Tiny BASIC

pro

TinyBasRV

Programové vybavení ver. 1.0

Mikropočítač TinyBasRV obsahuje interpreter jazyka Tiny BASIC s některými omezeními a naopak s řadou vylepšení. Interpreter zajišťuje uživatelské rozhraní (komunikaci s obsluhou) a také vykonávání uloženého programu.
Po zapnutí mikropočítače se na VGA displeji objeví hlášení „Tiny BASIC x.x“ udávající aktuální verzi interpreteru. Tento dokument by měl mít uvedenou stejnou verzi, jakou hlásí po startu počítač. Mikropočítač současně otestuje dostupný hardware a případně oznámí, že DISK f a expander PCF8575 na sběrnici I2C není dostupný a tím jsou omezeny některé funkce. Pro připojování disku „f“ nebo expanderu postupujte podle instrukcí „Co dělat, když...“ na konci tohoto návodu.

Při komunikaci s uživatelem očekává mikropočítač zadání příkazu na řádku ve dvou možných formátech:

STATEMENT (; STATEMENT) (; STATEMENT)

nebo

NUM STATEMENT (; STATEMENT) (; STATEMENT)

kde STATEMENT představuje příkaz jazyka Tiny BASIC, který je možno vykonat. Pokud začíná řádek příkazem, provede se ihned. Je-li na začátku řádku uvedeno číslo NUM (v rozsahu 0 až 127), příkaz se neprovádí, ale uloží se do paměti a stane se jedním z řádků programu, který čeká na spuštění.

Příklad:

PRINT 5+46

Vypíše ihned na řádek výsledek – tj. 51
Proti tomu

26 PRINT 5+46

neudělá zdánlivě nic. Tento řádek se uloží na pozici řádku číslo 26 do paměti programu v aktuálním „souboru“ (viz příkaz FILE). Do doby než bude program spuštěný, zůstává nečinně zapsaný v paměti. Teprve po spuštění programu se příkaz provede v okamžiku, kdy přijde na řadu zpracování řádku číslo 26.

Na řádku mohou být jeden, dva, nebo tři příkazy oddělené středníkem, které se vykonají postupně zleva. Některé příkazy např. REM, GOTO, IF způsobí, že další následující příkaz (nebo příkazy) na řádku se již neprovedou.

Psaní programu

Program nebo příkazy interpreteru se zadávají pomocí klávesnice na řádek do místa, kde bliká kurzor. Tisknutelné znaky se objevují přímo po zadání na obrazovce. Pokud je potřeba zadat více znaků „mezera“, je možné použít klávesu Tab, která se rozvine do 4 mezer. Případnou chybu je možné vrátit klávesou „BackSpace“. Posun na řádku pomocí šipek není možný, je pouze možné znaky postupně zpětně rušit.
Vymazání špatně napsaného řádku lze provést zadáním prázdného řádku. Na začátku řádku napíšeme pouze číslo rušeného řádku a pak zmáčkneme Enter. Oprava špatně zadaného řádku se provede automaticky po napsání nové správné verze řádku.
Příkazy jazyka Tiny BASIC se nevypisují postupně po jednotlivých znacích, ale vkládají se pomocí současného stisknutí 2 kláves. Příkazy se zadávají pomocí stisku jedné klávesy při současně stisknuté klávese „Levý CTRL“. Seznam povolených příkazů a jejich odpovídající klávesy jsou v souhrnné tabulce na poslední straně tohoto dokumentu.
Příkazy (STATEMENT) je možné zadat až 3 současně na jeden řádek. Většina příkazů očekává za

vlastním příkazem jeden nebo více parametrů ve specifickém přesně daném formátu. Seznam parametrů a jejich význam je uveden dále u popisu jednotlivých příkazů a také v tabulce na poslední straně. Hlavní parametry u příkazů jsou:

STRING
je textový řetězec uzavřený mezi uvozovky. V rámci řetězce mohou být libovolné tisknutelné znaky. Speciálním případem v rámci řetězce je kombinace dvou znaků - „\“ (zpětné lomítko) následované písmeny „n“, „t“ nebo „f“ – např. „\n“. Tyto kombinace způsobí:
n- přechod na nový řádek
t – tabulátor s posunem kurzoru na pozice dělitelné 8 (tj. 0,8,16,24)
f – vymazání obrazovky a návrat kurzoru vlevo nahoru

VAR
označuje proměnnou, do které se ukládají číselné hodnoty. Protože je použit 32bitový počítač, jsou také všechny proměnné 32bitové. Hodnoty uložené v nich mohou být v rozsahu -2 147 483 648 až 2 147 483 647. Uživatel má automaticky předdefinováno celkem 126 proměnných. Ty jsou rozděleny do dvou skupin. Buď mají své vlastní jméno a jsou označeny velkými písmeny anglické abecedy A až Z (celkem 26 proměnných).
Nebo tvoří indexované pole o velikosti 100 položek. K poli je možné přistupovat pomocí znaku „@“, za kterým bezprostředně následuje číslo indexu položky udané jednou nebo dvěma číslicemi, nebo písmeno označující proměnnou, kde je uložena hodnota indexu.

