TIT Basic Club

HT1080Z ASSEMBLER/EDITOR

Szövegszerkesztő, assembler fordító és programbelövő

BETÖLTESE

Az assembler/editor gépi kódu program, betöltése a SYSTEM utasitással történik, fileneve EDI, belépési pontja 17152. Igy betöltésekor a *? promptra EDI választ kell adni, betöltés után *?/17152-vel inditható és ujrainditható. Ezután a ?> INPUT prompt jelenik meg.

INPUT

A parancsokat a promptra válaszként kell megadni. Először a parancs betüjét, majd a cimet/cimeket kell beirni. A cimek decimálisak, hexadecimálisak /a végén H/ vagy oktálisak /a végén O/ lehetnek.
Ha több paraméter szükséges, akkor mindegyikre külön prompt jelenik meg.
A gépelési hibák visszaléptetéssel /balra nyil/ javithatók. A promptra NEWLINE választ adva az assembler/editor visszatér az INPUT parancshoz.

POINTER PARANCSOK

Az assembler/editor sor-orientált, miképp maga a Z80 assembler is. A forrásfile aktuális sorára belső pointer mutat.

T - top - a pointer a file elejére megy
B - bottom - a pointer a file végére megy
Dn - down - a pointer n sorral lejjebb megy /default n=1/
Un - up - a pointer n sorral feljebb megy /default n=1/
Pn - print - a képernyőre n sort listáz /default n=1/, a pointer ezután az utolsó listázott sorra mutat
L - locate - a forrásfileban megkeres egy karakterstringet, a pointert a megtalált stringre állitja, ha nem talál ilyen stringet, a pointer a file végére áll

ALTALÁNOS PARANCSOK

H - how big - kiirja a forrásfile start és end cimeit
K - kill - törli a forrásprogramot a memóriából
S - sort - az előző assemblálás során keletkezett szimbólumtáblát rendezi és kiirja, a parancs után egy betüt irva csak a megadott betüvel kezdődő szimbólumokat listázza
opció? V=video/képernyő, E=external/printer
A - assemble - a starttól az end pszeudo utasitásig terjedő forrásprogramot leforditja
opció? V=video/képernyő, E=external/printer, C=kazettára object szalagot ir
C esetén ujabb kérdések: address? filenév?
default a leggyorsabb mód, amig a forrásprogram hibás

KAZETTA PARANCSOK

A forrás- és object programok save-elhetők és load-olhatók, object program esetén a parancs után O betüt kell irni.

W - write - filet ir a kazettára, forrás módban a memóriában lévő forrásfilet irja ki, object módban prompt: address?, mindkét módban prompt: filenév?
R - read - kazettáról filet olvas, forrás módban a memóriában lévő kurrens file utánra tölt /egyszerü merge/, object módban a filet cimek szerint tölti be, mindkét módban prompt: filenév?
load-olás közben "L" látható, hibás load után "*", hibátlan load után a fileméret jelenik meg

V - valid - ellenőrzi, hogy a kiirt file hibátlan-e /ugyanaz, mint a read, de nem tölt a memóriába/

AZ ASSEMBLER

A forrásprogram utasitásai Z80 assembler utasitások. Minden programsor legfeljebb négy mezőt tartalmaz:

CIMKE:OPERÁTOR OPERANDUS;MEGJEGYZÉS

CIMKE

Az utasitásokat cimkékkel lehet megjelölni. Ha olyan operátort használunk, mint JP vagy CALL, akkor operandusként meg kell jelölni a célt /pl. hogy hova kell ugrani/. Az assembler megengedi szimbólikus nevek cimkékként történő alkalmazását.
Szimbólikus nevek kétféleképpen adhatók meg: az utasitás elejére irva /az assembler a programszámláló értékét rendeli hozzá/, , és EQUATE pszeudo utasitással /ezzel tetszés szerinti értéket lehet a szimbólumhoz rendelni, pl. BACKSPACE:EQU 8/.
A szimbólum első karaktere csak betü lehet, egyébként tartalmazhat számokat, hossza legfeljebb 7 karakter. Operátor nevek, regisztor nevek és feltétel kód nevek nem lehetnek szimbólumok.

OPERÁTOROK

Az assemblernek a Z80 utasitáskészlet 74 operátorán kivül néhány pszeudo utasitása is van, mégpedig:

END - end assembly, operandusa nincs
ezzel kell lezárni a forrásprogramot, különben hibaüzenetet kapunk

DS - define storage, egy operandus
megadott számu object byte-ot üresen hagy, pl. szöveg buffer, stack, stb. számára helyfoglalás

DW - define word, egy operandus
az object file-ban egy kétbyte-os szót generál, a byte-okat forditott sorrendben, ahogy az a Z80 utasitások számára szükséges, pl. BUFFER:DW VALUE

DB - define byte/s/, több operandus
az operandus értékét az object file-ba generálja, több operandus lehet vesszőkkel elválasztva, pl.: DB 6,93H,'I',80H
az operandusok kifejezések is lehetnek, a program számláló minden operandus után növekszik, mintha azok külön sorban lennének, ASCII stringek is lehetnek az operandusok, pl.: MSG: DB 'JOREGGELT'

EQU - equate, egy operandus
szimbólumhoz értéket rendel, pl. NEWLINE:EQU 12

ORG - origin, egy operandus
az object file kezdő helyét határozza meg, a programban tetszés szerint többször alkalmazható a kódrészek különböző kezdőpontokra helyezéséhez

LOAD - load memory, egy operandus
object kód memóriába betöltése az előállitáskor megadott cimre, ez a load teljesen független az assembler inditásakor megadott output opciótól, a load-olás a következő ORG-ig tart

OPERANDUSOK

Az operandusok száma nulla /pl. NOP/, egy /pl. CP/, kettő /pl. BIT/, és egy vagy kettő /pl. JR/ lehet. Az operandusok lehetnek regiszter nevek /pl. A, B, DE, IX, .../ és feltétel nevek /pl. Z, NZ, C, M, .../, ASCII literálok /' vagy " közé zárt karakterek/, számok /számjeggyel kezdődő decimális, hexadecimális vagy oktális értékek/, programszámláló /futáskor PC-t módositja/.

Az előző adattipusokból matematikai müveletekkel kifejezések alkothatók, amelyek mindenütt használhatók számok helyett.

MEGJEGYZÉS

Megjegyzések az assemblerben bárhol elhelyezhetők, megjegyzés vége a sor vége.

HIBAKEZELÉS

Ha az assembler hibát talál a forrásprogramban, megáll, hibaüzenetet ad, a hibás sort kiirja, a pointert erre a sorra állitja.

DOUBLE SYMBOL - többször deklarált szimbólum
RSVD - szimbólum neve lefoglalt szó /reserved word/
FULL - szimbólum tábla betelt
ORG - nincs kezdőpont /ORG utasitás/
OPND - operandus hiba
UNDEF - hiányzó szimbólum
EOP - hiányzó END
HUH? - süketlenség

ASSEMBLER LISTA

Képernyőre vagy printerre készülhet. A lista oldalanként készül, rövid szünetekkel. A szünet ideje alatt bármely billentyüt megnyomva az oldal bármely következő billentyü megnyomásáig a képernyőn marad. Az oldalméret képernyőn 15, printeren 16 sor.

