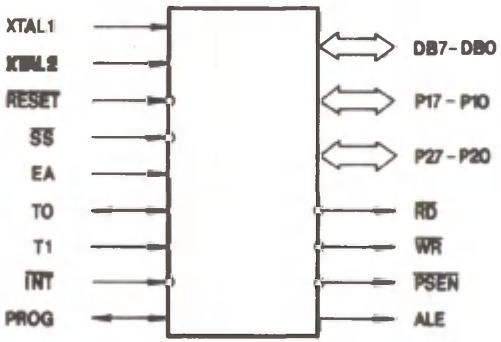
TARTU RIIKLIK ÜLIKOOL

MONOLIITARVUTI KM1816BE48

KIRJELDUS



TARTU 1988

TARTU RIIKLIK ÜLIKOOL
ÜKTU BÜROO

MONOLIITARVUTI KM1816BE48

KIRJELDUS

Matti Fischer

TARTU 1988

Kinnitatud füüsika-keemiateaduskonna
nõukogus 31. augustil 1987.a.

ОДНОКРИСТАЛЬНЫЙ МИКРОЭВМ КР1816ВЕ48.
Описание и принцип работы.
Составитель Матти Ф и ш е р.
На эстонском языке.
Тартуский государственный университет.
ЭССР, 202400, г.Тарту, ул.Юликооли, 18.
Vastutav toimetaja Ü. Haldra.
Paljundamisele antud 3.11.1987.
Formaat 60x84/16.
Rotaateripaber.
Masinakiri. Rotaprint.
Tingtrükipoogmaid 6,28.
Arvestuspoogmaid 6,0. Trükipoogmaid 6,75.
Trükiarv 700.
Tell. nr. 951.
Hind 20 kop.
TRÜ trükikoda. ENSV, 202400 Tartu, Tiigi t. 78.

Sissejuhatus.

Mõisted "mikroprotsessor" ja "integraallülituste mikroprotsessorkomplekt" seostuvad USA firmaga "Intel", kus 70ndate aastate alguses evitati seeriatootmisse ühel pooljuhtkristallil valmistatud protsessor (mikroprotsessor) ja rida protsessoriga ühildatavaid suuri integraallülitusi, mis koos protsessoriga võimaldasid lihtsalt koostada väikese ning odava universaalse juhtimisploki - mikroarvuti.

Integraallülituste mikroprotsessorkomplektide edasine areng on kulgenud kahes põhisuunas. Esimestele neljabitistele mikroprotsessoritele järgnesid kaheksa-, kuueteistkümne- ja kolmekümnekahebitised mikroprotsessorid, kasvanud on protsessorite töökiirus, laienenud on protsessoritega ühildatavate integraallülituste funktsionaalne skaala, kiiresti on kahanenud mikroskeemide hinnad. Ühtlasi on vähenenud lihtsate juhtimisarvutite koostamiseks vajalik minimaalne kiipide arv. Ka selles osas saavutas esimesena maksimaalselt edu firma "Intel", kus 70ndate aastate teisel poolel juurutati seeriatootmisse ühekiibiarvuti e. monoliitarvuti - integraallülitus, milles ühele pooljuhtkristallile on paigutatud kõik arvutile vajalikud sõlmed: protsessor, püsi- ja muutmälu, taktgeneraator, taimer, katkestuste kontroller ning infovahetust tagavad elektroonikaahelad.

Nõukogude Liidus on seeriatootmises kaheksabitise protsessoriga monoliitarvutid KM1816BE48, KM1816BE49, KM1816BE39 ja PBE035, mille prototüüpideks on firma "Intel" arvutid 8748, 8049, 8039 ja 8035. Need on väikeste koostisiseärasustega, lähedase tööpõhimõtte ning identse käsustikuga arvutid. Kuna meil veel puudub neid väga perspektiivseid integraallülitusi tutvustav raamat, siis püüame nende rakendamist kergendada käesoleva õppevahendi väljaandmisega.

Õppevahend "Monoliitarvuti KM1816BE48" koosneb kahest osast: "Kirjeldus" ja "Käsustik". Käesolevas vihikus on püütud KM1816BE48 kirjeldus esitada tasemel, mis on piisav selle ja temaga sarnaste arvutite rakendamiseks vajalike projekteerimisülesannete lahendamiseks. Kuigi vihiku lõpus ole-

vas lisas on toodud tabelid arvuti käsustiku lühikirjeldusega, võivad seal esitatud andmed osutuda liiga napiks kirjelduse nende lõikude mõistmisel, kus on juttu arvuti tööpõhimõttest seoses erinevate käskude täitmisega. Seetõttu on soovitav paralleelselt käesoleva vihikuga uurida ka õppevahendi teist vihikut "Käsustik", kus on esitatud arvuti kõigi käskude detailsed kirjeldused.

Õppevahend on adresseeritud eeskätt TRÜ füüsikaosakonna üliõpilastele kui konspekt mikroprotsessorite loengukursuse vastavast osast ning kui käsiraamat, mis lihtsustab ühekiibiarvuti rakendamist kursuse- ja diplomitööde tegemisel. Loodetavasti leevendab õppevahend ka füüsikaosakonna õppejõudude ja tehnilise personali arvutustehnikaalase kvalifikatsiooni tõstmisel kerkiva õppekirjanduse nappuse probleemi.

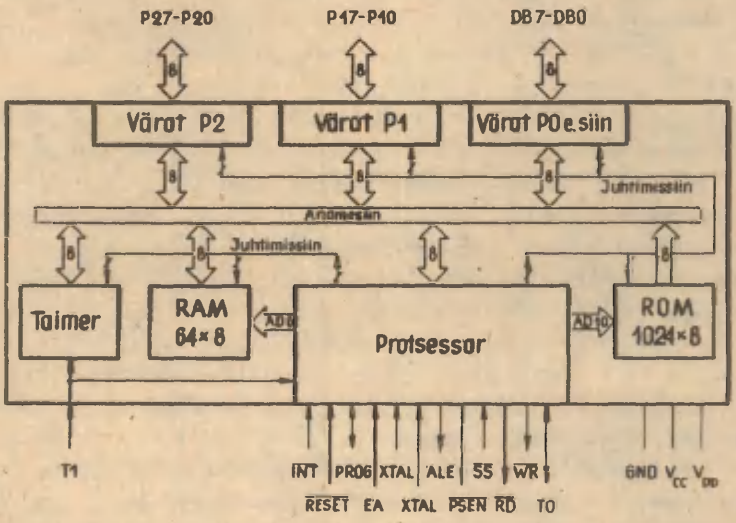
I ARVUTI EHITUS

1. Arvuti üldiseloomustus.

Monoliitarvuti KM1816BE48 on joonisel 1 kujutatud integraallülitus, mille koosseisu kuuluvad:

- kaheksabitine protsessor koos taktgeneraatori ning kaheni-voolise katkestuste kontrolleriga;
- elektriliselt programmeeritav ning ultraviolettkiirgusega kustutatav 1024 baidine püsimälu;
- muutmälu mahuga 64 baiti;
- kaheksabitised sisend-väljundväratid P0, P1 ja P2 ning kolm programmiliselt kontrollitavat sisendit T0, T1 ja INT;
- kaheksabitise loenduriga taimer.

Mikroskeemi, mis sisaldab arvuti kõiki põhisõlmi, nimetatakse monoliitarvutiks (vene keeles - однокристальный микроЭВМ, inglise keeles - microcontroller, single-chip microcomputer). Meie nimetame teda edaspidi lihtsalt arvutiks.



Joonis 1. Arvuti funktsionaalskeem.

Arvuti töökiirust annab ette viikudega XTAL ühendatav kvartskristall või väline signaalgeneraator. Sisendsignaali sagedus peab kuuluma vahemikku 1 - 6 MHz. Taktsignaal saadakse selle signaali sageduse jagamisel kolmega. Käskude täitmine toimub arvutis viietaktiliste masinatsüklite kaupa. Iga käsk täidetakse kas ühe või kahe masinatsükli jooksul.

Arvuti käsustik sisaldab 96 käsku. Neist 68 on ühebaidised ning ülejäänud 28 on kahebaidised käsud. Enamus käske on ühetsüklilised. Kõik kahebaidised ning 15 ühebaidist käsku on kahetsüklilised.

Väliste juhtimissignaalide abil saab arvutit seada järgmistesse töörežiimidesse:
- programmi täitmine;
- püsimälu programmeerimine;
- püsimälu kontroll-lugemine;
- häälestusrežiim;
- programmi käskhaavel täitmine.

Värat PO on kasutatav nii haviliku sisend-väljundväratina kui ka kahesuunalise kolmsolekulise multipleksse andmeja aadressisiinina (inglise keeles BUS), mis võimaldab arvutit väliselt laiendada. Püsimälu saab laiendada kolme kilobaidi võrra ning muutmälu 250 baidi võrra. Selle värati abil on arvutiga lihtne ühendada mikroprotsessorseeria KP580 mikroskeeme või arvutit ennast mõne teise mikroarvutiga.

Taimeris olevat loendurit saab seada kas arvuti teatud sisemiste impulsside või viigu T1 kaudu sisestatavate väliste impulsside arvu loendama. Sisemise signaali sagedus on 480 korda väiksem arvutit takteeriva kvartskristalli resonantssagedusest. Välise signaali sagedus peab aga olema vähemalt 45 korda väiksem nimetatud resonantssagedusest. Loenduri ületäitumist saab kasutada arvutilt taimerkatkestuse automaatseks nõudmiseks. Viigule $\overline{INT}$ antava signaaliga saab arvutilt nõuda väliskatkestust.

Programmi täitmise režiimis vajab arvuti vaid ühte toitepinget +5 V ning voolutarve ei ületa seejuures 135 mA. Kõigi sisend-väljundsignaalide pingetasemed vastavad TTL-loogika nivoodele. Monoliitarvutit KM1816BE48 valmistatakse 40neviigulises keraamilises korpuses massiga 7 grammi ning ta

sisaldab 18750 n-MOP transistori. Püsimällu salvestatud info garanteeritud säilivusaeg on vähemalt 15000 tundi ning kustutamis-programmeerimistsüklite lubatud arv on 25.

Lisaks arvutile KM1816BE48 toodetakse meil talle ehituselt ning tööpõhimõttelt väga sarnaseid arvuteid KM1816BE49, KM1816BE39 ja PBE035*. Need arvutid erinevad ükstaisest mälumahu, üldotstarbeliste sisend-väljundviikude arvu ning maksimaalse töökiiruse poolest nii, nagu on näidatud järgmises tabelis.

Tabel 1

Parameeter	TÜÜP			
	KM1816BE48	KM1816BE49	KM1816BE39	PBE035
Püsimälu maht (kilobaitides)	1K EPROM	2K	-	-
Muutmälu maht (baitides)	64	128	64	64
Masinatsükli kestus (μs)	2,5	1,4	1,4	2,5
Sisend-väljundviikude arv	27	27	15	15
Prototüüp (Intel USA)	8748	8049	8039	8035

Monoliitarvutid KM1816BE49 on projekteeritud suures seerias toodetavate automaatide valmistamiseks. Programm salvestatakse nende arvutite sisemisse püsimällu arvutite eneste valmistamisel ning see ei ole tarbija poolt enam muudetav.

Arvuteil KM1816BE39 ja PBE035 sisemine püsimälu puudub. Sisend-väljundväratite arv ja ehitus on neil täpselt samasugune kui arvutil KM1816BE48. Kuna väratit P0 ja värati P2 viike P23 - P20 kasutatakse välise püsimälu ühendamiseks arvutiga, siis üldotstarbeliste sisend-väljundviikude arv on neil kaheteistkümne võrra väiksem. Kõigi vaadeldud arvutite käsustik on identne.

* Juurutuspartiile PBE035 vastav seeriatähis on KM1816BE35.

(Toimetaja)

2. Arvuti plokkskeem.

Arvuti plokkskeem on toodud joonisel 2. Järgnevalt kirjeldame plokkide otstarvet, ehitust ja tööpõhimõtet.

2.1. Aritmeetika- ja loogikasektsioon.

Arvuti aritmeetika- ja loogikasektsiooni koosseisu kuuluvad aritmeetika- ja loogikaseade ehk ALU, akumulaator, akumulaatori fiksaator, vaheregister, programmiolekusõna PSW register ja kümnendkorrektsiooni plokk.

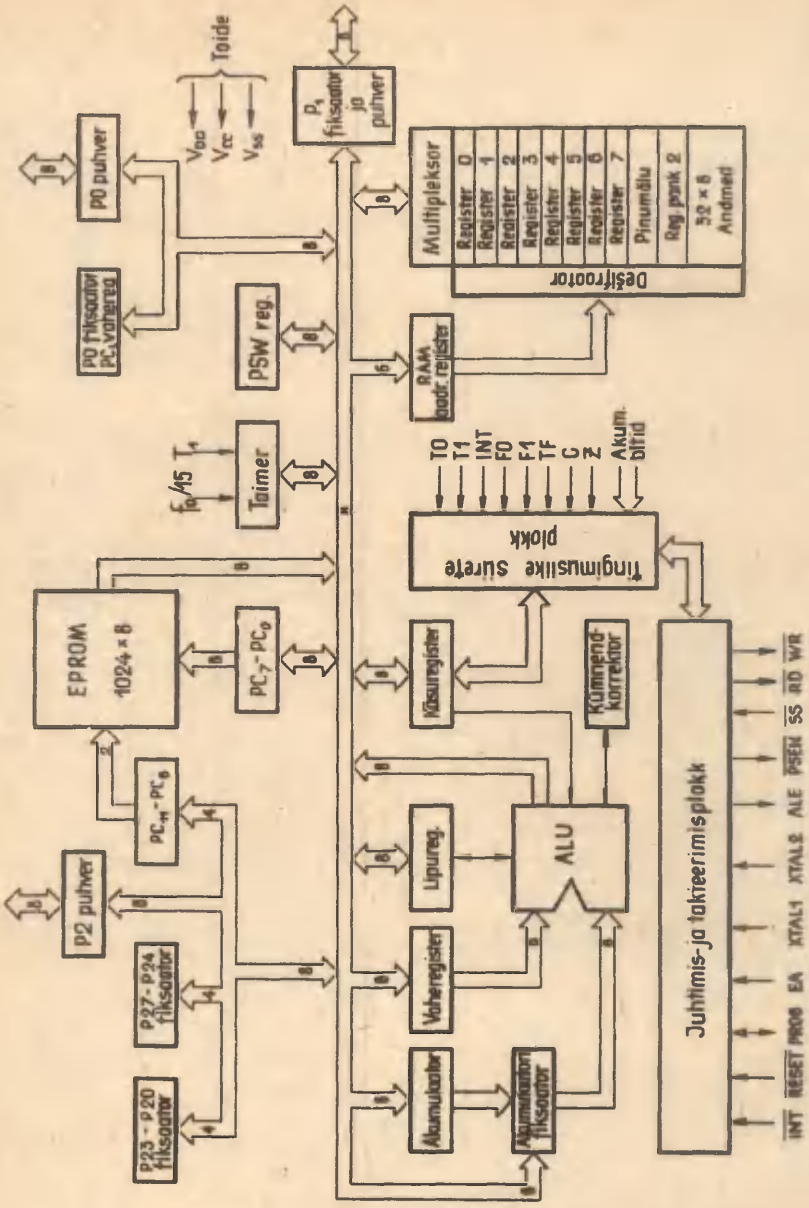
ALU (inglise keeles - arithmetic and logic unit) on seade, mis võimaldab kahendkoodis esitatud kaheksabitiste arvudega sooritada järgmisi tehteid (vt. tabelit 5 lk.93):

- liitmist käskudega ADDC või ADD;
- kümnendkorrektsiooni käsuga DA A;
- loogilist liitmist käskudega ORL ja ORLD;
- loogilist korrutamist käskudega ANL ja ANLD;
- välistavat loogilist liitmist käskudega XRL;
- inverteerimist käskudega CPL;
- nulli omistamist käskudega CLR;
- akumulaatoris oleva arvu nihutamist käskudega RL, RR, RLC ja RRC;
- akumulaatori kõrgema ja madalama poole sisude vahetamist käsuga SWAP A.

Akumulaator on kaheksabitine register, mida enamuse ALU-s sooritatavate tehete korral kasutatakse ühe operandi allikregistrina ning tulemi sihtregistrina. Tehte operand edastatakse ALU-sse akumulaatori fiksaatori kaudu ning tehte tulem ALU-st akumulaatorisse arvuti sisemise andmesiini kaudu. Akumulaator osaleb ka väratite kaudu välisseadmetega toimuvas andmevahetuses.

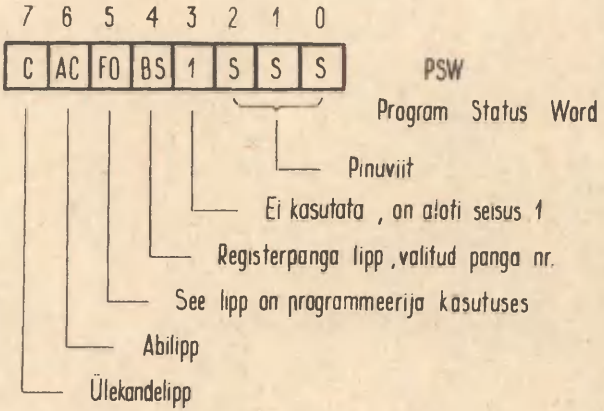
Kaheksabitise vaheregistri kaudu teisaldatakse ALU-sse seal sooritatava tehte operand. Kui tehte üks operand paikneb akumulaatoris, siis teine operand teisaldatakse ALU-sse alati vaheregistri kaudu.

Kaheksabitises registris PSW (inglise keeles - program status word) säilitatakse programmiolekusõna PSW. Selle sõ-



Joonis 2. Monoliitarvuti KM1816BE48 plokkskeem.

na bittide otstarvet selgitab joonis 3. Registri PSW kolmes madalamas bitis PSW_2- PSW_0 säilitatakse pinuviita. Siia salvestatud kahendarv näitab millisesse pesadepaari järgmine pinumällu salvestamine toimub. Igal pinumällu salvestamisel kasvatatakse viita automaatselt ühe võrra ja igal pinumälust lugemisel kahandatakse ühe võrra. Arvuti lähtestamisel signaaliga $\overline{RESET}$ seatakse viit automaatselt seisu 0. Täiendavad selgitused pinumällu salvestamise ja sealt lugemise kohta on toodud punktis 2.3.



Joonis 3. Programmiolekusõna vorming.

Olekuregistri bitti PSW_3 ei kasutata. Lugemisel on ta seisus 1. Bitt PSW_4 on registerpanga lipu BS (inglise keeles - bank select) bitt, mis näitab milline muutmälu registerpank on valitud. Käsuga SEL RB0 on ta seatav seisu 0. Siis on valitud registrid R0 - R7. Käsk SEL RB1 seab lipu BS seisu 1 ning valib registrid R0'- R7'.

Olekuregistri bitti PSW_5 kasutatakse programmeerija käsutuses oleva lipu F0 seisu säilitamiseks. Selle lipu seis on muudetav käskudega CLR F0 ja CPL F0 ning teda kasutab siirdekäsk JF0.

Kui arvude liitmisel toimub ülekanne tulemi kolmandast bitist neljandasse (madalamast poolest kõrgemasse), siis seatakse abilipp AC (inglise keeles - auxiliary carry) sei-

su 1, vastandjuhul seisu 0. Selle lipu seisu säilitatakse registri PSW kuuendas bitis. Kui baidiste arvude liitmisel saadakse tulem, mis on baidisest arvust suurem, siis seatakse ülekandelipp C (inglise keeles - carry) seisu 1, vastandjuhul seisu 0. Selle lipu seisu säilitatakse PSW kõrgeimas bitis ning ta on muudetav ka käskudega CLR C ja CPL C. Olekuregistrisse saab käsuga MOV PSW,A teisaldada akumulaatori sisu ning olekuregistris olevat arvu saab käsuga MOV A,PSW teisaldada akumulaatorisse.

Lahenduskäigu siirdumisel alamprogrammi või katkestavprogrammi täitmisele salvestatakse PSW neljas kõrgemas bitis PSW_7- PSW_4 olev arv automaatselt pinumällu ning ennistatakse alam- või katkestavprogrammist käsuga RETR naasmisel. Naasmiskäsul RET sellist toimet ei ole.

Arvuti võimaldab liita ka kahend-kümnendkoodis esitatud arve. Liitmiseks kasutatakse kahendarvude liitmise käske ADD või ADDC ning tulemit korrigeeritakse käsuga DA A. Korrigeerimises osaleb kümnendkorrektsiooni plokk. Sõltuvalt liitmistehte tulemist ning lippude C ja AC seisust teisaldatakse ALU-sse korrigeeriv arv 00_H, 06_H või 60_H, mis siis akumulaatoris olevale arvule liidetakse.

2.2. Püsimälu ja selle adresseerimine.

Universaalprotsessoritel baseeruvate arvutitega võrreldes on vaadeldavate monoliitarvutite omapäraks see, et püsimälu moodustab neil täiesti iseseisva mälupiirkonna, millest protsessor saab käsukoode ja konstantide väärtusi vaid lugeda. Käsustikus puuduvad käsud, mis võimaldaksid püsimälu piirkonda infot salvestada. Püsimälu adresseerimine (vajaliku mälupesa numbri esitamine) toimub arvutis alati 12bitise käsuloenduri PC kaudu. Seetõttu saab püsimälu maksimaalne maht olla vaid 4 kilobaiti (2^{12}=4096).

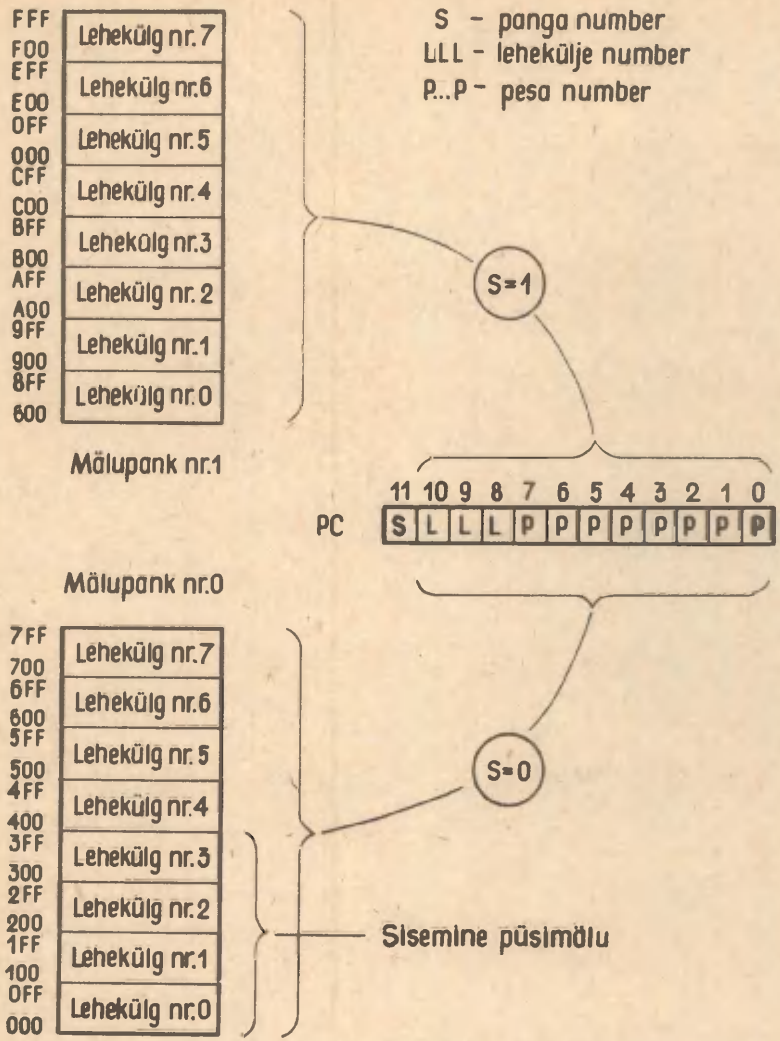
Püsimälu jaotub sisemiseks arvutikiibil paiknevaks mäluks ning väliseks laiendusmäluks. Arvutil KM1816BE48 on sisemise püsimälu maht 1024 baiti. See on elektriliselt programmeeritav ning ultravioletkiirgusega kustutatav mälu

(EPROM). Arvutil KM1816BE49 on kahe kilobaidine, arvutikiibi valmistamisel programmeeritud püsimälu, mille sisu ei ole kasutaja poolt enam muudetav. Mõlema arvuti püsimälu saab täiendavate mälukiipide lisamise teel laiendada nelja kilobaidini. Arvuteil KM1816BE39 ja PBE035 sisemist püsimälu ei ole. Nende töölerakendamiseks tuleb arvutiga ühendada välised mälukiibid, mille summaarne mälumaht võib samuti ulatuda nelja kilobaidini.

Kõigi vaadeldud arvutitüüpide maksimaalselt võimalik nelja kilobaidine püsimälu jaotub arvuti ehituse ning käsusüsteemi iseärasuste tõttu kaheks kahe kilobaidiseks mälupangaks. Esimest panka, mis paikneb aadresside vahemikus 000_H - $7FF_H$, nimetatakse mälupangaks nr.0. Teist, aadresside vahemikus 800_H - FFF_H paiknevat mälupiirkonda nimetatakse mälupangaks nr.1. Kumbki mälupank jaotub omakorda kaheksaks 256 baidiseks mäluleheküljeks. Arvuti KM1816BE48 püsimälu struktuuri selgitab joonis 4. Sisemine püsimälu paikneb panga nr.0 lehekülgedel 0 - 3. Ülejäänud püsimälu eksisteerib vaid siis, kui arvuti skeemi on väliste püsimälu kiipide abil laiendatud.

Käsuloenduri kõrgeimas bitis PC_{11} olev arv S määrab adresseeritava mälupanga numbri. Kui S=0, siis adresseeritakse mälupanka nr.0, vastandjuhul mälupanka nr.1. Käsuloenduri bittides PC_{10}- PC_8 paikneb mäluleheküljе numbrit määrav arv ning bittides PC_7- PC_0 leheküljel asuva mälupesa number. Kui näiteks (PC)=$30F_H$=001100001111, siis S=0, LLL=011 ja P...P==00001111 ning adresseeritakse panga nr.0 kolmandal leheküljel olevat mälupesa nr.$0F_H$=15_{10}.

Programmi täites loeb arvuti käsuloendurigа adresseeritavast püsimälu pesast käsu koodi ning kasvatab automaatselt käsuloenduri sisu ühe võrra. Seega sisaldab register PC käskude iga baidi püsimälust lugemise järel püsimälu selle pesa aadressi, milles paikneb järgmine käsubait. Registri PC sisu automaatne inkrementeerimine toimub siiski nii, et käsuloenduri kõrgeima biti PC_{11} seis ei muutu. Seisule $7FF_H$ järgneb seis 000_H, seisule FFF_H seis 800_H. Üleminek mälu ühest pangast teise on võimalik vaid käskude SEL MB0 ja SEL MB1 abil.



Joonis 4. Arvuti KM1816BE48 püsimälu struktuur.

4

Käsk SEL MB0 seab arvuti mälupanga trigeri DBF seisu 0 ning käsk SEL MB1 - seisu 1. Käsud JMP ja CALL edastavad trigeri DBF seisu käsuloenduri kõrgeimasse bitti PC_{11}. Kui oleme mälupanga nr.0 piires ja soovime lahenduskäiku suunata mälupanka nr.1, siis peame kasutama käsku SEL MB1 ja tingimatut siirdekäsku JMP. Öeldut selgitab järgmine näide.

Olgu püsimälus aadressil $11D_H$ paikneva käsu täitmise järel triger DBF seisus 0 ja olgu käsuloenduris järgmise mälupesa aadress $11E_H$ (vt. joonist 5). Järgmisena täidetakse mälupesas $11E_H$ paiknev käsk SEL MB1. Selle käsu koodi protsessorisse lugemisel kasvatatakse käsuloenduri sisu (PC):= =(PC)+1=$11F_H$. Käsu täitmisel seatakse triger DBF seisu 1.

Käsu aadress	$11D_H$	$11E_H$	$11F_H$	120_H
Käsu mnemokood		SEL MB1	JMP $4F0_H$	
Käsu 2ndkood		11110101	10000100	11110000
DBF triger	0	1	1	1
PC register	$11E_H$	$11F_H$	120_H	$CF0_H$

(PC):=$CF0_H$ | 1 | 1 | 0 | 0 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 |

Joonis 5. Mälupanga ümberlülimise näide.

