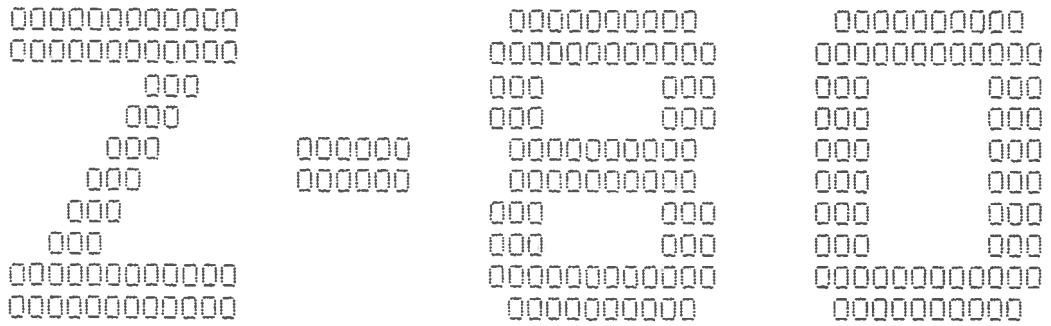
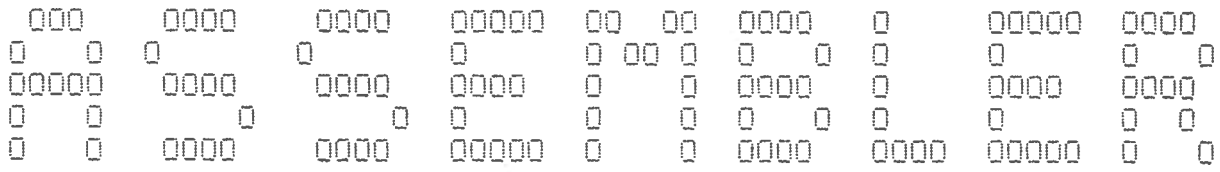
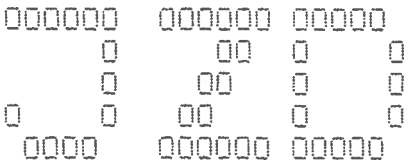
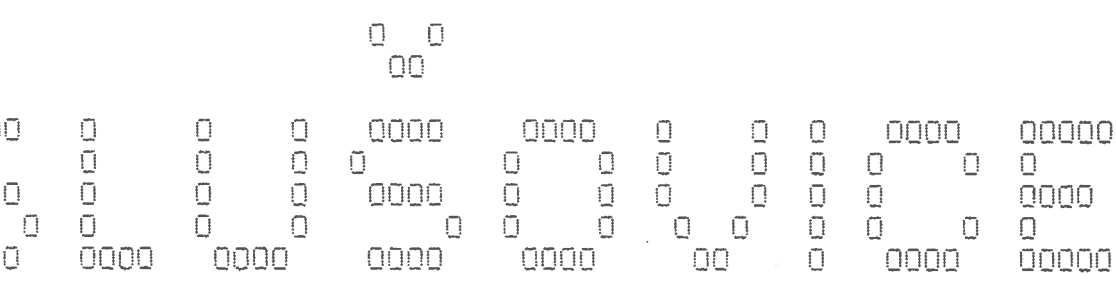








Opis knihy vydané v počtu 1000 kusů JZD Slušovice

Natištěno pro klub výpočetní techniky Karolinka

II. nezměněné vydání

O B S A H K N I H Y

únor 1988

A S S E M B L E R

Z - 8 0

= = = = = = = = = = = = = = = =

/ Opis knihy, již vydalo v počtu tisíc kusů JZD Slušovice /

P r a v i d l a

Program, zapsaný v assembleru Z-80, se skládá ze sledu instrukcí, z nichž každá je zakončena znakem konce řádku. /znak ASCII CR/.

Každá instrukce může mít maximálně 128 znaků.

Instrukce je tvořena po sobě jsoucími čtyřmi poli, oddělenými oddělovači:

Návěští	Op - kód	Operandy	Poznámka
NAV1:	LD	HL, NECO	; presun NECO do HL

Do těchto čtyř polí je nutné psát vždy v předepsaném pořadí, i když jsou některá pole vynechána.

A./ O D D Ě L O V A Č E

Operační kódy, operandy a návěští musí být od sebe odděleny jedním, nebo více oddělovači.

Oddělovače jsou: buď čárka, mezera, nebo tabulátor.

B./ P O Z N Á M K Y

Nejsou funkční částí programu. Překladač ji pouze přepíše do protokolu o překladu. Může začít v kterémkoliv sloupci, je uvedena středníkem, za nímž následuje libovolný počet znaků.

C./ N Á V Ě Š T Í

Definuje 16-ti bitovou adresovou konstantu s hodnotou, odpovídající poloze návěští v programu. Skládá se z jednoho, nebo více znaků, z nichž prvních šest tvoří klíč. První znak musí být abecední, ostatní abecedně číslicové /A ... Z, Ø ... 9/. Návěští začíná buď v prvním sloupci, nebo bezprostředně za návěštím musí následovat dvojtečka.

D./ O P E R A Č N Í K Ó D Y A O P E R A N D Y

Strojové instrukce jsou zastoupeny symboly, které se skládají z op-kódu a z žádného, nebo z několika operandů. Assembler také pracuje s pseudoinstrukcemi, které nemají smysl strojové instrukce, ale pouze informují překladač /např. o tom, má-li vynechat místo pro pole dat/.

E./ Z Á P I S Č Í S E L

Překladač může pracovat v těchto číslicových soustavách:

- binární - za číslem musí následovat B /10010B/
- osmičkové - -"- -"- Q /7635Q/
- desítkové - -"- -"- D nebo nic /96D, 96/
- šestnáctkové - -"- -"- H /hexadecimální/

Poznámka: Hexadecimální číslo nesmí začínat písmenem. Před případné písmeno je nutné napsat nulu /ØFDH,.../

F./ Z N A K O V É K o n s t a n t y

Překladač umožňuje definovat konstanty, které vyjadřují kódy ASCII znaků. Znak se musí zapsat do jednoduchých uvozovek /např. 'A' má hodnotu 41H/.

Zavedení znaku jednoduchých uvozovek se provádí zdvojením v textu /např.: LD A, '''' zavede do registru A hodnotu 27H, což odpovídá znaku jednoduchých uvozovek v kódu ASCII./.

G./ V E L K Á A M a L Á P Í S M E N A

Operační kódy a klíčová slova /identifikátory registrů/ musí být psány buď pouze malými, nebo pouze velkými písmeny.

Návěští může sestávat z kombinace malých a velkých písmen.

H./ A R I T M E T I C K É A L O G I C K É V Ý R A Z Y

Překladač umožňuje pracovat s řadou aritmetických a logických operací. Aritmetické výrazy mohou obsahovat závorky. Uzavře-li se ale celý výraz do závorek, znamená to odkaz na paměťovou lokaci. Výrazy jsou vyhodnocovány ve smyslu priority aritmetických výrazů a závorek. Možné aritmetické a logické výrazy udává následující tabulka:

operátor	funkce	priorita
+	unární plus	1
-	unární minus	1
.NOT.	negace	1
.RES.	výsledek	1
* *	umocnování	2
*	násobení	3
/	dělení	3
.MOD.	modulo	3
.SHR.	logický posun doprava	3
.SHL.	-"- -"- doleva	3
+	sčítání	4
-	odčítání	4
.AND. nebo &	logický součin	5
.OR. nebo ^	-"- součet	6
.EQ. nebo =	ekvivalence	7
.GT. nebo >	větší než	7
.LT. nebo <	menší než	7
.UGT.	bez znaménka větší než	7
.ULT.	-"- menší než	7

Při vyhodnocování se všude používá celočíselná aritmetika na 16 bitech. Záporná hodnota se vytvoří unární operací " - " /např.: LD HL, -36H/.

U srovnávacích operátorů /.EQ.; .GT.; .LT.; .UGT.; .ULT./ je výsledkem logická hodnota TRUE /1 ve všech bitech/ nebo FALSE /Ø ve všech bitech/.

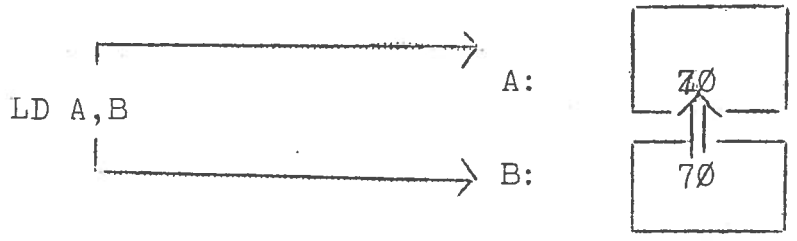
Operátor :RES. potlačuje hlášení chyby při přetečení.

Např.: LD BC, 7FFFH + 1 ; bude hlášena chyba
LD BC, .RES. 7FFFH + 1; nebude hlášena chyba

I./ V Ý Z N A M O P E R A N D Ů

Operand, vyskytující se za instrukcí, určuje objekt, ke kterému se instrukce vztahuje. Z hlediska programování může operand určovat objekt čtyřmi způsoby:

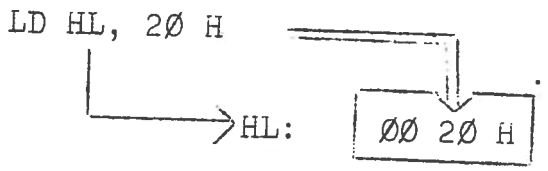
a/ Na místě operandu je označení registru, nebo registrového páru. Operandem /cílem přenosu, nebo jiné operace dané instrukcí/ je obsah tohoto registru nebo registrového páru /viz obr. 4.1. - 1/.



obr. 4.1. - 1

Poznámka: V tomto případě a v dalším textu nabývá slovo operand poněkud jiného významu, než při popisování adresových režimů.

b/ Na místě operandu je výraz, zapsaný buď dekadicky, binárně, oktalově, hexadecimálně, nebo jako adresová konstanta / návěští / nebo konstanta definovaná instrukcí EQU. Operandem je v tomto případě přímo tato konstanta /viz. obr. 4.1. - 2/.

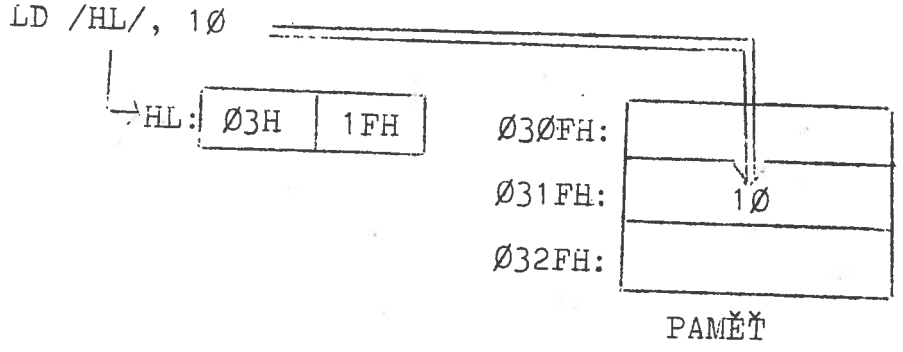


obr. 4.1. - 2

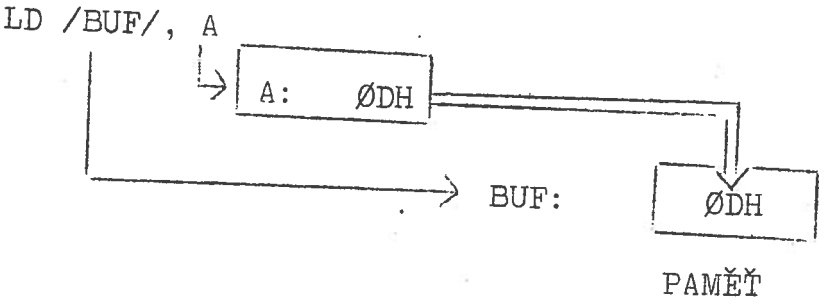
Příklad: LD A, 1ØH
LD B, 21
LD HL, LABEL

c/ Na místě operandu je výraz, návěští, nebo registrový pár, uzavřený do kulatých závorek. Je-li v závorkách číslo nebo návěští, je operandem obsah paměťové lokace, na kterou ukazuje číslo, nebo návěští. Je-li v závorkách uzavřen registrový pár, je operandem obsah paměťové lokace, jejíž adresa je ve zmíněném registrovém páru.

Poznámka: U instrukcí skoku, ve kterých se vyskytuje na místě operandu registrový pár, uzavřený do kulatých závorek, je cílem skoku adresa, kterou obsahuje registrový pár.



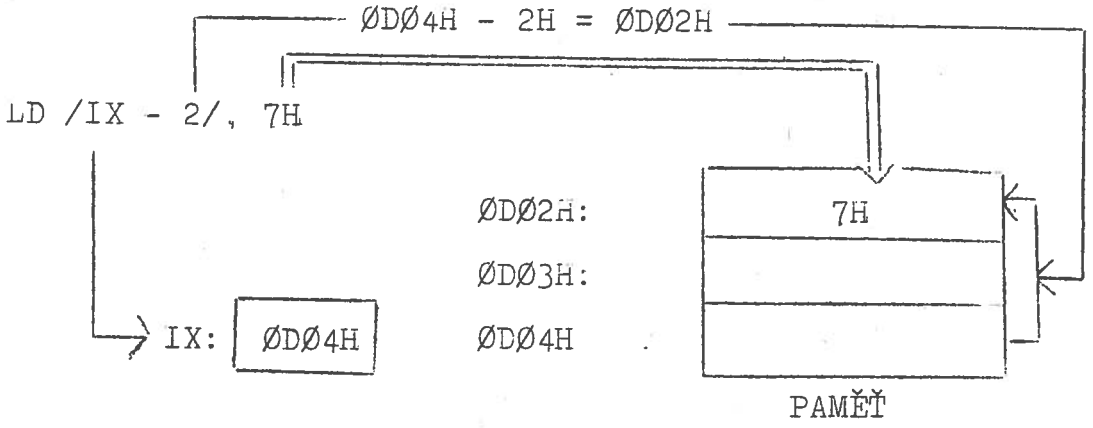
obr. 4.1. - 3



obr. 4.1. - 4

d/ Je-li na místě operandu uzavřen v kulatých závorkách některý z index registrů s indexem posunu /IX + d/, bude místem přenosu paměťová lokace, která je dána součtem obsahu index registru a indexem posunu d.

Index posunu může být číslo v některé z výše uvedených číselných soustav v rozsahu jednoho byte.
/viz. obr. 4.1. - 5/.



obr. 4.1. - 5

1.2. P S E U D O I N S T R U K C E

Pseudoinstrukce mají stejný formát jako strojové instrukce, nemají však zanásledek generaci kódu instrukce, ale řídí generaci cílového kódu.

A./ D E F I N I C E B Y T E

DEFB I.

Pseudoinstrukce DEFB generuje jeden byte s hodnotou, danou operandem I., na běžné hodnotě referenčního čítače / na tom místě, kde je pseudoinstrukce v programu uvedena/.

Příklad:
```
DVE   : DEFB 2
CISLO : DEFB 3 + 3
ZNAK  : DEFB 'V'
```

Poznámka: Hodnota operandu musí být v rozsahu -128 až 255. Operand může být buď čís lo, nebo aritmetický, či logický výraz v absolutním režimu /tzn. vyčíslitelný, v době překladu/.

B./ D E F I N I C E S L O V A

DEFW I.

