INTEL 8085 gépi utasításkészlete

ADATMOZGATÓ UTASÍTÁSOK

MOV r1,r2
(r1) <= (r2)
Az **r2** regiszter tartalmát átírja az **r1** regiszterbe.
1 byte: 01DD DSSS
1 gépi ciklus, 4 fázis, flag-eket nem állítja.

MOV r,M
(r) <= ((H)(L))
Az **r** regiszterbe átírja a **HL** tartalmával kijelölt memóriarekesz tartalmát.
1 byte: 01DD D110
2 gépi ciklus, 7 fázis, flag-eket nem állítja.

MOV M,r
((H)(L)) <= (r)
A **HL** tartalmával kijelölt memóriarekeszbe átírja az **r** regiszter tartalmát.
1 byte: 0111 0SSS
2 gépi ciklus, 7 fázis, flag-eket nem állítja.

MVI r,data
(r) <= data
Az **r** regiszterbe beírja az utasítás 2. byte-jában megadott adatot.
1. byte: 00DD D110
2. byte: data
2 gépi ciklus, 7 fázis, flag-eket nem állítja.

MVI M,data
((H)(L)) <= data
A **HL** tartalmával kijelölt memóriarekeszbe beírja az utasítás 2. byte-jában megadott adatot.
1. byte: 0011 0110
2. byte: data
3 gépi ciklus, 10 fázis, flag-eket nem állítja.

LXI rp, data16
(rh) <= 3.byte, (rl) <= 2.byte
A kijelölt **rp** regiszterpárba beírja az utasítás 2. és 3. byte-jában megadott adatot (**data16**).
1. byte: 00RP 0001
2. byte: adat (**data16**) felsô byte
3. byte: adat (**data16**) alsó byte
3 gépi ciklus, 10 fázis, flag-eket nem állítja.

LDA addr
(ACC) <= ((3.byte)(2.byte))
Az utasítás 3. és 2. byte-jában megadott címû memóriarekesz (**addr**) tartalmát beírja az **ACC**-be.
1. byte: 0011 1010
2. byte: cím (**addr**) alsó byte
3. byte: cím (**addr**) felsô byte
4 gépi ciklus, 13 fázis, flag-eket nem állítja.

STA addr
((3.byte)(2.byte)) <= (ACC)
Az **ACC** tartalmát beírja az utasítás 3. és 2. byte-jában megadott címû (**addr**) memóriarekeszbe.
1. byte: 0011 0010
2. byte: cím (**addr**) alsó byte
3. byte: cím (**addr**) felsô byte
4 gépi ciklus, 13 fázis, flag-eket nem állítja.

LHLD addr
(L) <= ((3.byte)(2.byte))
(H) <= ((3.byte)(2.byte)+1)
Az utasításban megadott címû (**addr**) memóriaszó, vagyis két byte tartalmát átmásolja a **HL** regiszterpárba.
1. byte: 0010 1010
2. byte: cím (**addr**) alsó byte
3. byte: cím (**addr**) felsô byte
5 gépi ciklus, 16 fázis, flag-eket nem állítja.

SHLD addr
((3.byte)(2.byte)) <= (L)
((3.byte)(2.byte)+1) <= (H)
A **HL** regiszterpár tartalmát átmásolja az utasításban megadott címû (**addr**) memóriaszóba, vagyis 2 memóriabyte-ba.
!. byte: 0010 0010
2. byte: cím (**addr**) alsó byte
3. byte: cím (**addr**) felsô byte
5 gépi ciklus, 16 fázis, flag-eket nem állítja.

LDAX rp
(A) <= ((rp))
Az **rp** regiszterpár tartalmával megcímzett memóriabyte-ot átmásolja az **ACC**-be. Cak a **BC** és a **DE** regiszterpár jelölhetô ki.
1. byte: 00RP 1010
2 gépi ciklus, 7 fázis, flag-eket nem állítja.

STAX rp
((rp)) <= (ACC)
Az **rp** regiszterpár tartalmával megcímzett memóriabyte-ba másol-ja az **ACC**-t. Csak a **BC** és **DE** regiszterpár jelölhetô ki.
1. byte: 00RP 0010
2 gépi ciklus, 7 fázis, flag-eket nem állítja.