Nüüd tuleb täitmisele käsk JMP $4F0_H$, mille kood paikneb pesades $11F_H$ ja 120_H. Selle käsu täitmine kulgeb järgmiselt:

1. Protsessorisse loetakse käsukood pesast $11F_H$ ning kasvatatakse käsuloenduri sisu (PC):=(PC)+1=120_H;
2. Käsukood desifreeritakse, mälust loetakse käsu teine bait ning (PC):=(PC)+1=121_H;
3. Käsu JMP täitmisel teisaldatakse:
   a) trigeri DBF seis käsuloenduri kõrgeimasse bitti PC_{11};
   b) käsukoodi esimese baidi kolmes kõrgemas bitis olev

mälulehekülje number $100=4_H$ käsuloenduri bittidesse PC_{10}- PC_8;

c) käsukoodi teises baidis olev mälupesa number $11110000=F0_H$ käsuloenduri bittidesse PC_7- PC_0.

Seega tuleb järgmisena täitmisele käsk, mis paikneb aadressil $CF0_H$ s.t. mälupanga nr.1 neljanda lehekülje pesas nr.$F0_H$. Kui käsk SEL MB1 puuduks, siis lahenduskäik jätkuks aadressil $4F0_H$ paiknevast käsust.

Alamprogrammi siirdumist võimaldava käsu CALL täitmisel kantakse esmalt käsuloenduri sisu pinumällu ning siis laaditakse siirdeaadress käsuloendurisse täpselt samuti kui käsu JMP täitmisel. Alamprogrammist käsuga RET või RETR naasmisel registri PC esialgne sisu ennistatakse, s.t. lahenduskäik siirdub käsule, mis paikneb programmi põhitekstis käsu CALL järel.

Oluline on siin meeles pidada, et käsud RET ja RETR ei muuda trigeri DBF seisu. Kui käsk CALL ja tema poolt kutsutav alamprogramm paiknevad erinevates mälupankades, tuleb käsu CALL ja talle programmi põhitekstis järgneva uue käsu JMP või CALL vahepealses lõigus hoolitseda trigeri DBF õigeksseadmise eest käsuga SEL. Vastandjuhul, kas me seda soovime või mitte, lülib järgmine käsk JMP või uus käsk CALL mälupanga jällegi ümber.

Kolm püsimälu pesa aadressidega 000_H, 003_H ja 007_H on eriotstarbelised. Arvuti lähtestamisel signaali $\overline{\text{RESET}}$ abil seatakse käsuloenduri kõik bitid seisu 0 ning programmi täitmine siirdub seega käsule, mis paikneb püsimälu alguspesas aadressiga 000_H.

Sisendi $\overline{\text{INT}}$ kaudu nõutava väliskatkestuse rahuldamisel seab arvuti käsuloendurisse aadressi 003_H. Seega peab väliskatkestust teenindava katkestavprogrammi esimene käsk paiknema pesas 003_H.

Arvuti sisemise taimeri poolt nõutava taimerkatkestuse rahuldamisel seatakse käsuloendurisse aadress 007_H. Seega peab taimerkatkestust teenindava katkestavprogrammi esimene käsk paiknema mälupesas aadressiga 007_H.

Oluline on veel teada, et katkestavprogrammi täitmisel on käsuloenduri kõrgeim bitt PC_{11} alati seisus 0. Katkestav-

programmi tekstis olevad SEL-käsud muudavad küll trigeri DBF seisu, kuid käsud JMP ja CALL seda käsuloenduri kõrgeimasse bitti PC_{11} ei kanna. Seetõttu peavad katkestavprogramm ja tema poolt kasutatavad alamprogrammid kindlasti paiknema mälupangas nr.0. Segaduste vältimiseks ei soovitata katkestavprogrammis käske SEL kasutada.

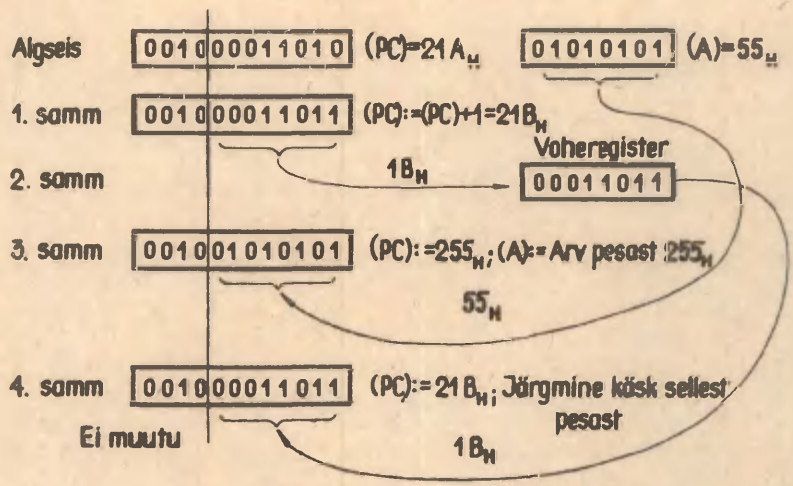
Peale programmi võib püsimällu ka konstante ning nende tabeleid salvestada. Püsimällu salvestatud suuruste lugemiseks on arvuti käsusüsteemis käsud MOVP ja MOVP3.

Käsu MOVP korral seatakse käsuloenduri bittidesse PC_7- PC_0 akumulaatoris olev arv, kusjuures PC kõrgemad bitid jäävad muutmata. Seega võimaldab käsk MOVP akumulaatorisse kanda arvu püsimälu käsiloleva lehekülje pesast, mille number oli akumulaatoris enne käsu täitmist. Käsu täitmise järel PC seis ennistatakse, s.t. programmi täitmine jätkub käsule MOVP programmi tekstis järgnevast käsust. Toome selgituseks järgmise näite.

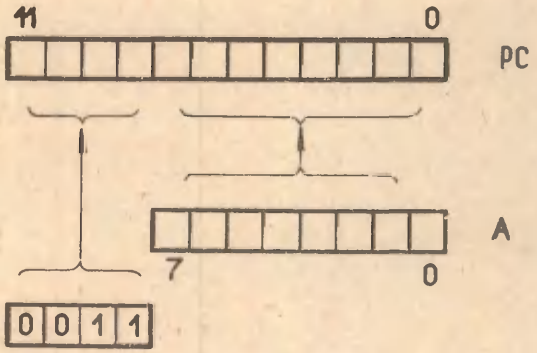
Olgu käsk MOVP püsimälu pesas, mille aadress on $21A_H$ ja enne selle käsu täitmisele asumist olgu akumulaatoris arv (A)=55_H (vt. joonist 6). Käsukoodi arvutisse lugemisel kasvatatakse käsuloenduri sisu (PC):=(PC)+1=$21B_H$. Käsukoodi dešifreerimise järel teisaldatakse käsuloenduri bittides PC_7- PC_0 olev arv $1B_H$ vaheregistrisse ning akumulaatoris olev arv 55_H bittidesse PC_7 - PC_0. Seega on nüüd (PC)=255_H ning akumulaatorisse loetakse arv püsimälu panga nr.0 teise lehekülje pesast nr.55_H. Lõpuks PC sisu ennistatakse - vaheregistris hoiul olnud arv $1B_H$ teisaldatakse tagasi käsuloendurisse nii, et (PC)=$21B_H$. Järgmisena tuleb täitmisele käsk, mis paikneb püsimälu selles pesas.

Käsu MOVP3 täitmisel loetakse akumulaatorisse arv püsimälu panga nr.0 kolmanda lehekülje pesast, mille number enne käsu täitmist oli akumulaatoris. Käsu täitmise järel jätkub programmi täitmine käsust, mis programmi tekstis järgneb käsule MOVP3. Aparatuurselt erineb käsu MOVP3 täitmine käsu MOVP täitmisest selle poolest, et 2. sammu ajal (vt. joonist 6) paigutatakse PC-registri kõik 12 bitti hoiule, kolmanda sammu ajal teisaldatakse käsuloenduri bittidesse PC_{11}- PC_8 kahendarv 0011 ning bittidesse PC_7- PC_0 akumulaa-

toris olev arv (vt. joonist 7). Käsu täitmine lõpeb 2.sammu ajal salvestatud PC täieliku sisu ennistamisega.



Joonis 6. Adresseerimine käsu MOVP täitmisel.



Joonis 7. Mälupesa aadressi moodustamine käsu MOVP3 täitmisel.

Tingimatu siirdekäsk JMPP ja tingimuslikud siirdekäsud võimaldavad programmi täitmist jätkata püsimälu käsiloleva

5

lehekülje suvalisel aadressil paiknevast käsust. Käsu JMPP täitmisel loeb protsessor siirdepesa aadressi püsimälu käsiloleva lehekülje sellest pesast, mille number on akumulaatoris. Tingimuslike siirdekäskude täitmisel kontrollitakse esmalt, kas tingimus on täidetud või mitte. Kui on, siis teisaldatakse käsuloenduri bittidesse PC_7- PC_0 käsu teises baidis olev arv.

Käskude MOVP, JMPP ja kõigi tingimuslike siirdekäskude korral toimub adresseerimine reeglina püsimälu käsiloleva lehekülje piires. Erandi moodustab nende käskude korral olukord, kus käsk paikneb mälulehekülje viimases pesas, mille number on FF. Sellisel juhul käsukoodi protsessorisse lugemisega kaasnev käsuloenduri sisu kasvatamine seab käsuloendurisse järgmise lehekülje pesa nr.00 aadressi.

Pöördume tagasi leheküljel nr.16 toodud näite juurde (vt. joonist 6). Kui algseisus oleks (PC)=$2FF_H$, siis 1.sammu ajal saaks (PC):=(PC)+1=300_H ning 3.sammu ajal saaks käsuloendurisse juba aadress (PC)=355_H. Adresseeritakse mitte 2.lehekülje, vaid 3.lehekülje pesa nr.55_H. Käskude JMP, CALL ja MOVP3 korral seda nn. lehekülje rajaprobleemi, ei esine, kuna nende käskude täitmisel modifitseeritakse käsuloenduri kõigi bittide PC_{11}- PC_0 seis.

2.3. Muutmälu ja selle adresseerimine.

Arvuti sisemine muutmälu sisaldab 64 mälupesa, mille aadressid asuvad vahemikus 00_H- $3F_H$. Vajaduse korral on muutmälu võimalik väliselt laiendada veel 256 mälupesa võrra. Välise muutmälu aadressid asuvad vahemikus 00_H- FF_H. Kõik muutmälu pesad on 8 bitised.

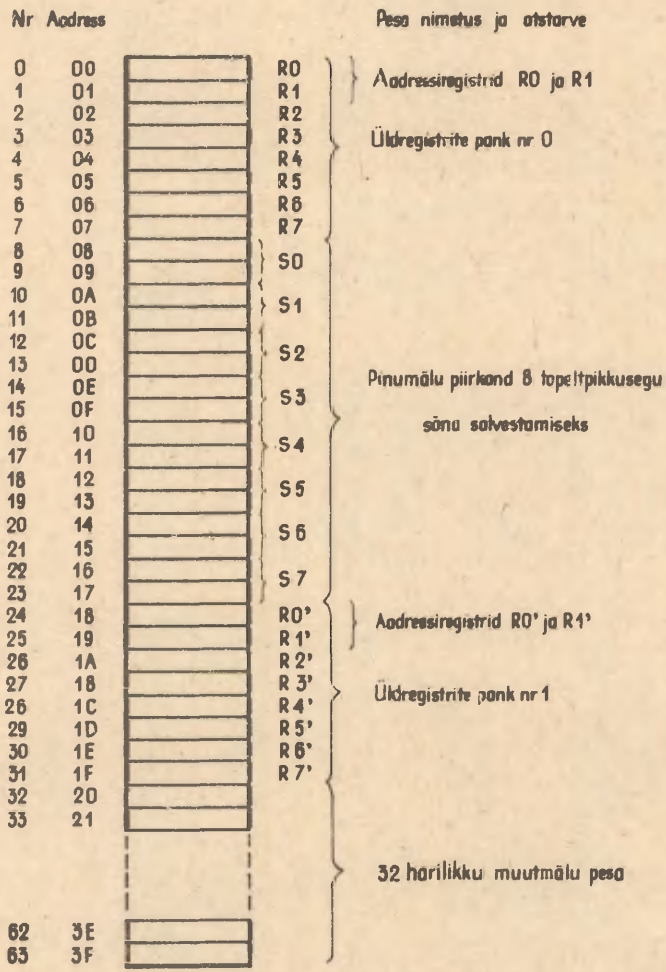
Nagu näeme, ei ole väline muutmälu sisemise jätkuks, vaid kujutab endast täiesti iseseisvat mäluplokki. See on tingitud arvuti arhitektuuri ja käsustiku omapärast. Käsusüsteemis on vaid kaks käsku, mis võimaldavad välise muutmälu poole pöörduda. Käsk MOVX A,@R loeb arvu välisest muutmälust akumulaatorisse ja käsk MOVX @R,A - salvestab akumulaatoris oleva arvu välisesse muutmällu. Kõik ülejäänud muutmäluga seotud käsud kasutavad arvuti sisemist muutmälu.

Valdava enamuse sisemist muutmälu kasutavate käskude täitmiseks kulub üks masinatsükkel ehk 5 takti. Erandi moodustavad vahetute andmete salvestamise käsud MOV @R,#data ja MOV R,#data ning siirdekäsud CALL, RET, RETR ja DJNZ R,addr, mille täitmiseks kulub 2 masinatsüklit ehk 10 takti. Mõlemad MOVX tüüpi andmevahetuskäsud täidetakse 10 takti jooksul. Seetõttu nimetatakse sisemist muutmälu vahel ka ülioperatiivmäluks (scratchpad memory).

Väline muutmälu on ühefunktsionaalne. Ta on ette nähtud vaid andmete salvestamiseks ja lugemiseks. Sisemine muutmälu on mitmefunktsionaalne. Pooled selle mälu pesadest on kasutatavad mitmel erineval otstarbel, sõltuvalt lahendatava ülesande tüübist ning programmi kirjutava programmeerija suvast.

Sisemise muutmälu struktuur on toodud joonisel 8. Kaheksa esimest mälupesa R0 - R7 ning kaheksa mälupesa R0'- R7' aadresside vahemikus 18_H- $1F_H$ kannavad üldregistrite nime. Esimesed moodustavad registrite panga nr.0 ja teised - registrite panga nr.1.

Arvuti käsusüsteemis on 12 käsku, mis kasutavad üldregistrites olevaid andmeid aritmeetika- ning loogikatehete sooritamiseks või võimaldavad andmeid neisse registritesse salvestada või sealt lugeda. Nende käskude koodi kolmes madalamas bitis on esitatud vajaliku registri number. Selle kohta öeldakse, et need käsud kasutavad fikseeritud adresseerimisviisi - käsukoodis on fikseeritud register, milles sisaldub operand. Toome näiteks käsu MOV A,RN, mille koodi üldkuju on 11111NNN. See käsk teisaldab akumulaatorisse arvu registrist, mille numbrit näitab kahendarv NNN. Kui näiteks soovime akumulaatorisse teisaldada andmeid registrist R5 või R5', siis tuleb kasutada käsku MOV A,R5 ning käsukood peab omama konkreetset kuju 11111101 = FD_H. Kas andmed teisaldatakse akumulaatorisse registrist R5 või registrist R5', sõltub programmiolekusõnast PSW, mille vorming on toodud joonisel 3. Kui olekusõna neljandas bitis olev lipp BS on seisus 0, siis pöördutakse automaatselt registerpanga nr.0 registrite poole, vastandjuhul panga nr.1 registrite poole. Panga ümberlülimiseks on arvuti käsusüsteemis kaks

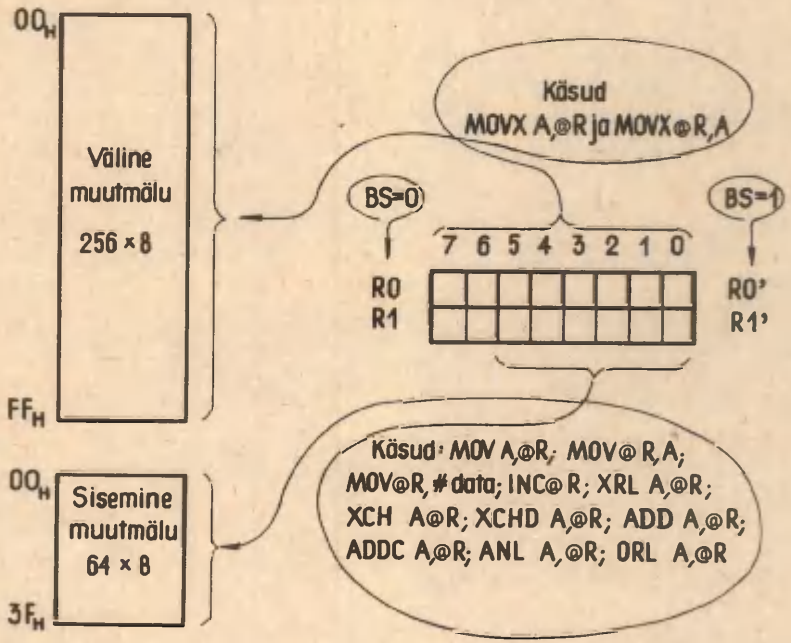


Joonis 8. Sisemise muutmälu struktuur.

spetsiaalset käsku. Käsk SEL RB0 seab lipu BS seisu 0 ja käsk SEL RB1 - seisu 1.

Kõigi sisemise ning välise muutmälu pesade poole pöördumisel on kasutatav veel kaudadresseerimismoodus - käsk valib ühe aadressiregistritest, milles olevat arvu kasutatakse aadressina. Kui lipp BS on seisus 0, siis on kaudadresseerimisel aadressiregistriks kas R0 või R1, vastandjuhul R0' või R1'. Seega on mõlema registerpanga kaks esimest registrit kasutatavad ka kui aadressiregistrid. Kui registrite poole pöördumisel saab programmeerija kasutada nii fikseeritud kui ka kaudset adresseerimisviisi, siis muutmälu kõigi ülejäänud pesade poole pöördumisel saab ta kasutada vaid kaudset adresseerimisviisi (vt. joon.9).

Kuna sisemine muutmälu sisaldab vaid 64 mälupesa, siis selle mälu pesa aadress saadakse aadressiregistri kuuest ma-



Joonis 9. Muutmälu adresseerimine aadressiregistrite abil.

dalamast bitist (2^6=64). Väline muutmälu sisaldab kuni 256 mälupesa ning selle poole pöördumisel kasutatakse aadressi saamiseks aadressiregistri kõiki kaheksat bitti. Öeldut selgitab joonis 9 ja järgmine näide.

Oletame, et me soovime akumulaatoris olevat arvu vahetada arvuga mälupesast, mille aadress on $0F_H$. Kuna pesa $0F_H$ ei ole registerpesa, siis teda saab adresseerida vaid registri kaudu. Toimime järgmiselt:

```
SEL RB0 / valime registerpanga nr.0;
MOV R0,#0F_H / kanname registrisse R0 aadressi 0F_H;
XCH A,@R0 / vahetame akumulaatori sisu selle mälupesa
            sisuga, mille aadress on registris R0.
```

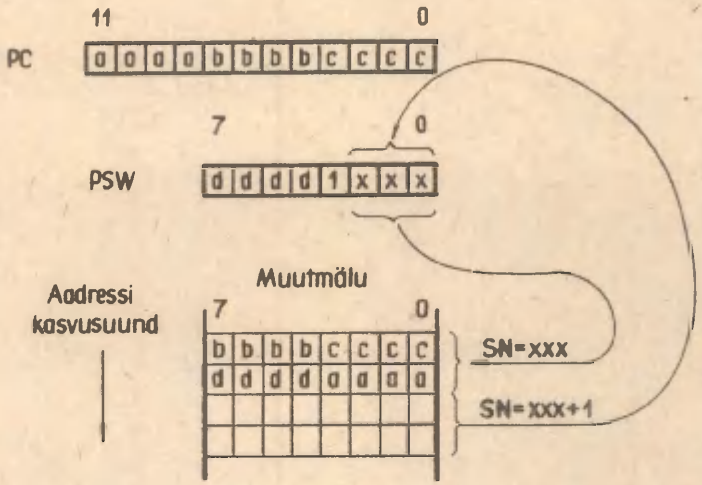
Registrite R0, R1, R0' ja R1' abil saab adresseerida ka neid registreid eneseid, kui selleks peaks soov tekkima.

Aadresside vahemikus 08_H- 17_H paiknevad sisemise muutmälu pesad on kasutatavad kui 16 harilikku mälupesa, mis on adresseeritavad aadressiregistrite abil, kuid nad võivad olla kasutuses ka kui 8 kahebaidist LIFO-tüüpi pinumälu pesa S0 - S7. Mõiste LIFO tuleneb ingliskeelsest väljendist "last in - first out", mida tuleb tõlgendada nii, et pinumälust loetakse esimesena sinna viimati kirjutatu.

Programmeerija ei saa seda mälupiirkonda pinuna kasutada, kuna käsustikus pole selleks vajalikke käske. Pinu kasutab arvuti käsuloenduri PC sisu ning programmiolekusõna PSW nelja kõrgema biti automaatseks säilitamiseks katkestav- või alamprogrammi siirdumise eel ning käsuloenduri sisu ennistamiseks neist programmidest põhiprogrammi naasmisel. Selle info pinumällu pakkimise korda selgitab joonis 10. Neist programmidest käskudega RET või RETR põhiprogrammi naasmisel käsuloenduri esialgne sisu ennistatakse. Käsu RETR täitmisel ennistatakse ka PSW_7 - PSW_4 esialgne sisu. Käsul RET sellist toimet ei ole.

Pinumälu adresseerimine toimub PSW kolmes madalamas bitis säilitatava pinuviida abil. See arv viitab millisesse pinumälu pesadepaari (vt. joonist 8) infot järgmisel korral salvestatakse. Arvuti lähtestamisel signaaliga RESET seatakse pinuviit nulli. Esimene pinumällu salvestamine toimub pesadepaari S0. Iga salvestamise järel kasvatatakse pinuviita

ühe võrra ja iga lugemise eel kahandatakse ühe võrra.



Joonis 10. Info pinumällu pakkimise kord.

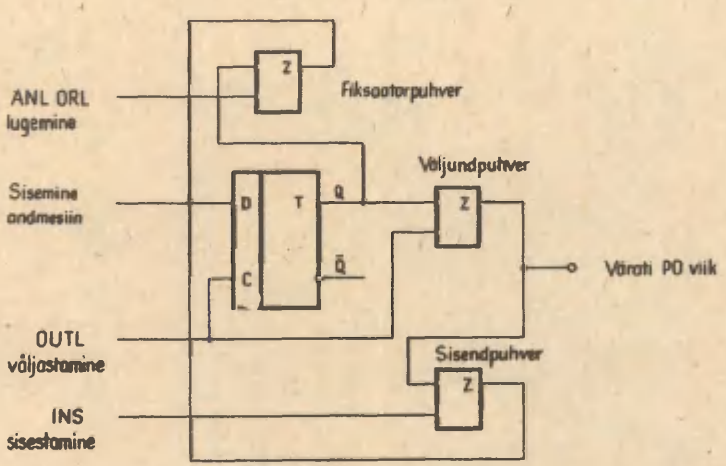
Programmeerija peab alati ise hoolitsema selles eest, et alamprogrammide kasutamissügavus alamprogrammide või katkestavprogrammide sees ei ületaks seitset. Vastandjuhul toimub pinumälu ületäitumine. Viida seis 111 teiseneb järgmisel salvestamisel seisuks 000. Kui nüüd enne lugemist veel kord salvestada, siis paaris S0 olev info kirjutatakse uuega üle ning seal varem säilitatu läheb kaduma. Kui viida seis on 000, mis viitab pesadepaarile S0, siis käsu RET või RETR täitmisel toimuval viida kahandamisel saadakse sinna arv 111, mis viitab paarile S7. Infot loetakse mälupesadest aadressidega 16_H ja 17_H.

2.4. Andmevahetusplokk.

Arvuti andmevahetusploki ülesandeks on infovahetuse korraldamine arvuti ja väliste seadmete vahel. Andmevahetusplokk sisaldab kolme 8-bitist sisend-väljundväratit P0, P1 ja P2 ning kolme mitmefunktsionaalset sisend-väljundviiku T0, T1 ja $\overline{INT}$.

Väratit P0 on võimalik kasutada nii üldotstarbelise sisend-väljundväratina kui ka kahesuunalise kolmeolekulise sadressi- ja andmesiini väratina. Seetõttu on käskude mnemokoodis ta nimeks BUS. Väratid P1 ja P2 on kasutatavad kvaasikahesuunaliste fikseeritud väljunditega sisend-väljundväratitena. Signaal arvuti viikudel T0, T1 ja $\overline{INT}$ on kontrollitav tingimuslike siirdekäskudega JNT0, JT0, JNT1, JT1, JNI ning käskudega STRT CNT, STOP TCNT ja ENT0 CLK.

Infovahetust värati P0 viikude kaudu illustreerib joonis 11. Kõik kaheksa sisend-väljundahelat koosnevad D-trigeril baseeruvast fiksaatorist ja puhvritest. Kõigi viikude puhvreid juhitakse paralleelselt. Seetõttu on kõik viigud korraga seatud sisenditeks või väljunditeks.



Joonis 11. Andmevahetus värati P0 kaudu.

Värati PO seab fikseeritud väljundiks käsk OUTL BUS,A, mis laadib akumulaatoris oleva arvu trigeritesse ning avab väljundpuhvrid. Sisendpuhvrid ja fiksaatorpuhvrid on seejuures kolmandas olekus. Fikseeritud arv püsib väljundeis muutumatuna uue väljastamiseni.

Välisseadmete operatiivseks juhtimiseks on käskudega ANL BUS,#data ning ORL BUS,#data võimalik sooritada loogikatehteid väratisse fikseeritud arvuga. Nende käskude täitmisel avatakse korraks fiksaatorpuhvrid ning väratisse fikseeritud arv teisaldatakse andmesiini kaudu protsessorisse, sooritatakse loogikatehe käsu teises baidis vahetult antud arvuga ning tehte tulem fikseeritakse värati trigeritesse. Käsud ANL ja ORL modifitseerivad värati PO väljundsignaale vaid siis, kui värat on käsuga OUTL seatud fikseeritud väljundväratiks. Vastandjuhul on väljundid kolmandas olekus. Tehte tulem paigutatakse küll trigeritesse, kuid mitte väljundviikudele. Käsud ORL ja ANL värati tööreŽiimi ei muuda.

Kahebaidine käsk INS A,BUS seab värati PO sisendväratiks. Seejuures seatakse väljundpuhvrid kolmandasse olekusse ning sisendpuhvrid avatakse käsu teise tsükli teise takti vältel. Selles taktis toimub värati viikudel oleva info teisaldamine arvuti sisemise andmesiini kaudu akumulaatorisse. Käskude OUTL ja INS täitmisel väljastab arvuti välisseadmetele juhtimissignaalid vastavalt $\overline{WR}$ ja $\overline{RR}$ väljunditesse.

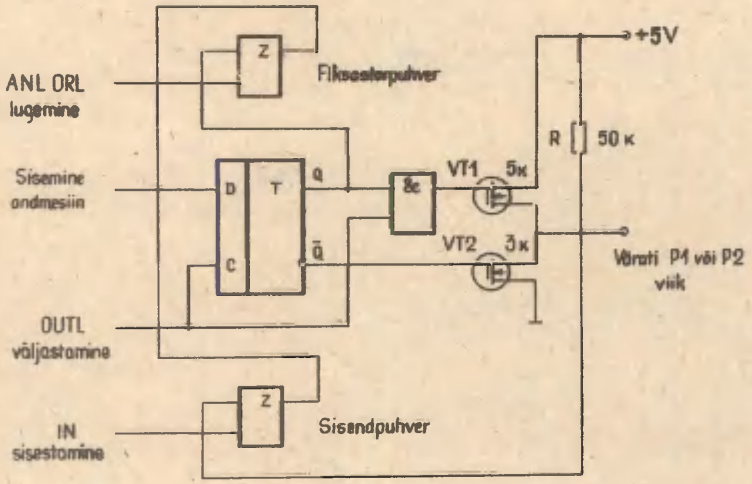
Lisaks vaadeldud kahele tööreŽiimile on värat PO kasutatav veel kui kahesuunaline kolmeolekuline aadressi- ja andmesiini värat. See tagab arvuti laiendamisvõimaluse väliste püsi- ning muutmäludega, võimaldab arvutiga siduda KP580-seeria suuri kiipe KP580BB51, KP580BB55, KP580BV53, KP580BB79 jne. ning ühendada meie arvuti abiseadmena mikroprotsessoril KP580VK80 baseeruvasse mikroarvutisse.