Pseudoinstrukce DEFW generuje jedno slovo /16 bitů/, dané hodnotou operandu I1, na běžné hodnotě referenčního čítače. Nejprve se ukládá nižší byte, pak vyšší byte, což je stejný formát jako při ukládání adres v paměti. Jestliže je operand znaková hodnota, ukládá se v tom pořadí, ve kterém je zapsán. Operand může být absolutní, relativní, nebo externí aritmetický, nebo logický výraz, tzn., že může obsahovat návěští, které může být deklarováno jako externál.

Příklad:	Object code	Návěští	Op-kód	Operand
	BD4Ø		DEFW	NAV
	Ø4ØØ		DEFW	2 + 2
	3Ø31	CISL:	DEFW	Ø1

C./ D E F I N I C E H L Á Š E N Í

DEFM

Pseudoinstrukce DEFM generuje posloupnost byte, odpovídající ASCII kódu posloupnosti znaků v poli operandu. Řetězec v poli operandu je tvořen posloupností znaků, uzavřených do jednotlivých jednoduchých uvozovek. Počet znaků nesmí být větší jak 63. Případné návěští bude ukazovat na první znak v posloupnosti.

Příklad:	Object code	Návěští	Op-kód	Operand
	612762	TU:	DEFM	'a' 'b'

D./ D E F I N I C E T E X T U

DEFT

Pseudoinstrukce DEFT generuje posloupnost byte, odpovídající textu v poli operandu v kódu ASCII stejně jako pseudoinstrukce DEFM, ale kromě toho uloží jako první byte údaj o délce textu /binárně počet znaků/.

Příklad: TEXT: DEFT 'POZOR'

E./ D E F I N I C E P A M Ě T I

DEFS I.

Pseudoinstrukce DEFS rezervuje paměť tak, že zvyšuje hodnotu referenčního čítače o hodnotu danou operandem I. Jakýkoliv výraz, vyskytující se v poli operandu, musí být pro překladač předem definován.

Příklad: BUFFER: DEFS 128

F./ D E F I N I C E N Á V Ě Š T Í

DEFL I.

Pseudoinstrukce DEFL se používá k přiřazení hodnoty výrazu v poli operandu I. k symbolu v poli návěští stejně jako pseudoinstrukce EQU. Rozdíl je v tom, že DEFL se může vyskytnout vícekrát ve spojení se stejným návěštím, má tedy platnost vždy od místa definice až po další předefinování.

```
Příklad:    BUFFER  DEFL    2ØH
                    LD      /BUFFER/, A; zápis na adresu 2ØH
            BUFFER  DEFL    5ØH
                    LD      /BUFFER/, A; zápis na adresu 5ØH
```

G./ D E F I N I C E S Y M B O L U

EQU I.

Používá se k přiřazení hodnoty výrazu I. v poli operandu k symbolu v poli návěští. Jakýkoliv symbol, který se ve výrazu vyskytuje, musí být pro překladač předem definován.

```
Příklad:      RADKU:    EQU 2Ø        ; počet řádků
            SLOUPCU:    EQU 1Ø        ; počet sloupců
              PRVKU:    EQU 1Ø * 2Ø   ; počet prvků matice
```

H./ K O N E C Z D R O J O V É H O P R O G R A M U

END

Pseudoinstrukce END označuje konec zdrojového programu. Jakýkoliv text za touto pseudoinstrukcí není uvažován. Případné návěští dostane hodnotu volného slova za programem.

```
Příklad:      KONEC:    END           ; konec programu
```

I./ N A S T A V E N Í R E F E R E N Č N Í H O Č Í T A Č E

ORG I.

Tato pseudoinstrukce se používá ve zdrojových programech, určených pro absolutní překladače a slouží k nastavení referenčního čítače. Referenční čítač je touto instrukcí nastaven na adresu, danou výrazem I. Tuto pseudoinstrukci můžeme použít v programu zcela libovolně. Část programu, následující za touto pseudoinstrukcí, se překládá od adresy, dané výrazem v poli operandu.

Příklad: Při přerušení se skáče na adresu Ø38H. Máme zajistit přechod na obslužný podprogram, začínající na adrese 3ØØØH.

```
          ORG    38H            ; nastavení referenč. čítače
          JP     INTER          ; skok na obslužný program
          ORG    3ØØØH          ; nastavení referenč. čítače
INTER:                          ; adresa obsluhy
```

J./ D E F I N I C E G L O B Á L U

GLOBAL I.

Pseudoinstrukce GLOBAL slouží k navázání relativních programů spojovačem. V poli operandu I. jsou uvedena návěští /oddělená mezerami nebo čárkami/, která chceme "nabídnout" ostatním relativním programům, aby se na ně mohly odvolávat.

Příklad: Vytvořme podprogram se vstupními body VSTUP a POKRACUJ.

```
          GLOBAL    VSTUP, POKRACUJ ; pseudoinstrukce GLOBAL
VSTUP:      :                       ; deklarace návěští VSTUP
POKRACUJ:                           ; deklarace návěští
                                      POKRACUJ
```

K./ DEFINICE EXTERNÁLU

EXTERNAL I.

Pseudoinstrukce EXTERNAL slouží k navázání relativních programů spojovačem. V poli operandu I. jsou návěští, oddělená čárkami, nebo mezerami. Tato návěští nesmějí být deklarována ve stejném programu, jako ve kterém jsou definována jako externál. Naopak musí být definována v jiném relativním programu jako globál.

Příklad: Máme zajistit spojení se vstupním bodem "VSTUP" podprogramu, který bude sestavován s tímto programem.

```
EXTERNAL   VSTUP      ; definice externálu
     :
CALL VSTUP            ; použití externálu
```

1.3. MAKROINSTRUKCE

Některé překladače dovolují definovat makroinstrukce a vyvolávat je kdekoliv v textu programu. Při definici makroinstrukce se mohou nedefinovat také vstupní parametry. Aby nedocházelo ke zdvojené deklaraci návěští při každém vyvolání makroinstrukce, používá se generátoru symbolů. K modifikaci instrukce nazákladě parametrů makra se používá pseudoinstrukce podmíněného překladu.

Vzhledem k tomu, že syntax makroinstrukcí je dán typem konkrétního překladače, nebudeme se popisem k makroinstrukcí v dalším textu zabývat.

1.4. STAVOVÉ INDIKÁTORY

Stavové indikátory jsou některé bity registru F, popř. F'. Tyto indikátory poskytují informace o stavu CPU a především způsobu dokončení poslední aritmetické operace. Umístění stavových bitů v registru F je zřejmé z následujícího vyobrazení. /obr. 4.4. - 1/.

7	6	5	4	3	2	1	Ø
S	Z	X	H	X	P/V	N	C

obr. 4.4. - 1

kde:

C	indikátor přenosu
N	-"- sčítání/odčítání
P/V	-"- parity /přetečení
H	-"- polovičního přenosu
Z	-"- nuly
S	-"- znaménka
X	nepoužito

A./ I N D I K Á T O R P Ř E N O S U

C

Indikátor přenosu udává vznik přenosu ze znaménkového bitu při aritmetických operacích sčítání/odčítání. Dojde-li k přenosu ze znaménkového bitu, je bit C nastaven. Dále bude bit C nastaven instrukcí DAA za předpokladu, že jsou splněny podmínky pro převod do BCD kódu. Logickými operacemi OR, AND, XOR bude indikátor nulován. Je nastavitelný instrukcí SCF a potřebujeme-li provést jeho komplement, použijeme instrukcí CCF.

Poznámka: Pro rozlišení od registru C se někde používá označení CY.

B./ I N D I K Á T O R S Č Í T Á N Í / O D Č Í T Á N Í

N

Používá se pouze pro instrukci DAA. Udává, byla-li předcházející instrukce sčítání /N=Ø/ nebo odčítání /N=1/.

C./ I N D I K Á T O R P A R I T Y / P Ř E T E Č E N Í

P/V

Při aritmetických operacích dojde k nastavení tohoto bitu, jestliže výsledek aritmetické operace je větší než kapacita registru, v němž se aritmetická operace provádí.

Při logických operacích a rotacích tento bit zaznamenává paritu výsledku. Je-li parita sudá, je bit nastaven /P/V=1/, v opačném případě je nulován /P/V=Ø/.

Při instrukcích CPI, CPIR, CPD, CDDR a instrukcích blokového přenosu LDI, LDIR, LDD, LDDR zaznamenáváme tímto bitem stav čítače BC. Sníží-li hodnotu čítače na Ø, je bit P/V snulován, jinak /BC≠Ø/ zůstává nastaven.

Při instrukcích LD A,I a LD A,R bude P/V bit nastaven obsahem klopného obvodu uvolnění přerušení IFF2.

Při čtení byte z V/V zařízení instrukcí IN r,/C/ bude indikovat P/V bit paritu přenášeného byte.

D./ I N D I K Á T O R P O L O V I Č N Í H O P Ř E N O S U

H

Indikátor H se používá v souvislosti s instrukcí DAA. Indikuje přenos mezi 3. a 4. bitem při provádění aritmetických operací.

E./ I N D I K Á T O R N U L Y

Z

Osmibitové aritmetické a logické operace nastaví bit Z /Z=1/, je-li výsledek ve střadači nula. V opačném případě dojde ke snulování tohoto bitu /Z=Ø/.

Při instrukcích srovnávání bude tento bit nastaven při shodě srovnávaných veličin.
Při testování bitu bude indikátor Z obsahovat komplement testovaného bitu.
Při instrukcích INI, INA, OUTD bude indikátor Z nastaven v případě, že obsah registru B je 1.
Při instrukcích IN r, /C/ bude indikátor Z nastaven v případě nulového vstupu.

F./ I N D I K Á T O R Z N A M É N K A

S

V aritmetických operacích a některých vstupních operacích /IN r, /C// se zde ukládá stav nejvýznamnějšího bitu registru výsledku, udávající znaménko. Bude-li obsah registru záporný, bude bit S nastaven /S=1/.

1.5. S K U P I N A I N S T R U K C Í P R O P Ř E N O S D A T

Tato skupina instrukcí umožňuje přesuny dat mezi registry, mezi registry a pamětí nebo zásobníkem a popřípadě umožňuje provést výměnu dat mezi registry, nebo celých skupin registrů.

A./ P Ř E S U N D A T

LD I., II. I. ⟵ II. MOV r, r´

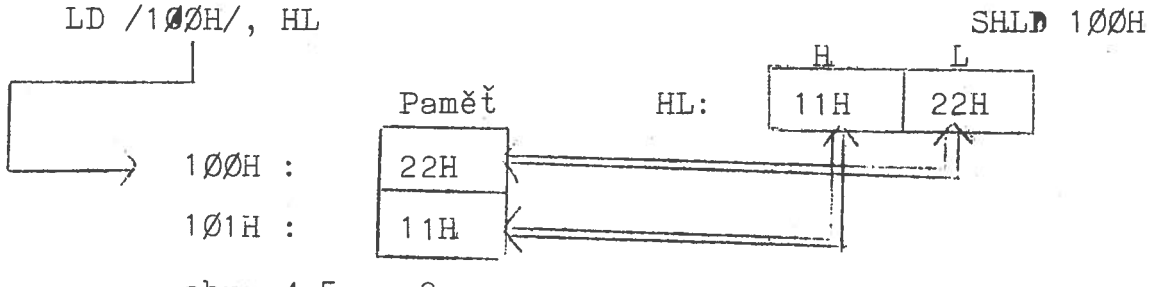
Instrukce LD se používá pro přesun jednoho nebo dvou byte dat z místa daného operandem II., na místo dané operandem I.

Nastavení indikátorů: nezměněno

Možné kombinace operandů: Je zřejmé, že všechny kombinace přenosů nemohou do instrukční množiny Z-80 patřit. Všechny povolené kombinace ukazuje tabulka 4.5 - 1.

Poznámka: Instrukce, které ukládají do paměti informaci velikosti 2 byte, ukládají jednotlivé byte v pořadí:
nižší byte, vyšší byte.

Příklad:
Máme uložit na adresu 1ØØH obsah registru HL. /obr. 4.5 - 2/



obr. 4.5. - 2

Mozne kombinace instrukce ld 1.,2.

2.\1.	(BC	DE	HL	IX+d	IY+d	NN)	A	B	C	D	E	H	L	I	R	IX	IY	SP	BC	DE	HL
(BC)	.	.	.	.	.	.	X	.	.	.	.	.	.	.	.	.	.	.	.	.	.
(DE)	.	.	.	.	.	.	X	.	.	.	.	.	.	.	.	.	.	.	.	.	.
(HL)	.	.	.	.	.	.	X	X	X	X	X	X	X	.	.	.	.	.	.	.	.
(IX+d)	.	.	.	.	.	.	X	X	X	X	X	X	X	.	.	.	.	.	.	.	.
(IY+d)	.	.	.	.	.	.	X	X	X	X	X	X	X	.	.	.	.	.	.	.	.
(NN)	.	.	.	.	.	.	X	.	.	.	.	.	.	.	.	X	X	X	X	X	X
A	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	.	.	.	.	.	.
B	.	.	X	X	X	.	X	X	X	X	X	X	X	.	.	.	.	.	.	.	.
C	.	.	X	X	X	.	X	X	X	X	X	X	X	.	.	.	.	.	.	.	.
D	.	.	X	X	X	.	X	X	X	X	X	X	X	.	.	.	.	.	.	.	.
E	.	.	X	X	X	.	X	X	X	X	X	X	X	.	.	.	.	.	.	.	.
H	.	.	X	X	X	.	X	X	X	X	X	X	X	.	.	.	.	.	.	.	.
L	.	.	X	X	X	.	X	X	X	X	X	X	X	.	.	.	.	.	.	.	.
F	.	.	.	.	.	.	X	.	.	.	.	.	.	.	.	.	.	.	.	.	.
R	.	.	.	.	.	.	X	.	.	.	.	.	.	.	.	.	.	.	.	.	.
N	.	.	X	X	X	.	X	X	X	X	X	X	X	.	.	.	.	.	.	.	.
IX	.	.	.	.	.	X	.	.	.	.	.	.	.	.	.	.	X	.	.	.	.
IY	.	.	.	.	.	X	.	.	.	.	.	.	.	.	.	.	X	.	.	.	.
SP	.	.	.	.	.	X	.	.	.	.	.	.	.	.	.	.	X	.	.	.	.
BC	.	.	.	.	.	X	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.
DE	.	.	.	.	.	X	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.
HL	.	.	.	.	.	X	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.
NN	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	X	X	X	X	X	X

Tabulka 4.5.-1. Mozne kombinace operandu instrukce LD

Poznamka : V dalsim textu budeme pouzivat techto symbolu :

n cislo velikosti 1 Byte
nn cislo velikosti 2 Byte
x kombinace je povolena

B./ V Y B Í R Á N Í Z Á S O B N Í K U

POP I. $I._H \leftarrow$ /SP+1/; $I._L \leftarrow$ /SP/;

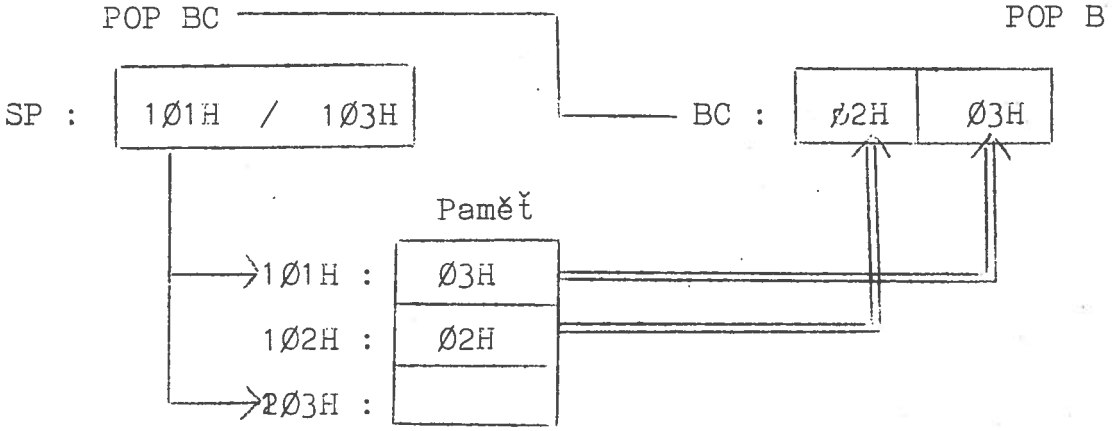
SP $\leftarrow$ SP+2

Instrukce přesune dva byte z vrcholu zásobníku do jednoho z indexregistrů nebo registrového páru tak, že vezme nižší byte z místa, kam ukazuje SP registr /ukazatel zásobníku/, inkrementuje SP registr, vezme vyšší byte a znovu inkrementuje SP registr, aby ukazoval na následující vrchol zásobníku.