EDITOR PARANCSOK

Zn - ZAP - a pointer helyétől kezdve n sort töröl /default n=1/
E - ENTER - beszurás a pointer helye elé, sorszámokat ad, beszurás vége csak "." karaktert tartalmazó sor
N - NEW - a pointer sorát uj sorra cseréli
A sorszámok csupán az editálást segitik, a filenak nem részei.

DEBUG /programbelövő/ PARANCSOK

G - GOTO - vezérlésátadás a felhasználói programnak, prompt: belépési cim? breakpoint cim? /default az aktuális programszámláló/, breakpoint default nincs, ha van, akkor egy cim felhasználásával a byte megőrződik és egy RST 38H utasitás kerül beszurásra
C - COPY - kezdőcimtől végcimig terjedő memóriablokk áthelyezése célcimre
F - FILL - kezdőcimtől végcimig feltölti a memóriát egy DATA konstanssal
X - EXAMINE - a felhasználói gépi kód státuszát irja ki, felül a fő regiszterekkel, a második sorban az alternativ regiszterekkel
M - MODIFY - a memória tartalmát vizsgálja és módositja, a megadott cimet és tartalmát kiirja, a tartalom átirható, majd a következő címre ugrik, kilépés "."-tal
Q - QUERY - a memóriából 8 byte-ot kiir a következő formában: cim, hexadecimális, ASCII a 7. bit nélkül, ASCII a 7. bittel /a harmadik mező megjeleniti azokat a karaktereket, amelyekben a 7. bit 1, a negyedik mező ezeket grafikus karakterekként mutatja/, a kontroll karakterek "."-tá alakitva jelennek meg. Q a 0000 cimtől Qnnnn az nnnn cimtől kezdi a kiirást, majd Q onnan folytatja tovább.

TIT Basic Club

Z80 ASSEMBLER UTASITASOK

Rövid áttekintés

A központi egység 1 byte-os regiszterei: B,C,D,E,H,L,A,I,R. Ezek közül:

A - akkumulátor
I - interrupt vektor 2. byte-ja
R - felfrissitő regiszter

a.többi általános regiszter. Dupla regiszterek: AF,BC,DE,HL,IX,IY, IV,SP,PC. Ezek közül

PC - program számláló
SP - stack pointer
IV - interrupt vektor
IX,IY - indexregiszterek
AF - flageket tartalmazó regiszter /A reg.+flagek/

AF-fel csak egyetlen müveletet lehet végezni. Ezenkivül munkaregiszterek is vannak, mégpedig F',A',B',C',D',E',H',L'.

1 byte-os információk:

a regiszterek tartalma
1 byte-os adat: dd
egy cim tartalma: (cim)
dupla regiszterben tárolt cim tartalma: (HL), (DE), (BC)

2 byte-os információk:

a dupla regiszterek tartalma
2 byte-os adatok: dddd

LOAD /adatátvitel/

Tipusai: regiszterből regiszterbe, mindkét helyen állhat A,B,C,D,E,H, L,I,R, de az I és R csak az A-val együtt lehet
dupla regiszterből dupla regiszterbe: csak SP-be a HL,IX,IY párokból
regiszterbe adat: A,B,C,D,E,H,L regiszterbe egy byte-os adat
dupla regiszterbe 2-byte-os adat: BC,DE,HL,IX,IY,SP párokba
regiszterbe cim tartalma: csak A-ba
dupla regiszterbe cimtől két byte tartalma: BC,DE,HL,IX,IY,SP
cimre regisztert: csak A-t
cimtől 2 byte-ra dupla regisztert: BC,DE,HL,IX,IY,SP

Indirekt cimzések:

dupla regiszterben adott cimre regiszter: BC,DE,HL,IX+távolság, IY+távolság, de BC,DE csak A-val, a többi B,C,D,E,H,L-lel is
regiszterbe egy dupla regiszterben adott cimen álló byte: mint előbb
dupla regiszterben adott cimre adat: HL,IX+távolság,IY+távolság

BIT /bit vizsgálat/, SET /bit bekapcsolás/, RES /bit kikapcsolás/

A megadott sorszámu bittel végez müveletet, regiszterek: A,B,C,D,E,H, L, (HL), (IX+távolság), (IY+távolság). A BIT állit flag-eket, a SET és RES nem állit.

ADD /összeadás/, ADC /maradékos összeadás/

ADD: az A regiszterhez ad egy egybyte-os információt: B,C,D,E,H,L, (HL), (IX+távolság), (IY+távolság), adat
kétbyte-os ADD: HL,IX,IY regiszterekhez BC,DE,SP vagy önmaga hozzáadása
ADC: mint ADD, de paritás, maradék, előjel, nulla flag-eket állitja
kétbyte-os ADC: csak HL-hez BC,DE,HL,SP

SUB /kivonás/, SBC /kivonás átvitellel/

SUB az A-ból kivon egy 1-byte-os információt: A,B,C,D,E,H,L, (HL), (IX+távolság),(IY+távolság),dd
SBC kivonja A-ból a C-flaget /átvitel/ és a fenti 1-byte-os információk valamelyikét
SBC dupla regiszterre: kivonja HL-ből a C-t, és egy regiszterpárt: BC,DE,HL,SP

INC /növelés eggyel/, DEC /csökkentés 1-gyel/

argumentumaik: A,B,C,D,E,H,L, (HL), (IX+távolság), (IY +távolság),BC,DE, HL,SP,IX,IY

Logikai müveletek: AND /és/, OR /vagy/, XOR /kizáró vagy/, CP /hasonlitás/

mindegyiknél az egyik regiszter A, a másik 1-bytes-os információ mint a SUB utasitásnál

Vezérlésátadó utasitások: CALL /szubrutinhiyás/, JP /ugrás/, JR /relativ ugrás/, RET /visszatérés/,DJNZ/csökkentés és felt. ugrás/
feltételek: semmi/-/, nulla/Z/, nem nulla/NZ/, átvitel/C/, nincs átvitel/NC/, páros/PE/, páratlan/PO/, pozitiv/P/, negativ/M/
hivásnál és ugrásnál cimet, relativ ugrásnál távolságot adunk meg
DJNZ csökkenti B-t és ha az eredmény nem nulla, az adott helyre ugrik, relativ ugrással

DI /kizárja az interrupt lehetőségét/
EI /megengedi az interrupt lehetőségét/
RETI /visszatér az interrupt módból/
RETN /visszatér a DI módból/
EX /felcseréli (SP) -t HL,IX,IY valamelyikével, DE-t HL-lel, AF-et AF'-vel/
EXX /felcseréli a regisztereket a vesszős regiszterekkel/
CPL /A komplemense/
DAA /A decimális alakja/
NEG /A -1-szerese/
NOP /nem csinál semmit sem/
CCF, SCF /váltja, ill. 1-re állitja a C-t/
HLT /megállitás/
IN /portról bevitel/, OUT /portra kivitel/
a C-ben lévő adat a port: (C), és A,B,C,D,E,H,L lehet a szállitandó, vagy befogadó regiszter, csak A-val szerepelhet a port száma
PUSH /stackbe bevisz/, POP /stackből kivisz/
regiszterek: AF,BC,DE,HL,IX,IY
RST /1-byte-os szubrutinhivás, kezdőcim lehet 0,8,10,18,20,28,30,38 hexa/
IM /interrupt módok, 0,1,2/
SLA /balra shift/, SRL /jobbra shift/, SRA /jobbra shift, első bit marad/
SLA,SRL első bit nulla, mindháromnál regiszterek mint SUB de dd nincs
Forgatások: utasitáskódban R/rotate/, L/bal/, R/jobb/, C/C-flag/, regiszterek mint előbb
RRD,RLD /jobbra ill. balra forgat félbyteok között, a három félbyte A négy utolsó bitje, (HL) első ill. utolsó 4 bitja/
CPDR,CPIR,CPD,CPI /block compare utasitások, HL tartalmát hasonlitja A-éval, BC-ben számolja, hogy hányszor/
INDR,INIR,IND,INI /a C-ben megadott portról vár adatokat, HL-ben számol, az A-ba kerül az adat/
LDDR,LDIR,LDD,LDI /betöltés/
OTDR,OTIR,OUTD,OUTI /mint IN, de nem be, hanem kivisz adatokat/

