Gediminas GRAŽULEVIČIUS

MIKROPROCESORIAI

Projekto kodas

VP1-2.2-ŠMM-07-K-01-047

VGTU Elektronikos fakulteto
I pakopos studijų programų
esminis atnaujinimas

Vilnius „Technika“ 2012

VILNIAUS GEDIMINO TECHNIKOS UNIVERSITETAS

Gediminas GRAŽULEVIČIUS

MIKROPROCESORIAI

Praktinės užduotys ir metodikos nurodymai

G. Gražulevičius. Mikroprocesoriai: praktinės užduotys ir metodiniai nurodymai. Vilnius: Technika, 2012. 158 p. [5,25 aut. l. 2012 06 04]

Leidinyje pateikta daugiau kaip du šimtai į problemą orientuotų praktinių 8 ir 16 skilčių bendrosios paskirties mikroprocesorių programavimo užduočių. Pateiktos daugelio užduočių sprendimo metodikos bei pavyzdžiai, taip pat užduotys savarankiskam darbui. Nagrinėjama 8 ir 16 skilčių bendrosios paskirties mikroprocesorių komandų sistema, jų programavimas mašininiais kodais ir assemblerio programavimo kalba.

Šis technologijos mokslų srities, elektros ir elektronikos inžinerijos mokslo krypties leidinys skirtas VGTU Elektronikos fakulteto pirmosios pakopos studijų studentams, studijuojantiems automatikos, elektronikos inžinerijos, informacinių sistemų inžinerijos ir kompiuterių inžinerijos studijų programose mikroprocesorių, mikroprocesorių ir jų programavimo dalykus. Leidinys taip pat gali būti naudingas ir kitų aukštųjų mokyklų bei universitetų studentams.

Recenzavo:

doc. dr. Nerijus Paulauskas, VGTU Kompiuterių inžinerijos katedra,
doc. dr. Edvardas Matkevičius, VGTU Automatikos katedra

Leidinyje parengtas vykdamas projektą „VGTU Elektronikos fakulteto I pakopos studijų programų esminis atnaujinimas“. Leidinio rengimą ir leidybą finansavo Vilniaus Gedimino technikos universitetas ir Europos socialinis fondas (sutartis Nr. VP1-2.2-ŠMM-07-K-01-047).

VGTU leidyklos TECHNIKA 1328-S mokomosios metodinės literatūros knyga
<http://leidykla.vgtu.lt>

Redaktorė *Kristina Noreikienė*
Maketuotoja *Rasa Labutienė*

eISBN 978-609-457-146-6
doi:10.3846/1328-S

© Gediminas Gražulevičius, 2012
© Vilniaus Gedimino technikos universitetas, 2012

TURINYS

ĮVADAS	4
1. AŠTUONIŲ SKILČIŲ BENDROSIOS PASKIRTIES MIKROPROCESORIŲ KOMANDŲ SISTEMA IR PROGRAMAVIMAS	6
1.1. Mikroprocesoriaus Intel® 8080 funkcionavimas ir ciklų organizavimo principas	6
1.2. Komandų formatai ir operandų adresavimo būdai	19
1.3. Mikroprocesoriaus Intel® 8080 komandų sistema ir programavimas	50
2. ŠEŠIOLIKOS SKILČIŲ BENDROSIOS PASKIRTIES MIKROPROCESORIŲ KOMANDŲ SISTEMA IR PROGRAMAVIMAS	87
2.1. Mikroprocesoriaus Intel® 8086 atminties organizacija	87
2.2. Komandų formatai ir operandų adresavimo būdai	91
2.3. Mikroprocesoriaus Intel® 8086 komandų sistema ir programavimas	121
ATSAKYMAI	130
LITERATŪRA	131
1 PRIEDAS	
Mikroprocesoriaus Intel® 8080 komandų sistema	132
Mikroprocesoriaus Intel® 8080 komandų grupės ir šešioliktainiai kodai	148
2 PRIEDAS	
Mikroprocesoriaus Intel® 8086 pagrindinės komandos	152

ĮVADAS

Projektuojant šiuolaikinę mikroprocesorinę sistemą, daugiausia laiko ir darbo sąnaudų reikalauja programinės įrangos kūrimas ir derinimas. Šiuo metu programos, skirtos bendrosios paskirties mikroprocesoriams, dažniausiai rašomos vadinamosiomis aukšto lygio programavimo kalbomis, tokiomis kaip Basic, Pascal, Fortran, C, C++ ir kt. Tačiau, jei keliamas uždavinys, kad parašyta programa būtų kompaktiška ir maksimaliai efektyviai vykdoma, be assemblerio kalbos neapsieisime. Programuojant assembleriu, geriausiai išnaudojamos šiuolaikinio mikroprocesoriaus galimybės. Assembleris yra žemo lygio programavimo kalba, skirta tam tikros architektūros mikroprocesorių mašininėms komandoms žymėti. Aukšto ir žemo lygio programavimo kalbas galima palyginti su automobilio pavarų dėžėmis. Aukšto lygio programavimo kalba atitiktų automatinę pavarų dėžę, nes ji yra patogi, tačiau mažiau efektyvi ir ekonomiška. Assembleris atitiktų mechaninę pavarų dėžę, nes ji lanksti, efektyvi, tačiau reikalauja daugiau vartotojo pastangų. Taigi assembleriu programuoti yra žymiai sudėtingiau, nes reikia žinoti mikroprocesoriaus architektūrą, t. y. registrų kiekį ir darbo su jais būdus, duomenų ir komandų formatą, atminties organizaciją, operandų adresavimo būdus, pertraukčių aptarnavimo būdus, komandų sistemą ir kt.

Dauguma projektuojamų mikroprocesorių sistemų gali tik saugoti parašytas taikomąsias programas. Programinei įrangai projektuoti galima

taikyti mikrokompiuterius arba mokomąsias mikroprocesorines sistemas, o jų neturint – specialiąsias programas, vadinamas emuliatoriais.

Leidinyje pateikta daugiau kaip du šimtai į problemą orientuotų praktinių 8 ir 16 skilčių bendrosios paskirties mikroprocesorių programavimo užduočių. Pateiktos daugelio užduočių sprendimo metodikos bei pavyzdžiai, taip pat užduotys savarankiškam darbui.

Leidinio struktūra ir turinys atitinka dalykų „Mikroprocesoriai“ ir „Mikroprocesoriai ir jų programavimas“ programą. Leidinį sudaro du skyriai: pirmajame pateikiamos 8 skilčių bendrosios paskirties mikroprocesorių programavimo praktinės užduotys ir metodikos nurodymai, antrajame – 16 skilčių bendrosios paskirties mikroprocesorių programavimo praktinės užduotys ir metodikos nurodymai.

1. AŠTUONIŲ SKILČIŲ BENDROSIOS PASKIRTIES MIKROPROCESORIŲ KOMANDŲ SISTEMA IR PROGRAMAVIMAS

1.1. Mikroprocesoriaus Intel® 8080 funkcionavimas ir ciklų organizavimo principas

1. Atliekant operaciją, buvo suformuoti tokie požymių registro **F** turiniai:

<i>a.</i> 03_{16} ;	<i>f.</i> 57_{16} ;
<i>b.</i> 02_{16} ;	<i>g.</i> 87_{16} ;
<i>c.</i> 06_{16} ;	<i>h.</i> 83_{16} ;
<i>d.</i> 07_{16} ;	<i>i.</i> 97_{16} ;
<i>e.</i> 12_{16} ;	<i>j.</i> 46_{16} .

Nustatykite ir užrašykite požymių registro **F** skilčių reikšmes ir pateikite operacijos, kuri suformuotą nustatytus požymius, pavyzdį.

1. a. Sprendimas.

Prisiminkime, kas yra požymių registras. Mikroprocesoriuje Intel® 8080 požymių registras žymimas raide **F**. Tai 8 skilčių registras, skirtas požymiams, suformuotiems atliekant operacijas, saugoti. Požymis fiksuojamas į atitinkamą požymių registro **F** skiltį įrašant vienetą arba nulį. Pateiksime požymių registro **F** skilčių reikšmes:

7	6	5	4	3	2	1	0
S	Z	0	AC	0	P	1	CY

- Nulinė – žemiausioji skiltis **CY** (angl. *Carry*) – pernašos: **CY** = 1, jei operacijos rezultatas netelpa į 8 skiltis (įvyksta pernaša iš aukščiausios skilties arba buvo skolintasi atliekant atimties veiksmą), **CY** = 0 – pernašos nėra;
- Antroji skiltis **P** (angl. *Parity*) – lyginumo: **P** = 1, jei rezultato dvejetainiame kode vienetų skaičius yra lyginis, **P** = 0 – jei nelyginis;
- Ketvirtoji skiltis **AC** (angl. *Auxiliary Carry*) – pagalbinės pernašos: **AC** = 1, kai įvyksta pernaša iš dvejetainio skaičiaus trečiosios skilties, **AC** = 0 – pernašos nėra;
- Šeštoji skiltis **Z** (angl. *Zero*) – nulio arba nulinio rezultato: **Z** = 1, jei operacijos rezultatas lygus nuliui, **Z** = 0 – jei nelygus nuliui;
- Septintoji – aukščiausioji skiltis **S** (angl. *Sign*) – ženkle: **S** = 1, jei operacijos rezultatas yra neigiamas, **S** = 0 – jei teigiamas.

Pirmoji, trečioji ir penktoji požymių registro **F** skiltys nekinta, nes jos netaikomos.

Pirmiausia duotą požymių registro **F** turinį 03_{16} reikia perversi iš šešiolyktainės į dvejetainę skaičiavimo sistemą. Pereidami iš šešiolyktainės skaičiavimo sistemos į dvejetainę, kiekvieną šešiolyktainį simbolį išreiškiamo keturių skilčių dvejetainiais skaičiais (tetradomis):

$$\begin{array}{ccccc}
 03_{16} = & 0 & & 3 & \\
 & \downarrow \downarrow & & \downarrow \downarrow & \\
 & \boxed{0000} & & \boxed{0011} & = 00000011_2
 \end{array}$$

Taigi matome, kad, atlikus operaciją, požymių registro **F** turinys yra lygus 00000011_2 , vadinasi, buvo suformuotas pernašos požymis, t. y. į pernašos skiltį **CY** įrašytas vienetasis. Tai reiškia, kad operacijos rezultatas netilpo į 8 skiltis atliekant sudėties operaciją arba buvo skolintasi atliekant atimties operaciją (atimant didesnę skaičių iš mažesnio). Reikia pažymėti, kad pernašos skiltis **CY** atimant vadinama paskolos skiltimi. Pateiksime sudėties su pernaša pavyzdį:

$$\begin{array}{r}
 1010\ 1100_2 = 172_{10} \\
 + \\
 0111\ 0000_2 = 112_{10} \\
 \hline
 1\ 0001\ 1100_2 = 284_{10}
 \end{array}$$

CY →

Mikroprocesoriaus Intel® 8080 8 skilčių aritmetinis-loginis įtaisas gali atlikti keturias aritmetines operacijas (sudėtį ir atimtį, su pernaša ir be jos), keturias logines ir keturias poslinkio operacijas. Atliekant aritmetines ir logines operacijas, vienas iš operandų turi būti kaupiklyje. Įvykdytos operacijos rezultatas lieka kaupiklyje.

Kaip matome, sudėties rezultatas netilpo į 8 skilčių kaupiklį. Tai rodo vienetą požymių registro **F** pernašos skiltyje **CY**. Gautas rezultatas yra neteisingas, kadangi su 8 skilčių mikroprocesoriumi galima operuoti teigiamais sveikaisiais skaičiais nuo 0 iki 255 arba sveikaisiais skaičiais nuo $-128 (-2^7)$ iki $+127 (2^7 - 1)$.

2. Duotas komandos dvejetainis kodas. Nustatykite ir užrašykite duotos komandos šešioliktainį ir mnemoninį kodą. Paaiškinkite, kokią operaciją komanda atlieka.

a. 00110110	m. 00101010	u. 11100110
00110000;	00001010	00000001;
b. 00001111;	00100001;	v. 11101110
c. 10111001;	n. 00100010	11111111;

d. 01000111;	00000001	w. 11010001;
e. 00101001;	00100010;	x. 00110010
f. 00000111;	o. 00111010	00100000
g. 00100101;	11111110	00000000;
h. 00000010;	00001111;	y. 00111011;
i. 00110001	p. 00011111;	z. 11001101
10101011	q. 00010111;	11001100
00101111;	r. 00100111;	00001010.
k. 00010011;	s. 11110101;	
l. 00011010;	t. 10110001;	

2. a. Sprendimas.

Pirmiausia operandus iš dvejetainės skaičiavimo sistemos pervesime į šešioliktainę skaičiavimo sistemą. Tai atliekame taip: dvejetainį skaičių, pradėdami nuo žemiausiosios skilties, suskaidome į skilčių ketvertukus – tetradas. Kiekvieną tetrada išreikštą dvejetainį skaičių užrašome šešioliktainiais simboliais:

$$00110110_2 = \boxed{0011} \boxed{0110}$$

3

6

= 36₁₆

Kadangi duota komanda yra dviejų baitų, antrąjį baitą pervedami tokiu pat būdu. Gauname komandos šešioliktainį kodą 36₁₆ ir 30₁₆. Pirmasis baitas komandoje visada yra operacijos kodo baitas, pagal kurį iš 1 priede pateiktos lentelės randame komandos mnemoninį kodą. Taigi duota komanda yra **MVI M, 30**, kuri persiunčia 8 bitų tiesioginį operandą (antrąjį komandos baitą) į atminties ląstelę, kurios adresą nurodo registrų pora **H** (**H** ir **L** registrai).

3. Pradiniai mikroprocesoriaus registrai **A**, **B** ir požymių registro **F** turiniai yra tokie: $A = 65_{16}$, $B = 0E_{16}$, $F = 03_{16}$. Nustatykite ir užrašykite šių registrų turinius įvykdę duotą komandą.

a. ADD B;	j. ORA A;	s. STA 2F00;
b. SUB B;	k. XRA B;	t. INR B;
c. ADC B;	l. XRA A;	u. CALL 2100;
d. SBB B;	m. RRC;	v. RAL;
e. ADC A;	n. RLC;	w. DCX B;
f. SBB A;	o. JMP 0A11;	x. NOP;
g. ANA B	p. RAR;	y. CMP B;
h. ANAA;	q. CPI 63;	z. CMA.
i. ORA B;	r. SUI FA;	

3. a. *Sprendimas.*

Pirmiausia operandus iš šešioliktainės skaičiavimo sistemos pervesime į dvejetainę skaičiavimo sistemą:

$$65_{16} = \begin{array}{cc} 6 & 5 \\ \downarrow & \downarrow \\ \boxed{0110} & \boxed{0101} \end{array} = 01100101_2$$

Atlikę tokią pačią procedūrą ir su kitais operandais, gauname, kad $A = 65_{16} = 01100101_2$, $B = 0E_{16} = 00001110_2$, $F = 03_{16} = 00000011_2$.

Komanda **ADD B** prideda registro **B** turinį prie kaupiklio **A** turinio. Rezultatas įrašomas į kaupiklį **A**. Ši komanda yra aritmetinė, todėl keičia požymių registro **F** skilčių reikšmes (1 priedas).

Kaip matome, įvykdžius komandą **ADD B** su duotais operandais, sudėties metu įvyko pernaša iš trečiosios skilties:

$$\begin{array}{r}
 0110\ 0101_2 = 65_{16} \\
 + \quad 0000\ 1110_2 = 0E_{16} \\
 \hline
 0111\ 0011_2 = 73_{16} \\
 \text{AC} \rightarrow
 \end{array}$$

Todėl požymių registro **F** pagalbinės pernašos skiltyje **AC** buvo įrašytas vienetasis. Taigi, įvykdę duotą komandą, gavome: **A** = 01110011₂ = 73₁₆, **B** = 00001110₂ = 0E₁₆, **F** = 00010010₂ = 12₁₆.

4. Duotas programos, kuri atmintyje prasideda adresu 2000₁₆, šešioliktainis kodas.

2000	01	202C	<i>x</i>
2001	30	202D	<i>y</i>
2002	20	202F	<i>z</i>
2003	21	2030	<i>w</i>
2004	2C		
2005	20		

2006	7E
2007	2C
2008	86
2009	2E
200A	2F
200B	96
200C	02
200D	76

Nustatykite ir užrašykite mikroprocesoriaus vykdomų komandų mnemoninius kodus (1 priedas). Apskaičiuokite, koks bus rezultatas w , jei $x = 05_{16}$, $y = 07_{16}$, $z = 01_{16}$.

5. Nubraižykite ir paaiškinkite duotos komandos laiko diagramą. Apskaičiuokite komandos vykdymo trukmę, jei mikroprocesoriaus taktinis dažnis $f_i = 2$ MHz.

a. MOV A, M;	j. STAX B;	s. LHLD 2005;
b. MVI C, 0A;	k. LDAX H;	t. SHLD 20FF;
c. MVI M, 02;	l. ADD D;	u. JMP 0A00;
d. LXI H, 2233;	m. OUT 08;	v. JNZ 2511;
e. STA 2160;	n. DAD D;	w. CALL 0701;
f. DCR B;	o. INX SP;	x. DAA;
g. IN FA	p. DCR M;	y. PUSH B;
h. LDA 200C;	q. XTHL;	z. POP PSW.
i. ORA M;	r. SPHL;	

5. e. *Sprendimas.*

Mikroprocesorius gali funkcionuoti tik tada, kai jis įjungtas į sistemą, turinčią taktinių impulsų generatorių, operatyviają ir pastoviąją atmintis, išorines duomenų, adresų ir valdymo magistrales, maitinimo šaltinį ir kt.

Taktinių impulsų generatorius sinchronizuoja mikroprocesoriaus darbą. Nagrinėjamame mikroprocesoriuje taikomos dvi periodinės identiškos stačiakampių taktinių impulsų sekos C1 ir C2, perstumtos viena kitos atžvilgiu per 0,5 periodo (1.1 pav.).

Mikroprocesorinė sistema funkcionuoja sinchroniškai su taktiniais impulsais. Mikroprocesoriaus būsena išlieka nepakitusi per taktinį intervalą (taktą). Tam tikras taktinių intervalų skaičius (3–5) sudaro vadinamąjį procesoriaus ciklą. Jų gali būti nuo 1 iki 5: **M1**, **M2**, **M3**, **M4**, **M5**. Kiekviename cikle mikroprocesorius atlieka vieną kreipimąsi į atmintį arba į įvesties ir išvesties įtaisą. Vieno procesoriaus ciklo metu nuskaitomas komandos baitas, duomenų baitas arba vienas iš adreso baitų.

1.1 pav. Stačiakampių taktinių impulsų sekos

Komandos ciklu vadinamas laiko intervalas, kurio reikia komandai nuskaityti iš atminties ir jai įvykdyti (1.2 pav.).

1.2 pav. Komandos ciklas

Komandos ciklą sudaro nuo 1 iki 5 procesoriaus ciklų. Konkretus jų skaičius priklauso nuo operacijos sudėtingumo ir lygus kreipčių į atmintį arba į įvesties ir išvesties įtaisą skaičiui.

Pirmieji trys taktai visuose procesoriaus cikluose taikomi duomenims keistis su atmintimi ir įvesties bei išvesties įtaisais. Taktai **T4** ir **T5** (jeigu jis yra) skirti vidinėms mikroprocesoriaus operacijoms atlikti.

Pirmuoju taktu **T1** komandų skaitiklio **PC** turinys perduodamas į adresų magistralę. Adresą priima atmintis. Pradedama komandos baitą skaityti iš nurodytos atminties ląstelės.

Antrajame takte **T2** tikrinama, ar parengti duomenys. Apie tai informuoja loginis vienetas mikroprocesoriaus įėjime **READY**.

Jeigu šiame įėjime yra loginis nulis, nustatoma mikroprocesoriaus laukimo būseną, kurioje kiekvienas kitas taktas laikomas kaip taktas **T2**. Tai trunka tol, kol įėjime **READY** nepasirodo loginis vienetas. Pasirodžius šiam signalui, mikroprocesorius išeina iš laukimo būsenos ir prasideda trečiasis taktas **T3**. Šiame takte komandos baitas iš duomenų magistralės įrašomas į mikroprocesoriaus komandų registrą.

Ketvirtajame takte **T4** iššifruojamas priimtas komandos baitas ir nustatoma, ar nereikia papildomai kreiptis į atmintį. Jeigu nereikia (vieno baito komanda ir operandai yra mikroprocesoriaus registruose), tai šiame takte, arba pritaikius papildomą taktą, **T5** atliekama komandoje užkoduota operacija. Taigi minimali operacijos trukmė – keturi taktai.

Jeigu reikia papildomai kreiptis į operatyviąją atmintį, po takto **T4** ciklas **M1** užbaigiamas ir pradedamas vykdyti ciklas **M2**.

Komanda **STA 2160** persiunčia kaupiklio **A** turinį į atminties ląstelę, kurios adresas nurodytas antrajame ir trečiajame komandos baituose. Ši komanda yra trijų baitų ilgio ir, tarkime, ji yra įrašyta į atmintį pradedant adresu 2000₁₆:

2000	3A – operacijos kodo baitas (OKB),
2001	60 – žemesnysis adreso baitas (ADRL),
2002	21 – aukštesnysis adreso baitas (ADRH).

Vadinasi, komandai iš atminties nuskaityti reikės trijų procesoriaus ciklų ir vieno ciklo duomenims iš kaupiklio į atminties ląstelę įrašyti:

Kaip matome, komanda **STA 2160** vykdoma 13 taktinių intervalų. Kadangi mikroprocesoriaus taktinis dažnis $f_t = 2 \text{ MHz}$, tai vieno taktinio intervalo trukmė $t_t = 1 / f_t = 1 / 2000000 = 0,0000005 \text{ s} = 0,5 \text{ } \mu\text{s}$. Apskaičiuojame komandos vykdymo trukmę: $t = N \times t_t = 13 \times 0,5 = 6,5 \text{ } \mu\text{s}$.

6. Pateikite po vieną sąlyginės kreipties į paprogramį ir sąlyginės grįžties iš paprogramio komandos pavyzdį. Nubraižykite ir paaiškinkite pateiktos komandos laiko diagramą, kai:

- Kreipties ir grįžties sąlyga yra įvykdyta;
- Kreipties ir grįžties sąlyga yra neįvykdyta.

7. Išnagrinėkite duotą programos fragmentą ir nustatykite, koks bus kaupiklio **A** turinys, įvykdžius duotą programos fragmentą, bei kokia bus programos vykdymo trukmė. Pradiniai duomenys prieš operaciją yra šie: **A** = 00_{16} , **C** = $A2_{16}$, **D** = 26_{16} , $2112_{16} = F2_{16}$, **H** = BC_{16} , **L** = EC_{16} , mikroprocesoriaus taktinis dažnis $f_t = 2 \text{ MHz}$.

a. 2005 MVIA, 0E	d. 201F MVIA, AA
2007 ADD L	2021 ANA D
2008 JNZ 200C	2022 ORA C
200B MOV A, C	2023 ANI 82
200C INR A	2025 RRC
200D MOV C, A	

<i>b.</i>	2013	MOV A, C	<i>e.</i>	21F1	MVI M, 01
	2014	ADI 00		21F3	LDA 2112
	2016	JZ 201C		21F6	MOV B, M
	2019	LDA 2112		21F7	XRA B
	201C	DCR A		21F8	RLC
<i>c.</i>	2101	JMP 2109	<i>f.</i>	2022	MOV M, A
	2104	MOV A, C		2023	INR M
	2105	LDA 2112		2024	INR L
	2108	SUB A		2025	MVI M, 10
	2109	HLT		2027	MOV B, M
				2028	LDA BCED
				202B	ORA B

7. a. Sprendimas.

Sprendimą pateiksime lentelės pavidalu. Komandų vykdymo trukmę taktais randame 1 priede pateiktame mikroprocesoriaus Intel® 8080 komandų sistemos aprašyme.

Adresas ₁₆	Komandos mnemonika	Atliekama operacija	Operacijos rezultatas	Taktų skaičius
2005	MVI A, 0E	Vieno baido tiesioginis operandas (antrasis komandos baitas) persiunčiamas į kaupiklį A ($0E_{16} \rightarrow A$).	$A = 0E_{16}$	7
2007	ADD L	Registro L turinys pridodamas prie kaupiklio A turinio ($A + L \rightarrow A$).	$A = FA_{16}$	4
2008	JNZ 200C	Jei gauta suma nėra lygi nuliui ($A + L \neq 0, Z = 0$), tai pereinama ir pradedama vykdyti komanda, kurios adresas yra $200C_{16}$ (antrasis ir trečiasis komandos baitai).	Pereiga į adresą $200C_{16}$	10
200B	MOV A, C	Registro C turinys persiunčiamas į kaupiklį A ($C \rightarrow A$).	Operacija nevykdoma	5

Adresas ₁₆	Komandos mnemonika	Atliekama operacija	Operacijos rezultatas	Taktų skaičius
200C	INR A	Kaupiklio A turinys didinamas vienetu ($A + 1 \rightarrow A$).	$A = FB_{16}$	5
200D	MOV C, A	Kaupiklio A turinys persiunčiamas į registrą C ($A \rightarrow C$).	$C = A = FB_{16}$	5

Sudėję visų vykdomų komandų taktų skaičius, gauname, kad šis programos fragmentas įvykdomas per 31 taktinį intervalą (komanda **MOV A, C** nevykdoma, kadangi įvyksta pereiga į adresą 200C₁₆). Kadangi mikroprocesoriaus taktinis dažnis $f_t = 2$ MHz, tai vieno taktinio intervalo trukmė $t_t = 0,5$ μs. Apskaičiuojame programos fragmento vykdymo trukmę: $t = N \times t_t = 31 \times 0,5 = 15,5$ μs.

8. Apskaičiuokite 7 užduotyje duoto programos fragmento vykdymo trukmę, kai mikroprocesoriaus taktinis dažnis:

- $f_t = 2,67$ MHz;
- $f_t = 3,13$ MHz;
- $f_t = 200$ kHz;
- $f_t = 2,5$ MHz;
- $f_t = 5$ MHz;
- $f_t = 6$ MHz.

9. Išnagrinėkite duotą programos fragmentą ir nustatykite, koks bus kaupiklio A arba išvesties prievado, kurio adresas yra 05₁₆, turinys, įvykdžius duotą programos fragmentą, bei kokia bus programos vykdymo trukmė. Pradiniai duomenys prieš operaciją yra šie: įvesties prievadas 06₁₆ = 01₁₆, mikroprocesoriaus taktinis dažnis $f_t = 2$ MHz.

a.	2000	MVI A, 05	d.	2000	MVI A, EE
	2002	MVI B, 00		2002	SUI 11
	2004	INR B		2004	JM 2002
	2005	SUB B		2007	NOP
	2006	JNZ 2004		2008	NOP
	2009	MOV A, B		2009	NOP
	200A	OUT 05		200A	OUT 05

<i>b.</i>	2000	ANI 00	<i>e.</i>	2000	MVI A, 0A
	2002	INR A		2002	MOV C, A
	2003	CPI FF		2003	NOP
	2005	JNZ 2002		2004	NOP
	2008	HLT		2005	DCR A
<i>c.</i>	2103	IN 06	<i>f.</i>	2006	JNZ 2003
	2105	MVI B, 64		2009	MOV A, C
	2107	MVI C, FA		2000	MVI A, 07
	2109	DCR C		2002	SUI 01
	210A	JNZ 2109		2004	JP 2002
	210D	DCR B		2007	NOP
	210E	JNZ 2107		2008	NOP
	2111	OUT 05		2009	NOP
				200A	OUT 05

10. Apskaičiuokite 9 užduotyje duotų programų fragmentų vykdymo trukmę, kai mikroprocesoriaus taktinis dažnis:

- $f_t = 5 \text{ MHz}$;
- $f_t = 6 \text{ MHz}$;
- $f_t = 3 \text{ MHz}$.

1.2. Komandų formatai ir operandų adresavimo būdai

- 11.** Parašykite programą, kuri persiųstų mikroprocesoriaus kaupiklio turinį į tris pasirinktus bendrosios paskirties registrus.

II. Sprendimas.

Šiai užduočiai įvykdyti taikysime registrinį operandų adresavimo būdą, kadangi užduotyje prašoma kaupiklio turinį persiųsti į tris laisvai pasirinktus bendrosios paskirties registrus. Esant šiam adresavimui, operandai yra registruose, kurių adresai nurodomi vieninteliame operacijos kodo baite. Kadangi registrų mažai, jiems adresuoti pakanka trijų dvejetainių skilčių (1.1 lentelė).

1.1 lentelė. Registrų adresai

Registras	Adresas ₂ komandoje	Registras	Adresas ₂ komandoje
B	000	H	100
C	001	L	101
D	010	M	110
E	011	A	111

Šioje lentelėje **M** – atminties ląstelė, kurios adresas nurodomas registrų poroje **H** (**H** ir **L** registrai).

Pvz., komandos **MOV R1, R2** formatas yra toks:

7	6	5	4	3	2	1	0
0	1	R1			R2		

Įrašę vietoje **R1** (registras, į kurį siunčiami duomenys) ir **R2** (registras, iš kurio siunčiami duomenys) konkrečių registrų adresus, gauname dvejetainį komandos kodą.

Tarkime, **R1** yra registras **B**, o **R2** – registras **A**. Įrašę į komandos formatą atitinkamus registrų adresus 000 ir 111 (1.1 lentelė), gauname dvejetainį komandos kodą: $01000111_2 = 47_{16}$. Šis šešiolyktainis kodas atitinka komandą **MOV B, A**. $B \leftarrow A$ (1 priedas).

Tai reiškia, kad kaupiklio **A** turinys persiunčiamas į registrą **B**. Taigi komandos su registriniu operandų adresavimu yra vieno baido ilgio: 1 baitas

C7	C6	C5	C4	C3	C2	C1	C0	– operacijos kodo baitas (OKB)
----	----	----	----	----	----	----	----	--------------------------------

Operandų adresai nurodomi vieninteliame operacijos kodo baite tiesiogiai.

Tarkime, kad kaupiklio **A** turinys yra lygus AA_{16} , jį persiūsime į tris pasirinktus bendrosios paskirties registrus **D**, **H** ir **L**.

Adresas ₁₆	Komandos kodas ₁₆	Komandos mnemonika	Komentaras
2000	57	MOV D, A	Kaupiklio A turinys persiunčiamas į registrą D ($A \rightarrow D, D = AA_{16}$).
2001	67	MOV H, A	Kaupiklio A turinys persiunčiamas į registrą H ($A \rightarrow H, H = AA_{16}$).
2002	6F	MOV L, A	Kaupiklio A turinys persiunčiamas į registrą L ($A \rightarrow L, L = AA_{16}$).
2003	76	HLT	Stabdomas mikroprocesoriaus darbas.

- 12.** Parašykite programą, kuri užpildytų mikroprocesoriaus kaupiklio turiniu visus mikroprocesoriaus bendrosios paskirties registrus.
- 13.** Parašykite programą, kuri sudėtų dviejų mikroprocesoriaus registrų turinius ir gautą rezultatą įrašytų į pasirinktą atminties ląstelę.
- Rezultatui įrašyti taikyti šalutinį registrinį operandų adresavimo būdą;
 - Rezultatui įrašyti taikyti tiesioginį operandų adresavimo būdą.

13. a. Sprendimas.

Šalutinis registrinis operandų adresavimo būdas skiriasi nuo registro tuo, kad registruose laikomas ne pats operandas, o šio operando adresas atmintyje. Kadangi adresas yra 16 bitų, jam saugoti taikoma registrų pora, turinti savo atskirą dvejetainį adresą. Kadangi registrų poros yra tik keturios, joms adresuoti pakanka dviejų dvejetainių skilčių (1.2 lentelė).

1.2 lentelė. Registrų porų adresai

Registrų pora	Adresas, komandoje
B pora (registrai B ir C)	00
D pora (registrai D ir E)	01
H pora (registrai H ir L)	10
PSW (kaupiklis A ir požymių registras F)	11

Pvz., komandos **MOV R, M** formatas yra toks:

7	6	5	4	3	2	1	0
0	1	R			M		

Įrašę vietoje **R** (registras, į kurį siunčiami duomenys) ir **M** (atminties ląstelės, kurios adresą nurodo registrų pora **H** (**H** ir **L** registrai), turinys) konkrečius adresus, gauname dvejetainį komandos kodą.

Tarkime, **R** yra registras **A**. Įrašę į komandos formatą atitinkamus adresus 111 ir 110 (1.1 lentelė), gauname dvejetainį komandos kodą: $01111110_2 = 7E_{16}$. Šis šešiolyktainis kodas atitinka komandą **MOV A, M**. $A \leftarrow M[HL]$ (1 priedas).

Tai reiškia, kad į kaupiklį **A** persiunčiami duomenys iš atminties ląstelės, kurios adresą nurodo registrų pora **H** (**H** ir **L** registrai), **M** čia reiškia registrų porą **H**.

Šiuo tikslu taip pat gali būti taikomos **B** (**B** ir **C** registrai) arba **D** (**D** ir **E** registrai) registrų poros. Pvz., **LDAX B. A ← M[BC]; STAX B. M[BC] ← A**. Čia **M[BC]** reiškia atminties ląstelę, kurios adresu yra registrų poros **B** turinys.

Taigi komandos su šalutiniu registriniu operandų adresavimu yra vieno baido ilgio:

1 baitas

C7	C6	C5	C4	C3	C2	C1	C0	– operacijos kodo baitas (OKB)
----	----	----	----	----	----	----	----	--------------------------------

Tarkime, kad sudėsime kaupiklio **A** ir registro **E** turinius. Kaupiklį mes pasirenkame būtinai, kadangi, atliekant aritmetines, logines ir poslinkio operacijas, jis yra operando šaltinis, t. y. vienas iš operacijoje dalyvau-

jančių operandų būtinai turi būti kaupiklyje. Atlikus operaciją, jame lieka rezultatas.

Priimkime, kad kaupiklio **A** turinys yra lygus 03_{16} , o registro **E** turinys yra lygus 02_{16} .

Gautą sudėties rezultatą (sumą) 05_{16} įrašysime į pasirinktą atminties ląstelę, kurios adresas yra $200A_{16}$.

Adresas ₁₆	Komandos kodas ₁₆	Komandos mnemonika	Komentaras
2000	21	LXI H, 200A	Į registrų porą HL persiunčiamas dviejų baitų tiesioginis operandas (antrasis ir trečiasis komandos baitai), t. y. operando adresas atmintyje ($0A_{16} \rightarrow L$, $20_{16} \rightarrow H$, $HL = 200A_{16}$).
2001	0A		
2002	20		
2003	83	ADD E	Registro E turinys pridamas prie kaupiklio A turinio ($A + E \rightarrow A$, $A = 05_{16}$).
2004	77	MOV M, A	Gauta dviejų registrų turinių suma iš kaupiklio A persiunčiama į atminties ląstelę, kurios adresas yra registrų poroje HL ($A \rightarrow M[HL]$, $A \rightarrow M[200A_{16}]$, $200A_{16} = 05_{16}$).
2005	76	HLT	Stabdomas mikroprocesoriaus darbas.

13. b. Sprendimas.

Komandos su tiesioginiu operandų adresavimu yra dviejų arba trijų baitų ilgio. Čia prievado numeris (adresas) yra antrasis komandos baitas, o operando adresas yra antrasis ir trečiasis baitai:

1 baitas

C7	C6	C5	C4	C3	C2	C1	C0	– operacijos kodo baitas (OKB)
----	----	----	----	----	----	----	----	--------------------------------

2 baitas

A7	A6	A5	A4	A3	A2	A1	A0	– prievado numeris (adresas) (N)
----	----	----	----	----	----	----	----	----------------------------------

1 baitas

C7	C6	C5	C4	C3	C2	C1	C0	– operacijos kodo baitas (OKB)
----	----	----	----	----	----	----	----	--------------------------------

2 baitas

A7	A6	A5	A4	A3	A2	A1	A0	– žemesnysis adreso baitas (ADRL)
----	----	----	----	----	----	----	----	-----------------------------------

3 baitas

A7	A6	A5	A4	A3	A2	A1	A0	– aukštesnysis adreso baitas (ADRH)
----	----	----	----	----	----	----	----	-------------------------------------

Atminties adresai yra 16 bitų, todėl, kaip ir 16 bitų duomenys, laikomi dalimis: ADRL ir ADRH – žemesnysis ir aukštesnysis adreso, kuriame yra operandas, baitai. Pvz., **LDA ADR. A** ← **M[ADR]**. Čia **M[ADR]** – operando adresas atmintyje (antrasis ir trečiasis komandos baitai).

Tai reiškia, kad į kaupiklį **A** persiunčiami duomenys iš atminties ląstelės, kurios adresą nurodo antrasis ir trečiasis komandos baitai.

Tarkime, kad kaupiklio **A** turinys yra lygus AB_{16} , o registro **L** turinys yra lygus 03_{16} .

Gautą sudėties rezultatą (sumą) AE_{16} įrašysime į pasirinktą atminties ląstelę, kurios adresas yra 2110_{16} .

Adresas ₁₆	Komandos kodas ₁₆	Komandos mnemonika	Komentaras
2000	85	ADD L	Registro L turinys pridedamas prie kaupiklio A turinio ($A + L \rightarrow A$, $A = AE_{16}$).
2001	32	STA 2110	Gauta dviejų registrų turinių suma iš kaupiklio A persiunčiama į atminties ląstelę, kurios adresas yra antrasis ir trečiasis komandos baitai ($A \rightarrow M[ADR]$, $A \rightarrow M[2110_{16}]$, $2110_{16} = AE_{16}$).
2002	10		
2003	21		
2004	76	HLT	Stabdomas mikroprocesoriaus darbas.