Nastavení indikátorů: nezměněno

Možné operandy I. : IX, IY, HL, DE, BC, AF

Příklad: Vložme do registrového páru BC obsah vrcholu zásobníku /viz. obr. 4.5. - 3/.



obr. 4.5. - 3

C./ V K L Á D Á N Í D O Z Á S O B N Í K U

PUSH I. /SP-2/ $\leftarrow$ $I._L$; /SP-1/ $\leftarrow$ $I._H$;

SP $\leftarrow$ SP-2

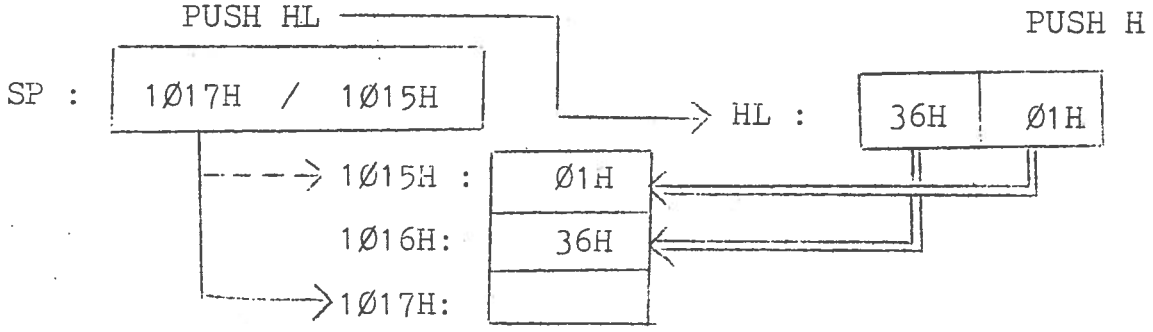
Instrukce přesune obsah registrového páru nebo indexregistru na vrchol zásobníku, určený obsahem SP registru /ukazatel zásobníku/. Adresa v SP registru ukazuje na poslední zaváděný byte do zásobníku.

Při provedení instrukce PUSH se SP registr napřed dekerementuje, pak se zavede vyšší byte, znovu se dekrementuje a zavede se nižší byte.

Nastavení indikátorů: nezměněno

Možné operandy : IX, IY, HL, DE, BC, AF.

Příklad: Vložme registrový pár HL do zásobníku /obr. 4.5.-4/



obr. 4.5. - 4

D./ V Ý M Ě N A

EX I., II. I. ⟷ II.

Instrukce Ex provádí vzájemnou výměnu dvou objektů, určených operandy I. a II.

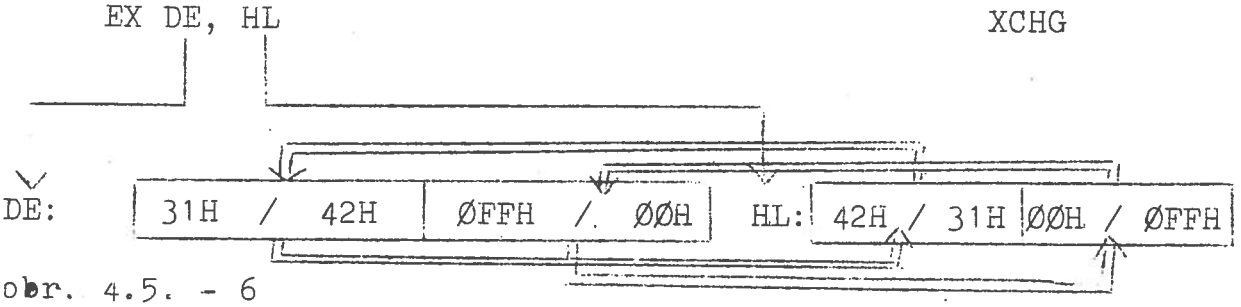
Nastavení indikátorů : nezměněno

Možné kombinace operandů jsou uvedeny v tab. 4.5. - 5.

I.	II.	HL	AF´	IX	IY
DE		x			
AF			x		
/SP/		x		x	x

Tab. 4.5. - 5 Možné kombinace operandů instrukce EX

Příklad: Proveďme záměnu registrů HL a DE /obr. 4.5. - 6/



obr. 4.5. - 6

E./ V Ý M Ě N A S K U P I N Y R E G I S T R Ů

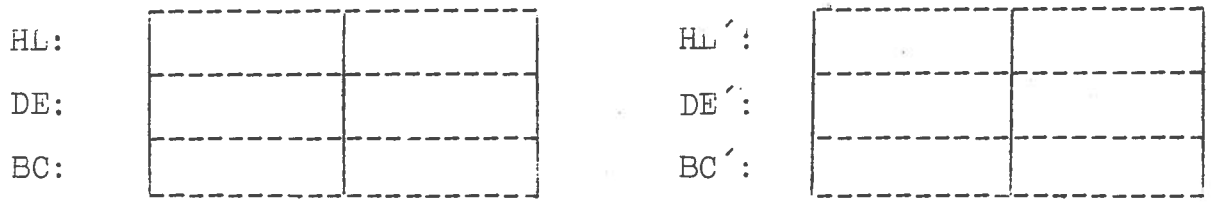
EXX HL ⟷ HL´; DE ⟷ DE´; BC ⟷ BC´

Instrukce EXX provede výměnu registrů HL, DE, BC za skupinu náhradních registrů HL, DE, BC.

Nastavení indikátorů: nezměněno

Příklad: Proveďme výměnu hlavních a náhradních registrů /obr. 4.5. - 7/

EXX



obr. 4.5. - 7

1.6. SKUPINA INSTRUKCÍ PRO BLOKOVÝ PŘENOS A VYHLEDÁVÁNÍ

Tyto instrukce slouží k přenášení bloku dat a k prohledávání bloku dat. Značně zjednodušují práci s většími bloky dat.

A./ BLOKOVÝ PŘENOS S INKREMENTACÍ BEZ OPAKOVÁNÍ

LDI /DE/ ⟵ /HL/ , DE ⟵ DE+1, HL ⟵ HL+1,
BC ⟵ BC+1

Instrukce provádí přenos jednoho byte dat z paměťové lokace, dané obsahem registrového páru HL, na paměťovou lokaci, danou obsahem registrového páru DE. Současně provádí inkrementaci registrového páru DE, registrového páru HL a dekrementaci registrového páru BC.

Nastavení indikátorů:
S nezměněn
Z -"-
H nulován
P/V nastaven, je-li BC-1≠Ø
N nulován
C nezměněn

Příklad:

Posunejme data v paměti o jeden byte "dolů", počínaje adresou 1ØØØH /max 1ØØØ byte/, až nalezneme bytes obsahem ØFFH.

```
      LD    BC, 1ØØØ      ; nastavení počítadla
      LD    HL, 1ØØØH     ;   -"-     počáteční adresy
      LD    DE, ØFFFH     ;   -"-     cílové adresy
      LD    A, ØFFH       ;   -"-     hodnoty hledaného byte
ZNOVU:
      CP    HL            ; porovnání
      JR    Z, KONEC      ; souhlasí-li, konec
      LDI                 ; nesouhlasí-li, přenesení byte
      JP    PE, ZNOVU     ; není-li 1ØØØ přenosů, znovu
KONEC:
```

B./ BLOKOVÝ PŘENOS S INKREMENTACÍ A OPAKOVÁNÍM

LDIR /DE/ ⟵ /HL/; DE ⟵ DE+1; HL ⟵ HL+1;
BC ⟵ BC-1;
opakování až do stavu BC = Ø.

Instrukce přenese obsah paměťové lokace, jejíž adresa je v registrovém páru DE. Inkrementuje registrový pár DE, registrový pár HL a dekrementuje registrový pár BC. Tuto činnost opakuje tak dlouho, dokud je obsah registrového páru BC různý od nuly. /viz. obr. 4.6. -1/. Je-li obsah registrového páru BC roven nule, CPU pokračuje instrukcí bezprostředně následující.

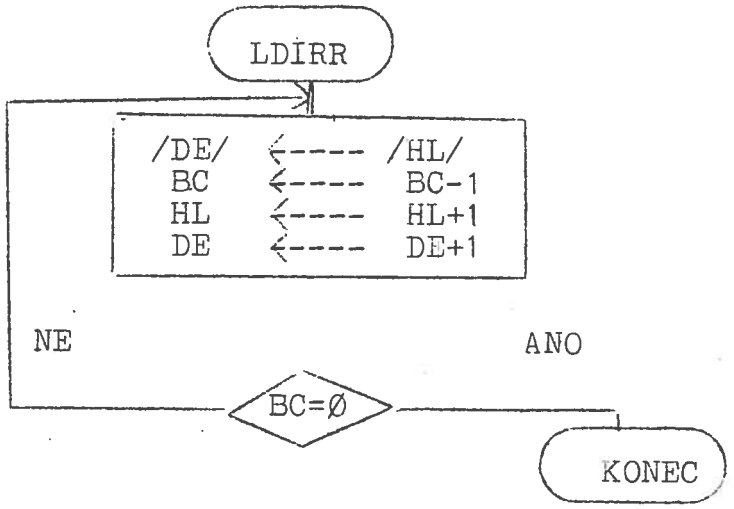
Poznámka: Je-li před provedením instrukce nastaven registrový pár BC na nulu, instrukce LDIR se bude provádět průchodem přes 64 kbyte.

Nastavení indikátorů:

S	nezměněn
Z	-"-
H	nulován
P/V	-"-
N	-"-
C	nezměněn

Příklad: Mějme za úkol snulovat paměťovou oblast, počínaje adresou ØAØØØH a konče adresou ØFFFFH.

```
LD HL, ØAØØØH          ; nastavení zdrojové adresy
LD DE, ØAØØ1H          ;    -"-    cílové adresy
LD BC, ØFFFH-ØAØØØH    ;    -"-    počítadla
LD A, Ø                ;
LD /ØAØØØH/,A          ; nulování první lokace
LDIR                   ; přenesení nul do celého pole
```



obr. 4.6. - 1

C./ BLOKOVÝ PŘENOS S DEKREMENTACÍ BEZ OPAKOVÁNÍ

LDD /DE/ <— /HL/; DE <— DE-1; HL <— HL-1;
BC <— BC-1

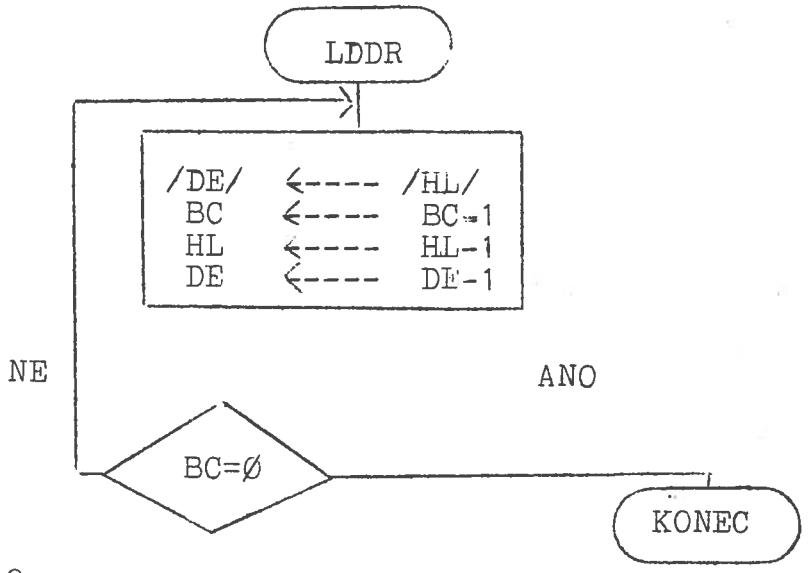
Instrukce provádí přenos obsahu paměťové lokace, jejíž adresa je obsažena v registrovém páru HL, na paměťovou lokaci, jejíž adresu obsahuje registrový pár DE. Provádí dekrementaci registrového páru DE, registrového páru HL a registrového páru BC.

Nastavení indikátorů:

S	nezměněn
Z	-"-
H	nulován
P/V	nastaven, je-li BC-1≠Ø
N	nulován
C	nezměněn

Příklad: nulujme blok dat od adresy ØFFFH "dolů" až po nalezení znaku 'A' nebo až po adresu 1ØØØH.

```
        LD A, Ø                 ; nulování registru A
        LD /ØFFFFH/, A          ;   -"-    první adresy
        LD A,  A                ; srovnávací znak
        LD BC, ØFFFFH-1ØØØH     ; nastavení počítadla
        LD HL, ØFFFFH           ;   -"-    počáteční adresy
        LD DE, ØFFFFH           ;   -"-    cílové adresy
ZNOVU:  EX DE, HL
        CP /HL/                 ; komparace
        EX DE, HL
        JR Z, KONEC             ; je-li nalezen srovnávací
                                  znak, tak skok na konec
        LDD                     ; nenalezen, nulování
        JP PE, ZNOVU            ; konec bloku
KONEC:                          ; ano, pokračujeme dále
```



obr. 4.6. - 2

D./ BLOKOVÝ PŘENOS S DEKREMENTACÍ A OPAKOVÁNÍM

LDDR /DE/ <--- /HL/, DE <-- DE-1, HL <--- HL-1,
BC <-- BC-1
opakování až do stavu BC = Ø.

Instrukce provede přenos paměťové lokace, jejíž adresu obsahuje registrový pár HL, na paměťovou lokaci, jejíž adresu obsahuje registrový pár DE. Provede dekrementaci registrového páru DE, registrového páru HL a registrového páru BC. Tuto činnost opakuje tak dlouho, dokud obsah registrového páru BC není roven nule.

Jestliže obsah registrového páru BC je roven nule, CPU pokračuje instrukcí bezprostředně následující. Vývojový diagram instrukce je na obr. 4.6. - 2.

Nastavení indikátorů:

S	nezměněn
Z	-"-
H	nulován
P/V	-"-
N	-"-
C	nezměněn

Příklad: Přenesme blok dat, počínaje adresou ØAØH a konče adresou 1ØØØH o 5 byte "nahoru".

```
LD HL, 1ØØØH                    ; nastavení počáteční adresy
LD DE, 1ØØ5H                    ;   -"-      cílové adresy
LD BC, 1ØØØH-ØAØH+1             ;   -"-      počítadla
LDDR                            ; provedení přenosu
```

E./ BLOKOVÉ PROHLEDÁVÁNÍ S INKREMENTACÍ BEZ OPAKOVÁNÍ

CPI A - /HL/; HL <--- HL+1; BC <--- BC - 1

Instrukce CPI provede odečtení obsahu paměťové lokace, jejíž adresa je obsažena v registrovém páru HL, od registru A. Provede inkrementaci registrového páru HL a dekrementaci registrového páru BC.