Värati PO kasutamisel kahesuunalise siiniväratina fikseeritakse väljastatavad andmed fiksaatortrigeritesse ning väljundpuhver avatakse täidetava masinatsükli nende taktide vältel, mis on aparatuurselt fikseeritud andmevahetusprotokollis. Kui väljundpuhver on avatud, siis sisendpuhver on kolmandas olekus ja vastupidi. Andmete sisestamiseks avatakse andmevahetusprotokollis fikseeritud momentidel sisend-

puhvrid, mis värati viikudel oleva signaali kannavad arvuti sisemisele andmesiinile. Seejuures väljastab arvuti laiendusskeemide töö sünkroniseerimiseks juhtimissignaale ALE, $\overline{PSEN}$, $\overline{RH}$ ja $\overline{WH}$. Kui selles reŝiimis oleva värati kaudu andmeid parasjagu ei sisestata või ei väljastata, on ta viigud kolmandas olekus.

Andmevahetusprotokolli illustreerivad ajadiagrammid koos selgitustega on esitatud järgmises peatükis, kus kirjeldatakse arvuti tööreŝiime.

Väratid P1 ja P2 on oma ehituselt identsed. Info sisestamist ja väljastamist nende väratite kaudu illustreerib joonis 12. Iga viiguga seotud skeem sisaldab fiksaatortrigerit, sisend- ja fiksaatorpuhvrit ning võtmeid VT1 ja VT2. Käsuga OUTL P,A väljastatavad andmed fikseeritakse D-trigeritel baseeruvates fiksaatorites. Trigeri seadmisel seisu Q=1 avab ühe takti pikkune väljastamisimpulss NING-elemendi ja seega ka transistori VT1. Avatud transistori VT1 5 kΩ-ne takistus kiirendab viigu ülesminekuprotsessi seisust 0 sei-



Joonis 12. Andmevahetus väratite P1 ja P2 kaudu.

su 1. Seejärel VT1 sulgub ning viik hoitakse seisus 1 takisti R abil. Signaali 0 fiksaatorisse kirjutamisel $\overline{Q}$=1 ning avatud transistori VT2 väike 3 kΩ-ne takistus hoiab viigu seisus 0.

Käskudega ANL P,#data ja ORL P,#data on võimalik väljunditesse seada sinna fikseeritud arvu ja käsu teises baidis antud vahetu operandiga teostatud loogikatehte tulemit. Selleks avatakse korraks fiksaatorpuhvrid ning väratisse fikseeritud arv teisaldatakse arvuti sisemise andmesiini kaudu protsessorisse. Loogikatehte tulem fikseeritakse värati trigerites ning edastatakse väljundviikudele.

Väratite viikudel olevat infot on võimalik protsessorisse sisestada käsuga IN A,P. Selle käsu täitmisel avanevad korraks sisendpuhvrid ning viikudel olev signaal teisaldub arvuti sisemise andmesiini kaudu akumulaatorisse.

Värateid P1 ja P2 nimetatakse kvaasikahesuunalisteks väratiteks seetõttu, et nende kasutamisel sisendväratitena tuleb esmalt sisendviik seada käsuga OUTL P,A seisu 1, mille väline skeem siis vajaduse korral nulli surub. Viik, mille protsessor on seadnud seisu 0, ei ole väliselt seisu 1 seatav, kuna avatud transistori VT2 takistus on väga väike - 3 kΩ. Kui värati P0 kõik viigud võisid samsaegselt olla ainult väljundviigud või ainult sisendviigud, siis väratite P1 ja P2 viikude otstarve on individuaalselt valitav. Sisendiks võib olla iga viik, mille fiksaatoritriger on seatud seisu 1.

Värati P2 viigud P23 - P20 on välise püsimälu poole pöördumisel kasutatavad aadressi nelja kõrgema biti väljundviikudena. Sinna kantakse käsuloenduri bittides PC_{11}- PC_8 olev arv. Sellele režiimile vastava andmevahetusprotokolli ajadiagrammid on esitatud järgmises peatükis.

Firma "Intel" on meie arvuti prototüübi projekteerimisel realiseerinud arvuti sisend-väljundkanalite laiendamisvõimaluse värati P2 viikude P23 - P20 kaudu. Selleks on projekteeritud spetsiaalne kiip Intel 8243, mis võimaldab luua veel neli neljabitist sisend-väljundväratit. Need võimalused on täielikult realiseeritud ka meie arvutis. Nelja täiendavat väratit nimetatakse väratiteks P4, P5, P6 ja P7. Andme-

vahetust nende väratitega võimaldavad arvuti käsud MOVD P,A, MOVD A,P, ANLD P,A ja ORLD P,A. Detailselt on seda laiendusvõimalust kirjeldatud õppevahendi kolmandas peatükis.

Sisend-väljundviik T0 on kasutatav sisendina, mida kontrollivad kahebaidised siirdekäsud JNT0 ja JT0. Kui sisendis T0 on signaal 0, siis käsu JNT0 täitmisel siirdub programmi täitmine käsiloleva mälulehekülje selles pesas olevale käsule, mille number on käsu JNT0 teises baidis. Kui T0=1, siis toimib analoogiliselt käsk JT0.

Käsuga ENT0 CLK on viik T0 seatav väljundiks, mille kaudu arvuti väljastab takteerimissignaali sagedusega $f_o/3$. Kui arvuti taktgeneraatoris kasutatava kvartskristalli sagedus on 6 MHz, siis väljastatakse käsu ENT0 CLK täitmise järel väljundi T0 kaudu signaali sagedusega 2 MHz. Takteerimissignaali väljastamist viigule T0 on võimalik katkestada vaid arvuti lähtestamisega signaali $\overline{\text{RESET}}$ abil. Viik T0 on kasutatav ka kui eriotstarbeline sisend arvuti sisemise püsimälu programmeerimisel.

Viik T1 on kasutatav sisendina, mida kontrollivad kahebaidised tingimuslikud siirdekäsud JNT1 ja JT1. Käsk STRT CNT ühendab viigu T1 taimeris oleva loenduri sisendiga (joon. 13). Seega on T1 kasutatav ka sisendina väliste sündmuste arvu loendamisel. Punktis 2.8 on kirjeldatud viigu T1 kasutamist täiendava väliskatkestuste sisendina. Käsk STOP TCNT katkestab ühenduse viigu T1 ja taimeris oleva loenduri vahel.

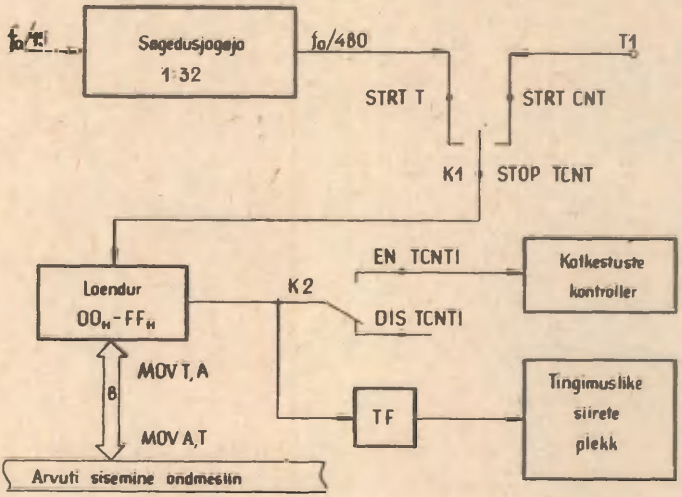
Sisendviik $\overline{\text{INT}}$ on kasutatav arvutilt väliskatkestuste nõudmiseks. Aktiivne on seejuures sisendsignaal 0. Sisend $\overline{\text{INT}}$ on kontrollitav ka kahebaidise tingimusliku siirdekäsuga JNI. Kui sisendis $\overline{\text{INT}}$ on signaal 0, siis siirdub programmi täitmise JNI teises baidis märgitud mälupesas olevale käsule. Seda võimalust kasutatakse tavaliselt enne katkestavprogrammist väljumist kontrollimaks, kas katkestusnõudmine on kõrvaldatud. Kui katkestavprogrammist väljuda enne katkestusnõudmise kõrvaldamist, siis siirdub lahenduskäik kohe katkestavprogrammi tagasi.

Arvuti lähtestamisel signaaliga $\overline{\text{RESET}}$ seatakse värati P0 viigud kolmandasse olekusse, väratite P1 ja P2 viikudele

väljastatakse signaal 1, viigu T0 side sisemise taktgeneraatoriga ning viigu T1 side taimeris oleva loenduriga katkestatakse ja katkestused keelatakse.

2.5. Taimer.

Taimer võimaldab protsessorit koormamata pidada ajaarvestust või loendada väliste impulsside arvu. Taimeri tööpõhimõtet illustreerib joonis 13. Ploki koosseisu kuuluvad sagedusjagaja, loendur, taimerlipu triger ning programmiliselt juhitavad kommutatsioonielemendid K1 ja K2.



Joonis 13. Taimeri funktsionaalskeem.

Taimeri tähtsaim sõlm on 8-bitine kahendloendur. Algseis on loendurisse suvalisel momendil laaditav käsuga MOV T,A, mis teisaldab loendurisse akumulaatoris oleva arvu. Loenduri hetkseis on kontrollitav käsuga MOV A,T, mis teisaldab loenduris oleva arvu akumulaatorisse.

Loendur on käivitatav käskudega STRT T ja STRT CNT. Esi-

mene käsk nullib sagedusjagaja seisu ning kommuteerib võtme K1 abil loenduri sisendisse signaali sagedusjagaja väljundist. Seda töömoodust nimetatakse taimerrežiimiks. Teine käsk kommuteerib loenduri sisendisse signaali viigult T1 ning seab taimeri nn. loendusrežiimi. Käivitatud loenduri seis kasvab ühe võrra sisendsignaali igal üleminekul seisust 1 seisu 0. Loendamise peatab loenduri sisu muutmata käsk STOP TCNT või arvuti lähtestamine signaaliga RESET.

Loenduri ületäitumine - üleminek seisus FF_H seisu 00_H, seab taimerlipu trigeri TF (inglise keeles - timer flag) seisu 1. Seda lippu kontrollib tingimussiirde käsk JTF. Kui lipp (TF)=1, siis jätkatakse programmi täitmist püsimälu käsiloleva lehekülje selles pesas paiknevast käsust, mille number paikneb käsu JTF teises baidis. Triger TF seatakse automaatselt seisu 0 käsu JTF igal täitmisel või arvuti lähtestamisel.

Taimerkatkestusi lubav käsk EN TCNTI lülib võtme K2 asendisse, milles loenduri ületäitumissignaal antakse ka katkestuste kontrollerisse. See signaal seab katkestuste kontrolleris (vt. joonist 15) paikneva taimerkatkestuste trigeri seisu 1 ning arvutilt nõutakse taimerkatkestust. Nõudmise rahuldamisel siirdub programmi täitmine automaatselt püsimälu pesas 007_H olevale käsule ning taimerkatkestuste triger seatakse seisu 0. Katkestusnõudmise annulleerib ka taimerkatkestusi keelav käsk DIS TCNTI ning arvuti lähtestamine.

Väliskatkestustel on taimerkatkestuste ees prioriteet. Kui mõlemad katkestused on lubatud ning nende nõudmine laekub samaaegselt, siis rahuldab arvuti esmalt väliskatkestusnõudmise. Kuna katkestused on katkestuse teenindamise ajal automaatselt keelatud, siis väliskatkestuse nõudmine parasjagu teenindatavat taimerkatkestust ei katkesta.

Taimerrežiimi kasutatakse ajaarvestuse pidamiseks. Loenduri sisendsignaal saadakse siin sagedusjagajast, mille sisendis oleva masinatsüklite kordussagedusliku signaali sagedus on 15 korda madalam arvutit takteeriva kvartskristalli resonantssagedusest f_0 ning jagamistegur on võrdne 32-ga. Seega on loenduri sisendis oleva signaali sagedus $f=f_0/480$.

Ajaarvestuse täpsuse määrab loenduri sisendsignaali periood, mis f_o= 6 MHz korral on 80 μs.

Ajavahemik loenduri käivitamisest käsuga STRT T kuni loenduri ületäitumiseni on maksimaalne, kui loendusregister enne käivitamist nullida. Minimaalse ajavahemiku saame algväärtuse FF_H laadimisel loendurisse. Juhul f_o= 6 MHz vältavad need ajavahemikud vastavalt 20,48 ms ja 80 μs.

Loendusrežiimi seatud taimer on kasutatav välissündmuste arvu loendamiseks või ajaarvestuse pidamiseks. Loenduri seis muutub viigul T1 oleva signaali igal ülemineku1 seisust 1 seisu 0. Sisendsignaali sageduse ülemine piir f_{max}= $=f_0/45$ ning sisend peab igas perioodis olema nullseisus vähemalt kolme f_0 perioodi (arvuti töötakti) vältel. Juhul f_0=6 MHz on f_{max}=133 kHz (periood 7,5 μs) ning nullsignaali minimaalne vältus igas perioodis on 500 ns. Sisendsignaali sageduse alumine piir ei ole reglementeeritud ning nivood peavad vastama TTL standardile.

2.6. Tingimussiirete plokk.

Tingimussiirete plokk osaleb programmis hargnemist võimaldavate tingimuslike siirdekäskude täitmisel. Ülevaate käsustikus sisalduvatest tingimuslikest siirdekäskudest ning nende poolt kontrollitavatest siirdumistingimustest annab järgmine tabel.

Mnemokood	Kontrollitav suurus	Siirdumistingimus
JZ addr	Akumulaatori sisu	$(A)=00_H$
JNZ addr	Akumulaatori sisu	$(A)\neq 00_H$
JBb addr	Akumulaatori suvaline bitt	$(a_b)=1$
JNC addr	Ülekandelipu seis	(C)=0
JC addr	Ülekandelipu seis	(C)=1
JF0 addr	Lipu F0 seis	(F0)=1
JF1 addr	Lipu F1 seis	(F1)=1
JTF addr	Taimerlipu seis	(TF)=1
JNT0 addr	Signaal sisendis T0	(T0)=0
JT0 addr	Signaal sisendis T0	(T0)=1

Mnemokood	Kontrollitav suurus	Siirdumistingimus
JNT1 addr	Signaal sisendis T1	(T1)=0
JT1 addr	Signaal sisendis T1	(T1)=1
JNI addr	Signaal sisendis $\overline{INT}$	($\overline{INT}$)=0
DJNZ RN, addr	Üldregistri RN sisu	(RN)$\neq 00_H$

Kõigi vaadeldud käskude esimene bait sisaldab käsukoodi ning teine bait siirdeaadressi. Kui käsu täitmisel kontrollitav siirdumistingimus on tõene, siis laaditakse käsu teises baidis olev siirdeaadress käsuloenduri bittidesse PC_7- PC_0. Näeme, et tingimussiire toimub reeglina püsimälu käsiloleva lehekülje piires. Erandi moodustavad juhtumid, kus üks käsubaitidest paikneb mäluleheküIje viimases pesas. Miks? Vaata punkti 2.2 lõpus analüüsitud juhtumit.

Siirdumistingimuste kontrollimiseks sisaldab arvuti spetsiaalset püsimälu, milles säilitatakse baidiseid testsõnu. Tingimuse kontrollimisel sooritatakse ALU-s loogikatehe tingimust sisaldava baidise sõna ning vastava testsõnaga. Seejärel kontrollitakse spetsiaalse elektroonikaahela abil kas loogikatehte tulem võrdub nulliga või mitte. Sõltuvalt täidetavast siirdekäsust on tingimust sisaldavaks sõnaks kas akumulaatori sisu või lippude ning sisendsignaalide seisu põhjal komplekteeritud baidine sõna või üldregistri sisu.

Näeme, et kasutatav kontrollimismoodus võimaldab analüüsida nii baidist sõna tervikuna kui ka selle suvalise biti seisu. Ka käskude JZ, JNZ või DJNZ täitmisel kontrollitakse aparatuurselt akumulaatoris või dekrementeeritavas registris oleva arvu võrdumist nulliga, mistõttu arvutis puudub spetsiaalne mäluelement nullilipu Z säilitamiseks. Märgime veel, et arvuti plokkskeemis (vt. joonist 2) näidatud lipuregistri täpset otstarvet ning seal säilitatava sõna vormingut ei ole kasutatud kirjanduse abil õnnestunud välja selgitada.

Universaalprotsessoritega võrreldes on meie arvuti tingimussiirete ploki oluliseks puuduseks see, et ta ei võimalda siirdumist mälu mistahes pesas olevale käsule. Samuti puudub otsene võimalus tingimuslikult alamprogrammi siirdumi-

seks või sealt põhiprogrammi naasmiseks.

2.7. Arvuti lähtestamine.

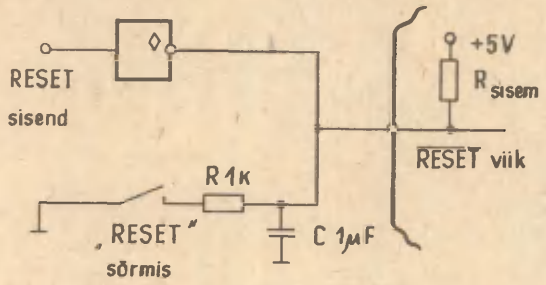
Programmi täitmise aparatuurset katkestamist ning lahenduskäigu automaatset siirdumist püsimälu fikseeritud mälupesas paiknevale käsule on meie arvutis võimalik esile kutsuda kolmel erineval viisil. Kõige kõrgema prioriteediga katkestust põhjustav signaal on sisendi $\overline{\text{RESET}}$ kaudu antav lähtestamissignaal. Prioriteedilt järgmised on väliskatkestussignaal $\overline{\text{INT}}$ ning arvuti sisemine taimerkatkestussignaal. Nende poolt põhjustatud muudatustega arvuti töös tutvume järgmises punktis.

Arvuti lähtestamisele, signaali 0 andmisele lähtestussisendisse $\overline{\text{RESET}}$, reageerib arvuti järgmiselt:

- programmi täitmine katkestatakse;
- käsuloenduri PC ja olekuregistri PSW sisu nullitakse;
- nulli seatakse ka lipud F1, TF ja TBF;
- taimeris paikneva loenduri töö peatatakse;
- lõpetatakse takteerimissignaali väljastamine viigule TO;
- katkestused keelatakse;
- juhul (EA)=0 seatakse P0 viigud kolmandasse olekusse;
- väratite P1 ja P2 viigud seatakse sisenditeks (seisu 1).

Signaali ALE väljastamist ($\overline{\text{RESET}}$)=0 ei katkesta. Kui (EA)=1, siis ei katke ka signaali $\overline{\text{PSEN}}$ väljastamine ning värat P0 käitub nii nagu välismälus paikneva programmi täitmisel. Signaali $\overline{\text{RESET}}$ lõppemisel jätkub programmi täitmine püsimälu panga nr.0 pesas 000_H olevast käsust, valitud on registerpank nr.0 ning pinuviit viitab nullindale pinumälu pesadepaarile.

Elektriline skeem, mis arvuti toitepinge lülimisel tagab automaatse lähtestamise ning võimaldab seda veel suvalisel momendil teha kas välissignaali või sõrmise "RESET" abil, on toodud joonisel 14. Toitepinge lülimisel peab pinge viigul $\overline{\text{RESET}}$ püsima allpool 0,5 V nivood vähemalt 50 ms vältel. Arvuti lähtestamiseks töö käigus tuleb nullsignaali viigul $\overline{\text{RESET}}$ hoida vähemalt nelja masinatsükli täitmiseks kuluva aja vältel (signaali f_0 kuuekümne perioodi vältel).



Joonis 14. Arvuti lähtestamine.

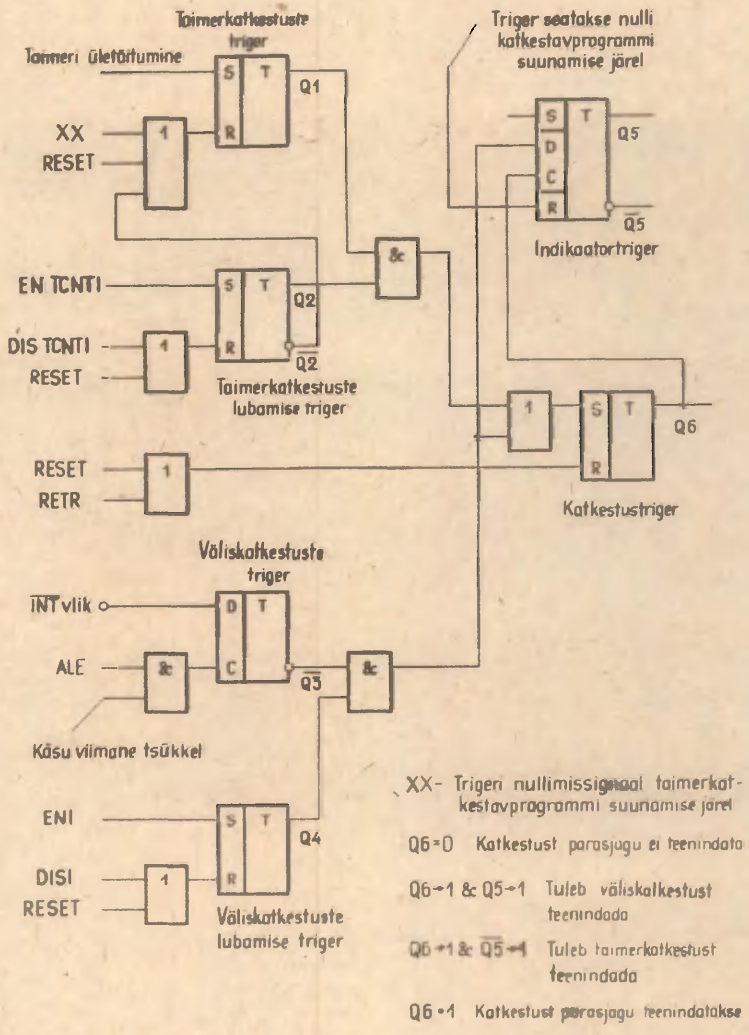
Signaal $\overline{\text{RESET}}$ katkestab ka katkestavprogrammi täitmise ning toimib isegi siis kui katkestused on keelatud.

2.8. Katkestused.

Arvutis on realiseeritud kahenivooline katkestussüsteem. Välisseadmed saavad arvutilt katkestust nõuda signaali 0 seadmisega arvuti katkestussisendisse $\overline{\text{INT}}$. Katkestuse võib esile kutsuda ka arvuti taimerplokis oleva loenduri ületäitumine - üleminek seisust FF_H seisu 00_H. Edaspidi nimetame esimest katkestust väliskatkestuseks ning teist taimerkatkestuseks.

Katkestusploki loogikaskeem on toodud joonisel 15. Mõlemad katkestused on programmiliselt lubatavad ning keelatavad. Taimerkatkestusi lubab käsk EN TCNTI, mis seab taimerkatkestusi lubava trigeri seisu Q2=1. Taimerkatkestused keelab käsk DIS TCNTI, mis selle trigeri nullib. Väliskatkestusi lubav käsk EN I seab väliskatkestusi lubava trigeri seisu Q4=1 ning keelav käsk DIS I - seisu null. Mõlemad katkestused keelatakse automaatselt arvuti lähtestamisel signaaliga $\overline{\text{RESET}}$.

Kui taimerkatkestused on lubatud (Q2=1), siis lülib loenduri ületäitumissignaal taimerkatkestusnõudmisi registreeriva taimerkatkestuste trigeri seisu Q1=1. Kui taimerkatkestused on keelatud, siis $\overline{Q2}$=1 ning taimerkatkestuste trigeri



Joonis 15. Katkestusploki loogikaskeem.

nullimissisendis R on aktiivne signaal 1, mis loenduri ületäitumissignaalil taimerkatkestuste trigerit seisu Q1=1 lülida ei luba. Seega registreerib taimerkatkestuste triger katkestusnõudmise vaid siis, kui taimerkatkestused on lubatud.

Signaali sisendis $\overline{INT}$ kontrollib katkestusplokk regulaarselt iga käsu täitmisel. Väliskatkestusnõudmisi registreeriva D-trigeri lülimine toimub arvuti masinatsüklite kordussageduslikku signaali ALE vältel. Kahetsükliliste käskude täitmisel toimub see käsu teisele masinatsüklile vastava signaali ALE vältel. Katkestust nõudev nullsignaal peab sisendis $\overline{INT}$ vältama vähemalt kolmele masinatsüklile vastava aja jooksul, et tagada väliskatkestuste trigeri lülitumist seisu Q3=1.

Katkestustrigeri ja indikaatortrigeri otstarve on järgmine. Katkestustrigeri väljundsignaali Q6 positiivne front, üleminek seisust 0 seisu 1, on selleks signaaliks, mis poolelioleva käsu täitmise järel põhiprogrammi täitmise katkestab. Sama signaal antakse ka indikaatortrigeri takteerimissisendisse C. Kui katkestustriger läheb seisu Q6=1 väliskatkestusnõudmise tõttu, siis seatakse indikaatortriger seisu Q5=1, kui taimerkatkestusnõudmise tõttu, siis seisu $\overline{Q5}$=1. Seega inditseerib indikaatortriger kumba katkestust tuleb arvutil teenindama hakata. Oluline on siin märkida järgmist. Kui mõlemad katkestusnõudmised laekuvad samaaegselt, siis läheb indikaatortriger ikkagi seisu Q5=1. Seega on väliskatkestustel taimerkatkestuste ees prioriteet. Esmalt teenindatakse väliskatkestust ning alles seejärel järjekorras seisvat taimerkatkestust. Parasjagu teenindatavat taimerkatkestust väliskatkestusnõudmine siiski ei katkesta.

Signaal 1 väljundis Q6 näitab seda, et parasjagu toimub katkestavprogrammi täitmine. Uus üleminek seisust 0 seisu 1 saab siin aset leida alles pärast katkestustrigeri nullimist, mis toimub käsu RETR täitmisel või arvuti lähtestamisel. Seetõttu peab iga katkestavprogramm lõppema käsuga RETR. Katkestavprogrammi täitmise ajal on katkestused reeglina aparatuurselt keelatud ning neid ei saa sel ajal isegi käskudega EN I või EN DCNTI lubada. Katkestuste lubamiseks kat-

kestavprogrammi täitmise ajal tuleb kasutada fiktiivset käsku RETR. Seda võimalust on selgitatud õppevahendi teises vihikus käsu RETR kirjelduses toodud näites.

Katkestusnõudmise laekumisel, signaali Q6 üleminekul seisust 0 seisu 1, lõpetab arvuti esmalt poolelioleva käsu täitmise. Seejärel salvestatakse pinumällu käsuloenduri PC sisu ja olekuregistri PSW neli kõrgemat bitti ning laaditakse käsuloendurisse katkestavprogrammi esimese käsu aadress püsimälus. Katkestavprogrammide algusaadressid on arvutis aparatuurselt fikseeritud. Väliskatkestust teenindava katkestavprogrammi esimene käsk peab paiknema püsimälu panga nr.0 pesas aadressiga 003_H ning taimerkatkestavprogrammi esimene käsk sama panga pesas 007_H. Mõlemad katkestavprogrammid ning nende poolt kasutatavad alamprogrammid peavad tervikuna paiknema püsimälu pangas nr.0, sest katkestavprogrammide täitmise ajal käsud JMP ja CALL mälupanga trigeri TBF seisu käsuloenduri kõrgeimasse bitti PC_{11} ei kanna.

Katkestusnõudmise rahuldamisel nullitakse automaatselt indikaatortriger ning taimerkatkestusnõudmise rahuldamisel ka taimerkatkestuste triger. Seega annulleeritakse taimerkatkestusnõudmine selle rahuldamisel automaatselt. Tõsisem on probleem väliskatkestusnõudmise õigeaegse annulleerimisega. Arvutil puudub spetsiaalne juhtimissignaal, mis katkestust nõudvale seadmele teataks, et nõudmine on rahuldatud. Ühelt poolt peab aktiivne nullsignaal sisendis $\overline{INT}$ püsima katkestusnõudmise rahuldamiseni. See ajavahemik võib kujuneda üsna pikaks juhul, kui nõudmine esitatakse taimerkatkestuse ajal. Teiselt poolt peab katkestussignaal sisendist $\overline{INT}$ olema kindlasti kõrvaldatud enne arvuti väliskatkestavprogrammist põhiprogrammi naasmist. Vastandjuhul siirdub programmi täitmine kohe katkestavprogrammi tagasi. Viimase õnnetuse vältimiseks võib naasmiskäsu RETR eel organiseerida käsu JNI abil katkestussignaali kõrvaldamist ootav tsükkel. Probleemi korrektseks lahendamiseks tuleb lisaks sellele väratite P1 või P2 üks viik loovutada katkestusnõudmise rahuldamist signaliseerivaks väljundviiguks, mille kaudu saab katkestavprogrammi alguses väljastada vajaliku kestusega vastus-

signaali.