XCHG
(H) <=> (D), (L) <= > (E)
Megcseréli a **HL** és a **DE** regiszterpárok tartalmát.
1. byte: 1110 1011
1 gépi ciklus, 4 fázis, flag-eket nem állítja.

ARITMETIKAI UTASÍTÁSOK

ADD r
(A) <= (A) + (r)
Az **ACC**-be írja az **ACC** és az **r** regiszter tartalmának összegét.
1. byte: 1000 0SSS
1 gépi ciklus, 4 fázis, állítja a Z,S,P,CY,AC -t.

ADD M
(A) <= (A) + ((H)(L))
Az **ACC**-be írja az **ACC** és a **HL** által kijelölt memóriabyte tartal-mának összegét.
1. byte: 1000 0110
2 gépi ciklus, 7 fázis, állítja a Z,S,P,CY,AC -t.

ADI data
(A) <= (A) + data
Az **ACC**-be írja az **ACC** tartalmának és az utasítás 2. byte-jának (**data**) összegét.
1. byte: 1100 0110
2. byte: data
2 ciklus, 7 fázis, állítja a Z,S,P,CY,AC-t.

ADC r
(A) <= (A) + (r) + CY
Az **ACC**-be írja az **ACC** és az **r** regiszter tartalmának és **CY** flag összegét.
1. byte: 1000 1SSS
1 ciklus, 4 fázis, állítja a Z,S,P,CY,AC-t.

ADC M
(A) <= (A) + ((H)(L)) + CY
Az **ACC**-be írja az **ACC** és a **HL** regiszterpár tartalmával kijelölt memóriabyte és a **CY** összegét.
1. byte: 1000 1110
2 gépi ciklus, 7 fázis, állítja a Z,S,P,CY,AC-t.

ACI data
(A) <= (A) + data + CY
Az ACC-be írja az ACC tartalmának, az utasítás 2. bytejának (**data**), és a **CY** összegét.
1. byte: 1100 1110
2. byte: data
2 gépi ciklus, 7 fázis, állítja a Z,S,P,CY,AC-t

SUB r
(A) <= (A) - (r)
Az **ACC**-be írja az **ACC** és a kijelölt **r** regiszter tartalmának különbségét.
1. byte: 1001 0SSS
1 gépi ciklus, 4 fázis, állítja a Z,S,P,CY,AC-t.

SUB M
(A) <= (A) - ((H)(L))
Kivonja az **ACC** tartalmából a **HL** regiszterpár tartalmával kijelölt memóriabyte tartalmát, a különbségét az **ACC**-ba írja.
1. byte: 1001 0110
2 gépi ciklus, 7 fázis, állítja a Z,S,P,CY,AC-t.

SUI data
(A) <= (A) - data
Az **ACC**-ba írja az **ACC** tartalmának és az utasítás 2. byte-jának (**data**) különbségét.
1. byte: 1101 0110
2. byte: data
2 gépi ciklus, 7 fázis, állítja a Z,S,P,CY,AC-t

SBB r
(A) <= (A) - (r) - CY
Az **ACC** tartalmából kivonja a **r** regiszter tartalmát és **CY**-t, a különbséget az **ACC**-ba írja.
1. byte: 1001 1SSS
1 gépi ciklus, 4 fázis, állítja a Z,S,P,CY,AC-t.

SBB M
(A) <= (A) - ((H)(L)) - CY
Az **ACC** tartalmából kivonja a **HL** regiszterpár tartalmával kijelölt memórabyte-ot és a **CY**-t, a különbséget az **ACC**-ba írja.
1. byte: 1001 1110
2 gépi ciklus, 7 fázis, állitja a Z,S,P,CY,AC-t.

SBI data
(A) <= (A) - data -CY
Az **ACC** tartalmából kivonja az utasítás 2. byte-ját (**data**) és a **CY**-különbséget az **ACC**-ba írja.
1.byte: 1101 1110
2.byte: data
2 gépi ciklus, 7 fázis, állítja a Z,S,P,CY,AC-t.