14. Parašykite programą, kuri padidintų pasirinktą kiekį kartų vieno iš bendrosios paskirties registrų turinį ir gautą rezultatą įrašytų į pasirinktą atminties ląstelę.
 - a. Rezultatui įrašyti taikyti šalutinį registrinį operandų adresavimo būdą;
 - b. Rezultatui įrašyti taikyti tiesioginį operandų adresavimo būdą.
15. Parašykite programą, kuri sumažintų pasirinktą kiekį kartų vieno iš bendrosios paskirties registrų turinį ir gautą rezultatą įrašytų į pasirinktą atminties ląstelę.
 - a. Rezultatui įrašyti taikyti šalutinį registrinį operandų adresavimo būdą;
 - b. Rezultatui įrašyti taikyti tiesioginį operandų adresavimo būdą.
16. Parašykite programą, kuri užpildytų visus mikroprocesoriaus registrus skirtingais 8 bitų operandais, esančiais pasirinktose skirtingose atminties ląstelėse.
 - a. Operandams adresuoti taikyti šalutinį registrinį operandų adresavimo būdą;

- b. Operandams adresuoti taikyti tiesioginį operandų adresavimo būdą;
- c. Operandams adresuoti parinkti skirtingus operandų adresavimo būdus.

17. Parašykite programą, kuri persiųstų 8 bitų operandą į pasirinktą atminties ląstelę.

- a. Operandui persiųsti taikyti šalutinį registrinį operandų adresavimo būdą;
- b. Operandui persiųsti taikyti tiesioginį operandų adresavimo būdą;
- c. Operandui persiųsti taikyti tiesioginį operandą.

17. c. Sprendimas.

Tiesioginis operandas – tai vieno arba dviejų baitų duomenys, prijungti prie operacijos kodo baito (antrasis arba antrasis ir trečiasis komandos baitai). Komandos su tiesioginiu operandu yra dviejų arba trijų baitų ilgio:

1 baitas

C7	C6	C5	C4	C3	C2	C1	C0	– operacijos kodo baitas (OKB)
----	----	----	----	----	----	----	----	--------------------------------

2 baitas

D7	D6	D5	D4	D3	D2	D1	D0	– 8 bitų duomenys (D8)
----	----	----	----	----	----	----	----	------------------------

1 baitas

C7	C6	C5	C4	C3	C2	C1	C0	– operacijos kodo baitas (OKB)
----	----	----	----	----	----	----	----	--------------------------------

2 baitas

D7	D6	D5	D4	D3	D2	D1	D0	– žemesnysis 16 bitų duomenų baitas (D16L)
----	----	----	----	----	----	----	----	--

3 baitas

D7	D6	D5	D4	D3	D2	D1	D0	– aukštesnysis 16 bitų duomenų baitas (D16H)
----	----	----	----	----	----	----	----	--

16 bitų duomenys, laikomi dalimis: žemesnysis baitas – D16L, aukštesnysis baitas – D16H.

Komandų su tiesioginiu operandu formatas yra panašus į komandų formatą su tiesioginiu operandu adresavimu. Skirtumas tik tas, kad, kai operandas yra tiesioginis, po operacijos kodo baito seka ne adreso, o duomenų baitai. Pvz., komanda **MVI R, D8**. $R \leftarrow D8$. Čia **D8** – duomenų baitas, einantis po operacijos kodo baito (antrasis komandos baitas).

Tai reiškia, kad į registrą **B** persiunčiamas 8 bitų tiesioginis operandas (antrasis komandos baitas).

Tarkime, kad 8 bitų tiesioginis operandas yra lygus 01_{16} , jį persiūsi-me į pasirinktą atminties ląstelę, kurios adresas yra 2040_{16} .

Adresas ₁₆	Komandos kodas ₁₆	Komandos mnemonika	Komentaras
2000	21	LXI H, 2040	Į registrų porą HL persiunčiamas dviejų baitų tiesioginis operandas (antrasis ir trečiasis komandos baitai), t. y. operando adresas atmintyje ($40_{16} \rightarrow L$, $20_{16} \rightarrow H$, $HL = 2040_{16}$).
2001	40		
2002	20		

Adresas ₁₆	Komandos kodas ₁₆	Komandos mnemonika	Komentaras
2003 2004	36 01	MVI M, 01	Į atminties ląstelę, kurios adresas yra registrų poroje HL, persiunčiamas vieno baito tiesioginis operandas (antrasis komandos baitas) ($01_{16} \rightarrow M[HL]$, $01_{16} \rightarrow M[2040_{16}]$, $2040_{16} = 01_{16}$).
2005	76	HLT	Stabdomas mikroprocesoriaus darbas.

18. Parašykite programą, kuri į visus mikroprocesoriaus bendrosios paskirties registrus įkeltų konstantą 11_{16} .

19. Parašykite programą, kuri sukeistų vietomis du 8 bitų operandus, esančius pasirinktose skirtingose atminties ląstelėse.

- Operandams adresuoti taikyti šalutinį registrinį operandų adresavimo būdą;
- Operandams adresuoti taikyti tiesioginį operandų adresavimo būdą;
- Operandams adresuoti parinkti skirtingus operandų adresavimo būdus.

19. a. Sprendimas.

Tarkime, kad pirmasis operandas 11_{16} yra atminties ląstelėje, kurios adresas yra 2030_{16} , antrasis operandas yra $22_{16} - 2060_{16}$.

Adresas ₁₆	Komandos kodas ₁₆	Komandos mnemonika	Komentaras
2000 2001 2002	21 30 20	LXI H, 2030	Į registrų porą HL persiunčiamas dviejų baitų tiesioginis operandas (antrasis ir trečiasis komandos baitai), t. y. pirmojo operando adresas ($30_{16} \rightarrow L$, $20_{16} \rightarrow H$, $HL = 2030_{16}$).

Adresas ₁₆	Komandos kodas ₁₆	Komandos mnemonika	Komentaras
2003 2004 2005	01 60 20	LXI B, 2060	Į registrų porą BC persiunčiamas dviejų baitų tiesioginis operandas (antrasis ir trečiasis komandos baitai), t. y. antrojo operando adresas ($60_{16} \rightarrow C$, $20_{16} \rightarrow B$, $BC = 2060_{16}$).
2006	56	MOV D, M	Pirmasis operandas iš atminties ląstelės, kurios adresas yra registrų poroje HL, persiunčiamas į registrą D ($M[HL] \rightarrow D$, $M[2030_{16}] \rightarrow D$, $D = 11_{16}$).
2007	0A	LDAX B	Antrasis operandas iš atminties ląstelės, kurios adresas yra registrų poroje BC, persiunčiamas į kaupiklį A ($M[BC] \rightarrow A$, $M[2060_{16}] \rightarrow A$, $A = 22_{16}$).
2008	77	MOV M, A	Kaupiklio A turinys persiunčiamas į atminties ląstelę, kurios adresas yra registrų poroje HL ($A \rightarrow M[HL]$, $A \rightarrow M[2030_{16}]$, $2030_{16} = 22_{16}$).
2009	7A	MOV A, D	Registro D turinys persiunčiamas į kaupiklį A ($D \rightarrow A$, $A = 11_{16}$).
200A	02	STAX B	Kaupiklio A turinys persiunčiamas į atminties ląstelę, kurios adresas yra registrų poroje BC ($A \rightarrow M[BC]$, $A \rightarrow M[2060_{16}]$, $2060_{16} = 11_{16}$).
200B	76	HLT	Stabdomas mikroprocesoriaus darbas.

20. Parašykite programą, kuri sudėtų du 8 bitų operandus, esančius pasirinktose skirtingose atminties ląstelėse, ir gautą rezultatą įrašytų į pasirinktą trečiąją atminties ląstelę.

a. Operandams adresuoti taikyti šalutinį registrinį operandų adresavimo būdą;

- b. Operandams adresuoti taikyti tiesioginį operandų adresavimo būdą;
- c. Operandams adresuoti taikyti tiesioginį operandą;
- d. Operandams adresuoti parinkti skirtingus operandų adresavimo būdus.

20. a. Sprendimas.

Tarkime, kad pirmasis operandas 05_{16} yra atminties ląstelėje, kurios adresas yra 2010_{16} , antrasis operandas yra $06_{16} - 2011_{16}$, t. y. operandai yra gretimose atminties ląstelėse.

Gautą sudėties rezultatą (sumą) $0B_{16}$ įrašysime į pasirinktą atminties ląstelę, kurios adresas yra 2012_{16} .

Adresas ₁₆	Komandos kodas ₁₆	Komandos mnemonika	Komentaras
2000	21	LXI H, 2010	Į registrų porą HL persiunčiamas dviejų baitų tiesioginis operandas (antrasis ir trečiasis komandos baitai), t. y. pirmojo operando adresas ($10_{16} \rightarrow L$, $20_{16} \rightarrow H$, $HL = 2010_{16}$).
2001	10		
2002	20		
2003	7E	MOV A, M	Pirmasis operandas iš atminties ląstelės, kurios adresas yra registrų poroje HL, persiunčiamas į kaupiklį A ($M[HL] \rightarrow A$, $M[2010_{16}] \rightarrow A$, $A = 05_{16}$).
2004	2C	INR L	Registro L turinys didinamas vienetu ($L + 1 \rightarrow L$, $L = 11_{16}$).
2005	86	ADD M	Antrasis operandas iš atminties ląstelės, kurios adresas yra registrų poroje HL, pridamas prie kaupiklio A turinio ($A + M[HL] \rightarrow A$, $A + M[2011_{16}] \rightarrow A$, $A = 0B_{16}$).
2006	2C	INR L	Registro L turinys didinamas vienetu ($L + 1 \rightarrow L$, $L = 12_{16}$).

Adresas ₁₆	Komandos kodas ₁₆	Komandos mnemonika	Komentaras
2007	77	MOV M, A	Gauta suma iš kaupiklio A persiunčiama į atminties ląstelę, kurios adresas yra registrų poroje HL ($A \rightarrow M[HL]$, $A \rightarrow M[2012_{16}]$, $2012_{16} = 0B_{16}$).
2008	76	HLT	Stabdomas mikroprocesoriaus darbas.

Sprendžiant šią užduotį, registrų poroje **HL** esantį atminties ląstelės, kurioje yra pasirinktas operandas, adresą galima nurodyti ir taikant kitą komandą, pvz., komandą **INR L** pakeisti registrų poros **HL** turinio inkrementavimo komanda **INX H**.

Jei pasirinktume, kad operandai yra nutolusiose viena nuo kitos atminties ląstelėse, t. y. pirmasis operandas 05_{16} yra atminties ląstelėje, kurios adresas yra 2010_{16} , o antrasis operandas yra $06_{16} - 2050_{16}$, šio uždavinio sprendimas būtų šiek tiek kitoks.

Adresas ₁₆	Komandos kodas ₁₆	Komandos mnemonika	Komentaras
2000	21	LXI H, 2010	Į registrų porą HL persiunčiamas dviejų baitų tiesioginis operandas (antrasis ir trečiasis komandos baitai), t. y. pirmojo operando adresas ($10_{16} \rightarrow L$, $20_{16} \rightarrow H$, $HL = 2010_{16}$).
2001	10		
2002	20		
2003	7E	MOV A, M	Pirmasis operandas iš atminties ląstelės, kurios adresas yra registrų poroje HL, persiunčiamas į kaupiklį A ($M[HL] \rightarrow A$, $M[2010_{16}] \rightarrow A$, $A = 05_{16}$).
2004	2E	MVI L, 50	Į registrą L persiunčiamas vieno baito tiesioginis operandas (antrasis komandos baitas) ($50_{16} \rightarrow L$, $L = 50_{16}$).
2005	50		

Adresas ₁₆	Komandos kodas ₁₆	Komandos mnemonika	Komentaras
2006	86	ADD M	Antrasis operandas iš atminties ląstelės, kurios adresas yra registrų poroje HL, pridedamas prie kaupiklio A turinio ($A + M[HL] \rightarrow A$, $A + M[2050_{16}] \rightarrow A$, $A = 0B_{16}$).
2007 2008	2E 12	MVI L, 12	Į registrą L persiunčiamas vieno baito tiesioginis operandas (antrasis komandos baitas) ($12_{16} \rightarrow L$, $L = 12_{16}$).
2009	77	MOV M, A	Gauta suma iš kaupiklio A persiunčiama į atminties ląstelę, kurios adresas yra registrų poroje HL ($A \rightarrow M[HL]$, $A \rightarrow M[2012_{16}]$, $2012_{16} = 0B_{16}$).
200A	76	HLT	Stabdomas mikroprocesoriaus darbas.

Sprendžiant šią užduotį, registrų poroje **HL** esantį atminties ląstelės, kurioje yra pasirinktas operandas, adresą galima nurodyti ir taikant kitą komandą, pvz., komandą **MVI L, D8** pakeisti dviejų baitų tiesioginio operando persiuntimo komanda **LXI H, D16**.

- 21.** Parašykite programą, kuri atimtų du 8 bitų operandus, esančius pasirinktose skirtingose atminties ląstelėse, ir gautą rezultatą įrašytų į pasirinktą trečiąją atminties ląstelę.
- Operandams adresuoti taikyti šalutinį registrinį operandų adresavimo būdą;
 - Operandams adresuoti taikyti tiesioginį operandų adresavimo būdą;
 - Operandams adresuoti taikyti tiesioginį operandą;
 - Operandams adresuoti parinkti skirtingus operandų adresavimo būdus.

22. Parašykite programą, kuri atimtų du 8 bitų operandus, esančius pasirinktose skirtingose atminties ląstelėse, ir gautą rezultatą įrašytų į dėklą nurodytu adresu.

- Operandams adresuoti taikyti šalutinį registrinį operandų adresavimo būdą;
- Operandams adresuoti taikyti tiesioginį operandų adresavimo būdą;
- Operandams adresuoti taikyti tiesioginį operandą;
- Operandams adresuoti parinkti skirtingus operandų adresavimo būdus.

22. b. Sprendimas.

Dėklas yra specialioji operatyviosios atminties sritis, skirta grąžties iš paprogramių adresams, tarpiniams skaičiavimų rezultatams, konstantoms ar duomenims ir pan. laikinai saugoti, t. y. laikina informacijos saugykla. Dėklą programuotojas gali susikurti (adresuoti), taikydamas specialųjį 16 skilčių registrą **SP**, vadinamą dėklo rodykle (angl. *Stack Pointer*) arba dėklo viršūne (angl. *Stack Top*).

Dėklas veikia LIFO (angl. *Last In, First Out*) principu (*paskutinis į, pirmas iš*), t. y. duomenys į dėklą įrašomi iš eilės tokia tvarka, kokia pateikiami, tad skaityti iš dėklo galima tik vėliausiai įrašytus duomenis. Juos nuskaičius, bus galima skaityti prieš juos buvusius duomenis ir t. t. Tokį dėklo veikimo principą galima vaizdžiai paaiškinti lėkščių rietuvės pavyzdžiu (1.3 pav.).

1.3 pav. Lėkščių rietuvė

Norint paimti pirmąją lėkštę, pirmiausia reikia nukelti trečiąją, o paskui antrąją lėkštę, t. y. pirmiausia į dėklą įrašytus duomenis bus galima nuskaityti vėliausiai.

Dėklo rodyklėje **SP** saugomas dėklo pradžios adresas, kuris nurodo, kur turi būti įrašyti arba iš kur turi būti nuskaityti duomenys. Duomenims rašyti į dėklą yra skirta komanda **PUSH**. Jos veikimo principas parodytas 1.4 paveiksle.

1.4 pav. Komandos **PUSH** veikimas

Komandoje **PUSH** nurodytos registrų poros **RP** turinys įrašomas dviejose dėklo atminties ląstelėse, kurių adresus nurodo dėklo rodyklė **SP**. Pirmojo registrų poros registro **RPH** turinys įrašomas į dėklo atminties ląstelę, kurios adresas vienetu mažesnis nei esama dėklo rodyklės **SP** reikšmė (1), o antrojo registrų poros registro **RPL** turinys įrašomas į dėklo atminties ląstelę, kurios adresas dviem vienetais mažesnis nei esama dėklo rodyklės **SP** reikšmė (2). Dėklo rodyklės **SP** turinys, įvykdžius **PUSH** komandą, sumažėja dviem vienetais. Taigi dėklas užpildomas, t. y. duomenys į dėklą įrašomi adresų mažėjimo kryptimi.

Tarkime, kad pirmasis operandas 02_{16} yra atminties ląstelėje, kurios adresas yra 2081_{16} , antrasis operandas yra $0C_{16} - 2090_{16}$, t. y. operandai yra nutolusiose viena nuo kitos atminties ląstelėse.

Gautą atminties rezultatą (skirtumą) $0A_{16}$ įrašysime į dėklą adresu 2015_{16} .

Adresas ₁₆	Komandos kodas ₁₆	Komandos mnemonika	Komentaras
2000	31	LXI SP, 2015	Į dėklo rodyklę SP persiunčiamas dviejų baitų tiesioginis operandas (antrasis ir trečiasis komandos baitai), t. y. nustatoma dėklo viršūnė ($2015_{16} \rightarrow SP$, $SP = 2015_{16}$).
2001	15		
2002	20		
2003	3A	LDA 2081	Pirmasis operandas iš atminties ląstelės, kurios adresas yra antrasis ir trečiasis komandos baitai, persiunčiamas į kaupiklį A ($M[ADR] \rightarrow A$, $M[2081_{16}] \rightarrow A$, $A = 02_{16}$).
2004	81		
2005	20		
2006	47	MOV B, A	Kaupiklio A turinys persiunčiamas į registrą B ($A \rightarrow B$, $B = 02_{16}$).
2007	3A	LDA 2090	Antrasis operandas iš atminties ląstelės, kurios adresas – antrasis ir trečiasis komandos baitai, persiunčiamas į kaupiklį A ($M[ADR] \rightarrow A$, $M[2090_{16}] \rightarrow A$, $A = 0C_{16}$).
2008	90		
2009	20		
200A	90	SUB B	Registro B turinys atimamas iš kaupiklio A turinio ($A - B \rightarrow A$, $A = 0A_{16}$).
200B	F5	PUSH PSW	Kaupiklyje A gautas dviejų 8 bitų operandų skirtumas kartu su požymių registro F turiniu persiunčiami į dėklą nurodytu adresu ($A \rightarrow M[SP - 1]$, $F \rightarrow M[SP - 2]$, $SP - 2 \rightarrow SP$, $A \rightarrow M[2014_{16}]$, $F \rightarrow M[2013_{16}]$, $2014_{16} = 0A_{16}$, $2013_{16} = 14_{16}$, $SP = 2013_{16}$).
200C	76	HLT	Stabdomas mikroprocesoriaus darbas.

- 23.** Parašykite programą, kuri sudėtų du 8 bitų operandus, esančius pasirinktose skirtingose atminties ląstelėse, ir gautą rezultatą įrašytų į dėklą nurodytu adresu.

- a. Operandams adresuoti taikyti šalutinį registrinį operandų adresavimo būdą;
- b. Operandams adresuoti taikyti tiesioginį operandų adresavimo būdą;
- c. Operandams adresuoti taikyti tiesioginį operandą;
- d. Operandams adresuoti parinkti skirtingus operandų adresavimo būdus.

24. Parašykite programą, kuri sudėtų du 8 bitų operandus, esančius dėkle, ir gautą rezultatą įrašytų atgal į dėklą.

- a. Operandai yra gretimose dėklo atminties ląstelėse;
- b. Operandai yra nutolusiose viena nuo kitos dėklo atminties ląstelėse.

24. a. *Sprendimas.*

Duomenims skaityti iš dėklo yra skirta komanda **POP**. Jos veikimo principas parodytas 1.5 paveiksle.

1.5 pav. Komandos **POP** veikimas

Dėklo atminties ląstelių, kurių adresus nurodo dėklo rodyklė **SP**, turiniai įrašomi į komandoje **POP** nurodytą registrų porą **RP**. Atminties ląstelės, kurios adresas sutampa su esama dėklo rodyklės **SP** reikšme, turinys įrašomas į antrąjį registrų poros registrą **RPL** (1), o atminties ląstelės, kurios adresas vienetu didesnis nei esama dėklo rodyklės **SP** reikšmė,

turinys įrašomas į pirmąjį registrų poros registrą **RPH** (2). Dėklo rodyklės **SP** turinys, įvykdžius **POP** komandą, didėja dviem vienetais. Taigi duomenys iš dėklo skaitomi adresų didėjimo kryptimi.

Tarkime, kad dėklas atmintyje yra nurodytas adresu 2070_{16} . Pirmasis 8 bitų operandas 03_{16} yra dėkle adresu 2070_{16} , antrasis 8 bitų operandas $0A_{16} - 2071_{16}$, t. y. operandai yra gretimose dėklo atminties ląstelėse.

Gautą sudėties rezultatą (sumą) $0D_{16}$ įrašysime atgal į dėklą.

Adresas ₁₆	Komandos kodas ₁₆	Komandos mnemonika	Komentaras
2000	31	LXI SP, 2070	Į dėklo rodyklę SP persiunčiamas dviejų baitų tiesioginis operandas (antrasis ir trečiasis komandos baitai), t. y. nustatoma dėklo viršūnė ($2070_{16} \rightarrow SP$, $SP = 2070_{16}$).
2001	70		
2002	20		
2003	C1	POP B	Pirmasis ir antrasis 8 bitų operandai persiunčiami iš dėklo į B ir C registrus ($M[SP] \rightarrow C$, $M[SP + 1] \rightarrow B$, $SP + 2 \rightarrow SP$, $M[2070_{16}] \rightarrow C$, $M[2071_{16}] \rightarrow B$, $C = 03_{16}$, $B = 0A_{16}$, $SP = 2072_{16}$).
2004	78	MOV A, B	Registro B turinys persiunčiamas į kaupiklį A ($B \rightarrow A$, $A = 0A_{16}$).
2005	81	ADD C	Registro C turinys pridedamas prie kaupiklio A turinio ($A + C \rightarrow A$, $A = 0D_{16}$).
2006	F5	PUSH PSW	Kaupiklyje A gauta dviejų 8 bitų operandų suma kartu su požymių registro F turiniu persiunčiama į dėklą nurodytu adresu ($A \rightarrow M[SP - 1]$, $F \rightarrow M[SP - 2]$, $SP - 2 \rightarrow SP$, $A \rightarrow M[2071_{16}]$, $F \rightarrow M[2070_{16}]$, $2071_{16} = 0D_{16}$, $2070_{16} = 00_{16}$, $SP = 2070_{16}$).
2007	76	HLT	Stabdomas mikroprocesoriaus darbas.

25. Parašykite programą, kuri atimtų du 8 bitų operandus, esančius dėkle, ir gautą rezultatą įrašytų atgal į dėklą.

- Operandai yra gretimose dėklo atminties ląstelėse;
- Operandai yra nutolusiose viena nuo kitos dėklo atminties ląstelėse.

25. b. *Sprendimas.*

Tarkime, kad pirmasis 8 bitų operandas 08_{16} yra dėkle adresu 2092_{16} , antrasis 8 bitų operandas $06_{16} - 2095_{16}$, t. y. operandai yra nutolusiose viena nuo kitos dėklo atminties ląstelėse.

Gautą atimties rezultatą (skirtumą) 02_{16} įrašysime atgal į dėklą.

Adresas ₁₆	Komandos kodas ₁₆	Komandos mnemonika	Komentaras
2000	31	LXI SP, 2092	Į dėklo rodyklę SP persiunčiamas dviejų baitų tiesioginis operandas (antrasis ir trečiasis komandos baitai), t. y. nustatoma dėklo viršūnė ($2092_{16} \rightarrow SP$, $SP = 2092_{16}$).
2001	92		
2002	20		
2003	C1	POP B	Pirmasis 8 bitų operandas persiunčiamas iš dėklo į C registrą ($M[SP] \rightarrow C$, $SP + 2 \rightarrow SP$, $M[2092_{16}] \rightarrow C$, $C = 08_{16}$, $SP = 2094_{16}$).
2004	79	MOV A, C	Registro C turinys persiunčiamas į kaupiklį A ($C \rightarrow A$, $A = 08_{16}$).
2005	31	LXI SP, 2095	Į dėklo rodyklę SP persiunčiamas dviejų baitų tiesioginis operandas (antrasis ir trečiasis komandos baitai), t. y. nustatoma nauja dėklo viršūnė ($2095_{16} \rightarrow SP$, $SP = 2095_{16}$).
2006	95		
2007	20		
2008	C1	POP B	Antrasis 8 bitų operandas persiunčiamas iš dėklo į C registrą ($M[SP] \rightarrow C$, $SP + 2 \rightarrow SP$, $M[2095_{16}] \rightarrow C$, $C = 06_{16}$, $SP = 2097_{16}$).
2009	91	SUB C	Registro C turinys atimamas iš kaupiklio A turinio ($A - C \rightarrow A$, $A = 02_{16}$).

Adresas ₁₆	Komandos kodas ₁₆	Komandos mnemonika	Komentaras
200A	F5	PUSH PSW	Kaupiklyje A gautas dviejų 8 bitų operandų skirtumas kartu su požymių registro F turiniu persiunčiamas į dėklą nurodytu adresu ($A \rightarrow M[SP - 1]$, $F \rightarrow M[SP - 2]$, $SP - 2 \rightarrow SP$, $A \rightarrow M[2096_{16}]$, $F \rightarrow M[2095_{16}]$, $2096_{16} = 02_{16}$, $2095_{16} = 10_{16}$, $SP = 2095_{16}$).
200B	76	HLT	Stabdomas mikroprocesoriaus darbas.

26. Parašykite programą, kuri sukeistų vietomis du 8 bitų operandus, esančius dėkle.

- Operandai yra gretimose dėklo atminties ląstelėse;
- Operandai yra nutolusiose viena nuo kitos dėklo atminties ląstelėse.

27. Parašykite programą, kuri persiųstų 16 bitų operandą į pasirinktas atminties ląsteles.

- Operando baitams persiųsti taikyti šalutinį registrinį operandų adresavimo būdą;
- Operando baitams persiųsti taikyti tiesioginį operandų adresavimo būdą;
- Operando baitams persiųsti taikyti tiesioginį operandą.

27. a. *Sprendimas.*

Tarkime, kad 16 bitų operandas yra lygus 0102_{16} , jį persiūsime į atminties ląsteles, kurių adresai yra 2023_{16} ir 2024_{16} .

Adresas ₁₆	Komandos kodas ₁₆	Komandos mnemonika	Komentaras
2000	21	LXI H, 2023	Į registrų porą HL persiunčiamas dviejų baitų tiesioginis operandas (antrasis ir trečiasis komandos baitai), t. y. 16 bitų operando žemesniojo baito adresas atmintyje ($23_{16} \rightarrow L$, $20_{16} \rightarrow H$, $HL = 2023_{16}$).
2001	40		
2002	20		

Adresas ₁₆	Komandos kodas ₁₆	Komandos mnemonika	Komentaras
2003 2004	3E 02	MVI A, 02	Į kaupiklį A persiunčiamas vieno baido tiesioginis operandas (antrasis komandos baitas), t. y. 16 bitų operando žemesnysis baitas ($02_{16} \rightarrow A, A = 02_{16}$).
2005 2006	06 01	MVI B, 01	Į registrą B persiunčiamas vieno baido tiesioginis operandas (antrasis komandos baitas), t. y. 16 bitų operando aukštesnysis baitas ($01_{16} \rightarrow B, B = 01_{16}$).
2007	77	MOV M, A	16 bitų operando žemesnysis baitas iš kaupiklio A persiunčiamas į atminties ląstelę, kurios adresas yra registrų poroje HL ($A \rightarrow M[HL], A \rightarrow M[2023_{16}], 2023_{16} = 02_{16}$).
2008	2C	INR L	Registro L turinys didinamas vienetu ($L+1 \rightarrow L, L = 24_{16}$).
2009	70	MOV M, B	16 bitų operando aukštesnysis baitas iš registro B persiunčiamas į atminties ląstelę, kurios adresas yra registrų poroje HL ($A \rightarrow M[HL], A \rightarrow M[2024_{16}], 2024_{16} = 01_{16}$).
200A	76	HLT	Stabdomas mikroprocesoriaus darbas.

- 28.** Parašykite programą, kuri sukeistų vietomis du 16 bitų operandus, esančius atmintyje.
- Operandų baitams adresuoti taikyti šalutinį registrinį operandų adresavimo būdą;
 - Operandų baitams adresuoti taikyti tiesioginį operandų adresavimo būdą;
 - Operandų baitams adresuoti parinkti skirtingus operandų adresavimo būdus.

28. b. *Sprendimas.*

Tarkime, kad pirmasis 16 bitų operandas 1122_{16} yra gretimose atminties ląstelėse 2080_{16} ir 2081_{16} , t. y. $2080_{16} - 22_{16}$, o $2081_{16} - 11_{16}$. Antrasis 16 bitų operandas 3344_{16} yra irgi gretimose atminties ląstelėse 2100_{16} ir 2101_{16} , t. y. $2100_{16} - 44_{16}$, o $2101_{16} - 33_{16}$.

Adresas ₁₆	Komandos kodas ₁₆	Komandos mnemonika	Komentaras
2000	2A	LHLD 2080	Pirmasis 16 bitų operandas iš atminties ląstelių, iš kurių pirmosios adresas yra antrasis ir trečiasis komandos baitai, persiunčiamas į registrus L ir H ($M[ADR] \rightarrow L$, $M[ADR + 1] \rightarrow H$, $M[2080_{16}] \rightarrow L$, $M[2081_{16}] \rightarrow H$, $L = 22_{16}$, $H = 11_{16}$).
2001	80		
2002	20		
2003	EB	XCHG	Registrai H ir L turiniai sukeičiami su registrai D ir E turiniais ($H \leftrightarrow D$, $L \leftrightarrow E$, $D = 11_{16}$, $E = 22_{16}$).
2004	2A	LHLD 2100	Antrasis 16 bitų operandas iš atminties ląstelių, iš kurių pirmosios adresas yra antrasis ir trečiasis komandos baitai, persiunčiamas į registrus L ir H ($M[ADR] \rightarrow L$, $M[ADR + 1] \rightarrow H$, $M[2100_{16}] \rightarrow L$, $M[2101_{16}] \rightarrow H$, $L = 44_{16}$, $H = 33_{16}$).
2005	00		
2006	21		
2007	22	SHLD 2080	Antrasis 16 bitų operandas iš registrai L ir H persiunčiamas į atminties ląsteles, iš kurių pirmosios adresas yra antrasis ir trečiasis komandos baitai ($L \rightarrow M[ADR]$, $H \rightarrow M[ADR + 1]$, $L \rightarrow M[2080_{16}]$, $H \rightarrow M[2081_{16}]$, $2080_{16} = 44_{16}$, $2081_{16} = 33_{16}$).
2008	80		
2009	20		

Adresas ₁₆	Komandos kodas ₁₆	Komandos mnemonika	Komentaras
200A	EB	XCHG	Registų H ir L turiniai sukeičiami su registų D ir E turiniais ($H \leftrightarrow D$, $L \leftrightarrow E$, $H = 11_{16}$, $L = 22_{16}$).
200B	22	SHLD 2100	Pirmasis 16 bitų operandas iš registų L ir H persiunčiamas į atminties ląsteles, iš kurių pirmosios adresas yra antrasis ir trečiasis komandos baitai ($L \rightarrow M[ADR]$, $H \rightarrow M[ADR + 1]$, $L \rightarrow M[2100_{16}]$, $H \rightarrow M[2101_{16}]$, $2100_{16} = 22_{16}$, $2101_{16} = 11_{16}$).
200C	00		
200D	21		
200E	76	HLT	Stabdomas mikroprocesoriaus darbas.

29. Parašykite programą, kuri sudėtų du 16 bitų operandus, esančius atmintyje, ir gautą rezultatą įrašytų atgal į atmintį.

- Operandų baitams adresuoti taikyti šalutinį registrinį operandų adresavimo būdą;
- Operandų baitams adresuoti taikyti tiesioginį operandų adresavimo būdą;
- Operandų baitams adresuoti taikyti tiesioginį operandą;
- Operandų baitams adresuoti parinkti skirtingus operandų adresavimo būdus.

29. d. Sprendimas.

Tarkime, kad pirmasis 16 bitų operandas 3322_{16} yra gretimose atminties ląstelėse 2001_{16} ir 2002_{16} , t. y. $2001_{16} - 22_{16}$, o $2002_{16} - 33_{16}$. Antrasis 16 bitų operandas 2244_{16} yra irgi gretimose atminties ląstelėse 2023_{16} ir 2024_{16} , t. y. $2023_{16} - 44_{16}$, o $2024_{16} - 22_{16}$.

Gautą sudėties rezultatą (sumą) $3322_{16} + 2244_{16} = 5566_{16}$ įrašysime į atminties ląsteles 2025_{16} ir 2026_{16} , t. y. $2025_{16} - 66_{16}$, o $2026_{16} - 55_{16}$.

Adresas ₁₆	Komandos kodas ₁₆	Komandos mnemonika	Komentaras
2000	11	LXI D, 3322	Į registrų porą DE persiunčiamas dviejų baitų tiesioginis operandas (antrasis ir trečiasis komandos baitai), t. y. pirmasis 16 bitų operandas ($22_{16} \rightarrow E, 33_{16} \rightarrow D, DE = 3322_{16}$).
2001	22		
2002	33		
2003	2A	LHLD 2023	Antrasis 16 bitų operandas iš atmin-ties ląstelių, iš kurių pirmosios adre-sas yra antrasis ir trečiasis komandos baitai, persiunčiamas į registrus L ir H ($M[ADR] \rightarrow L, M[ADR + 1] \rightarrow H, M[2023_{16}] \rightarrow L, M[2024_{16}] \rightarrow H, L = 44_{16}, H = 22_{16}$).
2004	23		
2005	20		
2006	7B	MOV A, E	Registro E turinys persiunčiamas į kau-piklį A ($E \rightarrow A, A = 22_{16}$).
2007	85	ADD L	Registro L turinys pridedamas prie kau-piklio A turinio, t. y. sudedami žemes-nieji 16 bitų operandų baitai ($A + L \rightarrow A, A = 66_{16}$).
2008	6F	MOV L, A	Kaupiklio A turinys persiunčiamas į re-gistrą L ($A \rightarrow L, L = 66_{16}$).
2009	7A	MOV A, D	Registro D turinys persiunčiamas į kau-piklį A ($D \rightarrow A, A = 33_{16}$).
200A	8C	ADC H	Registro H ir požymių registro F perna-šos skilties CY turiniai pridedami prie kaupiklio A turinio, t. y. sudedami aukš-tesnieji 16 bitų operandų baitai ($A + H + CY \rightarrow A, A = 55_{16}$).
200B	67	MOV H, A	Kaupiklio A turinys persiunčiamas į re-gistrą H ($A \rightarrow H, H = 55_{16}$).

Adresas ₁₆	Komandos kodas ₁₆	Komandos mnemonika	Komentaras
200C 200D 200E	22 25 20	SHLD 2025	Gauta 16 bitų operandų suma iš registrų L ir H persiunčiama į atminties ląsteles, iš kurių pirmosios adresas yra antrasis ir trečiasis komandos baitai ($L \rightarrow M[ADR]$, $H \rightarrow M[ADR + 1]$, $L \rightarrow M[2025_{16}]$, $H \rightarrow M[2026_{16}]$, $2025_{16} = 66_{16}$, $2026_{16} = 55_{16}$).
200F	76		
		HLT	Stabdomas mikroprocesoriaus darbas.

Sprendžiant šią užduotį, 16 bitų operandams sudėti galima pritaikyti komandą **DAD**, kuri prie registrų poros **HL** turinio prideda nurodytos registrų poros turinį ir gautas sudėties rezultatas įrašomas į registrų porą **HL**. Šiuo atveju programa žymiai sutrumpėja, kadangi nereikia atskirai sudėti žemesniųjų ir aukštesniųjų 16 bitų operandų baitų.

Adresas ₁₆	Komandos kodas ₁₆	Komandos mnemonika	Komentaras
2000 2001 2002	11 22 33	LXI D, 3322	Į registrų porą DE persiunčiamas dviejų baitų tiesioginis operandas (antrasis ir trečiasis komandos baitai), t. y. pirmasis 16 bitų operandas ($22_{16} \rightarrow E$, $33_{16} \rightarrow D$, $DE = 3322_{16}$).
2003 2004 2005	2A 23 20		
		LHLD 2023	Antrasis 16 bitų operandas iš atminties ląstelių, iš kurių pirmosios adresas yra antrasis ir trečiasis komandos baitai, persiunčiamas į registrus L ir H ($M[ADR] \rightarrow L$, $M[ADR + 1] \rightarrow H$, $M[2023_{16}] \rightarrow L$, $M[2024_{16}] \rightarrow H$, $L = 44_{16}$, $H = 22_{16}$).
2006	19	DAD D	Registrų poros DE turinys pridedamas prie registrų poros HL turinio, t. y. sudedami du 16 bitų operandai ($HL + DE \rightarrow HL$, $HL = 5566_{16}$).