Katkestavprogrammi siirdumisel salvestab arvuti põhiprogrammi naasmise aadressi (käsuloenduri PC sisu) ning programmiolekusõna PSW neli kõrgemat bitti automaatselt pinumällu. Naasmiskäsu RETR täitmisel nende registrite sisu ennistatakse. Katkestatud protsess ennistub täielikult siiski vaid siis, kui katkestavprogramm ei riku põhiprogrammi poolt kasutatavate registrite ning muutmälu pesade sisu. Selle tagamiseks soovitatakse üks registerpankadest reserveerida põhiprogrammi ning teine katkestavprogrammide tarbeks. Katkestavprogrammi alguses tuleb siis käsuga SEL RB valida katkestavprogrammidele reserveeritud registerpank ning akumulaatori sisu üldregistrisse salvestada. Katkestavprogrammi lõpus tuleb enne naasmiskäsku RETR akumulaatori sisu ennistada ning valida põhiprogrammile reserveeritud registerpank.

Teatavasti loendab loendusrežiimi seatud taimer viigu T1 kaudu arvutisse sisestatavaid impulsse. See võimaldab viiku T1 kasutada täiendava väliskatkestussisendina. Kui taimerkatkestused on lubatud ning loendusregistrisse on käsuga MOV T,A laaditud arv FF_H, siis põhjustab loenduri käivitamise järel signaali T1 esimene üleminek seisust 1 seisu 0 loenduri ületäitumise ning taimerkatkestuse. Katkestussüsteemi täiendava laiendamise võimalusi selgitatakse põhjalikumalt kolmandas peatükis.

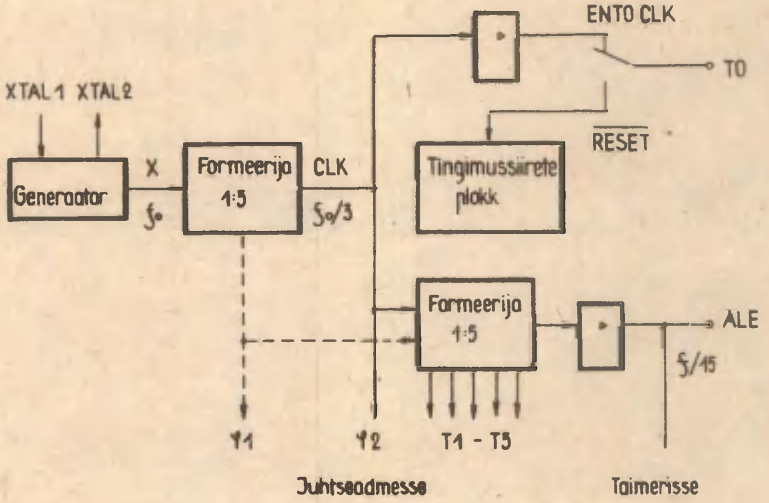
2.9. Juhtimis- ning takteerimisplokk.

Selle ploki koosseisu kuuluva juhtseadme ülesandeks on arvuti erinevate sõlmede koostöö ning häireteta andmevahetuse korraldamine. Juhtseade valib juhtimisalgoritmi vastavalt välissignaali EA poolt määratud töörežiimile, seda täpsustavate välissignaalide $\overline{RESET}$, $\overline{SS}$, T0 ja $\overline{INT}$ seisule ning parasjagu täidetava käsu koodile. Juhtimisalgoritmi realiseerimiseks väljastab ta arvuti sõlmedele nende tööd korraldavaid ning sünkroniseerivaid juhtimissignaale.

Käskude täitmine toimub arvutis masinatsüklite kaupa. Sõltuvalt käsust kulub selleks kas üks või kaks masinatsük-

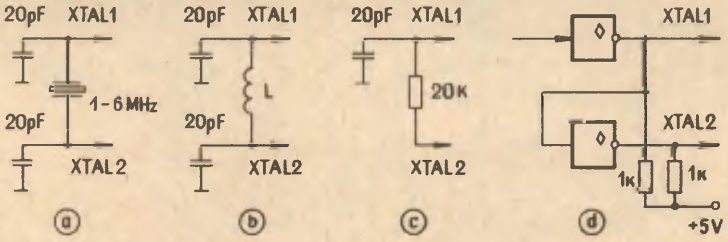
lit. Iga masinatsükkel koosneb omakorda viiest taktist. Juhtseadme ning seega ka kogu arvuti töörütmi määravad takteerimisseadme poolt genereeritud signaalid.

Takteerimisseadme põhisõlmedeks on signaalgeneraator ning kaks impulsside formeerijat. Seadme tööpõhimõtet illustreerib joonis 16.



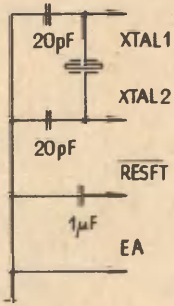
Joonis 16. Takteerimisseadme põhimõtteskeem.

Signaalgeneraator on võimendi, mille sisendiks on viik XTAL1 ning üheks väljundiks viik XTAL2. Nende viikudega tuleb ühendada generaatori töösagedust määrav sageduslikult selektiivne tagasiside ahel või väline signaalgeneraator. Välisteguritest nõrgalt sõltuva stabiilse takteerimissignaali saamiseks tuleb viikudega XTAL ühendada kvartskristall nii, nagu on näidatud joonisel 17a. Kvartskristalli sagedus peab arvutil KM1816BE48 ja PBE035 kuuluma vahemikku 1 - 6 MHz, arvuteil KM1816BE49 ja KM1816BE39 vahemikku 1 - 11 MHz. Takteerimissageduse etteandmiseks võib generaatori viikudega



Joonis 17. Tagasisideahela või välise signaalgeneraatori ühendamine viikudega XTAL.

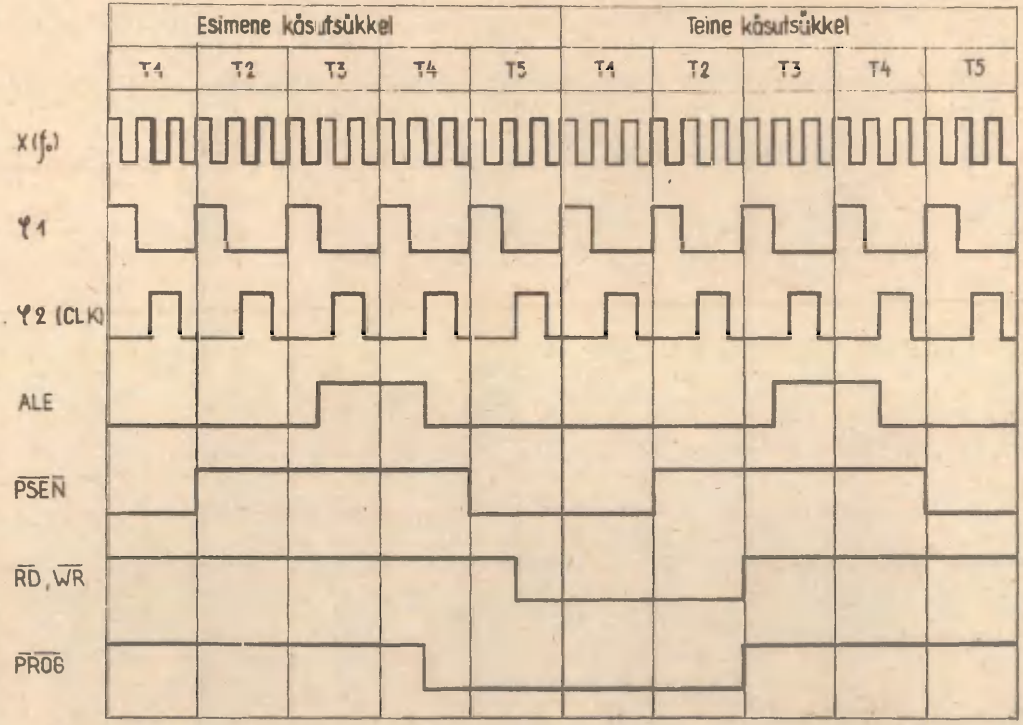
XTAL ühendada ka RC-ahela, LC-ahela või välise signaalgeneraatori nii, nagu on näidatud joonistel 17b, c ja d. Joonisel 18 on toodud skeem, mis tagab arvuti lähtestamise toite lülimisel ning arvuti töö sisemise püsimäluga. Generaatori väljundsignaal X on samas faasis viigul XTAL2 oleva signaaliga. Selle signaali sagedust tähistame f_0-ga.



Joonis 18. Kvartsi ühendamine.

Väljundsignaal X antakse sagedusjagajal baseeruva formeerija sisendisse. Formeerija väljundist saame takteerimissignaali $\varphi 2$, mille sagedus on $f_0/3$ ning perioodi pikkus on võrdne arvuti ühe töötakti kestusega. Algallikas [7] joon. 12-15 toodud ajadiagrammide alusel võib oletada, et juhtseade kasutab kahefaasilist takteerimissignaali $\varphi 1$ ja $\varphi 2$ (joonis 19). Kuna algallikates toodud skeemidel ei ole signaali $\varphi 1$ näidatud ega tekstis kommenteeritud, siis oleme joonisel 16 ta tähistanud punktiiriga.

Käsu ENTO CLK täitmise järel on arvuti viik TO seatud signaali $\varphi 2$ e. CLK väljundiks ning tingimussiirde käsud JTO ja JNTO ei ole enam kasutatavad. Viiku TO saab uuesti sisendiks seada vaid arvuti lähtestamissignaali $\overline{\text{RESET}}$ abil.



Joonis 19. Takteerimis- ning juhtimissignaalide ajadiagrammid.

Teine formeerija väljastab juhtseadmele taktiimpulsse T1 - T5 ning masinatsüklite korduvussageduslikku signaali ALE. Viimase sagedus on $f_0/15$ ning ta olemasolu väljundviigul ALE on selle tunnuseks, et takteerimisseade töötab korrektselt. Signaali ALE kasutab sisendsignaalina taimerrežiimi seatud taimer.

Vajaduse korral väljastab juhtimis- ja takteerimisplokk lisaks juhtimissignaalile ALE veel täiendavaid laiendusskeemide tööd korraldavaid signaale $\overline{\text{PSEN}}$, $\overline{\text{RH}}$, $\overline{\text{WR}}$ ning $\overline{\text{PROG}}$. Signaalide T1 - T5 kuju ning täpne otstarve ei ole teada. Teiste siin mainitud signaalide ajadiagrammid on toodud joonisel 19. Arusaamatuste vältimiseks mainime, et juhtimissignaalid $\overline{\text{PSEN}}$, $\overline{\text{RH}}$, $\overline{\text{WR}}$ ja $\overline{\text{PROG}}$ ei ole regulaarsed signaalid. Signaali $\overline{\text{PSEN}}$ väljastab arvuti välise püsimälu lugemisel, signaali $\overline{\text{RH}}$ käskude MOVX A,@R ning INS BUS täitmisel, signaali $\overline{\text{WH}}$ käskude MOVX @R,A ning OUTL BUS täitmisel ja signaali $\overline{\text{PROG}}$ käskude MOVD A,P, MOVD P,A, ANLD P,A ja ORLD P,A täitmisel. Korras kiibi korral ei esine kunagi situatsiooni, kus samaaegselt väljastatakse neist signaalidest kahte või enamat.

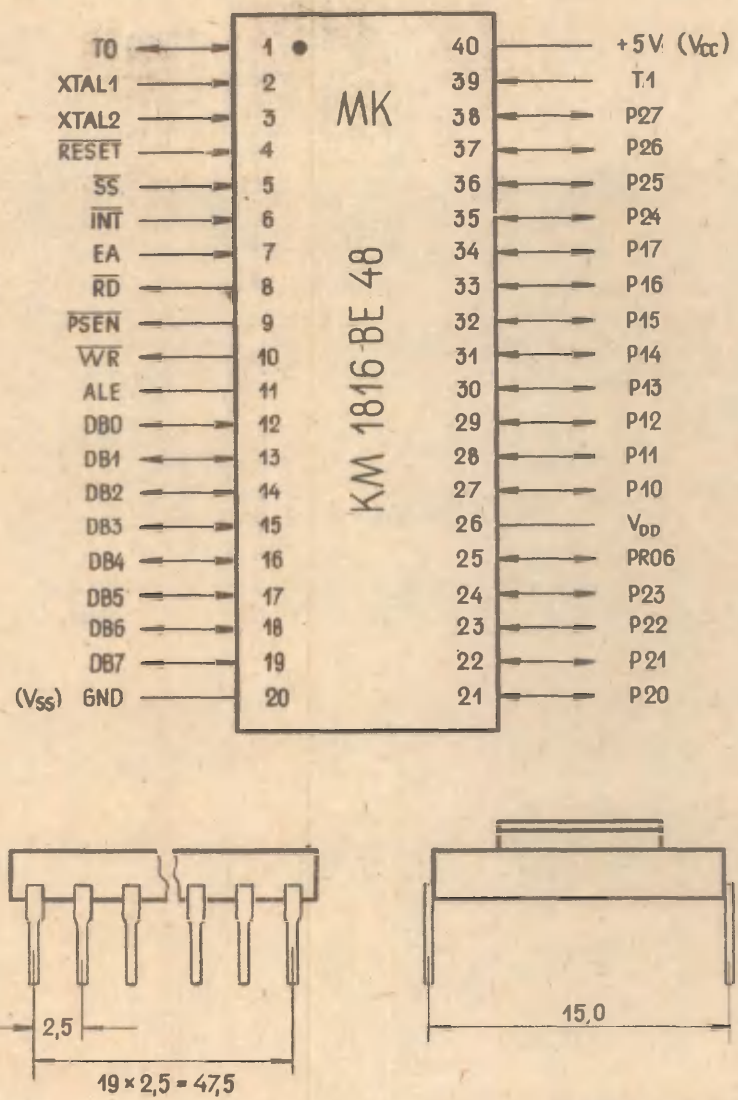
3. Arvuti ühendusskeem.

Arvuti ühendusskeem on toodud joonisel 20. Järgnevalt esitame kõigi viikude ning neile vastavate signaalide otstarbe lühida selgituse.

DB0 - DB7 (inglise keeles - databus) on kahesuunalise sisend-väljundvärati P0 viigud. Arvuti laiendamisel välise püsi- või muutmäluga kasutatakse neid viike multipleksse aadressi- ning andmesiini viikudena. Käsustikus - BUS.

P10 - P17 ja P20 - P27 (inglise keeles - port) on kvaasikahesuunaliste sisend-väljundväratite P1 ja P2 viigud. Värati P2 viike P20 - P23 kasutatakse veel välise püsimälu aadressi nelja kõrgema biti väljastamiseks ning andmevahetuseks laiendusväratitega P4, P5, P6 ja P7.

ALE (inglise keeles - address latch enable) on viik, mille kaudu töötav arvuti väljastab iga masinatsükli alguses



Joonis 20. Arvuti ühendusskeem.

sünkroimpulsi ALE. Seda kasutatakse viikude DB0 - DB7 kaudu väljastatava aadressi laadimiseks välismälu aadressi lukkregistrisse või takteerimissignaalina, mille sagedus on $f_0/15$.

$\overline{RD}$ (inglise keeles - read) on lugemissignaali $\overline{RD}$ väljundviik, mida kasutatakse andmete välisest muutmälust või välisseadmest viikude DB0 - DB7 kaudu arvutisse lugemisel

$\overline{WR}$ (inglise keeles - write) on salvestussignaali väljundviik, mida kasutatakse andmete välismällu või välisseadmesse salvestamisel viikude DB0 - DB7 kaudu.

$\overline{PSEN}$ (inglise keeles - program store enable) on lugemissignaali väljundviik, mida kasutatakse käsukoodide välisest püsimälust viikude DB0 - DB7 kaudu lugemisel.

EA (inglise keeles - external accsess) on sisendviik arvuti töörežiimi valikuks.

$\overline{SS}$ (inglise keeles - single step) on sisendviik, mida kasutatakse arvuti juhtimiseks programmi käskhaaval täitmise režiimis. Aktiivse nullsignaali ilmumine sisendisse $\overline{SS}$ peatab arvuti pärast poolelioleva käsu täitmist.

$\overline{INT}$ (inglise keeles - interrupt) on väliskatkestuste sisendviik, mis on kontrollitav ka tingimusliku siirdekäsuga JNI.

T0 (inglise keeles - test) on sisend-väljundviik. Lähtestamise järel on ta tingimuslike siirdekäskudega JT0 ning JNT0 kontrollitav sisendviik. Käsu ENT0 CLK täitmise järel väljastab arvuti selle viigu kaudu takteerimissignaali sagedusega $f_0/3$. Programmi salvestamisel arvuti sisemisse püsimällu kasutatakse seda viiku kas salvestamise või salvestatu kontroll-lugemise režiimide valikuks.

T1 (inglise keeles - test) on sisendviik, millel olev signaal on kontrollitav tingimuslike siirdekäskudega JT1 ning JNT1. Käsuga STRT CNT on ta seatav arvuti taimeris paikneva loenduri sisendiks.

$\overline{RESET}$ (inglise keeles - reset) on sisendviik, mida kasutatakse arvuti lähtestamiseks. Programmi salvestamisel arvuti sisemisse püsimällu laadib selle viigu kaudu antav impulss viikude DB0 - DB7 ja P20 - P21 kaudu sisestatava aadressi käsuloendurisse.

XTAL1 ja XTAL2 (inglise keeles - external crystal) on

viigud arvuti tööd takteeriva kvartskristalli, RC või LC ahela või välise generaatori ühendamiseks arvutiga.

PROG (inglise keeles - program) on väljundviik, mida kasutatakse laiendusväratite P4 - P7 juhtimiseks või sisendviik, mida kasutatakse programmi salvestamisel arvuti sisemisse püsimällu.

GND (inglise keeles - ground) on viik, millega ühendatakse arvuti toiteallika nullklemm.

V_{CC} on toitepinge +5 V viik.

V_{DD} on viik, millele töörežiimis antakse pinge +5 V ning programmi püsimällu salvestamisel pinge +24,5 V.

Näeme, et enamus viike on meile juba tuttavad. Ülejäänutega tutvume põhjalikumalt õppevahendi järgmistes peatükkides arvuti erinevate töörežiimide selgitamisel.

II ARVUTI TÖÖREŽIIMID

Arvuti töörežiimi määrab signaal viigul EA ning täpsustavad signaalid viikudel $\overline{INT}$, $\overline{RESET}$, $\overline{SS}$ ja T0. Signaal 0 viigul EA seab arvuti sisemises või välises püsimälus paikneva programmi täitmise režiimi. Selles režiimis alustab arvuti lähtestamise järel alati programmi täitmist käsust, mis paikneb sisemise püsimälu nullindas pesas. Kui käsuloenduri PC sisu saab seejuures suuremaks sisemise püsimälu viimase pesa aadressist, siis jätkab arvuti automaatselt käsukoodide lugemist välisest püsimälust. Arvutite KM1816BE48 ja KM1816BM49 jaoks on vaadeldav režiim põhiliseks töörežiimiks. Arvuteil KM1816BE39 ja PBE035 sisemist püsimälu ei ole. Neil tuleb viigule EA anda signaal 1.

Signaal 1 viigul EA seab arvuti (sõltumata tüübist) välises püsimälus paikneva programmi täitmise režiimi. Lähtestamise järel alustab arvuti programmi täitmist käsust, mis paikneb välise püsimälu nullindas pesas. Arvuteile KM1816BE39 ja PBE035 on see põhiliseks töörežiimiks. Arvutite KM1816BE48 ja KM1816BE49 korral nimetame režiimi, milles sisemist püsimälu ei kasutata, häälestusrežiimiks.

Mõlemas seni vaadeldud töörežiimis toimivad arvutile välised juhtimissignaalid $\overline{RESET}$, $\overline{INT}$ ja $\overline{SS}$. Signaal $\overline{RESET}$ põhjustab arvuti lähtestamise. Kui väliskatkestused on lubatud, siis katkestab signaal $\overline{INT}$ põhiprogrammi täitmise ning suunab lahenduskäigu katkestavprogrammi. Signaali $\overline{SS}$ kasutatakse programmi käskhaaval täitmisel. Kui signaal $\overline{SS}$ läheb seisu 0, siis arvuti lõpetab poolelioleva käsu täitmise ning peatub.

Arvuti seadmiseks sisemise püsimälu programmeerimise või sinna salvestatu kontroll-lugemise režiimi tuleb viigule EA anda pinge, mis kuulub vahemikku 21,5 - 24,5 V. Kui viigul T0 on seejuures signaal 0, siis on tegemist püsimälu programmeerimisega, vastandjuhul salvestatu kontroll-lugemisega.

Lisaks juba mainitud töörežiimidele käsitleme selles peatükis ka arvuti andmevahetust välise muutmäluga, kuna see toimub sarnaselt käsukoodide lugemisele välisest püsimälust.

Andmevahetust laiendusväratitega P4, P5, P6 ja P7 vaatleme õppevahendi viimases peatükis.

Arvuti tööpõhimõtte paremaks selgitamiseks kasutame signaalide ajadiagrammide visandeid. Arvuti KM1816BE48 ajadiagrammide detailsed joonised koos neil kujutatud signaalide staatiliste ning dünaamiliste parameetrite arvväärtustega on esitatud õppevahendi lisas.

4. Sisemises püsimälus paikneva programmi täitmine.

Teatavasti jagunevad arvuti käsustikus olevad käsud ühebaidisteks ühetsüklilisteks käskudeks, ühebaidisteks kahetsüklilisteks ning kahebaidisteks kahetsüklilisteks käskudeks. Alustame kõige lihtsamast - sisemises püsimälus paiknevatest ühetsüklilistest käskudest koosneva programmi täitmisest, mil laiendusmälude ning sisend-väljundväratite poole pöördumist ei toimu. Arvuti tööd sel lihtsaimal juhul illustreerib joonis 21.

Masinatsükkel					Masinatsükkel				
T1	T2	T3	T4	T5	T1	T2	T3	T4	T5
Lug.	Deš.	Täitmine			Lug.	Deš.	Täitmine		
aadr.	PC+1	Järgm. käsu aadr.			PC+1	Järgm. käsu aadr.			

Joon. 21. Ühetsükliliste sisemises püsimälus paiknevate käskude täitmine.

Iga masinatsükli esimese takti T1 vältel loeb arvuti käsukoodi käsuregistrisse. Juhtimisalgoritmi täpsustamiseks tarvilik käsukoodi dešifreerimine toimub teise takti T2 vältel ning käsu täitmine ülejäänud taktide T3 - T5 vältel.

Püsimälu adresseeritakse arvutis käsuloenduri PC abil. Pärast käsukoodi lugemist kasvatatakse käsuloenduri sisu ühe võrra ning paralleelselt käsu täitmisega esitatakse püsimä-

lule juba järgmise käsupesa aadress. Püsimälul on käsukoodi lugemiseni sellest pesast nüüd piisavalt aega loetava valmisseadmiseks.

Kuna sisemist püsimälu adresseeritakse mitu töötakti enne koodi lugemist, siis võib selgusetuks jääda millisest masinatsükli taktist algab lähtestamisjärgse esimese käsu täitmine. Selgub, et ka esimene masinatsükkel algab taktist T1 - käsukoodi lugemisest. Lähtestamissignaali andmisel sisendisse $\overline{\text{RESET}}$ nullitakse automaatselt käsuloenduri sisu ning püsimälule esitatakse nullinda mälupesa aadress. Tehniliste tingimuste kohaselt peab lähtestamissignaal kestma vähemalt neljale masinatsüklile vastava aja vältel ning püsimälule jääb loetava valmisseadmiseks aega enam kui küllalt.

Erandlikuks käsuks ühetsükliliste käskude seas on käsk ENTO CLK. Selle täitmise järel väljastab arvuti viigule TO takteerimissignaali CLK. Ülejäänud ühetsüklilistest käskudest koosneva programmi täitmisel puudub igasugune võimalus väliste vahenditega arvuti sisulist tööd jälgida. Ainsateks formaalseteks välisilminguteks on signaalid viikudel XTAL ja ALE. Sarnane on situatsioon ka nende kahetsükliliste käskude korral, mis sisend-väljundväratite kaudu infot ei vaheta ning välise muutmälu poole pöördumist ei vaja. Nende käskude täitmine toimub sarnaselt joonisel 21 toodud ajadiagrammile. Teine masinatsükkel kulub täiendava sisemise infovahetuse-töötluse korraldamiseks. Arvuti kasutajal puudub praktiline vajadus täpselt teada, kuidas see toimub. Nende käskude ajadiagrammid, milliste korral selline vajadus võib tekkida, on algallikates toodud ning meil on need esitatud tabelis 2.

Eraldi kommentaare vajab tabelis toodud andmetest töötakt T4. Selle töötakti keskel läheb signaal viigul ALE seisust 1 seisu 0 (vt. joonist 19). Seega antakse taimeris oleva loenduri sisendisse aktiivne signaal iga masinatsükli töötakti T4 keskel, ka siis, kui see tabelis näidatud ei ole. Välismälude poole pöördumisel tuleb ALE tagumist fronti kasutada mälupesa aadressi fikseerimiseks.

Tingimuslike siirdekäskude puhul on tabelis esimese

Käskude täitmise ajadiagramm

Tabel 2.

Käsu mnemokood	Esimene käsutsükkel					Teine käsutsükkel				
	T1	T2	T3	*T4	T5	T1	T2	T3	*T4	T5
Ting. siire	Käsukoodi pushmalust lugemine		Tingim. kontr.	Tingim. kontr.	-	Vahetu arvu pushmalust lugemine	-	(PC):= aadr.	-	-
ANL P,#data	Käsuloenduri PC sisu kasvatamine (PC+1)		-	Täitmert sisu kasvatamine ühe võrra	Väratis oleva arvu lugemine		-	Käsuloenduri sisu kasv. (PC+1)	Tuleml laadimine varatisse	-
ORL P,#data			-				-			-
ANL BUS,#data			-				-			-
ORL BUS,#data			-				-			-
ANLD P,A			Programmeer. sõna valj.		Andmete valj. P20-P23		Andmete lugemine			-
ORLD P,A										-
MOVD P,A										-
MOVD A,P			-		-					-
IN A,P			-		-					-
INS A,BUS			-		-					-
MOVX A,@R			Aadr. valj.		Andmete valj.					-
MOVX @R,A			-		-					-
OUTL P,A			-		-					-
OUTL BUS,A			-		-					-

Tabel 2 järg

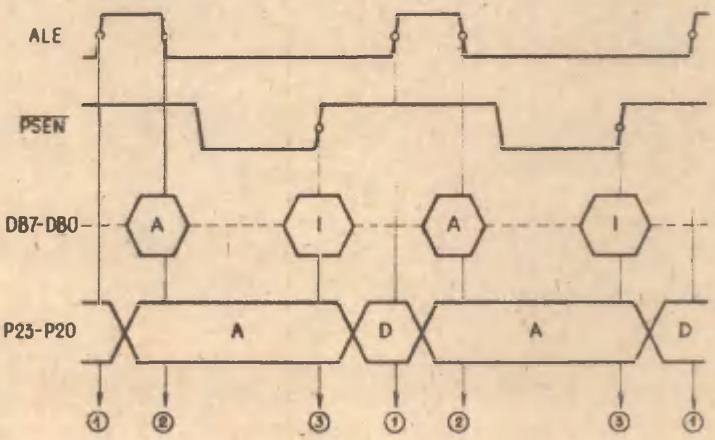
Mnemokood	T1	T2	T3	*T4	T5
STRT T	Käsukoodi püsimälust lugemine	Käsuloenduri PC sisu kasvatamine (PC+1)	-	-	Loenduri käivitam.
STRT CNT			-	-	Loenduri käivitam.
STOP TCNT			-	-	Loenduri seiskam.
EN I			-	Katkest. lubamine	
DIS I			-	Katkest. keelam.	
ENTO CLK			-	TO sead. väljund.	

masinatsükli lahtris T4 märgitud "tingimuse kontrollimine". Algallika [7] tabeli 12-11 andmetel toimub selles taktis vaid viigul T1 oleva signaali kontrollimine ja sedagi vaid firma poolt hiljem täiustatud arvuti tüüpides. Algallikas [1] joon. 25 on selles lahtris arusaamatu tekst "инкрем. по условию".

Märgime, et tabelis 2 ning joonisèl 19 toodud ajadiagrammid ei ole piisavad arvuti töö kôigi aspektide täielikuks môistmiseks. Toome ühe näite. Tabelist selgub, et käsu EN I täitmisel lubatakse katkestused taktis T4. Teisalt (vt. punkti 2.8.) on teada, et arvuti registreerib katkestusnôudmise juhul, kui katkestused on lubatud, sisendis $\overline{INT}$ on signaal 0, (ALE)=1 ning on käsu viimane tsükkel. Toodud andmete pôhjal ei selgu kas juhul ($\overline{INT}$)=0 algab katkestuse teenindamine käsu EN I täitmisele järgnevast masinatsüklist vôi alles pärast järgmise käsu täitmist. Ehk teisiti - kas käsu EN I täitmine toimub taktis T4 enne ALE tagumist fronti vôi mitte. Asja katseline uurimine näitab, et katkestuse teenindamine algab kohe käsu EN I täitmise järel. Sarnaseid küsimusi kerkib arvuti tulevasel kasutajal ilmselt hulgi. Kahjuks ei vôimalda käesoleva ôppevahendi maht kôiki probleeme ammendavalt käsitleda.

