LABORATEGIA

ERREPASOA

8085ean **oinarri hamaseitarra** erabiltzen da.

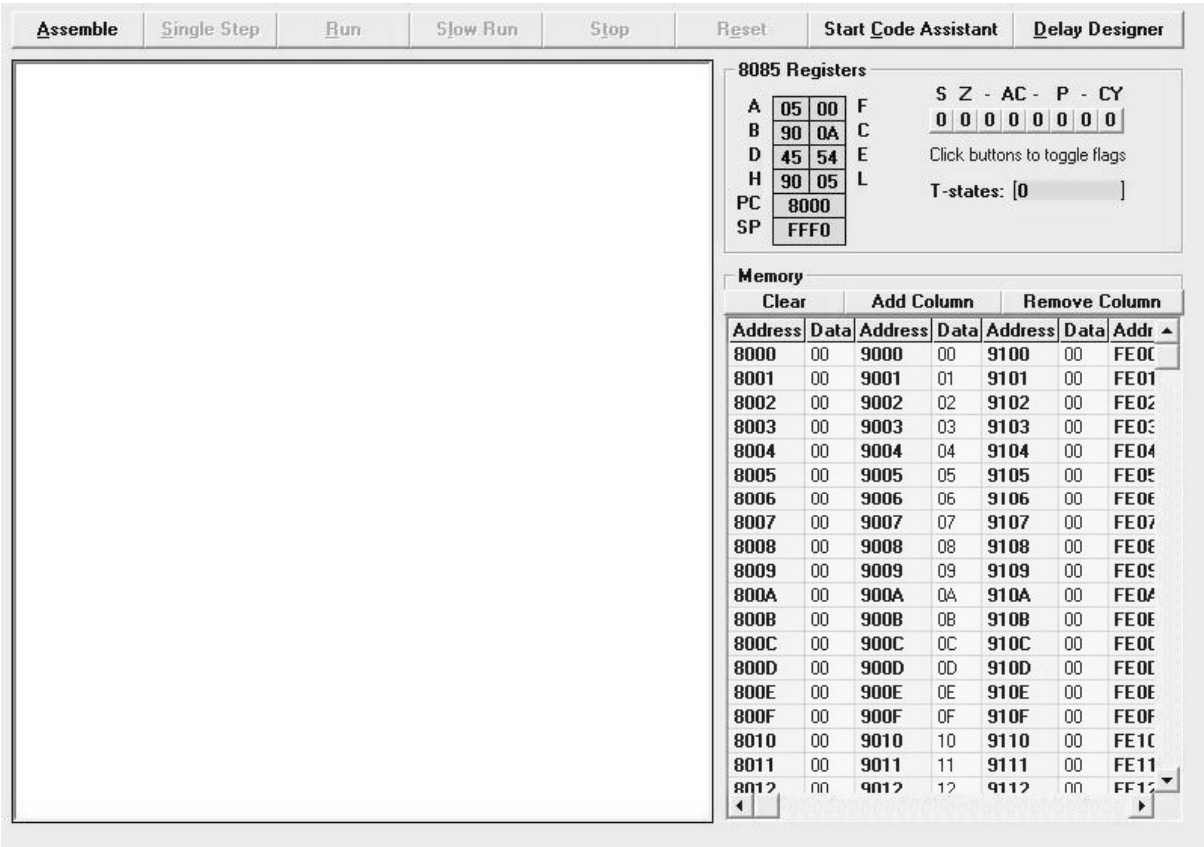
a) Zein balio har ditzake digitu hamaseitar batek?

b) Zenbat digitu bitar dagozkio digitu hamaseitar bati?

c) Zenbat digitu hamaseitar dituzte erregistroek? Beraz, zein balio adieraz daitezke hamaseitarrean, bitarrean eta hamartarrean?

d) Zenbat digitu behar dira memoriako posizio bat zehazteko? Orduan, zenbat erregistro behar dira memoriaren erakusle moduan erabili ahal izateko?

TRANSFERENTZIA-AGINDUAK



1. Goiko irudiak adierazten digun 8085aren egoeran honako agindu hauek exekutatuko balira, zein balio geratuko lirateke mikroprozesadorearen erregistroetan? Osatu taula, agindu bakoitza egoera horretatik abiatuta (ez hiru aginduak jarraian).

		A	B	C	D	E	H	L
a)	**mov b,d**							
b)	**mov a,m**							
c)	**mov m,e**							

d) Hiruretako batek ez du erregistro horietatik bat ere aldatzen, memoriako eduki bat baizik. Zein aginduk? Zein da aldatzen duen memoriako posizioa? Zein aldaketa eragiten du?

e) Zer da **F** erregistroa (zeren berri ematen du)? Goiko eragiketa horiekin aldatuko al da bere balioa? Zergatik?

f) Zer da **PC** erregistroa? Goiko eragiketa horiekin aldatuko al da? Zergatik eta nola?

2. Memoriako 9003 posizioan dagoen datua 9008ra eraman nahi da. Gauzatu mugimendu hori bi agindurekin.

3. 1. ariketako irudiko egoeratik abiatuz, esan zer gertatuko litzatekeen honako ariketa hauek exekutatzean (zer aldatzen den eta nola):

a) **sta** 9001

b) **lda** 9001

c) **shld** 9001

d) **lhld** 9001

4. 1. ariketan, zein erregistro-bikote erabili da memoriaren erakusle gisa? Nola hasieratu daiteke? Hau da, 900Ah posizioaren erakusle moduan erabili nahi badugu, zein eragiketa exekutatu beharko dugu?
Oharra: *memorian datu array bat izatekotan, arrayaren lehen elementuaren helbidearekin hasieratu dezakegu erakuslea, eta gero, erakuslea gehituz, arraya zeharkatu.*

5. Zein desberdintasun dago honako agindu hauen artean? (Azaldu bakoitzak zer egiten duen)

a) **lda addr** *(A) ← [addr]*

b) **ldax rp** *(A) ← [(rp)]*

c) **sta addr** *[addr] ← (A)*

d) **stax rp** *[(rp)] ← (A)*

6. Zein da **xchg** aginduaren eragina? Zertarako izango zaigu erabilgarri?

AGINDU ARITMETIKOAK

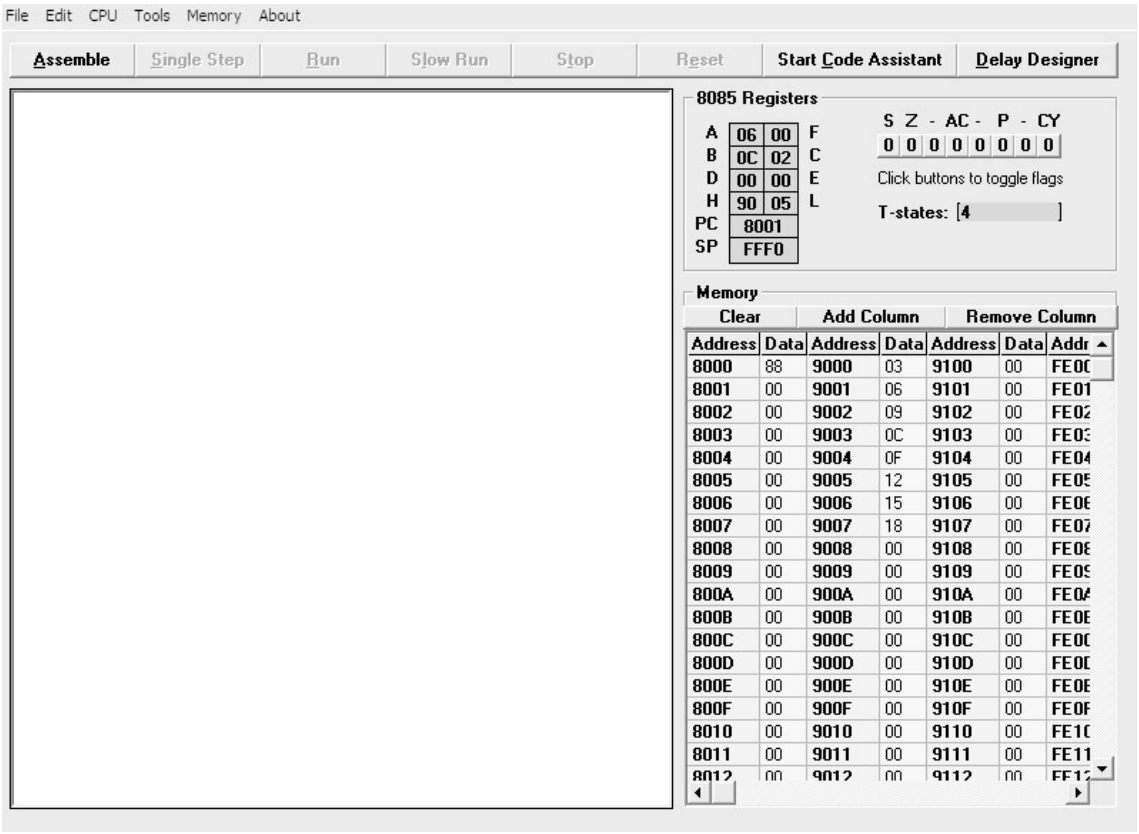
Eragiketa aritmetiko gehienetan metagailuak parte hartzen du. Aginduan erregistro bat edo memoria posizio bat zehazten da modu esplizituan, eta metagailuan egongo da eragiketa gauzatzeko behar den bigarren elementua. Eragiketaren emaitza metagailuan idatziko da, eta agindua exekutatu aurretik metagailuan zegoen datua galduko da. Beraz:

a) Metagailuan dagoen datua berriro erabili nahiko bada, eragiketa exekutatu aurretik beste nonbait (erregistro batean edo memoriako posizio batean) gorde beharko dugu (transferentzia-aginduak erabiliz).
b) Lortutako emaitza metagailuan baino ez dago; hurrengo agindua exekutatzean nahigabean galdu ez dezagun, beste nonbait kopiatu beharko da.

Transferentzia-aginduetan ez bezala, hauetan flag (indikadore) gehienak aktibatzen dira.

1. Honako aginduek gauzatzen dutena azaldu eta jatorriko egoera irudikoa izanik, esan zein emaitza geratuko litzatekeen metagailuan (lehenengoa adibide moduan eginda dago).

***Oharra**: aginduak ez dira bata bestearen atzean exekutatzen*



a) ADD b

b) ADD c

c) ADD a

d) ADD m

e) ADI 08

f) ADC b

g) ADC c

h) ADC m

i) ACI 08

j) SUB b

k) SUB c

l) SUB a

m) SUB m

n) SUI 08

o) SBB b

p)SBB c

q) SBB m

r) SBI 08

2. 9000h posiziotik aurrera 12 elementuko array bat dago. Demagun begizta bat egiten dugula, zenbatzaile baten laguntzaz arrayaren elementuak banan-banan gehitzeko. Idatzi honako pauso hauek gauzatzeko beharko diren aginduak:

a) Hasieratu erakusle moduan erabiliko den erregistro-bikotea.

a) Begiztaren barruan, gehitu arrayaren i. elementua (egokitzen dena).

b) Erakuslea eguneratu (orain arrayaren hurrengo elementua erakuts dezan)

AGINDU LOGIKOAK

1) Zein agindu logikok eraldatzen dituzte flag direlakoak? Zein flag-engan dute eragina?

2) a) Idatzi AND, OR, XOR eta NOT oinarrizko eragiketa logikoen egia-taula.

b) Nola gauzatu NOT eragiketa XOR ate batekin?

3) 8085ean agindu multzo logikoa AND, XOR, OR eta CMPk osatzen dute. CMPk konparaketa bat egiten du emandako erregistro (CMP r), memoria-eduki (CMP M) edo zehaztutako balio baten (CMP byte) eta metagailuaren balioaren artean. Galtzen al da metagailuan zegoen balioa? Non gordetzen da emaitza? Nola izango dugu erabilgarri eragiketa horren emaitza?

4) MASKARAK. 8 digituko zenbaki bitar ezezagun bat daukagu ($XXXXXXXX_2$). Agindu logikoak erabilita:

a) Nola jarri bit guztiak '1' (11111111_2)?

b) Nola jarri bit guztiak '0'an (00000000_2)?

c) Nola jarri 4 bit esanguratsuenak '1'ean, azken 4 biten balioa aldatu gabe?

d) Nola jarri 4 bit esanguratsuenak '0'an, azken 4 biten balioa aldatu gabe?

e) Nola jarri bit guztiak '0'an MSB delakoa izan ezik (hau dagoen bezala mantendu)? Zein dira eragiketa horren ahalezko emaitzak?

5) Demagun aurreko ariketako zenbaki bitar ezezaguna 10011101_2 dela. Jarri hamaseitarrean zenbaki hori eta aurreko ariketako a)-e) ataletako maskarak.

6) Memoriako 9002 posizioan dagoen datuarekin, honako eragiket hauek gauzatu nahi dira:

a) Bere bit guztiak ezeztatu eta emaitza 9003 posizioan utzi.

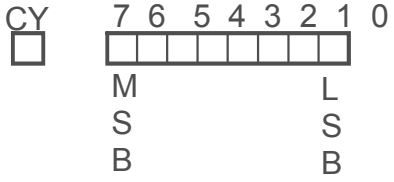
b) lsb aldatu gabe, beste guztiak '1'ean jarri eta emaitza 9004 posizioan utzi.

c) MSB aldatu gabe, beste guztiak '0'an jarri eta emaitza 9005 posizioan utzi.

7) Memoriako 9100 posizioan dagoen datua 9101ekoa baino handiagoa ala txikiagoa den jakin nahi da. Zer egin dezakegu (eragiketa logikoak erabilita) hori jakiteko? Gauza bera (9100eko datua) b erregistroan dagoen datuarekin eta 12_{10} zenbakiarekin.

BIRAKETA AGINDUAK

1) Irudikatu RLC, RRC, RAL eta RAR aginduek erregistroaren bitengan duten eragina.



2) Zein erregistroren edukia biratzen da?

3) Erregistroaren edukia 40h da. Zein da bi RLC edo bi RAL eragiketa jarraian egitearen arteko desberdintasuna? Gauza bera edukia 01h bada eta RRC edo RAR (behin) exekutatzen bada.

4) Erregistro horren edukia 10001011_2 bada, zein da bere balio hamartarra? Zein izango da RLC eta RRC eragiketen emaitza (bai oinarri bitarrean, bai hamartarrean ere). Eta edukia 00010000_2 den kasuan? (Balio hamartarra eta biraketen emaitzak). Orduan, biraketa ordez desplazamendua egingo bagenu eta hustutako posizioak betetzeko '0' bat sartuko bagenu, zer lortzen dugu zenbaki bitar bat posizio bat ezkerrera higitzen dugunean? Eta eskuinera higitzean?

5) Zein da zenbaki baten baterako osagarria (definizioa eta praktikan nola egiten dugun)? Eta birako osagarria?

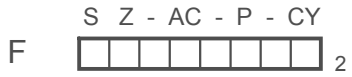
6) CMA eta CMC aginduen arteko ezberdintasunak (azaldu bakoitzak egiten duena).

ADARKATZE-AGINDUAK

1- Bete ezazu hurrengo taulan egoera bitek (flag direlakoek) balio-bata edo bestea hartzen dutenean adierazten dutena.

Egoera-bita	Balioa	Esanahia: aurreko eragiketan…
S	0	
S	1	
Z	0	
Z	1	
P	0	
P	1	
CY	0	
CY	1	

2- Nolakoa izan da aurreko eragiketaren emaitza F erregistroaren balioa honako hau bada?



a) F=04h

b) F=11h

c) F=C4h

3- Aurreko aginduaren emaitza negatiboa bada, jauzi egin NEGATIVO etiketara.

4- Emaitza negatiboa bada, jauzi egin NEGera; bestela EZ_NEGera.

ARIKETA OSOAK

1- Azaldu zer egiten duen honako programa honek.

```
mvi a,00
buklea:
lda 9000
cpi 00
jz fin
jm negatiboa
jmp buklea
negatiboa:
;hemen jartzen ez ditugun agindu batzuk egongo dira
amaiera:
```

2- Zer egiten du honako programa honek?

```
mvi a,00
buklea:
lda 9000
mov b,a
lda 9001
sub b
jz amaiera
jmp ezberdinak
ezberdinak:
;hemen jartzen ez ditugun agindu batzuk egongo dira
amaiera:
```

3- 9000 posizoan array baten lehen elementua dago. Array hori 10 elementuz osatuta dago, eta badakigu guztiak balio positiboak direla. Idatzi arrayaren elementuak zeharkatzen dituen programa bat, balio bakoitza dekrementatuz. Arrayaren amaierara iristean berriro hasiko da, beste unitate batean gehidekrementatzeko. Horrela jarraituko du balio horietako bat zerora iritsi arte.