INR r
(r) <= (r) + 1
Eggyel megnöveli a **r** regiszter tartalmát. Cak a **CY** flag nem változik meg.
1. byte: 00DD D100
1 gépi ciklus, 4 fázis, állítja a Z,S,P,AC-t.

INR M
((H)(L)) <= ((H)(L)) + 1
Eggyel megnöveli a **HL** regiszterpár tartalmával kijelölt memóriarekesz tartalmát. Csak a **CY** flag nem változik meg.
1. byte: 0011 0100
3 gépi ciklus, 10 fázis, állítja a Z,S,P,AC-t.

DCR r
(r) <= (r) - 1
Eggyel csökkenti az **r** regiszter tartalmát. Cak a **CY** flag nem változik meg.
1. byte: 00DD D101
1 gépi ciklus, 4 fázis, állítja a Z,S,P,AC-t.

DCR M
((H)(L)) <= ((H)(L)) - 1
Eggyel csökkenti a **HL** regiszterpár tartalmával kijelölt memória-byte tartalmát. Cak a **CY** flag nem változik meg.
1. byte: 0011 0101
3 gépi ciklus, 10 fázis, állítja a Z,S,P,AC-t.

INX rp
((rh)(rl)) <= ((rh)(rl)) - 1
A **rp** regiszterpár tartalmát eggyel nóveli. Flag-eket nem állítja.
1. byte: 00RP 1011
1 gépi ciklus, 6 fázis, flag-eket nem állítja.

DCX rp
((rh)(rl)) <= ((rh)(rl)) - 1
Eggyel csökkenti a **rp** regiszterpár tartalmát.Flag-eket nem állítja.
1.byte: 00RP 1011
1 gépi ciklus, 6 fázis, flag-eket nem állítja.

DAD rp
(H)(L) <= (H)(L) + (rh)(rl)
A **HL** regiszterpár tartalmához hozzáaadja a kijelölt **rp** regiszterpár tartalmát, és az eredményt a **HL**-be írja vissza.Csak a **CY** flag változik, a legfelsô helyértékrôl származó átviteltôl függôen..
1. byte: 00RP 1001
3 gépi ciklus, 10 fázis, állítja a CY flag-et.

DAA
(A) bcd <= (A) bin
A BCD számok bináris összeadásából származó **ACC** tartalmat átalakítja BCD kódba (két lépéses 6-korrekció)
1. byte: 0010 0111
1 gépi ciklus, 4 fázis, flag-eket nem állítja.

LOGIKAI UTASÍTÁSOK

ANA r
(A) <= (A) **ÉS** (r)
Az **ACC** és az **r** regiszter tartalmának logikai **ÉS** kapcsolatát visszaírja az **ACC**-ba. Mindig **CY**=0, és **AC**=1.
1. byte: 1010 0SSS
1 gépi ciklus, 4 fázis, állítja a Z,S,P,CY,AC-t.

ANA M
(A) <= (A) **ÉS** ((H)(L))
Az **ACC** és a **HL** tartalmával kijelölt memóriabyte logikai **ÉS** kapcsolatát visszaírja az **ACC**-ba. Mindig **CY**=0, és **AC**=1.
1. byte: 1010 0110
2 gépi ciklus, 7 fázis, állítja a Z,S,P,CY,AC-t

ANI data
(A) <= (A) **ÉS** data
Az **ACC** tartalmának és az utasítás 2. byte-jának (**data**) logikai **ÉS** kapcsolatát visszaírja az **ACC**-ba. Mindig **CY**=0, és **AC**=1.
1. byte: 1110 0110
2. byte: data
2 gépi ciklus, 7 fázis, állítja Z,S,P,CY,AC-t.

XRA r
(A) <= (A) **KIZÁRÓVAGY** (r)
Az **ACC** és az **r** regiszter tartalmának **KIZÁRÓ VAGY** kapcsolatát visszaírja az **ACC**-ba. Mindig **CY**=0, és **A**=0.
1.byte: 1010 1SSS
1 gépi ciklus, 4 fázis, állítja Z,S,P,CY,AC-t

XRA M
(A) <= (A) **KIZÁRÓVAGY** ((H)(L))
Az **ACC** és a **HL** tartalmával kijelölt memóriabyte **KIZÁRÓ VAGY** kapcsolatát visszaírja az **ACC**-ba. Mindig **CY**=0, és **AC**=0
1.byte: 1010 1110
2 gépi ciklus, 7 fázis, állítja Z,S,P,CY,AC-t.