Nastavení indikátorů:

S	nastaven, je-li výsledek záporný
Z	-"- , -"- Ø/A = /HL//
H	-"- , je-li přenos mezi 3. a 4. bitem
P/V	-"- , je-li BC-1 ≠ Ø
N	-"-
C	nezměněn

Příklad: Máme otestovat, je-li od paměťové lokace 1ØØH uložen textový řetězec 'A' znak. V případě, že ne, provést skok na návěští ERROR.

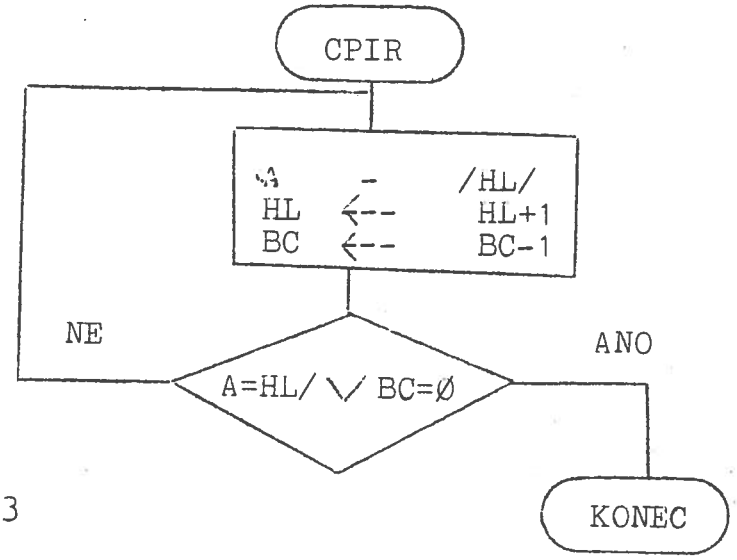
```
LD HL, 1ØØH                     ; nastavení adresy prvního znaku
LD A, 'Z'                       ;   -"-      prvního znaku
CPI                             ; komparace
JP NZ, ERROR                    ; v případě nerovnosti odskok
                                  na ERROR
LD A, 'N'                       ; další znak
CPI                             ; komparace
JP NZ, ERROR                    ;
LD A, 'A'                       ; další znak .... atd
atd ....
```

F./ PROHLEDÁVÁNÍ BLOKU S INKREMENTACÍ A OPAKOVÁNÍM

CPIR A-/HL/; HL <-- HL+1; BC <--- BC-1;
opakování až do stavu BC = Ø ∨ A = /HL/

Instrukce provádí odečtení paměťové lokace, jejíž adresu obsahuje registrový pár HL, od obsahu registru A. Inkrementuje registrový pár HL a dekrementuje registrový pár BC. Jestliže obsah registrového páru BC je různý od nuly a zároveň výsledek posledního odečtení nebyl roven nule, celá činnost se opakuje.

Je-li obsah registrového páru BC roven nule, nebo byl-li výsledek posledního odečítání roven nule, pokračuje se instrukcí bezprostředně následující. Vývojový diagram instrukce je na obr. 4.6. - 3.



obr. 4.6. - 3

Nastavení indikátorů: S nastaven, je-li výsledek záporný
Z -"- , je-li A = /HL/
H -"- , při přenosu mezi 3.a 4. bitem
P/V -"- , je-li BC-1 ≠ Ø
N -"-
c nezměněn

Příklad: Máme nalézt první znak 'E' v bloku dat, počínaje adresou 2ØH a konče adresou 1ØØH. Nenalezneme-li tento znak, pak provést skok na návěští NENI.

```
LD A, 'E'          ; nastavení srovnávacího znaku
LD HL, 2ØH         ;   -"-     první adresy
LD BC, 1ØØH-2ØH+1  ;   -"-     počítadla
CPIR               ; provedení hledání
JR NZ, NENI        ; testování, byl-li znak
                     nalezen
```

G./ PROHLEDÁVÁNÍ BLOKU S DEKREMENTACÍ BEZ OPAKOVÁNÍ

CPD A-/HL/; HL <-- HL-1; BC <-- BC-1

Instrukce provede odečtení paměťové lokace, jejíž adresa je v registrovém páru HL, od obsahu registru A. Provede dekrementaci registrového páru HL a registrového páru BC.

Nastavení indikátorů: S nastaven, je-li výsledek záporný
Z -"- , je-li A = /HL/
H -"- , při přenosu mezi 3.a4. bitem
P/V -"- , je-li BC-1 ≠ Ø
N -"-
C nezměněn

Příklad: Zkontrolujeme, je-li blok dat, počínaje lokací 1ØØØH a konče lokací 2ØØØH, nulový. Není-li nulový, nechť je proveden odskok na návěští ERROR.

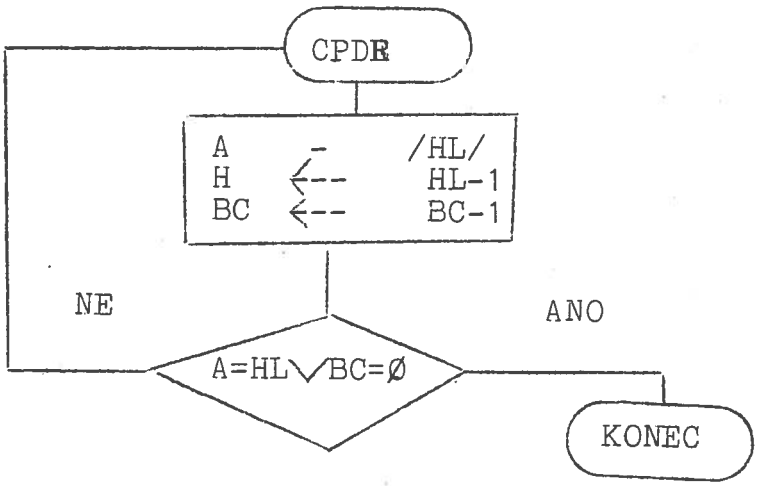
```
        LD HL, 2ØØØH        ; nastavení adresy paměti
        LD BC, 1ØØ1H        ;   -"-     počítadla
        LD A, Ø             ;   -"-     srovnávacího byte
ZNOVU:  CPD                 ; komparace
        JR NZ, ERROR        ; není-li nula, skok na ERROR
        J P PE, ZNOVU       ; není-li konec, skok na ZNOVU
```

H./ PROHLEDÁVÁNÍ BLOKU S DEKREMENTACÍ A OPAKOVÁNÍM

CPDR A- /HL/, HL <-- HL-1, BC <-- BC-1 CCDR
opakování až do stavu BC = Ø ∨ A = /HL/

Instrukce provádí odečtení paměťové lokace, jejíž adresu obsahuje registrový pár HL, od obsahu registru A. Provede dekrementaci registrového páru HL a dekrementaci registrového páru BC. Tuto činnost opakuje, dokud není obsah registrového páru BC roven nule, nebo dokud není výsledek posledního odečítání roven nule.

Je-li obsah registrového páru BC roven nule, nebo je-li výsledek posledního odečítání roven nule, CPU pokračuje instrukcí bezprostředně následující. /viz. obr. 4.6.-4/



obr. 4.6.-4

Nastavení indikátorů: S nastaven, je-li výsledek záporný
Z -"- , je-li A = /HL/
H -"- , je-li přenos mezi 3. a 4. bitem
P/V -"- , je-li BC-1 ≠ Ø
N -"-
C nezměněn

Příklad: Hledejme poslední znak 'D' v operační paměti. Pokud žádný nenalezneme, skok na návěští ERROR.

```
LD  A, 'D'          , nastavení srovnávacího znaku
LD  HL, ØFFFFH      ,   -"-     adresy
LD BC , Ø           ,   -"-     počítadla
CPDR                , provádění hledání
JP  NZ, ERROR       , test nalezení znaku
```

1.7. SKUPINA INSTRUKCÍ PRO ARITMETICKÉ A LOGICKÉ OPERACE

Aritmetické operace CPU Z-80 umožňují práci s 8 bitovým registrem, nebo 16 bitovým registrem či registrovým párem. Instrukce DAA umožňuje pracovat i v BCD kódu.

A./ A R I T M E T I C K É S Č Í T Á N Í

ADD I., II. I. <-- I. + II.

Instrukce ADD provádí součet dvou operandů I. a II. a uloží jej na místě operandu I.

Nastavení indikátorů:

8 bitová operace ADD	16 bitová operace ADD
S nastaven, je-li výsledek záporný	nezměněn
Z -"- , je-li výsl. nula	-"-
H -"- , je-li přenos mezi 3. a 4. bitem	nastaven, je-li přenos z 11. bitu
P/V -"- v případě přetečení	nezměněn
N nulován	-"-
C nastaven při přenosu ze znaménkového bitu	nastaven při přenosu ze znaménkového bitu

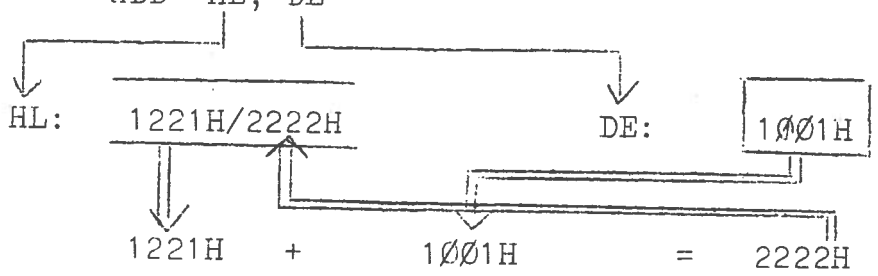
Možné kombinace operandů ukazuje tabulka 4.7. - 1.

I. II.	IX	IY	DE	HL	BC	SP	/HL/	/IX+d/	/IY+d/	A	B	C	D	E	H	L	n
A							x	x	x	x	x	x	x	x	x	x	x
HL			x	x	x	x											
IX	x		x		x	x											
IY		x	x		x	x											

Tab. 4.7. - 1 Možné kombinace operandů instrukce ADD.

Příklad: Máme sečíst obsah registrových párů HL a DE a výsledek uložit do registrového páru HL /obr. 4.7.-2/.

ADD HL, DE DADD



obr. 4.7. - 2

B./ A R I T M E T I C K É S Č Í T Á N Í S P Ř E N O S E M

ADC I., II. I. <-- I. + II. + CY

Instrukce ADC provede součet operandu I. s operandem II. a s indikátorem C. Výsledek uloží na místo operandu I.

Nastavení indikátorů:

S nastaven, je-li výsledek záporný
Z -"- , -"- nula
H -"- , je-li přenos mezi 3. a 4. bitem /u 8 bitové/, nebo mezi 11. a 12.bitem /u 16 bitové/
P/V -"- v případě přetečení
N nulován
C nastaven při přenosu ze znaménkového bitu

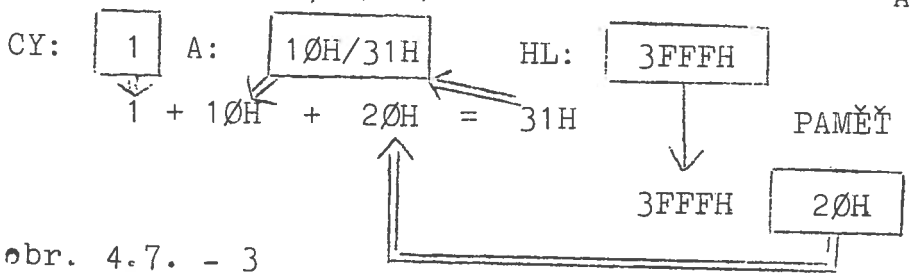
Možné kombinace operandů jsou uvedeny v tabulce 4.7. - 5.

I. II.	n	/HL/	/IX+d/	/IY+d/	A	B	C	D	E	H	L	BC	DE	HL	SP
A	x	x	x	x	x	x	x	x	x	x	x				
HL												x	x	x	x

Tab. 4.7. - 5 Možné kombinace operandů instrukce ADC.

Příklad: Proveďte součet registru A s obsahem paměťové lokace, jejíž adresu obsahuje registrový pár HL. V případě, že byl indikátor C nastaven, výsledek zvětšete o jedničku. /viz. obr. 4.7. - 3/

ADC A, /HL/ ADC Am M



obr. 4.7. - 3

G./ A R I T M E T I C K É O D Č Í T Á N Í

SUB I. A <-- A - I.

Instrukce SUB provádí odčítání operandu I. od obsahu registru A a výsledek uloží v registru A.

Nastavení indikátorů:

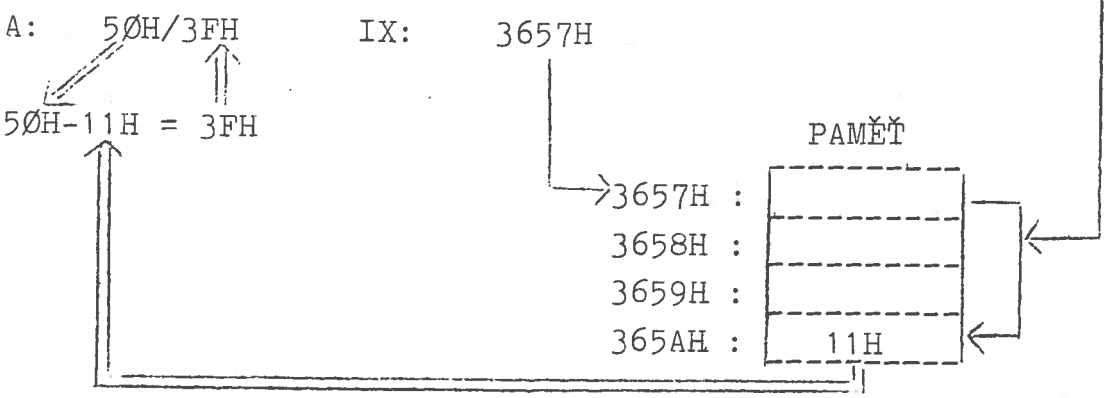
S	nastaven,	je-li výsledek záporný
Z	-"-	-"- nula
H	-"-	při přenosu mezi 3. a 4. bitem
P/V	-"-	v případě přetečení
N	-"-	
C	-"-	při přenosu ze znaménkového bitu

Možné operandy I.: /HL/; /IX+d/; /IY+d/; A; B; C; D; E; H; L; n

Příklad: Od registru A máme odečíst obsah paměťové lokace s adresou o tři větší, než je obsah registru IX. /viz. obr. 4.7. - 4/.

SUB A, /X/

SUB /IX+3/ -------- 3H+5657H = 365AH



obr. 4.7. - 4

D./ A R I T M E T I C K É O D Č Í T Á N Í S P Ř E N O S E M

SBC I., II. I. <--- I.-II. - CY

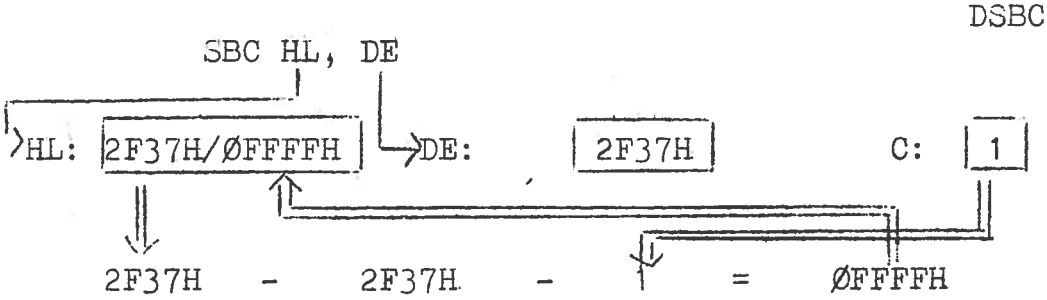
Instrukce SBC provede odečtení druhého operandu II. a indikátoru C od prvního operandu I. a výsledek uloží na místo prvního operandu I.