4- Azaldu zer egiten duen honako programa hauek, agindu bakoitzak eta programa osoak.

```
lhld 9100
mvi b, 00
mvi c, 10
mvi d, 08
buklea:
mov a,d
cmp m
jm hurrengoa
inr b
hurrengoa:
inx hl
dcr c
jz amaiera
jmp buklea
amaiera:
```

ERREPASOA

8085ean **oinarri hamaseitarra** erabiltzen da.

a) [Emaitza]: 16 balio desberdin har ditzake: 0,1,2,3,4,5,6,7,8,9,A,B,C,D,E eta F

b) [Emaitza]: Lau zifra bitar.

c) [Emaitza]: 2 digitu hamaseitar; beraz, adieraz daitekeen baliorik handiena $FF_{16}=11111111_2=255_{10}$

d) [Emaitza]: Lau digitu hamaseitar (16 bitar). Erregistro bakoitzak bi hamaseitar (zortzi bitar) baino ez ditu, beraz, **bi erregistro** behar dira.

TRANSFERENTZIA-AGINDUAK

1.

		A	B	C	D	E	H	L
a)	**mov b,d**	05	45	0A	45	54	90	05
b)	**mov a,m**	05	90	0A	45	54	90	05
c)	**mov m,e**	05	90	0A	45	54	90	05

d) [Emaitza]: c) aginduak. HL erregistro-bikoteak erakutsitako memoria-helbidea aldatzen du; kasu honetan, 9005 memoria helbidean E erregistroaren edukia idatziko du, 54 balioa, hain zuzen ere.

e) [Emaitza]: Egoera biten (S, Z, AC, P, CY) informazioa duen erregistroa da. Aurreko eragiketa guztiak transferentziakoak izanik (eta horiek flag-engan eraginik ez dutenez) ez da aldaketarik izango beren balioan.

f) [Emaitza]: PC erregistroa programaren zenbatzailea da (exekutatu behar den hurrengo aginduaren helbidea du). Aurreko eragiketek zuzenean eraldatzen ez badute ere, edozein eragiketaren exekuzioak erregistro horren gehikuntza automatiko bat dakar, oraingo eragiketaren ondoren exekutatu behar denaren helbidea erakuts dezan.

2.
[Emaitza]: **lda 9003** (A)←[9003]
sta 9008 [9008]←(A)

3.
a) **sta** 9001
[Emaitza]: **[9001]←(A)** memoriako 9001h helbidean gordeko du (metagailuaren balioa) 05h

b) **lda** 9001
[Emaitza]: **(A)←[9001]** metagailuan gordeko du memoriako (9001h helbidean dagoen) 01h.

c) **shld** 9001
[Emaitza]: **[9001]←(L) eta [9002]←(H),** 9001 memoriako helbidean (L erregistroaren edukia) 05h gordeko du, eta hurrengo posizioan, hau da, 9002h helbidean, (Hren edukia) 90h.

d) **lhld** 9001
[Emaitza]: **(L)←[9001] eta (H)←[9002],** L erregistroan (9001h helbidean dagoen datua) 01h balioa gordeko du, eta H erregistroan (9002h posizioko datua) 02h balioa.

4.
[Emaitza]: HL erregistro-bikotea erabili da. **lxi h, 900A**:

5.
a) **lda addr** *(A) ← [addr]*
[Emaitza]: metagailuan addr memoriako helbidearen edukia idazten da.

b) **ldax rp** *(A) ← [(rp)]*
[Emaitza]: rp erregistro-bikoteak zehazten duen memoria-helbidean dagoen datua idazten da metagailuan.

c) **sta addr** *[addr] ← (A)*
[Emaitza]: Emandako addr helbidean metagailuaren balioa gordetzen da.

d) **stax rp** *[(rp)] ← (A)*
[Emaitza]: rp erregistro-bikoteak zehaztutako memoria-helbidean gordetzen da metagailuaren balioa.

Beraz, a eta c kasuetan zuzenean zehazten da memoria-helbidea; b eta d kasuetan, berriz, erregistro-bikote bat erabiltzen da memoriaren erakusle gisa.

6.
[Emaitza]: (H)←→(D) eta (L)←→(E) HL eta DE erregistro-bikoteen edukiak trukatzen ditu. Begizta batean bi array batera maneiatu nahi badira, HL bikotea denez mov eragiketekin erabil daitekeen bakarra, erakusle bien edukiak trukatuz goaz.

(H)←→(D) y **(L)←→(E)**

AGINDU ARITMETIKOAK

1.
a) ADD b
[Emaitza]: (A) ← (A)+(B); 12h ← 06h+0Ch; metagailuan 06h zegoen, B erregistroan 0Ch (12_{10}); beraz, metagailuan gordeko den emaitza 12h da, (6_{10}+12_{10}=18_{10}) =12h.

b) ADD c
[Emaitza]: (A) ← (A)+(C); 08h ← 06h+02h; metagailuan 06h zegoen, eta C erregistroan 02h; beraz, metagailuan gordeko den emaitza 08h da, (6_{10}+2_{10}=8_{10}) =08h.

c) ADD a
[Emaitza]: (A) ← (A)+(A); 0Ch ← 06h+06h; metagailuan 06h zegoenez, bertan honako emaitza hau gordeko da:(6_{10}+6_{10}=12_{10}) =0Ch.

d) ADD m
[Emaitza]: (A) ← (A)+[(HL)]; 18h ← 06h+12h; metagailuan 06h balioa zegoen eta HL erregistro-bikoteak erakutsitako memoria-posizioan (hau da, 9005h helbidean) 12h (18_{10}), zegoen; beraz, (6_{10}+18_{10}=24_{10}) =18h da metagailuan gordeko den emaitza.

e) ADI 08
[Emaitza]: (A) ← (A)+08h; 0Eh ← 06h+08h; aginduaren mnemonikoan «i» horrek esaten digu aginduaren eragigaia berehalako datua dela, alegia 08h zuzenean datua dela (batugaia, kasu honetan). Metagailuan 06h zegoenez, bertan gordeko den emaitza ($6_{10}+8_{10}=14_{10}$)=0Eh izango da.

f) ADC b
[Emaitza]: (A) ← (A)+(B)+(CY); 12h ← 06h+0Ch+0b. Aginduaren mnemonikoan agertzen den «C» horrek esaten digu bururakoak ere parte hartzen duela eragiketan (batuketa, kasu honetan). Baina irudian CY='0' dela ikusten dugunez (00h=0_{10}), eragiketa horrek eta a) atalekoak emaitza bera emango dute: ($6_{10}+12_{10}+0_{10}=18_{10}$) =12h.

g) ADC c
[Emaitza]: (A) ← (A)+(C)+(CY); 08h ← 06h+02h+0b. Metagailuan 06h balioa zegoen, C erregistroan 02h eta bururakoaren bitak «0» balioa duenez (ez dago bururakorik gehitzeko), metagailuan ($6_{10}+2_{10}+0_{10}=8_{10}$) 08h balioa geratuko da.

h) ADC m
[Emaitza]: (A) ← (A)+[(HL)]+(CY); 18h ← 06h+12h+0b. Metagailuan 06h zegoen, HL erregistro-bikoteak zehaztutako memoria-helbidean (9005h) 12h eta bururako bitak «0» balio du; beraz, metagailuan geratuko den emaitza ($6_{10}+18_{10}+0_{10}=24_{10}$) 18h izango da.

i) ACI 08
[Emaitza]: (A) ← (A)+08h+(CY); 0Eh ← 06h+08h+0b; metagailuan 06h balioa zegoen, bururako bita «0» da eta berehalako eragigaia 08h; beraz, metagailuan geratzen den emaitza ($6_{10}+8_{10}+0_{10}=14_{10}$) 0Eh da.

j) SUB b
[Emaitza]: (A) ← (A)-(B); FAh ← 06h-0Ch; b erregistroan dagoen balioa (0Ch=12_{10}) metagailuan dagoena (06h=6_{10}) baino handiagoa denez, kenketaren emaitza balio negatibo bat izango da: ($6_{10}-12_{10}=-6_{10}$) FAh.

k) SUB c
(A) ← (A)-(C); 04h ← 06h-02h; metagailuan 06h eta C erregistroan 02h balioak genituenez, metagailuan gordeko den emaitza ($6_{10}-2_{10}=4_{10}$) 04h da.

l) SUB a
[Emaitza]: (A) ← (A)-(A); 00h ← 06h-06h; metagailuari bere balio bera kendu badiogu, metagailuan geratuko den emaitza ($6_{10}-6_{10}=0_{10}$) 00h izango da.

