16324, kde je uložená nižšia hodnota tabuľky 00HEX, a na adrese -16323, kde je uložená vyššia hodnota adresy tabuľky 85 HEX.

Na tieto dve miesta vložíme našu novú tabuľku s novými znakmi.

Vypočítate to takto:

kód motívu vložte na adresu 7000 H. Odčítajte 32 x 8, t.j. 256 byte. Dostanete báзовú adresu novej tabuľky, t.j. 6F00 H. K tejto adrese treba ešte pripočítať +1; vtedy možno použiť na výpis všetkých 8 riadkov motívu, pretože pri výpise znaku ASCII, ktorý je v rasti 5 x 7 bodov, sa pre väčšiu rýchlosť výpisu preskakuje posledný riadok.

Záver výpočtu: Na ukazovateľ tabuľky vložte nižšiu hodnotu báзickej adresy 01 [HEX] = 1 [DEC] a vyššiu hodnotu 6F [HEX] = 111 [DEC]. Túto modifikáciu môžete vykonať príkazom CODE alebo POKE.

Program:

Komentár:

|                           |                     |
|---------------------------|---------------------|
| 5 REM * AUTICKO *         |                     |
| 10 POKE - 15878, 176      | modifikácia bodu    |
| 20 GCLEAR                 |                     |
| 30 SCALE 0, 10, 0, 10     |                     |
| 40 A\$ = "21016F223CC0C9" | modifikácia tabuľky |
| 50 CODE A\$               |                     |
| 60 MOVE 1, 1              |                     |
| 70 LABEL 2, 2; "  "       | vykreslenie motívu  |
| 80 END                    |                     |

|  
medzera



## H. PŘÍKAZY NA VSTUP/VÝSTUP

V tejto časti návodu Vás bližšie oboznámime s interfaceovou doskou. Robíme to preto, že pomocou tohto modulu (vstupno - výstupných kanálov) mikropočítač komunikuje s okolím a aj z toho dôvodu, že od stupňa poznania jednotlivých kanálov závisí to, v akej miere využijete mikropočítač. Najskôr si stručne niečo povieme o jednotlivých kanáloch a potom o spôsobe programovania.

### 1. KANÁLY INTERFACEOVEJ DOSKY OSOBNÉHO MIKROPOČÍTAČA PMD-85

Interfaceová doska obsahuje:

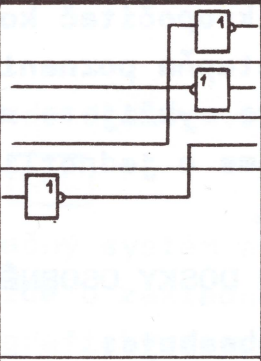
- dva paralelné programovateľné vstupy/výstupy (GPIO),
- jeden štandardný styk IMS - 2 (HP - IB),
- jeden štandardný sériový styk V.24,
- jeden styk pre bežný kazetový magnetofón,
- jeden časovač s podporou hodín reálneho času,
- aplikačný (všeobecný) vstup/výstup lokálnej zbernice mikropočítača.

Dekódovanie, t.j. výber jednotlivých kanálov je realizované obvodom MH 3205 z adresových liniek A7, A6, A5, A4.

#### 1.1 PARALELNÝ PROGRAMOVATEĽNÝ VSTUP/VÝSTUP

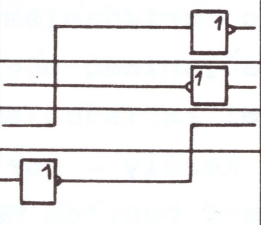
Tento interface je realizovaný obvodom MHB 8255 A a sprostredkúva styk so zariadeniami orientovanými na (paralelný styk (dierovač, snímač pásky, tlačiareň a pod.). Na prácu s periférnymi zariadeniami sú k dispozícii dva samostatne pracujúce kanály - porty A a B obvodu 8255 A. Každý kanál je oddelený budičom MH 8286, ktorý môže užívateľ ovládať prostredníctvom vyvedeného riadiaceho signálu pre vstup alebo výstup. V pokojovom stave je tento budič navolený (inicializovaný) na výstup, pripojením GND na túto linku je umožnený vstup údajov. Linky kanála A sú vyvedené na konektor K3 a linky kanála B na konektor K4. Okrem toho sú na konektory K3 a K4 vyvedené ešte po štyri linky portu C obvodu 8255 A a tri invertory MH 7405, ktoré môže užívateľ zo svojej strany použiť na úpravu riadiacich liniek kanála (linky portu C). Rozmiestnenie a označenie pripájacích miest konektorov K3 a K4 je na obr. 8.

PRIPÁJACIE MIESTA KANÁLA GPIO (kanál 4GPIO/0)

| Označ.prip.miesta | č.prip.miesta | pozn.   | č.prip.miesta | označ.prip.miesta                          |
|-------------------|---------------|---|---------------|--|
| GND               | 1             |  | 2             |  |
|                   | 3             |   | 4             |  |
|                   | 5             |   | 6             |  |
|                   | 7             |   | 8             | ovládanie smeru údajov<br>Log 0=DATA vstup |
| PC7               | 9             | DATA PORT<br>A  | 10            | PC6  |
| PC5               | 11            |   | 12            | PC4  |
| PA1 ↔             | 13            |   | 14            | ↔ PA0                                      |
| PA3 ↔             | 15            |   | 16            | ↔ PA2                                      |
| PA5 ↔             | 17            |   | 18            | ↔ PA4                                      |
| PA7 ↔             | 19            |   | 20            | ↔ PA6                                      |

KONEKTOR K3

PRIPÁJACIE MIESTA KANÁLA GPIO (kanál 4GPIO/1)

| Označ.prip.miesta | č.prip.miesta | pozn.   | č.prip.miesta | označ.prip.miesta                          |
|-------------------|---------------|---|---------------|--|
| GND               | 1             |  | 2             |  |
|                   | 3             |   | 4             |  |
|                   | 5             |   | 6             |  |
|                   | 7             |   | 8             | ovládanie smeru údajov<br>Log 0=DATA vstup |
| PC3               | 9             | KANÁL PORT<br>B   | 10            | PC2  |
| PC1               | 11            |   | 12            | PC0  |
| PB1 ↔             | 13            |   | 14            | ↔ PB0                                      |
| PB3 ↔             | 15            |   | 16            | ↔ PB2                                      |
| PB5 ↔             | 17            |   | 18            | ↔ PB4                                      |
| PB7 ↔             | 19            |   | 20            | ↔ PB6                                      |

KONEKTOR K4

Obr.8 ROZMIESTNENIE A OZNAČENIE PRIPÁJACÍCH MIEST KONEKTORA K3 a K4.