5. Välise püsimälu lugemine.

Monoliitarvutite seeria 1816 sisemist püsimälu omavad arvutid pöörduvad käsukoodide või vahetute andmete lugemiseks automaatselt välise püsimälu poole niipea, kui käsuloenduri PC sisu saab programmi täitmisel suuremaks sisemise püsimälu viimase pesa aadressist. Arvutite KM1816BE48 korral leiab see aset siis, kui (PC) > $1023_{10}=3FF_H$ ning arvutite KM1816BE49 korral, kui (PC)> $2047_{10}=7FF_H$. Kõik antud seeria arvutid, sõltumata tüübist, pöörduvad välise püsimälu poole ka siis, kui viigul EA on signaal 1. Seejuures kasutab arvuti väratit P0 multipleksse kahesuunalise aadressi- ning andmesiini väratina ja värati P2 nelja madalamat bitti aadressiväratina. Arvuti ja välise püsimälu koostöö sünkroniseerimi-



④ - Värati P2 kaudu väljastatavate andmete fikseerimine
② - Aadressi fikseerimine
③ - Sõna lugemine välisest püsimälust
A - Aadress D - Andmed I - Mälusõna

Joonis 22. Lugemine välisest püsimälust.

seks väljastab arvuti signaale ALE ja $\overline{PSEN}$. Info lugemist välisest püsimälust illustreerib joonisel 22 toodud ajadiagramm (vt. ka joonist 19 ja tabelit 2).

Käsukoodi või vahetult antud arvu püsimälust lugemine toimub masinatsükli esimese takti T1 lõpus. Selleks valmistumine algab juba eelmise masinatsükli taktis T3, mil mälupesa aadressi esitamiseks väljastab arvuti värati P0 viikudele DB0 - DB7 käsuloenduri madalama baidi PC0 - PC7 ning värati P2 viikudele P20 - P23 käsuloenduri kõrgema poolbaidi PC8 - PC11. Signaali ALE tagumine front järgmise takti T4 keskel on märguandeks selle kohta, et aadress on siinil stabiilne. Kuna siini on peatselt vaja mälupesa sisu lugemiseks arvutisse, siis peab väline skeem aadressi madalama baidi signaali ALE tagumise frondi ajal fikseerima.

Välismälust lugemise signaali $\overline{PSEN}$ väljastab arvuti töötakti T5 alguses pärast värati P0 viikude kõrgeoomilisse olekusse seadmist. See signaal lubab püsimälul andmebaidi siinile väljastada. Mälu poolt siinile seatav sõna peab siinil stabiilne olema lugemistakti T1 lõpus signaali $\overline{PSEN}$ tagumise frondi ajal, mil toimub andmebaidi arvutisse lugemine. Veidi pärast signaali $\overline{PSEN}$ tagumist fronti seab arvuti siini jälle kolmandasse olekusse ning valmistub järgmise aadressi väljastamiseks.

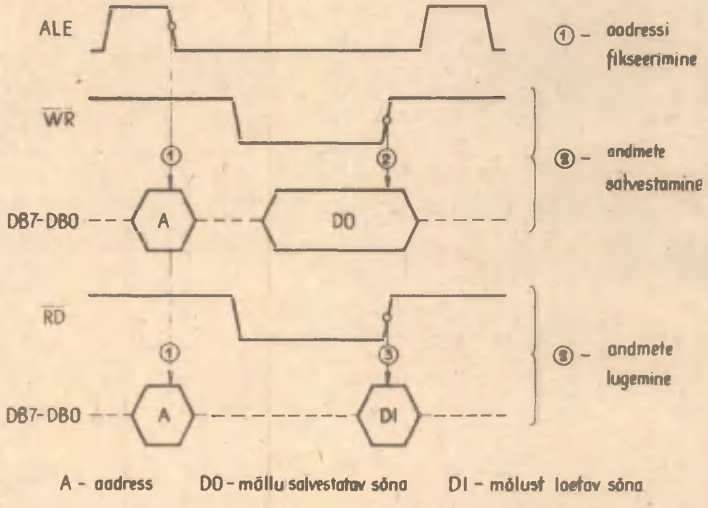
Aadressi kõrgema poolbaidi väljastamise ajaks teisaldatakse värati P2 viikude fiksaatorites olev info vaheregistrisse hoiule. Signaali $\overline{PSEN}$ tagumise frondi järel fiksaatorite sisu ennistatakse. See võimaldab äärmise vajaduse korral ka viike P20 - P23 väljundvärati viikudena kasutada. Sellisel juhul tuleb arvutit väliselt laiendada skeemiga, mis nende viikude kaudu väljastatava info fikseerib signaali ALE esimese frondi ajal.

Märgime veel, et lähtestamise vältel, mil $\overline{RESET}$=0, väljastab arvuti regulaarselt signaali ALE ning juhul EA=1 ka signaali $\overline{PSEN}$. Kui EA=1, siis väratid P0 ja P2 käituvad lähtestamise vältel nii nagu välise püsimälu lugemise režiimis. Detailsemalt on seda juhtu käsitletud paragrahvis 7.

6. Andmevahetue välise muutmäluga.

Andmevahetust välise muutmäluga korraldavad ühebaidised kahetsüklilised käsud MOV A,@R ja MOV @R,A. Käsu esimeses tsüklis toimub käsukoodi lugemine ning dešifreerimine ja teises tsüklis selle täitmine. Välise muutmälu pesade adresseerimisel kasutab arvuti kaudset adresseerimismoodust - käsukoodi aadressivälja bitid viitavad ühele aadressiregistritest R0, R1 või R0', R1', milles paikneb mälupesa aadress. Kuna aadress on 8 bitine, siis saab välise muutmälu maht olla vaid 256 mälupesa.

Sarnaselt välise püsimälu lugemisega kasutatakse ka välise muutmälu poole pöördumisel väratit P0 kahesuunalise multipleksse aadressi- ja andmesiini väratina. Värati P2 viigud välise muutmäluga toimuvas infovahetuses ei osale. Arvuti ning välise muutmälu tööd sünkroniseerivad aadressi fikseerimise signaal ALE, lugemissignaal $\overline{RD}$ ning salvestussignaal $\overline{WR}$. Andmevahetuse diagrammid on toodud joonisel 23 ning tabelis 2.



Joonis 23. Andmevahetue välise muutmäluga .

Kuna siini viigud DB7 - DB0 on kasutuses nii aadressi- kui ka andmesiinina, peab välise muutmälu skeem sisaldama aadressi fiksaatorit. Signaali ALE tagumise frondi ajal on arvuti poolt mälule väljastatav mälupesa aadress siinil stabiilne. Seega tuleb ALE tagumist fronti kasutada aadressi fikseerimiseks.

Info välisesse muutmällu salvestamisel teatab signaali $\overline{WR}$ esimene front mälule, et toimub andmete salvestamine. Andmed on siinil stabiilsed $\overline{WR}$-impulsi tagumise frondi ajal, mis peab salvestamise lõpule viima.

Info välisest muutmälust lugemisel teatab viigule $\overline{RD}$ antava signaali esimene front mälule, et arvuti on siini sisendiks lülinud ning mälu võib loetava pesa sisu siinile kanda. Andmed peavad siinil stabiilsed olema $\overline{RD}$-impulsi tagumise frondi ajal, mil toimub nende arvutisse lugemine.

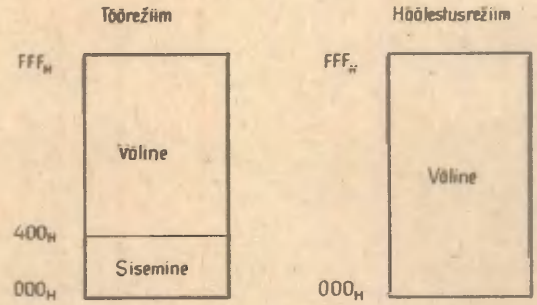
Välise püsimälu ning muutmälu kasutamise näiteid on kirjeldatud õppevahendi viimases peatükis.

7. Hääleetusrežiim.

Sisemist püsimälu omavad monoliitarvutid võivad püsimälu kasutada kahel erineval moel. Kui viigul EA on signaal 0, siis kasutavad nad sisemist või välist püsimälu. Lähtestamise järel alustavad nad alati sisemises püsimälus paikneva programmi täitmist. Kui käsuloenduri sisu saab seejuures suuremaks viimase sisemise püsimälu pesa aadressist, siis pöörduvad need arvutid automaatselt välise püsimälu poole. Arvutil KM1816BE48 algab välıne püsimälu selles režiimis aadressist 400_H ning arvutil KM1816BE49 - aadressist 800_H. Kui viigul EA on signaal 1, siis on sisemise püsimälu kasutamine keelatud ning lähtestamise järel hakkab arvuti täitma programmi, mille esimene käsk on välise püsimälu nullindas pesas. Öeldut illustreerib joonis 24.

Vaadeldud arvutitel on põhiliseks töörežiim, mille korral programmi täitmine algab sisemises püsimälus paiknevast programmist. Teist moodust kasutatakse arvuti häälestamisel, programmi silumisel ning diagnostika eesmärkidel. Seetõttu

nimetame sisemist püsimälu omavate arvutite korral esimest reŽiimi lihtsalt tööreŽiimiks ning teist - häälestusreŽiimiks.



Joonis 24. Arvuti KM1816ВЕ48 püsimälu aadressid.

Arvuti häälestamisel ning programmide silumisel kasutatakse sageli moodust, mida erialases kirjanduses nimetatakse mälu emuleerimiseks. Püsimälu emuleerimisel ühendatakse arvutiga püsimälu kiibi asemel muutmäludel baseeruv seade e. mälu emulaator, mis on ühendatav veel mingi teise arvutiga. Selle teise arvuti abil saab vajaliku programmi emulaatorisse salvestada. Seejärel lülitatakse emulaatormälu häälestatava arvutiga ning häälestatav arvuti seatakse häälestusreŽiimi - emulaatoris paiknevat programmi täitma. Nüüd on kohe näha, kas häälestatav arvuti või silutav programm teevad mis tarvis või mitte. Viimasel juhul saame mälu lülida jälle abiarvuti mälupiirkonda ja programmis vajalikud muudatused teha. Protsessi korratakse seni, kuni häälestatav arvuti töötab vajaliku programmiga laitmatult. Seejärel salvestatakse see programm püsikasutamiseks arvuti püsimällu või välise püsimälu kiipidesse.

Seeriatootmisse minevate arvutite ning nende programmvarustuse korrektsel projekteerimisel luuakse ka võimalused arvuti korrasoleku kontrollimiseks - valmistatakse diagnostikaseadmed ja diagnostikaprogrammid. Ka meie monoliitarvutite korral on see võimalik.

Sisemist püsimälu omavate arvutite korral võime diagnostikaprogrammi salvestada välise püsimälu algusse, kuhu arvuti tavalises töörežiimis kunagi ei pöördu. Lülides arvuti häälestusrežiimi, hakkab see lähtestamise järel välise püsimälu alguses paiknevat diagnostikaprogrammi täitma.

Sisemise püsimäluta monoliitarvutite korral võib diagnostika eesmärkidel kasutada spetsiaalset püsimälu kiipi, millega saame välise püsimälu kiibi asendada.

Korrektne režiimi vahetamine peab toimuma lähtestamise ajal, mil viigul $\overline{\text{RESET}}$ on signaal 0. Välise püsimäluga toimuva infovahetuse ajadiagrammid on häälestusrežiimis ning harilikus töörežiimis ühesugused (vt. joonist 22). Oluliseks erinevuseks nende režiimide vahel on arvuti käitumine lähtestamise vältel. Kui $\overline{\text{RESET}}$=0 ning EA=1, siis arvuti väljastab lisaks signaalile ALE regulaarselt ka välise püsimälu lugemise signaali $\overline{\text{PSEN}}$ ning värat P0 ja värati P2 neli madalamat bitti käituvad nii nagu välismälu regulaarse lugemise ajal. Signaali ALE tagumise frondi ajal on mainitud viigud väljunditeks, mille kaudu arvuti väljastab nulli seatud käsuloenduri PC sisu. Seega toimub lähtestamise ajal välise püsimälu regulaarne adresseerimine ning mälu on arvuti esimeses töötaktis valmis arvutile esimese käsu koodi edastama. See on nii ka sisemise püsimäluta arvutite KM1816BE39 ja PBE035 korral. Neile on režiim EA=1 ainsaks ning põhiliseks töörežiimike.

8. Sammtalitlus.

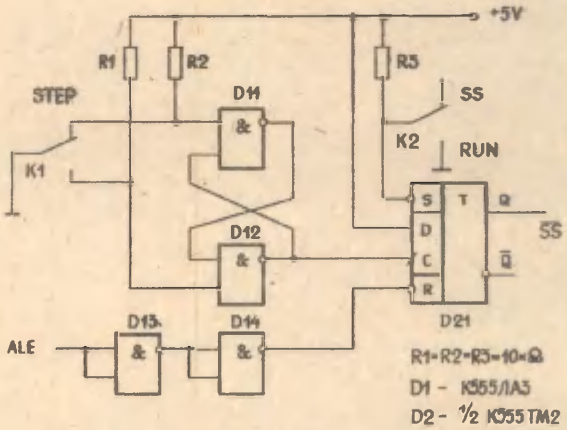
Sammtalitluseks e. programmi käskhaaval täitmiseks nimetatakse programmi silumist lihtsustavat võtet, mille korral protsessoril lastakse iga käsu täitmisel automaatselt peatuda ning järgmise käsu täitmisele asuda alles pärast lubava välise signaali saamist.

Meie arvuti seadmiseks sammtalitluse režiimi tuleb kasutada sisendit $\overline{\text{SS}}$ (inglise keeles - single step). Arvuti kontrollib seda sisendit iga käsu (kahetsüklilistel käskudel käsu teise tsükli) kolmandas taktis T3 niipea, kui sig-

naal ALE saab seisu 1. Kui ($\overline{SS}$)=0 & (ALE)=1 ning on käsil käsu viimane masinatsükkel, siis siirdub arvuti SS-režiimi - väljastab värati PO ja värati P2 madalama poole viikudele järgmisena täitmisele tuleva käsu aadressi, jätab väljundi ALE seisu 1 ning jääb ootama töö jätkamist lubava signaali 1 ilmumist sisendisse $\overline{SS}$. Selle ilmumisel jätkab arvuti tööd poolelijäänud kohast töötaktis T3. Töö jätkumise tunnuseks on üleminek 1→0 väljundis ALE. Kui soovime tööd ka järgmise käsu täitmisel peatada, tuleb enne ALE järgmist minekut seisu 1 anda uus peatussignaal ($\overline{SS}$)=0.

Tavaliselt kasutatakse SS-režiimi koos häälestusrežiimiga, mil arvuti töötab välise püsimäluga. Sellisel juhul väljastab ta niikuinii järgmisena täitmisele tuleva käsupesa aadressi värati PO ja värati P2 madalama poole viikudele. Harilikus tööreżiimis, kus väratid PO ja P2 võivad olla kasutuses üldotstarbeliste väljundväratitena, rikub SS-režiimi minekuga kaasnev aadressi väljastamine värati kaudu väljastatavad andmed. Neid on välisseadme jaoks võimalik päästa välise fikseerimise teel signaali ALE esimese frondi ajal.

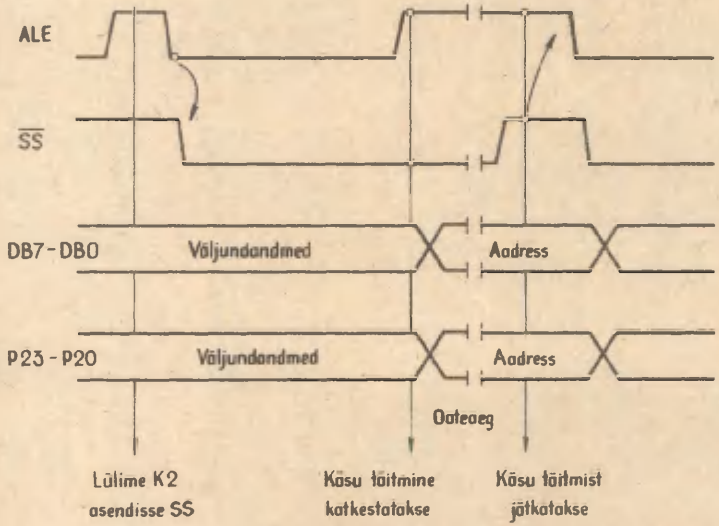
Sammtalitlust korrektselt realiseerida võimaldav elektroonikaekeem on toodud joonisel 25. Selle abil saab täpselt



Joonis 25. Skeem SS-režiimi kasutamiseks.

ajastada arvuti töö jätkamist lubava signaali ($\overline{SS}$)=1 kestust. Kui signaal (SS)=1 kestab liiga kaua, jõuab arvuti selle vältel rohkem kui ühe käsu täita. Joonisel 26 on toodud SS-režiimi ajadiagramm eeldusel, et väratid P0 ja P2 on väljundväratid ning kasutatakse joonisel 25 esitatud skeemi.

Kui lüliti K2 on asendis "RUN", siis on trigeri S-sisendis signaal 0, mis hoiab ta väljundi Q ning seega ka arvuti sisendi $\overline{SS}$ seisus 1. Arvuti on harilikus programmi täitmise režiimis. Kui viime lüliti K2 asendisse "SS", siis muutub aktiivseks trigeri nullimissisend R. Niipea kui ALE läheb seisu 0, viiakse ka triger seisu Q=0 ning arvutilt nõutakse peatumist.



Joonis 26. SS-režiimi ajadiagramm.

Peatust nõudva nullsignaali avastab arvuti järgmise masinatsükli taktis T3 niipea, kui ALE läheb seisu 1. Seepeale käsu täitmine katkestatakse. Arvuti ootab sisendisse $\overline{SS}$ töö jätkamist lubavat signaali 1. Selle signaali andmiseks tuleb vajutada sõrmisele "STEP", mis kutsub trigeri takteerimissisendis C esile ülemineku 0→1. Kuna sisend C on otse-

ne dünaamiline eisend, siis põhjustab üleminek 0→1 siin trigeri väljundi Q ning arvuti sisendi $\overline{SS}$ mineku seisu 1. Arvuti jätkab tööd. Niipea kui ALE läheb seisu 0, läheb ka triger seisu Q=0 ning arvutilt nõutakse jälle peatust. Arvuti avastab selle järgmise käsu täitmisel ning katkestab jälle oma töö. Kui sõrmis vabastada ning uuesti vajutada, siis täidetakse ka järgmine käsk jne.

9. Arvuti sisemise püsimälu programmeerimine.

Kasutaja poolt programmeeritavat EPROM-tüüpi sisemist püsimälu omab vaid arvuti KM1816BE48.. Programmeerimisele asumisel peab mälu olema kustutatud - kõigis mälupesades peab olema arv 00_H.

Püsimälu programmeerimine toimub pesahaaval. Iga mälupesa programmeerimine sisaldab järgmisi etappe: arvuti viiakse programmeerimisrežiimi, sisestatakse programmeeritava mälupesa aadress, aadress fikseeritakse, sisestatakse mälupessa salvestatav arv, programmeeriva impulsiga salvestatakse see arv valitud mälupessa, arvuti viiakse püsimällu salvestatu lugemise režiimi ja kontrollitakse salvestatu õigsust. Enne järgmise mälupesa programmeerimisele asumist tuleb eelmise programmeerimistsükkel täielikult lõpetada.

Püsimälu programmeerimisel kasutatavate viikude otstarve on järgmine:

XTAL1 - selle viigu kaudu sisestatakse takteerimissignaal, mille sagedus peab kuuluma vahemikku 1 - 6 MHz. On lubatud ka mainitud vahemikku kuuluva resonantssagedusega kvartskristalli ühendamine viikudega XTAL1 ja ja XTAL2 eespool kirjeldatud viisil.

$\overline{RESET}$ - selle viigu kaudu sisestatakse arvutisse programmeeritava mälupesa aadressi fikseeriv signaal.

TO - selle viigu kaudu seatakse arvuti programmeerimise või salvestatu kontrollimise režiimi.

EA - selle viigu kaudu seatakse arvuti programmeerimise ning salvestatu kontrollimise režiimi.

DB7 - DB0 - need viigud on kaeutusee kahesuunalise aadressi- ja andmesiini viikudena. Siitkaudu sisestatakse arvutisse programmeeritava mälupesa aadressi 8 madalamat bitti ja mällu salvestatav arv. Nende viikude kaudu väljastab arvuti mällu salvestatu õigsuse kontrollimisel mälupessa salvestatud arvu.

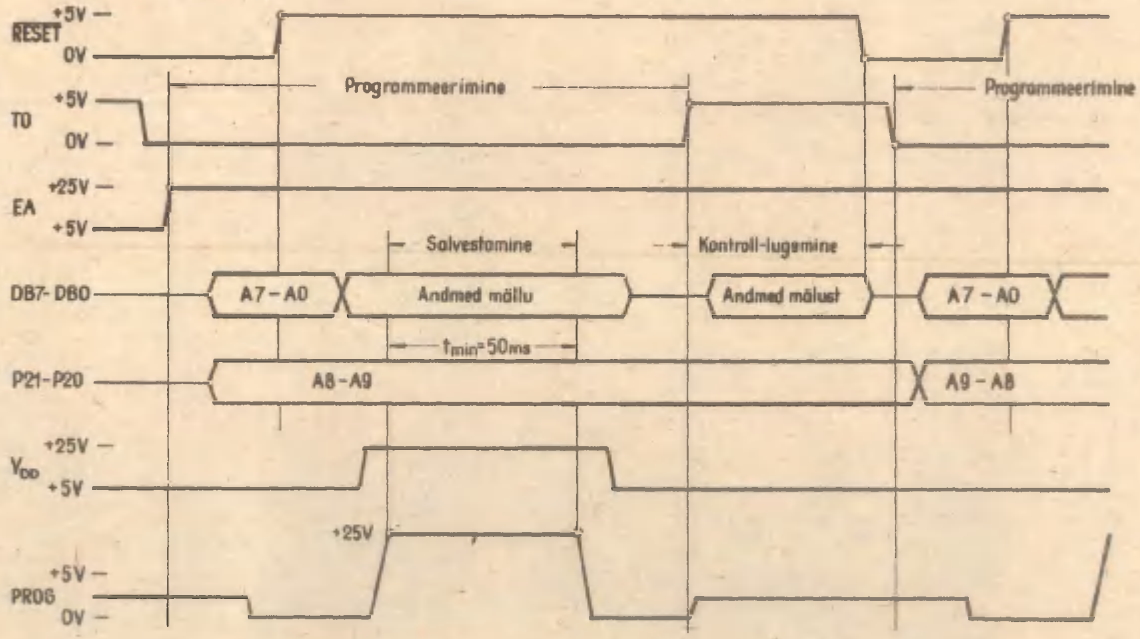
P21 - P20 - neid viike kasutatakse mälupesa aadressi kahe kõrgema biti sisestamiseks arvutisse.

V_{DD} - selle viigu kaudu antakse arvutile programmeerimispinge (toide).

PROG - selle viigu kaudu sisestatakse arvutisse programmeerimisimpulss.

Püsimälu programmeerimisel kasutatakse mikroarvutit sisaldavat abiseadet - programmaatorit, mis automaatselt korraldab kogu programmeerimiseks ning salvestatu õigsuse kontrollimiseks vajaliku töö. Programmeerimisega seotud signaalide ajadiagramm on toodud joonisel 27. Algallikate [3 ja 8] põhjal on programmeerimiseeskiri järgmine:

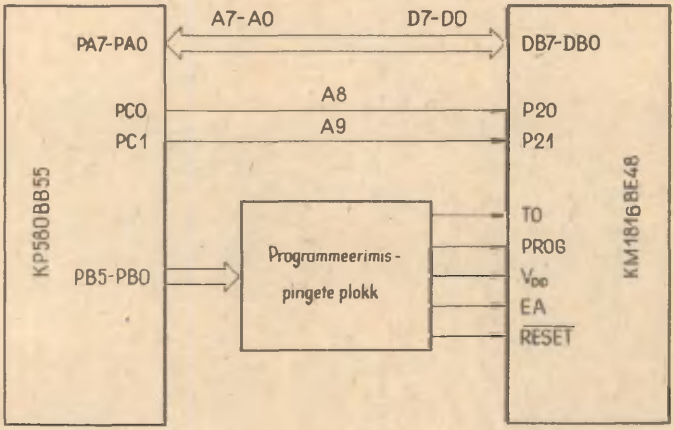
1. Viikudele V_{DD}, T0 ja EA tuleb anda pinge +5 V ning viigule $\overline{\text{RESET}}$ pinge 0 V. Nendes tingimustes tuleb arvutikiip pista programmaatoril olevasse pessa ning kontrollida signaali ALE olemasolu. Valepidi pessa pistetud arvutikiibi programmeerimiskatse rikub kiibi.
2. Viigule T0 tuleb anda pinge 0 V, mis valib programmeerimisrežiimi.
3. Viigule EA tuleb anda pinge +24,5 V, mis aktiviseerib programmeerimisrežiimi.
4. Värati P0 viikudele DB7 - DB0 tuleb anda mälupesa aadressi kaheksa madalamat bitti ning värati P2 viikudele P21 - P20 aadressi kaks kõrgemat bitti.
5. Viigule $\overline{\text{RESET}}$ tuleb rakendada pinge +5 V, mis fikseerib arvutis esitatud aadressi.
6. Värati P0 viikude kaudu tuleb nüüd arvutisse sisestada mälupessa salvestatav arv.
7. Viigule V_{DD} tuleb rakendada programmeerimispinge + 25 V.
8. Viigule PROG tuleb rakendada pinge 0 V ja seejärel vähemalt 50 ms kestev programmeerimispinge +25 V.



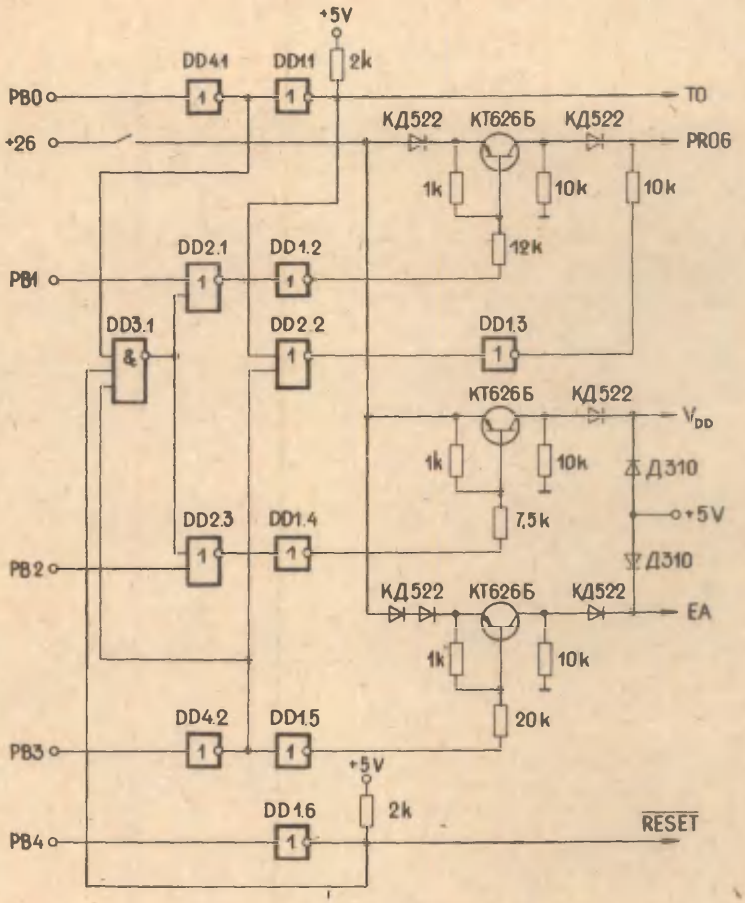
Joonis 27. Sisemise püsimälu programmeerimine ja kontroll-lugemine.

9. Viigul V_{DD} tuleb pinget vähendada +5 voldini. Sellega on mälupessa salvestamine lõpetatud ning algab salvestatu õigsuse kontrollimine.
10. Viigule T0 tuleb rakendada pinge +5 V, mis seab arvuti püsimällu salvestatu kontrollimise reĕiimi. Arvuti väljastab viikudele DB7 - DB0 mälupesa sisu. Programmaator peab lugema arvuti poolt väljastatud arvu ning võrdlemise teel kontrollima salvestatu õigsust.
11. Järgmise mälupesa programmeerimiseks tuleb viigule T0 ja seejärel viigule $\overline{\text{RESET}}$ rakendada pinge 0 V ning korrata eespool kirjeldatud protseduuri alates punktist nr. 4 kuni kogu vajalik info saab mällu salvestatud.
12. Enne arvutikiibi programmaatoril olevast pesast väljavõtmist tuleb arvuti viigud viia punktis nr. 1 näidatud seisu.