XRI data
(A) <= (A) **KIZÁRÓVAGY** data
Az **ACC** és az utasítás 2. byte-jának **KIZÁRÓ VAGY** kapcsolatát visszaírja az **ACC**-ba. Mindig **CY**=0, és **AC**=0.
1. byte: 1110 1110
2.byte: data
2 gépi ciklus, 7 fázis, állítja Z,S,P,CY,AC-t.

ORA r
(A) <= (A) **VAGY** (r)

Az **ACC** és az **r** regiszter tartalmának **VAGY** kapcsolatát visszaírja az **ACC**-ba.Mindig **CY**=0, és **AC**=0.
1.byte: 1011 0SSS
1 gépi ciklus, 4 fázis, állítja Z,S,P,CY,AC-t.

ORA M
(A) <= (A) **VAGY** ((H)(L))
Az **ACC** és a **HL** tartalmával kijelölt memóriabyte **VAGY** kapcsolatát visszírja az **ACC**-ba. Mindig **CY**=0, és **AC**=0.
1.byte: 1011 0110
2 gépi ciklus, 7 fázis, állítja Z,S,P,CY,AC-t.

ORI data
(A) <= (A) **VAGY** data
Az **ACC** és az utasítás 2. byte-jának (**data**) **VAGY** kapcsolatát visszaírja az **ACC**-ba. Mindig **CY**=0, és **AC**=0.
1. byte: 1111 0110
2. byte: data
2 gépi ciklus, 7 fázis, állítja Z,S,P,CY,AC-t.

CMP r
(A) - (r)
Kivonja az **ACC**-ból az **r** regiszter tartalmát és a flag-eket állítja. **Z**=1, ha (**A**) = (**r**) és **CY**=1, ha (**A**)<(**r**).
1. byte: 1011 1SSS
1 gépi ciklus, 4 fázis, állítja Z,S,P,CY,AC-t.

CMP M
(A) - ((H)(L))
Kivonja az **ACC**-ból a **HL** tartalmával kijelölt memóriabyte-ot és beállítja a flag-eket.**Z**=1, ha (**A**)=((**H**)(**L**)) és **CY**=1, ha (**A**)<((**H**)(**L**)).
1.byte: 1011 1110
2 gépi ciklus, 7 fázis, állítja Z,S,P,CY,AC-t.

CPI data
(A) - data
Kivonja az **ACC**-ból az utasítás 2. byte-ját (**data**) és beállítja a flag-eket. **Z**=1, ha (**A**)=**data** és **CY**=1, ha (**A**)<**data**.
1.byte: 1111 1110
2.byte: data
2 gépi ciklus, 7 fázis, állítja a Z,S,P,CY,AC-t.

RLC
(An+1)<=(An), (A0)<=(A7), CY<=(A7)
Az **ACC** tartalmát körbe lépteti balra. A 7. bit átírodik a **CY**-ba és a 0.bitbe is. Csak a **CY** flag változhat.
1. byte: 0000 0111
1 gépi ciklus, 7 fázis, csak a CY-t állítja.

RRC
(An)<=(An+1), (A7)<=(A0), CY<=(A0)
Az akkumulátor tartalmát körbe lépteti jobbra . A 0. bit beírodik a **CY**-ba és a 7. bitbe is. Csak a **CY** flag változhat.
1. byte: 0000 1111
1 gépi ciklus, 4 fázis, csak a CY-t állítja.

RAL
(An+1)<=(An), CY<=(A7), (A0)<=CY
Az **ACC** és a **CY** tartalmát balra lépteti körbe, a **CY**-on keresztül. A **CY** a 0. bitbe, a 7.bit a **CY**-ba írodik át. Csak **CY** flag változhat.
1. byte: 0001 0111
1 gépi ciklus, 4 fázis, csak a CY-t állítja.