## 1.2 INTERFACE IMS - 2

Známy tiež pod označením HP-IB. Je tak isto realizovaný obvodom MHB 8255A s podporou hardware, ktorý predstavuje iba výkonové prispôsobenie vstupno-výstupných liniek. Programová exekutíva umožňuje prácu v režime "CONTROLLER", ktorý je pevne navolený PMD - 85. Tento interface je riešený výlučne programovacími prostriedkami. V prípade, že užívateľ nepoužíva tento interface je zneho možné získať dva paralelné vstupno/výstupné kanály.

Na konektor K5 je vyvedených 16 signálnych liniek a 2 zemiacie linky.

Správy prenášané po týchto linkách môžu byť viacvodičové a jednovodičové. Viacvodičové správy sa prenášajú po dátových vodičoch DIO 1 ÷ 8 linky portu A obvodu MHB 8255 A. Jednovodičové správy sa prenášajú po vodičoch DAV, NRFD, NDAC, IFC, ATN, REN, SRQ, EOI - linky portu B a C obvodu 8255 A, ale môžu byť prenášané aj po individuálnom dátovom vodiči DIO. Popis jednotlivých liniek a ich rozmiestnenie a pripojenie na konektor K5 je uvedené v nasledujúcej tabuľke:

| označenie prip. miesta - funkcia        | č.prip. miesta | č.prip. miesta | označenie prip. miesta - funkcia           |
|---|----------------|----------------|--|
| nezapojené                              | 1              | 2              | NRFD indikácia priprav. jednot. príjmu dát |
| DAV - dáta plat.                        | 3              | 4              | nezapojené                                 |
| NDAC indikuje, že jednotka prijala dáta | 5              | 6              | nezapojené                                 |
| GND                                     | 7              | 8              | GND  |
| nezapojené                              | 9              | 10             | REN-dialk, ovl.                            |
| SRQ-vyžiad. obsl.                       | 11             | 12             | PA3-D3 ( dáta ) ↔                          |
| EOI-koniec alebo hlásenie               | 13             | 14             | PA2-D2 ( dáta ) ↔                          |
| nezapojené                              | 15             | 16             | ANT - pozor                                |
| nezapojené                              | 17             | 18             | PA1-D1 ( dáta ) ↔                          |
| nezapojené                              | 19             | 20             | IFC - nulovanie styku                      |
| PA0 - D0 ( dáta ) ↔                     | 21             | 22             | PA7 - D7 ( dáta ) ↔                        |

|                     |    |    |                   |
|---------------------|----|----|-------------------|
| PA6 - D6 ( dáta ) ↔ | 23 | 24 |                   |
| PA5 - D5 ( dáta ) ↔ | 25 | 26 | PA4-D4 ( dáta ) ↔ |
| nezapojené          | 27 | 28 | nezapojené        |
| nezapojené          | 29 | 30 | nezapojené        |

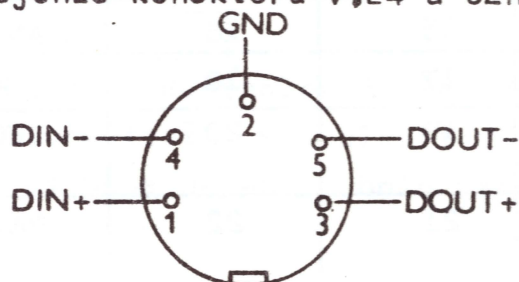
Obr. 9 KONEKTOR PRE IMS - 2 (HP-IB) - KANÁL 7

Na lepšie pochopenie činnosti tohto styku a na prácu s ním odporúčame preštudovať literatúru o IMS - 2.

### 1.3 SÉRIOVÝ STYK V.24

Sériový interface je realizovaný obvodom USART MHB 8251 (spoločný pre styk s magnetofónom). Vstupno výstupné linky sú galvanicky oddelené optočlenmi WK 164 13. Obvodová technika premeny signálu úrovne V.24 (V.24 - RS 232C, V.24 je prevzatý z amerického RS-232C) na úroveň TTL a opačne je jednoduchá. V našom prípade je použitá prúdová slučka, ktorá predstavuje jeden z najstarších sériových prechodov (používa sa na riadenie ďalekopisov a starších typov tlačiarň). Signál log 1 je predstavovaný prúdom 20 mA, log 0 prúdom do 4 mA. Potrebný prúd je dodávaný mikropočítačom (obvykle) a pripojené periférie je pasívne. Pri prepojení dvoch počítačov môže dôjsť ku kolízií, hlavne ak je interface s tranzistormi. Tento problém práve odstraňujú optické väzobné členy.

Z hľadiska druhu a rýchlosti prenosu je V.24 rozhraním pre asynchronný sériový prenos dát, kde jednotlivé znaky sú prenášané ako sled 8 bitov uvedených vždy jedným štartovacím bitom s nulovou úrovňou a ukončených jedným alebo dvoma stopbitmi s jedničkovou úrovňou. Používané prenosové rýchlosti sa pohybujú od 50 bitov až do 19200 bitov za sekundu. Zapojenie konektoru V.24 a označenie vývodov je na obr. 10.



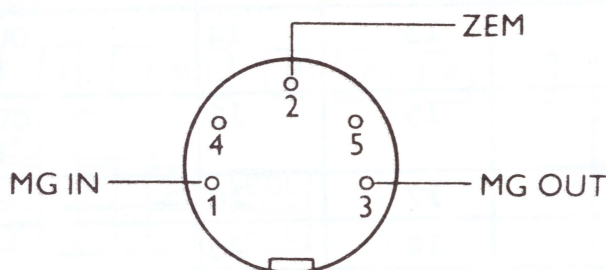
Obr. 10 PRIPOJOVACIE MIESTA KANÁLU V.24 (konektor K 6)



Vstupný signál (dátá) i vzorkovací kmitočť obvodu 8251 sú prepínané podľa voľby. Vzorkovací kmitočť 1200 Hz môže byť spoločný s magnetofónom, prípadne je možné, pomocou prepojky pripojiť na dosku z interného časovača 8253 kmitočť 2400 Hz.