Programmeerimissignaalide täpsed parameetrid on esitatud õppevahendi lisas. Joonistel 28 ja 29 on toodud programmaatori üks võimalik variant. Seadet juhitakse mikroprotsessoril KP580ИК80 baseeruva mikrokontrolleri rööpvärati KP580BB55 kaudu.



Joonis 28. Arvuti sisemise püsimälu programmeerimisseade.



DD1 ja DD4 - К555ЛН1
DD2 - К555ЛЕ1
DD3 - К555ЛА4

Joonis 29. Programmeerimispingete plokk.

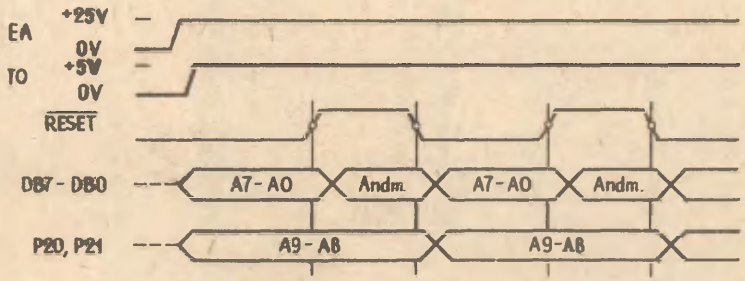
10. Sisemise püsimälu kontroll-lugemine.

Sisemise püsimälu kontroll-lugemise vajadus esineb alati enne selle mälu programmeerimist, mil on tarvis veenduda selles, et mälu on kustutatud. Sellise lugemise vajadus võib tekkida ka arvutil baseeruva seadme remontimisel, kui kaheldakse, et häired seadme töös on tingitud mälu osalisest kustumisest.

Arvuti viimiseks kontroll-lugemise reŽiimi tuleb:

1. Viigule $\overline{\text{RESET}}$ rakendada pinge 0 V;
2. Viigule EA rakendada pinge +24 V;
3. Viigule T0 rakendada pinge +5 V.

Seejärel tuleb esimese mälupesa aadress sisestada arvutisse selle värati P0 viikude DB7 - DB0 ning värati P2 viikude P21 - P20 kaudu (arvutil KM1816BE49 viikude P22 - P20 kaudu). Aadressi fikseerimiseks arvutis tuleb nüüd viigule $\overline{\text{RESET}}$ rakendada pinge +5 V. Aadressi fikseerimise järel seab arvuti viigud DB7 - DB0 väljunditeks, kuhu väljastab adresseeritud mälupesa sisu. Andmed püsivad väljundis kuni viigul $\overline{\text{RESET}}$ on pinge +5 V. Pinge 0 V andmisega viigule $\overline{\text{RESET}}$ algab järgmise mälupesa kontroll-lugemine jne. Öeldut illustreerib joonisel 30 toodud ajadiagramm.



Joonis 30. Sisemise püsimälu kontroll-lugemine.

11. Püsimälu kustutamine.

Püsimälu kustutamiseks tuleb kristallile läbi kiibil oleva akna juhtida ultraviolettkiirgus, mille lainepikkus $\lambda \approx 254$ nm. Kustutamiseks vajalik integraalne kiirgusdoos on orienteeruvalt 15 J/cm^2. Vajalikud tingimused tagab bakterioloogiline lamp DP5-8-0.4.2, kui kiip paikneb lambist 10 cm kaugusel ning kiirituse kestus on 60$\pm$5 minutit.

III ARVUTI VÄLINE LAIENDAMINE.

Kui arvuti sisemise mälu maht või sisend-väljundkanalite arv osutuvad püstitatud ülesande täitmiseks liialt napiks, siis võib arvutit väliselt laiendada. Arvuti käsustik ja ehitus võimaldavad otseselt laiendada:

- püsimälu kuni nelja kilobaidini;
- muutmälu kuni 320 baidini;
- sisend-väljundkanalite arvu täiendavate neljabitiste väratite P4 - P7 kasutuselevõtuga.

Kasutades spetsiaalseid võtteid saab nii mälu mahtu kui ka infokanalite arvu laiendada kuitahes suureks. Kuna arvuti värat P0 on kasutatav ka kahesuunalise siinina, mis on ühildatav seeria KP580 elementidega, siis on arvuti väliseks funktsionaalseks laiendamiseks kasutatavad ka selle seeria elemendid.

Kuigi arvuti laiendamisvõimalused on väga laiad, on ta projekteeritud eeskätt siiski lihtsamate automatiseerimisülesannete lahendamiseks. Enam kui kahe-kolme suure integraallülituse lisamist arvuti konfiguratsiooni ei peeta otstarbekaks. Keeruliste ülesannete lahendamisel osutub tunduvalt otstarbekamaks kasutada mingit universaalprotsessoril baseeruvat süsteemi, mis tagab komponentide sama arvu ja maksumuse juures süsteemile hoopis universaalsemad omadused.

12. Püsimälu väline laiendamine.

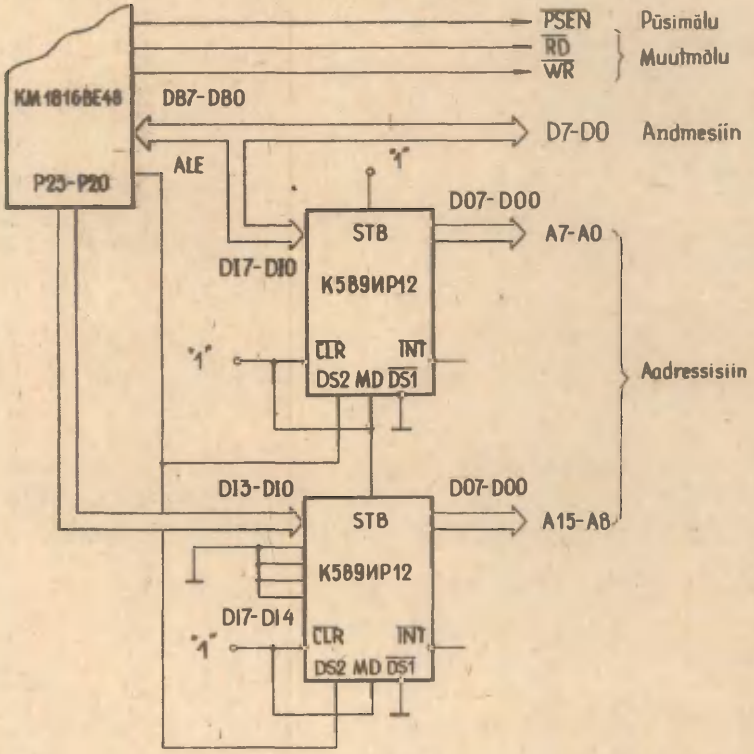
Püsimälu välisele laiendamisele asudes meenutame järgmist:

1. Püsimälu adresseerimist käsuloenduri PC kaudu kirjeldasime punktis 2.2. Sisemise püsimälu pesade poole pöördumisel väljastab arvuti vaid signaali ALE. Kui käsuloenduri sisu saab suuremaks sisemise püsimälu viimase pesa aadressist, siis läheb arvuti automaatselt välise püsimälu lugemise reŽiimi.

2. Välise püsimälu lugemise reŽiimi kirjeldasime punktis 5. Värati P0 viigud DB7 - DB0 on siin kasutuses kahesuu-

nalise multipleksse aadressi- ja andmesiini viikudena, värati P2 viikude P23 - P20 kaudu väljastab arvuti aadressi neli kõrgemat bitti ning lisaks signaalile ALE väljastab arvuti viigu $\overline{PSEN}$ kaudu veel püsimälu lugemissignaali. Väljastatav aadress on arvuti viikudel DB7 - DB0 stabiilne signaali ALE tagumise frondi ajal ja mälusõna loetakse arvutisse signaali $\overline{PSEN}$ tagumise frondi ajal.

Kuna meil multipleksse aadressi- ja andmesiiniga mälukiipe ei toodeta, tuleb nii püsi- kui ka muutmälu välisel laiendamisel aparatuuraelt moodustada väline aadressisiin. Seda võib teha nii, nagu on näidatud joonisel 31.



Joonis 31. Skeem välise aadressisiini moodustamiseks.

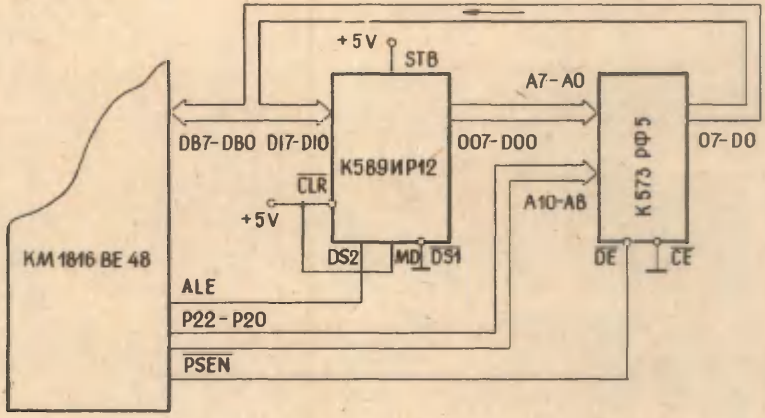
Aadressi fiksaatorina on kasutatud multifunktsionaalseid värateid К589ИР12, mis sisaldavad kaheksabitist fiksaatorit ja väljundpuhvrit. Kui viigul MD on signaal 1, siis väljundpuhvrid on avatud ja sisenditesse DI7 - DI0 antud arv fikseeritakse signaali $\overline{DS1}$ & DS2 nullimineva frondiga. Kuna meie juhul on $\overline{DS1}$=0 ja sisenditesse DS2 antakse signaal ALE, siis fikseeritakse arvuti poolt väljastatav aadress väratites just õigel momendil - signaali ALE tagumise frondi ajal. Väratite väljundid DO7 - DO0 moodustavad 16-bitise aadressisiini, mille neli kõrgemat bitti on muutumatult seisus 0, sest joonisel on alumise värati sisendites DI7 - DI4 signaal 0. Nõnda moodustatud 12-bitisesse aadressivahemikku võib paigutada 4 kilobaiti püsimälu, mille lugemiseks tuleb kasutada signaali $\overline{PSEN}$.

Kuna arvuti KM1816BE48 korral sisemise püsimälu esimest kilobaiti tavalises tööreŽiimis välismälu poole pöördumisel ei adresseerita, saame sinna salvestada arvuti diagnostikaprogrammi. Kui arvuti viia viigu EA kaudu häälestusreŽiimi (vt. punkti 7), siis kasutab ta ainult välist püsimälu ning hakkab programmi täitma alates välismälu algusaadressist 000_H.

Soovi korral võime välist püsimälu veelgi laiendada. Täiendavate mälupankade adresseerimiseks võime kasutada värati P2 ülejäänud viike P27 - P24, ühendades need värati К589ИР12 seni kasutamata sisenditega DI7 - DI4 või otse mälukiipide selekteerimise deŠifraatoriga. Nõnda saame välist püsimälu laiendada juba 64 kilobaidini. Sellise mooduse kasutamisel tuleb lisaks käsustikus olevatele mälupankade valiku käskudele SEL MB0 ja SEL MB1 kasutada käsku OUTL P,A täiendavalt loodud väliste pankade valikuks värati P2 nelja kõrgema biti kaudu.

Joonisel 32 on toodud eelmisest lihtsam püsimälu laiendusskeem, mis lisab kilobaidi välist töömälu ja kilobaidi püsimälu diagnostikaprogrammi jaoks. Fikseeritakse vaid aadressi kaheksa madalamat bitti. Aadressi kolm kõrgemat bitti antakse värati P2 viikude P22 - P20 kaudu otse mälukiibi К573РФ5 aadressisisenditesse. Selle mälu esimest kilobaiti arvuti tavalises tööreŽiimis ei adresseeri. Kui me vaa-

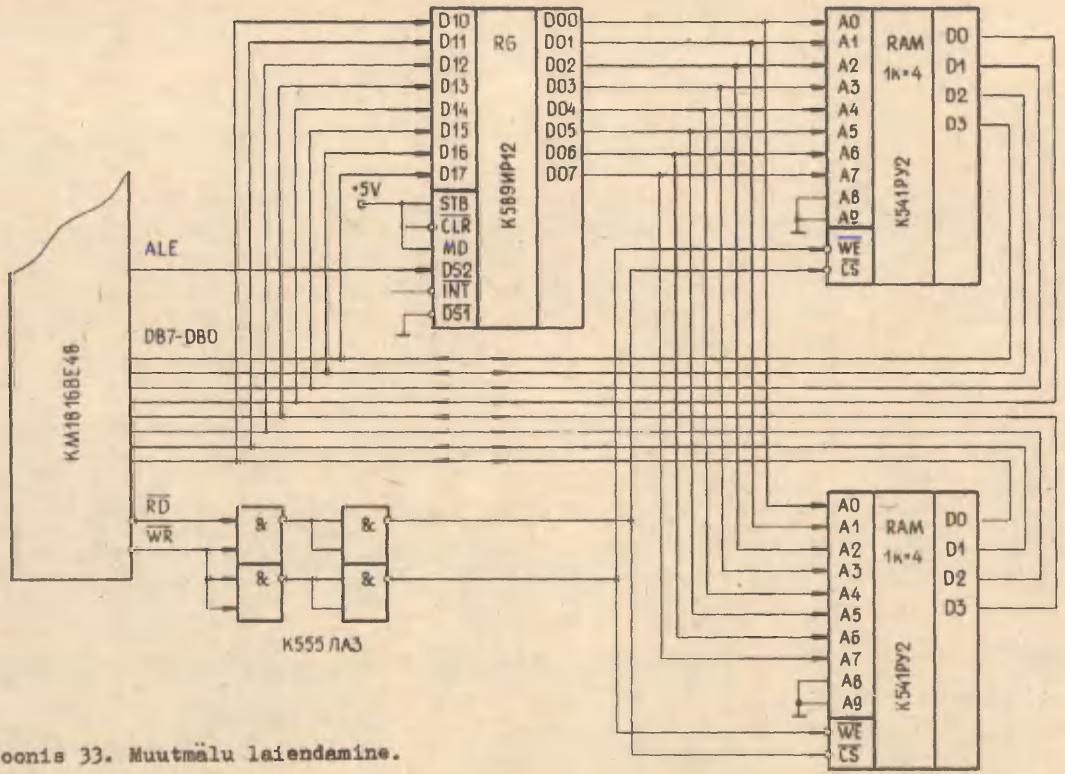
deldavat skeemi veidi muudaksime nii, et mälukiibi selekteerimisviigule $\overline{CE}$ annaksime läbi invertori signaali viigult P23, siis paikneks 2 kilobaiti välismälu aadresside vahemikus 800_H- FFF_H ning moodustaks välismälu panga nr.2.



Joonis 32. Püsimälu väline laiendamine.

13. Muutmälu väline laiendamine.

Joonisel 31 toodud aadressisiini saame kasutada ka muutmälu väliseks laiendamiseks. Kuna väline muutmälu on käskudega MOVX A,@R ja MOVX @R,A otseselt adresseeritav vaid 256 baidi ulatuses, siis tuleb kasutada aadressi bitte A7 - A0 ja sünkrosignaale $\overline{RH}$ ning $\overline{WH}$. Värati P2 viigud välise muutmäluga seotud infovahetuses ei osale. Soovi korral võime neid viike kasutada välise muutmälu täiendavate lehekülgede valikuks. Nelja aadreesijärku otseselt kasutades saaksime moodustada neli välise muutmälu lehekülge, lehekülje valiku dešifraatorit kasutades saaksime moodustada 16 välise muutmälu lehekülge. Seejuures tuleb enne vaetava mälulehekülje poole pöördumist värati P2 madalamasse poolde kä-



Joonis 33. Muutmälu laiendamine.

suga OUTL P,A väljastada vajaliku lehekülje number.

Joonisel 33 on toodud skeem väliee muutmälu mooduatamiseks mälukiipide K541PÜ2 abil. Kuna meil sobiva mahuga (256 x 8) muutmälu kiipe ei toodeta, on kasutatud kahte mälukiipi mahuga 1k x 4. Välise muutmälu mooduatamiael selliste kiipide abil, mille maht on 256 x 1 bitti, muutuks kiipide arv skeemis ebaaoovitavalt suureka.

Aadresai fikseerimine toimub joonisel 33 toodud skeemie nii, nagu joonisel 31 toodud ja eespool kirjeldatud skeemia. Mälukiipide valiku loogikatabel on järgmine:

Režiim	$\overline{RD}$	$\overline{WR}$	$\overline{CS}$	$\overline{WE}$
Mälu ei kasutata	1	1	1	1
Salvestamine	1	0	0	0
Mälu lugemine	0	1	0	1

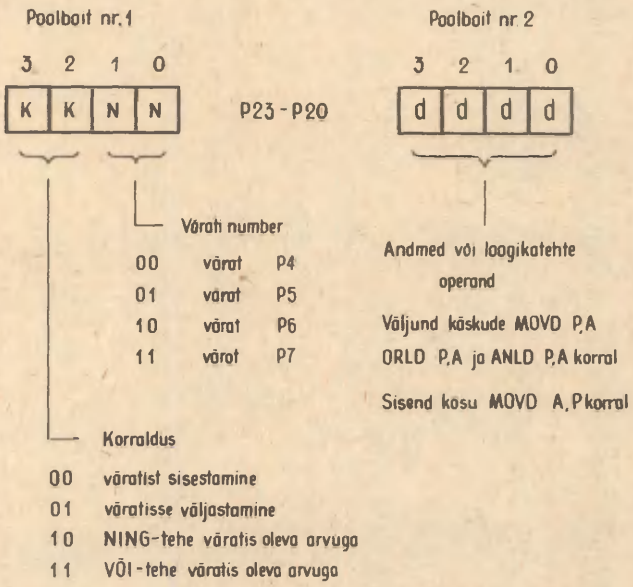
Režiimi $\overline{RD}$=$\overline{WR}$=0 ei esine, sest samaaegaelt ei saa salvestada ja lugeda.

Neljalehekülјelise välise muutmälu mooduatamiseks tuleb joonisel 33 toodud mälukiipide aadreasisiaendid A8 ja A9 ühendada värati P1 või P2 vabade viikudega ning kasutada neid viike mälulehekülgede valikuka. Neile viikudele käsuga OUTL P,A väljastatav kahendarv määrab mälulehekülje numbri. Kui näiteka väljaatame arvu 11_2, siis valime sellega mälulehekülje nr.3, mis mälukiipide aadreaaiväljaa paikneb aadreaside vahemikus 300_H- $3FF_H$.

14. Täiendavate aiaend-väljundkanalite moodustamine.

Vaatleme eamalt arvuti ehituae ning käaustiku poolt pakutavat võimaluat täiendavate neljabitiste väliate väratite P4 - P7 moodustamiseks. Firma Intel toodab selleks spetsiaalselt projekteeritud väratikiipi 8243, mia on arvutiga seotav värati P2 viikude P23 - P20 kaudu. Väratikiibi ja arvuti tööd sünkroniseerib signaal $\overline{PROG}$. Andmevahetust nende täiendavate väratitega korraldavad kahetsüklilised käsud MOVD P,A, MOVD A,P, ANLD P,A ja ORLD P,A. Käsu esimeses tsük-

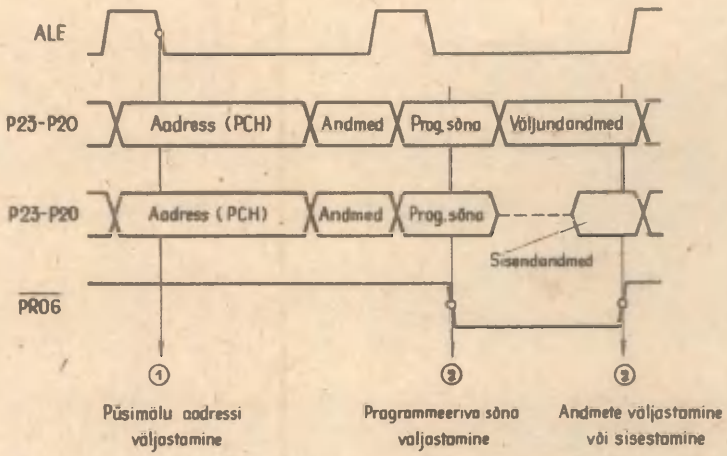
lis väljastab arvuti viikude P23 - P20 kaudu väratikiipi programmeeriva sõna, milles on kodeeritud värati number ning täidetava käsu tüüp. Andmevahetus välisväratitega toimub käsu teises tsüklis. Käsu MOVD A,P korral loeb arvuti väratis oleva arvu, käskude MOVD P,A ning ANLD P,A ja ORLD P,A korral väljastab väratisse salvestatava arvu või loogikatehte operandi. Mõlema sõna - programmeeriva sõna ja infosõna vorming on toodud joonisel 34.



Joonis 34. Laiendusvärateid programmeeriv sõna ja infosõna.

Välisväratitega toimuva andmevahetuse ajadiagramm on toodud joonisel 35. See on koostatud eeldusel, et arvuti loeb käsukoodi välisest püsimälust. Välise püsimälu lugemisel kasutab arvuti viike P23 - P20 aadressi nelja kõrgema biti väljastamiseks. Pärast käsukoodi lugemist käsu esimese tsükli alguses ennistab arvuti viikude P23 - P20 kaudu väljastatava arvu. Kolmandas taktis (vt. ka joonist 19 ning ta-

belit 2) väljastab arvuti nende viikude kaudu väratikiipi programmeeriva sõna. See on viikudel stabiilne neljanda



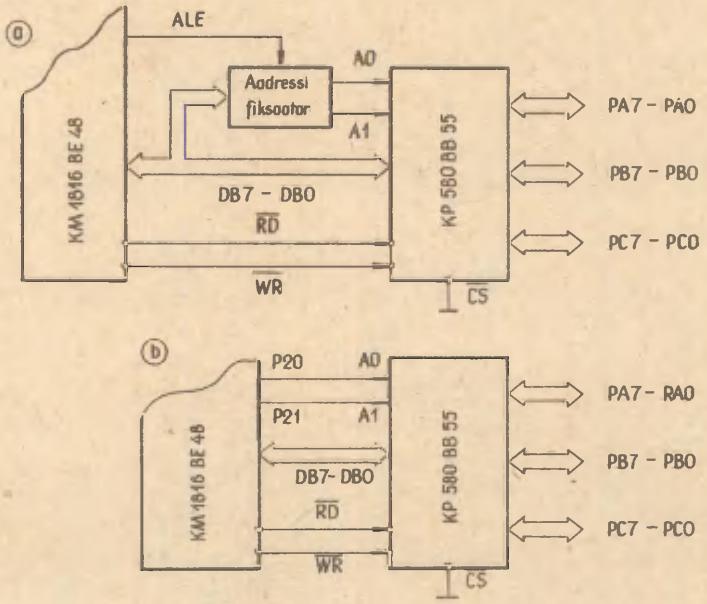
Joonis 35. Andmevahetus väratitega P4 - P7.

takti T4 keskel. Signaali $\overline{\text{PROG}}$ esimest fronti kasutatakse programmeeriva sõna väratikiibis fikseerimiseks. Laiendusväratitele mõeldud infosõna (väljastatava arvu või loogikatehte operandi) seab arvuti viikudele P23 - P20 taktis T5. Väratikiip fikseerib need käsu teise tsükli takti T2 lõpus. Fikseerimiseks kasutatakse signaali $\overline{\text{PROG}}$ tagumist fronti. Kui toimub väratist lugemine, siis kannab väratikiip väratis oleva arvu arvuti viikudele P23 - P20 veidi enne signaali $\overline{\text{PROG}}$ tagumist fronti, mil arvuti loeb neil viikudel oleva sõna akumulaatorisse.

Laiendusväratite P4 - P7 kasutamisel tuleb silmas pidada järgmist. Väratikiip Intel 8243 on projekteeritud nii, et loogikatehete sooritamine ning selle tulemi laiendusväratisse kandmine toimub väratikiibis. Laiendusväratite P4 - P7 kasutamise korral arvuti värati P2 viigud P23 - P20 enam üldotstarbelise sisend-väljundvärati viikudena kasutatavad ei ole. Käskude MOVD P,A, ANLD P,A ja ORLD P,A täitmise järel

väratisse varem väljastatud arvu ei ennistata. Värat jääb väljundväratiks, mille viikudel on laienduskiipi väljastatud infosõna. Käsu MOVD A,P täitmise järel jäävad viigud P23 - P20 sisenditeks, mis läbi kõrgeoomilise resistori on seatud seisu 1 (vt. joonise 12 juures toodud selgitusi).

Kuna meil Intel 8243 tüüpi väratikiipi ei toodeta*, on otstarbekam loobuda väratite P4 - P7 kasutamisest ning sisend-väljundkanalite arvu suurendada seeria KP580 väratikiipide abil. Joonisel 36 on toodud kaks skeemi rööpvärati KP580BB55 sidumiseks arvutiga. Mõlemal juhul paiknevad lai-



Joonis 36. Rööpvärati KP580BB55 sidumine arvutiga.

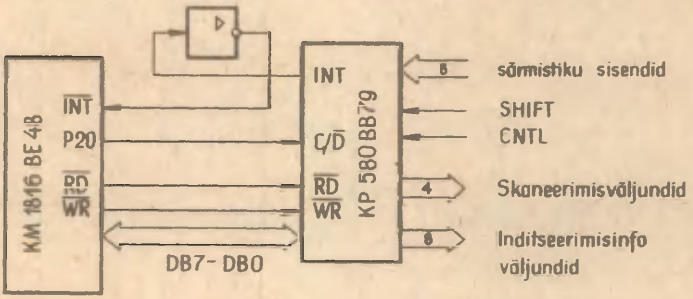
endusväratid arvuti välise muutmälu aadressiväljas ning nende poole tuleb pöörduda käskude MOVX @R,A või MOVX A,@R abil. Variandi a korral on värati aadressi fikseerimiseks kasutatud välist aadressifiksaatorit. Kui me laiendusmälu-

* Kiipi 8243 toodetakse meil nüüd kiibina KP580ИК43.
(Toimetaja).

sid ei kasuta, siis võime värati P2 viikude P20 ja P21 loovutamise hinnaga välisest aadressifiksaatoriet vabaneda. Seda võimalust illustreerib joonisel 36 variant b. Siin on enne välisvärati poole pöördumist tarvis käsuga OUTL P2,A viikudele P20 ja P21 kanda välisvärati aadress. Mõlemal juhul määravad välisvärati aadressibitid A0 ja A1 värati poole pöördumise otstarbe järgmiselt:

A1	A0	Pöördumise otstarve
0	0	Infovahetus väratiga A
0	1	Infovahetus väratiga B
1	0	Infovahetus väratiga C
1	1	Väratikiipi programmeeriva sõna väljastamine

Sarnaselt on arvutiga seotavad ka teised seeria KP580 elemendid. Joonisel 37 on toodud näide klaviatuuri- ja indikatsioonikontrolleri KP580BB79 ühendamise kohta. Siin on andmevahetuse otstarvet defineeriva signaali väljastamiseks kasutatud arvuti viiku P20. Kontrolleri katkestusväljund INT tuleb arvuti katkestussisendiga $\overline{INT}$ ühendada läbi invertori.

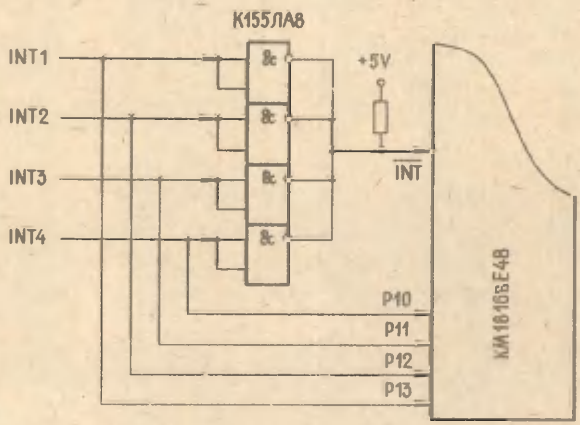


Joonis 37. Klaviatuuri- ja indikatsioonikontrolleri KP580BB79 ühendamine arvutiga.

15. Katkestussüsteemi väline laiendamine.

Arvuti katkestussüsteemi töötamispõhimõtet selgitasime punktis 2.8, kus kirjeldasime ka viigu T1 kasutamist täiendava väliskatkestussisendina. Kui ka seda jääb väheseks, siis on võimalik arvuti katkestussisendite arvu välise laiendusskeemi abil suurendada.