Příklady povolených jmen proměnných: A, .. Q, .. Z
Příklady jmen prvků pole: @0, .. @9, .. @10, .. @28, .. @99, @A .. @Z.

Při přímé indexaci pole číslicemi nedojde k překročení rozsahu pole, protože interpreter zpracuje pouze jednu či dvě číslice. Je-li však pro indexaci použita proměnná, může její obsah překročit velikost pole. Např. jestliže A obsahuje hodnotu 4786, pak odkaz na prvek pole @A způsobí hlášení chyby a ukončení programu.

EXPR
představuje matematický výraz s vyčíslitelnou hodnotou. Může jít přímo o číslo, obsah uložený v některé proměnné nebo algebraický výraz pro výpočet konkrétní hodnoty. Použitelné tvary výrazů popisuje matematika jazyka dále.

NUM
číslo přímo zadané pomocí těsně za sebou jdoucích dekadických číslic. Druhou možností je zadat číslo v binární formě. Binární číslo je uvedeno písmenem „b“, za kterým těsně za sebou následují číslice 0 nebo 1.

Matematika jazyka

Jazyk používá tzv. celočíselnou aritmetiku, není tedy možné zadávat desetinná čísla. Rozsah čísla může být od -2147483648 až do 2147483647. Čísla mimo toto rozmezí nebudou zpracována správně.
Výpočty zpracovávají 32bitová čísla v uvedeném rozsahu. Pro výpočty je možné používat operace sčítání, odčítání, násobení a dělení. Priorita operací (násobení a dělení mají přednost před sčítáním a odčítáním) je dodržena. Při stejné úrovni priority jsou výrazy zpracovávány zleva doprava.
Formálně lze matematiku zapsat:

EXPR = (|+|-)TERM ((+|-) TERM)^
kde TERM = FACT ((*|/) FACT)^
a FACT = VAR | NUM ..... viz výše.

Položky v závorkách jsou volitelné. Možnosti volby odděluje znak „|“. Závorky označené „^“ je možné opakovat až do konce řádku.
Příklady matematických výrazů (EXPR), které jsou povolené:

-5
A/3 + @23
10 + A*15 - @G/42*R
-114783648 – Q * R* S* T + 526 / C

Závorky není možno v rámci matematiky v programu používat. Pokud bychom potřebovali vypočítat např vzorec A=(B+C)/(D-E), je nutné použít rozdělení výpočtu do několika kroků – např.:

LET F=B+C; LET G=D-E; LET A=F/G

Příkazy jazyka

Při zadávání parametrů jednotlivých příkazů byl dále použit stejný formální popis, jako je popis výše uvedený u matematiky jazyka. Položky v závorkách jsou volitelné a závorky označené „^“ je možné opakovat až do konce řádku.

Příkaz: **PRINT**
Parametry: (STRING | EXPR) (,(STRING | EXPR))^
Funkce příkazu: Vypisuje na obrazovku od aktuální pozice údaje zadané v parametrech. STRING je vypsán v textové podobě bez uvozujících a zavírajících uvozovek. Výraz EXPR je vypsán jako číslo vyjádřené daným matematickým výrazem.

Příklady:

PRINT 5+3	vypíše na aktuální pozici obrazovky číslo 8
PRINT „Vysledek je „ , B	vypíše podle obsahu B např. Vysledek je 115
PRINT A, „\n“, B, „\n“, C	vypíše na řádky pod sebe postupně obsah proměnných A,B,C

Příkaz: **IF**
Parametry: EXPR RelOp EXPR
Funkce příkazu: Tento příkaz je těsně svázán s následujícím příkazem THEN, který musí bezprostředně následovat na stejném řádku. Tvoří jedinou na sebe vázanou dvojici příkazů a představují tak vlastně jeden dvojitý příkaz.

Příkaz: **THEN**
Parametry: STATEMENT
Funkce příkazu: Spolu s příkazem IF vytvářejí rozhodovací podmínku umožňující podmíněné zpracování jiného příkazu nebo větvení programu. Dvojice IF-THEN provede příkaz STATEMENT v parametru THEN pouze tehdy, je-li splněna relační podmínka EXPR RelOp EXPR v parametru u příkazu IF.
RelOp je porovnávací podmínkou mezi dvěma výrazy EXPR. Může být ve tvaru:

=	jsou oba výrazy stejné?
<	je první výraz menší než druhý?
<>	jsou výrazy rozdílné?
<=	je první výraz menší nebo roven druhému?
>	je první výraz větší než druhý?
><	jsou výrazy rozdílné?
>=	je první výraz větší nebo roven druhému?
&	je bitový logický součin výrazů nenulový?