#### 1.4 INTERFACE PRE KAZETOVÝ MAGNETOFÓN

Je realizovaný s podporou obvodu USART 8251 a jednoduchým dekóderom dát a hodín. V princípe je záznam robený modulovaním impulzov o frekvencii 1200 Hz dátami s obvodu USART. Generátorom kmitočtu 1200 Hz je obvod MHB 9500, tento kmitočť je súčasne hodinami pre USART (TxC). Takto zmodulovaná dátová zmes je nahraná cez konektor na pásku MG. Späťne signál z pásky MG sa najprv amplitúdovo upraví v obvodoch MAA 748 a A 301 D. Takto sa získava už číslicový signál, ktorý slúži ako vstupný signál na dekódovanie dát a hodín pre USART. Z každej hrany signálu sa za pomoci obvodu 7486 generujú krátke impulzy, ktoré spúšťajú monostabilný obvod 74121 so striedou výstupného signálu 3/4. Tento výstupný signál slúži ako hodiny (RxC) na vzorkovanie v USART-e. Vstupné dáta (RxD) sú ešte spracované klopným obvodom MH 7474, ktorý má za úlohu ich upraviť tak, ako keď išli z USART-u pri vysielaní. Zapojenie konektora K7 - konektor pre MG, je na obr. 11.



Obr. 11 ZAPOJENIE KONEKTORA PRE MG

#### 1.5 ČASOVAČ

Je realizovaný obvodom 8253 a generátorom sekundových impulzov MHB 1116. Obvod 8253 obsahuje tri samostatné časovače software prístupné. Jeden z nich (TIMER 0) je úplne k dispozícii užívateľovi - prístupný je cez aplikačný konektor.

Druhý (TIMER 1) má zabudované vstupné hodiny  $\emptyset 2$  (TTL) a užívateľ má k dispozícii len výstup časovača s jeho ovládaním. Tretí časovač (TIMER 2) má sekundové prvky na vstupe CLK a možnosť vonkajšieho ovládania. Používa sa ako hodiny reálneho času.

#### 1.6 VSTUPNO-VÝSTUPNÝ APLIKAČNÝ KONEKTOR

Pre užívateľa je vyvedená časť lokálnej zbernice mikropočítača, ktorá umožňuje pripojiť rôzne interfaceové obvody. Všetky signály sú oddelené pomocou obvodov 8286 a 74 $\emptyset$ 4. Z napájacích napätí je vyvedené +5V (max. do odberu 0,3A), +12V a -5V. Ostatné linky konektora K2 sú obsadené signálmi patriacimi interface pre časovač. Popis a rozmiestnenie pripájacích miest konektora K2 je na obr.12.

| Označ.prip.<br>miesta                            | č.prip.<br>miesta | č.prip.<br>miesta | Označ.prip.<br>miesta                        |
|--|-------------------|-------------------|--|
| GND  | 1                 | 30                | +12V   |
| BA6  | 3                 | 4                 | BA3  |
| BA2  | 5                 | 6                 | adresy BA5                                   |
| BA4  | 7                 | 8                 | BA $\emptyset$                               |
| BA1  | 9                 | 10                | BA7  |
| GATE 1-ovládanie<br>TIMER 1                      | 11                | 12                | OUT $\emptyset$ -výstup<br>TIMER $\emptyset$ |
| GATE $\emptyset$ -ovládanie<br>TIMER $\emptyset$ | 13                | 14                | OUT 1-výstup<br>TIMER 1                      |
| INT - Interrupt                                  | 15                | 16                | CLK $\emptyset$ hodiny<br>TIMER $\emptyset$  |
| I/OR   | 17                | 18                | DB5  |
| RESET  | 19                | 20                | DB6  |
| $\emptyset 2$ (TTL)                              | 21                | 22                | DB7  |
| I/OW   | 23                | 24                | dáta DB3                                     |
| DB1  | 25                | 26                | DB2  |
| DB $\emptyset$                                   | 27                | 28                | DB4  |
| -5V  | 2                 | 29                | +5V  |

Obr.12 PRIPÁJACIE MIESTA KONEKTORA K2



## 2. PROGRAMOVANIE I/O KANÁLOV

V programovacom jazyku BASIC - G existujú samostatné príkazy na ovládanie vstupno-výstupných kanálov. Všeobecný povel na výstup údajov je OUTPUT, na vstup ENTER. Každý kanál (GPIO, HP-IB, V.24, TIMER) možno naprogramovať na určitý režim a potom ho využívať prostredníctvom spomenutých príkazov (OUTPUT a ENTER). V prvom rade je potrebné špecifikovať správny kanál. Počet kanálov je 8 s adresou  $\emptyset \div 7$ .

Ďalej špecifikujeme konkrétny výstupný register. Pripájacie miesta pre každý kanál sú uvedené v predchádzajúcej časti. Okrem týchto dvoch základných príkazov existujú ešte dopĺňajúce príkazy (funkcie) na prácu I/O. Sú to:

| PRÍKAZ (FUNKCIA) | VÝZNAM                                  |
|------------------|---|
| CONTROL          | ovládanie príslušného I/O kanálu        |
| STATUS           | funkcia priraďujúca hodnotu I/O kanálu  |
| BIT              | funkcia na testovanie určitého bitu     |
| IN               | funkcia na načítanie hodnoty z I/O kan. |
| OUT              | zápis do I/O kanálu                     |

Obr.13

### 2.1 POVELY NA VÝSTUP ÚDAJOV

#### OUTPUT

Syntax: OUTPUT  $k$ , zoznam výrazov,

kde:  $k$  je číslo kanálu v rozsahu  $\emptyset \div 7$

$r$  je číslo výstupného registra v rozsahu  $\emptyset \div 255$  a závisí od druhu vstupného kanála.

Zoznam výrazov reprezentuje množinu možných výrazov, ktoré sú definované pri príkaze PRINT.

Druhy kanálov.