m) SUB m
[Emaitza]: (A) ← (A)-[(HL)]; F4h ← 06h-12h. Metagailuan 06h balioa zuen eta HL erregistro-bikoteak markatzen duen memoria-helbideak (9005h) 12h (18_{10}); hortaz, metagailuan geratzen den emaitza ($6_{10}-18_{10}=-12_{10}$) F4h izango da.

n) SUI 08
[Emaitza]: (A) ← (A)-08h; FEh ← 06h-08h. Metagailuan zegoen balioari (06h) berehalako eragigaia (08h) kentzen badiogu, metagailuan geratzen den emaitza ($6_{10}-8_{10}=-2_{10}$) =FEh izango da.

o) SBB b
[Emaitza]: (A) ← (A)-(B)-(CY); FAh ← 06h-0Ch-0b. Metagailuan 06h balioa zegoen, B erregistroan 0Ch (12_{10}) eta bururako bitak «0» balio du; beraz, metagailuan geratzen den balioa (j ataleko ariketan bezala) ($6_{10}-12_{10}-0_{10}=-6_{10}$) =FAh da.

p) SBB c
[Emaitza]: (A) ← (A)-(C)-(CY); 04h ← 06h-02h-0b. Bururakoa zeroan dagoenez, metagailuan zegoen 06h balioari C erregistroko 02h balioa kentzen zaio, eta emaitza metagailuan geratzen da ($6_{10}-2_{10}-0_{10}=4_{10}$) 04h.

q) SBB m
[Emaitza]: (A) ← (A)-[(HL)]-(CY); F4h ← 06h-12h-0b. Metagailuan 06h zegoen eta HL erregistroak zehaztutako memoria-helbidean (9005h) 12h (18_{10}) balioa zegoen. Bururako bita «0» izanik, metagailuan gordeko den emaitza (6_{10}-18_{10}-0_{10}=-12_{10}) F4h izango da.

r) SBI 08
[Emaitza]: (A) ← (A)-08h-(CY); FEh ← 06h-08h-0b; metagailuan 06h zegoen; berehalako eragigaia 08h da eta bururakoa zehazten duen bita «0»; hortaz, (6_{10}-8_{10}-0_{10}=-2_{10}) FEh emaitza gordeko da metagailuan.

2.
a) [Emaitza]: HL erregistro-bikotea erabiliko dugu erakusle moduan; beraz, arrayaren lehen elementuak duen memoriako helbidearekin hasieratuko dugu:

lxi h, 9000 (HL) ← 9000h

b) [Emaitza]: HL erakuslea eguneratuta badago, i. elementuaren helbidea izango du; hortaz, honako agindu hau erabil daiteke:

inr m [(HL)] ← [(HL)]+1

c) [Emaitza]: Arrayaren elementuak ondoz ondoko memoria-helbideetan daudenez, HL erregistro-bikoteak duen balioa unitate batean gehitzearekin nahikoa izango da horrek arrayaren hurrengo elementua erakuts dezan:

inx h (HL) ← (HL)+1

AGINDU LOGIKOAK

1) [Emaitza]: Agindu logiko guztiek flag guztiengan dute eragina.

2) a) [Emaitza]:

AND		
X	Y	F
0	0	0
0	1	0
1	0	0
1	1	1

OR		
X	Y	F
0	0	0
0	1	1
1	0	1
1	1	1

XOR		
X	Y	F
0	0	0
0	1	1
1	0	1
1	1	0

NOT	
X	Y
0	1
1	0

b) [Emaitza]: XOREn egia-taulan, ikus daiteke x=1 denean, F beti Yren kontrakoa dela. Beraz, nahikoa da XOREn bi sarreren bat =1 dela egiaztatzearekin.

3) [Emaitza]: Metagailuko balioa ez da galtzen. Emaitza berez inon gordetzen ez bada ere, 1. galderan esan dugun bezala egoera bit guztietan eragina izango duenez, horiek esango digute datu bat bestea baino handiagoa, txikiagoa edo berdina den. Horrela, konparaketaren emaitzan baldintzatutako jauziak egiteko erabilgarri izango zaigu.

4)

a) [Emaitza]: A+1=1, beraz, egin beharreko eragiketa: **ORI FFh** (FF_{16})=(11111111_2).
b) [Emaitza]: A·0=0, beraz, egin beharreko eragiketa: **ANI 00h** (00_{16})=(00000000_2).
c) [Emaitza]: A+1=1 eta A+0=A, beraz, eragiketa: **ORI F0h** ($F0_{16}$)=(11110000_2).
d) [Emaitza]: A·0=0 y A·1=A, beraz, eragiketa **ANI 0Fh** ($0F_{16}$)=(00001111_2)
e) [Emaitza]: A·0=0 y A·1=A, beraz, eragiketa **ANI 80h** (80_{16})=(10000000_2)
MSB=0 bada emaitza: **00h**, berriz, MSB=1 bada emaitza: **80h**.

5) [Emaitza]:

a) Maskara FFh, Emaitza FFh.
b) Maskara 00h, Emaitza 00h.
c) Maskara F0h, Emaitza FDh.
d) Maskara 0Fh, Emaitza 0Dh.
e) Maskara 80h, Emaitza 80h.

6) [Emaitza]:

a)
LDA 9002h
ANI 00h
STA 9003h

b)
LDA 9002h
ORI FEh
STA 9004h

c)
LDA 9002h
ANI 80h
STA 9005h

7) [Emaitza]:

a) lda 9100 (A) ← [9100]
lxi h, 9101 (HL) ← 9101
cmp m (A)-[(HL)]

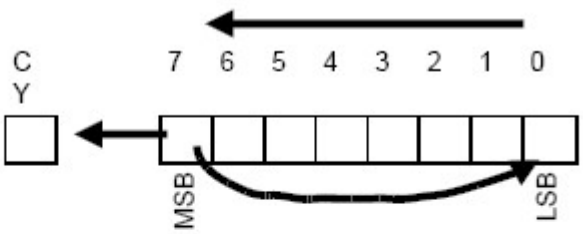
b) lda 9100 (A) ← [9100]
cmp b (A)-(B)

c) lda 9100 (A) ← [9100]
cpi 0C (A) – 0Ch (0Ch=12_{10})

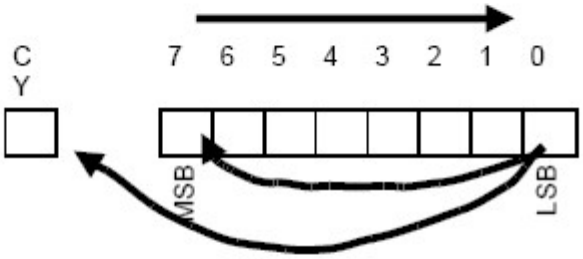
BIRAKETA-AGINDUAK

1) [Emaitza]:

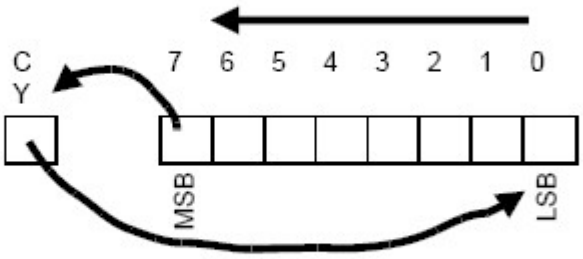
RLC



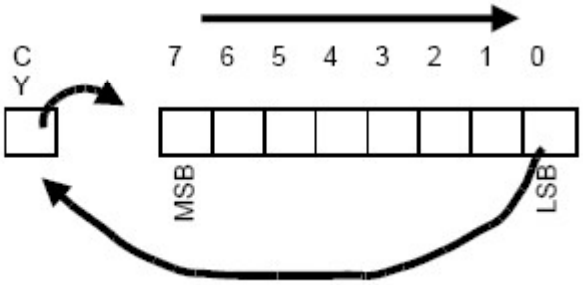
RRC



RAL



RAR



2) [Emaitza]: Metagailuan, A erregistroan.