Adresas ₁₆	Komandos kodas ₁₆	Komandos mnemonika	Komentaras
2007	22	SHLD 2025	Gauta 16 bitų operandų suma iš registrų L ir H persiunčiama į atminties ląsteles, iš kurių pirmosios adresas yra antrasis ir trečiasis komandos baitai. ($L \rightarrow M[ADR]$, $H \rightarrow M[ADR + 1]$, $L \rightarrow M[2025_{16}]$, $H \rightarrow M[2026_{16}]$, $2025_{16} = 66_{16}$, $2026_{16} = 55_{16}$).
2008	25		
2009	20		
200A	76	HLT	Stabdomas mikroprocesoriaus darbas.

- 30.** Parašykite programą, kuri atimtų du 16 bitų operandus, esančius atmintyje, ir gautą rezultatą įrašytų atgal į atmintį.
- Operandų baitams adresuoti taikyti šalutinį registrinį operandų adresavimo būdą;
 - Operandų baitams adresuoti taikyti tiesioginį operandų adresavimo būdą;
 - Operandų baitams adresuoti taikyti tiesioginį operandą;
 - Operandų baitams adresuoti parinkti skirtingus operandų adresavimo būdus.
- 31.** Parašykite programą, kuri sudėtų du 16 bitų operandus, esančius atmintyje, ir gautą rezultatą įrašytų į dėklą nurodytu adresu.
- Operandų baitams adresuoti taikyti šalutinį registrinį operandų adresavimo būdą;
 - Operandų baitams adresuoti taikyti tiesioginį operandų adresavimo būdą;
 - Operandų baitams adresuoti taikyti tiesioginį operandą;
 - Operandų baitams adresuoti parinkti skirtingus operandų adresavimo būdus.
- 32.** Parašykite programą, kuri atimtų du 16 bitų operandus, esančius atmintyje, ir gautą rezultatą įrašytų į dėklą nurodytu adresu.

- a. Operandų baitams adresuoti taikyti šalutinį registrinį operandų adresavimo būdą;
- b. Operandų baitams adresuoti taikyti tiesioginį operandų adresavimo būdą;
- c. Operandų baitams adresuoti taikyti tiesioginį operandą;
- d. Operandų baitams adresuoti parinkti skirtingus operandų adresavimo būdus.

32. b. Sprendimas.

Tarkime, kad pirmasis 16 bitų operandas $77A_{16}$ yra gretimose atminties ląstelėse 2060_{16} ir 2061_{16} , t. y. $2060_{16} - AA_{16}$, o $2061_{16} - 77_{16}$. Antrasis 16 bitų operandas 1188_{16} irgi yra gretimose atminties ląstelėse 2062_{16} ir 2063_{16} , t. y. $2062_{16} - 88_{16}$, o $2063_{16} - 11_{16}$.

Gautą atimties rezultatą (skirtumą) $77A_{16} - 1188_{16} = 6622_{16}$ įrašysime į dėklą adresu 2092_{16} .

Adresas ₁₆	Komandos kodas ₁₆	Komandos mnemonika	Komentaras
2000	31	LXI SP, 2092	Į dėklo rodyklę SP persiunčiamas dviejų baitų tiesioginis operandas (antrasis ir trečiasis komandos baitai), t. y. nustatoma dėklo viršūnė ($2092_{16} \rightarrow SP$, $SP = 2092_{16}$).
2001	92		
2002	20		
2003	2A	LHLD 2060	Pirmasis 16 bitų operandas iš atminties ląstelių, iš kurių pirmosios adresas yra antrasis ir trečiasis komandos baitai, persiunčiamas į registrus L ir H ($M[ADR] \rightarrow L$, $M[ADR + 1] \rightarrow H$, $M[2060_{16}] \rightarrow L$, $M[2061_{16}] \rightarrow H$, $L = AA_{16}$, $H = 77_{16}$).
2004	60		
2005	20		
2006	EB	XCHG	Registrai H ir L turiniai sukeičiami su registrai D ir E turiniais ($H \leftrightarrow D$, $L \leftrightarrow E$, $D = 77_{16}$, $E = AA_{16}$).

Adresas ₁₆	Komandos kodas ₁₆	Komandos mnemonika	Komentaras
2007 2008 2009	2A 62 20	LHLD 2062	Antrasis 16 bitų operandas iš atminties ląstelių, iš kurių pirmosios adresas yra antrasis ir trečiasis komandos baitai, persiunčiamas į registrus L ir H ($M[ADR] \rightarrow L, M[ADR + 1] \rightarrow H, M[2062_{16}] \rightarrow L, M[2063_{16}] \rightarrow H, L = 88_{16}, H = 11_{16}$).
200A	7B	MOV A, E	Registro E turinys persiunčiamas į kaupiklį A ($E \rightarrow A, A = A_{16}$).
200B	95	SUB L	Registro L turinys atimamas iš kaupiklio A turinio, t. y. atimami žemesnieji 16 bitų operandų baitai ($A - L \rightarrow A, A = 22_{16}$).
200C	6F	MOV L, A	Kaupiklio A turinys persiunčiamas į registrą L ($A \rightarrow L, L = 22_{16}$).
200D	7A	MOV A, D	Registro D turinys persiunčiamas į kaupiklį A ($D \rightarrow A, A = 77_{16}$).
200E	8C	SBB H	Registro H turinys kartu su požymių registro F paskolos skilties CY turiniu atimami iš kaupiklio A turinio, t. y. atimami aukštesnieji 16 bitų operandų baitai ($A - (H + CY) \rightarrow A, A = 66_{16}$).
200F	67	MOV H, A	Kaupiklio A turinys persiunčiamas į registrą H ($A \rightarrow H, H = 66_{16}$).
2010	E5	PUSH H	Registų poroje HL gautas dviejų 16 bitų operandų skirtumas persiunčiamas į dėklą nurodytu adresu ($H \rightarrow M[SP - 1], L \rightarrow M[SP - 2], SP - 2 \rightarrow SP, H \rightarrow M[2091_{16}], L \rightarrow M[2090_{16}], 2091_{16} = 66_{16}, 2090_{16} = 22_{16}, SP = 2090_{16}$).
2011	76	HLT	Stabdomas mikroprocesoriaus darbas.

- 33.** Parašykite programą, kuri persiųstų 24 bitų operandą į pasirinktas atminties ląsteles. Operando baitams adresuoti taikyti skirtingus adresavimo būdus.
- 34.** Parašykite programą, kuri sukeistų vietomis du 24 bitų operandus, esančius atmintyje. Operandų baitams adresuoti taikyti skirtingus adresavimo būdus.
- 35.** Parašykite programą, kuri sudėtų du 24 bitų operandus, esančius atmintyje, ir gautą rezultatą įrašytų atgal į atmintį. Operandų baitams adresuoti taikyti skirtingus adresavimo būdus.
- 36.** Parašykite programą, kuri atimtų du 24 bitų operandus, esančius atmintyje, ir gautą rezultatą įrašytų atgal į atmintį. Operandų baitams adresuoti taikyti skirtingus adresavimo būdus.
- 37.** Parašykite programą, kuri sudėtų du 24 bitų operandus, esančius atmintyje, ir gautą rezultatą įrašytų į dėklą nurodytu adresu. Operandų baitams adresuoti taikyti skirtingus adresavimo būdus.
- 38.** Parašykite programą, kuri atimtų du 24 bitų operandus, esančius atmintyje, ir gautą rezultatą įrašytų į dėklą nurodytu adresu. Operandų baitams adresuoti taikyti skirtingus adresavimo būdus.
- 39.** Parašykite programą, kuri iš atminties ląstelių, pradedant adresu 202D₁₆, iš eilės persiųstų šešis baitus į mikroprocesoriaus bendrosios paskirties registrus pradedant registru **B**.
- a. Baitams persiųsti taikyti tik **LDA ADR** komandą;
 - b. Baitams persiųsti taikyti tik **LHLD ADR** komandą;
 - c. Baitams persiųsti taikyti tik **MOV R, M** komandą;
 - d. Baitams persiųsti taikyti tik **LDAX RP** komandą.
- 40.** Parašykite programą, kuri iš atminties ląstelių, pradedant adresu 2019₁₆, iš eilės persiųstų šešis baitus į atminties ląsteles pradedant adresu 202F₁₆.
- a. Baitams persiųsti taikyti tik **LDA ADR** ir **STA ADR** komandas;

- b.* Baitams persiųsti taikyti tik **LHLD ADR** ir **SHLD ADR** komandas;
- c.* Baitams persiųsti taikyti tik **MOV R, M** ir **MOV M, R** komandas;
- d.* Baitams persiųsti taikyti tik **LDAX RP** ir **STAX RP** komandas.

1.3. Mikroprocesoriaus Intel® 8080 komandų sistema ir programavimas

- 41.** Parašykite programos fragmentą, kuris:
- Įrašytą konstantą $A0_{16}$ į atminties ląstelę, kurios adresas yra $202C_{16}$;
 - Įrašytą atminties ląstelės, kurios adresas yra $220F_{16}$, turinį į registrą **D**;
 - Įrašytą registro **E** turinį į atminties ląstelę, kurios adresas yra 2456_{16} ;
 - Sukeistų vietomis dviejų atminties ląstelių, kurių adresai yra 2102_{16} ir $21F2_{16}$, turinius;
 - Sukeistų registro **B** ir atminties ląstelės, kurios adresas yra registrų poroje **D** (registrai **D** ir **E**), turinius;
 - Įrašytą registrų poros **B** (registrai **B** ir **C**) turinį į atminties ląstelę, kurių adresai yra $2F00_{16}$ ir $2F01_{16}$.
 - Įrašytą atminties ląstelių, kurių adresai yra $2E00_{16}$ ir $2E01_{16}$, turinius į registrų porą **D** (registrai **D** ir **E**);
 - Įrašytą nulį į registrus **A**, **B**, **C**, **D** ir **E**.
- 42.** Išnagrinėkite duotą programos fragmentą ir nustatykite, kokie bus nurodytų registrų turiniai įvykdžius duotą programos fragmentą.
a. MVI A, 2D	b. MVI B, 51	c. MVI A, 14
ADI 0E	MOV A, B	SUI 2D
Kaupiklis A = ?	SUI 2A	Kaupiklis A = ?
	Kaupiklis A = ?	

d. **LXI B, B0FF****MOV A, B****ADD C**Kaupiklis **A** = ?Požymis **CY** = ?e. **LXI B, 01FF****MOV A, C****ADI 01****MOV L, A****ADD C**Registras **L** = ?f. **LXI H, F001****LXI B, 0F88****DAD B**Registras **H** = ?Registras **L** = ?42. c. *Sprendimas.*

Komanda **MVI A, 14** persiunčia 8 bitų tiesioginį operandą 14_{16} į kaupiklį **A**. Komanda **SUI 2D** atima 8 bitų tiesioginį operandą $2D_{16}$ iš kaupiklio **A** turinio, t. y. iš skaičiaus 14_{16} atimamas didesnis skaičius $2D_{16}$. Mikroprocesorius atliks tokias operacijas:

$$\begin{array}{r}
 00010100_2 = +14_{16} \\
 - \quad 00101101_2 = +2D_{16} \quad \text{atėminys užrašomas papildomuoju kodu:} \\
 \hline
 11010010_2 \quad \text{inversija} \\
 + \quad 1 \\
 \hline
 11010011 = -D3_{16} \\
 + \quad 00010100_2 = +14_{16} \\
 \hline
 11100111_2 = -E7_{16}, \text{ taigi kaupiklis } A = E7_{16}.
 \end{array}$$

43. Išnagrinėkite duotą programos fragmentą ir nustatykite, kokie bus nurodytų registrų turiniai įvykdžius duotą programos fragmentą. Pradiniai duomenys prieš operaciją yra šie: **A** = $2F_{16}$, **B** = $D0_{16}$, **C** = $4A_{16}$, **D** = $0A_{16}$, **E** = $F2_{16}$, **H** = CE_{16} , **L** = $E1_{16}$, **M** = $0B_{16}$.

a. **ADD B****SUB C****ADD D**Kaupiklis **A** = ?b. **ADD M****MOV C, A****MVI B, 00****DAD B**Registras **H** = ?Registras **L** = ?c. **ADD E****MOV E, A****MOV A, D****ADI 01****MOV D, A**Registras **D** = ?Registras **E** = ?

d. **DAD B**
MOV A, E
ADD D

Kaupiklis **A** = ?
 Registras **H** = ?
 Registras **L** = ?

e. **MOV A, C**
ADI FF
INR B

MOV C, A
 Registras **B** = ?
 Registras **C** = ?

f. **ADD C**
MOV L, A
INX H

Registras **H** = ?
 Registras **L** = ?

44. Išnagrinėkite duotą programos fragmentą ir nustatykite, kokie bus registų **A**, **B** ir **C** turiniai įvykdžius duotą programos fragmentą.

a. **MVI B, 0A**
MVI C, 05
MVI A, 0C
STA 2300
LXI H, 2302
MOV M, C
MOV A, B
LDA 2302

b. **MVI C, 08**
MVI B, 0B
MVI A, 33
STA 2440
LXI B, 2440
MOV A, B
MOV A, C
LDAX B

c. **MVI A, 0F**
MVI C, 01
LXI B, FA02
MOV A, C
ADD B
STA 2600
CMA

d. **LXI B, 1234**
LXI H, 2500
MVI M, 07
LDA 2500
INR B
ADD C
MOV B, A

e. **LXI B, 0202**
LXI H, 2110
MVI M, 07
LDA 2110
ANA B
MOV C, A
DCR A

f. **MVI A, 1D**
MVI B, 0B
MVI C, 04
SUB C
MOV B, A
STA 2050
LXI H, 204F
INX H
MOV C, A
MOV A, M

45. Atminties ląstelių, kurių adresai yra 2100_{16} , 2101_{16} , 2102_{16} ir 2103_{16} , turiniai atitinkamai yra lygūs 00_{16} , 11_{16} , 22_{16} ir 33_{16} . Išnagrinėkite duotą programos fragmentą ir nustatykite, kokie bus minėtų atminties ląstelių turiniai įvykdžius duotą programos fragmentą.

a. LHL D 2102
LDA 2100
MOV B, L
MOV C, H
MVI D, 02
MOV E, A
SHLD 2100
STA 2102	b. LXI H, 2100
MOV A, M
STA 2103
ADD A
INR L
MOV B, M
MOV A, B
STA 2102	c. LXI H, 2100
LXI B, 2101
LXI D, 2102
MOV H, L
MOV B, C
MOV D, E
SHLD 2100
SHLD 2102
d. LHL D 2100
XCHG
LHL D 2102
SHLD 2100
XCHG
SHLD 2102	e. LHL D 2102
XCHG
LHL D 2100
SHLD 2100
XCHG
SHLD 2102	f. LDA 2100
STA 2103
LDA 2101
STA 2102
LDA 2102
STA 2101
LDA 2103
STA 2100

46. Išnagrinėkite duotą programos fragmentą ir nustatykite, kokie bus registrų **A**, **H**, **L** ir atminties ląstelių, kurių adresai yra 2400_{16} , 2401_{16} , 2500_{16} ir 2501_{16} , turiniai įvykdžius duotą programos fragmentą.

a. LXI H, 2400
MVI A, 08
MOV M, A
INX H
SUI 02
MOV M, A
LHL D 2400
SHLD 2500
RLC
RLC
RLC
ANI 98 | b. LXI H, 2400
MVI A, 02
MOV M, A
INR L
ADI 0A
MOV M, A
LHL D 2400
SHLD 2500
RLC
RLC
RLC
ANA L | c. LXI H, 2400
MVI A, 07
MOV M, A
INX H
ADI 01
MOV M, A
LHL D 2400
SHLD 2500
RRC
RRC
ORA H |
|--|---|---|

- 47.** Parašykite programą, kuri padidintų arba sumažintų pasirinktą kiekį kartų mikroprocesoriaus kaupiklio turinį, ir gautą rezultatą įrašytų:
- Į pasirinktą atminties ląstelę;
 - Į dėklą nurodytu adresu.
- 48.** Parašykite programą, kuri sudėtų du 16 bitų operandus, esančius dėkle, ir gautą rezultatą įrašytų atgal į dėklą.
- Operandai yra gretimose dėklo atminties ląstelėse;
 - Operandai yra nutolusiose viena nuo kitos dėklo atminties ląstelėse.
- 49.** Parašykite programą, kuri atimtų du 16 bitų operandus, esančius dėkle, ir gautą rezultatą įrašytų atgal į dėklą.
- Operandai yra gretimose dėklo atminties ląstelėse;
 - Operandai yra nutolusiose viena nuo kitos dėklo atminties ląstelėse.
- 50.** Parašykite programą, kuri sukeistų vietomis du 16 bitų operandus, esančius dėkle.
- Operandai yra gretimose dėklo atminties ląstelėse;
 - Operandai yra nutolusiose viena nuo kitos dėklo atminties ląstelėse.
- 51.** Parašykite programą, kuri sudėtų du 24 bitų operandus, esančius dėkle, ir gautą rezultatą įrašytų atgal į dėklą.
- Operandai yra gretimose dėklo atminties ląstelėse;
 - Operandai yra nutolusiose viena nuo kitos dėklo atminties ląstelėse.
- 52.** Parašykite programą, kuri atimtų du 24 bitų operandus, esančius dėkle, ir gautą rezultatą įrašytų atgal į dėklą.
- Operandai yra gretimose dėklo atminties ląstelėse;
 - Operandai yra nutolusiose viena nuo kitos dėklo atminties ląstelėse.

- 53.** Parašykite programą, kuri sukeistų vietomis du 24 bitų operandus, esančius dėkle.
- Operandai yra gretimose dėklo atminties ląstelėse;
 - Operandai yra nutolusiose viena nuo kitos dėklo atminties ląstelėse.
- 54.** Parašykite programą, kuri, nenaudodama kitų mikroprocesoriaus registų, sukeistų registų poros **B** (registrai **B** ir **C**) turinį su registų poros **D** (registrai **D** ir **E**) turiniu.
- 55.** Parašykite programą, kuri požymių registro **F** turinį persiustų į registrą **L**.
- 56.** Į atminties ląsteles, kurių adresai yra 2010_{16} , 2011_{16} ir 2012_{16} , reikia įrašyti 8 bitų operandą x . Parašykite skirtingas programas šiai užduočiai įvykdyti. Išrinkite trumpiausią, t. y. mažiausiai atminties ląstelių užimančią programą.
- 57.** Parašykite programą, kuri 8 bitų operandą x , esantį atminties ląstelėje $200A_{16}$, užrašytų papildomuoju kodu ir gautą rezultatą įrašytų:
- Į pasirinktą atminties ląstelę;
 - Į dėklą nurodytu adresu.
- 58.** Parašykite programą, kuri 16 bitų operandą x , esantį atminties ląstelėje $200E_{16}$ ir $200F_{16}$, užrašytų papildomuoju kodu ir gautą rezultatą įrašytų:
- Į pasirinktą atminties ląstelę;
 - Į dėklą nurodytu adresu.
- 59.** Parašykite programą, kuri invertuotų visas požymių registro **F** skilčių reikšmes.
- 60.** Du 8 bitų operandai x ir y yra atminties ląstelėse, kurių adresai yra 2400_{16} ir 2401_{16} . Parašykite programą, kuri apskaičiuotų z_1 ir z_2 , ir gautą rezultatą įrašytų į atminties ląsteles, kurių adresai yra 2402_{16} ir 2403_{16} .

$$z_1 = x + y - 2;$$

$$z_2 = x - y + 10.$$

60. Sprendimas.

Pirmiausia visus operandus pervesime į šešioliktinę skaičiavimo sistemą: $2_{10} = 2_{16}$, $10_{10} = 0A_{16}$.

Adresas ₁₆	Komandos kodas ₁₆	Komandos mnemonika	Komentaras
2000	21	LXI H, 2400	Į registrų porą HL persiunčiamas dviejų baitų tiesioginis operandas (antrasis ir trečiasis komandos baitai), t. y. pirmojo operando x adresas atmintyje ($00_{16} \rightarrow L$, $24_{16} \rightarrow H$, $HL = 2400_{16}$).
2001	00		
2002	24		
2003	4E	MOV C, M	Pirmasis operandas x iš atminties ląstelės, kurios adresas yra registrų poroje HL, persiunčiamas į registrą C ($M[HL] \rightarrow C$, $M[2400_{16}] \rightarrow C$, $C = x$).
2004	2C	INR L	Registro L turinys didinamas vienetu ($L + 1 \rightarrow L$, $L = 01_{16}$), t. y. registrų poroje HL suformuojamas antrojo operando y adresas atmintyje 2401_{16} .
2005	56	MOV D, M	Antrasis operandas y iš atminties ląstelės, kurios adresas yra registrų poroje HL, persiunčiamas į registrą D ($M[HL] \rightarrow D$, $M[2401_{16}] \rightarrow D$, $D = y$).
2006	79	MOV A, C	Registro C turinys, t. y. operandas x , persiunčiamas į kaupiklį A ($C \rightarrow A$, $A = x$).
2007	82	ADD D	Registro D turinys pridedamas prie kaupiklio A turinio, t. y. sudedami operandai x ir y ($A + D \rightarrow A$, $(x + y) \rightarrow A$).

Adresas ₁₆	Komandos kodas ₁₆	Komandos mnemonika	Komentaras
2008 2009	D6 02	SUI 02	Vieno baido tiesioginis operandas (antrasis komandos baitas) atimamas iš kaupiklio A turinio ($A - 02 \rightarrow A$), t. y. $(x + y - 2) \rightarrow A$, $A = z_1$.
200A	2C	INR L	Registro L turinys didinamas vienetu ($L + 1 \rightarrow L$, $L = 02_{16}$), t. y. registrų poroje HL suformuojamas atminties ląstelės adresas 2402 ₁₆ .
200B	77	MOV M, A	Gautas rezultatas z_1 iš kaupiklio A persiunčiamas į atminties ląstelę, kurios adresas yra registrų poroje HL ($A \rightarrow M[HL]$, $A \rightarrow M[2402_{16}]$), t. y. $2402_{16} = z_1$.
200C	79	MOV A, C	Registro C turinys, t. y. operandas x , persiunčiamas į kaupiklį A ($C \rightarrow A$, $A = x$).
200D	92	SUB D	Registro D turinys atimamas iš kaupiklio A turinio, t. y. iš operando x atimamas operandas y ($A - D \rightarrow A$, $(x - y) \rightarrow A$).
200E	C6	ADI 0A	Vieno baido tiesioginis operandas (antrasis komandos baitas) pridedamas prie kaupiklio A turinio ($A + 0A \rightarrow A$), t. y. $(x - y + 10) \rightarrow A$, $A = z_2$.
200F	2C	INR L	Registro L turinys didinamas vienetu ($L + 1 \rightarrow L$, $L = 03_{16}$), t. y. registrų poroje HL suformuojamas atminties ląstelės adresas 2403 ₁₆ .

Adresas ₁₆	Komandos kodas ₁₆	Komandos mnemonika	Komentaras
2010	77	MOV M, A	Gautas rezultatas z_2 iš kaupiklio A persiunčiamas į atminties ląstelę, kurios adresas yra registrų poroje HL ($A \rightarrow M[HL]$, $A \rightarrow M[2403_{16}]$), t. y. $2403_{16} = z_2$.
2011	76	HLT	Stabdomas mikroprocesoriaus darbas.

- 61.** Penki 8 bitų operandai x_1, x_2, x_3, x_4 ir x_5 yra atminties ląstelėse, kurių adresai atitinkamai yra $2201_{16}, 2202_{16}, 2203_{16}, 2204_{16}$ ir 2205_{16} . Parašykite programą, kuri atliktų tokius veiksmus:
- $$2201_{16} \leftarrow x_1 + x_5$$
- $$2202_{16} \leftarrow x_5 - x_2$$
- $$2203_{16} \leftarrow x_4 + x_1 + x_3$$
- $$2204_{16} \leftarrow x_4 - x_1 - x_2$$
- $$2205_{16} \leftarrow x_1 + x_2 + x_3 + x_4$$
- 62.** Parašykite programą, kuri 8 bitų operandą, esantį registre **L**, persuntų per keturias skiltis į kairę.
- 63.** Parašykite programą, kuri 16 bitų operandą, esantį registrų poroje **D** (registrai **D** ir **E**), perstumtų per vieną skiltį į dešinę.
- 64.** Parašykite programą, kuri 16 bitų operandą, esantį registrų poroje **H** (registrai **H** ir **L**), perstumtų per vieną skiltį į kairę.
- 65.** Parašykite programą, kuri suglaudintų du neglaudintus dvejetainius-dešimtainius skaičius, esančius atminties ląstelėse, kurių adresai yra 2200_{16} ir 2201_{16} , ir gautą rezultatą įrašytų į pasirinktą atminties ląstelę. Priimti, kad žemesnysis skaičius yra atminties ląstelėje, kurios adresas yra 2200_{16} .

65. Sprendimas.

Tarkime, kad žemesnysis neglaudintas dvejetainis-dešimtainis skaičius 03 yra atminties ląstelėje, kurios adresas yra 2200_{16} , aukštesnysis neglaudintas dvejetainis-dešimtainis skaičius yra 07 – 2201_{16} . Gautą rezultatą, t. y. suglaudintą dvejetainį-dešimtainį skaičių, įrašysime į atminties ląstelę, kurios adresas yra 2220_{16} .

Adresas ₁₆	Komandos kodas ₁₆	Komandos mnemonika	Komentaras
2000 2001 2002	3A 01 22	LDA 2201	Aukštesnysis neglaudintas dvejetainis-dešimtainis skaičius iš atminties ląstelės, kurios adresas – antrasis ir trečiasis komandos baitai, persiunčiamas į kaupiklį A ($M[ADR] \rightarrow A$, $M[2201_{16}] \rightarrow A$, $A = 07_{16} = 00000111_2$).
2003	07	RLC	Kaupiklio A turinys perstumiamas per vieną skiltį į kairę ($A = 00001110_2$).
2004	07	RLC	Kaupiklio A turinys perstumiamas per vieną skiltį į kairę ($A = 00011100_2$).
2005	07	RLC	Kaupiklio A turinys perstumiamas per vieną skiltį į kairę ($A = 00111000_2$).
2006	07	RLC	Kaupiklio A turinys perstumiamas per vieną skiltį į kairę ($A = 01110000_2 = 70_{16}$).
2007	4F	MOV C, A	Kaupiklio A turinys persiunčiamas į registrą C ($A \rightarrow C$, $C = 70_{16}$).
2008 2009 200A	3A 00 22	LDA 2200	Žemesnysis neglaudintas dvejetainis-dešimtainis skaičius iš atminties ląstelės, kurios adresas – antrasis ir trečiasis komandos baitai, persiunčiamas į kaupiklį A ($M[ADR] \rightarrow A$, $M[2200_{16}] \rightarrow A$, $A = 03_{16} = 00000011_2$).
200B	81	ADD C	Registro C turinys pridamas prie kaupiklio A turinio ($A + C \rightarrow A$, $A = 73_{16}$).

Adresas ₁₆	Komandos kodas ₁₆	Komandos mnemonika	Komentaras
200C	32	STA 2220	Gautas suglaudintas dvejetainis-dešimtainis skaičius iš kaupiklio A persiunčiamas į atminties ląstelę, kurios adresas – antrasis ir trečiasis komandos baitai ($A \rightarrow M[ADR]$, $A \rightarrow M[2220_{16}]$), t. y. $2220_{16} = 73$.
200D	20		
200E	22		
200F	76	HLT	Stabdomas mikroprocesoriaus darbas.

66. Parašykite programą, kuri išglaudintų glaudintą dvejetainį-dešimtainį skaičių, esantį atminties ląstelėje, kurios adresas yra 2330_{16} , ir gautą rezultatą įrašytų į deklą nurodytu adresu.

67. 8 bitų operandas x yra atminties ląstelėje, kurios adresas yra 2020_{16} . Parašykite programą, kuri apskaičiuotų z ir gautą rezultatą įrašytų į atminties ląstelę, kurios adresas yra 2025_{16} . Daugybai ar dalybai iš 2^n (n – poslinkių skaičius) atlikti taikyti ciklinio poslinkio komandas.

a. $z = ((x - 3) \wedge (x - 10)) \vee (x \times 2)$;

b. $z = (x + 5) \wedge (x - 5) \vee (x \times 4)$;

c. $z = (x + 4) \wedge ((x + 15) \vee (x / 2))$;

d. $z = ((x - 2) \vee (x + 6)) \wedge (x / 4)$;

e. $z = ((x / 8) \wedge (x - 13)) + \neg x$;

f. $z = \neg x + ((x \wedge 2) \vee (x \times 8))$.

67. a. *Sprendimas.*

Pirmiausia visus operandus pervesime iš dešimtainės į šešioliktinę skaičiavimo sistemą: $3_{10} = 03_{16}$, $10_{10} = 0A_{16}$.

Adresas ₁₆	Komandos kodas ₁₆	Komandos mnemonika	Komentaras
2000 2001 2002	21 20 20	LXI H, 2020	Į registrų porą HL persiunčiamas dviejų baitų tiesioginis operandas (antrasis ir trečiasis komandos baitai), t. y. operando x adresas atmintyje ($20_{16} \rightarrow L$, $20_{16} \rightarrow H$, $HL = 2020_{16}$).
2003	7E	MOV A, M	Operandas x iš atminties ląstelės, kurios adresas yra registrų poroje HL, persiunčiamas į kaupiklį A ($M[HL] \rightarrow A$, $M[2020_{16}] \rightarrow A$), t. y. $A = x$.
2004 2005	D6 03	SUI 03	Vieno baito tiesioginis operandas (antrasis komandos baitas) atimamas iš kaupiklio A turinio ($A - 03 \rightarrow A$, $(x - 3) \rightarrow A$).
2006	47	MOV B, A	Kaupiklio A turinys, t. y. $(x - 3)$ rezultatas, persiunčiamas į registrą B ($A \rightarrow B$, $B = (x - 3)$).
2007	7E	MOV A, M	Operandas x iš atminties ląstelės, kurios adresas yra registrų poroje HL, persiunčiamas į kaupiklį A ($M[HL] \rightarrow A$, $M[2020_{16}] \rightarrow A$), t. y. $A = x$.
2008 2009	D6 0A	SUI 0A	Vieno baito tiesioginis operandas (antrasis komandos baitas) atimamas iš kaupiklio A turinio ($A - 0A \rightarrow A$, $(x - 10) \rightarrow A$).
200A	A0	ANA B	Registro B turinys, t. y. $(x - 3)$ rezultatas, logiškai sudauginamas su kaupiklio A turiniu, t. y. $(x - 10)$ rezultatu ($A \wedge B \rightarrow A$, $((x - 3) \wedge (x - 10)) \rightarrow A$).
200B	5F	MOV E, A	Kaupiklio A turinys, t. y. $((x - 3) \wedge (x - 10))$ rezultatas, persiunčiamas į registrą E ($A \rightarrow E$, $E = (x - 3) \wedge (x - 10)$).

Adresas ₁₆	Komandos kodas ₁₆	Komandos mnemonika	Komentaras
200C	7E	MOV A, M	Operandas x iš atminties ląstelės, kurios adresas yra registrų poroje HL, persiunčiamas į kaupiklį A ($M[HL] \rightarrow A$, $M[2020_{16}] \rightarrow A$), t. y. $A = x$.
200D	07	RLC	Kaupiklio A turinys, t. y. operandas x , pastumiamas per vieną skiltį į kairę, tai atitinka daugyba iš dviejų ($(x \times 2) \rightarrow A$).
200E	B3	ORA E	Registro E turinys, t. y. $((x - 3) \wedge (x - 10))$ rezultatas, logiškai sudedamas su kaupiklio A turiniu, t. y. $(x \times 2)$ rezultatu ($A \vee E \rightarrow A$), t. y. $((x - 3) \wedge (x - 10)) \vee (x \times 2) \rightarrow A$, $A = z$.
200F 2010 2011	32 25 20	STA 2025	Gautas rezultatas z iš kaupiklio A persiunčiamas į atminties ląstelę, kurios adresas – antrasis ir trečiasis komandos baitai ($A \rightarrow M[ADR]$, $A \rightarrow M[2025_{16}]$), t. y. $2025_{16} = z$.
2012	76	HLT	Stabdomas mikroprocesoriaus darbas.

- 68.** Parašykite programą, kuri apskaičiuotų keturių atmintyje esančių 8 bitų operandų aritmetinį vidurkį ir gautą rezultatą įrašytų į pasirinktą atminties ląstelę. Operandams adresuoti parinkti skirtingus operandų adresavimo būdus.
- 69.** Du 8 bitų operandai x ir y yra registruose **B** ir **C**. Parašykite programą, kuri apskaičiuotų z ir gautą rezultatą įrašytų į registrą **D**. Daugybai ar dalybai iš 2^n (n – poslinkių skaičius) atlikti taikyti ciklinio poslinkio komandas.
- $z = (x \times 8) \wedge (y / 16)$;
 - $z = (x / 4) \vee (y \times 2)$;
 - $z = (x \times 8) \vee (y / 16)$;

$$d. z = (x / 16) \vee (y / 4).$$

- 70.** Trys 8 bitų operandai x_1 , x_2 ir x_3 yra atminties ląstelėse, kurių adresai atitinkamai yra 2100_{16} , 2101_{16} ir 2102_{16} . Parašykite programą, kuri apskaičiuotų z_1 , z_2 ir z_3 ir gautą rezultatą įrašytų į atminties ląsteles, kurių adresai yra 2201_{16} , 2202_{16} ir 2203_{16} .

$$z_1 = (x_1 \wedge x_2) \vee x_3;$$

$$z_2 = (x_1 + x_2) \vee x_2;$$

$$z_3 = (x_3 - x_1) \vee x_2.$$

- 71.** Parašykite programą nurodytam reiškiniui apskaičiuoti. Daugybai ar dalybai iš 2^n (n – poslinkių skaičius) atlikti taikyti ciklinio poslinkio komandas.

$$a. y = (47 - 30 / 2) / 2 + 13;$$

$$b. y = (36 + 32 / 4) / 2 - 10;$$

$$c. y = (60 + 132 / 2) / 8 - 6;$$

$$d. y = (24 \times 4 - 8 \times 2) / 4;$$

$$e. y = (200 / 8 - 1) / 4;$$

$$f. y = 15 \times 8 - (140 / 2 + 11).$$

71. a. Sprendimas.

Pirmiausia visus operandus pervesime iš dešimtainės į šešioliktainę skaičiavimo sistemą. Kadangi užduotyje taikomi didesni dešimtainiai skaičiai, jiems perversi į šešioliktainę skaičiavimo sistemą taikysime universalią taisyklę, t. y. dešimtainės skaičiavimo sistemos skaičių nuosekliai dalydami iš kitos skaičiavimo sistemos pagrindo, gausime liekanas, kurias surašę atvirkščia tvarka (iš eilės nuo pirmos iki paskutinės) gausime skaičių kitoje skaičiavimo sistemoje. Taigi perveskime pirmąjį operandą 47_{10} į šešioliktainę skaičiavimo sistemą:

$$\begin{array}{ll} 47_{10} / 16 = 2_{10} & \text{liekana } 15_{10} = F_{16} \\ 2_{10} / 16 = 0_{10} & \text{liekana } 2_{10} = 2_{16} \end{array} \quad \begin{array}{l} \text{—} \\ \text{—} \end{array}$$

$$47_{10} = 2 F_{16}$$

Kiti operandai pervedami taip pat.

Adresas ₁₆	Komandos kodas ₁₆	Komandos mnemonika	Komentaras
2000 2001	3E 1E	MVI A, 1E	Į kaupiklį A persiunčiamas vieno baido tiesioginis operandas (antrasis komandos baitas), t. y. antrasis operandas $30_{10} = 1E_{16}$ ($1E_{16} \rightarrow A, A = 1E_{16}$).
2002	0F	RRC	Kaupiklio A turinys, t. y. antrasis operandas, pastumiamas per vieną skiltį į dešinę, tai atitinka dalyba iš dviejų ($(1E_{16} / 2 \rightarrow A, A = 0F_{16})$).
2003	47	MOV B, A	Kaupiklio A turinys, t. y. antrasis operandas, persiunčiamas į registrą B ($A \rightarrow B, B = 0F_{16}$).
2004 2005	3E 2F	MVI A, 2F	Į kaupiklį A persiunčiamas vieno baido tiesioginis operandas (antrasis komandos baitas), t. y. pirmasis operandas $47_{10} = 2F_{16}$ ($2F_{16} \rightarrow A, A = 2F_{16}$).
2006	90	SUB B	Registro B turinys atimamas iš kaupiklio A turinio, ($A - B \rightarrow A, A = 20_{16}$).
2007	0F	RRC	Kaupiklio A turinys pastumiamas per vieną skiltį į dešinę, tai atitinka dalyba iš dviejų ($(20_{16} / 2) \rightarrow A, A = 10_{16}$).
2008 2009	C6 0D	ADI 0D	Vieno baido tiesioginis operandas (antrasis komandos baitas), t. y. trečiasis operandas $13_{10} = 0D_{16}$, pridedamas prie kaupiklio A turinio ($A + D8 \rightarrow A, A + 0D_{16} \rightarrow A, A = 1D_{16}$).
200A	76	HLT	Stabdomas mikroprocesoriaus darbas.