V nasledujúcej tabuľke sú zahrnuté kanály, ktoré sú v interpretácii BASIC-G. Užívateľ môže definovať rozšírenie niektorých existujúcich kanálov premodifikovaním príslušných pamäťových buniek v tabuľke skokov.

| Adresa tab.<br>[HEX] |      | Druh kanálu    | Adresa exe-<br>kutívy [HEX] | Poznámka      |
|----------------------|------|----------------|-----------------------------|---------------|
| 0                    | 2088 | -              | FFFF                        | neobsadený    |
| 1                    | 208A | sériový (V.24) | 2215                        | spoločný s MG |
| 2                    | 208C | -              | FFFF                        | neobsadený    |
| 3                    | 208E | -              | FFFF                        | neobsadený    |
| 4                    | 2090 | paralel.(GPI ) | 209F                        | paralelný     |
| 5                    | 2092 | -              | -                           | neobsadený    |
| 6                    | 2094 | -              | -                           | neobsadený    |
| 7                    | 2096 | IMS-2 (HPIB)   | 2008                        | IMS-2         |

Obr.14 TABUĽKA KANÁLOV

PRÍKLAD: Ak by sme chceli umiestniť na kanál č.2 napr. GPIO (pozorovateľný I/O kanál), je potrebné zo stránky hardware pripojiť pripájacie miesto CS obvodu 8255 na "selekt". dekódera I/O (pozri adresovú tabuľku

|    |    |    |    |    |    |    |    |
|----|----|----|----|----|----|----|----|
| A7 | A6 | A5 | A4 | A3 | A2 | A1 | A0 |
|----|----|----|----|----|----|----|----|

k dekóderu I/O

A7=0; A3=A2=A1

a zo stránky software je potrebné vložiť na pamäťovú bunku 208CH nižšiu hodnotu vektora exekutívy GPIO, teda 9F, na nasledujúcu bunku vyššiu hodnotu - 20.  
Obdobne to platí i na rozšírenie ďalších kanálov.

Je potrebné upozorniť užívateľa, že príslušný kanál je aktívny iba po jeho inicializácii. Neplatí to pre kanál 7, ktorý sa pred použitím nemusí programovať. Je automaticky nastavený vlastnými procedúrami.

Napríklad: Ak máte tlačiareň s adresou 01, môžete vytlačiť text príkazom:

OUTPUT 701, "TEXT"

alebo pre výpis "listingu" programu: LIST # 701;



### 2.1.1 SÉRIOVÝ VÝSTUPNÝ KANÁL

Syntax: OUTPUT 1; zoznam výrazov.

Tento kanál je realizovaný obvodom USART MHB 8251, ktorý vzhľadom na to, že je prepínateľný s interfaceom kazetového magnetofónu, má pevne priradenú vzorkovaciu frekvenciu 1200 Hz (prípadne 2400 Hz), čo je potrebné mať na pamäti pri voľbe prenosovej rýchlosti. Sériový kanál je programovateľný, čo znamená, že pred príkazom OUTPUT je potrebné zvoliť jeho pracovný režim.

Voľba pracovného režimu sériového kanála sa vykoná vyslaním dvoch údajov. Prvý reprezentuje nastavenie režimu USART a je odlišný čo sa týka asynchronného alebo synchronného prenosu dát. Z nasledujúcich dvoch tabuliek možno zostaviť potrebný kód - decimálne číslo, ktoré sa vypočíta ako súčet z povinných parametrov.

| počet stop bitov |     |     | hodnota parity - ENABLE |        |             | počet znakov |   |   |    | rýchlosť prenosu [BAUD] |     |     |          |
|------------------|-----|-----|-------------------------|--------|-------------|--------------|---|---|----|-------------------------|-----|-----|----------|
| 1                | 1,5 | 2   | párny                   | nepár. | uvoľ. parit | 5            | 6 | 7 | 8  | 1x                      | 16x | 64x | Syn. MOD |
| 64               | 128 | 192 | 32                      | ∅      | 16          | ∅            | 4 | 8 | 12 | 1                       | 2   | 3   | ∅        |

Obr.15 ASYNCHRÓNNY REŽIM

| znak Sync. |     | Identifikácia Sync. |                  | parita |         |            | počet znakov |   |   |    |
|------------|-----|---------------------|------------------|--------|---------|------------|--------------|---|---|----|
| jeden      | dva | Syndet je INPUT     | Syndet je OUTPUT | pár.   | ne-pár. | uvoľne-nie | 5            | 6 | 7 | 8  |
| 128        | ∅   | 64                  | ∅                | 32     | ∅       | 16         | ∅            | 4 | 8 | 12 |

Obr.16 SYNCHRONNÝ REŽIM

**Príklad:** Ak chcete naprogramovať sériový kanál ako asynchrónny, počet stop bitov má byť 1, bez parity, počet znakov - 7 bitov a prenosová rýchlosť 1200 bit/sek. (1200 Hz). Potom z tabuľky pre asynchrónny režim dostanete nasledovné údaje:

$$64 + 0 + 8 + 1 = 73$$

toto číslo bude vyslané na USART príkazom CONTROL ako prvé. Ďalej USART vyžaduje riadiacu inštrukciu, ktorú musíte zostaviť tak, aby to vyhovovalo vašim požiadavkám. Potrebný kód zostavíte podobne z nasledujúcej tabuľky.

|     |   |
|-----|---|
| 128 | tzv. HUNT MODE<br>len pre SYNC.MODE                       |
| 64  | INTERNAL RESET  |
| 32  | "REQUEST"<br>vysielanie $\overline{\text{RST}} = \log 0$  |
| 16  | RESET ERROR   |
| 8   | SBRK - BREAK<br>Normálne = 0<br>$8 = \text{TxD} = \log 0$ |
| 4   | Príjem uvoľniť  |
| 2   | Terminál "READY"<br>$\overline{\text{DTR}} = \log 0$      |
| 1   | Vysielanie uvoľniť  |

Obr.17

Stavové slovo zo sériového kanála je osembitové a obsahuje tieto údaje:

|     |        |    |    |    |         |        |        |
|-----|--------|----|----|----|---------|--------|--------|
| DSR | SYNDET | FE | OE | PE | TXEMPTY | Rx RDY | Tx RDY |
|-----|--------|----|----|----|---------|--------|--------|



kde: DSR - DATA SET READY - indikuje úroveň log 0, je pripravený módem.

SYNDET - iba pre asynchrónny režim, indikuje, že USART našiel SYNC znak.

FE - iba pre asynchrónny režim, indikuje chýbajúci STOP bit. Nuluje sa vyslaním RESET ERROR v riadiacej inštrukcii USART.

OE - OVERRUN ERROR - znamená, že systém neprevzal v čas dáta a dochádza k preplneniu USARTu. Nuluje sa tak isto ako príznak FE.

PE - PARITY ERROR - chyba v parite. Nuluje sa tak isto, ako príznak FE.