3) [Emaitza]:

A=$01000000_2=40_{16}=64_{10}$

RLC A=$10000000_2=80_{16}=128_{10}$ CY=0

RLC A=$00000001_2=01_{16}=01_{10}$ CY=1

A=$01000000_2=40_{16}=64_{10}$ exekutatu aurretik CY=0 suposatuz:

RAL A=$10000000_2=80_{16}=128_{10}$ CY=0

RAL A=$00000000_2=00_{16}=0_{10}$ CY=1

A=00000001_2=01_{16}=1_{10}

RRC A=10000000_2=80_{16}=128_{10} CY=1

RAR A=11000000_2=$C0_{16}$=192_{10} CY=0

4) [Emaitza]:

A=100010112=8B16=13910

RLC A=000101112=1716=**2310 CY=1**

RRC A=110001012=C516=**19710CY=1**

A=000100002=1016=1610

RLC A=001000002=2016=**3210 CY=0**

RRC A=000010002=0816=**810 CY=0**

Ezkerrerantz higitzean, bit bakoitzaren pisua handitu egiten da; zenbakia 2rekin biderkatzen da beraz, baina MSB (b_{n-1}) galtzen denez, balio hori kendu behar zaio: **shl (N)= $2 \cdot N - b_{n-1} \cdot 2^n$**

Eskuinerantz higitzean, bit bakoitzaren pisua txikitu egiten da; ondorioz, zenbakia 2rekin zatitzen da, baina lsb (b_0) galtzen denez, balio hori kendu behar zaio: **shr (N)= $N/2 - b_0 \cdot 2_{-1}$**

5) [Emaitza]:

A zenbakiaren baterako osagarria:

2^n-1-|A| ; n: A zenbakiaren bit kopurua.

Praktikan: A erregistroaren bit guztien ezezkoa egin.

A zenbakiaren birako osagarria:

2^n-|A| ; n: A zenbakiaren bit kopurua.

Praktikan: A zenbakiaren baterako osagarriari bat gehitu.

6) [Emaitza]:

C**MA** aginduak A erregistroan dauden biten ezezkoa egiten du.

CMC aginduak CY flag-aren ezezkoa egiten du.

ADARKATZE-AGINDUAK

1-

Egoera bita	Balioa	Esanahia: aurreko eragiketan…
S	0	emaitza positiboa da
S	1	emaitza negatiboa da
Z	0	emaitza zeroren desberdina da
Z	1	emaitza zero da
P	0	emaitza parea da (1 zenbaki parea)
P	1	emaitza bakoitia da (1 zenbaki bakoitia)
CY	0	emaitzan ez da bururakorik egon
CY	1	emaitzan bururakoa egon da

2-

a) F=04h
[Emaitza]: F=04h=00000100_2 ; S=0; Z=0; AC=0; P=1; CY=0;
Emaitza bakoitia, positiboa, zeroren desberdina eta ez da bururakorik egon.

b) F=11h
[Emaitza]: F=11h=00010001_2 ; S=0; Z=0; AC=1; P=0; CY=1;
Emaitza bakoitia, positiboa eta zeroren desberdina da. Gainera, barne bururako bat (3. bitetik 4.era) (AC='1') eta bururako bat (CY='1') eman dira.

c) F=C4h
[Emaitza]: F=C4h=11000100_2 ; S=1; Z=1; AC=0; P=1; CY=0;
Ez da posible emaitza zero (Z='1'), negatiboa (S='1') eta bakoitia (P='1') izatea.

3- [Emaitza]: jm NEGATIBOA
(…)
NEGATIBOA:

4- [Emaitza]: jm NEG
jmp EZ_NEG
(…)
NEG:
(…)
EZ_NEG:

ARIKETA OSOAK

1-

```
mvi a,00        Metagailua zerora hasieratzan da,
buklea:
lda 9000        9000h-ko edukia irakurtzen da [a←(9000)]
cpi 00          eta zerorekin konparatzen da
jz amaiera      berdinak badira, programa amaitzen da
jm negatiboa    <0 bada, "negatiboa"-ra jauzi egiten du
jmp buklea      bestela (>0), buklea exekutatzen du berriz
negatiboa:
;hemen jartzen ez ditugun agindu batzuk egongo dira
amaiera:
```

Programak 9000h posizioko datua konprobatzen du, hori positiboa den bitartean; negatiboa bada, (zehazten ez den) kodigo zati bat exekutatzen da, eta zero bada programa amaitzen da.

2-

```
mvi a,00            Metagailua zerora hasieratzan da,
buklea:
lda 9000            9000h-ko datua irakurtzen da metagailura,
mov b,a             b erregistroan gordetzen da,
lda 9001            9001h-ko datua irakurtzen da metagailura,
sub b               konparatzen dira (a←a-b) eta
jz amaiera          berdinak badira programa amaitzen da,
jmp ezberdinak      bestela "ezberdinak"-era jauzi egiten du
ezberdinak:
;hemen jartzen ez ditugun agindu batzuk egongo dira
amaiera:
```

Programak 9000h eta 9001h posizioetako balioak konparatzen ditu eta kodigo bat exekutatzen du (hemen zehaztugabea), horiek desberdinak izatekotan.

3- [Emaitza]:

```
hasieratu:
lxi hl,9000
mvi b,0a
buklea:
dcr m
jz amaiera
inx hl
dcr b
jnz buklea
jmp hasieratu
amaiera:
```

4- [Emaitza]:

```
lhld 9100  ; 9100 helbideko edukia L erregistrora
           ; eta 9101ekoa Hra
mvi b, 00  ; b erregistroa zerora hasieratu
```

```
mvi c, 10  ; c erregistroa 10 baliora hasieratu
mvi d, 08  ; d erregistroa 08 baliora
buklea:
mov a,d    ; (a)<-- (d)=08h
cmp m      ; (a)-[(HL)] konparatzen ditu:
           ; metatzailearen edukia (8) eta (1. elementuaren
           ; helbide 9100 posizioko edukia duen) arrayaren
           ; elementua, X datua
jm hurrengoa ; emaitza negatiboa bada (8 > X)
           ; "hurrengoa"-ra jauzi egin b gehitu
inr b      ;gabe, bestela (8 < X) b gehitu
hurrengoa:
inx hl     ; erakuslea gehitu
dcr c      ; c erregistroa dekrementatu
jz amaiera ; c-ren edukia zero bada amaierara jauzi egin
jmp buklea ; bestela begiztan beste buelta bat
amaiera:
```

9100h eta 9101h helbideetan dauden bi digitu hamaseitarrek osatzen duten helbidea array baten 1. elementuaren helbidea da. Array horrek 16 elementu dauzka eta, beraz, C erregistroan 10h balioa sartuko dugu begizta zeharkatzeko zenbatzaile moduan erabiltzeko. B erregistroa ere zenbatzaile bat izango da; kasu honetan, 08 baino balio txikiagoa duen arrayaren elementu kopurua zenbatzeko erabiliko dena.