RAR
(An)<=(An+1), (A7)<=CY, CY <= (A0)
Az **ACC** és a **CY** tartalmát jobbra lépteti körbe, a **CY**-on keresztül.A **CY** a 7. bitbe, a 0. bit a **CY**-ba irodik át.Csak **CY** flag változhat.
1. byte: 0001 1111
1 gépi ciklus, 4 fázis, csak a CY-t állítja.

CMA
(A)<= **NOT** (A)
Az **ACC** tartalmát bitenként megnegálja.
1. byte: 0010 1111
1 gépi ciklus, 4 fázis, flag-eket nem állítja.

CMC
CY<= **NOT** CY
A **CY** flag-et megnegálja.
1. byte: 0011 1111
1 gépi ciklus, 4 fázis, csak CY-t állítja.

STC
CY <= 1
Logikai **1**-be állítja a **CY** flag-et.
1. byte: 0011 0111
1 gépi ciklus, 4 fázis, csak CY-t állítja.

UGRÓ UTASÍTÁSOK

Feltételek megadása:

mnem. kód	flag érték	opkód CCC
NZ	Z=0	000
Z	Z=1	001
NC	CY=0	010
C	CY=1	011
PO	P=0	100
PE	P=1	101
P	S=0	110
M	S=1	111

JMP addr
(PC) <= ((byte3)(byte2))
Feltétel nélkül az utasításban megadott címen (**addr**)folytatja a programvégrehajtást.
1. byte: 1100 0011
2. byte: addr alsó byte
3. byte: addr felsô byte
3 gépi ciklus, 10 fázis, flag-eket nem állítja.

Jcond addr
ha **cond**=IGAZ, akkor (PC) <= ((byte3)(byte2))
Ha az utasításban megadott feltétel teljesül, akkor az utasításban megadott címen (**addr**), egyébként a **Jcond** utáni utasításon folytatja a programvégrehajtást. Az utasítás végrehajtási ideje lerövidül, ha a feltétel nem teljesül.
1. byte: 11CC C010
2. byte: addr alsó byte
3. byte: addr felsô byte
2/3 ciklus, 7/10 fázis, flag-eket nem állítja.

CALL addr
((SP)-1) <= (PCfelsô)
((SP)-2) <= (PCalsó)
(SP) <= (SP)-2
(PC) <= ((byte3)(byte2))
Feltétel nélküli szubrutinhívás, amely elöször kimenti a stack-be a **CALL** utáni utasítás címét majd a szubrutin-kezdôcímmel (**addr**) felülírja a **PC**-t és kettôvel csökkenti a **SP** tartalmát.
1. byte: 1100 1101
2. byte: addr alsó byte
3. byte: addr felsô byte
5 gépi ciklus, 18 fázis, flag-eket nem állítja

Ccond addr
ha **cond**=IGAZ, akkor
((SP)-1 <= (PCfelsô)
((SP)-2 <= (PCalsó)
(SP) <= (SP)-2
(PC) <= ((byte3)(byte2))
Feltételes szubrutínhívás, amelynek eredménye a CALL utasítás eredményével azonos, ha a feltétel teljesül, egyébként a **Ccond** utáni utasításon folytatódik a programvégrehajtás. A végrehajtási idô lerövidül, ha a feltétel nem teljesül.
1. byte: 11CC C100
2. byte: addr alsó byte
3. byte: addr felsô byte
2/5 ciklus, 9/18 fázis, flag-eket nem állítja.

RET
(PCalsó) <= ((SP))
(PCfelsô) <= ((SP)+1)
(SP) <= (SP)+2
Feltétel nélküli visszatérés szubrutinból a meghívó **CALL** tip. uta-sítás utáni utasításra mutató, a stack-be kimentett visszatérési címre. A stack-bôl visszaolvasott cím felülírja a **PC**-t, a **SP** kettôvel megnô.
1. byte: 1100 1001
3 gépi ciklus, 10 fázis, flag-eket nem állítja.

Rcond
ha **cond**=IGAZ, akkor
(PCalsó) <= ((SP))
(PCfelsô) <= ((SP)+1)
(SP) <= (SP)+2
Feltételes visszatérés szubrutinból. Ha az utasításban megadott feltétel teljesül akkor a szubrutint meghívó **CALL** tip. utasítás utáni utasításra mutató, a stack-be kimentett visszatérési címen folytatódik a programvégrehajtás, egyébként az **Rcond** utáni utasításon. A stack-bôl visszaolvasott cím felülírja a **PC**-t, a **SP** kettôvel megnô. A végrehajtási idô függ a feltétel teljesülésétôl.
1. byte: 11CC C000
1/3 gépi ciklus, 6/12 fázis, flag-eket nem állítja.