Tx EMPTY - TRANSMITTER EMPTY - používa sa na indikáciu konca prenosu pre módem.

Rx RDY - RECEIVER READY - indikuje, že si systém môže prevziať prijaté dáta. Najčastejšie sa používa, ako "HANDSHAKE".

Tx RDY - TRANSMITTER READY - indikuje, že systém môže poslať ďalšie údaje (dáta) na vysielanie. Najčastejšie sa používa ako "HANDSHAKE".

Na zistenie stavového slova zo sériového kanála sa použije príkaz STATUS.

Príkaz pre OUTPUT:

100 OUTPUT 1; A\$: vyslanie obsahu reťazca A\$ na sériový kanál.

## 2.1.2 PARALELNÝ VÝSTUPNÝ KANÁL (GPIO)

Je realizovaný programovateľným paralelným stykovým obvodom MHB 8255 A.

Syntax: OUTPUT 4r; zoznam výrazov

kde: r znamená konkrétny výstupný register a môže byť v rozsahu 00 ÷ 07. Kanál GPIO je rozdelený na dve rovnocenné skupiny.

Skupina 0 - v tejto skupine sa nachádza PORT A a časť PORT-u C s vyššími linkami.

Skupina 1 - sem patrí PORT B a druhá (dolná časť) PORT-u C.

Jednotlivé skupiny a porty A,B,C môžu byť programované v rôznych režimoch (jednoduchý) vstup/výstup; (obojsmerný) vstup/výstup s podporou "HANDSHAKE"..  
Výber jednotlivkej skupiny alebo portu je daný hodnotou r, pre ktorú platí nasledujúca tabuľka :

|           | PORT     | REŽIM                                 | r  | CONTROL CODE |
|-----------|----------|---------------------------------------|----|--------------|
| SKUPINA 0 | A        | jednoduchý výstup (vstup)             | 00 | 128          |
|           | C(vyšší) | jednoduchý výstup (vstup)             | 02 | 128          |
|           | A        | výstup (vstup) s podporou handshake   | 03 | 160          |
|           | A        | obojsmerný vstup (výstup) + handshake | 05 | 192          |
| SKUPINA 1 | B        | jednoduchý výstup (vstup)             | 01 | 128          |
|           | C(nižší) | jednoduchý výstup (vstup)             | 02 | 128          |
|           | B        | výstup (vstup) s podporou handshake   | 04 | 132          |

Obr. 18

Kódy pre r = 06 a 07 nie sú obsadené a užívateľ im môže priradiť špeciálnu výstupnú (alebo vstupnú) exekutívu tak, že uloží príslušný vektor exekutívy na nasledujúcu pamäťovú bunku (najskôr sa ukladá nižšia hodnota).

| výstup | r  | adresa tabuľky [HEX] |
|--------|----|----------------------|
|        | 06 | 20E6, 20E7           |
|        | 07 | 20E9, 20EA           |

| vstup | r  | adresa tabuľky [HEX] |
|-------|----|----------------------|
|       | 06 | 21F8, 21F9           |
|       | 07 | 21FB, 21FC           |

Takto je možné napr. rozšíriť za pomoci obidvoch skupín styk pre BCD výstup a pod.

Režim jednotlivých skupín či portov je potrebné naprogramovať príkazom CONTROL. Tabuľka na obr. 18 obsahuje kombináciu skupín 0,1 keď sú všetky porty vo funkcii výstupu, ale je



možné voliť aj iné kombinácie. Riadiace slovo - kód pre voľbu kanálu GPIO je dané v nasledujúcej tabuľke a získava sa sčítaním jednotlivých vyžadovaných úkonov kanála.

|     |                                      |                          |           |
|-----|--------------------------------------|--------------------------|-----------|
| 128 | Nastavenie riadiaceho slova          |                          |           |
| 0   | jednoduchý (MOD = 0 )                | REŽIM                    | SKUPINA 0 |
| 32  | s podporou handshake (MOD = 1)       |                          |           |
| 64  | obojsmerný I/O + handshake (MOD = 2) |                          |           |
| 16  | vstup                                | PORT A                   |           |
| 0   | výstup                               |                          |           |
| 8   | vstup                                | PORT C<br>VYŠŠIE<br>BITY |           |
| 0   | výstup                               |                          |           |
| 0   | jednoduchý (MOD = 0 )                | REŽIM                    | SKUPINA 1 |
| 4   | s podporou (MOD = 1)                 |                          |           |
| 2   | vstup                                | PORT B                   |           |
| 0   | výstup                               |                          |           |
| 1   | vstup                                | PORT C<br>NÍŽŠIE<br>BITY |           |
| 0   | výstup                               |                          |           |

Obr.19

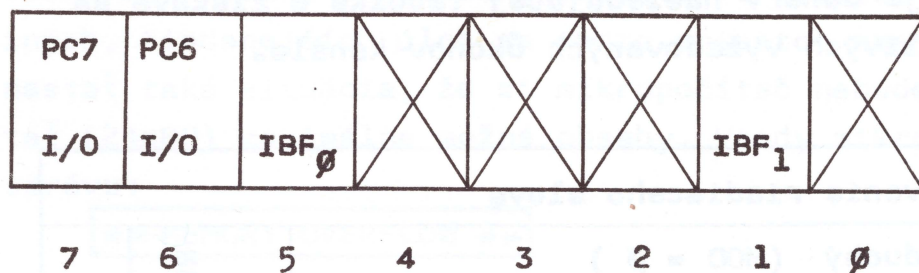
**Príklad:** Treba naprogramovať kanál GPIO tak, aby skupina 0 pracovala v režime výstup s podporou dvojdrotového handshake, a v skupine 1 port B ako vstupný.

Riadiace slovo =  $128 + 32 + 0 + 0 + 2 = 162$

Tento kód bude zapísaný príkazom CONTROL do riadiaceho registra kanála GPIO.

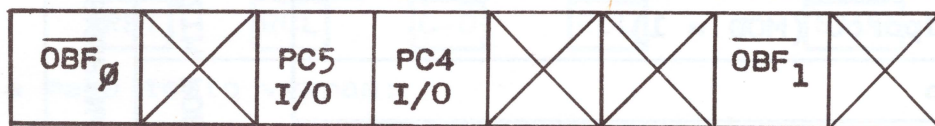
Kanál GPIO poskytne v režime (MOD = 1 a MOD = 2) stavové slovo, ktoré je možné programovo testovať na požadované hodnoty príkazom STATUS a BIT.