- 72.** Parašykite programą, kuri skaičiuotų nurodytą reiškinį ir gautą rezultatą u įrašytų į pasirinktą atminties ląstelę. Operandai x, y, z , ir w yra pasirinktose atminties ląstelėse. Operandams adresuoti taikyti

skirtingus adresavimo būdus. Daugybai ar dalybai iš 2^n (n – poslinkių skaičius) atlikti taikyti ciklinio poslinkio komandas.

a. $u = (x + y + z) - 2w$;

b. $u = (x - y - z) + 4w$;

c. $u = (x + y) / z - w$;

d. $u = x / (y - z) + w$;

e. $u = 4(x + y) + 2z + 4w$;

f. $u = 2(x - y) - 4z + w / 2$;

g. $u = x / 2 + (y - z) / 4 - w$;

h. $u = (x + y) / 2 + z / 4 + w$;

i. $u = (x + y) / 2 + z / 4 + (4w - 2) / 4$;

j. $u = (2x - y) / 4 - z + w / 8$;

k. $u = (x - y) / 8 + 2z - 4w + 3$;

l. $u = 4x - 2(y - z) + w / 2 - 5$;

m. $u = 2(x + y) - 6 + 4(z + w)$;

n. $u = x + 4y + 7 - (z + w) / 2$;

o. $u = (2x + 3 - 4y) / 8 + (w - (x / 2)) / (2 - z)$;

p. $u = 4(x + 5) + (x + y) / (2 - 2w) + z / 4$.

73. Parašykite programą, kuri skaičiuotų 72 užduotyje nurodytą reiškinį ir gautą rezultatą u įrašytų į dėklą nurodytu adresu. Operandai x , y , z , ir w yra pasirinktose atminties ląstelėse. Operandams adresuoti taikyti skirtingus adresavimo būdus. Daugybai ar dalybai iš 2^n (n – poslinkių skaičius) atlikti taikyti ciklinio poslinkio komandas.

74. Du 8 bitų operandai x ir y yra atminties ląstelėse, kurių adresai atitinkamai yra $20F1_{16}$ ir $20F2_{16}$. Parašykite programą, kuri per išvesties prievadą, kurio adresas yra FE_{16} , išvestų didesnįjį iš operandų.

74. *Sprendimas.*

Adresas ₁₆	Komandos kodas ₁₆	Komandos mnemonika	Komentaras
2000 2001 2002	3A F1 20	LDA 20F1	Pirmasis operandas x iš atminties ląstelės, kurios adresas – antrasis ir trečiasis komandos baitai, persiunčiamas į kaupiklį A ($M[ADR] \rightarrow A$, $M[20F1_{16}] \rightarrow A$, $A = x$).
2003	47	MOV B, A	Kaupiklio A turinys, t. y. operandas x , persiunčiamas į registrą B ($A \rightarrow B$, $B = x$).
2004 2005 2006	3A F2 20	LDA 20F2	Antrasis operandas y iš atminties ląstelės, kurios adresas – antrasis ir trečiasis komandos baitai, persiunčiamas į kaupiklį A ($M[ADR] \rightarrow A$, $M[20F2_{16}] \rightarrow A$, $A = y$).
2007	4F	MOV C, A	Kaupiklio A turinys, t. y. operandas y , persiunčiamas į registrą C ($A \rightarrow C$, $C = y$).
2008	90	SUB B	Registro B turinys atimamas iš kaupiklio A turinio, t. y. iš operando y atimamas operandas x ($A - D \rightarrow A$, $(y - x) \rightarrow A$).
2009 200A 200B	F2 10 20	JP 2010	Jei atimties rezultatas (skirtumas) yra teigiamas, t. y. $y - x > 0$ (operandas y yra didesnis už operandą x), tai pereinama ir pradedama vykdyti komanda, kurios adresas yra antrasis ir trečiasis baitai. Jei atimties rezultatas (skirtumas) yra neigiamas, t. y. $y - x < 0$ (operandas x yra didesnis už operandą y), tai pereigos nėra ir vykdoma paskui einanti komanda.
200C	78	MOV A, B	Registro B turinys, t. y. operandas x , persiunčiamas į kaupiklį A ($B \rightarrow A$, $A = x$).
200D 200E	D3 FE	OUT FE	Kaupiklio A turinys, t. y. operandas x , išvedamas per išvesties prievadą, kurio adresas yra antrasis komandos baitas.

Adresas ₁₆	Komandos kodas ₁₆	Komandos mnemonika	Komentaras
200F	76	HLT	Stabdomas mikroprocesoriaus darbas.
2010	79	MOV A, C	Registro C turinys, t. y. operandas y, persiunčiamas į kaupiklį A ($C \rightarrow A, A = y$).
2011 2012	D3 FE	OUT FE	Kaupiklio A turinys, t. y. operandas y, išvedamas per išvesties prievadą, kurio adresas yra antrasis komandos baitas.
2013	76	HLT	Stabdomas mikroprocesoriaus darbas.

- 75.** Pakeiskite 74 užduoties programą taip, kad operandų palyginimas būtų atliekamas taikant palyginimo komandą.
- 76.** Parašykite programą, kuri palygintų dviejų atminties ląstelių turinius ir didesnį iš jų įrašytų į pasirinktą trečiąją atminties ląstelę.
- 77.** Parašykite programą, kuri palygintų dviejų atminties ląstelių turinius ir mažesnį iš jų įrašytų į pasirinktą trečiąją atminties ląstelę.
- 78.** Parašykite programą, kuri palygintų du 8 bitų operandus, esančius pasirinktose skirtingose atminties ląstelėse, iš didesniojo atimtų mažesnį ir gautą skirtumą įrašytų į dėklą nurodytu adresu.
- 79.** Parašykite programą, kuri skaičiuotų nurodytą reiškinį ir gautą rezultatą z įrašytų į atminties ląstelę, kurios adresas yra 2230_{16} . 8 bitų operandai x ir y įvedami atitinkamai per įvesties prievadus, kurių adresai yra 01_{16} ir 02_{16} . Priimti, kad x , y ir z yra sveikieji skaičiai intervale nuo -128 iki 127 . Operandams adresuoti taikyti skirtingus adresavimo būdus. Daugybai ar dalybai iš 2^n (n – poslinkių skaičius) atlikti taikyti ciklinio poslinkio komandas.
- $z = 3x + 1$;
 - $z = 5x - 5$;
 - $z = 6x + 2$;
 - $z = (4x - 8y) / 2$;

$$e. \quad z = \begin{cases} x - y, & \text{kai } x \geq 8, \\ x + y, & \text{kai } x < 8; \end{cases}$$

$$f. \quad z = \begin{cases} 3x + y, & \text{kai } x \geq 10, \\ 5x - y, & \text{kai } x < 10; \end{cases}$$

$$g. \quad z = \begin{cases} x + 2y + 1, & \text{kai } x > 5, \\ 2x - 3y + 3, & \text{kai } x \leq 5; \end{cases}$$

$$h. \quad z = \begin{cases} x + 1, & \text{kai } x \leq 15, \\ x + y, & \text{kai } 15 < x \leq 30. \end{cases}$$

79. a. *Sprendimas.*

Adresas ₁₆	Komandos kodas ₁₆	Komandos mnemonika	Komentaras
2000 2001	DB 01	IN 01	Operandas x per įvesties prievadą, kurio numeris (adresas) nurodytas antrajame komandos baite, įvedamas į kaupiklį A ($\text{PORT} \rightarrow A, 01_{16} \rightarrow A, A = x$).
2002	47	MOV B, A	Kaupiklio A turinys, t. y. operandas x , persiunčiamas į registrą B ($A \rightarrow B, B = x$).
2003	87	ADD A	Kaupiklio A turinys pridamas prie kaupiklio A turinio, t. y. operandas x padidinamas du kartus, tai atitinka daugybą iš dviejų ($A + A \rightarrow A, A = 2x$).
2004	80	ADD B	Registro B turinys pridamas prie kaupiklio A turinio, t. y. operandas x padidinamas tris kartus, tai atitinka daugybą iš trijų ($A + B \rightarrow A, A = 3x$).
2005	3C	INR A	Kaupiklio A turinys didinamas vienetu, t. y. gaunamas rezultatas z ($A + 1 \rightarrow A, (3x + 1) \rightarrow A, A = z$).

Adresas ₁₆	Komandos kodas ₁₆	Komandos mnemonika	Komentaras
2006	32	STA 2230	Gautas rezultatas z iš kaupiklio A persiunčiamas į atminties ląstelę, kurios adresas yra antrasis ir trečiasis komandos baitai ($A \rightarrow M[ADR]$, $A \rightarrow M[2230_{16}]$), t. y. $2230_{16} = z$.
2007	30		
2008	22		
2009	76	HLT	Stabdomas mikroprocesoriaus darbas.

Kaip matyti iš pateikto sprendimo pavyzdžio, operandą, esantį kaupiklyje **A**, galima padauginti iš 2^n , t. y. iš 2, 4, 8 ir t. t., kartojant komandą **ADD A**. Pvz., norint padauginti kaupiklio **A** turinį iš keturių, komandą **ADD A** reikės pakartoti du kartus. Pridedant ar atimant konstantas, ne didesnes kaip du, dviejų baitų komandą **ADI D8** ar **SUI D8** patogiu pakeisti inkrementavimo **INR R** ar dekrementavimo **DCR R** komandomis.

79. e. Sprendimas.

Adresas ₁₆	Komandos kodas ₁₆	Komandos mnemonika	Komentaras
2000	DB	IN 02	Operandas y per įvesties prievadą, kurio numeris (adresas) nurodytas antrajame komandos baite, įvedamas į kaupiklį A ($PORT \rightarrow A$, $02_{16} \rightarrow A$, $A = y$).
2001	02		
2002	47	MOV B, A	Kaupiklio A turinys, t. y. operandas y , persiunčiamas į registrą B ($A \rightarrow B$, $B = y$).
2003	DB	IN 01	Operandas x per įvesties prievadą, kurio numeris (adresas) nurodytas antrajame komandos baite, įvedamas į kaupiklį A ($PORT \rightarrow A$, $01_{16} \rightarrow A$, $A = x$).
2004	01		
2005	4F	MOV C, A	Kaupiklio A turinys, t. y. operandas x , persiunčiamas į registrą C ($A \rightarrow C$, $C = x$).

Adresas ₁₆	Komandos kodas ₁₆	Komandos mnemonika	Komentaras
2006 2007	D6 08	SUI 08	Vieno baito tiesioginis operandas (antrasis komandos baitas) atimamas iš kaupiklio A turinio ($A - D8 \rightarrow A$, $A - 08_{16} \rightarrow A$, $(x - 8) \rightarrow A$).
2008 2009 200A	F2 11 20	JP 2011	Jei atimties rezultatas (skirtumas) yra teigiamas, t. y. $x \geq 8$, tai pereinama ir pradedama vykdyti komanda, kurios adresas yra antrasis ir trečiasis baitai. Jei atimties rezultatas (skirtumas) yra neigiamas, t. y. $x < 8$, tai pereigos nėra ir vykdoma pasikui einanti komanda.
200B	79	MOV A, C	Registro C turinys, t. y. operandas x , persiunčiamas į kaupiklį A ($C \rightarrow A$, $A = x$).
200C	80	ADD B	Registro B turinys pridedamas prie kaupiklio A turinio, t. y. prie operando x pridedamas operandas y ($A + B \rightarrow A$, $(x + y) \rightarrow A$, $A = z$).
200D 200E 200F	32 30 22	STA 2230	Gautas rezultatas z iš kaupiklio A persiunčiamas į atminties ląstelę, kurios adresas – antrasis ir trečiasis komandos baitai ($A \rightarrow M[ADR]$, $A \rightarrow M[2230_{16}]$), t. y. $2230_{16} = z$.
2010	76	HLT	Stabdomas mikroprocesoriaus darbas.
2011	79	MOV A, C	Registro C turinys, t. y. operandas x , persiunčiamas į kaupiklį A ($C \rightarrow A$, $A = x$).
2012	90	SUB B	Registro B turinys atimamas iš kaupiklio A turinio, t. y. iš operando x atimamas operandas y ($A - B \rightarrow A$, $(x - y) \rightarrow A$, $A = z$).

Adresas ₁₆	Komandos kodas ₁₆	Komandos mnemonika	Komentaras
2013	32	STA 2230	Gautas rezultatas z iš kaupiklio A persiunčiamas į atminties ląstelę, kurios adresas yra antrasis ir trečiasis komandos baitai ($A \rightarrow M[ADR]$, $A \rightarrow M[2230_{16}]$), t. y. $2230_{16} = z$.
2014	30		
2015	22		
2016	76	HLT	Stabdomas mikroprocesoriaus darbas.

- 80.** 8 bitų operandas z yra įvedamas per įvesties prievadą, kurio adresas yra $0A_{16}$. Atmintyje, pradedant adresu 2100_{16} , yra funkcijos $f = \cos x$ skaičiavimo paprogramis, čia x – kaupiklio A turinys prieš vykdant paprogramį. Nustatykite ir užrašykite išvesties prievado, kurio adresas yra $0F_{16}$, reikšmę įvykdžius duotas programas.

<p><i>a.</i> IN 0A
ADD A
DCR A
CALL 2100
INR A
OUT 0F
HLT</p>	<p><i>b.</i> IN 0A
ADD A
ADD A
INR A
CALL 2100
RLC
OUT 0F
HLT</p>	<p><i>c.</i> IN 0A
RRC
RRC
ADI 05
CALL 2100
SUI 0B
OUT 0F
HLT</p>
<p><i>d.</i> IN 0A
RLC
RLC
RLC
CALL 2100
ADI 10
OUT 0F
HLT</p>	<p><i>e.</i> IN 0A
ADD A
ADD A
ADD A
CALL 2100
RRC
OUT 0F
HLT</p>	<p><i>f.</i> IN 0A
SUI 03
RRC
CALL 2100
SUI 04
OUT 0F
HLT</p>

- 81.** Parašykite programą, kuri skaičiuotų duotą reiškinį. Atmintyje, pradedant adresu $210A_{16}$, yra funkcijos $f = \sin x$ skaičiavimo paprogramis, čia x – kaupiklio A turinys prieš vykdant paprogramį. 8 bitų

operandas a įvedamas per įvesties prievadą, kurio adresas yra 05_{16} .
Gautą rezultatą z išvesti per išvesties prievadą, kurio adresas yra 06_{16} .

$$z = 1 + \sin a + \sin 2a.$$

- 82.** Atmintyje, pradedant adresu 2050_{16} , yra dviejų operandų, esančių registruose **D** ir **E**, sandaugos skaičiavimo paprogramis. Daugybės rezultatas (sandauga) įrašomas į registrą **E**. Taikant šį paprogramį, parašykite cikliškai veikiančią programą, kuri skaičiuotų reiškinių:

$$z = x + x^2 + x^3 + x^4 + x^5 + x^6,$$

čia x – operandas, įvedamas per įvesties prievadą, kurio adresas yra 15_{16} .

Gautą rezultatą z persiūsti į atminties ląstelę, kurios adresas yra 2300_{16} .

82. Sprendimas.

Kadangi operandą reikės penkis kartus kelti laipsniu, t. y. penkis kartus reikės kreiptis į dviejų operandų sandaugos paprogramį, tai kreipčių į paprogramį skaičių įrašysime į registrą **L**. Tarpinius sudėties rezultatus saugosime registre **C**.

Adresas ₁₆	Komandos kodas ₁₆	Komandos mnemonika	Komentaras
2000	31	LXI SP, 2090	Į dėklo rodyklę SP persiunčiamas dviejų baitų tiesioginis operandas (antrasis ir trečiasis komandos baitai), t. y. nustatoma dėklo viršūnė ($2090_{16} \rightarrow \text{SP}$, $\text{SP} = 2090_{16}$).
2001	90		
2002	20		
2003	DB	IN 15	Operandas x per įvesties prievadą, kurio numeris (adresas) nurodytas antrajame komandos baite, įvedamas į kaupiklį A ($\text{PORT} \rightarrow \text{A}$, $15_{16} \rightarrow \text{A}$, $\text{A} = x$).
2004	15		
2005	57	MOV D, A	Kaupiklio A turinys persiunčiamas į registrą D ($\text{A} \rightarrow \text{D}$, $\text{D} = x$).

Adresas ₁₆	Komandos kodas ₁₆	Komandos mnemonika	Komentaras
2006	5F	MOV E, A	Kaupiklio A turinys persiunčiamas į registrą E ($A \rightarrow E, E = x$).
2007 2008	2E 05	MVI L, 05	Į registrą L persiunčiamas vieno baidtiesioginis operandas (antrasis komandos baitas), t. y. kreipčių į paprogramį skaičius ($05_{16} \rightarrow L, L = 05_{16}$).
2009 200A 200B	CD 50 20	CALL 2050	Besąlygiškai kreipiamasi į paprogramį, kurio pradžios adresą nurodo antrasis ir trečiasis komandos baitai (1 kreiptis: $D \times E \rightarrow E, x \times x \rightarrow E, E = x^2$; 2 kreiptis: $D \times E \rightarrow E, x \times x^2 \rightarrow E, E = x^3$; 3 kreiptis: $D \times E \rightarrow E, x \times x^3 \rightarrow E, E = x^4$; 4 kreiptis: $D \times E \rightarrow E, x \times x^4 \rightarrow E, E = x^5$; 5 kreiptis: $D \times E \rightarrow E, x \times x^5 \rightarrow E, E = x^6$).
200C	83	ADD E	Registro E turinys pridedamas prie kaupiklio A turinio (1 ciklas: $A + E \rightarrow A, (x + x^2) \rightarrow A$; 2 ciklas: $A + E \rightarrow A, (x + x^2 + x^3) \rightarrow A$; 3 ciklas: $A + E \rightarrow A, (x + x^2 + x^3 + x^4) \rightarrow A$; 4 ciklas: $A + E \rightarrow A, (x + x^2 + x^3 + x^4 + x^5) \rightarrow A$; 5 ciklas: $A + E \rightarrow A, (x + x^2 + x^3 + x^4 + x^5 + x^6) \rightarrow A, A = z$).
200D	4F	MOV C, A	Kaupiklio A turinys persiunčiamas į registrą C (1 ciklas: $A \rightarrow C, C = x + x^2$; 2 ciklas: $A \rightarrow C, C = x + x^2 + x^3$; 3 ciklas: $A \rightarrow C, C = x + x^2 + x^3 + x^4$; 4 ciklas: $A \rightarrow C, C = x + x^2 + x^3 + x^4 + x^5$; 5 ciklas: $A \rightarrow C, C = x + x^2 + x^3 + x^4 + x^5 + x^6, C = z$).
200E	2D	DCR L	Registro L turinys mažinamas vienetu (1 ciklas: $L - 1 \rightarrow L, L = 04_{16}$; 2 ciklas: $L - 1 \rightarrow L, L = 03_{16}$; 3 ciklas: $L - 1 \rightarrow L, L = 02_{16}$; 4 ciklas: $L - 1 \rightarrow L, L = 01_{16}$; 5 ciklas: $L - 1 \rightarrow L, L = 00_{16}$).

Adresas ₁₆	Komandos kodas ₁₆	Komandos mnemonika	Komentaras
200F 2010 2011	C2 09 20	JNZ 2009	Jei dekrementavimo rezultatas nelygus nuliui, tai pereinama ir pradama vykdyti komanda, kurios adresas yra antrasis ir trečiasis baitai. Jei atimties rezultatas (skirtumas) yra teigiamas, tai pereigos nėra ir vykdoma paskui einanti komanda.
2012	79	MOV A, C	Registro C turinys, t. y. gautas rezultatas z , persiunčiamas į kaupiklį A ($C \rightarrow A$, $A = z$).
2013 2014 2015	32 00 23	STA 2300	Gautas rezultatas z iš kaupiklio A persiunčiamas į atminties ląstelę, kurios adresas – antrasis ir trečiasis komandos baitai ($A \rightarrow M[ADR]$, $A \rightarrow M[2300_{16}]$), t. y. $2300_{16} = z$.
2016	76	HLT	Stabdomas mikroprocesoriaus darbas.

- 83.** Asemblerio kalba parašykite programą, kuri skaitytų 8 bitų operandą x iš įvesties prievado, kurio adresas yra FF_{16} . Jei iš įvesties prievado nuskaitytas operandas x yra lygus 200_{10} , kreiptis į paprogramį, kuris operandą x inkrementuoja du kartus ir gautą rezultatą įrašo į registrą **D**.
- 84.** Asemblerio kalba parašykite cikliškai veikiančios programos, kuri vykdytų nurodytas užduotis, fragmentą.
1. Persiųsti tiesioginį operandą į mikroprocesoriaus registrą;
 2. Dekrementuoti registro turinį;
 3. Kartoti 2 užduotį tol, kol gautas rezultatas nelygus nuliui.
- Tiesioginis operandas yra lygus:
- a. 110_{10} ;
 - b. 1010_{10} .

Pastaba: variante *a* taikyti mikroprocesoriaus registrą **D**, *b* – registrų porą **D** (registrai **D** ir **E**).

84. a. *Sprendimas.*

Tiesioginį operandą $110_{10} = 6E_{16}$ persiūsiame į registrą **D**.

Žymė	Komandos mnemonika	Komentaras
	MVI D, 6E	; Į registrą D persiunčiamas vieno baido tiesioginis ; operandas (antrasis komandos baitas), t. y. 8 bitų ; operandas $110_{10} = 6E_{16}$ ($6E_{16} \rightarrow \mathbf{D}$, $\mathbf{D} = 6E_{16}$).
Z1:	DCR D	; Dekrementuojamas registro D turinys ($\mathbf{D} - 1 \rightarrow \mathbf{D}$).
	JNZ Z1	; Jei dekrementavimo rezultatas nelygus nuliui, tai ; pereinama ir pradedama vykdyti komanda, pažymėta ; žyme Z1 . Jei dekrementavimo rezultatas lygus nuliui, ; tai pereigos nėra ir vykdoma paskui einanti komanda.

84. b. *Sprendimas.*

Kadangi tiesioginis operandas $1010_{10} = 03F2_{16}$ yra 16 bitų, jį persiūsiame į registrų porą **D** (registrai **D** ir **E**). Vienas iš galimų programos fragmentų gali būti toks:

Žymė	Komandos mnemonika	Komentaras
	LXI D, 03F2	; Į registrų porą DE persiunčiamas dviejų baitų ; tiesioginis operandas (antrasis ir trečiasis komandos ; baitai), t. y. 16 bitų operandas $1010_{10} = 03F2_{16}$; ($F2_{16} \rightarrow \mathbf{E}$, $03_{16} \rightarrow \mathbf{D}$).
Z1:	DCX D	; Dekrementuojamas registrų poros D (registrai D ir ; E) turinys ($\mathbf{DE} - 1 \rightarrow \mathbf{DE}$).
	MOV A, E	; Registro E turinys persiunčiamas į kaupiklį A ; ($\mathbf{E} \rightarrow \mathbf{A}$, $\mathbf{A} = F1_{16}$).

Žymė	Komandos mnemonika	Komentaras
	ORA D	; Registro D turinys logiškai sudedamas su kaupiklio ; A turiniu ($A \vee D \rightarrow A$).
	JNZ Z1	; Jei loginės sudėties rezultatas nelygus nuliui, tai ; pereinama ir pradedama vykdyti komanda, pažymėta ; žyme Z1. Jei loginės sudėties rezultatas lygus nuliui, ; tai pereinamos nėra ir vykdoma paskui einanti komanda.

Atliekant komandą **DCX**, nėra keičiamos požymių registro **F** skilčių reikšmės, todėl registrų poros **D** (registrai **D** ir **E**) turiniui tikrinti, prieš są-lyginės pereinamos komandą **JNZ**, reikia atlikti registrų **D** ir **E** turinių loginę sudėtį. Akivaizdu, kad $D \vee E = 0$ tik tada, kai $D = E = 0$.

- 85.** Pakeiskite 84 užduoties programą taip, kad ciklas, jei gautas rezulta-tas nelygus nuliui, būtų organizuojamas taikant palyginimo koman-dą.
- 86.** Assemblerio kalba parašykite cikliškai veikiančios programos, kuri vykdytų nurodytas užduotis, fragmentą.
1. Persiųsti tiesioginį operandą į mikroprocesoriaus registrą;
 2. Inkrementuoti registro turinį;
 3. Kartoti 2 užduotį tol, kol gautas rezultatas nelygus nuliui.
- Tiesioginis operandas yra lygus:
- a. 128_{10} ;
 - b. 1024_{10} .
- Pastaba:* variante a taikyti mikroprocesoriaus registrą **B**, b – regis-trų porą **B** (registrai **B** ir **C**).
- 87.** Assemblerio kalba parašykite programą, kuri persiųstų konstantą 05_{16} į mikroprocesoriaus kaupiklį, paskui kaupiklio turinį dekrementuotų iki nulio ir sustotų.

- 88.** Asemblerio kalba parašykite programą, kuri iš eilės užpildytų atminties sritį nuo 2100_{16} iki $21FF_{16}$ adreso konstantomis atitinkamai nuo FF_{16} iki 00_{16} .
- 89.** Asemblerio kalba parašykite programą, kuri nuolat iš eilės užpildytų atminties sritį nuo 2200_{16} iki 2280_{16} adreso, konstantomis, įvedamomis per įvesties prievadą, kurio adresas yra 02_{16} .
- 90.** Asemblerio kalba parašykite programą, kuri nuolat skaitytų 8 bitų operandus iš įvesties prievado, kurio adresas yra 04_{16} . Jei iš įvesties prievado nuskaitytas operandas yra didesnis už 64_{10} , išvesti vieneta į išvesties prievadą, kurio adresas yra 05_{16} . Jei iš įvesties prievado nuskaitytas operandas yra mažesnis už 64_{10} , išvesti nulį į išvesties prievadą, kurio adresas yra 05_{16} . Jei nuskaitytas operandas lygus 64_{10} , iš įvesties prievado skaityti kitą operandą.
- 91.** Atmintyje, pradedant adresu 2300_{16} , yra dvidešimties 8 bitų operandų masyvas. Ieškomasis operandas x yra atminties ląstelėje, kurios adresas yra 2100_{16} . Asemblerio kalba parašykite programą, kuri atliktų operando x paiešką masyve ir surasto operando adresą įrašytų į atminties ląsteles, kurių adresai yra 2500_{16} ir 2501_{16} . Jei ieškomojo operando masyve nėra, į atminties ląsteles, kurių adresai yra 2500_{16} ir 2501_{16} , įrašyti nulius.
- 92.** Atmintyje, pradedant adresu 2400_{16} , yra dvidešimties 8 bitų operandų masyvas. Asemblerio kalba parašykite programą, kuri atliktų didžiausio operando paiešką masyve ir surastą didžiausią operandą įrašytų į atminties ląstelę, kurios adresas yra 2100_{16} . Priimti, kad visi operandai masyve yra teigiami sveikieji skaičiai.
- 93.** Pakeiskite 92 užduoties programą taip, kad ji atliktų mažiausio operando paiešką masyve ir surastą mažiausią operandą įrašytų į atminties ląstelę, kurios adresas yra $210A_{16}$.
- 94.** Atmintyje, pradedant adresu 2200_{16} , yra dvidešimties 8 bitų operandų masyvas. Asemblerio kalba parašykite programą, kuri atliktų di-

džiausio ir mažiausio operando paiešką masyve, iš surasto didžiausio operando atimtų surastą mažiausią ir gautą skirtumą įrašytų į atminties ląstelę, kurios adresas yra 2110_{16} . Priimti, kad visi operandai masyve yra teigiami sveikieji skaičiai.

95. Asemblerio kalba parašykite programą, kuri 8 bitų operandų masyve nustatytų neigiamų (aukščiausioje skiltyje yra vienetas) operandų skaičių ir jį įrašytų į atminties ląstelę, kurios adresas yra 2400_{16} . Operandų masyvas prasideda adresu 2211_{16} , o jo dydis nurodytas atminties ląstelėje, kurios adresas yra 2210_{16} .
96. Asemblerio kalba parašykite programą, kuri apskaičiuotų 8 bitų operandų eilutės sumą. Operandų eilutė prasideda adresu 2140_{16} , o jos ilgis nurodytas adrese $213F_{16}$.
 - a. Priimti, kad gauta suma bus 8 bitų skaičius ir pernašos galima nepaisyti. Gautą sumą įrašyti į atminties ląstelę, kurios adresas yra 2300_{16} ;
 - b. Priimti, kad gauta suma bus 16 bitų skaičius. Gautą sumą įrašyti į atminties ląsteles, kurių adresai yra 2300_{16} ir 2301_{16} .
97. Atmintyje, pradedant adresu 2100_{16} , yra penkiolikos 8 bitų operandų masyvas. Asemblerio kalba parašykite programą, kuri perkeltų masyvą į kitą atminties sritį pradedant adresu 2200_{16} .
98. Asemblerio kalba parašykite programą, kuri 8 bitų operandų masyve apskaičiuotų lyginių operandų sumą ir ją įrašytų į atminties ląstelę, kurios adresas yra 2310_{16} . Operandų masyvas prasideda adresu 2201_{16} , o jo dydis nurodytas atminties ląstelėje, kurios adresas yra 2200_{16} . Priimti, kad gauta suma bus 8 bitų skaičius ir pernašos galima nepaisyti.
99. Asemblerio kalba parašykite programą, kuri 8 bitų operandų masyve apskaičiuotų nelyginių operandų sumą. Operandų masyvas prasideda adresu 2201_{16} , o jo dydis nurodytas atminties ląstelėje, kurios adresas yra 2200_{16} . Priimti, kad gauta suma bus 16 bitų skaičius. Gautą sumą įrašyti į atminties ląsteles, kurių adresai yra 2240_{16} ir 2241_{16} .

- 100.** Asemblerio kalba parašykite programą, kuri apskaičiuotų vienetų skaičių operando dvejetainiame kode. Operandas yra registre **H**. Apskaičiuotą vienetų skaičių įrašyti į registrą **L**.
- 101.** Asemblerio kalba parašykite programą, kuri apskaičiuotų nulių skaičių operando dvejetainiame kode. Operandas yra registre **H**. Apskaičiuotą vienetų skaičių įrašyti į registrą **L**.
- 102.** Asemblerio kalba parašykite programą, kuri apskaičiuotų vienetų ir nulių skaičių operando dvejetainiame kode. Operandas yra registre **E**. Apskaičiuotą vienetų skaičių įrašyti į registrą **H**, o nulių skaičių – į registrą **L**.
- 103.** Asemblerio kalba parašykite programą, kuri didėjančia tvarka išrikiuotų dešimt 8 bitų operandų, esančių atmintyje pradedant adresu 2100_{16} .
- 104.** Asemblerio kalba parašykite programą, kuri sudaugintų du 8 bitų operandus, esančius atminties ląstelėse 2100_{16} ir 2101_{16} , ir gautą rezultatą įrašytų į atminties ląsteles 2180_{16} ir 2181_{16} . Priimti, kad operandai yra teigiami sveikieji skaičiai didesni už nulį.

104. Sprendimas.

Mikroprocesoriaus Intel® 8080 komandų sistemoje daugybos komandos nėra, todėl, norint sudauginti du operandus, reikia rašyti tai atliekančią programą. Daugybos veiksmą keisime pakartotine sudėtimi.

Tarkime, pirmasis 8 bitų operandas $B1_{16}$ yra atminties ląstelėje, kurios adresas yra 2100_{16} , o antrasis 8 bitų operandas yra $04_{16} - 2101_{16}$.

Gautą daugybos rezultatą (sandaugą) $B1_{16} + B1_{16} + B1_{16} + B1_{16} = 02C4_{16}$ įrašysime į atminties ląsteles, kurių adresai yra 2180_{16} ir 2181_{16} , t. y. $C4_{16} - 2180_{16}$, $02_{16} - 2181_{16}$.

Žymė	Komandos mnemonika	Komentaras
	LDA 2100	; Pirmasis operandas iš atminties ląstelės, kurios ; adresas yra antrasis ir trečiasis komandos baitai, ; persiunčiamas į kaupiklį A ($M[ADR] \rightarrow A$, ; $M[2100_{16}] \rightarrow A$, $A = B1_{16}$).
	MOV E, A	; Kaupiklio A turinys persiunčiamas į registrą E ; ($A \rightarrow E$, $E = B1_{16}$).
	MVI D, 00	; Į registrą D persiunčiamas vieno baito tiesioginis ; operandas (antrasis komandos baitas) ($00_{16} \rightarrow D$, D ; $= 00_{16}$).
	LDA 2101	; Antrasis operandas iš atminties ląstelės, kurios ; adresas – antrasis ir trečiasis komandos baitai, ; persiunčiamas į kaupiklį A ($M[ADR] \rightarrow A$, ; $M[2101_{16}] \rightarrow A$, $A = 04_{16}$). Šį operandą priimame ; kaip sudėties veiksmo kartojimo skaičių.
	MOV C, A	; Kaupiklio A turinys persiunčiamas į registrą C ; ($A \rightarrow C$, $C = 04_{16}$).
	LXI H, 0000	; Į registrų porą HL persiunčiamas dviejų baitų ; tiesioginis operandas (antrasis ir trečiasis komandos ; baitai) ($00_{16} \rightarrow L$, $00_{16} \rightarrow H$).
Z1:	DAD D	; Registrų poros DE turinys pridamas prie registrų ; poros HL turinio ($HL + DE \rightarrow HL$).
	DCR C	; Dekrementuojamas registro C turinys ($C - 1 \rightarrow C$).
	JNZ Z1	; Jei dekrementavimo rezultatas nelygus nuliui, tai ; pereinama ir pradama vykdyti komanda, pažymėta ; žyme Z1. Jei dekrementavimo rezultatas lygus nuliui, ; tai pereigos nėra ir vykdoma paskui einanti komanda.

Žymė	Komandos mnemonika	Komentaras
	SHLD 2180	; Gautas daugybos rezultatas (sandauga) iš registrų L ; ir H persiunčiamas į atminties ląsteles, iš kurių ; pirmosios adresas yra antrasis ir trečiasis komandos ; baitai ($L \rightarrow M[ADR]$, $H \rightarrow M[ADR + 1]$, $L \rightarrow$; $M[2180_{16}]$, $H \rightarrow M[2181_{16}]$, $2180_{16} = C4_{16}$, ; $2181_{16} = 02_{16}$).
	HLT	; Stabdomas mikroprocesoriaus darbas.

- 105.** Asemblerio kalba parašykite programą, kuri skaičiuotų skaičių nuo 0 iki 8 faktorialą. Faktorialui skaičiuoti parašykite atskirą paprogramį.
- 106.** Asemblerio kalba parašykite programą, kuri padalytų 16 bitų operandą, esantį atminties ląstelėse 2400_{16} ir 2401_{16} , iš 8 bitų operando, esančio atminties ląstelėje 2402_{16} , ir gautą rezultatą (dalmenį) įrašytų į atminties ląsteles 2500_{16} ir 2501_{16} , o liekaną – į 2502_{16} ir 2503_{16} .
- 107.** Atmintyje yra du dešimties 8 bitų operandų masyvai, kurie atitinkamai prasideda adresais 2200_{16} ir 2300_{16} . Asemblerio kalba parašykite programą, kuri iš eilės sudėtų abiejų masyvų operandus, t. y. pirmojo masyvo pirmąjį operandą sudėtų su antrojo masyvo pirmuoju operandu ir t. t. Gautus rezultatus iš eilės įrašyti į atmintį pradedant adresu 2400_{16} .
- 108.** Atmintyje, pradedant adresu 2200_{16} , yra penkiasdešimties 8 bitų operandų masyvas. Asemblerio kalba parašykite programą, kuri atliktų lyginių operandų paiešką ir surastus lyginius operandus iš eilės įrašytų į atmintį pradedant adresu 2300_{16} .
- 109.** Atmintyje, pradedant adresu 2600_{16} , yra penkiasdešimties 8 bitų operandų masyvas. Asemblerio kalba parašykite programą, kuri nustatytų neigiamų (aukščiausioje skiltyje yra vienetas), teigiamų ir lygių nuliui operandų skaičius ir juos įrašytų atitinkamai į atminties ląsteles, kurių adresai yra 2700_{16} , 2701_{16} ir 2702_{16} .

- 110.** Du šešių skaitmenų dešimtainiai skaičiai užrašyti suglaudinta dvejetainė-dešimtaine forma. Kiekvienas skaičius užima po tris baitus atminties. Pirmasis skaičius yra atmintyje pradedant adresu 6000_{16} , o antrasis – pradedant adresu 6100_{16} . Asemblerio kalba parašykite programą, kuri sudėtų šiuos skaičius, ir gautą rezultatą tokia pat formą įrašytų į atmintį pradedant adresu 6200_{16} .
- 111.** Asemblerio kalba parašykite programą operatyviajai atminčiai testuoti. Atminties ląstelės testuojamos palyginant įrašytą informaciją su nuskaityta informacija (pirmajame cikle įrašomi ir nuskaitomi vienetai, antrajame cikle įrašomi ir nuskaitomi nuliai). Testuojamos atminties sritis nuo 2200_{16} iki 2300_{16} adreso. Aptikus klaidą, išvesti vienetą į išvesties prievadą, kurio adresas yra 02_{16} .
- 112.** Asemblerio kalba parašykite programos fragmentą, kuris sukurtų $0,5$ s vėlinimą, kai mikroprocesoriaus taktinis dažnis:
- $f_t = 2,5$ MHz;
 - $f_t = 5$ MHz;
 - $f_t = 200$ kHz;
 - $f_t = 3,13$ MHz;
 - $f_t = 2,67$ MHz;
 - $f_t = 6$ MHz.