Pokud je podmínka pravdivá, provede se příkaz STATEMENT, v opačném případě se tento příkaz neprovádí.
Pozor! Pokud je podmínka nepravdivá, nepokračuje se prováděním řádku za klíčovým slovem THEN a program přechází na další řádek programu. Tuto vlastnost lze využít při podmíněném provádění více příkazů současně. Při splněné podmínce se v provádění řádku pokračuje a protože na řádku mohou být až 3 příkazy, je možné po vyhodnocení jedné podmínky provést či neprovést až 3 příkazy.
Příklady:

IF A>5 THEN PRINT A	obsah A se vypíše pouze, když je větší než 5
IF A*5=100 THEN GOTO 35	v případě, že je v A hodnota 20, skočí program na řádek

	35 jinak se pokračuje následujícím řádkem
IF C&b1000 THEN BEEP 1000,100	je-li bit č.3 v proměnné C log. 1, zazní tón
IF A=0 THEN LET A=1; PRINT B	při nulové hodnotě A se A změní na 1 a vypíše se B

Příkaz: **GOTO**
Parametry: EXPR
Funkce příkazu: Jde o funkci skoku, kterým se přenese aktuální pozice zpracovaných řádků na řádek, jehož číslo udává výraz EXPR. Příkaz má význam pouze v režimu zpracovávání spuštěného programu. Pokud jej uživatel zadá jako příkaz v režimu zadávání příkazů, neprovede se nic. Parametr příkazu je obecný výraz. Programátor ale musí zajistit, aby hodnota tohoto výrazu nepřekročila rozsah 0 až 127. Při pokusu „skočit“ mimo povolené číslo řádku je ohlášena chyba a vykonávání programu se zastaví.

Příklady:

GOTO 23	přenese vykonávání programu na řádek číslo 23
GOTO 10+I*5	podle hodnoty v proměnné I skáče program na řádek v násobcích 5

Příkaz: **INPUT**
Parametry: VAR (,VAR)^
Funkce příkazu: Zadávání numerických hodnot do jednotlivých proměnných. Programátor může zvolit jako parametr, které hodnoty má uživatel zadat a ty se postupně uloží do dané proměnné. Uživatel je postupně vyzván, aby zadal číslo. Při zadávání je vypsáno jméno plněné proměnné. U proměnné typu pole je uživatel vyzván, aby zadal pouze „@“ (index není uveden). Čísla lze zadat pouze v dekadické formě.

Příklady:

INPUT L,H	uživatel je vyzván k zadaní hodnot do proměnných L a pak H
INPUT @20, @38	po uživateli bude 2x požadováno vložení hodnoty do @

Příkaz: **LET**
Parametry: VAR = EXPR nebo @VAR = EXPR(,NUM)^
Funkce příkazu: Přiřazovací příkaz určuje, jaká hodnota EXPR se vloží do proměnné VAR. Příkaz je možné použít také k současnému naplnění dalších hodnot do více položek pole. Po zadání hodnoty do určeného prvku pole, je možné vložit do následujících prvků numerické hodnoty oddělené čárkou. Počet zadaných hodnot je omezen pouze délkou řádku nebo dosažením konce pole.

Příklady:

LET I=52	do I se zapíše hodnota 52
LET Q= 23*W-L/3	výraz 23*W-L/3 se nejprve vypočítá a výsledek se zapíše do Q
LET @23=4,12,8,63,47	prvky pole 23 až 27 jsou naplněny postupně zadanými hodnotami

---

Příkaz: **GOSUB**
Parametry: EXPR
Funkce příkazu: Volání podprogramu, který může být na vyhrazených řádcích programu. Tento příkaz se velmi podobá příkazu GOTO a provádí prakticky totéž. Navíc se však uloží návratová hodnota pro příkaz RETURN. Návratová hodnota je číslo řádku o 1 větší, než má řádek, na kterém je právě zpracovávaný příkaz GOSUB. Po vykonání podprogramu se příkazem RETURN vrátí program na řádek daný návratovou hodnotou. V rámci podprogramu je možné volat jiný podprogram a takto do sebe podprogramy zanořovat. Maximální úroveň je ale omezena na 8 do sebe vnořených podprogramů. Pokus o zavolání devátého podprogramu z osmého vnořeného podprogramu způsobí ohlášení chyby a ukončení programu.

Příklady:

GOSUB 44	zavolá se podprogram uložený na řádku 44, po jeho ukončení příkazem RETURN se bude pokračovat na následujícím řádku

---

Příkaz: **RETURN**
Parametry: ---
Funkce příkazu: Ukončení podprogramu. Příkaz vezme návratovou hodnotu uloženou příkazem GOSUB a vloží ji do ukazatele na další prováděný řádek. Volat příkaz RETURN bez předchozího zavolání příkazu GOSUB (tj. chybějící návratová hodnota) způsobí ohlášení chyby a ukončení programu.