Stavové slová pre režim s podporou handshake.



Obr. 20 - VSTUPNÁ KONFIGURÁCIA

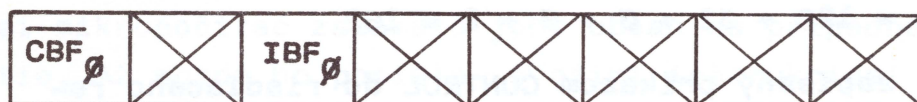
kde: PC7 je voľná linka pre vstup/výstup,  
 PC6 je voľná linka pre vstup/výstup,  
 IBF<sub>0</sub> je indikácia, že vstupný register portu A je naplnený,  
 IBF<sub>1</sub> je indikácia, že je naplnený vstupný register portu B.



Obr. 21 - VÝSTUPNÁ KONFIGURÁCIA

kde:  $\overline{\text{OBF}}_0$  indikuje, že je naplnený výstupný register portu A,  
 PC5, PC4 - voľné linky na vstup/výstup,  
 $\overline{\text{OBF}}_1$  indikuje, že je naplnený výstupný register portu B.

Stavové slovo pre obojsmerný I/O režim.



Obr. 22

kde:  $\overline{\text{OBF}}_0$  a IBF<sub>0</sub> majú predchádzajúci význam.

Príklad: 500 OUTPUT 403 ; "COMPUTER"



Reťazec znakov bude vyslaný na kanál GPIO a komunikácia bude s podporou handshake.

### 2.1.3 VÝSTUPNÝ KANÁL IMS - 2 (HPIB)

Syntax: OUTPUT 7 adr; zoznam výrazov,

kde: adr, - znamená adresu zariadenia na IMS - 2,

Použitie: výstup výrazov na zariadenie so štandardnou zbernicou IMS - 2..

Príklad: 100 OUTPUT 701; SIN(X)

Hodnota výrazu SIN(X) bude vyslaná na zariadenie s adresou 01, ktoré je nastavené ako "LISTENER".

Poznámka: Tento kanál netreba inicializovať, ale takisto možno k nemu pristupovať s príkazom CONTROL, STATUS.

### CONTROL

Syntax: CONTROL k, adr; výraz [výraz ....]

kde: k je číslo I/O;

adr, je adresa registra, do ktorého sa má zapísať výraz alebo postupnosť výrazov.

Výraz môže byť zadaný explicitne alebo implicitne a predstavuje celočíselnú hodnotu v rozsahu  $0 \div 255$ .

V nasledujúcej tabuľke sú uvedené všetky možné adresy kanálov prístupné týmto príkazom:

| kanál | register (PORT)    | adr.   |
|-------|--------------------|--------|
| 1     | dátový<br>riadiaci | 0<br>1 |
| 4     | A                  | 0      |
|       | B                  | 1      |
|       | C                  | 2      |
|       | riadiaci           | 3      |

| kanál | register (PORT) | adr. |
|-------|-----------------|------|
| 7     | A               | 0    |
|       | B               | 1    |
|       | C               | 2    |
|       | riadiaci        | 3    |

Obr. 23

Použitie: príkaz CONTROL je možné používať na jednoduché ovládanie registrov príslušného kanála vrátane jeho inicializácie.

Príklad: 10 CONTROL 4,3; 160  
20 OUTPUT 403 ; A\$

Príklad znázorňuje inicializáciu kanála 4 a to skupiny 0 na výstup údajov (dát) s podporou handshake (pozri Obr.18). V riadku 20 sa vykoná prenos dát (reťazec A\$) na kanál 4, skupina 0, port A.

40 CONTROL 1,1; 64 .... vnútorné nulovanie  
50 CONTROL 1,1; 73,37 ..... inicializácia

Inicializácia sériového kanála - 1 podľa obr.16,17.

100 CONTROL 4,0; 255

Nastavenie výstupného registra kanálu 4 port A na hodnotu 255(FFH).

## 2.2 POVELY PRE VSTUP DÁT

### ENTER

Syntax: ENTER kr; zoznam premenných

kde: k je číslo kanálu

r je konkrétny register v kanáli

Zoznam premenných je množina premenných definovaná rovnako, ako pri príkaze INPUT.



"k" a "r" sa určujú tým istým spôsobom ako pri príkaze OUTPUT. Pre užívateľa je potrebné vedieť, že max. počet prijatých znakov v rámci jednej premennej môže byť 80 a tento blok musí byť ukončený znakom návrat vozu (CR) a nový riadok (LF); v opačnom prípade môže dôjsť ku zničeniu celého obsahu pamäte.

Použitie : na vstup údajov zo zvoleného kanála do zoznamu premenných.

Poznámka : príkaz ENTER je možné použiť iba po inicializácii kanála !

Príklad : 1Ø CONTROL 4,3; 128 + Ø + 16 + Ø + 2 + Ø

Nastavenie kanála 4 na vstup údajov do portu A v móde Ø a v skupine 1 nastavenie portu B ako vstupný v jednoduchom režime (mód Ø).

2Ø ENTER 4ØØ; XØ vstup dát do XØ cez register ØØ port A

5Ø ENTER 4Ø1; X vstup dát do X cez register Ø1 port B

## STATUS

Syntax: STATUS k, adr.

kde: adr, a k majú rovnaký význam ako pri povelu CONTROL (kanál - adresa registra kanála).

Použitie: na jednoduché priradenie hodnoty obsahu zvoleného registra z príslušného kanála do zvolenej premennej. Príkaz sa môže použiť po inicializácii príslušného kanála príkazom CONTROL.

Príklad: 1Ø CONTROL 4,3; 146

2Ø A = STATUS 4,Ø

4Ø PRINT BIT (STATUS 4,Ø)

inicializácia kanála 4

priradenie obsahu portu A premennej A

7 tlačenie hodnoty siedmeho bitu portu A kanála 4.

### PRÁCA S MAGNETOFÓNOM

Interpretér BASIC-G má päť príkazov na prácu s magnetofónom. Stratégia zápisu, čítanie a vyhľadávanie v čítane kontroly správnosti zápisu je rovnaká ako pri operačnom systéme. Odlišuje sa iba tým, že BASIC-G Vám svojimi príkazmi uľahčuje túto činnosť.