Joonisel 38 on toodud skeem, mis võimaldab luua neli võrdse prioriteediga katkestussisendit INT1 - INT4. Signaali 1 andmisel ühte neist sisendeist (või korraga mitmesse) ilmub katkestust nõudev signaal 0 ka arvuti sisendisse $\overline{\text{INT}}$. Värati P1 viikudele P13 - P10 ilmub info katkestust nõudva seadme numbri kohta. Katkestusnõudmise rahuldamisel siirdub programmi täitmine pesas 003_H algavasse katkestavprogrammi, mis peab esmalt lugema viikudel P13 - P10 oleva info katkestust nõudva seadme numbri kohta, vajaduse korral lahendama prioriteediküsimuse ning seejärel suunama programmi täitmise katkestust nõudnud seadet teenindavasse katkestavprogrammi lõiku.



Joonis 38. Neli võrdse prioriteediga katkestussisendit.

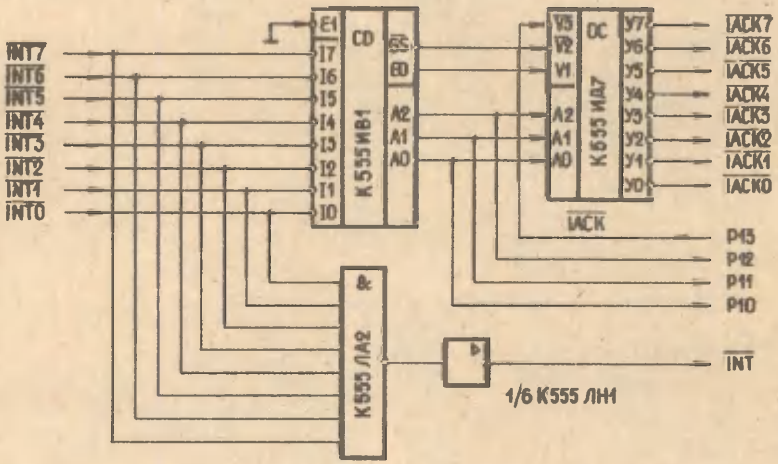
Vaadeldud lihtsa skeemi oluliseks puuduseks on nõudmise rahuldamisest signaliseeriva tagasisideahela puudumine,

mistõttu nõudmise esitanud seadmel võib tekkida raskusi nõudmise õigeaegse kõrvaldamisega. Joonisel 39 toodud laiendusskeemil sellist puudust ei ole. Siin on aparatuurselt lahendatud ka prioriteediprobleem. Kaheksast katkestussisendist $\overline{INT0}$ - $\overline{INT7}$ on kõige kõrgem prioriteet, sisendil $\overline{INT7}$ ning kõige madalam - sisendil $\overline{INT0}$.

Vaadeldav skeem funktsioneerib järgmiselt. Põhiprogrammi täitmise ajal peavad värati P1 viigud P13 - P10 olema seatud sisendeiks - olema seisus 1. Signaal 1 viigul P13 suleb deŝifraatori K155ИД7. Katkestust nõudva nullsignaali ilmumisel ühte või korraga mitmesse sisendisse $\overline{INT0}$ - $\overline{INT7}$, ilmub aktiivne nullsignaal ka arvuti katkestussisendisse $\overline{INT}$. Šifraatori K555ИВ1 väljundeisse ilmub kahendarv, mis on võrdne aktiivse katkestussisendi numbriga (kõrgeima prioriteediga aktiivse sisendi numbriga). Katkestavprogrammi siirdumisel loeb arvuti sisendeist P12 - P10 katkestust nõudva seadme numbri, seab viigu P13 seisu 0 ning siirdub katkestust nõudvat seadet teenindavasse katkestavprogrammi lõiku. Nullsignaal viigul P13 valib deŝifraatori, mille väljundeist Y0 - Y7 muutub aktiivseks see väljund, mis vastab katkestust nõudnud ning parasjagu teenindatavale seadmele. Vastussignaali $\overline{IACK}$ saab kasutada katkestusnõudmise kõrvaldamiseks.

Skeemi töötamispõhimõtte detailse analüüsi lihtsustamiseks on joonisel 39 toodud nii ŝifraatori kui ka deŝifraatori loogilist otstarvet defineerivad tabelid. Juhime veelkord tähelepanu sellele, et K555ИВ1 on seade, mille sisend I7 on kõige kõrgema ja sisend I0 kõige madalama prioriteediga. Madalama prioriteediga sisendi kaudu nõutud katkestusest informeerib skeem arvutit alles pärast seda, kui kõik kõrgema prioriteediga nõudmised on rahuldatud ning vastavad nõudmissignaalid sisendeist $\overline{INT}$ kõrvaldatud.

Selle paragrahvi ja ka peatüki lõpetuseks üks küsimus neile lugejatele, kes on tuttavad seeria KP580 elementidega: "Miks ei ole siin katkestussüsteemi laiendamiseks soovitatud katkestuskontrollerit KP580ВН59 ?"



K555 ИВ1

Sisendid									Väljundid				
E1	I0	I1	I2	I3	I4	I5	I6	I7	A1	A2	A0	ES	EO
1	X	X	X	X	X	X	X	X	1	1	1	1	1
0	1	1	1	1	1	1	1	1	1	1	1	1	0
0	X	X	X	X	X	X	X	0	0	0	0	0	1
0	X	X	X	X	X	X	0	1	0	0	1	0	1
0	X	X	X	X	X	0	1	1	0	1	0	0	1
0	X	X	X	X	0	1	1	1	0	1	1	0	1
0	X	X	X	0	1	1	1	1	1	0	0	0	1
0	X	X	0	1	1	1	1	1	1	0	1	0	1
0	X	0	1	1	1	1	1	1	1	1	0	0	1
0	0	1	1	1	1	1	1	1	1	1	1	0	1

X - suvaline seis

K555 ИД7

Sisendid					Väljundid							
V1	V*	A2	A0	A1	У0	У1	У2	У3	У4	У5	У6	У7
X	1	X	X	X	1	1	1	1	1	1	1	1
0	X	X	X	X	1	1	1	1	1	1	1	1
1	0	0	0	0	0	1	1	1	1	1	1	1
1	0	0	0	1	1	0	1	1	1	1	1	1
1	0	0	1	0	1	1	0	1	1	1	1	1
1	0	0	1	1	1	1	1	0	1	1	1	1
1	0	1	0	0	1	1	1	1	0	1	1	1
1	0	1	0	1	1	1	1	1	1	0	1	1
1	0	1	1	0	1	1	1	1	1	1	0	1
1	0	1	1	1	1	1	1	1	1	1	1	0

$V^* = \overline{V3} \vee \overline{V2}$

Joonis 39. Katkestusesüsteemi laiendamine.

IV LISA

Siin on lisaks eespool esitatud kirjeldusele toodud rida arvuti ekspluatatsioonitingimusi ning parameetreid täpsustavaid andmeid, mis muudatusteta on võetud arvuti KM1816BE tehnilisest dokumentatsioonist [1, 2]. Lisa viimases punktis on õppevahendi suhtelise iseseisvuse tagamiseks esitatud veel arvuti käskude koondtabelid. Käsustiku põhjalik kirjeldus on toodud õppevahendi teises vihikus [9].

1. Ekspluatatsioonieeskirjad.

Arvuti KM1816BE48 staatilised ja dünaamilised parameetrid ning signaalide ajadiagrammid erinevate töörežiimide korral on esitatud lisas 3. Kaitset staatilise elektrivälja kahjustava toime eest tuleb korraldada vastavalt standardile OCT B11.073,062 - 76. Staatilise potentsiaali lubatud piirväärtus on miinus 30 volti.

Trükkplaadile soovitatakse arvutikiip monteerida mikroskeemide pesa abil, et teda oleks ümberprogrammeerimiseks lihtsam kätte saada. Ka plaadil olevat kiipi on lubatud programmeerida. Kiipi on lubatud pessa paigutada või välja võtta vaid pingestamata viikude korral. Eriti tähelepanelik tuleb olla programmeerimisrežiimi korral, kuna selles režiimis kasutatavad pinged võivad mikroskeemi pöördumatult kahjustada. Vajalikud nõuded on toodud lk. 60.

Et vältida püsimällu salvestatud info kustumist valguse mõjul, tuleb kiibi aken katta lindiga ПВХ ГОСТ 16214-70. Selle nõude rahuldamise korral garanteeritakse püsimällu salvestatu säilimine töötavas arvutis 15000 tunni vältel ja laos hoitavas arvutikiibis 5 aasta jooksul.

Mikroskeem KM1816BE48 on signaalinivoode poolest ühilduv TTL seeriate 133 ja 155 elementidega. Signaalide andmiseks viikudele XTAL1, XTAL2 ja $\overline{\text{RESET}}$ tuleb kasutada nende seeriate avatud kollektoriga elemente ning koormustakisti valida nii, et signaalide parameetrid oleksid tabelis 3 lubatud piirides.

Arvuti väratite P0, P1 ja P2 viigud soovitatakse skeemi lülitada läbi siini formeerijate. Kiibi vahetusse lähedusse (mitte kaugemale kui 50 mm) soovitatakse toiteliinidele paigutada filtreerivad kõrgsageduslikud kondensaatorid mahtuvusega 0,022 - 0,15 μF.

Mikroskeemi trükkplaadile jootmisel soovitatakse kasutada joodist ХПОССУ 61-05 ГОСТ 21931-76 ja räbustit ФКС ОСТ 11 029.001-74.

Mikroskeemi tähisena tuleb skeemi tellimisel ja konstruktordokumentatsiooni koostamisel kasutada tähist: Mikroskeem КМ1816ВЕ48 ЬКО.348.839-01 ТУ.

2. Kasutamise piirtingimused.

1. Töötemperatuuride piirkond -10 kuni +70°C;
2. Säilitamistemperatuuride piirkond -50 kuni +50°C;
3. Maksimaalne tarbitav võimsus 750 mW;
4. Ülejäänud eeskirjad on toodud tabelis 3.

3. Staatilised ja dünaamilised parameetrid.

Tabelis 4 on dokumentide [1] ja [2] põhjal esitatud arvuti parameetrite arvväärtused erinevate tööreŽiimide korral. Tabeli lahtris "Tähis" on parameetri tähis esitatud vastavalt sellele, kuidas antud parameetrit on joonistel 40 - 43 toodud ajadiagrammidel tähistatud. Tähisteta parameetreid ei ole neil joonistel näidatud. Püsimälu programmeerimisega seotud parameetrid kehtivad temperatuuride vahemikus 25±10°C, ülejäänud parameetrid - temperatuuride vahemikus -10°C kuni 70°C.

PARAMEETRITE PIIRVÄÄRTUSED

Tabel 3.

Parameeter ja selle mõõtühik	Tähis	Norm			
		Lubatud piir		Riknemispiir	
		Mitte alla	Mitte üle	Alla	Üle
1. Toitepinged; V	U_{DD} U_{CC}	-	5,25	-0,5	7,0
2. Sisendpinge viikudel XTAL1, XTAL2, $\overline{SS}$, T0, T1, $\overline{INT}$, P17-P10, P27-P20, DB7-DB0, $\overline{RESET}$, EA; V.	U_I	-	5,25	-0,5	7,0
3. Pinge viigul V_{DD} programmeerimisrežiimis; V.	U_{DD}	-	26,0	-	25,0
4. Väljundvool signaali kõrge nivoo korral; mA.	I_{OH}	-0,4	-	-0,8	-
5. Väljundvool signaali madala nivoo korral; mA.	I_{OL}	-	2,0	-	3,0
6. Koormuse mahtuvus; pF.	C_L	-	190	-	500
7. Pinge viikudel PROG ja EA programmeerimis-režiimis; V.	U_{PROG} U_{EA}	-	24,5	-	25
8. Summaarne tarbimisvool; mA.	$I_{CC}+I_{DD}$	-	-	-	150

3.1. STAATILISED PARAMEETRID.

Tabel 4.

Parameeter ja selle mõõtühik	Tähis	Norm		Märkusi
		Mitte alla	Mitte üle	
1	2	3	4	5
1. Toitepinge; V.	U_{DD} U_{OC}	4,75	5,25	
2. Väljundpinge kõrge nivoo viikudel DB7 - DB0; V.	U_{12}	2,4		I=-100 µA
3. Väljundpinge kõrge nivoo viikudel $\overline{RD}$, $\overline{WR}$, $\overline{PSEN}$ ja ALE; V.	U_9	2,4		I=-100 µA
4. Väljundpinge kõrge nivoo viikudel P17 - P10, P27 - P20, PROG ja TO; V.		2,4		I=-40 µA
5. Väljundpinge madal nivoo viikudel DB7 - DB0; V.	U_{14}		0,45	I=2,0 mA
6. Väljundpinge madal nivoo viikudel $\overline{RD}$, $\overline{WR}$, $\overline{PSEN}$ ja ALE; V.	U_8		0,45	I=1,8 mA
7. Väljundpinge madal nivoo viigul PROG; V.			0,45	I=1,0 mA
8. Väljundpinge madal nivoo viikudel P17 - P10, P27 - P20 ja TO; V.			0,45	I=1,6 mA
9. Sisendpinge kõrge nivoo; V.		2,0	U_{CC}	
10. Sisendpinge kõrge nivoo viikudel XTAL1, XTAL2 ja $\overline{RESET}$; V.		3,8	U_{CC}	
11. Sisendpinge madal nivoo; V.		-0,5	0,8	

1	2	3	4	5
12. Sisendpinge madal nivoo viikudel XTAL1, XTAL2 ja RESET; V.		-0,5	0,6	
13. Programmeerimispinge kõrge nivoo; V.	U_{16}	24,0	26,0	
14. Programmeerimispinge madal nivoo; V.	U_{17}	4,75	5,25	
15. Sisendpinge kõrge nivoo viigul PROG püsimälu programmeerimisel; V.	U_{18}	21,5	24,5	
16. Sisendpinge madal nivoo viigul PROG püsimälu programmeerimisel; V.	U_{19}		0,2	
17. Sisendpinge kõrge nivoo viigul EA püsimälu programmeerimisel; V.	U_{20}	21,5	24,5	
18. Sisendpinge madal nivoo viigul EA püsimälu programmeerimisel; V.	U_{21}	4,75	5,25	
19. Lekkevoolu tugevus sisendites; μA.		-10	10	$U_{GND} < U < U_{OC}$
20. Sisendvoolu tugevus; μA.		-500		$U < U_{GND}+0,45$
21. Viigu V_{DD} tarbimisvoolu tugevus; mA.		5	15	
22. Viikude V_{CC} ja V_{DD} summaarse tarbimisvoolu tugevus; mA.		60	135	
23. Viigu V_{DD} tarbimisvoolu tugevus püsimälu programmeerimisel; mA.			30,0	
24. Viigu PROG sisendvoolu tugevus püsimälu programmeerimisel; mA.			16,0	

1	2	3	4	5
25. Viigu EA sisendvoolu tugevus püsimälu programmeerimisel; mA.			1,0	
26. Sisend-väljundviigu mahtuvus; pF.			20	
27. Sisendviigu mahtuvus; pF.			10	
3.2. DÜNAAMILISED PARAMEETRID.				
28. Masinatsükli kestus; μs.	t_1	2,5	15,0	
29. Signaali ALE kestus; ns.	t_2	400		
30. Signaali $\overline{\text{PSEN}}$ kestus; ns.	t_3	700		
31. Signaali $\overline{\text{WR}}$ kestus; ns.	t_4	700		
32. Signaali $\overline{\text{RD}}$ kestus; ns.	t_5	700		
33. Signaali $\overline{\text{PROG}}$ kestus; ns.	t_6	1510		
34. Ajavahemik aadressi väljastamisest viikudele DB7 - DB0 kuni signaali ALE tagumise frondini; ns.	t_7	150		
35. Ajavahemik aadressi väljastamisest viikudele P23 - P20 kuni signaali ALE tagumise frondini; ns.	t_8	150		
36. Ajavahemik signaali ALE tagumisest frondist kuni aadressi kadumiseni viikudelt DB7 - DB0; ns.	t_9	80		
37. Ajavahemik signaali ALE tagumisest frondist kuni aadressi kadumiseni viikudelt P23 - P20; ns.	t_{10}	80		
38. Ajavahemik signaali $\overline{\text{PSEN}}$ esimesest frondist kuni stabiilsete andmeteni viikudel DB7 - DB0; ns.	t_{11}	500		

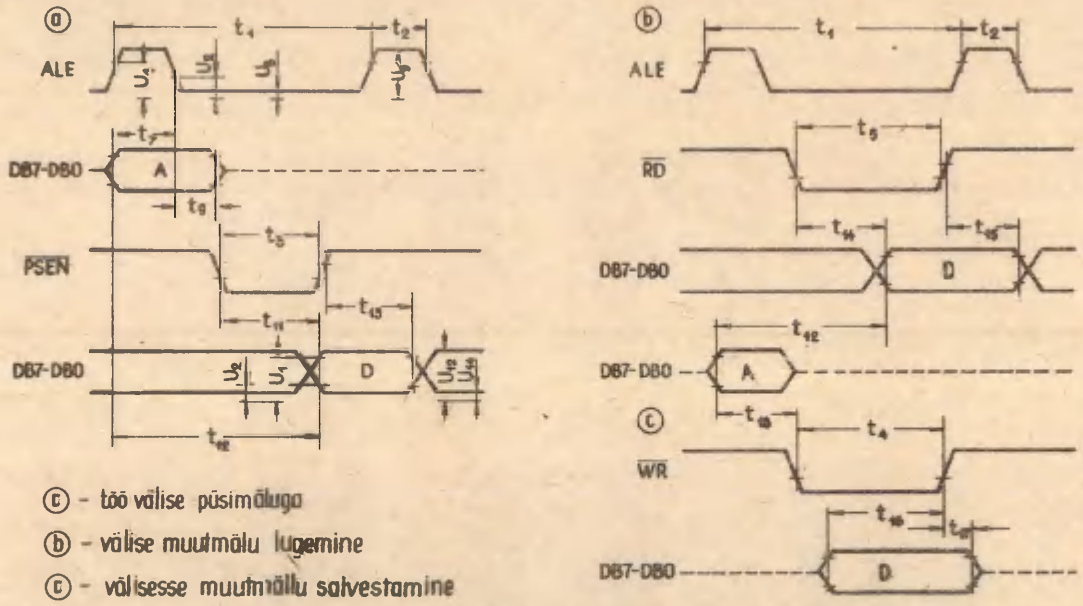
1	2	3	4	5
39. Ajavahemik aadressi stabiliseerumisest kuni andmete stabiliseerumiseni viikudel DB7 - DB0; ns.	t_{12}		950	
40. Ajavahemik signaali $\overline{PSEN}$ tagumisest frondist kuni andmete kadumiseni viikudelt DB7 - DB0; ns.	t_{13}	0	200	
41. Ajavahemik signaali $\overline{RD}$ esimesest frondist kuni andmete stabiliseerumiseni viikudel DB7 - DB0; ns.	t_{14}		500	
42. Ajavahemik signaali $\overline{RD}$ tagumisest frondist kuni andmete kadumiseni viikudelt DB7 - DB0; ns.	t_{15}	0	200	
43. Ajavahemik andmete stabiliseerumisest viikudel DB7 - DB0 kuni signaali $\overline{WR}$ tagumise frondini; ns.	t_{16}	500		
44. Ajavahemik signaali $\overline{WR}$ tagumisest frondist kuni andmete kadumiseni viikudelt DB7 - DB0; ns.	t_{17}	120		
45. Ajavahemik aadressi väljastamisest viikudele DB7 - DB0 kuni signaali $\overline{WR}$ esimese frondini; ns.	t_{18}	230		
46. Ajavahemik signaali ALE esimesest frondist kuni andmete kadumiseni viikudelt P23 - P20; ns.	t_{19}	150		
47. Ajavahemik andmete väljastamisest viikudele P23 - P20 kuni signaali ALE esimese frondini; ns.	t_{20}	400		
48. Ajavahemik signaali $\overline{PROG}$ esimesest frondist kuni täiendavaid väräteid programmeeriva sõna kadumiseni viikudelt P23 - P20; ns.	t_{21}	140		

1	2	3	4	5
49. Ajavahemik täiendavaid väärateid programmeeriva sõna väljastamisest viikudele P23 - P20 kuni signaali $\overline{\text{PROG}}$ esimese frondini; ns.	t_{22}	110		
50. Ajavahemik signaali $\overline{\text{PROG}}$ tagumisest frondist kuni täiendavatele väratitele väljastatava infosõna kadumiseni viikudelt P23 - P20; ns.	t_{23}	65		
51. Ajavahemik infosõna väljastamisest täiendavatele väratitele viikude P23 - P20 kaudu kuni signaali $\overline{\text{PROG}}$ tagumise frondini; ns.	t_{24}	220		
52. Ajavahemik signaali $\overline{\text{PROG}}$ esimesest frondist kuni täiendavate väratite kaudu sisestatava infosõna stabiliseerumiseni viikudel P23 - P20; ns.	t_{25}		810	
53. Ajavahemik signaal $\overline{\text{PROG}}$ tagumisest frondist kuni täiendavate väratite kaudu sisestatava infosõna kõrvaldamiseni viikudelt P23 - P20; ns.	t_{26}	0	150	
54. Ajavahemik programmeerimispinge U_{DD} stabiliseerumisest kuni programmeeriva impulsini PROG.	t_{27}	$4t_{CY}$		
55. Ajavahemik programmeeriva impulsi PROG lõppemisest kuni programmeerimispinge väljalülimiseni; ns.	t_{28}	0		
56. Programmeeriva impulsi PROG kestus; ns.	t_{29}	50	60	
57. Ajavahemik signaaliga TO programmeerimisrežiimi viimisest kuni aadressi fikseerimiseni signaaliga $\overline{\text{RESET}}$.	t_{30}	$4t_{CY}$		

1	2	3	4	5
58. Ajavahemik programmeerimispinge U_{DD} väljalülitamisest kuni programmeerimisrežiimi lõpetamiseni signaaliga TO.	t_{31}	$4t_{CY}$		
59. Signaali RESET kestus.	t_{32}	$4t_{CY}$		Esim. RESET imp. kestus peab olema $8t_{CY}$
60. Ajavahemik püsimällu salvestatu kontroll-lugemise režiimi viimisest signaaliga TO kuni mälupesa sisu väljastamiseni viikudele DB7 - DB0.	t_{33}		$4t_{CY}$	
61. Signaalide U_{DD} ja PROG tagumise frondi kestus; μs.	t_{34}	0,5	2,0	
62. Signaalide U_{DD} ja PROG esimese frondi kestus; μs.	t_{35}	0,5	2,0	
63. Ajavahemik mälupesa aadressi stabiliseerumisest viikudel DB7 - DB0 kuni aadressi fikseerimiseni signaaliga RESET.	t_{36}	$4t_{CY}$		
64. Ajavahemik aadressi fikseerimisest signaaliga RESET kuni aadressi kõrvaldamiseni viikudelt DB7 - DB0.	t_{37}	$4t_{CY}$		
65. Ajavahemik mälupessa salvestatava sõna stabiliseerumisest viikudel DB7 - DB0 kuni programmeeriva impulsini PROG.	t_{38}	$4t_{CY}$		
66. Ajavahemik programmeeriva impulsi PROG tagumisest frondist kuni mälupessa salvestatava sõna kõrvaldamiseni viikudelt DB7 - DB0.	t_{39}	$4t_{CY}$		

1	2	3	4	5
67. Masinatsükli kestus programmeerimisel; μs.	t_{CY}	5,0		
68. Ajavahemik signaali RESET algusest kuni programmeerimis- ja kontrollrežiimi aktiviseeriva signaalini EA.	t_{41}	$4t_{CY}$		
69. Kontrollrežiimi kestus.	t_{42}	$4t_{CY}$		
70. Ajavahemik programmeerimispinge 0 andmisest viigule PROG kuni mällu salvestatava sõna seadmiseni viikudele DB7 - DB0; μs.	t_{43}	20		
71. Ajavahemik, mille jooksul tuleb pinget 0 sisendviigul PROG hoida pärast mällu salvestatava sõna kõrvaldamist viikudelt DB7 - DB0; μs.	t_{44}	20		
72. Signaalide esimeste ja tagumiste frontide kestus (v.a. programmeerimisrežiim); ns.			15 15	
73. Signaalide esimeste ja tagumiste frontide kestus programmrežiimis (v.a. signaalid U_{DD} ja PROG); ns.			50 50	
74. Signaalide pingenivoo, mis on loetud ajavahemiku alguse või lõpu arvestuspunktiks; V.	U_1 U_2	2,0	 0,8	

Märkus: 1. Tähiseta parameetreid ei ole joonistel 40 - 43 näidatud.
2. Püsimälu programmeerimisega seotud parameetrid kehtivad temperatuuride vahemikus $25 \pm 10°0$, ülejäänud parameetrid - temperatuuride vahemikus -10°C kuni 70°C.

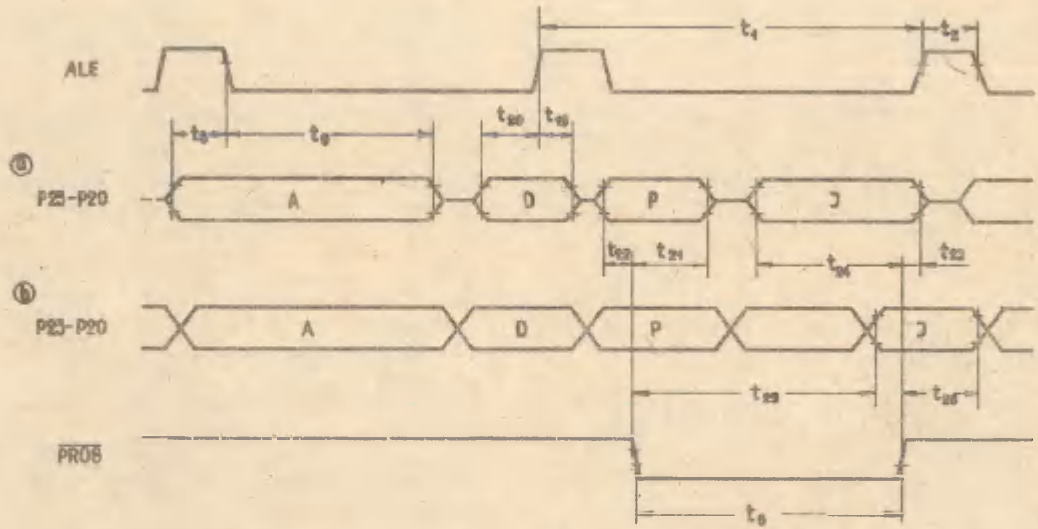


ⓐ - töö välise püsimäluga

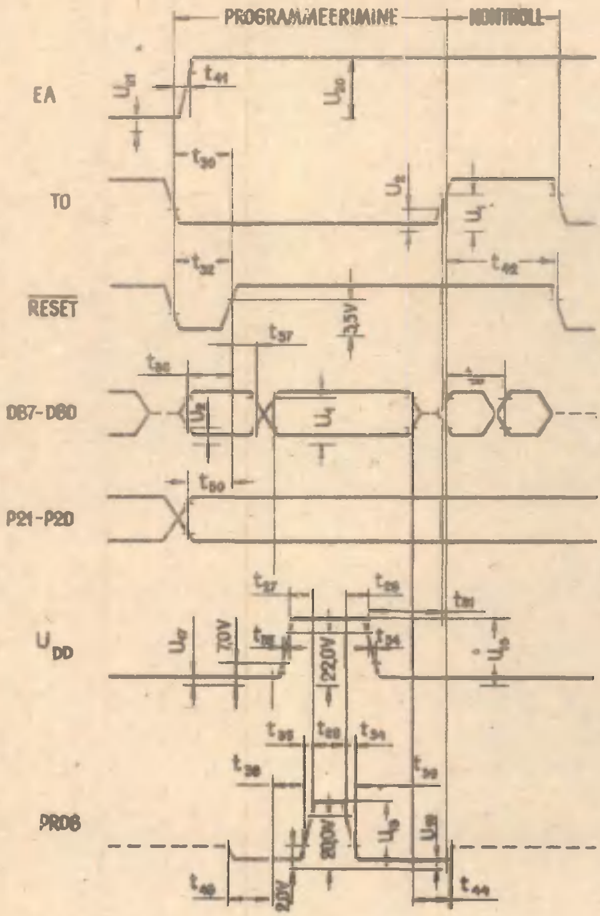
ⓑ - välise muutmälu lugemine

ⓒ - välisesse muutmällu salvestamine

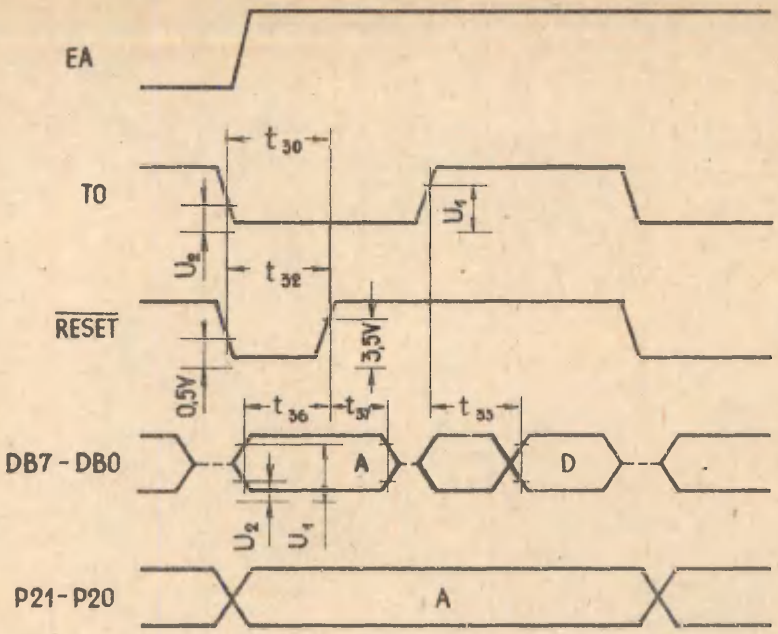
Joonis 40. Mikroarvuti ajadiagrammid välismälude kasutamisel.