Nastavení indikátorů:

S	nastaven,	je-li výsledek záporný
Z	-"- ,	-"- nula
H	-"- ,	je-li přenos mezi 3. a 4. bitem, nebo 11. a 12.bitem
P/V	-"-	v případě přetečení
N	-"-	
C	-"-	při přenosu ze znaménkového bitu

Možné kombinace operandů jsou uvedeny v tabulce 4.7. - 5.

Příklad: Máme odečíst od registrového páru HL registrový pár DE. V případě, že byl indikátor C nastaven, zmenšete výsledek o jedničku /viz. obr. 4.7. - 6/.



obr. 4.7. - 6

E./ L O G I C K Ý S O U Č I N

AND I. A <--- A /\ I.

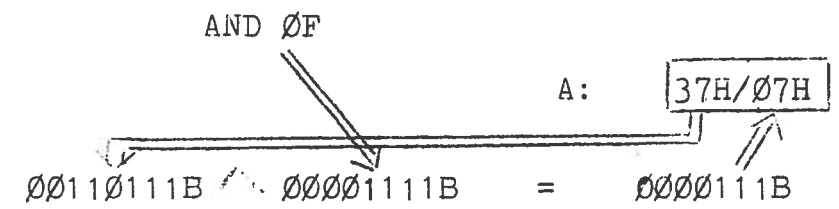
Instrukce AND provádí logický součin mezi registrem A a operandem I. tak, že provede logický součin sobě odpovídajících bitů a výsledek uloží vždy do příslušného bitu registru A.

Je-li bit nastaven, odpovídá logické hodnotě True, je-li nulován, odpovídá logické hodnotě False.

Nastavení indikátorů:
S nastaven, je-li výsledek záporný
Z -"- , -"- nulový
H -"-
P/V -"- v případě sudé parity
N nulován
C -"-

Možné operandy: /HL/; /IX+d/; /IY+d/; A; B; C; D; E; H; L; n

Příklad: Proveďme snulování vyšších čtyř bitů registru A a nižší čtyři bity ponechejme nezměněny. /obr. 4.7. - 7/.



obr. 4.7. - 7

F./ L O G I C K Ý S O U Č E T

OR I. A <--- A \/ I.

Instrukce OR provádí logický součet mezi registrem A a operandem I. tak, že provede logický součet sobě odpovídajících bitů a výsledek uloží vždy do odpovídajícího bitu registru A.

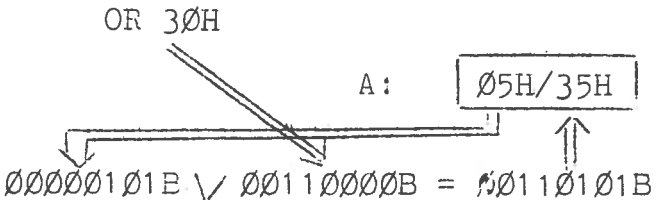
Je-li bit nastaven, odpovídá logické hodnotě True, je-li nulován, odpovídá logické hodnotě False.

Nastavení indikátorů: S nastaven, je-li výsledek záporný
Z -"- -"- nula
H nulován
P/V nastaven v případě sudé parity
N nulován
C nulován

Možné operandy: /HL/; /IX+d/; /IY+d/; A; B; C; D; E; H; L; n

Příklad: K binárnímu číslu Ø - 9 uloženému v registru A, přidejme do prvních čtyř bitů číselnou kombinaci ØØ11, aby vznikl znak, odpovídající číslu v registru A. /viz. obr. 4.7. - 8/.

ORI 3ØH



obr. 4.7. - 8

/Poznámka: vpravo jsou uvedeny instrukce pro mikroprocesor 8080/

G./ E X K L U Z I V N Í S O U Č E T

XOR I. A <--- A ⊕ I.

Instrukce XOR provádí exkluzivní součet mezi registrem A a operandem I. tak, že provede exkluzivní součet /součet modulo 2/ sobě odpovídajících bitů a výsledek uloží vždy do příslušného registru A.

Je-li bit nastaven, odpovídá logické hodnotě True, je-li nulován, odpovídá logické hodnotě False.

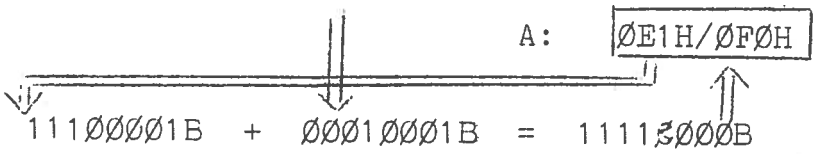
Nastavení indikátorů: S nastaven, je-li výsledek záporný
Z -"- -"- nula
H nulován
P/V nastaven v případě sudé parity
N nulován
C nulován

Možné operandy: /HL/; /IX+d/; /IY+d/; A; B; C; D; E; H; L; n

Příklad: Proveďme inverzi bitu Ø a 4 registru A /obr. 4.7-9/

XRI 11H

XOR 11H



obr. 4.7. - 9

H./ S R O V N Á N Í

CMP r, r´

CP I. A - I.

Instrukce CP provede odečtení operandu I. od registru A, ale výsledek nikam neuloží. O výsledku provedené operace nás informuje registr stavových indikátorů.

Nastavení indikátorů:
S nastaven, je-li výsledek záporný
Z -"- , -"- nula
H -"- , je-li přenos mezi 3. a 4. bitem
P/V -"- v případě přetečení
N -"-
C -"- při přenosu ze znaménkového bitu

Možné operandy: /HL/, /IX+d/, /IY+d/, A, B, C, D, E, H, L.

Příklad: Je-li obsah registru A písmeno ´A´, proveďme skok na návěští NAVA, je-li ´B´, skok na NAVB, je-li ´C´, skok na NAVC, jinak skok na návěští OSTATNI.

```
CP  Á´        ; srovnání s ´A´
JR Z, NAVA    ; je-li Z = 1, skok na NAVA
CP  B´        ; srovnání s ´B´
JR Z, NAVB    ; je-li Z = 1, skok na NAVB
CP  C´        ; srovnání s ´C´
JR Z, NAVC    ; je-li Z = 1, skok na NAVC
JR OSTATNI    ; jinak skok na OSTATNI
```

I./ I N K R E M E N T A C E

INR r

INC I. I. <--- I. + 1

Instrukce INC provede přičtení jedničky k operandu I. a výsledek uloží zpět na místo operandu I.

Nastavení indikátorů:

8 bitová operace	16 bitová operace
S nastaven, je-li výsledek záporný	nezměněn
Z nastaven, je-li výsledek nula	nezměněn

Nastavení indikátorů:

	8 bitová operace	16 bitová operace
S	nastaven, je-li výsledek záporný	nezměněn
Z	-"- -"- nula	-"-
H	-"- , je-li přenos mezi 3. a 4. bitem	-"-
P/V	-"- při přetečení	-"-
N	nulován	-"-
C	nezměněn	-"-

Možné operandy: /HL/, /IX+dP, /IY+d/, A, B, C, D, E, H, L, BC, DE, HL, IX, IY, SP

Příklad: Mějme zvětšit obsah paměťové lokace, v programu označené návěštím TADY, o 2.

```
        LD HL, TADY          ; nastavení HL na adresu
        INC HL               ; zvětšení o 1
        INC HL               ; zvětšení o 1
```

J./ D E K R E M E N T A C E

DCR r

DEC I. I. ---- I. - 1

Instrukce DEC provede odečtení jedničky od operandu I. a výsledek uloží zpět na místo operandu I.

Nastavení indikátorů:

	8 bitová operace	16 bitová operace
S	nastaven, je-li výsledek záporný	nezměněn
Z	-"- -"- nula	-"-
H	-"- , je-li přenos mezi 3. a 4. bitem	-"-
P/V	-"- při přetečení	-"-
N	-"-	
C	nezměněn	

Možné operandy: /HL/, /IX+d/, /IY+d/, A, B, C, D, E, H, L, BC, DE, HL, IX, IY, SP.

Příklad: Mějme provést odečtení 1 od registru A a je-li registr A po odečtení nula, skok na návěští DAL.

```
DEC A                    ; odečtení 1 od A
JR Z, DAL                ; je-li Z = 1, skok na DAL
```

K./ O P R A V A P R O B C D A R I T M E T I K U

DAA

Instrukce DAA provádí opravu ve střadači tak, aby výsledek předchozí aritmetické operace, prováděné v binárním kódu, odpovídal operaci, prováděné v BCD kódu.

Přesná činnost instrukce DAA je uvedena v tab. 4.7. - 10.

Operace	CY před DAA	Hodnota v A bit 7-4	H před DAA	Hodnota v A bit 3-0	Číslo připočtené k A	CY po DAA
	0	0-9	0	0-9	00	0
	0	0-8	0	A-F	06	0
ADD	0	0-9	1	0-3	06	0
	0	A-F	0	0-9	60	1
ADC	0	9-F	0	A-F	66	1
	0	A-F	1	0-3	66	1
INC	1	0-2	0	0-9	60	1
	1	0-2	0	A -F	66	1
	1	0-3	1	0-3	66	1
SUB	0	0-9	0	0-9	00	0
SBC	0	0-8	1	6-F	FA	0
DEC	1	7-F	0	0-9	A0	1
NEG	1	6-F	1	6-F	9A	1

Nastavení indikátorů:

S nastaven, je-li znaménkový bit střadače 1
Z nastaven, je-li obsah střadače nula
H nedefinován
P/V nastaven, jestliže střadač obsahuje po operaci sudou paritu
N nezměněn
C viz. tabulka

Příklad: Binárně sečítáme dvě čísla instrukcí ADD.

```
   00010101   =   15 D
+  00100111   =   27D
   ----------------------
   00111100   =   30 H
```

Vidíme, že interpretace v BCD kódu není možná a součet v BCD aritmetice je nesprávný. Provedeme instrukci DAA.

```
   00111100
+  00000110
   -----------
   01000010   =   42 D   =   15 D + 27 D
```

Po provedení instrukce DAA odpovídá BCD interpretace registru číslu 42, což je dekadický součet čísel 15 a 27.

L./ K O M P L E M E N T

CPL A ⟵ $\bar{A}$ CMA

Instrukce CPL provede jedničkový doplněk střadače A /Jedničky jsou nahrazeny nulami a naopak/.

Nastavení indikátorů: nezměněno.

Příklad: Obsah střadače A je 01101100. Po instrukci CPL bude obsah střadače A 10010011.

M./ N E G A C E

NEG A <--- Ø - A

Instrukce NEG provede negaci střadače /odečtení obsahu střadače od nuly, neboli změnu znaménka/.

Nastavení indikátorů:

S nastaven, je-li výsledek záporný
Z -"- -"- nula
H -"- , je-li přenos mezi 3. a 4. bitem
P/V -"- , byl-li obsah střadače před operací 8ØH
N nastaven
C nastaven, jestliže obsah střadače před operací není nula.

Příklad: Obsah střadače A je 00000001.
Po operaci NEG bude obsah střadače A 11111111.

1.8. SKUPINA ŘÍDÍCÍCH INSTRUKCÍ

Tato skupina instrukcí slouží k nastavení režimu CPU, povolení a znemožnění přerušení, zastavení CPU a práci se stavovým indikátorem C.

A./ I N V E R S E I N D I K Á T O R U C

CCF CY <--- $\overline{CY}$ CMC

Instrukce CCF provede inverzi indikátorového bitu C v registru F.

Nastavení indikátorů:
S nezměněn
Z nezměněn
H převezme předchozí nastavení C
P/V nezměněn
N nulován
C invertován

B./ N A S T A V E N Í I N D I K Á T O R U C

SCF CY <--- 1 STC

Instrukce SCF provede nastavení indikátoru přenosu C.

Nastavení indikátorů:
S nezměněn
Z nezměněn
H nulován
P/V nezměněn
N nulován
C nastaven

C./ P R Á Z D N Á O P E R A C E

NOP

Instrukce NOP nemá za následek žádnou činnost.

Nastavení indikátorů : nezměněno

d./ Z A S T A V E N Í

HALT HLT

Instrukce HALT má za následek zastavení činnosti CPU a provádění instrukce NOP až do přijetí přerušení, nebo vynulování CPU.

Nastavení indikátorů: nezměněno

E./ Z A M A S K O V Á N Í P Ř E R U Š E N Í

DI IFF <--- Ø

Instrukce DI vynuluje klopný obvod IFF pro povolení maskovatelného přerušení, tím znemožní přijetí maskovatelného přerušení až do jeho uvolnění instrukcí EI.

Nastavení indikátorů: nezměněno

F./ U V O L N Ě N Í P Ř E R U Š E N Í

EI IFF <--- 1

Instrukce EI provede nastavení klopného obvodu IFF pro povolení maskovatelného přerušení a tím uvolní přerušení.

Poznámka : Přerušení může být přijato až po dokončení následující instrukce.

Nastavení indikátorů: nezměněno

G./ N A S T A V E N Í R E Ž I M U P Ř E R U Š E N Í

IM I.

Instrukce IM slouží k nastavení režimu maskovatelného přerušení. I. může být číslice Ø, 1 nebo 2. Tím dojde k nastavení CPU na režim přerušení Ø, 1 nebo 2.

Nastavení indikátorů: nezměněno

Příklad: Nastavení režimu přerušení "1".

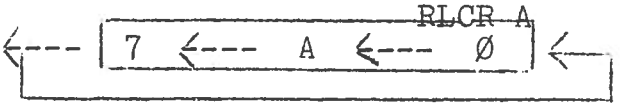
```
IM 1        ; nastavení režimu přerušení
```

1.9. SKUPINA INSTRUKCÍ ROTACÍ A POSUVU

Tato skupina instrukcí dovoluje pracovat s registrem, nebo vybranou paměťovou lokací jako s posloupností bitů, které můžeme posouvat či rotovat doleva, nebo doprava, popřípadě do posuvu nebo rotacezařadit indikátor C.

A./ R O T A C E S T Ř A D A Č E V L E V O , d o C Y

RLCA



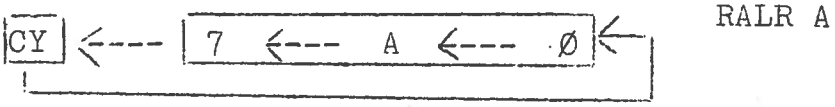
Instrukce RLCA provádí rotaci střadače A doleva. Sedmý bit se přesune do uvolněného bitu nula a také do indikátoru C.

Nastavení indikátorů:
A S nezměněn
Z -"-
H nulován
P/V nezměněn
N nulován
C obsahuje sedmý bit střadače

Příklad:
CY: Ø A: 10010110
Po provedení instrukce RLCA :
CY: 1 A: 00101101

B./ R O T A C E S T Ř A D A Č E V L E V O , přes C Y

RLA



Instrukce RLA provádí rotaci střadače A doleva, přes bit CY. Sedmý bit se přesouvá do indikátoru C a obsah indikátoru C do nultého bitu.

Nastavení indikátorů:
S nezměněn
Z -"-
H nulován
P/V nezměněn
N nulován
C obsahuje sedmý bit střadače

Příklad: CY: Ø A: 10010110
Po provedení instrukce RLA:
CY: 1 A: 00101100

C./ R O T A C E S T Ř A D A Č E V P R A V O do C Y

RRCA

Instrukce RRCA provádí rotaci střadače A doprava, do CY. Nultý bit se přesouvá do indikátoru C a do sedmého bitu střadače.