RST n
((SP)-1) <= (PCfelsô)
((SP)-2) <= (PCalsó)
(SP) <= (SP)-2
(PC) <= 8***n** (ahol n=0,1,2..,7)
Egy byte-os szubrutinhívó utasítás, a szubrutin kezdôcíme az utasításban megadott **n** (a mûveleti kód **NNN** bitjeinek deci-mális értéke) szám nyolcszorosa (decimális 0, 8, ..48, 56 mivel **n**<8).
1. byte: 11NN N111
3 gépi ciklus, 12 fázis, flag-eket nem állítja.

PCHL
(PCfelsô) <= (H)
(PCalsó) <= (L)
Az utasítás átírja a **HL** regiszterpár tartalmát a **PC**-be, így a programvégrehajtás a **HL**-ben tárolt címen folytatódik.
1. byte: 1110 1001
1 gépi ciklus, 6 fázis, flag-eket nem állítja.

<u>Stack, I/O és vezérlô utasítások</u>

PUSH rp
((SP)-1 <= (rp_felsô)
((SP)-2 <= (rp_alsó)
(SP) <= (SP)-2
Az kijelölt **rp** regiszterpár (**SP** nem lehet !) tartalmát kimenti a stack-be, a **SP**-t kettôvel csökkenti.
1. byte: 11RP 0101

3 gépi ciklus, 12 fázis, flag-eket nem állítja

PUSH PSW
((SP)-1 <= (A)
((SP)-2 <= (flag_byte)
(SP) <= (SP)-2
A (**SP**)-1 címre az **ACC** tartalmát, a (**SP**)-2 címre a flag-ekbôl képzett byte-ot menti ki. A **SP**-t kettôvel csökkenti.
Flag_byte:

D7	D6	D5	D4	D3	D2	D1	D0
S	Z	X	AC	X	P	X	CY

ahol X=közömbös
1. byte: 1111 0101
3 gépi ciklus, 12 fázis, flag-eket nem állítja.

POP rp
(rp_alsó) <= ((SP))
(rp_felsô) <= ((SP)+1)
(SP) <= (SP)+2
A **SP** által kijelölt két memóriabyte tartalmát (stack-be kimentett érték) átírja a kijelölt **rp** regiszterpárba. A **SP**-t kettôvel megnöveli.
1. byte: 11RP 0001
3 gépi ciklus, 10 fázis, flag-eket nem állítja.

POP PSW
(flag_byte) <= ((SP)
(A) <= ((SP)+1
A stack-bôl (amelyet a **SP** elölt ki) visszaállítja a flag-ek, valamint az **ACC** tartalmát. A **SP**-t kettôvel megnöveli.
1. byte: 1111 0001
3 gépi ciklus, 10 fázis, állítja Z,S,P,CY,AC-t

XTHL
(L) <= > ((SP))
(H) <=> ((SP)+1
Megcseréli a **HL** regiszterpár és a stack 2 legfelsô byte-jának (**SP** jelöli ki) tartalmát. A **SP** nem változik.
1. byte: 1110 0011
5 gépi ciklus, 16 fázis, flag-eket nem állítja.

SPHL
(SP) <= (H)(L)
A **HL** regiszterpár tartalmát közvetlenül átírja a **SP**-be.
1.byte: 1111 1001
1 gépi ciklus, 6 fázis, flag-eket nem állítja.

IN port
(A) <= input port
Az **ACC**-ba írja az utasításban megadott címû bemeneti port-ról beolvasott byte-ot.
1. byte: 1101 1011
2. byte: port
3 gépi ciklus, 10 fázis, flag-eket nem állítja.

OUT port
output port <= (A)
Az ACC tartalmát kiviszi az utasításban megadott címû kimeneti portra.
1. byte: 1101 0011
2. byte: port
3 gépi ciklus, 10 fázis, flag-eket nem állítja.