8085aren bloke-diagrama

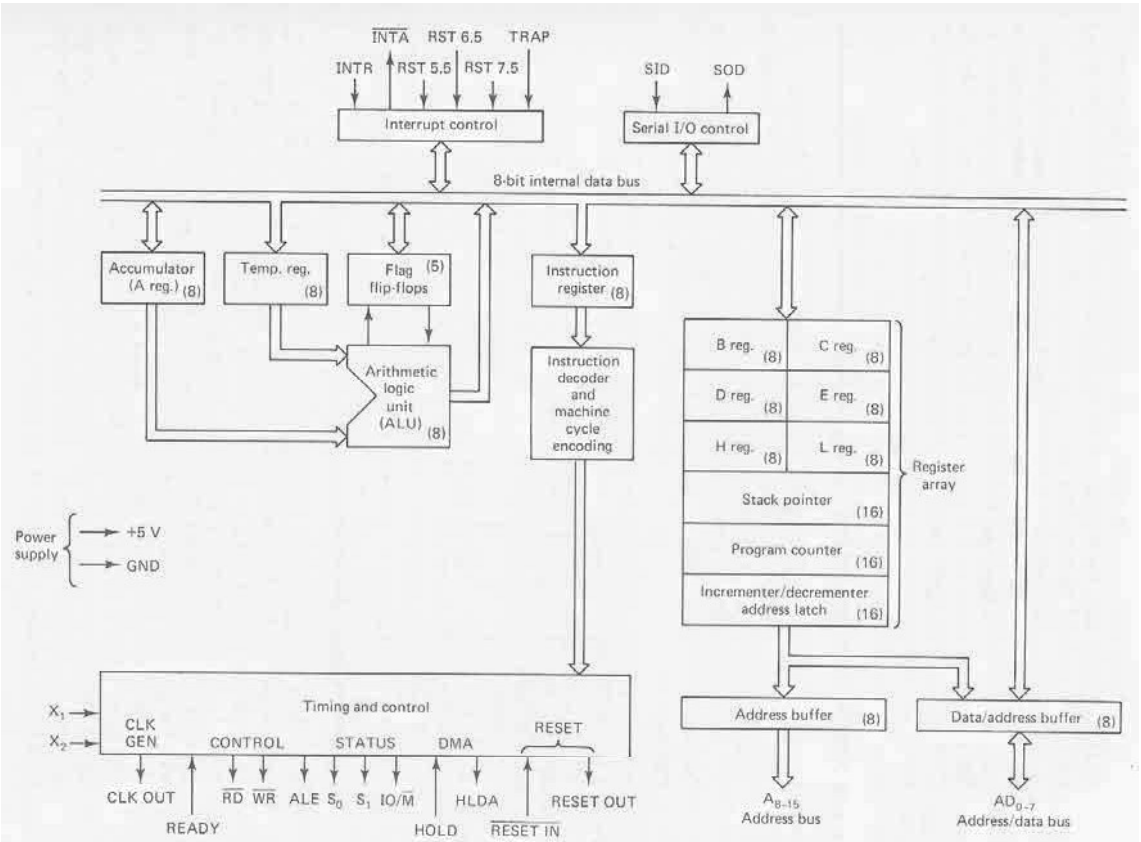
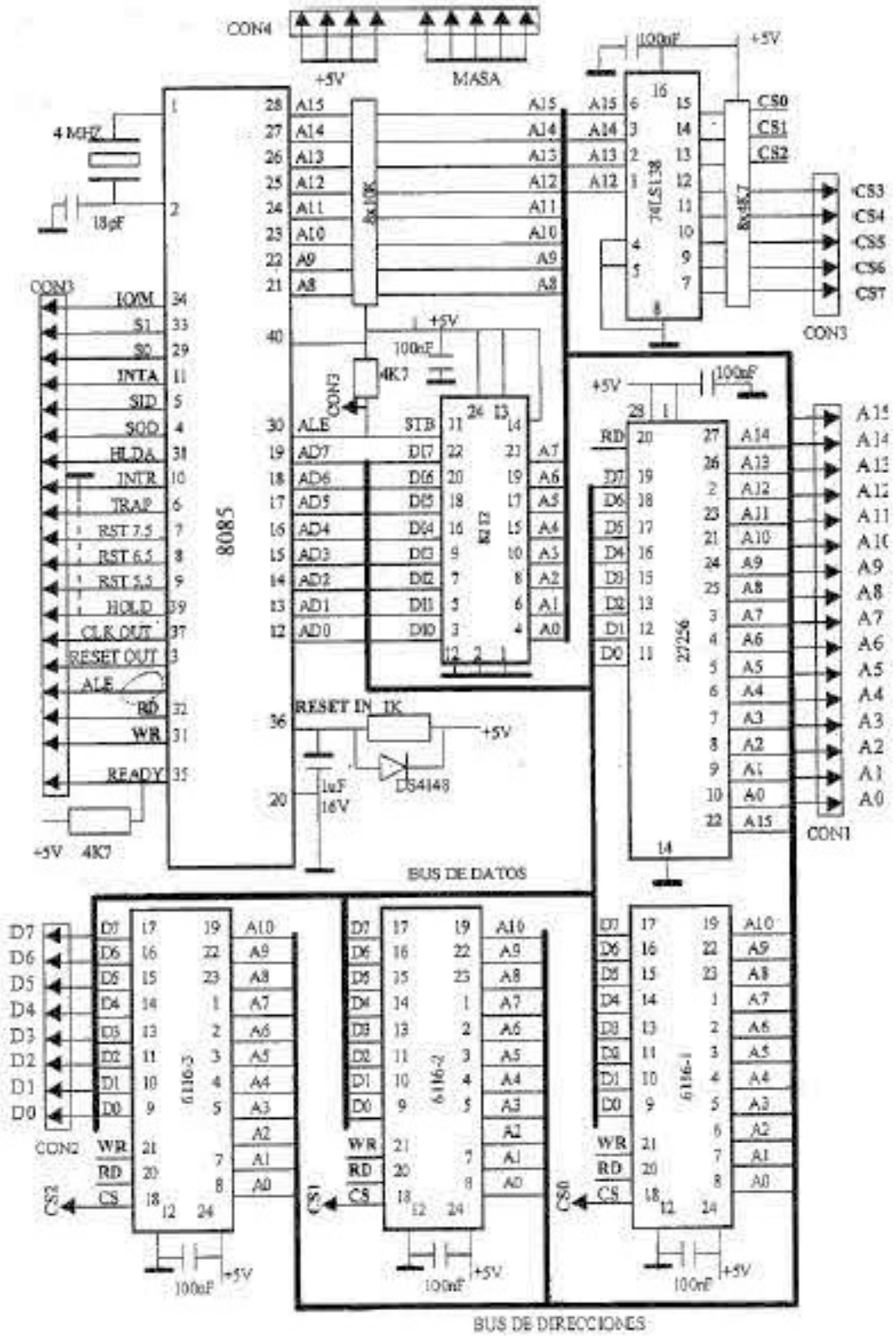


Figure 3–1 The functional block diagram of the Intel 8085A microprocessor. (Diagram courtesy of Intel Corporation.)

8085aren erregistroak eta kontrol-unitatea



8085aren agindu-jokoa

MNEMONIKOA	ADIERAZPENA	FLAGS
TRANSFERENTZIAKO AGINDUAK		
MOV r1,r2	(r1)←(r2)	BAT EZ
MOV r,M	(r)←[(HL)]	BAT EZ
MOV M,r	[(HL)] ←(r)	BAT EZ
MVI r,byte	(r)←byte	BAT EZ
MVI M,byte	[(HL)]← byte	BAT EZ
LXI rp,bikoitza	(rpl)←1° byte (rph)←2° byte	BAT EZ
LDA addr	(A)←[addr]	BAT EZ
STA addr	[addr]←(A)	BAT EZ
LHLD addr	(L)←[addr] (H)←[addr+1]	BAT EZ
SHLD addr	[addr]←(L) [addr+1]←(H)	BAT EZ
LDAX rp	(A)←[(rp)]	BAT EZ
STAX rp	[(rp)]←(A)	BAT EZ
XCHG	(H)↔(D) (L)↔(E)	BAT EZ
AGINDU ARITMETIKOAK		
ADD r	(A)←(A)+(r)	GUZTIAK
ADD M	(A)←(A)+[(HL)]	GUZTIAK
ADI byte	(A)←(A)+byte	GUZTIAK
ADC r	(A)←(A)+(r)+C Y	GUZTIAK
ADC M	(A)←(A)+[(HL)]+CY	GUZTIAK
ACI byte	(A)←(A)+byte+CY	GUZTIAK
SUB r	(A)←(A)-(r)	GUZTIAK
SUB M	(A)←(A)-[(HL)]	GUZTIAK
SUI byte	(A)←(A)-byte	GUZTIAK
SBB r	(A)←(A)-(r)-CY	GUZTIAK
SBB M	(A)←(A)-[(HL)]-CY	GUZTIAK
SBI byte	(A)←(A)-byte-CY	GUZTIAK
INR r	(r)←(r)+1	Z, S, P, AC
INR M	[(HL)]←[(HL)]+1	Z, S, P, AC
DCR r	(r)←(r)-1	Z, S, P, AC
DCR M	[(HL)]←[(HL)]-1	Z, S, P, AC
INX rp	(rp)←(rp)+1	BAT EZ
DCX rp	(rp)←(rp)-1	BAT EZ
DAD rp	(HL)←(HL)+(rp)	CY
DAA	(A) BCD egokitze	BAT EZ
AGINDU LOGIKOAK		
ANA r	(A)←(A) and (r) (CY)←0, (AC)←1	GUZTIAK
ANA M	(A)←(A) and [(HL)] (CY)←0, (AC)←1	GUZTIAK
ANI byte	(A)←(A) and byte (CY)←0, (AC)←1	GUZTIAK
XRA r	(A)←(A) xor (r)	GUZTIAK
XRA M	(A)←(A) xor [(HL)]	GUZTIAK
XRI byte	(A)←(A) xor byte	GUZTIAK
ORA r	(A)←(A) or (r)	GUZTIAK
ORA M	(A)←(A) or [(HL)]	GUZTIAK
ORI byte	(A)←(A) or byte	GUZTIAK
CMP r	(A)-(r)	GUZTIAK
CMP M	(A)-[(HL)]	GUZTIAK
CPI byte	(A)-byte	GUZTIAK