112. a. *Sprendimas.*

Rašant realaus laiko programas, tokias kaip šviesoforo valdymo, skaitmeninio laikrodžio, įvairių procesų valdymo ir kt., labai svarbu nuolat stebėti laiką, t. y. sudaryti įvairios trukmės vėlinimo (laukimo) arba skaičiavimo paprogramius ar programos fragmentus. Panagrinėsime programiniu būdu sudaromus vėlinimus. Yra trys būdai:

- 1) taikant komandą **NOP**;
- 2) taikant 8 arba 16 bitų skaitiklius;
- 3) taikant įterptus ciklus.

Komanda **NOP** yra „tuščia“, t. y. nieko nekeičianti (išskyrus komandų skaitiklį **PC**), keturiais taktiniais intervalais vėlinanti komanda. Žinant mikroprocesoriaus taktinį dažnį, galima gana tiksliai apskaičiuoti, kiek

kartų reikės kartoti komandą **NOP**, kad gautume norimos trukmės vėlinimą. Komandą **NOP** galima pakeisti kitomis „nieko neatliekančiomis“ komandomis, pvz., **MOV A, A**. Ši komanda vėlins penkiais taktiniais intervalais.

Skaičiavimas yra patogus būdas sudaryti norimos trukmės vėlinimus. Žinant skaičiavimo paprogramyje esančių komandų vykdymo trukmę, labai lengva apsiskaičiuoti skaitiklio reikšmę, kuri užtikrins reikiamos trukmės vėlinimą.

Vėlinimo programos fragmentas, taikant 8 bitų skaitiklį:

Žymė	Komandos mnemonika	Komentaras	Taktų skaičius
	MVI C, XX	; Į registrą C persiunčiamas vieno baido ; tiesioginis operandas (antrasis komandos ; baidas), t. y. 8 bitų skaitiklis XX_{16} ; ($XX_{16} \rightarrow C, C = XX_{16}$).	7
Z1:	DCR C	; Dekrementuojamas registro C turinys, ; t. y. dekrementuojamas 8 bitų skaitiklis ; $XX_{16} (C - 1 \rightarrow C, XX_{16} - 1 \rightarrow C)$.	4
	JNZ Z1	; Jei dekrementavimo rezultatas nelygus ; nuliui, tai pereinama ir pradedama ; vykdyti komanda, ; pažymėta žyme Z1.	10

Komandos **DCR C** ir **JNZ Z1** bus vykdomos XX_{16} kartų, taigi taktinių intervalų skaičius, reikalingas šiam programos fragmentui įvykdyti, yra lygus: $N = 7 + (XX - 1) \times 14 + 14$. Iš čia 8 bitų skaitiklio turinys yra lygus: $XX = ((N - 21) / 14) + 1$.

Tarkime, kad $XX = 5$, tuomet $N = 7 + (5 - 1) \times 14 + 14 = 7 + 56 + 14 = 77$. Kai mikroprocesoriaus taktinis dažnis yra $f_t = 2$ MHz, vienas taktinis intervalas trunka: $t_l = 1 / f_t = 1 / 2000000 = 0,0000005$ s = 0,5 μs. Taigi vėlinimo trukmė: $t = N \times t_l = 77 \times 0,5 = 38,5$ μs.

Maksimali vėlinimo trukmė gaunama, kai yra didžiausias galimas 8 bitų skaičius: $XX = 255_{10} = FF_{16}$. Kai mikroprocesoriaus taktinis dažnis

$f_t = 2 \text{ MHz}$, maksimali vėlinimo trukmė: $t = (7 + (255 - 1) \times 14 + 14) \times 0,5 = 1788,5 \text{ } \mu\text{s}$.

Vėlinimo programos fragmentas, taikant 16 bitų skaitiklį:

Žymė	Komandos mnemonika	Komentaras	Taktų skaičius
	LXI B, XXXX	; Į registrų porą BC persiunčiamas dviejų ; baitų tiesioginis operandas (antrasis ir ; trečiasis komandos baitai), t. y. 16 bitų ; skaitiklis $XXXX_{16}$; $(XXXX_{16} \rightarrow BC, BC = XXXX_{16})$.	10
Z1:	DCX B	; Dekrementuojamas registrų poros B ; (registrai B ir C) turinys, t. y. dekremen- ; tuojamas 16 bitų skaitiklis $XXXX_{16}$; $(BC - 1 \rightarrow BC, XXXX_{16} - 1 \rightarrow BC)$.	5
	MOV A, C	; Registro C turinys persiunčiamas į ; kaupiklį A ($C \rightarrow A$).	5
	ORA B	; Registro B turinys logiškai sudedamas su ; kaupiklio A turiniu ($A \vee B \rightarrow A$).	4
	JNZ Z1	; Jei loginės sudėties rezultatas nelygus ; nuliui, tai pereinama ir pradedama ; vykdyti komanda, pažymėta žyme Z1.	10

Taktinių intervalų skaičius, reikalingas šiam programos fragmentui įvykdyti, yra lygus: $N = 10 + (XXXX - 1) \times 24 + 24$. Iš čia 16 bitų skaitiklio turinys yra lygus: $XXXX = ((N - 34) / 24) + 1$.

Maksimali vėlinimo trukmė gaunama, kai yra didžiausias galimas 16 bitų skaičius: $XXXX = 65535_{10} = \text{FFFF}_{16}$. Kai mikroprocesoriaus taktinis dažnis yra $f_t = 2 \text{ MHz}$, maksimali vėlinimo trukmė: $t = (10 + (65535 - 1) \times 24 + 24) \times 0,5 = 0,786425 \text{ s}$.

Vėlinimo programos fragmentas, taikant įterptus ciklus:

Žymė	Komandos mnemonika	Komentaras	Taktų skaičius
	MVI B, YY	; Į registrą B persiunčiamas vieno baido ; tiesioginis operandas (antrasis komandos ; baitas), t. y. 8 bitų daugiklis ; YY_{16} ($YY_{16} \rightarrow B, B = YY_{16}$).	7
Z2:	MVI C, XX	; Į registrą C persiunčiamas vieno baido ; tiesioginis operandas (antrasis komandos ; baitas), t. y. 8 bitų skaitiklis XX_{16} ; ($XX_{16} \rightarrow B, B = XX_{16}$).	7
Z1:	DCR C	; Dekrementuojamas registro C turinys, ; t. y. dekrementuojamas 8 bitų skaitiklis ; XX_{16} ($C - 1 \rightarrow C, XX_{16} - 1 \rightarrow C$).	4
	JNZ Z1	; Jei dekrementavimo rezultatas nelygus ; nuliui, tai pereinama ir pradedama ; vykdyti komanda, pažymėta žyme Z1. ; Jei dekrementavimo rezultatas lygus nuliui, ; tai pereigos nėra ir vykdoma paskui ; einanti komanda.	10
	DCR B	; Dekrementuojamas registro B turinys, ; t. y. dekrementuojamas 8 bitų daugiklis ; YY_{16} ($B - 1 \rightarrow B, YY_{16} - 1 \rightarrow B$).	4
	JNZ Z2	; Jei dekrementavimo rezultatas nelygus ; nuliui, tai pereinama ir pradedama ; vykdyti komanda, pažymėta žyme Z2.	10

Taktinių intervalų skaičius, reikalingas įterptam ciklui įvykdyti, yra lygus: $N_i = 7 + (XX - 1) \times 14 + 14$.

Taktinių intervalų skaičius, reikalingas visam programos fragmen-tui įvykdyti, yra lygus: $N = 7 + (YY - 1) \times (N_i + 14) + 14$.

Maksimali vėlinimo trukmė gaunama, kai yra didžiausi galimi 8 bitų skaičiai: $XX = YY = 255_{10} = FF_{16}$. Kai mikroprocesoriaus taktinis

dažnis yra $f_i = 2 \text{ MHz}$, maksimali vėlinimo trukmė $t = (7 + (255 - 1) \times ((7 + (255 - 1) \times 14 + 14) + 14) \times 0,5 \approx 0,456068 \text{ s}$.

Užduotyje reikalaujama sukurti programos fragmentą, kuris sukurtų 0,5 s vėlinimą, vadinasi, vėlinimui gauti taikysime programinį būdą su 16 bitų skaitikliu. Mikroprocesoriaus taktinis dažnis $f_i = 2,5 \text{ MHz}$. Apskaičiuojame vieno taktinio intervalo trukmę: $t_i = 1 / f_i = 1 / 2500000 = 0,0000004 \text{ s} = 0,4 \text{ } \mu\text{s}$. Apskaičiuojame taktinių intervalų skaičių, reikalingą 0,5 s vėlinimo paprogramiui įvykdyti: $N = t / t_i = 0,5 / 0,0000004 = 1250000$. Taigi 16 bitų skaitiklio turinys reikalingas 0,5 s vėlinimui gauti: $XXXX = ((N - 34) / 24) + 1 = ((1250000 - 34) / 24) + 1 \approx 52083_{10} = \text{CB73}_{16}$.

0,5 s vėlinimo programos fragmentas atrodo taip:

LXI B, CB73

Z1: DCX B

MOV A, C

ORA B

JNZ Z1

2. ŠEŠIOLIKOS SKILČIŲ BENDROSIOS PASKIRTIES MIKROPROCESORIŲ KOMANDŲ SISTEMA IR PROGRAMAVIMAS

2.1. Mikroprocesoriaus Intel® 8086 atminties organizacija

113. Duotas fizinis atminties ląstelės adresas 47560_{16} . Užrašykite šios atminties ląstelės loginį adresą.

113. Sprendimas.

Fizinis atminties ląstelės adresas formuojamas iš segmento adreso (kartotinis visuomet 10_{16}) ir adreso segmento viduje. Atminties ląstelė segmento viduje nurodoma vadinamuoju efektyviuoju adresu (EA) arba poslinkiu Disp (angl. *Displacement*).

Fizinį 20 bitų adresą mikroprocesorius Intel® 8086 apskaičiuoja sudėdamas per keturias skiltis į kairę perstumtą 16 bitų segmento adresą, nurodytą viename iš segmentų registrų, su 16 bitų efektyviuoju adresu arba poslinkiu šio segmento pradžios atžvilgiu (2.1 pav.).

2.1 pav. Fizinio adreso formavimas

Fizinis atminties ląstelės adresas loginiais adresais gali būti užrašytas nevienareikšmiškai. Fizinis adresas 47560_{16} gali būti užrašytas tokiais loginiais adresais: $4000:7560_{16}$ arba $4110:6460_{16}$, arba $4002:7540_{16}$ ir pan.

114. Duota: $\mathbf{BX} = 1420_{16}$, $\mathbf{DI} = 11\mathbf{B}2_{16}$, $\text{Disp L} = \mathbf{F}2_{16}$, $\text{Disp H} = 01_{16}$, $\mathbf{SP} = 2400_{16}$, $\mathbf{DS} = \mathbf{A}000_{16}$, $\mathbf{SS} = 1200_{16}$. Apskaičiuokite operando efektyvųjį ir fizinį adresą atmintyje, kai:

- Adresavimas tiesioginis;
- Adresavimas šalutinis registrinis;
- Adresavimas bazinis su 8 bitų poslinkiu;
- Adresavimas bazinis su 16 bitų poslinkiu;
- Adresavimas indeksinis su 8 bitų poslinkiu;
- Adresavimas indeksinis su 16 bitų poslinkiu;
- Adresavimas bazinis-indeksinis su 8 bitų poslinkiu;
- Adresavimas bazinis-indeksinis su 16 bitų poslinkiu;
- Adresavimas bazinis-indeksinis be poslinkio;
- Adresavimas dėklinis.

115. Duota: $AX = 1532_{16}$, $BX = 0F00_{16}$, $CX = 0053_{16}$, $DX = 00FE_{16}$, $SP = 3F00_{16}$, $BP = 0080_{16}$, $SI = 1000_{16}$, $DI = 0150_{16}$, $DS = 0650_{16}$, $ES = 0B4A_{16}$, $SS = 1100_{16}$, $CS = 2A00_{16}$, $IP = 11F0_{16}$. Apskaičiuokite:

- 1) Kitos komandos, kuri bus vykdoma, fizinį adresą atmintyje;
- 2) Dėklo viršūnės fizinį adresą atmintyje;
- 3) Operando efektyvųjį ir fizinį adresą atliekant šias komandas:
 - a. **MOV AX, [BX + SI + B234];**
 - b. **MOV BX, [0200];**
 - c. **SUB BYTE PTR [1A00], 05;**
 - d. **MOV CX, [DI];**
 - e. **MOV AX, [SI + 60];**
 - f. **MUL WORD PTR [BP - 0F];**
 - g. **IDIV BYTE PTR [SI + 000A];**
 - h. **AND [BX + DI + 03], AL;**
 - i. **MOVSF;**
 - j. **LDSW;**
 - k. **STOSB;**
 - l. **CMPSB;**
 - m. **PUSH AX;**
 - n. **POP [BX + 0100].**

116. Duota: $AX = 1A28_{16}$, $BX = 1800_{16}$, $CX = 0085_{16}$, $DX = 10FE_{16}$, $SP = 3A00_{16}$, $BP = 0280_{16}$, $SI = 1200_{16}$, $DI = 5150_{16}$, $DS = 2450_{16}$, $ES = 0300_{16}$, $SS = 1000_{16}$, $CS = 2700_{16}$, $IP = 1000_{16}$.

Apskaičiuokite operando efektyvųjį ir fizinį adresą atmintyje atlikdami duotas komandas, kurių šešiolyktainiai (mašininiai) kodai yra šie:

a. 00 39;	h. 89 36 4B 91;
b. 0B AC E8 41;	i. 88 07;
c. 8B 7D 67;	j. 80 0E BE 90 01;
d. 8A 15;	k. 8B 1D;
e. 8B 1E 78 14;	l. 80 3C 22;
f. FE 06 50 99;	m. 8A 47 01;
g. C7 06 BD 90 00 00;	n. F7 36 D6 91.

Pastaba: pasinaudoti 2 priede pateikta informacija.

- 117.** Duota: $AX = 0001_{16}$, $BX = 1000_{16}$, $CX = 0002_{16}$, $DX = 0003_{16}$,
 $SP = 1000_{16}$, $BP = 1200_{16}$, $SI = 1300_{16}$, $DI = 1400_{16}$, $DS = 1000_{16}$,
 $ES = 1200_{16}$, $SS = 2000_{16}$, $CS = 3000_{16}$, $IP = 1010_{16}$.

Apskaičiuokite operando efektyvųjį ir fizinį adresą atmintyje pagal adresavimo baitą:

a. 06_{16} ;	g. 12_{16} ;	m. $8C_{16}$;
b. 17_{16} ;	h. $1C_{16}$;	n. $4E_{16}$;
c. 14_{16} ;	i. $4C_{16}$;	o. $4F_{16}$;
d. 88_{16} ;	j. 48_{16} ;	p. $4B_{16}$;
e. 10_{16} ;	k. 15_{16} ;	q. 13_{16} ;
f. $8B_{16}$;	l. 11_{16} ;	r. 89_{16} ;

ir paskui einančius baitus 30_{16} ir 04_{16} .

Pastaba: pasinaudoti 2 priede pateikta informacija.

- 118.** Duotas fizinis atminties ląstelės adresas 18000_{16} . Parašykite programos fragmentą, kuris nustatytų duomenų segmento adresą.
- 119.** Duotas fizinis atminties ląstelės adresas 56987_{16} . Parašykite programos fragmentą, kuris nustatytų papildomo duomenų segmento adresą.
- 120.** Parašykite programos fragmentą, kuris atmintyje adresu $3000:1900_{16}$ organizuotų dėklą.

2.2. Komandų formatai ir operandų adresavimo būdai

121. Parašykite programą, kuri inkrementuotų arba dekrementuotų kaupiklio **AX** turinį ir gautą rezultatą nusiųstų į registrą **BX**.

121. Sprendimas.

Šiai užduočiai įvykdyti taikysime registrinį operandų adresavimo būdą. Operandai, esant šiam adresavimui, yra registre arba registruose, kurių adresai nurodomi vieninteliam operacijos kodo baite arba adresavimo baite.

Pateiksime komandas, taikančios registrinį adresavimą, pavyzdį (2.2 pav.).

2.2 pav. Komandos, taikančios registrinį adresavimą, veikimas

Tai reiškia, kad į registrą **BX** persiunčiami 16 bitų duomenys iš kaupiklio **AX**.

Taigi komandos su registriniu operandų adresavimu yra vieno arba dviejų baitų ilgio. Be to, joms vykdyti reikia minimalaus mikroprocesoriaus laiko, nes operandai yra jo viduje.

Tarkime, kad kaupiklio **AX** turinys yra lygus 0010_{16} , jį dekrementuosime ir gautą rezultatą persiūsime į registrą **BX**.

Komandos mnemonika	Komentaras
DEC AX	; Dekrementuojamas kaupiklio AX turinys ; $(r16 - 1 \rightarrow r16, AX - 1 \rightarrow AX, AX = 000F_{16})$.
MOV BX, AX	; Kaupiklio AX turinys persiunčiamas į registrą ; BX $(r16 \rightarrow r16, AX \rightarrow BX, BX = 000F_{16})$.
HLT	; Stabdomas mikroprocesoriaus darbas.

- 122.** Parašykite programą, kuri užpildytų mikroprocesoriaus kaupiklio **AX** turiniu registrus **BX, CX, DX, SI, DI, BP** ir **SP**.
- 123.** Parašykite programą, kuri į visus mikroprocesoriaus duomenų registrus įkeltų 16 bitų konstantą.

123. Sprendimas.
Šiai užduočiai išspręsti taikysime tiesioginį operandą.
Tiesioginis operandas Data užima 1 arba 2 baitus komandos kodo pabaigoje. Žemesnysis baitas Data L visuomet nurodomas pirmasis.
Pateiksime komandos, taikančios tiesioginį operandą, pavyzdį (2.3 pav.).

2.3 pav. Komandos, taikančios tiesioginį operandą, veikimas

Tai reiškia, kad 16 bitų tiesioginis operandas (komandos antrasis ir trečiasis baitai) persiunčiamas į registrą **CX**.

Tarkime, kad 16 bitų tiesioginis operandas yra lygus 3333_{16} , jį persiūsime į duomenų registrus **AX**, **BX**, **CX** ir **DX**.

Komandos mnemonika	Komentaras
MOV AX, 3333	; 16 bitų tiesioginis operandas persiunčiamas į ; kaupiklį AX ($\text{imm16} \rightarrow \text{r16}$, $3333_{16} \rightarrow \text{AX}$, ; $\text{AX} = 3333_{16}$).
MOV BX, 3333	; 16 bitų tiesioginis operandas persiunčiamas į ; registrą BX ($\text{imm16} \rightarrow \text{r16}$, $3333_{16} \rightarrow \text{BX}$, ; $\text{BX} = 3333_{16}$).
MOV CX, 3333	; 16 bitų tiesioginis operandas persiunčiamas į ; registrą CX ($\text{imm16} \rightarrow \text{r16}$, $3333_{16} \rightarrow \text{CX}$, ; $\text{CX} = 3333_{16}$).
MOV DX, 3333	; 16 bitų tiesioginis operandas persiunčiamas į ; registrą DX ($\text{imm16} \rightarrow \text{r16}$, $3333_{16} \rightarrow \text{DX}$, ; $\text{DX} = 3333_{16}$).
HLT	; Stabdomas mikroprocesoriaus darbas.

124. Parašykite programą, kuri persiųstų 16 bitų operandą į atmintį.

- Operandui persiųsti taikyti šalutinį registrinį operandų adresavimo būdą;
- Operandui persiųsti taikyti tiesioginį operandų adresavimo būdą;
- Operandui persiųsti taikyti tiesioginį operandą;
- Operandui persiųsti taikyti bazinį operandų adresavimo būdą;
- Operandui persiųsti taikyti indeksinį operandų adresavimo būdą;
- Operandui persiųsti taikyti bazinį-indeksinį operandų adresavimo būdą;
- Operandui persiųsti taikyti dėklinį operandų adresavimo būdą.

124. a. Sprendimas.

Komandose su šalutiniu registriniu adresavimu operando 16 bitų efektyvusis adresas (EA) yra viename iš registrų, kurį nustato adresavimo baito laukas **R/M**. Šiuo tikslu gali būti taikomas vienas iš registrų **BX**, **SI** arba **DI**.

Pateiksime komandos, taikančios šalutinį registrinį adresavimą, pa-vyzdį (2.4 pav.).

2.4 pav. Komandos, taikančios šalutinį registrinį adresavimą, veikimas

Tai reiškia, kad į registrą **DX** persiunčiami 16 bitų duomenys iš atminties ląstelių, pirmosios iš kurių adresą nurodo registras **BX**.

Tarkime, kad 16 bitų operandas yra lygus 2222₁₆, jį persiųsime į atmintį adresu 0001:1050₁₆.

Komandos mnemonika	Komentaras
MOV AX, 0001	; 16 bitų tiesioginis operandas persiunčiamas į ; kaupiklį AX ($\text{imm16} \rightarrow \text{r16}$, $0001_{16} \rightarrow \text{AX}$, ; $\text{AX} = 0001_{16}$).
MOV DS, AX	; Kaupiklio AX turinys persiunčiamas į duomenų ; segmento registrą DS, t. y. nustatomas duomenų ; segmento adresas ($\text{r16} \rightarrow \text{sr16}$, $\text{AX} \rightarrow \text{DS}$, ; $\text{DS} = 0001_{16}$).
MOV DI, 1050	; 16 bitų tiesioginis operandas persiunčiamas į ; registrą DI, t. y. persiunčiamas 16 bitų operando ; adresas atmintyje ($\text{imm16} \rightarrow \text{r16}$, $1050_{16} \rightarrow \text{DI}$, ; $\text{DI} = 1050_{16}$).
MOV DX, 2222	; 16 bitų tiesioginis operandas persiunčiamas į ; registrą DX ($\text{imm16} \rightarrow \text{r16}$, $2222_{16} \rightarrow \text{DX}$, ; $\text{DX} = 2222_{16}$).
MOV [DI], DX	; Registro DX turinys persiunčiamas į atminties ; ląsteles, pirmosios iš kurių adresas yra nurodytas ; registre DI ($\text{r16} \rightarrow \text{m16}$, $\text{DX} \rightarrow \text{M}[\text{DI}]$, $\text{DX} \rightarrow$; $\text{M}[1050_{16}]$, $1050_{16} = 22_{16}$, $1051_{16} = 22_{16}$).
HLT	; Stabdomas mikroprocesoriaus darbas.

124. b. Sprendimas.

Tiesioginis operandų adresavimas yra toks adresavimas, kai operando adresas nurodomas pačiame komandos kode. Duomenų apdorojimo komandose tiesioginis adresas nurodomas po adresavimo baido einančiu poslinkiu Disp L ir Disp H.

Pateiksime komandas, taikančios tiesioginį adresavimą, pavyzdį (2.5 pav.).

2.5 pav. Komandos, taikančios tiesioginį adresavimą, veikimas

Tai reiškia, kad į registrą **DX** persiunčiami duomenys iš atminties ląstelių, pirmosios iš kurių adresas yra trečiasis ir ketvirtasis komandos baitai.

Tarkime, kad 16 bitų operandas yra lygus 1973₁₆, jį persiūsime į atmintį adresu 0100:1300₁₆.

Komandos mnemonika	Komentaras
MOV AX, 0100	; 16 bitų tiesioginis operandas persiunčiamas į ; kaupiklį AX (imm16 → r16, 0100 ₁₆ → AX, ; AX = 0100 ₁₆).
MOV DS, AX	; Kaupiklio AX turinys persiunčiamas į duomenų ; segmento registrą DS, t. y. nustatomas duomenų ; segmento adresas (r16 → sr16, AX → DS, ; DS = 0100 ₁₆).

Komandos mnemonika	Komentaras
MOV BX, 1973	; 16 bitų tiesioginis operandas persiunčiamas į ; registrą BX ($\text{imm16} \rightarrow \text{r16}$, $1973_{16} \rightarrow \text{BX}$, ; $\text{BX} = 1973_{16}$).
MOV [1300], BX	; Registro BX turinys persiunčiamas į atminties ; ląsteles, pirmosios iš kurių adresas yra nurodytas ; komandos kode ($\text{r16} \rightarrow \text{m16}$, $\text{BX} \rightarrow$; $\text{M}[\text{Disp H Disp L}]$, $\text{BX} \rightarrow \text{M}[1300_{16}]$, ; $1300_{16} = 73_{16}$, $1301_{16} = 19_{16}$).
HLT	; Stabdomas mikroprocesoriaus darbas.

124. c. Sprendimas.

Tarkime, kad 16 bitų tiesioginis operandas yra lygus 0103_{16} , jį persiųsiame į atmintį adresu $2000:1140_{16}$.

Komandos mnemonika	Komentaras
MOV AX, 2000	; 16 bitų tiesioginis operandas persiunčiamas į ; kaupiklį AX ($\text{imm16} \rightarrow \text{r16}$, $2000_{16} \rightarrow \text{AX}$, ; $\text{AX} = 2000_{16}$).
MOV DS, AX	; Kaupiklio AX turinys persiunčiamas į duomenų ; segmento registrą DS, t. y. nustatomas duomenų ; segmento adresas ($\text{r16} \rightarrow \text{sr16}$, $\text{AX} \rightarrow \text{DS}$, ; $\text{DS} = 2000_{16}$).
MOV WORD PTR [1140], 0103	; 16 bitų tiesioginis operandas persiunčiamas į ; atminties ląstelę, pirmosios iš kurių adresas yra ; nurodytas komandos kode ($\text{imm16} \rightarrow \text{m16}$, ; $0103_{16} \rightarrow \text{M}[\text{Disp H Disp L}]$, $0103_{16} \rightarrow$; $\text{M}[1140_{16}]$, $1140_{16} = 03_{16}$, $1141_{16} = 01_{16}$).
HLT	; Stabdomas mikroprocesoriaus darbas.

124. d. Sprendimas.

Kai adresavimas yra bazinis, operando efektyvusis adresas (EA) gaunamas sudedant registrų **BX** arba **BP** turinius su 8 arba 16 bitų poslinkiu. Jeigu poslinkis yra 8 bitų, jis pateikiamas papildomuoju kodu iš skaičių intervalo nuo -128 iki $+127$.

Pateiksime komandos, taikančios bazinį adresavimą, pavyzdį (2.6 pav.).

2.6 pav. Komandos, taikančios bazinį adresavimą, veikimas

Tai reiškia, kad duomenys iš atminties ląstelių, pirmosios iš kurių adresas yra registro **BP** turinio ir 16 bitų poslinkio (trečiasis ir ketvirtasis komandos baitai) suma, atimami iš registro **BX** turinio.

Tarkime, kad 16 bitų operandas yra lygus 2233₁₆, jį persiūsimė į atmintį adresu 0020:1500₁₆.

Komandos mnemonika	Komentaras
MOV AX, 0020	; 16 bitų tiesioginis operandas persiunčiamas į ; kaupiklį AX ($\text{imm}_{16} \rightarrow \text{r16}$, $0020_{16} \rightarrow \text{AX}$, ; $\text{AX} = 0020_{16}$).
MOV DS, AX	; Kaupiklio AX turinys persiunčiamas į duomenų ; segmento registrą DS, t. y. nustatomas duomenų ; segmento adresas ($\text{r16} \rightarrow \text{sr16}$, $\text{AX} \rightarrow \text{DS}$, ; $\text{DS} = 0020_{16}$).
MOV BX, 1400	; 16 bitų tiesioginis operandas persiunčiamas į ; registrą BX, t. y. persiunčiama operando adreso ; atmintyje dedamoji ($\text{imm}_{16} \rightarrow \text{r16}$, $1400_{16} \rightarrow$; BX , $\text{BX} = 1400_{16}$).
MOV DX, 2233	; 16 bitų tiesioginis operandas persiunčiamas į ; registrą DX ($\text{imm}_{16} \rightarrow \text{r16}$, $2233_{16} \rightarrow \text{DX}$, ; $\text{DX} = 2233_{16}$).
MOV [BX+0100], DX	; Registro DX turinys persiunčiamas į atminties ; ląsteles, pirmosios iš kurių adresas yra registro ; BX turinio ir 16 bitų poslinkio suma ($\text{r16} \rightarrow \text{m16}$, ; $\text{DX} \rightarrow \text{M}[\text{BX} + \text{Disp H Disp L}]$, $\text{DX} \rightarrow$; $\text{M}[1500_{16}]$, $1500_{16} = 33_{16}$, $1501_{16} = 22_{16}$).
HLT	; Stabdomas mikroprocesoriaus darbas.

124. e. Sprendimas.

Indeksinis adresavimas skiriasi nuo bazinio tik tuo, kad taikomi ne bazės registras **BX** ir bazės rodyklė **BP**, o indeksų registrai **SI** ir **DI**. 8 bitų poslinkis pateikiamas papildomuoju kodu iš skaičių intervalo nuo –128 iki +127.

Pateiksime komandas, taikančios indeksinį adresavimą, pavyzdį (2.7 pav.).

2.7 pav. Komandos, taikančios indeksinį adresavimą, veikimas

Tai reiškia, kad duomenys iš atminties ląstelės, kurios adresas yra registro **SI** turinio ir 16 bitų poslinkio (trečiasis ir ketvirtasis komandos baitai) suma, persiunčiami į registro **CX** žemesniąją dalį **CL**.

Tarkime, kad 16 bitų operandas yra lygus 7711₁₆, jį persiūsime į atmintį adresu 0000:1600₁₆.

Komandos mnemonika	Komentaras
MOV AX, 0000	; 16 bitų tiesioginis operandas persiunčiamas į ; kaupiklį AX (imm16 → r16, 0000 ₁₆ → AX, ; AX = 0000 ₁₆).
MOV DS, AX	; Kaupiklio AX turinys persiunčiamas į duomenų ; segmento registrą DS, t. y. nustatomas duomenų ; segmento adresas (r16 → sr16, AX → DS, ; DS = 0000 ₁₆).

Komandos mnemonika	Komentaras
MOV SI, 1700	; 16 bitų tiesioginis operandas persiunčiamas į ; registrą SI, t. y. persiunčiama operando adreso ; atmintyje dedamoji ($\text{imm16} \rightarrow \text{r16}$, $1700_{16} \rightarrow \text{SI}$, ; $\text{SI} = 1700_{16}$).
MOV DX, 7711	; 16 bitų tiesioginis operandas persiunčiamas į ; registrą DX ($\text{imm16} \rightarrow \text{r16}$, $7711_{16} \rightarrow \text{DX}$, ; $\text{DX} = 7711_{16}$).
MOV [SI-0100], DX	; Registro DX turinys persiunčiamas į atminties ; ląsteles, pirmosios iš kurių adresas yra registro SI ; turinio ir 16 bitų poslinkio skirtumas ($\text{r16} \rightarrow$; m16 , $\text{DX} \rightarrow \text{M}[\text{SI} - \text{Disp H Disp L}]$, $\text{DX} \rightarrow$; $\text{M}[1600_{16}]$, $1600_{16} = 11_{16}$, $1601_{16} = 77_{16}$).
HLT	; Stabdomas mikroprocesoriaus darbas.

124. f. Sprendimas.

Kai adresavimas yra bazinis-indeksinis, efektyvusis adresas (EA) formuojamas sumuojant bazės ir indeksų registrų turinius su poslinkiu. 8 bitų poslinkis pateikiamas papildomuoju kodu iš skaičių intervalo nuo -128 iki +127.

Pateiksime komandos, taikančios bazinį-indeksinį adresavimą, pavyzdį (2.8 pav.).

Tai reiškia, kad kaupiklio **AX** turinys persiunčiamas į atminties ląsteles, pirmosios iš kurių adresas yra registrų **BX** ir **SI** turinių bei 16 bitų poslinkio (trečiasis ir ketvirtasis komandos baitai) suma.

Tarkime, kad 16 bitų operandas yra lygus AABB_{16} , jį persiūsime į atmintį adresu $1200:1\text{A}00_{16}$.

2.7 pav. Komandos, taikančios bazinį-indeksinį adresavimą, veikimas

Komandos mnemonika	Komentaras
MOV AX, 1200	; 16 bitų tiesioginis operandas persiunčiamas į ; kaupiklį AX ($imm16 \rightarrow r16$, $1200_{16} \rightarrow AX$, ; $AX = 1200_{16}$).
MOV DS, AX	; Kaupiklio AX turinys persiunčiamas į duomenų ; segmento registrą DS, t. y. nustatomas duomenų ; segmento adresas ($r16 \rightarrow sr16$, $AX \rightarrow DS$, ; $DS = 1200_{16}$).

Komandos mnemonika	Komentaras
MOV SI, 1700	; 16 bitų tiesioginis operandas persiunčiamas į ; registrą SI, t. y. persiunčiama operando adreso ; atmintyje dedamoji ($\text{imm16} \rightarrow \text{r16}$, $1700_{16} \rightarrow \text{SI}$, ; $\text{SI} = 1700_{16}$).
MOV BX, 0200	; 16 bitų tiesioginis operandas persiunčiamas į ; registrą BX, t. y. persiunčiama operando adreso ; atmintyje dedamoji ($\text{imm16} \rightarrow \text{r16}$, $0200_{16} \rightarrow$; BX , $\text{BX} = 0200_{16}$).
MOV DX, AABB	; 16 bitų tiesioginis operandas persiunčiamas į ; registrą DX ($\text{imm16} \rightarrow \text{r16}$, $\text{AABB}_{16} \rightarrow \text{DX}$, ; $\text{DX} = \text{AABB}_{16}$).
MOV [BX+SI+0100], DX	; Registro DX turinys persiunčiamas į atminties ; ląsteles, pirmosios iš kurių adresas yra registrų ; BX ir SI turinių ir 16 bitų poslinkio suma ; ($\text{r16} \rightarrow \text{m16}$, $\text{DX} \rightarrow \text{M}[\text{BX}+\text{SI}+\text{Disp H Disp L}]$, ; $\text{DX} \rightarrow \text{M}[1\text{A}00_{16}]$, $1\text{A}00_{16} = \text{BB}_{16}$, ; $1\text{A}01_{16} = \text{AA}_{16}$).
HLT	; Stabdomas mikroprocesoriaus darbas.

124. g. Sprendimas.

Dėklinis adresavimas taikomas komandose **PUSH** ir **POP** (rašyti į dėklą ir skaityti iš dėklo). Operando adresas nurodomas dėklo rodyklėje **SP** ir automatiškai mažinamas arba didinamas dviem vienetais įrašant į dėklą arba skaitant iš jo. Dėklas užpildomas adresų mažėjimo kryptimi. Dėklas gali keistis duomenimis su registrais, segmentų registrais ir atmintimi.

Pateiksime komandos, taikančios dėklinį adresavimą, pavyzdį (2.9 pav.).

2.9 pav. Komandos, taikančios dėklinį adresavimą, veikimas

Tai reiškia, kad registro **BX** turinys persiunčiamas į dvi dėklo atminties ląsteles, pirmosios iš kurių adresas yra vienetu mažesnis nei esama dėklo rodyklės **SP** reikšmė, o antrosios – dviem vienetais mažesnis nei esama dėklo rodyklės **SP** reikšmė.

Tarkime, kad 16 bitų operandas yra lygus 5522_{16} , jį persiųsime į dėklą adresu $0010:1322_{16}$.

Komandos mnemonika	Komentaras
MOV AX, 0010	; 16 bitų tiesioginis operandas persiunčiamas į ; kaupiklį AX ($\text{imm16} \rightarrow \text{r16}$, $0010_{16} \rightarrow \text{AX}$, ; $\text{AX} = 0010_{16}$).
MOV SS, AX	; Kaupiklio AX turinys persiunčiamas į dėklo ; segmento registrą SS, t. y. nustatomas dėklo ; segmento adresas ($\text{r16} \rightarrow \text{sr16}$, $\text{AX} \rightarrow \text{SS}$, ; $\text{SS} = 0010_{16}$).

Komandos mnemonika	Komentaras
MOV SP, 1322	; 16 bitų tiesioginis operandas persiunčiamas į ; dėklo rodyklę SP, t. y. nustatoma dėklo viršūnė ; ($\text{imm16} \rightarrow \text{r16}$, $1322_{16} \rightarrow \text{SP}$, $\text{SP} = 1322_{16}$).
MOV CX, 5522	; 16 bitų tiesioginis operandas persiunčiamas į ; registrą CX ($\text{imm16} \rightarrow \text{r16}$, $5522_{16} \rightarrow \text{CX}$, ; $\text{CX} = 5522_{16}$).
PUSH CX	; Registro CX turinys persiunčiamas į atminties ; ląsteles, pirmosios iš kurių adresas yra vienetu ; mažesnis nei esama dėklo rodyklės SP reikšmė, o ; antrosios – dviem vienetais mažesnis nei esama ; dėklo rodyklės SP reikšmė ($\text{r16} \rightarrow \text{m16}$, $\text{CX} \rightarrow$; $\text{M}[\text{SP}]$, $\text{CX} \rightarrow \text{M}[1322_{16}]$, $1321_{16} = 55_{16}$, ; $1320_{16} = 22_{16}$).
HLT	; Stabdomas mikroprocesoriaus darbas.

125. Parašykite programą, kuri padidintų arba sumažintų pasirinktą kiekį kartų vieno iš duomenų registrų turinį ir gautą rezultatą įrašytų į atmintį.

- Rezultatui įrašyti taikyti šalutinį registrinį operandų adresavimo būdą;
- Rezultatui įrašyti taikyti tiesioginį operandų adresavimo būdą;
- Rezultatui įrašyti taikyti bazinį operandų adresavimo būdą;
- Rezultatui įrašyti taikyti indeksinį operandų adresavimo būdą;
- Rezultatui įrašyti taikyti bazinį-indeksinį operandų adresavimo būdą;
- Rezultatui įrašyti taikyti dėklinį operandų adresavimo būdą.