Nastavení indikátorů:
S nezměněn
Z -"-
H nulován
P/V nezměněn
N nulován
C obsahuje nultý bit střadače

Příklad: CY: Ø A: 10010110
Po provedení instrukce RRCA:
CY Ø A: 01001011

D / R O T A C E S T Ř A D A Č E V P R A V O přes C Y

RRA

RARR A

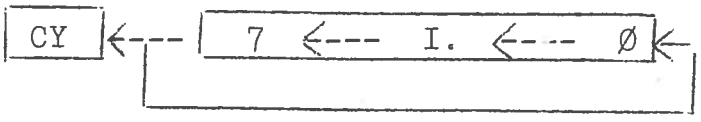
Instrukce RRA provádí rotaci střadače A vpravo, přes bit CY. Nultý bit se přesouvá do indikátoru C a stav indikátoru C do sedmého bitu střadače.

Nastavení indikátorů:
S nezměněn
Z -"-
H nulován
P/V nezměněn
N nulován
C obsahuje nultý bit střadače

Příklad: CY: Ø A: 10010110
Po provedení instrukce RRA:
CY: Ø A: 01001011

E./ R O T A C E O P E R A N D U V L E V O d o C Y

RLC I.



Instrukce RLC provádí rotaci operandu I. o jeden bit doleva. Sedmý bit se přesune do indikátoru C.

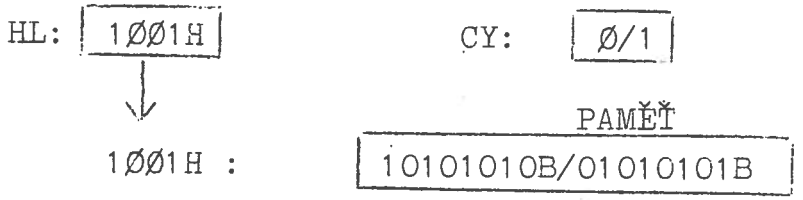
Nastavení indikátorů:
S nastaven, je-li výsledek záporný
Z -"- -"- nulový
H nulován
P/V nastaven v případě sudé parity
N nulován
C obsahuje sedmý bit operandu

Možné operandy: /HL/, /IX+d/, /IY+d/, A, B, C, D, H, E, L.

Příklad: Rorujme paměťovou lokaci, jejíž adresu obsahuje registrový pár HL o jeden bit vlevo /viz. obr. 4.9. - 1/.

RLOR M

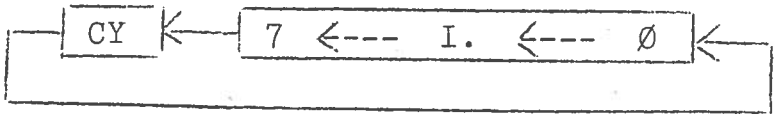
RLC /HL/



OBR. 4.9. - 1

F./ R O T A C E O P E R A N D U V L E V O přes C Y

RL I.



Instrukce RL provádí rotaci operandu I. vlevo, přes indikátor C. Sedmý bit operandu se přesune do indikátoru C a stav indikátoru C se přesune do nultého bitu operandu I.

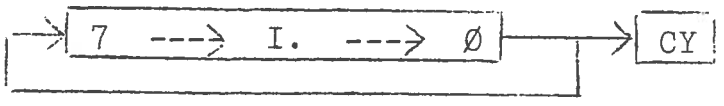
Nastavení indikátorů:
S nastaven, je-li výsledek záporný
Z -"- -"- nula
H nulován
P/V nastaven v případě sudé parity
N nulován
C obsahuje sedmý bit operandu I.

Možné operandy: /HL/, /IX+d/, /IY+d/, A, B, C, D, E, H, L.

Příklad: CY: 1 D: 10100110
Stav registrů po provedení instrukce RL D:
CY: 1 D: 01001101

G./ R O T A C E O P E R A N D U V P R A V O do C Y

RRC I.



Instrukce RRC provádí rotaci operandu I vpravo, do indikátoru CY. Nultý bit operandu se přesune do sedmého bitu operandu.

Nastavení indikátorů:
S nastaven, je-li výsledek záporný
Z nastaven, je-li výsledek nula
H nulován
P/V nastaven v případě sudé parity
N nulován
C obsahuje nultý bit operandu I.

Možné operandy: /HL/, /IX+d/, /IY+d/, A, B, C, D, E, H, L.

Příklad: CY: Ø E: 00001111
Stav registrů po provedení instrukce RR E:
CY: 1 E: 00000111

I./ A R I T M E T I C K Ý P O S U V V L E V O

SLAR A

SLA I. CY <--- 7 <--- I. <--- Ø <--- Ø

Instrukce SLA provádí aritmetický posuv operandu I. doleva, do indikátoru C. Sedmý bit operandu se přenese do indikátoru C a nultý bit operandu se obsadí nulou.

Nastavení indikátorů:
S nastaven, je-li výsledek záporný
Z -"- -"- nula
H nulován
P/V nastaven v případě sudé parity
N nulován
C obsahuje sedmý bit operandu I.

Možné operandy: /HL/, /IX+d/, /IY+d/, A, B, C, D, E, H, L.

Příklad: CY: 1 L: 00111011
Stav registrů po provedení instrukce SLA L:
CY: Ø L: 01110110

J./ A R I T M E T I C K Ý P O S U V V P R A V O

SRAR A

SRA I. ⟲ 7 ---> I. ---> Ø ---> CY

Instrukce SRA provádí aritmetický posuv operandu I. vpravo, do indikátoru C. Nultý bit operandu se přesune do indikátoru C a sedmý bit operandu zůstane zachován.

Nastavení indikátorů:
S nastaven, je-li výsledek záporný
Z -"- -"- nula
H nulován
P/V nastaven v případě sudé parity
N nulován
C obsahuje nultý bit operandu I.

Možné operandy: /HL/, /IX+d/, /IY+d/, A, B, C, D, E, H, L

Příklad: CY: 1 A: 10000000
Stav registrů po provedení instrukce SRA A:
CY: Ø A: 11000000

K./ P O S U V V P R A V O

SRLR A

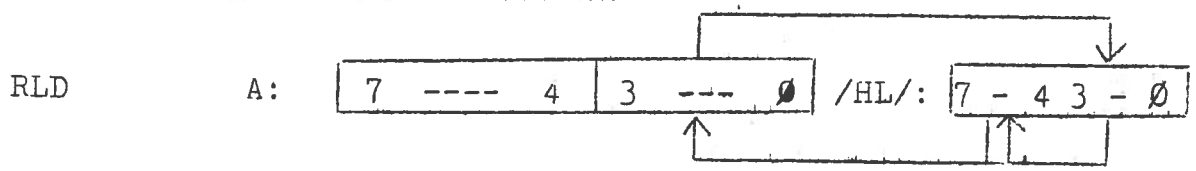
SRL I. Ø ---> [7 ---> I. ---> Ø] ---> [CY]

Instrukce SRL provádí posuv operandu I. vpravo. Bit Ø se přesouvá do indikátoru C a bit 7 se plní nulou.

Nastavení indikátorů:
- S nastaven, je-li výsledek záporný
- Z -"- -"- nula
- H nulován
- P/V nastaven v případě sudé parity
- N nulován
- C obsahuje bit Ø operandu I.

Příklad: CY: 0 B: 10101111
Stav registrů po provedení instrukce SRL:
CY: 1 B: 01010111

L./ B C D R O T A C E V L E V O



Instrukce RLD provádí BCD rotaci mezi nižší polovinou byte registru A a obsahem paměťové lokace, jejíž adresa je obsažena v registrovém páru HL. Nižší polovina byte registru A je přenesena do nižší poloviny byte paměťové lokace, nižší polovina byte paměťové lokace je přenesena do vyšší poloviny byte a vyšší polovina byte paměťové lokace je přenesena do nižší poloviny byte registru A.

Nastavení indikátorů:
- S nastaven, je-li obsah střadače záporný
- Z nastaven, je-li obsah střadače nula
- H nulován
- P/V nastaven, má-li střadač sudou paritu
- N nulován
- C nezměněn

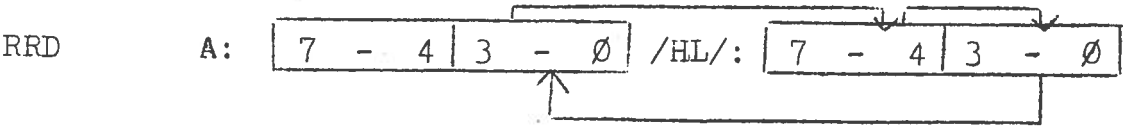
Paměť

Příklad: A: 56H HL: Ø3ØØH Ø3ØØH: 2ØH

Stav registrů a paměti po provedení instrukce RLD:

A: 52H HL: Ø3ØØH Ø3ØØH: Ø6H

M./ B C D R O T A C E V P R A V O



Instrukce RRD provádí BCD rotaci mezi nižší polovinou byte registru A a obsahem paměťové lokace, jejíž adresa je obsažena v registrovém páru HL. Nižší polovina byte střadače je přenesena do vyšší poloviny byte paměťové lokace, vyšší polovina byte paměťové lokace je přenesena do nižší poloviny byte a nižší polovina byte paměťové lokace je přenesena do nižší poloviny byte střadače.

Nastavení indikátorů:
- S nastaven, je-li obsah střadače záporný
- Z nastaven, je-li obsah střadače nula
- H nulován
- P/V nastaven, má-li střadač sudou paritu
- N nulován
- C nezměněn

Příklad: A: 31H HL: 3ØØØH 3ØØØH: Paměť FFH

Stav registrů a paměti po provedení instrukce RRD:

A: 3FH HL: 3ØØØH 3ØØØH: 1FH

1.10. SKUPINA INSTRUKCÍ PRO PRÁCI S BITEM

Následující skupina instrukcí umožňuje testovat, nastavovat, nebo nulovat kterýkoliv bit registru, nebo paměťové lokace.

A./ T E S T B I T U

BIT b, II. $Z \leftarrow \overline{II._b}$

Instrukce BIT přenese komplement bitu "b" operandu II. do indikátorového bitu Z.

Nastavení indikátorů:
- S nedefinován
- Z obsahuje komplement specifikovaného bitu
- H nastaven
- P/V nedefinován
- N nulován
- C nezměněn

Možné kombinace operandů: Jsou možné všechny kombinace pro b = Ø až 7 a operand II.: /HL/, /IX+d/, /IY+d/, A, B, C, D, E. H, L.

Příklad: Z: Ø HL: ØFFFFH ØFFFFH: 10001000

Stav registrů po provedení instrukce BIT 2, HL:

Z: 1 HL: ØFFFH ØFFFFH: 10001000

paměť

B./ N A S T A V E N Í B I T U

SET b, II. II.$_b$ <--- 1

Instrukce SET provede nastavení bitu "b" operandu II.

Nastavení indikátorů: nezměněno

Možné kombinace operandů:
Jsou možné všechny kombinace pro b = Ø až 7 a operand II.
/HL/, /IX+d/, /IY+d/, A, B, C, D, E, H, L.

Příklad: IX: 1ØØØH

	Paměť
1ØØØH:	ØØH
1ØØ1H:	ØØH

Stav registrů a paměti po provedení instrukce SET Ø, /IX+1/
IX: 1ØØØH

1ØØØH:	ØØH
1ØØ1H:	Ø1H

C./ N U L O V Á N Í B I T U

RES b, II. II.$_b$ <--- Ø

Instrukce RES provede nulování bitu "b" operandu II.

Nastavení indikátorů: nezměněno

Možné kombinace operandů: Jsou možné všechny kombinace pro b = Ø až 7 a operand II.: /HL/, /IX+d/, /IY+d/, A, B, C, D, E, H, L.

Příklad: A: ØFFH
Stav registru po provedení instrukce RES 3,A
A: ØF7H

1.11. SKUPINA INSTRUKCÍ SKOKU

Následující skupina instrukcí umožňuje měnit obsah programového čítače a tím provádět skok v programu. Skok může být nepodmíněný /tvrdý/, nebo podmíněný stavovými indikátory.

A./ N E P O D M Í N Ě N Ý S K O K

JP I. PC <--- I.

PCHL
PCIX
PCIY
JMPnn

Instrukce JP provede zavedení operandu I. do registru PC. Výsledkem je skok na paměťovou lokaci, danou operandem I.

Nastavení indikátorů: nezměněno

Možné operandy: nn, /HL/, /IX/, /IY/

Poznámka: Je-li operandem některý z 16 bitových registrů, uzavřený v kulaté závorce, t.j. /HL/, /IX/, /IY/, neznamená to, že operandem /adresou odskoku/ je obsah paměťové lokace, jejíž adresa je v registru, ale že operandem je přímo obsah registru.

Příklad: Proveďme skok na adresu, danou obsahem registrového páru HL. /viz. obr. 4.11. - 1/.

JP /HL/ PCHL

Ø47H - Ø49H:	LD HL, 3F47H
Ø4AH:	JP /HL/
3F47H:	
ØFØØØH - ØFØØ2H :	JP 3F47H

obr. 4.11. - 1

B./ P O D M Í N Ě N Ý S K O K

JNC JPO JM ...

JP I., II Jestliže I. je True, pak PC <--- II, jinak pokračuj

Podmíněná instrukce JP provede zavedení adresy, dané operandem II., do registru PC, pouze je-li splněna podmínka, daná operandem I. Není-li splněna podmínka, CPU pokračuje instrukcí bezprostředně následující.

Podmínka je dána předepsaným stavem těchto indikátorů:

Z, C, P/V, S /viz. tab. 4.11.-1/.

Nastavení indikátorů: nezměněno

Možné operandy: I.: NC, C, NZ, Z, PE, PO, M, P
II.: nn

Příklad: Je-li registr A nulový, máme pokračovat na adrese NUL. Je-li záporný, pokračovat na adrese NEGAT.

```
AND A       ; nastavení indikátorů
JP Z,NUL    ; odskok v případě nuly v A
JP M,NEGAT  ; odskok v případě A je menší než Ø
```

I.	Indikátor	Stav indikátoru	Stav, splňující podmínku
NZ	Z	nulován	výsledek nenulový
Z	Z	nastaven	výsledek nulový
NC	C	nulován	bez přenosu
C	C	nastaven	s přenosem
PO	P/V	nulován	lichá parita
PE	P/V	nastaven	sudá parita
P	S	nulován	výsledek nezáporný
M	S	nastaven	výsledek záporný

Tab. 4.11.-1 Tabulka podmínek podmíněných inst. JP, CALL a RET

C./ R E L A T I V N Í S K O K

JMPR e

JR I. PC <--- PC + e

Instrukce JR provede relativní skok na adresu, danou operandem I. Skok je relativní svou interpretací ve strojovém kódu. Má uložen pouze jeden byte operandu, proto odskok nesmí být větší jak -126 nebo + 129 byte /ne instrukcí/.

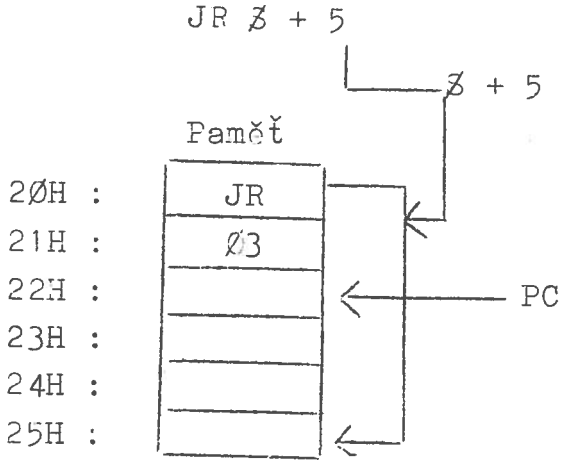
Nastavení indikátorů: nezměněno

Možné operandy: Operand I. je adresa odskoku /rozdíl "e" si překladač vypočte sám/.