Joonis 41. Infovahetuse ajadiagrammid täiendavate väratitega: a)käskude MOVD P,A ANLD P,A ja ORLD P,A täitmisel; b) käsu MOVD A,P täitmisel. Tähised: A - käsupesa aadressi 4 kõrgemat bitti; D - viikude P23 - P20 kaudu väljastatav info; P - laiendusväratit programmeeriv sõna; I - infosõna.



Joonis 42. Sisemise püsimälu programmeerimise ja kirjutatu kontrollimise ajadiagrammid.



Joonis 43. Sisemise püsimälu kontrollimise ajadiagrammid.

4. Arvuti käsustik.

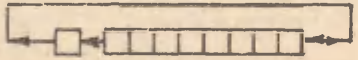
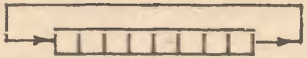
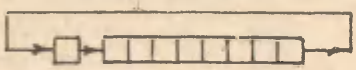
Tabelis 5 on toodud käsustiku lühikirjeldus. Käsud on siin esitatud funktsionaalsete gruppide kaupa. Iga käsutüübi kirjeldus sisaldab käsu mnemoonikut, koodi, pikkust baitides, käsu täitmiseks kuluvate masinatsüklite arvu ja käsu toimet selgitavat algoritmi. Tabelis kasutatud tähistuste selgitus on esitatud tabeli lõpus.

Tabelis 6 on esitatud arvuti kõigi käskude loend nende mnemoonikute alfabeetilises järjekorras ja tabelis 7 on kõik käsud reastatud nende koodi kasvamise järjekorras.

Käskude lühikirjeldus.

Tabel 5.

MNEMOKOOD	KOOD	BAITE	TSÜK-LEID	KÄSU KIRJELDUS/ TOIME LIPPUDELE C ja AC
AKUMULAATORIGA SEOTUD KÄSUD				
ADD A,R	01101NNN	1	1	(A):=(A)+(R) / C,AC
ADD A,@R	0110000N	1	1	(A):=(A)+SM(R) / C,AC
ADD A,#data	03_H #data	2	2	(A):=(A)+(b2) / C,AC
ADDC A,R	01111NNN	1	1	(A):=(A)+(R)+(C) / C,AC
ADDC A,@R	0111000N	1	1	(A):=(A)+SM(R)+(C) /C,AC
ADDO A,#data	13_H #data	2	2	(A):=(A)+(b2)+(C) / C,AC
ANL A,R	01011NNN	1	1	(A):=(A)∧(R)
ANL A,@R	0101000N	1	1	(A):=(A)∧SM(R)
ANL A,#data	53_H #data	2	2	(A):=(A)∧(b2)
ORL A,R	01001NNN	1	1	(A):=(A)V(R)
ORL A,@R	0100000N	1	1	(A):=(A)VSM(R)
ORL A,#data	43_H #data	2	2	(A):=(A)V(b2)
XRL A,R	11011NNN	1	1	(A):=(A)⊕(R)
XRL A,@R	1101000N	1	1	(A):=(A)⊕SM(R)
XRL A,#data	$D3_H$ #data	2	2	(A):=(A)⊕(b2)
INC A	17_H	1	1	(A):=(A)+1
DEC A	07_H	1	1	(A):=(A)-1
CLR A	27_H	1	1	(A):=0
CPL A	37_H	1	1	(A):=($\overline{A}$)

MNEMOKOOD	KOOD	BAITE	TSÜK-LEID	KÄSU KIRJELDUS/ TOIME LIPPUDELE C ja AC
DA A	57_H	1	1	kümnendkorrektsioon
SWAP A	47_H	1	1	$(A_{4-7}):=:(A_{0-3})$
RL A	$E7_H$	1	1	
RLC A	$F7_H$	1	1	 / C
RR A	77_H	1	1	
RRC A	67_H	1	1	 / C
SISEND-VÄLJUNDSEADMETEGA SEOTUD KÄSUD				
IN A,P	00001ONN	1	2	(A):=(P) (P=P1,P2)
OUTL P,A	00111ONN	1	2	(P):=(A) (P=P1,P2)
ANL P,#data	10011ONN #data	2	2	(P):=(P)∧(b2) (P=P1,P2)
ORL P,#data	10001ONN #data	2	2	(P):=(P)V(b2) (P=P1,P2)
INS A,BUS	08_H	1	2	(A):=(PO)

MNEMOKOOD	KOOD	BAITE	TSÜK-LEID	KÄSU KIRJELDUS/ TOIME LIPPUDELE C ja AC
OUTL BUS,A	02_H	1	2	(PO):=(A)
ANL BUS,#data	98_H #data	2	2	(PO):=(PO)∧(b2)
ORL BUS,#data	88_H #data	2	2	(PO):=(PO)V(b2)
MOVD A,P	000011NN	1	2	$(A_{0-3}):=(P),(A_{4-7}):=0$
MOVD P,A	001111NN	1	2	$(P):=(A_{0-3})$
ANLD P,A	100111NN	1	2	$(P):=(P)\wedge(A_{0-3})$
ORLD P,A	100011NN	1	2	$(P):=(P)V(A_{0-3})$
				} (P=P4,P5,P6,P7)
REGISTRITEGA SEOTUD KÄSUD				
INC R	00011NNN	1	1	(R):=(R)+1
INC @R	0001000N	1	1	SM(R):=SM(R)+1
DEC R	11001NNN	1	1	(R):=(R)-1
SIIRDEKÄSUD				
JMP addr	NNN00100 addr	2	2	$(PO_{8-10}):=(b1_{5-7}),(PCL):=(b2),(PC_{11}):=(DBF)$
JMPP @A	$B3_H$	1	2	$(PC_{0-7}):=M(A)$
DJNZ R,addr	11101NNN addr	2	2	(R):=(R)-1. Kui (R)≠0, siis (PCL):=(b2), muidu (PC):=(PC)+2
JC addr	$F6_H$addr	2	2	kui (C)=1 } siis (PCL):=(b2), muidu (PC):=(PC)+2
JNC addr	$E6_H$addr	2	2	kui (C)=0 }

MNEMOKOOD	KOOD	BAITE	TSÜK-LEID	KÄSU KIRJELDUS/ TOIME LIPPUDELE C ja AC
JZ addr	$C6_H$addr	2	2	kui (A)=0,
JNZ addr	96_Haddr	2	2	kui (A)≠0,
JTO addr	36_Haddr	2	2	kui (TO)=1,
JNTO addr	26_Haddr	2	2	kui (TO)=0,
JT1 addr	56_Haddr	2	2	kui (T1)=1,
JNT1 addr	46_Haddr	2	2	kui (T1)=0, } siis (PCL):=(b2),
JF0 addr	$B6_H$addr	2	2	kui (F0)=1, muidu (PC):=(PC)+2
JF1 addr	76_Haddr	2	2	kui (F1)=1,
JTF addr	16_Haddr	2	2	kui (TF)=1,
JNI addr	86_Haddr	2	2	kui ($\overline{INT}$)=0,
JBb addr	NNN10010 addr	2	2	kui (A_b)=1
ALAMPROGRAMMIDEGA SEOTUD KÄSUD				
CALL addr	NNN10100 addr	2	2	SM(SP):=(PC)+(PSW), (SP):=(SP)+1, (PC_{8-10}):=($b1_{5-7}$), (PCL):=(b2),(PC_{11}):=(DBF)
RET	83_H	1	2	(SP):=(SP)-1, (PC):=SM(SP)
RETR	93_H	1	2	(SP):=(SP)-1, (PC)+(PSW):=SM(SP)
DIPPUDEGA SEOTUD KÄSUD				
CLR C	97_H	1	1	(C):=0 / C
CPL C	$A7_H$	1	1	(C):=($\overline{C}$) / C
CLR F0	85_H	1	1	(F0):=0

MNEMOKOOD	KOOD	BAITE	TSÜK-LEID	KÄSU KIRJELDUS/ TOIME LIPPUDELE C ja AC
CPL F0	95_H	1	1	(F0):=($\overline{F0}$)
CLR F1	$A5_B$	1	1	(F1):=0
CPL F1	$B5_H$	1	1	(F1):=($\overline{F1}$)
TEISALDUSKÄSUD				
MOV A,R	11111NNN	1	1	(A):=(R)
MOV A,@R	1111000N	1	1	(A):=SM(R)
MOV A,#data	23_H #data	2	2	(A):=(b2)
MOV R,A	10101NNN	1	1	(R):=(A)
MOV @R,A	1010000N	1	1	SM(R):=(A)
MOV R,#data	10111NNN #data	2	2	(R):=(b2)
MOV @R,#data	1011000N #data	2	2	SM(R):=(b2)
MOV A,PSW	$C7_H$	1	1	(A):=(PSW)
MOV PSW,A	$D7_H$	1	1	(PSW):=(A) / C,AC
XCH A,R	00101NNN	1	1	(A):=:(R)
XCH A,@R	0010000N	1	1	(A):=:SM(R)
XCHD A,@R	0011000N	1	1	(A_{0-3}):=:$SM(R)_{0-3}$
MOVX A,@R	1000000N	1	2	(A):=VM(R)
MOVX @R,A	1001000N	1	2	VM(R):=(A)
MOVP A,@A	$A3_H$	1	2	(A):=M(A) (käsilolevalt leheküljelt)
MOVP3 A,@A	$E3_H$	1	2	(A):=M(A) (panga nr.0 leheküljelt nr.3)

MNEMOKOOD	KOOD	BAITE	TSÜK-LEID	KÄSU KIRJELDUS/ TOIME LIPPUDELE C ja AC
TAIMERIGA SEOTUD KÄSUD				
MOV A,T	42_H	1	1	(A):=(T)
MOV T,A	62_H	1	1	(T):=(A)
STRT T	55_H	1	1	käivitab taimeri (sisemine signaal)
STRT CNT	45_H	1	1	käivitab taimeri (signaal viigul T1)
STOP TCNT	65_H	1	1	seiskab taimeri
EN TCNTI	25_H	1	1	lubab taimerkatkestused
DIS TCNTI	35_H	1	1	keelab taimerkatkestused
JUHTIMISKÄSUD				
EN I	05_H	1	1	lubab väliskatkestused
DIS I	15_H	1	1	keelab väliskatkestused
SEL RB0	$C5_H$	1	1	(BS):=0 } registerpanga valik
SEL RB1	$D5_H$	1	1	(BS):=1 } registerpanga valik
SEL MB0	$E5_H$	1	1	(DBF):=0 } mälupanga valik
SEL MB1	$F5_H$	1	1	(DBF):=1 } mälupanga valik
ENTO CLK	75_H	1	1	väljastab takteerimissignaali viigule TO
NOP	00_H	1	1	tühikäsk

Tabelis 5 toodud kirjeldustes on kasutatud järgmisi tähistusi:

A	akumulaator; A_7 - A_0 - akumulaatori bitid;
A_{0-3}	akumulaatori 4 madalamat bitti A_3 - A_0;
A_{4-7}	akumulaatori 4 kõrgemat bitti A_7 - A_4;
R	üks registritest { R0, R1 või R0 - R7 / R0', R1' või R0'- R7';
PC	12- bitine käsuloendur; PCL - käsuloenduri bitid PC_7 - PC_0; PC_{11} - käsuloenduri kõrgeim bitt;
PSW	programmiolekusõna; olekuregister; olekuregistri bitid: PSW_7 - PSW_0;
C	ülekandelipp;
AC	abilipp;
F0, F1	lipud F0, F1;
SP	pinuviit: PSW bitid PSW_2 - PSW_0;
BS	registerpanga lipp: olekuregistri bitt PSW_4;
DBF	mälupanga triger;
P	üks väratitest P1, P2 või P4 - P7;
BUS	värat P0;
T	taimeri loendusregister;
T0, T1, $\overline{INT}$	arvuti vastavate viikude kaudu sisestatavad signaalid;
addr	8-bitine aadress;
b1, b2	käsu esimene, teine bait;
M	püsimälu pesa;
M(A)	püsimälu käsiloleval leheküljel akumulaatoriga adresseeritud pesa sisu;
SM	sisemise muutmälu pesa;
SM(R)	registriga R adresseeritud sisemise muutmälu pesa sisu;
$SM(R)_{0-3}$	registriga R adresseeritud sisemise muutmälu pesa neljas madalamas bitis olev arv;
SM(SP)	arv pinumälu pesade sellest paarist, mis vastab pinuviidale SP;
VM	välise muutmälu pesa;

VM(R)	registriga R adresseeritud välise muutmälu pesa sisu;
(·)	sulgudes paikneva objekti sisu;
#data	8-bitine vahetu operand;
@R	operand registriga adresseeritud sisemise või välise muutmälu pesast;
@A	operand akumulaatoriga adresseeritud püsimälu pesast;
:=	ühepoolne omistamine;
:=:	kahepoolne omistamine;
Λ	loogilise korrutamistehte märk;
V	loogilise liitmistehte märk;
$\oplus$	välistava loogilise liitmistehte märk;
(PC)+(PSW)	pinumällu salvestatav info; PC_{11} - PC_0 ning PSW_7 - PSW_4 sisu;
NNN	käsu aadressivälja bitid;
(BCD)	kahend-kümnendkoodi tähis (binary coded decimal);
H	16ndkoodi tähis.

Käskude loend mnemokoodide tähestikulises järjekorras.

Tabel 6.

MNEMOKOOD	KOOD	MNEMOKOOD	KOOD	MNEMOKOOD	KOOD
ADD A,R0	68	ANL A,#data	53	DEC R6	CE
ADD A,R1	69	ANL BUS,#data	98	DEC R7	CF
ADD A,R2	6A	ANL P1,#data	99	DIS I	15
ADD A,R3	6B	ANL P2,#data	9A	DIS TCNTI	35
ADD A,R4	6C	ANLD P4,A	9C	DJNZ R0,addr	E8
ADD A,R5	6D	ANLD P5,A	9D	DJNZ R1,addr	E9
ADD A,R6	6E	ANLD P6,A	9E	DJNZ R2,addr	EA
ADD A,R7	6F	ANLD P7,A	9F	DJNZ R3,addr	EB
ADD A,@R0	60	CALL addr 1k0	14	DJNZ R4,addr	EC
ADD A,@R1	61	CALL addr 1k1	34	DJNZ R5,addr	ED
ADD A,#data	03	CALL addr 1k2	54	DJNZ R6,addr	EE
ADDC A,R0	78	CALL addr 1k3	74	DJNZ R7,addr	EF
ADDC A,R1	79	CALL addr 1k4	94	EN I	05
ADDC A,R2	7A	CALL addr 1k5	B4	EN TCNTI	25
ADDC A,R3	7B	CALL addr 1k6	D4	ENTO CLK	75
ADDC A,R4	7C	CALL addr 1k7	F4	IN A,P1	09
ADDC A,R5	7D	CLR A	27	IN A,P2	0A
ADDC A,R6	7E	CLR C	97	INC A	17
ADDC A,R7	7F	CLR F0	85	INC R0	18
ADDC A,@R0	70	CLR F1	A5	INC R1	19
ADDC A,@R1	71	CPL A	37	INC R2	1A
ADDC A,#data	13	CPL C	A7	INC R3	1B
ANL A,R0	58	CPL F0	95	INC R4	1C
ANL A,R1	59	CPL F1	B5	INC R5	1D
ANL A,R2	5A	DA A	57	INC R6	1E
ANL A,R3	5B	DEC A	07	INC R7	1F
ANL A,R4	5C	DEC R0	C8	INC @R0	10
ANL A,R5	5D	DEC R1	C9	INC @R1	11
ANL A,R6	5E	DEC R2	CA	INS A,BUS	08
ANL A,R7	5F	DEC R3	CB	JB0 addr	12
ANL A,@R0	50	DEC R4	CC	JB1 addr	32
ANL A,@R1	51	DEC R5	CD	JB2 addr	52

MNEMOKOOD	KOOD	MNEMOKOOD	KOOD	MNEMOKOOD	KOOD
JB3 addr	72	MOV A,R4	FC	MOVD A,P7	0F
JB4 addr	92	MOV A,R5	FD	MOVD P4,A	3C
JB5 addr	B2	MOV A,R6	FE	MOVD P5,A	3D
JB6 addr	D2	MOV A,R7	FF	MOVD P6,A	3E
JB7 addr	F2	MOV A,@R0	F0	MOVD P7,A	3F
JC addr	F6	MOV A,@R1	F1	MOVP A,@A	A3
JF0 addr	B6	MOV A,T	42	MOVP3 A,@A	E3
JF1 addr	76	MOV PSW,A	D7	MOVX A,@R0	80
JMP addr lk0	04	MOV R0,A	A8	MOVX A,@R1	81
JMP addr lk1	24	MOV R1,A	A9	MOVX @R0,A	90
JMP addr lk2	44	MOV R2,A	AA	MOVX @R1,A	91
JMP addr lk3	64	MOV R3,A	AB	NOP	00
JMP addr lk4	84	MOV R4,A	AC	ORL A,R0	48
JMP addr lk5	A4	MOV R5,A	AD	ORL A,R1	49
JMP addr lk6	C4	MOV R6,A	AE	ORL A,R2	4A
JMP addr lk7	E4	MOV R7,A	AF	ORL A,R3	4B
JMPP @A	B3	MOV R0,#data	B8	ORL A,R4	4C
JNC addr	E6	MOV R1,#data	B9	ORL A,R5	4D
JNI addr	86	MOV R2,#data	BA	ORL A,R6	4E
JNT0 addr	26	MOV R3,#data	BB	ORL A,R7	4F
JNT1 addr	46	MOV R4,#data	BC	ORL A,@R0	40
JNZ addr	96	MOV R5,#data	BD	ORL A,@R1	41
JTF addr	16	MOV R6,#data	BE	ORL A,#data	43
JT0 addr	36	MOV R7,#data	BF	ORL BUS,#data	88
JT1 addr	56	MOV @R0,A	A0	ORL P1,#data	89
JZ addr	C6	MOV @R1,A	A1	ORL P2,#data	8A
MOV A,#data	23	MOV @R0,#data	B0	ORLD P4,A	8C
MOV A,PSW	C7	MOV @R1,#data	B1	ORLD P5,A	8D
MOV A,R0	F8	MOV T,A	62	ORLD P6,A	8E
MOV A,R1	F9	MOVD A,P4	0C	ORLD P7,A	8F
MOV A,R2	FA	MOVD A,P5	0D	OUTL BUS,A	02
MOV A,R3	FB	MOVD A,P6	0E	OUTL P1,A	39

MNEMONKOOD	KOOD	MNEMONKOOD	KOOD	MNEMONKOOD	KOOD
OUTL P2,A	3A	STRT T	55	XCHD A,@R1	31
RET	83	SWAP A	47	XRL A,R0	D8
RETR	93	XCH A,R0	28	XRL A,R1	D9
RL A	E7	XCH A,R1	29	XRL A,R2	DA
RLC A	F7	XCH A,R2	2A	XRL A,R3	DB
RR A	77	XCH A,R3	2B	XRL A,R4	DC
RRC A	67	XCH A,R4	2C	XRL A,R5	DD
SEL MB0	E5	XCH A,R5	2D	XRL A,R6	DE
SEL MB1	F5	XCH A,R6	2E	XRL A,R7	DF
SEL RB0	C5	XCH A,R7	2F	XRL A,@R0	D0
SEL RB1	D5	XCH A,@R0	20	XRL A,@R1	D1
STOP TCNT	65	XCH A,@R1	21	XRL A,#data	D3
STRT CNT	45	XCHD A,@R0	30		

Käskude loend käsukoodi kasvamise järjekorras.

Tabel 7.

KOOD	MNEMOKOOD	KOOD	MNEMOKOOD	KOOD	MNEMOKOOD
00	NOP	20	XCH A,@RO	40	ORL A,@RO
01	----	21	XCH A,@R1	41	ORL A,@R1
02	OUTL BUS,A	22	----	42	MOV A,T
03	ADD A,#data	23	MOV A,#data	43	ORL A,#data
04	JMP addr lk0	24	JMP addr lk1	44	JMP addr lk2
05	EN I	25	EN TCNTI	45	STRT CNT
06	----	26	JNTO addr	46	JNT1 addr
07	DEC A	27	CLR A	47	SWAP A
08	INS A,BUS	28	XCH A,RO	48	ORL A,RO
09	IN A,P1	29	XCH A,R1	49	ORL A,R1
0A	IN A,P2	2A	XCH A,R2	4A	ORL A,R2
0B	----	2B	XCH A,R3	4B	ORL A,R3
0C	MOVD A,P4	2C	XCH A,R4	4C	ORL A,R4
0D	MOVD A,P5	2D	XCH A,R5	4D	ORL A,R5
0E	MOVD A,P6	2E	XCH A,R6	4E	ORL A,R6
0F	MOVD A,P7	2F	XCH A,R7	4F	ORL A,R7
10	INC @RO	30	XCHD A,@RO	50	ANL A,@RO
11	INC @R1	31	XCHD A,@R1	51	ANL A,@R1
12	JBO addr	32	JB1 addr	52	JB2 addr
13	ADDC A,#data	33	----	53	ANL A,#data
14	CALL addr lk0	34	CALL addr lk1	54	CALL addr lk2
15	DIS I	35	DIS TCNTI	55	STRT T
16	JTF addr	36	JTO addr	56	JT1 addr
17	INC A	37	CPL A	57	DA A
18	INC RO	38	----	58	ANL A,RO
19	INC R1	39	OUTL P1,A	59	ANL A,R1
1A	INC R2	3A	OUTL P2,A	5A	ANL A,R2
1B	INC R3	3B	----	5B	ANL A,R3
1C	INC R4	3C	MOVD P4,A	5C	ANL A,R4
1D	INC R5	3D	MOVD P5,A	5D	ANL A,R5
1E	INC R6	3E	MOVD P6,A	5E	ANL A,R6
1F	INC R7	3F	MOVD P7,A	5F	ANL A,R7

KOOD	MNEMOKOOD	KOOD	MNEMOKOOD	KOOD	MNEMOKOOD
60	ADD A,@R0	80	MOVX A,@R0	A0	MOV @R0,A
61	ADD A,@R1	81	MOVX A,@R1	A1	MOV @R1,A
62	MOV T,A	82	----	A2	----
63	----	83	RET	A3	MOVP A,@A
64	JMP addr lk3	84	JMP addr lk4	A4	JMP addr lk5
65	STOP TCNT	85	CLR F0	A5	CLR F1
66	----	86	JNI addr	A6	----
67	RRC A	87	----	A7	CPL C
68	ADD A,R0	88	ORL BUS,#data	A8	MOV R0,A
69	ADD A,R1	89	ORL P1,#data	A9	MOV R1,A
6A	ADD A,R2	8A	ORL P2,#data	AA	MOV R2,A
6B	ADD A,R3	8B	----	AB	MOV R3,A
6C	ADD A,R4	8C	ORLD P4,A	AC	MOV R4,A
6D	ADD A,R5	8D	ORLD P5,A	AD	MOV R5,A
6E	ADD A,R6	8E	ORLD P6,A	AE	MOV R6,A
6F	ADD A,R7	8F	ORLD P7,A	AF	MOV R7,A
70	ADDC A,@R0	90	MOVX @R0,A	B0	MOV @R0,# data
71	ADDC A,@R1	91	MOVX @R1,A	B1	MOV @R1,# data
72	JB3 addr	92	JB4 addr	B2	JB5 addr
73	----	93	RETR	B3	JMPP @A
74	CALL addr lk3	94	CALL addr lk4	B4	CALL addr lk5
75	ENTO CLK	95	CPL F0	B5	CPL F1
76	JF1 addr	96	JNZ addr	B6	JF0 addr
77	RR A	97	CLR C	B7	----
78	ADDC A,R0	98	ANL BUS,#data	B8	MOV R0,# data
79	ADDC A,R1	99	ANL P1,#data	B9	MOV R1,#data
7A	ADDC A,R2	9A	ANL P2,#data	BA	MOV R2,#data
7B	ADDC A,R3	9B	----	BB	MOV R3,#data
7C	ADDC A,R4	9C	ANLD P4,A	BC	MOV R4,#data
7D	ADDC A,R5	9D	ANLD P5,A	BD	MOV R5,#data
7E	ADDC A,R6	9E	ANLD P6,A	BE	MOV R6,#data
7F	ADDC A,R7	9F	ANLD P7,A	BF	MOV R7,#data

KOOD	MNEMOKOOD	KOOD	MNEMOKOOD
C0	----	E0	-----
C1	----	E1	-----
C2	----	E2	-----
C3	----	E3	MOVP3 A,@A
C4	JMP addr lk6	E4	JMP addr lk7
C5	SEL RB0	E5	SEL MB0
C6	JZ addr	E6	JNC addr
C7	MOV A,PSW	E7	RL A
C8	DEC R0	E8	DJNZ R0,addr
C9	DEC R1	E9	DJNZ R1,addr
CA	DEC R2	EA	DJNZ R2,addr
CB	DEC R3	EB	DJNZ R3,addr
CC	DEC R4	EC	DJNZ R4,addr
CD	DEC R5	ED	DJNZ R5,addr
CE	DEC R6	EE	DJNZ R6,addr
CF	DEC R7	EF	DJNZ R7,addr
D0	XRL A,@R0	F0	MOV A,@R0
D1	XRL A,@R1	F1	MOV A,@R1
D2	JB6 addr	F2	JB7 addr
D3	XRL A,#data	F3	----
D4	CALL addr lk6	F4	CALL addr lk7
D5	SEL RB1	F5	SEL MB1
D6	----	F6	JC addr
D7	MOV PSW,A	F7	RLC A
D8	XRL A,R0	F8	MOV A,R0
D9	XRL A,R1	F9	MOV A,R1
DA	XRL A,R2	FA	MOV A,R2
DB	XRL A,R3	FB	MOV A,R3
DC	XRL A,R4	FC	MOV A,R4
DD	XRL A,R5	FD	MOV A,R5
DE	XRL A,R6	FE	MOV A,R6
DF	XRL A,R7	FF	MOV A,R7

ÕPPEVAHENDI KOOSTAMISEL KASUTATUD MATERJALID

1. Микросхема КМ1816ВЕ48. Техническое описание и инструкция по эксплуатации. ИТ3.480.038 ТО. 1984, 106 lk.
2. Микросхемы интегральные КМ1816ВЕ48. Технические условия бК0.348.839-0ТУ. 1984, 50 lk.
3. Руководство пользователя микроЭВМ МС -48. Перевод № А-26177. Всесоюзный центр переводов научно-технической литературы и документации, М. 1978, 638 стр.
4. Е.И.Крылов, Однокристальные микроЭВМ серий К1814, К1820, К1816, Микропроцессорные средства и системы, № 2, 1985, стр. 3 - 7.
5. А.В.Кобылинский, Г.П.Липовецкий, Однокристальные микроЭВМ серии 1816, Микропроцессорные средства и системы, № 1, 1986, стр.10 - 19.
6. Цикл статей: "Кросс-средства для автоматизации разработки, тестирования и отладки программ однокристальных микроЭВМ серии КМ1816", Микропроцессорные средства и системы, № 3, 1986, стр. 23 - 33.
7. **Microcontroller Handbook. - Intel Corporation Literature Department, Santa Clara, 1985, pp. 12-1 - 15-17.**
8. **A.Osborne, B.Kane, OSBORNE 4 & 8-Bit Microprocessor Handbook, OSBORNB/McGraw-Hill, Berkeley, California, 1981, pp. 6.1 - 6.D22.**
9. **Mikroarvuti KM1816BE48 käsustik, Tartu, 1986, 106 lk.**
10. **8048 Family Applications Handbook. - Intel Corporation Literature Department, Santa Clara, 1980, pp. 1-1 - 13-6.**

Sisukord.

Tabelid.