126. Parašykite programą, kuri sukeistų vietomis du 16 bitų operandus, esančius atmintyje. Priimti, kad: 1) operandai yra tame pačiame duomenų segmente; 2) operandai yra skirtinguose duomenų segmentuose.

- a. Operandams adresuoti taikyti šalutinį registrinį operandų adresavimo būdą;
 - b. Operandams adresuoti taikyti tiesioginį operandų adresavimo būdą;
 - c. Operandams adresuoti taikyti bazinį operandų adresavimo būdą;
 - d. Operandams adresuoti taikyti indeksinį operandų adresavimo būdą;
 - e. Operandams adresuoti taikyti bazinį-indeksinį operandų adresavimo būdą;
 - f. Operandams adresuoti parinkti skirtingus operandų adresavimo būdus.
- 127.** Parašykite programą, kuri sudėtų arba atimtų du 16 bitų operandus, esančius atmintyje, ir gautą rezultatą įrašytų atgal į atmintį. Priimti, kad: 1) operandai ir gautas rezultatas yra tame pačiame duomenų segmente; 2) operandai ir gautas rezultatas yra skirtinguose duomenų segmentuose.
- a. Operandams adresuoti ir rezultatui įrašyti taikyti šalutinį registrinį operandų adresavimo būdą;
 - b. Operandams adresuoti ir rezultatui įrašyti taikyti tiesioginį operandų adresavimo būdą;
 - c. Operandams adresuoti ir rezultatui įrašyti taikyti bazinį operandų adresavimo būdą;
 - d. Operandams adresuoti ir rezultatui įrašyti taikyti indeksinį operandų adresavimo būdą;
 - e. Operandams adresuoti ir rezultatui įrašyti taikyti bazinį-indeksinį operandų adresavimo būdą;
 - f. Operandams adresuoti ir rezultatui įrašyti parinkti skirtingus operandų adresavimo būdus.
- 128.** Parašykite programą, kuri sudėtų arba atimtų du 16 bitų operandus, esančius atmintyje, ir gautą rezultatą įrašytų į deklą nurodytu adresu. Priimti, kad: 1) operandai yra tame pačiame duomenų segmente; 2) operandai yra skirtinguose duomenų segmentuose.

- a.* Operandams adresuoti taikyti šalutinį registrinį operandų adresavimo būdą;
- b.* Operandams adresuoti taikyti tiesioginį operandų adresavimo būdą;
- c.* Operandams adresuoti taikyti bazinį operandų adresavimo būdą;
- d.* Operandams adresuoti taikyti indeksinį operandų adresavimo būdą;
- e.* Operandams adresuoti taikyti bazinį-indeksinį operandų adresavimo būdą;
- f.* Operandams adresuoti parinkti skirtingus operandų adresavimo būdus.

129. Parašykite programą, kuri sudėtų arba atimtų du 16 bitų operandus, esančius dėkle, ir gautą rezultatą įrašytų atgal į dėklą.

- a.* Operandai yra gretimose dėklo atminties ląstelėse;
- b.* Operandai yra nutolusiose viena nuo kitos dėklo atminties ląstelėse.

130. Parašykite programą, kuri sukeistų vietomis du 16 bitų operandus, esančius dėkle.

- a.* Operandai yra gretimose dėklo atminties ląstelėse;
- b.* Operandai yra nutolusiose viena nuo kitos dėklo atminties ląstelėse.

131. Parašykite programą, kuri persiųstų 32 bitų operandą į atmintį.

- a.* Operando žodžiams persiųsti taikyti šalutinį registrinį operandų adresavimo būdą;
- b.* Operando žodžiams persiųsti taikyti tiesioginį operandų adresavimo būdą;
- c.* Operando žodžiams persiųsti taikyti tiesioginį operandą;
- d.* Operando žodžiams persiųsti taikyti bazinį operandų adresavimo būdą;
- e.* Operando žodžiams persiųsti taikyti indeksinį operandų adresavimo būdą;

- f. Operando žodžiams persiūsti taikyti bazinį-indeksinį operandų adresavimo būdą;
- g. Operando žodžiams persiūsti taikyti dėklinį operandų adresavimo būdą.

132. Parašykite programą, kuri sukeistų vietomis du 32 bitų operandus, esančius atmintyje. Priimti, kad: 1) operandai yra tame pačiame duomenų segmente; 2) operandai yra skirtinguose duomenų segmentuose.

- a. Operandų žodžiams adresuoti taikyti šalutinį registrinį operandų adresavimo būdą;
- b. Operandų žodžiams adresuoti taikyti tiesioginį operandų adresavimo būdą;
- c. Operandų žodžiams adresuoti taikyti bazinį operandų adresavimo būdą;
- d. Operandų žodžiams adresuoti taikyti indeksinį operandų adresavimo būdą;
- e. Operandų žodžiams adresuoti taikyti bazinį-indeksinį operandų adresavimo būdą;
- f. Operandų žodžiams adresuoti parinkti skirtingus operandų adresavimo būdus.

132. 1) b. Sprendimas.

Tarkime, kad pirmasis 32 bitų operandas 11223344_{16} yra segmento 0002_{16} atminties ląstelėse 1100_{16} , 1101_{16} , 1102_{16} ir 1103_{16} , t. y. $1100_{16} = 44_{16}$, $1101_{16} = 33_{16}$, $1102_{16} = 22_{16}$, ir $1103_{16} = 11_{16}$. Antrasis 32 bitų operandas 55667788_{16} yra tame pačiame segmente atminties ląstelėse 1200_{16} , 1201_{16} , 1202_{16} ir 1203_{16} , t. y. $1200_{16} = 88_{16}$, $1201_{16} = 77_{16}$, $1202_{16} = 66_{16}$, ir $1203_{16} = 55_{16}$.

Komandos mnemonika	Komentaras
MOV SI, 0002	; 16 bitų tiesioginis operandas persiunčiamas į ; registrą SI ($imm16 \rightarrow r16$, $0002_{16} \rightarrow SI$, ; $SI = 0002_{16}$).

Komandos mnemonika	Komentaras
MOV DS, SI	; Registro SI turinys persiunčiamas į duomenų ; segmento registrą DS, t. y. nustatomas duomenų ; segmento adresas ($r16 \rightarrow sr16$, $SI \rightarrow DS$, ; $DS = 0002_{16}$).
MOV AX, [1100]	; Atminties ląstelių, pirmosios iš kurių adresas yra ; nurodytas komandos kode, turiniai, t. y. pirmojo ; 32 bitų operando žemesnysis žodis persiunčiamas ; į kaupiklį AX ($m16 \rightarrow r16$, $M[Disp\ H\ Disp\ L] \rightarrow$; AX, $M[1100_{16}\ 1101_{16}] \rightarrow AX$, $AX = 3344_{16}$).
MOV BX, [1102]	; Atminties ląstelių, pirmosios iš kurių adresas yra ; nurodytas komandos kode, turiniai, t. y. pirmojo ; 32 bitų operando aukštesnysis žodis ; persiunčiamas į registrą BX ($m16 \rightarrow r16$, ; $M[Disp\ H\ Disp\ L] \rightarrow BX$, $M[1102_{16}\ 1103_{16}] \rightarrow$; BX, $BX = 1122_{16}$).
MOV CX, [1200]	; Atminties ląstelių, pirmosios iš kurių adresas yra ; nurodytas komandos kode, turiniai, t. y. antrojo ; 32 bitų operando žemesnysis žodis persiunčiamas ; į registrą CX ($m16 \rightarrow r16$, $M[Disp\ H\ Disp\ L] \rightarrow$; CX, $M[1200_{16}\ 1201_{16}] \rightarrow CX$, $CX = 7788_{16}$).
MOV DX, [1202]	; Atminties ląstelių, pirmosios iš kurių adresas yra ; nurodytas komandos kode, turiniai, t. y. antrojo ; 32 bitų operando aukštesnysis žodis ; persiunčiamas į registrą DX ($m16 \rightarrow r16$, ; $M[Disp\ H\ Disp\ L] \rightarrow DX$, $M[1202_{16}\ 1203_{16}] \rightarrow$; DX, $DX = 5566_{16}$).
MOV [1100], CX	; Registro CX turinys persiunčiamas į atminties ; ląsteles, pirmosios iš kurių adresas yra nurodytas ; komandos kode, t. y. antrojo 32 bitų operando ; žemesnysis žodis persiunčiamas į pirmojo 32 bitų ; operando žemesniojo žodžio vietą ($r16 \rightarrow m16$, ; $CX \rightarrow M[Disp\ H\ Disp\ L]$, $CX \rightarrow M[1100_{16}$; $1101_{16}]$, $1100_{16} = 88_{16}$, $1101_{16} = 77_{16}$).

Komandos mnemonika	Komentaras
MOV [1102], DX	; Registro DX turinys persiunčiamas į atminties ; ląsteles, pirmosios iš kurių adresas yra nurodytas ; komandos kode, t. y. antrojo 32 bitų operando ; aukštesnysis žodis persiunčiamas į pirmojo 32 ; bitų operando aukštesniojo žodžio vietą ($r16 \rightarrow$; $m16$, $DX \rightarrow M[\text{Disp H Disp L}]$, $DX \rightarrow M[1102_{16}$; $1103_{16}]$, $1102_{16} = 66_{16}$, $1103_{16} = 55_{16}$).
MOV [1200], AX	; Kaupiklio AX turinys persiunčiamas į atminties ; ląsteles, pirmosios iš kurių adresas yra nurodytas ; komandos kode, t. y. pirmojo 32 bitų operando ; žemesnysis žodis persiunčiamas į antrojo 32 bitų ; operando žemesniojo žodžio vietą ($r16 \rightarrow m16$, ; $AX \rightarrow M[\text{Disp H Disp L}]$, $AX \rightarrow M[1200_{16}$; $1201_{16}]$, $1200_{16} = 44_{16}$, $1201_{16} = 33_{16}$).
MOV [1202], BX	; Registro BX turinys persiunčiamas į atminties ; ląsteles, pirmosios iš kurių adresas yra nurodytas ; komandos kode, t. y. pirmojo 32 bitų operando ; aukštesnysis žodis persiunčiamas į antrojo 32 ; bitų operando aukštesniojo žodžio vietą ($r16 \rightarrow$; $m16$, $BX \rightarrow M[\text{Disp H Disp L}]$, $BX \rightarrow M[1202_{16}$; $1203_{16}]$, $1202_{16} = 22_{16}$, $1203_{16} = 11_{16}$).
HLT	; Stabdomas mikroprocesoriaus darbas.

133. Parašykite programą, kuri sudėtų arba atimtų du 32 bitų operandus, esančius atmintyje, ir gautą rezultatą įrašytų atgal į atmintį. Priimti, kad: 1) operandai ir gautas rezultatas yra tame pačiame duomenų segmente; 2) operandai ir gautas rezultatas yra skirtinguose duomenų segmentuose.

- Operandų žodžiams ir rezultatui adresuoti taikyti šalutinį registrinį operandų adresavimo būdą;
- Operandų žodžiams ir rezultatui adresuoti taikyti tiesioginį operandų adresavimo būdą;

- c. Operandų žodžiams ir rezultatui adresuoti taikyti bazinį operandų adresavimo būdą;
- d. Operandų žodžiams ir rezultatui adresuoti taikyti indeksinį operandų adresavimo būdą;
- e. Operandų žodžiams ir rezultatui adresuoti taikyti bazinį-indeksinį operandų adresavimo būdą;
- f. Operandų žodžiams ir rezultatui adresuoti parinkti skirtingus operandų adresavimo būdus.

134. Parašykite programą, kuri sudėtų arba atimtų du 32 bitų operandus, esančius atmintyje, ir gautą rezultatą įrašytų į dėklą nurodytu adresu. Priimti, kad: 1) operandai yra tame pačiame duomenų segmente; 2) operandai yra skirtinguose duomenų segmentuose.

- a. Operandų žodžiams adresuoti taikyti šalutinį registrinį operandų adresavimo būdą;
- b. Operandų žodžiams adresuoti taikyti tiesioginį operandų adresavimo būdą;
- c. Operandų žodžiams adresuoti taikyti bazinį operandų adresavimo būdą;
- d. Operandų žodžiams adresuoti taikyti indeksinį operandų adresavimo būdą;
- e. Operandų žodžiams adresuoti taikyti bazinį-indeksinį operandų adresavimo būdą;
- f. Operandų žodžiams adresuoti parinkti skirtingus operandų adresavimo būdus.

135. Parašykite programą, kuri sudėtų arba atimtų du 32 bitų operandus, esančius dėkle, ir gautą rezultatą įrašytų atgal į dėklą.

- a. Operandai yra gretimose dėklo atminties ląstelėse;
- b. Operandai yra nutolusiose viena nuo kitos dėklo atminties ląstelėse.

136. Parašykite programą, kuri sukeistų vietomis du 32 bitų operandus, esančius dėkle.

- a. Operandai yra gretimose dėklo atminties ląstelėse;

- b. Operandai yra nutolusiose viena nuo kitos dėklo atminties ląstelėse.

137. Parašykite programą, kuri sudėtų du 8 bitų operandus, esančius registre **DX**, ir gautą rezultatą įrašytų į pasirinktą atminties ląstelę. Suma neturi viršyti FF_{16} .

- Rezultatui įrašyti taikyti šalutinį registrinį operandų adresavimo būdą;
- Rezultatui įrašyti taikyti tiesioginį operandų adresavimo būdą;
- Rezultatui įrašyti taikyti bazinį operandų adresavimo būdą;
- Rezultatui įrašyti taikyti indeksinį operandų adresavimo būdą;
- Rezultatui įrašyti taikyti bazinį-indeksinį operandų adresavimo būdą;
- Rezultatui įrašyti taikyti dėklinį operandų adresavimo būdą.

137. a. Sprendimas.

Nagrinėjamo mikroprocesoriaus duomenų registrai sudaryti iš dviejų dalių: aukštesniosios **H** (angl. *High*) ir žemesniosios **L** (angl. *Low*), turinčių po 8 skiltis, į kurias galima kreiptis atskirai.

Tarkime, kad pirmasis operandas 05_{16} yra registro **DX** žemesniojoje dalyje **DL**, o antrasis 06_{16} yra aukštesniojoje dalyje **DH**.

Gautą sudėties rezultatą (sumą) $0B_{16}$ įrašysime į atmintį adresu $0001:1200_{16}$.

Komandos mnemonika	Komentaras
MOV AX, 0001	; 16 bitų tiesioginis operandas persiunčiamas į ; kaupiklį AX ($imm16 \rightarrow r16$, $0001_{16} \rightarrow AX$, ; $AX = 0001_{16}$).
MOV DS, AX	; Kaupiklio AX turinys persiunčiamas į duomenų ; segmento registrą DS, t. y. nustatomas duomenų ; segmento adresas ($r16 \rightarrow sr16$, $AX \rightarrow DS$, ; $DS = 0001_{16}$).

Komandos mnemonika	Komentaras
MOV SI, 1200	; 16 bitų tiesioginis operandas persiunčiamas į ; registrą SI, t. y. persiunčiamas operando adresas ; atmintyje ($\text{imm16} \rightarrow \text{r16}$, $1200_{16} \rightarrow \text{SI}$, ; $\text{SI} = 1200_{16}$).
ADD DH, DL	; Registro DX žemesniosios dalies DL turinys ; pridamas prie registro DX aukštesniosios dalies ; DH ($\text{r8} + \text{r8} \rightarrow \text{r8}$, $\text{DH} + \text{DL} \rightarrow \text{DH}$, $\text{DH} = 0\text{B}_{16}$).
MOV [SI], DH	; Registro DX aukštesniosios dalies DH turinys ; persiunčiamas į atminties ląstelę, kurios adresas ; nurodytas registre SI ($\text{r8} \rightarrow \text{m8}$, $\text{DH} \rightarrow \text{M}[\text{SI}]$, ; $\text{DH} \rightarrow \text{M}[1200_{16}]$, $1200_{16} = 0\text{B}_{16}$).
HLT	; Stabdomas mikroprocesoriaus darbas.

138. Parašykite programą, kuri sudėtų arba atimtų du 32 bitų operandus, esančius registruose **AX**, **BX** ir **SI**, **DI**, ir gautą rezultatą įrašytų į atmintį adresu ES:1400_{16} .

- Rezultatui įrašyti taikyti šalutinį registrinį operandų adresavimo būdą;
- Rezultatui įrašyti taikyti tiesioginį operandų adresavimo būdą;
- Rezultatui įrašyti taikyti bazinį operandų adresavimo būdą;
- Rezultatui įrašyti taikyti indeksinį operandų adresavimo būdą;
- Rezultatui įrašyti taikyti bazinį-indeksinį operandų adresavimo būdą.

139. Parašykite programą, kuri persiųstų 8 bitų operandą į pasirinktą atminties ląstelę.

- Operandui persiųsti taikyti šalutinį registrinį operandų adresavimo būdą;
- Operandui persiųsti taikyti tiesioginį operandų adresavimo būdą;
- Operandui persiųsti taikyti tiesioginį operandą;
- Operandui persiųsti taikyti bazinį operandų adresavimo būdą;
- Operandui persiųsti taikyti indeksinį operandų adresavimo būdą;

- f.* Operandui persiūsti taikyti bazinį-indeksinį operandų adresavimo būdą;
- g.* Operandui persiūsti taikyti dėklinį operandų adresavimo būdą.

140. Parašykite programą, kuri užpildytų visus mikroprocesoriaus duomenų registrus skirtingais 8 bitų operandais, esančiais pasirinktose skirtingose atminties ląstelėse.

- a.* Operandams adresuoti taikyti šalutinį registrinį operandų adresavimo būdą;
- b.* Operandams adresuoti taikyti tiesioginį operandų adresavimo būdą;
- c.* Operandams adresuoti taikyti bazinį operandų adresavimo būdą;
- d.* Operandams adresuoti taikyti indeksinį operandų adresavimo būdą;
- e.* Operandams adresuoti taikyti bazinį-indeksinį operandų adresavimo būdą;
- f.* Operandams adresuoti taikyti dėklinį operandų adresavimo būdą.

141. Parašykite programą, kuri sukeistų vietomis du 8 bitų operandus, esančius pasirinktose skirtingose atminties ląstelėse. Priimti, kad: 1) operandai yra tame pačiame duomenų segmente; 2) operandai yra skirtinguose duomenų segmentuose.

- a.* Operandams adresuoti taikyti šalutinį registrinį operandų adresavimo būdą;
- b.* Operandams adresuoti taikyti tiesioginį operandų adresavimo būdą;
- c.* Operandams adresuoti taikyti bazinį operandų adresavimo būdą;
- d.* Operandams adresuoti taikyti indeksinį operandų adresavimo būdą;
- e.* Operandams adresuoti taikyti bazinį-indeksinį operandų adresavimo būdą;
- f.* Operandams adresuoti parinkti skirtingus operandų adresavimo būdus.

142. Parašykite programą, kuri sudėtų arba atimtų du 8 bitų operandus, esančius pasirinktose skirtingose atminties ląstelėse, ir gautą rezultatą įrašytų į pasirinktą trečiąją atminties ląstelę. Priimti, kad: 1) operandai ir gautas rezultatas yra tame pačiame duomenų segmente; 2) operandai ir gautas rezultatas yra skirtinguose duomenų segmentuose.

- a. Operandams adresuoti ir rezultatui įrašyti taikyti šalutinį registrinį operandų adresavimo būdą;
- b. Operandams adresuoti ir rezultatui įrašyti taikyti tiesioginį operandų adresavimo būdą;
- c. Operandams adresuoti ir rezultatui įrašyti taikyti bazinį operandų adresavimo būdą;
- d. Operandams adresuoti ir rezultatui įrašyti taikyti indeksinį operandų adresavimo būdą;
- e. Operandams adresuoti ir rezultatui įrašyti taikyti bazinį-indeksinį operandų adresavimo būdą;
- f. Operandams adresuoti ir rezultatui įrašyti parinkti skirtingus operandų adresavimo būdus.

143. Parašykite programą, kuri sudėtų arba atimtų du 8 bitų operandus, esančius pasirinktose skirtingose atminties ląstelėse, ir gautą rezultatą įrašytų į deklą nurodytu adresu. Priimti, kad: 1) operandai yra tame pačiame duomenų segmente; 2) operandai yra skirtinguose duomenų segmentuose.

- a. Operandams adresuoti taikyti šalutinį registrinį operandų adresavimo būdą;
- b. Operandams adresuoti taikyti tiesioginį operandų adresavimo būdą;
- c. Operandams adresuoti taikyti bazinį operandų adresavimo būdą;
- d. Operandams adresuoti taikyti indeksinį operandų adresavimo būdą;
- e. Operandams adresuoti taikyti bazinį-indeksinį operandų adresavimo būdą;

f. Operandams adresuoti parinkti skirtingus operandų adresavimo būdus.

144. Parašykite programą, kuri sudėtų arba atimtų du 8 bitų operandus, esančius dėkle, ir gautą rezultatą įrašytų atgal į dėklą.

a. Operandai yra gretimose dėklo atminties ląstelėse;

b. Operandai yra nutolusiose viena nuo kitos dėklo atminties ląstelėse.

145. Parašykite programą, kuri sukeistų vietomis du 8 bitų operandus, esančius dėkle.

a. Operandai yra gretimose dėklo atminties ląstelėse;

b. Operandai yra nutolusiose viena nuo kitos dėklo atminties ląstelėse.

146. Parašykite programą, kuri registrų **AX**, **BX**, **CX** ir **DX** turinius įrašytų į dėklą nurodytu adresu.

147. Parašykite programą, kuri iš dėklo persiųstų 16 bitų operandus į mikroprocesoriaus duomenų registrus.

148. Parašykite programą, kuri iš atminties, pradedant adresu 64000_{16} , nuskaitytų dešimt baitų informacijos, ir iš eilės įrašytų į atmintį pradedant adresu 81025_{16} . Taikyti simbolių eilučių apdorojimo komandą **MOVSB** arba **MOVSW**.

148. Sprendimas.

Simbolių eilučių elementai gali būti baitai arba žodžiai. Pirminės simbolių eilutės elementai adresuojami taikant registrą **SI**, o patys elementai pagal nutylėjimą įrašomi į duomenų segmentą **DS**. Apdorota simbolių eilutė visuomet įrašoma į papildomą duomenų segmentą **ES**, o jos elementams adresuoti taikomas registras **DI**.

Jei prieš eilutės elementų apdorojimo komandą įterpiamas vieno baito kartojo prefiksas **REP**, tai atitinkama operacija bus kartojama tol, kol ciklo skaitiklio **CX** turinys taps lygus 0. Vykdamas eilutės apdorojimo komandą, automatiškai koreguojamas registrų **SI** ir **DI** turinys, atsižvel-

giant į požymių registro **FL** krypties skilties **D** reikšmę. Jei **D** = 0, po kiekvienos operacijos atitinkamų indeksų registrų turinys didinamas vienetu, jei veiksmai atliekami su baitais, ir dviem – jei veiksmai atliekami su žodžiais. Jei **D** = 1, indeksų registrų turiniai mažinami vienetu arba dviem.

Pateikiame simbolių eilučių apdorojimo komandos **MOVSB** kartu su kartojimo prefiksu **REPZ** pavyzdį (2.10 pav.).

2.10 pav. Simbolių eilučių apdorojimo komandos **MOVSB** veikimas

Komanda **MOVSB** arba **MOVSW** kopijuoja baitą arba žodį iš **DS:SI** ir persiunčia į **ES:DI**. Kad būtų persiųsta visa eilutė, taikomas kartojimo

prefiksas **REPZ**, kuris kartoja komandas **MOVS** registre **CX** nurodytą kartų skaičių.

Tarkime, kad dešimt baitų atmintyje yra išdėstyti pradedant adresu 6200:2000₁₆, t. y. fizinis adresas atmintyje yra 64000₁₆, ir perkelsime juos pradedami adresu 8000:1025₁₆, t. y. fizinis adresas atmintyje yra 81025₁₆.

Komandos mnemonika	Komentaras
MOV AX, 6200	; 16 bitų tiesioginis operandas persiunčiamas į ; kaupiklį AX (imm16 → r16, 6200 ₁₆ → AX, ; AX = 6200 ₁₆).
MOV DS, AX	; Kaupiklio AX turinys persiunčiamas į duomenų ; segmento registrą DS, t. y. nustatomas duomenų ; segmento adresas (r16 → sr16, AX → DS, ; DS = 6200 ₁₆).
MOV AX, 8000	; 16 bitų tiesioginis operandas persiunčiamas į ; kaupiklį AX (imm16 → r16, 8000 ₁₆ → AX, ; AX = 8000 ₁₆).
MOV ES, AX	; Kaupiklio AX turinys persiunčiamas į papildomą ; duomenų segmento registrą ES, t. y. nustatomas ; papildomas duomenų segmento adresas ; (r16 → sr16, AX → ES, ES = 8000 ₁₆).
MOV SI, 2000	; 16 bitų tiesioginis operandas persiunčiamas į ; registrą SI, t. y. persiunčiamas operando adresas ; atmintyje (imm16 → r16, 2000 ₁₆ → SI, ; SI = 2000 ₁₆).
MOV DI, 1025	; 16 bitų tiesioginis operandas persiunčiamas į ; registrą DI, t. y. persiunčiamas operando adresas ; atmintyje (imm16 → r16, 1025 ₁₆ → DI, ; DI = 1025 ₁₆).
MOV CX, 000A	; 16 bitų tiesioginis operandas persiunčiamas į ; registrą CX, t. y. persiunčiamas komandos ; MOVSB kartojo skaičius (imm16 → r16, ; 000A ₁₆ → CX, CX = 000A ₁₆ = 10 ₁₀).

Komandos mnemonika	Komentaras
CLD	; Į požymių registro FL krypties skiltį D įrašomas ; nulis, t. y. duomenų eilutė apdorojama nuo ; pradžios ($0 \rightarrow D$, $D = 0$).
REPZ	; Paskui einanti komanda MOVSB kartojama ; CX kartų, t. y., kol $CX \neq 0000$.
MOVSB	; Baitas persiunčiamas iš DS:SI į ES:DI.
HLT	; Stabdomas mikroprocesoriaus darbas.

Sprendžiant šią užduotį duomenų eilutei persiųsti, komandą **MOVSB** galima pakeisti komanda **MOVSW**, t. y. eilutė būtų skaitoma žodžiais. Kartojimų skaičius registre **CX** būtų du kartus mažesnis.

- 149.** Atlikite 148 užduotį, kai atminties segmentai sutampa.
- 150.** Parašykite programą, kuri 128 atminties ląsteles, pradedant adresu 21200_{16} , užpildytų nuliais. Taikyti simbolių eilučių apdorojimo komandą **STOSB** arba **STOSW**.
- 151.** Parašykite programą, kuri 2 KB atminties, pradedant adresu 53000_{16} , užpildytų 16 bitų konstanta $ABCD_{16}$. Taikyti simbolių eilučių apdorojimo komandą **STOSW**.
- 152.** Užrašykite duotos komandos formatą ir raskite šešioliktainius (mašininis) kodus. Pasinaudoti 2 priede pateikta informacija.
- MOV BL, DL;**
 - MOV BX, 1234;**
 - MOV [DI], CL;**
 - MOV WORD PTR [BP + SI], 000A;**
 - MOV CX, [BX + DI + 0250];**
 - ADD AX, DX;**
 - ADD AX, 0007;**
 - SHR AX;**
 - DEC BX;**

- j. **SUB DX, [1200];**
- k. **ADD WORD PTR [1257], 0202;**
- l. **MOV DX, [BX];**
- m. **SUB CL, [SI + 0200];**
- n. **PUSH AX;**
- o. **POP [BX + SI + 0150];**
- p. **MUL [SI];**
- q. **DIV SI;**
- r. **XCHG AX, AX;**
- s. **OR AX, [0500];**
- t. **SHL [BX + 2000], CL;**
- u. **ADC BX, AX.**

152. a. *Sprendimas.*

2 priede randame komandos **MOV** į registrą iš registro formatą:

	7	0	7	0	
	100010 d W		Mod Reg R/M		
MOV BL, DL	100010	0 0	11	010 011	= 88 D3 ₁₆

Komanda **MOV BL, DL** persiunčia 8 bitų operandą iš registro **DX** žemesniosios dalies **DL** į registro **BX** žemesniąją dalį **BL**, t. y. operuojama 8 bitų duomenimis, esančiais registruose, vadinasi, operacijos kodo baido žemiausioji skiltis **W** = 0, adresavimo baido laukas **Mod** = 11 ir laukas **R/M** reiškia registro dvejetainį adresą. Priimame, kad **d** = 0 ir lauke **Reg** įrašome registro siųstuvo, t. y. **DL** dvejetainį adresą, o lauke **R/M** – registro imtuvo, t. y. **BL** dvejetainį adresą. Užpildę visus laukus, turime komandos **MOV BL, DL** kodą: 10001000 11010011₂ = 88 D3₁₆.

2.3. Mikroprocesoriaus Intel® 8086 komandų sistema ir programavimas

- 153.** Parašykite programą, kuri atimtų 32 bitų operandą, esantį registruose **CX** ir **DX**, iš 32 bitų operando, esančio kaupiklyje **AX** ir registre **BX**, ir gautą rezultatą įrašytų į registrus **SI** ir **DI**.
- 154.** Parašykite programą, kuri apskaičiuotų dviejų 64 bitų operandų skirtumą. Pirmasis operandas yra atmintyje adresu DS:4000, antrasis – DS:BX. Gautą skirtumą įrašyti į atminties ląstelę adresu ES:DI.
- 155.** Parašykite programą, kuri sudaugintų 16 bitų konstantas $001C_{16}$ ir $058D_{16}$, ir gautą rezultatą įrašytų į atmintį adresu 1230:SI.
- 156.** Parašykite programą, kuri sudaugintų teigiamą skaičių 14_{16} su neigiamu skaičiumi 67_{16} ir gautą rezultatą įrašytų į registrą **CX**. Taikyti komandą **NEG**.
- 157.** Parašykite programą, kuri sudaugintų 8 bitų sveikąjį skaičių, esantį registre **BH**, su 16 bitų sveikuoju skaičiumi, esančiu registre **CX**. Gautą rezultatą įrašyti į atmintį adresu ES:BX. 8 bitų skaičiui išplėsti iki 16 bitų (išplėsti ženklo skilties reikšmę) taikyti komandą **CBW**.
- 158.** Parašykite programą, kuri padalytų 16 bitų skaičių 4926_{16} iš 16 bitų skaičiaus 2687_{16} ir gautą rezultatą įrašytų į atmintį adresu DS:DI.
- 159.** Parašykite programą, kuri padalytų neigiamą skaičių 8900_{16} iš teigiamo skaičiaus 2400_{16} ir gautą rezultatą įrašytų į atmintį adresu 0005:1200₁₆.

Pastaba: 16 skilčių sveikieji skaičiai yra intervale nuo -8000_{16} iki $+7FFF_{16}$. -8900_{16} nepatenka į šį intervalą, vadinasi, reikia atlikti 32 skilčių skaičiaus dalybą iš 16 skilčių skaičiaus.

- 160.** Parašykite programą, kuri apskaičiuotų septynių 8 bitų operandų, esančių atmintyje, aritmetinį vidurkį ir gautą rezultatą įrašytų į dėklą nurodytu adresu.
- 161.** Parašykite programą, kuri sudaugintų du 32 bitų operandus, esančius atmintyje, ir gautą rezultatą įrašytų atgal į atmintį.
- 162.** Parašykite programą, kuri sudėtų du dvejetainius-dešimtainius skaičius 6 ir 8 ir gauto rezultato ASCII kodą įrašytų į registrą **DX**. Taikyti komandą **AAA**.
- 163.** Parašykite programą, kuri sudėtų du dvejetainius-dešimtainius skaičius 78 ir 17 ir gauto rezultato ASCII kodą įrašytų atitinkamai į registrus **CL**, **BH** ir **BL**. Taikyti komandą **AAA**.
- 164.** Parašykite programą, kuri sudėtų du dvejetainius-dešimtainius skaičius 87 ir 71 ir gauto rezultato ASCII kodą įrašytų atitinkamai į kaupiklį **AX** ir registrą **BX**. Taikyti komandą **AAA**.
- 165.** Parašykite programą, kuri sudėtų keturių skaitmenų BCD skaičius, esančius kaupiklyje **AX** ir registre **BX**, ir gautą rezultatą įrašytų į registrą **CX** ir **DX**. Taikyti komandą **DAA**.
- 166.** Parašykite programą, kuri sudėtų du dvejetainius-dešimtainius skaičius 54 ir 76 ir gautą rezultatą įrašytų į registrą **CX**. Taikyti komandą **DAA**.
- 167.** Parašykite programą, kuri iš dvejetainio-dešimtainio skaičiaus 82 atimtų dvejetainį-dešimtainį skaičių 37 ir gautą rezultatą BCD formoje įrašytų į registrą **BL**. Taikyti komandą **DAS**.
- 168.** Parašykite programą, kuri apskaičiuotų dviejų šešių skaitmenų (24 bitai) BCD skaičių skirtumą. Pirmasis operandas yra atmintyje adresu 1000:1500₁₆, antrasis – 1000:3000₁₆. Gautą skirtumą įrašyti į atminties ląstelę adresu 2000:2000₁₆. Taikyti komandą **DAS**.

- 169.** Parašykite programą, kuri iš dvejetainio-dešimtainio skaičiaus 9 atimtu dvejetainį-dešimtainį skaičių 3 ir gauto rezultato ASCII kodą įrašytų į registrą **DL**. Taikyti komandą **AAS**.
- 170.** Parašykite programą, kuri iš dvejetainio-dešimtainio skaičiaus 2 atimtu dvejetainį-dešimtainį skaičių 7 ir gauto rezultato ASCII kodą įrašytų į registrą **DH**. Taikyti komandą **AAS**.
- 171.** Parašykite programą, kuri sudaugintų du dvejetainius-dešimtainius skaičius 2 ir 5 ir gauto rezultato ASCII kodą įrašytų į registrą **CX**. Taikyti komandą **AAM**.
- 172.** Parašykite programą, kuri pervestų 8 skilčių dvejetainį skaičių į ne-glaudintą 8 skilčių BCD skaičių. Taikyti komandą **AAM**.
- 173.** Parašykite programą, kuri padalytų dvejetainį-dešimtainį skaičių 14 iš dvejetainio-dešimtainio skaičiaus 7 ir gautą rezultato ASCII kodą įrašytų į registrą **DX**. Taikyti komandą **AAD**.
- 174.** Parašykite programą, kuri padalytų dvejetainį-dešimtainį skaičių 32 iš dvejetainio-dešimtainio skaičiaus 6 ir gautą rezultato ASCII kodą įrašytų į atmintį adresu DS:SI. Taikyti komandą **AAD**.
- 175.** Parašykite programą, kuri pervestų 8 skilčių glaudintą BCD skaičių, esantį registre **CH**, į 8 skilčių dvejetainį skaičių ir gautą rezultatą įrašytų į registrą **DX**. Taikyti komandą **AAD**.
- 176.** Parašykite programą, kuri skaičiuotų nurodytą reiškinį ir gautą rezultatą u įrašytų į atmintį adresu $0001:1000_{16}$. 16 bitų operandai x, y, z , ir w yra pasirinktose atminties ląstelėse. Operandams adresuoti taikyti skirtingus adresavimo būdus. Daugybai ar dalybai iš 2^n atlikti taikyti aritmetinio ir loginio poslinkio komandas.
- $u = 2x + 4y - 2z + w;$
 - $u = x - y / 2 + z - w / 8;$
 - $u = 8x - 4y - z / 4 + w;$
 - $u = 4x + y / 16 + 8z + w / 8;$

- e. $u = (x + y + z) - 2w;$
- f. $u = (x - y - z) + 4w;$
- g. $u = (x + y) / z - w;$
- h. $u = x / (y - z) + w;$
- i. $u = 4(x + y) + 2z + 4w;$
- j. $u = 2(x - y) - 4z + w / 2;$
- k. $u = x / 2 + (y - z) / 4 - w;$
- l. $u = (x + y) / 2 + z / 4 + w;$
- m. $u = (x + y) / 2 + z / 4 + (4w - 2) / 4;$
- n. $u = (2x - y) / 4 - z + w / 8;$
- o. $u = (x - y) / 8 + 2z - 4w + 3;$
- p. $u = 4x - 2(y - z) + w / 2 - 5;$
- q. $u = 2(x + y) - 6 + 4(z + w);$
- r. $u = x + 4y + 7 - (z + w) / 2;$
- s. $u = (2x + 3 - 4y) / 8 + (w - (x / 2)) / (2 - z);$
- t. $u = 4(x + 5) + (x + y) / (2 - 2w) + z / 4.$

177. Parašykite programą, kuri skaičiuotų nurodytą reiškinį ir gautą rezultatą u įrašytų į atmintį adresu 1202:1600₁₆. 16 bitų operandai x, y, z , ir w yra pasirinktose atminties ląstelėse. Operandams adresuoti taikyti skirtingus adresavimo būdus.