TIT Basic Club

Z80 assembler utasitások táblázata

Mnemonic	Leirás	Feltétel kódok
ADC HL,ss	HL=ss+CY-t HL-be	SZPC
ADC A,r	A+r+CY-t A-ba	SZPC
ADC A,n	A+n+CY-t A-ba	SZPC
ADC A,/HL/	A+/HL/+CY-t A-ba	SZPC
ADC A,/IX+d/	A+/IX+d/+CY-t A-ba	SZPC
ADC A,/IY+d/	A+/IY+d/+CY-t A-ba	SZPC
ADD A,n	A+n-t A-ba	SZPC
ADD A,r	A+r-t A-ba	SZPC
ADD A,/HL/	A+/HL/-t A-ba	SZPC
ADD A,/IX+d/	A+/IX+d/-t A-ba	SZPC
ADD A,/IY+d/	A+/IY+d/-t A-ba	SZPC
ADD HL,ss	HL+ss-t HL-be	C
ADD IX,pp	IX+pp-t IX-be	C
ADD IY,rr	IY+rr-t IY-ba	
AND r	A ÉS r-t A-ba	SZP C=0
AND n	A ÉS n-t A-ba	SZP C=0
AND /HL/	A ÉS /HL/-t A-ba	SZP C=0
AND /IX+d/	A ÉS /IX+d/-t A-ba	SZP C=0
AND /IY+d/	A ÉS /IY+d/-t A-ba	SZP C=0
BIT b,r	r b-ik bitjének vizsgálata	SZP
BIT b,/HL/	/HL/ b-ik bitjének vizsgálata	SZP
BIT b,/IX+d/	/IX+d/ b-ik bitjének vizsgálata	SZP
BIT b,/IY+d/	/IY+d/ b-ik bitjének vizsgálata	SZP
CALL cc,nn	nn szubrutint hivja, ha cc	
CALL nn	nn szubrutint hivja	
CCF	komplemens carry flag	C
CP r	A:r hasonlitás	SZPC
CP n	A:n hasonlitás	SZPC
CP /HL/	A:/HL/ hasonlitás	SZPC
CP /IX+d/	A:/IX+d/ hasonlitás	SZPC
CP /IY+d/	A:/IY+d/ hasonlitás	SZPC
CPD	blokk hasonlitás ismétlés nélkül	SZP
CPDR	blokk hasonlitás ismétléssel	SZP
CPI	blokk hasonlitás ismétlés nélkül	SZP
CPIR	blokk hasonlitás ismétléssel	SZP
CPL	A komplemense /egyes komplemens/	
DAA	A decimális igazitása /decimal adjust/	SZP
DEC r	r-t eggyel csökkenti	SZP
DEC /HL/	/HL/-t eggyel csökkenti	SZP
DEC /IX+d/	/IX+d/-t eggyel csökkenti	SZP
DEC /IY+d/	/IY-d/-t eggyel csökkenti	SZP
DEC IX	IX-t eggyel csökkenti	
DEC IY	IY-t eggyel csökkenti	
DEC ss	regiszterpárt csökkenti	
DI	interrupt elnyomás /disable interrupt/	
DJNZ e	B-t csökkenti és JR, ha B≠0	
EI	interrupt engedélyezés /enable/	
EX /SP/,HL	/SP/ és HL felcserélése /exchange/	
EX /SP/,IX	/SP/ és IX felcserélése	
EX /SP/,IY	/SP/ és IY felcserélése	
EX AF,AF'	elsődleges AF-t aktiválja /set prime AF	
EX DE,HL	DE és HL felcserélése active/	
EXX	felcseréli B—L-t B'—L'-vel	
HALT	megállás	

Mnemonic	Leirás	Feltétel kódok
IM 0	0-s interrupt módot beállitja	
IM 1	1-es interrupt módot beállitja	
IM 2	2-es interrupt módot beállitja	
IN A,/n/	n-ből inputtal tölti A-t	
IN r,/C/	/C/-ből inputtal tölti r-et	SZP
INC r	r-t eggyel növeli	SZP
INC /HL/	/HL/-t eggyel növeli	SZP
INC /IX+d/	/IX+d/-t eggyel növeli	SZP
INC /IY+d/	/IY+d/-t eggyel növeli	SZP
INC IX	IX-et eggyel növeli	
INC IY	IY-t eggyel növeli	
INC ss	regiszterpárt növeli	
IND	blokk i/o input /C/-ből	SZP
INDR	blokk i/o input ismétléssel	SZP
INI	blokk i/o input /C/-ből	SZP
INIR	blokk i/o input ismétléssel	SZP
JP /HL/	feltétlen ugrás /HL/-re	
JP /IX/	feltétlen ugrás /IX/-re	
JP /IY/	feltétlen ugrás /IY/-ra	
JP cc,nn	nn-re ugrás, ha cc	
JP nn	feltétlen ugrás nn-re	
JR C,e	relativ ugrás, ha van carry	
JR e	feltétlen relativ ugrás	
JR NC,e	relativ ugrás, ha nincs carry	
JR NZ,e	relativ ugrás, ha nem nulla	
JR Z,e	relativ ugrás, ha nulla	
LD A,/BC/	A-t tölti /BC/-vel	
LD A,/DE/	A-t tölti /DE/-vel	
LD A,I	A-t tölti I-vel	SZP
LD A,/nn/	A-t tölti nn cimről	
LD A,R	A-t tölti R-rel	SZP
LD /BC/,A	A-t tárolja /BC/-re	
LD /DE/,A	A-t tárolja /DE/-re	
LD /HL/,n	n-et tárolja /HL/-re	
LD dd,nn	regiszterpárt tölti nn-nel	
LD dd,/nn/	regiszterpárt tölti nn cimről	
LD HL,/nn/	HL-t tölti nn cimről	
LD /HL/,r	r-et tárolja /HL/-re	
LD I,A	I-t tölti A-val	
LD IX,/nn/	IX-et tölti nn-nel	
LD IX,nn	IX-et tölti nn cimről	
LD /IX+d/,n	n-et tárolja /IX+d/-re	
LD /IX+d/,r	r-et tárolja /IX+d/-re	
LD IY,nn	IY-t tölti nn-nel	
LD IY,/nn/	IY-t tölti nn cimről	
LD /IY+d/,n	n-et tárolja /IY+d/-re	
LD /IY+d/,r	r-et tárolja /IY+d/-re	
LD /nn/,A	A-t tárolja nn cimre	
LD /nn/,dd	regiszterpárt tárolja nn cimpárra	
LD /nn/,HL	HL-t tárolja nn cimre	
LD /nn/,IX	IX-et tárolja nn cimre	
LD /nn/,IY	IY-t tárolja nn cimre	
LD R,A	R-et tölti A-val	
LD r,r'	r-et tölti r'-vel	
LD r,n	r-et tölti n-nel	
LD r,/HL/	r-et tölti /HL/-lel	
LD r,/IX+d/	r-et tölti /IX+d/-vel	
LD r,/IY+d/	r-et tölti /IY+d/-vel	