Příklady:

RETURN	ukončí podprogram a vrátí program na řádek za příkazem GOSUB

Příkaz: **CLEAR**
Parametry: ---
Funkce příkazu: Vymazání aktuálního programu. Po vykonání tohoto příkazu budou všechny řádky programu prázdné. Pokud byl v aktuálním souboru (viz příkaz FILE) nějaký program zapsán, bude kompletně vymazán.

Příklady:

CLEAR	vymaže aktuální program

Příkaz: **LIST**
Parametry: (NUM)
Funkce příkazu: Výpis aktuálního programu v paměti. Příkaz může být zadán bez parametru. V tomto případě vypíše celý aktuální program uložený v paměti počítače. Pokud je jako parametr uvedeno číslo, vypíše pouze konkrétní řádek. Číslo může být libovolné, ale platné je pouze nejnižších 7 bitů (tzv. aritmetika modulo 128).
Příklady:

LIST	vypíše na obrazovku celý program
LIST 15	vypíše řádek 15 programu

LIST 130 vypíše řádek 2 programu

Příkaz: **RUN**
Parametry: ---
Funkce příkazu: Spustí provádění programu. Po spuštění je zahájeno postupné vykonávání řádků programu od řádku 0. Není-li na daném řádku příkaz skoku (GOTO nebo GOSUB) je po provedení aktuálního řádku vykonán řádek s číslem o 1 větším. Řádky, na nichž není uložen žádný příkaz, jsou sice prohledávány, ale není na nich prováděna žádná akce. Představují pouze drobné zpoždění v programu (řádek se musí přečíst). U časově kritických částí programu je vhodné nenechávat zbytečně prázdné řádky. Pokud ukazatel řádků překročí hodnotu ze 127 na 128 je program automaticky ukončen.
Speciálním případem je uvedení příkazu RUN v souboru FILE 0 (u disku „h“) na první pozici řádku 0. Najde-li počítač na této pozici příkaz RUN, spustí automaticky po svém zapnutí tento program. Počítač tak nemusí mít připojenou klávesnici a může být po odladění programu používán např. k ovládání nějakého zařízení.

Příklady:
RUN spustí zapsaný program

Příkaz: **END**
Parametry: ---
Funkce příkazu: Ukončí provádění programu. Je-li při vykonávání programu zpracován příkaz END, program se ukončí a počítač se vrátí do režimu vyčkávání na uživatelský příkaz. Příkaz END je jediný příkaz, jehož stisknutí patřičných kláves (tj. Ctrl + N) je v průběhu přechodu mezi řádky za běhu programu testováno. Při zadání tohoto příkazu na klávesnici je běžící program ukončen.

Příklady:
END ukončí běžící program

Příkaz: **TIME**
Parametry: VAR
Funkce příkazu: Do proměnné VAR uloží hodnotu z trvale běžícího interního čítače. Čítač je od startu počítače zvětšen o 1 vždy, když proběhne vykreslení jedné grafické linky na VGA monitoru. Kvůli synchronizaci s frekvencí oscilátoru běží čítač s frekvencí 31914 Hz. To znamená, že každou sekundu je jeho obsah zvětšen o 31914. Protože je čítač 32bitový, musí dojít k jeho přetečení. Po dosažení hodnoty 2 147 483 647, k čemuž dojde asi po více než 18 hodinách provozu, skočí hodnota na nejmenší záporné číslo -2 147 483 648 a pokračuje v dalším růstu o 1. Další přetečení pak nastanou vždy za více než dalších 37 hodin.
Funkci je možné využít nejenom pro hlídání uplynutí určitého časového intervalu během vykonávání programu, ale také např. jako generátor náhodného čísla. Při stisknutí klávesy a uložení aktuálního času si stačí vypočítat zbytek po dělení času číslem, které určuje rozsah generátoru.

Příklady:

TIME T	uloží čas do proměnné T
TIME R; LET Q=R/6; LET R=R-Q*6	v proměnné R získáme náhodné číslo 0 až 5 (např. pro hod kostkou)

Příkaz: **CURSOR**
Parametry: EXPR
Funkce příkazu: Nastaví kurzor (tj. místo, kde se bude na obrazovce zapisovat) na určenou pozici. Levý horní roh obrazovky má pozici 0 a pravý dolní roh má pozici 799. Pozice rostou na řádcích zleva doprava stejně, jako když se píše. Pokud je pozice mimo povolený rozsah, příkaz neprovede nic a pozice kurzoru zůstává na původní hodnotě.
Jestliže je potřeba nastavit pozici kurzoru pomocí čísla řádku, pak lze použít výpočet, kdy se číslo řádku vynásobí 32 a přičte pozice znaku v rámci řádku.