MNEMONIKOA	ADIERAZPENA	FLAGS
BIRAKETA ETA FLAG AGINDUAK		
RLC	Biraketa ezkerra	CY
RRC	Biraketa eskuinera	CY
RAL	Biraketa ezkerra CYren bidez	CY
RAR	Biraketa eskuinera Cyren bidez	CY
CMA	(A) 1eko osagarri	BAT EZ
CMC	Aldaraztua (CY)	CY
STC	(CY)←1	CY
ADARKATZE AGINDUAK		
ccc=NZ jauzi zero ez (Z=0), ccc=Z jauzi zero (Z=1), ccc=NC jauzi bururako ez (CY=0), ccc=C jauzi bururakoa (CY=1) , ccc=PO jauzi bakoitia (P=0), ccc=PE jauzi bikoitia (P=1), ccc=P jauzi positiboa (S=0), ccc=M jauzi negatiboa (S=1)		
JMP addr	(PC)←addr	BAT EZ
Jccc addr	ccc=1, (PC)←addr; ccc=0, (PC)←(PC)+3	BAT EZ
CALL addr	Gorde PC pilan (PC)←addr	BAT EZ
Cccc addr	ccc=1,gorde PC pilan (PC)←addr; ccc=0, (PC)←(PC)+3	BAT EZ
RET	PC pilatik berreskuratu	BAT EZ
Rccc	ccc=1, atera PC pilatik Si ccc=0 (PC)←(PC)+1	BAT EZ
RSTn	(PC)←n x 8	BAT EZ
PCHL	(PC)←(HL) + 1	BAT EZ
PILA MANEIATZEKO AGINDUAK		
PUSH rp	[(SP)-1]←(rpl) [(SP)-2]←(rph) (SP)←(SP)-2	BAT EZ
PUSH PSW	[(SP)-1]←(A) [(SP)-2]←(RE) (SP)←(SP)-2	BAT EZ
POP rp	(rph) ←[(SP)] (rpl)← [(SP)+1] (SP)←(SP)+2	BAT EZ
POP PSW	(RE) ←[(SP)] (A)← [(SP)+1] (SP)←(SP)+2	BAT EZ
XTHL	(L) ↔[(SP)] (H)↔ [(SP)+1]	BAT EZ
SPHL	(SP) ← (HL)	BAT EZ
SARRERA-IRTEERA AGINDUAK		
IN ataka	(A)←[ataka]	BAT EZ
OUT ataka	[ataka]←(A)	BAT EZ
ETENDURA-KONTROLEKO AGINDUAK		
EI	Etendurak gaitu	BAT EZ
DI	Etendurak desgaitu	BAT EZ
HLT	Gelditu mikroprozesadorea	BAT EZ
NOP	Ezer ez	BAT EZ
RIM	Irakurri serie lerroa eta etenduren egoera	BAT EZ
SIM	Idatiz serie lerroan eta etendurak programatu	BAT EZ

RE: Satatus Register
PSW: Processor Status Word

Inplementa ezazu gelako praktiketan diseinatutako UALa eta konproba ezazu horren funtzionamendua.

Diseina ezazu lau biteko A eta B zenbakiak (A[A3, A2, A1, A0] eta B[B3, B2, B1, B0]) sarrera moduan izanik, honako eragiketa hauek gauzatzen dituen UAL bat:

> *Aritmetikoak: A; Aren gehikuntza (A+1); batuketa (A+B); kenketa (A-B)*
> *Logikoak: B; AND (A·B); OR (A+B); XOR (A⊕B)*

Marraztu eragiketa aritmetikoetarako taula eta zirkuitua, eta eragiketa logikoetarako taula eta zirkuitua etapa baterako. Zenbat hautespen-seinale behar izan dituzu? Murriztu al daiteke kopuru hori?

Zirkuitu integratuak:

7400	quad 2-input NAND gates
7402	quad 2-input NOR gates
7404	hex inverting gates
7408	quad 2-input AND gates
7410	triple 3-input NAND gates
7411	triple 3-input AND gates
7427	triple 3-input NOR gates
7432	quad 2-input OR gates
7486	quad 2-input exclusive-OR gates
7483	4-bit binary adder with fast carry
7485	4-bit magnitude comparator
74138	3-line to 8-line decoder/demux
74139	2 fully independent 2-to-4-line decoders/demultiplexers
74153	dual 1-of-4 line data selectors/mux
74154	4-line to 16-line decoder/demux

OHARRA: zirkuitu integratuen ezaugarri-orrien kopia bat (datasheets) eskuragarri dago laborategian kontsultatzeko.

L3: Transferentzia aginduak

1. Hartu 9101h memoria posizioko datua eta gorde metagailuan.

2. Gorde metagailuan dagoen datua memoriako 9000h posizioan.

3. Kargatu memoriako 9000h posizioko datua L erregistroan eta 9001h posizioko datua H erregistroan.

4. Gorde L eta H erregistroen edukia 9000h eta 9001h memoria-posizioetan hurrenez hurren.

5. Kopiatu 9001h memoria-posizioko datua 9101h posizioan.

6. BC erregistroak erakusle moduan erabilita, gorde metagailuan horiek adierazitako memoria-helbidearen edukia.

7. Gorde metagailuaren edukia memorian, BC erregistro-bikoteak adierazitako posizioan.

8. Trukatu HL eta DE erregistro-bikoteen edukiak.

9. Gorde B erregistroaren edukia E erregistroan.

10. HL erregistro-bikotea memoriaren erakusle moduan erabiliz, gorde metagailuan horiek adierazitako helbideko edukia.

11. HL erregistro-bikotea memoriaren erakusle moduan erabiliz, gorde C erregistroaren edukia horiek adierazitako memoria-helbidean.

12. mov E,B; mov A,m eta mov m,C aginduak exekutatzen badira, zer gertatuko da F erregistroarekin? Eta PC erregistroarekin? (Arrazoitu erantzuna)

13. Kargatu metagailua 05h balioarekin.

14. Hasieratu HL erregistroak memoriaren 900A helbidearen erakusle moduan.

15. Gorde 05h datua memoriako 9001h posizioan, erakuslerik erabili gabe.

16. Gorde 05h datua memoriako 9001h posizioan, erakuslea erabilita.

17. Osatu taula kodigo zati hori mihiztatzean memorian 8000h helbidetik aurrera kargatzen diren balioekin. Pausoz pauso exekutatu eta PCaren edukiari erreparatuz erlazionatu aginduak eta memoriako balioak. Zer ondorioztatzen duzu?

mvi B, 06
mov A, B
lda 9000

Address	Data
8000	
8001	

1. Batu B erregistroaren edukia metagailuarenari, eta kendu emaitzari 7_{10} balioa (07h datua).

2. HL erregistro-bikoteak erakutsitako memoria-helbideko datua batu metagailuarenari, eta kendu emaitzari B erregistroaren edukia.

3. Batu 12_{10} balioa (0Ch balioa) metagailuaren edukiari eta kendu emaitzari C erregistroaren edukia.

4. Kendu metagailuaren edukiari HL erregistro-bikoteak erakutsitako memoria-helbidean dagoen datua eta gehitu emaitzari 10_{10} balioa (0Ah).

5. Dekrementatu B erregistroaren edukia eta gehitu unitate batean C erregistroarena.

6. Gehitu unitate batean HL erregistro-bikoteak erakutsitako memoria-posizioaren edukia.

7. Gutxitu unitate batean HL erregistro-bikoteak erakutsitako memoria posizioaren edukia.

8. Batu B erregistroaren edukia eta aurreko eragiketaren bururakoa metagailuari; ondoren, gehitu unitate batean B erregistroan dagoen datua.

9. Batu metagailuari 9_{10} balioa (09h) eta aurreko eragiketaren bururakoa; ondoren, gutxitu unitate batean C erregistroan dagoen datua.

10. Kendu metagailuari B erregistroaren edukia bai eta aurreko eragiketaren bururakoa ere; ondoren, gehitu unitate batean C erregistroan dagoen datua.

11. Kendu metagailuari 2_{10} balioa (02h) eta aurreko eragiketaren bururakoa; ondoren, gutxitu unitate batean C erregistroan dagoen datua.

12. Batu metagailuaren edukiari HL erregistro-bikoteak erakutsitako memoria-helbidean dagoen datua eta aurreko eragiketaren bururakoa.

13. Kendu metagailuaren edukiari HL erregistro-bikoteak erakutsitako memoria-helbidean dagoen datua eta aurreko eragiketaren bururakoa.