Mnemonic	Leirás	Feltétel kódok
LD SP,HL	SP-t tölti HL-lel	
LD SP,IX	SP-t tölti IX-szel	
LD SP,IY	SP-t tölti IY-nal	
LDD	blokk töltés előre ismétlés nélkül	P
LDDR	blokk töltés előre ismétléssel	P=0
LDI	blokk töltés hátra ismétlés nélkül	P
LDIR	blokk töltés hátra ismétléssel	P=0
NEG	A negáltja /kettes komplemense/	SZPC
NOP	üres utasitás	
OR r	A VAGY r-t A-ba	SZP C=0
OR n	A VAGY n-t A-ba	SZP C=0
OR /HL/	A VAGY /HL/-t A-ba	SZP C=0
OR /IX+d/	A VAGY /IX+d/-t A-ba	SZP C=0
OR /IY+d/	A VAGY /IY+d/-t A-ba	SZP C=0
OTDR	blokk output hátra ismétléssel	SZP
OTIR	blokk output előre ismétléssel	SZP
OUT /C/,r	r kivitele /C/-be	
OUT /n/,A	A kivitele az n portra	
OUTD	blokk output hátra ismétlés nélkül	SZP
OUTI	blokk output előre ismétlés nélkül	SZP
POP IX	stackből kivesz IX-be	
POP IY	stackből kivesz IY-ba	
POP qq	stackből kivesz qq-ba	
PUSH IX	IX-ből stackre tesz	
PUSH IY	IY-ból stackre tesz	
PUSH qq	qq-ból stackre tesz	
RES b,r	r b-ik bitjét kikapcsolja	
RES b,/HL/	/HL/ b-ik bitjét kikapcsolja	
RES b,/IX+d/	/IX+d/ b-ik bitjét kikapcsolja	
RES b,/IY+d/	/IY+d/ b-ik bitjét kikapcsolja	
RET	szubrutinból visszatér	
RET cc	szubrutinból visszatér, ha cc	
RETI	interruptból visszatér	
RETN	nem maszkolható interruptból visszatér	
RL r	r-t carry-n át balra forgatja	SZPC
RL /HL/	/HL/-t carry-n át balra forgatja	SZPC
RL /IX+d/	/IX+d/-t carry-n át balra forgatja	SZPC
RL /IY+d/	/IY+d/-t carry-n át balra forgatja	SZPC
RLA	A-t carry-n át balra forgatja	C
RLC r	r-t balra körbe forgatja	SZPC
RLC /HL/	/HL/-t balra körbe forgatja	SZPC
RLC /IX+d/	/IX+d/-t balra körbe forgatja	SZPC
RLC /IY+d/	/IY+d/-t balra körbe forgatja	SZPC
RLCA	A-t balra körbe forgatja	C
RLD	/HL/ bcd jegyet balra forgatja	SZP
RR r	r-t carry-n át jobbra forgatja	SZPC
RR /HL/	/HL/-t carry-n át jobbra forgatja	SZPC
RR /IX+d/	/IX+d/-t carry-n át jobbra forgatja	SZPC
RR /IY+d/	/IY+d/-t carry-n át jobbra forgatja	SZPC
RRA	A-t carry-n át jobbra forgatja	C
RRC r	r-t jobbra körbe forgatja	SZPC
RRC /HL/	/HL/-t jobbra körbe forgatja	SZPC
RRC /IX+d/	/IX+d/-t jobbra körbe forgatja	SZPC
RRC /IY+d/	/IY+d/-t jobbra körbe forgatja	SZPC
RRCA	A-t jobbra körbe forgatja	C
RRD	/HL/ bcd jegyet jobbra forgatja	SZP
RST p	ujrainditás p cimtől	

Mnemonic	Leirás	Feltétel kódok
SBC A,r	A-r-CY-t A-ba	SZPC
SBC A,n	A-n-CY-t A-ba	SZPC
SBC A,/HL/	A-/HL/-CY-t A-ba	SZPC
SBC A,/IX+d/	A-/IX+d/-CY-t A-ba	SZPC
SBC A,/IY+d/	A-/IY+d/-CY-t A-ba	SZPC
SBC HL,ss	HL-ss-CY-t HL-be	SZPC
SCF	carry flag-et állitja	C=1
SET b,/HL/	/HL/ b-ik bitjét állitja	
SET b,/IX+d/	/IX+d/ b-ik bitjét állitja	
SET b,/IY+d/	/IY+d/ b-ik bitjét állitja	
SET b,r	r b-ik bitjét állitja	
SLA r	r balra aritmetikai eltolása	SZPC
SLA /HL/	/HL/ balra aritmetikai eltolása	SZPC
SLA /IX+d/	/IX+d/ balra aritmetikai eltolása	SZPC
SLA /IY+d/	/IY+d/ balra aritmetikai eltolása	SZPC
SRA r	r jobbra aritmetikai eltolása	SZPC
SRA /HL/	/HL/ jobbra arithetikai eltolása	SZPC
SRA /IX+d/	/IX+d/ jobbra aritmetikai eltolása	SZPC
SRA /IY+d/	/IY+d/ jobbra aritmetikai eltolása	SZPC
SRL r	r jobbra logikai eltolása	SZPC
SRL /HL/	/HL/ jobbra logikai eltolása	SZPC
SRL /IX+d/	/IX+d/ jobbra logikai eltolása	SZPC
SRL /IY+d/	/IY+d/ jobbra logikai eltolása	SZPC
SUB A,r	A-r-t A-ba	SZPC
SUB n	A-n-t A-ba	SZPC
SUB /HL/	A-/HL/-t A-ba	SZPC
SUB /IX+d/	A-/IX+d/-t A-ba	SZPC
SUB /IY+d/	A-/IY+d/-t A-ba	SZPC
XOR r	A KIZÁRÓ_VAGY r-t A-ba	SZP C=0
XOR n	A KIZÁRÓ_VAGY n-t A-ba	SZP C=0
XOR /HL/	A KIZÁRÓ_VAGY /HL/-t A-ba	SZP C=0
XOR /IX+d/	A KIZÁRÓ_VAGY /IX+d/-t A-ba	SZP C=0
XOR /IY+d/	A KIZÁRÓ_VAGY /IY+d/-t A-ba	SZP C=0

Jelmagyarázat

Utasitás mező:

b	0-7 bitmező
c	feltétel kódok /NZ,Z,NC,C,PO,PE,P,M/
d	+127-től -128 index eltolás /displacement/
dd	regiszterpár /BC,DE,HL,SP/
e	relativ ugrás eltolás +127-től -128
n	közvetlen adat vagy cimérték
pp	regiszterpár /BC,DE,HL,SP/
qq	regiszterpár /BC,DE,HL,SP/
r	regiszter /B,C,D,E,H,L,A/
r'	regiszter /mint r/
ss	regiszterpár /BC,DE,HL,SP/
t	RST mező: cim=t*8

Feltétel kódok:

S	előjel flaget állitja
Z	nulla flaget állitja
P	paritás/tulcsordulás flaget állitja
C	carry flaget állitja

Megjegyzés: /rr/ helyett mindenütt (rr) olvasandó.