Příklady:

CURSOR A	přesune pozici kurzoru na hodnoty podle proměnné A
CURSOR 32*5,15	přesune kurzor na šestý řádek shora a 16. pozici na tomto řádku
CURSOR 1024	neudělá nic (hodnota je mimo povolený rozsah)

Příkaz: **PUTCH**
Parametry: EXPR nebo x EXPR (kde x je znak viz dále) nebo p EXPR, EXPR
Funkce příkazu: Na pozici určené kurzorem (tj. místem, kde se bude na obrazovce zapisovat) vypíše znak jehož hodnota je určena výrazem parametru. Pro výpis je použito nejnižších 8 bitů. Vypisuje se znak v rozsahu 0 až 255 (pro vyšší čísla to bude zbytek po dělení 256). Vykreslený znak odpovídá tzv. ASCII tabulce. Např. písmeno „A“ má hodnotu 65.
V rámci rozsahu ASCII tabulky budou zobrazovány pouze tisknutelné znaky od 32 do 127. Nad tento rozsah budou v rozmezí 128 až 159 zobrazovány semigrafické znaky (především čáry vhodné pro různé rámečky). Za semigrafickými znaky je pak oblast, jejíž hodnota se rozbalí jako klíčové slovo jazyka BASIC. Např. Hodnota 160 způsobí vypsání slova „PRINT“.
Zbývající znaky jsou netisknutelné a místo nich se objeví tečka. Výjimkou jsou pouze 3 hodnoty, které odpovídají znakům „\n“, „\t“ a „\f“, jak bylo uvedeno u parametru STRING výše:

- 9 „\t“ - přesun kurzoru na řádku na nejbližší vyšší pozici tabulátoru 0,8,16 nebo 24
- 10 „\n“ - přechod na nový řádek
- 12 „\f“ - vymazání obrazovky a kurzor na pozici 0,0

Kurzor se posouvá stejně jako při vypisování textu pomocí příkazu PRINT. Na konci řádku se přejde na nový řádek a případně na poslední pozici se posune celá obrazovka.
V případě, že je před EXPR uveden jeden z následujících znaků, posouvá se kurzor zvláštním způsobem:

- . kurzor zůstane na stejné pozici
- > kurzor se posune o jedno místo doprava
- < kurzor se posune o jedno místo doleva
- v kurzor se posune o jedno místo dolů
- ^. kurzor se posune o jedno místo nahoru

V těchto případech se nepřechází na nový řádek nebo sloupec. Kurzor se posouvá (a přetéká) v rámci stále stejného řádku nebo sloupce. Tímto způsobem je možné vypisovat pouze tisknutelné znaky v rozsahu 32 až 159.

Poslední možností je vložit znak přímo na určenou pozici na displeji bez ovlivnění kurzoru. Po znaku „p“ je uvedena hodnota znaku EXPR, který je vložen na pozici zadanou druhým parametrem. Pozice na displeji je v rozsahu 0 až 799 stejně, jako je u příkazu CURSOR. Takto je opět možné vypisovat pouze tisknutelné znaky v rozsahu 32 až 159.

Příklady:

PUTCH 49	vypíše na pozici kurzoru číslici 1
PUTCH p68, 32*12+10	vypíše písmeno D na řádku 12, pozice 10
PUTCH 12	vymaže obrazovku a nastaví kurzor na levý horní roh
PUTCH .65	vypíše písmeno A a kurzor ponechá na stejné pozici
PUTCH v67	vypíše písmeno C a přesune kurzor na nižší řádek se stejnou pozicí (v případě nejnižšího řádku přejde na nejvyšší řádek)

Příkaz: **GETCH**
Parametry: VAR nebo pEXPR, VAR
Funkce příkazu: Do proměnné VAR uloží hodnotu znaku, který se nachází na pozici kurzoru na obrazovce. Pomocí příkazu CURSOR je možné nastavit kurzor na libovolnou pozici a pomocí příkazu GETCH je možné prohledávat, co bylo na dané pozici zapsáno.
Druhou možností je určit po znaku „p“ přímo pozici na displeji v rozsahu 0 až 799, ze které se vezme znak a ten se uloží do proměnné VAR.

Příklady:

CURSOR 392; GETCH C	do proměnné C je vložen znak z pozice 12,8 na obrazovce
GETCH p32*15+4, L	do proměnné L je vložen znak z pozice 4,15 na obrazovce

---

Příkaz: **INKEY**
Parametry: VAR
Funkce příkazu: Příkazem se otestuje, zda uživatel stiskl nějakou klávesu. Pokud nebyla stisknuta žádná klávesa, je do proměnné VAR uložena hodnota 0. Jestliže došlo ke stisknutí některé klávesy před voláním příkazu, je do proměnné uložena odpovídající hodnota znaku podle ASCII tabulky. Např. Hodnota 66 znamená, že uživatel zmáčkl „B“.

Příklady:

INKEY C	do proměnné C je vložen znak již stisknuté klávesy, nebo 0 v případě, že nic nebylo stisknuto

---

Příkaz: **FILE**
Parametry: (EXPR)
Funkce příkazu: Podle velikosti připojené paměti má uživatel k dispozici nejméně 1 a nejvíce 16 souborů pro zapsání programu o 128 řádcích. Mezi těmito programy se přepíná pomocí příkazu FILE, kde výraz EXPR určuje, jaký soubor (program) bude aktivní. S tímto programem se pak pracuje, může se do něj zapisovat nebo číst. EXPR může nabývat libovolné hodnoty, ale platné jsou pouze nejnižší 4 bity (zbytek po dělení 16). Takže např. hodnota 17 je ekvivalentní hodnotě 1.