14. Gehitu unitate batean HL erregistro-bikotearen edukia (hurrengo memoria helbidea erakuts dezaten).

15. Gutxitu unitate batean HL erregistro-bikotearen edukia (aurreko memoria-helbidea erakuts dezaten).

1. Gauzatu metagailuaren eta B erregistroaren edukien arteko AND eragiketa logikoa, eta utzi emaitza B erregistroan.

2. Gauzatu metagailuaren eta 9001h memoria-helbidean dagoen datuaren arteko AND eragiketa logikoa.

3. Metagailuan dagoen datuaren pisurik handieneko bita (MSB) '1' edo '0' den jakin nahi dugu; horretarako, maskara bat aplikatuko diogu metagailuari, bit guztiak, hau izan ezik, '0'an jartzeko. Zein da egin behar dugun eragiketa logikoa (balioa zehaztuz)?

4. Gauzatu metagailuaren eta C erregistroaren edukien arteko OR eragiketa logikoa, eta utzi emaitza C erregistroan.

5. Gauzatu metagailuaren eta 900Ah memoria-helbidean dagoen datuaren arteko OR eragiketa logikoa.

6. Metagailuan dagoen datuaren 0,1 eta 5 pisuko bitak '1'ean jarri nahi ditugu, besteak eraldatu gabe; horretarako, maskara bat aplikatuko diogu metagailuari. Zein eragiketa logiko egin behar dugu (balioa zehaztuz)?

7. Gauzatu metagailuaren eta D erregistroaren edukien arteko XOR eragiketa logikoa, eta utzi emaitza D erregistroan.

8. Gauzatu metagailuaren eta 9000h memoria helbidean dagoen datuaren arteko XOR eragiketa logikoa.

9. Metagailuan dagoen datuaren ukatua lortu nahi dugu; horretarako, maskara bat aplikatuko diogu metagailuari. Zein eragiketa logiko gauzatu behar dugu (balioa zehaztuz)?

10. Metagailuaren eta B erregistroaren edukiak konparatu nahi ditugu, baina hauek eraldatu gabe. Zein eragiketa logiko erabiliko dugu? Non gordetzen da emaitza? Nola jakingo dugu A<B, A>B edo A=B den?

11. Eta metagailuaren edukia eta 9000 memoria-helbidean dagoen datua konparatu nahi baditugu?

12. Eta metagailuaren edukia 10 balio hamartarra (0Ah) baino txikiagoa den jakin nahi badugu?

1. Egin 88h balio hamaseitarra zati bi biraketa-agindu batekin eta utzi emaitza B erregistroan.

2. Egin 11h balio hamaseitarra bider bi biraketa-agindu batez baliatuz, eta utzi emaitza metagailuan.

3. Zein da beheko a) eta b) programen emaitza? Arrazoitu erantzuna (metagailuaren edukia irudikatu pasuz-pausu).

```
a) mvi a, 81          b) mvi a, 81
   ral                   ral
   ral                   rlc
```

4. Zein da beheko a) eta b) programen emaitza? Arrazoitu erantzuna (metagailuaren edukia irudikatu pasuz-pausu).

```
a) mvi a, 81          b) mvi a, 81
   rlc                   rlc
   rlc                   ral
```

5. Egin B erregistroan dagoen datuaren 1erako osagarria (emaitza Bn utzi).

6. Gauzatu C erregistroan dagoen datuaren 2rako osagarria (emaitza Cn utzi).

1. Gutxitu unitate batean metagailuaren balioa, eragiketa honen emaitza zero bada, egin jauzi «zero» labelera eta gorde B erregistroan 0_{10} balioa; emaitza ez bada zero, ordea, egin jauzi «ez-zero» labelera eta kopiatu B erregistroan metagailuaren balioa. *Kontuz! Besterik adierazi ezean, exekuzio sekuentziala izango da, baina eragiketa bakarra burutu behar du!*

2. Batu A eta B erregistroen edukia, emaitzak bururakoa eman badu, egin jauzi «bururakoa» labelera, eta gehitu Aren edukia D erregistroari bururakoarekin ; ez badu bururakorik eman, ordea, egin jauzi «ez-bururakorik» labelera eta burutu (A)+(D) eragiketa bururakorik gabe.

3. A eta B erregistroen kenketa egin eta emaitzaren arabera erregistro bat gehitzen duen programa idatzi: positiboa (edo zero) bada C erregistroa, eta negatiboa bada D erregistroa.

4. Paritate bikoitiarekin bidalitako hamar datu batzuk 9000h memoria posiziotik aurrera biltegiratu dira; 9000h posizioko datutik hasita paritatea konprobatzen duen programa idatzi, errorerik egon bada, «errorea» azpierrutina exekutatu behar dugu posizio horretako bit guztiak «0»an jarriz; paritatea zuzena bada exekuzio normala jarraitu (konprobatu hurrengo posizioko datua).

11. 9000h memoria-helbideko datuak «1» kopuru bakoitia badu, sartu 01h balioa B erregistroan. Sartu 02h balioa kontrako kasuan.

12. Kopiatu metagailuan B erregistroaren edukia, egin azpierrutina bateri deia, bertan dekrementatu metagailuko balioa unitate batean eta, emaitza zero bada, itzuli azpierrutinatik.

13. Zein desberdintasun dago beheko bi programen artean?

```
jmp salto1              call salto1
mvi a,01                mvi a,01
salto1:                 jmp fin
mvi b,01                salto1:
                        mvi b,01
                        ret
                        fin:
```

14. Zein desberdintasun dago beheko bi programen artean?

```
lda 9000                lda 9000
adi 00                  adi 00
jp positiboa            cp positiboa
mvi b,01                mvi b,01
positiboa:              jmp fin
mvi d,01                positiboa:
                        mvi d,01
                        ret
                        fin:
```

Beheko programak bi zenbaki jasotzen ditu, bata 1. atakatik eta bestea 2.etik; C eta D erregistroetan kopiatzen ditu eta «batuBCD» azpierrutinari deitzen dio. Zer egiten du azpierrutina horrek?

Idatzi programa eta exekuta ezazu pausoz pauso funtzionamendua konprobatzeko. Erreparatu PC eta SP erakusleek hartzen dituzten balioei. Osatu beheko taula, agindu bakoitza identifikatuz (kontuan izan guztiek ez dutela berdin okupatzen):

mnemonikoa	**helbidea**	**agindua hex**
in 01		
mov c,a		
in 02		
mov d,a		
mvi b,00		
call sumarBCD		
mov a,b		
out 03		
jmp fin		
nop		
nop		
nop		
nop		
nop		
sumarBCD:		
mov a,c		
add d		
mov b,a		
ret		
fin:		
mov e,a		

Kudeaketaren eta Informazio Sistemen Informatikaren Ingeniaritzako Gradua
Ordenagailuen Egitura 2014/2015
L9: programa osoak

Idatzi eta konprobatu honako programa hauetarako kodigoak.

1- Irakurri memoriako 9000h posizioko edukia eta, 07h baino txikiagoa bada, gorde bere balioa 9010h posizioan.

2- Irakurri memoriako 9000h posizioko edukia eta, 07h baino handiagoa edo berdina bada, gorde 0Fh 9010h posizioan.

3- Memoriako 9000h posizioan array baten lehen elementua dago. Horren luzera 10 elementukoa dela jakinik, idatzi 9100h posizioan arrayaren elementu guztien batura gordeko duen programa.

4- Gelako praktiketan, 2 zenbaki biderkatzeko eta baten faktoriala kalkulatzeko fluxu-diagramak egin ziren (GP3). Idatzi eta konprobatu 8085ean faktoriala kalkulatzeko kodigoa.

5- Idatzi 9001h posizioko balioa etengabe irakurtzen duen programa eta, balioa 03h bada, eragiketarik gabeko bi atzerapen sartzen dituena ondoren, 3. atakan 01h balioa jartzeko.

6- Sentsore batek «1»ean jartzen ditu 9001h posizioko byte-aren bit guztiak, ura maila jakin batera iristen denean. Hori gertatzen denean, gure programak zerbitzu-azpierrutina deitu behar du, bertan metagailuaren balioa pilan gordeko du eta 9001h posizioko bitak berriro «0»an jarriko ditu, ondoren programa deitzailera itzuliz.

7- Memoriako 9000h posizioan array baten lehen elementua dago. Horren luzera 10 elementukoa dela jakinik, idatzi arrayaren elementuak txikienetik handienera ordenatuko dituen programa.

8- Memoriako 9000h posizioan array baten lehen elementua dago. Elementu guztiak balio positiboak direla jakinik, bilatu arraya osatzen duten 10 elementuetatik txikiena eta gorde bere posizio erlatiboa (arrayaren barruan duena) arrayaren azken elementuaren hurrengo memoria-posizioan.