TIT Basic Club

HT1080Z ROM rutinok

cimke	érték	tipus	leirás
			USR-ből visszatérés /nem RET-tel/
BSCPAR	0A9A	JP	Basic programba visszatérés paraméterrel
BASIC	1A19	JP	Basic-hez visszatérés /READY üzenet, 1A19 helyett 06CC vagy 0072 lehetséges/
GETPAR	0AF7	CALL	az USR/A/ A paraméterét betölti HL-be, ez kell hogy a user szubrutin első utasitása legyen
DELAY	0060	CALL	BC-be n-et töltve n üres ciklus várakozás
			aritmetikai ROM rutinok
ADDINT	0BD2	CALL	DE és HL összege, eredmény HL-ben és SA-ban, ha nincs tulcsordulás; SA-ban, ha van /flag=4/
ADDSP	0716	CALL	RA és SA összege SA-ban
ADDDP	0C77	CALL	SA és SA_1 összege SA-ban
CPINT	0A39	CALL	DE és HL összehasonlitása
CPDP	0A78	CALL	SA és SA_1 összehasonlitása
CP16	0018	RST	DE és HL mint 16-bites előjel nélküli egészek összehasonlitása
DIVINT	2490	CALL	DE osztva HL-lel, eredmény SA-ban egyszeres pont.
DIVSP	08A2	CALL	RA osztva SA-val, eredmény SA-ban
DIVDP	0DE5	CALL	SA osztva SA_1-gyel, eredmény SA-ban
MLTINT	0BF2	CALL	DE szorozva HL-lel, ha nincs tulcsordulás /flag=2/, akkor eredmény HL-ben és SA-ban, ha van /flag=4/, eredmény csak SA-ban
MLTSP	0847	CALL	RA szorozva SA-val, eredmény SA-ban
MLTDP	0DA1	CALL	SA szorozva SA_1-gyel, eredmény SA-ban
POWER	13F7	CALL	RA az SA-dik hatványon, eredmény SA-ban
SUBINT	0BC7	CALL	DE minusz HL, ha nincs tulcsordulás /flag=2/, eredmény HL-ben és SA-ban, ha van /flag=4/, csak SA-ban
SUBSP	0713	CALL	RA minusz SA, eredmény SA-ban
SUBDP	0C70	CALL	SA minusz SA_1, eredmény SA-ban
			Basic változók keresése
VARPTR	260D	CALL	HL a változó névre mutat, eredmény a változó cime DE-ben
			Adatmozgatás
LDDEHL	09D2	CALL	/HL/ egyszeres pontosságu számot /DE/-be viszi /FLAGSP kell hozzá/
LDRAHL	09C2	CALL	/HL/ egyszeres pontosságu számot RA-ba viszi
LDRASA	09BF	CALL	SA-t RA-ba viszi
LDHLDE	09D3	CALL	/DE/ egyszeres pontosságu számot /HL/-be viszi /FLAGSP kell hozzá/
LDHLSA	09CB	CALL	SA-t átviszi /HL/-be /a HL által mutatott helyre/
LDSTSA	09A4	CALL	SA-t a stackbe viszi
LDSARA	09B4	CALL	RA-t SA-ba viszi
LDSAHL	09B1	CALL	/HL/ egyszeres pontosságu számot SA-ba viszi
			Duplapontosságu adatmozgatás
FLAGDP	0AEC	CALL	SA typeflag-jét duplapontosságura állitja
MVALT	09FC	CALL	SA-t SA_1-be viszi /FLAGDP kell hozzá/
MVDEHL	09D3	CALL	/HL/ számot /DE/-be viszi /FLAGDP kell/
MVHLDE	09D2	CALL	/DE/ számot /HL/-be viszi /FLAGDP kell/
MVSAHL	09F7	CALL	/HL/-t SA-ba viszi /FLAGDP kell/
SA	411D	EQU	SA első byte-ja
SA1	4127	EQU	SA_1 első byte-ja
SAFLAG	40AF	EQU	SA typeflag-je

MVVAR	0982	CALL	/DE/-ből /a DE által mutatott területről/ a typeflag által mutatott számu byte-ot átvisz /HL/-be /typeflag kell hozzá/

Stringkezelő rutinok

BSCTXT	40A4	EQU	ezen a cimen kezdődik a Basic program /szöveg/
CPSTR	25A1	CALL	két string összehasonlitása, HL és BC a két stringre mutat, D és E tartalmazza a stringek hosszát
MOVEA	09D6	CALL	/DE/-ből /HL/-be adatmozgatás, byteszám A-ban
MOVEB	09D7	CALL	mint MOVEA, de byteszám B-ben
MVSTR	21E3	CALL	stringet a stringterületre mozgat, HL a buffer első byte-jára, BC a változónévre mutat

Hibakezelés

ERTRAP	41A6	EQU	hibakezelő ide ugrik, innen RET a Basicbe
ERCODE	409A	EQU	itt tárolódik a hibakód; ha ezen a cimen nem 00_H van, akkor a READY üzenet után kinyomtatásra kerül a hiba sora és a Basic EDIT modba megy
BSCSTM	40E6	EQU	pointer az utoljára végrehajtott Basic utasitás végére /00 vagy ":"/
RUNSTM	1D1E	JP	HL egy Basic utasitás ":" végére vagy egy sor 00 végére mutat, a végrehajtás a következő utasitásnál folytatódik

Matematikai függvények

/1. tipus: SA-ban egész, egyszeres vagy kétszeres pontosságu szám, typeflag mutatja, hogy milyen; eredmény SA-ban, typeflag-et állitja,
2. tipus: SA-ban egyszeres pontosságu szám, eredmény SA-ban/

ABS	0977	CALL	1.tipus
ATN	15BD	CALL	2.tipus
COS	1541	CALL	2.tipus
EXP	1439	CALL	2.tipus
FIX	0B26	CALL	2.tipus
FLAGDP	0AEC	CALL	SA typeflag-jét duplapontosságura állitja
FLAGIN	0A9D	CALL	SA typeflag-jét egész tipusra állitja
FLAGSP	0AEF	CALL	SA typeflag-jét egyszeres pontosságura állitja
INT	0B37	CALL	1.tipus
LOG	0809	CALL	2.tipus
RND	14C9	CALL	1.tipus /ha SA-ban 0 van, akkor 0 és 1 köztit ad/
RANDOM	01D3	CALL	randomizál
RSETSA	0778	CALL	SA-t kinullázza
SGN	098A	CALL	1.tipus
SIN	1547	CALL	2.tipus
SQR	13E7	CALL	2.tipus
TAN	15A8	CALL	2.tipus

Konverzió

CSAASC	0FBD	CALL	SA-t konvetálja ASCII-ra /typeflag áll./, az eredmény /decimálisan/ a 4130_H cimen kezdődő bufferben, eredmény végén 00_H /HL/=4130
CSAINT	0A7F	CALL	SA-t egésszé, eredmény SA-ban /typeflag állitandó/
CSASP	0AB1	CALL	SA-t egyszeres pontosságura, eredm. SA-ban /typefl/
CSADB	0ADB	CALL	SA-t duplapontosságura, eredmény SA-ban /typeflag/
VAL	1E5A	CALL	decimális számot megadó stringet /végén 00_H byte/ binárissá konvertál, HL az első karakterre mutat, eredmény DE-ben