Velikost použité paměti určuje, kolik souborů má uživatel k dispozici. Je-li použita nejmenší paměť 24C32, nemá příkaz FILE žádný význam, protože je k dispozici pouze soubor s číslem 0 a ten je stále aktivní. Větší paměť 24LC64 má již dva soubory – 0 a 1. Výraz EXPR pak umožňuje přepínání mezi soubory, podle toho, je-li výraz lichý nebo sudý. U paměti 24LC128 lze volit pomocí EXPR postupně soubory 0,1,2,3,0,1,2,3,0,1... Teprve paměť 24LC512 využije možnost celého rozsahu tohoto příkazu a volí soubory číslo 0 až 15.
Příkaz FILE se bude asi nejčastěji používat při zadávání a spouštění programů uživatelem. Jeho použití je možné i v rámci spuštěného programu. Získává se tím mocný nástroj, který umožní spojovat různé soubory a přeskakovat za běhu mezi nimi.
Pokud běží např. program v souboru číslo 0 a na řádku 25 vykoná příkaz FILE 3, pak program pokračuje řádkem 26 v souboru 3. Takto lze obejít limitující velikost programu na 128 řádků. Při použití paměti 24LC512 a využívání možnosti napsat až 3 příkazy na řádek, lze do paměti umístit teoreticky program o velikosti až kolem 6000 příkazů. V kombinaci s příkazem DISK (viz dále) toto číslo vzroste až na 12000.
V případě, že se parametr neuvede (nebo je špatně zadaný výraz), vypíše tento příkaz na novém řádku číslo aktuálního souboru.

Příklady:

FILE 10	přepne na soubor číslo 10 (pouze pro 24LC512), na soubor 0 (pro 24C32, 24LC64) a na soubor 2 (pro 24LC128 a 24LC256)
FILE	vypíše číslo aktuálního souboru

Příkaz: **DISK**
Parametry: (h|f)
Funkce příkazu: Příkazem se přepíná mezi pamětmi na desce počítače. Jedna z pamětí je v provedení SMD pevně připájená na desku a nelze ji měnit. Tato paměť představuje „harddisk“ počítače a má označení h. Druhou paměť v provedení DIL je možné zasunout a opět vyjmout z patice a má označení f. Obvod je možné označit za „floppy disk“ - vyměnitelné médium. Po startu počítače je aktivní DISK h a v něm soubor FILE 0.
Parametr určuje, se kterým diskem se bude dále pracovat – bude aktivní.
Jestliže není při startu vložený DISK f, oznámí to počítač při úvodním hlášení po zapnutí. Pak samozřejmě není možno pomocí příkazu DISK f přepnout na tento disk. Příkaz nic neprovede, a ani také nehlásí chybu. Zůstane aktivní stále DISK h.
Při nezadání nebo nesprávné volbě parametru vypíše příkaz na nový řádek označení aktivního disku.

Příklady:

DISK f	přepne na DISK f, jestliže je vložený; jinak neprovede nic
DISK	vypíše aktivní DISK

Příkaz: **DELAY**
Parametry: EXPR
Funkce příkazu: Pozastavuje běh programu na dobu určenou parametrem EXPR. Hodnota výrazu určuje dobu v milisekundách. Povolené hodnoty jsou 1 až 1 000 000. Mimo tento rozsah nedělá příkaz nic.

Příklady:
DELAY 5*100 program počká 500 milisekund a pak pokračuje dále

---

Příkaz: **BEEP**
Parametry: EXPR, EXPR
Funkce příkazu: Malý reproduktorek na desce vydá tón o frekvenci určené prvním parametrem po dobu definovanou druhým parametrem. První parametr udává frekvenci v Hz a povolené hodnoty jsou v rozsahu 100 až 4000 Hz. Vzhledem k parametrům těchto malých zvukových bzučáků budou různé frekvence různě hlasité a větší rozsah hodnot nemá smysl.
Druhý parametr definuje dobu, po kterou bude tón znít v milisekundách. Povolené hodnoty jsou mezi 20 a 100 000 milisec. Při nedodržení rozsahu parametrů se příkaz neprovede – nedělá se nic.

Příklady:
BEEP 800, 1500 zazní tón 800 Hz po dobu 1,5 sekundy

Příkaz: **REM**
Parametry: Libovolný text
Funkce příkazu: Prázdný příkaz, který neprovádí nic. Slouží pouze jako komentář pro programátora, který si může napsat krátkou poznámku do programu. Text až do konce řádku je považován za komentář.