Příklad:
```
TOP:    LD ...      ; začátek
        .
        .
        .
        JR TOP      ; návrat na začátek
```

Poznámka: K vyjádření běžné adresy /adresy instrukce, ve které je symbol uveden/ se používá symbolu $.

Příklad: Provečme skok o 5 byte vpřed, vzhledem k prvnímu byte instrukce /viz. obr. 4.11. - 2/.



obr. 4.11. - 2

D./ R E L A T I V N Í P O D M Í N Ě N Ý S K O K

JR I., II. jestliže I. je True, pak PC <--- PC + e, jinak pokračuj JRC ...

Podmíněná instrukce JR provede relativní skok na adresu danou operandem II., pouze je-li splněna podmínka, daná operandem I.

Není-li splněna podmínka, pokračuje se instrukcí bezprostředně následující. Skok je relativní svou interpretací ve strojovém kódu. Má uložen pouze jeden byte operandu, proto rozsah skoku nesmí být větší jak -126 a + 129 byte /ne instrukcí/.

Nastavení indikátorů: nezměněno

Možné operandy: I.: je podmínka /viz. tab. 5. - 2/
II.: adresa odskoku /rozdíl e si vypočte překladač sám/

I.	Indikátor	Stav indikátoru	Stav splňující podmínku
C	C	nastaven	s přenosem
NC	C	nulován	bez přenosu
Z	Z	nastaven	výsledek nulový
NZ	Z	nulován	výsledek nenulový

Tabulka 5. - 2

Příklad: Vytvořme čekací smyčku s délkou úměrnou obsahu registru A.

```
ZPET:  DEC A          ; zmenšení o jedničku
       JR NZ,ZPET     ; není-li A=Ø, znovu
```

E./ S K O K P O D M Í N Ě N Ý Č Í T A Č E

DJNZ I. B <--- B - 1 ; jestliže B=Ø, pak pokračuj,
jinak PC <--- PC + e

Instrukce DJNZ provádí dekrementaci registru B a v případě, že je obsah registru po dekrementaci různý od nuly, CPU provede skok na adresu, danou operandem I. Skok je interpretován jako relativní, proto má rozpětí -126 až +129 byte.

Nastavení indikátorů: nezměněno

Možné operandy: Adresa odskoku /překladač si rozdíl e vypočte sám/

Příklad: Vyvolejme 7 krát podprogram SUBR s parametrem v registru A, pohybujícím se od Ø do 6.

```
       LD B, 7        ; nastavení čítače cyklu
       LD A, Ø        ; nastavení prvního parametru
ZPET:  CALL SUBR      ; vyvolání podprogramu
       INC A          ; zvýšení parametru
       DJNZ ZPET      ; test na konec, jinak zpět
```

1.12. SKUPINA INSTRUKCÍ VOLÁNÍ A NÁVRATU

Pomocí těchto instrukcí je možné tvořit strukturu podprogramů. Při volání podprogramu se do zásobníku ukládá návratová adresa, je tedy možno jeden podprogram využívat více programy.

A./ V O L Á N Í P O D P R O G R A M U

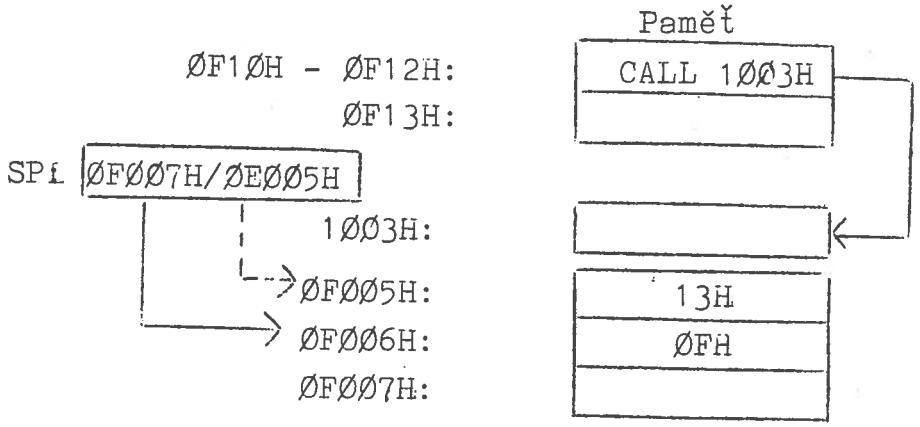
CALL I. /SP-1/ <--- PC_H ; /SP-2/ <--- PC_L ;
PC <--- I. ; SP <--- SP-2

Instrukce CALL provádí volání podprogramu. Návratová adresa se uloží do zásobníkové paměti tak, jako by se ukládaly instrukcé PUSH a do programového čítače se zavede adresa, daná operandem I.

Nastavení indikátorů: nezměněno

Možné operandy: Návěští, nebo absolutní adresa.

Příklad: Uložme adresu následující instrukce do zásobníku a proveďme skok do podprogramu o adrese 1ØØ3H. /viz. obr. 4.12. - 1./.



Obr. 4.12. - 1

B./ P O D M Í N Ě N É V O L Á N Í P O D P R O G R A M U

CALL I., II. jestliže I. je True, pak /SP-1/ <--- PC_H; /SP-2/ <--- PC_L; PC <--- II.; SP <--- SP-2, jinak pokračuj

Podmíněná instrukce CALL provádí skok do podprogramu, je-li splněna podmínka, daná operandem I. V opačném případě se pokračuje instrukcí bezprostředně následující.

Návratová adresa se při skoku do podprogramu uloží na vrchol zásobníku, stejně jako u nepodmíněné instrukce CALL.

Podmínka je dána předepsaným stavem jednoho z indikátorů Z, C, P/V, S /viz. tab. 4.11. - 1/.

Nastavení indikátorů: nezměněno

Možné kombinace operandů: Jsou povoleny všechny kombinace podmínak
I.: NZ, Z, NC, C, PO, PE, P, M a
operandu II: Absolutní adresa nebo návěští

Příklad: Je-li v registru A znak 'A', proveďte vyvolání podprogramu CHARA. V opačném případě pokračovat.

```
CP 'A'              ; srovnání s 'A'
CALL Z, CHARA       ; Z=1, volání CHARA
```

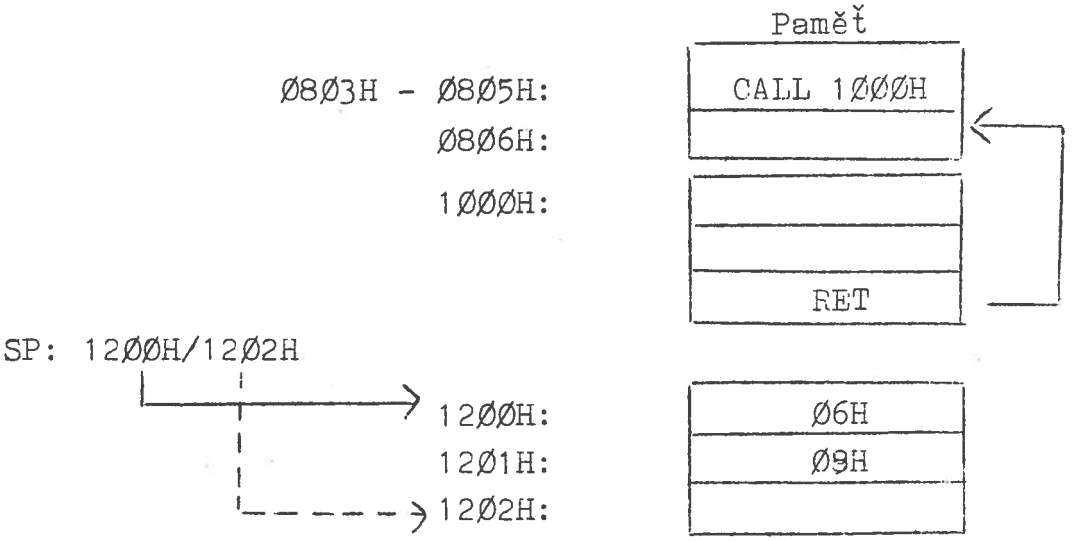
C./ N Á V R A T Z P O D P R O G R A M U

RET PC_L <--- /SP/ ; PC_H <--- /SP+1/; SP <--- SP+2

Instrukce RET provede návrat z podprogramu zpět k programu, který vyvolal daný podprogram. Návratovou adresu vybere z vrcholu zásobníku ve stejném formátu, jak ji tam uloží instrukce CALL nebo PUSH. Adresa je uložena v paměti v pořadí nižší a vyšší byte ve smyslu rostoucí adresy paměti.

Nastavení indikátorů: nezměněno

Příklad: Proveďme skok na adresu, danou obsahem zásobníku, čímž provedeme návrat z podprogramu. /viz. obr. 4.12.-2/.



obr. 4.12. - 2

D./ P O D M Í N Ě N Ý N Á V R A T Z P O D P R O G R A M U

RET I. jestliže I. je True, pak PC_L <--- /SP/ ; PC_H <--- /SP+1/, SP <--- SP+2

Podmíněná instrukce RET provede podmíněný návrat z podprogramu. Je-li splněna podmínka daná operandem I., programový čítač PC se naplní adresou, uloženou na vrcholu zásobníku. Vopačném případě se pokračuje instrukcí bezprostředně následující.

Nastavení indikátorů: nezměněno

Možné operandy: I. je některá z podmínek C, NC, Z, NZ, PE, PO, M /viz. tab. 4.11. - 1/

Příklad: Je-li v A registru znak 'A', proveďme návrat, jinak budeme pokračovat.

```
CP 'A'          ; testování střadače na znak 'A'
RET Z           ; je-li Z=1, pak návrat
```

E./ N Á V R A T Z P Ř E R U Š E N Í

RETI $PC_L \leftarrow$ /SP/ , $PC_H \leftarrow$ /SP+1/, SP $\leftarrow$ SP+2

Instrukce se používá k návratu z obslužného programu přerušení. Provádí se jednak skok na adresu, danou vrcholem zásobníku, stejně jako instrukce RET. Druhým úkolem této instrukce je oznámit perifernímu zařízení, jehož přerušení bylo přijato, že končí jeho obsluha.

Tato instrukce neprovádí uvolnění přerušení, které bylo zamaskováno přijetím přerušení. Proto je nutno před instrukcí RETI uvolnit přerušení instrukcí EI, aby mohlo být po ukončení obslužného programu znovu přijato přerušení.

Nastavení indikátorů: nezměněno

F./ N Á V R A T Z N E M A S K O V A T E L N É H O P Ř E R U Š E N Í

RETN $PC_L \leftarrow$ /SP, $PC_H \leftarrow$ /SP+1/, PC $\leftarrow$ PC+2

Instrukce RETN se používá k návratu z nemaskovatelného přerušení. Provádí skok na adresu, danou vrcholem zásobníku, stejně jako instrukce RET. Dále uvede klopný obvod uvolnění přerušení do takového stavu, jaký byl před přijetím nemaskovatelného přerušení.

Nastavení indikátorů: nezměněno

G./ R E S T A R T

RST I. /SP-1/ $\leftarrow$ PC_H; /SP-2/ $\leftarrow$ PC_L; $PC_H \leftarrow$ Ø; $PC_L \leftarrow$ I.; SP $\leftarrow$ SP-2

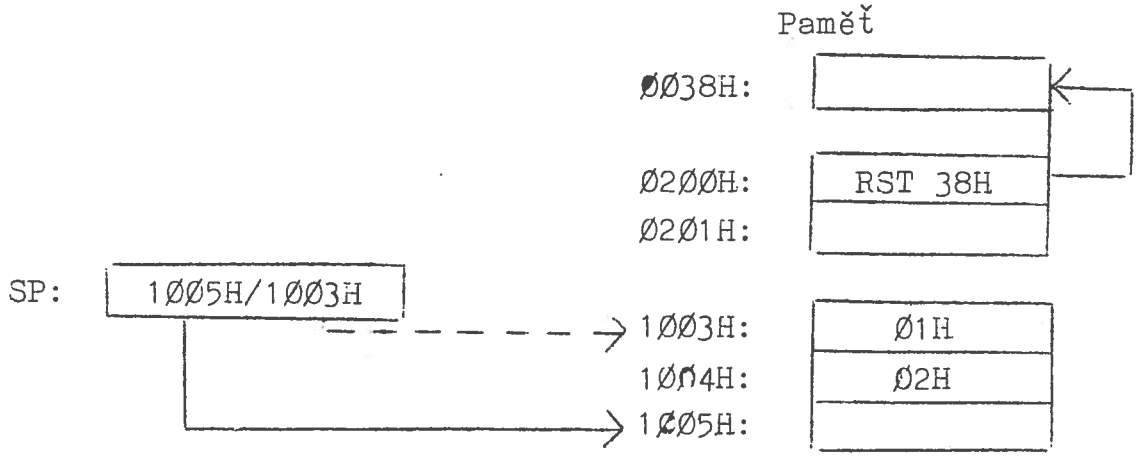
Instrukce RST provede uložení běžného stavu programového čítače PC na vrchol zásobníkové paměti, stejně jako instrukce CALL.

Programový čítač naplní adresou, danou operandem I. Adresa může nabývat Ø - 7 násobku čísla 8.

Tato instrukce se používá při přerušení v režimu Ø.

Poznámka: Povšimněme si, že instrukce RST 38H má strojový operační kód ØFFH. Tzn., že při programové chybě, která má za následek čtení z neexistující paměti /v tomto stavu se chová sběrnice jako připojená na signál úrovně H/, se provede instrukce RST 38H.

Příklad: Proveďme skok do podprogramu, který začíná na adrese 38H /viz. obr. 4.12.-3/.



obr. 4.12.-3

1.13. SKUPINA INSTRUKCÍ VSTUPŮ A VÝSTUPŮ

Následující skupina instrukcí dovoluje pracovat s periferními zařízeními. Obsahuje také instrukce blokového přenosu dat k periferním zařízením.

A./ I N S T R U K C E V S T U P U

INn INPr

IN I., II. I. <--- II.

Instrukce IN provádí přenos jednoho byte dat z periferního zařízení, daného operandem II. do registru, daného operandem I.

Nastavení registrů:

Při kombinaci IN I., /n/ nezměněno.
Při kombinaci operandů IN I., /C/ následující:

S nastaven, je-li hodnota vstupujících dat záporná
Z nastaven, jsou-li vstupní data nulová
H nulován
P/V nastaven v případě sudé parity
N nulován
C nezměněn

Možné kombinace operandů jsou uvedeny v tab. 4.13. - 1

II. I.	A	B	C	D	E	H	L
/n/	x						
/c/	x	x	x	x	x	x	x

Tab. 4.13. - 1. Kombinace operandů instrukce IN a OUT

Poznámka: /n/ znamená, že se V/V přenos týká adresy periferního zařízení n.
/c/ znamená, že se V/V přenos týká adresy periferního zařízení, jehož adresa je uložena v registru C.
Tato symbolika platí pro všechny V/V instrukce.

Příklad: Máme přenést ze zařízení o adresách ØAH až 1FH po jednom byte dat a uložit je postupně od adresy ØFFØH.

```
        LD C, ØAH          ; nastavení první V/V adresy
        LD B, 16H          ; nastavení počítadla
        LD HL, ØFFØH       ; nastavení první adresy paměti
ZPET:   IN A, /C/          ; přenesení dat do A
        LD HL, /A/         ; uložení do paměti
        INC HL             ; zvýšení adresy v paměti
        INC C              ; zvýšení V/V adresy
        DJNZ ZPET          ; testování konce, jinak znovu
```

B./ B L O K O V Ý V S T U P
S I N K R E M E N T A C Í B E Z O P A K O V Á N Í

INI /HL/ ⟵ /C/ ; B ⟵ B-1; HL ⟵ HL+1

Instrukce INI provede přenos z periferního zařízení o adrese, která je uložená v registru C, na paměťovou lokaci, jejíž adresa je uložena v registru HL. Dále provede dekrementaci registru B a inkrementaci registru HL.