Input/output

CLS	01C9	CALL	képernyő törlés
KBDSCN	002B	CALL	billentyüzet vizsgálata /scan/, eredmény A-ban
OUTSEL	409C	EQU	output eszköz választás, -1 a kazetta, 0 a képernyő, 1 a printer
OUTSTR	28A7	CALL	string outputja, a stringet 00_H vagy 22_H /" kódja/ terminálja, output eszköz választás OUTSEL-lel
OUTCHR	032A	CALL	A regiszterből karakter kivitele /eszközválasztás OUTSEL-lel/
RDSECT	46DD	CALL	lemez olvasás és irás, mindkettőnél /C/=drive,
WTSECT	46E6	CALL	/D/=track, /E/=sector, /HL/=data buffer, Z=1, ha sikeres volt, egybéként hibaüzenet kód A-ban

Grafika

GRAPH	0150	JP	Használata: visszatérési cim HL-be, PUSH HL, A-ba 0 /POINT/ vagy 1 /RESET/ vagy 80_H /SET/, PUSH AF, A-ba x-koordináta, PUSH AF, A-ba y-koordináta, végül JP GRAPH

Összehasonlitások /CPxxx rutinok/ eredménye

ha egyenlők	Z=1
ha nem egyenlők	Z=0
ha az első kisebb	C=0
ha a második kisebb	C=1

Software accumulator /SA/

40AF	411D	411E	411F	4120	4121	4122	4123	4124
typeflag	←----------			számok				----------→
2					LSB	MSB		
4					LSB	byte2	MSB	EXP
8	LSB	byte2	byte3	byte4	byte5	byte6	MSB	EXP

A $40AF_H$ /16559$_D$/ cimen található typeflag értéke szerint értelmeződik SA tartalma. Ha typeflag=2, akkor a 4124/16673/ és 4122/16674/ cimeken egész szám van /LSB=less significant byte, alsó byte, MSB=more significant byte, felső, értékesebb byte/. Ha 4, akkor egyszeres pontosságu szám, ha 8, akkor duplapontosságu szám, ezek elhelyezését a táblázat mutatja.
SA_1 másodlagos software accumulator a 4127-412E /16679-16686/ cimeken, csak duplapontosságu számokra használható.
Egyszeres pontosságu számok és egész számok regiszterekben is tárolhatók. Egész számok a DE és HL regiszterekben tárolódnak. Egyszeres pontosságuak gyakran az E,D,C,B regiszterekben /sorra LSB,byte2,MSB,EXP/, ezt nevezzük regiszter accumulatornak /RA/.

Megjegyzés: /rr/ jelentése az rr által mutatott cim tartalma /adatmozgatás, duplapontosságu adatmozgatás, stringkezelő rutinok/.

TIT Basic Club

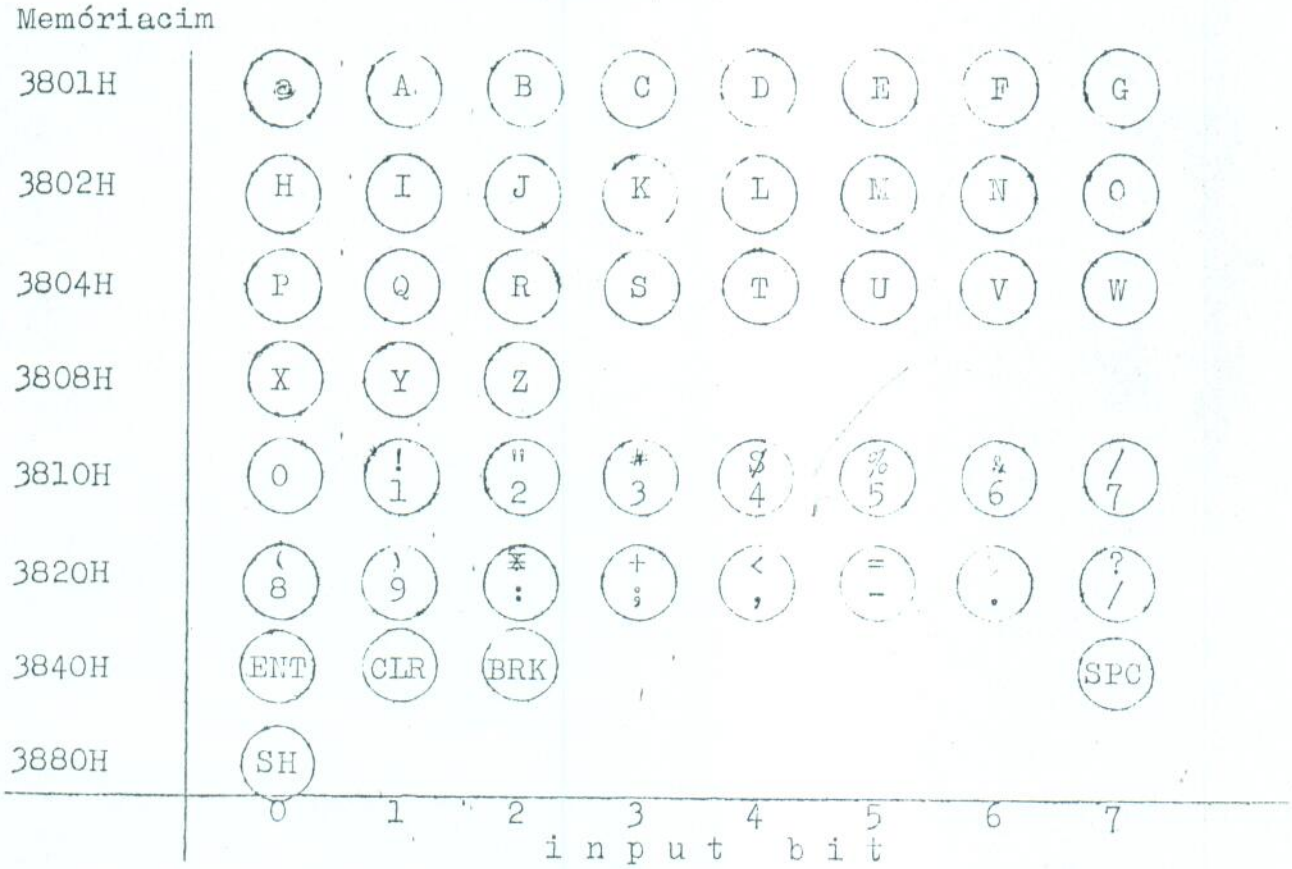
HT1080Z memória elrendezése

hexa cim	decimális cim	tartalom
0000-2FFF	00000-12287	Basic interpreter
3000-3BFF	12288-15359	input-output terület /lásd a köv. táblát/
3C00-3FFF	15360-16383	video display memória
4000-42E8	16384-17128	Basic munkaterület /lásd a köv. táblát/
42E9-7FFF	17129-32767	Basic program, változok táblázata, tömbváltozók táblázata, stack terület, string terület, védett terület