Příklady:
REM Dale je vypocet plochy kruhu může být libovolný text do konce řádku

---

Příkaz: **COPY**
Parametry: (h|f)NUM nebo sNUM,NUM,NUM
Funkce příkazu: Umožňuje kopírovat zdrojový text programu mezi soubory nebo kopírovat řádky v rámci aktivního souboru.
První možnost způsobí, že obsah aktivního souboru, se kterým se pracuje přepíše soubor zadaný v parametru. Písmena h nebo f určují disk, na nějž se aktivní soubor zapíše a číslo NUM určuje cílový soubor na tomto disku. Zkopírovány jsou kompletně všechny řádky 0 až 127. Vytvoří se přesná kopie souboru na daném disku.
Druhé zadání parametrů uvedené písmenem s kopíruje řádky v rámci aktivního souboru. První číslo (od 0 do 127) udává zdrojový řádek, kde se začne kopírovat. Druhý parametr (od 0 do 127) volí místo, kam se začne ukládat. Třetí parametr (1 až 127) pak určuje, kolik řádků bude kopírováno. Pokud je při kopírování dosaženo konce souboru, kopírování se ukončí.

Příklady:
COPY f6 zkopíruje aktivní soubor na disk f do souboru č. 6
COPY s25,50,20 kopíruje 20 řádků v aktivním souboru od řádku 25 na řádky 50 a výše

---

Příkaz: **AINP**
Parametry: VAR
Funkce příkazu: Počítač má standardně připravený jeden analogový vstup umožňující měřit napětí v rozsahu 0 až 5V. Napětí 0V je měřeno jako hodnota 0, maximálnímu napětí 5V odpovídá hodnota 1023. Je potřeba zohlednit, že nejde o přesný měřicí přístroj. Rozsah měření a přesnost jsou ovlivněny více faktory. Pro běžné amatérské použití by to však mělo postačovat. Hodnota změřená příkazem AINP je uložena do proměnné VAR.
Napětí mimo uvedený rozsah není možné měřit a takové napětí může velmi pravděpodobně počítač poškodit!

Příklady:
AINP @32 změř analogové napětí na vstupu A a ulož ho do prvku pole s indexem 32

Příkaz: **DINP**
Parametry: VAR, (1|2) - pro verzi s krystalem, bez krystalu: VAR, (1|2|3|4)
Funkce příkazu: Při použití krystalu jako zdroje přesné frekvence má počítač na desce připravené 2 digitální vstupy. Není-li krystal osazený a počítač je taktovaný vnitřním oscilátorem, je počet vstupů rozšířen na 4. Příkazem se přečte stav vstupu s uvedeným číslem do proměnné VAR. Hodnota uložená do proměnné může být buď 0, nebo 1 podle logické úrovně na vstupu. Nepřipojený vstup se chová jako logická 1. Použitím příkazu se přepne do režimu vstupu se slabým přítahem na log 1. Až do použití příkazu DOUT zůstává vývod jako vstupní.

Příklady:
DINP R,2 do proměnné R se přečte logický stav na digitálním vývodu 2

Příkaz: **DOUT**
Parametry: EXPR, (1|2) - pro verzi s krystalem, bez krystalu: EXPR, (1|2|3|4)
Funkce příkazu: Příkaz opačný k předcházejícím příkazu DINP. Po jeho provedení se z daného digitálního vývodu stane výstupní s určenou logickou úrovní. Příkazem se zapíše hodnota EXPR na určený výstup. Je-li hodnota 0, pak je na výstupu také logická 0. Každá jiná hodnota pak zapisuje na výstup logickou 1. Až do použití příkazu DINP zůstává vývod jako výstupní.

Příklady:
DOUT 23,2 na digitálním vývodu 2 se nastaví výstup s logickou 1 (hodnota není 0)

Příkaz: **I2CW**
Parametry: EXPR
Funkce příkazu: Počítač má poměrně málo digitálních vstupů a výstupů (2 nebo 4 podle osazení desky). Pokud by tento počet nedostačoval pro zamýšlený projekt, je možné vstupy a výstupy rozšířit pomocí desky s expanderem PCF8575. Pak se počet rozšíří o dalších 16 digitálních vstupů nebo výstupů. Tyto expandery je možné připojit pomocí konektoru s vyvedenou sběrnicí I2C na desce.
Hotové desky lze často zakoupit jako doplňky k modulům Arduino apod. Pozor na zapojení

konektorů na desce expanderu! Deska s PCF8575 jde do konektoru obvykle zapojit přímo, ale je lepší si to ověřit!
Příkaz odešle 16 spodních bitů z výrazu EXPR do desky expanderu. Předpokládá se standardně 16bitový expander. Má tedy význam posílat pouze hodnotu do max. 65535.

Příklady:

I2CW 61455	8 prostředních bitů v expanderu PCF8575 budou 0 doplněné shora i zdola 1

Příkaz: **I2CR**
Parametry: VAR
Funkce příkazu: Doplňkový příkaz k předcházejícímu příkazu I2CW – viz výše. Umožňuje číst stav vstupů z desky s expanderem PCF8575 do proměnné VAR.
Opět se standardně předpokládá 16bitový expander PCF8575. Přečte hodnotu do max. 65535.