Nastavení indikátorů:
S nedefinován
Z nastaven, je-li B-1 = Ø
H nedefinován
P/V nedefinován
N nastaven
C nezměněn

Příklad: Přenesme blok dat, vstupujících ze zařízení s adresou Ø1H o délce 8ØH byte, na adresu paměti 1ØØH. Mezi jednotlivými přenosy volat podprogram WAIT. /Podprogram WAIT nepřepisuje registry BC a HL./.

```
        LD B, 8ØH          ; nastavení počítadla
        LD C, Ø1H          ; nastavení adresy zařízení
        LD HL, 1ØØH        ; nastavení adresy paměti
ZNOVU:  INI                ; provedení přenosu
        JR Z, KONEC        ; testování konce
        CALL WAIT          ; volání WAIT
        JR ZNOVU           ; znovu
KONEC:
```

C./ B L O K O V Ý V S T U P
S I N K R E M E N T A C Í A O P A K O V Á N Í M

INIR /HL/ <--- /C/ ; B <--- B-1; HL <--- HL+1;
opakování do stavu B = Ø.

Instrukce INIR provádí přenos ze zařízení o adrese, dané registrem C, na paměťovou lokaci, danou registrovým párem HL, dekrementuje B registr a inkrementuje HL registr. Tuto činnost provádí tak dlouho, dokud B ≠ Ø. Když obsah registru B dosáhne nuly, CPU pokračuje instrukcí bezprostředně následující.

Nastavení indikátorů:

S	nedefinován
Z	nastaven
H	nedefinován
P/V	nedefinován
N	nastaven
C	nezměněn

Příklad: Přenesme blok délky 64 byte ze zařízení o adrese 1ØH do paměti od adresy 1ØØØH.

```
        LD B, 64           ; nastavení počítadla
        LD C, 1ØH          ; nastavení adresy V/V zařízení
        LD HL, 1ØØØH       ; nastavení cílové adresy
                             přenosu
        INIR               ; provedení přenosu
```

Poznámka: Protože se provádí napřed dekrementace B registru a teprve potom test na nulovost, má nastavení registru B na nulu před instrukcemi INIR, INDR, OTIR, OTDR za následek přenos 256 byte dat.

D./ B L O K O V Ý V S T U P S D E K R E M E N T A C Í
B E Z O P A K O V Á N Í

IND /HL/ <--- /C/; B <--- B-1; HL <--- HL-1

Instrukce IND provádí přenos jednoho byte z periferního zařízení o adrese uložené v registru C na paměťovou lokaci danou registrem HL. Zároveň provede dekrementaci registru B a dekrementaci registrového páru HL.

Nastavení indikátorů:
S nedefinován
Z nastaven, je-li B-1 = Ø
H nedefinován
P/V nedefinován
N nastaven
C nezměněn

Příklad: Přenesme blok dat ze zařízení o adrese ØDH délky 2ØH byte do bloku paměti, začínajícího adresou 1Ø1ØH tak, že první vstupující byte bude ukládán na konec oblasti, další před něj atd. Mezi každým přenosem volat čekací podprogram WAIT /WAIT nepřepisuje HL a BC/.

```
        LD BC, 2ØØDH          ; nastavení počítadla
                                na adresu periferie
        LD HL, 1Ø1ØH+2ØH-1    ; nastavení adresy paměti
ZPET:   IND                   ; provedení přenosu
        JR Z, KONEC           ; testování konce
        CALL WAIT             ; volání WAIT
        JR ZPET               ; znovu
KONEC:
```

E./ B L O K O V Ý V S T U P S D E K R E M E N T A C Í A O P A K O V Á N Í M

INDR /HL/ <--- /C/; B <--- B-1; HL <--- HL-1; opakování až do stavu B = Ø.

Instrukce INDR provádí přenos ze zařízení, daného obsahem registru C, na paměťovou lokaci, danou obsahem registrového páru HL, dekrementuje registr B a registrový pár HL.
Tuto činnost provádí tak dlouho, dokud obsah registru B není roven nule.
Když registr B dosáhne nuly, CPU pokračuje instrukcí bezprostředně následující.

Nastavení indikátorů:
S nedefinován
Z nastaven
H nedefinován
P/V nedefinován
N nastaven
C nezměněn

Příklad: Přenesme data ze zařízení o adrese ØCH a ukládejme je sestupně od adresy ØFFFFH. Je zapotřebí převést celkem 2ØØ byte.

```
        LD BC, 2ØØ*1ØØH+ØCH   ; nastavení počítadla
                                 na V/V adresu
        LD HL, ØFFFFH         ; nastavení adresy paměti
        INDR                  ; provedení přenosu
```

F./ I N S T R U K C E V Ý S T U P U

OUT II., I. II. <--- I. OUTn OUTPr

Instrukce OUT provádí přenos jednoho byte do periferního zařízení o adrese, dané operandem II., z registru, daného operandem I.

Nastavení indikátorů: nezměněno

Možné kombinace operandů udává tabulka 4.13. - 1.

Příklad: Přenesme na zařízení ØAH byte ØFFH.

```
LD A, ØFFH          ; přenášená data do A
OUT /ØAH/, A        ; přenesení obsahu A
                      na adresu ØAH
```

G./ B L O K O V Ý V Ý S T U P
S I N K R E M E N T A C Í B E Z O P A K O V Á N Í

OUTI /C/ <--- /HL/; B <--- B-1; HL <--- HL+1

Instrukce OUTI provede přenesení jednoho byte z paměťové lokace, jejíž adresu obsahuje registr HL, na periferní zařízení, jehož adresu obsahuje registr C. Dále je dekrementován registr B a inkrementován registrový pár HL.

Nastavení indikátorů:
S nedefinován
Z nastaven, je-li B-1=Ø
H nedefinován
P/V nedefinován
N nastaven
C nezměněn

Příklad: Přenesme dva byte do periferního zařízení s adresami ØCH a ØFFH z paměťových lokací 1ØH a 11H.

```
LD C, ØCH           ; nastavení adresy periferie
LD HL, 1ØH          ; nastavení adresy paměti
OUTI                ; provedení prvního přenosu
LD C, ØFFH          ; nastavení nové adresy periferie
OUTI                ; provedení druhého přenosu
```

H./ B L O K O V Ý V Ý S T U P
S I N K R E M E N T A C Í A O P A K O V Á N Í M

OTIR /C/ <--- /HL/; B <--- B-1; HL <--- HL+1;
opakování až do stavu B = Ø.

Instrukce OTIR provede přenesení jednoho byte dat z paměťové lokace, jejíž adresa je dána obsahem registrového páru HL, na periferní zařízení, jehož adresu obsahuje registr C. Dále provede dekrementaci registru B a inkrementaci registrového páru HL. Tuto činnost opakuje tak dlouho, dokud obsah registru B je různý od nuly. Když obsah registru B je roven nule, CPU pokračuje instrukcí bezprostředně následující.

Nastavení indikátorů:
S nedefinován
Z nastaven
H nedefinován
P/V nedefinován
N nastaven
C nezměněn

Příklad: Přenesme blok 256 byte dat, uložený v paměti, od adresy 1ØØØH na periferní zařízení s adresou ØC4H.

```
LD BC, ØØC4H      ; nastavení počítadla na V/V adresu
LD HL, 1ØØØH      ; nastavení adresy paměti
OTIR              ; provedení přenosu
```

I./ B L O K O V Ý V Ý S T U P S D E K R E M E N T A C Í B E Z O P A K O V Á N Í

OUTD /C/ <--- /HL/; B <--- B-1; HL <--- HL-1

Instrukce OUTD provádí přenos jednoho byte dat z paměťové lokace, jejíž adresu obsahuje registrový pár HL, na periferní zařízení, jehož adresa je dána obsahem registru C. Dále provádí dekrementaci registru B a dekrementaci registrového páru HL.

Nastavení indikátorů:
S nedefinován
Z nastaven, je-li B-1 = Ø
H nedefinován
P/V nedefinován
N nastaven
C nezměněn

Příklad: Přenesme ØH bytů z paměti sestupně na periferní zařízení s adresou Ø47H. Adresa posledního byte v paměti je 4ØØH. Před každým přenosem vyčkejme, až se objeví na periferii s adresou Ø48H nula.

```
        LD BC, 8Ø'7H      ; nastavení počítadla na adresu periferie
        LD HL, 4ØØH       ; nastavení adresy paměti
ZNOVU:  IN A, /Ø48H/      ; přenos stavové informace
        CP Ø              ; testování nulovosti
        JR NZ, ZNOVU      ; není nula, znovu
        OUTD              ; je nula, provedení přenosu
        JR NZ, ZNOVU      ; není B-1 = Ø, znovu
```

J./ B L O K O V Ý V Ý S T U P S D E K R E M E N T A C Í A O P A K O V Á N Í M

OTDR /C/ <--- /HL/; B <--- B-1; HL <--- HL-1;
opakování až do stavu B = Ø.

Instrukce OTDR provádí přenos z paměťové lokace, jejíž adresu obsahuje registrový pár HL, na periferní zařízení, jehož adresa je v registru C. Dále dekrementuje registrový pár HL a registr B. Tuto činnost opakuje tak dlouho, dokud je registr B různý od nuly. Jakmile je registr B roven nule, CPU pokračuje instrukcí bezprostředně následující.

Nastavení indikátorů:

S	nedefinován
Z	nastaven
H	nedefinován
P/V	nedefinován
N	nastaven
C	nezměněn

Příklad: Přenesme blok dat, počínaje adresou 1ØØØH a konče adresou F5ØH, na periferní zařízení s adresou ØCØH.

```
LD B, 1ØØØH-ØF5ØH+1     ; nastavení počítadla
LD C, ØCØH              ; nastavení adresy
                          periferie
LD HL, 1ØØØ             ; nastavení adresy paměti
OTDR                    ; provedení přenosů
```

K O N E C

2. Programy, překladače a spojovače

Programy mohou být na různých stupních svého vývoje v různých formách. Většinou při vývoji programů projde program těmito formami: zdrojová forma programu /zdrojový program/, relativní forma programu /relativní program/ a absolutní forma programu /absolutní program/.

2.1. Zdrojový program

Zdrojový program je symbolickou formou programu. Jednotlivé instrukce tvoří řetězce znaků. Tyto řetězce znaků vytváří mnemonický kód instrukce. Např. instrukce CP /compare/ je instrukce pro srovnání.

V této formě programu se mohou vyskytovat poznámky. Jsou to řetězce znaků, začínajících středníkem. Poznámky pomáhají udržet přehlednost programu, nejsou však obsaženy v dalších formách programu, takže vlastní /cílový/ program nijak neprodlužují.

V našem případě bude zdrojový program znamenat program, psaný v jazyku Assembler Z-80. Assembleru se také říká "jazyk symbolických adres". Adresy, na které se odvoláváme v programu, jsou většinou symbolické. Prakticky to znamená, že když uvedeme návěští před některou instrukcí, přiřadí se tomuto návěští adresy, na které je uložen první byte této instrukce. Uvedeme-li v jiné instrukci toto návěští jako operand, dosadí se automaticky za toto návěští adresa /tedy číslo velikosti 2 byte/, přiřazená tomuto návěští. Je patrné, že při jakémkoliv posunu instrukcí nemusíme měnitadresy, ve kterých se na tyto adresy odvoláváme.

Zdrojový program zpracovává překladač. Je to program, do kterého vstupuje zdrojový program a vystupuje relativní nebo absolutní program. Ukázka zdrojového programu v příloze.

2.2. Relativní program

Je to program, ve kterém jsou instrukce zobrazeny již tak, jako v cílovém programu, ale adresy /to znamená odvolávky na adresy v jednotlivých instrukcích/ nejsou absolutní, ale relativní. Jsou vztaženy k programu tak, jako by program začínal na adrese Ø bez ohledu na to, kde bude začátek programu uložen v operační paměti.

Relativní program obsahuje navíc tzv. "Globály" a "Externály".

Externál je symbol, pro který platí stejná pravidla jako pro návěští, ale není uveden v daném programu jako návěští, ale jako externál. Tzn., že adresa bude externálu přiřazena až v dalším zpracování programu.

Globál je návěští, uvedené v daném programu, a je k dispozici v dalším zpracování programu jako protějšek externálům. Ve zdrojovém programu je poznačen jako Globál pomocí pseudoinstrukce "GLOBAL".

V relativním programu jsou globály a externály poznačeny i se symbolickým návěštím.

Relativní program zpracovává sestavovací program /Linker/. Linker musí také dostat ještě jednu důležitou informaci, kde v paměti bude uložena první instrukce programu, aby mohl ze všech relativních adres vytvořit absolutní adresy.

Linker většinou sestavuje více programů dohromady. Tyto programy na sebe navazují pomocí záznamů Globál a Externál. To návěští, které je v jednom programu uvedeno jako externál, musí mít v jiném programu odpovídající záznam Globál /musí si odpovídat řetězce, tvořící návěští/. Linker pak dosadí za externál adresu, danou odpovídajícím globálem v jiném programu. Výsledkem je pak absolutní program.

2.3. Absolutní program

Absolutní program je program zpracovatelný CPU. Vyznačuje se dvěma parametry: zaváděcí adresou a vstupním bodem.

Zaváděcí adresa je adresa, kde program začíná a odkud se má uložit do paměti.

Vstupní bod je adresa, na které se má program spustit /adresa, na které je uložená instrukce, která se má provést jako první/.

Poznámka: Zaváděcí adresa může, ale také nemusí souhlasit se vstupním bodem.

2.4. Absolutní překladač

Je to program, do něhož vstupuje zdrojový program, a jehož výstupem je přímo absolutní program. Adresa, od které se má program sestavit, je uvedena přímo ve zdrojovém programu pseudoinstrukcí ORG.

2.5. Relativní překladač

Je to program, do něhož vstupuje zdrojový program, a jehož výstupem je relativní program. Zdrojový program pro relativní překladač nesmí obsahovat pseudoinstrukci ORG.

2.5.1. Protokol o překladu

Obsahuje zdrojový program včetně poznámek, doplněný číslem řádku, adresou instrukce a poznámkou, jde-li o relativní adresu operandu a jedná-li se o externál nebo globál. Výpis se provádí na tiskárně.

2.5.2. Křížové odkazy

Překladač většinou dovede vypsat na periferní zařízení tzv. tabulku křížových odkazů. Je to abecední seznam návěští s číslem řádků, na kterém je návěští uvedeno jako návěští ve zdrojovém programu a s číslem řádků, ve kterých se provádí odkaz na toto návěští.

Tabulka křížových odkazů slouží k orientaci v programu, hlavně při studiu rozsáhlejších programů.

2.6. Sestavovací program

Je to program, do něhož vstupují relativní programy a vystupuje sestavený absolutní program. Jako parametr se zadává sestavovacímu programu adresa, od které se má začít absolutní program sestavovat.

2.6.1. Zaváděcí mapa

Zaváděcí mapu vytváří spojovací program. Jsou v ní uvedeny zaváděcí body a vstupní body všech sestavovaných modulů. Dále je uveden abecední seznam všech identifikátorů typu globál. U každého identifikátoru je uvedena adresa a modul, v němž je deklarován.

Mapa dále obsahuje identifikátor cílového programu, jeho délku a vstupní bod.