- a. $u = (x + y)^2 / z - w;$
- b. $u = (x - y) / w + z^2;$
- c. $u = x^2 / 2 + (y - z) / 4 - w^3;$
- d. $u = (x^2 + y) / 8 + z^2 / w + (4z - y) / 4;$
- e. $u = 2x^3 + (x^2y - 2zw) / 4 - 8x;$
- f. $u = (x^2 + y^2 + 4z^3) / 2 + (w + 2x - z^2) / 4;$
- g. $u = ((x + y^2)^3 + 2w) / 2 - z^2 / 2 + (2w + y) / 4;$
- h. $u = (2x^2 + y + z) / 2w + ((2x + z)^2 - w^2) / x^2;$
- i. $u = (xy) + (zw).$

178. Duotas vėlinimo paprogramis:

Žymė	Komandos mnemonika	Taktų skaičius
	MOV CX, XXXX	4
Z1:	DEC CX	2
	NOP	3
	JNZ Z1	16

Apskaičiuokite *XXXX*, kad gautumėte 50 ms vėlinimą, kai mikroprocesoriaus taktinis dažnis:

- $f_t = 5 \text{ MHz}$;
- $f_t = 8 \text{ MHz}$;
- $f_t = 10 \text{ MHz}$.

179. Pakeiskite 46 užduoties vėlinimo paprogramį, kad gautumėte:

- 1 s vėlinimą;
- 1 min vėlinimą;
- 5 min vėlinimą.

Priimti, kad mikroprocesoriaus taktinis dažnis $f_t = 10 \text{ MHz}$. Taikyti įterptus ciklus (žr. 112 užduoties sprendimą).

180. Parašykite programą, kuri iš atminties, pradedant adresu 54273_{16} , cikliška skaitytų 256 baitus informacijos ir siųstų į išorinį įtaisą. Išorinis įtaisas prijungtas prie mikroprocesorinės sistemos išvesties prievado, kurio adresas yra $1A_{16}$. Po kiekvieno 256 baitų skaitymo turi būti 1 s pauzė. Taikyti simbolių eilučių apdorojimo komandą **LODSB**.

181. Parašykite programą, kuri skaitmeninio-analoginio keitiklio išėjime generuotų 1 Hz sinusinį signalą. 360 baitų sinusinio signalo imčių yra atmintyje pradedant adresu 05200_{16} . Skaitmeninis-analoginis keitiklis prijungtas prie mikroprocesorinės sistemos išvesties prievado, kurio adresas yra $1D_{16}$. Taikyti simbolių eilučių apdorojimo komandą **LODSB**.

Pastaba: 1) vienam sinusinio signalo periodui sugeneruoti reikia persiųsti 360 baitų į skaitmeninio-analoginio keitiklio įėjimą; 2) 1 Hz sinusinio signalo periodas yra 1 s, tai reiškia, kad reikia 360 signalo imčių (baitų) persiųsti į skaitmeninio-analoginio keitiklio įėjimą per 1 s (1000 ms). Taigi laikas, reikalingas vienam baitui persiųsti, yra lygus: $1000 \text{ ms} / 360 \text{ baitų} = 2,78 \text{ ms} / \text{baitui}$.

182. Parašykite programą, kuri į septynių segmentų displėjų iš eilės išvestų šešioliktinius skaičius nuo 0 iki F. Kiekvienas skaičius septynių segmentų displėjuje turi šviesti 1 s. Septynių segmentų displėjus prijungtas prie mikroprocesorinės sistemos išvesties prievado, kurio adresas yra $1C_{16}$. Taikyti simbolių eilučių apdorojimo komandą **LODSB**.

183. Parašykite programą, kuri į išvesties prievadus, kurių adresai atitinkamai yra 01_{16} ir 02_{16} , vieno žodžio per sekundę greičiu persiųstų 1024 baitus informacijos. 1024 baitų duomenų blokas yra atmintyje pradedant adresu 11300_{16} . Taikyti simbolių eilučių apdorojimo komandą **LODSW**.

184. Parašykite programą, kuri nuskaitytų 64 baitus iš įvesties prievado, kurio adresas yra 25_{16} , ir įrašytų į atmintį pradedant adresu 12345_{16} . Taikyti simbolių eilučių apdorojimo komandą **STOSB** arba **STOSW**.

185. Parašykite programą, kuri įjungtų išorinį įtaisą, iš jo nuskaitytų 4 KB informacijos ir jį išjungtų. Išorinis įtaisas prijungtas prie mikroprocesorinės sistemos įvesties prievado, kurio adresas yra $1E_{16}$. Išorinis įtaisas valdomas mikroprocesorinės sistemos išvesties prievado, kurio adresas yra $1F_{16}$, pirmuoju išvadu, t. y., išvedus vienetą, įtaisas įjungiamas, išvedus nulį – išjungiamas. Nuskaitytus duomenis įrašyti į atmintį pradedant adresu 52340_{16} . Taikyti simbolių eilučių apdorojimo komandą **STOSB**.

186. Atmintyje, pradedant adresu 90008_{16} , yra 512 baitų masyvas. Parašykite programą, kuri atliktų skaičiaus 73_{16} paiešką masyve. Jei skaičius rastas, programa turi įrašyti surasto skaičiaus adresą į dėklą

ir vienetą į požymių registro **FL** pernašos skiltį. Jei skaičius nerastas, programa turi įrašyti nulį į požymių registro **FL** pernašos skiltį. Taikyti simbolių eilučių apdorojimo komandą **SCASB**.

- 187.** Atmintyje, pradedant adresu 4000_{16} , yra 1 KB masyvas. Parašykite programą, kuri atliktų skaičiaus $ABCD_{16}$ paiešką masyve. Jei skaičius rastas, programa turi įrašyti surasto skaičiaus adresą į registrą **BX** ir vienetus – į kaupiklį **AX**. Jei skaičius nerastas, programa turi įrašyti nulius į kaupiklį **AX**. Taikyti simbolių eilučių apdorojimo komandą **SCASW**.
- 188.** Atmintyje, pradedant adresu 13000_{16} , yra 768 spausdinamų simbolių ASCII kodų masyvas. Parašykite programą, kuri apskaičiuotų klausukų skaičių masyve. Gautą klausukų skaičių įrašyti į registrą **BX**. Klausuko ASCII kodas yra $3F_{16}$. Taikyti simbolių eilučių apdorojimo komandą **SCASB**.
- 189.** Atmintyje, pradedant adresais 73000_{16} ir 42000_{16} , yra du 256 baitų masyvai. Parašykite programą, kuri palygintų baitus duotuose masyvuose. Jei baitai yra vienodi, programa turi įrašyti vienetą į požymių registro **FL** pernašos skiltį. Jei baitai yra skirtingi, programa turi įrašyti nulį į požymių registro **FL** pernašos skiltį ir persiųsti pirmojo skirtingo baito adresą į dėklą nurodytu adresu. Taikyti simbolių eilučių apdorojimo komandą **CMPSB**.
- 190.** Atmintyje, pradedant adresais 73000_{16} ir 72000_{16} , yra du 1024 žodžių masyvai. Parašykite programą, kuri palygintų žodžius duotuose masyvuose. Jei žodžiai yra vienodi, programa turi išvesti vienetą į išvesties prievadą, kurio adresas yra 06_{16} . Jei žodžiai yra skirtingi, programa turi išvesti nulį į išvesties prievadą, kurio adresas yra 06_{16} , ir persiųsti paskutiniojo skirtingo žodžio adresą į dėklą nurodytu adresu. Taikyti simbolių eilučių apdorojimo komandą **CMPSW**.
- 191.** Atmintyje, pradedant adresais 45005_{16} ir $4700A_{16}$, yra du vienodo dydžio duomenų masyvai. Kiekvieno masyvo dydis baitais yra atminties ląstelėse, kurių adresai yra 45003_{16} ir 45004_{16} . Parašykite

programą, kuri duotuose masyvuose apskaičiuotų skirtingų baitų skaičių ir šį skaičių įrašytų į registrą **BP**. Taikyti simbolių eilučių apdorojimo komandą **CMPSB**.

- 192.** Atmintyje, pradedant adresu $4160:1500_{16}$, yra keturiasdešimties žodžių masyvas. Parašykite programą, kuri duoto masyvo žodžius išdėstytų atvirkščia tvarka pradedant adresu $4160:2000_{16}$, t. y. pirmasis masyvo žodis taptų paskutiniu, paskutinysis – pirmuoju ir t. t. Taikyti simbolių eilučių apdorojimo komandas **LODSW** ir **STOSW**.
- 193.** Atminties srityje nuo 65013_{16} iki 66000_{16} adreso yra du 64 baitų masyvai. Parašykite programą, kuri sukeistų vietomis duotus masyvus. Priimti, kad masyvai yra tame pačiame atminties segmente.
- 194.** Parašykite programą, kuri persiųstų 8 bitų operandus nuo 00_{16} iki FF_{16} į atmintį pradedant adresu 30000_{16} . Taikyti simbolių eilučių apdorojimo komandą **STOSB**.
- 195.** Parašykite programą, kuri persiųstų 8 bitų BCD skaičius nuo 00 iki 99 į atmintį pradedant adresu 40000_{16} . Taikyti simbolių eilučių apdorojimo komandą **STOSB** bei komandą **DAA**.
- 196.** Atmintyje, pradedant adresu 75000_{16} , yra šimtas 16 skilčių skaičių. Parašykite programą, kuri surastų didžiausią 16 skilčių skaičių ir įrašytų jį į registrą **DX**.
- 197.** Atmintyje, pradedant adresu 75000_{16} , yra šimtas 16 skilčių skaičių. Parašykite programą, kuri surastų mažiausią 16 skilčių skaičių ir įrašytų jį į atmintį adresu $7100:4400_{16}$.
- 198.** Parašykite programą, kuri skaičiuotų:
- $$\sum_{n=1}^{20} x_n^2,$$
- čia x_n – sveikieji 8 skilčių skaičiai atmintyje, pradedant adresu $4040:0340_{16}$.

Gautą rezultatą įrašyti į registrą **BX**. Taikyti simbolių eilučių apdorojimo komandą **LODSB**.

- 199.** Atmintyje, pradedant adresu $2000:0100_{16}$, yra šimto skaičių masyvas. Parašykite programą, kuri duotame masyve suskaičiuotų lyginius ir nelyginius skaičius, ir lyginių skaičių skaičių įrašytų į registrą **DX**, o nelyginių – į **BX**. Taikyti simbolių eilučių apdorojimo komandą **LODSW**.
- 200.** Atmintyje, pradedant adresu $0100:1100_{16}$, yra šimto skaičių masyvas. Parašykite programą, kuri duotame masyve suskaičiuotų teigiamus ir neigiamus skaičius ir teigiamų skaičių skaičių įrašytų į registrą **BX**, o neigiamų – į **DX**. Taikyti simbolių eilučių apdorojimo komandą **LODSW**.
- 201.** Parašykite programą, kuri 256 baitus atminties, pradedant adresu $2000:1200_{16}$, užpildytų šešioliktainiais skaičiais 00, 11, 22, 33,..., FF, 00, 11, 22, 33,..., FF, 00, 11, 22, 33,..., FF ir t. t. Taikyti simbolių eilučių apdorojimo komandą **STOSB**.

ATSAKYMAI

7. *a.* $A = FB_{16}$, $t = 15,5 \mu s$; *b.* $A = 6F_{16}$, $t = 20 \mu s$; *c.* $A = 00_{16}$, $t = 8,5 \mu s$; *d.* $A = 41_{16}$, $t = 13 \mu s$; *e.* $A = E2_{16}$, $t = 19 \mu s$; *f.* $A = 10_{16}$, $t = 28 \mu s$.
9. *a.* **PORT 05** = 05_{16} , $t = 62 \mu s$; *b.* $A = FF_{16}$, $t = 2,812 ms$; *c.* **PORT 05** = 01_{16} , $t = 0,189 s$; *d.* **PORT 05** = 77_{16} , $t = 74 \mu s$; *e.* $A = 0A_{16}$, $t = 123,5 \mu s$; *f.* **PORT 05** = FF_{16} , $t = 82,5 \mu s$.
42. *a.* $A = 3B_{16}$; *b.* $A = 27_{16}$; *c.* $A = E7_{16}$; *d.* $A = AF_{16}$, **CY** = 1; *e.* $L = 00_{16}$; *f.* $H = FF_{16}$, $L = 89_{16}$.
43. *a.* $A = BF_{16}$; *b.* $H = CE_{16}$, $L = 1B_{16}$; *c.* $D = 0B_{16}$, $E = 21_{16}$; *d.* $A = FC_{16}$, $H = 9F_{16}$, $L = 2B_{16}$; *e.* $B = D1_{16}$, $C = 49_{16}$; *f.* $H = CE_{16}$, $L = 7A_{16}$.
44. *a.* $A = 05_{16}$, $B = 0A_{16}$, $C = 05_{16}$; *b.* $A = 33_{16}$, $B = 24_{16}$, $C = 40_{16}$; *c.* $A = 03_{16}$, $B = FA_{16}$, $C = 02_{16}$; *d.* $A = 3B_{16}$, $B = 3B_{16}$, $C = 34_{16}$; *e.* $A = FF_{16}$, $B = 02_{16}$, $C = 00_{16}$; *f.* $A = 19_{16}$, $B = 19_{16}$, $C = 19_{16}$.
45. *a.* $2100_{16} = 22_{16}$, $2101_{16} = 33_{16}$, $2102_{16} = 00_{16}$, $2103_{16} = 33_{16}$; *b.* $2100_{16} = 00_{16}$, $2101_{16} = 11_{16}$, $2102_{16} = 11_{16}$, $2103_{16} = 00_{16}$; *c.* $2100_{16} = 00_{16}$, $2101_{16} = 00_{16}$, $2102_{16} = 00_{16}$, $2103_{16} = 00_{16}$; *d.* $2100_{16} = 22_{16}$, $2101_{16} = 33_{16}$, $2102_{16} = 00_{16}$, $2103_{16} = 11_{16}$; *e.* $2100_{16} = 00_{16}$, $2101_{16} = 11_{16}$, $2102_{16} = 22_{16}$, $2103_{16} = 33_{16}$; *f.* $2100_{16} = 00_{16}$, $2101_{16} = 11_{16}$, $2102_{16} = 11_{16}$, $2103_{16} = 00_{16}$.
46. *a.* $A = 10_{16}$, $H = 06_{16}$, $L = 08_{16}$, $2400_{16} = 08_{16}$, $2401_{16} = 06_{16}$, $2500_{16} = 08_{16}$, $2501_{16} = 06_{16}$; *b.* $A = 00_{16}$, $H = 0C_{16}$, $L = 02_{16}$, $2400_{16} = 02_{16}$, $2401_{16} = 0C_{16}$, $2500_{16} = 02_{16}$, $2501_{16} = 0C_{16}$; *c.* $A = 0A_{16}$, $H = 08_{16}$, $L = 07_{16}$, $2400_{16} = 07_{16}$, $2401_{16} = 08_{16}$, $2500_{16} = 07_{16}$, $2501_{16} = 08_{16}$.
80. *a.* **PORT 0F** = $(\cos(2z - 1)) + 1$; *b.* **PORT 0F** = $2(\cos(3z + 1))$; *c.* **PORT 0F** = $(\cos((z/4) + 5)) - 11$; *d.* **PORT 0F** = $(\cos 8z) + 16$; *e.* **PORT 0F** = $(\cos 4z) / 2$; *f.* **PORT 0F** = $(\cos((z - 3) / 2)) - 4$.

LITERATŪRA

- Antonakos, J. L. (2007). *The Intel Microprocessor Family: Hardware and Software Principles and Applications*. USA, New York: Thomson Delmar Learning. 618 p. ISBN 1-4180-3845-8.
- Berger, A. S. (2005). *Hardware and Computer Organization*. USA, Burlington: Newnes. 512 p. ISBN 0-7506-7886-0.
- Brey, B. B. (2009). *The Intel Microprocessors 8086/8088, 80186/80188, 80286, 80386, 80486, Pentium, Pentium Pro processor, Pentium II, Pentium III, Pentium 4, and Core2 with 64-bit extensions: Architecture, Programming, and Interfacing*. USA, New Jersey: Pearson Prentice Hall. 925 p. ISBN 0-13-502645-8.
- Deksnyš, V.; Jastramskas, V. (2000). *Įterptinės sistemos: vadovėlis*. 1 dalis. Kaunas: Technologija. 195 p. ISBN 9986-13-816-7.
- Ganguly, A. K. (2009). *Architecture, Programming and Applications of Advanced Microprocessors*. UK, Oxford: Alpha Science. 376 p. ISBN 1-84265-481-0.
- Godse, A. P.; Godse, D. A. (2010). *Microprocessor & Microcontroller*. India, Technical Publications Pune. 2008. ISBN 81-8431-765-7.
- Gražulevičius, G. (2008). *Mikroprocesorinė technika: mokomoji knyga*. I dalis. Vilnius: Technika. 224 p. ISBN 978-9955-28-280-8.
- Irvine, K. R. (2011). *Assembly Language for x86 Processors*. USA, New Jersey: Pearson Prentice Hall. 743 p. ISBN 0-13-137709-4.
- Matkevičius, E. (2009). *Mikroprocesoriniai valdymo įtaisai: mokomoji knyga*. I dalis. Vilnius: Technika. 176 p. ISBN 978-9955-28-478-9.
- Mazidi, M. A. (2010). *The x86 PC: Assembly Language, Design, and Interfacing*. USA, Boston: Pearson Prentice Hall. 786 p. ISBN 0-13-609226-8.
- Routt, W. A. (2007). *Microprocessor Architecture, Programming, and Systems Featuring the 8085*. USA, New York: Thomson Delmar Learning. 271 p. ISBN 1-4180-3241-7.
- Triebel, W. A. (2003). *The 8088 and 8086 Microprocessors: Programming, Interfacing, Software, Hardware and Applications*. USA, New Jersey: Pearson Prentice Hall. 1019 p. ISBN 0-13-093081-4.

1 PRIEDAS

Mikroprocesoriaus Intel® 8080 komandų sistema

Komandų aprašymuose taikomi sutartiniai žymenys:

- **R, R1, R2** – vieno iš registrų **A, B, C, D, E, H** arba **L** turinys (8 bitai);
- **M** – atminties ląstelė, kurios adresą nurodo registrų pora **H** (**H** ir **L** registrai);
- **M[HL]** – atminties ląstelės, kurios adresą nurodo registrų pora **H** (**H** ir **L** registrai), turinys (8 bitai);
- **D8** – 8 bitų tiesioginis operandas (antrasis komandos baitas);
- **D16** – 16 bitų tiesioginis operandas (antrasis ir trečiasis komandos baitai);
- **D16L** ir **D16H** – žemesnysis ir aukštesnysis 16 bitų tiesioginio operando baitai;
- **ADR** – operando 16 bitų adresas (antrasis ir trečiasis komandos baitai);
- **M[ADR]** – 8 bitų operandas, kurio 16 bitų adresas – antrasis ir trečiasis komandos baitai;
- **RP** – registrų poros **B** (**B** ir **C** registrai), **D** (**D** ir **E** registrai), **H** (**H** ir **L** registrai) arba dėklo rodyklės **SP** turinys (16 bitų);
- **RPL** ir **RPH** – žemesnysis ir aukštesnysis registrų poros registrai;
- **M[RP]** – atminties ląstelės, kurios adresą nurodo registrų pora **B** (**B** ir **C** registrai) arba **D** (**D** ir **E** registrai), turinys (8 bitai);
- **M[SP]** – dėklas;
- **SP** – 16 bitų dėklo rodyklė;
- **PC** – 16 bitų komandų skaitiklis;
- **PSW** – procesoriaus būsenos žodis (kaupiklio **A** ir požymių registro **F** turiniai (16 bitų));

- **PORT** – įvesties arba išvesties prievado 8 bitų numeris (adresas) (antrasis komandos baitas);
- **N** – pertraukties aptarnavimo paprogramio numeris;
- **n** – registro bito numeris (bitai numeruojami iš dešinės į kairę nuo 0 iki 7);
- **CY** – požymių registro **F** pernašos skiltis;
- **P** – požymių registro **F** lyginumo skiltis;
- **AC** – požymių registro **F** pagalbinės pernašos skiltis;
- **Z** – požymių registro **F** nulio skiltis;
- **S** – požymių registro **F** ženklo skiltis;
- $\wedge$ – loginis IR;
- $\vee$ – loginis ARBA;
- $\forall$ – loginis išskirtinis (griežtasis) ARBA;
- $\neg$ – inversija;
- $\leftarrow$ – perdavimas;
- $\leftrightarrow$ – sukeitimas.

Komanda, atliekama operacija, pavyzdys	Formatas ¹	B/C/T ²	Ciklai ³	AB ⁴	Keičiami požymiai ⁵				
					S	Z	AC	P	CY
					D ₇	D ₆	D ₄	D ₂	D ₀
1	2	3	4	5	6	7	8	9	10
Duomenų persiuntimo komandos									
Persiųsti (perkelti) registro turinį į kitą registrą (angl. <i>Move Register to Register</i>): MOV R1, R2. R1 ← R2 ; pvz., MOV B, C.	01dddsss	1/1/5	S	RA	–	–	–	–	–
Persiųsti (perkelti) atminties ląstelės, kurios adresą nurodo registrų pora H (H ir L registrai), turinį į registrą (angl. <i>Move Memory to Register</i>): MOV R, M. R ← M[HL] ; pvz., MOV A, M.	01ddd110	1/2/7	FR	ŠR	–	–	–	–	–
Persiųsti (perkelti) registro turinį į atminties ląstelę, kurios adresą nurodo registrų pora H (H ir L registrai) (angl. <i>Move Register to Memory</i>): MOV M, R. M[HL] ← R ; pvz., MOV M, E.	01110sss	1/2/7	FW	ŠR	–	–	–	–	–
Persiųsti (perkelti) 8 bitų tiesioginį operandą (antrąjį komandos baitą) į registrą (angl. <i>Move Immediate to Register</i>): MVI R, D8. R ← D8 ; pvz., MVI B, 25.	00ddd110 D8	2/2/7	FR	TO	–	–	–	–	–
Persiųsti (perkelti) 8 bitų tiesioginį operandą (antrąjį komandos baitą) į atminties ląstelę, kurios adresą nurodo registrų pora H (H ir L registrai) (angl. <i>Move Immediate to Memory</i>): MVI M, D8. M[HL] ← D8 ; pvz., MVI M, A5.	00110110 D8	2/3/10	FRW	TO	–	–	–	–	–
Persiųsti (įkrauti) 16 bitų tiesioginį operandą (antrąjį ir trečiąjį komandos baitus) į nurodytą registrų porą (angl. <i>Load Immediate to Register Pair</i>): LXI RP, D16. RPL ← D16L, RPH ← D16H ; pvz., LXI B, 2060.	00rp0001 D16L D16H	3/3/10	FRR	TO	–	–	–	–	–

1	2	3	4	5	6	7	8	9	10
Persiūsti (įkrauti) atminties ląstelės, kurios adresas nurodytas antrajame ir trečiajame komandos baituose, turinį į kaupiklį A (angl. <i>Load Accumulator Direct</i>): LDA ADR. A ← M[ADR] ; pvz., LDA 0900 .	00111010 ADRL ADRH	3/4/13	FRRR	TA	–	–	–	–	–
Persiūsti (išsaugoti) kaupiklio A turinį į atminties ląstelę, kurios adresas nurodytas antrajame ir trečiajame komandos baituose (angl. <i>Store Accumulator Direct</i>): STA ADR. M[ADR] ← A ; pvz., STA 0300 .	00110010 ADRL ADRH	3/4/13	FRRW	TA	–	–	–	–	–
Persiūsti (įkrauti) atminties ląstelių, iš kurių pirmosios adresas nurodytas antrajame ir trečiajame komandos baituose, turinius į registrus H ir L (angl. <i>Load H and L Direct</i>): LHLD ADR. L ← M[ADR] , H ← M[ADR+1] ; pvz., LHLD F005 .	00101010 ADRL ADRH	3/5/16	FRRRR	TA	–	–	–	–	–
Persiūsti (išsaugoti) registrų H ir L turinius į atminties ląsteles, iš kurių pirmosios adresas nurodytas antrajame ir trečiajame komandos baituose (angl. <i>Store H and L Direct</i>): SHLD ADR. M[ADR] ← L , M[ADR+1] ← H ; pvz., SHLD 1020 .	00100010 ADRL ADRH	3/5/16	FRRWW	TA	–	–	–	–	–
Persiūsti (įkrauti) atminties ląstelės, kurios adresą nurodo registrų pora B (B ir C registrai) arba D (D ir E registrai), turinį į kaupiklį A (angl. <i>Load Accumulator Indirect</i>): LDAX RP. A ← M[RP] ; pvz., LDAX B .	00rp1010	1/2/7	FR	ŠR	–	–	–	–	–
Persiūsti (išsaugoti) kaupiklio A turinį į atminties ląstelę, kurios adresą nurodo registrų pora B (B ir C registrai) arba D (D ir E registrai) (angl. <i>Store Accumulator Indirect</i>): STAX RP. M[RP] ← A ; pvz., STAX D .	00rp0010	1/2/7	FW	ŠR	–	–	–	–	–

1	2	3	4	5	6	7	8	9	10
Sukeisti registrų H ir L turinius su registrų D ir E turiniais (angl. <i>Exchange H and L with D and E</i>): XCHG. H ↔ D, L ↔ E.	11101011	1/1/4	F	RA	–	–	–	–	–
Aritmetinės komandos									
Pridėti registro turinį prie kaupiklio A turinio (angl. <i>Add Register to Accumulator</i>): ADD R. A ← A + R; pvz., ADD B.	10000sss	1/1/4	F	RA	+	+	+	+	+
Pridėti atminties ląstelės, kurios adresą nurodo registrų pora H (H ir L registrai), turinį prie kaupiklio A turinio (angl. <i>Add Memory to Accumulator</i>): ADD M. A ← A + M[HL]; pvz., ADD M.	10000110	1/2/7	FR	ŠR	+	+	+	+	+
Pridėti 8 bitų tiesioginį operandą (antrąjį komandos baitą) prie kaupiklio A turinio (angl. <i>Add Immediate to Accumulator</i>): ADI D8. A ← A + D8; pvz., ADI 05.	11000110 D8	2/2/7	FR	TO	+	+	+	+	+
Pridėti registro turinį ir požymių registro F pernašos skilties CY turinį prie kaupiklio A turinio (angl. <i>Add Register to Accumulator with Carry</i>): ADC R. A ← A + R + CY; pvz., ADC E.	10001sss	1/1/4	F	RA	+	+	+	+	+
Pridėti atminties ląstelės, kurios adresą nurodo registrų pora H (H ir L registrai), turinį ir požymių registro F pernašos skilties CY turinį prie kaupiklio A turinio (angl. <i>Add Memory to Accumulator with Carry</i>): ADC M. A ← A + M[HL] + CY; pvz., ADC M.	10001110	1/2/7	FR	ŠR	+	+	+	+	+
Pridėti 8 bitų tiesioginį operandą (antrąjį komandos baitą) ir požymių registro F pernašos skilties CY turinį prie kaupiklio A turinio (angl. <i>Add Immediate to Accumulator with Carry</i>): ACI D8. A ← A + D8 + CY; pvz., ACI 22.	11000110 D8	2/2/7	FR	TO	+	+	+	+	+

1	2	3	4	5	6	7	8	9	10
Atimti registro turinį iš kaupiklio A turinio (angl. <i>Subtract Register from Accumulator</i>): SUB R. A $\leftarrow$ A – R ; pvz., SUB D .	10010sss	1/1/4	F	RA	+	+	+	+	+
Atimti atminties ląstelės, kurios adresą nurodo registrų pora H (H ir L registrai), turinį iš kaupiklio A turinio (angl. <i>Subtract Memory from Accumulator</i>): SUB M. A $\leftarrow$ A – M[HL] ; pvz., SUB M .	10010110	1/2/7	FR	ŠR	+	+	+	+	+
Atimti 8 bitų tiesioginį operandą (antrąjį komandos baitą) iš kaupiklio A turinio (angl. <i>Subtract Immediate from Accumulator</i>): SUI D8. A $\leftarrow$ A – D8 ; pvz., SUI 01 .	11010110 D8	2/2/7	FR	TO	+	+	+	+	+
Atimti registro turinį kartu su požymių registro F pernašos skilties CY turiniu iš kaupiklio A turinio (angl. <i>Subtract Register from Accumulator with Borrow</i>): SBB R. A $\leftarrow$ A – (R + CY); pvz., SBB D .	10011sss	1/1/4	F	RA	+	+	+	+	+
Atimti atminties ląstelės, kurios adresą nurodo registrų pora H (H ir L registrai), turinį kartu su požymių registro F pernašos skilties CY turiniu iš kaupiklio A turinio (angl. <i>Subtract Memory from Accumulator with Borrow</i>): SBB M. A $\leftarrow$ A – (M[HL] + CY); pvz., SBB M .	10011110	1/2/7	FR	ŠR	+	+	+	+	+
Atimti 8 bitų tiesioginį operandą (antrąjį komandos baitą) kartu su požymių registro F pernašos skilties CY turiniu iš kaupiklio A turinio (angl. <i>Subtract Immediate from Accumulator with Borrow</i>): SBI D8. A $\leftarrow$ A – (D8 + CY); pvz., SBI 22 .	11011110 D8	2/2/7	FR	TO	+	+	+	+	+
Inkrementuoti registro turinį (angl. <i>Increment Register</i>): INR R. R $\leftarrow$ R + 1; pvz., INR B .	00ddd100	1/1/5	S	RA	+	+	+	+	–

1	2	3	4	5	6	7	8	9	10
Inkrementuoti atminties ląstelės, kurios adresą nurodo registrų pora H (H ir L registrai), turinį (angl. <i>Increment Memory</i>): INR M. M[HL] ← M[HL] + 1 ; pvz., INR M.	00110100	1/3/10	FRW	ŠR	+	+	+	+	–
Dekrementuoti registro turinį (angl. <i>Decrement Register</i>): DCR R. R ← R – 1 ; pvz., DCR C.	00ddd101	1/1/5	S	RA	+	+	+	+	–
Dekrementuoti atminties ląstelės, kurios adresą nurodo registrų pora H (H ir L registrai), turinį (angl. <i>Decrement Memory</i>): DCR M. M[HL] ← M[HL] – 1 ; pvz., DCR M.	00110101	1/3/10	FRW	ŠR	+	+	+	+	–
Inkrementuoti registrų poros turinį (angl. <i>Increment Register Pair</i>): INX RP. RP ← RP + 1 ; pvz., INX D.	00rp0011	1/1/5	S	RA	–	–	–	–	–
Dekrementuoti registrų poros turinį (angl. <i>Decrement Register Pair</i>): DCX RP. RP ← RP – 1 ; pvz., DCX H.	00rp1011	1/1/5	S	RA	–	–	–	–	–
Pridėti registrų poros turinį prie registrų poros H (H ir L registrai) turinio (16 bitų sudėtis) (angl. <i>Add Register Pair to H and L or Double Add</i>): DAD RP. HL ← HL + RP ; pvz., DAD B.	00rp1001	1/3/10	FEE	RA	–	–	–	–	+
Kaupiklio A turinio dešimtainė korekcija (angl. <i>Decimal Adjust Accumulator</i>): DAA. Jei A3–0 > 9 arba AC = 1 , tai A3–0 ← A3–0 + 6 , jei A7–4 > 9 arba CY = 1 , tai A7–4 ← A7–4 + 6 .	00100111	1/1/4	F	–	+	+	+	+	+
Loginės komandos									
Logiškai sudauginti registro turinį su kaupiklio A turiniu (angl. <i>Logical AND Register with Accumulator</i>): ANA R. A ← A ∧ R ; pvz., ANA H.	10100sss	1/1/4	F	RA	+	+	+	+	+

1	2	3	4	5	6	7	8	9	10
Logiškaiai sudauginti atminties ląstelės, kurios adresą nurodo registrų pora H (H ir L registrai), turinį su kaupiklio A turiniu (angl. <i>Logical AND Memory with Accumulator</i>): ANA M. $A \leftarrow A \wedge M[HL]$; pvz., ANA M.	10100110	1/2/7	FR	ŠR	+	+	+	+	+
Logiškaiai sudauginti 8 bitų tiesioginį operandą (antrąjį komandos baitą) su kaupiklio A turiniu (angl. <i>Logical AND Immediate with Accumulator</i>): ANI D8. $A \leftarrow A \wedge D8$; pvz., ANI 01.	11100110 D8	2/2/7	FR	TO	+	+	+	+	+
Logiškaiai sudėti registro turinį su kaupiklio A turiniu (angl. <i>Logical OR Register with Accumulator</i>): ORA R. $A \leftarrow A \vee R$; pvz., ORA B.	10110sss	1/1/4	F	RA	+	+	+	+	+
Logiškaiai sudėti atminties ląstelės, kurios adresą nurodo registrų pora H (H ir L registrai), turinį su kaupiklio A turiniu (angl. <i>Logical OR Memory with Accumulator</i>): ORA M. $A \leftarrow A \vee M[HL]$; pvz., ORA M.	10110110	1/2/7	FR	ŠR	+	+	+	+	+
Logiškaiai sudėti 8 bitų tiesioginį operandą (antrąjį komandos baitą) su kaupiklio A turiniu (angl. <i>Logical OR Immediate with Accumulator</i>): ORI D8. $A \leftarrow A \vee D8$; pvz., ORI 02.	11110110 D8	2/2/7	FR	TO	+	+	+	+	+
Logiškaiai sudėti modulių du registro turinį su kaupiklio A turiniu (angl. <i>Logical Exclusive OR Register with Accumulator</i>): XRA R. $A \leftarrow A \vee R$; pvz., XRA B.	10101sss	1/1/4	F	RA	+	+	+	+	+
Logiškaiai sudėti modulių du atminties ląstelės, kurios adresą nurodo registrų pora H (H ir L registrai), turinį su kaupiklio A turiniu (angl. <i>Logical Exclusive OR Memory with Accumulator</i>): XRA M. $A \leftarrow A \vee M[HL]$; pvz., XRA M.	10101110	1/2/7	FR	ŠR	+	+	+	+	+

1	2	3	4	5	6	7	8	9	10
Logiškai sudėti modulių du 8 bitų tiesioginį operandą (antrąjį komandos baitą) su kaupiklio A turiniu (angl. <i>Logical Exclusive OR Immediate with Accumulator</i>): XRI D8. A ← A ∨ D8 ; pvz., XRI 03 .	11101110 D8	2/2/7	FR	TO	+	+	+	+	+
Palyginti registro turinį su kaupiklio A turiniu (angl. <i>Compare Register with Accumulator</i>): CMP R. A – R ; pvz., CMP B .	10111sss	1/1/4	F	RA	+	+	+	+	+
Palyginti atminties ląstelės, kurios adresą nurodo registrų pora H (H ir L registrai), turinį su kaupiklio A turiniu (angl. <i>Compare Memory with Accumulator</i>): CMP M. A – M[HL] ; pvz., CMP M .	10111110	1/2/7	FR	ŠR	+	+	+	+	+
Palyginti 8 bitų tiesioginį operandą (antrąjį komandos baitą) su kaupiklio A turiniu (angl. <i>Compare Immediate with Accumulator</i>): CPI D8. A – D8 ; pvz., CPI 05 .	11111110 D8	2/2/7	FR	TO	+	+	+	+	+
Cikliškai pastumti kaupiklio A turinį į kairę per vieną skiltį (angl. <i>Rotate Accumulator Left</i>): RLC. An+1 ← An, n = 0–6, A0 ← A7, CY ← A7 .	00000111	1/1/4	F	–	–	–	–	–	+
Cikliškai pastumti kaupiklio A turinį į dešinę per vieną skiltį (angl. <i>Rotate Accumulator Right</i>): RRC. An ← An+1, n = 0–6, A7 ← A0, CY ← A0 .	00001111	1/1/4	F	–	–	–	–	–	+
Cikliškai pastumti kaupiklio A turinį ir požymių registro F pernašos skiltį CY į kairę per vieną skiltį (angl. <i>Rotate Accumulator Left through Carry</i>): RAL. An+1 ← An, n = 0–6, A0 ← CY, CY ← A7 .	00010111	1/1/4	F	–	–	–	–	–	+

1	2	3	4	5	6	7	8	9	10
Cikliškai pastumti kaupiklio A turinį ir požymių registro F pernašos skiltį CY į dešinę per vieną skiltį (angl. <i>Rotate Accumulator Right through Carry</i>): RAR. An $\leftarrow$ An+1 , n = 0–6 , A7 $\leftarrow$ CY , CY $\leftarrow$ A0 .	00011111	1/1/4	F	–	–	–	–	–	+
Invertuoti kaupiklio A turinį (angl. <i>Complement Accumulator</i>): CMA. A $\leftarrow$ $\neg$ A .	00101111	1/1/4	F	–	–	–	–	–	–
Invertuoti požymių registro F pernašos skilties CY reikšmę (angl. <i>Complement Carry</i>): CMC. CY $\leftarrow$ $\neg$ CY .	00111111	1/1/4	F	–	–	–	–	–	+
Nustatyti požymių registro F pernašos skilties CY reikšmę, lygią vienetui (angl. <i>Set Carry</i>): STC. CY $\leftarrow$ 1 .	00110111	1/1/4	F	–	–	–	–	–	+
Pereigų komandos									
Besąlygiškai pereiti ir vykdyti komandą, kurios adresą nurodo antrasis ir trečiasis komandos baitai (angl. <i>Unconditional Jump</i>): JMP ADR. PC $\leftarrow$ M[ADR] ; pvz., JMP 0600 .	11000011 ADRL ADRH	3/3/10	FRR	TA	–	–	–	–	–
Pereiti ir vykdyti komandą, kurios adresą nurodo antrasis ir trečiasis komandos baitai, jei skaičiavimo rezultatas nelygus nuliui (Z=0) (angl. <i>Jump on No Zero</i>): JNZ ADR. PC $\leftarrow$ M[ADR] ; pvz., JNZ 0A00 .	11000010 ADRL ADRH	3/3/10	FRR	TA	–	–	–	–	–
Pereiti ir vykdyti komandą, kurios adresą nurodo antrasis ir trečiasis komandos baitai, jei skaičiavimo rezultatas lygus nuliui (Z=1) (angl. <i>Jump on Zero</i>): JZ ADR. PC $\leftarrow$ M[ADR] ; pvz., JZ 2100 .	11001010 ADRL ADRH	3/3/10	FRR	TA	–	–	–	–	–