Ide o tieto príkazy:

| PRÍKAZ | VÝZNAM  |
|--------|---|
| SAVE   | zápis programu na pásku kazetového magnetofónu (MG) |
| LOAD   | čítanie programu z MG pásky                         |
| CHECK  | kontrola a vyhľadávanie čítaného súboru na MG páske |
| DSAVE  | zápis zvoleného poľa na MG pásku                    |
| DLOAD  | čítanie dát do poľa z MG pásky                      |

Zameriame sa na posledné dva príkazy, ktoré umožňujú dátové spracovanie na magnetofónovú pásku a z nej.

Zásadne možno zapísať alebo načítať iba obsah dátového alebo reťazcového poľa, ktoré bolo v programe deklarované s patričnou dimenziou, ktorá sa pri čítaní ešte kontroluje s dimenziou nahratého poľa.

Postup pri programovaní je takýto:

#### ZÁPIS DÁT:

dimenzovanie príslušného poľa, napr.: DIM A (10 10)

príkaz na zápis v patričnej sekvencii, napr.,

DSAVE 4; A (0) " DATA "

(odporúčame ešte predtým vydať správu, aby bolo možné nastaviť MG)

? "ZAPNI MG"

po ukončení príkazu na zápis tak isto vydať správu

? "MG STOP"

#### ČÍTANIE DÁT:

dimenzovať príslušné pole, napr.,

DIM B (1 00)

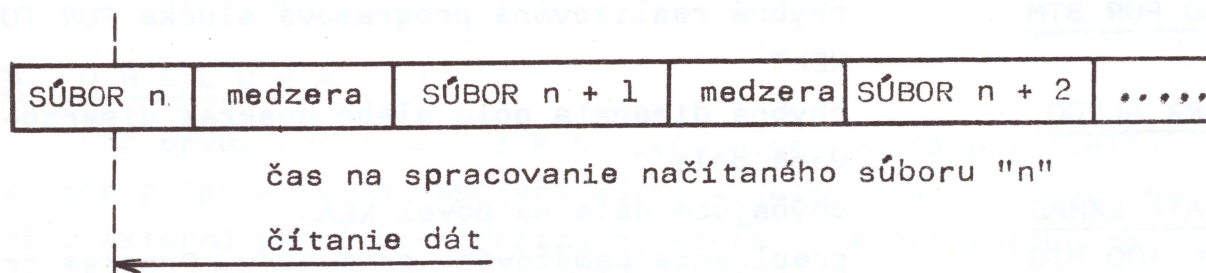


v patričnej sekvencii vložiť príkaz na čítanie dát, napr.

DLOAD 4; B (Ø)

Iste ste si všimli, že pri výkonných príkazoch je dané pole napísané pre nultý prvok. Je to potrebné z toho dôvodu, že sa tým spoznáva, o akú dimenziu ide. V opačnom prípade Vám bude mikropočítač oznamovať chyby.

Tieto príkazy môžete využiť na triedenie súborov alebo na ich spracovanie. Číslo súboru (záznamu) môžete zadať aj implicitne a použiť cyklus na načítanie a spracovanie polí. Túto skutočnosť - využitie plynulého chodu magnetofónu so simultánnou spolupracou s programom BASIC - treba zvážiť už pri nahrávaní jednotlivých súborov tak, ako ukazuje obrázok.



Úloha: a/ Napíšte program na uloženie troch súborov, v ktorých bude jednorozmerné pole obsahujúce tri rozličné texty.

b/ Napíšte program na vyhľadanie správneho textu z predchádzajúcich troch súborov a zobrazte daný text.

### HLÁSENIE CHÝB

V prípade chyby Vám mikropočítač podá správu do dialógového riadka a oznámi, o aký typ chyby ide. Výpis chyby môže mať dve formy, ktoré závisia od toho, či ide o priamy alebo programový režim,

\*\*\* ERROR MESSAGE \*\*\*

priamy režim

\*\*\* ERROR MESSAGE IN LINE \*\*\*

v idúcom programe

Všetky správy sú v anglických skratkách a obsahujú 1Ø znakov.

SYNTAX ERR

chybne zadaný príkaz

FNC. PARAM.

parameter funkcie nesprávny

|                    |   |
|--------------------|---|
| <u>SUBSCR.RNG.</u> | výraz subscript je mimo definovaného rozsahu, napr. príkazom DIM  |
| <u>ONLY IN PG</u>  | príkaz možno použiť iba v programovom kóde DEF, INPUT   |
| <u>OVERFLOW</u>    | prekročený rozsah výpočtu   |
| <u>DV BY ZERO</u>  | FLOATING POINT NUMBER   |
| <u>TYPE COVN</u>   | delenie nulou   |
| <u>CANT CONT</u>   | chybný prevod, napr. prevod medzi numeric-kou premennou a reťazcovou premennou  |
| <u>NO FOR STM</u>  | chybný povel CONT na pokračovanie v interpre-tovaní programu, Pred týmto povelom bol pou-žitý povel, ktorý spôsobil zmenu v konfigu-rácii programu. |
| <u>ARR ALLOG</u>   | chybne realizovaná programová slučka FOR TO NEXT  |
| <u>DATA EXHAU</u>  | chybná dimenzia poľa alebo dvakrát dimenzo-vané pole  |
| <u>PG TOO BIG</u>  | chýbajúce dáta na povel READ  |
| <u>STRING LONG</u> | preplnenie pamäťového priestoru. Program tre-ba skompresovať vynechaním medzier a použi-tím viacnásobných príkazov na jeden programo-vý riadok.     |
| <u>NO STR. SPC</u> | dĺžka reťazca presahujúca 255 znakov  |
| <u>NUMB. NONEX</u> | žiadne voľné miesto pre string  |
| <u>STR. ALGRT</u>  | neexistujúci programový riadok GOTO, GOSUB, THEN  |
| <u>STOP</u>        | reťazcový výraz je v tomto príkaze príliš dl-hý alebo príliš zložitý. Treba ho rozdeliť na dva alebo na viacero jednoduchých príka-zov.             |
| <u>RETURN ERR</u>  | zastavený program   |
| <u>INPUT ERR</u>   | použitý príkaz RETURN bez korešpondujúceho príkazu GOSUB  |
| <u>FILLE BOUND</u> | chyba pri vkladaní dát, napr. nečíselný znak do číselnej premennej  |
|                    | prekročený maximálny počet záznamu MG   |



FILLE ERROR

FIELD LOST

FILLE SMALL

chyba pri prenose dát  
prekročený žiadaný počet vstupných dát,  
napr. pri INPUT  
dátový súbor z pásky MG nekorešponduje pries-  
torovo s dimenzovaním