EI
INTE FF <= 1
A megszakítás elfogadása engedélyezett lesz az **EI u**táni utasí-tás végrehajtása **után.**
1.byte: 1111 1011
1 gépi ciklus, 4 fázis, flag-eket nem állítja.

DI
INTE FF <= 0
A megszakításkérés elfogadása letiltottá válik közvetlenül a **DI** utasítás végrehajtása után. Az utasítás végrehajtása közben nem figyeli a megszakításkéréseket.
1. byte: 1111 0011
1 gépi ciklus, 4 fázis, flag-eket nem állítja.

HLT
A CPU HALT állapotba kerül a **HLT** végrehajtása közben kiadott 2. ALE impulzus után (2. gépi cklus elején).
1. byte: 0111 0110
1.5 gépi ciklus, 5 állapot, flag-eket nem állítja.

NOP
Üres utasítás bármely adat megváltozása vagy mûvelet nélkül.
1. byte: 0000 0000
1 gépi ciklus, 4 fázis, flag-eket nem állítja.

RIM
(A) <= megszakítás maszk
Az utasítás az ACC-ba olvassa az alábbi információkat:
A0: RST5.5 maszkbitje (ha 1, akkor tiltott)
A1: RST6.5 maszkbitje (ha 1, akkor tiltott)
A2: RST7.5 maszkbitje (ha 1, akkor tiltott)
A3: INTE FF (általános megszakít. engedélyezés)
A4: megszakításkérés az RST5.5-ön

A5: megszakításkérés az RST6.5-ön
A6: megszakításkérés az RST7.5-ön
A7: SID bemenetrôl (soros adatbemenet) beolvasott érték
1. byte: 0010 0000
1 gépi ciklus, 4 fázis, flag-eket nem állítja.

SIM
Az utasítás **elôtt** betöltött **ACC** tartalom alapján beállítja a megszakítás-maszkokat és a **SOD** soros adatkimenetet.
A3=1 engedélyezi a maszkbeállítást az **A0,A1,A2** biteken
A0=1 és **A3**=1 letiltja az RST5.5 megszakítást,
A1=1 és **A3**=1 letiltja az RST6.5 megszakítást,
A2=1 és **A3**=1 letiltja az RST7.5 megszakítást,
A4=1 törli az RST7.5 tároló flip-flopot,
A5: közömbös
A6=1 engedélyezi a SOD kimenet beállítását,
A7 => SOD, ha **A7**=1.
1. byte: 0011 0000
1 gépi ciklus, 4 fázis, flag-eket nem állítja.

JELÖLÉSEK

bitek sorszámozása: 7 = byte legfelsô helyértéke
0 = byte legkisebb helyértéke
(..) : regiszter, regiszterpár, vagy memória tartalma
A, ACC : 8 bites akkumulátor
addr : 16 bites memóriacím
data : 8 bites adat
data16 : 16 bites adat
byte2 : utasítás 2. byte-ja
byte3 : utasítás 3. byte-ja
port : kiviteli vagy beviteli port 8 bites közvetlen címe
r, r1, r2 : az A,B,C,D,E,H,L 8 bites regiszter
(jelölésük a mnemonikban: A B C D E H L)
rp : a BC, DE, HL, SP 16 bites regiszterpár
(jelölésük a mnemonikban: B D H SP)
DDD,SSS: regiszterkijelölô bitkombináció az utasításkódban
111 => A
000 => B
001 => C
010 => D
011 => E
100 => H
101 => L
RP : regiszterpárt kijelölô bitkombináció az utasításban
00 => BC
01 => DE
10 => HL
11 => SP
PC : 16 bites programszámláló
SP : 16 bites stack pointer
Ai : az akkumulátor i-edik bitje
Z,S,P,CY,AC: flag-ek
Z => zero flag
S => elöjel (sign) flag
P => páratlan paritás flag
CY => átvitel (carry) flag
AC => segédátvitel (auxiliary carry) flag
<= , = > : egyirányú adatmozgatás iránya
<=> : kétirányú adatcsere
NNN : 3bites kombináció, amely 0..7 közötti számot jelöl
n : az NNN kombinációval decimális szám