1	2	3	4	5	6	7	8	9	10
Pereiti ir vykdyti komandą, kurios adresą nurodo antrasis ir trečiasis komandos baitai, jei nėra pernašos ($CY=0$) (angl. <i>Jump on No Carry</i>): JNC ADR. PC $\leftarrow$ M[ADR] ; pvz., JNC 2150 .	11010010 ADRL ADRH	3/3/10	FRR	TA	–	–	–	–	–
Pereiti ir vykdyti komandą, kurios adresą nurodo antrasis ir trečiasis komandos baitai, jei yra pernaša ($CY=1$) (angl. <i>Jump on Carry</i>): JC ADR. PC $\leftarrow$ M[ADR] ; pvz., JC 20DE .	11011010 ADRL ADRH	3/3/10	FRR	TA	–	–	–	–	–
Pereiti ir vykdyti komandą, kurios adresą nurodo antrasis ir trečiasis komandos baitai, jei skaičiavimo rezultate vienetų skaičius nelyginis ($P=0$) (angl. <i>Jump on Parity Odd</i>): JPO ADR. PC $\leftarrow$ M[ADR] ; pvz., JPO 2A10 .	11100010 ADRL ADRH	3/3/10	FRR	TA	–	–	–	–	–
Pereiti ir vykdyti komandą, kurios adresą nurodo antrasis ir trečiasis komandos baitai, jei skaičiavimo rezultate vienetų skaičius lyginis ($P=1$) (angl. <i>Jump on Parity Even</i>): JPE ADR. PC $\leftarrow$ M[ADR] ; pvz., JPE 2FFF .	11101010 ADRL ADRH	3/3/10	FRR	TA	–	–	–	–	–
Pereiti ir vykdyti komandą, kurios adresą nurodo antrasis ir trečiasis komandos baitai, jei skaičiavimo rezultatas teigiamas ($S=0$) (angl. <i>Jump on Positive</i>): JP ADR. PC $\leftarrow$ M[ADR] ; pvz., JP 2205 .	11110010 ADRL ADRH	3/3/10	FRR	TA	–	–	–	–	–
Pereiti ir vykdyti komandą, kurios adresą nurodo antrasis ir trečiasis komandos baitai, jei skaičiavimo rezultatas neigiamas ($S=1$) (angl. <i>Jump on Minus</i>): JM ADR. PC $\leftarrow$ M[ADR] ; pvz., JM 3550 .	11111010 ADRL ADRH	3/3/10	FRR	TA	–	–	–	–	–
Besąlygiškai kreiptis į paprogramį, kurio pradžios adresą nurodo antrasis ir trečiasis komandos baitai (angl. <i>Unconditional Call</i>): CALL ADR. M[SP] $\leftarrow$ PC + 3 , SP $\leftarrow$ SP – 2 , PC $\leftarrow$ ADR ; pvz., CALL 0900 .	11001101 ADRL ADRH	3/5/17	SRRWW	TA	–	–	–	–	–

1	2	3	4	5	6	7	8	9	10
Kreiptis į paprogramį, kurio pradžios adresą nurodo antrasis ir trečiasis komandos baitai, jei skaičiavimo rezultatas nelygus nuliui ($Z=0$) (angl. <i>Call on No Zero</i>): CNZ ADR. M[SP] ← PC + 3, SP ← SP – 2, PC ← ADR; pvz., CNZ 0700.	11000100 ADRL ADRH	3/3/11 3/5/17	SRR SRRWW	TA	–	–	–	–	–
Kreiptis į paprogramį, kurio pradžios adresą nurodo antrasis ir trečiasis komandos baitai, jei skaičiavimo rezultatas lygus nuliui ($Z=1$) (angl. <i>Call on Zero</i>): CZ ADR. M[SP] ← PC + 3, SP ← SP – 2, PC ← ADR; pvz., CZ 1700.	11001100 ADRL ADRH	3/3/11 3/5/17	SRR SRRWW	TA	–	–	–	–	–
Kreiptis į paprogramį, kurio pradžios adresą nurodo antrasis ir trečiasis komandos baitai, jei nėra pernašos ($CY=0$) (angl. <i>Call on No Carry</i>): CNC ADR. M[SP] ← PC + 3, SP ← SP – 2, PC ← ADR; pvz., CNC 2340.	11010100 ADRL ADRH	3/3/11 3/5/17	SRR SRRWW	TA	–	–	–	–	–
Kreiptis į paprogramį, kurio pradžios adresą nurodo antrasis ir trečiasis komandos baitai, jei yra – pernašą ($CY=1$) (angl. <i>Call on Carry</i>): CC ADR. M[SP] ← PC + 3, SP ← SP – 2, PC ← ADR; pvz., CC 2110.	11011100 ADRL ADRH	3/3/11 3/5/17	SRR SRRWW	TA	–	–	–	–	–
Kreiptis į paprogramį, kurio pradžios adresą nurodo antrasis ir trečiasis komandos baitai, jei skaičiavimo rezultate vienetų skaičius nelyginis ($P=0$) (angl. <i>Call on Parity Odd</i>): CPO ADR. M[SP] ← PC + 3, SP ← SP – 2, PC ← ADR; pvz., CPO 1010.	11100100 ADRL ADRH	3/3/11 3/5/17	SRR SRRWW	TA	–	–	–	–	–
Kreiptis į paprogramį, kurio pradžios adresą nurodo antrasis ir trečiasis komandos baitai, jei skaičiavimo rezultate vienetų skaičius lyginis ($P=1$) (angl. <i>Call on Parity Even</i>): CPE ADR. M[SP] ← PC + 3, SP ← SP – 2, PC ← ADR; pvz., CPE 1E00.	11101100 ADRL ADRH	3/3/11 3/5/17	SRR SRRWW	TA	–	–	–	–	–
Kreiptis į paprogramį, kurio pradžios adresą nurodo antrasis ir trečiasis komandos baitai, jei skaičiavimo rezultatas teigiamas ($S=0$) (angl. <i>Call on Positive</i>): CP ADR. M[SP] ← PC + 3, SP ← SP – 2, PC ← ADR; pvz., CP 2E00.	11110100 ADRL ADRH	3/3/11 3/5/17	SRR SRRWW	TA	–	–	–	–	–

1	2	3	4	5	6	7	8	9	10
Kreiptis į paprogramį, kurio pradžios adresą nurodo antrasis ir trečiasis komandos baitai, jei skaičiavimo rezultatas neigiamas ($S=1$) (angl. <i>Call on Minus</i>): CM ADR . M [SP] $\leftarrow$ PC + 3, SP $\leftarrow$ SP - 2, PC $\leftarrow$ ADR ; pvz., CM A000 .	11111100 ADRL ADRH	3/3/11 3/5/17	SRR SRRWW	TA	–	–	–	–	–
Besąlygiškai grįžti iš paprogramio (angl. <i>Unconditional Return</i>): RET . PC $\leftarrow$ M [SP], SP $\leftarrow$ SP + 2.	11001001	1/3/10	FRR	ŠR	–	–	–	–	–
Grįžti iš paprogramio, jei skaičiavimo rezultatas nelygus nuliui ($Z=0$) (angl. <i>Return on No Zero</i>): RNZ . PC $\leftarrow$ M [SP], SP $\leftarrow$ SP + 2.	11000000	1/1/5 1/3/11	S SRR	ŠR	–	–	–	–	–
Grįžti iš paprogramio, jei skaičiavimo rezultatas lygus nuliui ($Z=1$) (angl. <i>Return on Zero</i>): RZ . PC $\leftarrow$ M [SP], SP $\leftarrow$ SP + 2.	11001000	1/1/5 1/3/11	S SRR	ŠR	–	–	–	–	–
Grįžti iš paprogramio, jei nėra pernašos ($CY=0$) (angl. <i>Return on No Carry</i>): RNC . PC $\leftarrow$ M [SP], SP $\leftarrow$ SP + 2.	11010000	1/1/5 1/3/11	S SRR	ŠR	–	–	–	–	–
Grįžti iš paprogramio, jei yra pernaša ($CY=1$) (angl. <i>Return on Carry</i>): RC . PC $\leftarrow$ M [SP], SP $\leftarrow$ SP + 2.	11011000	1/1/5 1/3/11	S SRR	ŠR	–	–	–	–	–
Grįžti iš paprogramio, jei skaičiavimo rezultate vienetų skaičius nelyginis ($P=0$) (angl. <i>Return on Parity Odd</i>): RPO . PC $\leftarrow$ M [SP], SP $\leftarrow$ SP + 2.	11100000	1/1/5 1/3/11	S SRR	ŠR	–	–	–	–	–
Grįžti iš paprogramio, jei skaičiavimo rezultate vienetų skaičius lyginis ($P=1$) (angl. <i>Return on Parity Even</i>): RPE . PC $\leftarrow$ M [SP], SP $\leftarrow$ SP + 2.	11101000	1/1/5 1/3/11	S SRR	ŠR	–	–	–	–	–

1	2	3	4	5	6	7	8	9	10
Grįžti iš paprogramio, jei skaičiavimo rezultatas teigiamas ($S=0$) (angl. <i>Return on Positive</i>): RP. PC $\leftarrow$ M[SP] , SP $\leftarrow$ SP + 2 .	11110000	1/1/5 1/3/11	S SRR	ŠR	–	–	–	–	–
Grįžti iš paprogramio, jei skaičiavimo rezultatas neigiamas ($S=1$) (angl. <i>Return on Positive</i>): RM. PC $\leftarrow$ M[SP] , SP $\leftarrow$ SP + 2 .	11111000	1/1/5 1/3/11	S SRR	ŠR	–	–	–	–	–
Kreiptis į pertraukčių aptarnavimo paprogramį (angl. <i>Restart</i>): RST N. PC $\leftarrow$ $8 \times N$; pvz., RST 1 .	11nnn111	1/3/11	SWW	ŠR	–	–	–	–	–
Persiųsti (perkelti) registrų poros H (H ir L registrai) turinį į komandų skaitiklį PC (angl. <i>Jump to H and L Indirect – Move H and L to Program Counter</i>): PCHL. PC $\leftarrow$ HL .	11101001	1/1/5	S	RA	–	–	–	–	–
Dėklo komandos									
Įrašyti registrų poros turinį į dėklą (angl. <i>Push Register Pair on Stack</i>): PUSH RP. M[SP – 1] $\leftarrow$ RPH , M[SP – 2] $\leftarrow$ RPL , SP $\leftarrow$ SP – 2 ; pvz., PUSH B .	11rp0101	1/3/11	SWW	ŠR	–	–	–	–	–
Įrašyti procesoriaus būsenos žodį (kaupiklio A ir požymių registro F turinius) į dėklą (angl. <i>Push Processor Status Word on Stack</i>): PUSH PSW. M[SP – 1] $\leftarrow$ A , M[SP – 2] $\leftarrow$ F , SP $\leftarrow$ SP – 2 .	11110001	1/3/11	SWW	ŠR	–	–	–	–	–
Skaityti registrų poros turinį iš dėklo (angl. <i>Pop Register Pair off Stack</i>): POP RP. RPL $\leftarrow$ M[SP] , RPH $\leftarrow$ M[SP + 1] , SP $\leftarrow$ SP + 2 ; pvz., POP D .	11rp0001	1/3/10	FRR	ŠR	–	–	–	–	–
Skaityti procesoriaus būsenos žodį (kaupiklio A ir požymių registro F turinius) iš dėklo (angl. <i>Pop Processor Status Word off Stack</i>): POP PSW. F $\leftarrow$ M[SP] , A $\leftarrow$ M[SP + 1] , SP $\leftarrow$ SP + 2 .	11110001	1/3/10	FRR	ŠR	+	+	+	+	+

1	2	3	4	5	6	7	8	9	10
Sukeisti dėklo aukštesniųjų ląstelių turinius su registru poros H (H ir L registrai) turiniu (angl. <i>Exchange Top of Stack with H and L</i>): XTHL. L $\leftrightarrow$ M[SP] , H $\leftrightarrow$ M[SP + 1] .	11100011	1/5/18	FRRWW	ŠR	–	–	–	–	–
Persiųsti (perkelti) registru poros H (H ir L registrai) turinį į dėklo rodyklę SP (angl. <i>Move HL to Stack Pointer</i>): SPHL. SP $\leftarrow$ HL .	11111001	1/1/5	S	RA	–	–	–	–	–
Įvesties ir išvesties komandos									
Įvesti duomenų baitą per įvesties prievadą, kurio numeris (adresas) nurodytas antrajame komandos baite, į kaupiklį A (angl. <i>Input</i>): IN PORT. A $\leftarrow$ PORT ; pvz., IN 05 .	11011011 PORT	2/3/10	FRI	TA	–	–	–	–	–
Išvesti kaupiklio A turinį per išvesties prievadą, kurio numeris (adresas) nurodytas antrajame komandos baite (angl. <i>Output</i>): OUT PORT. PORT $\leftarrow$ A ; pvz., OUT 0F .	11010011 PORT	2/3/10	FRO	TA	–	–	–	–	–
Mikroprocesoriaus valdymo komandos									
Leisti pertrauktis (angl. <i>Enable Interrupts</i>): EI .	11111011	1/1/4	F	–	–	–	–	–	–
Drausti pertrauktis (angl. <i>Disable Interrupts</i>): DI .	11110011	1/1/4	F	–	–	–	–	–	–
Stabdyti mikroprocesoriaus darbą (angl. <i>Halt</i>): HLT .	01110110	1/2/7	FE	–	–	–	–	–	–
Nėra operacijos (angl. <i>No Operation</i>), t. y. „tuščia“ keturių taktų trukmės komanda keičianti tik komandų skaitiklį PC : NOP. PC $\leftarrow$ PC + 1 .	00000000	1/1/4	F	–	–	–	–	–	–

¹ sss – registro, iš kurio siunčiami duomenys, dvejetainis adresas komandoje: **B** = 000, **C** = 001, **D** = 010, **E** = 011, **H** = 100, **L** = 101, **M** = 110, **A** = 111; ddd – registro, į kurį siunčiami duomenys, dvejetainis adresas komandoje: **B** = 000, **C** = 001, **D** = 010, **E** = 011, **H** = 100, **L** = 101, **M** = 110, **A** = 111; ADRL ir ADRH – žemesnysis ir aukštesnysis 16 bitų adreso, kuriame yra operandas, baitai (antrasis ir trečiasis komandos baitai); rp – registrų poros dvejetainis adresas komandoje: **BC** = 00, **DE** = 01, **HL** = 10, **SP** = 11); nnn – pertraukties aptarnavimo paprogramio dvejetainis numeris komandoje: **RST 0** = 000, **RST 1** = 001, **RST 2** = 010, **RST 3** = 011, **RST 4** = 100, **RST 5** = 101 **RST 6** = 110, **RST 7** = 111.

² B – komandos ilgis baitais; C – procesoriaus ciklų skaičius; T – komandos vykdymo trukmė taktiniais intervalais arba taktais.

³ F – komandos išrinkimo ciklas, trunkantis keturis taktinius intervalus arba taktus; S – komandos išrinkimo ciklas, trunkantis penkis taktinius intervalus arba taktus; R – skaitymo iš atminties ciklas; W – rašymo į atmintį ciklas; I – įvesties iš išorinio įtaiso ciklas; O – išvesties į išorinį įtaisą ciklas; E – „tuščias“ procesoriaus ciklas.

⁴ AB – operandų adresavimo būdas; RA – registrinis operandų adresavimo būdas; ŠR – šalutinis registrinis operandų adresavimo būdas; TO – tiesioginis operandas; TA – tiesioginis operandų adresavimo būdas; - – neapibrėžtas operandų adresavimo būdas.

⁵ + – keičiama požymių registro **F** skilties reikšmė; - – nekeičiama požymių registro **F** skilties reikšmė.

Mikroprocesoriaus Intel® 8080

komandų grupės ir šešioliktiniai kodai

Duomenų persiuntimo komandos

Persiūsti (perkelti)		Persiūsti (perkelti)		Persiūsti (perkelti) tiesioginį operandą				
MOV	{ A, A	7F	MOV	{ E, A	5F	MVI	{ A, D8	3E
	{ A, B	78		{ E, B	58		{ B, D8	06
	{ A, C	79		{ E, C	59		{ C, D8	0E
	{ A, D	7A		{ E, D	5A		{ D, D8	82
	{ A, E	7B		{ E, E	5B		{ E, D8	1E
	{ A, H	7C		{ E, H	5C		{ H, D8	26
	{ A, L	7D		{ E, L	5D		{ L, D8	2E
	{ A, M	7E		{ E, M	5E		{ M, D8	36
				Persiūsti (įkrauti) tiesioginį operandą				
MOV	{ B, A	47	MOV	{ H, A	67	LXI	{ B, D16	01
	{ B, B	40		{ H, B	60		{ D, D16	11
	{ B, C	41		{ H, C	61		{ H, D16	21
	{ B, D	42		{ H, D	62		{ SP, D16	31
	{ B, E	43		{ H, E	63	Persiūsti (įkrauti)		
	{ B, H	44		{ H, H	64	LDAX B	0A	
	{ B, L	45		{ H, L	65	LDAX D	1A	
	{ B, M	46		{ H, M	66	LHLD ADR	2A	
MOV	{ C, A	4F	MOV	{ L, A	6F	LDA ADR	3A	
	{ C, B	48		{ L, B	68	Persiūsti (išsaugoti)		
	{ C, C	49		{ L, C	69	STAX B	02	
	{ C, D	4A		{ L, D	6A	STAX D	12	
	{ C, E	4B		{ L, E	6B	SHLD ADR	22	
	{ C, H	4C		{ L, H	6C	STA ADR	32	
	{ C, L	4D		{ L, L	6D	Sukeisti		
	{ C, M	4E		{ L, M	6E	XCHG	EB	
MOV	{ D, A	57	MOV	{ M, A	77			
	{ D, B	50		{ M, B	70			
	{ D, C	51		{ M, C	71			
	{ D, D	52		{ M, D	72			
	{ D, E	53		{ M, E	73			
	{ D, H	54		{ M, H	74			
	{ D, L	55		{ M, L	75			
	{ D, M	56						

Aritmetinės komandos

Pridėti ¹	Atimti ¹	Inkrementuoti ³
ADD { A 87 B 80 C 81 D 82 E 83 H 84 L 85 M 86	SUB { A 97 B 90 C 91 D 92 E 93 H 94 L 95 M 96	INX { B 03 D 13 H 23 SP 33
Pridėti tiesioginį Operandą ¹	Atimti tiesioginį Operandą ¹	Dekrementuoti ³
ADI D8 C6	SUI D8 D6	DCR { A 3D B 05 C 0D D 15 E 1D H 25 L 2D M 35
Pridėti su pernaša ¹	Atimti su paskola ¹	DCX { B 0B D 1B H 2B SP 3B
ADC { A 8F B 88 C 89 D 8A E 8B H 8C L 8D M 8E	SBB { A 9F B 98 C 99 D 9A E 9B H 9C L 9D M 9E	Dešimtainė korekcija ¹
Pridėti tiesioginį operandą su pernaša ¹	Atimti tiesioginį operandą su paskola ¹	DAA 27
ACI D8 CE	SBI D8 DE	
Pridėti 16 bitų ²	Inkrementuoti ³	
DAD { B 09 D 19 H 29 SP 39	INR { A 3C B 04 C 0C D 14 E 1C H 24 L 2C M 34	

¹ Keičiamos požymių registro F skilčių CY, P, AC, Z ir S reikšmės.² Keičiama požymių registro F skilties CY reikšmė.³ Keičiamos požymių registro F skilčių P, AC, Z ir S reikšmės (išimtis: komandos INX ir DCX nekeičia jokių požymių registro F skilčių reikšmių).

Loginės komandos

Logiškai sudauginti ⁴		Logiškai sudėti modulių du ⁴		Palyginti su tiesioginiu operandu ⁴			
ANA	A	A7	XRA	A	AF	CPI D8	FE
	B	A0		B	A8	Cikliškai pastumti ⁵	
	C	A1		C	A9	RLC	07
	D	A2		D	AA	RRC	0F
	E	A3		E	AB	RAL	17
	H	A4		H	AC	RAR	1F
	L	A5		L	AD	Invertuoti ⁵	
M	A6	M	AE	CMA	2F		
Logiškai sudauginti su tiesioginiu operandu ⁴		Logiškai sudėti modulių du su tiesioginiu operandu ⁴		Cikliškai pastumti ⁵			
ANI D8 E6		XRI D8 EE		RLC 07			
Logiškai sudėti ⁴		Palyginti ⁴		RRC 0F			
ORA	A	B7	CMP	A	BF	RAL 17	
	B	B0		B	B8	RAR 1F	
	C	B1		C	B9	Invertuoti ⁵	
	D	B2		D	BA	CMA 2F	
	E	B3		E	BB	CMC 3F	
	H	B4		H	BC	Nustatyti ⁵	
	L	B5		L	BD	STC 37	
M	B6	M	BE				
Logiškai sudėti su tiesioginiu operandu ⁴							
ORI D8 F6							

⁴ Keičiamos požymių registro **F** skilčių **CY**, **P**, **AC**, **Z** ir **S** reikšmės.

⁵ Keičiama požymių registro **F** skilties **CY** reikšmė (išimtis: komanda **CMA** nekeičia jokių požymių registro **F** skilčių reikšmių).

Pereigų komandos

Pereiti		Grįžti iš paprogramio		Kreiptis į pertraukčių aptarnavimo paprogramį
JMP ADR	C3	RET	C9	
JNZ ADR	C2	RNZ	C0	
JZ ADR	CA	RZ	C8	
JNC ADR	D2	RNC	D0	
JC ADR	DA	RC	D8	
JPO ADR	E2	RPO	E0	
JPE ADR	EA	RPE	E8	
JP ADR	F2	RP	F0	
JM ADR	FA	RM	F8	
PCHL	E9			
				RST {
				0 C7
				1 CF
				2 D7
				3 DF
				4 E7
				5 EF
				6 F7
				7 FF
Pertraukčių aptarnavimo paprogramių adresai				
		Komanda	Komandos kodas ₁₆	Paprogramio adresas ₁₆
Kreiptis į paprogramį				
CALL ADR	CD			
CNZ ADR	C4	RST 0	C7	0000
CZ ADR	CC	RST 1	CF	0008
CNC ADR	D4	RST 2	D7	0010
CC ADR	DC	RST 3	DF	0018
CPO ADR	E4	RST 4	E7	0020
CPE ADR	EC	RST 5	EF	0028
CP ADR	F4	RST 6	F7	0030
CM ADR	FC	RST 7	FF	0038

Dėklo komandos

Įrašyti į dėklą		Skaityti iš dėklo ⁶		Sukeisti
PUSH {	B C5	POP {	B C1	XTHL E3
	D D5		D D1	Persiųsti (perkelti)
	H E5		H E1	SPHL F9
	PSW F5		PSW F1	

Įvesties ir išvesties bei mikroprocesoriaus valdymo komandos

Įvesti per prievadą		Leisti pertrauktis		Stabdyti (sustoti)
IN PORT	DB	EI	FB	HLT 76
Išvesti per prievadą		Drausti pertrauktis		Nėra operacijos
OUT PORT	D3	DI	F3	NOP 00

⁶ Keičiamos požymių registro **F** skilčių **CY**, **P**, **AC**, **Z** ir **S** reikšmės (išimtis: komandos **POP B**, **POP D** ir **POP H** nekeičia jokių požymių registro **F** skilčių reikšmių).

2 PRIEDAS

Mikroprocesoriaus Intel® 8086 pagrindinės komandos

Komandų aprašymuose taikomi sutartiniai žymenys:

- **OKB** – operacijos kodo baitas (pirmasis komandos baitas);
- **AB** – adresavimo baitas (antrasis komandos baitas), jo gali ir nebūti;
- **r** – 8 arba 16 bitų operandai registruose;
- **m** – 8 arba 16 bitų operandai atmintyje;
- **imm** – 8 arba 16 bitų tiesioginis operandas;
- **moffs** – 8 arba 16 bitų komandoje adresuojami operandai, netaikant adresavimo baito;
- **sreg** – segmento registras;
- **acc** – kaupiklis;
- **port** – prievado numeris (adresas);
- **d** – operacijos kodo baito bitas nurodantis duomenų perdavimo kryptį. Jeigu **d** = 1, operandas perduodamas į registrą, nustatomą lauke **reg**, t. y. kai **d** = 1, registras yra imtuvas, kai **d** = 0, registras yra siųstuvas;
- **w** – operacijos kodo baito žemiausias bitas, nurodantis operandų, su kuriais atliekamos operacijos, bitų skaičių. Jeigu **w** = 1, operacijos vykdomos su 16 bitų operandais, kai **w** = 0–8 bitų operandais;
- **s** – operacijos kodo baito bitas nurodantis tiesioginio operando tipą. Jeigu **sw** = 01, 16 bitų tiesioginis operandas yra teigiamasis sveikasis skaičius nuo 0 iki 65535. Kai **sw** = 11, tiesioginis operandas yra 8 bitų, užrašomas papildomu kodu ir gali reikšti skaičius nuo –128 iki +127;

- **mod** – dviejų skilčių adresavimo baido laukas, nurodantis operandų adresavimo būdą. Nuo jo priklauso, kaip bus interpretuojamas trijų skilčių adresavimo baido laukas **r/m**, kuris gali reikšti arba registrą, arba atminties ląstelės adresą;
- **reg** – trijų skilčių adresavimo baido laukas, nurodantis registrą kaip vieną iš komandos operandų;
- **sr** – segmento registro dvejetainis adresas;
- **rel8, rel16** – santykinis adresas, kuris pereigų komandose pridedamas prie komandų rodyklėje **IP** esančio esamos komandos adreso;
- **r16/m16** – registras arba atminties adresas, kuriuose yra pereigos adresas;
- **ptr16:16** – 4 baitų tolima (angl. *Far*) rodyklė į pereigos adresą segmentas:poslinkis;
- **m16:16** – tolima rodyklė, kuri yra nurodytame atminties adrese segmentas:poslinkis;
- **Disp L** ir **Disp H** – tiesioginio adreso arba poslinkio, segmento pradžios atžvilgiu, žemesnysis ir aukštesnysis baitai;
- **Data L** ir **Data H** – tiesioginio operando žemesnysis ir aukštesnysis baitai;
- **Seg L** ir **Seg H** – segmento adreso žemesnysis ir aukštesnysis baitai;
- **()** – nebūtinai komandos kodo baitai;
- **/** – arba.

Mnemonika ir aprašymas	Komandos kodas					
	76543210	76543210	76543210	76543210	76543210	76543210
	OKB	AB				
1	2	3	4	5	6	7
Duomenų persiuntimo komandos						
MOV						
r, r/m; r/m, r	100010dw	mod reg r/m	(Disp L)	(Disp H)		
r/m, imm	1100011w	mod 000 r/m	(Disp L)	(Disp H)	Data L	(Data H)
r, imm	1011wreg	Data L	(Data H)			
acc, moffs	1010000w	Disp L	Disp H			
moffs, acc	1010001w	Disp L	Disp H			
sreg, r/m	10001110	mod 0sr r/m	(Disp L)	(Disp H)		
r/m, sreg	10001100	mod 0sr r/m	(Disp L)	(Disp H)		
PUSH						
r/m	11111111	mod 110 r/m	(Disp L)	(Disp H)		
r	01010reg					
sreg	000sr110					
PUSHF						
POP						
r/m	10001111	mod 000 r/m	(Disp L)	(Disp H)		
r	01011reg					
sreg	000sr111					
POPF						
XCHG						
r, r/m; r/m, r	1000011w	mod reg r/m	(Disp L)	(Disp H)		
acc, r; r, acc	10010reg					
IN						
acc, port	1110010w	port				
acc, DX	1110110w					
OUT						
port, acc	1110011w	port				
DX, acc	1110111w					
Aritmetinės komandos						
ADD						
r, r/m; r/m, r	000000dw	mod reg r/m	(Disp L)	(Disp H)		
r/m, imm	100000sw	mod 000 r/m	(Disp L)	(Disp H)	Data L	(Data H)
r/m, imm	100000sw	mod 000 r/m	Data L			
acc, imm	0000010w	Data L	(Data H)			
ADC						
r, r/m; r/m, r	000100dw	mod reg r/m	(Disp L)	(Disp H)		
r/m, imm	100000sw	mod 010 r/m	(Disp L)	(Disp H)	Data L	(Data H)
r/m, imm	100000sw	mod 010 r/m	Data L			
acc, imm	0001010w	Data L	(Data H)			
SUB						
r, r/m; r/m, r	001010dw	mod reg r/m	(Disp L)	(Disp H)		
r/m, imm	100000sw	mod 101 r/m	(Disp L)	(Disp H)	Data L	(Data H)
r/m, imm	100000sw	mod 101 r/m	Data L			
acc, imm	0010110w	Data L	(Data H)			

1	2	3	4	5	6	7
SBB						
r, r/m; r/m, r	000110dw	mod reg r/m	(Disp L)	(Disp H)		
r/m, imm	100000sw	mod 011 r/m	(Disp L)	(Disp H)	Data L	(Data H)
r/m, imm	100000sw	mod 011 r/m	Data L			
acc, imm	0001110w	Data L	(Data H)			
INC						
r/m	1111111w	mod 000 r/m	(Disp L)	(Disp H)		
r	01000reg					
DEC						
r/m	1111111w	mod 001 r/m	(Disp L)	(Disp H)		
r	01001reg					
CMP						
r, r/m; r/m, r	001110dw	mod reg r/m	(Disp L)	(Disp H)		
r/m, imm	100000sw	mod 111 r/m	(Disp L)	(Disp H)	Data L	(Data H)
r/m, imm	100000sw	mod 111 r/m	Data L			
acc, imm	0011110w	Data L	(Data H)			
NEG						
r/m	1111011w	mod 011 r/m	(Disp L)	(Disp H)		
AAA						
	00110111					
DAA						
	00100111					
AAS						
	00111111					
DAS						
	00101111					
AAM						
	11010100	00001010				
AAD						
	11010101	00001010				
MUL						
r/m	1111011w	mod 100 r/m	(Disp L)	(Disp H)		
IMUL						
r/m	1111011w	mod 101 r/m	(Disp L)	(Disp H)		
DIV						
r/m	1111011w	mod 110 r/m	(Disp L)	(Disp H)		
IDIV						
r/m	1111011w	mod 111 r/m	(Disp L)	(Disp H)		
CBW						
	10011000					
CWD						
	10011001					
Loginės komandos						
AND						
r, r/m; r/m, r	001000dw	mod reg r/m	(Disp L)	(Disp H)		
r/m, imm	1000000w	mod 100 r/m	(Disp L)	(Disp H)	Data L	(Data H)
acc, imm	0010010w	Data L	(Data H)			
OR						
r, r/m; r/m, r	000010dw	mod reg r/m	(Disp L)	(Disp H)		
r/m, imm	1000000w	mod 001 r/m	(Disp L)	(Disp H)	Data L	(Data H)
acc, imm	0000110w	Data L	(Data H)			
XOR						
r, r/m; r/m, r	001100dw	mod reg r/m	(Disp L)	(Disp H)		
r/m, imm	1000000w	mod 110 r/m	(Disp L)	(Disp H)	Data L	(Data H)
acc, imm	0011010w	Data L	(Data H)			

1	2	3	4	5	6	7
TEST						
r/m, r	1000010w	mod reg r/m	(Disp L)	(Disp H)		
r/m, imm	1111011w	mod 000 r/m	(Disp L)	(Disp H)	Data L	(Data H)
acc, imm	1010100w	Data L	(Data H)			
NOT						
r/m	1111011w	mod 010 r/m	(Disp L)	(Disp H)		
SHL / SAL						
r/m, 1	1101000w	mod 100 r/m	(Disp L)	(Disp H)		
r/m, CL	1101001w	mod 100 r/m	(Disp L)	(Disp H)		
SHR						
r/m, 1	1101000w	mod 101 r/m	(Disp L)	(Disp H)		
r/m, CL	1101001w	mod 101 r/m	(Disp L)	(Disp H)		
SAR						
r/m, 1	1101000w	mod 111 r/m	(Disp L)	(Disp H)		
r/m, CL	1101001w	mod 111 r/m	(Disp L)	(Disp H)		
ROL						
r/m, 1	1101000w	mod 000 r/m	(Disp L)	(Disp H)		
r/m, CL	1101001w	mod 000 r/m	(Disp L)	(Disp H)		
ROR						
r/m, 1	1101000w	mod 001 r/m	(Disp L)	(Disp H)		
r/m, CL	1101001w	mod 001 r/m	(Disp L)	(Disp H)		
RCL						
r/m, 1	1101000w	mod 010 r/m	(Disp L)	(Disp H)		
r/m, CL	1101001w	mod 010 r/m	(Disp L)	(Disp H)		
RCR						
r/m, 1	1101000w	mod 011 r/m	(Disp L)	(Disp H)		
r/m, CL	1101001w	mod 011 r/m	(Disp L)	(Disp H)		
Pereigų komandos						
CALL						
rel16	11101000	Disp L	Disp H			
r16/m16	11111111	mod 010 r/m	(Disp L)	(Disp H)		
ptr16:16	10011010	Disp L	Disp H	Seg L	Seg H	
m16:16	11111111	mod 011 r/m	(Disp L)	(Disp H)		
JMP						
rel8	11101011	Disp L				
rel16	11101001	Disp L	Disp H			
r16/m16	11111111	mod 100 r/m	(Disp L)	(Disp H)		
ptr16:16	11101010	Disp L	Disp H	Seg L	Seg H	
m16:16	11111111	mod 101 r/m	(Disp L)	(Disp H)		
RET						
(near)	11000011					
imm (near)	11000010	Data L	Data H			
(far)	11001011					
imm (far)	11001010	Data L	Data H			
JE / JZ						
rel8	01110100	Disp L				

1	2	3	4	5	6	7
JL / JNGE						
rel8	01111100	Disp L				
JLE / JNG						
rel8	01111110	Disp L				
JB / JNAE						
rel8	01110010	Disp L				
JBE / JNA						
rel8	01110110	Disp L				
JP / JPE						
rel8	01111010	Disp L				
JO						
rel8	01110000	Disp L				
JS						
rel8	01111000	Disp L				
JNE / JNZ						
rel8	01110101	Disp L				
JNL / JGE						
rel8	01111101	Disp L				
JNLE / JG						
rel8	01111111	Disp L				
JNB / JAE						
rel8	01110011	Disp L				
JNBE / JA						
rel8	01110111	Disp L				
JNP / JPO						
rel8	01111011	Disp L				
JNO						
rel8	01110001	Disp L				
JNS						
rel8	01111001	Disp L				
LOOP	11100010	Disp L				
LOOPZ /						
LOOPE						
rel8	11100001	Disp L				
LOOPNZ /						
LOOPNE						
rel8	11100000	Disp L				
JCXZ						
rel8	11100011	Disp L				
INT	11001101					
INT 3	11001100					
INTO	11001110					
IRET	11001111					
Mikroprocesoriaus valdymo komandos						
CLC	11111000					
CMC	11110101					

1	2	3	4	5	6	7
STC	11111001					
CLD	11111100					
STD	11111101					
CLI	11111010					
STI	11111011					
HLT	11110100					
WAIT	10011011					
Simbolių eilučių apdorojimo komandos						
REP /						
REPE /						
REPZ	11110011					
MOVSB /						
MOVSW	1010010w					
CMPSB /						
CMPSW	1010011w					
SCASB /						
SCASW	1010111w					
LODS /						
LODSW	1010110w					
STOSB /						
STOSW	1010101w					

Mikroprocesoriaus Intel® 8086 registrų adresai

Laukas reg arba r/m	mod = 11		
	w = 0	w = 1	Segmentų registrai
000	AL	AX	ES
001	CL	CX	CS
010	DL	DX	SS
011	BL	BX	DS
100	AH	SP	–
101	CH	BP	–
110	DH	SI	–
111	BH	DI	–

Operandų adresavimo būdai

r/m	Efektyvusis adresas		
	mod = 00	mod = 01	mod = 10
000	BX + SI	BX + SI + Disp L	BX + SI + Disp H Disp L
001	BX + DI	BX + DI + Disp L	BX + DI + Disp H Disp L
010	BP + SI	BP + SI + Disp L	BP + SI + Disp H Disp L
011	BP + DI	BP + DI + Disp L	BP + DI + Disp H Disp L
100	SI	SI + Disp L	SI + Disp H Disp L
101	DI	DI + Disp L	DI + Disp H Disp L
110	Disp H Disp L	BP + Disp L	BP + Disp H Disp L
111	BX	BX + Disp L	BX + Disp H Disp L