ZOZNAM PRÍKAZOV A FUNKCIÍ V BASIC-G

VÝZNAM

TVAR

|                       |  |   |                                      |
|-----------------------|--|---|--------------------------------------|
| EDITOVACIE            | LIST<br>LIST n<br>LLIST<br>LLIST n   | AUTO<br>AUTO n,m<br>NEW   | SAVE n<br>LOAD n<br>ROM n<br>CHECK n |
| RIADIACE              | RUN<br>RUN n<br>GOTO n<br>CONT   | MONIT   |                                      |
| FUNKCIE               | SIN (X)<br>COS (X)<br>TAN (X)<br>ATN (X)<br>LOG (X)<br>EXP (X)<br>SQR (X)<br>ABS (X) | INP (I)<br>RND (X)<br>SGN (X)<br>FRE (X)<br>SPC (I)<br>TAB (I)<br>BIT A,B<br>STATUS A,B | INKEY<br>PEEK (I)<br>USR (I)         |
| REŤAZCOVÉ<br>FUNKCIE  | ASC (X\$)<br>FRE (X\$)<br>LEFT\$ (X\$,I)   | RIGHT\$ (X\$,I)<br>MID\$ (X\$,I)<br>LEN (X\$)   | STR \$(X)<br>VAL (X\$)               |
| ŠTANDARDNÉ<br>PRÍKAZY | BEEP<br>DEF FNC<br>DIM<br>DSAVE<br>DLOAD<br>FOR-TO-NEXT<br>STEP                      | PRINT<br>GOTO<br>IF - THEN<br>LET<br>ON<br>PAUSE  | STOP<br>REM<br>CLEAR<br>NULL (I)     |

|                             |                                |                        |                    |
|-----------------------------|--------------------------------|------------------------|--------------------|
|                             | DISP<br>GCLEAR<br>GOSUB-RETURN | READ-DATA              |                    |
| GRAFICKÉ<br>PRÍKAZY         | SCALE<br>MOVE<br>AXES          | PLOT<br>BMOVE<br>BPLOT | LABEL<br>FILL      |
| VSTUPNO-VÝSTUPNÉ<br>PRÍKAZY | OUTPUT<br>ENTER                | CONTROL<br>STATUS A,B  | LIST #             |
| PRÍKAZY PRE<br>SYSTÉM 8080  | CODE<br>ROM<br>USR             | PEEK<br>POKE           | INP<br>OUT<br>WAIT |

# I. TECHNICKÉ ÚDAJE

**Napájacie napätia:**

|               |   |
|---------------|---|
| +5V $\pm$ 5%  | zvlnenie max. 50mV<br>prúdový odber max. 2A   |
| +12V $\pm$ 5% | zvlnenie max. 50mV<br>prúdový odber max. 1A   |
| -5V $\pm$ 5%  | zvlnenie max. 50mV<br>prúdový odber max. 0,4A |

**Celkový príkon počítača:** max. 35 VA

**Kapacita operačnej pamäte:** 48 kB

**Kapacita pevnej pamäte (Monitor):** 4 kB

**Kapacita pevnej pamäte (Interpreter Basic - G):** 9 kB

**Rozmery mikropočítača:** 313 x 285 x 65 mm

**Hmotnosť mikropočítača:** 1915 g



## J. ZÁVER

V doteraz uvedených článkoch ste sa oboznámili s najdôležitejšími prácami v obidvoch programových módoch, v operačnom systéme a v móde vyššieho programovacieho jazyka BASIC - G..

Nie je to všetko, ale na začiatok práce s mikropočítačom to postačí. Po zvládnutí zadaných úloh máte predpoklad dobre pracovať s mikropočítačom. Pre využitie vlastností a operačných možností mikropočítača je dôležité oboznámenie sa s vlastným popisom jazyka BASIC - G, ktorý je popísaný v príslušnej literatúre..

T E S L A   Bratislava k.p.

## DODÁVANÉ PRÍSLUŠENSTVO

- 1 ks návod na obsluhu a použitie
- 1 ks záručný list
- 1 ks baliaci list
- 1 ks pripájacia šnúra - vf 1 PF 897 14
- ~~1 ks pripájacia šnúra - napájanie 1 PF 897 15~~

## ODPORÚČANÉ PRÍSLUŠENSTVO

TV prijímač - vf vstup na 9. kanáli  
TV monitor,  
štandardný kazetový magnetofón (napr. K 10 MIRA ),  
všetky meracie a registračné zariadenia, ktoré obsahujú  
interfejs IMS 2,  
ďalšie zariadenia, ktoré využívajú paralelný alebo sériový  
prenos údajov.

## Operačný systém

Operačný systém v rozsahu 4 Kbytov je uložený v pamätiach ROM typu ROM 256K a začína sa na adrese 0000 H, ukončený je špinou k dispozícii pracovnej pamäte ROM 256K. Práca stránok pamäti ROM sa adresy 0000 H.

Operačný systém zabezpečuje organizáciu obrábkovania vstupných údajov, ktoré sa začínajú od adresy 0000 H, obrábkovanie vstupných údajov pre ROM, výpis znakov ASCII a organizáciu obrábkovania výstupných údajov. Rozdelenie pracovnej pamäte počítača ROM - 256 K je na obr. 2. 3.



## O B S A H

|  |     |
|--|-----|
| A. ÚVOD - ARCHITEKTÚRA MIKROPOČÍTAČA .....                 | 1   |
| B. OBSLUHA .....   | 7   |
| C. ZOBRAZOVANIE .....                                      | 15  |
| D. MONITOR .....   | 16  |
| E. PODPROGRAMY MODULU MONITOR .....                        | 26  |
| F. PROGRAMOVÉ MODULY EDIT, KLÁV, PRTOUT, INPÓL, OSIO ..... | 29  |
| G. PROGRAMOVANIE V JAZYKU BASIC - G .....                  | 35  |
| H. PRÍKAZY NA VSTUP/VÝSTUP .....                           | 79  |
| I. TECHNICKÉ ÚDAJE .....                                   | 100 |
| J. ZÁVER .....   | 101 |

---

Vydal: k.p. Tesla Bratislava v náklade 10 000 ks. Júl 1985  
Vytlačili: Západoslovenské tlačiarne, n.p. Bratislava,  
závod 50, Partizánske