Cimtáblázat

37E0	14304	interrupt address port, bit7=1:óra, bit6=1:disk
37E1	14305	disk drive select
37E4	14308	1. vagy 2. magnó választás /0 vagy 1/
37E8	14312	nyomtató address port
37EC	14316	disk parancs/státusz
37ED	14317	disk track választás
37EE	14318	disk szektor választás
37EF	14319	disk adat
3800-3840	14336-14400	billentyüzet address portok
3C00-3FFF	15360-16383	video display memória
4003-4005	16387-16389	RST 10H vezérlésátadási cim
4012-4015	16402-16405	RST 38H vezérlésátadási cim
4015-401C	16405-16412	billentyüzet ellenőrző blokk
401D-4024	16413-16420	video ellenőrző blokk
4025-402C	16421-16428	nyomtató ellenőrző blokk
403D	16445	magnó port és printsize flag /bit3/ másolat
4040-4046	16448-16454	idő-terület /25msec ütés,sec,min,h,yy,dd,mm/
4047-4048	16455-16456	DOS terület kezdete
4049-404A	16457-16458	DOS memóriaméret
408E-408F	16526-16527	USR kezdőcim
4099	16537	INKEY$ tárolómező
409A	16538	hibakód
409C	16540	output flag /-1,0 vagy 1/
40A0-40A1	16544-16545	string terület kezdőcim
40A2-40A3	16546-16547	Basic program kurrens sorszáma
40A4-40A5	16548-16549	Basic program kezdőcime
40AA-40AC	16554-16556	RND mag
40AF	16559	SA random vált. tipus flag állitja
40B1-40B2	16561-16562	Basic számára elérhető utolsó memóriacim
40D6-40D7	16598-16599	string terület köv. használható byte-ja
40DF-40E0	16607-16608	system szalag belépési pontja
40E6-40E7	16614-16615	utoljára végrehajtott Basic utasitás terminátora
40EA-40EB	16618-16619	hibás sor száma
40F9-40FA	16633-16634	változótábla kezdete
40FB-40FC	16635-16636	tömbváltozó-tábla kezdete
40FD-40FE	16637-16638	tömbváltozó-tábla utolsó byte-ja
4101-411A	16641-16666	változótipus tábla
411D-4124	16669-16676	SA /software accumulator/
4127-412E	16679-16686	SA1 /alternate software accumulator/
4152-41AF	16722-16815	DOS interface terület
41A6-41A8	16806-16808	Basic hibaüzenet terület

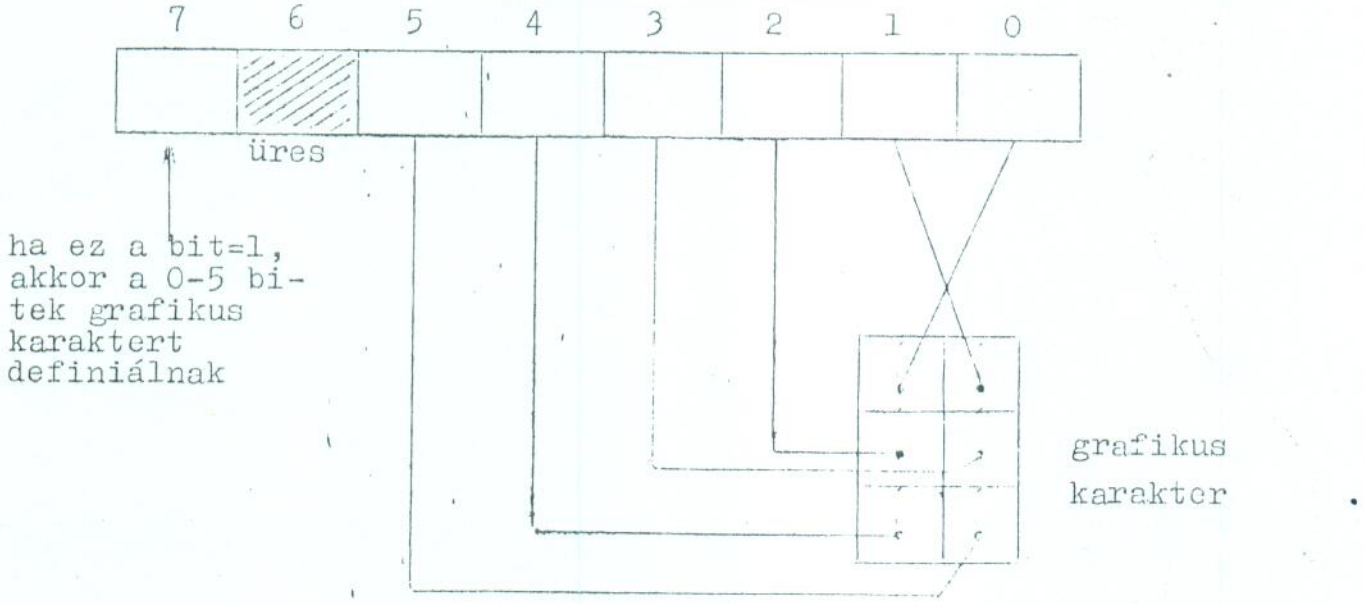
Billentyüzet cimzése



Röviditések: ENT=enter,return, CLR=clear, BRK=break, SPC=space,blank, SH=shift

Példa: ha az "S" billentyüt megnyomjuk, a 3804H cimen 08H lesz, az összes többi cimen 00H

Grafikus karakterek



Példa: a BASIC SET/63,16/ utasitása az 5. karakteres sor 31. oszlopában lévő grafikus karakter középső sorának jobb oldali képelemét állitja, tehát a 15360+351=15711=3D5F/hexa/ cim 3-as blokkját, assemblerben mindössze két utasitás:

```
LD  HL,3D5FH
SET 3,/HL/
```

BASIC táblázatok

Egyszerü változótábla

Erre mutat 40F9-40FA, a tábla elemei a következő tipusuak lehetnek:

Egész változó:
typeflag /2/
változónév második karaktere
változónév első karaktere
érték /LSB, alsó byte/
érték /MSB, felső, értékesebb byte/

Egyszeres pontosságu:
typeflag /4/
változónév második karaktere
változónév első karaktere
érték /LSB/
érték /2. byte/
érték /MSB/
kitevő

Duplapontosságu:
typeflag /8/
változónév második karaktere
változónév első karaktere
érték /7 byte-on: LSB,2.byte,...,6.byte,MSB/
kitevő

String változó:
typeflag /3/
változónév második karaktere
változónév első karaktere
string hossza
kezdőcim LSB-je
kezdőcim MSB-je

Tömbváltozó tábla

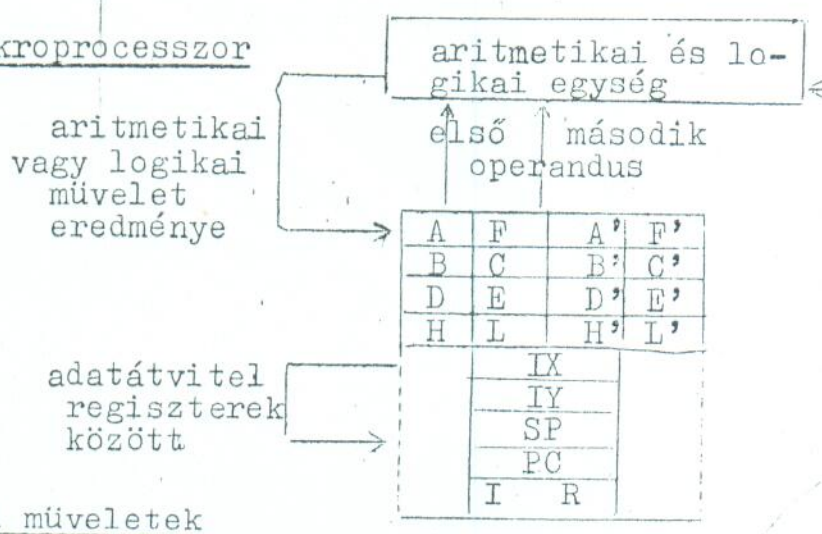
typeflag
változónév második karaktere
változónév első karaktere
tömb tárolóterület mérete /LSB/
tömb tárolóterület mérete /MSB/
dimenziók száma
első dimenzió LSB-je
első dimenzió MSB-je

BASIC kulcsszó tábla

hexa cim	kulcs-szó	hexa cim	kulcs-szó
4152	CVI	417C	FIELD
4155	FN	417F	GET
4158	CVS	4182	PUT
415B	DEF	4185	CLOSE
415E	CVD	4188	LOAD
4161	EOF	418B	MERGE
4164	LOC	418E	NAME
4167	LOF	4191	KILL
416A	MKI	4197	LSET
416D	MKS	419A	RSET
4170	MKD	419D	INSTR
4173	CMD	41A0	SAVE
4176	TIME	41A3	LINE
4179	OPEN		

Ezek a három byte-os területek a lemez nélküli rendszerekben mind JP 012DH utasitást tartalmaznak és hibaüzenetet adnak. A cimeket átirva ezekhez a kulcsszavakhoz tartozó saját rutinokkal bővithetjük a BASIC-et.