Příklady:

I2CR H	přečte stav z expanderu PCF8575 do proměnné H

Příklad programu

Jako příklad krátkého programu byl vybrán postup pro zobrazení všech tisknutelných znaků na obrazovce:

```
0  REM  Printable characters
1  PUTCH  12;  LET  A=32
2  PUTCH  A;  LET  A=A+1
3  IF  A<160  THEN  GOTO  2
```

Řádek 0 je pouze komentář s názvem programu a neprovádí nic.
Řádek 1 nejprve vysláním znaku 12 vymaže obrazovku a pak inicializuje proměnnou A na 32.
Řádek 2 vypíše znak z A na displej a zvedne hodnotu proměnné A o jedničku.
Řádek 3 otestuje, zda bylo již dosaženo konečně hodnoty a pokud ne, pak se vrací na řádek 2.

Pomocí tohoto programu je možné zobrazit tisknutelné znaky na obrazovce. Po spuštění se objeví 4 řádky znaků. Řádky začínají znakem s hodnotou odpovídající 32, 64, 96 a 128.

Co dělat když...

Při používání počítače mohou nastat situace, které vedou k zablokování chodu přístroje. V těchto případech může pomoci následující postup:

1) Jestliže obsahuje spuštěný program nekonečnou smyčku, je možné jeho běh přerušit zmáčknutím tlačítka RESET. Tím se u počítače provede tzv. studený start, kdy jsou všechny proměnné nastaveny na nulovou hodnotu.
   Druhou možností je zadat v průběhu provádění programu příkaz END (Ctlr+N). Takto přerušený program ponechá hodnotu proměnných takovou, jaká byla v okamžiku přerušení. Zadání příkazu END obvykle zanechá tento příkaz nebo pouze písmeno „n“ na příkazové řádce. Je potřeba provést dodatečné smazání tohoto zbytku příkazu.

2) Pokud po zapnutí počítače dojde k manipulaci s připojenou klávesnicí (odpojování a připojování), může dojít k narušení její komunikace. Klávesnice pak nereaguje nebo zadává chybné znaky. Problém vyřeší zmáčknutí tlačítka RESET.

3) Počítač má programy uložené v sériové EEPROM paměti připojené na sběrnici I2C. V případě nějakých rušivých signálů na sběrnici se může paměť zablokovat a počítač přestane reagovat úplně. Často nepomůže ani stisknutí tlačítka RESET. V tomto případě je potřeba odpojit počítač od displeje a vypnout napájení. Po několika sekundách ve vypnutém stavu je možné počítač opět zapnout a připojit k displeji.

4) **Paměť do patice pro disk „f“ vkládejte nebo vytahujte vždy ve vypnutém stavu (počítač bez napájení)!** Hazardní stavy na sběrnici při manipulaci s pamětí mohou poškodit její obsah nebo obsah paměti představující disk „h“. Podobně je potřeba přistupovat i k připojování nebo odpojování expanderu s obvodem PCF8575.

Příkazy a jejich parametry:

Příkaz	Ctrl +	Parametry	Poznámka
PRINT	**P**	(STRING | EXPR) (,(STRING | EXPR))^	
IF	**U**	EXPR RelOp EXPR	7 různých operátorů
THEN	**T**	STATEMENT	
GOTO	**G**	EXPR	0 až 127
INPUT	**I**	VAR (,VAR)^	
LET	**L**	VAR = EXPR nebo @VAR = EXPR(,NUM)^	
GOSUB	**H**	EXPR	0 až 126
RETURN	**Y**		
CLEAR	**X**		
LIST	**K**	(NUM)	Modulo 128
RUN	**R**		
END	**N**		
TIME	**V**	VAR	31914 Hz pulsy
CURSOR	**W**	EXPR	0 až 799
PUTCH	**C**	EXPR nebo x EXPR nebo p EXPR, EXPR	x zvol / pZnak, Pozice
GETCH	**,**	VAR nebo p EXPR, VAR	pPozice, Proměnná
INKEY	**S**	VAR	
FILE	**F**	(EXPR)	Modulo 16
DISK	**D**	(h|f)	
DELAY	**M**	EXPR	1 až 1000000
BEEP	**Z**	EXPR, EXPR	Frekvence, Doba
REM	**E**	Libovolný text	
COPY	**.**	(h|f)NUM nebo sNUM,NUM,NUM	sZdroj, Cíl, Počet
AINP	**A**	VAR	
DINP	**B**	VAR, (1|2) nebo VAR, (1|2|3|4)	Krystal / bez krystalu
DOUT	**O**	EXPR, (1|2) nebo EXPR, (1|2|3|4)	Krystal / bez krystalu
I2CW	**Q**	EXPR	16 bitů
I2CR	**J**	VAR	16 bitů

^ - možnost opakování

| - možnost volby z více možností