Z80 mikroprocesszor



Logikai müveletek

```
LD  A,/HL/   binárisból   LD A,/HL/  2.bitet  LD A,/HL/    2.bitet
OR  30H      ASCII-t      OR 4,A      1-re     AND A,ØFBH  0-ra állitja
                          LD /HL/,A            LD /HL/,A
```

Aritm., logikai és összehasonlitó müveletek

Az ábra lényege: 8-bites müveletnél egyik regiszter az A, másik oprendus A,H,L,B,C,D,E regiszterből vagy memóriából jöhet, eredmény az A-ba kerül. Lehet regiszterek vagy memóriacimek tartalmát 1-gyel növelni /INC/ vagy csökkenteni /DEC/. 16-bites müveletek HL,IX,IY között. /Az utasitástáblázatból ki lehet olvasni, hogy pontosan milyen kombinációk lehetnek./

Adatátvitel LD-vel regiszterből regiszterbe, regiszterből memóriába, memóriából regiszterbe, utasitás operandus mezőjéből regiszterbe, memóriába, regiszterből i/o eszközre, i/o eszközről regiszterbe lehet adatot átvinni.

Példa

```
ORG 7000H
LOAD 7000H
CALL 01C9H
LD A,"Q"
LD /3D20H/,A
HALT
END
```

Példa

```
ORG  7000H
LOAD 7000H
LD A,"*"
LD HL,3C00H
LD B,255
CIKL: LD /HL/,A
INC HL
DEC B
JR NZ,CIKL
VEGE: JR VEGE
END
```

Példa

```
ORG 7000H
LOAD 7000H
CALL 01C9H
LD HL,SZOVEG
LD DE,VIDEO+512
LOOP: LD A,/HL/
CP "@"
JP Z,1A19H
LD /DE/,A
INC HL
INC DE
JR LOOP
VIDEO: EQU 3C00H
SZOVEG: DB "COMPUTER@"
END
```

Példa:

```
LD HL,0      forrás
LD DE,3C00H  cél
LD BC,1024   bytesz
LDIR
```

Szorzás	A	C
LD A,/HL/	17	
SLA A	34	
LD C,A	34	34
SLA A	68	34
SLA A	136	34
ADD A,C	170	34
LD /HL/,A	170	34

16-bites összeadás

```
LD A,/IX+1/   alsó byte
ADD A,/IY+1/
LD /IX+1/,A
LD A,/IX/     felső byte
ADC A,/IY/    carryt is
LD /IX/,A     hozzáadja
```

Input/output

```
LD A,/3801H/ első sor
OR A   0-e? trükk!
JP Z,NOKEY  nem nyomták
LD B,FFH    meg
K2: INC B
SRL A   toljuk,mig nem 0
JP NZ,K2
LD A,B
LD B,Ø
LD HL,TABLE  karakter-
ADD HL,BC    táblából tölt
LD A,/HL/
```

Cimzési módok

```
ADD A,23     azonnali, A-hoz 23-t hozzáad
LD BC,3700H  azonnali, BC dupla reg.-be 3700H-t t
ADD A,D      regiszter cimzés, A-hoz D tartalmát
             adja hozzá
LD A,/1000H/ közvetlen, a regisztert a memória-
LD /5000H/,A cim tartalmával tölti, a regiszter
             a megadott cimre tárolja
LD HL,1000H  regiszter indirekt: HL-be tölti a
LD A,/HL/    cimet, majd a HL-ben lévő cimről
             tölti az A-t
JR LOOP      relativ cimzés /-128---+127/
LD B,/IX+40/ indexelt cimzés IX+40-ről tölt
```

Koncert

```
        LD C,/időtartam/
CONT:   LD B,/frekvencia/
        LD A,1
        OUT /0FFH/,A felső fél hullám
LOOP1:  DJNZ LOOP1   várakozik /frekv./
        LD B,/frekvencia/
        LD A,2       alsó fél hullám
        OUT /0FFH/,A
LOOP2:  DJNZ LOOP2   várakozik
        DEC C        időt csökkenti
        JP NZ,CONT   következő hullám
```

Assembler szubrutin

Betöltése előtt Basic számára elérhető memória felső vége /16561-16562/ átirandó ORG-nál kisebbre, vagy bekapcsoláskor a READY?-re /memória méret?/ kell ennél kisebbet megadni.

USR hivása

16526-16527 USR kezdőcim mezőbe beirni az assembler rutin kezdőcimét:

100 POKE 16526,xxx: POKE 16527,yyy
110 X=USR/A/

visszatérés lásd a ROM rutinok elején

Memória elrendezés

a MEM táblában nem szereplő fontos érték 16416-16417 a kurzor cime

Peek és Poke

Basic program mérete = vált.tábla kezdete - Basic pgm. kezdőcime
PRINT PEEK/16634/*256+PEEK/16633/-PEEK/16549/*256-PEEK/16548/

Programsorok felépitése:

következő sorra mutató pointer /2/ sorszám /2/ utasitás kódja /1/ szövegrész,változók,müveleti jelek stb. /xx/ sorvége jel /1/ file vége jel/2/

Merge

első programot betöltjük, ekkor 40F9-40FA /vált.tábla kezdete/ az első program utáni első cimre mutat, ebből kivonunk 2-t /töröljük a file vége jelet/, átmásoljuk 40A4-40A5-re /Basic program kezdőcime/, betöltjük a második programot, majd az eredeti cimet visszairjuk 40A4-40A5-re

?PEEK/16548/:?PEEK/16549/ leirjuk /x,y/
CLOAD prog1
?PEEK/16633/:?PEEK/16634/ kivonunk belőle 2-t /marad u1,v1/
POKE 16548,v1: POKE 16549,u1
CLOAD prog2
POKE 16548,x:POKE 16549,y visszairjuk az eredeti kezdőcimet

Basic és Assembler

Képernyőt végigcsikozza 1. SET-tel
2. PRINT@ CHR$ -ral
3. STRING$/64,CHR$/176//-tal
4. POKE-kal
5. assembler szubrutinnal

```
VIDEO:  EQU 3C00H
BSCPAR: EQU 0A09H
        ORG 7EC0H
        LD BC,1024
        LD HL,VIDEO
LOOP:   LD /HL/,176
        INC HL
        DEC BC
        LD A,C
        OR B
        JR NZ,LOOP
        JP BSCPAR
        END
```